Basic DromEd Tutorial

Welcome to DromEd, Looking Glass Studios' Thief: The Dark Project level editor, and the tool you need to create your own Thief missions. This tutorial is designed to walk you through the basic functions of DromEd, by teaching you to create rooms, place objects, activate enemies, equip Garrett with some weapons, and specify objectives and difficulty levels. It's important that you follow this tutorial from beginning to end; by skipping sections, you may very well miss out on some critical information.
Also keep in mind that DromEd has been released to the public "as is," meaning the program is unsupported by Looking Glass. We hope we've given you enough information to get started…but you'll have to take it from there. **In other words, do not call Looking Glass or EIDOS asking for help with DromEd!** We wish you success in designing Thief levels.

I. What's On The Screen

When you load up DromEd, you'll be presented with the main program interface, separated into four distinct sections:

1.) A menu bar at the top of the screen, as seen in any Windows-based program.
2.) Four separate view windows, which allow you to examine your levels from four different perspectives. These windows, starting in the upper left-hand corner and moving clockwise, are "3D View," "Top," "Right," and "Front." You can use the bracket keys to cycle through the different windows, with the white outline indicating which is currently the "active" window. For the purposes of this tutorial, don't worry about the active window or its uses.
3.) Several buttons, located in the bottom left-hand section of the screen, used for a variety of functions.
4.) A command line, located in the bottom right-hand section of the screen (in the small box outlined in light blue), used to enter specific DromEd commands.

[**Important Note:** In order to properly use DromEd, your desktop must be set larger than DromEd's default resolution of 640x480. If your desktop is set to 640x480, and you run DromEd, important parts of the interface screen – like the menu bar at the top – will be cut off.]
During the course of this tutorial, we'll explore each of these interface elements in-depth.

II. Brushes and Portalizing

If you have no experience building levels for 3D games, the first thing you need to do is familiarize yourself with the term "brush." In DromEd, anything that gets created is called a brush. Rooms are referred to as "room brushes," lights are called "light brushes," etc. The most common brush is the "operation brush," used to create terrain (hollowed-out "rooms" and solid objects). DromEd allows the user to create spaces using six different shapes of operation brush: cube, cylinder, pyramid, corner-apex pyramid, wedge, and dodecahedron. You can think of the default Thief level as being completely solid and stretching infinitely in all directions. In other words, before you add any brushes, the world is just an infinitely huge block of solid. You carve away from that block to create rooms, stairs, structures, and other unique architectural features. So, to create a square room, you would use a cube-shaped operation brush, filled with air, to carve away a square in the center of the existing solid block. If all of this is a bit confusing, don't worry: everything will become clear once we start building some rooms. As you can see by looking at the different view windows, there's already a beginning operation brush in place (a cube) and you're standing right in the middle of it, as indicated by the violet-colored icon. Now look down at the bottom of the screen, to the small, funky-looking texture affectionately referred to as "Jorge." That's the default texture for the selected brush, but we'll discuss textures later in the tutorial. For now, look at the buttons underneath Jorge, specifically, the one that reads "Op<Fill Air >." Click on the arrows to see the different types of brushes you can create. After you've seen all the choices, go back to "Fill Air," because we want to create an air brush, which is essentially a hollowed-out room. Remember, the existing universe is already a giant, solid block, so we need to fill a brush with air to create a room; creating a brush and then filling it in with solid will have no effect, for obvious reasons.

Now that you've set the brush selection to "Fill Air," it's time to "portalize" the level so the change actually takes place in the game world. For the purposes of designing levels with DromEd, portalizing is the process by which your brushes, which are created on the 2D grid, are transformed into 3D space. So, go to the menu bar on the top of the screen, click "Tools," then click "Portalize." Be patient as DromEd processes the brushes. Small levels with just a few brushes usually portalize in just a couple of seconds, but when your levels get larger and more complex, portalization can take a couple of minutes or more. Look at the center of the white bar on the very bottom of the DromEd screen; when it reads "done," the level has been portalized. Whenever you alter the terrain in the level, using one of the 2D views, you will need to portalize to see new changes in the 3D View window and in the game itself.

III. Viewing Brushes / Moving around in the View Windows

Okay, you've portalized the level. It may seem like nothing actually happened, but don't panic: that's just because the 3D View window defaults to a wireframe representation of the 3D world, and it's tough to notice any changes. Move the mouse cursor to the 3D View window and press and hold the right mouse button. Doing so allows you to select from a list of options for the view you currently have the mouse cursor over. We'll discuss these options in more depth a bit later. For now, highlight the choice that reads "solid + selection," then release the right mouse button. This will change the 3D View representation from wireframe to solid, with a white outline indicating the currently selected brush (in this case, the *only* brush).

Now, you may still be wondering why the 3D View hasn't changed much; right now it should look like we've just gone from a wireframe display to a completely black display with a few white lines. Actually, you're standing in the exact center of the operation brush you just created, but the "room" is shrouded in darkness so it's impossible to see anything. When you create Thief rooms using DromEd, everything is pitch black – you need to actually place light sources (either light brushes, or light-emitting objects, like torches or electric lamps). But, for the sake of simple level designing, we can take a shortcut and use the "light_bright" command, which completely illuminates the level and eliminates all darkness and shadow. [**Important Note:** While the term "room" can be used to refer to operation brushes you create, it's important to know that DromEd does not yet consider them rooms. So, even if you create a detailed banquet hall, DromEd

won't see the space as a room, but a large air brush. Don't worry about that for now, though – we'll get to this issue later on in the tutorial.]
To enter the command, either left click on the command line or press semicolon (note: you'll need to hit Shift and the ";" key). Type "light_bright" (as with all commands, do **not** use quotation marks) and hit the "Enter" key. Now that you've entered in the "light_bright" command, you'll notice (hey, you may even be panicking) that the 3D View window still hasn't changed. That's because there needs to be some "movement" in the 3D world before "light_bright" activates. In the 3D View window: the "A" and "D" keys are used to rotate left and right; the "W" and "S" keys are used to move forward and backward; the "Z" and "C" keys are used to slide left and right; and the "Q" and "E" keys are used to move up and down. Hit one of those keys, and the 3D View window will change – you should now be hovering in the center of a small, very ugly room. Use the "R" and "V" keys to look up and down, and the "F" key to recenter your view. Remember how you chose the "solid + selection" representation for the 3D View window, by pressing and holding the right mouse button? Move the mouse cursor over one of the 2D view windows (top, right, or front), and do the same thing. Choose "teleport camera" to instantly jump to that point in the 2D display. The selection defaults to "teleport camera," so you should get into the habit of moving the mouse cursor to different spots in the 2D windows, and right clicking to instantly jump to those positions. [**Important Note:** Don't try this when the mouse cursor is positioned over the 3D View window. The 3D View menu defaults to "solo view," and accidentally choosing this option will find DromEd's grid sections replaced by a giant 3D view. If this happens, just right click on the 3D View window to revert DromEd back to normal mode, and bring back the four small windows.] Try teleporting outside the operation brush. In the 3DView window, you'll be able to see the operation brush from a distance, just sort of sitting there in space. It's important to realize, though, that during the course of creating your level, you never want to allow the player or any AIs to enter any "solid" part of the terrain. The player and AIs can travel through water and air, but putting them in solid is a sure way to "break" your level.
Now it's time to enter into the 3D world in "game mode" and see the room as Garrett would. Go back inside the room and hit "Alt+G" to enter into the game. You are now, for all intents and purposes, in Thief. You can hit the "Escape" key to access different options, like changing video cards or reconfiguring keyboard keys. [**Note:** Make sure you're actually inside the confines of your operation brush before you enter into game mode, or you'll run into problems.] You may notice that some things appear to be missing – Garrett doesn't have a health meter, there are no items in your inventory, and there are no sound effects. That's because DromEd doesn't yet recognize your brush as an actual room, and you haven't yet placed a Garrett character model (referred to as a "Starting Point") in the game world. But don't worry about those things just yet; we'll come back to them later.
When you're ready to go back to the editor hit "Alt+E." It's now time to modify the room even further, to make it more recognizable.

IV: Using Textures

So, we've created a simple room…but it doesn't exactly look like a room. That's because the whole thing is using Jorge, the default texture. In order to create Thief-style rooms, complete with carpeting, cobblestones, and wooden planks, we need to load in texture families. Go to the command line, type in “add_family core” and hit “Enter” to load in the “core” set of Thief textures. Now, let’s add another texture family, but this time use a shortcut. Go to the command line again and type “a” – then hit the “Tab” key to cycle alphabetically through all recognizable commands. Use this method to select (or simply type in) the command “add_family rescore” to load in Thief’s core set of residential textures.

To see the textures that you’ve loaded in, bring up the texture palette by pressing “Alt+T.” For now, we’ll make the whole room one texture. Click on one of the brick textures so that its name is highlighted in violet. At the end of the texture palette, you’ll notice a few buttons; click the one that reads “Put on Brush.” You’ll see that texture instantly applied to the entire room. Occasionally, during the course of level design, you’ll need to portalize the level before you see new textures applied in the 3D View window. When this is the case, DromEd will display a message to that effect on the bottom of your window. In fact, you should get used to checking the message bar on the bottom of the screen for important information. If something goes wrong during the course of your level design, an error message can often be found in that space.

At this point it’s quite possible that something has gone wrong, and clicking the “Put on Brush” button actually applied the selected texture to just one side of the room. If that happens, it’s because at some point you clicked on a face of the wall, thereby selecting that as the “active” face. Using the “Put on Brush” command doesn’t necessarily put a texture on the entire brush – it puts it on the selected face. It just so happens that when you first start DromEd, none of the faces is selected; instead, the program starts with the brush in “default” mode. Basically, imagine the default texture as the original color of an entire room. By adding different textures to different faces of the room (walls and ceiling), you’re not replacing the default texture, but “painting” over it. So, if you were to create an operation brush with a default brick texture, and then chose other textures for the walls, floor, and ceiling, the default texture would still be the brick you originally chose – it would just be hidden under the other textures.

It’s easy to tell which face you have selected. If you still have the texture palette up, remove it by hitting “Alt+T” again. Now, in the 3D View window, click one of the faces of the operation brush. It will highlight in orange, indicating that it is now the selected face. You can also cycle through the different faces by repeatedly hitting the comma (“,”) key on the keyboard. If you cycle through completely, so that none of the faces is highlighted in orange and the brush is outlined in solid white, then you once again have the brush’s default selected. To quickly see which face of the brush is selected, and which texture has been selected for that brush, look down to the bottom center of the screen, to the buttons that read “Face” and “Texture.” You can also use these buttons to directly apply textures to any face of the brush, including the default. To reset the default texture of the room, cycle through the available textures using the arrows to the left and right of the “Texture” button. When you’ve found one you like, hit the “Reset” button, found above the “Face” and “Texture” brushes. This will reset the default texture to the one you selected, essentially allowing you to start from scratch.

Now, hit “Alt+T” to bring the texture palette back up. Click a texture so that its name is highlighted in violet, and then click a face to see the selected texture applied to that face. Use this method to apply textures to the walls and floor. Then, hit the “R” key to look up at the operation brush’s “ceiling.” Click the “Sky” button (also found at the end of the

texture palette) so that its name is highlighted in violet, and then click the ceiling to place the star texture. Hit "Alt+T" again to remove the texture palette, and then press "Alt+G" to enter to game and take a look around. It should look as if you're standing is a tall-walled, open-air courtyard, with the starry night sky above. It's important to understand that, for level design purposes, the sky is just an illusion. It's actually just a ceiling with a modified texture, and not an actual, limitless sky. So, while it looks as if you could fly up to the heavens, you'd really smash your head against a relatively low ceiling (in this case, the room is still set to the default height – 16 feet). When you're done admiring your craftsmanship, hit "Alt+E" to enter back into DromEd.
T means T. Select a brush and type T. The brush will be textured by whatever is displayed. Type T again and you'll see the next texture applied to your brush

V. Resizing Brushes

Up until now, we've simply played around with the existing, default operation brush. Let's modify the brush to make it a bit bigger. Look at the lower left-hand corner of the screen and find the three buttons that read "D," "W," and "H." These stand for depth, width, and height respectively, and are measured in an approximation of feet. You'll notice, then, that the default brush is a perfect cube, measuring 16x16x16 feet. Let's leave the height at 16 feet, but modify the depth and width. Here we have a few options. The easiest way to change the size of a brush is to press and hold the control key, position the mouse cursor over one of the 2D windows, press and hold the left mouse button, and move the mouse to resize the brush. Move the mouse cursor over the "Top" view window and practice resizing the room. Note how resizing the brush in this manner only affects the two "visible" dimensions – width and depth. To use this method to resize the third dimension (in this case, height), press and hold the right mouse button and move the mouse left and right. Experiment with this method in all of the 2D view windows.
There are other methods of resizing brushes as well, particularly if you want to get more precise in your measurements. Try left clicking on the arrows to the left and right of the measurements to resize the brush in very small increments. It's also important to know that the "D," W," and "H" indicators (as well as "X," "Y," "Z," "H," "P," "B," and others) are buttons. Move the mouse cursor over the "D" indicator and click and hold the left mouse button; notice how the letter turns purple. With the left mouse button held down, move the mouse left and right to alter the brush's depth. You can use this method to alter any of the brush's dimensions. There is also another method for resizing a brush, especially for entering in precise measurements, and that is simply to type in the height of any given dimension. Let's use this method to make the room 24x32x16. Click the number next to the "D" button so that it highlights in purple. Then, simply type in the desired height in feet – in this case 24 – and press "Enter"; the depth of the brush is now 24 feet. Use this method to change the width to 32 feet as well. Now, portalize the level so that these changes take effect in the 3D world.

VI. Creating New Brushes

Now that we have an operation brush in place, let's create another operation brush from scratch and attach it to the existing one. In the lower left-hand corner of the screen, under the heading "Create," are several buttons, with the "Brush" button already highlighted. These buttons allow you to create new elements, like brushes, lights, and objects; the option to create a brush is the default. To create a new brush, choose the type of brush you want to create (in this case we want the default, an “air” brush). Left click on one of the 2D view windows and keep the button held down. Now, move the mouse to create a brush; when the brush is big enough, release the left mouse button to place the brush. Brushes can be repositioned in much the same way they can be resized. In the lower left-hand corner of the screen are indicators for a brush’s “X,” “Y,” and “Z” coordinates; you can use these brushes to modify the brush’s position. Or, press and hold the “Shift” key; move the mouse cursor to one of the 2D view windows; click and hold the left mouse button; and move the mouse to reposition the brush. Resize the brush so that its measurements are depth=8, width=8, and height=16. Then, use the “top” view window to reposition the brush so that its southern wall connects to the existing brush’s northern wall. After you’ve done that, move the mouse cursor over the “Right” or “Front” view window and reposition the new brush so that its bottom side aligns with the existing brush’s bottom side; this will essentially line up the new brush so that it’s on the same level as the existing brush (i.e. they share the same “floor”). It should now look as if you have an open-air courtyard whose northern wall connects to a corridor. To the north of the corridor, create a room whose southern wall connects to the northern wall of the corridor. When the brushes are lined up properly, portalize the level to see the changes reflected in the 3D View window. Feel free to play with the textures, and hit “Alt+G” to jump into the game and walk around the rooms as Garrett would. If everything went smoothly, you will have created an open-air courtyard, connected to a corridor, connected to a room.

Now, use this method to create a circular column somewhere in the northern room. To change the shape of the brush from a cube to a cylinder, go up to the menu bar and click “Shapes” then “Cylinder.” You can modify the number of sides in the cylinder (or in a pyramid, if you were to create one) by clicking “Shapes,” then “Sides in Base,” and entering in the number of sides you want (the default is 6). Let’s choose a cylinder with 10 sides.

Now that we’ve selected a cylinder, we need to tell DromEd to create a solid brush, as opposed to the air brush we just made. Look at the bottom center of the screen, to the button called “Op.” Use the arrows to the left and right of the button to cycle through the different brush types, and stop when the selection reads “fill solid.” [**Note:** “Fill Solid” is actually the first choice in the list] When this is done, create the solid brush somewhere in the northern room, using the same method you used to create the air brush. Resize the brush so that it stretches from floor to ceiling, and actually connects to the floor and ceiling. Portalize the level to see these changes reflected in the 3D View window, and enter into the game to see the column more closely.

VII. Placing Objects/Lightsourcing

You’ll recall that “light_bright” is still enabled, so the spaces you’ve created are completely illuminated in the 3D View window and in the game itself. Let’s change that by placing a torch in the game world.

Objects in DromEd reside in an “object hierarchy,” which can be accessed via the menu bar at the top of the screen. Click “Editors” then “Object Hierarchy” to bring up the object hierarchy. All of the game’s objects reside somewhere in this hierarchy, but for now we’re only concerned with placing a torch. Objects are located in one of five categories – “Sound,” “SFX,” “fnord,” “physical,” and MotArchetypes. Click the plus sign (“+”) next to “Physical” to expand that tree, then expand “Lights.” You’ll notice

that “Torches” can also be expanded; doing so will bring up an item called “ConTorch.” Sometimes, the name next to a “+” sign is simply a heading, and you need to expand that tree to see the objects you can place. In other cases, the headings are objects in themselves, as is the case with the “Torch.” Click “Torch” so that it’s highlighted in blue, and then click the “Create” button on the right-hand side of the object hierarchy window. You’ve basically just told DromEd to get a torch object ready for creation.
To place the torch in the game world, position the mouse cursor over one of the 2D windows, and click and drag to create a box, just as if you were creating a new brush. When you release the mouse button, the object will be created in the center of the box. Create a total of six torches – three in the southernmost room, near the west, south, and east walls, and two in the northernmost room, near the west and east walls. Now, examine the torches in the 3D View window – you’ll notice that their brackets are all facing a particular direction, south. That’s because all objects, when placed in the game world, “face” southward…but we can attach the torches properly to any wall by rotating them. [**Important Note:** look down to the lower left-hand corner of the of the DromEd screen and find the column of buttons under the “Create” heading. In order to create an object, you first need to click the “Object” button so that it’s highlighted in violet. It just so happens that when you select an object for creation out of the hierarchy, the “Object” button is automatically selected.]
Up until now we’ve changed a brush’s “X,” “Y,” and “Z” planes and modified its depth, width, and height. It’s also possible to modify a brush’s heading, pitch, and bank to rotate it into a desired position. Look at the lower left-hand corner of the screen, to the buttons marked “H,” “P,” and “B.” Use these buttons as you did the previous ones, and practice rotating one of the torches. Or, press and hold the “Alt” key, press and hold the left mouse button, and then move the mouse to rotate the torch. Rotate the torch so that its heading is 270 (but its pitch and bank are both 0), and affix it to the western wall. Use this method to rotate the remaining torches and place them on the other walls.
[**Important Note:** Some objects are actually complex composites of several smaller objects. The torch is a perfect example: it’s actually made up of the torch, the flame, and the smoke. When you move the torch, it’s important that you do indeed move the *torch*…and not the other elements. When you move the torch, it may seem as if the flame and smoke are left behind. Don’t worry – they’ll naturally follow the object. So, when you enter into the game by pressing “Alt+G,” you’ll notice that the torch looks just as it should. When you return to DromEd and look at the 2D view windows, you’ll see that everything is back to where it should be.]
[Note: When you move light sources around, you’ll often need to use the “Light” function to update the lighting. To do so, go to the menu bar and click on “Tools,” and then click “Light.” This will update the level’s light sourcing.]
Now that we’ve placed torches on the walls, let’s place a table on the floor, in the middle of the southern room. Tables, like all objects, reside in the object hierarchy; but instead of fishing through the hierarchy tree, let’s take a shortcut. If you know the name of the object, you can use the “find_object” command to instantly jump to its position in the hierarchy. Click on the command prompt, and enter in the command “find_obj table.” (without the period, of course) When you hit “Enter,” you’ll go directly to the standard table in the object hierarchy. You can either keep this one, or choose one of the available variations, like “Cabinet Table.” Create the table just as you did the torches, and place it

in the game world. Chances are, the table will be hovering a few feet off the ground. Instead of manually moving the table, you can have it automatically "grounded" by clicking the "Floor Me" button, located at the bottom center of the screen. [**Note:** "Floor Me" only works if the object you wish to have grounded is hovering in the air *over* the floor you want to have it grounded to. If, for example, you place a street lamp in the world, its base is sticking through the floor, and you want to have it aligned perfectly with the floor, you'll need to first raise the street lamp into the air and then hit the "Floor Me" button. Using the "Floor Me" button when an object is already sticking through a floor will have no (or worse, an undesirable) effect.]
Now, using the method you just employed for creating and placing the table, we'll create and place a guard in the northern room. [**Important Note:** If you try to create the default choice – "guard" – it actually places a white wedge in the game world. That's because "Guard" is a merely a heading in the object hieracrchy, and not an actual object. DromEd interprets "unknown" objects using white wedges. In this case, you need to click the "+" sign next to the heading "Guard" to expand that part of the tree, and choose a *specific* kind of guard.] Find and create a "sword guard" (under guards\swordsmen\grunts). Place the sword guard in the northern room. If you accidentally created a wedge, you can delete it by selecting it as the active object and hitting the "Delete" key. Note that you can use this method to delete *any* kind of brush, even entire "rooms."
Now that you're an expert with the object hierarchy, do this: place a table somewhere in the center of the northern room; place the object "BaffordScepter" (the scepter from Thief's Lord Bafford mission) in the room; and then rotate and move the scepter so that it's lying, horizontally, on the table. Now you've actually got something to steal.
Here's a really helpful tip for creating multiple objects: instead of creating another table from the hierarchy, we can simply copy the existing table. To copy a brush (including operation brushes, objects and lights), simply highlight the brush in question and hit the "Insert" key. This will create a copy in the same exact position as the original brush, with the copy now highlighted as the active brush. So, while it may look as if nothing happened, you've actually made a copy of the existing brush…it's just that the copy is in the same exact place as the original. If you go to move the original brush, you'll actually move the copy. So, use this method to copy the existing table and place the new table in the northern room.
You'll notice that the game world is still fully illuminated, because we've still got the "light-bright" command enabled. Now that we've placed some torches, we can disable this feature by typing in "light_bright" again and moving around in the 3D View window. Hit "Alt+G" to enter into the game to see the lighting effects more closely.

VIII. Bringing the Level to Life

The first thing we need to do is load in a script. A script is a basic set of commands, used by DromEd to activate different in-game elements, like specific AI behaviors (such as the factory workers in Cragscleft Prison.) Click on the command line and type "script_load convict" to load in the ConVict (**Con**ditions for **Vict**ory) script. This is the script DromEd needs to activate victory conditions (mission objectives), doors, and weapons. If you forget to load the script, these and some other in-game elements will fail to function, so it's good to get into the habit of loading in the script whenever you start a new map. The script will get saved with the level and you won't ever have to load it for this particular level again. Scripts are powerful tools, but since they are part of the

game's source code, script editing isn't available to the public. However, "ConVict" should be perfectly adequate for your level building needs.
By now you'll have noticed that entering into game mode doesn't quite have the desired effect: Garrett doesn't have a health meter; there are no sound effects; and the guard you placed is just sort of standing there, with his arms out. That's because DromEd recognizes the brushes you've created and the guard object you've placed…but doesn't yet recognize you as a player, or the brushes as rooms, and the guard's AI is unable to function because the level hasn't been processed for AI's yet.

The first thing we need to do is place the Starting Point, and tell Dromed to start the player model there. To place a Starting Point, go to the object hierarchy and expand the branches under the "fnord" tree until you come to "marker." (Don't expand past "marker.") Place the marker object in the southernmost room, in the northwest corner, and make a note of its number, which will be located in parentheses, after the name "A Marker" (found at the bottom center of the screen). When the marker is in place, look at the bottom center of the screen and find the "Properties" button; click it. This will bring up a separate "Properties" dialog box, with the marker's name – "A Marker" – already highlighted. Click the box's edit button and change the name to "StartingPoint." When the name has been changed, click the window's "Done" button to return to the main DromEd screen. Now, click the "Links" button at the bottom center of the DromEd screen, located directly below the "Properties" button. This will bring up the "Links" dialog box. Click the "Add" button, which in turn will bring up a small dialog box. This dialog box will have the following fields: "Flavor," "From," and "To." Click the arrow next to the "Flavor" field and select "PlayerFactory" from the drop-down menu. [**Note:** Use the "PlayerFactory" *without* the tilde (~).] In the "From" field, enter the number of the Starting Point you created. In the "To" field, type in the name "Garrett." Click the "OK" button when you're done, and then click the "Links" box's "OK" button to return to the main DromEd screen. Press "Alt+G" to enter into the game. Notice that you now enter into the game world in the precise spot you placed the Starting Point, and Garrett now has a health meter, as indicated by the small shield icons in the lower left-hand corner of the screen. Go back to DromEd, and we'll turn the operation brushes into actual rooms, complete with sound effects.
[**Note:** If you get killed by the guard (or if you accomplish all your objectives, but we'll get into that later in the tutorial), you'll need to exit the program and restart DromEd. To avoid this, place the line "no_endgame" (without quotation marks) in your user.cfg file. This will allow you to stay in game mode if you get killed or complete your objectives.]

To turn a brush into a room, at least as far as DromEd is concerned, you need to encapsulate that brush in a separate room brush. Look down at the bottom center of the DromEd screen, to the column of "Create" buttons. Click the last button – "Room" – so that it's highlighted in violet. With the button active, go to one of the 2D view windows and create a violet room brush around the brushes you've already created. Make sure you expand the brush in the other views so that it completely surrounds the existing brush (if it doesn't already do so). While it's okay to use this method for now, there's a more precise method of creating room brushes. Click on a brush in one of the 2D view windows so that it's active. Then, press and hold the "Shift" key and hit the "Insert" key; this will create a room brush around the selected brush. What you're probably asking

yourself right now is this: Why would I want to encapsulate all my individual brushes in separate room brushes, when I could just create one giant room brush around all the brushes I've created? Well, room brushes serve to separate the spaces realistically, particularly where Thief's sound effects are concerned. If you created a giant area, say one equivalent to the "Bonehoard," and surrounded it in one big room brush, all of the sounds in that room brush would appear to be in one big room. By encapsulating each brush in a room brush, you're helping to channel the game's sound effects in a much more natural way; sounds will appear to come from around corners; a guard at the other end of the map won't hear your footsteps; and so forth.
When you've created a room brush around the existing brushes, you need to build a room database so DromEd can examine the room/s and calculate sound propagation, etc. Go to the menu bar at the top of the screen and click "Tools," then click "Build Room Database." When this is done, hit "Alt+G" to enter into the game and walk around; notice that you can now hear sound effects, like the crackle of the torches and your own footsteps…but the guard is still standing motionless. Enter back into DromEd and we'll remedy that.
To activate AIs (guards, servants, monsters, etc.) you need to build a pathfinding database, which basically allows the AIs to find their way around the rooms you've created. Note that you must create room brushes and build the room database before you can build a pathfinding database. So, go back to the menu bar at the top of the screen; click "Tools," and then click "Compute Pathfinding Database." When that's done, enter into the game to see the Hammerite guard come to life. If you placed the Starting Point in the northwest corner of the southern room, and the guard in the northern room, you'll be out of his field of view and therefore out of harm's way. You way want to jump out, taunt him, and then run feebly around in circles before he bashes your brains in with that humongous warhammer. Yeah, you're right – that's not much fun. Let's give Garrett some weapons to even the odds. Go back to DromEd, and we'll equip the player model with a default inventory.
[**Important Note:** As you build more in-depth levels, there may be times when AIs cease to function as you'd expect: they continuously walk into walls, can't climb stairs, and refuse to enter certain rooms. If that happens, it's probably because you haven't updated the pathfinding database. Whenever you add new terrain to your level, the AIs don't know it's there unless you tell them…and you tell them by clicking "Compute Pathfinding Database" under the "Tools" section of the menu bar. Get into the habit of updating the pathfinding database whenever you build new terrain.]

IX. Acoustic Settings

DromEd supports environmental acoustic settings using EAX, or "Environmental Audio Extensions." This step is *optional*, but it will greatly enhance the game's audio effects. EAX effects are supported in DromEd's "game mode," and in *Thief* version 1.33 or higher. Keep in mind, however, that you need an EAX-compatible sound card to benefit from these settings.

DromEd allows you to set individual acoustic characteristics for every room brush. Simply select a room brush in the editor, choose a pre-set room type from the table below, and use the hot-key indicated on the table to set the room's EAX type. That's all there is to it.

The following table lists the various EAX settings; while some of the names may seem a bit arcane, you should get a general idea of what kinds of sounds each setting will produce. The "Small Dead" setting, for example, would be suitable for any small room with very little reverb, such as a walk-in closet. Pick the room type that sounds closest to the thing you want, and feel free to experiment.

Ctrl-F1	Small Dead
Ctrl-F2	Small Normal
Ctrl-F3	Small Live
Ctrl-F4	Large Dead
Ctrl-F5	Large Normal
Ctrl-F6	Large Live
Ctrl-F7	Dead Hallway
Ctrl-F8	Normal Hallway
Ctrl-F9	Live Hallway
Ctrl-F10	Tunnels
Ctrl-F11	Caverns
Ctrl-F12	Sewers

X. Creating a Default Inventory

Creating a Starting Point basically creates an in-game representation of Garrett without any inventory. To start a mission with a default selection of items (weapons and powerups), you need to create those items and then "link" them to the Starting Point. Search through the object hierarchy or use the "find_obj" command to create and place in the game world a blackjack, sword, and broadhead arrow. [**Note:** The broadhead arrow is called simply "broadhead."] After the objects are created, make a note of each one's object number.

We'll start by linking the blackjack to the Starting Point. Select the Starting Point, and click the "Links" button, just as you did before. In the "Flavor" field, select "Contains." In the "From" field, type in the number of the Starting Point (or just "StartingPoint"). In the "To" field, type in the number of the blackjack. Click "OK" and then click "OK" on the "Links" window to return to the main DromEd screen. You have just linked the blackjack to the player model, meaning you will start the game with that weapon. Don't worry about the blackjack appearing in the world as an actual object during the course of the game: when you create a "contains link," the player contains that object in his/her inventory, and it no longer exists in the game world. Now that you've linked to the blackjack, use the same procedure to link to the sword (make sure you use the correct object number for each object you link to).

Linking the broadhead arrow to the player is a bit more complex, because we want to give Garrett multiple arrows, and not just the one we placed in the game world. Before we link the broadhead arrow to the Starting Point, we need to modify the broadhead object so that it gives the player multiple arrows. Click on the broadhead arrow, and then click the "Properties" button to bring up the "Properties" window. Then, click the "Add" button to bring up a list of possible properties. Choose "Engine Features" then "Stack Count," and type in the number of arrows you want to start out with. Let's choose thirty (30) arrows. When you've finished changing the number of arrows, link the broadhead arrow object to the Starting Point just as you did with the sword and blackjack. Note that you don't actually have to equip garret with a bow; once you acquire arrows, the bow is automatically available.

Anyone's who's played Thief knows that one of the game's coolest elements is the ability to extinguish torches using water arrows, to create your own beautiful, concealing darkness. Using the methods already described, give Garrett two water arrows. It's important to note that, for the purposes of creating levels with DromEd, water arrows (and the same is true for moss arrows, fire arrows, and gas arrows.) are actually referred to as *crystals*, and are found in the hierarchy under Physical\Tulz\Crystal.

Now that you've got an inventory, go back into the game world and show that Hammerite who's boss!

[**Note:** If you use your two water arrows to extinguish the two torches in the guard's room, that room will be plunged into complete darkness, as those two torches are the only sources of light!]

XI. Creating AI Patrol Routes

The AIs in Thief: The Dark Project, as evidenced by the guard we placed in this tutorial level, are very intelligent: although they remain stationary when nothing's going on, they can hear your footsteps, hunt you down, and even run away when threatened. These are all behaviors that are inherent to the AIs, and you as the level designer don't need to worry about controlling them. And, while complex scripts are not available, you can modify AI behavior by assigning patrol routes. Basically, a patrol route allows an AI, let's say a guard, to walk a predefined circuit, by travelling from one marker (similar to a waypoint in a flight sim) to another, to another, and so on.

To define a patrol route, you need place a circuit of markers to indicate that route. In the object hierarchy, go to fnord\Marker\TrolPt. Place four TrolPt markers (**not** LookBackPt markers) in a square pattern around the room; these are the markers the guard will follow on his route. Now, it's important to know that an AI will start a patrol by heading to the TrolPt marker that is closest to him in 3D space. How do we make sure the AI starts the patrol at a particular marker? Simple: we create the first marker (or move it, if it was created somewhere else) directly inside of the selected AI. When placing markers, it's important that they always exist inside in the game world (meaning, inside an air brush), close to the floor (up to around 4 feet) where the AI will walk.

Now that the markers have been created, we need to link them together. Click on all the markers and make a note of their object numbers. Then, click on the first one – it should be inside the guard – and bring up its "Links" box (we're basically going to do something very similar to linking the default weapons to the Starting Point). Click the "Add" button to bring up the smaller dialog box. In the "Flavor" field, use the drop-down menu to select "AIPatrol." [**Note:** Do **not** use "~AIPatrol." Names with the tilde (~) in front of them are "return" links, and we'll discuss them in just a bit.] In the "From" field, put the number of the currently selected marker. In the "To" field, put the number of the next marker you want the guard to travel to. We placed the markers in a square pattern around the room, so we want the guard to follow a square patrol route. So, in the "To" field, put the number of the next marker in the square patrol route. Use this method to link all the markers together. Remember that we want the guard to walk in a continuous loop, so, when you get to the last marker, link from it back to the first marker. That way, when the guard gets to the last marker, he'll head back to the first marker and start his patrol all over again.

There's one more thing we need to before the guard will follow the markers, however. When you first place an AI, it will remain stationary because that's its default position. Although you've placed the TrolPt markers, you haven't told the AI to follow them. To change this, select the guard, and then bring up his "Properties" box. Click the "Add" button, and then select "AI," "Ability Settings," and "Patrol: Does patrol." This will bring up a small dialog box containing a small, white, unchecked selection box. Click on the selection box to place a check mark there, and then click the "OK" button to close out the dialog box. Then, click the "Done" button on the "Properties" box to return to the main DromEd screen. By checking the box, you have enabled that AI's ability to go on patrol.

You can also enable an AI's "random patrol," command, so that instead of following the TrolPt markers in order, the AI chooses a path at random. To do this for the guard, follow the same steps you used for enabling "Patrol: Does patrol," but select "Patrol: Random sequence." This will enable the guard's ability to patrol at random. [**Note:** "Patrol: Random sequence" is found on the list directly underneath "Patrol: Does patrol."] It's important to note that if you do decide to activate the "Patrol: Random sequence" ability, you still need to activate the "Patrol: Does patrol" ability so the guard knows he's supposed to patrol in the first place.

You may notice that when you open up a marker's "Links" box and go to link from that marker to the next, there's already another link there. It looks similar to the one you're about to create, but there's a tilde (~) at the beginning of the "Flavor" name, and the "From" and "To" links are different. The links with the tilde are called "return" links. Basically, you as the level designer create a link from one marker to the next marker. But DromEd recognizes that one marker also links to a *previous* marker. While you may create three markers, numbered 23, 24, and 25, and link from one to the next, DromEd goes one step further by linking them backward as well. So, let's say you create a link from marker 23 to 24. Then, you create a link from marker 24 to 25. You have specified that marker 24 links to 25…but DromEd has already linked marker 24 back to 23.

Return links are very similar to normal links except for the following: their flavor names begin with a tilde; they point in the opposite direction; and they have the opposite meaning. Say, for example, you have a chest, and a "contains" link from the chest to a potion. What the link means is that, "The chest contains the potion." The potion can't be seen in the game, but when someone opens the chest the potion goes into his or her inventory. At the same time the "contains" link is created from the chest to the potion, a "~contains" link is created automatically from the potion to the chest. This link means, "The potion is contained by the chest." The important thing for you to remember about return links is that although you will see them frequently, their placement is a normal occurrence that gets managed automatically by DromEd. You don't have to worry about them at all.

Now that you fully understand the concepts of linking, enter into game mode and watch as the guard patrols around the room. See if you can sneak up behind him and knock him out with the blackjack before stealing the scepter. [**Important Note:** If you enabled the "Patrol: Random sequence" ability, you may notice that at some point the guard walks into the column, and just continues walking into it, instead of trying to go around. That's because you need to update the pathfinding database, as mentioned earlier in the tutorial.]

XII. Creating Mission Objectives

If you've followed this tutorial all the way through, from beginning to end, you should now have the knowledge to make fairly detailed Thief maps. You can create terrain, place objects, give yourself weapons, and even define patrol routes for your AIs. Now we come to the most advanced (read: confusing) part of this tutorial: creating mission objectives. If you enter into the current level, you can kill the guard and steal the scepter…but then you're just sort of stuck there, hanging around with nothing to do. In other words, you have to "pretend" that stealing the scepter is your goal, because DromEd doesn't yet recognize that (or anything else) as an actual objective.

In order to assign objectives, you must specify a mission's quest data. Go to the top of the screen, to the menu bar, and click "Editors." Then, click "Mission Quest Data." Don't worry – nothing is supposed to happen. That's because you haven't yet specified any quest data. When you *have* specified this information, clicking "Mission Quest Data" will bring up a small "Quest Data" dialog box, containing all the objective and difficulty information for that mission. So, we need to enter in this information. Understanding these concepts, at least for the purposes of level creation, is a bit tricky…but this tutorial should at least teach you the basic principles needed for the creation of real missions, with real objectives.

In order to implement objectives, you must first understand how DromEd views those objectives. So, this section of the tutorial will first explain the concepts behind objectives and difficulty levels, and then teach you to actually implement them in your missions.

First and foremost, every objective you include must be given a number, and the first objective **must** be given the number 0. So, if you have three objectives, they would be numbered 0, 1, and 2. Again, they **must** be numbered in this way.

Next, each objective has a "state," which indicates whether or not it has been completed or not. DromEd recognizes four states for objectives: "incomplete," "complete," "inactive," and "failed." If you've played Thief, you will have seen, on the "Objectives" screen, visible representations of each of these states: "incomplete" is marked by an empty box, meaning that objective has not yet been fulfilled; "complete" is marked by a green check mark, meaning that objective has been satisfied (like when you steal an objective item); "inactive" is marked by a red circle with a line through it, meaning that objective is no longer active or applicable; and "failed" is marked by a red "x" (though at that point, because of the failure of an objective, the mission will usually end before you can even notice this mark on the "Objectives" screen).

DromEd uses the following numbering convention for objective states:

0=incomplete
1=complete
2=inactive
3=failed

When making Thief levels with DromEd, each objective **must** be marked with a 0 (zero) for "incomplete," meaning that at the beginning of a mission, each objective has yet to be

accomplished by the player. These objectives will be marked either “complete” or “failed” according to the actions of the player.
The next thing a budding level designer needs to know is that in Thief, objectives can be either visible or invisible. Visible objectives are those the player can see on the “Objectives” screen right at the start of the game; so, when starting a Thief mission with visible objectives, the player knows exactly what he or she is supposed to do. A good example of a visible objective can be seen in Thief’s “Lord Bafford’s Manor” mission, where the player has to steal Lord Bafford’s scepter: “Steal the jeweled scepter with as little notice as possible.” An invisible objective, on the other hand, is one the player is not aware of at the start of a mission. A good example of an invisible objective can be seen in Thief’s “Assassins” mission, where the player, after tailing two would-be hit men back to Ramirez’s mansion, has to then steal the purse from Ramirez’s belt. The objective to steal the purse is an invisible objective, because it doesn’t show up on the player’s “Objectives” list at the beginning of the mission; instead, that objective is added after the player tails the two assassins back to the mansion. Subsequently, if you set off the alarm during the course of the mission, exiting Ramirez’s mansion will give you yet another objective: to make it back to your “home turf.”
For the purposes of designing Thief missions with DromEd, it is imperative that you realize something about mission objectives before you actually begin the creation process: the “ConVict” script **only** supports visible objectives! So, you could not create a mission with the complexity of “Assassins,” because all of the objectives **must** be visible to the player at the start of that mission. So, you could easily create a mission that required the player to enter a certain building, steal a certain object, and then retreat to a certain location…but the player would have to be aware of all of these objectives at the beginning of the mission. You could not, however, create a mission that required the player to steal a certain object…and then gave that player a new objective after the object was actually stolen.
DromEd uses the following numbering convention for objective visibility/invisibility:
0=invisible
1=visible
So, as stated, each objective must be numbered 1, for “visible.” [**Important Note:** By now, you may be wondering exactly *where* these numbers come into play. Don’t worry – that will be explained shortly.]
You now know that, for the purposes of level design with DromEd, each objective has a “state” (incomplete, complete, inactive and failed) and a visibility (visible, invisible). Well, there are obviously different *kinds* of objectives as well. The public release of DromEd supports four different kinds of objectives, with the following numbering scheme:
1=Steal an object
2=Kill a creature
3=Get a certain amount of loot
4=Go to a location
[**Note:** Objectives are covered in more detail in the “ConVict” document. There are some variations of the objectives you see here. For example, you could easily have an objective that *prevents* you from killing a certain creature or creature type. Also, in order

to learn how to create loot objectives (ie., steal 500 gold pieces on “Normal” level), you **must** refer to the ConVict document!]]
You could include just one or all four of these objective types in your mission. So, the player could be required to steal an object…or the player could be required to steal an object, kill a creature, get a certain amount of loot, and go to a certain location.
You can even use multiple objectives of the same type, if you wish. For example, the player could be required to steal two different objects, kill three different monsters, and then go to a certain location.
So, how do you enter all this data into DromEd? The first thing you should do is write down on a piece of paper all the objectives you want, starting with the number 0. Using the sample level we’ve made, let’s have two objectives: stealing the scepter and killing the guard. This would read as:

0=Steal the scepter
1=Kill the guard

Now, before we can tell DromEd what the actual objectives are, we have to indicate that each one will be “incomplete,” and “visible,” as already stated in the tutorial; each of these commands must be entered in separately. The command used for entering in objective data is “quest_create_mis” -- without the quotation marks, of course. Get used to this command, because we’ll be using it quite often.

The command for entering in the “state” of an objective is “goal_state_x, 0” – where “x” is a variable representing the number of that particular goal, and “0” is the number corresponding to the state we want – incomplete. So, the entire command you would need to enter (in the command line) is exactly as follows:

quest_create_mis goal_state_0, 0

[**Important Note:** The first number is always the number of the objective you’re referring to, and the second number is always the number that corresponds to a specific DromEd command. Always use the above format: number, comma, space, number.]

After this line is entered in, go back up to the menu; go to “Editors,” and “Mission Quest Data.” You’ll notice that the small “Quest Data” dialog box now comes up, and lists the info you’ve entered. Click the “Cancel” button to close the window.

[**Note:** You can use the “Quest Data” window to directly edit variables you’ve entered in. Just highlight the command, click “OK,” and you can change the command’s “Name” and “Value.”]

Confused? Let’s dissect that line to see what you’re telling DromEd to do. “Quest_create_mis” is the standard command for entering in objective data. Then, you use a space and type in the rest of the command. “Goal_state_0, 0” means that objective 0 (steal the scepter) is going to be marked with a 0, meaning it is incomplete. What all this means is that when you start the mission, the objective “Steal the scepter” will have an unchecked box next to it on the “Objectives” screen, because that objective has yet to be fulfilled.

Of course (now things get even trickier) the objective “Steal the Scepter” won’t appear on the “Objectives” screen at all unless you mark that objective as visible, meaning the player can see it right at the beginning of the mission. To mark our first objective as visible, use the following command exactly as you see here:

quest_create_mis goal_visible_0, 1

Hopefully, by now you are gaining an understanding of how objective data is entered into the command line. With the above line, we told DromEd that objective 0 (steal the scepter) would be marked with a 1, meaning it is visible at the start of a mission.

[**Important Note:** So, in case you haven't already figured it out, you must enter in the "goal_state" and "goal_visible" commands for **every** objective you have, and each one **must** be made "incomplete" and "visible."]

Now that we've set the state and visibility of the first goal (steal the scepter), follow the examples above and set the state and visibility of the second goal (kill the guard).

[**Important Note:** If you screw something up and need to delete a comand you've entered in, use the command "quest_delete" and then the command you entered. So, if you wanted to delete the line "quest_create_mis goal_state_0, 0" you would type in:

quest_delete goal_state_0

You would **not** use the comma and second number because you're not trying to set anything for goal 0 – you're simply trying to delete goal 0. So, you need only indicate the number of the objective (goal), which in this case is 0, for our first objective, "Steal the scepter."]

Okay, we've told DromEd that we've got two objectives, and that each one will be visible and marked as incomplete at the beginning of the mission…but we haven't yet told DromEd *what* those objectives are. For each objective, we must enter in two commands: one to indicate what the objective is, and one to indicate what the *subject* of that objective is. So, for the first objective – "Steal the scepter" – we need to tell DromEd two things: 1.) The player needs to steal an object; 2.) The object the player needs to steal is the jeweled scepter. To do this, we use the commands "goal_type" and "goal_target."

Type in:

quest_create_mis goal_type_0, 1

The above line tells DromEd that objective 0 will require the player to steal an item.

Now, find the scepter object you placed on the table, click it, and make a note of its number. Then, type in:

quest_create_mis goal_target_0, x (where "x" is the number of the scepter)

The above line tells DromEd that the subject of objective 0 (which we just defined as "steal an object") is the scepter.

Using the examples above, let's enter in the data for the second objective – "Kill the guard." First, we would use the following command:

quest_create_mis goal_type_1, 2

The above line tells DromEd that objective 1 will require the player to kill a creature.

Then, we would use:

quest_create_mis goal_target_1, x (where “x” is the number of the guard)

The above line tells DromEd that the subject of objective 1 (which we just defined as “kill a creature”) is the guard.

Before you can get the objectives enabled, there’s one more thing you must do, and that is to assign the “VictoryCheck” script to your Starting Point. Click on the Starting Point to select it, and then click the “Properties” button to bring up the “Properties” window. The, click “Add,” and choose “S,” and then “Scripts.” This will bring up a small “Scripts” box with a few different fields. In the “Script 0” field, type in “VictoryCheck” (one word – without the quotation marks) and then click the “OK” button. Click the appropriate buttons to close out the windows and return to DromEd.

Now that you’ve actually got real objectives, enter into game mode by hitting “Alt+G.” Once there, hit the “O” key to bring up the “Objectives” screen. You should see two small, empty boxes to the left of the screen, with no text next to them. Those objectives are, in order from top to bottom: Steal the scepter; and kill the guard. The reason there are no text descriptions of the objectives is because this information hasn’t been entered in yet. Don’t worry – we’ll come to that a bit later. For now, it’s enough to know that we have two objectives: the first requires you to steal the scepter; the second requires you to kill the guard. Hit the “Done” button on the “Objectives” screen to return to game mode. Then, kill the guard – either nail him with an arrow from a distance, or get behind him and backstab with the sword. [**Note:** If you knock the guard out with the blackjack, he’ll be merely unconscious, and not dead. To finish him off, you’ll have to hack away at his unconscious body with your sword.] When the guard is dead, return to the “Objectives” screen and note that there is now a green check mark in the second box. That’s because you’ve completed the second objective – “Kill the Guard.” Exit the “Objectives” screen and steal the scepter – the mission will end, meaning you also completed the first objective – “Steal the scepter.”

XIII. Creating Difficulty Levels

If you’ve made it this far, you’re to be commended! Learning how to implement real objectives into your missions is the test of a true Thief level designer. And, if you can do that, you can take the objective formula even further and incorporate difficulty levels into your missions.

The first thing you need to realize when creating different difficulty levels is that you first need to create all the objectives, as discussed in the previous section of this tutorial. So, even if you have objectives that only show up on certain difficulty levels, those objectives need to be created before you do anything else. Since we already have two objectives in place, let’s use those to illustrate the differences in difficulty levels. Right now, our objectives are: “Steal the scepter” and “Kill the guard.” Whenever we enter the game, we must accomplish these objectives. But let’s modify things a bit, so that the “Kill the guard” objective is only required when we play the game on “Expert” difficulty level.

DromEd recognizes three different difficulty levels, with the following numbering scheme:

0=Normal

1=Hard

2=Expert

To tell DromEd which objectives will be available on which difficulty level, there are two different commands we can use: "goal_min_diff" and "goal_max_diff." "Goal_min_diff" is the *minimum* difficulty that objective would be available. "Goal_max_diff" is the *maximum* difficulty that objective would be available. So, let's say the "Normal" and "Hard" difficulty levels required you to steal 300 gold worth of loot, and the "Expert" difficulty level required you to steal 500 gold worth of loot. The goal "Steal 300 gold worth of loot" would use the "goal_max_diff" command, set to the number 1, for Hard difficulty. So, basically, this would tell Dromed that the goal "Steal 300 gold worth of loot" would apply to all difficulty levels, up to the maximum difficulty level of "Hard." We don't want to apply this objective beyond the "Hard" difficulty level because we want the "Expert" level to require stealing 500 gold worth of loot. To specify that objective, we would use the "goal_min_diff" command and set the number to 2, for "Expert" difficulty. This would tell DromEd that the goal "Steal 500 gold worth of loot" would only apply to difficulty level of "Expert" and above. Of course, there is no difficulty level higher than "Expert," but that's okay. That just means the program will apply that difficulty level to *just* "Expert," which is exactly what we want.

Let's get back to our tutorial level. Remember, we want the objective "Steal the scepter" to apply on all difficulty levels, and the objective "Kill the guard" to apply only on the "Expert" difficulty level. So, for the "Steal the scepter" objective, we wouldn't need to specify any conditions. For the "Kill the guard" objective, however, we would need to enter in the following command line:

quest_create_mis goal_min_diff_1, 2

The above line basically tells DromEd that objective 1 ("Kill the guard") will be applied to the minimum difficulty level of 2, which is "Expert."

[**Note:** Remember, "Kill the Guard" is objective 1, because objective 0 is "Steal the scepter."]

Now it's time to test out the difficulty levels. Normally (and this is something that will be covered in the next section), a person playing your mission would select a difficulty level through the pre-mission screen, just as they would when playing any Thief level. But during the course of level design, you need to be able to check your work in DromEd. You do this by using the difficulty level selection command. To play the mission on a particular difficulty, type in:

quest_create_mis difficulty, x (where "x" is the difficulty level you want to play at)

So, to play the tutorial mission on the "Normal" difficulty level, type in:

quest_create_mis difficulty, 0

Now, enter into game mode and hit the "O" button to bring up the "Objectives" screen. You should notice that there is now only one small box…meaning there is only one objective. That's because we're playing on the "Normal" difficulty level, and we set the "Kill the guard" objective as an "Expert" difficulty level. So, on "Normal" difficulty, this objective wouldn't even be available. If we had wanted to play the mission on the

“Expert” difficulty level, thereby enabling that second objective, we would have used the command:

quest_create_mis difficulty, 2

You should only use the “set difficulty” command to test out a particular difficulty level through DromEd. If you set a particular difficulty level, and then save the file, the mission will be “stuck” on that level, even if the player chooses another difficulty level on the pre-game screen. So, if we set the tutorial mission to “Expert” difficulty and then saved the mission, anyone who played the mission would be forced to do so at the “Expert” level. If you *do* save the mission while it’s set to a particular difficulty level, you must delete the difficulty command and resave in order for the mission to operate correctly. So, you would use this command:

quest_delete difficulty

XIV. Other Helpful Commands and Tips

If you’ve followed this tutorial from beginning to end, you should now have the ability to build basic Thief 2 maps. By experimenting with DromEd, you can discover the program’s full capabilities. But here are a few helpful commands and tips to get you started.

- To learn more, remember that you can load up Thief levels with DromEd and poke around to see how some complicated things were created. You can probably learn a lot, but remember a few things:
    - You might not have the right tools available to do the all of the things that were done in the game. Try not to beat your head against the wall.
    - It’s not necessarily true that everything in all Thief levels was done perfectly and optimally. Some examples may be better than others.
- When you design Thief missions, it often helps to disable the game’s gravity so you can “fly” around the game world and examine your work more closely. Hit “Shift+Q” to disable gravity, and “P” to turn gravity back on.
- Similarly, you can toggle the ability of AIs to see you with the command “aiawareofplayer,” (again, no quotation marks or comma) and toggle player collision detection with the command “physics.” With regard to toggling “physics” on and off, remember that even though you can now fly through solid space, doing so can lead to weird errors (although it is unlikely to fatally corrupt your work, format your hard drive, or anything permanent like that).
- Use the “show_stats” command to enable in-game statistics, like frame rate and polygon count. When creating Thief levels, you want to keep the polygon count low – shoot for 250 or below – to maximize game speed.
- To start DromEd with different default settings, you can place the following lines in your user.cfg directory: :
    - “edit_screen_size 800,600” – This will start DromEd in 800x600 mode.

    - "editorcam_from_game – When you place the Starting Point, you will start the mission at this point every time you enter into the game. With the "editorcam_from_game" command, you can enter into the game at the same point you were at in DromEd.
    - "no_endgame" – This allows you to continue playing if you're killed in the game world or complete all your objectives.
- When you first start DromEd, the program defaults to a grid size of 16. 16 is fine for creating large brushes like rooms, but in order to make smaller brushes, you'll need a smaller grid. Find the grid field in the lower left-hand corner of the DromEd screen – try using a size 11 grid for smaller brushes, like creating small columns and solid details.
- On the menu bar, choose "Tools" and then "Optimize" to optimize the level. This essentially "tightens" things up, by reducing your polygon count and making things run more smoothly. Keep in mind, however, that "Optimizing" takes longer than "Portalizing."
- While you, as Garrett, can "mantle" up ledges and walls, AIs have much more restricted movement; they cannot climb up terrain obstacles that are much higher than a foot or two. This is most important to remember when building stairs. To make sure AIs can climb stairs, the stairs should be 1ft. deep by ¾ foot high.
- Get into the habit of saving your work regularly. On the menu bar, choose "Save Mission" to save your level in progress. DromEd may crash on you, and it may not always be your fault. Saving helps to avoid frustration.
- If, during the course of level creation, something seems wrong, try:
    - Portalizing
    - Building the Pathfinding Database
    - Lighting
- DromEd does some automatic color-coding to help you tell which brushes are closest to the camera. If you don't like it (because you can't see the lines very well), try pressing "Ctrl+7" to disable it.
- Normally your camera and the 2-D grids are synchronized such that the camera is always in the center of the grid. The menus you get when you right click on one of the 2-D view screens include the option "asynch all," which desynchronizes this. To resynch, select "synch all" from the same menu.
- The command "cam_to_brush" will teleport your camera to the currently selected brush. This works well in conjunction with "find_obj x" where "x" is the object's name or number.
- The full list of texture families is: dungeon, city, church, cave, catacomb, concastle, bafford, sewer, rescore, temple, mine, ruined, metals, core, ancient, maw, ramirez, lostcty, newcity, keeper, mech, newkeep, basement, tower2, waterhw, skyhw, sky, porttest, water

- Lighting – There are three lighting modes: Quick, Raycast, and Objcast. You can choose between them by selecting “Tools” on the menu bar. Quick lighting is the default. Raycast lighting is more realistic but takes longer. With Objcast lighting, some stationary objects will cast shadows.
- Ambient Lighting - Initially, the ambient light is set to 0, so unlit areas are completely dark. To set it to a higher value (say, 20), use the command “ambient 20.” Use the dialog box by replacing the word “intensity” with the numeric value.
- Properties and Links – As you can see by looking at the “Properties” dialog box and “Links” dialog box, there are lots of links and properties in DromEd. These are one way in which a designer can get interesting and complicated things to happen in missions. However, some properties and links may seem to have redundant effects, or no effects at all. Furthermore, some of them are intended for use by the designer while others are intended to be managed automatically during the game (and thus could be very problematic if set by the designer). We can’t cover these in any detail here, but as always you are encouraged to check out Thief missions for examples. Below are brief descriptions of some useful properties.
  - Game::Damage Model::Hit Points – Determines how much damage an AI (and some objects) can take before dying.
  - Engine Features::Locked – When placed on a door or chest and set to “true,” this makes the object locked.
  - Engine Features::Key Src – When placed on a key, this specifies which lock (or locks) the key opens.
  - Engine Features::Key Dst – When placed on a locked object, this specifies which keys fit the lock.
  - Dark Gamesys::Pick Cfg – This is how you specify that (and how) a locked door can be lockpicked open.
  - AI::AI Ability Settings::Combat:Non-hostile – Controls whether the AI is hostile to the player.
  - AI::AI Ability Setting::Flee: Conditions for Flee – Controls when the AI will flee and when he will fight.
  - AI::AI Core::Alertness Cap – Can be used to make an AI frozen at a particular alertness level or range thereof.
  - AI::Utility::Flee Point – When placed on a marker object, it specifies that location as a good one for AIs to flee to. The value ranks the location from 0 to 100.
  - Difficulty::Destroy – Putting this on an object and selecting 0, 1, and/or 2 will cause the object to disappear for all selected difficulty levels. Note: This only happens when you enter the mission normally through the pre-mission screens. Optionally, you can run the command “process_difficulty”, but only do this when you are in the game, not the editor!

- Timing terrain creation – During Portalization, operation brushes are processed in a specific sequential order. Press Tab to cycle between all of the brushes in your level, and look at the lower-left corner of the editor to a field called "Time." Time starts at 0 and goes to the last brush in your level. Note that "fill" operations (such as "fill water") replace all other terrain within their brush dimensions. Therefore, if you make a neat statue out of "fill solid" brushes stacked like building blocks, then place a "fill air" brush such that it covers the whole statue, the statue will disappear when you portalize. This is because the "fill air" operation is processed later in time than the many "fill solid" operations that make up the statue. To fix it, you could move the "fill air" brush to earlier in time, before any of the "fill solid" brushes take place. You can do so by clicking in the time value box and typing in the new number, or pressing the "<" and ">" buttons next to it. If you are having trouble making the shapes of terrain that you want, pay attention to what sequential order the operation brushes are processed in. Lastly, note that the conversion brushes (such as "water->solid") do not necessarily effect all terrain within their brush dimensions.
- Always using a grid of size **11 or larger** will help to prevent a large variety of potential problems caused by irregular sizes, shapes, and rotations of terrain. The grid is "on" when the "Use" button (found below the grid number) is purple (not gray). If you even get weird error messages and/or crashes from the renderer (which you may recognize because they mention things like "polyhedrons," "portals," "planes," etc.), try reducing the complexity of your mission's terrain by using operation brushes that are more regularly shaped. Go through and grid-snap all of your brushes by moving them slightly when the grid is on. Furthermore, remember that very high poly counts can lead to problems, so don't let your lines-of-sight get too long or too detailed. In addition to causing renderer problems, complex terrain can potentially get the player stuck and confuse the AI. Use "hilight_check_snap" and "hilight_do_snap" commands to snap unaligned brushes to the nearest grid line.
- Filtering – To the lower right of the "Create" button is a "Filter" button. Sometimes DromEd's 2D windows get crowded and you want to look at only objects, or only terrain, etc. When you press "Filter" you are presented with a number of buttons you can use to toggle on and off various types of brushes ("Terr", "Light", etc.). Remember to toggle them all back on before portalizing, lighting, etc.
- Area brushes – Area brushes are another way to do filtering, this time by proximity rather than type. Area brushes are cubic brushes that you can create by selecting them on the "Create" menu, then dragging in a 2D window. When an area brush is the current brush, you can use it to filter. Press "Me Only" to show only those brushes that are inside the area brush. Press "Not Me Only" to disable that filtering. You can also name area brushes, select them by name (with the "Search" button), and activate some area brushes and not others. Among other things, area brushes are useful for speeding up portalization when you are focusing your work on a small area, such as a single room. Again, make sure you shut off all filtering before trying to portalize, light, etc., the whole level.

- Multibrushes – Multibrushes are a good tool for cutting-and-pasting brush groups. So, for example, if you make a row of pillars on one side of a room and you want to have the same exact row of pillars on the opposite side of the room, you could multibrush them, press Insert to make a copy of the multibrush, then move the new multibrush to the other side of the room. To add brushes to a multibrush, hold down "Shift" while selecting a brush. Remove brushes from a multibrush in the same way. You will note that when you are in multibrush mode, the word "multibrush" appears where the textures normally appear, at the bottom of the DromEd screen. To get out of multibrush mode, select a brush that is not in the multibrush while "Shift" is *not* pressed. The "Multibrush" menu gives you other multibrush functionality, such as saving and loading multibrushes, which is especially useful for porting multibrushes between two different missions. Lastly, multibrushes were primarily designed with terrain brushes (i.e. operation brushes) in mind, but also work to a degree with other brush types.

- Room Brushing – Room brushes are your way, as the designer, to specify the "general shape and form" of your 3D terrain for DromEd. A hallway may have a carpet running down the middle, skylights above, columns along the sides, and some decorative trim along the top that sticks out a little. All of these things may change the 3D shape in small ways, but with regard to sound propagation, you want to specify the "general shape and form" as an elongated cubic shape with the same location and dimensions of the hallway. The easiest way to so is as follows. When you made the hallway, you probably started with a long "fill air" brush and then added other brushes for the details. Select that first "fill air" brush, then hold down "Shift" and press "Insert." DromEd will make a room brush that is just like the original brush, but slightly bigger. Another useful tool is the "show sel" button, which will draw lines to show you how the currently selected room brush is connected (for the purposes of sound propagation) to adjacent rooms. Be warned that good room brushing can take a lot of time and effort, but can make the sound in your levels much better. Here is the list of room brushing rules.

    - All air and water space in your levels must be contained within at least one room brush. This is particularly important for the planes of the floor, from which all footstep sounds originate.

    - Room brushes are allowed to contain solid space (this is only for convenience of placing room brushes; remember that no sounds can actually originate inside solid space).

    - When two room brushes overlap, sound will propagate between them. Use the "show sel" button to confirm that two room brushes intersect.

    - Room brushes that are rotated or that intersect with other room brushes in strange ways may not propagate sound correctly. Use the "show sel" button to see if an intersection problem has occurred. You will know there's a problem if the room brushes intersect in 3D space but do not have a line between them when you press "show_sel." If so, try changing the shapes, positions, and rotations of your room brushes. Use more room brushes if necessary.

    - The center of a room brush must never be contained within another room brush.
    - Don’t forget to “Build Room Database” to process your room brushes.
    - If you include “check_rooms” in your user.cfg, some useful diagnostics may happen when you build the room database. The first thing it will do is highlight in yellow all room brushes whose center is inside another room brush. You should fix all of these room brushes. If no such room brushes occur, the command will highlight in yellow all room brushes that intersect with other room brushes inside solid space. These are not technical problems, but you might want to check to make sure you really meant for those room brushes to intersect.

- Water brushes – To create water, try using the “fill water” and “flood” brushes. Two things you need to know: 1.) Water brushes are like air brushes, except that they fill the entire brush with water. This water will erase any solid you may have already created there. 2.) Flood brushes fill an existing air brush with water, but do *not* delete any solids that are already there. To get the water to look like water, you will need to make sure it is flow brushed (see below).
- Flow brushes – Flow brushes are how you make water in a Thief level move, look, and behave like water. They are fairly complicated, so they will only be touched on here. A flow brush applies to the water space that the center of the flow brush is in and sometimes water space adjacent to that as well. The boundaries of the flow brush should contain the surface of the water. Each flow brush is belongs to a “group.” To edit the properties for a group of flow brushes, click on the flow brush to select it, and then look down to the bottom center of the screen. Click the “Edit Group” button to bring up the properties box for that group. Different flow brush “groups” have different flowing properties, such as “x change/ sec” which specifies how fast the water flows along the X axis. In order to use water textures, the “texture name” field for a flow brush group should be one of the following: “bl” or “gr” (“bl” if you want blue water, “gr” if you want green water). The easiest way to get basic water flow is the shotgun method – lots of large flow brushes. Make sure you press “update in world” and/or reportalize.
- Don’t cross the beams. If you get hurt, remember that we warned you. We wish you luck, but remember: DromEd is unsupported. You asked for it.

==============================

Trimfect:
On ControlDevices.

If you have advanced to the point of using switches and triggers, read on.

(Note: I assume you have loaded ConVict and done other necessary preparation, so they are not covered here.)

Basicly the ControlDevice (CD) triggers an event in its target. If you were to add CD-link from a switch to a door, it would cause the door to open when the switch is pulled.

I. First things first: What is a link?

Link is basicly a link between two objects. There are many different kind of links - some can be used to give items to guards, some make guards to walk around and some trigger events. The DromEd tutorial gives you the basic idea of what a "link" is.

Adding a link:

Create a room with a switch and a door. Now check the door's ID (a number in brackets) and write it down (Later when the map gets bigger, you will need to write down these number quite often, and its very handy to keep a pen and a paper near by). Next select the switch and click "links". Click "add" and choose ControlDevice from the list. Type the ID number of the switch on the upper field and the ID of the door to lower field. Basicly that's it. Now the door opens and closes from the switch.

II. Next thing second: What is a script?

Thief uses scripts to accomplish various tasks. Door uses a script called "StdDoor", button use a script "StdButton". Lights that can be turned on or off from a switch uses "AnimLight". Conversation "TrapConverse". Books can be read with "StdBook". Are you even a bit confused? Good, you should be. There are numerous scripts that do different tasks and there is no list of scripts available, so search the real missions or wait till someone does a good tutorial/list. Basicly the script is the heart of the object. Add "StdDoor" to a book and you got yourself a book that works like a door. (Then there is the FrobInfo and other small systems that may prevent a script from working, but I will not cover that here, maybe later.)

Script: TrigRoomPlayer:

When player goes near the Bafford's front gate, the guards start to talk about bears. It is done by a simple script called "TrigRoomPlayer". Once again, make a room and create 2 different room-brushes in it, separating the room from the middle. Do not leave any area not covered by the room brushes. Build the room database. Select other of the room brushes and click "create" and from there "Add". (be at the root of the room-tree when you create the new room) Type in a name, ie. "TriggerRoom". And yes, it is a concrete room when DromEd asks you. Check for sure the room is now the TriggerRoom. If it is not, click "create", select TriggerRoom and click "create". (Sometimes you can't see the change immediately, so select something else and then the roombrush again.)

Now click "Edit" -> "Add" -> S -> Scripts. Type "TrigRoomPlayer" to field "Script 0". Hit OK and Done. Now create a porcullis from Physical -> Terrainlike -> Doors -> slidy_door -> porcullis -> select one of them. Create the porcullis and check it's ID. Select the TriggerRoom room brush and click links. Add a ControlDevice from the roombrush to the porcullis. Build room database and all the other things. When you enter the TriggerRoom, the porcullis should open and when you leave the area, it closes.

More advanced things:

A door needs the TrigDoorOpen script in order to function as a ControlDevice

Inverter that can be found from "fnord -> TrapTrig -> Inverter", can be used to invert the effect of ControlDevice. Using the the same room, where the porcullis is, create an inverter. Now link the inverter to the porcullis and the TriggerRoom to the inverter with a ControlDevices.

Remove the CD-link from the TriggerRoom to the Porcullis. Now the gate opens when you are NOT in TriggerRoom.

More practical usage if the Inverter can be found from doors. If you want a door that closes when another door opens from a switch, you have to use an inverter to accomplish this.

I think I've confused everyone's head enough and shut up for now. Just remember, that it is all logical. Once you understand the basics, it really is quite simple. And yes, I know I'm not much of a tutorial writer.

==============================

Texture families:
The full list of texture families is: dungeon, city, church, cave, catacomb, concastle, bafford, sewer, rescore, temple, mine, ruined, metals, core, ancient, maw, ramirez, lostcty, newcity, keeper, mech, newkeep, basement, tower2, waterhw, skyhw, sky, porttest, water
Don't use texture in RED; they confuse DromEd.

==============================

Lights:
DISCLAIMER: I make NO guarantee that this will work. Any loss or problems as a result of following these steps is your own fault. This tutorial is for people who have actually read the LG main tutorial as well as tried to actually build something in Dromed.

In this example, it is really not necessary to use a RelayTrap. Just make the link connected directly from the Switch to the OldElecWallLight.

1) Create physical:Lights:lanterns:ElecWallLight:OldElecWallLight(-3352) and place it in a location which you would like to be lit up by a player when they throw the switch.

2) Create Physical:Gizmo:Switches:Levers:Up/Down Switch(-464) and place it somewhere in your world.

3) Create fnord:TrapTrig:RelayTrap(-1704) and place it in your world, preferably close to the switch and out of the way.

4) Create the following link in the Up/Down Switch(#1):

FLAVOR:ControlDevice FROM:An Up/Down Switch(#1) TO:A RelayTrap(#2)

5) Create The following Link in the RelayTrap(#2):

FLAVOR:ControlDevice FROM:A RelayTrap(#2) TO:An OldElecWallLight(#3)

Notes: Since the default position of the switch is up, but the light is default off, you may want to rotate the switch 180 degrees so that the lever is at the bottom.

==============================

Elevators:
I noticed someone asking for help on how to make elevators; so since I was able to make it work for me, I'll post the step by step process.

DISCLAIMER: I make NO guarantee that this will work. Any loss or problems as a result of following these steps is your own fault. This tutorial is for people who actually read the LG main tutorial as well as tried to actually build something in Dromed.

1) Create physical:TerrainLike:Lift:ElevPlatform (-487)

and place it in it's starting point location.

2) Create fnord:Marker:TerrPt (-1322) [#1]

and place it z-1.5 ft below the elevator platform (e.g. if the elevator platform is at (x,y,z) position (10, 10, 10), then the TerrPt should be placed at (10, 10, 8.5). Another good rule of thumb is that the top of the bounding box of the TerrPt is just below the surface of the floor.

3) Create two more TerrPts at two more levels where you would like the elevator to stop.

4) On each level, the number of buttons which need to be created is equal to the number of TerrPts. So, in our example, since we created three TerrPts (i.e. floors) we will need to create a total of nine buttons. (For four floors, we would need to create 16 buttons.)

5) Close to EACH of the TerrPts (i.e. so that they may be pressed while standing on the elevator) create physical:Gizmo:Switches:Buttons:ElevatorButton(-462)

6) Let's assume TerrPt(#1) is on the ground floor, while TerrPt(#2) is on the second, and TerrPt(#3) is in the basement. (As you know, each object has a unique id in Dromed; these I have labeled 1, 2, and 3 for simplicity.) The physical layout of your objects would look something like this:

TerrPt(#2) ElevatorButton(#12)

ElevatorButton(#11)

ElevatorButton(#10)

TerrPt(#1) ElevatorButton(#9)

ElevatorButton(#8)

ElevatorButton(#7)

TerrPt(#3) ElevatorButton(#6)

ElevatorButton(#5)

ElevatorButton(#4)

Following this, only the links need to be created.

7) When you choose buttons 12, 9, or 6, make the following links:

FLAVOR:ControlDevice FROM:ElevatorButton(#12) TO:TerrPt(#2)

FLAVOR:ControlDevice FROM:ElevatorButton(#9) TO:TerrPt(#2)

FLAVOR:ControlDevice FROM:ElevatorButton(#6) TO:TerrPt(#2)

8) Repeat this procedure for the rest of the buttons where 11, 8, and 5 are linked to TerrPt(#1) and 10, 7, and 4 are linked to TerrPt(#3)

9) For the TerrPts, make the following links:

FLAVOR:TPath FROM:TerrPt(#2) TO: TerrPt(#1)

FLAVOR:TPath FROM:TerrPt(#1) TO: TerrPt(#3)

FLAVOR:TPath FROM:TerrPt(#3) TO: TerrPt(#2)

(This must be a closed loop of links)

10) In the Data field (i.e. the fourth column) of the links which you just created in the previous step, you must change the values of TPath from {0.00,0,FALSE} to {5.00,0,TRUE} so that the elevator can move.

11) For the elevator to initially start at TerrPt(#1) add the following link to the ElevPlatform(-487):

FLAVOR:TPathInit FROM:ElevPlatform(#?) TO:TerrPt(#2)

(Don't ask me why it has to be the TerrPt before the one you want in the loop)

Notes: In order to change the orientation of the elevator, you must change the orientation of the TerrPts.

Be sure to add Shape->Joint Positions: {0;0;0;0;0;0} to all elevator buttons to make them visible initially.

=========================

Windows:

This is a simple topic that I guess a lot of people don't know much about, but it's extremely useful.

If you've started building a level of any size at all, you'll notice that the 2d windows get real messy after a short while. This can't really be fixed by filtering, as filtering out all the "terrain" is pretty counterproductive when you're still trying to build. So, ta-da! **Area brushes** to the rescue!

When you've finished a certain area of your level, (say, a few rooms, or a basement or something) click on "Area" and drag out a brush that completely surrounds the bit that you're finished with. You don't want this brush to be any bigger than it has to be.

After the brush is of proper size, you'll see some new options on the left side. The first thing to do is give the area a name, by clicking on "name" and typing something clever. After that, you can play around with the buttons above. "Activate" sets the area brush (and everything inside it) as "active". "Show Hot" filters out everything but the active areas. "Me Only" filters out everything but the selected brush.

Be careful with one thing though. Stuff that's filtered out no longer shows up in the 3d window and doesn't get portalized (so it doesn't exist in game mode). If you want to go through all of your level, click "show all" to show all areas and portalize. In the meantime, when you only work with certain areas, the "persistent_player_pos" command is your friend.

============================

Signposts:

Okay, I think the objects you are looking for are in the Hierarchy path physical

decorative:signs - all sorts of plaques, signs and whatnot. The kind of object you're looking for was named PosterScroll or something like that - an unpickable, readable scroll.

To actually make them produce text, you edit the object properties and add a Book:Text attribute on it. On this you give the name of the text file that the contents are in. The text file should have a .str extension, such as 'message3.str', and reside in the path /books/english. In the Text attribute settings you give the name of the file without the .str extension.

Now, if you place the 'text' attribute on a sign, plaque or something like that, the message will appear on the playing screen. If, however, you place it in a book or carryable scroll, it will open a new window with an appropriate 'book page' background to display the text in. Only the latter type of object is capable of showing more than 10-12 lines of text, spread over several pages (see the format of the text files from those in GatB or from the original Thief books.crf).

==

- I want to put a fake door on a wall. I've tried creating solid brush of various sizes and then applying the texture. Either the texture never appears (I re-portalize just about every keystroke), or I get the texture split right half swapped with left half.
- I've loaded the unstripped missions to look at how it was done. I don't see any difference in the brush or texture settings. What am I doing wrong?
- I have similar problems placing fake windows.
- Select the brush, apply the texture, then manipulate the "U", "V", "Scale", and "Rot" values at the bottom of the screen.
- "U" moves the horizontal texture origin, "V" the vertical, and the other two are self-explanatory.
- I wish it was as simple as Datoyamnazischmutz says, but it's a little trickier than that. First, apply the brush (I prefer to use an air brush so that you get that door step) and apply the map in the 3d viewport. Then, <u>make sure your textures aren't visible on the screen</u>, and select the object. If it's texture doesn't appear on the bottom center of the screen, hit reset near the bottom.
- Then, hit the box that says "default" until it highlights the face you want to alter. Then that "U, V, Scale and Rot" stuff works. One thing about Rot (rotation), I think you can only use intervals of 90. If you don't see the texture alter in the 3d vieweport as you change the values, portalize and they should.
- It's just that simple. You'll get the hang of it.

==

Candle Flames:

In the Object Hierarcy go to SFX -> FireFX -> Flame...

It will insert it with many effects like orange, yellow, smoke, and sparks. Delete all of those except for the flame itself. Resize it to your liking by manipulating the H,W,D numbers. Also, in order to have it not sound like a roaring fire, you can delete the AmbientHacked schema from the Properties.

==

ThiefGold: Filename ID Description

GAR001 This door is bolted, and there is no way to open it

GAR002 Its locked, I need a key to open it

GAR003 Its locked, I can try picking the lock or finding the right key

GAR004 Aahh, got it!

GAR005 Container like this might have some good inside it

GAR006 That's too high

GAR007 Crystal are good for powering up arrows

GAR008 This will be useful

GAR009 Rope arrows will on stick in wooden surfaces

GAR010 Always good to grab loot

GAR011 Now this is worth the price of admission

GAR012 Now the bow will shoot fire arrows

GAR013 Now the bow will shoot water arrows

GAR014 Now the bow will shoot moss arrows

GAR015 Now the bow will shoot gas arrows

GAR016 I'd rather not walk out there into the light

GAR017 If I stick close to the walls, I have a chance of not being seen

GAR018 I better wait till his back is turned

GAR019 Too many for me to take all at once

GAR020 I'd be better off looking for another way around

GAR021 I shouldn't leave a body laying around

GAR022 Perfect. Only one, and he is looking the other way

GAR023 Just keep looking the other way...

GAR024 I think I lost him

GAR025 I hate blind corners - you never know what you are going to run into

GAR026 It always pays to listen to a door before opening it

GAR027 If I can hit him blackjack without being seen, he will drop without a sound

GAR028 That was bloody. I should move the body, but if someone walks by they might figure out something's up

GAR029 Ooohh... This is not good!

GAR030 I can pick that guy off with an arrow, but if I miss, I will give myself away

GAR031 Lots of pyrotechnics, but not very stealthy. I've got to be very careful with those

GAR032 Water arrows are most useful for putting out torches

GAR033 Moss arrows are meant to fire on noisy surfaces so that I can walk on them without making a racket

GAR034 Not much point in firing those into the floor

GAR035 I should remember slashing around can make a lot of noise

GAR036 I can't stay underwater too long or I'll drown!

GAR037 I've only got few seconds of air left!

GAR038 I'll never make any headway against this current!

GAR039 Barrels like these, can provide good cover in a pinch
GAR040 Crates like these are useful for concealment
GAR041 Every key opens something, somewhere
GAR042 Aaah, not so secret anymore, is it?
GAR043 Aargh, I'm making too much noise walking on this tile
GAR044 Walking over metal is noisy, I should be careful
GAR045 I have a bad feeling about this
GAR046 Hmm. This looks interesting
GAR047 I wonder what happened here
GAR048 Looks like this hasn't worked in years
GAR049 I wonder how I can get up there
GAR050 Hmm... There must be a way down there
GAR100 No luck here
GAR101 I'm not opening this
GAR102 Can't get through here
GAR103 Can't go this way
GAR104 Not this way
GAR105 Ahh, no point in going here
GAR106 I needn't bother here
GAR107 Nothing useful this way
GAR108 Nothing back here I care about
GAR110 Doesn't seem to do anything
GAR111 Doesn't do much
GAR120 Hmm.. this can't be opened now
GAR121 Looks like I can't open it
GARC0101 Alright old man, lets get you out here and me to my money
GARC0103 Good thing you are dying, Cutty, or I'd have to kill you for stiffing me, again.
GARHMMM Hmm...
GARM0101 The keepers were training me to be one of them, but I found other uses for those skills
GARM0201 Hmm... There are too many to try to get by here
GARM0202 Only one guard, still it would be nice to drop on him
GARM0203 Inside, at last
GARM0204 Hah! Could you possibly be any more helpful?!
GARM0205 Better not leave too many bodies laying around
GARM0206 I wonder if he reads them or if they're just for show
GARM0207 Since I'm in here, I might as well pick up something for myself
GARM0208 Hah! Its a throne room, how pretentious can you get?
GARM0209 Here we go...
GA1V0301 I should be near the prison entrance by now
GA1V0302 He can't see me while I stick to the shadows, but I'll have to cross the light in order to get to the stairs.
GA1V0303 Argh, clanky metal plating, I better tread softly
GA1V0304 Locked from the other side

GA1V0305 Ha, better not let anyone bang that gong, or I'll have more company than I can handle
GA1V0306 Ha! Just what the world needs - more hammers
GA1V0307 More than one guy, that will make things trickier
GA1V0308 Looks like I'm on the right track
GA1V0309 I hate blind corners, you never know what you are going to run into
GA1V0310 I have a bad feeling about this
GA1V0311 Better stick to the shadows here, if they see me, I'm in trouble
GA1V0312 Right, cell block 4, Cutty better appreciate this
GA1V0313 This seems too easy
GA1V0314 How many guards do they have in here anyway?
GA1V0315 But which one is Cutty, and how I'm going to get the door open?
GA1V0316 Just keep looking the other way
GA1V0317 I bet the door controls are in there
GA1V0318 Well this was a lost cause, time to get out before I end up a cell next door
GA1V0319 Hah, that must be how they spend all those tithes
GA1V0320 Wonder if I have time to stop for a snack?
GA1V0321 No point going that way, it will be crawling with hammers
GA1V0322 They take all this stuff way too seriously
GA1V0323 If I keep going up, I'm going to run out of building
GA1V0324 No good going this way, nowhere to hide
GA1V0325 Cutty won't be needing this anymore
GA1V0326 Aah, this is what I'm looking for
GA1V0327 At least I can swim better than anyone carrying a sledge hammer
GA1V0328 That's a long hammer
GA1V0329 Interesting ride, I wonder if any of the hammers take it
GA1V0330 Well there's Basso. If he weren't unconscious it would be simpler, but now I have to carry him out
GARM0401 Not my most elegant entrance, but it worked
GARM0402 Felix, you don't look so good
GARM0403 Damn, there has to be another way in
GARM0404 What's that sound?
GARM0405 Whoa! You know you can't take it with you, but I can
GARM0406 The horn of Quintus, I presume?
GARM0407 I take this is the fabled horn of Quintus. Hope no one minds if I just take it with me.
GARM0408 So much for Felix, then
GARM0409 It’s a long way down
GARM0410 Huh, how rude
GARM0411 hah, tombs with piped-in music. How classy
GARM0412 Nice view, I wonder if the dead appreciate it
GARM0413 Good thing I'm not afraid of heights
GARM0414 Sounds like I'm getting close
GARM0415 Hmm, these guys don’t look too mean
GARM0501 So these are Ramirez's boys, its about time I deal with Ramirez once and for all

GARM0502 It may be time for me to move along now, I wouldn't want to wear out my welcome
GARM0503 Maybe I should head back to my own neighbourhood, these fellows don't seem friendly
GARM0504 What sort of a lunatic keeps these things as pets
GARM0505 Damn, I guess I lost their trail
GARM0506 That shot was meant for me
GARM0507 These creeps are going to lead right back to whoever sent them to kill me, but only if I can tail them without them noticing me
GARM0601 A glass of jam, this sword was just waste of time!
GARM0602 Damn, its a fake!
GARM0603 Information on Constantine, almost as good as gold
GARM0604 Time to take my new sword and get out of this crazy place
GARM0605 Magical traps, I guess he is serious about his privacy
GARM0701 Looks like this place was built by old pals the keepers. I wonder what they are hiding in here
GARM0901 What have we here
GARM0902 I'm not swimming across there
GARM0903 Creepy
GARM0904 Anybody home?
GARM0905 I wonder if the keepers have seen this
GARM0906 One down, one to go
GARM0907 How could the keepers keep this place a secret?
GARM0908 Looks like this must have been some sort of library
GARM0909 According to the map, this was some sort of market place
GARM0910 Time to raid some tombs
GARM0911 I'm not too crazy about the neighborhood
GARM0912 For me to make use of my map, I'm going to have to find some sort of landmark to go by
GARM0913 So this is the lost city... haha, its not lost no more
GARM1001 Damn, I took too much time
GARM1002 I'm a wall builder... This is no place to be reading poetry
GARM1003 I'm a wall builder, let my walls endure from season to season, from year to year... blah blah blah...
GARM1101 I don’t like the looks of this
GARM1102 Ouh...
GARM1103 Hmm...
GARM1104 What do we have here
GARM1105 These stairs have seen better days
GARM1106 Looks like this doesn't work anymore
GARM1107 If I don’t get this thing to work, I'm not going to get very far
GARM1108 What was that?!
GARM1109 Now how am I going to get out of here
GARM1110 Oh master builder, we ask thee to bless our brother who has died in our service. Forgive him... blah blah blah...
GARM1202 Whoa! This is interesting, but it doesn't say anything about the eye

GARM1203 Whoa! Not even the hammers can make heads or tales of this, but it doesn't look good
GARM1301 What happened here, and where are all the Hammerites?
GARM1302 Damn, looks like Constantine got here before me
GARM1303 This looks like it would make a good escape route for me and the high priest
GARM1401 Well this is my way out, I better not mess with it
GARM1402 That can't go anywhere uh pleasant
GARM1403 All you other demon thing you can just stay home now
GARM1404 Ouch, better not to try that again
GARM1405 You gotta be kidding me
GARM1406 Hah... Now that's big!
GARM1407 Ahh... Interesting
GARM1408 I've had enough of this!
GARM1409 Let's just disable that portal and give hammers a fighting chance
gar06100 Nighty night
gar06101 That pays the rent
gar06102 Ah, a back door
gar06103 Phew, sewage is definitely not the glamorous part of the job
gar06104 Lord Randall’s would have smelled better
gar06105 They’re so busy gambling their money way, they won’t notice if I take a bit
gar06106 Now I can get my vase
gar06107 That’s what I needed to hear
gar06108 I love it when money falls into my lap
gar10100 Hmm, this looks like the way out. That’ll come in useful later.
gar10101 So this is where the mages hide their dirty little secrets.
gar10102 Looks like Captain Regalio’s been holding out on the mages. Good thing he can’t hold out on me.
gar10103 This’ll make any fence crack a smile.
gar10104 Now for the talisman
gar12100 Guess I’m not getting any information from Geary after all
gar12101 Empty. Now what?
gar12102 The curtains went down on this guy a long time ago, but he’s the best lead I have.
gar12103 This looks like the way in.
gar12104 Huh, do people really pay money to see this stuff?
gar12105 Ah, a hidden entrance.
gar12106 Well, I’d like to stay for the finale, but I’ve got a feeling it’s time to move on.
gar12107 Well, that was a surprise.
gar99100 I’ll just sneak out through the windows.
gar99102 Better get to work
gar99103 Taffer?
gar99104 Better get to the shadows before they see me
gar99105 I can’t hide here forever
gar99106 This guy needs to lose some weight (grunt)
gar99107 That shut him up
gar99108 These guys just have no sense of humor

garOT01 I’m not even gonna comment
garOT02 I’d call that a bug
garOT03 Hey, get back here!
garOT04 Oh, yeah. He’s gone.
garOT05 Somebody light a match.
garOT06 What the hell is a taffer, anyway?
garOT07 I hate modern trends
garOT08 I think Bafford’s got too may bats in his belfry
==

A List of Garrett's Comments in Thief 2 - by SilentButDeadly

gar0100: Ok Basso you sit tight 'till I give the signal.

gar0101: Better check my map.

gar0102: This must be the door Basso mentioned.

gar0103: Good thing the butler is out for the night, he left his lights on.

gar0104: Upstairs... Basso isn't going to need to come up here.

gar0105: Ohhhhh... Better stop and check out that light around the corner.

gar0106: This hallway is too well lit, I better put a water arrow into that torch.

gar0107: Carefull...

gar0108: Another gaurd up ahead, time to knock him out while his back is turned.

gar0109: Lets duck aside and wait for this.

gar0110: Lets uh... duck aside and wait for this little "chap" to finish.

gar0111: Steady... Steady!

gar0113: Looks like the coast is clear for Basso now.

gar0201: Shouldn't be too hard to find a spot to climb out of this place.

gar0202: Hmm... doesn't look like I can get up there.

gar0203: A fall from this height could hurt!

gar0204: I bet a gaurd or two around here may have a key for me to pickpocket.

gar0205: This place has lots of shadows to hide in.

gar0206: That floor looks loud... I better find a way to walk across it quietly.

gar0207: I need to find a key for this.

gar0208: Look whats on that guy's belt!

gar0209: I hear this guy is quite a popular musician, his unpublished work should be valuable.

gar0210: This place looks like an inventer's shop, theres probably a valuable gadjet or two around here.

gar0211: Mynell makes the best steaks in the entire city, his reciepies my catch me a pretty penny.

gar0212: Germ is the man that makes those magic lenses, theres probably some valuable lenses in here.

gar0213: That looks like Captain Davidson's cargo ship, he's a shady guy... smuggling in piracey.

gar0214: A large open door... that is bound to attract a gaurd or two, we should probably find a way to get it shut again.

gar0215: Hehe... Yo ho ho and a bottle of rum!

gar0216: Eewww... so that Mynell's secret ingredient!

gar0217: Hmm... the numbers above these doors may be important.

gar0218: Only thing left for me to do is to make my exit.

gar0219: If I aim right, I can hit that button with an arrow and activate it.

gar0401: I should take a look at those instructions.

gar0402: Ohhhhh ya! He's gone.

gar0403: He won't need that anymore.

gar0404: Real classy, I guess it pays to be an officer.

gar0405: Nice view!

gar0406: Now! On to the vault.

gar0407: Looks like this Mosley character has got a thing for plants... I guess she can afford to.

gar0408: Lets see what I can use against our Lt. Hagen.

gar0409: Guess I could use some target pratice.

gar0410: I should be able to find something in here.

gar0411: Mechanists, always the do gooders.

gar0412: I better do this quick.

gar0413: Letting them out would spoil the plan.

gar0414: I don't think I want to go that way.

gar0415: Hagen doesn't seem to.. keen on Lt. Mosly does he?

gar0416: I should be careful in here, there are likely a few traps around.

gar0417: Hey, I know who this guy is.

gar0418: No thanks, I'm not ever that hungry!

gar0419: I should write that down............ now, onto the vault.

gar0420: Well, this is a snug fit.

gar0421: Heh, yeah right, life is tough when you don't have anyone waiting on you hand and foot.

gar0501: Better get moving before they see me.

gar0502: A sewer never looked so appealing.

gar0503: Judging from the gaurd post, I bet thats Smith's place.

gar0504: Hmmm, wipple and hellan away, thats where that prisoner in Shoalsgate said he stashed some loot.

gar0505: Home sweet home.

gar0506: Looks like im going to need a new address, but im going to have to get to my stash first.

gar0507: Good, now time to relocate.

gar0601: These Mechanists are just as bad as the Hammers.

gar0602: It's no big secret that the Mechanists are the ones suppling shariff Truart with those machines.

gar0603: I could really learn to hate these guys.

gar0605: Strange, I wonder what kind of work the Mechanists do here.

gar0606: Better not disturb them, I can hear the meeting just fine from outhere.

gar0607: I'd like to get a copy of that key myself.

gar0608: I wonder how I knew I was gonna need this wax key press.

gar0609: If I can find some soft wax I can get an impression of the key and make a duplicate later.

gar0610: Heh, wrong key.

gar0611: This is the right key.

gar0612: That wasn't the right key.

gar0613: Good thing I can still use this wax.

gar0614: Nope!

gar0615: Hope this is a safe place to drop this key and make a copy of it.

gar0616: I better come back here and replace this key when im done with it.

gar0617: That ought to do it.

gar0618: I think my work here is done.

gar0701: I wonder if any of the outside windows can be opened.

gar0702: Looks like I might be able to climb onto the roof.

gar0703: Some people in this city are too rich for their own good, luckly they have me to give them a hand.

gar0704: Now to find some usefull information.

gar0705: I'll never be able to cross through here with all those secrity machines active.

gar0706: Mybee I can find a way to deactivate them.

gar0707: The vault isn't going to open until that locking bar is out of the way.

gar0708: What a weird contraption, lets see what these controls do.

gar0709: Hehehe, now I know im a master thief, waltsing right into a bank vault, time to find the recording.

gar0801: Lt. Mosely, out for a little strole, lets see where you take me.

gar0802: Somehow I doubt she dropped her note be accident, I better see what it says.

gar0803: Looks like the good Lt. is looking for Pagens.

gar0804: Lets see who comes to claim the note.

gar0805: A cemetary, heh, that's original!

gar0806: Wasn't expecting that.

gar0807: Well I'm not gonna wait around for him to come out, fortunatly, cemetaries don't spook me.

gar0808: Where did he go? Guess there must be another way out of here after all.

gar0809: Damn! I guess I lost the trail.

gar0901: Hmmm, looks like a gear is missing.

gar0902: There is probably something in this room that will tell me who killed Truart.

gar0903: Damn! Someone beat me to the Shariff, I better keep it low profile or I'll be pinned as the killer.

gar0904: I better get out of here now.

gar0905: Truat doesn't do much to hide his vanity.

gar0906: If I knew being Shariff paid that well, I would of changed careers a long time ago.

gar1101: Hmmm, that leter the Pagen is carrying, It's gonna be late.

gar1102: These people didn't have a chance against the Mechanists.

gar1103: I hope this guy doesn't run out of blood.

gar1104: Looks like I'm still on the right track.

gar1105: So... it's been the Mechanists behind my problems this whole time.

gar1106: No need for alarm ladies, just passing through.

gar1107: The front doors are locked, looks like Karras wants a captive audience.

gar1108: So, thats the castle of the future, heh, I'll take my tenement any day of the week.

gar1109: So, Karras doesn't even show up for his own party.

gar1110: If I follow this road north, it should take me right to the Mechanist Tower.

gar1111: One thing is for sure... this.. KARRAS guy has lost his mind!

gar1201: Better find a way in before I catch cold.

gar1202: Ahh, storage, I'm sure Gervaiceus has lots of things down here that he'll never miss, aleast not until It's too late.

gar1203: Hmmm, I may have to stop in here for a drink later.

gar1204: Mybee I'll stop in for a little light reading.

gar1205: Either someone is hiding the evidence or these guys got lost in ten feet of passage.

gar1206: No wonder the spooks are restless around here.

gar1207: Ash, you are a bastard.

gar1301: Ahh, early morning mist, all respectable people should be in bed now, time for us disrebtuable types to get to work.

gar1302: So this is what all the fuse was about, that old bragger has made quite a spectacle for his bobbles.

gar1303: I'll have to grab a bottle, and later drink a toast to Lord Gervaiceus for his generousity.

gar1304: Alright, thats the last one.

gar1305: Hmmm, another Mechanist, guess the Hammers just arn't "sheak" any more.

gar1306: Looks like I'll be able to get one of those "Cultivators" when I come back.

gar1307: Looks like I'll be able to get one of those "Cultivators" while I'm here.

gar1308: A "Cultivator", that must be what Karras is writing to Gervaiceus about, what a nice bonus!

gar1309: A "Cultivator", another nice bonus.

gar1401: Hmmm, that must be that Cetus Amicus, guess I'll sneak aboard and have a look around.

gar1401A: Lotus?

gar1401C: I'm no Mechanist, I'm Garrett.

gar1401E: Knew I would come? but..

gar1401G: Don't worry, I'll find Cavador, Karras will be stopped!

gar1401I: Yes, my friend, what can I do?

gar1402: Tight fit, I hope It's a short ride.

gar1403: If I were Cavador... where would I be?

gar1404: Berrrr, I wonder what they keep on ice.

gar1405: Hmmm, doesn't look very dangerous, good thing I know better.

gar1406: It's a long way up.

gar1407: The wheel seems to be missing a peg.

gar1408: What the!?

gar1409: This should fetch a nice price.

gar1501: I'm back in the Lost City, ahhh, should of known, guess I'll take the back door home.

gar1502: Ahhh, knowing where Cavador is, thats half the battle.

gar1503: What does he eat!? Metal rivets?

gar1504: Looks like my old root is still open.

gar1505: This guy makes the Hammers sound sane.

gar1506: Archeologist seems so much more dignified than "thief".

gar1507: Guess things have changed a bit.

gar1508: Too bright for an ambush here, I prefer one of those dark corridors.

gar1601: Ya, just keep prataling away Karras.

gar1602: We'll see how well your protective chamber works against your own rust gas.

gar1603: This is just what I was looking for, now I've got a real plan.

gar1604: Hmmm, there must be instructions on this thing somewhere.

gar1605: Asuming the blue prints didn't leave anything out, this thing ought to do the trick.

gar1606: Now I just have to switch the signal towers to use the beacon instead of Karras's instructions.

gar1607: What if Karras has anyway to detect what I'm doing, I'll know soon.

gar1608: Karras didn't pipe up about the sabatage, I don't think he knows.

gar9901: I'm not going to find what I'm looking for this way.

gar9902: I need to find a better way than this.

gar9903: I bet this is the right way.

gar9904: eh, I think I'm on the right track.

gar9905: That should do it.

gar9906: Hehehe, I think I deserve a pat on the back or that one.

gar9907: That was helpful.

gar9908: Oops.

gar9909: Damn! I didn't mean for that to happen.

gar9910: Unhn, that hurt.

gar9911: Time to go.

gar9912: Now to be on my way.

gar9913: Time to get out of here.

gar9914: This doesn't look safe.

gar9915: This looks like a dangerous area.

gar9916: Better watch my step around here.

gar9917: This looks like a hard place to sneak through.

gar9918: It would be tricky to get through here undetected.

gar9919: I can't stay underwater too long, or I'll drown.

gar9920: Huh, not so secret anymore, is it?

gar9921: I wonder how I can get up there?

gar9922: Ahhh, a back door.

gar9923: Can't go this way.

gar9924: Ahhh, no point in going here.

gar9925: I needen bother here.

gar9926: Nothing back here I care about.

gar9927: This looks like the way in.

gar9928: Ahhh, a hidden entrance.

gar9930: Ahhh, my favorite year.
==

Creating Automaps - A Tutorial

by Caeyr

First a few words about the way Thief handles maps:

The basic map is the one you see when you are nowhere special - nothing is marked in blue. Room maps are parts of the basic map, with a part of them blue. Thief copies one of these on the basic map when the room you are in has an Automap property, but more on that later. If that all sounds rather strange, take a look at the maps that come with Thief (which is never a bad idea). There are three kinds of files necessary for a functioning automap:

-Pageddd.pcx

The basic map. Obviously you can have more than one, but numbering should always start at 001 with ddd being a three digit number.

-PdddRnnn.pcx

The room maps for the basic map named Pageddd.pcx. The numbering is similar: ddd should be the same as that of the basic map to which it belongs, nnn is the number of the room map and should begin with 000. For each of these, numbering should be continually, without gaps.

-Pdddra.bin

This file contains positioning information for the room maps. When Thief copies the room map unto the basic map, it needs to know where to put it. That information is found here.

Creating the map:

I can't really help with drawing the map, because you've got to know yourself what you want it to look like.

What you'll need to do is use some graphics editing program and load the blank map named Page001.pcx that came with DromEd. Sketch your level unto it. Save it as Page001.pcx (I don't need to remind you that you'll want to keep the blank map for further projects, do I?). Now select a part of the map you want to display as automap and copy it to a new file/image. Edit this smaller image to give your room the blueish color. (I haven't found a perfect way to do this yet, as I'm not really talented in editing graphics. I'm working with Picture Publisher, so what I did was select the part of the image I wanted to turn blue and used 'Hue Shift'. The problem is, it only works in True Color and I had to convert the image back to the Thief palette afterwards - Thief wants the maps to be 256 color and have its usual palette, else it won't work properly.)

When you're done with painting, save the room map as P001R000.pcx. You can create room maps for P001R001and onwards the same way.

Positioning room maps:

Probably the most complicated part: For each of your created room maps you need to get the X and Y values of the upper left corner where it should appear on the basic map, and the room map's width and height. Those values are needed for the positioning file. Now create a file named P001ra.bin with a size of 8*(number of your rooms) bytes. You can do that easily with a text editor (like Notepad) by writing the above amount of characters inside. For the next part you should use a Hex-Editor. Open the file and enter the values for X,Y,width,height. Each of these values go in two bytes, so that there's 8 bytes for any room map. Remember that low byte comes first (That means, if you have an X value of, say 270, which is 010E in Hex, you'll write 0E 01 into the first two bytes). The first 8 bytes are for the room map P001R000, the next for P001R001 and so on. Enter the values for each room.

For your mission to use these files, they need to be put into the \intrface\missxx\english directory, where xx is the number of your mission.

Things to do in DromEd:

For DromEd to use the automaps, select a room brush. Edit its properties and Add: Room->Automap. Enter the number of the basic map into the first line and the number of the room map in the second line. (If you want to show the basic map Page001.pcx with the room map P001R000.pcx on it, first is 1, second is 0).

Another thing do do is set the map_max_page and map_min_page. The numbers you need to enter here correspond with those of your Pageddd.pcx files. (If you have 4 basic maps numbered Page001-Page004, map_min_page should be set to 1 and map_max_page to 4. Those variables are entered like the other quest variables with quest_create_mis.)

DromEd should now display the proper automap when you are inside the room.

One thing to note: When I created room brushes they were all named 'default room', and on adding the Automap property, the others seemed to inherit it. What I did to get rooms with different automaps was to go into the Object Hierarchy and select Rooms (as opposed to Archetypes etc.) from the dropdown list. In there already are usually 'Default Room' with 'Base Room'. For a another room type, press Add and enter a name for the special room. Then press Create. You can now create this room brush like any other (even Shift-Insert works). Remember that an automap added to this room will also show up in other rooms of the same type (I wonder if there's a way around that).

On another note: It seems that, in the game, if you enter a room without an automap property after leaving one with the property, it will show the basic map without any room map (as it should be). If that is true in all cases, it would be advisable to always give the room around the StartingPoint an automap.

Editing the Pdddra.bin file by hand is rather tedious, so I've begun to write a program to make positioning easier. I'm actually rather happy, because yesterday I got my 'Map Positioning Tool' to load .pcx files properly. Formerly I had to convert them to .bmp files because those were already supported. I hadn't expected that I would have to face the Windows API personally when I began with my little project.

Nonetheless it worked rather well and I even ventured to try 'enhancing' GatB with automaps because it already had a good map. Unfortunately I gave up because I had problems finding the room brushes inside all those lines.

Anyway, I might release this tool if there is demand for it (although it still needs some tweaking). Anybody interested?

==

You can add brushes to a multi-brush by:

Clicking on Add Brush to Group in the Multibrush menu. Or use add_brush_num nn, where nn is the number of that brush.

Make an area brush around whatever you wish to turn into a multibrush, then click on multi-brush-me.

==

- clicking the grid "use" button in the lower left corner of DromEd is a "snap to grid" button.
- you can place a light source that is not associated with an object by clicking "light" and creating one in the 2d window (I think this is how they made the hall lights in Cragscleft after you go up the stairs from the mines)
- don't forget to portalize
- when you create the purple box "room", it comes out flat. You have to resize the 3rd dimension so it envelopes your level areas.
- if you equip Garrett with the wrong sword, he can't use any weapons

Scolling 2D windows is tricky. There are 2 methods that I use.

1) Either rightclick your viewpoint to place where you want to center the screen and then hit ALT-SPACE. Use zoom and unzoom to move faster.
2) Press] (I think) and then CTRL-X to enable X-Mouse. Now you can scroll the window where your mouse is with the keypad. There was the adjustment of how much it scrolls, but I can't remember what it was.

==

When you create a brush (other than an object brush) such as a room brush, you rectangle it out with the mouse in two out of three dimensions, depending on which 2d view you're in; the third dimension defaults to the dimension of the last brush you had selected. If it was a light or something small, it'll be zero (due to grid) but if it was your big room-digging air brush, it should be large.

Another useful thing is the dromed command brush_to_room N, which creates a room brush larger by N than the brush you have selected (again, good to use on your room-digging air brush).

==

How to make a secret door

First off... You have your default room, right? Make a starting point (->fnord->marker, and under links make flavor PlayerFactory, the from the Marker's ID and the to "Garrett").Add another room, a small one perfectly 4 wide, 8 high and however deep you want (preferably 4 or more.) Add another room about 16 wide by 16 high by 50 deep.

Connect them all, so you start in the original DromED supplied room, then there's the 'doorway' and the other room. Light them as you wish, and make them rooms (shift+ins.)

In the objects hierarchy, add physical -> terrainlike -> doors -> slidy_door -> 4x8secret_door. Place it in the doorway, perfectly aligned with the wall of your starting room. Under it's properties, add ->door->spinny_door (or slidy, but i haven't tried that yet) and in the windows you get, set Closed Angle to 0 and Open Angle to 90. Portalize and everything, and go into game mode. Go up to the door and try to open it. (It should open ok) Go back into editor mode, and if the door had opened away from you, that's good, but if it opened towards you, rotate it 180 degrees. Hit Alt-T and look at the texture you're using for the wall the door is supposed to blend into. Note the path and filename (when you click on the texture you want, the filename is in the middle of the status bar at the bottom of the DromED window. Go under the door's properties, and add -> Shape -> TxtRepl r0. In the window that pops up, type in the path and filename, ex: "fam\bafford\b22" and click ok, done. Back in the 3D view, the door's texture should look like your wall. If not, reportalize and go ingame.

===

The pausing patrol...

In the long room, at one end, create a normal Guard. Add -> Ai -> Ability Settings -> Does Patrol. When the window pops up, click on the check box so that it is checked. Then add -> MetaProperty ->Ai Behaviors ->AI_B_m2 -> M-Front Gate Guard. Exit all the windows, and add a hammerite guard (or any other object.) If it's a human, add the same Front Gate Guard Property. Place the guard near the middle of the room. Add a ->fnord->marker->TrolPnt and place it near the normal guard, not the hammerite. Write down it's ID number. Let's say it's X. Add another TrolPnt near the other end of the room, so that to get from one to the other, the guard must practically walk through the hammerite. Check this one's ID num, let's say it's Y. Under TrolPnt X's links, add AiPatrol with from as X and to as Y. Do the same in Y's links, with Y as from and X as to. Update Pathfinding database, AI Room Database, etc... and go into game mode. You should start out in the main room. Open your door, go through, and you should see your guard patrol, and the hammerite should just be there, maybe whistle, etc... They will probably ignore you (because of the m-front gate guard property.) Go back into DromED, and click on the normal guard. Under links, add AIWatchObj.

Your resolution should be 1024x768 (at least) otherwise you won't see the top drop down option box. In that option box, select "Self Entry." In radius, type in 10, although increasing and decreasing this number might be needed. In height, put 10 (i haven't tried any other values yet.) Then in Reuse (a bit lower down the window) put 5000. Under that, in Reset, put 500 as well. This is how long after entering, the values will reset. So if it's 0, as soon as he finishes doing what we'll tell him to do later, he'll do it again, and again. Under Response 1, select Face. Type in the ID number of your Hammerite in Argument 1. Making him look at something works better, because just using wait makes him freeze exactly as he is. Then, under Response 2, select wait. Under Argument 1 type in 2000. This is the time in milli sec. that he'll wait, so 1 second. Under Argument 2, typ e “IdleGesture 0” (sans quotes) so he won’t freeze position, or “Search, Scan” to make him peer into the distance like he's searching. Hit enter if you don't see the OK button, or just click it if you do. Hit done, etc... until you get rid of all the windows, and go into game mode. You should see him walk up to the hammerite, look at him, pause for 1 second and continue. In my demo level, I have added another AiWatchObj in the guards properties, so that if he sees you, he'll get hostile and lose the m-front gate guard metaproperty.

==

How to make books

You first need to use notepad or something and write out the page. Here's an example, this was from Bafford's Manor:

```
page_0: "Cedric -
Please speak to Cook about last night's dinner. While, \
technically, the menu conformed to my instructions, I \
suspect that the lamb was somewhat older than this spring's, \
and I am in no way fooled by his practice of warming the \
salad to disguise wilting. If Cook is incapable of finding \
adequate ingredients, he can be replaced - and if he offers \
those same excuses about the Stonemarket shortages, please \
remind him that the grocery budget is a good fifty percent \
above last year's figures, and even he should be able to procure \
adequate victuals at those prices.
- Lord Bafford"
```

That's how you would type it out. If you were to have another page, you would then start it out "page_1:" and so on. You then save it as a .str file. Now, you would make a folder called "books". And in that, make another folder and name it a language. Okay, go back to DromEd and look at the properties in the book. Press add::book::text. Now type in the name of the text file you made without the extension.

==

Books, Screen text, and "Readable" Objects

This tutorial is intended to teach the basics of creating books and the required elements. It will also explain how to apply those concepts to arbitrary objects, enabling you to "read" a loaf of bread, for example. I assume that you have read all the DromEd documentation (in the \docs folder after installing DromEd), and have a working level in which to place your book. That means that you have taken all the preliminary steps, including loading ConVict, creating a starting point and linking it to Garrett, applying room brushes, and anything else that may be needed. I make no guarantees about anything I say here, and have actually only spent a few minutes experimenting, but I believe I understand the concepts. I have also tested all of my examples.

The Basics

There are four basic elements that are required to let Garrett read something:

1) The text file. All books are kept in a text file with the .str extension. The file must be placed in the Thief\Books\English directory. The \English directory is not necessary if

you are not planning on including multi-lingual support. In fact, it should probably only be used if you are. If you don't use the \English directory, book files go directly in the \Books directory.

2) The frob information. "Frob," in DromEd, means "use." The frob information is contained in the FrobInfo property and describes what happens when an object is used. A book must be "frobbable."

3) The StdBook script. The book must have a Script property with StdBook in one of the fields. I don't know if it has to be the first field, but I imagine that it will work in any of them. (Sometimes StdScroll is used instead. I'm not clear on the distinction, except for what I mention in the final example.)

4) The Book properties. There are two types of Book properties. Text is required (for obvious reasons) and that is where you type the name of the file that contains the text you wish to be displayed. The Art property tells which paper texture to use when displaying the text. If it is absent, the text is simply displayed at the top of the screen. (I have not tested this thoroughly, but it seems to be the case.)

Adding a Book to Your Level

First, open Notepad to create your text file. Notepad is a good choice because it only saves in plain text format. WordPerfect, for example, saves in a completely different format that DromEd does not understand. Likewise with MS Word.

In notepad, type the following:

```
Page_0: "This is the text that will be displayed. You must
always start your book on page zero and count up from
there, using this exact format for each page..."

Page_1: "the word Page, followed by an underscore, followed
by the page number. Then a space and a double quote before
beginning the text. It must also end with a double quote."
```

Save this file in the Thief\Books\English directory with the name testfile.str. Windows might have a nasty trick in store for you here. Notepad may tag on an extra extension, making the file name testfile.str.txt. You may not even be able to see this extra extension. Right click on the file and select properties. Look at the extension on the MS-DOS Filename. It should be .str. If it is not, here is how to avoid this problem. When saving, enclose the whole filename in double quotes, like this: "testfile.str". I have heard that changing the Save as Type from Text documents to All Files (*.*) may fix it also.

Now, open DromEd and load your level. (If DromEd is open when you save your text file it may not find it. Close DromEd and restart it to correct the problem.) Go to the Object Hierarchy and select a book from Physical-Household-Book. I am using the HammerBook. After creating the book, go to its properties. Books already have two of the four things we need: FrobInfo and the StdBook script. If you search through the parent objects you will find them. Books also come supplied with the Book-Art property. We only need to add Book-Text and the actual file. We've already got the file, so all that's left is the Book-Text property. Click Add-Book-Text. Type testfile (our filename, minus the extension) in the new dialog box and click OK. Click Done to exit the Properties box and go to game mode. You should be able to read your book. (I must admit at this point that I can only read the first page. I have this problem in the original Thief missions also, so I hope that everything works properly for you. If not, let me know.)

Other Readables

Plaques (found under Physical-Decorative) are made in much the same way as books. You need only add the text file and the Book-Text property. Note that plaques do not have a Book-Art property, the lack of which causes their text to be drawn directly on the playing screen. Only the first page is displayed. Adding the Book-Art property to a plaque causes it to be read like a book. Possible art types are: graveD10, ledger, ledger2, parch, parch2, parch3, pbook, pbook2, plaque, and stoned4. I have only tried a few of these. They come from directory names in the Books.crf file. There is a good chance that you'll never need to use them.

But, enough easy stuff. Let's do something different. How about a readable Horn of Quintus? Add a Horn of Quintus to your level (from Physical-Treasure-Quest items) and go to its properties. The horn already has FrobInfo from one of its parents, but we need to change it. We do that by adding FrobInfo, which will override the original. Click Add-Engine Features-FrobInfo. The World Action describes what happens when you use the horn as it sits in the world. It is set to Move, which means you pick it up. Click the World action button and select Script. This tells DromEd that the horn's Script should be activated when it is frobbed in the world. It has no script yet; we will add it shortly. You can leave the Move action or remove it. If it is removed, the horn will not be able to be picked up. It will only display text. If you leave it, the text will be displayed when the horn is picked up. Click OK to exit the dialog box and return to the Properties dialog.

Now we will add the StdBook script. Click Add-S-Scripts. Type StdBook in the first field. This script tells Thief to display the text associated with the object and is activated when we frob the horn because of the frob information we set before. Click OK to leave the dialog and return to the Properties dialog.

Since we will use our test file from before, we need not create a new one. That leaves only one thing to add of our four requirements. The Book-Text property. Remember this one? Click Add-Book-Text and type testfile. Click OK, click Done and go to game mode. You should be able to "read" the Horn of Quintus.

Now for one last example. Go back to the Horn's properties. Click its FrobInfo and then click edit. Notice the Inv Action button. This describes what actions to take when an object is frobbed from Garrett's inventory. Click on it and select script, then click on World action and deselect script. Select Move again if you turned it off earlier. Now edit the Scripts property and change it to StdScroll. I don't know why this has to be changed.

It seems to be required for reading something from your inventory. Go back to game mode. Now the horn is read when you use it from your inventory.

I hope this helps you figure out how books work and how to use them. As always, remember to experiment and look at examples. Have fun.

-Kevin Goodsell, 6-19-99

==

You know how when everyone is new to DromEd, you have bunch of troubles with brushes not being solid or air like you thought? I just discovered something after many hours with DromEd. Selecting a brush, then changing the type (solid, air, etc) has no affect WHEN YOU HAVE THE TEXTURE VIEW UP even though you see the selection change the type.

Turn the texture palette off (Alt-T) first, then change the brush type. Now it works.

I think I've also noticed that if you change ANYTHING with the brushes to render the current portal out of date, then you can nolonger apply textures to faces and see your results. The textures will only apply if the portalization is up to date apparently, even if the brush whose texture you're modifying IS up to date.

Now if only I could figure out why doing a texture "apply to brush" doesn't always work. (It will appear to not apply a texture when the portalization is out-of-date. If there is an asterisk annuciator in the lower left corner near the mission file name, then you must portalize to see texture changes.)

==

--Lockpicking, by Ishy-

I've spent about 1/2 an hour working this out so I thought I'd share it. First, give Garrett the lockpicks (Doh!). Use LockPickSM and LockPickLG rather than BothLockPicks. Then, make the door, chest or whatever you want to be picked. Select it and press the "properties" button. Add "engine features: Locked" and check the box, then add "Dark gamesys: PickCfg" Now edit PickCfg. There are 3 different possible stages of the lock, and each has 4 settings. If you leave all of these settings on a stage blank, that stage will be ignored.

- **LockBits**: This specifies which pick will work with this stage. 0 means neither will work, 1 means the Square-toothed will work, 10 means the Triangle-toothed one will work, and 11 means either will work.
- **Pins**: This is the number of pins in this stage of the lock. Each pin inside a stage is picked with the same pick, and makes a click once it is picked.
- **TimePct**: I think this is the time it takes to pick each pin in the stage.
- **Flags**: I can't tell what "reset on fail" does. "Randomise time" appears to, as you would expect, randomise the time taken to pick each pin.

An Example:

For a lock on which you have to change picks once, and the second stage takes longer than the first:

- LockBits 1-----1
- Pins 1---------2
- TimePct 1------1
- Flags 1--------[none]
- LockBits 2-----10
- Pins 2---------5
- TimePct 2------1
- Flags 2--------[none]
- Lockbits 3-----0
- Pins 3---------0
- TimePct 3------0
- Flags 3--------[none]

==

Locked Portcullis
I found this by checking "Cragscleft Prison".

1. Create and place your portcullis. Try Port4Vert5Horz (-1264).
2. Create and place your lockbox. Use LockBox (-4587). Add the following properties:
   2.1. Dark GameSys:PickCfg{1; 1; 10; [None]; 10; 1; 10; [None]; 1; 1; 10; [None]}
   2.2. Tweq:LockState{On, Reverse; [None]; 0.00; 0}
3. Go back to the portcullis and add the following properties:
   3.1. Door:Translating{…Closed…}
   3.2. **Be sure there is not EngineFeatures:Locked{}**
   3.3. MetaProperty:FrobInert
      3.3.1. Engine Features:FrobInfo{ [None]; [None]; [None]}
4. Add this link to the portcullis:FLAVOR{LOCK};FROM{portcullis};TO{lockbox}

By the way, when I finally added the FrobInert to my portcullis, suddenly the sound started working on my mission. Before that, sound was sporadic at best.

==

Topic: Here's a list of motions available in conversations
Sorry, subject should say "motions", not "actions", but can't change it now.
I've figured out the motions available in Conversations. I don't think I've missed anything, but if anyone knows better correct me.

To make an actor make a motion during a conversation, set that step's action to "Play sound/motion". Arguments 1 and 2 are left blank, and argument 3 should be set to one of these (the part within quotes, the rest is explanatory):
"Conversation 0, Quux 0" - Cross arms behind back
"Conversation 0, Baz 0" - Jump and shake arms a little as if agitated
"Conversation 0, Baz 0, Quux 0" - Raise arms as if to say I don't know
"Conversation 0, Bar 0" - Lean backward as if surprised
"Conversation 0, Bar 0, Quux 0" - Cross arms in front
"Conversation 0, Bar 0, Baz 0" - Move arms as if counting or conducting music
"Conversation 0, Foo 0" - Raise left hand and nod as if greeting
"Conversation 0, Foo 0, Baz 0" - Nod head as in agreement or acknowledgement
"Conversation 0, Foo 0, Bar 0" - Take a step forward with one foot, then back
"Conversation 0, Foo 0, Bar 0, Baz 0" - Look right
Other combinations of Foo, Bar, Baz, and Quux only create duplicates of some of the above motions.
Now, if only the speech were so easy, my first two missions would be done soon.
I'm really really sorry about that. As it turns out, that could have been handled with flags of "Conversation 0: Cross Arms"
"Conversation 1: Jump and shake arms"
but I hadn't grasped the concept of counting beyond 0 yet.
Oh, and the only thing you missed is:
quux foo is "cross arms"
quux bar is "keep arms crossed"
So if you played either by itself, they'd look much the same, but if you wanted a guy to cross his arms and keep them there for a while, play quux/foo and then play several quux bar.
incidentally, the name of the bar baz schema is "george bush wave"
==

When you want to clone an object consisting of mutliple brushes, or objects, you can use the following trick to align it on the grid.

NEVER CLONE A MULTI-BRUSH THAT HAS PARTICLE EFFECTS! DromEd MAY DUPLICATE ALL OBJECTS IN YOUR MISSION!

First: align every brush you want to clone on the grid. This is usually the case, anyway.

Then select each of them with shift-click, to make a multi-brush.

Then press "INS" to clone the multibrush. Do that in the view you want to use to drag the multi-brush, e.g. top-view.

Use "+" or "-" to zoom in that view, and drag your multi-brush with shift-mousedrag around.

Drag it to the position you want it.

Now to align it. Do you see the coordinates on the lower left corner of the DromEd window? They are relative to the origin of the multi-brush. Just enter integer numbers, quarters (e.g. 8.00 or 8.25) for them. Watch how your multibrush moves. When you make these numbers something like 5.00, then your multibrush will be aligned to the grid.

Even easier is it to enter the offsets directly, without even dragging the brush or object.

After positioning manually, you must de-select the multibrush, then use the “hilight_check_snap 1” and “hilight_do_snap 1” commands to force alignment of the individual brushes. Use “hilight_clear” to clear the hilight.

You can clone an object easily more than one time. Just clone it once, enter an offset (e.g. 4), then clone it again, and enter 8 as offset. So you get objects that are evenly spaced.

===

<u>Making an location based alarm:</u>

I`ll try to describe how to build an alarm that goes off when the player enteres an area and which could be disabled by a switch:

<u>Items needed:</u>

An alarm system that works like described in the tutorial of A Little Mouse, a bigzaparc (-4550), two ham novices (-1730), a guard(-51),a switch and a portcullis

First you arrange some big zap arcs in the area where the alarm should be triggered. Remember a ID of one of them, or put a Marker there and use its ID.

Now you put yor Hammer Novice somewhere near them and set the following on him:

Renderer: Has Refs: False (An invisible Hammer)

AI: Core: Alertness Cap None,None,None

Speech: Voice: (delete the value here)

OK, now we have a dumb Hammer here, very useful, of course he is.

Now put this link on him:

Flavor: AIWatchObject

From:Your Invisble Hammer

To.The ID of the big zap arc

Player intrusion

3

10

...

frob object

one of your alarm buttons

Now when you touch your big zap arcs the alarm will go off.

Good, but how to disable.

Here is my idea, it will just work one time per playing but I`m working on a better one right now:

Make a room which somewhere in solid space away from your level.

Put in your guard and another Ham Novice (yeah, I hate this guys).

Separate them by portcullis and give them a light source and a room.

Lock the portcullis and give the guard the link:

AIAttack

From:the guard

To: the poor Ham Novice

Priority Absolute

Now let`s edit the Ham Novice:

AI: Core: Team: Bad2

AI:Attrbutes: Vision: Null (all my Hams seem to be blind, but it`s important)

AI:Attributes: Hearing: Null (this is new)

Now make a switch, which will be your Main Power Switch.

Edit it to be default on position ON.

Link it with Control device to the portcullis (maybe an Inverter between them).

Link it with Control device to all your "laser beams".

Now try and turn it. Ok, the laser beams disapper but the evil Hammerite will activate the alarm.

Time for a little trick:

When you pull the switch a poor blind Hammerite is just to receive his death, and that`s the point:

Put an Receptron stim on the Hammerite which will be like this:

Object: A Ham Novice (the one in the seperate room)

Stimulus: SlashStim

No Min

No Max

Effect: Destroy object

Target Object: A Ham Novice (the evil one who will activate the alarm)

That`s it! If done right your pulling of the switch will cause the beams to disappear, the portcullis to raise and the guard to kill the Hammerite.

This will result in the other Hammerite being removed so that he can`t do his job in activating the alarm.

Hope this is useful for some of you...

I`ll make other vesions of this (including blowing the player up) later and it will be in my mission.

Angus MacNokker

==

MAKING A CUSTOM SPELLCASTER

Creature; Model Name - Voice
Haunt; exphamz4 - vhaunt
HammerZombie; - hammzom vzomb
Zombie; expZOM - vzomb
Murus; expmurus - vmurus
Apparition; expbeneg - vappart
Apebeast; expconms - vapes
BugBeast ; bug3 - vbug
Frogbeast; expfrog - vfrog
Trickster; expcon - vcon
Spiderbeast; spidbea - vspider
Craybeast; mawcray - vcray
FireElement; vfire - fire1
ham guard; exphamh4 - hammrs
ham worker; expham2 - hammrs
ham novice; exphnovi - hammrs
ham priest; exphamw2 - hammrs
ham corpse; expbeneg - hammrs
Maleserv1; expserv2 - normal1
Femserv1; expwo1 - normal2
Femserv2; expwo2 - normal2
Femserv3; expwo3 - normal2
Merchant; merchant - normal1
sword guard; mguard - guards
BaffGuard; mguard - guards
RamirezGuard; expramg2 - guards
Con Guard; exphcngs - guards
BaffSergeant; expbafes - guards
RamirezSarge; expramsg - guards
bow man; bowman2 - guards
RamirezBowMan; expramb2 - guards
ConBowman; expcobo - guards
thief; expgarsw
KeeperAgent; expkeeph
Burrick; burrick - vburr
DeadBurrick1; burdead - vburr
DeadBurrick2; burdead2 - vburr
Crayman; crayman - vcray
Spider; spidey7 - vspider
HugeSpider; spidey7 - vspider
SewerSpider; spidally - vspider

WHY IS THIS USEFUL?
The following tutorial should explain why.
MAKING A CUSTOM SPELLCASTER
This is only one way to do it, it is meant to encourage you to use your imagination and think of some interesting variations or completely different creatures. First, prepare everything (load script etc.) Then make your room desired size about 30x30 will suffice, insert a light near the top of the room. Portalize. Next make the starting point at the opposite end of room with the light. Insert an Apparition, physical>creature>undead, place it near the light and floor it.

Now there's a few things you need to change in the Apparitions Properties:

i) An Apparition>Renderer>Extra Light Delete This

ii) Apparition>Renderer>Extra Light and Transparency Delete Both

iii) Apparition>Shape>Model Name (double click) change to merchant

iv) Apparition>Speech>Voice (double click) change to vapes

Now delete all of the GhostShots (they will be in the middle of the room. Create Physical>Projectile>CritterShots>firebolt

Click on the Apparition and then goto the Links, Add Contains with From (apparition number) To (firebolt number) CulpableFor should have appeared if not add it with the same From and To Add AIProjectile with From (Apparition number) To (firebolt number)

Now for some finishing touches, Create SFX>hauntsmoke make two of them

Now for both goto Links and Add ParticleAttachment, double click on ID number change the following; Type : Joint Joint : left fingers (for one) right fingers (for other) Now change the Properties for both; Add>Renderer>Self Lit :10

Additional things you can change, goto the Links for the firebolt and double click on the ID number for AIProjectile, now change; Selection Desire to Very High, Accuracy to Very High, Check the Leads Target box, Launch Joint to right wrist.

It is possible to change the max distance that the firebolt can be fired among other things by going to AI>Ability Settings>Ranged Combat in the Properties of the Apparition

Surround the room with a Room, Compute Pathfinding Database etc. Portalize!

This is only changing basic things, there are many other things to change too, for example Act/React>Receptrons, this will allow you to make the creature react differently to what you do, for example affected by holy arrows. You can do really complex things with this. I'm looking for a way to make creatures levitate like a fire elemental, it's only a matter of time.

Experiment! Put your creature up for download! And have fun!

The_Thief

===

How to get a full list of DromEd Commands

Luckily, there is a feature within DromEd that generates a list of all the commands the program can use. First, load up DromEd, and then go to your little console box in the lower right hand corner. Go and type in: **dump_cmds dump_cmds.txt**

Walla! There should now be a text file in the directory you keep DromEd in. This should provide you with a complete listing of commands with explainations. If that isn't the cat's pajamas.

===

CreateTrap to create items behind a locked door

It's possible in Thief to get an item hidden behind a locked door. Remember the locker with the notes in Cragscleft? Load it up again and see - you can get the notes while the locker is still locked if you look around and keep pressing "use".

I had a similar problem in my mission. In case anyone else is also having this problem, here's how I fixed it.

Put a CreateTrap in the locker and add a Contains link from the CreateTrap to the actual object you want in the locker. Then add a ControlDevice link from the locker door to the CreateTrap. Add the TrigDoorOpen script to the door (check off "don't inherit) and voila - you can't get the object until you open the door, because the object isn't there yet.

===

Useful DromEd commands

hilight_by_prop	: give property name, highlights obj's with it
hilight_nonaxial	: highlights any terrain with non-90 angles
hilight_media	: highlight terrain w/media_op of type <arg>
hilight_texture	: highlight terrain w/texture id <arg>
hilight_split_obj	: highlight objects crossing a portal
multibrush_the_highlight:	make hilight objs the multibrush
hilight_brush	: hilight current (0) or brush_id
hilight_check_snap	: hilight unangled unsnapped brushes, or if (1) all unsnapped brushes
hilight_do_snap	: grid snap all hilight brushes...
hilight_list	: list objs, or, if arg 1, all brush ids
hilight_global	: if true, we will hilight everything, else just active
hilight_autoclear	: do we autoclear old hilight and make it only active
hilight_use	: set which hilight bit to use (bitfield, must be just 1)
hilight_clear	: clear current highlights (all if 0, else bitfields)
hilight_activate	: turn on hilight bits
hilight_deactivate	: turn off hilight bits
hilight_add_prop	: Add a named property to all hilit objects
hilight_rem_prop	: Add a named property to all hilit objects
hilight_render	: boolean toggle
hilight_obj_type	: hilight all instances of an archetype

hilight_room_id	Hilight the specified room brush
show_raycasts	display light raycasts
rooms_build	Convert rooms to internal rep
fix_rooms	Fix dangling room pointers
hilight_room_id	Hilight the specified room brush
compress_br_ids	compact brush ids
floor_object	function
wall_object	function
ceil_object	function
spiral_serf	build spiral stair, 0 for dialog
stair_serf	build straight stair, 0 for dialog
set_grid	force brush to grid
add_family	add another texture family
compress_family	remove unused textures from family or <all>
remove_family	remove family, use <all> to clear all
ai_aware_of_player	Toggle AI awareness of player
aiawareofplayer	Toggle AI awareness of player
script_load	Load a script file, add to mission
script_drop	drop a script file from mission.
script_drop_all	drop all script files from mission.
find_obj	Look up an object by name or number
persistent_player_pos	Toggle whether the player comes back to game mode
brush_select	int function
obj_brush_select	set current brush to objID
cam_to_brush	move camera to look at cur brush
dump_cmds	dump command list to file

===

Making Books Inventoriable

Book are not normally "inventoriable", that is, you can't take them. You can only read them. However, if you want a book (or other item) to be "takable to inventory" do this:

- create your object (in this case, a book)
- click Properties
- Add->EngineFeatures->FrobInfo; there are three fields and a check box. Books already have "script" set in the first field. Select the first field and choose MOVE. Now the field will say "move,script". By this action, you have made your book takeable yet still readable. In the second field (which currently says "none"), select SCRIPT. This second field is executed when the item is in inventory and the player frobs it.
- Now define the specific scripts the frob info refers to. Add->S->scripts. A book will already have "StdBook" in the first script field (inherited from parent books). In the second field, put "StdScroll".

StdBook and StdScroll are very similar scripts. The only difference (I observed) is that one is for frobbing readables from the world (StdBook) and the other is for frobbing readable within inventory (StdScroll).

==

Does anyone know what the four model types (OBB, Sphere, Sphere Hat, None) are?

An OBB is an "oriented bounding box." Basically it's a rectangular prism, a box with depth, width, height, and orientation. All OBB physics models must be location- and rotation-controlled; they do not move freely under physics. Usually this means they don't move at all, but there are other sorts of "controlled" movement, as with doors and elevators. Contrary to what you might think, crates and the like are not OBBs; they are sphere-hats (see below).

A sphere is what it sounds like, mostly. Objects with sphere physics models can actually be controlled collections of spheres in certain pre-programmed circumstances, as with creatures. Generally, though, it's a single sphere moving freely under the influence of physics.

A sphere-hat can be pictured as a sphere with a square mortarboard hat on top. The "hat" is a square as big on each side as the sphere's diameter. The hat is always on top of the sphere in the world's frame of reference; it does not rotate with the sphere. Sphere-hats are used for objects which can move freely under physics but can be stacked or stood upon, as with crates.

None is what it sounds like. The object has no physics model. It is nonexistent as far as physics is concerned.

The command "show_phys_models" can be helpful to understand this. It will overlay a wireframe representation of all physics models on your 3D view. Sphere-hats just look like spheres, though; it doesn't draw the hat.

==

CONCRETE ROOM

In tutorials that I've read, when making room brushes w/ the WelcomeRoom script in making Type 4 Miss. Objectives it said that you must make the room brush a CONCRETE ROOM....

What does this mean and what do I have to do to make it work????

It means you have to change the brush to "fill solid". You know, like concrete.

No, seriously, a concrete object is a "real" one in your level (well, virtually real) while an archetype is a "template" for an object.

Archetypes have negative indices to distinguish them from concrete objects, which have positive indices.

Rooms are a little hard to get the hang of at first, because a concrete room still sort of acts like an archetype room.

Bring up the object hierarchy dialog and switch to rooms in the combo box at the top. When you add a new room type here, it asks you if you want it to be concrete. If you say yes, the new room type is stored in your .mis file, if not, it becomes an archetype and you would have to save the gamesys to keep it, and it would be available in all missions.

So, once you've created a concrete room, you still bring up the object hierarchy to select it and press "Create" to make an instance of it. That's why I said it's still sort of like an archetype: you can have many instances of a "concrete" room. (So it isn't really concrete, is it?)

==

For a switch that starts on:

Load your mission

In the editor, issue the command "set game_mode_backup 0". This will prevent DromEd from undoing the changes you make by playing in game mode, so watch out and don't save your mission after playing.

Go into game mode

Flip the switch. Wait for it to finish animating its joint.

Go back into editor mode.

Bring up the switch's properties, and note the values of its "Shape/Joint Positions" and "Tweq/Joints" properties.

"set game_mode_backup 1" again, because you're done with all that.

Load your mission.

Set your lever's properties to the ones you noted it as having after being flipped.

Here's a different approach:

Instead of editing the switch, edit what it controls. i.e. if you've linked it to a door, edit the door's properties so that it begins the game in the open position.

Well, you have to do both, really. The lever has a notion of which direction causes it to send a "Turn On" message and which causes a "Turn Off". If the thing that it controls starts in the "on" state (which is open in the case of a door) then the lever should be editing into the already-flipped state to correspond.

If you want, you can also play tricks with putting inverters in the way, flipping the switch upside-down, and such, but actually editing the switch keeps everything more consistent.

I was only considering how you get the switch to start on. The thing it controls is another matter, and will vary according to the type of thing in question.

==

Ok, DOORS THE BASICS

First create the door you want somewhere in your level. We'll position it in a minute.

Then you need to hole in your wall to put a door in.

Create an air brush of the same height and width as your door. The depth will obviously be the same as your wall.

Not down the x/y coords of the airbrush and select your door.

Click floorme to get your door flush with the floor. Then copy those coords into your door's coords. This will centre it in it's hole. You might need to rotate it by 90 degrees.

Now, what kind of door did you pick:

(1) Slidy: (easy)

You need to make a hole in the wall for the door to slide into.

When you've done that, go to game mode and test it. It if slides the wrong way, go back to the editor and rotate it 180 degrees.

(2) Spinny: (harder)

Ok, go into game mode and test the door.

(i) Is the hinge on the correct side? If no then rotate the door by 180 deg.

(ii) Does it spin correctly? (i.e. towards or away from you). If not, bring up the door's properties, and find the 'rotation' property. Change the 'clockwise' tickbox to anticlockwise

and change the open angle from 90 to 270 or 270 to 90, depending on what it was.

Locking Doors:

Bring up properties, Select Add>EngineFeatures>Locked and tick the box. Do not lock the door when you have linked the door to a lockbox (flavour=lock). Using a lockbox automatically makes the door locked (distinct status from setting the locked bit).

Setting up doors to be picked:

Bring up properties, Select Add>DarkGameSys<PICKCFG.

You can have up to three stages to your pickcfg.

LockBits are set as follows: 0=Ignore this stage; 1=square lockpick required; 10=triangle lockpick req'd; 11=random pick req'd

Pins is the number of 'clicks' for this stage. Numbers in range 1-5 are good.

TimePct is the time this stage takes. Again numbers in range 1-5 are best.

The flags aren't implemented as far as I know.

Setting up doors to work with keys:

1. Make a key.
2. Key Properties Add>EngineFeatures>KeySrc set the two boxes to unique numbers for this key.
3. Door Properties Add>EngineFeatures>KeyDst set the two boxes to the same numbers as the key.
4. By copying these numbers you can have multiple keys for one door or vice versa.
5. Selecting MasterBit on a key makes it open all doors than can be opened with a key (with matching RegionID index). Handy for debugging your level.

Setting up doors to work with switches:

Make a position switch e.g. BigFloorLever and note down it's object number

Select the BigFloorLever, click Links.

Select Add, then ControlDevice and type BigFL number in FROM and door number in TO.

AIs and doors.

To allow an AI to go through a locked door, set up a key to work with the door (this key does not have to be available to the player).

Select the AI, click Links, Select Contains, From=AI number, To=key number, Data=Generic Contents.

Note: this will NOT put the key on his/her belt. You'll need a separate key for that, another contains Link, and Data=Belt.

==

TrigRoomPlayer

You edit your room's properties and add S->Scripts and type 'TrigRoomPlayer' in the top field. This can be used in conjunction with ControlDevice links and other things of that nature.

Oh, a word of caution though. If you edit one room's properties, you will change the properties of every room of the same archetype. So, if you want to put this script on only a few different rooms, you will need to add a new concrete room archetype so that every room will not have this script added.

You can trigger sounds by making Fnord->Ambient Sound and add the script "AmbientActivate" to it.

Link the room to the sound with a Control Device and the sound will play each time you enter the room.

==

Candle

You shouldn't need to douse a candle with a water arrow, right? Here's how to create a more realistic candle.

Create object SFX > FireFX > Flame > DynFlame. Delete everything but the flame (it's also called DynFlame.)

Resize DynFlame to an appropriate candle flame size. I used 0.13 d, 0.13 w, 0.3 h.

Create object physical > lights > Candle+Stick1.

Add property A:AmbientHacked to DynFlame. Change radius to a more appropriate setting. (From how far away can you hear a candle flame?) Note, if you delete AmbientHacked in the lower section (parent archetype), all DynFlames will lose sound! Don't do that.

Align DynFlame with Candle+Stick1. I'm sure you can use an attachment link for this, or do it the simple way.

Add property Engine Features > FrobInfo to DynFlame. Change World Action to 'delete'.

Since the previous incarnation of this tutorial, people have been complaining of disappearing DynFlames. This is easy to correct. Add prop Tweq > DeleteState. Press the button next to AnimS, and uncheck On.

That's it! Now you can walk up to your candle, center it on the screen, and right click. The flame will disappear.

Only problem - the flame will not 'light up' like other selectable objects. There is probably a way to link up the candlestick with the flame, so, when the candlestick is frobbed, the flame is extingushed. I might try that.

==

LIGHTNING TUTORIAL

It's quite an interesting effect, and it took me hours to figure out, SO YOU BETTER LIKE IT. I assume you know everything about room brushes, StartingPoint, script_load convict, etc. If you don't, you shouldn't be reading this, because I'll only confuse you.

Okay, there should be an 'outside' air brush where you want the lightning effect to take place. And it should be surrounded with a room brush.

1. Add the script 'TrigRoomPlayer' to the room.
2. Create an OldElecWallLight. Resize to 0.01, 0.01, 0.01, and change the dimensions prop to the same. Now, move this speck to a place where you want your lightning effect to center. I used the top, lower left corner. Dunno what it'll look like in other places.
3. Create a VoTrap. Link the Room to the VoTrap with ControlDevice. Link the VoTrap to -3692 with SoundDescription. This is unlock_earthwar. It sounds somewhat like thunder. You can replace it, if you desire, but that's another matter entirely.
4. Create a RelayTrap. Name it 'PapaSmurf' as not to confuse it with the other RelayTrap you'll create. Link the room to PapaSmurf with ControlDevice. Link PapaSmurf to OldElecWallLight with ControlDevice.
5. Create another RelayTrap. Let's call it 'BrainySmurf.' Link OldElecWallLight to BrainySmurf with ControlDevice.
6. Create an Inverter. Link BrainySmurf to the Inverter with ControlDevice. Link the Inverter to OldElecWallLight with ControlDevice.
7. Create a DestroyTrap. Link BrainySmurf to the DestroyTrap with ControlDevice. Link the DestroyTrap to PapaSmurf with ControlDevice. Link the DestroyTrap to the VoTrap with ControlDevice. Link the DestroyTrap to BrainySmurf with ControlDevice.

Basically, this is what will happen:

You'll enter the room. The room will trigger PapaSmurf, who turns on the OldElecWallLight. The room will also trigger a VoTrap, which is the sound you'll hear. OldElecWallLight will then trigger BrainySmurf, who will, with the inverter, turn OldElecWallLight off again, and trigger the DestroyTrap. The DestroyTrap will kill PapaSmurf and the VoTrap so the effect and sound occurs only once. It will also kill BrainySmurf as to stop the flickering.

That's it! Why did it take so long for me to figure this out? I guess I'm an idiot.

The effect itself should look like flickering lightning. If you want, I can create an 'example mission' so you can watch it in action. Note that this lightning *does* affect your light meter.

===

To trigger a sound with a ControlDevice, set up the link as normal and add the script 'ActivateAmbient' to the AmbientHacked marker.

Add a CD link from an Old Elec Lite to the ambient sound. So this way when the light flickers the sound plays.

===

Flickering HangSpotFlush
This example is taken from Cragscleft Prison mission near the guard station for cell blocks 3,4.

1. Create FlickerTrigger1
   - FlickerState: {On; [None]; 0; 0}
   - Flicker: {Continue; NoLimit,SimSmallRad; Scripts; [None]; 3000}
2. Create FlickerTrigger2
   - FlickerState: {On; [None]; 0; 0}
   - Flicker: {Continue; NoLimit,SimSmallRad; Scripts; [None]; 2000}
3. Create HangSpotFlush
   - Scripts: {AnimLight;;;;FALSE}
   - Renderer->AnimLight: {flip between min&max;63;63;125;0;0;0,0,0;FALSE}
   - Renderer->Light: {0;0,0,0;0}
4. Add Link: FLAVOUR(ControlDevice) From(FlickerTrigger1) To(HangSpotFlush)
5. Add Link: FLAVOUR(ControlDevice) From(FlickerTrigger2) To(HangSpotFlush)

===

I noticed that scripts seem to have a hierarchy, so I poked around a little and here is what I found... Not sure how helpful this is, but it seemed interesting. Anyone know of a convict script that is not in this list?

RootScript ActivateAmbient
RootScript AI
AI CorpseFrobHack
AI Elemental
Elemental FireElement
AI QuaffHeal
AI TrigAIAlert
RootScript AnimLight
AnimLight Extinguishable
AnimLight WindowShade
WindowShade ControlWindowShade
RootScript Arrow
RootScript Attack

Attack AttackActivate
RootScript BlackJack
RootScript Burplauncher
RootScript CloneContactDmg
RootScript CloneContactFrob
RootScript Container
RootScript Crystal
RootScript DeathSentence
RootScript DescribeSounds
DescribeSounds AmbientSounds
DescribeSounds VOSounds
RootScript DoFlameSource
RootScript Door
Door SecureDoor
Door StdDoor
StdDoor ToggleDoor
Door TrigDoorOpen
RootScript EatFood
EatFood AirPotion
EatFood HolyH2O
EatFood SpeedPotion
RootScript ElevatorSounds
RootScript FrobSlay
RootScript GoMissing
RootScript HolyFont
RootScript JAccuse
RootScript JunkWebs
RootScript Legible
Legible StdBook
Legible StdScroll
RootScript LoadoutBox
LoadoutBox LoadoutCache
RootScript Lock
RootScript LockPick
RootScript LockSounds
RootScript LootSounds
RootScript MovingTerrain
MovingTerrain StopAtWaypoints
RootScript NotifyRegion
RootScript OnOffSounds
RootScript OutDamnSpot
RootScript PressurePlate
PressurePlate CollapseFloor
PressurePlate TrigPPlate
PressurePlate TrigPPlateImmed

?? PressurePlateInactive PressurePlateActive PressurePlateDeactivating
PressurePlateActivating
RootScript Physics
Physics ActiveMine
Physics CollisionStick
CollisionStick DeployRope
Physics Mine
Mine GasMine
Physics MossLump
Physics MossSpore
Physics PlayNoisemaker
Physics StickyWebs
Physics StdController
StdController Gong
StdController StdButton
StdController StdTwoState
StdTwoState StdLever
StdLever JumperSwitch
StdLever StdGauge
StdController TweqOnOff
Physics TrigOBB
TrigOBB Glyph
Physics ZombieRegen
Physics FlashBomb
Physics Corpsed
RootScript ReloadTweqEmit
RootScript Room
Room TrigRoomCreature
Room TrigRoomPlayer
Room TrigRoomPlayerTrans
Room VictoryChecker
VictoryChecker VictoryCheck
Room WelcomeRoom
RootScript RopeFX
RootScript Sanctifier
RootScript SpeedyPlayer
RootScript StdKey
RootScript StdElevator
RootScript StdParticleGroup
StdParticleGroup FireElemSparx
RootScript StdTerrpoint
RootScript Sword
RootScript TrapConverse
RootScript TrapDeadfall
RootScript TrapDestroy
RootScript TrapInverter

RootScript TrapOnFilter
RootScript TrapRelease
TrapRelease TrapCreate
RootScript TrapRequirement
TrapRequirement TrapRequireAll
TrapRequirement TrapRequireAny
RootScript TrapRelay
RootScript TrapSetQVar
RootScript TrapTweqEmit
RootScript TrapTeleporter
RootScript TrigFlicker
RootScript TrigSchemaDone
RootScript TrigSlain
RootScript TrigUnlock
RootScript TrigWorldFocus
RootScript Waypoint
RootScript WatchMe
==

To set an EmitTrap that spits out one firebolt each time a pushbutton object is pressed.
I tried this:

1. Create an EmitMouthWdMet whatever and put it on a wall. It's just decoration.
2. Create an EmitTrap; position it in the EmitMouth and point it in the proper direction. Use "cam_to_brush" to verify that it's pointing correctly.
3. Add Tweq->Emit and specify firebolt.
4. Add S->Scripts:ReloadTweqEmit
5. Add a pushbutton; add a link flavor(controldevice) from the pushbutton to the EmitTrap.

==

Apparently, some doors allow light to shine through, unless set renderer->shadow 10.

==

The script itself is called "TrigRoomPlayer."

I don't have Dromed open, but this is basically what you need to do:

I assume you have convict loaded, etc.

Lay down your room brush.

Highlight the room brush and hit 'create.' Highlight Base Room and hit 'add'. Name your room "Wyclef" or whatever else you choose to. Highlight the newly added room and hit 'create.'

Your room brush should still be highlighted. Hit 'edit' at the bottom of the screen. Add property S > Scripts. Change one of the fields to "TrigRoomPlayer."

Back at the main editing window, create a fnord > TrapTrig > SoundTrap > VoTrap. It doesn't really matter where you place it, but put it in some spot where you'll remember it. Link the Room to this VoTrap w/ControlDevice.

Find the Garrett voice-over you want to play in the Hierarchy. Note the number. Link the VoTrap to that number w/SoundDescription.

Now he should say something when you enter the room (every time). When you want Garrett to speak only once, you could link the room brush to a RelayTrap. The RelayTrap links first to the VOTrap and then to a DestroyTrap. The DestroyTrap has ControlDevice links to the RelayTrap and to the VOTrap. Thus, the room brush sends a TurnOn signal to the RelayTrap. The RelayTrap first sends a TurnOn signal to the VOTrap, followed by a TurnOn signal to the DestroyTrap. The VOTrap starts the SoundDescription. The DestroyTrap destroys the RelayTrap and the VOTrap. The next time Garrett enters the room, the traps are already destroyed, so nothing happens.

==

Stair_serf

The stairs are built ascending (first solid brush) from Bottom to Top, starting with the selected face (or South for default) of the current air-fill brush. Use the block height multiplied by 1.5 for the number of steps, yielding ¾' high steps. The steps rise from bottom to top, and progress from South to North.

- Best results achieved by deselecting "slats".
- You'll probably have to select the start stair block;
  - Bring up the texture palette and select your default texture.
  - Press "reset" texture.
  - Press TAB.
  - Repeat until all stairsteps are reset to matching texture.

WARNING: Always use the "default" selection, rather than selecting a face. Sometimes, DromEd screws-up in the latter case; it builds solid steps, but the steps never materialize in portalization…strange bug, probably screwed brush timing. You may have to change the heading and dimensions of the air fill brush to get the steps to come out right. Also, using stair_serf may generate steps that are not snapped properly and when you snap them, they imbed themselves into each other making portalization extremely slow (at the "merge polys" step).

Never imbed solid steps within each other. Always try to make them adjacent.

A better method of making stairs that is less sensitive to grid snapping glitches, is to carve the steps out of a solid block. Create solid block; create air brush 6.0 wide, 1.0 deep, and about 10.0 high. Iteratively place and clone the air brush, rising 0.75 with each step. The brick steps leading to the basement in "The Sword" mission is an example of this technique.

==

Wedge

Wedge blocks are drawn from bottom face to top face. Side_3 is the hypotenuse.

==

Unlit Torch

Select the Torch object:

- Set AmbientHacked->Flags:TurnedOff
- Set Tweq->ModelState{[None];[None];0;0}
- Set Renderer->AnimLight{Zero Brightness;…}
- Set Shape->ModelName:Newt01

==

Up/Down Switch Starting in ON state controlling an OldElecWallLight in ON state

Up/Down Switch

    Shape->Joint Positions: {0.70,0,0,0,0,0}
    Tweq->JointState: {Reverse;[none];On,Reverse;[none];…}

```
OldElecWallLight
        AmbientHacked: {0;0;[none];;;}
        Tweq->FlickerState: {On; [None]; 1161; 1}
        Tweq->JointsState: {[None]; [None]; On; On; [None]...}
        Renderer->Self Illumination: 1.00
```

==

Here's a trick: you can make ambient light points with a *negative* light intensity, to make an area more shadowy. It won't look darker on screen, but your light gem will go dark when you stand there. Convenient when you've got a place that *looks* like it's in shadow, but your visibility gem stays lit (and you don't want it to).

==

I'm using VideoFramer (www.flickerfree.com) for my mission briefings and thought I'd post a few tips. It's a beta and not perfect, but works pretty well.

It can't handle large numbers of files. Once I import a certain number (lots because they're mostly still frames) and save the project, next time I try to load it gives an error message about not being able to access a file and won't load the project.

I think this may be related to path/filename length, so keep the files in a root directory if possible.

A better solution is to load up all the frames for a particular section of your video, and save them as an intermediate AVI. Then import these into your final project, which will only need to import a few files instead of hundreds. It also makes it easier when you "mess up" a project, as you don't have to re-import hundreds of single frames. Save the intermediate AVIs as "full frames" (no compression) so you won't lose any data. (Compressed frames may look "fuzzy".) You can use compression for the final AVI.

If you are using a section of an existing AVI, VideoFramer can import it and truncate the parts you don't need, but it's still good to also save this an an intermediate AVI (no compression) to save work in the final project.

You can type in exact frames for the start/stop frames of objects (single frames, collections, and videos are objects) but this can get confusing, as the other objects don't move to "take up the slack" and have to be moved manually.

If you make all your major sections as intermediate AVI's, you can just import each of these into the final project in the order it appears, then right-click each one in order and select Add. This starts the next clip exactly where the previous one left off, as opposed to dragging the object into the track. If you import still frames that you want to appear for several frames, adjust this immediately after you add the image to the track, so the next one you add by "right click/Add" will automatically be positioned correctly.

Another tip: VideoFramer shows thumbnail versions of each object, but doesn't always refresh them correctly, so you can't tell which item you want to select for editing. Bring any other window to the foreground and make sure it obscures VideoFramer, then switch back, and the thumbnails will get refreshed.

Anybody else using this product that has comments?

Doesn't it say 'made using video framer' or something similar in the bottom left corner?

That's one reason I chose not to use it.

Personally I used MGI VideoWave for my briefing, it has some useful effects.

For some additional effects I used the Image Transitions in Anim Shop Pro.

VideoFramer is useful for adding sound to the briefing, but if you save it's still going to have that annoying message at the bottom of your avi

==

WARNING: Didn't grab coplanar case!

What does this message mean? I get in the CSGMERGE window when I optimize.
It probably means there are some brushes that are placed a little awkwardly, and the DromEd optimizer thing doesn't like 'em.... Have you snapped everything to the grid? I had that message come up a few times and snapping my brushes cleared it up. The easiest way is to use the hilight_check_snap and hilight_do_snap commands.

- Caused by two adjacent air brushes that weren't quite aligned (snapped) to each other.
- Caused by air brush within solid brush that wasn't aligned exactly 0/90/180/270 degrees with the containing wall (used as an open window).

==

Using one button to open/close a door

Should work, I'm using a button to open/close a door and didn't have to do anything special. Just the standard red button .
Oh, one difference, I made the button a FrobProxy for the door. Maybe that's the key.

==

Limiting Knockouts

Actually, the whole thing was prompted by other peoples revelations on mission objectives. I read the messages about using scripts from other levels which weren't available in convict, and I decided to play about with the editor to see if I could reproduce the missing objective of bring an object to a certain place.

This in turn revolved around the tidbit about changing the goal variables midgame, to which I saw too methods from previous posts.

The first one was by using a 'TrapSetQvar' to change a goal variable from a control device link, the second was by using a receptron to change it.

The actual method that I have uses the latter, not the former, but I'll talk about that later.

Using the Receptron, you can set it to change a goal variable depending on a certain stimulus. Setting it to knockout will mean that if, say, the guard that it is assigned to is knocked out, the goal will turned complete.

Not that useful.

However the trick is simply to compound this slightly. Using a 'blue room' (I think thats the term), I created a potted plant which could knock things out (done via Act/React Sources:knockout) and rested it on a wooden door.

I then gave the door hitpoints, 5 to start with.

Underneath this door, I created another door, and to this one, I added the receptron which completed the objective.

The final step was by changing the receptron on the guard, so that when he is knocked out, instead of triggering the objective, he damages an object for two hitpoints, and the object is the door.

Repeat this for multiple guards, and you have the following situation.

Knockout one guard, door loses 2 of its 5 hps. Second time, it has 1 left. And after the third knockout, it disappears. (Oh, I had to change a property on the middle door so that when it is at 0hps it is destroyed). Once it has disappeared the potted plant falls, hits the second door and delivers the knockout that sets the objective.

This could be adjusted for any number of knockouts.

De deeer!

Of course, one side-effect is that every person who you want to be counted as being 'donotknockoutmorethan 3 times' would have to have this receptron link on them. I have quickened the process by creating my own archetype, but that means that I have to save a cow file, and I think I heard that I couldn't distribute that or something?

Well, thanks for listening. I'll probably expand to say what other areas it can be applied to later on, and I might possibly write a better tutorial.

I have just realised that this does allow you to reproduce the affect of having to use object A on object B. Just simply use a receptron for HolyStim and give item A a holy source. Must be the time, this is addictive and Ihave to get up in 5 hours for work.

One other side affect of the blackjack limiting is that it calculates via blackjack hits, not actual knockouts. Hitting one corpse three times has the same affect.

Another thing that would work with this is an objective of NOT going to a room, as you can trigger off a door open and shutting via the RoomTrigger script thing.

At the moment trying to puzzle out how to create my own receptron and sources. They seem to be acessing a yes or no type variable of the parent object. My main aim is to somehow link in the receptron so it triggers depending upon a custom quest variable (you don't have to use only the goal_state ones), as you can only use numbers on operators not other variables.

I also haven't looked at the AI things, as it might be cool if I could change this so that you are only allowed to be seen 10 times, but at the moment I can safely say that anything that sends (or emulates) a CD can now trigger an objective in steps or one go.

I also think that this can be customised for difficulty levels.

==

Don't use the Beds under " physical - Furniture - Bed " they crash DromEd,
Use only the beds under " physical - Furniture - Bedparts " they work right

==

If you load up Gatb and look at the Door11 that was used as a window, what Trim done was this:
Renderer->Alpha->0.50
Inventory->Max Pick Distance->0.50
Game->Damage Model->Hit->1

Game->Damage Model->State->Destroy
Meta Property->Materials->MatGlass
===

Tutorial: Bring Object to Place Objective.

Let's say you want an objective "Bring the Scroll to the Library".
First make the objective with:
quest_create_mis goal_state_0, 0
quest_create_mis goal_visible_0, 1
Then you'll need a bunch of stuff in your level:
1 Scroll (S) - any readable with StdScroll will do.
1 Library (L) - any room with a room brush round it.
1 Marker (M)
1 RequireAllTrap (RAT)
Now, set it up like this.
On the S add the script 'StdButton' (keeping StdScroll as well) and also Inventory>CannotDropThis.
Also, the S needs EngineFeatures>FrobInfo set with WorldAction as Move,Script.
The L wants to be a room brush with script 'TrigRoomPlayer'.
The Marker wants the script 'TrapSetQVar' and then Add>Editor>DesignNote and put '=1:goal_state_0' in the box.
Now make these links:
ControlDevice From:S To:RAT
ControlDevice From:L To:RAT
ControlDevice From:RAT To:M
Assuming you've loaded convict everything is ready to go. (You don't even need VictoryCheck on your StartingPoint for this.)
===

Water: The Basics.

First we need a hole to put the water in, so make an airbrush in the floor. Make it 16x16x8

Now we need to put some water in.

Click on the airbrush you just made. Make a copy of it by pressing 'Insert'. The copy will now be the selected brush. Change fill_air to fill_water. Reduce its height to 7 and line it up with the bottom of the air brush.

Now select Flow from the Create menu (middle of main DromEd button bar).

Make a flow brush that completely encloses the water brush. Click 'Edit flow group'. In the bottom box type 'gr' or 'bl' depending on whether you want blue or green water. You can give it current in this window if you want. (current x, y and z.) Click OK and portalize.

Now put a room brush round your room and another round the air brush that contains the water and Build the Room Database.

And that's it.

A few other things:

There are also flood brushes that make water go round solid objects placed in the water. But flood brushes are unreliable. It's much easier to make sure that you make the water before any objects you put in it.

If your water appears with a jorge texture then the most likely cause is that you have reached your 255 textures limit and there's no room for the water textures.

==

A way to have a mission load a gamesys other than dark.gam

There is. Check the list of commands, but it's something like "set_gamesys myfile" (where your gamesys is "myfile.gam" - exclude the .gam extension, it is assumed). Save the mission. Save the gamesys file in the Thief directory. In a normal mission, at the bottom of the screen, it says "dark.gam." In your modified mission, it should say "myfile.gam" instead. From that point you don't have to load the gamesys from the menu; it loads when you load the mission.
I seem to remember having trouble saving the gamesys from the menu; I started saving it using the command line (save_gamesys or something like that).
==

Double Spiral Staircase

Okay, I'm answering my own problem here, but I'm sure someone else will want a staircase like mine someday. So here's what I had to do:

Anything with more than 15 planks (apprx) crashed Dromed (this was for a 180 degree section of dual-spiral stair), so I had to keep my angle between planks OVER 10 degrees or so.

I ended up using 18 long planks, .75 high, with a 12 degree (NO LESS THAN TWELVE!) rotation on each lift section. This is kind of small, but not very, as long as you keep the planks above probably 14 long or so. It's a very pretty staircase, and I'm very proud of it, I may upload the multibrush somewhere, if anyone is interested (I don't have a site of my own), and please feel free to copy or adjust my measurements accordingly. Because the stairs are tight together with these measurements, you shouldn't have any problems with slits between beams, and any 18 wide (or so) column should be able to contain the stairs.

With my finished staircase, I'm planning on putting a flat brush at each "floor" level, with a 90-degree offset or some such, to allow for entering and leaving the tower of stairs. This is a pretty cool staircase, I must say, go ye, and try it out.

==

I created a merchant class in the store he has the frontgateguards meta property. However if I steal or even draw the sword on him he’s a happy man just whistling away.....
Question one
HOW do I make it so that when I steal or draw a weapon on him he loses his friendly attitude?
2) After he is hostile (yes hostile I dont want him running away because he can’t he’s trapped behind a counter) how do I make him push an alarm that will call several other guards...
He will react to a raised weapon, but not a drawn one. That's a limitation of AI->Responses->Threat Response.

As far as reacting when you steal an item, try this:
Add the StdButton script to all the items you can't steal. Also edit their Engine Features->FrobInfo so that WorldAction includes the Script flag.
Create a marker and add the TrapConverse script, and AI->Conversations->Conversation. Edit step 0 like this
Actor: ActorOne
Flags: None
Conversation: Action 0: Add/Remove Meta-property
Argument 1: Remove
Argument 2: M-Front Gate Guard
Now add an AIConversationActor link from the marker to the merchant. Edit the data to read "1" for actor one.
Now add a ControlDevice link from each item you can't steal to the marker.
Now, when you take an item, it triggers the ControlDevice link to the Conversation marker, which removes the M-Front Gate Guard property from the merchant.
Well, it would make them hostile next time they saw Garrett, regardless of whether they saw him take it or not.
Undercover uses custom mission scripts, I think. There are scripts called SuspiciousToFrob and SuspiciousToTake that aren't in convict.
Maybe try this:
Put a marker where each item is. Follow the steps above, but in the conversation, instead of removing the merchant's metaproperty, add an AIWatchObj link to the marker. This requires one conversation per item, and also you'll need to set up default AIWatchObj parameters for the merchant with AI->Utility->Watch: Watch link defaults and set the defaults to remove the metaproperty from him. Set it up to not kill the link.
That should make him react if he sees you take an item, or if you return to where the item was. Haven't tried exactly that, but it should work.
===
How exactly does the S->Scripts list work?
Now that we've been doing mods a lot to objects, you often need to run across this thing. How does Thief use this script list? So far, I've only tweq'ed with it but am not totally sure about it's operation. For example:
I just modified a sword which NORMALLY has frobinfo:
worldaction(move),invaction(script,focuscript),toolaction(script,focuscript)
The "move" script means "take into inventory, the script says there's a specified script to use (found in S->Script), and focuscript means to put a spinning icon of the object on the GUI. Right?
Ok, BUT! When you look at the S->Script list, it has: script0(sword), script1(), script2(), etc.
In order to add StdButton script when the sword is picked up, I had to change frobinfo worldaction to (move,script) so it calls a specified script. Then I had to *move* the "sword" script from script0 to script1, and place "StdButton" script in script0.
So, my interpretation is this: when Thief sees it needs a specified script, it pops the top of the script stack list to know which one to use. (In this case StdButton for worldaction.) This continues for invaction script with sword. But since there are no more items in the

script stack, the 3rd call of a script (with toolaction) will use the last used script, which was sword. Is that right?
But this gets really complicated as you start adding more script calls in other properties... sheesh, anyone know the whole picture on this stuff?
kfgecko, I have successfully used the StdButton script with the sword and many other objects, and I have never noticed different behavior based on which script is listed first. I always check "Don't Inherit," I wonder if that matters?
"Don't Inherit" means "Don't inherit scripts from the parent archetype".
The confusing part is that when you edit an object's script list, by default it shows the list of scripts from the parent.
For instance, if you edit a door's scripts, it shows StdDoor.
If you REPLACE StdDoor with TrigDoorOpen, and leave "Don't Inherit" UNCHECKED, the door will still work. If you CHECK "Don't Inherit", the door will not open. I've tried this.
I don't know what order they are executed in. That would be interesting to know.
==

How to add a holy sword to a level and have it replace the original sword for the rest of the level.

First off, let's make a holy sword out of a normal sword that also does not expose you when you have it drawn.

- create a regular sword from the object hierarchy

- edit the properties, click Add, and choose act/react sources

- click Add, and enter the object number of the sword, holystim, contact, and enter the value you want the holystim to have (a holy water arrow typically has 2)

- edit the properties again, click Add, and choose Weaponp->Exposure and enter 0

You now have a holy sword that doesn't affect your visibility!

However, when in game this sword will not be permanent until you make the following changes. The new sword needs to actually replace your old sword.

- from the object hierarchy, create a DestroyTrap and link it as a ControlDevice to the original sword the player starts with

- create another link, from the new sword as a ControlDevice to the DestroyTrap

- edit the properties of the new sword and add EngineFeatures->FrobInfo

- check the FrobInfo WorldAction script

- add S->Scripts; move the word "sword" from script0 to script1, add the word "StdButton" to script0

Effectively, when the player picks up the new sword, it will trigger the DestroyTrap to delete in the original sword from the players inventory.

Hope this helps everybody.

Yeah, the StdButton script is VERY useful in all sorts of situations. You can use it to trigger VOTraps when Garrett picks up an item, trigger Conversations (which can be used simply to add/remove links, metaproperties, trigger teleport traps, etc. to CHANGE THE SITUATION) basically anything that accepts a ControlDevice link.

Also, a lot of times when you create a sword in the level, it will mysteriously disappear on you - the object's bounding box is there but the sword itself is invisible. Set **Renderer->HasRefs to True** to get rid of this problem.

==

1. How I align double doors, that the other works exactly mirrored?
   Link FLAVOUR(ScriptParams) FROM(door#1) TO(door#2) DATA(Double)
2. How I give my keys own names, like basement key?
   make a key
   edit props
   go to add>inventory>object name
   type a sourcename (doesn't matter what it is), followed by a colon, a space, and then the name you want it to have in the game within quotation marks.
   for example, to have a key that shows up in your inventory as Basement Key, you would type
   MungMaster: "Basement Key"
3. How I can hide my loot or guards at other difficulty levels?
   select the object/guard or loot then edit his properties:
   then add/difficulty/destroye
   now select on which diff level you want that this object is destroyed (only use level 0,1 or 2). For example, if you want one guard which is here only on medium level,add destroye for 0 and 2,and if you want him only on easy level,add destroye for 1 and 2.

==

With Oblivion being released early next week, I thought Id share some things Ive learned about making a large level. first off there is a limit to the number of ojects and room brushes you can pack in. I dont have an exact count but if you recieve an error about "not enough concrete rooms" or " object count too high" these can be tweeked by increasing the object count in your DARK.CFG, the trick is to keep all the numbers adding up to the same max. I seem to top this editor out at about 2500 objects. also the first sign of too many objects is a crash when running into a creature that emits a FX, like burricks/bug beasts etc.

About poly counts, the main thing is to keep the depth of field short. if you have a large room, especialy with windows your going to blow out the editor. I trick is to place objects in field of view to reduce the field of depth. Another thing is to limit the amount of AI walking through the same room or space at the same time, this SLOWS down the program greatly. Object size distortion, objects can be enlarges to a degree but their overall property remains the same if left unedited, eg. dimentions frob etc. stairs, as you will see, I love stairs, the problem is these are object number hogs also they will kill poly counts, one way I kinda got around this was to not have stright stairs, at the bottom I bent the direction 90 deg to help hide the over all site of stairs, this greatly reduces the poly count.

AI, these are fun to tweak but can get way out of hand. In Oblivion I have over 200 pat. pt. and 60 guards to keep track of at any one time. Also the guards warp in and out depending on what your doing, the problem becomes what is any one guard doing at any time. my solution is to keep any markers you have for any particular guard right next to them inside a wall, this way you can keep track of each guards links.

A neat trick is to vary the guards attributes so their not all clones, e.g. change the vision of one to well above and another to below, or hearing above and below, the point is make them all a little different and the game becomes more alive. Guard behavior seems also dependent on the room brush their in, they will change speech, hearing and vision as they transition through room brushes, this can "break" a levels mood.

Sounds, these are difficult sometimes and I'm still battling them, however I hate the "snap" of an ambient on when I enter a room, this should transition. there is a fine line for too many sounds and too little, use them to enhance the mood of a certain room or area, use garrets speech to give clues and enhance a certain room or event.

Room design: its very easy to go way overboard on a room then you get the poly error or room too complex thing, I've been dancing this line. use textures to make up for room detail, for an example go look at the cathedral level with the lights on, you'll be surprised to the over all simplicity of the brushes and how the textures really fill out the buildings.

I could go on and on, level designs get better with exp. with the editor, oblivion got way out of hand, it started as a simple little level, but as we all discovered how to do things, I kept adding and adding, this created a monster that I hope has held up, if you download this level I had added a readme that says all the things I tried and had to delete or abandon due to editor limitations or max's. I hope some of these things helps some of you, I know a ramble on but oh well... if you have any more question I do my best to answer them... good luck

This 2500 object limit, is this included brushes too, or is it 2500 each?

I have well over 3000 brushes, 2500 are just for the objects

==

I don't know if any progress has been made on the fly-away door problem, but I successfully fixed two today and thought someone might benefit from knowing how and what caused them. The first was a standard spinny door that triggered an emitter trap. The cause of its flight was the TrigDoorOpen script. When it was added, the door started flying. To fix it I checked the "Do not inherit" box in the Scripts property.

The second was a FrobInert secret door, triggered by a doused flame. Since the doused flame sends a TurnOff signal, the initially closed door was told to close again. It didn't like that, so it flew away. Piping it through an inverter fixed the problem.

Some people have reported a problem where bumping a door causes it to make a sound like it just closed. I sort of found a cause for this. This particular door was a spinny door, but I wanted that big sliding door sound. Changing a door's sound is done by doing Add->Schema->Class Tags. Then type "DoorType *typename*" where *typename* is one of a handfull of possible types. Off the top of my head, there is Metal, Wood1sm, Wood2lg, sliding, Skylight, Cell, and a few others. The one that I wanted was sliding. This one and this one alone caused the sound problem. But, since many of the slidy doors use this schema, there must be something else that causes it. In other words, it is the combination of at least two factors, the sliding schema being one of them. I don't know what the other is.

==

Here is a way to teleport Garrett to new location when he enters a specific room.
First create a room by going to editors> object hierarchy. At the top where it says show tree, change this to rooms. Hilight default room and click add. Type in name of room (anything you want). Now click create, and then hilight the room brush where teleport takes place and while holding down shift press insert. With your new room hilighted, click edit then add>S>scripts. In Script 0 field type TrigRoomPlayer. Now create a teleport trap and destroy trap. Make sure you align teleport trap where you want Garrett to appear after teleporting. Note the id#'s of the room you created, teleport trap, destroy trap, and starting position. Use the following information for the links:
Room> link 1:Flavour: control device
From: room id#
To: teleport trap id#
link 2:Flavour: control device
From: room id#
To: destroy trap id#
teleport trap> Flavour: control device
From: teleport trap id#
To: starting position id#
destroy trap> Flavour: control device
From: destroy trap id#
To: teleport trap id#
Now when Garrett enters room you created, he should teleport to new location.
Note: if you want Garrett to keep teleporting everytime he enters room then don't use the destroy trap.

==

I found another method to teleport that could possibly offer more versatility, though it does involve saving your own gamesys file.

The basic gist is that you modify the archetype Avatar to emit a stimulus when you use an object. This way you can set up another object with a receptron to teleport the source (i.e. Garrett) to wherever you want.

This does work so you can set up a teleport switch, which you could even carry with you, but for some bizarre reason you can't directly give the Garrett Avatar a stimulus, as it doesn't translate into the player box object which is called when the game runs. A workaround is to give the avatar a receptron which rebounds the stimulus to its source. i.e

Teleport Switch
Holy Stim Source,
contact,on frob.
Avatar
----> Holy Stim Receptron
Stimulate Source
with holy stim 10
<-----

Receptron
Holy Stim Receptron
Teleport source
to marker.
Hmm, hope thats clear.
Pity about having to use a custom gamesys.
===
"Click the 'Editors' menu, then select 'Mision Parameters'.

Click 'Sky Rendering Mode' then 'OK'. Change the selection to 'Textures' and press 'OK'. Now click in the command window (lower right) and type 'load_sky dsky' (without the quotes)".

While this will give you a full moon, typing in 'load_sky nsky' will give you a blue sky with clouds !

===
No, I'm not back on my mission full-time yet, but I snuck in a few minutes to try and figure out my annoying fly-away door problem. The door would still fly away if I recreated it from scratch.

You know what was wrong with my door?

NOTHING. It was the key.

I added the StdButton script to the key so I could have it trigger a VOTrap when the key was taken. Problem was, I did a big NO NO even though I should have known better. When I added the script, I added StdButton to the script list and didn't check "don't inherit", so the key had TWO lock scripts. I edited the script list so that only StdButton was in it, and left "don't inherit" unchecked, so it would inherit the standard lock scripts, and now the door works fine.

That's one less problem I'll have to figure out when I get back to my mission. Hopefully it will help someone else too.

===
There is one basic cause of a fly away door that is easy to fall into and easy to fix.
When we change Properties, it is done as if the game is actually playing. So...

You make your door and set it locked. Jolly good. Now whilst testing your level you get a bit tired of shimmying into the giant Aardvark's esophagus and picking the chest lock in order to get your door key so what do you do?

You set the door to unlocked temporarily.

What happens in the game when a door is unlocked? It changes its Rotation attributes from closed to opening. Same thing happens in the editing. Hence when you Alt G, the door opens automatically.

Then, when you are ready to test your level with the door locked you simply set it to locked with the Properties. Then just as in during gameplay, THE DOOR IS AUTOMATICALLY SET TO CLOSING!!!

So when you Alt G you have a locked door that is in the middle of closing.

Bye,bye door.

Solution then is simple; When you have re-locked your door. Change the setting from 'closing' to 'closed'.

It cured all my door problems. In general, I think all fly away doors involve something similar.

Hope this is of use.

==

Actually, your root problem of temporarily unlocking the door has an easy circumvention.
Just create a master key in a blue room. The master key opens everything. Link it to the starting point. Before releasing your final version of the mission, just unlink the master key from the StartingPoint or set the property difficulty->destroy{0,1,2} to destroy the key for all difficulty settings.

==

So, it turns out that the *Flight Unlimited II* adventure builder includes the programs 3DS2E and BSP, which are the tools used to convert a .3DS (3D Studio) polygon model file into the .BIN format used by *Flight Unlimited II*.
Not coincidentally, the same tools are used by *Thief*. Looks like these versions of 3DS2E and BSP are, in fact, compatible with *Thief*. Just in case anybody wanted to import their own .3DS models.
The *Flight Unlimited II* adventure builder is a free download, but requires version 1.04 of *Flight Unlimited II* to install successfully.

Air -> Solid, Solid -> Air, Flood, Evaporate
I exclusively use "Solid" and "Air" brushes. I noticed there are also "Air -> Solid" and "Solid -> Air". What are they for?

- Air->Solid turns air into solid... and has no effect on water. As opposed to "Solid," which turns everything into solid.
- Solid->Air turns solid into air with no effect on water. As opposed to “Air,” which turns everything into air.

Think of them as like the "flood" operation, if the flood operation were named "Air->Water" instead.

- The “Flood” operation fills the air with water (not affecting solid); “air->water”.
- The “Evaporate” operation fills the water with air (not affecting solid); “water->air”.

Example:

1. Use a wedge brush with Flood. Now you’ve got some water floating in mid-air shaped like a wedge.
2. Intersecting the wedge, use a pyramid brush with “water->solid”. It will change part of the wedge to solid, leaving the remainder as water. Any part of the pyramid that extends beyond the wedge brush will remain as air (it won’t change air to solid).
3. Clone the wedge in step #1 and change the operation to “evaporate”. It will change the residual water back to air leaving just the odd shaped solid created by step #2.

It’s a very cool way to use logical AND, OR, XOR to get odd-shaped solids without worrying about overflowing into other regions!
WHAT IS “BLOCKABLE”?

==

I've been working on a way for a lockbox to control an elevator.

When you lock the box, the elevator moves to one floor, and when you unlock the box, it moves to the other floor. Well, I finally figured out how. You can, of course, make the lockbox frob anything you like with this technique, so I'm posting it.

Link a door in a blue room to the lockbox with flavour(lock). Add S->Scripts{TrigDoorOpen} to the door and delete the other scripts, be sure to uncheck the "Don't Inherit" checkbox. Link FLAVOUR(ControlDevice) FROM(door) TO(tpath1) (i.e., the terrain point for the unlocked elecvator position). Add an inverter, and link FLAVOUR(ControlDevice) FROM(inverter) TO(tpath2) (i.e., the terrain point for the elevator locked position). Add link FLAVOUR(ControlDevice) FROM(door) TO(inverter). When the lockbox is unlocked, the blue room door opens, which sends a TurnOn signal to tpath1 and sends the elevator to tpath1. When the lockbox is locked, the blue room door shuts, which sends a TurnOff signal to the inverter. The inverter sends a TurnOn signal to tpath2 and sends the elevator to tpath2. You also may be able to use s->scripts{TrigUnlock} on the Lockbox, rather than a hidden door, to generate the ControlDevice signals; use an inverter to one of the terrain points.

The remainder of this section is useless:

In a blue room, create a reallycrushingwall. Change it's movement speed to 5, and it's open position to seven. Change it's dimensions to 9.5, 4.5, and 1 (Obviously these numbers aren't written in stone, but this is what I've got and it works).

Next, create a flame (under SFX: FireFX) as close to the wall as it can be without actually firestimming it, such that the wall will cross through it when it moves.

Next, create your lockbox, and put a lock link from the wall to the box. Now, when the box is activated, the wall will move and be burned by the fire. Naturally, it will also be stimmed again when the box is unlocked.

Open up the crushing wall's properties and give it an act/react: receptron property. The receptron will be a firestim with a min strength of zero, the "No Max" box checked, with "frob object" selected as it's effect. The Target field should contain the object number of the up/down switch which is linked to your elevator.

Voila! There is a noticable delay, of course, between the time when the key is used and the lift begins to move, as the wall is not burned immediately. This suits my needs perfectly, as a delay is desired for what I'm doing.

You can adjust the timing by doing things like altering the size of the crushing wall, changing its speed, etc. The important thing is that it move slow enough to get burned, but fast enough to only get burned once.

Now just about anything frobbable can require a key for activation if you so desire.

Revision to the above.

In playtesting, the Crushing Wall in question turned out a bit unreliable due to the necessity of very specific tweaking. To get the whole setup to work reliably, use the following:

Crushing Wall dimensions: 9.5, 3.5, and 1

Door: Translating: Closed Position-0 Open Position- 6 Base Speed-9 Axis-Y

Assuming the Crushing Wall is located at 0,0,0, you'll want your flame at 0,3,0.

Now it should work with precision.

Revision to the above.

Okay, honestly I don't even know if anyone got any use out of this tidbit, but the fact is, it turned out to be ultimately too unreliable. It worked probably 90-95% of the time, which is of course unacceptable.

Didn't get much feedback last time, but good conscience won't let me leave that sloppy mess in the archives unfixed.

The following is a new, improved, consistent, and simpler method:

Create an LC_bridge and change it's size and physics dimensions to something small and manageable.

Place a skull on it. Surround the skull, except above and below, with solid brushes in the shape of a long rectangle. The skull thus sits in a sort of "box", resting on the bridge. Make sure that the side walls are close enough to keep the skull from rattling as it slides along when the bridge moves.

Sounds bizarre, but it's really quite simple. The idea is that when the bridge moves, the skull slides along a couple feet until it "bashes" into the opposite wall of the box. Helps if you increase the bridge's movement speed.

Now, add an act/react:receptron property to the skull. Min is zero, and check the "No Max" box. The stim is a bashstim, of course, with "frob object" as the effect. The target object can be an elevator, a set of lights, or whatever.

The above setup should live in a blue room, naturally.

Go through the usual procedures for making the bridge into a door with it's lock contained in a lockbox.

This arrangement has worked consistently throughout quite a bit of playtesting, and seems perfectly reliable. The one issue I did run into was that occasionally the skull would fall through the bridge. I remedied this by making the bridge a little thicker and increasing it's mass. I'm not sure which of these actions fixed the problem, but it works 100% of the time, now.

==

Using a LockBox to send a ControlDevice signal

Create a simple door in a blue room, add script TrigDoorOpen and delete the other scripts, be sure the "Do Not Inherit" flag is unchecked. Link: From(Door) To(LockBox) Flavour(LOCK). When the lockbox is unlocked, it will open the door. The door sends a TurnOn signal to whatever you want. When the lockbox is locked, the door is shut and sends a TurnOff signal. A s->scripts{TrigUnlock} may also work; haven't tried it on a lockbox.

==

SpotStreetLamp adjustment

When placing a SpotStreetLamp, change the dimensions under Physics:
Offset: -1.75, 0, -1
Size: 1, 1, 12
This allows Garrett and AI to walk near the SpotStreetLamp. Otherwise, the Bounding Box keeps away Garrett and AI.

==

how do you change the skin?

OK, I'll take you through the process of replacing a skin. First you need to know the name of the skin the model has; you could look for the skin in Thief Media Edit or you could edit the .bin file and look for xxxx.gif. Have your skin, convert it's palette to Thief's, rename the skin to the one you're replacing.
To find the appropriate directory, remember what .crf file it was in.
All creature skins are in mesh\txt, so create these subdirs within Thief's dir.
Place your skin in that directory.
So now you'd have your skin let's say 1.gif in the following directory;
...Thief\mesh\txt\1.gif
I don't think you have to bother with a txt16 version of your skin.

==

those who want to make or import models into Thief might find these places useful

http://www.3dcafe.com/
http://avalon.viewpoint.com/

Ambient Hacked Sound

Create an ambientSound object. Select properties and add “ambientHacked”:

1. Radius in feet, usually from 5 to 30.
2. Schema
   - m01start
   - m02tapping - ticking clock
   - m02wind
   - m02people1
   - m02rafters
   - m02vox
   - m02tiles
   - m02watercalm
   - m03wind
   - m03mines
   - m03zaparc
   - m03trans
   - m04holywater
   - m05fb - zombie/eerie
   - m05nightlp
   - m05rising - guitar anxious
   - m05sqlo - squeal low
   - m05tavern1, 2, 3
   - m05tunnel - wind howling low
   - m05water
   - m05watercalm
   - m05wind
   - m05windtr
   - m06nightlp
   - m06forest
   - m06trans
   - m06water
   - m14cicadas
3. Aux is usually omitted
   - m02basement2
   - m02cue1
   - m02cue2
   - m03birds
   - m03mines_var
   - m05l1 - guitar base tempo
   - m05l3 - stalking whistle
   - m05bellthump

==

I have noticed something about metaproperties.

If you already know what metaproperties are, skip to the next paragraph. Metaproperties are groups of properties that can be added or removed from objects either in edit mode or in game mode. One example is the M-Front Gate Guard metaproperty which is on the three guards at Bafford's gate. The metaproperty alters their reaction to the player so they don't attack. If you try to go in the front gate, this metaproperty is removed and they attack as normally. You can define your own new metaproperties, but it requires saving the gamesys. They can be edited under the object tree if you switch the select box from Archetypes to MetaProperties.
The issue I found is that metaproperties can override properties of an object's archetype, but not specific properties added to the concrete object. For instance, the ServantWithTray metaproperty changes the model of a servant to one that includes a tray, and also changes its motions so the hand holding the tray is extended. However, if you start with an AI other than the male servant, and change its model to the male servant's model, then the metaproperty, when added, will not change the model.
So, if you have a customized object, and you need to add metaproperties that will override properties you have added to the object, you will need to make the object an archetype and save the gamesys.
==

Dromed does not allow you to create new schemas.
I'm pretty sure schemas are stored in the gamesys because I've listed out dark.gam, all the exe's, dll's and osm's, and every other file just to be sure, and dark.gam is the only file that contains the schema names AND the names of the wav files. But you can't edit that in Dromed, as far as I know.
It would be REALLY nice (ahem, LGS) if the tool used to define schemas were made available, or if it's too tied in with proprietary stuff maybe release enough information about the format of dark.gam that someone could write a tool to do this. (I volunteer.)
Well, I just used a file monitor utility to watch what Dromed does when some commands related to schemas are run.
When "reload_schemas" is run, Dromed looks in a SCHEMA subdirectory under the main Thief directory for *.SPC, *.ARC, and *.SCH.
I don't know exactly what goes in these files but I think they are text files defining schemas. Any other hacker types want to help out from here?
Some commands related to schemas:
destroy_schemas : Destroy all schemas
build_motion_database : read in motion schemas as build database
reload_schemas : Load all schemas in path (don't destroy)
zggtvrk_load_schema : Load a schema file
zggtvrk_load_schemas : Load all schemas in path (destroy first)
==

A FlickerTrig cycles off and on. It could be used, for instance, to make a light go off and on at regular intervals.
The one I used in my mission teleports a marker to Garrett at regular intervals, so AI can face the marker. I could find no other way to make AI face the player. **(SEE MURUS.)**
==

How does Bro. Murus know to turn and watch Garrett?

After I beat my head against the wall to find a way to do this, and even modified the Garrett archetype so I could teleport a marker to him for AI to face, the answer is simple.

Put the string "Player" as the SECOND argument to a Face action.

Not the FIRST argument, which works with every other object.

For the example, load up RTC (miss11.mis) and type in the command box "find_obj 853" (Brother Murus id.) He has an AIWatchObj link to himself watching for player intrusion that simply faces the player.

==

One of the things Tweq can be used for is to make an object spin.
Such as a sword, for example.
Tweq->RotateState->AnimS = On
Tweq->Rotate->Halt = continue
AnimC = NoLimit
Primary axis = 3
Z-rate-low-high
X = 75
Y = 0
Z = 75
This will get you an object that spins on the Z (vertical) axis.

==

Making New Objectives

Wasn't this supposed to be impossible? I dunno that, but it is possible, that's for sure.

Warning: This tutorial is written in a place where I do not have DromEd, thus it is vague, try to read between the lines.

As I was making GatB ver1.1, I almost accidentaly run into a way of making new objectives. Thrilled with this new invetion, I started to hide current objectives only for the purpose of making the "new objectives" pop up. Needless to say, I was on completely wrong tracks, since you don't use it because you can, you use it where you need it. Keep that in mind. Oh, just for a note, there are no new objectives in GatB since they plain and simple didn't feel right.

On with the tutorial.

Make your first goal invisible:

quest_create_miss goal_visible_0, 0

Make a room and such, and a guard. Make the following act/react values:

Receptron

weapon

min 0

no max

Change mission objectives (?)

details->

goal_visible_0

set to

1

Now when you hit the guard with any weapon, a New Objectives pops up.

But what if you want new objectives to pop up when you enter a room? There is no "frob" -stimulus on act/react so that pretty much eliminates all the normal methods.

There HAS to be a better way of pulling this off, but for a reasons unknown, I didn't figure it out. Didn't think very long, but still.

Prepare for a strange thing:

Put a script "TrigRoomPlayer" on the room where you want to trigger the new objectives. Now link the room to a broadhead arrow with a control device. Huhh?

Let's go back a bit. Create a BlueRoom (Room outside the world, a place for odd stuff). Now add a door and a broadhead arrow there. Make the arrow point the door (for artistic reasons) and then click on the properties of the arrow.

Add script: StdDoor

Door: Translating

FrobInfo: Script, none, none

Now make it so that when you use the arrow, it moves towards the door. It has to go all the way, hit the door. Just change the values of the "door->translating" and it should work.

Now check the properties of the door. The real door. Add act/react and do the same things done with the guard, but stimulus is "poke". Now when the arrow hits the door, it activates the act-react, and that activates the new objectives.

Backing up a bit, you remember that you had linked the room to the broadhead with controldevice. So now when you enter the room, the broadhead is activated, it moves towards the door, thus poking it and activating the new objectives. Simple, eh?

You can also change the goal_state_n where

0: Uncomplete

1: Complete

2: No valid anymore

3: Failed

Adds a whole new world to explore.

Hope you understood even half of what I said.

Yeah, the tutorial says you can't do this, but actually you can.

You actually want "Set Quest Variable" not "Change Mission Objectives (?)" on that receptron. You can also set quest variables using trap scripts, as I mentioned in another thread.

I want to resurrect this thread for those who may not have seen it (or understood) the first time. This is an incredibly useful tool. I am using a similar idea to create an objective of type "bring object x to location y." Though we are normally limited to 4 types of objectives (Kill a creature, steal an object, go to an area, steal x amount of loot) I think you can find ways of creating much more varied objectives using this technique.

Note that for my purposes, I replace

goal_visible_0

set to

1

with

goal_state_0

set to

1

which checks off the goal in question. There are any number of possibilities.

<< You can also set quest variables using trap scripts, as I mentioned in another thread. >>

EvilSpirit (or anyone else), can you give more detail on using a QuestVarTrap to change quest variables? I can create the QuestVarTrap and have it be triggered by something, but I don't know how to tell the QuestVarTrap what variable to change.

I figure you have to set the Trap->QuestVar property, but I don't know what to set this property to. I've tried things like "goal_state_0, 1" "goal_state_0 = 1" etc. with no success.

Here's what EvilSpirit had to say:

EvilSpirit

Member posted May 28, 1999 11:56 AM

Actually, you might be able to expose, hide, and score mission goals using traps. Since the mission goal state is held in quest variables, you ought to be able to use the little-if-ever-used but never-removed-from-the-scripts Quest Variable Trap to do these things.

I'm going to assume that you've figured out basically how to use triggers and traps. In a nutshell, certain scripts cause objects to send "TurnOn" and "TurnOff" messages across ControlDevice links, and certain scripts have set responses to these messages. Levers and doors, for example, have trigger and trap behaviors in their default scripts, which is why you can link a lever to a door and have it control the door.

To set up a quest variable trap, just put the TrapSetQVar script on any object; a marker will do. Then write a short text command to describe the quest variable change you want to make. Since we never went and supported this "for real," the trap uses the text of the DesignNote property (ordinarily a property intended to let you put comments on objects).

Here's the comment from the TrapSetQVar script that describes the two kinds of quest variable changes I ever wrote:

```
/*
The Quest Var trap performs some operation on a quest variable
as specified by an "operator" string property.
The format of the quest var operator is :
the op is a single character indicating the operation to apply
to the quest var of the specified name. The argument is a single
non-negative integer argument which us used if the op is a
binary operator. Currently-supported ops are:
Let x be the named quest variable, and a be the argument:

Op | Effect of TurnOn | Effect of TurnOff
 = | x=a             | none
 ! | x|=a            | x&=(~a)

Obvious possibilities to be implemented are x+=a, x-=a, etc.
*/
```

So, to make goal 3 visible, you'd use the command "=1:goal_visible_3", or you could use "!1:goal_visible_3" and turn it on with a TurnOn and then off with a TurnOff.

No guarantee that this works, mind you. But it seems like it ought to.

This works! Note that if you use the "=1:goal_state_0" format there must not be a space after the ":".

can you make the change timed so that it occurs after a conversation? Like in Cragscleft where the objectives didn't change until after Cutty died. If so, how?

At the end of the conversation, last stage, have an actor frob a switch that has ControlDevice links to one or more QuestVarTraps.

==

QuestVarTraps.

Basically, create the objective with only a visibility and state. The type and target are useless. Create a QuestVarTrap (under fnord->TrapTrig in the object hierarchy). In its properties, add->Editor->design note. Type "=1:goal_state_*n*" without the quotes, where *n* is the number of the goal in question. This says that when this trap is triggered, the quest variable goal_state_*n* will be set to 1 (complete).

Now you link your three items to this trap as ControlDevices. Give them the StdButton script and add Script to the World section of the FrobInfo. That should do it.

One problem. This doesn't work in Thief Gold. Looks like they fixed the QuestVarTrap so that it uses its own field instead of the Design Note. So instead of adding the design note, add->Trap->QuestVar and type "=1:goal_state_*n* there. But then it'll probably crash original Thief. Try it anyway. Use both the design note and the QuestVar and it just might work in both if we're lucky. I'll have to reinstall original Thief and try it sometime.

==

The new objectives are actually pretty simple, mokkis. ThiefGold and Thief2.

1. Create the objective as you normally would using the quest_create_mis commands, only make it invisible (quest_create_mis goal_visible_3, 0)
2. Create a QuestVarTrap (found under fnord->TrapTrig in the heirarchy). Add the property Trap->QuestVar and set it to something like this:
   =1:goal_visible_3
   ...when the trap is triggered, this will set the goal_visible_3 variable to 1, making goal 3 visible, and it will appear to the player as a new objective with accompanying sound and whatnot.
3. Create something as a ControlDevice to that QuestVarTrap. Let's say you want the new objective to show up when you read a book. Give the book the StdButton script, and make it a ControlDevice for your QuestVar Trap. You're done.

Note that, if you make the book the kind that the player can pick up, then the new objective is triggered when the player *picks up* the book, rather than when they read it. For this reason you might want to make it a non-pickupable book, if possible.

==

I think guards noticing doused torches was intentionally left out of Thief.

If you *wanted* to implement it, you could do it like this: give the torch flame (not the torch itself) a TrigSlain script; link it as a ControlDevice to a CreateTrap that creates an AIWatchObj marker that causes an AI to react in some way. It has been discussed elsewhere on this forum whether you could have the AI relight the torch; I'm not sure if anybody ever figured out how to do it.

==

Very cool use of Water -> Solid and Evaporate in Cragscleft

If you know what I'm talking about, move along . This concerns the popular, intricately shadowed cellblock areas.

A cube above the prison bars creates the lights on the floor and walls, with pyramids carved out of the top and bottom of the cube. A light (spot) is placed where the two pyramids overlap. That single light creates the light shape above and below the fixture. The shape that is created on the floor is directly related (obviously) to the number of sides of the lower pyramid, and the angles at which they're set. So if you can imagine how the light will be "projected", it'll probably make creating a scene like that a lot easier.

That alone is interesting, but for anyone who's been wondering how the more unusual brush operations are used, such as me (especially since they're often found nowhere near any water in the original missions), this is really good to know.
The light fixture on the wall above the prison bars cannot be created with just a solid cube and two pyramids, **because the pyramids would cut into the solid that is beyond the air brush that creates the cellblock corridor**. (Make sure you know what I mean by that.) It might mess up the lighting and would look sloppy if examined. So this is how it's done:

- Create the main corridor with a **Fill Air** brush (time 1)
- Put a 1x1x1 **Fill Water** brush where you want the fixture (time 2)
- Create two pyramids with **Evaporate** and place them in the cube of water. Since Evaporate turns water into air, you now have a bit of water with two shapes carved out of it. However, the Evaporate brush has no effect on the wall of solid next to it. Now the water must be changed to solid. (times 3 and 4 for the pyramids)
- Put a 1x1x1 **Water -> Solid** brush in the exact same spot as the Fill Water brush. That only turns the *remaining water* into solid! (time 5)

All of this could probably be done without the water-related Op brushes (by patching the hole in the wall with a Fill Solid), but it's not at all as clever . And there are probably situations where those Ops *must* be used to get a desired effect, so it's good to know how to use them.
I hope this was helpful to some people. If you already knew it all, don't take it as condescension, I'm relatively new to this, and found it extremely interesting
==

How tweq moves the dial
By Steven Hindley.

open properties of transformer1
add Tweq/JointsState
AnimS = on
Joint1AnimS = on
add Tweq/Joints
Halt = continue
AnimC = OneBounce
CurveC = JitterLow,JitterHi
PrimaryJoint = 1
Joint1AnimC = OneBounce
Joint1CurveC = JitterLow,JitterHi
rate-low-high = 4 0 200 (use any numbers you like 4=rate 0=low point on dial 200=high point on dial)
add Script
TweqOnOff;;;;FALSE
then add a marker with a ambientsound tapping sound added to make it sound like its working. Now your transformer looks like its working. Got this information from the transformer in the cathedral's basement from the thief original missions

==

http://www.ridge7.demon.co.uk/thief/srctut.htm

Sources and Receptrons, for teleporting and timing.

Aim of Tutorial

This tutorial is intended to teach the basic of sources and receptrons, and to show what a nifty tool they can be. As well as showing basic implementation, it will also describe how to set up a system to teleport Garrett from place to place, without using TrigRoom Scripts

Assumed Knowledge

Though this tutorial is aimed at people new to Sources and Receptrons, it is assumed that the reader has basic Dromed knowledge, and knows how to set up a functional level with Convict loaded and starting points, as well as being able to navigate around Dromed reasonably well. Like all tutorials, do the one in the Dromed download first!

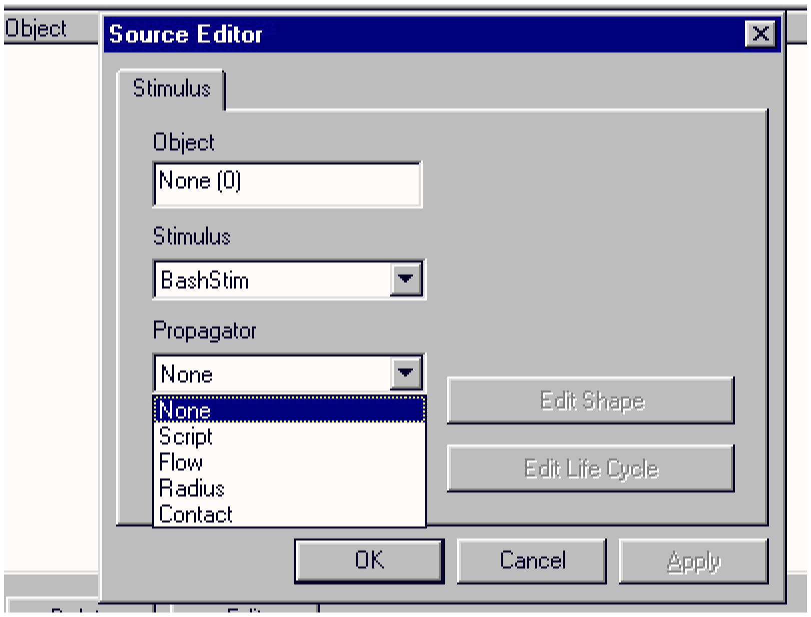
Brief Overview of Sources and Receptrons

To my mind, sources and receptrons are wonderful, as they allow the formation of complex mock scripts. Most of the puzzles that I have encountered I have solved using Sources and Receptrons.

In case you haven't used sources and receptrons before, I shall attempt to explain how to use them.

A source and receptron system is affectively a way to turn objects into weird 'keys'. One object is the source. It can be assigned a certain stimulus, either, HolyStim, FireStim, PokeStim, LightBright, WaterStim, etc. Another object is the receptron, it is set up to look for a certain type of stim, and then to do an interface with the game world in some way upon receiving that stim. An example this is of a Holy Water arrow hurting a zombie. The holy water arrow has a holy stimulus on it, and the zombie a receptron for that holy stim. When the holy arrow hits the zombie, the receptron triggers, and injures the zombie. Indeed, you can assign objects a holy stim and that object will then injure zombies like a holy arrow.

The actual situation is slightly more complicated than that, there are many different types of options. Each source has a specific strength, and each receptron looks between a range of strengths, and there are various types of 'links' that you can set.

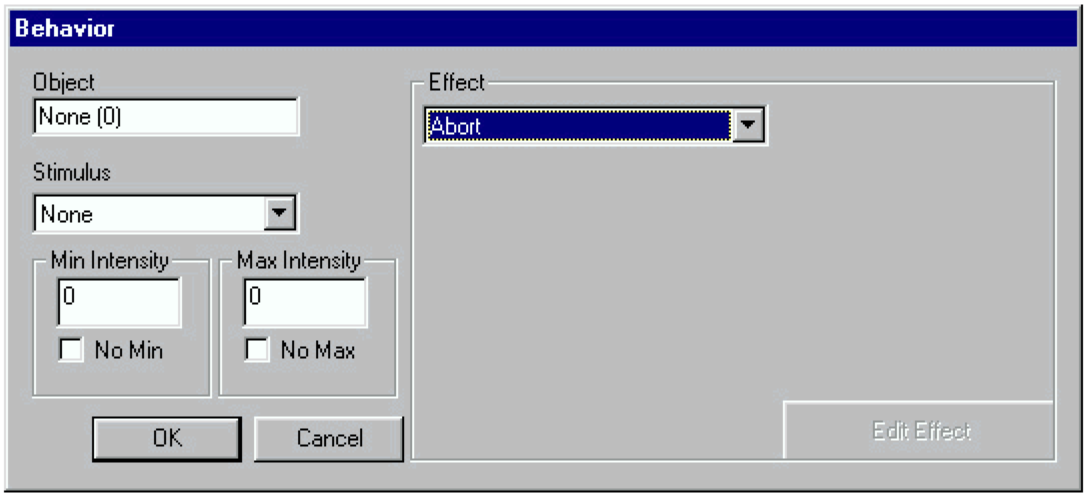


The screenshot above shows the source editor. To get this, in dromed, click on the properties of an object, then Add->Act/React->Sources,and then Add. Select the stimulus you wish to assign, and then select the propagator. This defines what triggers the link. I use either contact or radius. I am unsure what flow does, and script will probably involve some hardcoding so is useless for our purposes.

Radius is used for affects you want to trigger on a regular basis, or when someone steps within the area of an object. Contact is for when an object is either frobed, or collided with. If you select script then both Edit Shape and Edit Lifecycle will be activated. Edit Shape determines the radius of the source, and Lifecycle how regularly and how long the source 'fires' and allows any receptrons within the radius to trigger. If you select contact, then only Edit Shape will be available. You can then change the Collision Type. The default setting is collision, which triggers when object collide though you can activate any combination of frobs or weapon motions that you want. The final option is for intensity, which is simply a scale of how strong a source is.

One important thing to remember about both the source and recpetron window is that are a few slight bugs inherit with them. The first time you open up either window, you MUST add any source or receptron, as if you don't it will CRASH. Just add any old thing, making sure that the object setting is valid, and then exit the dialogue box as usual. The source\receptron window should re-appear twice, and if after the first time the source\receptron disappears, then you will have to add another. Once this has been done, you can safely delete the false source\receptron without worry.

For receptrons, you have to specify which source it is to check and between what range of intensities it wants to trigger from, and finally what to do when it encounters that source.


Behavior

Object: None (0)

Stimulus: None

Min Intensity: 0 ☐ No Min

Max Intensity: 0 ☐ No Max

Effect: Abort

There is a multitude of affects. You can set it up to teleport objects, clone objects, frob objects, add meta-properties, remove meta-properties, kill monsters, stimulate other objects, etc, and so you can see it is very versatile. One of the beauties is that you do not have to specify which specific object you want to target, as there are usually the options of either Me or Target. Setting me for teleport object means that you want to teleport the object that you are currently editing, and target means that you want to target the object where the source came from. Explore on what the possible effects are.

As you only have a limited amount of standard stimuli to use, you could use a system of giving each different set a separate stimulus. So Holy source A would have an intensity of 4 and Holy source B an intensity of 10, and then set the receptron so that if it receives a stim between 3 and 5 it heals Garrett, if it receives 9 or more it kills him. One other thing to watch is that on the receptron screen itself the max setting will always match the min setting. This is simply a slight bug, and the actual settings you specify when editting a receptron will work as they say.

This method is viable, but remember to choose a stimulus that will not be used in the course of the game normally. Do not choose HolyStim when your level will be crawling with zombies!

Teleporting Garrett

An example of a use of receptrons is to teleport Garrett.

The way I am going to describe involves making your own Gamesys file as it will edit your archetype settings. The Gamesys file is where Thief stores all the object's attributes, and as there was no need to change them under the normal play of the game, they all existed in one file, dark.gam. Whenever you modify an archetype, via Object Hierachy or by accidentally editting the properties of one, you are changing the gamesys, and if you don't save your own file they won't be remembered next time you load up.

First, create two rooms not connected to each other. Texture them up, and create the normal starting point in one (loading convict and other such essentials). Then, add a BigFloorLever to both rooms. (Physical->Gizmos->Switches->BigFloorLever), and name the one in the starting room Teleport1 and the other Teleport2. Now create two new markers in each room, one StartMark, and the other TargetMark.

The idea is that when you pull the first lever you will teleport to target mark, and when you press the other lever you will teleport to StartMark.

Now, we can't simply add a source to Garrett, as, you've guessed it, there isn't one (yet). We instead have to add a source to the Avatar, which is the template for Garrett. This template is what the PlayerFactory link calls when you first enter the level. It takes the template and creates a player box object within the game itself. We can't access this playerbox object, as it doesn't exist until the game actually runs so we have to modify the archetype. Before we do this, we better create our own custom gamesys.

Open up the Object Hierachy, and go to the Act/React section. Click on stimulus, and then add, and type in what you want to call your stim. I use TotalityStim. We now have created our own stimulus, but it won't be saved as part of your level normally, you have to manually do it by saving a gamesys file. Under file save choose file, save gamesys, and type in an intelligent name and save it in your thief root directory, I used 'actreact'. Now you have to set your level up to use your custom gamesys. In the command window type 'set_gamesys' followed by the name you typed. If you did it correctly, the bit down the bottom saying DARK.GAM should change to ACTREACT.GAM, or the name of your gamesys file. Save your mission normally, and change the name back to your level name. If you want to distribute this level you will have to also send that gamesys file.

Now that we have our custom stimulus, we now need to add the source to Avatar Garret. Under object properites edit, Physical->Avatar->Garrett. For some reason, we run into a problem here, as we can't give Garrett sources only receptrons. If you try to add a source, it is not transferred to the playerbox object, but receptrons arel. Scroll to where it says receptrons, and then edit them. Click on any of the three receptrons, and then press add to carry the settings across, make sure 'no max' is selected and 'no min' is not selected. Change the stimulus to the custom one, TotalityStim, and change the effect to Stimulate Object, select 'Source' for the first option and 'me'for the second, and then click Edit Effect. Type in the name of the custom stimulus under stimulus, and then type 10 under 'Then Add', then ok out.

So, now when a TotalityStim stimulates Garret he will stimulate that same object back with TotalityStim level 10. Now, as we have just edited an archetype we will need to RE-SAVE the gamesys

Next, select the lever in the start room, and give it a TotalityStim Source, (Add Act/React, sources). You will notice that you cannot specify a propagator. Resist the urge to panic, this happens as thief has yet to recognise your source as a proper one. Save your level, then reload it. This should reset dromed so you can use your source properly. Go back to the start lever, and edit the source. Change the propagator to contact, specifcy any non-zero intensity, say 4, and then edit the shape, click on the button with 'collision' on, de-select 'collision' and select 'FrobinWorld'. This means it will trigger when anything frobs it.

Then exit the source window, and add a receptron. Set it up for TotalityStim, set the min intesity to 9 and select no max. For effect, select teleport, in target object select the Source option, and under agent type in TargetMark. If we wanted we could adjust EditEffect to give us a displacement off the marker and a set heading once we teleport, but for this there is no real need to adjust these settings.

Go into game mode, when you pull the lever you should teleport to the other room. To get around the fact that you can't assign a source to Garret when you pull the lever, the lever stimulates you, and then you stimulate the lever which teleports you to the marker. To link up the other lever, add the source as before, and a receptron with the same details as before, but this time select the other marker, TargStart.

You should now be able to jump between the two at will. See the possibilities, you don't even need to use markers, you could use dropable objects, and switches to do this on the fly. For instance, in the level that comes with this bit, go to the table and pickup the scepter and the switch. Go around a bit and drop the scepter somewhere (usually by pressing y) and then teleport to the other room and then press the switch in your inventory. Guess what happens? Your own teleport button! You should be able to figure out how this works yourself now.

==

Conversations Tutorial

By Deep Qantas
(aka Teemu Salmela)

IV. An incentive for conversations

First, you should make an airbrush, a room around it and maybe add a torch too. Load the ConVict script and compute pathfinding and build room database. Just do all the stuff you usually do in the beginning.

After making this environment, you can add two guards, but make sure, that Garrett won't be visible to them. For example, you can put the torch so, that it's too dark for them to see you.

Edit the first ones properties and name him Tom. And of course the other one is Jerry. Change Tom's properties and add **Speech->Voice**. The text should be **'guards'**, but we have to change it to **'guard3'**. Jerry's voice must be changed too, but to **'guard2'**. It's important that the voices are right. Otherwise the conversation won't work. Tom is **'guard3'** and Jerry **'guard2'**.

Now we have two actors, but we need the actual conversation. Create a marker and name it 'BearConv' or 'MyOwnConv' or anything you like. I shall call it 'BearConv'. It doesn't matter where the marker is, but it would be wise to keep it somewhere near the actors.

Then we need to change the marker's properties. Add **S->Scripts** and put **'TrapConverse'** in the first line. Add **AI->Conversations->SaveConversation** and check the box. Add **AI->Conversations->Conversation** and just click 'OK' twice. Don't get confused by the big dialog box. It's somewhat large, but you can manage with 1024x768 or even 800x600 screen. The 'OK' is in the lower left corner. And pushing enter usually means 'OK'

Now we have actors and the conversation. We just need to connect them, so link 'BearConv' to 'Tom' with **'AIConversationActor'** and change the data to **'1'**. Link it to 'Jerry' as well but change the data to **'2'**. The numbers mean the number of the actor.

The conversation is working, but there is one flaw. The actors don't say anything yet, so we'll now edit the lines. Open up the properties of the 'BearConv' and edit that **AI->Conversations->Conversation**. Now you should see a window with 11 steps. The steps are executed in order and the next step doesn't start before the last one is finished. There are also these 'Abort' things, but we'll see to them later.

Double-click the '00' and the big dialog box appears on your screen. As you can see, there are 6 smaller steps, or actions, if you prefer. These are also for motions and pushing buttons, but as for now, we just use them for speech. Let's just edit the upper part of this box. Change the **'Actor'** to **'ActorOne'**, which means Tom and **'Conversation:Action 0'** to **'Play sound/motion'**. In the first argument, enter **'sg_c01'** and in the second argument**, 'LineNo 1'**. That **'sg_c01'** is the name of the conversation and **'LineNo 1'** is obviously the line number. Don't worry about these yet.

When you press 'OK', you're back at the properties. Open the conversation steps again and now choose '01'. Change **'Actor'** to **'ActorTwo'**, Tom, and use the same action and first argument, but the second argument should be **'LineNo 2'**. Just edit the '02' with **'ActorOne'** and **'LineNo 3'** and then we'll try how it sounds in game.

Before we are able to test it, we must make a ControlDevice. In this case, a switch or a button will do. Create a switch or a button and link it with **'ControlDevice'** to the 'BearConv'. Locate it somewhere near Garrett, so there isn't any trouble with the guards. Now just check portalizations and all those things one more time and test it.

If everything went fine, you should now have a three-line conversation that starts when you press the button. Not too great, eh? Well, this was just a demonstration. Now I have to explain all of this (sigh).

V. Advanced conversations

Let's start with the voices. As you remember, Tom's voice was **'guard3'** and Jerry's **'guard2'**. You might think 'Why on Earth can't I use some other voices?'. This is because the conversations are made for specific actors. The conversation made in demonstration is the same you can hear in front of Bafford's, and the voices of these actors are 'guard3' and 'guard2'. The freedom of choice is limited to using built-in conversations. Of course, this doesn't limit our creativity. You can take a part from here and a bit from there. If you like, you can finish that conversation by adding more lines starting from '03'.

Then there is this **'sg_c01'** and **'LineNo'**. That 'sg' is for 'sergeant' or something like that. There are also 'hm' and 'sv' for hammers and servants. 'c01' is for 'conversation 1'. I guess you could search the Snd.crf file for all the conversation, but I prefer studying the game levels. For example, another conversation **'sg_c12'** has also two actors, **'guard1'** and **'guard2'**. **'LineNo'** is of course the line number, but you'll have to be careful with this too. **'LineNo 1'** means guard 1 and for example **'LineNo 4'** means guard 2. THAT LAST STATEMENT IS INCORRECT. The “Line No” value is used in determining the “wav” file. See below for encoding technique.

Next thing is this marker. It's preferable to name it, so you can easily find it later. Those **'AIConversationActor'** links are quite self-explanatory. You just have to be careful with those numbers in 'Data'. If you forget them or they go wrong… Actor one should obviously be the one saying line one.

Now let's try something new. Open the conversation part **'01'** and change the second action, 'Action 1' to the following…

Actor: **ActorTwo**
Conversation: Action 1: **Play sound/motion**
Argument 3: **'Conversation 0, Baz 0'**.

Now try it out. Cool, eh? That **'Baz 0'** might confuse you, but don't worry, you're not the only one confused. Here's something else to try out…

"Conversation 0ZXY Quux 0"	Cross arms behind back
"Conversation 0ZXY Baz 0"	Jump and shake arms a little as if agitated
"Conversation 0ZXY Baz 0ZXY Quux 0"	Raise arms as if to say I don't know
"Conversation 0ZXY Bar 0"	Lean backward as if surprised

"Conversation 0ZXY Bar 0ZXY Quux 0"	Cross arms in front
"Conversation 0ZXY Bar 0ZXY Baz 0"	Move arms as if counting or conducting music
"Conversation 0ZXY Foo 0"	Raise left hand and nod as if greeting
"Conversation 0ZXY Foo 0ZXY Baz 0"	Nod head as in agreement or acknowledgement
"Conversation 0ZXY Foo 0ZXY Bar 0"	Take a step forward with one foot ZXY then back
"Conversation 0ZXY Foo 0ZXY Bar 0ZXY Baz 0"	Look right
"Conversation 0ZXY Quux 0ZXY Foo 0"	Cross arms
"Conversation 0ZXY Quux 0ZXY Bar 0"	Keep arms crossed

These motions make a little color for your otherwise so boring conversation. One thing to notice is that speech technically last for just a moment. This means, if you play motion before speech, the guard will move first and speak after his little show, so play sound first.

In addition to this **Play sound/motion** action, there are a lot of other possibilities. I'll now explain some of them. To tell the truth, I don't actually know the meaning of them all.

Goto object
Arg. 1: *Object number or name*
The actor walks to the object defined in argument 1.

Face
Arg. 1: -
Arg. 2: *Object number or name*
The actor turns towards the object defined in argument 2.

Frob object
Arg. 1: *Object number or name*
The actor pushes a button or pulls a switch or anything that is appropriate for the object. Note that the actor doesn't have to be anywhere near the object, and doesn't actually 'push' the object.

Wait
Arg. 1: *Amount*
Simply makes the actor to stand still for the amount of time. 1000 is one second.

Add link
Arg. 1: *Type of link*
Arg. 2: *To*
Arg. 3: *From*
With this you can add links in-game. For example, you have a button, but you want it to be active **after** the player has heard some valuable information. Like where the button is.

Remove link
Arg. 1: *Type of link*
Arg. 2: *To*
Arg. 3: *From*
This is the opposite of the **Add link**, but should be used more often. When you have a conversation, you should destroy the **ControlDevice** link to it as soon as possible, so it doesn't start all over again if player goes taffing around.

Add/Remove meta-property
Arg. 1: *'Add' or 'Remove'*
Arg. 2: *Name of meta-property*
This is quite like the two last ones, but this adds or removes the actor's meta-property. If you don't know what is a meta-property, then read the corresponding tutorial.

That's quite about it. There are some other actions that I didn't mention, but they're not so important and if you really need them, you can easily explore them yourself. One thing to remember is that every action needs an actor. Even if they don't actually do anything with the actor.

In our example, we used a button to trigger the event, but it would of course be much better to activate it with player entering a specific room. This is specified in another tutorial.

(In another tutorial, Totality? Advanced DromEd tricks, Lighting lanterns or something? ***)***.

And now as you should be able to do conversations on your own, why don't you try it out. Make a two-person conversation, which consists of motions and another actor walking to a button and pushing it, and then coming back. And make sure that the conversation cannot be started twice. Don't worry about the meaning of the speech, just make it work.

If you do something wrong, just check it from the tutorial. When you succeed, you can move to the next part.

VI. Garrett as an actor and recording sounds

Everybody remembers the infamous Cutty and the conversation in Cragscleft. This was the only conversation, where Garrett actually spoke in game and was done with some obscure method called **OldConversation** and is much related on script. It's simple to do an identical conversation with the normal method.

The problem is that player isn't actually a creature, so he/she cannot be an actor. If you're thinking about making the starting point an actor or an actor that is attached to Garrett or something like that… forget it. It's so much simpler to do this with **VOTraps**. As you remember from the last part, you can make actors to **frob** objects, and they don't move in any way. So we plainly make the actor to push a button which has a **ControlDevice** link to a **VOTrap** which makes the sound.

Let's start with the **VOTrap**. If you don't remember how it's done, let me refresh your memory. Create a button and a **VOTrap**. Make a **ControlDevice** link from the button to the **VOTrap** and a **SoundDescription** link from the **VOTrap** to the sound you want to hear. For example **'garm0201'** is fine. Move the button somewhere near the start and try it. If you hear 'Hmm… a few too many to get by here.' coming 'inside your head', your **VOTrap** is working.

Now we have to hide the button. Put it somewhere underground or just anywhere, where there is no room, so you could hear the button being pressed. When the actor now frobs the button, it will seem like Garrett just made a comment. Let's look at the conversation then.

Make the **'00'** a normal conversation part with someone as **Actor1**. If you want you can use one of the previous ones. When the actor has done his part we'll make our trick in **'01'**. We need two actions for this. The actor on both these actions is **Actor1**. The same one as in **'00'**, but this time he just frobs the button and waits. **Action0** is **Frob object** and **argument 1** is the button number or if you have named it, you can use the name. **Action1** is **Wait** and **argument 1** is… let's say **3000**.

Test it and you should now have a conversation with Garrett involved. Of course, the conversation doesn't make much sense, because their topics don't quite match. You can fix this by finding the sounds that make at least some reason or if you're feeling creative, you can record your own sounds.

To record sounds, you need an external program. I use GoldWave, but I suppose that almost anything will do. For playback rate **22050Hz** is good and saving format… well I use **8-bit mono**, but **16-bit mono** works too. Note that when you open the original sounds, if they seem to be **'Microsoft ADPCM; 4bit'** or something weird like that, you should change it to something more usual. If you save them in wrong format, it will crash Thief.

Then the directories: Easiest way to do this is to create a **\snd** directory under the Thief directory and the unzip the files from **snd.crf** to that directory. You should then have, for example, the file **C:\Thief\Snd\Guard3\English\Sg3c0101.wav** in right place. Then you just re-record them. And don't bother trying to put them back in the **snd.crf** for Thief looks first in the directories and then the Crf-files.

Yep. That's quite about it. You should now be able to your own conversations and make the actors do all the funny things. If something doesn't work, try to find the reason first by yourself and then refer to the tutorial.

If you need an example for this, try downloading my 'level' called 'Conversation at the Gate' from the Circle, http://www.thief-darkproject.com.

VII. What else to do with conversations

Conversation can be used in very multiple ways. Mastering it can make you be able to do most stunning tricks and solve the most serious problems on your levels. Of course, I admire it almost as much as Totality admires sources and receptrons… Right, Totality?

You may have seen Little Mouse's the Mouse Certified Alarm System. Now we're making something like that, but with a little modification. The alarm will go off if the guard doesn't press the button every two minutes. This isn't probably the only way to make this, but I'll show you how I did it. You can modify it on your own.

Create some area, where you have a button on the wall. On the other side of the wall (in the solid void), make a **blueroom** (a room which isn't seen in-game) and another button in there. Name the first button 'CountDownButton' and the second one 'AlarmButton'. Create one guard somewhere near the buttons and one **'FrogBeast'** inside the **blueroom**. Name the guard 'Mr Jacobson' and the frog 'TimerFrog'.

Now create patrol points (**TrolPt**) for Mr J and spread them around the area. Make sure they're on a height, where guards could touch them. Link the patrol points with each other with **AiPatrol** (for example, from first to second, second to third, third to first). Now give Mr J the property; **Ai->Ability->Patrol: Does patrol: True**, and place him upon one of the patrol points.

Now we need to make Mr Jacobson push the CountDownButton, so we create **a marker** named 'ButtonSpot' in front of the switch and add a **AiWatchObj** link from Mr J to a patrol point near the button. I suggest that you name the patrol point first. After we have created the link, we must edit its properties. Change the properties to following:

Watch kind:	Self Entry
Trigger:	
Radius	5
Hight	5
Req. awareness	(0) None
Line req.	None
Min. alertness	(0) None
Max. alertness	(1) Low
Exit:	
Link kill option	Don't kill
Kill like links	( )
No tesh once triggered	( )
Reuse delay	20000 (which is 20 seconds)

Reset delay	0
Step 1:	Goto object
Arg. 1	ButtonSpot
Step 2:	Face
Arg. 1	-
Arg. 2	CountDownButton
Step 3:	Play sound/motion
Arg. 1	-
Arg. 2	-
Arg. 2	WorldFrob 0, AtWaist 0, BellPull 0
Step 4:	Frob object
Arg. 1	CountDownButton

When you test it, Mr Jacobson should be patrolling around and when he comes by he goes over to push the button. You may need to move the **ButtonSpot** so he doesn't stand too far or too near the button.

Now we can make the actual conversation. Create a marker called 'TimerConv' in the blueroom, and make a **ControlDevice** link from the CountDownButton to it. Make the frog actor one with the AiConversationActor link. Give TimerConv the normal properties and copy the following to the conversation's properties in step **'00'**:

Actor:	ActorOne
Action 0	Wait
Arg. 1	120 000 (120 seconds)
Actor:	ActorOne
Action 1	Frob object
Arg. 1	AlarmButton

This means that the frog will wait two minutes and then set off the alarm. Every time the guard (or a sneaky thief) pushes the button, frog starts counting from the beginning.

Now we just need the alarm, so find the 'RamirezAlarm' and place it on the wall (note that some of the alarm lights don't work) and just add a **ControlDevice** link from the AlarmButton to the alarm.

Add two more guards to see how it works out, and don't forget a blackjack, so the alarm actually does go off (after mugging of Mr Jacobson).

You can find my version of this from the Circle.

VIII. Tips and hints

- If you don't seem to hear sounds, try building room database. Also check that rooms are correctly placed.
- If the sounds disappear completely after trying the level once, try re-launching DromEd.
- If an action doesn't seem to work, check that you have right arguments. If you're not sure, try rearranging them. Also verify that you have an actor selected.

- Try doing your own combinations on motions.
- Research the levels. For example, the following useful combination is servant's motion for putting down the tray in Ramirez's: **WorldFrob 0, WithTray 0, AtWaist 0, BellPull 0**
- Try fancy tricks. In Datoyminaytah's "The Gem" level, Farkas turns towards player, which is a spectacular result, considering it's done with DromEd. Actually, an easier method is to use AIWatchObj set for player intrusion on the AI; use the "Face" action with argument 2 "Player" (not argument 1).
- When recording, get some friends to do the sounds instead of manipulating your own voice over and over again. The result is much more pleasant.
- If you have trouble check other texts like Mucho Macho DromEd guide. They can be found from the Circle, http://www.thief-darkproject.com

If you need help, write me, teemu.salmela@kolumbus.fi or go to the Forum, http://www.ttlg.com.

And if you don't want to be labeled as a newbie, use the Forum's search function first!

Deep Qantas(Teemu Salmela)
teemu.salmela@kolumbus.fi
or ICQ# **36112496**
==

Conversation Name Encoding

A conversation name is encoded. To determine the actual "wav" file used, substitute the actor number at the underscore and append a 2-digit suffix for the line number.

Example sg_c19: This is SwordGuard conversation 19. The underscore "_" is replaced with the actor index, "1", "2", or "3". The "Line No" is appended as a right-justified, zero-filled two-digit suffix. Thus, for actor 1, Line No 1, the file name is "sg1c1901.wav" in the snd.crf (zip) file. For actor 3 (actor 2 is not used in this conversation), Line No 3, the file name is "sg3c1903.wav".

Understanding the encoding technique may help you to add new conversations without modifying the existing conversation files; just use the encoding technique.

==

Changing object names

In your objects Properties add 'Inventory' -> 'object name' and write down a text string (It can be whatever you want). This text string is used as an 'assign' to the actually name/text you want to use in the mission and should be put in a copy of the objnames.str file.
If you have checked out the directory content in FM's, (If not, do so) you noticed the 'Objnames.str' file is inside the 'strings' directory.
The 'objnames.str' is a plain textfile so use your favourite ascii editor to make one. The format to use the text string assigning to the new name/text is;
text_string: "New Text"
So, if you as a text string would use 'A_Testname', and the new name "The Cellar Key", the finished line of text would be;
A_Testname: "The Cellar Key"
... And so on for all the objects name/text changes.
PS. Don't forget to only put one text string rename per line.
==

Sources and Receptrons 4, The Magic Torch Relighter.

Note: Still under revision by the guild.

Aim of Tutorial

This tutorial will hopefully relate the necessary steps involved in getting an AI to perform a set sequence of actions indefinately whenever he notices a change in the environment. The method is versatile and could be applied to nearly anything that the player could interface with. This actual example used in the tutorial is of a Hammerite Novice going about relighting any doused torches he spots. Examples of other potential uses are a cleaner going about washing out the bloodstains, a hammerite healer, a necromancer who raises the bodies of corpses, etc.

Assumed Knowledge

This tutorial is for ADVANCED dromed users, who are fairly comfortable with intermediate dromed operation. A basic working knowledge of sources/receptrons is assumed.

IMPORTANT: the system involves assigning a source/receptron relay system on to the Avatar Garrett so that he can interface with objects via that method, and so needs a custom gamesys with such a system on it. See the tutorial on teleporting Garrett for details on how to do this.

The tutorial starts from a basic level with all necessary scripts, starting points loaded.

Discussion of the process

First let us decide our aim. We want to create a system that will allow our Relighter to walk up to and relight any doused torch. With most difficult tasks, it is perhaps better to take a simpler example first, and then to expand it, the simplest version of walking is to have it so that the AI only has worry about one torch.

This is easier, but just in case the solotion doesn't spring to mind lets dissect it further. We have to tackle three hurdles.

1. How does the AI notice that the torch has been doused?
2. How does the AI relight the torch?
3. How do we make it repeatable?

The first can be answered by using receptrons. If you analyse the torch archetype you'll notice that this is how the dousing procedure works anyway. When the torch receives a water or a knockout stim stimulus, it triggers the script that removes the flame. So by using these two receptrons, we can do 'something' once it has been doused.

The second can be answered by using the AI_WatchObj link. You can assign this from an AI to an object (which I call a trigger object), and when the object comes within range of the AI you can trigger a series of actions much like those available in a conversation. There is no relight torch action however, but using a simple control device link can turn torches off and on. We can simply get the relighter to frob the light switch as it were.

So, we can set it up so that when the torch is doused, an object that the relighter is set to watch is teleported in. The novice can then be told to walk up to the torch, wave his arms, and then frob the relight switch. To answer the third problem, we can then teleport the trigger object away again so that the loop can be repeated. When a torch is lit, it can send out a ControlDevice TurnOn signal. Likewise, when a torch becomes unlit (through whatever means), it can send out a ControlDevice TurnOff signal.

Getting a relighter to look after one torch

Within your basic level, create a series of passageways that form a loop or square. Add a torch on a wall nearby to your starting point, and create a hammerite novice, call him 'Relighter' and the torch 'theone'. As we don't really want him to get upset about our thief-like attributes, assign him to the good team. (AI->AI_Core->Team). Move him so that he is near the torch, and also place some water arrows on the ground nearby.

Next we need the switch to turn the light on and off. Actually, even though I say switch, the best thing is a button, as that way it will always turn the light back on again. If it was a lever, then the second time you doused the torch the lever would still send the 'off' signal, and it would not relight. Create the button in a blue room somewhere and aptly name it 'lightswitch'. Then add a control device link from the button to the torch. If you can access your blue room when in the level you can now go in and douse the torch and then hit the button to relight it.

The final step is to set up the trigger object and the relevant link. First create a markers in a blue room quite away from your proper level. Call it 'relightone' and then create a teleport trap called 'markstart' in roughly the same location. Assign a control device link from markstart to relightone so that we can reset the trick once it has completed. Now, assign an AIWatchObj link from the novice to relightone, and then highlight that link and click on data. You have to have a fairly beefy screen resolution to see the full window, but you can get away with 1024x768 if your taskbar is on the right or left hand of the screen. Change the watchkind to self-entry rather than player intrusion, as we want it to happen irrelevant of where the player is. Set the trigger radius to 10 and the height to ten. We can now program the responses. Step one is to 'goto object', and for argument one enter in 'relightone'. This gets the AI to walk to the torch. Step two is to 'frob object', argument 1 is 'lightswitch'. That is it for the AIWatchlink. We now need to set up the receptron that teleports the trigger object in the first place. Add a receptron to the torch, stimulus 'waterstim' , min 0 no max. Effect is to teleport object, target 'relightone' agent 'me' edit effect, Z = - 2. The z displacement is so that the marker does not appear in the torch, as otherwise the novice sometimes doesn't see it. In your actual level you may also want to add the knockout stim receptron to do the same thing, as that will also douse a torch. Finally, add a control device link from the button to the teleport trap, markstart, so that it is teleported out when the novice frobs it.

Assuming all went well, if you now go in the game, whenever you douse the torch the novice would relight it.

How to expand it to many torches?

Now that we have done it for one torch, how do we do it for many torches. Well, the simplest answer is to repeat the process over and over again, for all the torches in the level. This is extremely tedious, as we would have to have a light switch for every torch, a trigger object for every torch, and an AI watch object for every torch. If you have many torches, this becomes an aggravating alternative.

Well let us re-think our aims.

1. The relighter should notice any torch that has been doused without multiple AIWatchObj links.
2. The relighter should relight any torch that has been doused without multiple light switches.
3. The process should be repeatable.

The solution to this is harder, and took me a while to figure out. As you may have noticed from the title it involves sources and receptrons. I set myself the restriction of using only one AIWatchObj link as I didn't want to have to program dozens of links at the start of the level. I could have come up with a system of creating an AIWatchObj on the fly midgame, but I eventually realised that I could come up with my own custom AIWatch mechanism by using a radius source and receptron.

If the relighter is given a radius source that fires every couple of seconds indefinitely, and the torch a receptron looking for that source I could trigger a series of actions once the relighter got within range of the receptron, like an AIWatchObj link. Of course, the options avalaible on the receptron edit screen are different from those on the AIWatchObj link, and I needed the goto object one, which is not available. The trick is so that when the relighter nears the doused torch, it teleports the trigger object to its location. By expanding this method we have succeeded with the first aim. The doused torch will teleport the trigger object when the relighter nears it, meaning that we only have to have one AIWatchObj link (but lots of receptrons).

Of course, we have to deal with what we are going to add the receptron to first. We cannot add it to the torch object, as that way the sequence will trigger whether the torch is doused or not. Instead we change the receptrons upon the torch itself, so that it creates another marker beneath it once it has been doused rather than teleporting in the trigger object. This marker has the receptron on that will teleport in the trigger object once the AI gets close.

We now have to address aim 2. As we cannot use multiple light switches, we have to come up with an alternative method to light the torches. If you are a good taffer, you will remember that torches can be relight with fire arrows, or more specifically fire stims. If any fire stimulus hits a torch it will relight. Of course, we don't really want to use an area affect fire stim as that would kill anyone nearby, but we could use a custom stimulus, and then convert it to a fire stimulus that will stimulate only the torch. Within the sequence of events, we will now set it so that after the novice has gone to the trigger object another marker will be teleported in that has this custom stimulus on that relights the torch.

Finally there is aim 3 to assess, making the process repeatable. In other words, tidying up the system. With one torch, we had to teleport the triggerobject out the way again. This applies here as well. Another necessity is to remove the markers that trigger the teleportation of the triggerobject once the relighter is close, as otherwise we'll get stuck in a loop. The final thing is to teleport the marker with the stimulus that relights the torch. The reason for this is otherwise any torch that has just been relit by this method we continue to do so, even if you douse it again.

We can use the relighting marker to also serve as the destroyer of the other markers, and we can get the AI to frob another switch that teleports the relighting marker and the trigger object back out again.

That's the theory.

Putting it into place

Let's first add add another novice that patrols about, making sure that his torches don't blow themselves out. Create another novice on the good side, and some patrol points for him to walk around. This novice I have dubbed 'Relighter2'. To speed matters up, we are going to add our receptrons to the object hierachy so we don't have to add them every single time. Open up the object hierachy, scroll to the torch entry and click add. For our new sub-object enter in 'TotTorch'. If you haven't already got one in your gamesys, add a custom stimulus as well, 'TotalityStim' is what I'll refer to, but you could use a HolyWater stim if necesary (See sources and receptrons tut 1 for more details on this.). Save a custom gamesys, set your level to use it with set_gamesys, and then resave your level. Using your new TotTorch'es, create a whole bunch of them lining your patrol path.

If you remember the theory we need three different types of markers. The simplest is the trigger object, create that in your blue room and call it 'FrobMe'. The next simplest is the marker that relights the torches, simply create a marker called 'lighting', we'll get back to that later. The last are the markers that teleport the trigger object in once the hammerite relighter nears. As these are to be created on the fly we may as well make them a new object archetype. Edit the object hierachy, and add a new subtype of markers, named 'LightMe'. Then edit that archetype. To start with we want to add a TotalityStim receptron, min -6 max -4, and the effect is to teleport object, target: FrobMe, agent: me. Next is the corresponding source on the patrolling novice. Stimulus is obviously TotalityStim, propagator Radius, intensity -5 and the no max firings flag set. The period between firings and the radius should be dependent on the minimum distance between your torches. As in my tut level they close together, I'll set the radius to 5 and the time period to 1000 msecs. Before anything can happen we need to edit the torch archetype to create those 'LightMe's whenever any TotTorch is doused. Go to the object hierachy, and add a water stim receptron. Min 0, no max, and the effect is to create object, target: Lightme, agent: Me, edit effect z = -2. (Note: you may have to actually type in the number of LightMe in the object hierachy rather than enter in its name). Save the gamesys.

Next lets assign the AIWatchObj link from the patrolling relighter to the 'FrobMe' marker. Click the link select data, change player intrusion to self-entry, radius 10, height 10, first reponse is to Goto Object, arguement 1 'FrobMe'. The second reponse is to frob object, though I'll explain what we are going to frob in a bit. The third response is wait, arg1 of a 500. Fourth response is to Play a sound/motion, and arguement 3 is 'Conversation 0, Foo 0' so that he looks like he is actually doing something to the torch. The fifth and final response is another wait statement, this time for 3500 msecs.

The frob command with the missing arguement will trigger a teleport trap that teleports the 'lighting' marker to the patrolling relighter. Of course, we need the teleport trap to somehow mirror the novice's location. We could either attatch it to the model, or simply add the relevant script to the novice so that he becomes the teleport trap. Add the script 'TrapTeleporter' to the novice, and create a CD link from the novice to 'lighting'. Then create a button called 'sendlighting' with a CD link to our walking talking teleport trap. The arguement for the frob object command that we missed out is that button.

Now there's the tidying up phase. Add a TotalityStim source to the lighting marker. Propagator radius, intensity 2,radius 5, remove the line of sight flag, no max firings flag, time period 3000. This is our relight signal, so we need to tell the TotTorches to relight. Edit the archetype again, add a TotalityStim receptron, min 1 max 3. Effect to Stimulate Object, target: me, agent: me. Edit Effect, stimulus FireStim, MultiplyBy = 1. We also need to get rid of our lightme markers, edit their archetype as well, add the same receptron but the effect is to destroy object, target: me. Save the gamesys.

The final piece of tidying is to teleport FrobMe and lighitng markers back out again once the torch has been relit. Create two teleport traps, TeleFrob, TeleLight each with a CD links to FrobMe and lighting repspectively. Then create a RequireAny trap to act as a relay, with a CD link to both TeleTraps, and then a new button called reset with a CD link to the requireany trap. Then, go back to the AI and add a sixth repsonse, to frob 'reset'.

There we go. He'll now go about happily relighting torches. Of course, it may take him ages to get to the torch that you just doused? Unless we get him to change his patrol path on the fly, now there's an idea for another tutorial.

Tips and hints

Always remember the bug when editing with sources and receptrons - you must have a source or a receptron added to an object after you edit it for the first time. You can delete it afterwards.

By playing with timings, you'll be able to enhance the illusion that the novice is relighting the torch

Adjusting the offsets of the creation effect so that it is slightly away from the torch in the y-axis as well as the z when you create the lightme's might help

Also, be careful if your torches are closely packed together, as this will probably muck up the system.

==

There is an effect under receptrons named set property.

It has a target, agent, and edit effect. Target is the thing whose property you want to set, agent is the object whose property you want to copy. Under edit effect you have to type in the properties name.

For visibility it is:- AI_Visibility.

To get a list of the different properties start up a mono log file,

'mlog '

and then 'list_props'

this dumps all the property names to that named file.

That's all you realy need to know to get the basics done, but you have to juggle things about to be able to interface with Garret.

For instance, to make Garrett invisible you have to do some more work. Assuming Garret has a TotalityStim receptron on that bounces back the stimulus like I normallly have (see the tuts), you have to create a source, radius, no max firings, period LOW, about 10 -100. And then a receptron that looks for Totality and sets the property as above with Target Source and Agent.

The problem is if you add the current visibility tag to the something, then it will disappear after one exectution of Dromed. To get around it, at the start of the level you have a source receptron relay thing again that does the opposite, and sets the to have Garret's current light visibility. If he is in the dark when he frobs it, he will still be in the dark when he goes near that marker. All that is left is to get that marker to follow Garret about a bit.

Even though this does overide the light gem, they can still see you (even though you are apparently in total darkness). I aplogise if this did get anyone's hopes up, but I hoped that the vision check would be done on the current visibility. It must deliberately update the visibility before it runs the AIs through.

Dang again. It's still nice to know that you can change properties on the fly though, I definately know that it is an extremely quick and nice way to wake up knocked out guards.

I suppose you could use to tell the light gem to give false readings.

Found a possible work around that DOES work, though it will require more programming.

Have two guards in a blue room somewhere, one with vision set to null, the other to average or good. Then, have a switch that set's all the guards 'vision' attributes to null to go invisible, and one other to set them all back to good to go back visible.

Yes, of course you can add properties. The problem is how do you set their attributes? I haven't found anyway to specify them. Edit effect doesn't allow you to change them.

BTW, Don't use state properties. That's what the engine uses to keep track of it's current state. They will be overwritten whenever the state changes.

Use AI_VisModifier (AI: AI Core\Visibility Modifier) or AI_VisCtrl (AI: Utility\Visibility control). You'll have better luck with them, but only if you can figure out how to change their values.

It DOES allow you to 'change their values',

It takes the property you name under edit effect, and copies what it is for the Agent object to the Target object.

So if you have a guard whose Current State is alive, and use that as the agent and the target as a guard whose current state is dead, he'll spring to life.

If I guess correctly, that AI_Utility function sets how quickly Garret can move without being seen??? I suppose that if this could be set so that Garret is never visible no matter how fast he is moving or what light stance he is in then that would work. It all assumes that my guess is correct.

Yes, that's right. It's also got mods for how visible he is in low, med, high light levels, crouching, etc.

I tried AI_VisCtrl on the player. It worked. The guards couldn't see me, but they could hear me walking (which was amusing and what I wanted).

==

Tonight I'll just explain how a Flicker tweq can be used for timed events.

The trick is the TrigFlicker script. It changes the flicker into TurnOn and TurnOff signals which can then be sent to some other object via CDs. The easiest way to use it is through the fnord / FlickerTrigger object. Create one of these, attach a ControlDevice from it to say a door, set the FlickerState to On and go into game mode. Once every second the door will close or open.

Note, the Flicker tweqs first signal is TurnOff, the next is TurnOn and then it repeats.

Add a Flicker tweq to the FlickerTrigger and change the rate (it's in milliseconds) to something like 5000. Now the door doesn't look like an oversized fan any more since 5 seconds elapse before it cycles.

Change the Halt action to Stop Tweq, remove the nolimit flag, and set the Flicker State Frame # to 2 (the number of cycles you need to get one TurnOn signal) and you have a door that will open once after 10 seconds (5 seconds for each of the 2 cycles).

With a little extra work you should be able to make use of the TurnOff signals to put together your own haunted door, but I'll save that for another night **For Thief2, just specify the Timing value.**

==

TrapTrig scripts

TrigDoorOpen
Used on doors. Will send an on CD signal as soon as the door starts to open and will send an off CD signal once the door has closed. Replaces the StdDoor script otherwise you get the door zooming of into the distance syndrome. **When adding scripts to doors, be sure to delete the standard scripts and <u>uncheck</u> the "Do not inherit" option.**

TrigSlain
Sends an on CD signal when anything is slain (i.e killed), Unsure if you heal its hitpoints whether it will trigger again if reduce it to less than nought (it would be cool if it did, it would simplfy my in-game store a bit)

PlayNoisemaker
Makes the noisemaker sound when the object collides with something, whether it be the player for a stationary object, or anything else if the object can move.

TrigAIAlert
Sends an on CD signal when the AI becomes fully alerted of the player.

TrigUnlock
Sends an on CD signal when the door/lockbox is unlocked, and an off CD signal when the door/lockbox is locked again.

LootSounds
Plays the sound of coins rattling when you pick up the object.

StdButton
Allows the object to be used as used as a button, that is, it will send an "on" ControlDevice signal when frobbed.

ActivateAmbient
Add this script to an ambientSound object that has the “TurnedOff” flag set. Then send a TurnOn ControlDevice signal to the ambientSound object to turn on its sound. A “TurnOff” signal will turn off the ambientSound object.

AnimLight
Allows to object to be an animated light object: that is, it can be controlled by a Control Device; it can flicker, dim, or whatever.

TrigRoomPlayer
Allows a room to act as a Control Device, triggered when the player enters the room.

TrigWorldFocus
Sends a TurnOn CD signal when the object receives focus (i.e. is highlighted) and sends a TurnOff CD signal when the focus is removed. In order to function properly, the object must also have Engine Features / FrobInfo : World Action = FocusScript.

TweqOnOff
Causes a TurnOn signal to start the objects tweqs and and TurnOff signal to stop them. For example, a two state switch could be used to start and stop an object rotating.

TrigRoomCreature
Like TrigRoomPlayer except that it triggers if any creature enters the room (including the player).

TrapConverse
Allows a scripted conversation to be ran. See relevant tuts.

TrapTeleporter
Teleports the target object to the trap.

Two CD links one from the trap to the object to be teleported, and one to the trap which triggers the teleport.

TrapCreate
Seemingly redundant. 'Creates' object that trap contains at trap's location. One Contains link holding the object to be created, and one CD link to the trap that triggers the process. Works once?

TrapInverter
Works like a NOT gate on CD signals.
Two CD links, one in and one out. Turns an on signal into an off, and an off into an on.

TrapDestroy
Destroys a target object upon signal.
Two CD links, one to the object to be destroyed and one link from the trigger. Any number of objects may be specified, and the DestroyTrap will destroy itself upon completion.

TrapRelay
Sends the in signal to multiple outs.
Numerous CD links, usually multiple out links to other traps. Whatever the trap receives it will send out to all the outgoing CD links **in the order specified**.

RequireAll
Essentially an AND gate.
Numerous CD links, will only send the out signals when all of the input signals are on.

RequireAnd
Essentially an OR gate.
Numerous CD links, will send the out signals when any of the input signals are on.

StdBook
Applied to a non-pickupable object. Will display text when frobbed, either with no background or background depending upon the Book->Art, Book->Text properties.
Has to have Script in the world section of FrobInfo.

StdScroll
Applied to a readable object in inventory.Will display text when frobbed, either with no background or background depending upon the Book->Art, Book->Text properties.
Has to have Script in the Inv section of FrobInfo.

TrigFlicker
Allows a Flicker tweq to send TurnOn and TurnOff signals over a ControlDevice link. It's not clear if you have to have scripts selected in the MiscC of the tweq. FlickerTrigger in Thief does but it doesn't seem to matter either way.

TrapOnFilter
Takes input coming from a ~CD and filters out TurnOff signals. TurnOn signals are allowed through and are sent along to any CD's that the object has. If you want to filter out TurnOn signals instead then you have to put a TrapInverter both before and after the TrapOnFilter.

==

Simple combination lock using four switches.

I’ve set this up for four but once you get the idea you could use whatever you want.
Setup: four switches to open a door, crushing wall whatever… if one of the switches is thrown in the wrong secquence, an emmitter will fire at the player…
The comination for this example is 3241
Create four switches, sw1 sw2 sw3 sw4
Sw1-cd-require all
Sw1-cd-link1-cd-emittrap-cd-fireball (arrow, whatever)
Sw2-cd-require all
Sw2-cd-link2-cd-emittrap-cd-fireball
Sw2-cd-distroy link4
Sw3-cd-require all
Sw3-cd-destroy link2
Sw4-cd-require all
Sw4-cd destroy link1
Sw4-cdlink4-cd-emittrap-cd-fireball
RequireAll ControlDevice (door, crushing wall, whatever)

==

I am working on a tutorial for creating new sound schemas based on the schema files included in Thief Gold. However, the tutorial will have enough information that users of "Classic" Thief will still be able to create new schemas.
Here's something to get you started. Under your Thief directory, create a directory called "schema". Place in this directory a plain text file called "myschema.sch" (make sure notepad doesn't append .txt to the name) containing the lines below:

```
----- begin, don't include this line -----
// This is a comment
// Define new archtype under Object->Sound->Schema->AMB
schema MY_AMB
archetype AMB
//Duplicate of START MISSION
schema mystart
archetype MYAMB
volume -500
keehit_s // name of .wav file(s) minus extension
----- end, don't include this line -----
```

Now run DromEd, and in the command box type "zggtvrk_load_schema myschema.sch" (minus quotes.) Now open the Object Hierarchy, and under Object->Sound->Schema->AMB you should find MYAMB containing mystart.
In the command box, type "play_schema mystart" and you should hear it.
Actually, most (if not all) of the other parameters that can be specified in the .sch file can be edited in DromEd, if you edit the schema, so this example should get you pretty far. Just put your new .wav files in the Thief\SND directory, and specify them (minus extension) instead of "keehit_s".

If you do tweak the schema, don't try to hear the full effect with the "play_schema" command in edit mode, which is limited. Create a marker, add AmbientHacked and specify your schema, and listen to it in game mode.
OK, one or two more things you can't do in DromEd.
You can specify multiple .wavs by stringing them together like this:
wavname1 wavname2 wavname3
They can be on separate lines as well.
You can weight the individual wav's like this:
wavename1 freq 1 // don't play often
wavename2 freq 3
wavename3 freq 5 // this one gets played most
I don't know the maximum value for freq but the highest observed value is 7.
The tutorial will have an explanation of as many parameters as I can figure out, but most are easy enough to figure out by experimenting.
===

SplitPortalPolyhedronByPlane: polyhedron didn´t cross plane

First of all: MAKE A COPY OF THE MIS-FILE !

1.) Start DromEd new

2.) Portalize

3.) Compute Pathfinding Database

4.) Light

5.) Optimize

If now the error " SplitPortalPolyhedronByPlane: polyhedron didn´t cross plane "

comes still by using Optimize then try "clear_surface_cache" most of the time it doesn´t help but it is one off the quickest way´s without a danger for the Level)

First off all

MAKE A COPY OFF THE *.MIS FILE

first way (and most time the quickest) to eliminate the bad brushes

--- Try "hilight_check_snap" and then "hilight_do_snap"

try it with with different grid (11-16) and with many caution,

maybe it will make a loot of changes in your level if you use a to big grid.

second way to find the bad brushes

--- creat an area brush around the half of your level and, select " Me only ", and

Optimize. If you get the bug, try again with a smaller area brush, if not try the

other part. Make so on and make the part smaller and smaller.

If you have at last a little part of your level with ca. 20-30 brushes then

delete some of them and optimize new. By the end you will find the bad Brush.

I call this way try and error ;-)

third way to find the bad brushes

--- try this methode only if the last optimize is not to long ago

By crashing DromEd makes a file named " Crash000.log "

Look in this file, it will look like this example :

```
crash 1
Brush 1193 (face 0)
Brush 1167 (face 4)
Brush 1167 (face 3)
Brush 1167 (face 3)
Brush 1167 (face 3)
Brush 1167 (face 3)
Brush 1167 (face 3)
Brush 1197 (face 1)
Brush 1167 (face 3)
Brush 1167 (face 3)
Brush 1167 (face 1)
Brush 1167 (face 0)
Brush 1208 (face 0)
Brush 1208 (face 0)
Brush 1208 (face 0)
Brush 1208 (face 0)
Brush 1208 (face 0)
Brush 1208 (face 0)
Brush 1167 (face 5)
```

The Nummbers are the same you see in DromEd in the leftdown area on the screen

" Time > XXXXX < ".

By using the key " Tab " you can circle through the Brushes onward.

By using the key " Shift+Tab" you can circle through the Brushes backwards.

Now look what kind of Brushes that are and look where they are.

Now delete them from the highest number to the lowest

(Before you do this all MAKE A COPY OF THE MIS-FILE !!!!!!)

(If you make a screenshot of the Brushes you delete then you can later, after

Errorfixing make the good brushes new)

Now save the file and

1.) Start DromEd new

2.) Portalize

3.) Compute Pathfinding Database

4.) Light

5.) Optimize

If the error " SplitPortalPolyhedronByPlane: polyhedron didn´t cross plane " comes still, look in the new " Crash000.log "-file and make the same.

When Optimize works without error you must give maybe an other shape in this parts

of your level

The disadvantage of this method is, you delete brushes, maybe many brushes

and few of them are the bad you don´t want.

long time ago by the last optimize ==> many brushes to delete

==

Do you ever want to immerse yourself in Thief and enjoy the architecture or simply explore without worring about little things like death and/or annoying guards?

Try modifying your gamesys file!

Removing some of Garrett's metaproperties (like Firedamaged, etc) after copying anything that isn't Act/React (CsArrow: arrow_body [or whatever] and things like that), adding it to Garrett himself and not the metaproperty. Then check the Receptrons on Garrett himself, and remove Holy and Fire stims. Then look at avatar and remove anything left dangling down there (except for RestoreStim, if present).

Viola! Invincibility!

Now let's say that you've gotten incredibly lazy, and don't want to kill the things that get in your way, or have to draw your sword and click to slash that banner.... Simple! Edit your compass (compass2, that is). Give it some sources. For instance, SlashStim if you don't plan on Expert play (after all, Expert is for serious play) and HolyStim. Give them a Radius rather than a contact property, with Compass2 as the source (intensities are your choice; I used 6 for both). Set no max firings, and give it a radius of about 5-10. Line of Sight (Raycast) is recommended so you know when something is done.

Now, whenever your compass is selected, Garrett becomes an avatar of doom! Nearby torches go out, banners are slashed, guards find themselves bleeding to death from sudden lacerations that have no apparant cause.... Also, pick your compass and then cycle inventory to any other item. Your compass has been "left behind" for purposes of damaging things. "Leave" it at the doorway to a room, loot, and if any guards show up then you don't have to pay them any attention. Select it again to "pick it up." (Note that giving Garrett immunity to the stims you make is a very good idea, unless you've made

Garrett invincible as mentioned above; setting the period between firings to 500 or less is a good idea, as well).

And finally, The Thing Garrett! Set his Bash Factor to 20 or so. Now gently nudging a crate makes it go boom, and almost no wooden door can stay closed when you knock on it.

This is intended for amusement only. Since newbies generally don't come here as often, and it involves editing your dark.gam file (after backing up, of course), I posted here.

And a bit of trivia: did you know that Sandbags have a KnockoutStim source on them?

==

AIwatchobj are amazingly useful.

They essentially are the enablers to what many refer to as "scripted events" as well as others.

For most applications, it is best to have an AIwatchobj link from an AI to a regular marker object. This specifies the AI to be the one who will *react* and the marker as the "location for which the event will trigger".

Within the data of the link, you specify a radius around the marker for which the event will be triggered. Typically you want this about 5, no less than 3 (too small and it can be unreliable), although it can be much bigger as the situation calls for it. When the triggering object enters this radius, then the event will be triggered.

The "triggering object" is specified as the first variable in the data of the link. There are 2 choices: player intrusion (when Garrett enters radius) or self entry (when the AI who is AIwatchobj linked to the marker enters radius).

Don't select option "kill on completion" (or whatever the thing says) if you want it to occur repeatedly.

Line of sight specifies another condition for the trigger. In addition to the trigger by proximity, it also requires: none (no extra requiremnt), line of sight (AI must actually be able to SEE the thing; ie it won't trigger if the AI is behind a wall), raycast is almost the same as line-of-sight, but the AI doesn't have to look directly at the object. That is, raycast is like a light ray shining from the object and striking the AI, regardless of which way the AI is facing; the raycast won't go through anything that blocks light (i.e., solid terrain and doors that have the "Blocks Vision" enabled).

Note that the actions you list in the data will only occur within the radius of the marker itself. This doesn't matter for most things except "go to object" which can cause the AI to leave the radius.

I've found that you also need to select the "no test when triggered... " option or it won't work the first time, presumably since it performs some sort of test. I also set the height to between 8-10. For some reason leaving at 0 causes problems. You might try putting the marker about 1' off of the floor just in case you have way up in the air or totally floored.

You added AIWatchObj to a *marker*?

A marker is not an AI, so that won't work. The purpose of the link is for an *AI* to react as if it had noticed something.

If you only want the conversation to trigger when Garrett reaches a certain point, either use a BoundsTrigger or else start the conversation the "traditional" way with a custom room type with the TrigRoomPlayer script. Also, you can use the AIWatchObj for player intrusion to cause an AI to frob a hidden button. The button sends a ControlDevice TurnOn signal to a conversation marker that can start the conversation.

==

I mentioned before that you can control the size of the particle effect with the bounding box, this is both true and misleading. While it does increase the total area it also has an unfortunate (for weather anyway) side effect. Particles will appear in random locations within the box, on ALL axes. To avoid this and other problems, here is a general guidline for creating snow and rain and hail and blood from showerheads like in Blade:

1) Create a H2OstreamFlat for your base effect. It is in the object hirearchy-->SFX-->WaterFX-->H2Ostream-->H2OstreamFlat.

2) In it's properties open the Particle Launch Info window. Change only the x and y box mins and maxs to alter the length and width of your effect. The size of the cube representing the effect in DromEd will change accordingly only after you've entered game mode once. Also note that the effect has it's own set of axes, if you set the y max for 25 make sure to set the y min to -25. Consider the h2ostreamflat cube in DromEd to be composed of four equal quadrants, if ymax=25, ymin=o, xmax=25 and xmin=-25 then only half of the cube will contain particles.

3) Place the top of the effect at the height in your level from which you want the particles to fall. Even though our bbox is set to have no verticle dimension the cube still appears to have one in DromEd, why is that? This happens because the cube also reflects any physics imposed on the particles, in this case gravity.

Right now the paricles never reach the ground, to fix this we must change the max and min times in the Particle Launch Info window. These values represent how long each particle is visible. If the min=0 and max>0 then the particles will disppear at any point in their descent. For our purposes we want the min and the max to be the same. What numbers you use depend on how much time the particles need to fall to the ground. (If the particles act strangly after changing the timing just leave and enter game mode a second time.)

4) Now bring up the Particles window in the properties. Number of particles and Size of particles are self-explanatory. The x and y values for gravity can be used to simulate wind (note: this will change the size of the cube in DromEd) z simulates, um, gravity (remember, the sign denotes direction while the absolute value denotes magnitude.) Full.pcx in fam.crf (using winzip) will help you pick the right color for the color (palettized) field. Alpha is transparency. If the player starts in or near the particle effect then you want to check the 'Paricles start launched' box, otherwise it will look as if the rain, snow, etc just started to fall.

Note on the side, good light blue color for rain is index # **166**

so [Texture Name (paletted) : 166[

==

To Flinderize:

'From' is the object that needs to explode, 'To' is the archetype of the object that explodes from it. In the Data: 'count' tells how many of that particular archetype it explodes, 'impulse' is how fast they explode, and the checkbox for scattering determines if the objects actually explode out or just drop to the ground.

==

Hey, here's something useful
When you go into game mode always have an air brush selected in your 2d window. NEVER have a room brush selected; it could rename all your room brushes. Actually, never have an object selected either.
Courtesy of Lohan.

==

Topic: Is it possible to get a map during game play?

map_min_page and map_max_page, as you probably know, are mission quest variables that determine how many map pages are accessible to the player. Let's say you've got map_min_page = 1 and map_max_page = 2. Then, Thief will assume that you've got map pages named page001.pcx and page002.pcx. When the player first goes to the map, they'll see page001.pcx. They'll be able to hit the "right arrow" button to go to page 2 (page002.pcx). From that point, there's no "right arrow" button anymore; all they can do is hit the "left arrow" button to go back to page 1.

Now, let's say you include a third map page with your mission called page003.pcx, but you still set map_min_page = 1 and map_max_page = 2 at first. When the player first starts the mission, they won't be able to view that third page. In order for them to view that page, something has to happen to set map_max_page to 3. Here's where the QuestVarTrap comes in.

A QuestVarTrap can be used to set mission quest variables. For example, you can use it to set goal #3 to "complete" when triggered. I have never tried it, but it might also work on the map_max_page variable.

So, you create a QuestVarTrap. You set its Trap->QuestVar property to something like this:

=3:map_max_page

You make something a ControlDevice for the QuestVarTrap. That way, when the trap is triggered, it will set map_max_page to 3. Then, when the player goes to the map, they'll be able to see all 3 pages.

As a foot note I would like to mention that you could actually get a map seemingly to update as you walk around (i.e like an automap).

The trick is to essentially change the automap properties of each room on the fly so that when you walk to a certain place the overlay is changed. Unfortunately, as you can only have one overlay at a time being displayed on the map, it means that you need a lot of PCX files to cater for every possibility.

==

Fleepoints

Create a marker and give it a Fleepoint property between 1 and 100, 100 being a great place for AI to flee to and 1 being not such a great place.

Then, for any AI that you want to flee to that point, give them a ConditionForFlee property (found under the AI category, but I forget which sub-category). For example, you can make them flee on alert level 3 or on low hit points. When the condition for fleeing is met, the AI will pick a flee point based on (1) its Fleepoint value and (2) its location. They will favor fleepoints with high Fleepoint values as well as Fleepoints that are far from the player. Then, if all goes well, they'll run there.

As a side note, I believe any object can be given a Fleepoint property, not just a marker, but I'd imagine it wouldn't work if the AI cannot pathfind to that fleepoint.

==

Topic: Motion list

Right, well, as no one seemed to no my sitting down one, I have posted all that I found out. The model I used was a Bafford sergant, and the motions will change from model to model. The easiest way of checking if a motion is valid is by adding it as a motion tag and seeing if the model changes position, if it does it is valid, if it doesn't it isn't. Some seemed valid, but I could detect no discernible difference from another motion.
I lifted the conversation ones from Deep Qantas' tut, I hope he doesn't mind.
Enjoy!

WorldFrob 0, WithTray 0, AtWaist 0, BellPull 0 Frobs in front with left arm back
WorldFrob 0, AtWaist 0 Frobs in front with right hand forwayrd
WorldFrob 0, Lever Pulls a lever
WorldFrob, Door Opens Door
Stand2Sit Sits down from standing
Sit2Stand Gets up from sitting
Sit2Lie Goes from sitting to lying
Lie2Stand From lying to sitting
Crumple, Die Lies on face dead
Crumple, Knockout Lies on back dead
+OnStairs As if KOd on stairs
+IsConfined Falls to knees then back
+NearHazard It is allowable, don't know what it does.
Search, Peek Peers Forward
Search, Scan Leans forwards hunches body and turns head.
Discover, Pointout Points forwards
Discover, Recoil Recoils backwards
Discover, Challenge Points forwards real quickly.
Discover, Thwarted Shrugs shoulders in warming up kind of way.
Discover, Salute Salutes (Note discovers move only arms so variants if you play another motion immediately before hand)
IdleGesture 0 Wobbles side to side if guard (or yawns if servant)
Stalled, Poisoned Looks like he's throwing up

Stalled, Stunned Stunned routine
Stalled, Blinded Holds arms up to shield face, lots of walking
Stalled, Flail Moving arms about like a headless chicken
Stalled, Flail, InAir Can't see any obvious difference, more headless chicken
Stalled, Flail, InWater Drowning Routine (very nice endpoint)
+Rebalance Allowable though I don't know what it does.
ReceiveWound Get's knocked back if hit
(+Block& SevereWound & Attack No discenible difference on sergeant model
Special Attack)
ReceiveWound, MeleeCombat As if hit in chest, returns to on guard position
RangedCombat Whirls Hands then throws
RangedCombat, Directed A bit more discenable throw
RangedCombat, Direction Throws to one side
Conversation, Quux Cross arms behind back
Conversation, Baz Jump and shake arms a little as if agitated
Conversation, Baz, Quux Raise arms as if to say I don't know
Conversation, Bar Lean backward as if surprised
Conversation, Bar, Quux Cross arms in front
Conversation, Bar, Baz Move arms as if counting or conducting music
Conversation, Foo Raise left hand and nod as if greeting
Conversation, Foo, Baz Nod head as in agreement or acknowledgement
Conversation, Foo, Bar Take a step forward with one foot, then back
Conversation, Foo, Bar, Baz Look right
Conversation, Quux, Foo Cross arms
Conversation, Quux, Bar Keep arms crossed

ActorTags - Refine the motions in someway
Zombie
Hammer
WithSword
Unarmed
WithBow
Drunk
Constantine
Victoria
Apparition
Assassin
Ramirez
Haunt
Crumple 0, Die 0
Crumple 0, Die 0, Direction 1
Crumple 0, Die 0, Direction 2
Crumple 0, Die 0, Direction 3
Crumple 0, Die 0, Direction 4
Crumple 0, Die 0, IsConfined 0
Crumple 0, Knockout 0

Discover 0, Challenge 0
Discover 0, PointOut 0
Discover 0, PointOut 0, Direction 0
Discover 0, PointOut 0, Direction 5
Discover 0, PointOut 0, Direction 6
Discover 0, Recoil 0
Discover 0, Recoil 0, Direction 4
Discover 0, Recoil 0, IsConfined 0
Discover 0, Salute 0
Discover 0, Thwarted 0
IdleGesture
Locomote 0
Locomote 0, Direction 1
Locomote 0, Direction 2
Locomote 0, Direction 4
Locomote 0, LocoUrgent 0
Locomote 0, Search 0
MeleeCombat 0, Attack 0, Direction 0
MeleeCombat 0, Attack 0, Direction 5
MeleeCombat 0, Attack 0, Direction 6
MeleeCombat 0, Attack 0, SpecialAttack 0
MeleeCombat 0, Block 0
MeleeCombat 0, Block 0, Direction 1
MeleeCombat 0, Block 0, Direction 2
MeleeCombat 0, Block 0, Direction 5
MeleeCombat 0, Block 0, Direction 6
MeleeCombat 0, Locomote 0, Direction 1
MeleeCombat 0, Locomote 0, Direction 2
MeleeCombat 0, Locomote 0, Direction 3
MeleeCombat 0, Locomote 0, Direction 4
MeleeCombat 0, Locomote 0, LocoUrgent 0, Direction 1
MeleeCombat 0, Locomote 0, LocoUrgent 0, Direction 2
MeleeCombat 0, Stand 0
RangedCombat 0
ReadyItem 0
ReadyItem 0, ItemBow 0
ReadyItem 0, ItemSword 0
ReceiveWound 0
ReceiveWound 0, Direction 1
ReceiveWound 0, Direction 2
ReceiveWound 0, Direction 7
ReceiveWound 0, Direction 8
ReceiveWound 0, SevereWound 0
ReceiveWound 0, SevereWound 0, Direction 1
ReceiveWound 0, SevereWound 0, Direction 2
ReceiveWound 0, SevereWound 0, Direction 7

ReceiveWound 0, SevereWound 0, Direction 8
Search 0, Peek 0, Direction 1
Search 0, Peek 0, Direction 2
Search 0, Peek 0, Direction 3
Search 0, Scan 0
Stalled 0, Blinded 0
Stalled 0, Flail 0
Stalled 0, Flail 0, InAir 0
Stalled 0, Flail 0, InWater 0
Stalled 0, Flail 0, Long 0
Stalled 0, Poisoned 0
Stalled 0, Rebalance 0
Stalled 0, Stunned 0
Stalled 0, Stunned 0, Long 0
Stand 0
Stand 0, Halt 0
Stand 0, Halt 0, NearHazard 0
Stand 0, Search 0
Stand 0, Wheel 0
WorldFrob 0, AtChest 0
WorldFrob 0, AtWaist 0
WorldFrob 0, AtWaist 0, BellPull 0
WorldFrob 0, AtWaist 0, Door 0
WorldFrob 0, AtWaist 0, Lever 0
WorldFrob 0, OnFloor 0
+Action
+Activity
+Apebeast
+Apparition
+Assassin
+AtChest
+Attack
+AtWaist
+Bar
+Baz
+BellPull
+BlackJack
+Blinded
+Block
+BugDemon
+Challenge
+Constantine
+Conversation
+CrayMan
+Crumple
+Cutty

+Die
+Directed
+Direction
+Discover
+Door
+Drunk
+Facing
+Flail
+Foo
+Frog
+GLP180Turn
+GroundAction
+GroundLocoPair
+GroundLocoSngle
+Halt
+Hammer
+Haunt
+HurtLevel
+IdleGesture
+InAir
+InWater
+IsConfined
+ItemBow
+ItemSword
+KnockOut
+Lever
+Lie2Sit
+Lie2Stand
+Locomote
+LocoSpeed
+LocoUrgent
+Long
+MeleeAction
+MeleeCombat
+NearHazard
+OnFloor
+OnStairs
+Peek
+Pivot
+PlayerBow
+PlayerSword
+PlaySpecMotion
+PlyrBodyCarry
+PlyrBow
+PlyrCorpseCarry
+PlyrSword

+PlyrSwordSwing
+PointOut
+Poisoned
+PoseFrame
+PoseKind
+Quux
+Ramirez
+RangedCombat
+ReadyItem
+Rebalance
+ReceiveWound
+Recoil
+Resurrect
+Salute
+Scan
+Search
+Simple
+Sit2Lie
+Sit2Stand
+SpecialAttack
+Stalled
+Stand
+Stand2Sit
+StandWith90s
+StandWithTurns
+Stunned
+SwingLength
+SwingType
+SwordDirBlock
+SwordGenBlock
+SwordSwing
+Test
+Thwarted
+Unarmed
+Victoria
+Wheel
+WithBow
+WithSword
+WithTray
+WithTurns
+WorldFrob
+Zombie
I don't think I'm alone in wondering how exactly we use these motions. I've plugged them into the actor tag list just to see them, and used them in a Current Pose property, and I understand that you can play them in conversations, but what else is there?

Well, there is only one other place that I can think where you could use them, and that is signals.

In case you don't know, you can send a signal via the conversation or watchobj options and some of the responses.

As far as I can tell, anyone you has a response set up to that signal will go through a mini-conversation. It is quite useful for timing patrols so that one guard only starts off when he another gets to a certain point. (there is a slight added complication with this, as you have to muck about with receptrons to get the other guard to resume patrolling)

You get the general idea.

This should allow the areas in which you could use the motions to be slightly expanded, so that you could have a vomit arrow or something that temporarily forces a guard to, well, vomit (I suppose you could combine that with a particle effect.)

==

I can understand the guards not patroling until they have seen one another fairly well, but how to make a guard get suspicious after a certain amount of time completely escapes me. Have you managed to figure this out?

Well yes, though I don't think I infered in the post. I must try to be clearer.

Yeah, you should be able to do it quite easily, you could either have it so the guard get's suspicous after not seeing his friend for a certain time, or that he gets suspicous after so many of a select groups of friends gets knocked out.

I didn't really implement it in my level as I didn't really want to put my eggs in one basket. Erm, this is probably going to be complicated to explain. I don't suppose I can have a survey of who is interested in this and how much they currently know?

Well, Totality, I certainly wouldn't mind knowing this. Not a necessity at this stage (the AIs won't come in until later, for convenience) in my level design.

Even my best guesses have gaps in them. I'm only now starting to work with the trap items (a destroy trap to allow a new sword to be picked up and destroying the old one, and ending a certain special effect at the same time). I'm making good use of the things I know, thus far. A little hint to anyone having a problem with an area having just the right level of light, but with objects too dark: go to Properties, look under Renderer, and set Extra Light. 1.00 is the max and -1.00 is the min. Additive means that local light is taken into effect as well. 1.00 is full brightness. I found 0.15 to be good enough to look effective in darker areas.

==

Topic: Sit, Stand Gaurds

They are used as argument 3 of a "Play sound/motion" action in a pseudo-script (conversation, AIWatchObj link, or similar.)

Try this:

1. Create an AI.

2. Add an AIWatchObj link from it to itself.

3. Edit the data so it watches for player intrusion, never gets killed, and has a decent radius.

4. Under the first action, select "Play sound/motion", and in argument 3, type "Stand2Sit".

5. So the AI won't attack you, either add to it the M-Front Gate Guard or M-AlertCapZero metaproperties, or set its team to good or neutral. Don't use the command "aiawareofplayer" which toggles player invisibility to AI, because that also disables AIWatchObj links.

Now go into game mode and walk into the radius area of the link. The AI should sit down and then stand back up.

Well, these 'tags' are found under

Creature->CurrentPose

Set the type to motion name and then the motion to standtosit and the AI should then stay in that position. By making this a metaproperty you can take this off at will (this is how the sleeping servant is handled).

Of course, you could just do it the conversation way, and just put an indefinite wait statement at the end to force him to stay sitting.

===

Topic: How To: sitting guards

This is a gamesys modification that lets you create guards who are sitting, but jump up to attack when they see you.

Problems:

1) Sitting gaurds are still as statues.

2) The guard never sits back down after standing, but this could probably be implemented easily enough with a marker and self-entry AIWatchObj link.

Basic Idea: Create a meta-property that is added to the guard to make him sit, and removed when the guard is alerted.

Method:

1) Go to the object hierarchy. Select MetaProperties from the drop-down box at the top of the dialog. Click Add and type M-SittingGuard for the New Object Name. Click OK.

2) Click on the new MetaProperty to select it. Click Edit.

3) Add->Creature->Current Pose.

Type: Motion Name

Motion: Stand2Sit

Frac: 1.00

Model scale: 1.00

Ballistic: False

4) Add->AI->Ability Settings->Idling: Should fidget: False

5) Add->AI->AI Core->Standing motion tags: (leave blank)

Now when you create the guard himself, add this to his properties: Add->AI->Responses->Alert response.

Alert level: 3 (2 might work well, also)

Priority: Very high

Response: Step 1: Add/Remove Meta-property

Argument 1: Remove

Argument 2: M-SittingGuard

==

LAVA - From Apache's tutorial on lava
TO MAKE IT LOOK LIKE LAVA...
Use L3 or L4 for that lava look NOT "bl" or "gr" in the Texture Name box.
TO MAKE IT ACT LIKE LAVA
Highlight the flow you want to change.
Click on Properties
Click on Add
Add > Renderer > Water Flow Color Index
Set to 2
Add > Act/React > sources
When the "sources" edit box comes up.
Add.
This will bring up another window.
This has for sections.
Object = this is the number of the flow group.
Stimulus = this is the kind of effect it will have. Use this for lava FireStim.
Propagator = should be set to "flow"
Set "Intensity" to anything above 0 but not higher than 10, unless you want to kill Garrett instanty.
Then add the number for the flow again..It likes to delete it once you start adding the others.
Click on the OK button to close.
Then Click on the OK button to close the "Sources" box.
Then Click on DONE.
And there you have it..LAVA...tada
I hope this helps you out.

==

LAVA using Lava Flow Groups

1. Select the "Create" button to create a new brush.
2. Select the "Terrain" button to specify the kind of brush.
3. Drag out the Terrain brush to encompass the 3D region where you want the lava.
4. Now change the brush operation to "Fill Water" or to "Flood". If you choose "Fill Water", then the entire cubic region is changed to water. If you choose "Flood", then

only the air within the cubic region is changed to water; solids are unaffected. **Important**: Change the operation *after* creating the brush.

5. Select the "Flow" button to specify the kind of brush (a flow group).
6. Drag out the flow brush to surround the water region where you want the lava. It will create a water flow group. If you now click "Properties", then you will see that the flow group has inherited its properties from the parent "Water Flow Group"; click "Done" to get out of the properties dialog box. You need to change the parent flow from "water" to "lava" by following the next steps.
7. Select the "Edit Group" button.
8. Specify "L3" or "L4" (without quotes) in the texture name box. These are the lava names. Be sure that your texture palette is not full, because DromEd needs to load two textures, L3in and L3out, for example.
9. Click OK to get out of the dialog box.
10. Go up to the "Editors" menu and select "Object Hiearchy".
11. In the object dialog box, go up to the pull-down list box and choose "Flow Groups".
12. Expand the "Water Flow Group" by clicking on the plus-sign in the square. You should see the name of the flow group that you just created in step 6.
13. Left-click and hold on the flow group name. Drag the flow group to the "Lava Flow Group" and release the mouse.

You've now moved the flow group from "water" to "lava", thus changing it's parent flow from water to lava. All subsequent flows created from that particular flow group will now be lava instead of water.
==

VO TRAPS WITHOUT USING ROOM BRUSHES (the purple room outline)
(From Apache's tutorial on VOTraps sans room brushes)

Create the following items..
1. VOTrap -4862
2. BoundsTrig -2086

Place them where you want to voiceover to be triggered.
Link the BoundsTrigger to the VOTrap.

ID	Flavor	From	To
#######	ControlDevice	A BoundsTriggger	A VOTrap

Then link the VOTrap to a SoundDescription

ID	Flavor	From	To
#######	SoundDescription	A VOTrap	gar022 or whatever voice you want

----
Now to change the distance to the BoundsTrigger
Go into Properties
Expand Model
Attributes

Controls
Dimension
X=8   Y=8   Z=8
State
Type
The default setting is 8feet. Change the distance to whatever you want.
And thats it..No more messing with RoomBrushes and TrigRoomPlayer
NOTE
This method makes the VOTrap play everytime you run into the area.
So if you don't mind that, leave it as is. Otherwise do it this way.

Create the following items..
1. VOTrap (-4862)
2. BoundsTrig (-2086)
3. DestroyTrap (-2651)
Place them where you want to voiceover to be triggered.
Link the BoundsTrigger to the VOTrap.

ID	Flavor	From	To
#######	ControlDevice	A BoundsTriggger	A VOTrap

Link the BoundsTrigger to the DestroyTrap.

ID	Flavor	From	To
#######	ControlDevice	A BoundsTriggger	A DestroyTrap

Link the DestroyTrap to the BoundsTrigger.

ID	Flavor	From	To
#######	ControlDevice	A DestroyTrap	BoundsTriggger

Then link the VOTrap to a SoundDescription

ID	Flavor	From	To
#######	SoundDescription	A VOTrap	gar022 or whatever voice you want

----
Now to change the distance to the BoundsTrigger
Go into Properties
Expand Model
Attributes
Controls
Dimension
X=8   Y=8   Z=8
State
Type
The default setting is 8feet. Change the distance to whatever you want.

And thats it..No more messing with RoomBrushes and TrigRoomPlayer

==

VO Trap Tutorial with room brushes

(From Apache's tutorial on Votraps with room brushes)

1. Create a Room Brush where you want the sound to play upon entering. Assuming you already know how.
2. Highlight the room brush and hit 'create.' Highlight Base Room and hit 'add'. Name your room whatever you want.
3. Highlight the newly added room and hit 'create.'
4. When asked if this room is concrete. Answer YES
5. Your RoomBrush should still be HighLighted. Hit "Edit".
6. Add a TrigRoomPlayer script to the room brush by Add>S>Script and typing "TrigRoomPlayer" to the topmost field without the quotes.
7. Create a VOTrap object. From fnord>TrigTrap>SoundTrap>VOTrap (-4862)
8. Do a ControlDevice link from the Room Brush to the VOTrap.
9. ID#******* Flavor : ControlDevice From : RoomBrush To: VOTrap (-4862)
10. Look up the sound file name you want to play, for example, I used Sound:Schema:PLAYER_SPEECH:garm0404 and remember the ID# -1923. But you can just type in the name.
11. Add a SoundDescription Link to the VOTrap, and add the ID# or name to the To: field.
12. ID#****** Flavor : SoundDescription From : VOTrap To : garm0404 or -1923
13. Build RoomDataBase and Compute PathFinding DataBase.
14. Portalize.
15. All Done

==

How to: Set a slidy_door to start in the open position

1. Place the door where it would normally be when it is open. Be sure that it is properly aligned with floor, walls, and ceiling.
2. Select properties:
   - door->translating->status{open}
   - physics->model->attributes{gravity:=0.00%}
   - physics->misc->AI Collides With:FALSE

Be sure to verify that the translation axis and direction are correct. Otherwise, the door will shut in the wrong direction. If you don't set the gravity to zero, the door will fall through the floor when it is shut. When the door is shut, the gravity will revert to 100%. AI never collide with open doors, so be sure to set that flag to FALSE. When the door is shut, the flag will automatically revert to TRUE.

==

DromEd 2 Script Hierarchy (all the scripts you can use)

I believe these are all of the scripts that are available to us in DromEd 2. There are a bunch of new (and interesting) ones to experiment with!

Here is the script hierarchy for gen.osm (except for the victory check stuff which comes from convict.osm):

RootScript ActivateAmbient

RootScript AdvPickScript

```
RootScript AI
        AI CameraAlert
                CameraAlert CameraAlert2
        AI CorpseFrobHack
        AI NoPingBack
        AI QuaffHeal
        AI ReactsNoisemakers
        AI RoboSteam
        AI RobotSlayScript
        AI SummonMeBoiler
        AI SuspiciousReactions
        AI TrigAIAlert
        AI TrigBrainDead
        AI WorkerRobot
RootScript AllAIScript
RootScript AnimLight
        AnimLight Extinguishable
                Extinguishable GasLight
RootScript Arrow
RootScript Attack
        Attack AttackActivate
RootScript BlackJack
RootScript CamReturn
RootScript Clock
        Clock BigClockFace
RootScript CloneContactDmg
RootScript CloneContactFrob
RootScript Container
RootScript ContainWaterProp
RootScript Crystal
RootScript DeathSentence
RootScript DescribeSounds
        DescribeSounds AmbientSounds
        DescribeSounds VOSounds
RootScript DiffSpoof
RootScript DoFlameSource
RootScript Door
        Door SecurityDoor
        Door StdDoor
                StdDoor NonAutoDoor
        StdDoor ToggleDoor
    Door SubDoorJoints
    Door TrigDoorOpen
RootScript EatFood
    EatFood AirPotion
    EatFood HolyH2O
    EatFood TimedPotion
        TimedPotion InvisiPotion
        TimedPotion LoGravPotion
        TimedPotion SpeedPotion
RootScript ElevatorSounds
RootScript FrobFind
RootScript FrobSlay
RootScript FrobSounds
RootScript GoMissing
RootScript HolyFont
RootScript IllumeBearer
RootScript InstantiateSuspicious
RootScript Instrument
RootScript Invisible
RootScript JAccuse
RootScript JunkWebs
RootScript LastMissionLoot
RootScript Legible
    Legible StdBook
    Legible StdScroll
RootScript LightFlare
RootScript LoadoutBox
    LoadoutBox LoadoutCache
RootScript Lock
RootScript LockPick
RootScript LockSounds
RootScript LootSounds
RootScript Lugged
RootScript MapSupplement
RootScript MovingTerrain
    MovingTerrain StdElevator
        StdElevator OverheadDoor
        StdElevator PathElevator
    MovingTerrain StopAtWaypoints
```

```
RootScript NotifyRegion
RootScript Odometer
RootScript OnOffSounds
RootScript OutDamnSpot
RootScript Phantom
RootScript Physics
        Physics ActiveMine
                ActiveMine ActiveFlashMine
                ActiveMine ActiveGasMine
        Physics CamGrenade
        Physics CollisionStick
                CollisionStick DeployRope
                        DeployRope DeployVine
        Physics ConveyorBelt
        Physics Corpsed
        Physics FlashBomb
        Physics Mine
                Mine FlashMine
                Mine GasMine
        Physics MossLump
        Physics MossSpore
        Physics NoticesPlayerBumps
        Physics PhysARContact
        Physics PlayNoisemaker
        Physics ReGravitize
        Physics StdController
                StdController Gong
                StdController StdButton
                        StdButton NumberButton
                StdController StdTwoState
                        StdTwoState StdLever
                                StdLever DoubleButton
                                StdLever JumperSwitch
                                StdLever LeverNoChain
                                StdLever StdGauge
                StdController TweqOnOff
        Physics StickyWebs
        Physics TrigOBB
                TrigOBB Glyph
                TrigOBB TrigOBBPlayer
        Physics ZombieRegen
RootScript PressurePlate
        PressurePlate TrigPPlate
        PressurePlate TrigPPlateImmed
RootScript RainSound
RootScript ReloadTweqEmit
RootScript Room
        Room TrigRoomCreature
                TrigRoomCreature TrigRoomPopChange
        Room TrigRoomDeposit
        Room TrigRoomObject
                TrigRoomObject TrigRoomDelivery
        Room TrigRoomPlayer
        Room TrigRoomPlayerTrans
        Room VictoryChecker
                VictoryChecker VictoryCheck
RootScript RopeFX
RootScript Roulette
RootScript Sanctifier
RootScript SecretSounds
RootScript ShutUpYerDead
RootScript SlayFind
RootScript StdKey
RootScript StdParticleGroup
RootScript StdTerrpoint
RootScript StdTrap
        StdTrap QVarTrapTrigBase
                QVarTrapTrigBase TrapSetQVar
                QVarTrapTrigBase TrigQVar
        StdTrap TrapCapacitor
        StdTrap TrapConverse
        StdTrap TrapDeadfall
        StdTrap TrapDestroy
        StdTrap TrapDirection
        StdTrap TrapFindSecret
        StdTrap TrapLocker
        StdTrap TrapPatrol
        StdTrap TrapRelay
        StdTrap TrapRelease
        StdTrap TrapRequirement
```

```
                TrapRequirement TrapRequireAll
                TrapRequirement TrapRequireAny
                TrapRequirement TurretAlert
        StdTrap TrapRevert
        StdTrap TrapTeleporter
        StdTrap TrapTexture
        StdTrap TrapTweqEmit
        StdTrap TrigFlicker
RootScript Sword
RootScript TransformLock
RootScript TrigContained
RootScript TrigInvFrob
RootScript TrigSchemaDone
RootScript TrigSlain
RootScript TrigUnlock
RootScript TrigWorldFocus
RootScript TrigWorldFrob
RootScript Victrola
RootScript VisitDecal
RootScript Waypoint
        Waypoint StopHere
```

Here is the list in alphabetical order (note, all the scripts in the hierarchy are present – even those that were never intended to be directly callable):

ActivateAmbient
ActiveFlashMine
ActiveGasMine
ActiveMine
AdvPickScript
AI
AirPotion
AllAIScript
AmbientSounds
AnimLight
Arrow
Attack
AttackActivate
BigClockFace
BlackJack
CameraAlert
CameraAlert2

CamGrenade
CamReturn
Clock
CloneContactDmg
CloneContactFrob
CollisionStick
Container
ContainWaterProp
ConveyorBelt
Corpsed
CorpseFrobHack
Crystal
DeathSentence
DeployRope
DeployVine
DescribeSounds
DiffSpoof
DoFlameSource
Door
DoubleButton
EatFood
ElevatorSounds
Extinguishable
FlashBomb
FlashMine
FrobFind
FrobSlay
FrobSounds
GasLight
GasMine
Glyph
GoMissing
Gong
HolyFont
HolyH2O
IllumeBearer
InstantiateSuspicious
Instrument
Invisible
InvisiPotion

JAccuse
JumperSwitch
JunkWebs
LastMissionLoot
Legible
LeverNoChain
LightFlare
LoadoutBox
LoadoutCache
Lock
LockPick
LockSounds
LoGravPotion
LootSounds
Lugged
MapSupplement
Mine
MossLump
MossSpore
MovingTerrain
NonAutoDoor
NoPingBack
NoticesPlayerBumps
NotifyRegion
NumberButton
Odometer
OnOffSounds
OutDamnSpot
OverheadDoor
PathElevator
Phantom
PhysARContact
Physics
PlayNoisemaker
PressurePlate
QuaffHeal
QVarTrapTrigBase
RainSound
ReactsNoisemakers
ReGravitize

ReloadTweqEmit
RoboSteam
RobotSlayScript
Room
RootScript
RopeFX
Roulette
Sanctifier
SecretSounds
SecurityDoor
ShutUpYerDead
SlayFind
SpeedPotion
StdBook
StdButton
StdController
StdDoor
StdElevator
StdGauge
StdKey
StdLever
StdParticleGroup
StdScroll
StdTerrpoint
StdTrap
StdTwoState
StickyWebs
StopAtWaypoints
StopHere
SubDoorJoints
SummonMeBoiler
SuspiciousReactions
Sword
TimedPotion
ToggleDoor
TransformLock
TrapCapacitor
TrapConverse
TrapDeadfall
TrapDestroy

TrapDirection
TrapFindSecret
TrapLocker
TrapPatrol
TrapRelay
TrapRelease
TrapRequireAll
TrapRequireAny
TrapRequirement
TrapRevert
TrapSetQVar
TrapTeleporter
TrapTexture
TrapTweqEmit
TrigAIAlert
TrigBrainDead
TrigContained
TrigDoorOpen
TrigFlicker
TrigInvFrob
TrigOBB
TrigOBBPlayer
TrigPPlate
TrigPPlateImmed
TrigQVar
TrigRoomCreature
TrigRoomDelivery
TrigRoomDeposit
TrigRoomObject
TrigRoomPlayer
TrigRoomPlayerTrans
TrigRoomPopChange
TrigSchemaDone
TrigSlain
TrigUnlock
TrigWorldFocus
TrigWorldFrob
TurretAlert
TweqOnOff
VictoryCheck

VictoryChecker
Victrola
VisitDecal
VOSounds
Waypoint
WorkerRobot
ZombieRegen

==

How do I trigger sound on object pick up?

Create VOTrap
fnord-TrigTrap-SoundTrap-VOTrap (-4862)
Create the item you are using for the sound trigger. Open the properties and alter like this.
ADD-S-Scripts
Scrtpt0 - StdButton (unless there is something already there, then place it in the next open space)
ADD-EngineFeatures-FrobInfo
In the window that comes up there are 3 options.
1. World Action
2. Inv Action
3. Tool Action
Click on World Action and check the 'script' option. Click OK till you are exited out of the props window for that item.
Next link the object to the VOTrap like this
Flavor-ControlDevice
From-Object
To-VOTrap
Then link the VOTrap to the Garrett voice like this
Flavor-SoundDescription
From-VOTrap
To- (the name of the wave)
Thats it all done.
==

Thief 2 Garret wav files

gar0111- "Steady....steady"
gar0402- "Oh yeah...He's gone."
gar9930- "Ah, My favorite year."
gar9920- "Huh, not so secret anymore is it."
gar9921- "I wonder how I can get up there?"
gar9922- "Ahhh, back door."
gar9923- "Can't go this way."
gar9924- "Ah, no point in going here."
gar9925- "I needn't bother here."
gar0101- "I better check my map."
gar0100- "Ok Basso, you sit tight till I give the signal."
gar0418- "No thanks, not ever that hungry."
gar0420- "Well, this is a snug fit."
gar0421- "Huh, yeah right. Life is tough when you don't have anyone waiting on you hand and foot.
gar0405- "Nice view."
gar1306- "Looks like I'll be able to get one of those cultivators when I come back."
gar1307- "Looks like I'll be able to get one of those cultivators while I'm here."
gar1308- "A cultivator, that must be what Karras is writing to Gervaseous about. What a nice boss."

gar1309- "A cultivator what a nice bonus."
gar1401A- "Lotus"
gar0417- "Hey, I know who this guy is."
gar0112- "Well that's her door. Guess Basso can open it. He's not much of a sneak but he sure can handle a lock."
gar1401C- "I'm no Mechanist. I'm Garrett."
gar1401E- "Knew I would come. But."
gar1401G- "Don't worry I'll find Cavador. Karras will be stopped!"
gar1401I- "Yes my friend, what can I do?"
gar1603- "This is just what I was looking for. Now I've got a real plan."
gar0208- "Look what's on that guy's belt."
gar0209- "I hear this guy is quite a popular musician. His unpublished work should be valuable."
gar0102- "This must be the door Basso mentioned."
gar0103- "Good thing the butlers is out for the night. He left his lights on."
gar0104- "Upstairs, Basso isn't going to need to come up here."
gar0105- "Oh, better stop and check out that light around the corner."
gar0106- "This hallway is too well lit, gonna put a water arrow into that torch."?????
gar0107- "Careful."
gar0108- "Another guard up ahead. Time to knock him out while his back is turned."
gar0109- "Let's duck aside and wait for this."
gar0110- "Let's uh, duck aside and wait for this little chat to finish."
gar0218- "Only thing left for me to do, is to make my Exit."
gar0219- "If I aim right I can hit that button with an arrow and activate it."
gar0113- "Looks like the coast is clear for Basso now."
gar0201- "Shouldn't be to hard to find a spot to climb out of this place."
gar0202- "Hum, doesn't look like I can get up there."
gar0203- "A fall from this height could hurt."
gar0204- "I bet a guard or two around here may have a key for me to pickpocket."
gar0205- "This place has lots of shadows to hide in."
gar0206- "That floor looks loud. I better find a way to walk across it quietly."
gar0207- "I need to find a key for this."
gar0415- "Hagen doesn't seem too keen on Lt. Mosley does he."
gar0410- "I should be able to find something in here."
gar0411- "Mechanists, always to do-gooders.."
gar0210- "This place looks like an inventors shop. There's probably a valuable gadget or two around here."
gar0211- "Lynal makes the best steaks in the city. His recipes may catch me a pretty penny."
gar0213- "That looks like Cpt. Davidson's cargo ship. He's a shady guy, that’s into smuggling and piracy."
gar0214- "Large open door. That is bound to attract a guard or two. Should probably try to find a way to get it shut again."
gar0215- "Huh..Yo Ho Ho and a bottle of Rum."
gar0416- "I should be careful in here. There are likely a few traps around."
gar0216- "Ew..so that's Milons secret ingredient."

gar0217- "Hum..the numbers above these doors may be important."
gar0507- "Good..now time to relocate."
gar0601- "These Mechanists are just as bad as the Hammers."
gar0602- "It's not big secret that the Mechanists are the ones supplying Sherrif Turant with those machines."
gar0401- "I should take a look at those instructions."
gar0414- "I don't think I want to go that way."
gar0403- "He won't need that anymore."
gar0404- "Real classy. I guess it pays to be an officer."
gar1111- "One things for sure..this..Karras guy has lost his mind."
gar0406- "Now on to the Vault."
gar0407- "Looks like this Mosley character has got a thing for plants. I guess she can afford to."
gar0408- "Let's see what I can use against our Lt. Hagen."
gar0612- "That wasn't the right key."
gar0613- "Good thing I can still use this wax."
gar0614- "Nope"
gar0412- "I better do this quick."
gar0501- "Better get moving before they see me."
gar0502- "A sewer never looked so appealing."
gar0503- "Judging from the guard post I'll bet that's Smith's place."
gar0504- "Hum..Whipple and Helena Way. Thats where that prisoner in Soulsgate said he stashed some loot.'????
gar0505- "Home sweet Home."
gar0506- "Looks like I'm gonna need a new address. But I'm gonna have to get to my stash first."
gar0705- "I'll never be able to cross thru here with all those security machines active."
gar0706- "Maybe I can find a way to de-activate them."
gar0707- "The Vault isn't going to open until that locking bar is out of the way."
gar0603- "I could really learn to hate these guys."
gar0605- "Strange ..wonder what kind of work the Mechanists do here."
gar0606- "Better not disturb them. I can hear the meeting just fine from out here."
gar0607- "I'd like to get a copy of that key myself."
gar0608- "I wonder how I knew I was going to need this wax key press."
gar0609- "If I can find some soft wax, I can an impression of the key and make a duplicate later."
gar0610- "Huh..wrong key."
gar0611- "This is the right key."
gar0806- "Wasn't expecting that."
gar0807- "Well I'm not going to wait around for him to come out. Fortunately cemetaries don't spook me."
gar0808- "Where did he go..guess there must be another way out of here afterall."
gar0615- "Hope this is a safe place to drop this key and make a copy of it."
gar0616- "I better come back here and replace this key when I'm done with it."
gar0617- "That ought to do it."
gar0618- "I think my work here is done."

gar0701- "I wonder if any of the outside windows can be opened."
gar0702- "Looks like I might be able to climb on to the roof."
gar0703- "Some people in this city are to rich for thier own good. Lucky they have me to give them a hand."
gar0704- "Now to find some useful information."
gar1003- "I hope this guy doesn't run out of blood."
gar1004- "Looks like I am still on the right track."
gar1005- "So..it's been the Mechanists behind my problems this whole time."
gar0708- "What a weird contraption. Let's see what these controls do."
gar0709- "Hahaha..now I know I am a Master Thief waltzing right into a bank vault. Time to time the recording."
gar0801- "Lt. Mosley out for a little stroll. Let's see where you take me."
gar0802- "Somehow I doubt she dropped her note by accident. I better see what it says."
gar0803- "Looks like the good Lt. is working with the Pagans."
gar0804- "Lets see who comes to claim the note."
gar0805- "A cemetary..huh..that's original."
gar0413- "Letting them out would spoil the plan."
gar1109- "So Karras doesn't even show up for his own party."
gar1110- "If I follow this road north it should take me right to the Mechanist Tower."
gar0901- "Hum..looks like a gear is missing."
gar0902- "Theres probably something in this room that will tell me who killed Truart."
gar0903- "Dam..someone beat me to the Sheriff. I better keep a low profile or else I'll be pinned as the killer."
gar0904- "I better get out of here now."
gar0905- "Truart doesn't do much to hide his vanity."
gar0906- "If I knew being Sheriff paid that well I would have changed careers along time ago."
gar1001- "Hum..that letter the Pagans carring. It's going to be late."
gar1002- "These people didn't have a chance against the Mechanists."
gar1301- "Ah early morning mist, all respectable people should be in bed now. Time for us disreputable types to get to work."
gar1302- "So this is what all the fuss was about. That old braggards made quite a spectable for his babbles.
gar1303- "I'll have to grab a bottle and later drink a toast to Lord Grivasues for his generouslity."
gar1006- "Is this the Maw..looks different."
gar9928- "Ah..a hidden entrance."
gar1102- "That wasn't so tough. Maybe Tower security isn't as tight as I thought."
gar9929- "This will make a nice little bonus."
gar0809- "Damn, I guess I lost the trail."
gar9919- "I can't stay underwater to long or I'll drown."
gar0409- "Guess I could use some target practice."
gar1107- "The front doors are locked. Looks like Karras wants a captive audience."
gar1407- "The wheel seems to be missing a peg."
gar1408- "What the!!!"
gar1409- "This should fetch a nice price."

gar1201- "Better find a way in before I catch cold."
gar1202- "Ah..storage. I'm sure Grevasous has lots of things down here that he'll never miss. At least not until it's to late."
gar1203- "Hum..I may have to stop in here for a drink later."
gar1204- "Maybe I'll stop in for a little light reading."
gar1205- "Either someone's hiding the evidence or these guys got lost in 10 feet of passage."
gar1206- "No wonder the spooks are restless around here."
gar1207- "Ash you are a bastard."
gar1508- "To bright for an ambush here, I perfer one of those dark corridors."
gar1601- "Yeah, just keep pratling away Karras."
gar1602- "We'll see how well your protective chamber works against your own rust gas."
gar1304- "Right..thats the last one."
gar1305- "Humm..another Mechanist, guess the Hammers just aren't chic anymore."
gar1401- "Hmm..that most be the Seatus Amacus, guess I'll sneak aboard and have a look around."
gar1402- "Tight fit, I hope it's a short ride."
gar1403- "If I were Cavador where would I be."
gar1404- "Burrr..I wonder what they keep on ice."
gar1405- "Hum..doesn't look very dangerous. Good thing I know better."
gar1406- "It's a long way up."
gar9901- "Not gonna find what I'm looking for this way."
gar9902- "I need to find a better way than this."
gar9903- "I bet this is the right way."
gar9904- "I think I'm on the right track."
gar1501- "I'm back in the Lost City..ah..I should have known. Guess I'll take the back door home."
gar1502- "Ah..knowing where Cavador is, that's half the battle."
gar1503- "What does he eat, metal rivets.?"
gar1504- "Looks like my old route is still open."
gar1505- "This guy makes the Hammers sound sane."
gar1506- "Archaeologist sounds so much more dignified than Thief."
gar1507- "Guess things have changed a bit."
gar9911- "Time to go."
gar9912- "Now to be on my way."
gar9913- "Time to get out of here."
gar9914- "This doesn't look safe."
gar9915- "This looks like a dangerous area."
gar1604- "Hum..there must be instructions on this thing somewhere."
gar1608- "Karras didn't pipe up about the sabotage. I don't think he knows."
gar1605- "Assuming the blueprints didn't leave anything out. This thing ought to do the trick."
gar1606- "Now I just have to switch the signal towers to use the beacon instead of Karras's instructions."
gar1607- "Well if Karras has anyway of detecting what I am doing I'll know soon."
gar1104- "Hey the city looks almost bearable from up here."

gar1105- "Careful"
gar1106- "No need for alarm ladies, just passing thru."
gar1108- "So, that’s the castle of the future. I'll take my tenement any day of the week.'
gar9927- "This looks like the way in."
gar9905- "That should do it."
gar9906- "HaHaHa..I think I deserve a pat on the back for that one."
gar9907- "That was helpful."
gar9908- "Opps."
gar9909- "Damn! I didn't mean for that to happen."
gar9910- "Argh that hurt."
gar9916- "Better watch my step around here."
gar9917- "This looks like a hard place to sneak thru."
gar9918- "It would be tricky to get thru here undetected."
gar9926- "Nothing back here I care about."
gar1101- "Well well house breakers. how quaint."
gar1103- "Uh oh..time to get the goods and get out of here while I still can."
GAR0419- "I should write that down...Now on to the vault."
GAR0212- "Germ is the man that makes those magic lens. There are probably some valuable lens in here.'
===

This is a goal list for thief 2

GOAL_STATE_n, x 0 = not complete, 1 = complete

GOAL_VISIBLE_n, x 0 = not visible, 1 = visible

GOAL_TYPE_n, x 1 = steal an object, 2 = kill person, 3 = steal loot, 4 = go to location

goal_target_n, x x = (number) of item

GOAL_MIN_DIFF_n, x 0 = normal, 1 = hard, 2 = expert

GOAL_MAX_DIFF_n, x 0 = normal, 1 = hard, 2 = expert

GOAL_OPTIONAL_n, x

GOAL_REVERSE_n, x x = 1 = true reverse the goal_target, eg kill guard, would be, do not kill guard

GOAL_SATISFIED_n, x

GOAL_BONUS_n, x 1 = not visible

GOAL_SPECIAL_n, x (goes with bonus) 1 = true

GOAL_SPECIALS_n, x (goes with bonus) 1 = true

GOAL_FINAL_n, x

GOAL_IRREVERSIBLE_n, x

goal_gold_n:

GOAL_GEMS_n, x x = total value of gems

GOAL_LOOT_n, x x = total value of loot (gold, gems, items combined)

MAP_MAX_PAGE x maximum map page

MAP_MIN_PAGE x minimum map page

sometimes you have to restart dromed2 to start getting results as it store some infomation in a buffer and it doesnt restore the information in the buffer if you change it. The only way to change it is to save your level, exit dromed2, run dromed2 again, then load your level.

If you are using questvars objects to get a result you also have to add a script/trap control/once to the object property to make it trigger

---

Making A Guard Face Different Directions

1. AI->AI Settings->Idling: Directions
   - Min. Facing switch time: 5000 (milliseconds)
   - Max. Facing switch time: 8000
   - Facing 1: Direction: 0 (degrees – facing south)
   - Facing 1: Weight: 1 (positive value)
   - Facing 2: Direction: 180 (degrees – facing north)
   - Facing 2: Weight: 1 (positive value)
2. AI->AI Settings->Idling: Should Fidget: TRUE

==

Interesting Scripts in Undercover Mission (#10)

On the restricted doors, there is the SuspiciousToFrob script.
On the gold hammer at the entrance, there are SuspiciousToTake;GoMissing scripts, and the FrobInfo{Move,Script;[None];[None]}.
These scripts apparently only work in miss10, because it has these custom scripts.

==

Set BigFloorLever to start in ON position controlling an OPEN door:

- Shape->Joint Positions:{60.00; 0.00; 0.00; 0.00; 0.00; 0.00}
- Tweq->JointsState:{Reverse; [None]; On, Reverse; [None] ; [None] ; [None] ; [None] ; [None]}
- Links: ~ScriptParams: TO(door)
- Links: ControlDevice: FROM(door)

==

Make an immovable barrel

Edit properties:
Physics->Model->Controls {Location, Rotation;….}

==

You can reskin individual AI mesh without overriding the texture for all using that model, and without creating special models as I did for D1.

Edit an AI's properties. Add Renderer->Mesh Textures. You get a list of texture names, blank at first. You type in the name of the texture specified in the mesh .bin file under "texture in model", and the name of the texture to replace it with in "replace with". If the mesh uses more than one texture (I don't know of any yet) you can remap other ones on the same screen.

Include the .gif extension!

To find out what to put in the "texture in model" field:

1. Edit your AI and find out what model it uses (look under Shape->Model Name.)

2. Extract this file (with the .bin extension) from mesh.crf.

3. Edit the file and look for the .gif name near the beginning of the file. Use a hex editor if you have it, but it doesn't matter if you open it in a text editor as long as you can see the .gif filename, since you don't need to edit it.

Unfortunately, this does not work for objects that get their models and skins from obj.crf.

==

This tutorial demonstrates how to retexture a brush ingame. Only the specified texture will be changed.

To begin, we need to create a marker, or any other object with no scripts - such as a table - nearby the area you want to retexture.

Let's rename it 'TexChangePoint'

Add the EngineFeatures->Retexture Radius property, and enter the the maximum distance from this object that textures will be changed.

Next, we have to give it the script 'WindowShade'. Add S->Scripts and enter 'WindowShade' in script 0.

Now we have to add both Script->TerrReplaceOff and Script->TerrReplaceOn.

Now, load both the textures you wish to use (the texture to be changed and the texture to replace it with) by using either the 'load_a_texture' or 'add_family' commands.

Change Script->TerrReplaceOff to the name of the texture you wish to replace, for example 'fam/core/blustn'. To find out a textures name, open the texture palette and click on the appropriate texture. The texture's name will be displayed on the status bar.

Finally, change Script->TerrReplaceOn to the name of the texture you want the first texture to be replaced with.

The only other thing that remain to do is to create a lever or similar object and give it a ControlDevice link to the newly created TexChangePoint.

==

Reskinning AIs by Genocide - http://www.3dactionplanet.com/thief/dromed/tutorials/dual/reskinning.asp

The fine art of reskinning. This guide explains in detail reskinning AI with Corel Photo Paint & Paint Shop Pro, more programs may become available in the future but these are the most commonly used.

Contents:

- Corel Photo Paint
- Paint Shop Pro
- Troubleshooting
- Palette Downloads

Skinning AI with Corel Photo Paint

This was written according to Corel Photo Paint OEM version 7 on Windows 98 o/s. There should be little difference when using older and newer versions.

Step 1: Open the Mesh.crf from the Res directory in your Thief 2 folder or main directory in your Thief 1 folder with an unzipping utility such as WinZip and choose one of the skin .gif files.

Step 2: Extract the skin to any local folder.

Step 3: Open the skin with Corel Photo Paint by double clicking the icon or running the program and clicking open under 'File' in the menu toolbar.

Step 4: Once opened, edit the skin as desired with the Corel effect and paint tools.

Step 5: Convert the image to paletted with the corel thief palette which you can download from our server here. This is done by selecting 'Image' from the menu toolbar, then selecting 'Convert To ->' in the drop-down menu, here you need to choose 'Paletted 8-bit'. In the paletted options that pop up, choose 'Custom' & 'Error Diffusion'. In the window that pops up, load the corel thief palette and click ok.

Step 6: Once the skin has been converted save it in Thief 2->Mesh->txt16 for Thief 2 OR Thief->Mesh->Txt16. Make sure the M in Mesh is capitalized.

Step 7: Load the editor and open your level, create the creature that you changed the skin of and you will see the results of your reskinning. If changed a creature that is not in the hiearchy you will need to click 'properties' in dromed, then Add->Shape->Model Name->'X', where 'X' is the bin file name of the creature (without the quotes).

Skinning AI with Paint Shop Pro

This was written according to Paint Shop Pro v6.02 registered download on Windows 98 o/s. There should be little difference when using older and newer versions.

Step 1: Open the Mesh.crf from the Res directory in your Thief 2 folder or main directory in your Thief 1 folder with an unzipping utility such as WinZip and choose one of the skin .gif files.

Step 2: Extract the skin to any local folder.

Step 3: Open the skin with Paint Shop Pro by double clicking the icon or running the program and clicking open under 'File' in the menu toolbar.

Step 4: Once opened, edit the skin as desired with the PSP paint and plugin tools.

Step 5: Convert the image to paletted with the psp thief palette which you can download from our server here. This is done by selecting 'Colors' from the menu toolbar, then selecting 'Load Palette' in the drop-down menu. In the load options that pop up, select the palette from the local folder you

extraced it to, and by preference you can choose 'Nearest Color Matching', 'Error Diffision' or 'Maintain Indexes'. You may want to try all three to see which one works best with your skin (I prefer and recommend Error Diffusion, but sometimes Nearest Color Matching works best).

Step 6: Once the skin has been converted save it in Thief 2->Mesh->txt16 for Thief 2 OR Thief->Mesh->Txt16. Make sure the M in Mesh is capitalized.

Step 7: Load the editor and open your level, create the creature that you changed the skin of and you will see the results of your reskinning. If changed a creature that is not in the hiearchy you will need to click 'properties' in dromed, then Add->Shape->Model Name->'X', where 'X' is the bin file name of the creature (without the quotes).

Troubleshooting

The skin works but the colors look like a rainbow.

A: You did not convert it with the Thief Palettle.

The skin works but there are holes in the AI that I can see through.

A: The first color in the palette is the transparent one, in both my psp and corel palettes for download on this site the color is neon green, if you used neon green it will be transparent in game mode. If you would like to use a different color such as hot pink for the index color, edit the palette in the program, or contact me.

The skin does not show up at all.

A: You did not put the skin in the correct folder.

The skin is in the right folder but it still does not show up.

A: You may have not saved the skin as a CompuServe Bitmap (.gif). Make sure you did so.

Palette Downloads

Thief Palette w/ Transparent Neon Green for Corel Photo Paint (.CPL Corel Palette File)

Thief Palette w/ Transparent Neon Green for Paint Shop Pro (.PAL Palette File)

Credits

This guide was written and tested by Genocide. Please email all suggestions, comments, and questions.

==

Thief-TheCircle.com -- TEG Guides

Dark Engine Level Design - Stepwise Refinement Methodby VK-GayleSaver

All information in this document comes from my own experience, unless noted otherwise by references to previously created missions. DromEd was designed by programmers, and whichever way you turn, traces of that aphorism are visible. Therefore, in my view, the best way to design levels in DromEd is to follow a top-down strategy, partially based on Tim's Building Principles document; you should be familiar with it before you read this. I've developed a set of steps that should be taken in designing and shipping a Dark mission. Please be aware that this document does not cover artistic or general aspects of level design, such as fiction composition; however, these aspects are mentioned at appropriate locations. This document is best read from start to end as a tutorial, then used as a reference document - it does not implement a step-by-step strategy for level design - it simply discusses it; several Steps refer the reader back to previous Steps for additional information. Also be aware that you not only need to carry DromEd knowledge to this document, but also a bit of common sense. There are many things that are never explicitly discussed; these things can usually be deduced easily. I can't put the entire philosophy of level design from the beginning of first-person-perspective game into a ten-page document.

Pre-step: Plan your mission

You must make several important decisions before you start creating the mission. The first thing you need to think about is what kind of level it will be. If it is going to be a progressive mission in which the player must get from one point to another, as Thief 1's "Assassins" or Thief 2's "Ambush" and "Courier", it is beneficial to carve the mission from the solid world which starts the process in DromEd. If it is a centralized mission in which the player must infiltrate an installation, as most missions are in both games, creating large air brushes for world space is preferred. Please see Tim's document for details. Before building, you must also decide for yourself what the level should look like when it is done. The reason for this is consistency. Have you ever tried writing without deciding on both a beginning and an end for your composition? It is easier to wander off the main design path than you may think. A general image in your mind is all that is necessary, but a design on paper can help. When you have completed this planning, you can begin the building process. There will be much more design work to do ahead, and stopping to think after each Step can help you polish your levels. Also, this document assumes that the task of adding story to your mission is separate from building the mission, and is therefore not discussed here.

Step One: Create world space

The first step in my refinement method is the creation of world space using large air brushes or smaller brushes of specific shapes. If your level is centralized, it is easy to build world space for it. The dimensions of an entire-world air brush may differ, but the generous estimate is 80 feet in height and no more than 200 or 300 in either direction on the x-y plane. If you need more space than 300, you should create separate air brushes adjacent to the primary air brush. The reason for this is Dark's limitation; a 300 by 300 air brush may or may not work, and it is better to be safe than sorry. When texturing this air brush, you should choose a texture for the floor that you believe will best serve as the "outdoor" texture for this level; if you need a different texture for an area you can always carve another air brush inside the primary brush. The ceiling should generally be the sky. After you believe you have planted your world space for a centralized mission, you need to build your main building(s):

1. Create solid brushes of specific shapes and sizes.
2. To prepare a rough draft of your level's space.

As stated in Tim's document, try to use Grid Size 14 for this initial stage. Using any grid size smaller than 14 violates the principle of stepwise refinement, and can cause headaches when you build your interiors. Try to use basic shapes; if you need to aesthetically improve your buildings later on, you can use air brushes to carve out a façade. Leaving architectural improvements for later can also assist in keeping polygon counts low and will allow you to focus entirely on optimization when you begin making complex scenes. A final note: you can leave some room on one side of your building for the outskirts of your mission (for example, streets and other buildings). By making these outskirts before at least Step Two, Part Three, you could very well be locking yourself into a limited space.

In my limited view, this step is slightly different for progressive missions. In progressive missions, air brushes must be used to carve corridors inside the world. Therefore, you should begin with Step Two almost immediately, and consider the solid world to be your main building.

Add the following links:

For centralized missions, you will find yourself backtracking into this Step from Step Two. This is normal; the two steps are related closely and should be integrated. You may need to jump back to this step when resizing buildings to gain more space, or creating space for a new room if you decide that one is needed. Compare this to revising an essay; your final draft is based on your rough draft rather than written from scratch. Portions of Step One are really just a draft for Step Two, and you may have to edit them when you decide on something more final.

Resist the urge to add details or change the grid size. Details are hard to notice and easily forgotten due to the amount of them per mission, opening the door to the possibility of un-textured architectural quirks.

In addition, details can force you to build in a confined space. It is not necessary to texture during this Step. In fact, as I see it, texturing before the end of Step Two is a waste of time, as it breaks stepwise refinement. As long as no fine details are added, it should be easy to do all the texturing of your rough cut at the end of the second Step.

Step Two (Part One): Design the rooms in buildings

It is now time to make rooms inside your building or buildings, or, for a progressive mission, create the world. If you are creating a progressive mission, keep in mind that it is best to choose a base plane for your streets and stick with it. You should attempt to stay relatively close to this plane. Jumping in height while creating progressive levels makes them, in my opinion, less editable in the visual brush list. A progressive mission designer's Step Two (or Step One, for him) is considerably easier than a centralized mission designer, because all his world brushes are the same height. Effectively, he creates alleys and uses the walls of those alleys in place of buildings. In one sense, however, making progressive mission worlds is harder than making interiors for centralized missions: space management. In centralized missions, most interior space represents rooms with (relatively) thin walls. In progressive missions, on the other hand, the "interiors" are corridors with large gaps between them. One city's design might help you as you build your progressive space. Remember Manhattan? All the streets run at right angles to each other. Building progressive missions in a way such that the world brushes are perpendicular will ensure that when you continue onto Step Three, you have enough space to create niches and other eye candy in the building walls. For centralized missions, this Step is different, primarily because it often flows back into Step One. Your task in this Step is very simple: you must create the interiors of your buildings. Tim's document details many interesting points on how to build interiors, so I will instead focus on what kind of interiors to build. Remember that walls should be 3' thick. As you create your interior for the building, be it a mansion or a cave, think about the primary functions of the building. You need to make sure that rooms are big enough to serve the function you want them to serve and that they are shaped the way you want them to be shaped. If you need to, go back to Step One and change your building's external form in order to add or remove additional or unnecessary space. Generally, details should be saved until later; however, there is one notable exception. As you build unusual constructs, such as overhanging floors, make sure you add any supporting architecture to it, such as wooden beams or columns. When you add this architecture (make sure that it is indeed supporting the construct in question), you should texture it right away. Texturing guidelines are detailed in Step Two, Part Two.

When creating rough drafts of stairways, apply the Pythagorean theorem and trigonometric functions. Use this formula and its transformations or variations to help you:

```
Ns - number of stairs

h(x,y) - Pythagorean formula of x and y

SWh - stairwell height (selected by designer)

SWw - stairwell width (selected by designer)

STw - stair width (selected by designer).

NOTE: Stair height should be .75

     Ns ~h(STw, 0.75)

     ------------------

     h(SWw, SWh)
```

In this Step, you should prepare the space for the stairway by creating a solid wedge of the desired height and with. Even if you are going to make a stairwell without using the wedge, it can be a helpful guide. When selecting the stairwell height, remember that on average, each stair should be .75' high. In Step Three, you will fill in these wedges with actual stairs. Special stair combinations such as spiral staircases will not be discussed, but you can look at some original Thief missions to see how they were done.

If you really want to do it, you can carve out doorways in this Step, although that is not strictly necessary. As you build your interior, keep in mind that you're not finalizing anything yet. In Tim's words, "If any of the objects in your rough cut are sized such that they would be difficult to reduce in size, you probably haven't allowed yourself enough room..."

Think about room brushing. You will need to room brush quite soon, so try to affirm the general structure of your level into your mind. Design carefully, not haplessly. Keep in mind the room brushing tips from one of the DromEd tutorials.

Keep the AI in mind. As soon as you start inserting AIs, they will be assigned positions to guard and patrol routes, not to mention the ability to chase the player through most of the level. Try to remember Tim's info on the actual limitations of AI movement through space. Remember that as you add details in Step Three, spaces are going to get more "cramped". Before you finish, remember that now your large space is clearly defined in a centralized model. This may be a good time to go back and repeat Step One on the outskirts of your mission. When you perform following Parts and Steps, you should start with the outskirts and move into the center of your mission.

Step Two (Part Two): Texture your exteriors and interiors

In this Step, you return to the very start of the mission (whether it be the outskirts, for a centralized mission, or the start location, for a progressive mission) and begin your texturing work. For progressive missions, I would suggest picking your textures with the assumption that a lot of overwriting is going to be happening as you carve details out of the walls of your sky-corridors. If your mission is centralized, start on the outskirts and apply textures to solids as necessary. This is also the time to vary the shape of your buildings slightly and texture these shape variations (an action that takes your partly into Step Three). This may prove useful later when you discover that if you had not done so, you would have had some trouble manipulating textures intuitively. As you add these variations (such as arches, domes, balconies, and so on), however, ensure that your grid size remains 14, to maintain consistency with the rest of the building. You should add the finer details in Step Three. Use the "default texture" feature (by selecting it in the PnP and using Alt-T -> Put On Brush. It'll really save you some headaches when you need to copy similar objects.

When you are done texturing the exterior in your level, you can begin the interior. My personal advice is that you do the interiors as if you were wallpapering or painting them. When you plan the textures for the floor, also keep in mind what kind of surfaces you are plotting - hard or soft. As Tim mentions, you don't want to leave the player stranded on a hard surface unless there is a specific reason for doing so.

A personal aesthetic tip: go back over your entire level rough draft and make sure that every, and I mean every single texture makes sense in its context. Nothing can ruin a well-designed mission more than a few haplessly chosen textures. The reason for this probably lies at the core design of 3D-game textures. If a texture is applied incorrectly, it loses its three-dimensionality. The player's mind no longer comprehends the texture's role in the surroundings. The natural reaction is the same as invoked by a bad painting - the texture becomes flat, boring, or sometimes even utterly revolting. While you are doing this inspection, I also suggest rotating or aligning any textures gone astray. If a wooden platform changes its length-to-width ratio in a different direction, the wood texture on that platform should be rotated to match the pattern used on its previous section. In other words, if a non-uniform texture runs in one direction on a surface, and that surface changes relevant heading, the texture should change heading in the same direction as the surface.

Step Two (Part Three): Room brush the level

Room brushing is an interesting topic, and it has often been approached from a designer standpoint. In my view, novice designers cannot be expected to understand a generalized explanation of what a room brush is. Mathematically, room brushes are easily defined: they are abstract three-dimensional parallelepiped shapes that define the nature of sound playback and propagation for the Dark Engine. Room brushes have other functions as well, such as defining areas on the automap. When you room brush, you must follow a set of specific mathematical rules that I will not explain here (since I am not familiar with all of them myself). This section will instead deal with integrating room brushes into your general architecture.

When beginning to room brush, you need to think about sound. Generally speaking, every operation brush that represents a room should be room brushed; however, room brushing does not limit itself to actual rooms. Creating room brush intersections in small spaces can be very important for sound to propagate correctly. Whenever you want to isolate an area by means of a block of solid, you need to create separate room brushes. Remember that all room brushes are prisms. This means that unusually shaped rooms will no doubt be difficult to room brush. Be wary of room brushes extending far past the floor or ceiling - intersections of room brushes in a horizontal (flat) plane can result in weird effects, and should be used only if you have a crevice for the player to crawl through, such as a well opening.

Room brushing unusual spaces can, as noted elsewhere, take some guts. Think about some mathematically annoying constructs that get in the way of normal room brushing. Broadly, any operation brush with its center not in a horizontal or vertical plane is going to take some time and effort to room brush. The key is to intersect the room brush prisms in such a way that they don't intrude into one another. Remember that you can rotate room brushes, too. If the transition (opening) is between two surfaces that lie in non-parallel planes, you can use a small, banked connector brush that intersects both large brushes.

In a centralized mission, room brushing the outskirts can be tricky. Do not encapsulate the level into a giant room brush. Rather, place room brushes around the solid structures in the outskirts and intersect them in the alleys of space between the structures.

Room brushing should be tested as you go along. However, you have no way to test a room brush configuration if there is nothing to guide you. Start the player object in the area you want to test and drop a boulder in the room right outside the player's starting location. Repeat this process for other rooms - and make any necessary corrections - until you're sure that you've got everything the way you want it. After completing this trial, you need to return to the beginning for one last time and repeat this trial with an AI. Give the AI a property that prevents him from harming you. Since you are working with sound design, now might also be a good time to consider changing soft and hard surfaces around if any of them may harm the gameplay.

You may need to come back to room brush again during Step Three in case you build smaller openings that could not be constructed with grid size 14.

Step Three: Add architectural details

Architectural details create a visual mood for the mission. After this Step, your mission's space will be finalized, and you will be free to add lighting and objects. This Step is arguably the most controversial. It is very easy to tread too lightly in this Step, resulting in a level that seems to have been built correctly, yet in the end turns out bland and boring. It is also easy to overdo detail insertion, which can cause high polygon counts and cramped spaces that ruin the atmosphere. Since no other public document explains detail addition, I will attempt to relay my own knowledge to you. However, before you proceed, ensure that you read the "Gameplay" and "Stealth" sections in Tim's documents. You will be utilizing the information presented there extensively throughout this Step.

The aim of adding details is simple: to set the level's mood and make it more interactive. There are two main categories f details you will add during the course of level construction: architectural atmosphere and world objects. Both kinds of details are important to atmosphere, each in its own way. It is these details that make Thief's world unique among the countless 3D shooters and adventure games out there.

Architectural design is an art, and most of that art is exhibited in Step Three. When you made your building exteriors in Step One, you defined a generic shape for your buildings as per your plan. It's time to crank grid size down to 12 and begin creating the details. Assuming you followed Tim's advice and your sizes are big, round numbers (a side effect of using Grid Size 14), aligning details should be a snap. Begin by looking at real-world architecture and/or original Thief 1 FMs to examine how different buildings were constructed. Then, create your building decorations. Begin with functional details - anything the player will interact with. Then, move onto atmospheric details - change the shape of your building to be as interesting as possible without overcomplicating it. As you add these details, examine them closely from in game (preferably with gravity off), taking all the positions from which the player will have access to the scene. You should ask yourself questions regarding what you see. Is this a realistic scene? Will the player find it boring from one of the viewpoints? Are there any other viewpoints that can be added (by using windows, ledges, and simple raised terrain) to make this more interesting? Are the functional details useful in helping the player or are they just there, in the way? Are the atmospheric details successful in achieving their intended effect? To answer several of these questions may require that others play your rough draft. Please remember that this is the "no turning back" Step. It will be very hard to fix any errors in your space design after you have completed it. Add experimental lighting (using light brushes) to gain the ability to add play-testing AIs to your space and practice running away from them, slipping past them, and fighting them.

After you have created your architectural space and think you have it the way you want it, you enter the most difficult phase of design. This phase will integrate with Steps Four and Five, so do not overwork yourself lest you find you have to rid yourself of whatever you produce. Creating world objects using brushes is similar to adding architecture: it enhances the game world's atmosphere and it provides the player with interactivity (the principle that distinguishes games from virtual museum tours). Although I have not mentioned it explicitly prior, you should think about story throughout the creation of your mission (whether the story is yours or someone else's). This is where that story becomes important. As previously, work with the story- or game-relevant details first. Make sure that each object looks the way it is supposed to look and serves the purpose it is supposed to serve. Add AIs and experiment with their properties to make them fit the space they are navigating in. Begin planning AI patrol routes. When adding objects for atmosphere or effect, try to place them so the shadows they cast from lighting are interesting and provide a realistic tone for the setting you are trying to convey. Remember that not only the books, scrolls, or even AIs talk to the player about your mission. The mission itself - its architecture, its sound, its mood - is at least as important (arguably more, since in all missions the majority of the player's time is spent looking at your space, regardless of the mission's size or nature) in establishing atmosphere as the aforementioned objects.

Recall that this is where things get unimaginably complex. This and the next two Steps are often entangled in a web. When planning early AI design, think about visibility. To explain what I mean, I will use concrete examples. In Lord Bafford's Manor, there are two "watchtowers" to discourage the player from attempting invasion via the front door. These archer guards are located at forte points that put them at an advantage against anyone who is between the two towers and attempting to go upstairs. However, the guards can be easily dispatched from their backs - as long as the player douses the torch inside the tower to prevent the guard in the opposite tower from seeing what he just did. Right there, in that mission - in that very spot, in fact - is a design flaw that I discovered while playing the Thief demo. If you get the attention of the three guards at the front gate - even by attacking the guard in the opposite tower with an arrow - they will come running to search for you. From that point, killing the three guards is a simple matter of patience, because their sound-based search algorithms do not account for elevation - they will search for the player under the watchtower, and each new arrow the player puts into a guard will simply reset the alert timer on him and his brethren, giving the player plenty of opportunity to knock off all three guards. To avoid such situations, place light brushes in functionally important spots and set them to the brightness that is appropriate for that location, then play-test every visible viewpoint from which the AI can see the player or, more importantly, the player can see the AI. Ensure that all your objects and architectural details provide for these viewpoints correctly. Consider extinguishable lights. Mark points where lights must not be extinguishable with POST_IT markers (special numbered markers with Design Notes on them that I like to use to remind myself of things to do later). Take any other steps you feel are necessary to ensure that your objects serve their intended purpose.

Step Four: Lighting

Lighting is an under-discussed subject. Whereas it may seem that lighting is a natural (and mundane) topic, reality dictates that incorrect lights may severely damage a mission's polish. The reason for this is both the aesthetic value of and the many functions served by light in mission design. Older games, such as Duke Nukem 3D, required the designer to do his own lighting work. It is important to realize that a texture will look better if lit correctly, and so will architecture. The lights in a mission must be placed after careful consideration. Let us review what the psychological effects on the player from different combinations of lighting are.

Load a Thief mission in DromEd and turn on "light_bright". Play the mission as if it was built that way. What effect does it have on you? When lighting is relatively monotone, it results in a mental reaction that causes the player to believe that all the textures and architecture in the room have equal weight. This may not seem to be an issue in Thief level design, but how many missions have you seen with boring, unimaginative lighting, or lighting that did nothing to the mission's atmosphere? Mono lighting should only be used when you want to emphasize an area or when you need to make a "hot zone" that the player should not be able to easily cross.

Binary lighting, mentioned in Tim's document, is one of the most common types of lighting in Thief. The reason for this is the effect binary lighting creates on a player's mind and on gameplay. When there's a clear distinction between light and shadow, the player knows where he can hide right away. As for atmosphere, binary lighting removes the ambience effect, and thus reassures the player, since his visibility will drop as soon as he leaves the light. It is also easier to avoid binary lights. In a normal situation, the best way to create binary lighting is by placing the light source into a wall or ceiling, which causes a spot light to be generated.

Ambient-radius lighting is also a common type of lighting. It is useful when you need to light a long hallway and want to make it interesting for the player. Recall Thief Gold's "Song of the Caverns". The hallway with the arguing male and female stars and patrolling archer provides a good example of ambient-radius lighting. To create such lighting, simply place light sources close enough to each other so that the edges of their radii intersect.

Pressure lighting is less common. I use that term to refer to lighting that causes the player to feel tense, such as the casino in "Thieves Guild". Consider setting up lighting that will cause the player to flinch and put away his weapon, then surprising him with footsteps. The effect of pressure lighting (when combined with AI activity) is usually to alert the player. Do not hesitate to use this effect to liven up a mission. Creating such lighting usually involves some acrobatics: you need to put time into deciding where in your space you can put the tense spots, and place lights that affect the radii you want them to affect. If you would like to leave behind the radial form of standard lighting, use ambient lighting tools such as OmniLightPoints and light brushes.

There are other types of lighting combinations, of course, but for clarity's sake I will let you discover them on your own. Instead let us turn to the issue of AI interaction with lighting. Remember that lighting determines the player's visibility. You need to build a lighting system that maintains atmosphere while keeping the player both in sight and seeing. For this, to your aid come ambient lighting tools. Light brushes and the slightly more flexible OmniLightPoint and SpotLightPoint provide you with the opportunity to create your own light - whether it be from textures, as an extension to physical light objects, or simply ambience. You should never abandon the player. Resist the desire to rely on flares in Thief 2 to provide a path; flares were created to allow players to examine the world more closely (there's a chance the player didn't buy any or he doesn't have any left).

A nice touch is to add additional sources of lighting for each higher difficulty level. Or, alternatively, you can change extinguishable sources into non-extinguishable sources. Simply ensure that you allow the player to solve a lighting problem in accordance with the difficulty level that he is playing.

For your lighting scheme to be effective, you need to try and combine realism (don't put a burning torch in a cave or a mushroom into a manor unless you have a reason), atmosphere (create shadows that will change boring textures into interesting ones), and gameplay (never leave the player stranded; play on his nerves by changing his visibility as he walks by active spots). If you are capable of doing this, your mission will undoubtedly rise above those that pay little attention to or disregard meticulous lighting.

Step Five: Add AIs and script your level

It will be a certain cliché if I were to tell you that Dark's AI is more advanced than that of any first-person game to date. Thus it is redundant to convince you of the importance of utilizing that power correctly. In this section, I will discuss several thoughts on AI not already mentioned or detailed elsewhere.

When placing an AI, be sure you understand its role. Also, consider balance. For the easier difficulties, the player has an opportunity to confront an AI. Keep that in mind as you decide what AI to put where. As you place AIs, also remember realism. If you place a guard into a forbidden room, the player has to know why he is there. Burricks walking around a bedroom might cause some puzzlement as well. This point is sometimes overlooked, resulting in sometimes amusing but more often simply annoying scenes. In addition, remember that AIs in Thief are not cookie-cutter products. Each guard poses a puzzle for the player to solve in addition to helping move the storyline.

Let me digress to the technical side and discuss AI placement in terms of weak and strong points. Swordsmen are of no use if placed in places where they cannot easily get to the player. Archers are of no use when confronting players up front - if a player is allowed to kill an archer, he can do so easily. Keep in mind that groups of AIs, when placed and coordinated correctly, can present very devious traps for the player to overcome. On the other hand, when AIs are combined incorrectly, they can spoil the gameplay. Never force the player to play Quake. Over exaggerated? Perhaps. However, if you must force the player into combat, you must also ensure that he believes it was his own fault. There are some dark ways to do so into which I shall not go. Some things to keep in mind: patrolling archers on the ground very rarely, if ever, achieve success; in fact, they are more susceptible to attack, yet they can be effective when placed on a raised ledge where the player has no choice but to run; swordsmen should not be placed in light and left to stand in units unless there are other swordsmen, or, better yet, archers, to support them in case of an attack; slow archers such as spell casters and burricks must take care not to kill their own teammates and should be supported by melee attackers.

When you place an AI in a setting, you receive a set of stock properties. Thief is a dynamic engine, thanks in part to the efforts of Marc Leblanc. When you place an AI, it is best to edit its features to suit the puzzle you intend to create with this AI. Sometimes, it is also helpful to change guard behaviors depending on difficulty level. Which properties should you change? One of the first rules to follow when fooling around with creature properties is balance. If you change the Time Warp value to give the attacker a speed advantage, place a speed potion in the area. If you give him better vision, create expanses of soft surfaces and dark areas for players to hide in. If you play with his alertness, allow the player to distract him with noisemakers. One of the more exciting things to do with properties is create different "moods" for guards and other AIs - angry guards, bored guards, scared servants - was one of the original design goals for the new Thief AI (mostly written by Tom Leonard). If you create these mood variations, be sure that you find the appropriate place for them and that they are noticeable by the player - a bored guard in an easy-to-pass location won't do much good for the mission.

Creating AI patrol routes is a task that needs to be preplanned. Along a patrol route, the player should have the opportunity to run and hide if he sees a guard approaching. The routes should be carefully designed to cover enough ground in an effective way, yet sloppy enough for the player to be able to surpass any patrolling guards. Groups of guards walking together in a "train" can be effective, but most likely will not result in anything other than two knockouts instead of one. Instead of such a combination, it is better to use guards whose patrol routes intersect. Recall "Assassins". At Ramirez's mansion's entrance, the patrolling guards create an interesting dead corner. The player does not know how many guards there are, nor does he know if he will have time to knock out guards at the entrance. This forces him to consider moss-arrowing the stony ground - his only clue that guards are approaching. Then, he must douse the torch, which also takes away some of the visual cue. All this time, the player is kept on his toes as he creeps behind the mansion and the shadows therein. Such is a way that a simple T intersection can be turned into an AI nightmare. Place several of these throughout your level, and you will discover that whatever criticism for your artificial intelligence may have existed will be turned into praise. Such traps are characteristic of Thief's nature.

Other AIs, such as innocents and fire elementals, merit further discussion. Innocents exist in the game to add atmosphere, but that is most certainly not the only reason for their existence. They also exist to alert guards. If it were not for a feature in Thief I that allows a player to blackjack an innocent even after he begins to flee, then any innocent could scare the hell out of the player, despite the fact that they are unarmed without much work on the designer's part (remember the golden child in "Life of the Party"?). Since this is not so for that game, you may need to implement your own innocents, whose alertness is raised before they begin fleeing (a topic discussed in the next paragraph). Aside from innocents, an interesting subject to examine is the fire elemental. Fire elementals from Thief I have the ability to light their own way. Assume that when running from a fire elemental, the player stands no chance unless he can gain a significant head on the creature by twisting his way. Knowing this, add plenty of turns to areas where escape may be required. Fire elementals belong to a broader group of non-human AIs, each with his own attributes. For example, zombies cannot be blackjacked; therefore a design like that of St. Yora in "Return to The Cathedral" may be both frightening (and thus thrilling) and irritating. A uniting property of non-humans is that most of them do not have schemas with relevant sounds. With the exception of ApeBeasts, they cannot convey meaningful messages regarding their current state and/or mood to the player. Keep that in mind when you place these AIs in your level.

No level is complete without the addition of "flee points" that can be used by the AIs to guide their escape. I do not know the specifics of flee point creation; however, I do know that when you choose where an AI should flee, you should ensure that he flees to a location with other guards. For this to be possible, try to avoid letting AIs walk into dead-ends without support outside - by the time they reach their flee point, everyone may be knocked out or dead. Remember that not every guard must flee - certain guards, such as front gate guards, are better off fighting to death since they have no one to flee to. Test your flee points carefully. In Thief II, AI flee points are fully developed, and can be relied on. In Thief I, you may need to do some juggling before you get flee points to work the way you want them to.

Of course, it's very important to script your AIs. If it were not for scripted AIs, Thief would be called Hitman: Code 47. AIs that run for alarms when alerted, AIs that stand ready at the front gate to warn the player, AIs that converse with one another... There is a whole slew of scripted behaviors that you can utilize in your mission. This article will not go to any depth in describing how to place these AIs, merely because each AI is different, and explaining how to use different scripted AIs in different settings will take as much space as twice this article. Simply remember all the concepts mentioned here.

You are almost ready to test your mission. This is the time to take care of other technicalities (explained elsewhere), such as quest variable creation. However, before you consider the level finished, there is one last Step you need to perform.

Step Six: Equip the player (and test the mission in the process, too)

This is something that may seem very unimportant. However, in the end it is this minor design phase and no other that makes or breaks gameplay. Before designing a store for the player, you will need to give yourself a lot of everything using the start point and play your level on all three difficulty levels. What equipment did you use? Try various playing styles - this will ensure you predict everything. Avoid the urge to leave game mode to correct mistakes - by now, all major bugs should have been eliminated. Record on a piece of paper your equipment usage. Next, take a weighted average as appropriate to your level. When you get a set of numbers, curve them until you believe the loadout would be fit for the mission. Your next task will be deciding how much of the equipment should be free. After eliminating story-relevant equipment, you should be left with peripherals such as flash bombs, gas arrows, and healing potions to distribute between the store and starting equipment. Place several of each item, as appropriate for your mission, in your player's starting inventory. Everything that remains behind will be placed in the store. After adding special items such as Tips, add up the cost of the items. Based on your evaluations, create a chart of which players (in terms of style) are likely to buy what on which difficulty level. Then it's a matter of distributing items so that the player has the opportunity to buy whatever he needs. Tease the player - cause him to run out of money the first few times. It should come naturally, and with the help of mathematics it can be perfected into a nice little touch of realism. The reason this Step is so vital is because there is nothing in the world that can irritate the player more than running out of equipment when he shouldn't have to. I will not provide you with a lecture on power-ups, since everything that applies to other games in that respect applies to Thief as well. Remember that in Thief the player doesn't know what he's getting if it is in a container. A short synopsis of power-up placement techniques crammed into one word is: balance. Keep your power-ups balanced with gameplay. Consider power-up placement an extension of the equipment store.

Testing

You should not need to test your level extensively after completion of the Stepwise Refinement Process; if you've read this document, you know that testing along the way is better than testing all at once and discovering a slew of bugs. The testing you conducted in the last Step was probably the most extensive, and if you've carefully planned your design path, should not have turned up many bugs. After Step Six and another test run, it is time to allow others to test your level. They should not find many bugs, either; however, any flaws in your storyline design or illogicalities in architecture that may have slipped past you will probably be caught by them.

Conclusion

I hope that this article helped you in some way. The step-wise refinement method is only one way to do missions; yours may be better. However, for what a programmer's opinion is worth, I believe that these Steps can be a great help to you in creating missions that will be both enjoyable and professional. As mentioned before, this article did not explore overly general or overly specific aspects of level design. The reader of this article should discover things such as storyline creation, power-up placement, and loot location autonomously.

Footnotes:

1) By the term "building", I mean any interior space for your mission, be it a cave, an alien spaceship, or a mammoth statue from Hell. Any solid structure within a centralized mission is referred to as a "building" throughout this document.
2) When picking standard building sizes for standard human buildings, I believe that leaving around fifteen feet for the floors and five feet for the space between them works best. A 72' three-story building leaves enough space for any additional architecture you may need to put in later. Also, any irregular architecture (such as cathedrals) can be accommodated by making them a couple of stories tall (a 35 foot cathedral is high enough, trust me).
3) This is a point overlooked by many designers because of their lust to create stealth-only levels. The more ways there are to play your level, the less linear it will be considered. This does not mean you should force the player into combat; simply that you should give him the opportunity to fight if need be.

==

List of T2 Ambients.
By Inkyape
----

m01wind
m01crickets
m01night
m01trans
m01inside
m01turbine

m01pipes
m01drone
m01tones

m02_litewind
m02seawind
m02_cicads
m02machine1
m02fridge
m02mainL
m02orchestra
m02hits1
m02gearM
m02gearLG
m02machS1
m02machS2
m02machS3
m02machS4
m02machS5
m02machS6
m02machL1
m02machL2
m02machL3
m02machL4
m02machL5
m02machL6
m02machL7
m02machL8
m02machL9
m02creaks

m04wind
m04zaparc
m04hum
m04night
m04tank
m04turbine
m04windobj
m04fan
m04gears
m04throb
m04lava
m04dials
m04bsmt
m04bst_var
m04pulley

m04prison
m04prison_var

m05cicads
m05wind
m05nightL
m05drip
m05maintrem
m05mainhits

m04smith	-CRAZY GORDON SMITH
m04in1	
m04tones	
m04turret	
m06bell	
m06wind	-ERRIE OUTSIDE WIND
m06windtrans	-TRANSITION-MASKING WIND
m06wingedM	-WINGED STATUE MALE
m06wingedF	-WINGED STATUE FEMALE
m06brass	-CREEPY BRASS LOOP IN LAB
m06bubbles	-BUBBLES IN LAB
m06gearM	-MEDIUM STONE WHEEL-GEAR
m06squeaks	- METAL SQUEAKS FOR STONE GEAR
m06whispers	-GRAVEYARD WHISPERS
m06screams	-GRAVEYARD SCREAMS
m06laughs	-GRAVEYARD LAUGHS
m06insidehits	
m06mainL	
M06steam	-STEAMLOOP
m06machL1	-LARGE MACHINES
m06machL2	
m06machL4	
m06machL5	
m06machL6	
m06machL7	
m06machL8	
m16machL9	
m06machs3	-SMALL MACHINE
m07wax	-WAXCYLINDER with TRUART's VOICE
m07night	
m07cricket	
m07wind	-NIGHT WIND (LIGHT BREEZE)
m07basement	
m07mainL	
m07bass	
m07basementH	
m07birdtree	

m07domecym

m08cicads
m08wind
m08nightL
m08drip
m08insideL
m08hitsL

m09gregorian
m09wind
m09wind_roof
m09churchbell
m09voiceL1
m09tonebend
m09vlncalm1 -LOBBY STRING MUSIC
m09hissL -HOT HOUSE AMB STEAM HISS
m09hisses -HOT HOUSE STEAM SPURTS
m09pianoL1 -TO 2ND FLOOR TRANSITION
m09basementL
m09trem_hits -TREM "HITS"
m09string_hits -CREEPY STRING "HITS"
m10windlit
m10night
m10crickets
m10buzz
m10boing
m10sad -SAD MUSIC AT PAGAN VILLAGE
m10subson
m10bellthump
m10den -MAW TONES
m10plantbr -FLESH-EATING BREATH
m10substr -TRANS TO APE
m10windstil -HUM AT APE ENTRANCE
m10cicadas -APE TREES
m10forest
m10cave -TREEBEAST ROOM

M11start
M11wind
M11windlit
M11windgust
M11night
M11bellchurch
M11pipes
M11pump

M11zaparc	-BIG ZAP ARC
M11turbine	
M11ticking	
M11printing	
M11hum	
M11thump	-BELL THUMP
M11singing	
M11gears	
M11steam	
M11tunnel	
M11gregorian	
m11gearM1	-SMALL GRINDING STONE GEARS
m11gearM2	
m11gearM3	
m11gearLG	-LARGE GRINDING STONE GEARS
m11machS1	-SMALL MACHINES
m11machS2	
m11machS3	
m11machS4	
m11machS5	
m11machS6	
m11machL1	-LARGE MACHINES
m11machL2	
m11machL3	
m11machL4	
m11machL5	
m11machL6	
m11machL7	
m11machL8	
m16machL9	
m11wingedM	-WINGED STATUE MALE
m11wingedF	-WINGED STATUE FEMALE
m13wind	
m13windTrans	
m13rain	
m13holyW	-WATER HOLY
m13gregorian	
m13STS_amb	-SUBWAY TO SALLY AMBIENT
m13tones	
m14cave	
m14cavehit	
m14drips	
m14windlo	
m14fridge	
m14lighthouse	
m14windhi	

m14wail	-SUB LEVEL AMB LOOP
m14tension1	-SUB LEVEL TENSION
m14rain	
m14rainL	-SUB LEVEL AMB LOOP loud
M14steam	
m14insideL	
m14hitsL	

m15gearM	
m15machS1	
m15machS2	
m15machS3	
m15machS4	
m15machS5	
m15machS6	
m15machL1	
m15machL2	
m15machL3	
m15machL4	
m15machL5	
m15machL6	
m15machL7	
m15machL8	
m15machL9	
m15tapping	
m15wind	
m15cave1	
m15cave2	
m15trans	-Cavern trans to inside
m15inside	
m15quake1	
m15quake2	-Earthquake hi rumbles SOFT
m15quake3	-Earthquake all rumbles
m15lava1	-Lava1 -burbling
m15lava2	-Lava2 -more intense
m15lava3	-Lava3 -slurping
m15lava4	-Lava4 -soft bubbles
m15falls1	-lava falls
m15steam1	-lava steam -lava blasts
m15steam2	-lava steam -soft bubbling steam
m15crackle1	-lava crackle
m15crackle2	-lava crackle
m15bub1	-Bubble -random blurps
m15bub2	-Bubble fast
m15bub3	-bubble slow -soft bubble loop
m15churchbell	

m16gearM
m16gearLG
m16machS1
m16machS2
m16machS3
m16machS4
m16machS5
m16machS6
m16machL1
m16machL2
m16machL3
m16machL4
m16machL5
m16machL6
m16machL7
m16machL8
m16machL9
m16wingedM
m16wingedF
m16beacon -Beacon choir loop
m16squeakM
m16lava2 -Lava2 -more intense
m16steam2 -lava steam2 -soft bubbling steam
m16crackle1 -lava crackle1
m16bell1
m16fireV -VIOLENT FIRE
m16blade -BLADES (SAW)
m16topwind -WIND AT BEACONS
m16bigfridge -LARGE REFRIDGERATION UNITS
m16windLo
m16cicads
m16orchestra
m16insideL
m16hitsL
m16holyW

------------GLOBAL AMBIENTS-----------------

m00start -HIT THAT PLAYS WHEN YOU START THE MISSION
torch_flame
gaslight_lp
glowball_lp -COLLECTOR TOWER LIGHT
fire_flame -FIREPLACE FLAMES

strlight_lp -STREET LIGHT LOOP
vikglow_lp -VIK GLOW GAS
purple_lp   -PURPLE CRYSTALS
paglight_lp -WILL 'O WISP
cauldron_lp
sewerlite_lp
portal_lp
pbridge_lp -PARTICAL BRIDGE
transf_lp   -TRANSFORMERS
coil_lp
underwater
waterpool
waterpool_rvb -WATER POOL UNDERGROUND [w/reverb]
waterslow
waterslow_rvb -WATER FLOWING SLOW UNDERGROUND [w/reverb]
waterfast
waterfast_rvb -WATER FLOWING FAST UNDERGROUND (w/reverb)
waterwave_sm -WATER WAVES SM [like at edge of a pool]
waterwave_lg
waterfaucet
waterfount_sm
waterfount_lg
waterfall_sm
waterfall_med
waterfall_lg
bugcloud_lp
bughalo_lp   -BUG HALO AROUND ZOMBIES
==

Medusa AI kills player when player sees medusa AND medusa sees player:

Alright, this is working:

1. Create AI (medusa).
2. Add FrobInfo{FocusScript}.
3. Add s->Scripts{TrigWorldFocus;;;;FALSE}.
4. Add Inventory->MaxPickDistance{10}.
5. Create button named "MedusaSeesPlayer".
6. Create FnordLock and set locked->TRUE.
7. Create RelayTrap and link FLAVOUR(Lock) From(RelayTrap) To(FnordLock).
8. Add link FLAVOUR(ControlDevice) From(RelayTrap) To(whatever_kills_player). Probably create a QuestVarTrap for this purpose. I used a VOTrap for testing.
9. Create Inverter and link FLAVOUR(ControlDevice) From(Inverter) To(FnordLock).

10. Add link FLAVOUR(ControlDevice) From(medusa) To(inverter).
11. Add link FLAVOUR(AIWatchObj) From(Medusa) To(Medusa). Data: Player Intrusion, 10 foot radius, line-of-sight, reuse/reset 1000, Frob Object (MedusaSeesPlayer).

Now, when the player is within 10 feet of the medusa **and** the medusa can see the player, the medusa will frob the button MedusaSeesPlayer. The button sends a ControlDevice TurnOn signal to the RelayTrap, which happens to be locked currently. Nothing happens, the signal is killed. However, if at the same time, the player is focused on the medusa (i.e., **sees** the medusa), the TrigWorldFocus TurnOn signal is sent to the Inverter. The Inverter flips the signal to TurnOff and forwards to the Fnordlock, which unlocks the RelayTrap. Within 1000 milliseconds, the medusa will frob the button (if the medusa still sees the player). The unlocked RelayTrap will allow the signal through to the whatever_kills_player object. If the player averts his view, then RelayTrap will once again become locked and stop any signal from the button.

You may want to shorten the timing on the frob button to about 500 milliseconds.

The medusa hilights when the player looks at it, which may be a nice effect or not. I think there is a way to make the AI glow all the time, so the focus hilight isn't noticeable; use Renderer->ExtraLight{10;FALSE}. Other than that, this works great, within the 10 foot limitation. I don't think it will work any farther than that, but in an interior setting with tight corridors, this may not be an issue.

===

Using TrigWorldFocus and TrigDoorOpen on a locked door

1. Set Door properties s->scripts{TrigWorldFocus;TrigDoorOpen;;;FALSE}.
2. Set Door properties EngineFeatures->FrobInfo{Script,FocusScript}.
3. Create a Lockbox (this example requires a separate lock mechanism) and set the property EngineFeatures->Locked{TRUE}.
4. Create a DestroyTrap.
5. Create a RelayTrap#1.
6. Create a RelayTrap#2.
7. Create the trap to receive the TrigWorldFocus; the trap will activate once when the player focuses on the door. Example: VOTrap.
8. Create the trap to receive the TrigDoorOpen; the trap will activate everytime the door opens and not when the door is focused. Example: Switchable light.
9. Link: FLAVOUR(Lock) From(Door) To(Lockbox).
10. Link: FLAVOUR(Lock) From(RelayTrap#1) To(LockBox).
11. Link: FLAVOUR(ControlDevice) From(RelayTrap#1) To(Light).
12. Link: FLAVOUR(ControlDevice) From(RelayTrap#2) To(VOTrap).
13. Link: FLAVOUR(ControlDevice) From(RelayTrap#2) To(DestroyTrap).
14. Link: FLAVOUR(ControlDevice) From(DestroyTrap) To(RelayTrap#2).
15. Link: FLAVOUR(ControlDevice) From(DestroyTrap) To(VOTrap); omit this step when sharing the VOTrap among several triggers.
16. Link: FLAVOUR(ControlDevice) From(Door) To(RelayTrap#1).
17. Link: FLAVOUR(ControlDevice) From(Door) To(RelayTrap#2).

When the door is focused, a CD TurnOn signal is sent to RelayTrap#1, which is locked and stops the signal. A CD TurnOn signal is also sent to RelayTrap#2, which forwards the signal to the VOTrap and to the DestroyTrap. The VOTrap emits the voice-over, then the DestroyTrap destroys the RelayTrap#2 and the VOTrap. (Be sure the RelayTrap#2 specifies the VOTrap **before** the DestroyTrap.) Subsequent focus events generate CD TurnOn/TurnOff signals which only go to RelayTrap#1, which is locked. When the Lockbox is unlocked, the RelayTrap#1 is unlocked and the Door is unlocked, which automatically opens. The Door script TrigDoorOpen sends a ControlDevice TurnOn signal to the RelayTrap#1, which forwards the signal to the Light.
Note: This technique requires a separate locking mechanism. Directly locking the door won't work here, unless the door also has a s->scripts{Lock} script. Use the door in place of the Lockbox; I haven't tried this. Thus, the door would need s->scripts{Lock;TrigWorldFocus;TrigDoorOpen;;FALSE}.

===

A fake wall secret area

1. Create a doorway 4x8x1.
2. Place a spinny SecretDoor4x8; the slidy secretdoor uses custom model and doesn't work well with the example.
3. Set door properties:
   a. EngineFeatures->Locked{TRUE}
   b. EngineFeatures->FrobInfo{[NONE]}.
   c. Shape->TxtRepl0{corresponding wall texture}
   d. Physics->Misc->CollisionType{[NONE]}.

4. On the “secret” side, place an ambientSound, with AmbientHacked{5;0;RemoveProp,DestroyObj;m02cue1}.
5. Create a Marker, named FoundSecret:
    a. Book->Text{foundsec}, where “foundsec” is the file name for the text string <Page_0: “Found Secret!”>.
    b. EngineFeatures->FrobInfo{Script}.
    c. S->Scripts{StdBook}.
6. Create a Frog in a blue room near the secret fake wall. If the frog is too far away, it will freeze.
7. Link: FLAVOUR(AIWatchObj) From(Frog) To(AmbientSound) with DATA: Player Intrusion, radius(5), height(10), Kill link(After Trigger), NoTestAfterTrigger(True), Action:FrobObject(FoundSecret).

The player can pass through the secret door, because the CollisionType is set to NONE. When the player is on the “secret” side, the AmbientSound will trigger a one-shot queue sound and then destroy itself. Simultaneously, the AIWatchObj will cause the Frog to Frob the FoundSecret marker. The marker displays the text message on the screen.
===

Thiefgold: True property names

I used the “list_props” command to list the properties. Here is what I found for property names:

```
Object System\Transient = Transient
Object System\Immobile = Immobile
Object System\Donor Type = DonorType
Object System\Symbolic Name = SymName
Engine Features\Combine Type = CombineType
Engine Features\Stack Count = StackCount
Game: Damage Model\Death Stage = DeathStage
Game: Damage Model\Weapon Damage = WeaponDamage
Game: Damage Model\Weapon Type = WeaponType
Game: Damage Model\Slay Result = SlayResult
Game: Damage Model\Culpable = Culpable
Inventory\Type = InvType
Inventory\Cycle Order = InvCycleOrder
Shape\Scale = Scale
Shape\Model Name = ModelName
ModelNumber
Physics: Misc\Collision Type = CollisionType
SFX\Particle = Particle
\Position = Position
Renderer\Light = Light
Renderer\LightColor = LightColor
Renderer\Spotlight = Spotlight
Renderer\Anim Light = AnimLight
Renderer\Extra Light = ExtraLight
Motions\ActorTagList = MotActorTagList
Motions\Motor Controller = MotorController
Creature\Time Warp = TimeWarp
Motions\Sword Action Type = SwordActionType
Motions\Phys Limits = MotPhysLimits
Motions\Gait Desc = MotGaitDesc
Motions\Player Limb Offsets = MotPlyrLimbOff
```

```
Puppet
Scripts
Game: Damage Model\Hit Points = HitPoints
Game: Damage Model\Max Hit Points = MAX_HP
Renderer\Self Lit = SelfLit
Renderer\Shadow = Shadow
SFX\Heat Disks = HeatDisks
SFX\Spark = Spark
Renderer\Mesh Attach = MeshAttach
Firer
Physics: Projectile\Launcher Mass = LauncherMass
Engine Features\FrobInfo = FrobInfo
Engine Features\FrobLocally = FrobLocally
Physics: Model\Type = PhysType
Door\Rotating = RotDoor
Door\Translating = TransDoor
Tweq\Scale = CfgTweqScale
Tweq\Rotate = CfgTweqRotate
Tweq\Joints = CfgTweqJoints
Tweq\Models = CfgTweqModels
Tweq\Delete = CfgTweqDelete
Tweq\Flicker = CfgTweqBlink
Tweq\Emit = CfgTweqEmit
Tweq\Lock = CfgTweqLock
Tweq\Emit2 = CfgTweq2Emit
Tweq\Emit3 = CfgTweq3Emit
Tweq\Emit4 = CfgTweq4Emit
Tweq\Emit5 = CfgTweq5Emit
Tweq\ScaleState = StTweqScale
Tweq\RotateState = StTweqRotate
Tweq\JointsState = StTweqJoints
Tweq\ModelsState = StTweqModels
Tweq\DeleteState = StTweqDelete
Tweq\EmitterState = StTweqEmit
Tweq\FlickerState = StTweqBlink
Tweq\LockState = StTweqLock
Tweq\EmitterState = StTweq2Emit
Tweq\Emitter3State = StTweq3Emit
Tweq\Emitter4State = StTweq4Emit
Tweq\Emitter5State = StTweq5Emit
Weapon\BaseDamage = BaseWpnDmg
Weapon\CurDamage = CurWpnDmg
Weapon\Exposure = WpnExposure
Weapon\SwingExpose = SwingExpose
Engine Features\Locked = Locked
LockCnt
Engine Features\KeySrc = KeySrc
Engine Features\KeyDst = KeyDst
Inventory\Limb Model = InvLimbModel
Inventory\Render Type = InvRendType
Engine Features\Retexture Radius = TextureRadius
Inventory\Pick Bias = PickBias
Inventory\Max Pick Distance = PickDist
Engine Features\From Briefcase? = FromBriefcase
Editor\Auto-Multibrush = AutoVBR
Renderer\Flow Group = FlowGroup
Game\Bash Factor = BashFactor
```

```
Game\Bash Params = BashParams
CSArrow
CSProjectile
CSProperty
AmbientHacked
Book\Text = Book
Book\Art = BookArt
Script\TerrReplaceOff = TerrRepOff
Script\TerrReplaceOn = TerrRepOn
Script\TerrReplaceDestroy = TerrRepDestroy
Editor\Design Note = DesignNote
Trap\Quest Var = TrapQVar
Script\Timing = ScriptTiming
Creature\Creature Type = Creature
Creature\Is Non-Physical = NonPhysCreature
Creature\Current Pose = CretPose
Dark GameSys\PickSrc = PickSrc
Dark GameSys\PickCfg = PickCfg
Dark GameSys\PickState = PickState
Dark GameSys\FlashInvuln = NoFlash
Prox\Fungus = Fungus
Prox\Blood = Blood
Inventory\Object Name = GameName
Inventory\Tool Reach = ToolReach
Inventory\Can't Drop This = NoDrop
Dark Gamesys\Breath Config = BreathConfig
Dark Gamesys\Air Supply = AirSupply
SFX\Particle Type = ParticleType
Game: Dark\BloodType = BloodType
Game: Dark\BloodCause = BloodCause
Game: Dark\BloodMaxDamage = BloodMaxDmg
Dark Gamesys\Loot = Loot
Room\Automap = Automap
Inventory\Store = ItemStore
Inventory\Purchase Price = SalePrice
Inventory\Long Description = GameDesc
Difficulty\Destroy = DiffDestroy
Difficulty\Lock (Unlock) = DiffLock
Difficulty\Close (Open) Door = DiffClose
Difficulty\Turn On (Off) = DiffTurnOn
Difficulty\Script = DiffScript
AI: Ability Settings\Frog-beast: Explode range = DAI_FrogExpl
Dark GameSys\Stats = DarkStat
AI: AI Core\AI = AI
AI: AI Core\Efficiency settings = AI_Efficiency
AI: AI Core\Movement: z offset = AI_MoveZOffset
AI: AI Core\Movement: max speed = AI_MoveSpeed
AI: AI Core\Movement: turn rate = AI_TurnRate
AI: AI Core\Team = AI_Team
AI: AI Core\Standing motion tags = AI_StandTags
AI: AI Core\Sound tags = AI_SndTags
AI: AI Core\Motion tags = AI_MotTags
AI: State\Current alertness = AI_Alertness
AI: AI Core\Alertness cap = AI_AlertCap
AI: AI Core\Awareness delay (react time) = AI_AwrDel2
AI: State\Current visibility = AI_Visibility
AI: State\Current mode = AI_Mode
```

```
AI: AI Core\Vision description = AI_VisDesc
AI: AI Core\Visibility Modifier = AI_VisModifier
AI: AI Core\Vision Type = AI_VisType
AI: Utility\Visibility control = AI_VisCtrl
AI: AI Core\Awareness capacitor = AI_AwareCap
AI: AI Core\Alertness sense multipliers = AI_AlSnMul
AI: Ability Settings\Device: parameters = AI_Device
AI: Ability Settings\Turret: parameters = AI_Turret
AI: Ability Settings\Camera: parameters = AI_Camera
AI: Conversations\Conversation = AI_Converation
AI: Conversations\SaveConversation = AI_SaveConverse
AI: Attributes\Vision = AI_Vision
AI: Attributes\Hearing = AI_Hearing
AI: Attributes\Aggression = AI_Aggression
AI: Attributes\Dodginess = AI_Dodginess
AI: Attributes\Sloth = AI_Sloth
AI: Attributes\Verbosity = AI_Verbosity
AI: Attributes\Defensive = AI_Defensive
AI: Attributes\Aptitude = AI_Aptitude
AI: Ability Settings\Patrol: Does patrol = AI_Patrol
AI: Ability Settings\Patrol: Random sequence = AI_PtrlRnd
AI: Ability Settings\Idling: Should fidget = AI_Fidget
AI: Ability Settings\Investigation: Style = AI_InvKnd
AI: Ability Settings\Combat: Non-hostile = AI_NonHst
AI: Ability Settings\Ranged Combat = AIRCProp
AI: Utility\Sound value = AI_SndType
AI: Utility\Is Knockout = StimKO
AI: Utility\Marker: Vantage Point = AIVantagePt
AI: Utility\Marker: Cover Point = AICoverPt
AI: Utility\Angle Limits = AngleLimit
AI: Ability Settings\HtoHCombat: Distances = HTHCombatDist
AI: Ability Settings\HtoHCombat: Audio Response = HTHAudioResp
AI: Ability Settings\HtoHCombat: Motion Response = HTHMotionResp
AI: Ability Settings\HtoHCombat: Grunt Always = HTHGruntAlways
AI: Debug\HtoHModeOverride = HTHModeOverride
AI: Utility\Flee point = AI_FleePoint
AI: Ability Settings\Flee: Condition for flee = AI_FleeConds
AI: Utility\Watch: Watch link defaults = AI_WtchPnt
AI: Ability Settings\Idling: Directions = AI_IdleDirs
AI: State\Idling: Origin = AI_IdleOrgn
AI: Ability Settings\Idle: Returns to origin = AI_IdlRetOrg
AI: AI Core\Free sense knowledge = AI_FreeKnow
AI: AI Core\Sees projectiles = AI_SeesPrj
AI: AI Core\Projectile: Visible launch = AI_LaunchVis
AI: AI Core\Broadcast customization = AI_BcstSet
AI: Responses\Signal response = AI_SigRsp
AI: Responses\Threat response = AI_ThrtRsp
AI: Responses\Alert response = AI_AlrtRsp
AI: Responses\Body response = AI_BodyRsp
AI: Utility\Path avoid = AI_ObjAvoid
AI: Utility\Pathable object = AI_ObjPathable
AI: Ability Settings\Inform: Response options = AI_InfRsp
AI: Ability Settings: Inform\Inform others = AI_InfOtr
AI: Ability Settings: Inform\Max. Passes = AI_InfMxP
AI: Ability Settings: Inform\Inform delay = AI_InfDly
AI: Ability Settings: Inform\Min. dist to informed = AI_InfDst
AI: Ability Settings: Inform\Expiration = AI_InfExp
```

```
AI: Responses\Sense combat response = AI_CbtRsp
AI: State\Frozen = AI_Frozen
AI: AI Core\Uses doors = AI_UsesDoors
AI: Ability Settings\Non-combat: Respond to damage = AI_NCDmRsp
AI: AI Core\Suprise [0, 1, Rad] = AISuprise
AI: Utility\Blocks AI Vision = AI_BlkVis
Room\Acoustics = Acoustics
Room\Ambient = Ambient
Room\Gravity % = RoomGrav
Schema\Loop Params = SchLoopParams
Schema\Play Params = SchPlayParams
Schema\Priority = SchPriority
Schema\Message = SchMsg
Schema\Action = SchActionSnd
Schema\Last Sample = SchLastSample
Schema\Attenuation Factor = SchAttFac
Speech\Current Speech = Speech
Speech\Last Played = SpchNextPlay
Speech\Voice = SpchVoice
Speech\Pause Min = MinSpchPause
Speech\Pause Max = MaxSpchPause
Speech\Voice Index = VoiceIdx
Schema\Class Tags = Class Tags
Schema\Material Tags = Material Tags
Physics: Model\Attributes = PhysAttr
Physics: Model\State = PhysState
Physics: Model\Controls = PhysControl
Physics: Model\Dimensions = PhysDims
Physics: Misc\MovingTerrain = MovingTerrain
Physics: Terrain\Friction = Friction
Physics: Terrain\Elasticity = Elasticity
Physics: Terrain\Climbability = Climbability
Physics: Terrain\Can Attach = CanAttach
Physics: Projectile\Explode Me = PhysExplode
Physics: Projectile\Initial Velocity = PhysInitVel
Physics: Projectile\Faces Velocity = PhysFaceVel
Physics: Misc\Rope = PhysRope
Physics: Misc\Pressure Plate = PhysPPlate
Physics: Misc\AI Collides With = PhysAIColl
Shape\TxtRepl r0 = OTxtRepr0
Shape\TxtRepl r1 = OTxtRepr1
Shape\TxtRepl r2 = OTxtRepr2
Shape\TxtRepl r3 = OTxtRepr3
Renderer\Has Refs = HasRefs
Renderer\Render Type = RenderType
Shape\Joint Positions = JointPos
Renderer\Transparency (alpha) = RenderAlpha
Renderer\Self Illumination = SelfIllum
SFX\Particle Launch Info = PGLaunchInfo
SFX\Particles = ParticleGroup
SFX\FrameAnimationState = FrameAniState
SFX\FrameAnimationConfig = FrameAniConfig
SFX\FlashBombInfo = RenderFlash
Renderer\Bitmap Animation = BitmapAnimation
Renderer\Texture Anim Data = AnimTex
Renderer\Water Texture Color = WaterColor
Renderer\Water Flow Color Index = FlowColor
```

```
Texture\Index = TextureID
Editor\Brush Name = Brush
Editor\Has Brush = HasBrush
```

===

Making AI Statues -- this doesn't work

You can put conversation style motion names in the Current Pose property. Then the frac option in that property controls which pose is displayed, 0 being the first frame of the motion, and 1 the last.
If you used a statue as a base, uncheck Object System -> Immobile too, or Dromed will crash when you Objcast light.
I suppose I'll list everything you need to make a normal AI a statue too.

- Add AI->Core->Standing Motion Tags and leave it blank.
- Add Ai->Core->Alertness cap and set everything to 0.
- Add Ai->Ability Settings->Should fidget and set it to false.
- Add Speech->voice and blank it out too.
- Add the MatStone metaproperty, or metal or whatever.
- Maybe add a weaponstim abort receptron too.

Then you add the Creature stuff as described above.

===

Resetting lockboxes:

Add the Lock and TrigUnlock scripts to the lockbox, delete the other scripts listed, uncheck the "do not inherit" flag.

1. Link FLAVOUR(ControlDevice) FROM(lockbox) TO(FlickerTrig)
2. Link FLAVOUR(ControlDevice) FROM(FlickerTrig) TO(OnFilter)
3. Link FLAVOUR(ControlDevice) FROM(OnFilter) TO(Inverter)
4. Link FLAVOUR(ControlDevice) FROM(Inverter) TO(lockbox)

Set the FlickerTrig to something like 8000 milliseconds. Every 8000 milliseconds, it will send alternately a TurnOn or a TurnOff signal. Only the TurnOn signal gets through the OnFilter. The Inverter flips the TurnOn signal to TurnOff, which then goes to the lockbox.

The idea here is to cause the lockbox to respond/forward TurnOn/TurnOff signals (achieved by the "Lock" script), and to make the lockbox send a TurnOn signal when it is unlocked and a TurnOff signal when it is locked (achieved by the TrigUnlock script).

When the lockbox sends a TurnOn signal (by becoming unlocked) to the FlickerTrig, the FlickerTrig starts counting milliseconds. It eventually sends a TurnOff signal (via the OnFilter and Inverter) that locks the lockbox (via the "Lock" script on the lockbox) and that TurnOff signal is forwarded back to the FlickerTrig to turn it off, also.

You'll need to make an OnFilter by creating a TrapTrig object and adding the "TrapOnFilter" script to it. I suggest that you add an archetype under the "TrapTrig" for convenient usage. You'll want to save the custom gamesys under a unique name, and use the "set_gamesys" command to associate the mission with that gamesys file. If you don't understand this, then just make a concrete TrapTrig and edited its script list as described; don't bother with the custom gamesys.

A drawback to this approach is that should the lockbox become locked through some other means (like frobbing the door shut), it will prematurely stop the FlickerTrig. The next time the FlickerTrig receives a TurnOn signal, it will resume counting at where it left off. This may cause an apparently short reset period.

==

Destroying an AI, rather than slaying the AI

The DestroyTrap will destroy (remove from the world) the specified objects, except for AI, which are slain. If you want to remove the AI from the world forever, then:

1. Add AI property s->scripts{TweqOnOff;;;;FALSE}. Be sure to delete all other scripts and uncheck the "do not inherit" checkbox.
2. Add AI property Tweq->Delete{Destroy Obj;[None] ;[None] ;[None];0}.
3. Add AI property Tweq->DeleteState{[None]; [None];0;0}.
4. Create a ControlDevice object, like a button, that can send a TurnOn signal.
5. Link FLAVOUR(ControlDevice) FROM(button) TO(ai), for example.

Click on the button to send a ControlDevice TurnOn signal to the AI. The TweqOnOff script will start the Tweq->DeleteState and advance it from frame 0 to frame 1, which will cause the Tweq->Delete{Destroy Obj} to remove the AI from the world. The AI is not slain. If the AI had a property s->scripts{TrigSlain}, that script would not execute.

Apparently, the Player must be looking at (or within the line-of-sight) of the AI in order for this technique to delete the AI from the world. Try adding the OffScreen flag.

==

Tutorials - Converting Dromed 1 -> Dromed 2 by Conchong(compiler/editor)

Here is a guide, as promised a while ago, which brings together, updates and corrects various D1-D2 conversion threads. I converted The Docks in about 30 minutes using this method. Credit to many TTLG forum members for this.

1) Open up your mission in D1 and use the command "do report". Save report as textures.cmd

2) Open the report file and delete everything except the names of the textures. (they come in the format texture XxY ####. You only want to keep the name)

3) Play hunt the T2 texture. Unzip the fam.crf from the T2 CD1. Do a find file search for each of the texture names. When you find them put the line "load_a_texture folder texture". If you don't find the file search for one that looks similar with a pic browser and put that in place of the original texture. Make sure you keep the textures in the same order they were originally. Place the line "load_family blank" at the top of the file and save.

Alternative method, using Perl scripts

This was written by Bruny. You need Perl installed for this http://members.xoom.com/buxtehude/txtmap.pl

It's got a bug in it though. Replace the line -

```
print OUTFILE "load_family $TextureMap{$txt} $txt\n";
```

with

```
print OUTFILE "load_a_texture $TextureMap{$txt} $txt\n";
```

Generate the report "texrep" with the "do_report" command. Copy the perl program into your Thief 1 directory. Open your DOS prompt and cd to your Thief 1 directory, e.g., "cd \game\Thief", and execute the script by typing:

```
perl txtmap.pl texrep
```

It will generate a file called textures.cmd which you can then execute in Dromed 2 with the command "run textures.cmd".

4) Unzip all the thief 2 textures into your thief1/fam folder. Keep the original folder structure intact. Make sure you uninstall any active FM's first.

5) Reopen Dromed 1 and load the mission to convert

6) In the command box, type "run textures.cmd". All the textures in the mis are now T2 textures

7) Save the mission as a COW file.

8) Exit Dromed 1 and start Dromed 2.

9) Load the COW file.

10) Type "set_gamesys blah".

11) Save the mission as a MIS file and exit Dromed 2.

12) Open dark.cfg. Lower the "obj_min" value to something like "-18192". Save and exit.

13) Start Dromed 2 and load the MIS file.

14) Select Thief 2's Dark.gam file when it prompts you for the GAM file.

15) The level SHOULD load with no problems now and the object hierarchy should be Thief 2's.

16) Type "find_lost_objs" and then "purge_missing_objs" to clear Dromed 1 objects in the "missing" folder.

17) To finalize it, make sure you type "script_load convict" and "script_load gen".

18) Assuming it works, portalize and save the MIS file.

19) Exit Dromed and open dark.cfg.

20) Set the "obj_min" value back to the original value, which should be "-8192".

21) Save dark.cfg and exit. Now the mission should load with no problems and should be compatible with others.

Known Issues

* The texture replacement seems to be off with some missions (2 out of 3 I tried). This may be due to where Dromed 1 loaded water/lava textures into the palette. At least you have the correct palette to re-texture.

* Torches are backwards - you'll have to turn them round by hand. Other objects may be affected but I haven't spotted any yet

* Some AI may need changing

* The ambient schemas are different in T2

* Some motions are different (eg. Drunk 0)

*Custom gamesys objects will be lost. That may screw-up any traps/puzzles linkages.

==

Custom Textures

T2 skins must still be 256 color, BUT any 256 colors. HOWEVER T2 uses the first color in the palette (i.e. top left when you look at the pallette) as the transparency color, whatever that happens to be (or at least that's what I've found) AND, I believe, black...sorry!!

I had the same problem as you and had to do the following (I have Photoshop but I guess PSP will be similar)

1) go into the palette and change the top left color to something obvious - say bright red.

2) use the magic wand to select all the bright red pixels on your texture.

3) fill those pixels with one of the other 255 colors from your palette (I guess you'd use whichever one is nearest to the colour that was originally there)

4) use the magic wand to select any pure black pixels (not always that easy if you have other very dark pixels in your palette)

5) fill with the darkest non-black color you have.

6) go back to your pallette and change that top left colour from red (or whatever you used in step 1) to jet black.

7) save your teture.

I've just successfully gotten rid of all the 'holes' in about six new textures that I'd created by doing the above.

There is a tool called **Bright** (at http://homepages.compuserve.de/daxim5/shock/bright183.zip) which saves you an amount of work, as you can do batch processing with it. Procedure:

- Create your texture in true color depth. Give all pixels which shall appear transparent in game a dummy color which greatly differs from the used colors, for example magenta or green. Save as .tga

- Run the command

  ```
  bright -black file.tga
  ```

  To reduce the image to a 256-paletted .pcx, the index 0 colour is black, exactly what we want. Note that the Bright algorithm works much better than the internal one from PSP, the result is much more eye-pleasing.
- Load the .pcx and change all occurrences of the dummy color to index color 0, thus making them transparent. Save as .gif

Reducing a file onto an existing palette in PSP looks truly gruesome. But fear not - Bright does this job, too, but much better. Behold:

```
bright -pal palfile.pcx inputfile.tga
```

Everyone who deals with palette files should test out Bright.
==

New Texture Family

To make a new texture family (it was already posted several times ago, there are at least 2 tutorials on that somewhere, but hey, you were pleading, so I suppose I must take my time) :

1. Create all your textures in true color and save them as **.tga** (Targa format) into the same directory.
2. Reduce them to a shared 256-colour-palette with the colour on index 0=black thus:

   ```
   bright *.tga -common -black -writepal full.pcx
   ```
   ...whereas after completion you receive the output files with the same name but with **.pcx extension** and a dummy palette file named **full.pcx**
3. Make in your **fam** directory a custom subdirectory, for example **chiefd** and move all the .pcx files in there.
4. Fire up DromEd and **add_family chiefd**. That's it.

If you are stuck or encounter problems, just post again and describe the situation exactly. Neither 'didn't work' nor 'couldn't get it to work' are helpful.
==

From the recommendations Bright is great for reducing graphics, as long as you don't use black in your texture at all where you don't want transparency. I'm not sure if there is a way using Bright you can force the first colour to be something other than black, but I've read a few posts where people seem to have developed complex workarounds because of this.

If you happen to have Photoshop (sorry, I don't have PSP but there may be an equivalent process) you can relatively easily convert to a palette forcing a different color as color 0, one that is not likely to include pixels in your image.

. Open your 24 bit image.

. Image > Mode > Indexed Color

. Palette = Adaptive, Colors = 256

. Choose Forced = None, then Choose Forced = Custom... (this clears any existing forced colors)

. Click the top left gray box

. In the colors picker select or enter a color NOT present in your image - you can have painted areas you want transparent with this color previously. Lime green or hot pink are common options.

. If you want to force black also, click the box NEXT TO your new transparency color and select black. Yes this works for any other colors you want

. Choose your Dither options, Diffusion at 75% should be fine.

. Hit OK

Voila. I don't know if this will help anyone, and it does use Photoshop's color matching algorithms which aren't meant to be as accurate as Bright, but I thought I'd throw it out just in case...
==

EDITING SCHEMAS

by Sledge

It's easy enough to create new sounds for your level, but then the question becomes how you get them to work in DromEd. Assuming you have basic knowledge of Ambient Sounds, this tutorial will hopefully explain a little bit more about how Schema Files work. This should be useful for both Thief 1 and Thief 2.

WHERE THEY ARE:

The first thing you need to do is to get the schema files themselves. They are not by default in your Thief directory, even on full install. They should be on one of the Thief discs in a folder called Schemas. You need to grab all of the files in this folder, including Speech.spc and Envsound.spc. Make sure the "schema" folder is intact (in other words, you should have a folder called "schema" [no quotes] in your Thief 2 folder with all of the files in it).

-From now on, whenever you do a reload_schemas command in DromEd it will reload all of the information in these files. So don't remove any of them from your schema folder unless you're done making all of the changes you're going to make.

WHERE TO PUT THEM:

Once you have made your new schema files or edited the existing (although there's really no need to do the latter), you type reload_schemas in DromEd to compute the changes. After this, you save the gamesys. Beyond this, you don't need to include the schema files when you release your mission or use them ever again. However, it's good to leave them where they are so that if you decide to make any changes in the future they're in the right place.

Your sound files go in a folder called "snd" (no quotes) in your Thief 2 folder. Create an individual folder for each group of sounds, and then another folder within this one to specify language. In other words, the folders might go Thief2 > snd > mynewsounds > English. Then, in the English folder, you would place all of your wav files.

DromEd does not care how you organize your sounds or your schemas. It will compute everything in the "Schema" folder and everything in the "Snd" folder. The reason we separate things into their own schema files or our own folders is for convenience and convenience only.

Following with this idea, it is not necessary to edit the existing schema files if you do not like a sound. Just create a new one. Say, for example, you wanted M04wind to use 2 new wind sounds as well as the ones LGS created. You don't need to open up the existing amb_m04.sch file in order to do this. Simply create a new file, stick it at the end of the list, then re-create the information about m04wind from amb_m04.sch in your new schema file and add your sounds to the information. The reason for this is to not confuse things and to not mess up the original files should you do more than one mission with new sound. There's no reason not to simply use completely new schema files.

How you name your schema files is arbitrary.

HOW TO CREATE A SCHEMA FILE:

Open up Notepad and think of a name for your schema file. It really doesn't matter what it is. Let's say it's called "blog." Open up your text file, and before you do anything, click "Save As." Save the file as an "All Files" (not a txt file) and type the following exactly in the name line: "blog.sch" (WITH the quotes). Now the icon for the file appears and you've got an .sch file.

>From here on, the easiest thing for ambient schemas is to copy one whole entry from an existing schema into your own. Then, change the names to your own liking and use this as a model for all of your entries that follow.

For an AI voice schema, I've found it's best to copy an ENTIRE voice repetoire into your own, then edit it to your liking. The easiest schema file to use is Testvoice.sch because it is complete and is not cluttered with extraneous file names and comments.

It's going to be easiest for me to illustrate this through ostensive definitions and explanations, so what I've done below is copied a single entry from some schema files and then explained line by line what they do.

AMBIENT SCHEMA SCH FILES

This is an example of something I put in my FM in a new schema file:

```
//wind

schema invwind

archetype AMB_INV

mono_loop 0 0

volume -1

invwind
```

//wind = A comment which is only there for the person looking at the schema.

Whenever you have a "//" that means that this is a "comment" and does not have any meaning other than to that of the viewer.

schema invwind = This is the name that will show up in the gamesys in DromEd when you type in reload_schemas. You DO NOT have to edit anything in DromEd to get your new sounds (with one exception, see below). Everything can be done in the schema sch file. This is the name of the object that will "appear" in DromEd after you type reload_schemas. When you place an object with the AmbientHacked property in DromEd, you will specify this name under SCHEMA NAME and not the name of the wav file(s).

archetype AMB_INV = The name of the pre-existing "object" found in the object hierarchy under which the ambient sound is to be placed when you hit that reload_schemas command. In order for your reload_schemas command to work, you do have to make one small gamesys change. Create a new category for your sounds. The pre-existing ambients are classified according to mission. If you open up the Object Hierarchy, you'll notice that under Sounds > Schema > Ambients there is a category for each mission. For my mission, The Inverted Manse, I used AMB_INV as an abbreviation. To do this, find Ambients in the Object Hierarchy. Hilight it, then click "Add." Type in a fitting name. Save your gamesys. Then, when you write your schema files, make sure that this third line is exactly the same as the one in your gamesys for every entry.

mono_loop 0 0 = This means the sound consists of a single loop played repeatedly with no delay. The two numbers (in this case 0 0) are an interval of delay. 2000, 5000 would mean that the wav plays once, stops, then plays again at a random time somewhere between 2 and 5 seconds after the stop for another full loop, then stops again. If you want to have file with multiple sounds, then you would put poly_loop 2000, 5000 (I've never seen such a combination of sounds at 0 0 for obvious reasons). The wind I had in this case was consistent and played repeatedly.

volume -1 = this is the default volume for the sound, -1 being the loudest and anything beneath that being quieter (all numbers must be negative). This can be "overridden" when placing an ambient sound in DromEd, but the default volume is still important as it is easier to place unadjusted sounds... and you can't adjust volume when using a sound trap as opposed to an ambient marker, as far as I know.

invwind = This is the name of the actual wav file to be placed in a folder in the snd directory. When naming them here, multiple files are placed side by side with no comma.

Thus, if I had several wav files of different wind sounds that I wanted to be repeated after a slight delay, the entry in the .sch file might look something like this:

```
//more wind

schema newinvwind

archetype AMB_INV

poly_loop 2 10000 12000

no_repeat

volume -800

invwind2 invwind3 invwind4
```

For some reason there is always a "2" after poly_loop. If it is not there, the sound will not be created in the gamesys.

The no_repeat means that the sounds (in this case, invwind2, invwind3, and invwind4) will play randomly and one of those sounds will never be played twice in a row.

THE FINAL STEPS

Once you have completed your schema files and both these files and your wavs are in the right place, (as mentioned above) open up DromEd and type reload_schemas. I know I've mentioned this command several times, but you only need to do it once... at the end after everything else is done. Check to make sure your new sounds work, then save your gamesys. If you did everything right, the "objects" you specified will appear in the Object Hierarchy and you will have everything you need in order to use the new sounds. There is nothing else manual other than actually placing the sounds or voices in the editor. Also, remember to include all of the "old" LGS schemas in your "schema" folder. DromEd literally replaces all of the sounds in the Object Hierarchy with the ones from the "schema" folder, so you need all of the old ones there in addition to your new ones when you do this command.

==

Adding multiple in-game maps non-linear for ThiefGold

I solved this problem in "The Varyx Obelisk" by assigning a power-of-2 numeric value for each map trigger. A QuestVarTrap for each map would add the appropriate power-of-2 to an accumulator variable. For each possible combination of map triggers, I had a QuestVarTrig that would monitor for that accumulator sum and issue a ControlDevice TurnON signal to a pair of QuestVarTrap objects that would set the appropriate min:max ranges. Duplicate maps are required to cover all possibilities.

The scroll objects have property S->Scripts{StdButton;;;TRUE}, which makes the scroll act like a button, instead of a scroll, and property EngineFeatures->FrobInfo{Script,Delete;;} to cause execution of the script and automatic deletion of the scroll. The scrolls are linked to their corresponding QVarTrap objects that add the appropriate power-of-2 value to the accumulator.

For example, adding 3 in-game maps at different locations in any order:

Assign the first map power-of-2 value "1". Assign the second map power-of-2 value "2". Assign the third map power-of-2 value "4". (A fourth map is value "8", and so on.) Frobbing the first scroll deletes that scroll and sends a CD signal to the QVarTrap which adds "1" to the accumulator. Frobbing the second scroll deletes that scroll and sends a CD signal to another QVarTrap which adds "2" to the accumulator, and so on for all in-game map scrolls.

The QVarTrap property would look something like this: TRAP->QuestVar{+1:accum} for the first QVarTrap, TRAP->QuestVar{+2:accum} for the second QVarTrap, and TRAP->QuestVar{+4:accum} for the third QVarTrap. Be sure to create the "accum" variable with an initial value of zero.

In this example, you need a number of QVarTrig objects to monitor for all possible values of the accumulator. That number of objects is equal to (2 raised to power of the_number_map_scrolls) minus 1. In this example, that is (2**3)-1, which equals 7. You want to monitor the accumulator for the distinct values 1, 2, 3, 4, 5, 6, 7. Thus, this example requires 7 QVarTrig objects, and 14 QVarTrap objects to receive the signals from the QVarTrig objects. You also need the 3 QVarTrap objects that are linked to the scrolls.

If "N" is the number maps (distinct ranges), then you need (2**N)-1 QVarTrig objects, and N+2*((2**N)-1) QVarTrap objects, and N scroll objects.

- When the accumulator is 1, then **only** the first map (one range) has been triggered.
- When the accumulator is 2, then **only** the second map (one range) has been triggered.
- When the accumulator is 3, then **both** the first and second maps (two ranges) have been triggered.
- When the accumulator is 4, then **only** the third map (one range) has been triggered.

- When the accumulator is 5, then **both** the first and third maps (two ranges) have been triggered.
- When the accumulator is 6, then **both** the second and third maps (two ranges) have been triggered.
- When the accumulator is 7, then the first, second and third maps (all ranges) have been triggered.

The corresponding QVarTrig objects for each of these distinct "accum" monitors would send a CD signal to the corresponding pair of QVarTrap objects to set the "map_min_page" and "map_max_page" variables.

That's how I did it for ThiefGold. I have no idea if Thief2 is any easier, or whether there is a better way for ThiefGold.

Somebody please copy this to tutorial somewhere. It's hard to remember this after so many months... :eek:

==

CamVator Cutscene by Silentsleep and Saturnine.
=============

This tutorial is to give an example of a smootly scrolling cutscene-style view of a conversation or some sort of fly-by sequence.
This works for Thief 1 and Thief 2.

First of all, set up the elevator. Make 2 TerrPts along the route you want the camera to travel, it can be a straight line, or something a little fancier, but there's just 2 for this tutorial. Now select the first TerrPt give it a TPath link to the other TerrPt. Highlight the link and hit Data, and enter the speed you want to travel at. 3 is about right to keep pace with an average AI. Now link the second TerrPt to the first, with another TPath link. Leaving the speed at 0 is a good way to stop the elevator there, which is what I did.

Now make the CamVator. Type "find_obj raft" into the console, and create one. Resize its dimensions to 1,1,1 for convience. We don't want it getting in the way of anybody so open up its properties and go to Physics->Misc and change the Collision Type to None. Also, change the Moving Terrain option to false, if necessary. This'll make sure the raft starts sationary. Rename it to CamVator and finally, give it a TPathInit link to the first TerrPt in your sequence.

Make a button and give it a CD link to the second Terrpt. Frobbing this will start the CamVator moving slowly along the route. This button will be part of the final conversation. In the example mission, Its called TeleButton. You can make your CamVator unrendered now if you want, and T2 users should compute the pathfinding database.

Now for the conversation. Make an AI and put him near the first TerrPt, a little behind it is better. Add Ai->Core->Alertness Cap and set everything to 0 so he doesn't start attacking you. Make a marker, name it ConvPt and give it the TrapConverse script. Add the conversation properties too. Link from the marker to the AI with the AiConversationActor link and set the link data to 1. Make another button and CD link it to the ConvPt. This button will start the conversation, so you might end up using a BoundsTrigger or something in your final mission. Make a marker near the final TerrPt and name it something like EndPt.

So how is this going to work anyway? 🙂Well, we're going to give the player a PhysAttach link to the CamVator for the duration of the conversation. This will drag the player along the route of the CamVator. It's important to note that the route should carry the player _above_ the floor. If the player is able to get a foothold and try to walk or jump, things can get strange, possible even to the point of crashing Thief. In Thief 2 Adding the PhysAttach link will make the player zoom to the CamVator! It's a cool effect, but not what we want here, (Although possibly of some use to display the effects of a new weapon of some kind that can blow someone clear across a room at high speed!) (hmmm) In T1, the link will hold Garrett in his current position, and move him relative to the motion of the CamVator. Which is not what we want either. The solution is to teleport the player to the CamVator just before we add the PhysAttach link. So, open the properties of the CamVator and add the script "TrapTeleporter". CD link your TeleButton to the CamVator. The conversation will handle the teleportation.

Now open up the conversation property of the ConvPt. In the first and second Actions, enter this:

Add Link
ControlDevice
Player
CamVator

Frob Object
TeleButton

This Cd link the CamVator to the player, and teleports him to its location. The Telebutton also starts the Elevator moving. Next add this:

Add Link
PhysAttach
CamVator
Player

Goto Object
EndPt

Remove Link
PhysAttach
CamVator
Player

Now the player is attached to the elevator, and the AI starts walking to the EndPt. That's all our conversation is doing. You can enter in other things here, like speech, motions, or whatever you want. You conversation ends with removing the PhysAttach link. It's tempting to destroy the CamVator instead, but don't. The player will be locked in position.

The player should drop to the floor. You'll probably want to teleport him someplace else, so just set up another button and Teleport trap, CD link it to the player with your conversation, and frob the button.

That's it! Go into game mode from the button linked to the ConvPt, and you should be whisked off to follow the CamVator. Once you're finished, you can adjust the heading and pitch of the CamVator to give you a good initial view. In D2 however, its position will reset every time you compute the pathfinding, so don't forget to change it back.
==
==
==
==