Dark Disciples II : Editor Overview

DD2 comes with a complete level editor. Everything within the default game was created via the level editor. This document explains how to use the editor and create your own games. While the editor is, for the most part, fairly user friendly, a basic understanding of programming concepts will definitely be useful. If you understand what a *flag* is and the concept behind a *if…then…* statement, then you're already half way there.

Note: This guide is not definitive by any means. What it will do, is explain all the important concepts/options and any other features which may not be self explanatory. Also see the **Tutorial**.

Terminology

Here are a few terms that you need to understand before reading this guide.

• **Event -** An event is something specific that is placed on a map such as a door, a chest or a trap.

• **Script** – A script is an NPC conversation tree. To create a script, you need to place a Script event on the map, and then edit the script text.

• **Flag** – Flags are used to determine what events have or have not yet occurred in game. For example, an event might only occur if a flag is set to a particular value. Flags are also used in scripts. There are 7000 available flags (0-6999). By default they are all set to -1.

• **Status Flag –** A status flag is a specific type of flag. It is highlighted in red and is used to record the current status of an event. For example, a 'Complex door' event has a status flag that is used to record whether the door is currently open, closed, locked or broken.

Overview

There are actually two editors in DD2, and a bunch of databases:

Level editor – allows you to edit maps and modify values is a number of *databases*. This editor can be accessed by pressing CONTROL+U at the main menu screen, or by clicking the symbol in the bottom left hand corner.

Associated files: 'LevelX.dat', found in 'Mod/LevelData' directory.

Database editors – There are a bunch of databases built into the *Level editor*. These allow you to edit game objects that are not directly associated with a particular level. The databases are accessible from the 'DB1' and 'DB2' tabs on the right hand side of the level editor screen.

• **Objective** DB – Allows you to change the game main quest text (which appears in the 'Objective' selection from the main menu). You can also edit the start location of the hero (map number, X and Y coordinates) and the default quest number (refer to the *Quests* DB below).

Associated file: 'GameIntro.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

• **Quests** DB – Allows you to edit the text for each quest. In game, these appear in the Quest log and are added or removed as the hero progresses through the game. (To assign a quest to the player, you need to use the 'Set Quest' code in a script. Scripts are usually NPC conversations and are placed on the map with a Script Event).

Associated file: QuestText'.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

• **Item** DB – Allows you to edit the stats of every item in the game. For example, you could change the damage range of a weapon, the protective value of a piece of armour, or create a new *quest item*.

Associated file: 'Item.dat', found in 'Mod/GameData' directory. This file SHOULD NOT BE deleted – edit it only!. Upon restarting DD2, a new empty file will be created – however since items are partially hardcoded, redefining items will lead to strange behavior ingame.

• **Monster** DB – Allows you to edit monster statistics of default monsters, or create new ones.

Associated file: 'Monster.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

• **Modules** DB – This DB provides optional functionality. Think of a module in terms of a classic D&D (the pen and paper RPG) module. Essentially, this allows you to modularize scenarios and give them a pretty 'front end'. The player can select modules to play via the *Module* event.

Associated file: 'Module.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

• **Dreams** DB – Another entirely optional database. If you want the player to receive location based dreams when they 'rest', this is the DB to do it.

Associated file: 'Dream.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

• **Books** DB – A number of books can be designed here. Each can have up to several pages of text. Each book here is associated with a book item which, if used by the player, brings up a book page and the text you entered.

Associated file: 'Book.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

• **Penta Portal** DB – Again, optional. Up to 5 portals can be set up which allows the player to teleport instant between these locations. Each portal requires a portal stone to activate. Once activated, you can travel from any given pad to another one. A portal is placed using the *portal* event but is defined here.

Associated file: 'Penta.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

• **Forge Recipes** DB – Optional. A number of recipes can be defined which allows the player to 'brew' up special items. A forge is placed on a level by using the *forge* event. The recipes themselves are defined here.

Associated file: 'Forge.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

• **Skill Editor** DB – There are 11 skills built into DD2 which are available at character creation (You cannot create new skills). The Skill editor lets you modify the effect of the skills. For example, the Armsmaster skill gives you a combat 'to hit' bonus and a damage bonus which increase with high skill levels. You can alter the exact values as you see fit (for balancing purposes etc).

It also allows you to determine the number of stat points and skill points you begin with at character creation, and how many extra you gain per level up.

Associated file: 'SkillBonus.dat', found in 'Mod/GameData' directory. This file can be deleted. Upon restarting DD2, a new empty file will be created.

1.0 Directory Structure

User Mods:

The DD2 directory structure is deliberately set out to make swapping user mods in and out as easy as possible. Within the DD2 directory, is the 'Mod' directory. This contains the LevelData directory, the GateData directory and the SavedGames directory.

• **LevelData directory** – contains all the user constructed maps in the format:

LevelX.dat where X is the level number.

• **GameData directory** – contains all the user edited data files:

Book.dat	define readable book text
Custom.bmp	user defined graphics are imported from this file
Dream.dat	define dream text for when PC rests
Forge.dat	define recipes for making items

GameIntro.dat	define game objective and start location
Item.dat	all items defined here
Module.dat	independent quest blocks (modules) defined here
Monster.dat	defines all monster statistics
Option.dat	the game settings (e.g. difficulty level) are stored here
Penta.dat	defines penta portals
QuestText.dat	defines text descriptions for quests
SkillBonus.dat	data tables for skills and powers

• **SavedGames directory** – contains all the player saved game files and the 'master list' file called SaveList.dat.

To play someone's game, simply delete (or move) the existing Mod directory, and replace it with the new one. Note that the saved games are kept in the Mod directory for a reason – you can't run the same character through different user mods (trying to do so will cause crash outs or weird behavior).

Where are the music and sound effect files?

The music and sound effects for DD2 have been compiled into the executable and are not directly accessible by the user.

2.0 Level Editor

Each level (map) is comprised of 5 *layers*. Each layer is an array of 50x50 squares. You can create up to 130 maps (map 0 – 129).

• **Tile1 layer:** This layer is where the ground tiles (such as grass, walls etc) are placed.

• **Tile2 layer:** Tiles that sit on top of others (e.g. a rock or statue) would be placed here.

• **Access layer:** Determines what tiles can be walked through, or looked though (completely independent of the tile placed there).

• **Zone layer:** A map can be separated into different zones, each represented by a colour. Each zone can be given a different name. The main purpose of zones is to allow you to create separate areas on the same map: when the player opens up the map function in game, only the current zone is displayed. Zones can also be given specific traits such as extreme temperatures or antimagic properties.

Advanced use of zones: The yellow and blue zones have an extra option: a flag which you can set to 'suppress the top layer and access layer'. This extra functionality allows you to toggle on or off large areas of a map with a simple variable change.

To access this functionality simply nominate a flag number. Once set, all top layer and access tiles converted by the area of that zone will be suppressed – that is, they will not appear on the map. If you then unset the flag, the top layer and associated access will spring into existence. Note that this effect can also be achieved by

multiple TOGGLE or ALT TILE events. The reason why using the zone functionality is preferable for large changes is that each event uses up extra memory (the level file size will grow), while the zone usage does not.

The green and cyan zones can also be used to toggle on or off 'conditions'. For example, if a zone has the 'Very Hot' condition, unprotected PCs will loose Hp with every step. This environmental condition can be switched on and off using the associated flag.

• **Event layer:** Holds any events (such as a chest or door) placed on the map.

How do I move around the map?

Use the arrow keys. A map is comprised of a 50x50 area.

How do I place tiles on the map?

On the top right hand side of the editor screen are 8 boxes. These are 'slots' where you can place tiles (such as grass, a wall etc). One slot is highlighted with a red box. To select a slot, simply click on it. To add a new tile to that slot, click on the large 'T' button below (T being for tile). This opens up the tile selection dialog box. Select a tile. To actually draw that tile on the map, simply left click on the map. Note that you'll draw the tile that is in the highlighted slot.

You can draw tiles on the Tile1 (bottom layer) or Tile2 layer (top layer). Which layer you are drawing to is determined by the button just below the slots (which will read 'btm layer' or 'top layer' depending on what's currently selected).

To erase tiles, select a blank slot or select a blank tile and 'draw' over the tiles you want to erase.

AutoAcc: You'll notice a button labelled AutoAcc: On/Off next to the layer toggle button. When on, the program will automatically guess what access layer you want when you place a tile on the map. You'll want it on sometimes, and off other times since it won't necessarily always get it right (you'll see what I mean when you have a play with it).

How do I place access tiles on the map?

Select an access tile by clicking on the big 'A' button and draw it onto the map.

How do I place zone tiles on the map?

Select a zone tile by clicking on the big 'Z' button and draw it onto the map.

How do I place events (such as a door or chest) onto the map?

Select an event by clicking on the 'E' button. Click on the map to place the event. *Right click* on the placed event to bring up a dialog box with the event variables. Most

of the options such be fairly self explanatory – for example, you can decide what items will be inside the chest etc.

What are Status Flags and how do they work?

A status flag is a specific type of flag. It is highlighted in red and is used to record the current status of an event. For example, a 'Complex door' event has a status flag that is used to record whether the door is currently open, closed, locked or broken. Generally you'll want each status flag to be unique: there are 7000 flags available (0-6999).
A special function exists which will tally up all the flags used so far and which ones are still free. To access this function, click on the 'Util' tab, click on 'Flag search' option, and select 'Free Flags'. Note that this function loads and searches every existing level, so make sure you save the current level you're working on first, otherwise your changes will be lost.

How do I create NPCs?

Place a *script* event on the map and right click to edit it. Click on the 'Browse' button next to the 'Script Num:' entry. This will bring up the script editor. Note that each level can have up to 30 scripts. Different *script events* can access the same script, if you desire.

To create the new script, click on the 'create script' button. Each script comprises of 400 lines and, when created, increases the file size of the map.

Each script line comprises of a 3 character code segment and the data segment. There are a whole bunch of different codes that perform different functions. To select a code, click on the code box and select one from the menu. The data segment will then change to provide you with additional entries to add specific variables or text. For example, the TXT code allows you to enter a line of text which is displayed to the player. The best way to get to grips with how scripts are put together is to open up an existing level and check out an example.

Importing script text: Adding large swaths of text to a script can be laborious. Instead, you can add text by importing it from a file. Here's how:

Step 1: Create a text file called 'ScriptImport.txt' in the Dark Disciples 2 root directory.

Enter the text you want to import. The text can be separated into different 'chunks' which can be imported individually. Each chunk is separated by a '%' symbol, followed by a name. This line is referred to as an 'identifier'.

%introduction

Welcome, my friend. I'm glad to see you made it back alive! Did you bring the Sword of Doom with you?

1. Yes I did.
2. I couldn't find it...

%Yes

Good, good. We are one step nearer to completing our goal. Your next mission is to.....

%No

Well what the hell are you doing here? Go back into the dungeon and retrieve the Sword. Without it we cannot prevail!

Step 2: Load DD2, enter the editor and place a script event. Edit the script event. Add a TXT event at a place in the script where you want to import a chunk of text.

Step 3: You'll notice that a red exclamation mark appears on the right hand side of the line. Click this button. You're now presented with a menu listing the identifiers found in the ScriptImport.txt file. In the above example, they would be 'introduction', 'yes' and 'no'. Click the one you want and the relevant text will be imported into the script.

A few notes:

• Importing text into the scripts is not fool proof. It's a really good idea to save the level before inserting scripts, just in case the result is not what you expected. It's probably also possible to cause a cash if you try hard enough...

• You don't have to arrange the text in the file to line up with the script TXT lines – a single large paragraph will automatically be fitted into the script line format of the editor.

• If the imported text takes up more than 1 TXT line, new TXT lines are automatically generated and inserted (existing lines below the insertion point are automatically moved down). Be aware that there are a maximum of 400 lines per script – if inserting text causes this limit to be exceeded, the excess lines will 'disappear' of the bottom (i.e. erased).

• Lines beginning with a number (i.e. user options) are automatically made blue (i.e. they use the BTX code, not the TXT (black) code.

TAB functions

On the middle right side of the editor screen is a box with 6 tabs labelled Def, View, Clear, DB1, DB2 and Util. Clicking on each gives you a different set of functions:

Def tab – These functions allow you to define various elements *specific to the map you are currently editing*.

• **Define Zones –** Once you have defined an area as a particular zone, you can use this option to attribute properties to that zone, such as its name and environmental conditions associated with it.

• **Define Exits –** This option defines where the hero ends up if he walks off the edge of the map. Note that tiles on the edge of the map must be designated as valid exit tiles by placing 'Map Border Exit' *access* tiles there. The Define Exits option is simply used to designate *where* the player ends up – without the access tile, the player won't go anywhere regardless of what's entered on the Define Exits dialog box.

• **Define NPC scripts** – Each level can have up to 30 scripts. This option is simply another way of accessing the script editor. Scripts can also be accessed directly from the *script events* where they are 'called'.

• **Select music –** A simple option which allows you to select a music track for the level, if desired.

View TAB – These options allow you to turn layers on and off so you can get a better look at the map you're creating. Note that these options don't actually delete anything – they just determine what elements are currently visible. For example, if you want to turn off the access layer to get a better look at the map as it would appear to the player, you can do it here.

Clear TAB – If you want to erase a particular layer or start a new map from scratch, you can do it here. Note that, where applicable, the Tile1 layer will be reset with the currently selected tile. The Replace option allows you to replace a specific tile with another throughout the entire level. For example, if you decide you want to change the ground from sand to grass, this function allows you to do it quickly and easily.

DB1 and DB2 TABS – All the databases can be accessed with these options (see above for list of available databases). Unlike the options in the Def TAB, these are not associated with a particular level (i.e. a monster you created in the Monster database will be available to any level you create).

Util TAB – Here are a bunch of miscellaneous utility functions:

• **Flag Search –** These functions allow you to track down a particular flag (i.e. where is it used or referenced) or list currently unused (available) flags. Note that these functions need to load all the currently existing levels, one after the other, so you'll need to save your changes to the current level before using this (otherwise you'll lose

changes you made). At the end, the level you're currently editing will be reloaded (the whole process only takes a second).

• **Item Search –** If you're trying to locate where a particular item is located, this function will track it down for you. E.g. if you can't remember where you placed the 'Widget of Fear', this is the function you need. As with the flag search option, you need to save your changes before using it.

• **Tile Order –** This function allows you to select any tile (picture tile, access tile, zone tile, event tile, misc tile). The tiles are displayed in chronological order and are not sorted (like they are if you select them using the 'T', 'A', 'Z' or 'E' buttons). You'll probably never use this function.

• **Shift Level –** If you decide that the entire level needs to be shifted to the right one tile, or something like that, this function allows you to achieve it.

Other Miscellaneous buttons:

• **Map Overview button:** This option simply gives you a overview of the entire level on a single screen (and looks exactly like the map option that the player has access to ingame). You can also use this function to quickly 'teleport' to a particular area on the map or 'draw' outlines on the map (useful for roughly sketching a coastline etc).

• **Auto Access button:** When switched on, the editor automatically places access tiles when you place picture tiles on the Tile1 or Tile2 layers. For example, if you're placing a wall tile, the editor will assume that it's a blocked access. This function will get it right most of the time, but not always (when you use it, you'll usually need to switch it off at the end and make some manual adjustments).

Other Miscellaneous functions:

• **Copy and pasting events:** Events can be quickly copied and pasted. Place the mouse curser over the event you want to copy and press CONTROL+C. Move the mouse curser to the place where you want it pasted and press CONTROL+P.

• **Flooding areas:** If you want to flood an area on the map, place the mouse cursor in the desired area and press CONTROL+F. The tile that is used to flood the area, is the one that is currently selected (highlighted). This function is not reversible and it is advised that you experiment a little first to understand how it works, before using it 'for real'. This function can be used to flood access and zone layers too.

Note: The Tile1 and Tile2 layers are treated entirely separately for the purposes of flooding – if you have 'top layer' selected, you're only flooding within the top layer. If you have 'bottom layer' selected, you are only flooding within that layer. If a flood appears to go beyond the bounds of the area you intended, it's probably because you're flooding the wrong layer.

3.0 Events

Each event in the editor has a descriptive line so you should be able to work out what you're after. However, here is a list of the most common/important events to get you started:

Simple Door (under the Access tab): If you want a simple door that opens and closes but has no locks etc this is the event you're after. This event has the advantage that it doesn't require the use of a flag (you'll notice that simple doors reset (close) when you leave a level and return or reload a saved game).

Complex Door (under Access tab): Used for locked doors.

Portal (under Access tab): Used to define teleporters, stairs, cave entrances etc This is one of the main methods for transporting characters from one map to another.

Chest (under Goodies tab): Chests contain items and can be locked or trapped.

Search (under Goodies tab): Used for smashing open pots or barrels etc. Can contain an item (or coins) if desired.

Script (under NPC tab): Used for creating NPCs to talk to or hand out quests.

Monster (under NPC tab): Used for placing monsters (whose details are defined in the Monster database) on the map.

Shop (under Shop tab): Allows you to define shops where heroes can by stuff. Other establishments such as Trainers and Vaults are also located under the Shop tab.

4.0 Testing and Troubleshooting

Press CONTROL+U ingame to bring up the cheat menu. This provides a bucket of useful tools to help you test the game. In particular, the Utils tab allows you to teleport from place to place, alter the status of quests and alter flags.

Reset Move Layer:

The move layer is a 50x50 array that is used to hold the current position of all movable agents such as monsters, boats, pushable objects and so forth. Resetting the Move layer will reset all movable objects to their starting points.

When playing the game, the move layer is updated when you enter a new level. The move layer is stored in the character save game file. So if you save your game, edit the level (to add, say, a new monster), then reload the character, the new monster WON'T appear until the move layer is updated. You can achieve this by leaving and returning to the level in question, or click on the **Reset Move Layer** button.

5.0 Monster Editor

The monster editor is fairly straight forward and allows you to create and edit monsters in pretty much any way you see fit.

6.0 Item Editor

Unlike the Monster editor, the item editor is largely predefined. Item 1 is (and always will be) a potion of healing. You can edit the names of any of the predefined items, however it won't change their fundamental nature. I.e. you could rename the Potion of Healing to 'Sword of Doom' but it will still look like and behave like a potion of healing. This is because alot of the item behaviour has been hardcoded. However you may wish to rename some items for reasons of 'flavour'.

Conversely, Quest items (e.g. a key) can be created by the user and an appropriate icon imported by editing the appropriate section in the Custom.bmp file. Quest items are defined in the 'Custom' tab.

7.0 Customization

Need a wall tile that doesn't exist yet in DD2? Need to draw up your own monster or quest item? This section is a brief guide on how to add customized graphics to DD2.

The default graphic tiles in DD2 are imbedded in the .exe file, but you can add your own images. Goto the DD2/Mod/GameData folder and open the file Custom.bmp. Draw your new tiles in the appropriate areas. These new tiles will be available from the relevant 'custom' tabs in the editor.

Defining Acts:

By using the 'Act Title' event, you can set up your game into chapters. Each chapter has an associated picture which can be customized – simply edit the DarkChapterX.bmp in the Mod/GameData/ directory (where X = 1 to 4, for chapters 1 to 4).

Defining Intro Pics:

The pics displayed when you read the game objective (Objective option from the main menu) are also customizable – simply edit DarkObj1.bmp and DarkObj2.bmp in the Mods/GameData directory.

Defining Title page:

DarkTitle.bmp can also be edited to create a unique 'mod' title page.

Can I create custom items or monsters larger than 1x1?

Currently this is not possible. If there is sufficient interest/need, I'll add some larger tiles to the custom import feature.

What are the tiny tiles directly under the wall tiles in custom.bmp?

Each tile also has a miniature (8x8 pixels) representation for the map overview (press M ingame). So the first wall tile is represented on the map by the first miniature wall tile.

8.0 Help

This guide is fairly brief – if I were to cover every aspect of the editor, it would be 50 pages long. However, if you have read this guide and not understood something or have a question, please feel free to e-mail me: llafebre@bigpond.net.au.

Any queries will be added to the FAQ at the end of this document – if one person is having trouble with something, chances are many more are too.

BUGS – Spotted a bug? Report it!

REQUESTED FEATURES – I'm happy to receive any suggestions or requests for added features or events. I can't promise I'll implement every request, but I'll certainly take them all under consideration.

9.0 FAQs

Why do monsters / pushable objects / boats not appear when I play a level I created?

Refer to section 5 above (Testing and Troubleshooting) and specifically to the bit about resetting the move layer.

I've drawn walls on my map, but the hero just walks over them!

To define how the hero interacts with wall tiles, place access tiles (press A or click the 'A' button to select) in the appropriate squares (see above).

How do I set up conversation options using the Script event / Script editor?

The simplest and most intuitive way to learn how to construct NPC conversation trees is simply to load a few of the default campaign levels into the editor and check them out.