

Contents

Preface	1
Introduction	2
Getting Started.....	5
Making Maps.....	10
Adding Traps & Monsters	11

Preface

Back in 2006, Wizards of the Coast published the first set of *Dungeon Tiles*. For me, this was one of the best RPG products out there...system neutral, great looking, and very durable. At pretty regular intervals, new tiles came out and I was good and hooked. Sometime shortly thereafter, a java based tile editor was published in order to help make maps from the tiles. The `dungeon_tiles` group on yahoo was started around this time as well.

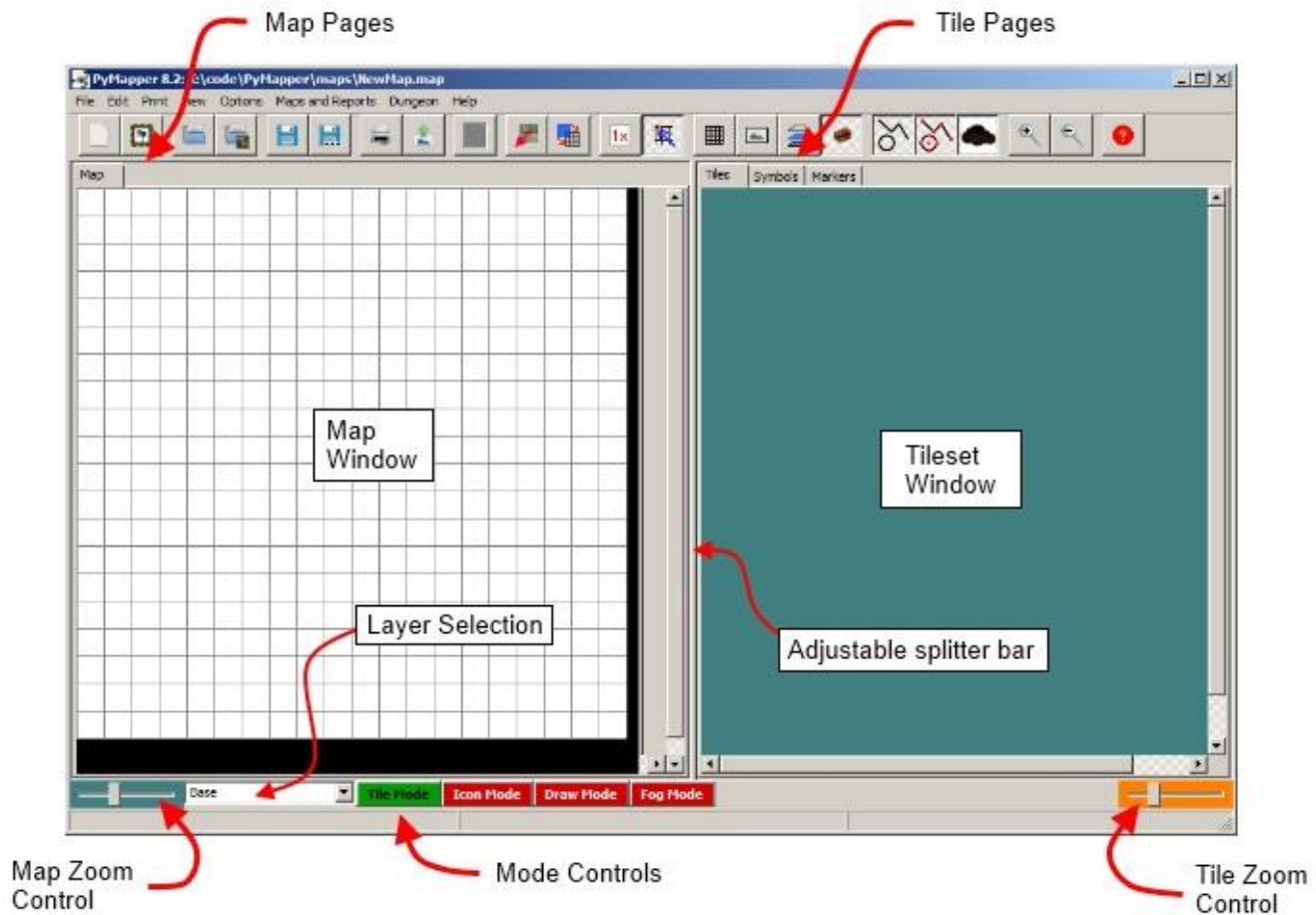
The mapper was decent...you could manage a few sets, and arrange the tiles to make maps in the window. There was even an export feature that allowed you to send a text string to someone else to share your work. But it had limitations. Tiles didn't rotate by themselves, so the mapper had to store an image for each side, rotated in the four directions. New tilesets (which were coming out at a regular pace) were a pain to add. Most importantly, it wasn't maintained by the developer. There were a few bugs, and no real way to address them. So, around the middle of 2009, I started teaching myself the python computer language, and started work on a new program to address some of the mapper's limitations, and to put new features into play. Thus, pymapper was born.

Pymapper has come a long way since the first alpha code was put together. I've tried to keep pymapper simple enough to be user-friendly, but as I've found over the years, what is intuitive to me doesn't mean that it is for everyone else. Hopefully this tutorial will help you out in using the program. As with any software, there are bound to be things you don't like. Please let me know how to improve pymapper. I try to implement the suggestions I receive, and there have been some great ones over the years. I can be reached at pymapper@gmail.com if you have suggestions, bugs, or just general feedback.

Welcome to pymapper! Let's go make some maps!

Introduction

The basic pymapper window looks something like this, depending on your operating system.



There are several principal areas in pymapper:

Tile Window: The tile window displays individual tiles loaded from the tileset files.

Map Window: Tiles from the tile window are dragged here to create maps.

Map Pages: The map window can be divided into individual pages to show different maps

Tile Pages: The tile window can be divided into individual pages to show different tilesets

Map Zoom Control: Adjusting the slider zooms the map window in and out.



Tile Zoom Control: Adjusting the slider zooms the tile window in and out.






















Mode Control: There are several modes to control the behavior of pymapper.

Adjustable splitter bar: Drag side to side to change the size of the map and tile windows

Layer Selection: Change the active layer for tile placement.

The toolbar contains the following tools:

	Create a new map file.
	Create a new geomorph file.

	Open an existing map file.
	Open an existing geomorph file.
	Save map file.
	Save map file with a new name.
	Print map file.
	Export an image
	Undo button (this is dimmed until something happens)
	Import tileset
	Import background image
	Tile quantity selection (cycles between the three images when pressed)
	Toggle snap-to-grid placement
	Toggle display of the gridlines
	Toggle display of the background image (if present)
	Toggle display of the layer manager dialog
	Toggle display of the treasure, monster, npc, and room icons (xml editor icons)
	Toggle display of drawing items on the map (lines, circles, etc.)
	Toggle display of the drawing handles on items
	Toggle display of fog-of-war items on the map window.
	Zoom in
	Zoom out
	Help button

In the location where you installed pymapper, you find some additional folders created on installation of the main program. The default location (on windows) for pymapper is at C:\pymapper\. You may have changed the default folder when you installed the program...at any rate the following folders are found in the installation location:

/artwork/ Pymapper stores the images for buttons and other resources here. Deleting this folder will cause pymapper to crash, so you really shouldn't do it.

/geomorphs/ Pymapper has the option of creating random dungeons from preset tile patterns called geomorphs. Check out the geomorphs section to learn more about them.

/maps/ This is a default folder where pymapper will store the map files you create. You are welcome to store maps anywhere, but this is where pymapper will look for maps by default.

/srd/ Information about various open source game systems is found here. Information about the d20 System Reference Document is included here. Pathfinder and 4th & 5th Edition Dungeons and Dragons will also be located here if they are included at some point.

/tiles/ This folder is where pymapper looks for tilesets. When you download or create a new tileset, it should be stored here in a separate folder. In the default installation, a subfolder named "TST" is included. Images created by Jonathan Roberts are stored here with the tileset definition file (and

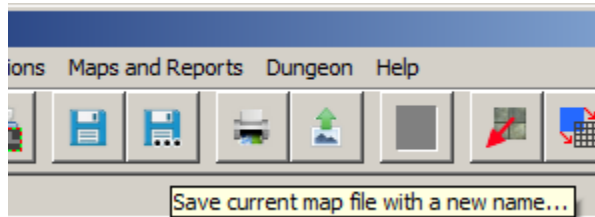
license). Pymapper will automatically look for tilesets here in the /tiles/ folder, and it is the only place where you can place new sets for pymapper to use.

When you obtain a new tileset, you should store the files in a new subfolder of the /tiles/ folder. Each tileset should have its own folder. So, you should have something like this after installing several sets (assuming that c:\pymapper\ is where you installed the program):

```
c:\pymapper\tiles\DN1\  
c:\pymapper\tiles\DN2\  
c:\pymapper\tiles\ET1\  
...and so on.
```

If you are creating your own tileset from images on your computer (more on that later), you don't need to create the folder or copy the images. Pymapper will do that for you as part of the creation process.

Most buttons and items in pymapper have tooltips that are revealed when you place the mouse over the item.



If you find an item that ought to have a tip (but doesn't) let me know at pymapper@gmail.com.

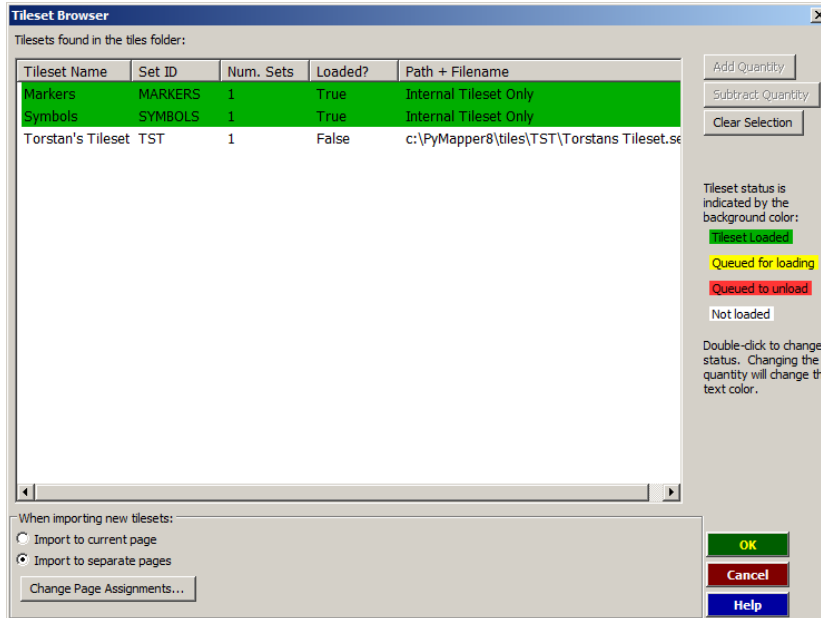
Pymapper only installs files in the installation folder. No modifications are made to the windows registry, and no files are installed anywhere else.

Getting Started

Loading Tilesets

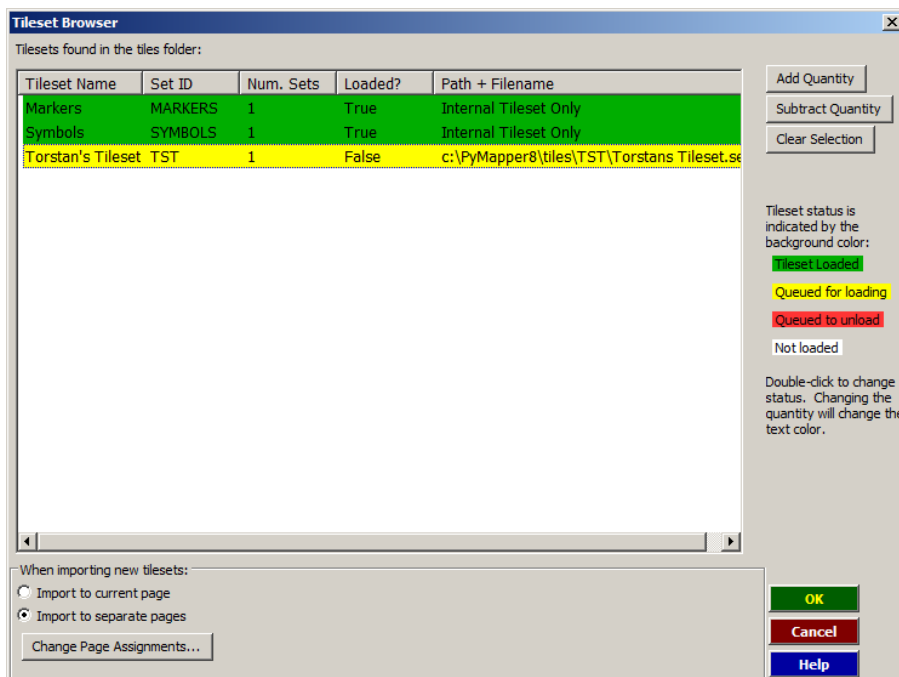
This first tutorial will walk you through the process of importing a tileset, creating and saving a map file, and basic program navigation.

From the *File* menu, select *Tileset Browser*. Pymapper automatically detects tilesets installed in the /tiles/ folder. The following dialog will show up:



The Tileset Browser dialog shows all available tilesets in the /tiles/ folder.

The main window shows all tilesets that pymapper has found in the /tiles/ folder. Lines in green indicate tilesets that are currently loaded and available for use. The *Torstan's Tileset* is installed by default with the pymapper program. To load a tileset, double click on the name, and the selection will change to yellow. Tilesets are loaded after you exit the dialog by pressing **OK**.



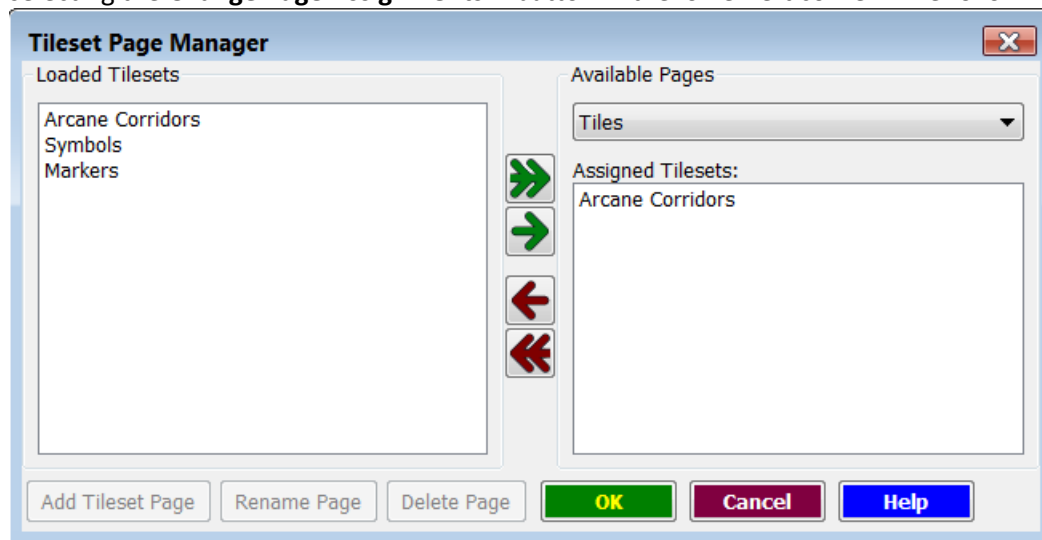
The yellow highlighted line indicates a tileset that will be loaded from disk once you exit the dialog.

If you own multiple copies of a tileset (or just want to use multiple copies of a set), select the tileset and press **Add Quantity**. Obviously, the **Subtract Quantity** button works the other way.

Tilesets can also be unloaded by using the *Tileset Browser*. Double click a green highlighted row, and it will turn red. Once you exit the browser, the tileset will be unloaded from memory.

When pymapper opens tilesets, it imports them to a page in the tile window. New tilesets can be imported to either the current page, or to a new page for each tileset (which is the default action). You can change this option by selecting the appropriate action in the lower left corner. Tilesets imported into separate pages will use the tileset name as the page name by default.

By default, pymapper will import a single tileset to a single page in the tile window. If you want to group several tilesets onto a single page, you can change the assignments for tilesets and pages by selecting the **Change Page Assignments...** button in the lower left corner. The following dialog appears:

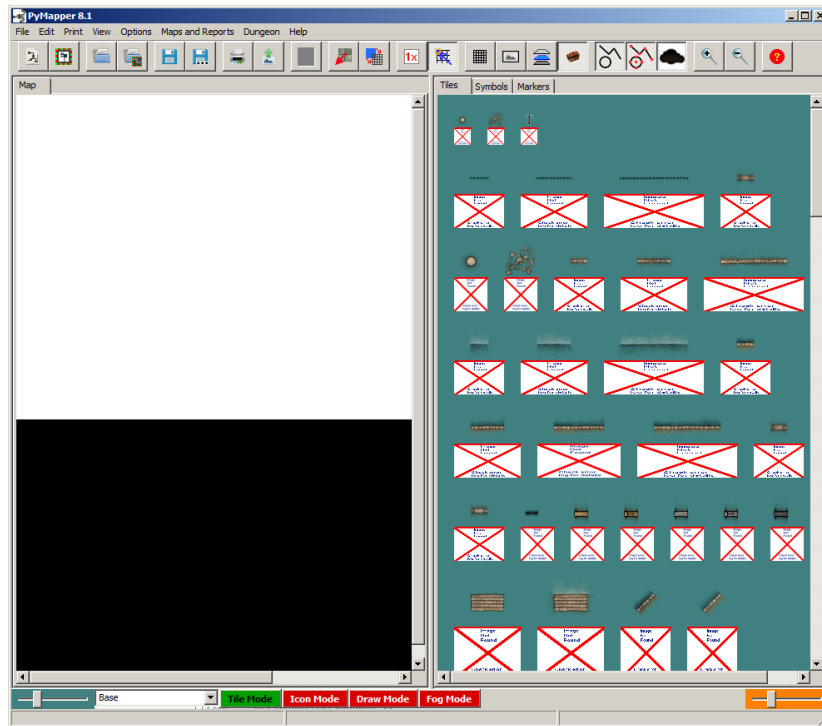


For this example, several other tilesets have been loaded.

On the left side of the dialog, the tilesets you currently have loaded are shown. On the right is a drop-down list of tile pages that have been created. On this dialog example, the Tiles page is shown, and the *Arcane Corridors* tileset has been assigned to it. To change the page assignments, use the buttons between the two panels. The single arrow assigns one tileset; the double arrow assigns all of them.

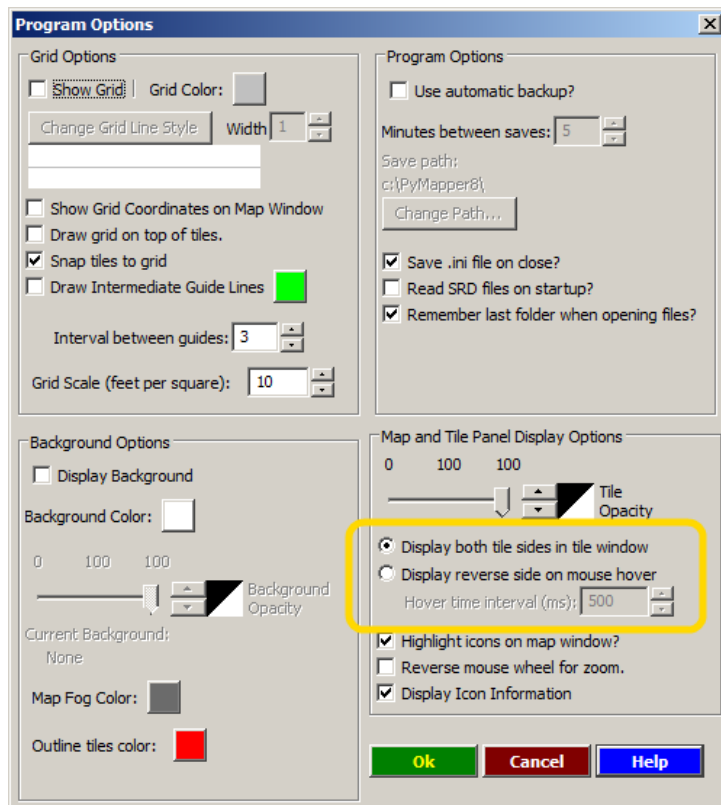
Click OK to exit. The tile window is updated with the tileset images.

Going back to our previous example, we had loaded the *Torstan's Tileset* into pymapper. Exiting the Tileset Browser, pymapper loads the tileset and we have the following:



Error images are displayed when no reverse side image is shown.

Pymapper can display both sides of a tile in the tile window. In *Torstan's Tileset*, there is no reverse side image, so this isn't really an error in our example case. When no image is found, an error image is displayed in place of the reverse side image. We can change the display of the tile window by going to the program options. From the *Options* menu, select *Program...* to bring up the options dialog.

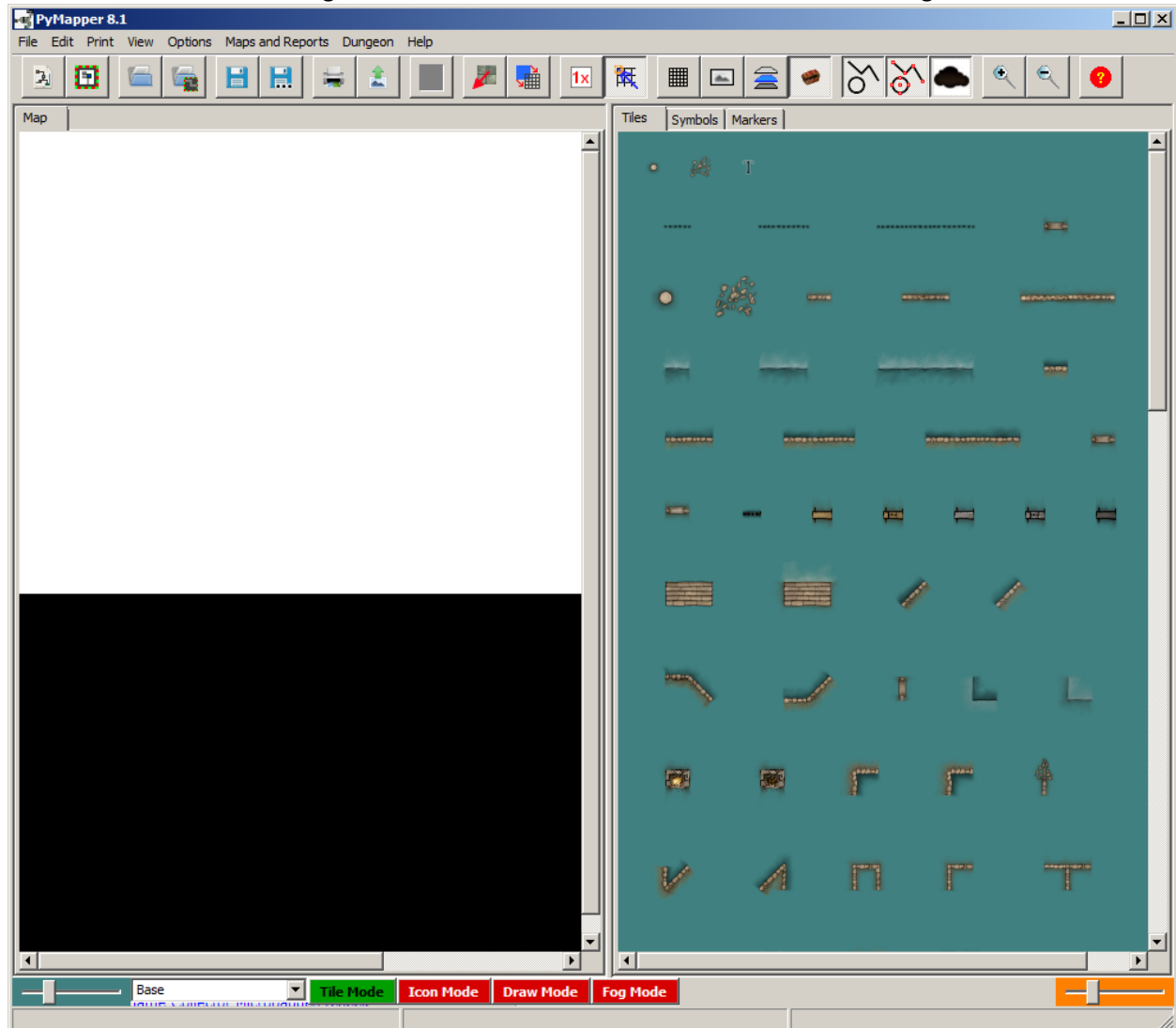


Many different options are controlled in this dialog.

For this example we will focus on changing the behavior of the tile panel with this option change.

In the highlighted area, change the option to the second item listed. This will allow us to only display the reverse side if we hold (hover) the mouse over the tile for a specified period of time. The default is 500 milliseconds (1/2 second) although you can change this value up or down as you like. Click OK and we get a more useable tile panel, as shown below.

The tiles shown now only have the front image shown. To show the reverse side, you can either double click a tile and it will change over, or push the space bar (with the mouse in the tile panel) and all of the tiles will switch over. Holding the mouse over a tile will also show the reverse image.



In addition to the default **Tiles** page, two other pages are part of the initial setup. **Symbols** and **Markers** pages on the tile panel will round out our discussion on the tile panel side of things.

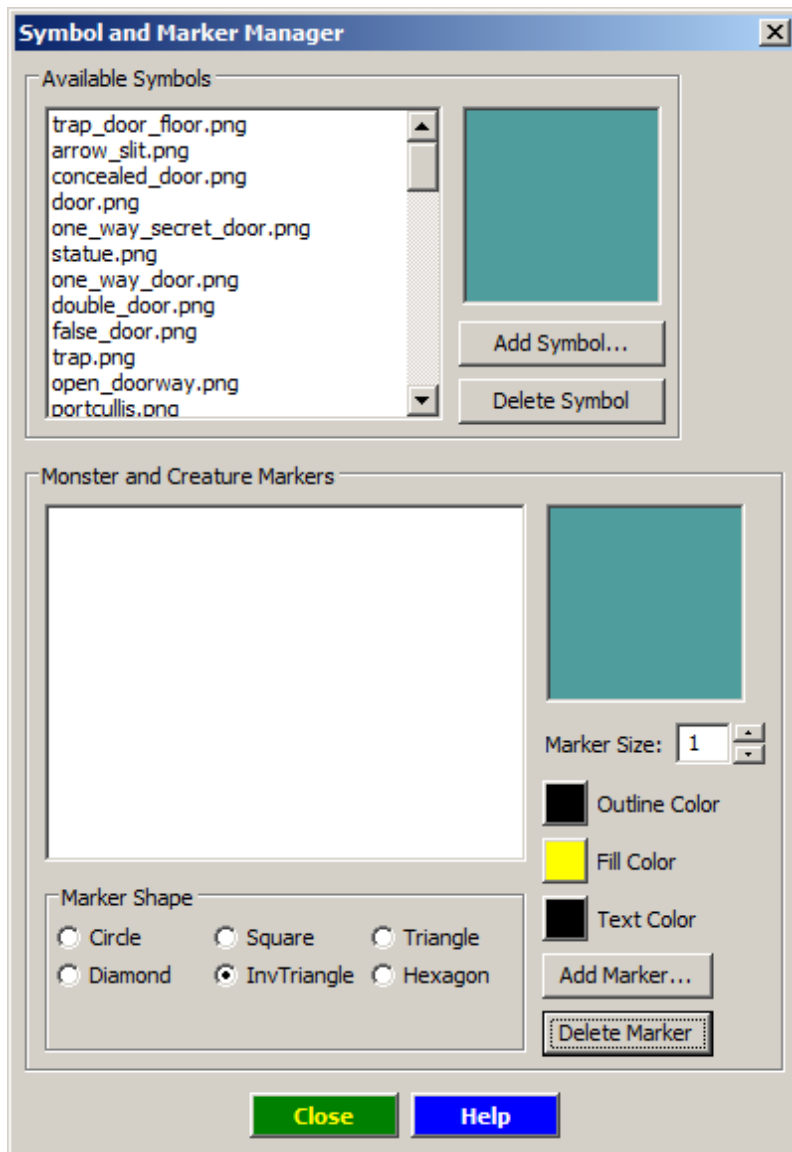
Symbols

Symbols in pymapper have properties similar to tiles. The principal difference is that tiles have no required size or resolution, whereas symbols must have dimensions in multiples of 100 pixels per side. Pymapper automatically determines the height and width of the symbol based on that resolution. Tiles are stored in the /tiles/ folder, and symbols are stored in /artwork/symbols/. All files in the

symbols folder are automatically loaded when pymapper starts (provided they meet the 100 pixel/side requirement) and are placed on the *Symbols* page.

Markers

Markers are user-defined symbols. There are six different shapes available, and text notes can be placed on the marker. To create new markers, select *Symbols and Markers...* from the *Options* menu.



Symbols can be managed through this dialog.

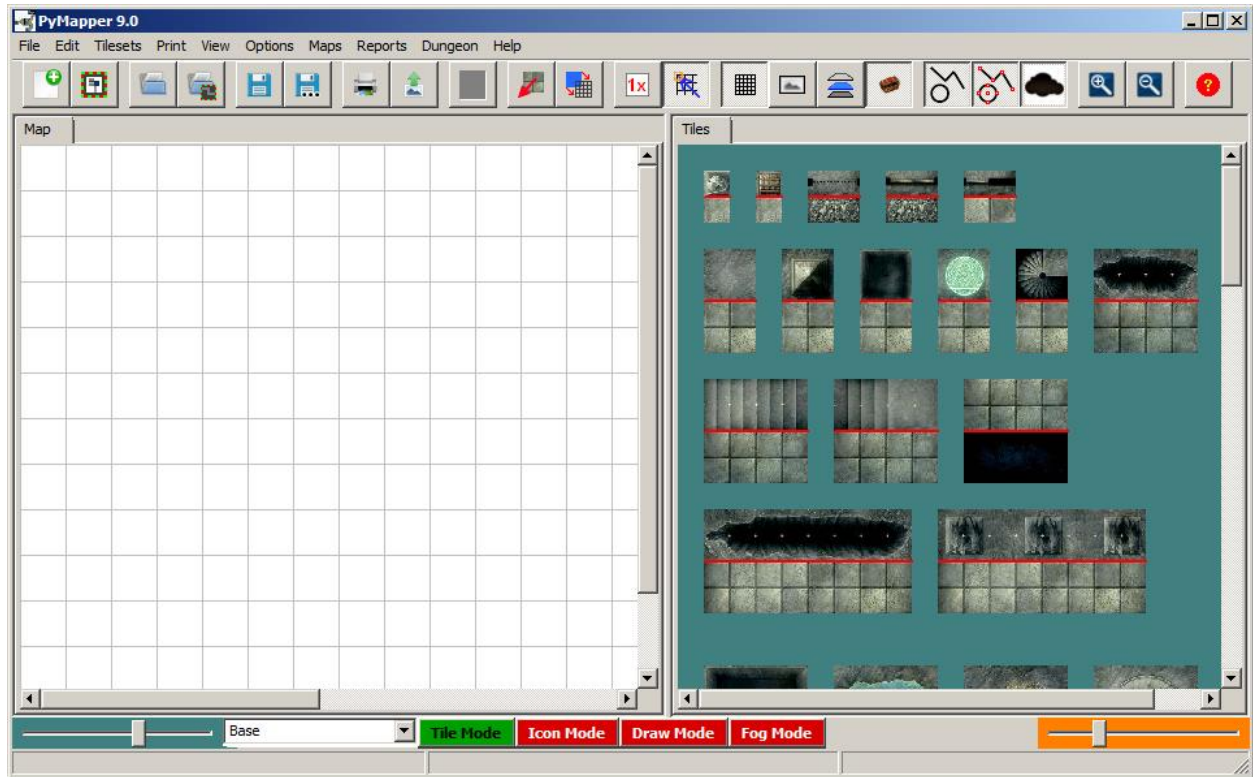
New markers are created with this dialog, and existing markers can be modified.



To define a marker, select the Add Marker... button. A text prompt will come up to set the text on the marker. Simply press OK without adding text if you want a blank marker. To change the properties of an existing marker, select it from the list, and change the option desired.

From a mapping standpoint, markers and symbols act exactly like tiles.

Making Maps

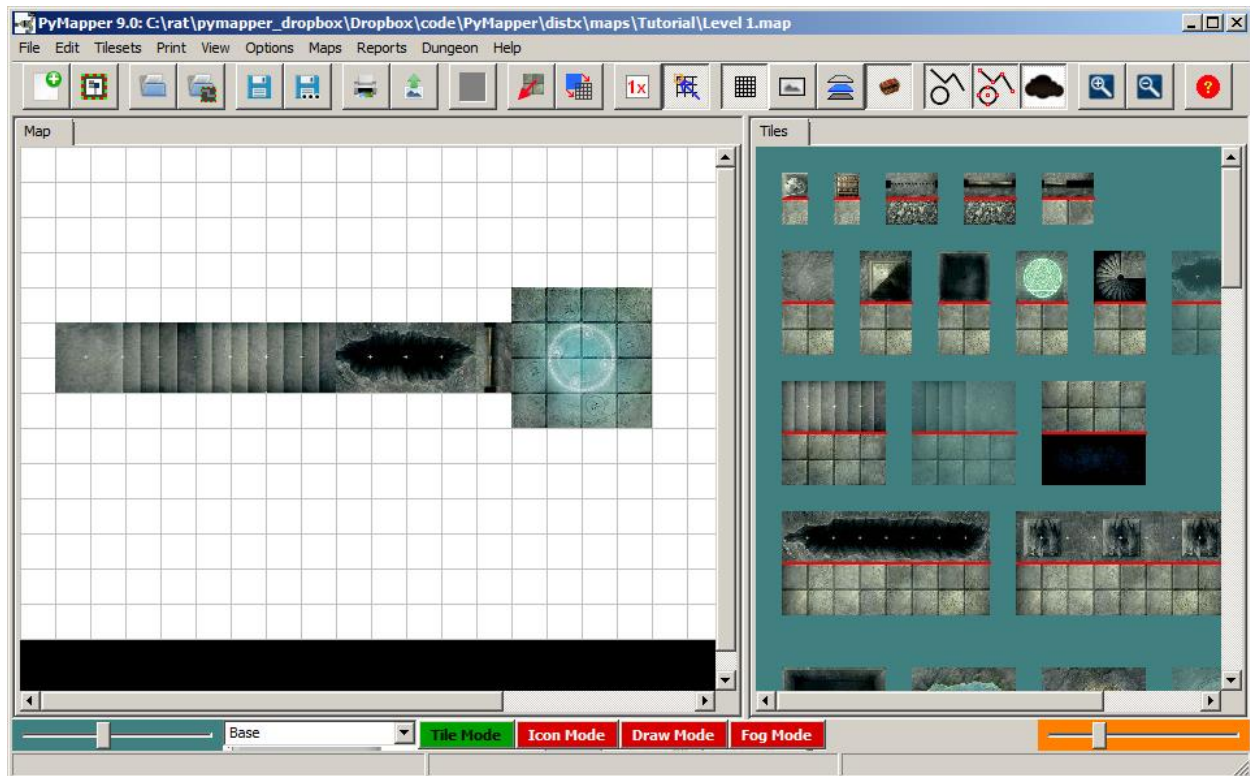
Loading tilesets is pretty cool you have to admit. But probably not the reason that you downloaded and installed pymapper. It certainly isn't reason I coded it. In this section, we'll go over map-making with the program, and several of the options that are available. This tutorial assumes that you have downloaded the *Dungeon Tiles 1 (DT1)* tileset from pymapper.com, and that it is loaded into pymapper. You should see something similar to this:



Pymapper can limit the number of tiles that are used in a map. This may be useful if you are using physical tiles, and you don't want to use more than you actually own. So, we can either make the tile placement limited or unlimited. The limited mode can be a single tile, or based on the number of sets you own. To change between modes, click on the button with the  or  image. You will notice that the tiles in the tile window may dim as you switch between modes.

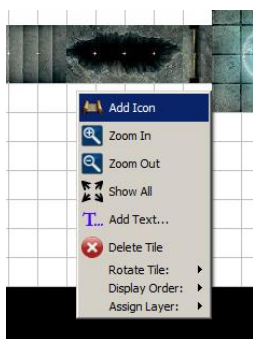
Drag some tiles onto the map from the tile window to create a simple map. If the tiles are dimmed, make sure that the TILE mode is active (it is the green button on the bottom of the window), or that the tile hasn't already been placed.

Alternately, you can open up the map file to match what is shown here in the tutorial. Open the *Level 1.map* file found in the `/maps/tutorial/` folder to start with the following map:



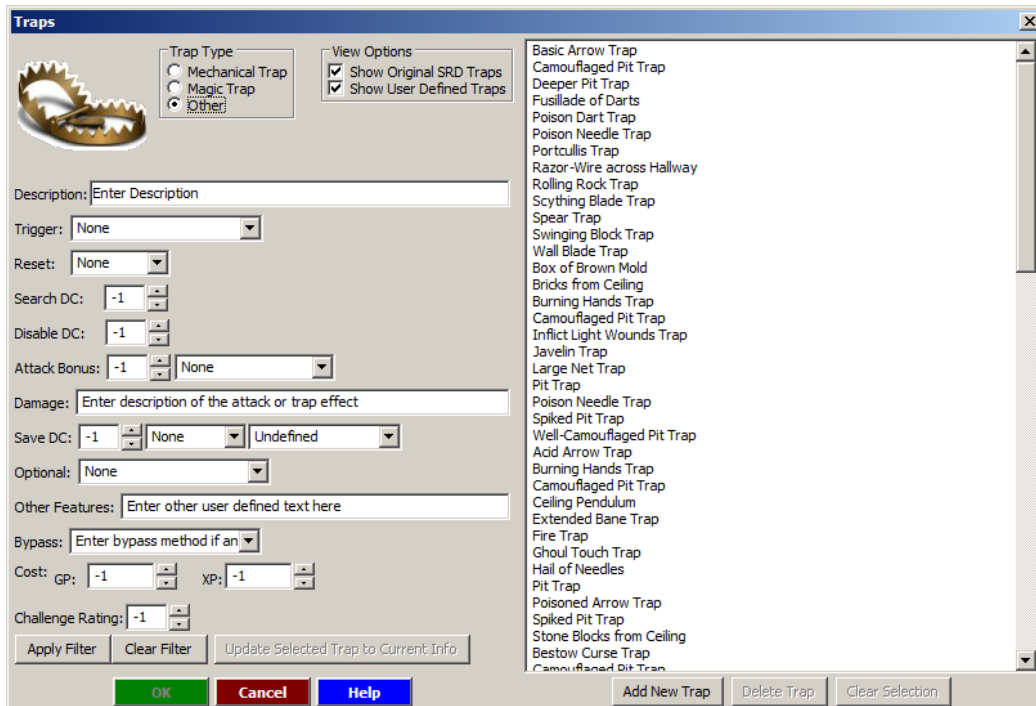
Adding Traps & Monsters

PyMapper comes with support for roleplaying games such as Pathfinder, and 5th Edition Dungeons and Dragons. We can add some additional map information to help the DM run the adventure. To add this supporting information, open the **Dungeon** menu and select **Load Dungeon Resources**. You can also change the program settings to load these automatically (using **Options|Program...**) when you start pymapper.



In this simple dungeon, we are going to set a trap for the characters as they come down the stairs. The previous inhabitants wanted to protect the summoning circle from unwelcome visitors, so they hid a pit trap at the base of the stairs. Right click on the map near the pit, and select **Add Icon** from the popup menu. Type in “Pit Trap” in the description prompt, and select OK. A yellow highlighted icon will appear on the screen where you clicked, and a text editor will appear.

Along the top menu row of the text editor, there is a menu named **d20 Item**. Select **Add Trap...** from this menu, and we will get a list of traps that are available in pymapper, as shown in the dialog box below.



From this list of traps, select “Camouflaged Pit Trap” near the top of the list. Selecting the OK button will return you to the text editor, and the text of the trap will have been added to the editor.

Note a series of buttons at the top of the text editor:



These buttons

change the type of icon that is displayed on the map screen. By default, the scroll is selected when you create an icon. When you add monster, NPC, trap, or treasure information from the d20 Item menu, the icon automatically changes to the appropriate symbol. The last button (the circled A) allows for custom symbols to be placed on the map as icons.

The text editor can remain open while you are using the other features of pymapper. Hold the mouse over the icon in the map window to show the description of the icon. While in **Icon Mode**, double click the icon to open it up. While in **Tile Mode**, hold down the CTRL key and double click to open the icon.

By default, the background highlighting color of icons is yellow. For traps, I like to change the background to red to indicate a trap that has not been sprung, and green for disabled traps. Right click on the icon, and select **Change Highlight Color...** to use a different color for the highlighting.

For those players that get past the trap, there is a locked door that protects the magic circle. It’s pretty simple, so while we could create another icon to describe it, instead we’ll just add a text annotation. Near the door, right click and select **Text Annotation...** from the popup menu. In the prompt, type in “Locked Door” to help remind us that the party can’t just barge in. Size, font, and background can be changed by another right-click on the text, or by going to **Options | Text Annotations** in the main menu.

There is a monster waiting just past the doors. We could just add a symbol, or a text annotation, but for this example we want to have the monster stats available to us as well. Right-click on the magic circle tile, and select **Add Icon**. At the description prompt, type in 'Ape' and click OK, and we get the icon text editor again. From the **d20 Item** menu, select **Add Monster...** if you are playing D&D 3.5 or Pathfinder, or select **Add 5E Monster...** if you are playing the Fifth Edition D&D RPG¹.

This is the Pathfinder/3.5 Edition monster information dialog. Select the desired monster from the list on the right. The monsters can be sorted by either CR or Monster type by selecting the appropriate filter. To find the Ape, select the CR filter, and choose 2.

¹ A note about the availability of monsters for 5th Edition: Wizards of the Coast has not yet released public domain or open gaming content for the latest revision to Dungeons & Dragons. The monster included with pymapper is taken from the DM Basic rules guide available at www.wizards.com. When (if?) Wizards releases open game content, additional monsters may become available in future updates of pymapper. Until that time, feel free to add your own monsters to pymapper.

Fifth Edition Monster Editor

Monster Name:

Type:

Size:

Challenge Rating:

XP: Alignment:

Armor Class: Passive Perception:

HP: Hit Dice:

Speed:

STR: DEX: CON: INT: WIS: CHA:

Attacks:

Actions:

Other Features:

Spells:

New Monster Add to Master List Update Monster Delete Monster

Close Cancel Help

Filters

☐ CR ☐ Monster Type

<1
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Aberration
Beast
Celestial
Construct
Dragon
Elemental
Fey
Fiend
Giant
Humanoid
Monstrosity
Ooze
Plant
Undead

Ape

This is the 5th Edition monster information dialog. Select the desired monster from the list on the right. The monsters can be sorted by either CR or Monster type by selecting the appropriate filter.

Once you have selected the monster, close the dialog. The monster's ID number (internal and unique inside pymapper) will be displayed in the text editor box. Close the text editor (it will prompt you to save if you haven't already) and a monster icon with a green background will appear on your map. When you move the mouse over the icon, an information dialog will pop up, showing you the details of the monster.

Pathfinder / d20 / 3.5 Edition Monsters

Monster Type:

Monster Name:

Type:

Size:

Challenge Rating:

Alignment:

Hit Points: HD:

Initiative:

Speed:

Armor Class:

Base Attack: Grapple:

Space: Reach:

Attack:

Full Attack:

Special Attacks:

Special Qualities:

Special Abilities:

Treasure:

The Pathfinder/3.5 monster information box shows the details for the monster we have selected.

Ape

19 Full HP: 19

AC XP: 100 PP: 13

Speed: 30' /30' Climb

Attacks

Melee: Fist +5, reach 5', one target. Hit: 6 (1d6+3) bludgeoning.
 Ranged: Rock +5, range 25'/50', one target. Hit: 6 (1d6+3) bludgeoning.
 Multiattack: The ape makes two fist attacks.

Athletics +5, Perception +3

Ability Scores and Modifiers

STR:16(+3) DEX:14(+2) INT:6(-2) CON:14(+2) WIS:12(+1) CHA:7(-2)

Conditions

Spells

The 5th Edition monster box shows information for the monster, and has an extra feature of being able to track the hit points of the creature. Click on the green HP button to re-roll a new HP total for the monster based on its hit dice. You can directly edit the total, use the spinner control, or press the -1, -5, or -10 buttons to change the hit points.

You can load the file "[Level 1 Complete.map](#)" from the tutorial folder to see the completed map.