Pro-D Total 4.3 Manual

What is Pro-D Total?

Pro-D Total gives you the essential tools for making games with procedurally generated content.

Pro-D Total's main features include random map generation, A* pathfinding, turn-based progression, object distribution, field of view and fog of war. Initially meant to satisfy the needs of roguelike developers Pro-D Total is now a tool for all games that include procedurally generated environments.

Pro-D Total is the most extensive version of Pro-D packages and includes all functionalities.

How to use this Manual?

Use 1: If you want to see the **online documentation** go to this link. We used Doxygen, a free documentation tool to make proper documentation possible for you.
Use 2: If you read further you will find a quickstart, an in depth explanation of how Pro-D works, a FAQ, Contact Info and an Update Log.

QuickStart

Pro-D Total contains an all in one test scene in the folder "**Pro-D/Scenes**". This scene will present you with a GUI that lets you explore most of the main features. **We highly encourage you to run the test scene and play with it**!

You should now the following before diving into the code:

1. The test scene uses the **Pro-DManager.cs** and the **Materializer.cs**. This shows a clear **distinction between generating the dungeon and actually showing it.** The GUI in the test scene works in two steps. First calling a world map generation via: Generator_Generic_World.Generate(). Then calling a prefab placement via: Materializer.Instance.MaterializeWorldMap()
2. Inside a World Generator, for example Generator_Generic_World.cs, a world is created which can consist of multiple maps. Each map is created using a Map Generator, for example Generator_Dungeon.cs.
3. **These generators are static**, so they don't need to be instantiated. They also serve as examples for you, on how to make your own generators. Each of them uses methods from the static **MethodLibrary.cs**.
4. **MethodLibrary.cs contains all methods for creating rooms, caverns, doors etc.** You should see these methods as building blocks and we encourage you to experiment with them and write your own map generator scripts. All the methods of the method library are carefully commented and

documented in our online documentation here.

How does Pro-D Total work?

1. **Cells, Maps, WorldMaps**
    a. **Worldmaps are made out of maps.** For example a 2 by 2 worldmap can be called "the Ancient Dungeon Ruins", and its maps can consist of "Dungeon Ruins - Entrance", "Dungeon Ruins - Maze" and "Dungeon Ruins - Lich's Lair" with one extra space left for another map.
    b. **Maps are made out of cells**. For example "Dungeon Ruins - Maze" will be a map of 49 by 49 Cells and will have Wall, Path, Trap, Entrance and Exit as some of these cells.

2. **Generators and the MethodLibrary.cs**
    a. **WorldMap Generators use Map Generators.** For example the worldmap "the Ancient Dungeon Ruins" can use Generator_DungeonRuins.cs and Generator_Maze to make its aforementioned individual maps.
    b. **Map Generators always use MethodLibrary.cs methods**: The MethodLibrary.cs consists all fundamental methods in making maps. For example Generator_DungeonRuins.cs uses the following and more:
        i. MethodLibrary.CreateRooms(...);
        ii. MethodLibrary.CreateMaze(...);
        iii. MethodLibrary.SetCellsOfTypeAToB(...);
        iv. MethodLibrary.CloseDeadEndCells(...);
        v. MethodLibrary.ReduceUCorridors(...);
        vi. MethodLibrary.ConvertUnreachableCells(...);

3. **Materializer.cs**
    a. **Materializer.cs uses the theme of a map and the resources folder to instantiate game objects.** If a map has its theme as "Terminal Theme", then the Materializer.cs will look into **"Pro-D\Visual Assets\Resources\Terminal Theme\Cells"** and search for respective Prefabs such as Wall, Path, Trap, Entrance and Exit. Please look into the necessary naming convention by navigating to aforementioned folder. You can easily use an atlas and assign your prefabs accordingly as seen in the Stone Theme or you can use individual tiles without an atlas as seen in the Terminal Theme.

4. **FogOfWar.cs**
    a. The FogOfWar.cs has to be on the same GameObject where the Pro-DManager.cs is. To see how it is set up, have a look at the ExampleSceneGUI.cs. The TurnBasedPlayerMovement.cs updates the FogOfWar everytime it moves. Other than that, it works just by itself.

5. **PathFinding.cs**
   a. The Pathfinding.cs has to be on the same GameObject like the Pro-DManager.cs. To see how it is set up, have a look at the ExampleSceneGUI.cs. The TurnBasedPlayerMovement.cs updates the pathfinding everytime it moves. Other than that, it works just by itself. To use pathfinding in your own scripts, use the PathFindingAlgorithm.cs.

6. **TurnManager.cs**
   a. The TurnManager.cs manages a list of actors, they're all required to implement the TurnBasedActor interface. The TurnManager.cs provides functions for adding actors to and removing actors from a scene . He calls the startTurn function for the actors in the correct order and expects the actors to call endTurn when they are done.

F.A.Q.

1. **Can I use 3D models with Pro-D?**
   Yes you can. Check out how our Materializer.cs works and explore the Resources folder for more. Read the Materializer.cs header in the manual.

2. **Does Pro-D generate maps in Play Mode or before I press play, in the Editor Mode?**
   a. **Pro-D can do both**, **however**, the test scene in the package will only let you generate maps in the play mode, that is after you press Play icon in Unity.

   b. Pro-D works in two steps. First step is generation and second step is materialization. You are capable of generating maps without viewing them if that's what you wish.

   c. If you want to generate maps on the Editor Mode **you have to write your own Editor script** to do so and it is possible since we've done it on earlier versions.

3. **I can't get pixel-perfect textures in my game. How do I resolve this issue? (Pre Unity 4.3)**
   a. Your texture size, your object's scale and your resolution need to be in sync for getting pixel-perfect images in Unity3D. Follow these steps:
      i. Switch your camera to Orthographic. This will display everything in 2-D.
      ii. Go to import settings of your textures and raise their compression to a higher resolution. Your texture's Filter Type should be Point and its Format should be highest possible.
      iii. Power of 2 is your friend. You get warped pixels when they need to split into half. For example I use a 32x32 pixel PNG file that is attached to a

prefab with scale of X:32 Y:1 Z:32. I display this in a resolution of 512x512 using 512 as my camera size.

iv. Go to Unity>Edit>Project Settings>Quality and Disable Anti Aliasing option.

v. If these don't help you then google and unity forums are your friend.

4. **What's your e-mail?**
   Unity3D@graylakestudios.com

5. **Is there a community forum where I can make posts and communicate with the other users?**
   Yes! We finally have a forum. We are using Reddit to host our subreddit and our community is located at r/Pro-D.

6. **I e-mailed you but you didn't reply. What's up with that?**
   a. If for whatever reason you're not getting answers, we highly recommend you try the community forum subreddit at r/Pro-D. There are people who run into similar problems.
   b. I usually reply immediately. I might be very busy due to the fact that I'm working at my own start-up. Feel free to e-mail me again in case I missed it. I would give it 24 hours and not expect a reply over Sundays.
   c. Also we don't receive notifications for reviews you've made in the asset store. This means if you complain about something there and we may not see it until later. I kindly advise you to make a post in the subreddit first, e-mail us your issues second since we want you to get help asap.

7. **Do you guys have a video tutorial or other media teaching Pro-D Total?**
   We will be making videos very soon. Check for video links on our page.

I want to contact you!

Here are the steps you should follow:

1. Post your issue/feedback/question at the forums here.
2. E-mail us at Unity3D@graylakestudios.com and tell us how we can help you.
3. Do **NOT** put a review on the asset store asking something because we don't get notified since **Unity doesn't send a notification e-mail to us when you make a comment!**

Update Log for Pro-D Total 4.3

If you want to keep posted about all upcoming updates and contests then you may join the mailing list here.

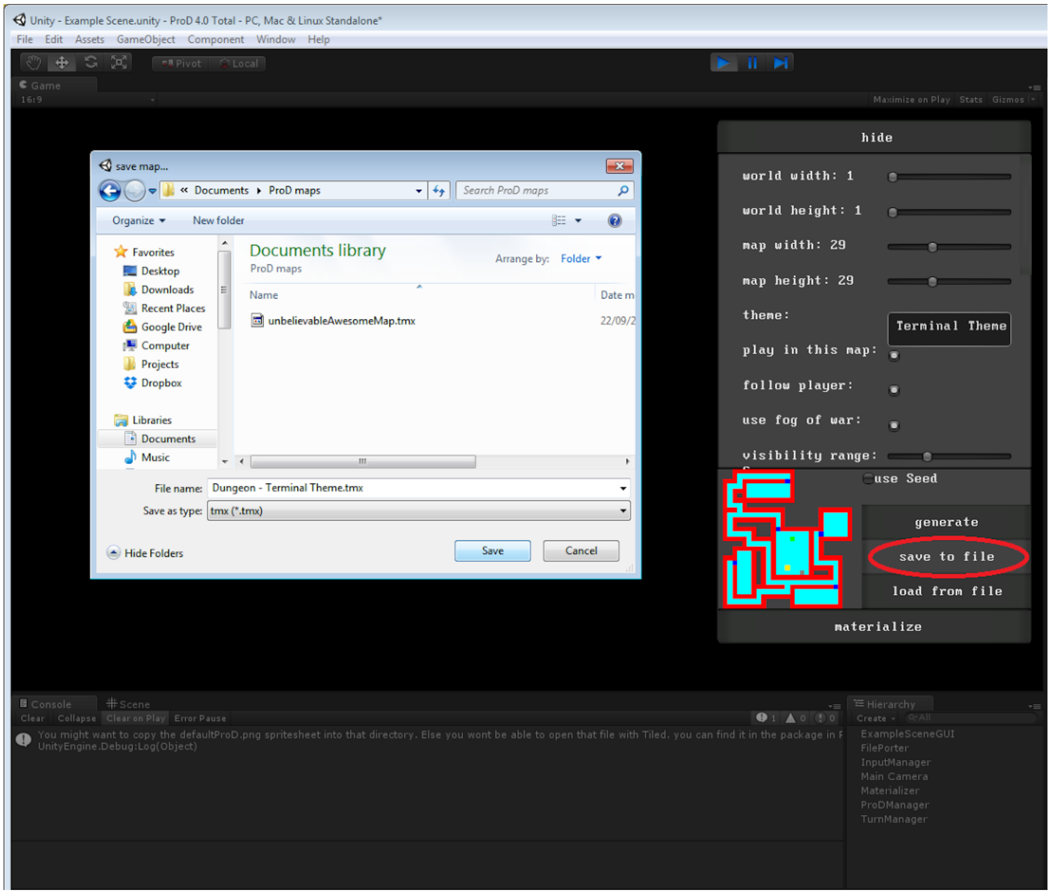
1. Added a new method that places stairs according to a minimum distance. Now staircases aren't placed next to each other and their distance can be adjusted.
2. Added a new generator type called Generator_Castle.cs. This new generator uses some old and some new methods to generate castles.
3. Added weights to the RandomTexture.cs and RandomSprite.cs. You can now assign weights to multiple visuals for one type of tile.
4. Added some more methods into the method library.
5. All themes now include outside path type of cell.
6. Various bug fixes.

Addendum: Pro-D Tiled

Let’s find out how you can use Tiled the popular free tile editing software to edit the maps you make in Pro-D. Since Pro-D 4.2 you can export your maps as .tmx files. These files can then be edited to give your procedural maps some finishing touches by hand. We assume that you have Tiled already installed. If not here’s the free download link for Tiled.

Step 1: Export your map.

The example scene that comes with Pro-D offers a quick and easy interface to generate maps. Adjust the settings to your liking and hit generate. You will see a preview of the map in the lower left corner of the HUD. Right next to it you will see the button “save to file”. When you press this button to save your file, a window should appear where you can now select the save location. Note that this only works in the editor mode of unity, because the file save dialog is provided by the Unity EditorUtility.



Unity - Example Scene.unity - ProD 4.0 Total - PC, Mac & Linux Standalone*

world width: 1
world height: 1
map width: 29
map height: 29
theme: Terminal Theme
play in this map:
follow player:
use fog of war:
visibility range:

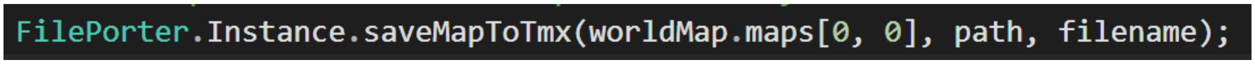
save map...

File name: Dungeon - Terminal Theme.tmx
Save as type: tmx (*.tmx)

unbelievableAwesomeMap.tmx 22/09/2

You might want to copy the defaultProD.png spritesheet into that directory. Else you wont be able to open that file with Tiled. you can find it in the package in F
UnityEngine.Debug:Log(Object)

You can also export your maps from your own code. To do this you need to have a GameObject with the FilePorter.cs script attached in your scene. Then it is just a simple method call:



```
FilePorter.Instance.saveMapToTmx(worldMap.maps[0, 0], path, filename);
```

The three parameters are the map you want to save, the path and the filename. If these parameters are alien to you, “read the section How Does Pro-D Total Work?” in the manual.

Step 2: Open exported map with Tiled.

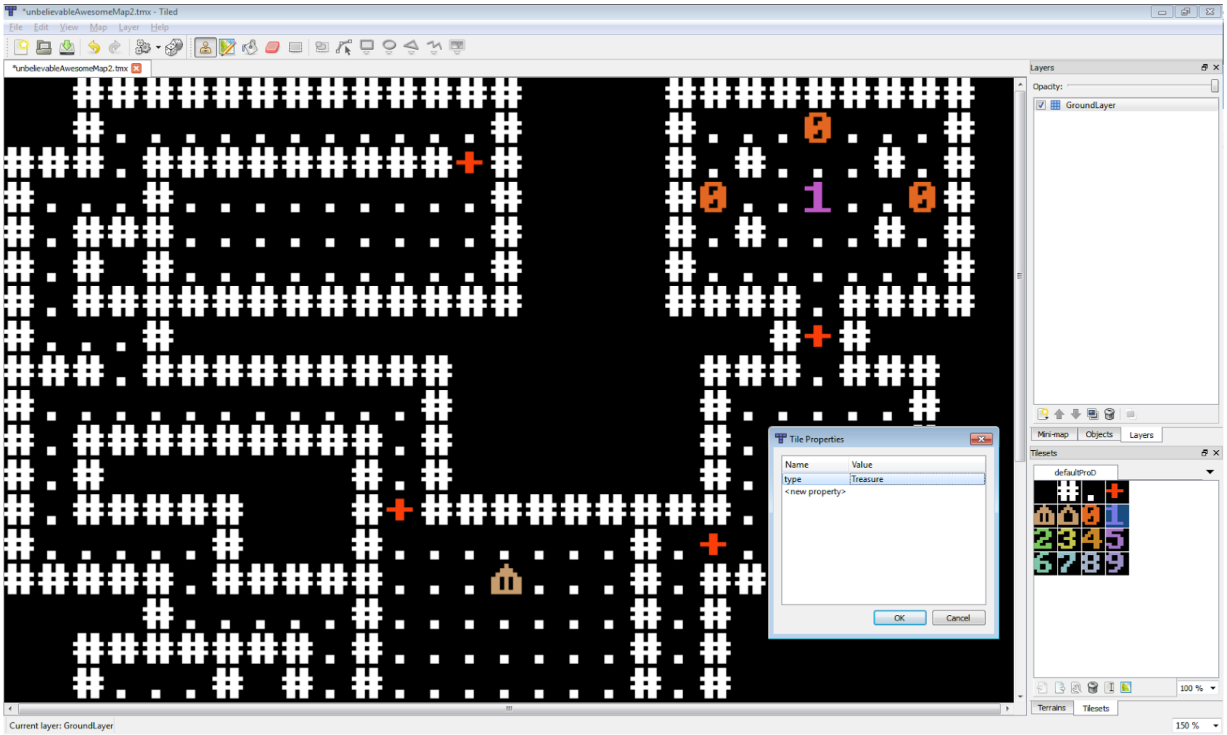
The .tmx file that was just created contains the data Tiled is seeking however .tmx file alone does not specify how the different types of tiles look like. In simpler words you need a tileset to view your .tmx with. Therefore we need to have a sprite sheet in the same directory to be able to open the file with Tiled. You can find the default sprite sheet in your package in “Pro-D\Visual Assets\FilePorter\defaultPro-D.png”. When you have copied the spritesheet into the same directory as your .tmx file, you can doubleclick the .tmx file to open the map in Tiled.

Step 3: Edit the map with Tiled.

You can now edit the map to your heart’s content! Just pick the tile you want to draw with in the lower right corner, and draw directly onto the map. As you can see, the default spritesheet supports the standard types (Abyss, Wall, Path. Door, Exit and Entrance), as well as 10 additional types.

If you created a Dungeon or a DwarfTown with the Pro-D default generators, the “0” will already be assigned with “PathWithGoblin”. This tile is used as spawn locations for the default goblins we used.

When the map you exported had more types, they are assigned to the additional types. When you want to define a new type in the editor, just right click on a unused type, and open the properties of the tile. There you can set a new property with the name “type”. The value is then your new type. Don’t forget to save your changes after you are done and always keep backups!



Name	Value
type	Treasure
<new property>	

Step 4: Import

You can either press the “load from file” button in the example scene, or call the corresponding method in your own code.

```
Map map = FilePorter.Instance.loadMapFromTmx(path, filename);
```

At this point you should make sure that every custom type you defined is supported by your theme to avoid blank tiles appearing. Now you are ready to materialize the map, press Materialize or run your own code to view your changes and enjoy!