PreProduction Naming Conventions for Game Maker

Neal Mehai

Daniel had posted on the facebook group what naming conventions he wanted. I spoke with the other programmers and they all agreed on Daniel's naming convention. I believe that letting the programmers decide on the naming conventions was the best choice. My reasoning is that the programmers are the first to work on our prototype and will work on it the longest.

This was Daniel's post on GPW that I have pasted here for the rest of the team to see just in case they missed it the first time.

" **NUMBERS**

Numbers should only be used when it makes sense that two objects should have the same name, but require a differentiating identifier.

Eg: You would like to call two different objects ***airplaneFastAgressive***. Only here should numbers be used to label them as: ***airplaneFastAgressive1*** and ***airplaneFastAgressive2***

CASCADING NAMING FORMAT

Names should clearly represent where the object belongs in relation to other objects, while maintaining a name that makes sense to the object. To do this objects should be named starting with their most general quality, followed by camel case (as opposed to underscore, which will be mentioned why later) descriptors of more in depth qualities. Once all necessary qualities are described, then and only then should the specific name of the object occur.

Think of this like a folder system on your computer. When determining how to place a picture you downloaded, the detail of the location of the picture increases as you go further down the list.

Eg: Pictures > Facebook > 2012 > TrackMeet > "nameOfPicture"
Now relating this to code...
object > Units > Towers > Fire > "type"
becomes...
Eg: ***obj_unitTowerFire1***

STATE DESCRIPTORS

Finally, at the end of everything any other necessary descriptors should be placed, separated by underscores in order to represent the end of a name, and the beginning of a different specific detail.

For example, in Game Maker, it is common to have multiple scripts that are called during different times, such as a script that is called upon creation of an object and then another script for that same object called during each step of its lifespan.

Eg:

scr_myScript_parent_create
scr_myScript_parent_step

<u>CONCLUSION</u>

Combining everything all in to one:
Cascading Naming Format: ***unitMonsterFire***
Numbers: ***monsterFire2***
State Descriptor: ***obj_"name"_ghost***

Would become:
obj_unitMonsterFire2_ghost

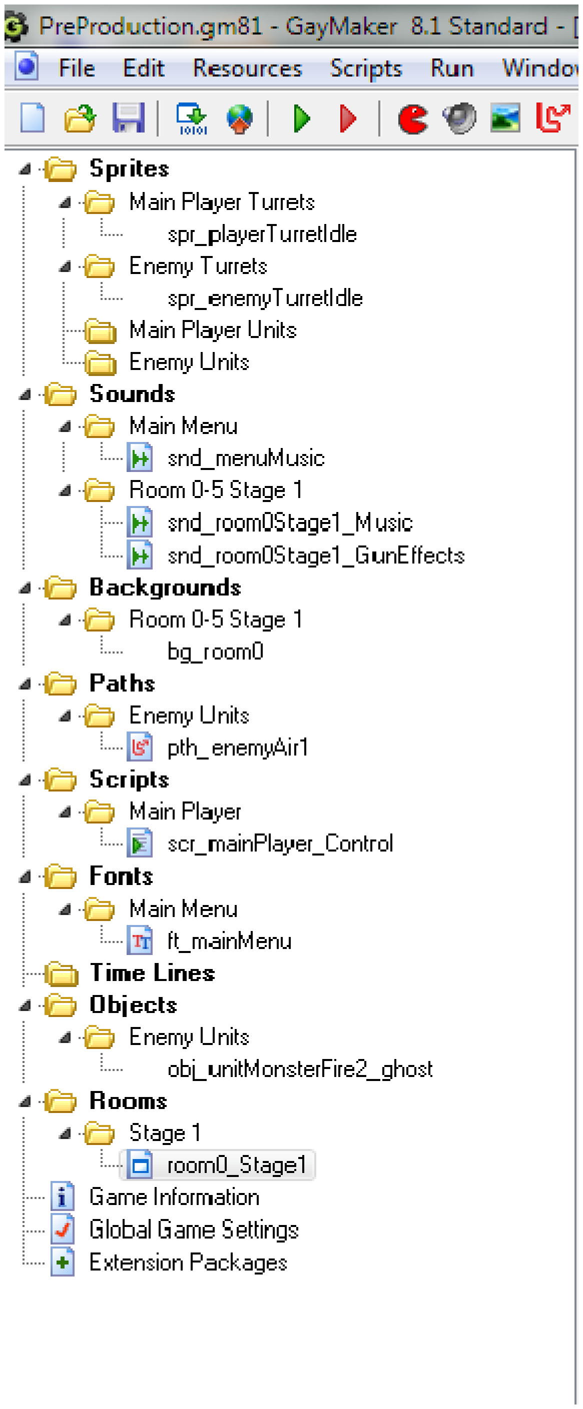
<u>More Examples:</u>

obj_unitMonsterFire2_ghost
scr_myScript_parent_create ”

(* symbol means I (Neal) have added these examples in)
**** spr_playerIdle***
****snd_menuMusic***
****bg_roomStart***
****pth_enemyUnitAir***
****ft_mainMenuFont***
****room0_Stage1***
****room1_Stage1***
******If you are not sure what format to use please refer to a programmer for the correct naming conventions******

This is all of Daniel’s post and all programmers agreed and at this point have started their tasks from the priority task list document. I want the game maker scripts, objects, rooms , etc to be organized from the start. I want subfolders in each category of scripts, objects, etc to be related to enemies, main player, and anything else that can be sub-categorized. Screenshots below should be help indicate what it should resemble.

Sub-Folder Image Example



```
Sprites
    Main Player Turrets
        spr_playerTurretIdle
    Enemy Turrets
        spr_enemyTurretIdle
    Main Player Units
    Enemy Units
Sounds
    Main Menu
        snd_menuMusic
    Room 0-5 Stage 1
        snd_room0Stage1_Music
        snd_room0Stage1_GunEffects
Backgrounds
    Room 0-5 Stage 1
        bg_room0
Paths
    Enemy Units
        pth_enemyAir1
Scripts
    Main Player
        scr_mainPlayer_Control
Fonts
    Main Menu
        ft_mainMenu
Time Lines
Objects
    Enemy Units
        obj_unitMonsterFire2_ghost
Rooms
    Stage 1
        room0_Stage1
Game Information
Global Game Settings
Extension Packages
```