

Game Programming Workshop (part I)

What is RenPy?

RenPy is a game engine used to create Visual Novels.

You can use RenPy to tell interactive stories using 2D images, sounds and music. It has a simple syntaxe based on Python that's easy to learn and enables the creation of simple or complex games.

What will you be doing?

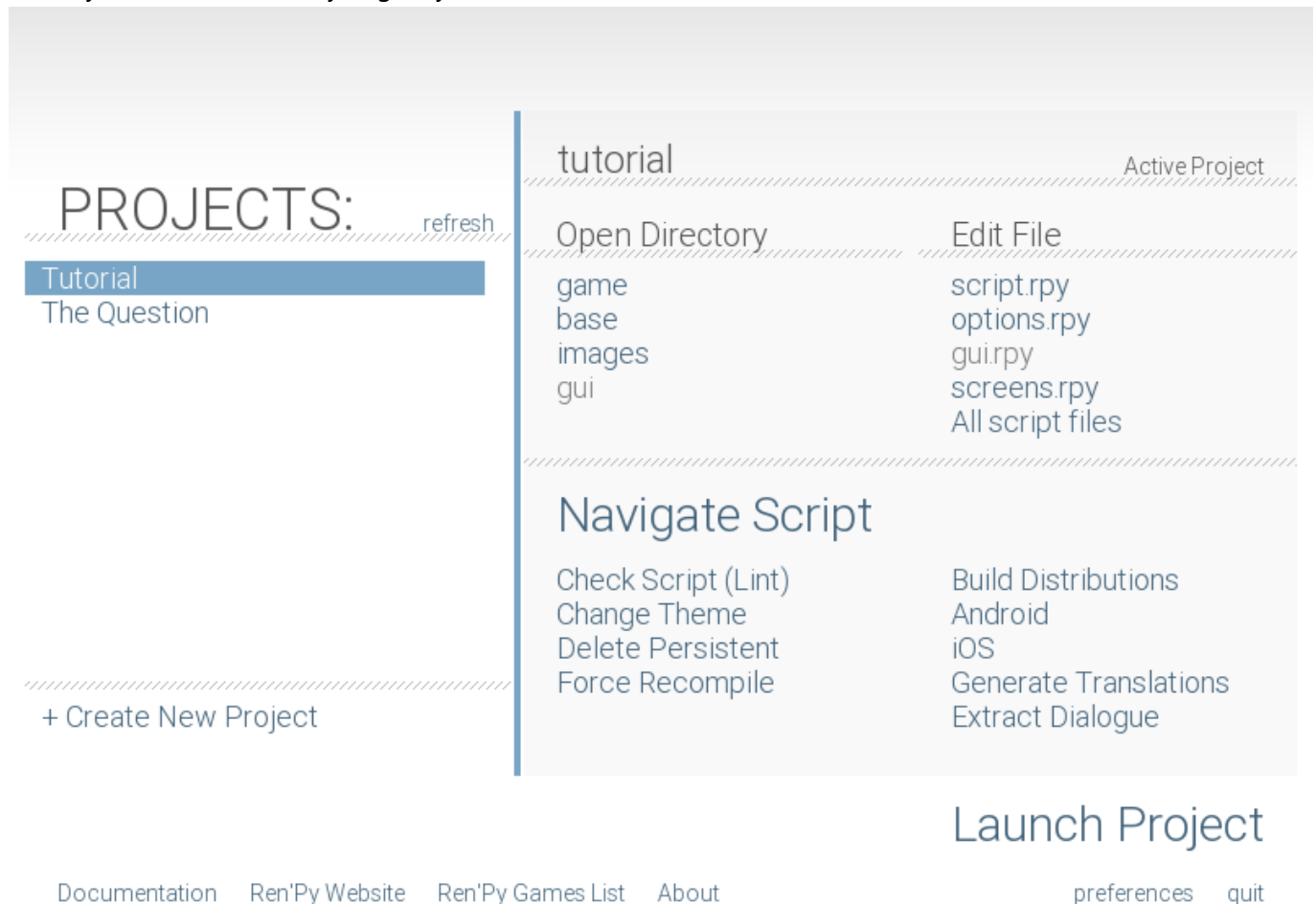
In this workshop you will learn the basics of RenPy and some aspects of Python and create your own Visual Novel.

Download RenPy

- Click [here](#) to download the RenPy game engine
- Create a folder on the Desktop for your project, you can name it whatever you want.
- Unzip the contents you just downloaded into your project folder
- Open the folder you unzipped and run RenPy.exe

The RenPy Engine - Create a new Project

When you launch the RenPy engine you will see this window



On the left you have the list of projects. Under the list you can find the + **Create New Project** option. Press this link to create a new project.

You will now be prompted to name your new project

PROJECT NAME

Please enter the name of your project:

My Question|

Due to package format limitations, non-ASCII file and directory names are not allowed.

Cancel

Continue

Documentation Ren'Py Website Ren'Py Games List About preferences quit

Give it a name and press **Continue**

Then, the engine will prompt you to choose the desired game resolution

CHOICE

What resolution should the project use? Although Ren'Py can scale the window up and down, this is the initial size of the window, the size at which assets should be drawn, and the size at which the assets will be at their sharpest.

The default of 1920x1080 is a reasonable compromise.

1280x720

1920x1080

2560x1440

3840x2160

Custom. The GUI is optimized for a 16:9 aspect ratio.

Cancel

Continue

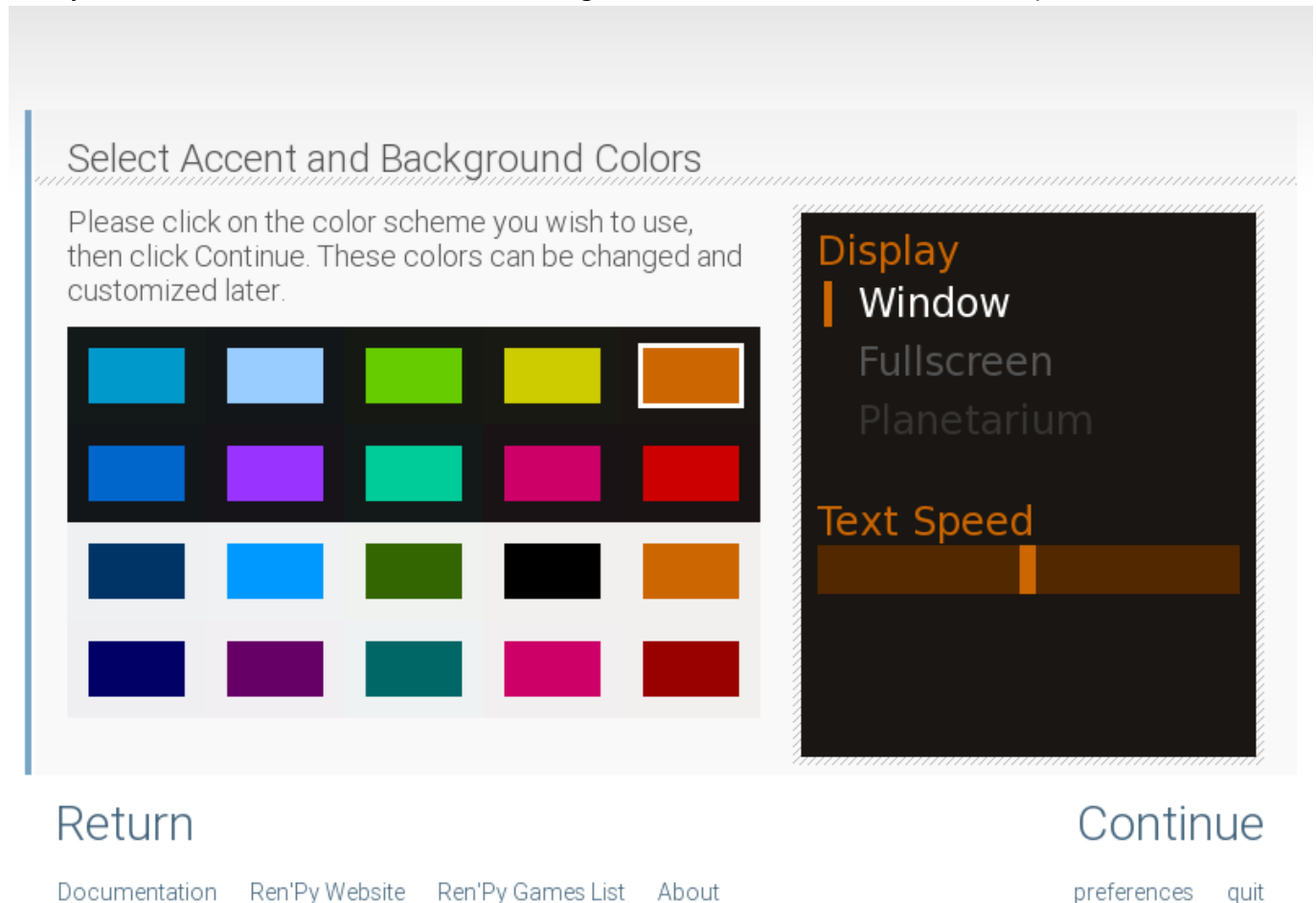
[Documentation](#) [Ren'Py Website](#) [Ren'Py 8.0.0n](#)

[update](#) [preferences](#) [quit](#)

[Ren'Py Sponsor Information](#)

Leave the default selected resolution for now and press **Continue**

Now you have to select the GUI accent and background colors, as well as some other options.



Choose whatever colors you prefer but leave the Window Display option selected for now, then press **Continue**

Once the game engine finishes processing everything, you'll return to the initial launcher window.

The RenPy Engine - How it Works

Your game is composed of files (also called scripts) stored in different folders (or directories).

Looking at the **Open Directory** section of the launcher you will find the following options:

- **base**: This is where all the files are contained. This folder has the same name as your game.
- **game**: contains all the scripts used in your game.
- **images**: stores all the images used in the game.
- **audio**: contains all the audio and music you use in the game.
- **gui**: this folder contains resources from the engine that you may use to customize your game's GUI.

Under the **Edit File** section you can find:

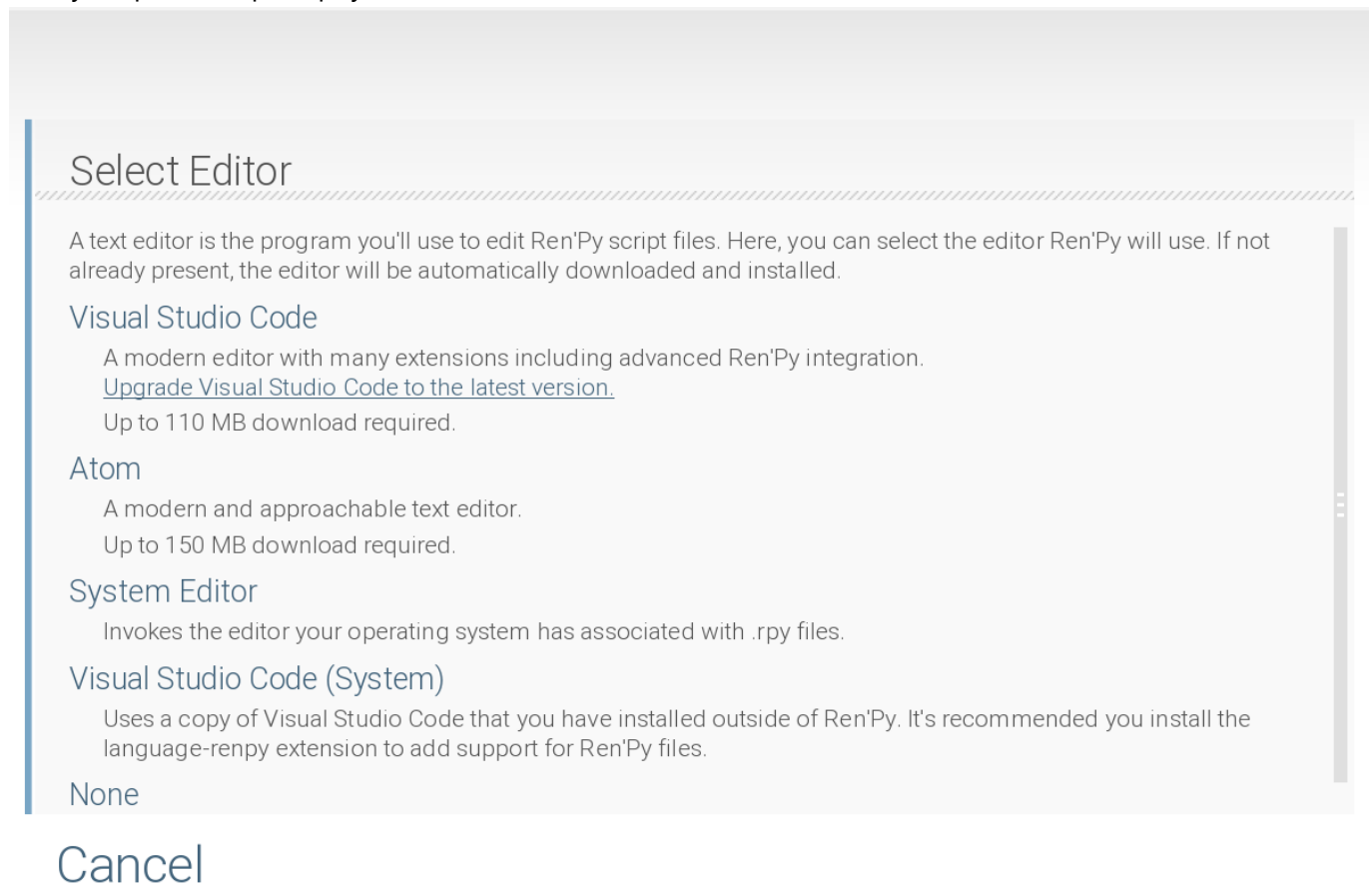
- **script.rpy**: this is the main script of your game where you'll add the code to be implemented.
- **options.rpy**: this script contains your game's configurations. We won't need to alter this for now, but all the possible configurations can be consulted in the documentation [here](#).
- **gui.rpy**: same as the options script, but with variables that change the game's GUI.

- **screens.rpy**: just like the GUI, the screens can also be customized. Once again we won't be changing this file, but you may consult the documentation for more information.

Let's create a Game

Now that you created a new project make sure it's selected under the **Projects** section. If not, simply click over the name.

Now click the **script.rpy** file to open it and we'll start coding our game. Since it's the first time you open RenPy scripts it will prompt you for the desired editor.



Select the **Visual Studio Code (System)** option at the end of the list. This will open the IDE we will be using for the rest of the workshop.

Once you open the script you will see something like this

```

1  # The script of the game goes in this file.
2
3  # Declare characters used by this game. The color argument colorizes the
4  # name of the character.
5
6  define e = Character("Eileen")
7  |
8
9  # The game starts here.
10
11 label start:
12
13     # Show a background. This uses a placeholder by default, but you can
14     # add a file (named either "bg room.png" or "bg room.jpg") to the
15     # images directory to show it.
16
17     scene bg room
18
19     # This shows a character sprite. A placeholder is used, but you can
20     # replace it by adding a file named "eileen happy.png" to the images
21     # directory.
22
23     show eileen happy
24
25     # These display lines of dialogue.
26

```

Adding Sprites to the project

Open the folder you downloaded named **Sprites** and copy the contents to the **images** folder of your project (use the RenPy launcher to open this folder)

This folder will contain all the images you need for the project. To show an image we will use the name without the extension (e.g. **.png**)

Defining the Characters

If you look at the example code you can see how to define a new character in RenPy. In the case of our game we will have 2 characters: Alex and you. To make it easier to tell who is speaking we're also going to add different colors to the text. It should look something like this

```

1  # The script of the game goes in this file.
2
3  # Declare characters used by this game. The color argument colorizes the
4  # name of the character.
5
6  define a = Character("Alex", color=■ "#d2d2d2")
7  define m = Character("Me", color=■ "#a6a6ff")
8

```

Note: Visual Studio Code provides a built-in color picker, you can change the colors to something else.

Start the game loop

Once you define the characters you will need to start the game. That's achieved with the `label start:` statement you find in the script.

Adding content to the game

Now we need to add a background. The script says it's a coffee shop, so we need to change the scene to show the *bg coffee shop.png* image

Then the narrator gives an introduction. Since the narrator is not a character, there is no need to add the character speaking before the line of text. Here's the code:



```
11 label start:
12
13     # Show a background. This uses a placeholder by default, but you can
14     # add a file (named either "bg room.png" or "bg room.jpg") to the
15     # images directory to show it.
16
17     scene bg coffee shop
18
19     "You go to your favorite coffee shop and spot your friend Alex sitting at a small table, nervously sipping on a latte."
20     "You go over to talk to them."
```

The script says Alex is nervous. Let's show the corresponding sprite and add the dialogue:

```
25 show alex sad
26
27 # These display lines of dialogue.
28
29 a "Hello..."
30 m "Hey Alex! You look like you've seen a ghost. What's going on?"
31 a "I've got a major career decision to make, and it's eating me alive."
32 a "I've been offered a steady, well-paying job with great benefits, but it's in a city I've never been to.
33 I'd have to move away from everything I know."
34 a "On the other hand, I've also received an offer to work for a startup in a field I'm passionate about, but it pays less,
35 and there's no guarantee of success."
36
37 # This ends the game.
38
39 return
40
```

Testing the game

Save the changes you made to the script and let's test the game. For that you need to close the editor and click **Launch Project** on the RenPy window.

Once you launch the project the game window will open with the main menu. Press **Start** and click on the game window to go through the interactions.  

Adding a choice menu

To make this interactive we need to help the character make a decision. For this we will add a menu with two options.

To add a menu we use the `menu:` statement. \

Then we need to add the two dialogue options that the player can choose. These will be two regular narrator lines of dialogue.

Since we want our decision to have different outcomes on the game we also need to add a `jump` statement so the engine knows what part of the dialogue to show the player.


Here's how it looks:

option

Now let's add the scene for the first option. In order for the program to know where to jump to, we need to add a `label` . We can also change the character sprite and background to show the game progression:

security

Once the player ends the interaction the game will end.

Now let's add the second option. It should look something like this: passion

Now save all the changes and run the game