**Screens**

(Abstract Class)

This class should be used as the base class for all the Screens to be used on the program. Screens have pre-defined methods and functionalities to control the layers and order of elements to be displayed in the screen that will make it easier for the developers to manipulate and implement how they want the Screen to be shown.

It also automatically handles some necessary yet repetitive function calls *(e.g., for loop for pygame.event.get(), display.update(), clock.tick())* so developers won’t have to worry anymore about any errors that may occur should they forgot to write or doubled those statements.

After implementing all the abstract methods, developers can easily add some additional methods to the subclasses and insert it to its corresponding main loop methods *(before\_looping(), while\_looping(), after\_looping()).* After completing the subclass, developers will just need to call the **display()** method to display the Screen and start its loop – no more additional lines of code, error checking, event handling, mouse detection, or nested if statements – just simply **display()** it.

**Attributes**:

self.screen  *(pygame.Surface) --* program’s main Surface.

self.bg\_color *(R, G, B) --* indexed color, pygame.Color.

self.bg\_music  *(BinaryIO) --* path of the audio file.

self.music\_playing *(bool)* -- Boolean variable to check if the music is playing or not.

self.running *(bool)* -- Boolean variable to check if the loop is running or not.

**Properties**:

self.size *(tuple[width, height])* – returns a tuple containing the width and height of the

Screen (this should return the current display’s width and

height).

self.width *(int)* – returns the width of the Screen

self.height *(int)* – returns the height of the Screen

**Abstract Methods**:

self.before\_looping() – blocks of code to be executed before starting the loop.

self.while\_looping() – this acts as the main function where developers will place

everything they want to display in layers to the screen, and although the method name suggests that this is the main loop, this will just serve as a block of code that will be executed every frame, unlike before\_looping that will be executed only once after **display()** is called.

self.after\_looping() – idk if this will even execute

**Instance Methods:**

self.fill() – fills the Screen with the passed *bg\_color*

play\_music() – stops any music that is currently playing if there’s any, and plays the

passed *bg\_music*

display\_cursor() – basically displays the cursor to the Screen

get\_events() – handles the event loop

display () – this is where the **while loop** is located. This handles almost every

necessary methods to be called for the Screen to function properly.

Text, timeline

Description automatically generated