# INITIAL SETUP

Before you can start working with the VNGhost frameword, you must first install a text editor that will allow you to run a live server. This will allow you to preview your work in your browser.

Visual Studio Code is a free, robust, and open-source text editor that will
https://code.visualstudio.com/

Once Visual Studio Code is installed, open the program and navigate to the Extensions: Marketplace by clicking on the Extensions symbol on the side-bar to the left of the window (symbol pictured below).

Next you will type 'Live server' in the *Search Extensions in Marketplace* text window at the top-right corner of the screen. Select **Live Server** by Ritwick Dey and install.
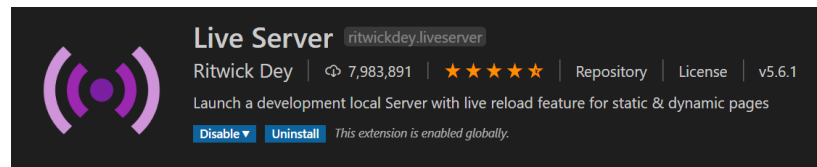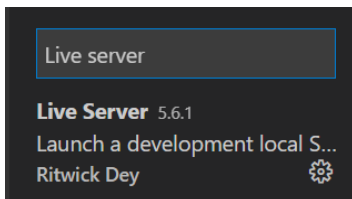
Once installed, you will now be able to initiate a Live Server to view your work within your browser. To access the Live Server within an html document, right click within the text window of your html document and click **Open with Live Server**.
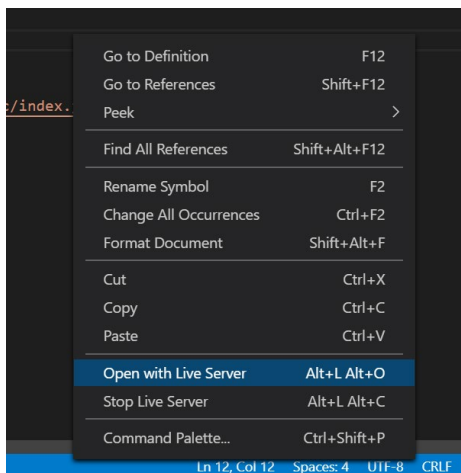
In order to update your csv files, you will need to download Python to run the **updatedata** python script.
https://www.python.org/downloads/

Make sure to check off 'add Python to PATH' if you see that come up in the installation process.

WORKFLOW

Now that you have installed the necessary programs to work with VNGhost, you can start working on your visual novel!

The overall workflow of this framework was created to be as simple as possible. The vast majority of your work will consist of the following steps..

1. Make edits to **timeline.csv** (found within the **vn_data** folder)
2. Save your progress by double-clicking the **updatedata** file (found within the **vn_data** folder)
3. Add image and sound files to the img and sound folders as needed.

That's it! Once you are familiar with how the framework will parse the timeline.csv file, you can hit the ground running.

_____

There are several folders within the VNGhost folder (pictured below) but you will not need to be familiar with all of them.

css	10/25/2020 3:13 PM	File folder	
img	10/27/2020 1:13 PM	File folder	
sound	10/24/2020 6:49 PM	File folder	
src	10/25/2020 3:39 PM	File folder	
vn_data	11/1/2020 10:07 AM	File folder	
index	10/25/2020 3:47 PM	Chrome HTML Do...	1 KB

img folder

> The img folder is where all of your images will live. For the sake of simplicity, there are no subfolders. Any time you are referencing an image within the csv files, you will be referencing images in this folder.

sound folder

> This is where your sounds will live. Any time you are referencing a sound within the csv files, you will be referencing sounds in this folder. If you are not implementing any sounds, you can ignore this folder.

vn_data folder

> This folder houses all of your information for your game. General setting for the game are edited within **config.csv** and all edits to the visual novel timeline are edited within **timeline.csv**. The file **updatedata.py** takes the information from **config.csv** and **timeline.csv** and updates the **data.js** file to be used by the framework. You will not need to make any edits to **data.js** or **updatedata.py**.

config	11/1/2020 9:58 AM	Microsoft Excel Co...	1 KB
data	11/1/2020 9:58 AM	JavaScript File	2 KB
timeline	11/1/2020 10:07 AM	Microsoft Excel Co...	2 KB
updatedata	10/25/2020 3:35 PM	Python File	1 KB

EXAMPLE OF HOW IT ALL WORKS

START SCREEN

The first slide that will show before the first node will be the start screen. It is composed of two images, the start_screen, which is an image that takes up the entire game window and the start_button, which is an image that appears at the center of the game window and 20% from the bottom. When clicked by the player, the start_button will initiate the first node of the game.

config.csv
start_screen (image)

config.csv
start_button (image)
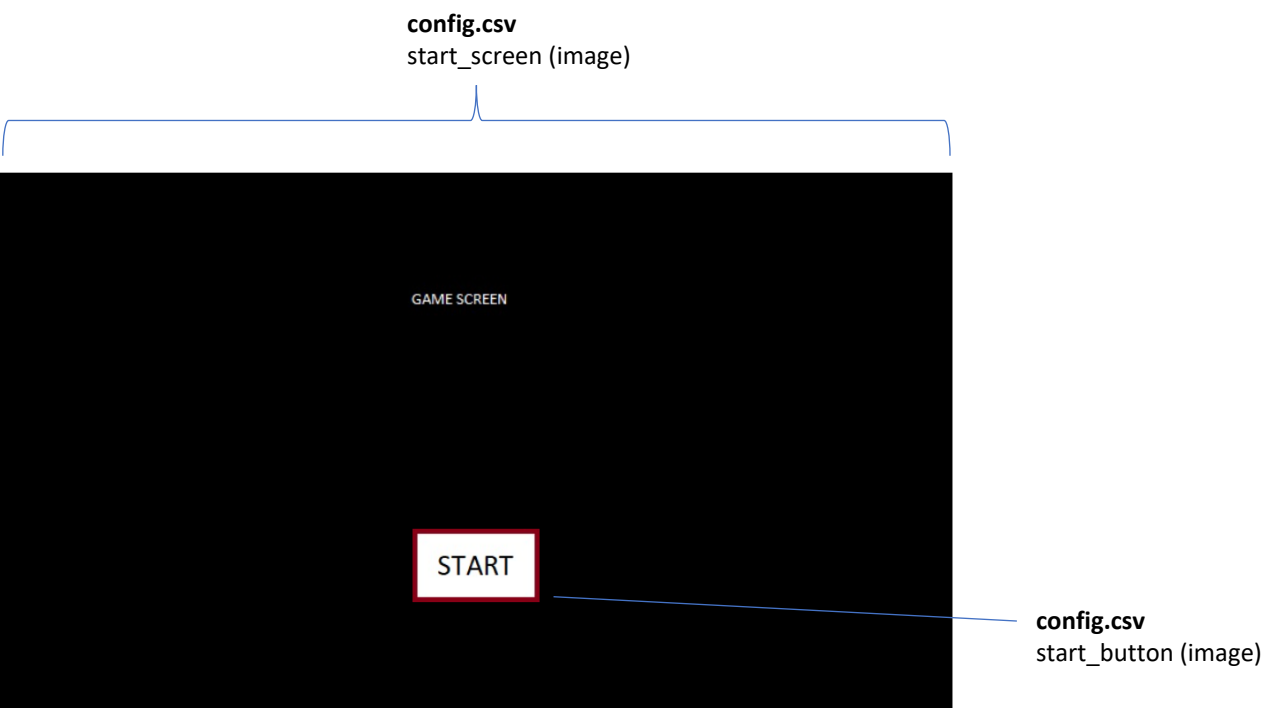
The start_screen is 747 px x 427 px and the start_button is 128 px x 73 px. The start_screen image should stay consistent with the size given here but the start_button can be a custom size of your choosing.

Both start_screen and start_button are found within the config.csv file.

M	N
start_screen	start_button
start_screen.jpg	start_button.jpg

WHAT IS A NODE?

A node is defined as a point at which lines or pathways intersect or branch; a central or connecting point. Within this framework, I will be defining each screen or slide in the visual novel as a node. So for your purposes, a node is a point in your story.
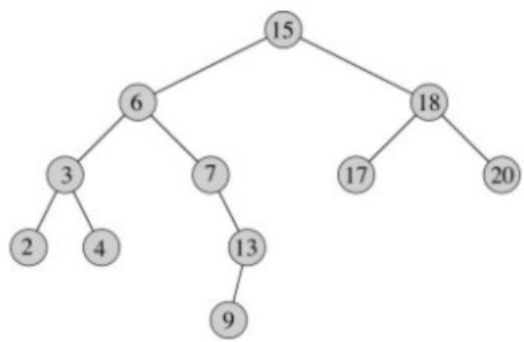
Each node in your Visual Novel will be represented within the timeline.csv as a row. Below you will see an example of a 'text' node...

...which is represented within the timeline as the row below. The node_id for this row is 1, which means that it will be the first node displayed after the Start Screen. All the information for each node is found in their row.

node_id	display_m	sound	sound_vol	fade_out_	loop_soun	loop_soun	setting_ba	setting_ba	setting_ba	setting_mi	setting_mi	setting_mi	setting_fr	setting_fr	setting_fr	text_color	text_font	name	name_pos	line_1_tex	line_2_tex	lin
1	text						hht_rough_forest01.19.jpg											Neil	left	Hey! Just v	base with	wa

Each node will have a designated output that will tell the framework which node to display next. If your Visual Novel is 100% linear with no branching paths, node 1 will be first followed by node 2 followed by node 3, etc. The output of node 1 would be node 2. Node 2's output would be node 3, etc.

TYPES OF NODE

There are four distinct types of nodes that you can code within the timeline.

1. TEXT

The Text node displays three lines of text within the dialogue box and then displays the next button. The next button will only appear once all the text has been added to the dialogue box. The next button, when pressed, will trigger the specified Output node. If a character name is specified, that will show that as well.

setting_back (image)

line_1_text (text)
line_2_text (text)
line_3_text (text)

img_left (image)

next_button.png
next-botton-hover.png
IF YOU WANT TO CHANGE THESE, JUST REPLACE THESE FILES IN THE IMG FOLDER.

name (image)
name_position -
(specify: left, center, right)

config.csv
text_box_img (image)

The diagram above lays out where each piece of the display is coming from within the timeline.csv row.

To set a node to TEXT, make sure that text is written as the value within the display_mode column (shown below as it appears within the timeline.csv file).

display_mode

text

To set a node to TEXT, make sure that text is written as the value within the display_mode column (shown below as it appears within the timeline.csv file).

To specify which node will appear when the user clicks on the next button, you will need to give your node an output value. This value will correspond to another row within you timeline.csv file.

output

2

In the example above, the output value is set to 2. This means that this current node is pointing to the node with the node_id of 2 as the next displayed node in the Visual Novel sequence.

2. SCENE

The Text node displays three lines of text within the dialogue box and then displays the next button. The next button will only appear once all the text has been added to the dialogue box. The next button, when pressed, will trigger the specified Output node. If a character name is specified, that will show that as well.

setting_back (image)

img_center (image)
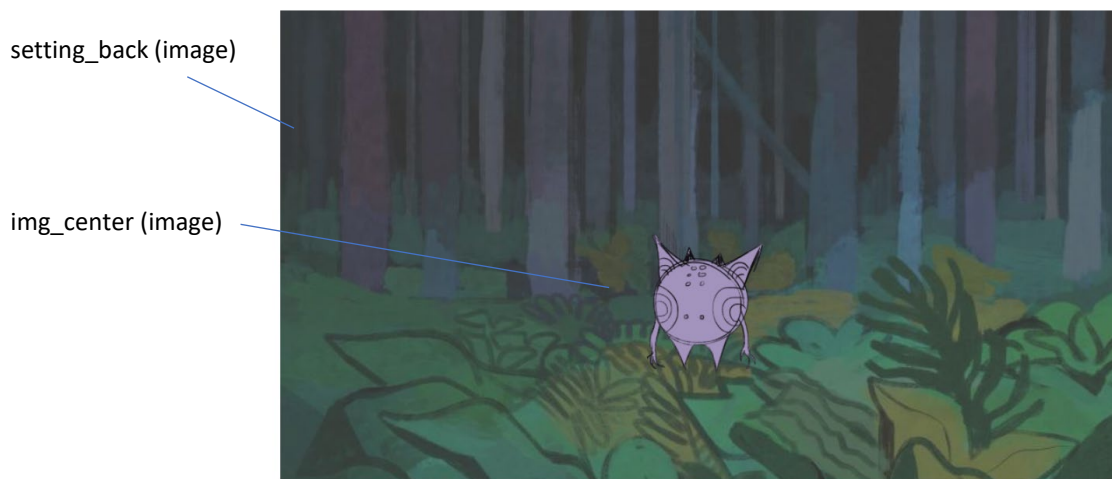
The diagram above lays out where each piece of the display is coming from within the timeline.csv row.

3. OPTION

The Text node displays three lines of text within the dialogue box and then displays the next button. The next button will only appear once all the text has been added to the dialogue box. The next button, when pressed, will trigger the specified Output node. If a character name is specified, that will show that as well.

setting_back (image)

img_left (image)

e_1_text (text)
e_2_text (text)
e_3_text (text)

next_button.png
next-botton-hover.png
IF YOU WANT TO CHANGE THESE, JUST REPLACE THESE FILES IN THE IMG FOLDER.

name (image)
name_position -
(specify: left, center, right)

text_box_img (image)

The diagram above lays out where each piece of the display is coming from within the timeline.csv row.

4. END

Setting a node's display_mode as 'end' will mark it as the game's end point. This node is similar to a scene node as it displays images without the text window, except this type of node has no time limit or output. This is where you would want to put an image signifying to the user that the game is over. Depending on how you structure your visual novel, you may want several different end nodes (like a choose your own adventure story), however for each run, the user can only ever be served one.

POSITION OF DISPLAYED IMAGES

Each image has a set location on the z-axis. The image setting_back will always be the furthest back and text_box_img will always be in the very front. Be mindful of how these images overlap. Multiple setting layers are available for option to implement layered background (png images with creative transparencies). These background layers can also be set as gifs to add looped movement or given an opacity value (between 0 and 1. Opacity with a .5 value is 50% opacity).

ANGLED VIEW

setting_back

setting_mid

setting_front

img_center

img_left

Img_right

text_box_img

VIEW FROM THE SIDE

Back >   < Front

setting_back   img_left   text_box_img
setting_mid   img_center
setting_front   Img_right
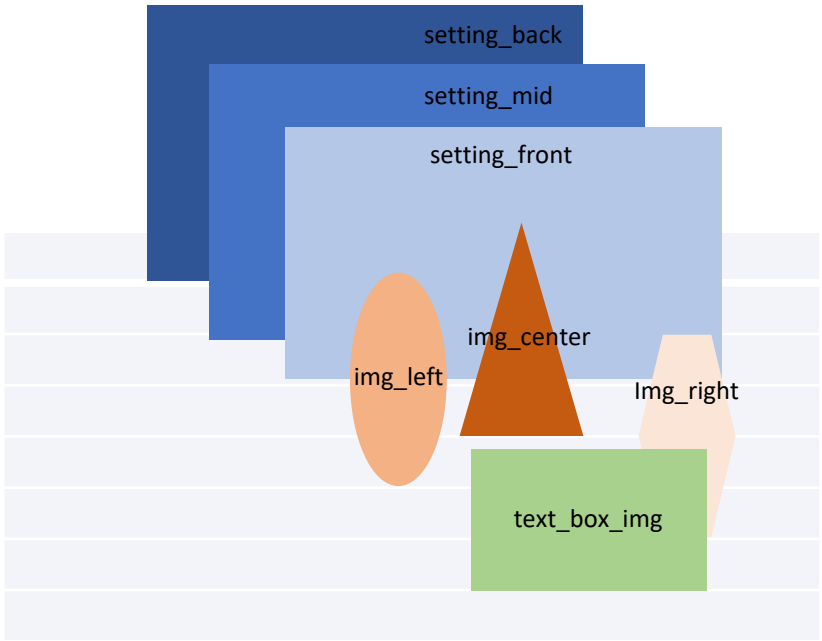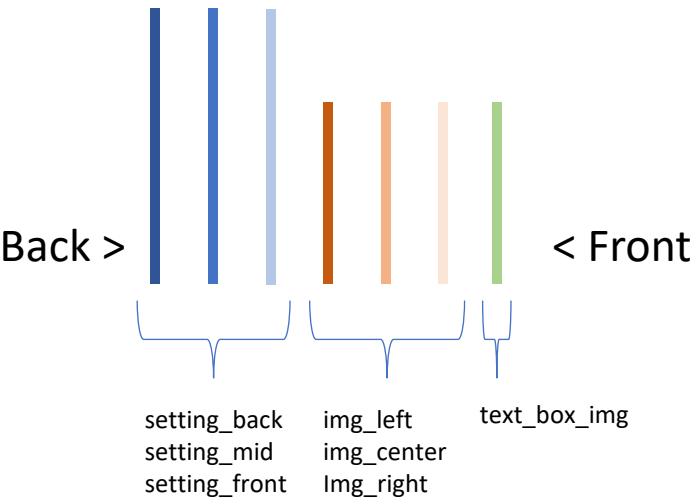
GLOSSARY

node_id: Reference number for the node. Not only does this define the node found in the given row within timeline.csv but is used as a reference value for any of the outputs.
Input type: Number.　　　1

display_mode: The type of display that you desire to use for the given node. This cell sets the template for the rest of the node. For instance, setting display_mode to text will format the node as Text so it will be using the name, name_position, line_1_text, etc cells.
Input type: text, scene, option, end　scene

sound: The sound file that will trigger at the beginning of its node. This draws from the Sound folder. Mp3 file type is suggested as it is compressed.
Input type: The name of the file in the *sound* folder with the file type suffix.　guitar.mp3

sound_volume: The set volume for the sound triggered in this node. It only effects the sound output of the sound specified in its node. This will only be used if you want to set the sound volume below its max volume.
Input type: Decimal number between 0 and 1. Using .5 will set the volume to 50% max volume, .25 will set the volume to 25% max volume.　　0.5

loop_sound: Functions the same as sound except the *sound* will continue to loop. It will not stop playing until specified in *fade_out_active_loop*.

loop_sound_volume: Functions the same as *sound_volume* but is setting the volume for *loop_sound*.

fade_out_active_loop: Will be expanded in future build. Currently this will fade out the last looped sound that was set. If you set a sound loop in node 2 and want to have that sound fade out when the player gets to node 26, you will set the loop in node 2 and then use this cell to fade out the sound in node 26.
Input type: Currently populating this cell with any value will trigger a fadeout of the looped sound. This will be expanded in a future build.　y

setting_back: The furthest back background image. This image should be sized to 747 px wide x 427 px high. If not, the image will be stretched to fill the space.
Input type: The name of the file in the *img* folder with the file type suffix.　hht_rough_forest01.19.jpg

setting_back_animation: The name of the type of animation the image will enact when the node first loads. There are set by the css documentation. Let me know if you have any specific requests.
Input: type: Currently the only animation values are 'slideLeft' and 'slideRight'. 'slideLeft' slides the image from off the screen, sliding from right to left. 'slideRight' does the opposite. More may be added in future builds.　slideLeft

setting_back_opacity: This will set the opacity for the *setting_back* image.
Input type: Decimal number between 0 and 1. Using .5 will set the opacity to 50%, .25 will set the opacity to 25%.　0.5

setting_mid: The next layer background up. If this is set to a solid image with no transparency, it will hide the layer underneath it. This layer can, however, be opaque or have some level of transparency to complement the layer underneath. Additionally, animations could be applied to give an interesting multi-plane effect.
Input type: The name of the file in the *img* folder with the file type suffix.　background2.png

GLOSSARY (cont.)

node_id: Reference number for the node. Not only does this define the node found in the given row within timeline.csv but is used as a reference value for any of the outputs.
Input type: Number.　　　1

display_mode: The type of display that you desire to use for the given node. This cell sets the template for the rest of the node. For instance, setting display_mode to text will format the node as Text so it will be using the name, name_position, line_1_text, etc cells.
Input type: text, scene, option, end　scene

sound: The sound file that will trigger at the beginning of its node. This draws from the Sound folder. Mp3 file type is suggested as it is compressed.
Input type: The name of the file in the *sound* folder with the file type suffix.　guitar.mp3

sound_volume: The set volume for the sound triggered in this node. It only effects the sound output of the sound specified in its node. This will only be used if you want to set the sound volume below its max volume.
Input type: Decimal number between 0 and 1. Using .5 will set the volume to 50% max volume, .25 will set the volume to 25% max volume.　0.5

loop_sound: Functions the same as sound except the *sound* will continue to loop. It will not stop playing until specified in *fade_out_active_loop*.

loop_sound_volume: Functions the same as *sound_volume* but is setting the volume for *loop_sound*.

fade_out_active_loop: Will be expanded in future build. Currently this will fade out the last looped sound that was set. If you set a sound loop in node 2 and want to have that sound fade out when the player gets to node 26, you will set the loop in node 2 and then use this cell to fade out the sound in node 26.
Input type: Currently populating this cell with any value will trigger a fadeout of the looped sound. This will be expanded in a future build.　y

setting_back: The furthest back background image. This image should be sized to 747 px wide x 427 px high. If not, the image will be stretched to fill the space.
Input type: The name of the file in the *img* folder with the file type suffix.　hht_rough_forest01.19.jpg

setting_back_animation: The name of the type of animation the image will enact when the node first loads. There are set by the css documentation. Let me know if you have any specific requests.
Input: type: Currently the only animation values are 'slideLeft' and 'slideRight'. 'slideLeft' slides the image from off the screen, sliding from right to left. 'slideRight' does the opposite. More may be added in future builds.　slideLeft

setting_back_opacity: This will set the opacity for the *setting_back* image.
Input type: Decimal number between 0 and 1. Using .5 will set the opacity to 50%, .25 will set the opacity to 25%.　0.5

setting_mid: The next layer background up. If this is set to a solid image with no transparency, it will hide the layer underneath it. This layer can, however, be opaque or have some level of transparency to complement the layer underneath. Additionally, animations could be applied to give an interesting multi-plane effect.
Input type: The name of the file in the *img* folder with the file type suffix.　background2.png

GLOSSARY (cont.)

node_id: Reference number for the node. Not only does this define the node found in the given row within timeline.csv but is used as a reference value for any of the outputs.
Input type: Number. 1

setting_mid_animation: Functions the same as *setting_back_animation*.

setting_mid_opacity: Functions the same as *setting_back_opacity*.

setting_front: The top layer of the background. If this is set to a solid image with no transparency, it will hide the layer underneath it. This layer can, however, be opaque or have some level of transparency to complement the layer underneath. Additionally, animations could be applied to give an interesting multi-plane effect.
Input type: The name of the file in the *img* folder with the file type suffix.

setting_front_animation: Functions the same as *setting_back_animation*.

setting_front_opacity: Functions the same as *setting_back_opacity*.

text_color: Currently not included. Will be added in a later build to change the text color during a specified node.

setting_back_opacity: This will set the opacity for the *setting_back* image.
Input type: Decimal number between 0 and 1. Using .5 will set the opacity to 50%, .25 will set the opacity to 25%.

name: The character name displayed within a text node. Appears at the top of the text window if specified. Not required to display text for the text node.
Input type: Plain text. Review how the program inputs the name. Depending on font size and the length of the name, your input may be too long and runoff to a second line. Neil

name_position: The character name displayed within a text node. Appears at the top of the text window if specified. Not required to display text for the text node.
Input type: left, right, or center depending on where you would like the name to display for the specified node. left

line_1_text: Text for the top line of text for a text node. This is displayed first when a text node initiates, followed by *line_2_text*. If this contains too many characters, it may spill off the text window. The amount of characters you can comfortably display in each line of text will change based on font and font size.
Input type: Plain text. Hey! Just wanted to say hi

line_2_text: Functions the same as *line_1_text*. This will display after *line_1_text*.

line_3_text: Functions the same as *line_1_text*. This will display after *line_2_text*.

option_description: Text displayed above the options for an option node. This field is not required but does allow. A good way to supply context for an option choice.
Input type: Plain text. What would you like to do?

GLOSSARY (cont.)

option_1_text: Text for the first option for an option node. If this contains too many characters, it may spill off the text window. The amount of characters you can comfortably display in each line of text will change based on font and font size.
<u>Input type:</u> Plain text. `Go to work.`

option_2_text: Functions the same as *option_1_text*. This will display after *option_1_text*.

option_3_text: Functions the same as *option_1_text*. This will display after *option_2_text*.

option_1_output: Specifies the next node to be displayed if *option_1_text* is selected by the user.
<u>Input type:</u> Number corresponding to an existing node_id. `4`

option_2_output: Functions the same as *option_1_output*.

option_3_output: Functions the same as *option_1_output*.

img_left: An image displayed on the left-hand side of the game window meant to be used for characters. The image is anchored to the bottom-left of the game window.
<u>Input type:</u> The name of the file in the *img* folder with the file type suffix.

img_center: Functions the same as *img_left* except this image is anchored to the bottom-center of the game window.

img_right: Functions the same as *img_left* except this image is anchored to the bottom-right of the game window.