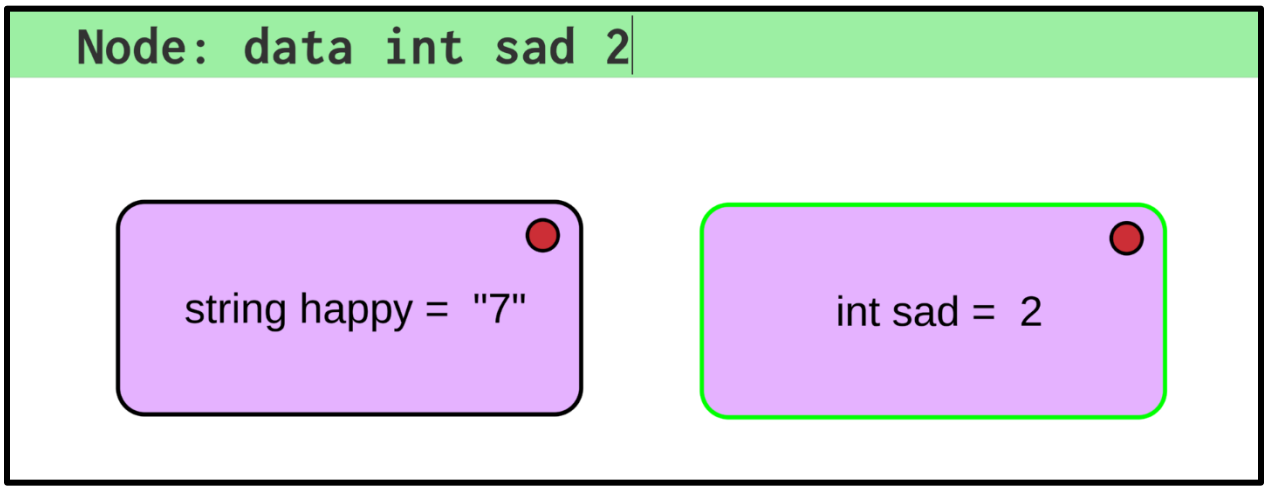
Project Deliverable 2 - Kostiantyn Makrasnov - Visual Scripting Language

My current progress on the project is right on course for completion by the finals date. I have begun a Unity 2D project and developed code to be able to create and move around generic nodes of code. I added a system in which the developer can edit nodes in the form of a dynamic input field that adjusts the hint text according to the selected node. Each generic node now contains an identity class that is populated by the user's interactions in the input field or connecting the nodes on the screen. Each time a new node is selected, its current code is displayed in the editing fields.



Currently only the data nodes of string, int and bool types are available for creation and customization. The other major types of nodes that are still in development include selection blocks as well as *for* and *while* loops. Finally, function nodes will allow people to either combine a group of existing nodes or import python or C++ code to execute for its output.

Some struggles that I ran into primarily relate to the way Unity separates its game world and UI objects. In my case, I would like to have the ability to zoom out and see a larger amount of created nodes or even the entire program. To do this, the nodes themselves have to be game objects that are not part of the UI, which stays static on camera position change. However, this then prevents me from having access to the many useful auto-scaling and layout features that Unity implements for its UI objects. This limitation leaves me to create my own layout code to display text on the nodes correctly and position every element in a way that is both readable and non-overlapping.

Besides implementing the creation process for other types of nodes, I will need to complete an execution object that would take in the node tree and compile all connections for a proper flow of information. Afterward, the same object would either output which node is causing errors or execute the tree and print out the final output of the sink node. I believe this will be the most challenging section of the project and therefore hope to allocate as much time as possible to this portion. As I test the different parts of the language, I will also gradually build up the language documentation through comments and backed up examples.