**Class Definition Document**

**Class Name: DataLoader**

**Class Type**: Base Class

**Description**: The DataLoader class is a base class that provides functionality for loading and managing data from a file. It includes methods for file picking, memory allocation, and storing data in a dictionary.

**Purpose of this Class**: The purpose of the DataLoader class is to abstract the process of loading data from a file and managing memory allocation. It provides a consistent interface for subclasses or client code to load and access data.

**Functions**:

**File\_Picker**

Purpose: Opens a file picker dialog to allow the user to select a file.

Inputs:

None

Return:

file\_path: The path to the file.

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**Allocate\_Memory**

Purpose: Allocates memory and initializes a dictionary with the specified size. It reads lines from a file and stores them in the dictionary.

Inputs:

file: The BasicML text file.

memory\_size: the number of lines in the BasicML file.

Return:

MEM: The lines from the file that have been converted to a dictionary.

Pre/Post Conditions: The file should be opened and accessible.

-------------------------------------------------------------------------------------------------------------------------------------------

**get\_data**

Purpose: Return the data to the caller.

Inputs:

None

Return:

Private\_data\_dictionary: Dictionary of the data from the BasicML file.

Pre/Post Conditions: The file should be opened and accessible.

-------------------------------------------------------------------------------------------------------------------------------------------

**get\_file\_path**

Purpose: Return the file path to the caller.

Inputs:

None

Return:

private\_data\_file\_path: The path to the file.

Pre/Post Conditions: The file should be opened and accessible.

-------------------------------------------------------------------------------------------------------------------------------------------

**Class Name: testingWindow**

**Class Type**: base class

**Description**: The testingWindow class serves as a base class for creating window objects used in software testing scenarios.

**Purpose of this Class**: The class provides a framework for creating and manipulating testing windows in software testing processes.

**Functions**:

**appendOutput**

Purpose: This function appends a given value to the testing window.

Inputs:

value: The value to be appended to the testing window.

Return:

None

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**Class Name: DataModel**

**Class Type**: base class

**Description**: The DataModel class represents a data model used in a software system. It encapsulates the data, accumulator, program counter, and memory operations.

**Purpose of this Class**: The class provides functionality for accessing and manipulating the data model, including memory operations, program counter control, and accumulator value management.

**Functions**:

**\_\_init\_\_**

Purpose: Initializes a new instance of the DataModel class.

Inputs:

data: A dictionary representing the memory contents of the data model.

accumulator (optional): A string representing the initial value of the accumulator. The default value is '+0000'.

pc\_location (optional): An integer representing the initial value of the program counter. The default value is 0.

Return:

None

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**get\_mem\_value**

Purpose: Retrieves the value from the memory at the specified location.

Inputs:

memory\_location: An integer representing the memory location to retrieve the value from.

Return:

The integer value stored in the memory location.

Pre/Post Conditions: Raises a TypeError if memory\_location is not an integer. Raises a ValueError if the value in the memory location is not convertible to an integer. Raises a MemoryError if the memory\_location is out of bounds.

-------------------------------------------------------------------------------------------------------------------------------------------

**set\_mem\_value**

Purpose: Sets the value in the memory at the specified location.

Inputs:

value: An integer representing the value to be stored in the memory.

memory\_location: An integer representing the memory location where the value should be stored.

Return:

True if the value was successfully set in the memory, False otherwise.

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**get\_pc**

Purpose: Retrieves the current value of the program counter.

Inputs:

None

Return:

An integer representing the current value of the program counter.

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**set\_pc**

Purpose: Sets the value of the program counter.

Inputs:

value: An integer representing the value to be set as the program counter.

Return:

True if the program counter value was successfully set, False otherwise.

Pre/Post Conditions: The value must be an integer and within the bounds of the memory.

-------------------------------------------------------------------------------------------------------------------------------------------

**increment\_pc**

Purpose: Increments the program counter by 1.

Inputs:

None

Return:

True if the program counter was successfully incremented, False otherwise.

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**get\_acc**

Purpose: Retrieves the current value of the accumulator.

Inputs:

None

Return:

An integer representing the current value of the accumulator.

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**set\_acc**

Purpose: Sets the value of the accumulator.

Inputs:

value: An integer representing the value to be set as the accumulator.

Return:

True if the accumulator value was successfully set, False otherwise.

Pre/Post Conditions: The value must be a valid integer within the specified range.

-------------------------------------------------------------------------------------------------------------------------------------------

**validate\_value**

Purpose: Validates whether a value is within the valid range for the data model.

Inputs:

value: The value to be validated.

Return:

True if the value is within the valid range, False otherwise.

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**Class Name: MainWindow**

**Class Type**: Child Class of QtWidgets.QMainWindow, Ui\_MainWindow

**Description**: The MainWindow class represents the main window of a graphical user interface (GUI) application. It inherits from the QtWidgets.QMainWindow and Ui\_MainWindow classes.

**Purpose of this Class**: The class provides functionality for the main window of the GUI application, including handling user interactions and displaying output.

**Functions**:

**\_\_init\_\_**

Purpose: Initializes a new instance of the MainWindow class.

Inputs:

\*args: Variable length argument list.

obj: An optional object.

\*\*kwargs: Keyword arguments.

Return:

None

Pre/Post Conditions: Calls the init method of the QtWidgets.QMainWindow and Ui\_MainWindow classes. Sets up the user interface, connects button signals to respective slots, and initializes the Enter button state.

-------------------------------------------------------------------------------------------------------------------------------------------

**chooseFileButtonClicked**

Purpose: Handles the button click event when the ChooseFile button is clicked.

Inputs:

None

Return:

None

Pre/Post Conditions: Clears the OutputText field, creates a DataLoader instance, initializes the DataModel instance (\_mem) with data from the DataLoader, and sets the FilePath text to the file path obtained from the DataLoader.

-------------------------------------------------------------------------------------------------------------------------------------------

**clearFilePath**

Purpose: Handles the button click event when the Clear button is clicked.

Inputs:

None

Return:

None

Pre/Post Conditions: Clears the OutputText and FilePath fields.

-------------------------------------------------------------------------------------------------------------------------------------------

**submitFilePath**

Purpose: Handles the button click event when the Start button is clicked.

Inputs:

None

Return:

None

Pre/Post Conditions: Clears the OutputText field, retrieves the file path from the FilePath text field, and if a valid file path exists, calls the main function passing the DataModel instance (\_mem) and the MainWindow instance (window). Displays the output and appends a completion message if a file path is provided, otherwise displays an error message.

-------------------------------------------------------------------------------------------------------------------------------------------

**enterClicked**

Purpose: Handles the button click event when the Enter button is clicked.

Inputs:

None

Return:

None

Pre/Post Conditions: Sets the Enter button state to checked.

-------------------------------------------------------------------------------------------------------------------------------------------

**displayOutput**

Purpose: Displays the provided value in the OutputText field.

Inputs:

value: The value to be displayed.

Return:

None

Pre/Post Conditions: Clears the OutputText field and sets the text to the provided value.

-------------------------------------------------------------------------------------------------------------------------------------------

**appendOutput**

Purpose: Appends the provided value to the OutputText field.

Inputs:

value: The value to be appended.

Return:

None

Pre/Post Conditions: Appends the provided value to the OutputText field.

-------------------------------------------------------------------------------------------------------------------------------------------

**getInput**

Purpose: Enables the Enter button and waits for it to be checked.

Inputs:

None

Return:

None

Pre/Post Conditions: Enables the Enter button and enters a loop until the Enter button is checked.

-------------------------------------------------------------------------------------------------------------------------------------------

**colorBox**

Purpose: Enables the user to enter color values to change the color of the interface.

Inputs:

None

Return:

None

Pre/Post Conditions: This creates a popup window where the user can enter hexadecimal color values and validates the value as a valid entry. If the entry is invalid, it displays an error message. Else, it changes the colors of certain elements of the interface.

-------------------------------------------------------------------------------------------------------------------------------------------

**Class Name: Ui\_MainWindow**

**Class Type**: interface

**Description**: This class represents the user interface for the main window of the UVSim application.

**Purpose of this Class**: The purpose of this class is to define the graphical user interface elements and functionality for the main window of the UVSim application.

**Functions**:

**setupUi**

Purpose: This function sets up the UI elements and properties of the main window. It initializes the main window, sets its size and palette, and creates various labels, buttons, and input fields.

Inputs:

MainWindow: The function takes the MainWindow object as a parameter.

Return:

None

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**retranslateUi**

Purpose: This function sets the text and labels for the UI elements in the main window. It translates and sets the text for various labels, buttons, and menu options.

Inputs:

MainWindow: The function takes the MainWindow object as a parameter.

Return:

None

Pre/Post Conditions: None

-------------------------------------------------------------------------------------------------------------------------------------------

**Class Name: TableWidget**

**Class Type**: Child Class of QTableWidget

**Description**: This class represents the user interface for the file edit window of the UVSim application.

**Purpose of this Class**: The purpose of this class is to define the graphical user interface elements and functionality for the file edit window of the UVSim application.

**Functions**:

**keyPressEvent**

Purpose:

Inputs:

event: the type of event that we are looking for

Return:

None

Pre/Post Conditions: The table widget object must be created

-------------------------------------------------------------------------------------------------------------------------------------------

**Class Name: FileEditBox**

**Class Type**: Child Class of QDialog

**Description**: This class represents the user interface for the file edit window of the UVSim application.

**Purpose of this Class**: The purpose of this class is to define the graphical user interface elements and functionality for the file edit window of the UVSim application.

**Functions**:

\_\_**init\_\_**

Purpose: Initializes a new instance of the FileEditBox class.

Inputs:

parent: parent object that the window inherits from

Return:

None

Pre/Post Conditions: Calls the init method of the QDialog class. Sets up the file editing box, loads in and displays the contents of the file, and connects button signals to respective slots.

-------------------------------------------------------------------------------------------------------------------------------------------

**handle\_submit**

Purpose: Allows the user to submit the edited BasicML code and builds a usable dictionary.

Inputs:

None

Return:

None

Pre/Post Conditions: FileEditBox must be initialized.

-------------------------------------------------------------------------------------------------------------------------------------------

**Class Name: ColorInputBox**

**Class Type**: Child Class of QDialog

**Description**: This class

**Purpose of this Class**: The purpose of this class is to

**Functions**:

**\_\_init\_\_:**

Purpose: initialize the color input box pop up with all of the necessary attributes.

Inputs:

parent: The window that the pop-up should be initialized in. This input is not required, it will be set to None by default.

Return:

None

Pre/Post Conditions: The main UVSim window must be loaded and the user must click the “change colors” button.

-------------------------------------------------------------------------------------------------------------------------------------------

**get\_inputs:**

Purpose: This function takes the inputs given by the user to change the primary and secondary colors of the UVSim

Inputs:

None

Return:

self.input\_text1.text(): user input in hex for the primary color

self.input\_text2.text(): user input in hex for the secondary color

Pre/Post Conditions: The UVSim must be loaded and the user must click the “change colors” button.

-------------------------------------------------------------------------------------------------------------------------------------------

**Class Name: InputDialog**

**Class Type**: Base Class

**Description**: This class sets up an input pop-up with a text entry box and an enter button.

**Purpose of this Class**: The purpose of this class is to get user input for the read function.

**Functions**:

**get\_input:**

Purpose:

Inputs:

None

Return:

self.input\_text.text(): the text inputted by the user.

Pre/Post Conditions: The input pop-up window must be created.

-------------------------------------------------------------------------------------------------------------------------------------------