Class Documentation for «Tales of Rydonia»

During the development, there is going to be a lot of code involved (you don’t say!). Logically we are going to create a lot of new classes during this entire project and documenting them all inline in the code files. But this is not enough and quite risky, to be honest. Should any of the code files be deleted or otherwise compromised, then the documentation of the classes and their functions is gone.

To prevent this scenario, we created this document. In the following table, we are going to document every self-developed class that we are going to use. This way, we can recreate any method and class that we used in the development, should anything happen to the code files at any time. This is also a good place to have a oversight of all classes and their methods to look up and see what exactly they are for. As an example, we are going to use the **TestClass**. Keep in mind that for every new class, a new table must be generated.

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| TestClass | Class description. | |
|  | **Method** | **Description** |
|  | Public void TestFunction()  string param1 | Method description  Param description |
|  | Public void TestFunction()  string param2 | Method description  Param description |
|  |  |  |

**Color - Description**

Methodnames()

Inherit or Interface, Code

Event Function, Overload, Object-Class, Click\_Event

Not finished

INHALTSVERZEICHNIS

[Class Documentation for «Tales of Rydonia» 1](#_Toc147321640)

[TestClass 1](#_Toc147321641)

[Logger 4](#_Toc147321642)

[GameLogger 4](#_Toc147321643)

[GameManager 5](#_Toc147321644)

[GameLogger 5](#_Toc147321645)

[Game Data Controller 6](#_Toc147321646)

[DataRemover 6](#_Toc147321647)

[GameData 6](#_Toc147321648)

[InventoryData 6](#_Toc147321649)

[TimeAndProgress 6](#_Toc147321650)

[GameDataController 7](#_Toc147321651)

[Story Dialogue 8](#_Toc147321652)

[StoryAsset 8](#_Toc147321653)

[StoryViewer 8](#_Toc147321654)

[StoryHolder 9](#_Toc147321655)

[StoryNode 9](#_Toc147321656)

[StoryUI 9](#_Toc147321657)

[Inventory 10](#_Toc147321658)

[Item 10](#_Toc147321659)

[StatItem 10](#_Toc147321660)

[ItemInfo 10](#_Toc147321661)

[ItemSlot 10](#_Toc147321662)

[Controller 11](#_Toc147321663)

[FileController 11](#_Toc147321664)

[XmlController 11](#_Toc147321665)

[CharacterSelect 11](#_Toc147321666)

[ObjectRotation 11](#_Toc147321667)

[SaveSlotImage 12](#_Toc147321668)

Logger

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| GameLogger | Logs Messages and Exceptions from other Classes.  Needs to be instantiated. (How-To in the Code Conventions Doc / Class) | |
|  | **Method** | **Description** |
|  | Public GameLogger()  string classname | Constructor Takes the Classname as Parameter to create the Log File |
|  | Private void CreateLogFile()  string classname | Creates a Log File with the name of the Class Name of the Class |
|  | Private void LogEntry()  string type  string message  int lineNumber | Creates a new Log Entry by creating a new Log Event Object  Type of Log (Exception, Information, Log, ...)  Message or Exception message to log  Linenumber where the Log happens |
|  | Private void AddLogEntry()  LogEvent log | Appends a new Log to the Log File Log Event Object, created in LogEntry() |
|  | Private void RemoveLogEntry()  LogEvent log | Removes all Log Entries that are older than 5 days. This Method is automatically called whenever AddLogEntry() is called.  Path of the Log File |

GameManager

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| GameManager | Is in Control of the Game, Story and handles the Scenes | |
|  | **Method** | **Description** |
| Event Function | Private void Start() | Start of the GameManager. More information in the class itself |
| Click\_Event | NewGame\_Click() | Opens the character select window and disables the select Images |
| Click\_Event | StartNewGame\_Click() | Checks if a character was selected and a Name was given.  Starts a new game and checks if a save placeholder is empty, else asks to override another placeholder. |
| Click\_Event | LoadGame\_Click() | Loads a saved Game. |
| Click\_Event | NextChapter\_Click() | When the next chapter Button is clicked. |
| Click\_Event | NextStory\_Click() | When the next story Button is clicked. |
| Click\_Event | BackToMainMenu\_Click() | When the back button is clicked. |

Game Data Controller

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| GameDataRemover | This function is only available in the Load Game Screen.  The Player can select a save slot and remove its data. | |
|  | **Method** | **Description** |
| Click\_Event | Public static Remove\_Data\_Click() | Deletes the selected Data. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| GameData | This class contains the Variables that are needed to save the status of the game. | |
| Object-Class | **Method** | **Description** |
|  | Public GameData()  SaveData saveData | Constructor to save an Object of type GameData.  Contains all necessary elements to save the status. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| InventoryData | This class saves and loads the Items from the Inventory. | |
|  | **Method** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| TimeAndProgress | Calculates the elapsed Time and Progress. | |
|  | **Method** | **Description** |
|  | Public static void StartTime() | Starts the Time. |
|  | Public static void StopTime() | Stops the Time. |
|  | GetElapsedTime() | Returns the elapsed Time. |
|  | CalculateProgress()  string chapter | Calculates the Progress.  The chapter is used to calculate the progress. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| GameDataController | This class loads and saves the status of the Game. | |
|  | **Method** | **Description** |
|  | Public class SaveData() | Provides a temporary object to store and load the data |
| Click\_Event | Public void LoadGame\_Click() | Starts either a new game or loads a selected one. |
|  | Public bool NewGame() | There is a limit of 3 Savedata files.  When a new Game is started, it checks if there are 3 savedata files or not.  If not, it just creates a new one,  else the User must choose an old save to override with the new Game. |
|  | Public void LoadGame() | Initializes the Save panel and loads the data into the placeholders. |
|  | Public void InitializeSaveDataPanel()  String text  Int index | Initializes the Savedata panel  Disables the check Images of all placeholders  Sets the Button text and the overview text.  Text for the Button, either Load Game or New Game.  Identifies which text from the xml file should be displayed. |
|  | Private void LoadDataIntoPlaceHolders() | Gets all save data files and stores the data in the placeholders. |
|  | Private void UpdatePlaceholderView()  Int placeholderNum  IReadOnlyList<SaveData> loadedData | Updates the Placeholder view with data.  If loaded data is null, then the placeholder is empty.  Placeholder number where the data has to be placed.  List of all loaded data to display. |
|  | Private static void LoadSelectedGame() | Loads the data for the selected game  In case the deserializing throws an error, the method is executed again. |
|  | Private void SaveNewGame() | Saves the name and the Character Properties as first Save for a New Game in a Json File. |
|  | Public void SaveGame()  SaveData save | Saves the status of the Game in a Json File.  The SaveData sent from StoryUI. |

Story Dialogue

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| StoryAsset | This class reads the content of the Story xml files.  Creates asset files by saving the node Information in Json Files. | |
|  | **Method** | **Description** |
|  | Public class NodeInfo() | This class is needed to evaluate if a node needs to be added or removed. |
|  | Public StoryAsset ReadNodes()  StoryAsset chapter | Reads the Node Information from the Xml File and puts them in the right order.  The name of the chapter is needed to find the according Json file. |
|  | Public void ReadJsonFile()  String path | Reads the Json File that matches the chapter, and deserializes each object into StoryNodeDataProperty object  The path to the Json file |
|  | Public void ReadNodesFromXmlFile()  XmlDocument xmlDock path | Reads the Xml file that matches the chapter and processes each node.  Remove nodes that ore not needed anymore.  Reads the properties of the rest of the nodes.  The XmlDocument info of the file to read. |
|  | Private void ReadProperties()  StoryNode node  XmlDocument xmlDoc | Reads the Properties from the Xml to the according node.  Node whose properties must be read.  The currently opened xml document. |
|  | Public void SetNodePosition() | Sets the position of all nodes. |
|  | Private static StoryNode CreateNode()  XmlNode node  string id  bool isChoice | Creates a new Node and a unique GUID an id, a label, a text and a type of the node (choice or story boolean).  Node name.  Id Attribute from xml File  Declares if node is a choice or not |
|  | Private void SaveNodesToJson() | Writes the nodes in a Json File. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| StoryViewer |  | |
|  | **Method** | **Description** |
|  | Public static bool OnOpenAsset()  Int instanceId | Shows the Editor Window, depending on if a Story is loaded or not. Returns the StoryAsset.  Instance ID of the Editor Window. |
|  | Private void DeleteAsset() | Custom menu item.  Deletes the asset file and Json file with the same name. |
| Event Function | Private void OnEnable() | When the Dialog is Enabled, initialize the node Styles. |
|  | Private void OnSelectionChanged() | Loads the selected chapter |
| Event Function | Private void OnGUI() | Gets called every time a change happens on the Editor. Displays the nodes. |
|  | Privates static DrawSurface() | Draws the Surface of the Editor Window. |
|  | Private bool DrawNode()  StoryNode node | Draws the nodes.  Next node to draw. |
|  | DrawConnections()  StoryNode node | Add Bezier Curve between the nodes to connect parent and child nodes. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| StoryHolder | Holds the Story and provides information about the selected nodes and next nodes. | |
|  | **Method** | **Description** |
|  | Public void LoadChapterProperties()  StoryAsset chapter | Sets the Information for the current chapter and nodes (New Game), or has to get the information from the savedata (Load Game).  Is either null or a new chapter. |
|  | Private StoryNode GetNextNode()  StoryNode selectedChoice | Returns the next story node.  Parent that contains the next choices nodes. |
|  | Public StoryNode GetNodeBefore() | Gets the node that was selected before. |
|  | Public List<StoryNode> HasMoreNodes()  StoryNode nodeToDisplay | Get all child nodes from a specific node.  From this node the child nodes are needed. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| StoryNode | Object Class for StoryNode. | |
| Object-Class | **Properties** | **Description** |
|  | Public void InitializeStoryNode()  StoryNodeDataProperty node | Initializes a new StoryNode.  Needed to serialize and deserialize the story nodes. |
|  | Public void AddChildNode()  String childId | Adds the node name to the child nodes list.  Child to add to the list |
|  | Public void RemoveChildNode()  String childId | Removes node from child nodes.  Child to remove from the list. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| StoryNodeData | Object Class to save the Story nodes. | |
| Object-Class | **Properties** | **Description** |
|  | Public StoryNodeData()  StoryNode node | Constructor to save an Object of type StoryNode.  Contains all properties. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| StoryNodeDataProperty | Object Class to cache the Story nodes. | |
| Object-Class | **Properties** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| StoryUI | Displays the Story  Takes the node information and puts in the Game objects so the Story can be played | |
|  | **Method** | **Description** |
|  | Private void Start() | Hands over the current chapter to the Story holder,adds the next button click Event and updates the UI. |
| Click\_Event | Private void Next\_Clickt() | When the next button is clicked, it loads the next part of the story |
|  | Private void UpdateUI()  Bool isSave  Bool isGameOver | Updates the UI, loads the next story or choice nodes and their properties. Saves the node state.  True the game is saved, false then not.  If true, then only the choices which haven't been selected are visible. |
|  | Private void NextChapter() | Loads the next Chapter, when the End of Chapter node is reached.  Loads the next Story, when the EndOfStoryNode is reached.  Loads the GameOver Screen, when the GameOver node is reached.  Loads the End of Tale Screen (End Credits screen), when the Game is finished. (Not yet developed) |
|  | Private void BuildChoiceList()  Bool isGameOver | Builds the choice list, depending on the count of the nodes. Some choices are only visible for players with the required background.  If true, then only the choices which haven't been selected are visible. |

Inventory

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| Item | This class gives a Scriptable object that is then used in the inventory as an item. | |
|  | **Method** | **Description** |
| Click\_Event | Public virtual void Use() | Uses the item. At the moment it only debugs |
| Event Function | Public virtual string GetItemDescription() | If there are any save files, then the LoadGame Button in the Menu is activated |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| StatItem | This item type can give boons to the player such as health, food etc. Could be used in a later state of the game | |
|  | **Method** | **Description** |
| Click\_Event | Public override void Use() | Uses the Statitem to give stats to the player at a later stage. |
| Datatype | Public enum StatitemType() | Defines Statitem types such as health items, food items etc. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| ItemInfo | This class returns the information about the item. Namely the item description as well as the item name. | |
|  | **Method** | **Description** |
| Event Function | Public void SetUp()  Text itemName  Text \_itemDescription | Uses the Statitem to give stats to the player at a later stage.  Return the name of the item.  Return the description of the item. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| ItemSlot | Adds an item slot to the inventory windows. I this slot, an item is then used to be put into the item slot. | |
|  | **Method** | **Description** |
| Event Function | Public void AddItem()  Item newItem | Add items to the item slots.  The item that should be added to the slot. |
| Click\_Event | Public void UseItem() | Uses the item and gets the function from the item class. |
| Event Function | Public void DestroySlot() | Destroys item slot when the item is used or removed by the user. |
| Click\_Event | Public void OnRemoveButtonClicked() | This function removes an item from the inventory. |
| Hover\_Event | Public void OnCourserEnter() | This function activates the tooltip over an item, when it is hovered over it. |
| Hover\_Event | Public void OnCourserExit() | This function deactivates the tooltip over an item, when the courser is removed from said item. |

Controller

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| FileController | Controls the CRUD of Files and Folders | |
|  | **Method** | **Description** |
|  | Public static void CreateFolders() | Creates Folders  SaveData: stores the Save Data from the Game  StoryAssets: stores the node Information as Json files |
|  | Public static void SortSaveFilest() | Sorts and renames the files after one was deleted |
|  | Public static void DeleteFile()  string path | Deletes a specific file.  path to the file, that needs to be deleted |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| XmlController | This class contains Methods to access Xml Files.  Returns XmlDocument, XmlReader, XmlWriter, XmlNode, Node InnerText or Content. | |
|  | **Method** | **Description** |
|  | Public static string GetMessageBoxText()  int index | Reads the MessageBox Nodes from the OutsourcedStrings/strings.xml  Determines the node to return. |
|  | Public static string GetInformationText()  int index | Reads the Information Nodes from the OutsourcedStrings/strings.xml  Determines the node to return. |
|  | Public static string GetChapterTitle()  StoryAsset currentChapter | Reads the title node from the current StoryAssets/StoryChapter.xml  Current Chapter that is loaded |
|  | Public static XmlDocument GetXmlDocOfStoryFile()  string chapter | Reads the Node information of the chapter  The current chapter that need's to be created |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| CharacterSelect-Controller | When a Character is selected. | |
|  | **Method** | **Description** |
| Click\_Event | Public void Character\_Click() | Loads the Character name in the invisible label, so the GameDataController knows which character to save. |
|  | Private void SetImage() | Sets the select Image, when a character is selected. |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| ObjectRotation-Controller | Rotates an Element around the Z Axis. | |
|  | **Method** | **Description** |
| Event Function | Private void Update() | Every frame the Game Object gets rotated |

|  |  |  |
| --- | --- | --- |
| **Class** | **Description** | |
| SaveSlotImage-Controller | When a save slot is clicked. | |
|  | **Method** | **Description** |
| Click\_Event | GameObject\_Click() | Sets the select Image, when a save slot is selected. |