Technical Design Document

Robert McDonagh - C00186196

# CRC Cards:

|  |  |
| --- | --- |
| Resource Manager |  |
| Add resource to queue  Load resource  Add texture  Add music file  Add sound effect  Check JSON object  Reload textures  Reload music  Reload sound effects  Getting time information of resources  Reload from XML file  Reload from JSON file  Reload from text file  Stores all textures  Stores all music files  Stores all sound effects  Stores all paths to resources  Stores all animation frames | Resource |

|  |  |
| --- | --- |
| Resource |  |
| Stores the key value for the resource |  |

|  |  |
| --- | --- |
| Texture |  |
| Stores the directory to the texture | Resource |

|  |  |
| --- | --- |
| Music |  |
| Stores the directory to the music file | Resource |

|  |  |
| --- | --- |
| Sound Effect |  |
| Stores the directory to the sound effect | Resource |

# Class Diagram:

Link: <https://drive.google.com/open?id=0B9TpPdUYFsghYW5wRGhnNzlrTTQ>

# Sequence Diagram:

## Resource Manager:

Link: <https://drive.google.com/open?id=0B9TpPdUYFsghdEFTSF9HNkpWZGM>

# Approaches:

## Resource Manager:

1. Create a singleton class that loads and stores all of the assets.
   1. Create resource manager
   2. Loads all assets
   3. Reads from JSON
   4. Reads from XML
   5. Reloads assets without restart
   6. Can be accessed anywhere in the application
2. Create a class that loads and stores all of the assets.
   1. Create resource manager
   2. Loads all assets
   3. Reads from JSON
   4. Reads from XML
   5. Reloads assets without restart
   6. Can only be accessed by the object created

# Features:

## Resource Manager:

Create a singleton that loads all of the resources into the game and stores them in maps.

### Task: Load resources from Text

Create a function that loads all the resources into the game by reading the values from a text file.

* void ResourceManager::loadResourcesFromText(string fileName)

// Passes the filename of the text file to read and adds all the resources to a queue.

* void ResourceManager::loadResourceQueue()

// Loops through the resource queue, loading one resource at a time.

* void ResourceManager::loadResource(Resource\* resource)

// Passes a resource that then loads a texture, sound effect or music file depending on

// the resource

### Task: Load resources from JSON

Create a function that loads all the resources into the game by reading the values from a JSON file.

* void ResourceManager::loadResourcesFromJSON(string fileName)

// Passes the filename of the text file to read and adds all the resources to a queue.

* void ResourceManager::loadResourceQueue()

// Loops through the resource queue, loading one resource at a time.

* void ResourceManager::loadResource(Resource\* resource)

// Passes a resource that then loads a texture, sound effect or music file depending on

// the resource

### Task: Load resources from XML

Create a function that loads all the resources into the game by reading the values from a XML file.

* void ResourceManager::loadResourcesFromXML(string fileName)

// Passes the filename of the text file to read and adds all the resources to a queue.

* void ResourceManager::loadResourceQueue()

// Loops through the resource queue, loading one resource at a time.

* void ResourceManager::loadResource(Resource\* resource)

// Passes a resource that then loads a texture, sound effect or music file depending on

// the resource

### Task: Reload assets without restart

Create a function that will reload resources that have been changed while the game is running.

* void ResourceManager::update(float deltaTime)

// Passes the delta time so to check each files time information

* Struct tm ResourceManager::getTimeInfo(const char\* path)

// Passes the path of the file and returns the time information for that file

* inline bool isOutOfDate(struct tm current, struct tm new)

// Passes the current time and new time informations to compare

* void ResourceManager::reloadTexture(string key)

// Passes the key value of the texture we want to reload, safely disposes of the old texture

// and replaces it with the newly edited one

* void ResourceManager::reloadFromXML()

// Reloads the animation frames from the xml file first used

* void ResourceManager::reloadFromJSON()

// Reloads the animation frames from the json file first used

* void ResourceManager::reloadFromText()

// Reloads the animation frames from the text file first used

### Task: Safely dispose of resources

Create a function that safely disposes of all resources.

* void ResourceManager::destroy()

// Deletes the instance of the resource manager which calls the destructor

* void ResourceManager::~ResourceManager()

// Iterates through each map and safely disposing of any resources