|  |
| --- |
| SFWR ENG 3Gc3 |
| SphereQuest |
| Project Documentation |
|  |
| **Jason Costabile, Ben Kybartas, & Kaitlin Smith** |
| **4/5/2010** |

|  |
| --- |
|  |

# Introduction

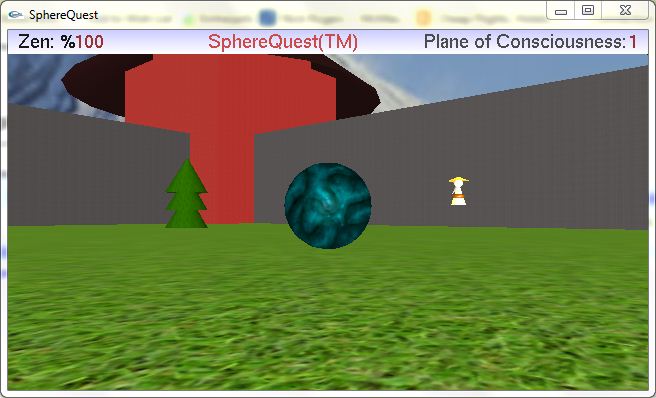
Welcome to SphereQuest! SphereQuest is a game designed to bring you inner peace by bringing you on a personal journey through your inner self. Find the wise men throughout the mazes and answer their riddles to ascend to a higher plane of consciousness!

# User Manual

HUD

Current Plane of Consciousness

Your Zen level



Obstacles

Wise man

Sphere (you!)

## Objective

The objective of SphereQuest is to reach the sixth Plane of Consciousness, where your spiritual journey of self-realization will come to an end. Before you can ascend to the next Plane, you must locate the wise man and successfully answer his riddle.

## Installation

To install SphereQuest, simply unzip the provided archive into your preferred directory. Run the executable file to play the game.

To uninstall, delete the SphereQuest folder where the unzipped files were placed.

## Movement

To navigate the sphere through each Plane, simply press the arrow keys on your keyboard. The sphere will glide in the direction of the arrow key currently pressed.

The sphere is not allowed to simply move where it pleases. Throughout the maze, you will find obstacles which you cannot pass though – walls, trees, and temples. You may not move travel through the wise men, but it is easy enough to move around them.

## Riddles

Once you have successfully located a wise man, approach him to be asked a riddle, along with a choice of three answers. Search your soul and think carefully about the answer. Once you have made your decision, press the function key that corresponds to your chosen answer – each answer is labelled with the key to press.

If you have chosen the correct answer, the sphere will move to the next Plane of Consciousness. This can be confirmed by reading the Plane of Consciousness indicator in the top-right corner of the HUD (heads up display).

If you have chosen an incorrect answer, you will lose a portion of your Zen. You may monitor your Zen level by reading the Zen meter in the top-left corner of the HUD. Approach the wise man again for another opportunity to answer his riddle.

## Winning and Losing

Winning SphereQuest occurs when you have successfully reached the sixth Plane of Consciousness before your Zen reaches zero.

Losing SphereQuest occurs when you have answered too many riddles incorrectly, causing your Zen to reach zero. If this happens, you will be presented with the option to try again. If you decide to try again, you will appear back on the first Plane of Consciousness, ready to retry your spiritual journey.

## Saving and Loading

If you need to take a break from your journey, you may save your current progress. Right-click anywhere within the game window to be presented with the Save/Load Menu. Choose a slot to save in and a save file will be created for you. To load a game you’ve saved, right-click again and choose the slot to load in which you saved your game. When saving a game, take care to not save over another journey, as it is not possible to recover a saved-over file.

## Cheats and Developers’ Shortcuts

Several shortcuts were implemented to aid in the development of SphereQuest, and can be used as cheats if desired.

To display grid lines, press the J key. Press it again to remove the grid lines.

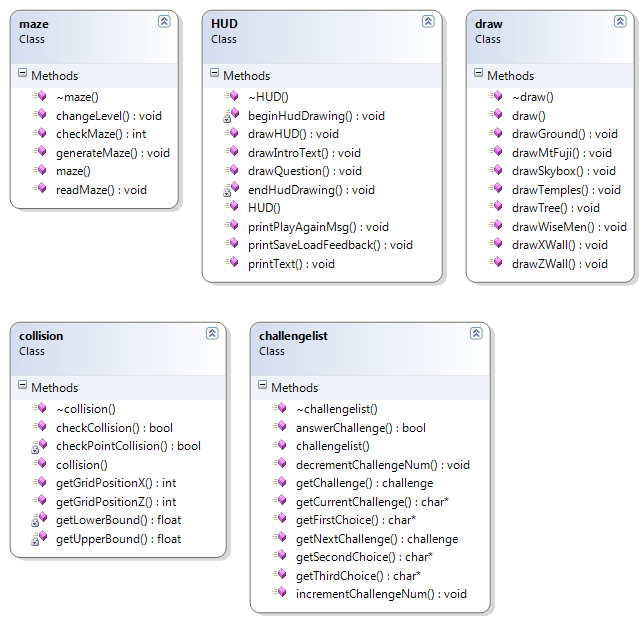
To skip the current Plane of Consciousness, press F4. To automatically win, press F5. Pressing F6 will cause you to automatically lose.

# Code Documentation

Please make sure to browse through the code as there are many comments documenting the purpose of individual code chunks.

## Class Breakdown

A quick breakdown of the classes in the SphereQuest project is given in the class diagram below.



The SphereQuest project does not use class inheritance (each class is independent), so no connections are drawn between the classes. Public methods are represented by a purple brick, which private methods are represented by a small lock beside the purple brick. Every class contains a constructor and destructor method.

## The maze Class

The maze class contains all code for creating the maze in which the user navigates through.

### The checkMaze Method

This method takes integers representing a row and column of the maze and returns an integer representing the item that is located in that grid location. The table below summarizes the grid items:

|  |  |
| --- | --- |
| Integer | Grid Item |
| 1 | A wall placed on the x-axis. |
| 2 | A wall placed on the z-axis. |
| 3 | A temple. |
| 4 | A wise man. |
| 5 | A tree. |

### The changeLevel Method

This void method takes an integer representing a level number and loads the corresponding maze file from the provided maze folder. The loading of the file is handled by the readMaze method described below. This method also modifies a global level variable and sets it to the changed level.

### The readMaze Method

This method takes a string representing a file name and loads that requested file to read the maze from. Maze files are located in a ‘mazes’ folder and the maze data is stored in text file. Please take the time to examine the maze files and note their simple structure.

If the requested maze file is loaded successfully, the method will read the file, character by character, and store the current character in the mazeLayout array declared globally in the maze class. If the method is unable to load the file, an error is printed to the console.

## The HUD Class

The HUD class takes care of drawing the elements of the heads up display.

### The printText Method

This method takes two floating point numbers representing screen coordinates, a character array representing the text to be printed, and three floating point number representing an RGB value. This method then applies the given colour, positions the text on the screen, and then prints the character array using glutBitmapCharacter.

### The drawIntroText Method

This method uses the printText method above to print the text which appears when the game first starts.

### The drawQuestion Method

This method takes several character arrays as inputs which represent the question to be printed as well as the three choices for answers. The printText method is used once again to display the text.

### The printPlayAgainMsg Method

This method accepts a Boolean value representing whether or not the player has won the game. If the value is true, this method uses the printText method to print a message indicating that the player is victorious. If the value is false, this method uses the printText method to print a message indicating that the player has lost. In both cases a line of text is also displayed prompting the user whether or not he would like to play again.

### The printSaveLoadFeedback

This method accepts two Boolean values, one representing a save or load took place, and whether or not the action was successful. Depending on the values of the Booleans, if statements choose which messages to print using the printText method.

### The drawHUD Method

This method’s responsibility is to display the HUD.

**Look into your Heart**