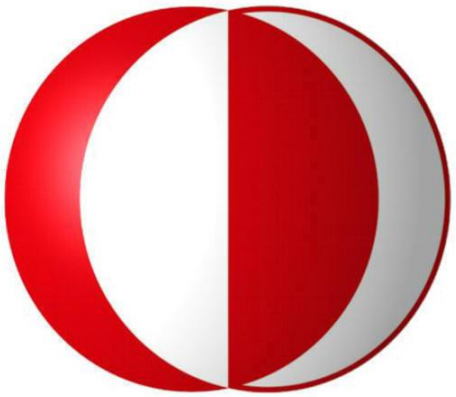
BIG CENG THEORY

Initial Design Report

Middle East Technical University



19.12.2011

INDEX

1. INTRODUCTION

This initial design document is prepared to give design descriptions for single-player educational physics game project Fizbot which is sponsored by Words To Inspire for Fatih project and carried out by Big Ceng Theory project group.

1.1. Problem Definition

Most of the high school students have difficulties in learning physics. The major reason is that the physics in high schools is taught as it is only a course subject which they have to pass and not related with the real world. Therefore, students cannot associate the knowledge they learned in class with real life and the knowledge cannot go further than a memorization. However, we know that the better way to learn physics is to visualize the concepts and relate them to daily life. Considering that students at that age are more proned to play games than studying physics which is also a severe problem, we can propose a solution that combines physics with computer games. From childhood to adulthood, games have an important role in learning, since it helps increasing imagination, creative thinking, curiosity and exploring the environment with fun. Therefore, a game that uses ninth grade physics as rules may have a great contribution for the solution of this problem. The reason why we chose ninth grade is it covers the very basic and significant parts of the physics and has too many examples in real world. Despite there are many educational physics game in the market, the need for such game is still exists. This is because the existing games are either question answer type which most students find boring or very funny but not sufficient for the level of high school students.

1.2. Purpose

The main purpose of this document is to explain all the design descriptions of Fizbot. The explanations are primarily targeted for programmers who will develop this project to satisfy all the requirements.

1.3. Scope

This software design document includes design patterns giving brief explanation about the goal of the project, design constraints, assumptions, dependencies, detailed data description with data dictionary, system architecture with its components, user interface, libraries and tools and time planning. The main objective is to provide design description so that by following this document, the construction of the system will easily satisfy the requirements in the SRS.

1.4. Overview

This document has 9 sections to state the initial design of the project. In the following section, we will give a brief system overview. In section 3, we will state the design constrains, assumptions and dependencies. In following section, detailed data description and data dictionary will be provided. In section 5, system architecture will be covered with its components. Section 6 will give information on interface design. Libraries and tools that will be used in the project are explained in section 7. Finally, a revised Gantt Chart is provided in section 8 in order to state the time line determined for the progress of the game.

1.5. Definitions, Acronyms and Abbreviations

SRS: Software Requirements Specification
IDD: Initial Design Document

1.6. References

FatihProjesi.(n.d.)Retrievedfromhttp:http://fatihprojesi.meb.gov.tr/site/

IEEE, IEEE Std 1016-1998 IEEE Recommended Practice for Software Design Descriptions. IEEE Computer Society.

2. SYSTEM OVERVIEW

In our game, the user will start to play the game first by selecting an avatar. This avatar will be the main character of the game. There will be two different choices for avatars. The user can choose either a girl or a boy avatar. After choosing an avatar the general story is as follows. The character will firstly encounter with a witch that curses the world by imprisoning two scientists who are very significant for world to develop namely Archimedes and Sir Isaac Newton. With the effect of the curse, those two scientists will not be able to work on their inventions and today's world will be in a chaos.

The character is expected to save those two scientists by going back in time and following the clues provided by "Fizbot" which is a robot wants to help the character. There are two levels hence two scientists to rescue. In every level there are two quests we want the player to solve. These quests we chose are related to the physics topics taught in 9^{th} grade and also to the scientist's research area. During solving two quests using the clues, the character will gain a key to open the door in which scientist is kept and move on to save the other in the same manner. Saving each scientist will develop world further and after saving both of them it will be the same as the one before the witch's curse.

The curse of the witch, the development of the world and the final scene after finishing the game will be provided as videos at the beginning, in between the levels and at the end of the game respectively.

The topics of the quests are: buoyancy of liquids, simple machines, conservation of energy and momentum and equilibrium respectively.

In the quest of buoyancy of liquids, the character will encounter with a pool which is filled with water. Inside the water there is a long, thin wood whose half is sank into the water and the character is not able to reach it. At the bottom of the wood there is a key attached and character must take this key to save the scientist at the end of the level. Nearby the character there are four valves containing different liquids. The aim of the character is to change the density of the water by adding different liquids to the pool by using valves in order to reach the key. If the density of water is decreased, the wood will sink more and character will not be able to reach it. In this case the liquid added to the pool will be thrown from the hole at the top of the pool and character will be able to add new liquids to the pool. Therefore, character needs to add a liquid that will increase the density of water in order to reach the key. When character reaches the key, he/she can move to the second task which is the simple machine quest.

The next task is a simple machine problem which the player has to solve a puzzle composed of rotors. In order to open a door, the player needs to find a code related to the rotors' number of rotation. After he/she solves three puzzles, he/she will be able to open the door using the key gained in the previous step. After saving the scientist player will move to next level.

The first task of the second level is related to the conservation of energy. In this quest the player will encounter with a deep rift. Therefore, he/she is not able to pass across to the other side. At the other side, there is a ruined university which is destroyed by the witch at the beginning of the game where Newton had studied. In order to enter the university, a bridge must be dropped from the entering of the university to cover the rift. There is also a ball that character will throw. Bridge will drop slowly when the ball is thrown by the character and reaches the corresponding buttons which are on the other side of the rift. For the first shoot there will be no friction on the rift. However, when a ball thrown by the character touches the button, bridge drops a little bit and some friction will be added to the environment. After character presses all four buttons, the bridge will be down completely with a key at the tip of it. This is the end of this task.

The next task is related to the momentum and equilibrium. In this task player needs to walk on a seesaw without disturbing the balance of it. In order to achieve this, he/she has to change the positions of the objects on the seesaw.

Passing all the four steps means completing the game and saving the world from the curse.

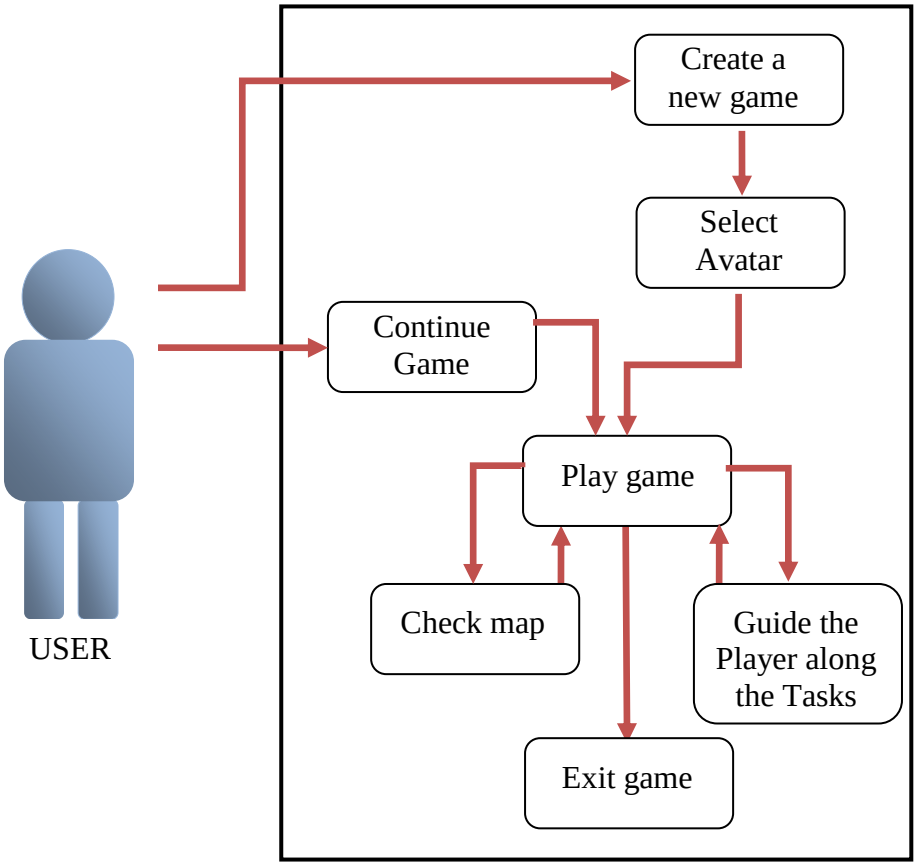


Figure-1 Product Functions

3. DESIGN CONSIDERATIONS

3.1. Design Assumptions, Dependencies and Constraints

The game will be developed on tablets and the operating system that these tablets are assumed to use is Android OS.02. Since the game is a part of the “Fatih Project”, the tablets are determined by Ministry of Education. The game will be developed using “Unity3D”, because it is a good way to design a game on Android platform. Moreover, the most important feature of Unity3D is that when a game is developed in Unity3D, it is capable of working with full functionality on multiple platforms and operating systems by only writing a plug in for different platforms and integrating the code. This feature will help us to adapt our code in different platforms easily. The tablets that are provided by Ministry of Education are able to meet the minimum requirements of Unity3D. The processor of the tablets is not determined yet, however it will be single or two core processor. The core of the processor will not affect the performance of the game significantly. It is assumed that the game will respond to the user in 0.1 second however this time can be changed according to the processor used in the tablets. The tablets have the necessary two-dimensional input hardware such as touch surface. The tablets supports also multi touching.

It is assumed to make two levels for completing the game and for each level students have to solve two different tasks in each level. However, according to the going process the number of the level or tasks can be increased. Moreover, the number of avatars is assumed to be two for this moment. However; this number can also be increased according to the process.

The game will be used only with one user. No other users can connect to the game at the same time. The number of avatars that the user will be able to select before starting to the game will be limited. The language that will be used in the game will be in Turkish. No other language will be provided to the user. Since the game will be first used for "Fatih Project", it will only be used in tablets.

3.2. Design Goals and Guidelines

- Maintainability: It is the ease that a product can be maintained in order to correct defects easily, meet new requirements, cope with a changed environment and make future maintenance easier.
- Reusability: helps the developer to add new functionalities to the code with slight or no modification. Using reusable modules and classes reduce implementation time, increase the likelihood that prior testing and use has eliminated bugs and localizes code modifications when a change in implementation is required.
- KISS principle: it will be a guideline the help us to maintain the design simple as possible during the design process. Moreover, avoids unnecessary complexity of the product.
- Usability: Since the product will be used by 9^{th} degree student, It is one of the most important design considerations for our project. Therefore, the tasks used in the game should be simple, understandable and informative for students.

4. DATA DESIGN

4.1. Data Description

Since our game is heavily story based, the main data object in the project is GameManager and AvatarManager objects. AvatarManager, SoundManager, TransitionManager, ItemManager, TaskManager, MapManager, SceneManager objects are all associated with the GameManager object. Also AvatarManager object is associated with TaskManager and ItemManager objects so that the story will be shaped according to the actions of the character on items in specific tasks. Since there is not database in the project we will use a txt file to keep the information of the saved game. Following parts will introduce these data objects and their attributes in detail.

4.1.1. Data Objects

AvatarManager: This data object is representing the avatar choice of the leading character of the game that is the center element and controlled by the user.

Figure-2 AvatarManager object

avatarID: holds the avatar information of the leading character that is selected by the player at the beginning of the game.

avatarName: holds the name of the leading character that is entered by the player at the beginning of the game if it is not entered a default name will be used during the game.

avatarImage: holds the avatar graphics designed by the graphics designer which represents the avatar of the player.

SceneManager: This data object is used for graphics of the game. They will be loaded according to their ids as the game processed.

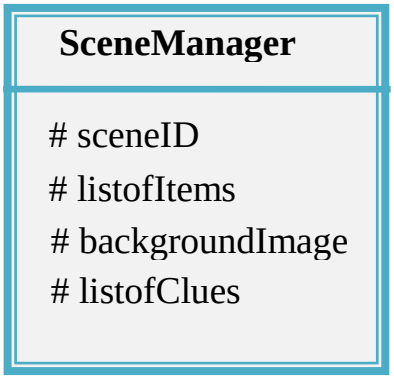


Figure-3 SceneManager object

sceneID: holds the id of the graphics to be loaded when necessary in order to compose the visuals of the game.

listofItems: this data object is representing all the items that can be seen in the game which are going to be helpful for user to solve problems and also the items which are going to be displayed in the current scene of the game. From witch and Fizbot to arbitrary objects in the game are all defined as items and separated by the type parameter.

backgroundImage: holds the graphics designed by the graphics designer for the non-altering background of the game. They are going to reflect the characteristical view of the current time which the character traveled through.

listofClues: holds the clue items that the character picked until he/she reaches the task. These items will be displayed on the screen and character will be able to read these clues whenever he/she wants.

ItemManager: This data object is holding information of each item which is in the listofItems of sceneManager object.

Figure-4 ItemManager object

itemImage: holds the graphics of the items in the current scene designed by graphics designer.

itemID: holds the id of the graphics of the item to load when necessary.

isonScene: represents whether an item is on the scene or not.

itemPosition: holds the position information of the item in the scene.

GameManager: This data object is representing the whole game and includes all the data objects stated in this section.

Figure-5 GameManager object

position: holds the position information of the character in the game.

timeLeft: holds the time remained in order to be able to give feedback to the player at the end of the game as points.

currentLevel: holds the level information which the character is at in the game currently.

currentTask: holds the task information which the character is at in the game currently.

MapManager: This data object is representing the map screen of the game in order to show user which year he/she is currently at and which tasks are completed, which are not.

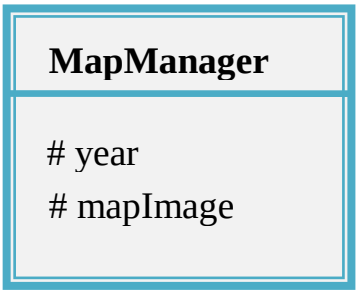


Figure-6 MapManager object

year: holds the year information of the game according to the levels. The reason to store this data is the game includes time travels and player may need to know which year he/she is currently at.

mapImage: holds the graphics for representing the map which is designed by the graphics designer.

TaskManager: This data object is representing each quest in the game.

Figure-7 TaskManager object

levelID: holds the id information of the level to load proper quest.

taskID: holds the id information of the task to load proper quest.

When new tasks are added to the project, it will be easy to adopt them to the game simply by creating new task object with new level and task id.

SoundManager: This data object is representing the sounds that will be used to inform the player about the situation. Also this object will hold the general music which will be played during the game.

Figure-8 SoundManager object

musicID: holds the id information of the music which will be played when necessary in the scene.

musicFile: holds the music to play according to the id when necessary.

TransitionManager: This data object is representing the transition videos that integrate the tasks to each other. This object will be very helpful for the player to understand the story in a continuous manner.

Figure-9 TransitionManager object

taskCompleted: holds the task information which is lastly passed to show the related video to the player.

levelCompleted: holds the level information which is lastly passed to show the related video to the player.

videoID: holds the video id which will be displayed when needed.

SavedGame.txt: This file holds the position, timeLeft, currentTask and currentLevel attributes of GameManager object and avatarID and avatarName attributes of AvatarManager object in order to resume a saved game from the position where the player quit the game. The attributes will be separated by new line character in the txt file.

The format of the graphics files which will be prepared by the graphics designer is not determined yet, however, they can be either Photoshop files or png files.

4.1.2. Relationships

In this section we will introduce the relationships between all the data objects described above.

AvatarManager-GameManager: GameMAnager object has only one AvatarManager object in the game.

ItemManager-GameManager: GameManager object may have one or more ItemManager objects representing that there can be one or more items available in the game environment.

MapManager-GameManager: GameManager object has only one MapManager object that is dynamically changed.

TaskManager-GameManager: GameManager object may have one or more TaskManager

objects.

AvatarManager-TaskManager: AvatarManager object will be interactive with a TaskManager objects to pass in order to proceed.

AvatarManager-ItemManager: AvatarManager object may have one or more ItemManager objects to use in order to solve problems.

SceneManager-GameManager: GameManager object may have one or more SceneManager objects to use in order to preceed the story of the game.

SoundManager-GameManager: Game object has one or more SoundManager object that is dynamically changed.

TransitionManager-GameManager: Game object has one or more TransitionManager object that is dynamically changed.

4.2. Data Dictionary

avatar: is a visual object in the game that represents the player. Players can separate themselves from other characters in the game by this visual. The most important part is that the avatar is under the control of the player.

character: means the player in the game environment, the hero/heroin of the game.

task or quest: is the each problem need to be solved by the user in levels. There 2 tasks in every level.

level: is the each step of the game that the player has to pass in order to succeed. There are 2 levels in the game.

Item: means all the objects in the game environment available for player to use in solving the problems, for example; bridge, key, tree, Fizbot, witch etc.

background: can be thought as contiguous pictures to compose the story of the game as the character walks.

5. SYSTEM ARCHITECTURE

5.1. Architectural Design

Our system will have a modular structure, therefore it is divided into subsystems which will accomplish the goal of the whole system. There will three components or subsystems, which are namely:

PreGame Module

Game Module

PostGame Module

Component diagram of these subsystems is given below:

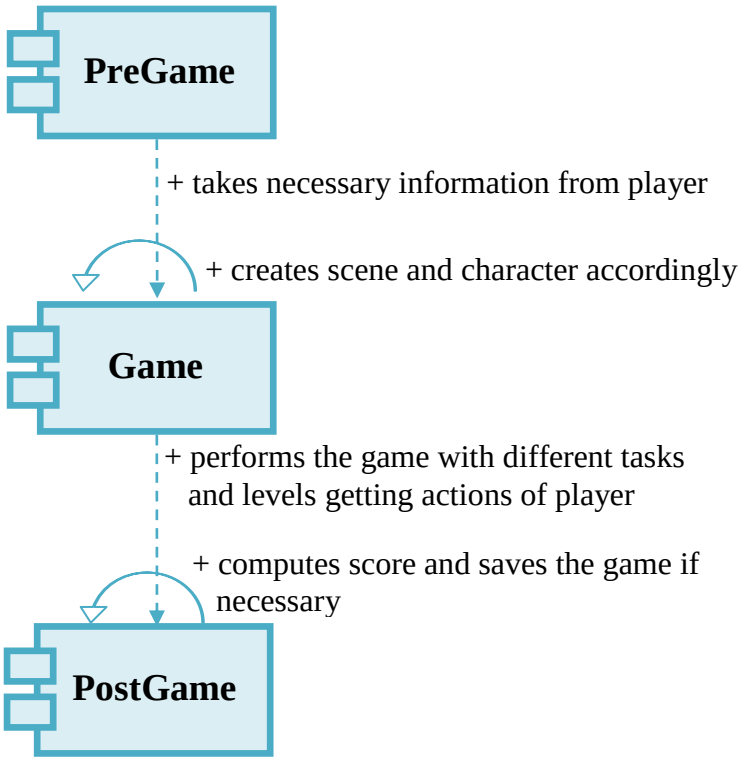


Figure-10 Component diagram of the system

First component of the system is PreGame. In that stage, necessary information from the player will be taken, based on which the game will be prepared for the player. After taking those information, scenes, characters, and if necessary items will be adjusted and placed accordingly.

The second one is Game component, which is composed of two submodules, namely levelTransition and gamePlaying. In the levelTransition as the name reveals itself, player will be informed about the previous and upcoming level of the game and system will make sure that player follows the game appropriately. The gamePlaying subsystem is going to be where the actual game scene and characters will take place.

The last one is PostGame subsystem. This subsystem gives us the opportunity to save the game in one determined file format. For this reason, operations on these files will be simpler. If desired by the player, game will be saved for the future play. If not, current condition will be shown.

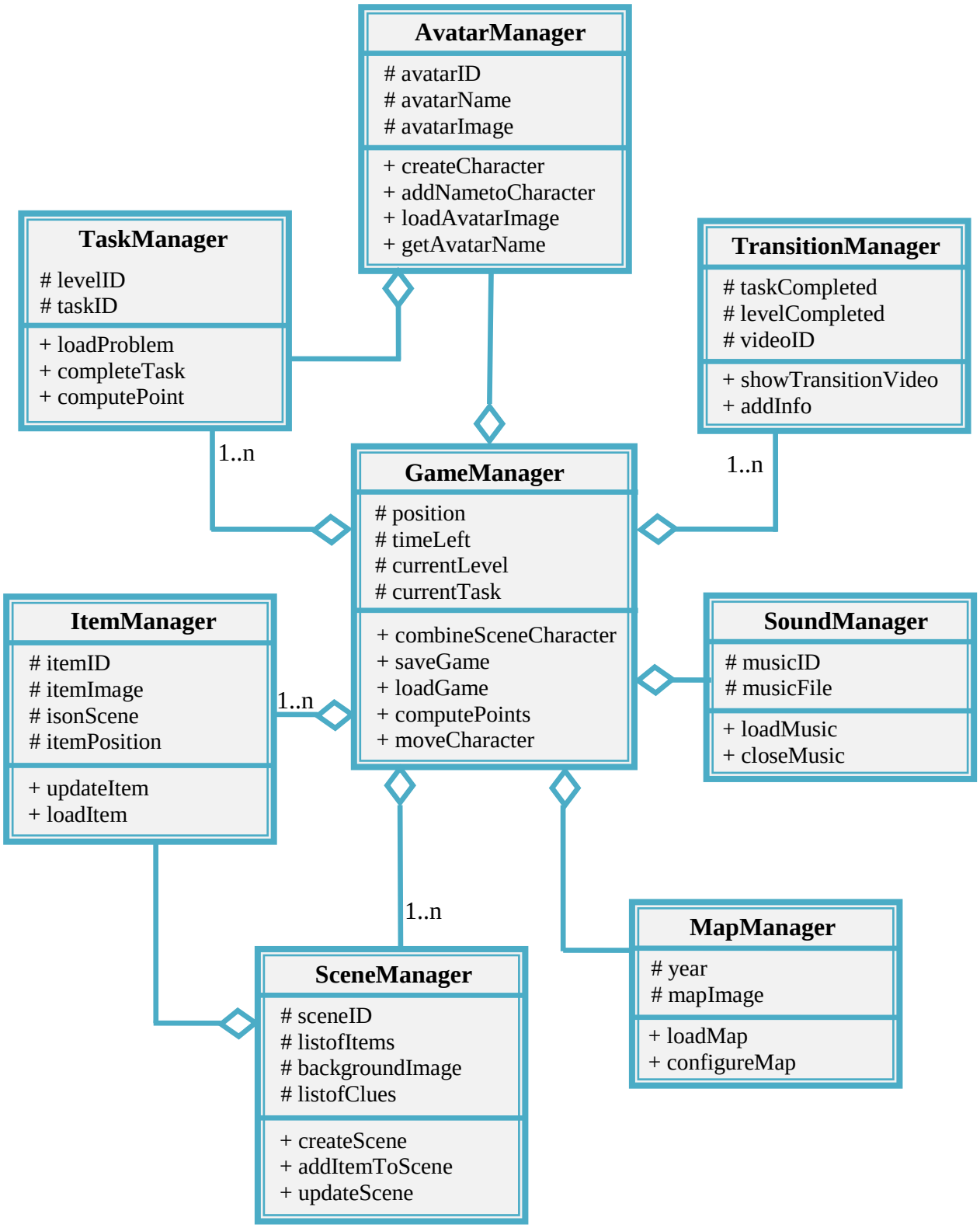


Figure-11 Class diagram

5.2. Description of Components

5.2.1. PreGame Component

5.2.1.1. Processing narrative for PreGame component

This is the subsystem which is responsible from taking all the configurations that the player will give to the system. After passing this system, player will never be able to change any configuration regarding the character, game or settings. To be able to make such changes, he/she will have to come back to that stage, make the desired changes and start the game all over again. This component will also be responsible from some user actions, such as loadGame, startNewGame, createCharacter. Besides those player-oriented actions, some systems actions will take place, such as startGame, drawScene.

5.2.1.2. PreGame Component Interface Description

For this subsystem, user will be supplied a menu which he/she can configure some settings and take some actions mentioned in the previous sections. The components' input interfaces are the events that player triggers. Such events are mouse clicks, keyboard entries etc.

In this stage, user will also be able to choose an avatar that will be used throughout the game. For this purpose output of such action will be the avatar choices that we are going to offer to the players.

5.2.1.3. PreGame Component Processing Detail

This component itself does not have an important algorithmic implementation. The important points of the implementation are handling the player choices and make the necessary adjustments accordingly. Also, if a previously saved game is going to be loaded, the correct flow of the game is crucial. Correct parsing of the saved game file is important.

5.2.1.4. Dynamic Behavior of PreGame Component

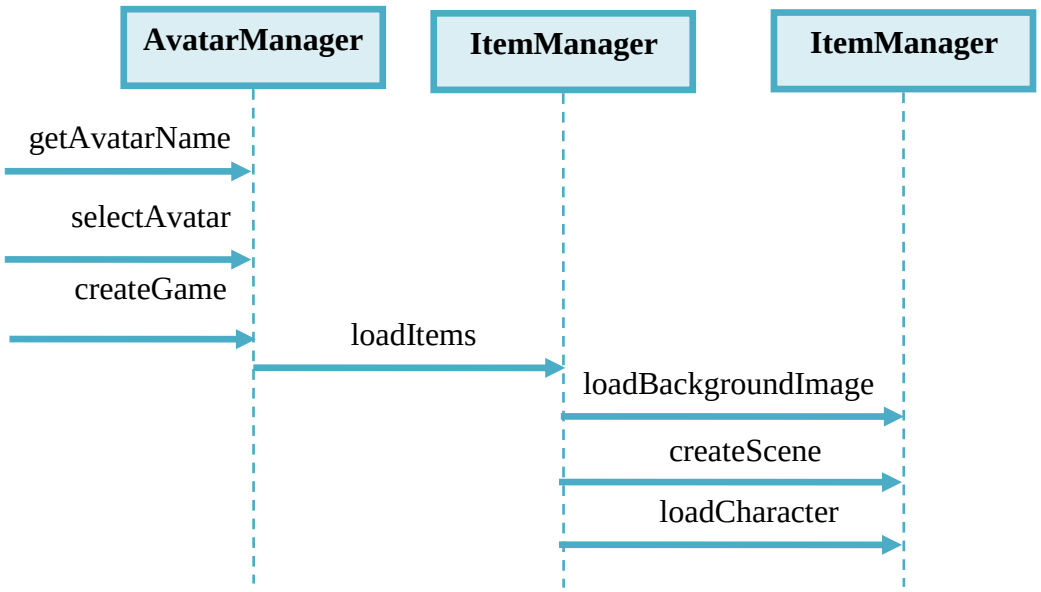


Figure-12 Sequence diagram of PreGame

5.2.2. Game Component

This is the topmost layer of the system. All components will be combined into this component to make users play the game we created.

5.2.2.1. Processing narrative for Game component

This subsystem is where the actual game playing takes place; however we find it useful to divide this subsystem into two subcategories, which are namely levelTransition and gamePlaying. In the levelTransition, transition between levels and parts within the levels will be done smoothly, related actions will be taken, and system will be prepared from the new level or part. On the other hand, in the gamePlaying stage, the player will be more effective on deciding which actions are going to be applied as he/she decides what to do while playing game.

5.2.2.2. Game Component Interface Description

When the player completes a part, a video will be shown to him/her. That way, the transition between the parts will be applied and the system will be able to be prepared for the next one. Also, by this video shown, player will be able to follow the story behind the game.

In the gamePlaying stage, user will have two controller buttons, as he/she touches those character on the screen will act accordingly, and user will be able to complete the game. Also map interface will be supplied to the player in order to make him be able to keep track how much he/she accomplished in the game so far.

5.2.2.3. Game Component Processing Detail

The algorithms in this stage will be implemented depending on the problems which player will have to tackle with. There will be a control mechanism to decide whether the player did the right thing to solve the problem in the current part of the level. So for each level, there will a different algorithm implemented.

5.2.2.4. Dynamic Behavior of Game Component

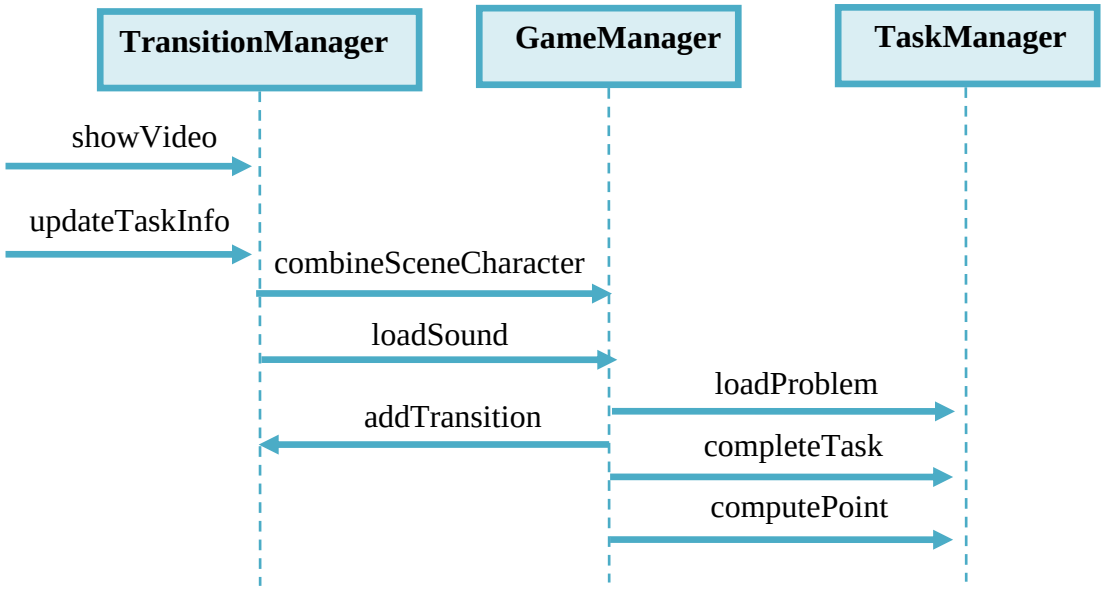


Figure-13 Sequence diagram of Game

5.2.3. PostGame Component

5.2.3.1. Processing narrative for PostGame component

This is the subsystem which is responsible from the actions that player wants to take after finishing the whole game or save the game for a future time. This will be enabled by some methods, such saveGame, finishGame, computeScore, showResults etc.

5.2.3.2. PostGame Component Interface Description

If the player interrupts the game and wants to save the game for a future use, he/she will have to click a button which is placed on the main game screen. If he/she does so, he/she will be redirected to a menu similar to the one in the beginning. By the touch on the screen, game will be saved. If he/she wants to return the game and continue, this will be possible as well.

If the player completes the whole game, after watching the ending video, he/she will be supplied an interface which he/she can see his/her point and how well he/she did during the game. After that, whether he/she wants to start a new game will be asked on the screen. Later on, necessary actions will be taken accordingly.

5.2.1.3. PostGame Component Processing Detail

All the performance of the player will be kept in some format throughout the game, so when he/she finishes the game, his/her point will be calculated based on those information.

The weight of the levels will be different from each other and they will be determined later on according to the difficulties of the levels.

5.2.1.4. Dynamic Behavior of PostGame Component

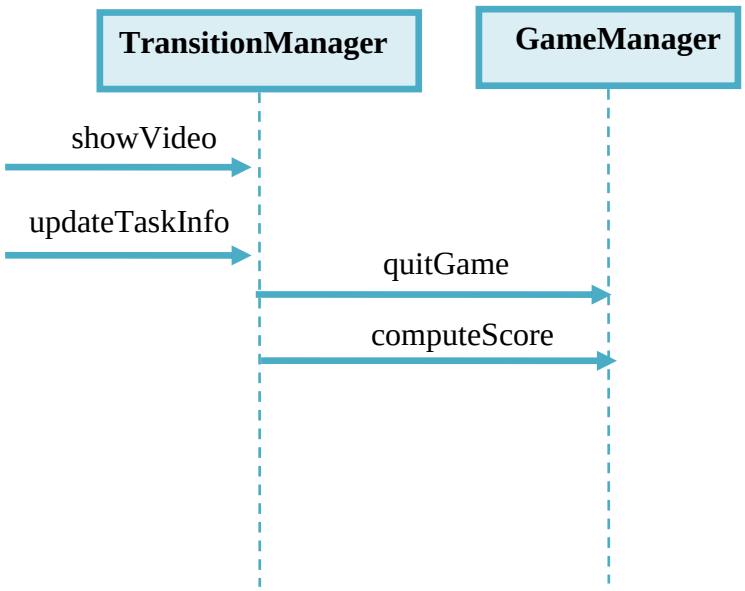


Figure-14 Sequence diagram of PostGame

5.3. Design Rationale

We chose to divide our system into those subsystems, because we wanted separate the actions within and out of the games. That way, the implementation of the whole system will work in a more organized way since different group members can deal with different stages without depending on each other, since the actions in the game will have no effect the ones which are not. Although the actions which are taken before the game effects how the game and the character are created, after completing that task, there is not a dependency or relation between those actions and stages and vice versa.

6. USER INTERFACE DESIGN

6.1. Overview of User Interface

The user starts to use the game by first clicking to the game icon. After that a main menu appears on the screen that user has two choices whether creating a new game or continuing the game which is saved last time. If user selects continuing game button, he/she will directly start to play the game where he/she was in last time. If new game is chosen, user again has two options. He/she can choose an avatar or start to play game without choosing avatar. When user chooses to play the game without selecting an avatar, a default avatar will be provided by the game. User can also click the help button to be able to learn the actions that the avatar can make or exit the game without playing.

User will be play two level in order to finish the game. In each level two different tasks must be solved by the user in order to continue the game. Before reaching one of the tasks, user will collect some paper of clue that will help the player while solving the task. And also these clues will increase the trial numbers of the player. At the beginning of the game a default trial number will be given to the user. This number will be five. User will be able to try to solve a question until 5 trials if he/she did not collected any clue paper. However, if clue papers are collected, the number of trial will increases according the the number of clue papers. When the number of trials is decreased to zero, the game will be over. User can touch one of the clue papers which are situated at the top of the screen to read the information written. In any time of the game, user can touch to map button, which will be unique to each level. The map helps user to see how many tasks are there and in which year is the user is in. Then he/she can return to game and resume playing. The player aim is to save a scientist in each level. Therefore, to save them he/she needs a key. This key is given to the player if the first task of the level is solved. If the second task of a level is solved, player will use the key that is collected in the first task and save the scientist. If user wants to quit the game, he/she can touch the exit button which is again on the screen. At this stage, the system asks user to exit game with or without saving.

6.2. Screen Images

This part gives information about the screen appearance of our game. However, the images used to represent screens in this section are not the actual screen interfaces that will be used in the game. These are only some prototype of screens.

6.2.1. Main Menu Screen

The “Main Menu Screen” has five buttons called “Karakter Seç”, “Oyuna Başla”, “Oyuna Devam Et”, “Yardım” and “Çıkış”. If the player did not save the previously played game “Oyuna Devam Et” button will be displayed but the player will not be able to touch it. If not he/she can choose to continue the game or start a new game.

Figure-15 Main menu screen

6.2.2. Select Avatar Screen

“Select Avatar Screen” helps user to select an avatar. At the moment there will be two different avatars. One of is a girl and the other is a boy. The user should write a name that will be his/her character name for the game. After that user can only touch one of the avatars that he/she wants to play with and then touch the “Oyuna Başla” button. From the avatar page, the player starts to play the game.

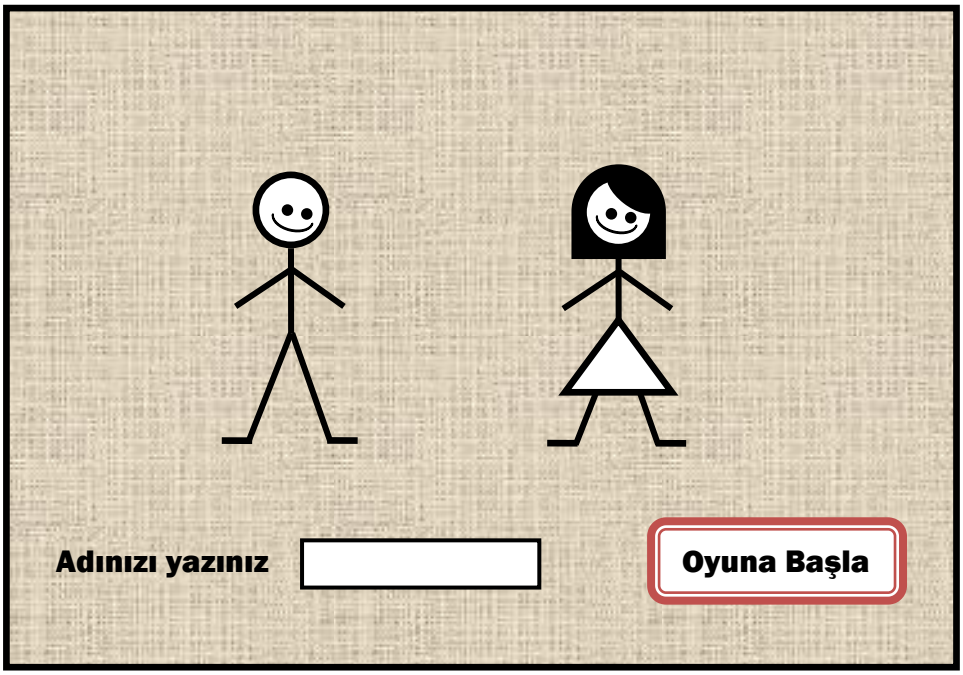


Figure-16 Select avatar screen

6.2.3. Help Screen

The “Help Screen” explains the user how to play the game. How to use buttons to move the avatar, how to collect the paper clues and keys, how to read the inside of a clue paper and how to use the key at the end of the task.

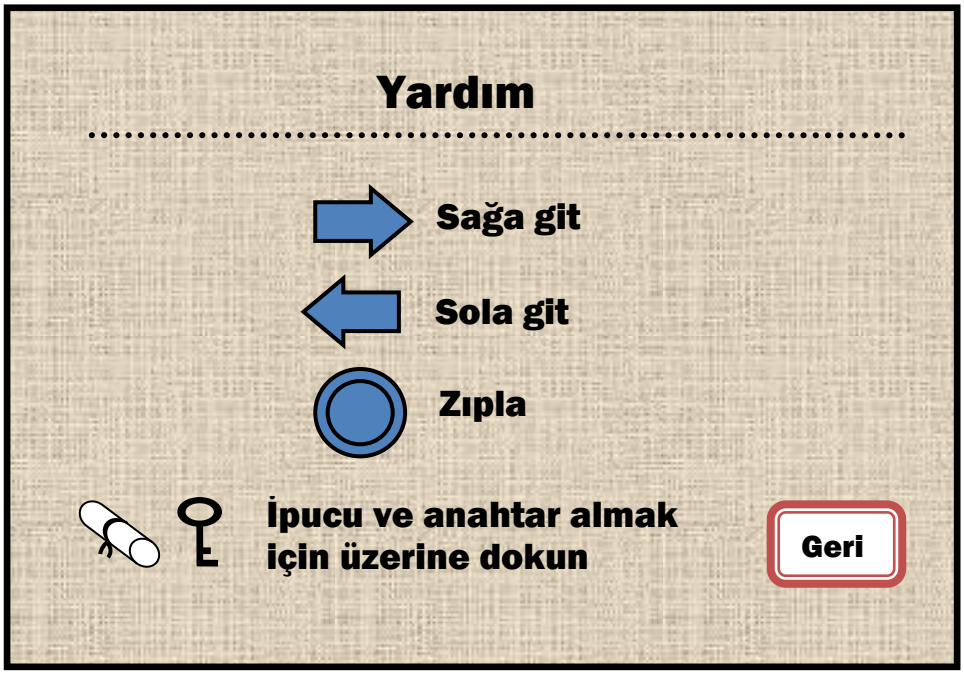


Figure-17 Help screen

6.2.4. Introductory video screen

"Introductory Video Screen" is displayed when the user wants to start the game. Before starting to play the game, this video will appear on the screen and will explain the story of the game to the player.

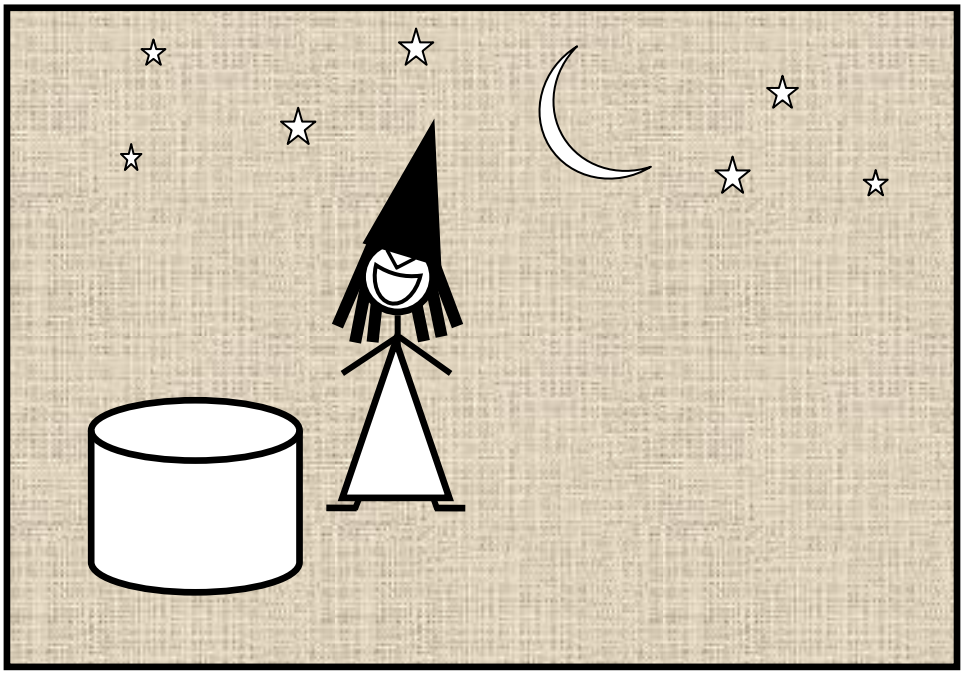


Figure-18 Introductory video screen

6.2.5. Level Screen

There are two levels in the game. The roots and the tasks that are going to be solved are different for each level. However, the places of the items on the screen are the same for both levels. Figure-19 and Figure-20 are the prototypes of two different task.

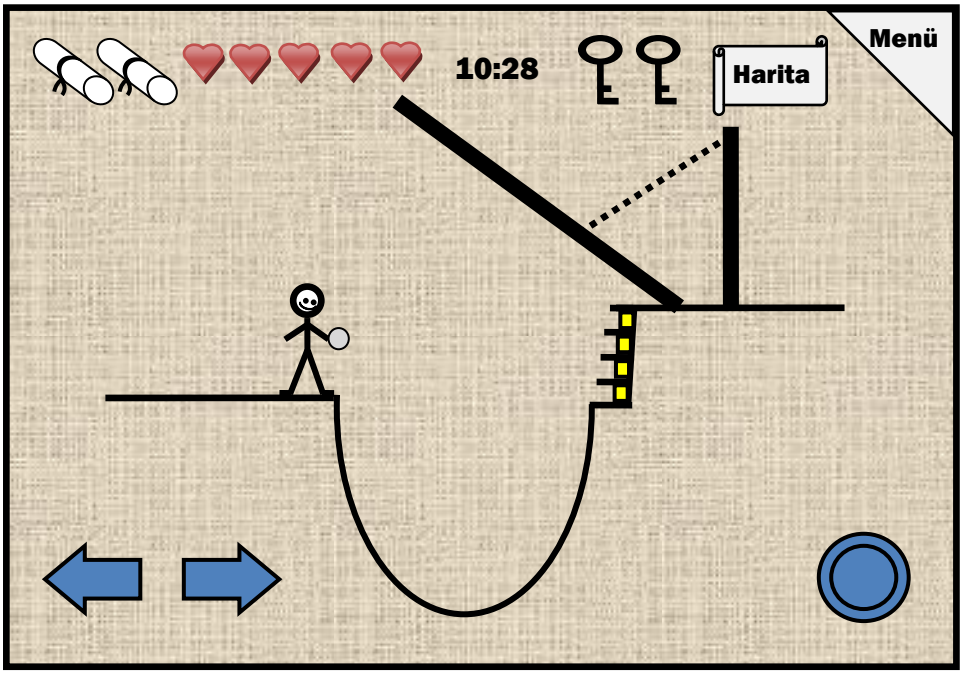


Figure-19 Level screen task - 1

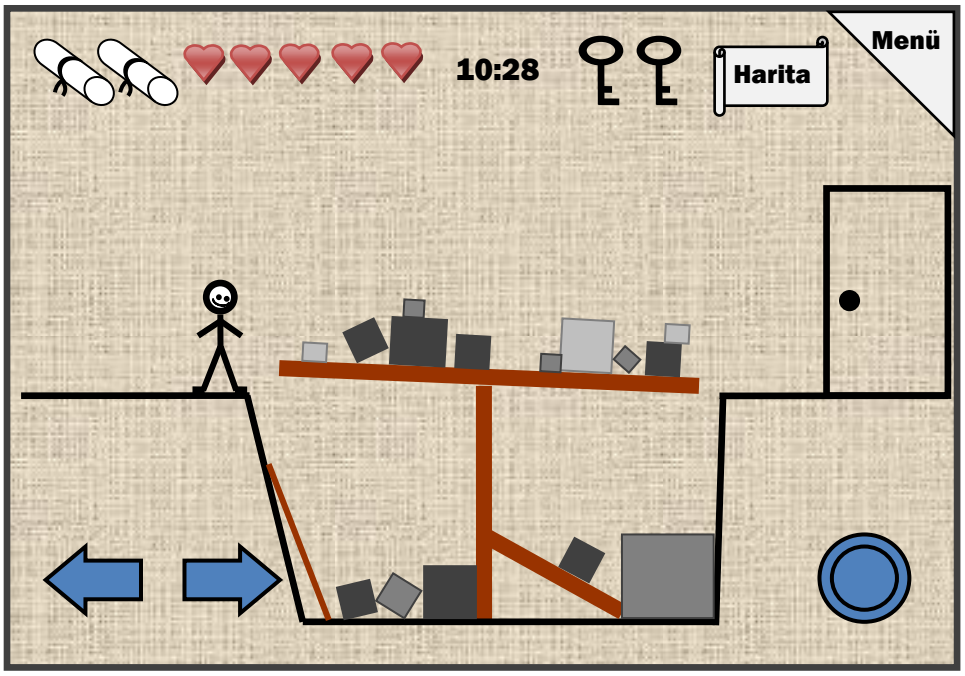


Figure-20 Level screen task - 2

At the top of the screen there will be icons that show the key and clue papers which are collected by the player during the game. If player wants to use one of these icons, he/she should touch on one of the icon. There are also “Menu” and “Harita” buttons. If the player touches the “Menu” a menu screen will be appeared. If the player touch the “Harita”, button a map will be appeared. There is also a counter that shows the time passed during the game and a number of trying rights which are represented with hearts in the screen. If

user gives wrong answers to the questions, the numbers of hearts will decreases. If user collects the clue paper in the game, he/she will get rights for each clue collected and the number of hearts will increase accordingly. The right, left arrows and the blue button allow user to move right and left and jump respectively. The figure shows a scene belonging to one of the task which will be in second level. There will be also other scene with different tasks during the game.

6.2.6. Main Menu Screen During the Game

There is a button called “Menu” on the “Level Screen” that helps player to exit game while playing or get some help. If player does not want to quit the game, he/she can touch the “Continue Game” button.

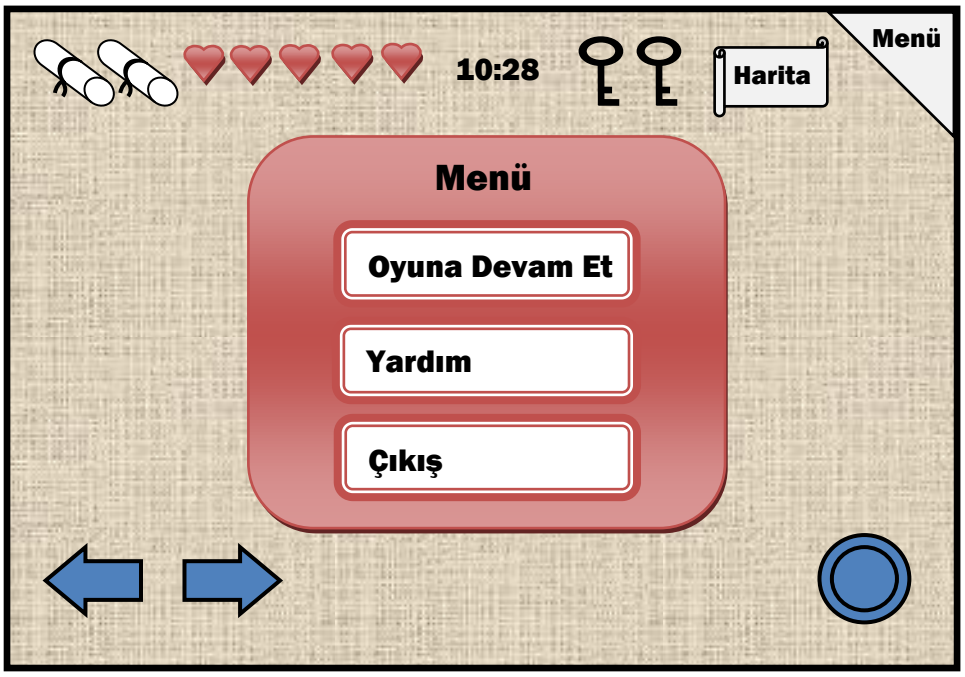


Figure-21 Main menu screen during the game

6.2.7. Map Screen

The player can see the “Map Screen” by touching the button called “Harita” in the “Level Screen”. “Map Screen” contains a map that shows the root that the player should follow during the game and remaining tasks. There is also a time line at the left of the map which indicates the year of the player is in.

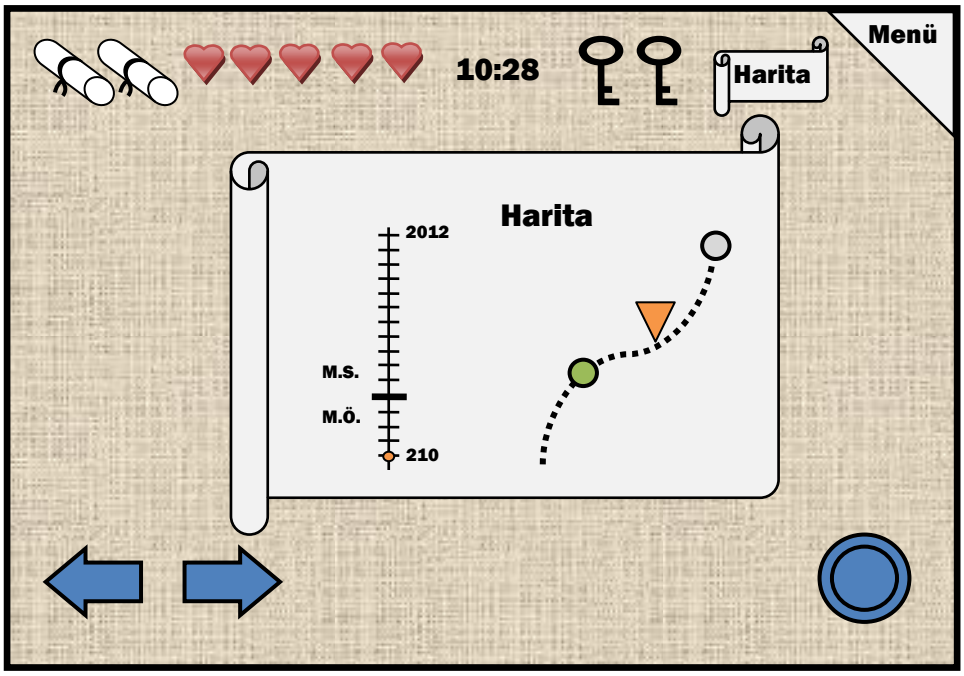


Figure-22 Map screen

6.2.8. Clue Screen

When a player is trying to solve a task in the game, he/she can get some help from the clue paper that are collected on the root. These clues give some information about how to solve a question. If the player wants to read what is written in the paper, he/she can touch one of the clue icons which are at the top of the “Level Screen”. After touching, a screen that shows the inside of the paper will appear on the screen and the player will be able to read it.

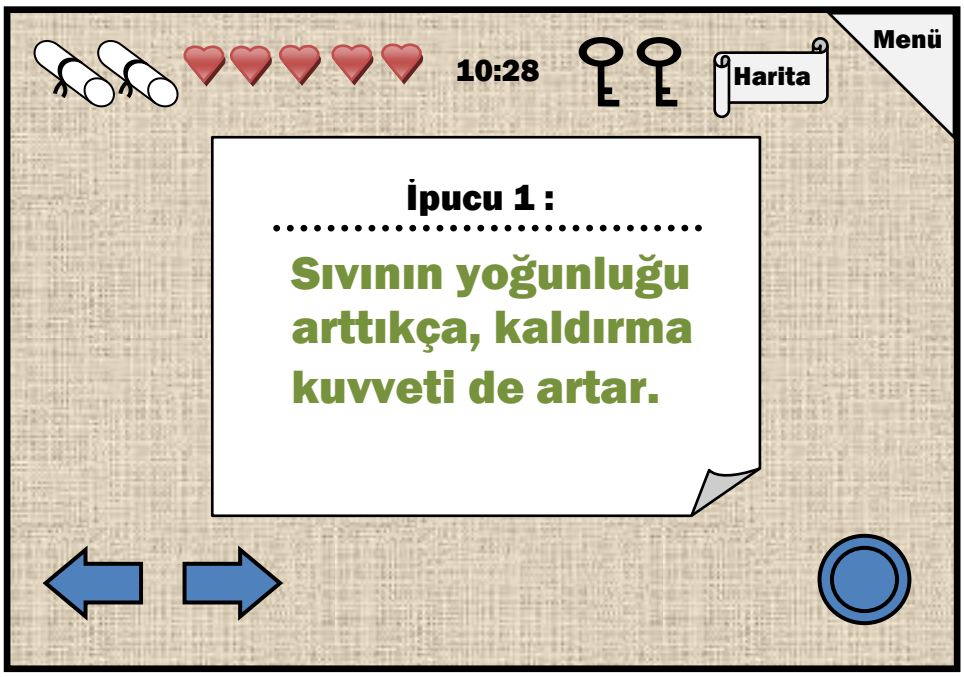


Figure-23 Clue screen

6.2.9. Level Video Screen

"Level Video Screen" is shown between the two levels. When the player completes the first level, there will be a very short video that shows the player that he/she has completed the level successfully. This video will give the player the impression that the world is developed more when a scientist is saved. Moreover, the time that the player is spent in that level will be appeared.

6.2.10. End Video Screen

"End Video Screen" is displayed when the whole game is completed by the player. This video gives impression to the player that he made something very useful and also to give some courage. Moreover, the time that the player is spent to complete the game will be appeared.

6.3. Screen Objects and Actions

The player first sees the "Main Menu Screen" in which there are several choices. Player can select an avatar, take some help about the game, start the game without selecting an avatar or continue the previously saved game. When player touches the "Avatar Seç" button, he/she will be directed to the avatar selecting screen. After selecting an avatar the user touches the "Oyuna başla" button and an introductory video will appear on the screen that explains the story of the game. After introductory video, the player starts playing the game from the first level. After completing the first level, a video will appear to show the player that the scientist is saved and the word become more developed and the time passed to complete the first level will be shown. Then the player will start to play the second level. When the second level is completed, an ending video will appear to give the player the impression that the world is saved and his/her total amount of time spent to play the both levels.

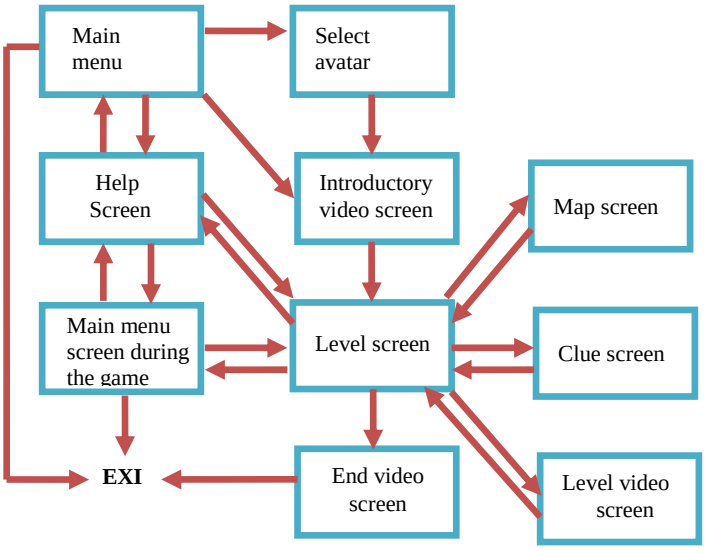


Figure-24 Game screen flowchart

7. LIBRARIES AND TOOLS

7.1. Libraries

Only thing we are going to use for this project is Unity 3D, hence we are not going to use any external libraries. But still, there will be some libraries of the unity which we are going to use:

1. Character Controller
2. Light Flares
3. Particles
4. Physics Material
5. Projectors
6. Scripts
7. Standard Assets
8. Toon Shading
9. Water

7.2. Tools

Unity: The ultimate tool for video game development, architectural visualizations, and interactive media installations – publish to the web, Windows, OS X, Wii, Android etc.

8. TIME PLANNING

By the end of the first semester, we are planning to finish the menu and the first task of the game, namely buoyancy of the liquids, which we will also present in the demo presentation. In the second semester, we are planning to finish the implementation of the rest of the game. We will mainly implement other three tasks and do system testing.

8.1. Term 1 Gantt Chart

Task	Assigned to	Start	End	Oct	Nov	Dec	Jan
Play with physics	Big Ceng Theory	21.10.2011	09.06.2012				
Analyzing existing games' stories, characters, inspiration elements, experiences, music, interactions with other fields.	All members	21.10.2011	18.11.2011				
Workshop with Words To Inspire, Ministry of National Education of Turkey and Game Developers	All members	19.11.2011	19.11.2011				
Determining physics subjects	Allmembers	20.11.2011	25.11.2011				
Determining the story of game	All members	20.11.2011	25.11.2011				
Determining the music of the game	All members	23.11.2011	03.12.2011				
Updating design and implementation reports 1	All members	03.12.2011	06.12.2011				
Designing graphics of the game	All members	03.12.2011	25.12.2011				
Designing the characters of the game	All members	03.12.2011	25.12.2011				
Implementation of the story of step 1	All members	26.12.2011	15.01.2012				
Testing step 1	All members	16.01.2012	22.01.2012				

Figure-25 Gantt Chart of term 1

8.2. Term 2 Gantt Chart

Task	Assigned to	Start	End	Feb	Mar	Apr	May	June
Implementation of the story of step 2	All members	13.02.2011	05.03.2012					
Testing step 2	All members	06.03.2012	13.03.2012					
Implementation of the story of step 3	All members	06.03.2012	31.03.2012					
Testing step 3	All members	01.04.2012	08.04.2012					
Implementation of the story of step 4	All members	01.04.2012	30.04.2012					
Testing	All members	01.05.2012	07.05.2012					
Implementation to fix problems in the test step	All members	01.05.2012	13.05.2012					
Quality Ensurance	All members	13.05.2012	20.05.2012					

Figure-26 Gantt Chart of term 2

9. CONCLUSION

This IDD is prepared to give brief information for the all design patterns of Fizbot. First, design constraints, assumptions and dependences introduced. Second data models are given with their descriptions. Then, system architecture is provided with all of its components. In the following sections, user interfaces and actions of objects are stated and the libraries and tools which will be used in the project are introduced. Finally, we provide 2 Gantt charts to demonstrate the detailed time planning we determined for two terms.

This document will be basis for the detailed design document and also will be very helpful when implementing the requirements of the project.