Requirements and analysis document for Treasure Pleasure.

Team: Skyriders
Date: 2018-10-16
Version: 1.0
Members: David Weber Fors, Felix Rosén, Jesper Naarttijärvi,
John Gidskehaug Lindström and Oskar Lyrstrand

1. Introduction

1.1. Purpose of the application

Excitement! Adventure! Treasure! This app will turn your dreary everyday stroll into an exciting adventure! The adventurous user is plunged into an exciting world where treasure is never far away. But be vigilant, else other shady characters may snatch your price from under your nose. Will you be the most famous adventurer in the land, achieving wealth and glory beyond imagination? Only the boldest user should apply.

Encourage physical activity through an competitive and enjoyable gaming experience.

1.1.1. Who will it benefit

Extraordinary gentlemen of exquisite taste. We believe the game can benefit anyone who enjoys a casual gaming experience while at the same time being outdoors. However the game will maybe mostly benefit those who aren't very physically active in their day to day life because of lack of motivation.

1.1.2. General description

TreasurePleasure is a competitive mobile multiplayer game for Android devices. It will use the users phone GPS to locate him/her on a shared **map** where multiple **players** compete with each other by collecting the most **items**.

Each **player** has a **backpack** with a maximum capacity, when the capacity is full the backpack must be emptied at the players **chest** before more things can be collected. Whoever has the most value contained in their **chest** at the end of a **season** wins that season and will appear on the **hall of fame.** However, there’s a catch. A player can choose to offer some or all of their hard earned score to increase different aspects of the game, e.g. slots in their backpack. A player must evaluate if a upgrade might increase their overall score before the end of the season.

1.1.3. In what situations can the application be used?

The application can be used to encourage a group of people to become more active by adding some flavour to their physical activity. It can also be used more casually by anyone who want to explore and compete within a virtual reality in their spare time.

1.2. Wordlist

- The **Map** is the main view of the game. We append the players location on a map served by google. From the map a player can interact with items, stores and settings

- A **Player** is one of several players on a map. Players are unique to each user. Players also hold a backpack and a treasure chest, where they can store items collected from different locations on the map.
- **Collectables** are randomly spawned items on the map. Collectables can be interacted with by players, and collected if the player is close enough to the collectable. These items are shared for all players so the one that picks an item up first gets it, and the collectable is removed from the global map. There is always a certain amount of collectables available, when one is collected a new collectable is randomly spawned. However a player increase this amount.
- A **Item** is a valuable in the game. There are different items and each item has a value and a unique type/id. The type can for example be diamond, stone and gold. Items are spawned according to their value, e.g. a diamond will spawn less frequent than wood.
- A **Backpack** is a local inventory that every player carries with them. Backpacks are unique to each player and can only carry a limited amount of items in them. When a players backpack is full the player has to empty it to his/hers treasure chest. A player can sacrifice some of their score to increase this amount.
- A **Treasure Chest** is stationary storage unit with unlimited size. A player has one unique treasure chest that the player chooses where to place when first starting the game[1]. When a player empty his/hers backpack into the treasure chest a total value is calculated and displayed.
- A **Avatar** is an image representing the player. A player has the opportunity to chose his avatar in game.
- A **Store** is a way for the player to buy products to amplify their stats. An example product is “The value multiplier”, this product increases the value of collected items from the map. A store is placed on the map and a player has to be close enough to interact with it.
- A **Score** is a players unique player score for the current season. The score is calculated using collected items value after the player has inserted them to his/hers treasure chest.
- A **Location** is real world location, represented by coordinates in latitude and longitude.
- A **Collect**: is an event where a Player picks up Collectable on the Map and puts in his or her backpack.

1.2.1. Future implementations

- **Settings** - Here one can access the highscore or change name and avatar.
- **Hall of Fame** - A list or high score of the best performing players.
- **Season** is a limited duration that a player can collect items and increments his/hers score. The season is a global season for all players. The player with the highest individual score after a season has ended won.

[1] Not yet implemented. Location is fixed given map constraints

2. Requirements

2.1. User Stories

Description: As a player I want to see a map pinpointing my location so that I get a sense of belonging in the game
Acceptance Criterias:

- A map should show a representation of the real world (streets, buildings and locations)
- Map is constrained to specific area

Description: As a player, I want objects to have a location; so that I know where to find them.
Acceptance criterias:

- Add location to Chest
- Add location to Upgrade Center
- Add hashmap with <item,location> in TreasurePleasure model

Description: As a player I want to see different items on the map so that I know what I can collect.
Acceptance criterias:

- There is always a certain amount of items on the map
- Items are spread out rather than clustered
- Items has value and type
- There are at least 3 items differing in appearance and value

Description: As a player, I want to a backpack so that I can collect items and plan my walk.
Acceptance criterias:

- The backpack holds an integer representing maximum capacity (number of items).
- Backpack has slots according to capacity.
- The backpack must display the contents
- Backpack has a function to erase all items when called

Description: As a player, I want to be able to pick up items that I'm near; because items are valuable to me.
Acceptance criterias:

- Have a function that can compare two distances
- The item must be within reaching distance for the player to pick it up
- Cannot add item to full backpack
- If item is added to backpack, it must be removed from the map
- if item is not added to backpack, it must remain on the map
- When an item is removed from the map a new item appears

Description: As a player, I want a storage chest, with unlimited capacity, to unload my backpack contents into. - So that I can stash all my valuables and get a higher score.
Acceptance criterias:

- Chest has unlimited capacity
- The chest can display the total value of contained items
- There must be a certain distance between the chest and the store
- The player must be close to the chest to be able to use it
- After function call: all items that were in the backpack must be in the chest.
- The player should be notified somehow that the backpack has been emptied.

Description: As a player, I want to be able to improve my character attributes. This will add a sense of progress to my game experience and forces me to make choice between my score right now and score at the end of the game.
Acceptance criterias:

- Player has different attribute that changes how the player can interact with the game.
- When I buy products that changes my attributes, I want my game experience to change in some way.
- There must be a certain distance between the chest and the upgrade center
- The store appears on the map
- When close enough player can interact with store
- Ability to choose items from backpack and trade them for Value multiplier (e.g. drop bonus increases by 5/100*(value of item))

Description: As a player I need a player class so that I can get a backpack, chest and upgrade center.

- Implement a class Player holding a Backpack, Chest and Upgrade Center
- All calls on functions in Backpack, Chest and Upgrade Center goes through Player

Description: As a player I want to have a choosable nick and an avatar so that I get a personalized experience.
Acceptance criterias:

- Can choose nick and avatar on first time playing game
- Player cannot take already taken nick
- Can change nick and avatar when in game
- Information about current nick and avatar is available

2.1.1. Not yet implemented

Description: As a new player I want the game to have seasons with certain duration so that a winner is determined and everything resets and gives equal opportunities to all.

- Implement a Class Season
- Season has an attribute integer or Date that defines the length or end of a season

Description: As a player I want to be able to see my and others current score to see how well I'm doing.
Acceptance criterias:

- Implement function returning the 5 players (in order) with highest value in their chest. (highscore)

Description: As a player I want to see the history of who has won previous season in case I am on that scoreboard.
Acceptance criterias:
- Implement function returning the current leader of the season (winner)

Description: As a developer I want to be able to play the game without needing to moving around, this is because I want to be able to test the game on a emulator while developing the game.
Acceptance criterias:
- Has a to toggle debug mode/god mode
- In debug mode, there's a way to start with a score so I can buy items
- In debug mode, I want to be able to pick up items even if I'm not near enough

Description: As a developer I want a data class so that i easily can change the global variables so that I can edit the pre-decided game values fast
Acceptance criterias:
- Implement a data class.
- Implement each piece of data as static so that this class don't need to be instantiated.

2.2. User interface

The main view is a map where various objects are displayed. The user of the application can interact with these objects if they are close enough, if not then they have to move to get closer to them. If e.g. a player is close enough to an item that is on the map the user can click on this item to collect it. The user can also interact with his or her chest and store, where items can be dropped of or traded for increased drop rate.

Map

The foreground

Player

- Located by gps coordinates
- Has luck, the higher luck the more valuable are picked up items

Backpack

Picked up items goes here. Has a maximum capacity but can be emptied at Treasure Chest.

Fortune Shop

Trade items for luck

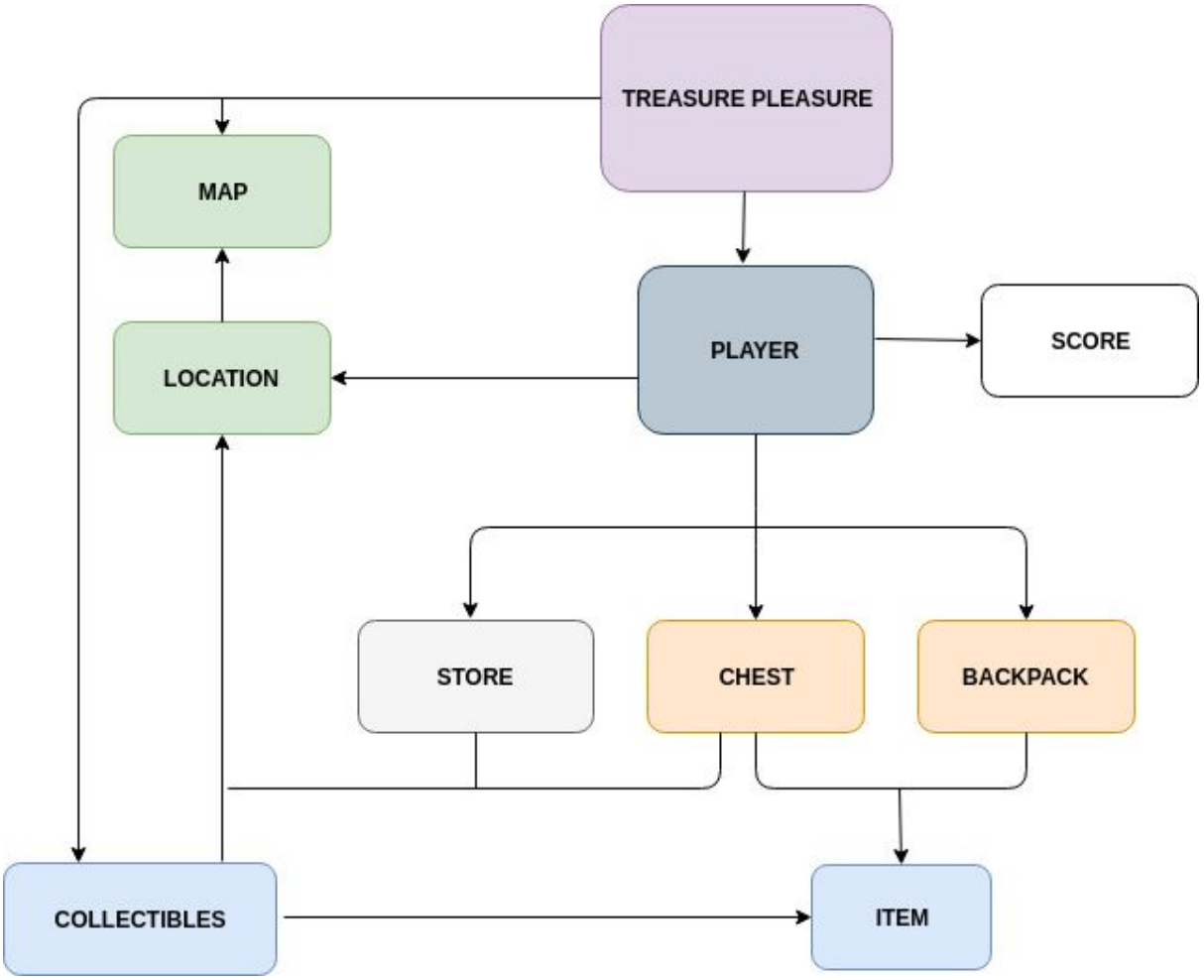
Items

Has value, weight and can be picked up by players

Treasure Chest

Contains items that has been dropped of by Player. Total item value counts toward Player Score.

3. Domain model



3.1. Class responsibilities

Each class has responsibilities. Below follows a short description of the different classes in the domain model.

Treasure Pleasure - is the hub of the model. Handles and redirects information. Collaborates with Player, Location, Backpack and Collectables to handle Collects. Stores user name and handles name update. In addition it keeps track of players.

Player - keeps track of player related properties. A player has a username, an avatar, a chest, a Value multiplier.

Location - Collectibles, Treasure Chest and Player have coordinates in the real world. In the app, players and collectibles are represented as icons on a map. As the user walks around in the real world, his coordinates are updated accordingly. Handles logic to check if a player is close enough to an item to collect it.

Collectables - is responsible for keeping track of Location for items populating the game. Can spawn new randomized items and assign the item a random location.

Backpack - is responsible to keep track of any items currently carried by a player. A backpack has room for a limited number of items. A backpack can be improved to hold more items.

Chest - keeps track of current score. Has a Location.

Item - keeps track of Item type and value.

Store - provides a place for a player to improve Value multiplier. Has a Location. Yet to be implemented.

Score - not in use. Is currently represented by a variable in Chest.

Season - responsible for handling Season. Yet to be implemented.