King’s Tasks Simulation Game

Software Design Document

Lenell White

Date 3/10/2023

INTRODUCTION:

This Software Design Document (SDD) describes the functionality and software design of a task management simulation game, King's Tasks. In the game, the user is assigned the role of a king whose job is gaining and managing his subjects. The objective is to keep the kingdom alive as long as possible. The player's score will be based on the number of days he can survive and the game ends when the kingdom's health drops to 0. Note: The term "king" will be used to refer to the human player, and the term "subject" will be used to refer to the autonomous agents doing the tasks.

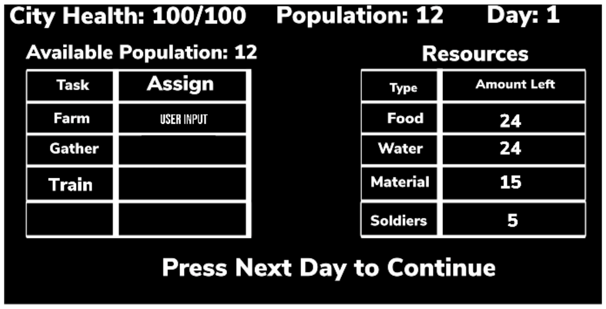
The overall point of this project is to introduce a task management game that displays personal knowledge of object-oriented programming. This has value because task management is widely used in industry and can potentially help train employees to develop management skills. This document provides a template, and guidance, for the simulation game.

This document contains two main sections:

1. Problem Description section describing the rules of the game as implemented.
2. Problem Solution section presenting a description of the King's Tasks software.

PROBLEM DESCRIPTION:

This project implements a text-based simulation game in which the user assigns autonomous agents, also to be known as subjects. The user is presented the values of his kingdom's health, kingdom population, day, tasks and resources as shown in Figure 1. Each "Day", the King will have to use these values to determine which tasks require the most importance.



City Health: 100/100 Population: 12 Day: 1

Available Population: 12

Task	Assign
Farm	USER INPUT
Gather	
Train	

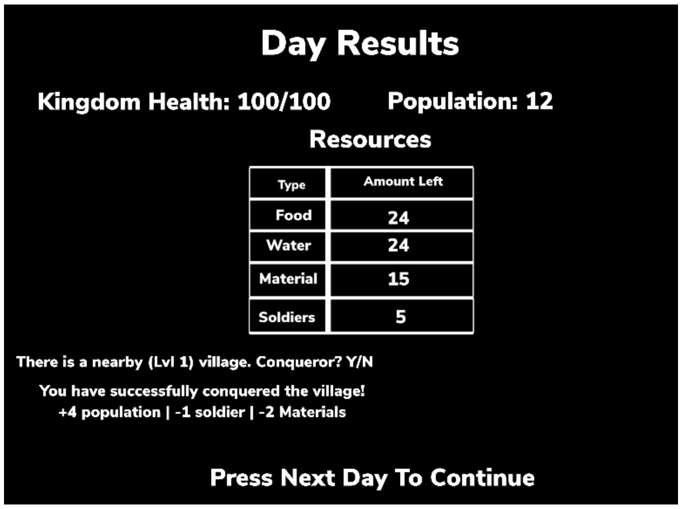
Resources

Type	Amount Left
Food	24
Water	24
Material	15
Soldiers	5

Press Next Day to Continue

Figure 1: King's Tasks User Interface

The kingdom after each day consumes resources (food, water, material) at the end of each day. The point of the task is to keep up with the consumption of materials as city health will fall if any are at 0. Figure 2 shows the events that occur after "Next Day" is selected. The "Day Results" interface show updated resources, city health, and population values. Within the "Day Results", are randomly generated text based events. These events will simulate real life events that can happen to a civilization to either benefit it or destroy it. Some events will be user decision based or be automatic. In Figure 2, we also see an example of a user-based decision event. After the Day Results, the cycle continue and the user goes on to the next day.



Day Results

Kingdom Health: 100/100 Population: 12

Resources

Type	Amount Left
Food	24
Water	24
Material	15
Soldiers	5

There is a nearby (Lvl 1) village. Conqueror? Y/N

You have successfully conquered the village!
+4 population | -1 soldier | -2 Materials

Press Next Day To Continue

Figure 2: Day Results Interface

Every 5 days, there will be a new task along with a new resource. The table below shows all the tasks and resources present in the game.

Task	Resource Produced	Resource Descriptions
Farm	Food	Feeds the total population. Used every day.
Gather	Water	Feeds the total population. Used every day
Train	Soldiers Influence	Aids in defending the kingdom in text-based events.
Study	Scholars Knowledge	Aids in defending the kingdom in text-based events.
Mine	Coal	Aids in defending the kingdom in text-based events. Used by population every day.
Fishing	Food	Feeds the total population. Used every day.
Study Healing	Medicine Doctors	Used by the population every day. Used in defending the kingdom in text-based events.
Entertain	Moral Population	Used by the population every day. Used in defending the kingdom in text-based events

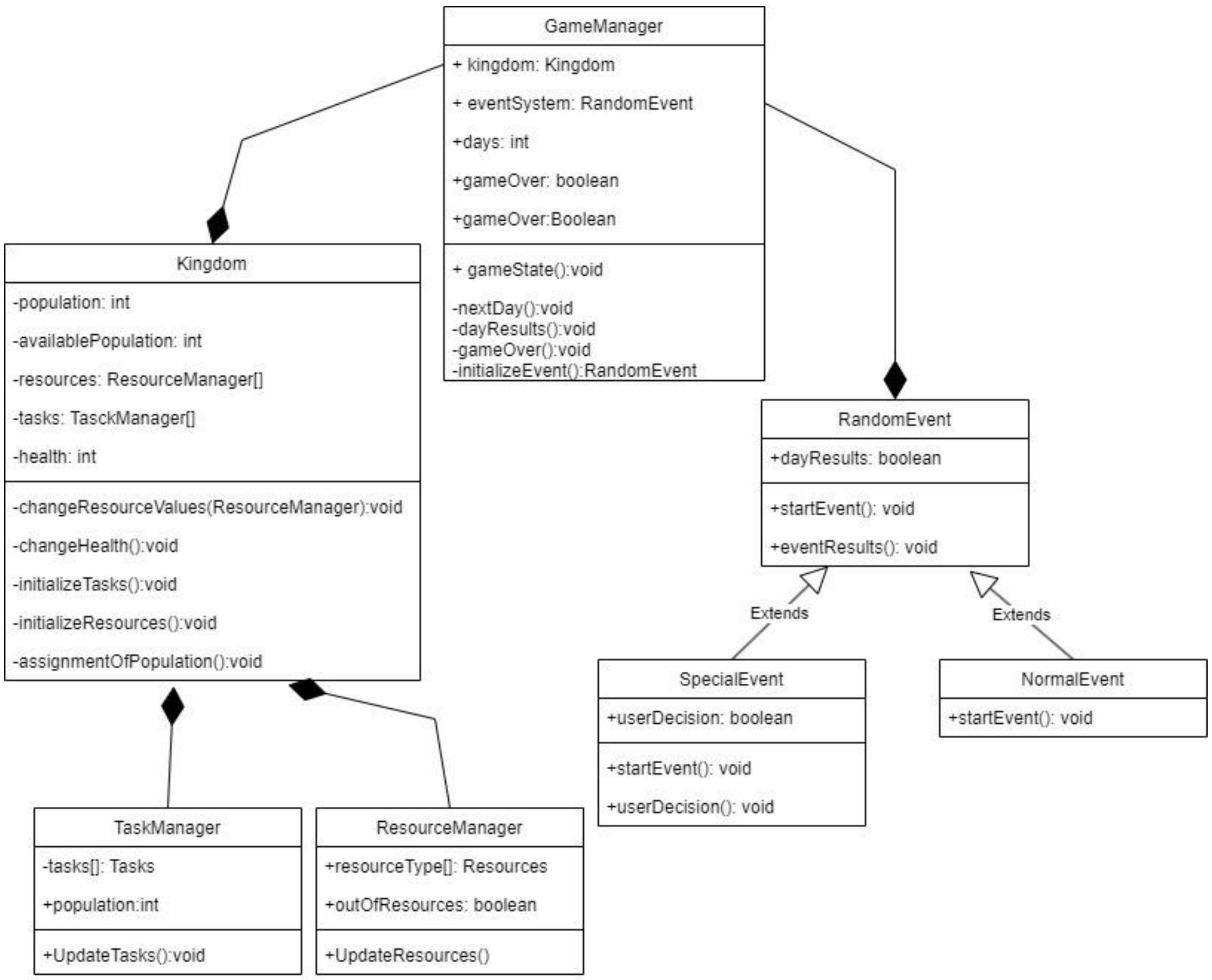
The rules of the game are provided below (Note: Specific numerical values may be adjusted for game play once play testing begins.):

1. King's Tasks objective is to keep the kingdom alive as long as possible.
2. The kingdom has a health of 100 out of 100.
3. The kingdom starts out with 10 population.
4. The kingdom starts out with The tasks of Farm, Gather, and Train.
5. The kingdom starts out with 20 food, 20 water, and 2 soldiers. (Likely to change)
6. During every "Day", the King assigns subjects to tasks based on the "Available Population".
7. If any resource is at 0, the Kingdom takes (-10) damage to its health.
8. The number of subjects assigned to a task cannot exceed the maximum "Available Population" value or be below 0.
9. Once a subject is placed in an activity, they cannot do anything else. (Likely to change)
10. A day shall end when the player selects "Next Day".
11. After each day, the following resources shall be used up:
    - Food and Water.
    - Coal, Medicine and Moral (Only when a certain day is reached)
    - Note: 1 subject consumes 3 food, 2 water, 2 Coal, 1 Moral. Medicine is used by ±¼ of the total population.
    - Note: Resources can be consumed through text-based events
12. The following tasks shall produce:
    - 1 Farm task = +2 Food
    - 1 Gather = +2 Water
    - 1 Train = +1 Soldiers & + 2 Influence
    - 1 Study = +3 Knowledge & +1 Scholar
    - 1 Mine = +3 Coal
    - 1 Fishing = +3 Food
    - 1 Study Healing = +2 Medicine & +1 Doctor
    - 1 Entertain = +1 Population & +2 Moral
13. Text-based events will either be automatically applied or Yes/No questions off user input.
14. Text-based events will be random, but still only use up or supply resources already available.
15. A new task and new resource associated with that task is added every 5 days.
16. Every two days a random text-based event will appear in the Day Results interface.

PROBLEM SOLUTION:

The overall flow of action and information for the game is listed below.

UML DIAGRAM



GameManager Class

This class runs the King's Tasks Game. It keeps track of days, creates the Kingdom and RandomEvent classes, and keeps track of the game status. Game statuses are game over, current day, and day results. This class is responsible for the following actions:

1. Creates an instance of the Kingdom and RandomEvent class.
2. Game Status between "Next Day", "Day Results", and Game Over

Kingdom Class

This class initializes the game by creating an array of tasks and resources through file input. It creates the tasks and requests input from the user until the user presses “Next Day”. This class is responsible for the following actions:

- Read from a file using the Random Event class or children of that class.
- Create an instance of the Task Manager
- Create an instance of the Resources.
- Keep track of health and population attributes.

Task Manager Class

Creates a table and reads from the File “Tasks”.

Resources Class

Creates a table and reads from the File “Resources”.

Random Event Class

During the “Day Results” phase, this is used to initialize events that happen to the kingdom. Events will take in player input if it is a special event.

Special Event

This class extends the Random Event Class. It will generate scenarios where the user will input yes or no. The output will be a result of the applied event.

Normal Event

This class extends the Random Event Class. It will generate scenarios with a predetermined outcome.

REFERENCES:

APPENDICES: