trAIner

An immersive artificial intelligence game for everybody!

Lucas Mahler, Patrick Gautheret, Kasparas Gudzius, Rahul Tak, Oleksandr Shlapak

University of Applied Sciences Ulm

January 24, 2019

Motivation

"We were supposed to make AI do all the work and we play games but we do all the work and the AI is playing games!"

-Andrey Karpathy

Outline

What is trAIner?

A single, cross platform game experience:

- Implementing the concept of Artificial Intelligence in a Java based application.
- trAIner is an absorbing, immersive game experience
- Lets the player develop and train theirown AI
- Lets the user build his own maps

Members:

On Github: github.com/tr-AI-ner/trAIner

- Patrick Gautheret: ElectrifyPowr
- Lucas Mahler: Lugges991
- Kasparas Gudzius: kasparasGud
- Rahul Tak: takrahul
- Oleksandr Shlapak: oleksandrshlapak

Targeted Audience

- Gaming Lovers
- Tech Enthusiasts
- AI Geeks

3 Different Modes:

- Play the Game!
- Build Your Own Maps!
- Let the AI Play!

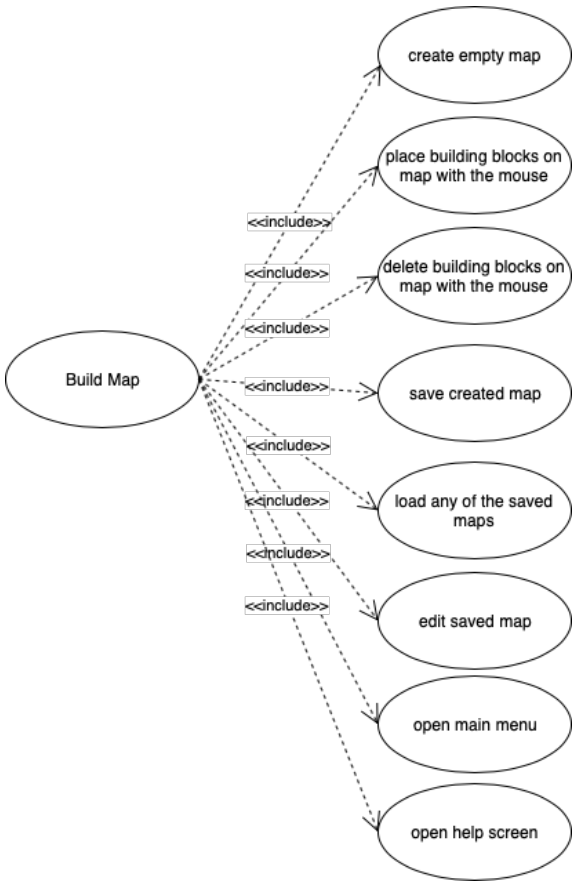
Builder Mode:

- Place/Delete building blocks
- Make use of enemies with different behavior
- Save & Load your own creations

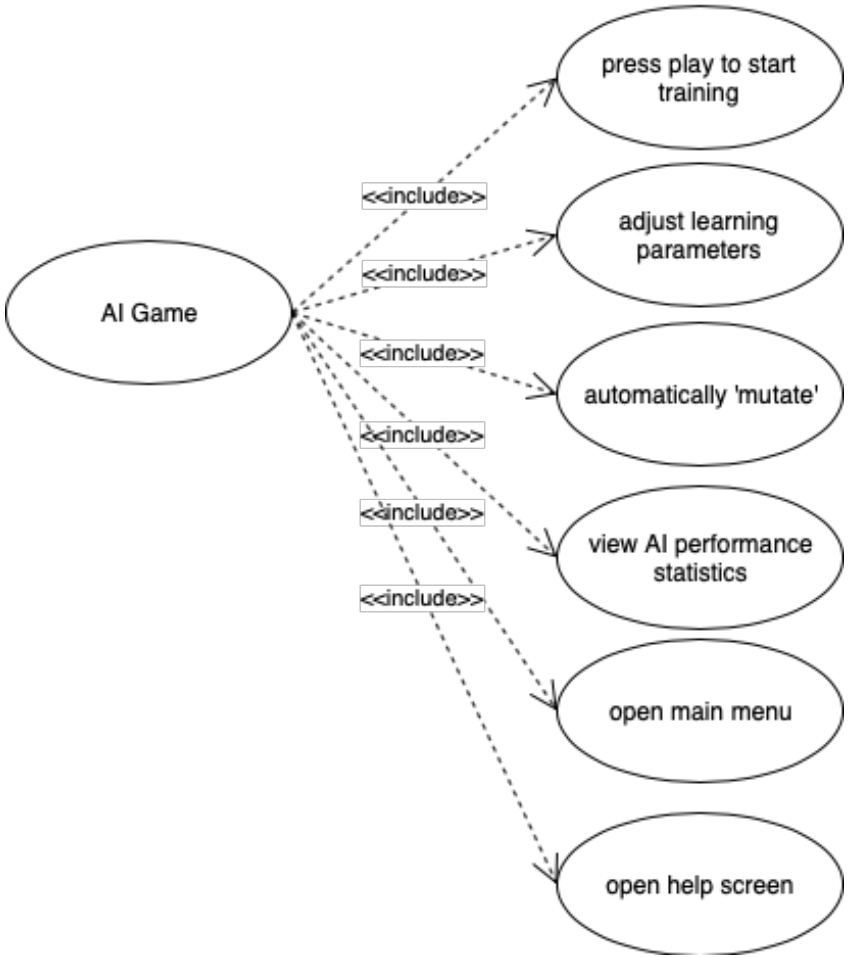
Player Mode:

- Play the default
- Play on your own maps
- Test your might on the challenges

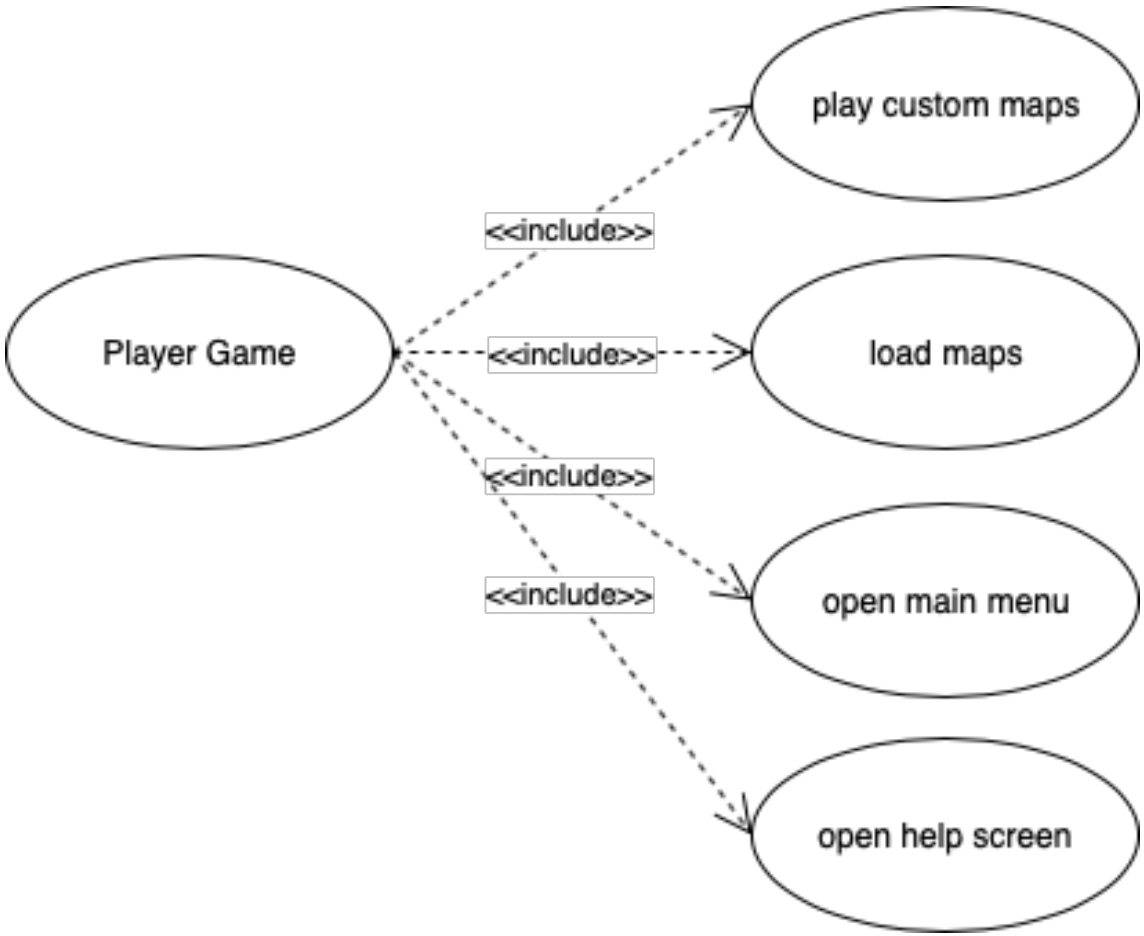
Use Cases


create empty map
place building blocks on map with the mouse
delete building blocks on map with the mouse
save created map
load any of the saved maps
edit saved map
open main menu
open help screen
Build Map
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>

Use Cases


AI Game
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>
press play to start training
adjust learning parameters
automatically 'mutate'
view AI performance statistics
open main menu
open help screen

Use Cases


play custom maps
<<include>>
Player Game
<<include>>
load maps
<<include>>
open main menu
<<include>>
open help screen

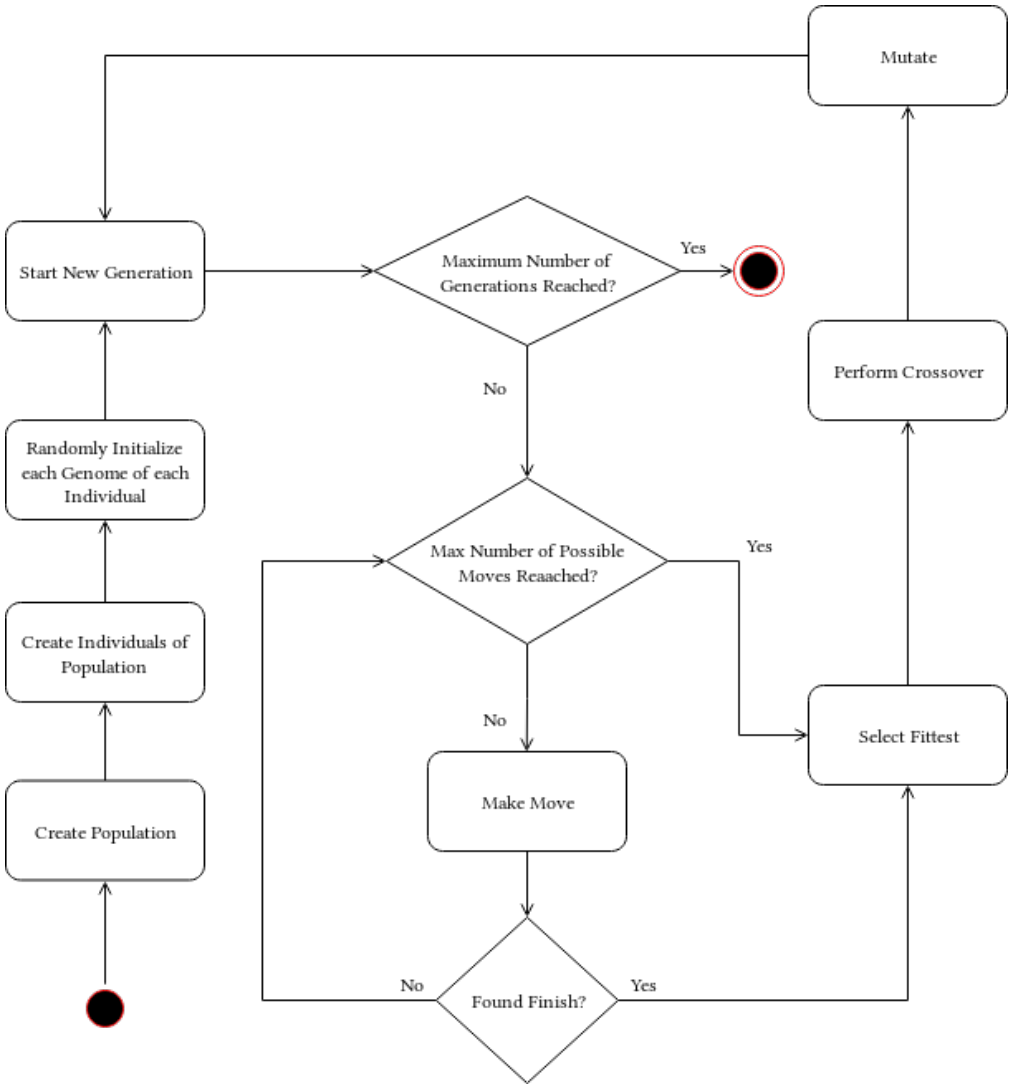
Tools and Technology

- Java
- Genetic Algorithms
- Local Storage

Genetic Algorithm

- Darwin as inspiration
- Natural concepts of survival of the fittest
- Implementation of selection, crossover and mutation

How Does the GA Work?



Functional & Non-Functional Requirements

Technical Constraints:

Programming Language	Java
Supported Platform	All platforms supporting Java
Budget	750 hours
Scope	A.I. based system
Schedule	04.10.2018 - 24.01.2019

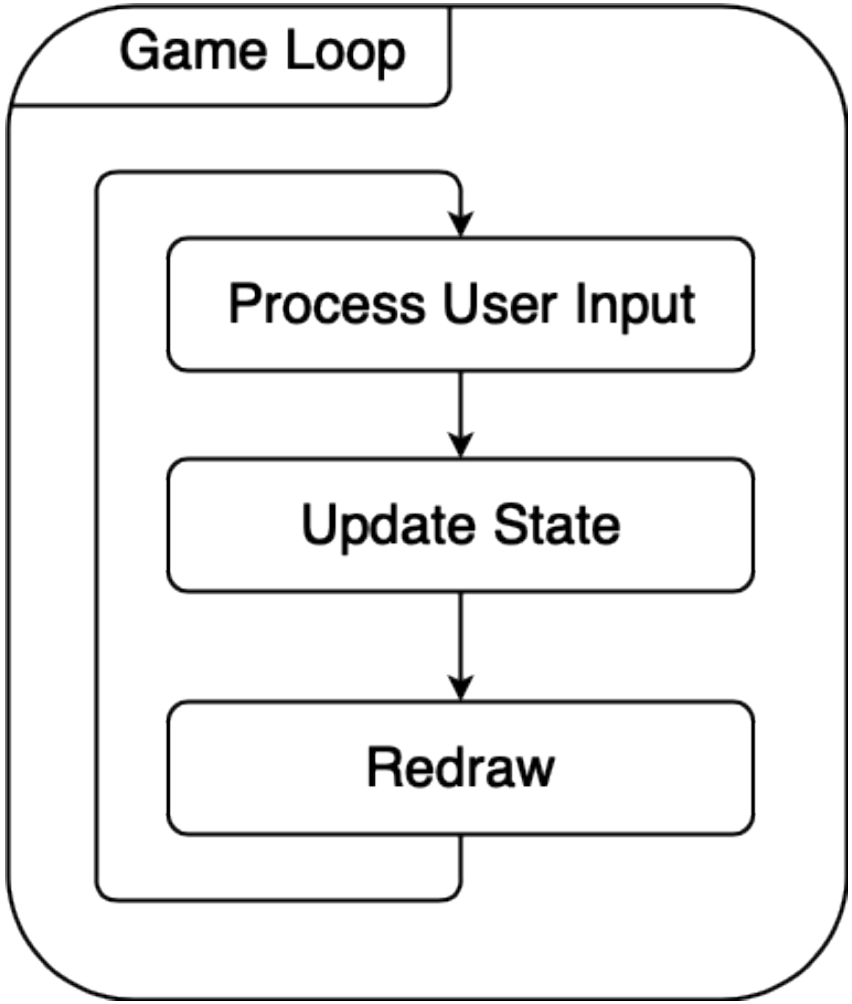
Non Functional Constraints:

Portability	Each Java capable device can run the game
Usability	User friendly and long-term satisfacting game
Cost	Minimal installation and setup efforts

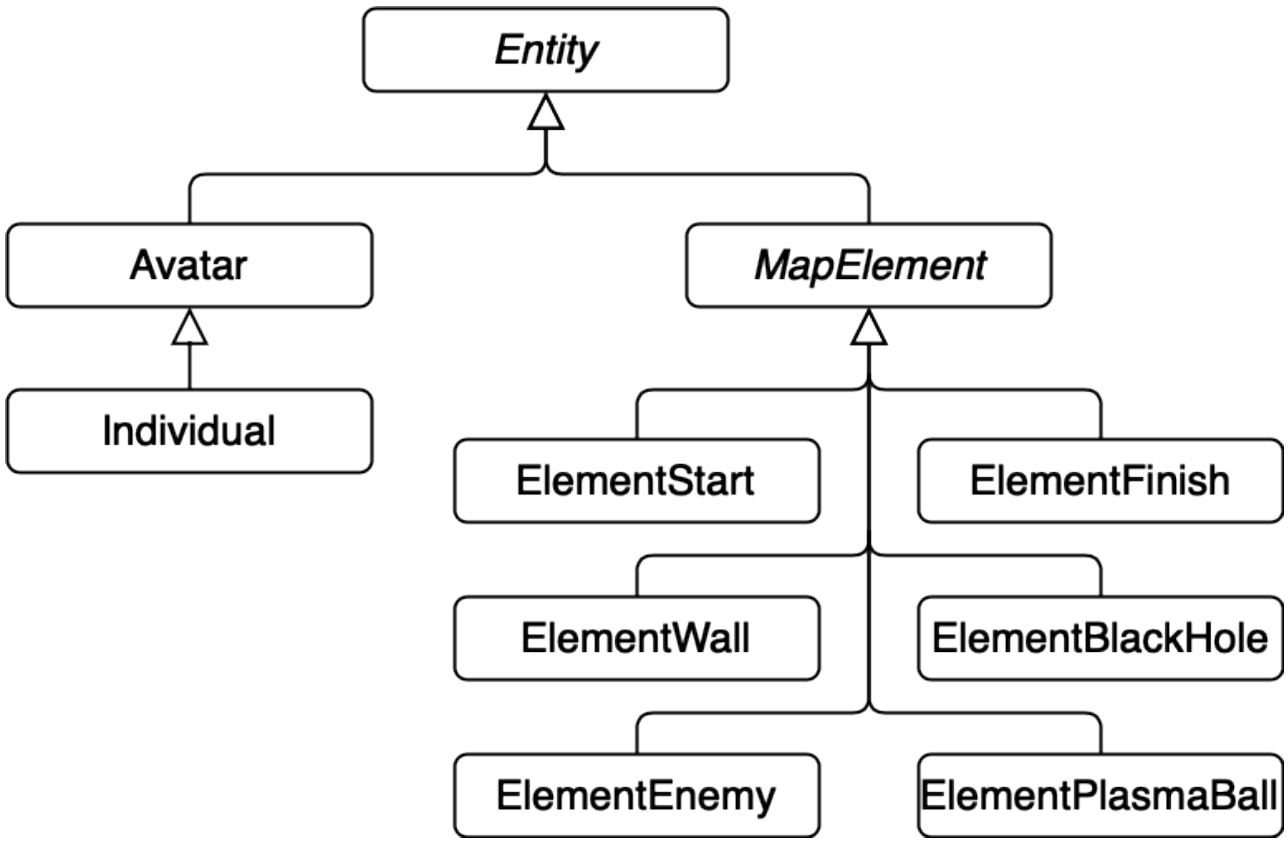
Architectural Patterns Used:

- Game Loop
- Type Object
- Double Buffer

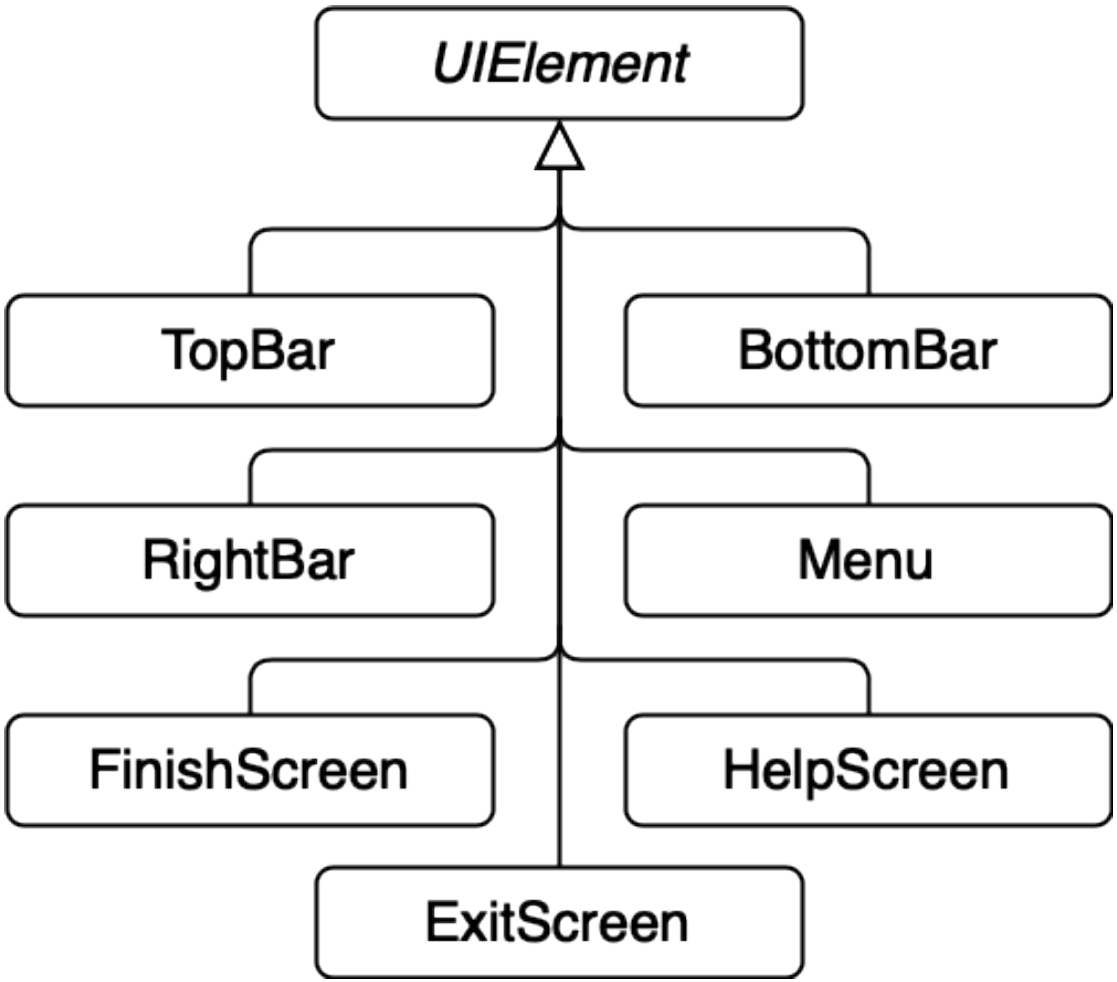
Game Loop



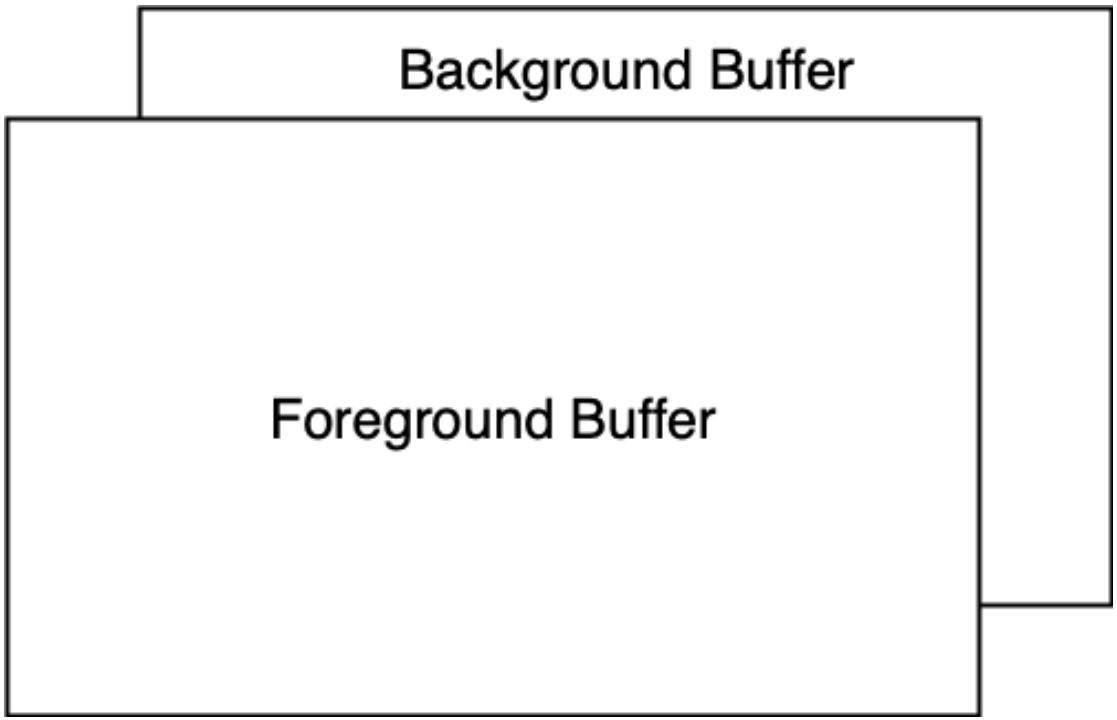
Type Object


Entity
Avatar
MapElement
Individual
ElementStart
ElementFinish
ElementWall
ElementBlackHole
ElementEnemy
ElementPlasmaBall

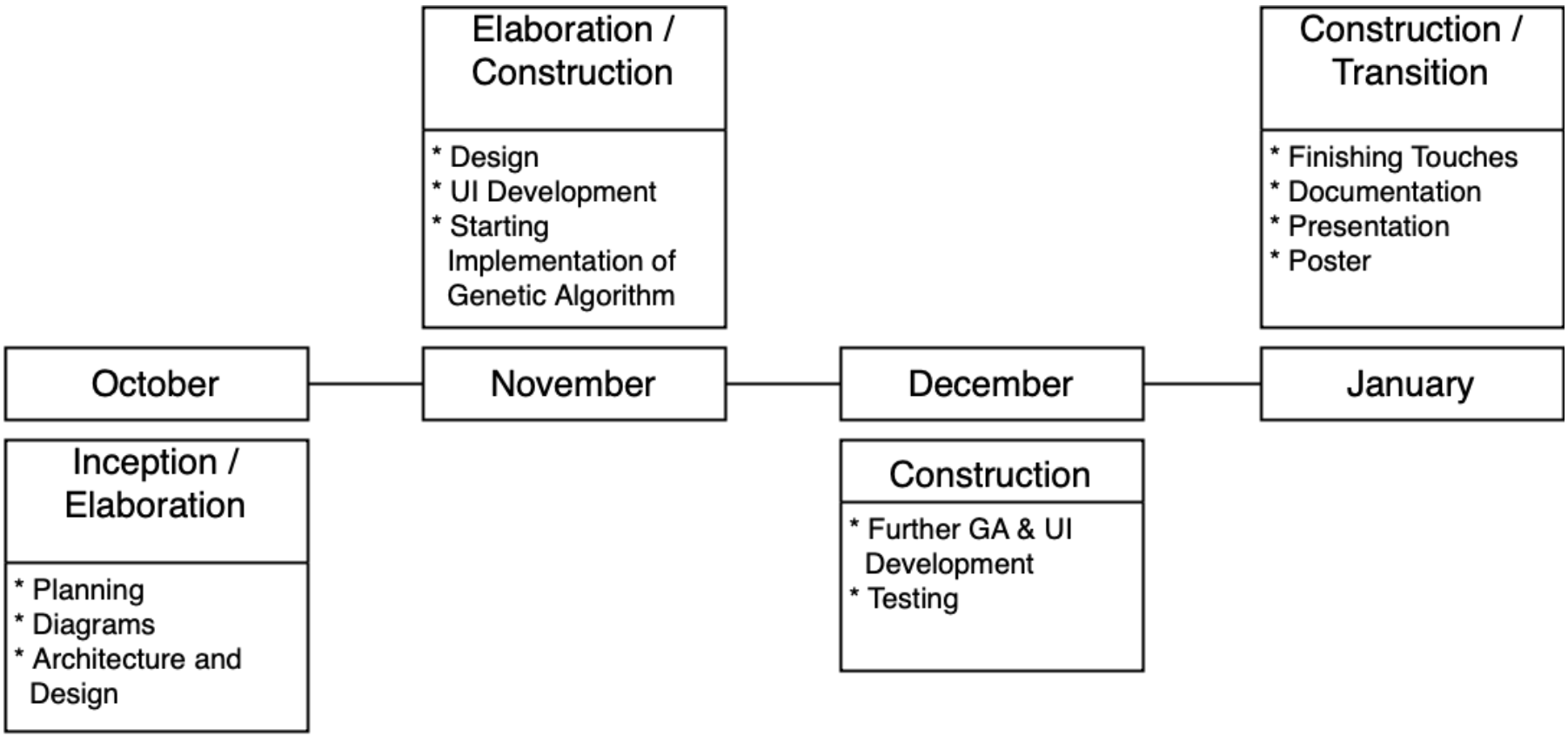
Type Object


UIElement
TopBar
BottomBar
RightBar
Menu
FinishScreen
HelpScreen
ExitScreen

Double Buffer



Timeline


October
Inception / Elaboration
* Planning
* Diagrams
* Architecture and Design
November
Elaboration / Construction
* Design
* UI Development
* Starting Implementation of Genetic Algorithm
December
Construction
* Further GA & UI Development
* Testing
January
Construction / Transition
* Finishing Touches
* Documentation
* Presentation
* Poster

Future Work

- Score board
- Reuse of pretrained AI
- Reinforcement learning
- UI/UX improvements

Lessons Learnt

- Managing project on Github (branching and merging)
- Importance of scrum meetings
- Proper time planning
- Importance of communication
- Coherent task distribution
- Adhering the coding standards (comments, diagrams etc.)

Demo: