Application Design Choices

These are the three main folders: Engine, Entity, and Map. They are:

- Engine:
    - This folder contains the engine (orchestrator) of the simulation. A base abstract class "Engine" is used as a template. An implemented "GameEngine" class contains the initial logic of the simulation. The Singleton pattern is used to only allow a single engine to run at any time.
- Entity:
    - This folder contains the classes related to the entity objects in the simulation. It includes a base abstract class called "Entity" that can cover any type of object in the simulation. Alongside this is an abstract "LivingEntity" class which contains further methods related to movement.
    - In our example, we use a "Bike" class to implement this moving object alongside an "Action" interface which all "LivingEntity" implement. For future work, the Builder pattern can be used to create and configure more complex entities with different types of action behaviours and states.
- Map:
    - This folder contains the classes related to the world or map of the simulation. An abstract base class of "Map" contains the details of which all "Map" type objects inherit from. In this example, we use a DefaultMap class to implement this.

Additional Dependencies and Plugins

Testing:

- JUnit plugins were used for unit testing. This ensures that any changes that break the application can be caught before these tests.
- Jacoco Maven Plugin is used to generate the test coverage. This enables developers to know where the code has not been tested.

Code Formatter:

- Spotless Maven Plugin is used to reformat the code to be compliant with industry standards (Google Java Style Guide). This enables every developer to create the same type of code for easier debugging and code sharing.

Javadoc Generator:

- The IntelliJ Javadoc generator was used to generate a standalone Javadoc web application to contain the application's API documentation.

CI/CD

GitHub was used to host this application. Alongside this, GitHub Pages was used to host the Javadocs and GitHub Actions were used to run the application's test cases in an isolated environment.

Future Works

For additional work, different types of maps and entities can be made with different properties. An end goal for the game can be defined as well, perhaps dodging other vehicles in the simulation and reaching a certain point on the map while collecting points or money on the way.