Predictions

Student name: Daniella Gordover
ID: 316464775
Email: daniellago@mta.ac.il
Bonuses: none

Predictions system

This system is a simulation engine tailored for researchers striving to recreate intricate scenarios. This system provides a personalized and adaptable environment, simplifying the creation of diverse simulations. Just upload an XML file outlining the simulation's framework. Whether it involves complex ecosystems, intricate societal dynamics, or a variety of phenomena, Predictions stands as the ideal tool to transform your research ideas into captivating simulations.

The project contains 3 modules:

1. **The engine module – where the magic happens. This module runs all the processes – from file validating, to running a simulation and etc.**
2. **The UI module – the user interface of the program – designed in JavaFX.**
3. **The DTO module – allows transferring data from the engine to the UI without compromising the original data.**

Engine module:

This module plays a crucial role by handling data, ensuring its accuracy based on user preferences.
Its initial task involves reading the XML file, meticulously analysing it for potential errors.
Once the file's correctness is verified, the Engine constructs the "WORLD" object. This entity encapsulates an entire universe, from environmental details to entity attributes.
Inside this "WORLD," you'll find a collection of Entities representing subjects for the simulation. Each Entity carries properties that make research diverse and insightful.
The simulation continues until the conditions specified in the XML file are met.
Notably, the Engine doesn't stop there – it archives past simulations, available for the user's future reference.

UI module:

In this project, we are introducing a new JavaFX application to the user.
At the upper section of the application, users can upload a file and monitor the activity of the thread pool as it operates.
The application consists of three main screens:

1. **Details Screen**:
   - This screen provides an overview of the file details before initiating a simulation.
   - File data is presented hierarchically as a tree structure, including entities, environment variables, rules, and other relevant information.
   - For each data member, the program displays detailed information.
   - Actions are organized as separate components, each with its own dedicated FXML file.

2. **New Execution Screen**:
   - Users can input population amounts for each entity on this screen.
   - It also offers the option to specify environment variable data, though this is not mandatory, and users can opt for random values.
   - Users can initiate new simulations from this screen, and it supports running multiple simulations via the "clear" and "set simulation data" buttons.

3. **Results Screen**:
   - This screen displays both ongoing and completed simulation data.
   - Upon the completion of a simulation, it provides details regarding the termination reason.
   - Users can access statistical analyses of the simulation data.
   - Additionally, users have the option to re-run the same simulation, effectively starting a new simulation using the same data as the previous one.
Facilitating the linkage between the UI and the Engine is the DTO module.

DTO module:

This module allows to “exchange” data between the engine and the UI modules, without transferring the actual data, but a copy of what is needed.

How it all works:

The engine module encompasses a framework that allows the XML file to seamlessly channel its data into the system.
Once the file is uploaded, the user gains the ability to observe simulation data and initiate simulations.
The file serves as a window into a virtual world, where research unfolds.
This world accommodates entities, each with its distinctive population count.
A compendium of rules resides within this world, designed to engage with entities based on activation conditions and unique properties.
Amidst this landscape, environment variables take their place, offering an array of possibilities to interact with entities.
When a simulation is set into motion, users can scrutinize the outcomes, viewing them through both entity lenses and property histograms.
Every entity adheres to the Entity Definition Interface, while properties find their definition within the Property Definition Interface.
Rules manifest through the Rules Interface, with each action stemming from an abstract Action class.
Stepping into the realm of environment variables, which bear semblance to properties, we encounter an Environment Manager (both in definition and activation). This manager curates a map of properties, facilitating smooth interaction.