There are two main parts of the app:

1. **app** directory contains all the view logic based on Angular 10;
2. **app-engine** directory contains the core logic of the game in vanilla typescript;

Inside the **app-engine** directory you can find two directories: **scene** and **script-runner.**

The script-runner directory mostly shouldn’t be touched.

The **app-engine/scene** directory contains all the code that makes up concrete scene’s game logic. Here is the place you need to add new features to the game! Let’s take a look at the files and directories. First the ones which *most often* shouldn’t be touched:

* directory **common** – contains common interfaces, abstract classes etc. which should be used by concrete scene implementations;
* **scene-model.service.ts** is independent of a concrete scene implementation and are used as a storage and single source of truth for the game’s state;
* **scene-accessors.service.ts** is also independent of a concrete scene and is used to provide accessors to the model for anyone;

Now the ones you gotta change to add new features:

* **scene-init.service.ts** – this is an important service which handles the initialization flow game as well as hooks existing concrete scene implementations to the backbone of the game. Whenever you get a new scene implementation, you should add its reader, writer, skulptAPI and tracker services to the *buildersMap* object so it can be initialized.
* **Any concrete scene directory** (e.g. *raccoon*, *pandemic*, *console*) including the ones you’re about to create – these are subject to modifications.

A concrete scene directory should contain the following:

* **entities** – anything domain specific described in form of interfaces/enums etc.
* **model** – a common interface which defines the structure of all entities interconnected
* **readers** – services which expose methods for getting and processing any model data and state
* **writers** – services which expose methods for modifying the model data and state
* **tracker** – a special service which is used for logging user’s actions during a pilot run of the app in web workers. The log collected by this service is then used to run the app in the view
* **builder service** – a special service used to assemble a model from a config, and also associate accessors and skulptAPI to it
* **skulpt service** – adds a scene’s API to the python interpreter so the user can interact with it
* **validation service** – is used together with the skulpt service to validate user input