➗

Logical Architecture

Package Hierarchy for `conkueror`

Package: `domain`

- **Package:** `domain` `.model`
  - **Package:** `domain` `.model.game`
  - **Package:** `domain` `.model.map`
  - **Package:** `domain` `.model.army`
  - **Package:** `domain` `.model.player`
  - **Package:** `domain` `.model.deck`
  - **Package:** `domain` `.model.card`
- **Package:** `domain` `.controller`
  - **Package:** `domain` `.controller.app`
  - **Package:** `domain` `.controller.listeners`

Package: `view`

- **Package:** `view.` `screens`
- **Package:** `view.` `components`
- **Package:** `view.` `graphics`

Description

Main Packages:

- The `domain.model` package contains classes that define the game logic and mechanics, including the game itself, the game map, the players, and the decks and cards used in the game.
- The `domain.controller` package contains classes that handle user input and manage the game state, including the main application and event listeners.
- The `view` package contains three sub-packages: `view.screens` , `view.components` , and `view.graphics` . These packages contain classes that handle rendering and displaying the game to the user, including different screens, individual UI components, and graphics rendering.

Domain Model Packages

The `domain.model` package contains classes that define the game logic and mechanics, including the game itself, the game map, the players, and the decks and cards used in the game.

- `domain.model.game` : This package contains classes related to the game itself, such as the `Game` class, which manages the game state and controls the flow of the game.

- **`domain.model.map`**: This package contains classes related to the game map, such as the `Map` class, which defines the layout of the game map and the various territories on it.
- **`domain.model.army`**: This package contains classes related to armies, such as the `Army` class, which represents a group of units that can be moved around the game map and used to attack other players.
- **`domain.model.player`**: This package contains classes related to the players, such as the `Player` class, which represents a single player in the game.
- **`domain.model.deck`**: This package contains classes related to decks and cards, such as the `Deck` and `Card` classes, which define the different types of cards that can be used in the game.
- **`domain.model.card`**: This package contains classes related to individual cards, such as the `ArmyCard`, `ChanceCard` and `TerritoryCard` classes.

Domain Controller Packages

The `domain.controller` package contains classes that handle user input and manage the game state, including the main application, event listeners, and network communications.

- **Package: `domain.controller.app`**: This package contains classes related to the main application, such as the `App` class, which serves as the entry point for the game and manages the overall application state.
- **Package: `domain.controller.listeners`**: This package contains classes that listen for user input and handle events, such as the `InputListener` class, which listens for mouse and keyboard events.

User Interface Packages

The `view` package contains three sub-packages: `view.screens`, `view.components`, and `view.graphics`. These packages contain classes that handle rendering and displaying the game to the user, including different screens, individual UI components, and graphics rendering.

- **Package: `view.screens`**: This package contains classes related to different screens that the user can interact with, such as the `MenuScreen` and `GameScreen` classes.
- **Package: `view.components`**: This package contains classes related to individual UI components that are used throughout the game, such as buttons, text fields, and menus.
- **Package: `view.graphics`**: This package contains classes related to graphics rendering, such as the `MapGraphics` and `Animation` classes, which define the appearance and behavior of various game elements.

Module Directory Structure

- **`conkueror/src`**: directory with the three main packages — the source code — **`domain`**`.model`, **`domain`**`.controller`, `view`
- **`conkueror/assets`**: all the assets for the game — avatars, buttons, fonts, background images etc.