Tic Tac Toe Documentation

- ## Unity version

The project was created using Unity version **2021.3.33f1 LTS**. To ensure the game functions correctly, it is important to run the project on the **same version or a newer version**.

- ## How to start a game

To correctly launch the game, switch to the initialization scene named **Boot**. Also don't forget to add **Boot, Main Menu Scene, Main Game Scene,** to the build settings.

- ## Project Entry Point

The entry point of the project is a script named **Bootstrapper**.

- ## Composition root

In this project, Zenject is used as the DI (Dependency Injection) framework. To find bindings of services and configs, look at the **GameInstaller** script. This script works within the **ProjectContext**.

- ## Game Settings

To change the game's initial settings, such as one round time, initial bundle name, computer think time, and game mode, go to the config file: **Assets->Configs->InitialGameConfig**.

- ## Asset bundle builder

Asset bundle builder is a custom window that helps create new bundles. To open this window in the Unity editor, follow these steps: **Window->Custom->Asset Bundle Builder**.

- ## Unit Tests

The project includes unit tests, which can be run using the Test Runner. The tests cover functions such as **Win, Lose, Draw, Undo, and Hint**.

- ## Game modes

Player vs Player, local multiplayer game mode: Two human players play on one screen.

Player vs Computer: A human player plays against a computer player. In this mode, hint, restart, undo, and round timer features are provided.

Computer vs Computer. Computer plays against a computer.

- ## Ready asset bundles

In the project, there are ready asset bundles that can be loaded at runtime. Here are their names: **defaultbundle, secondbundle, strangeonebundle, another strange bundle**

strangeonebundle, another strange bundle, they are very similar, differing only in the circle sign.

- ## Main Project Flow

The Boostrapper registers states in the state machine and transitions the state machine to the BootstrapState. In the BootstrapState, static data for the levels and the standard game bundle are loaded.

Next, the state machine transitions to the MainMenuState. This state loads the MainMenuScene and initializes the menu.

When the Start button is pressed, the game transitions to the LoadMainGameSceneState. Here, the MainGameScene is loaded, the game world is initialized, UI is created, and a 3x3 grid is generated.

Afterwards, the game transitions to the GameLoopInitialState. In this state, players and the DrawWinService are initialized.

Then, the game transitions to the PlayerTurnState. In this state, a player makes a move, and after the player's turn, the DrawWinService checks for win or draw conditions. If the game continues, it transitions back to the PlayerTurnState with the next player.

If the DrawWinService detects a win, the game transitions to the WinState. If it's a draw, the game transitions to the DrawState. From here, the game can return to the MainMenuState, or if the Restart button is pressed, it transitions back to the GameLoopInitialState.