SimpleGameUI / SimpleGameUI

Namespace **Playniax.UI.SimpleGameUI**

Inherits from **MonoBehaviour**

Script can be found in **Assets/Playniax/Framework/SimpleGameUI/Editor/SimpleGameUI.cs**

Class SimpleGameUI

Description

The SimpleGameUI will manage the different screens and can hold game data that has to be maintained throughout the game session like number of lives, current level, purchases, etc.

Public fields	Description
`string backgroundScene`	Determines what scene to load when your app starts. backgroundScene will run on the background of the SimpleGameUI. Left blank the SimpleGameUI will load the first level or last level played depending on settings. Don't forget to add these scenes to the Unity Build Settings or they won't load!
`bool startPaused = true`	Determines the scene starts paused or not (TimeScale = 0).
`bool newGameEverytime = true`	Whether to reset the game or not on every play.
`string[] levelSettings`	The levels or scenes to load as levels. Don't forget to add these scenes to the Unity Build Settings or they won't load!
`bool allowTransition = true`	Whether to show the transition scene or not.
`string transition`	The transition scene will be shown after each level is completed.
`bool allowAdvertisements = true`	Whether to show an ad between loading.
`string[] advertisements`	The scene(s) containing the ad(s). Don't forget to add these scenes to the Unity Build Settings or they won't load!
`Font font`	Fallback font.

Public Methods	Description
`static Font GetFont(Font font)`	Returns the fallback font.

`void Intermission(bool state)`	Returns the state of the intermission playing or not.
`virtual string ResetGame()`	Resets the game.
`void AboutButton()`	Is called when player selects the about button.
`void BackButton()`	Is called when player selects the back button.
`void ExitButton()`	Is called when player selects the exit button.
`void GameOver()`	Will show the Game Over page. Usually this is managed by the SimpleGameUI or other helpers but there might be circumstances where you want to manage it yourself.
`void GameOver(float delay)`	Will show the Game Over page but after some delay. Usually this is managed by the SimpleGameUI but there might be circumstances where you want to manage it yourself.
`int GetLevels()`	Returns number of levels.
`void ResetButton()`	Is called when player selects the reset button.
`void ResetYes()`	Is called when player selects the yes button.
`void ResetNo()`	Is called when player selects the no button.
`void SettingsButton()`	Is called when player selects the settings button.
`void ShopButton()`	Is called when player selects the shop button.
`int GetCurrentLevel()`	Retrurns the current level.
`void LevelUp()`	Will stop current level and load the next level.
`void Reload()`	Will stop current level and reload it.
`void PauseButton()`	Is called when player selects the pause button.
`void PlayButton()`	Is called when player selects the play button.
`void ReplayButton()`	Is called when player selects the replay button.

Some examples of how to use the public fields or methods:

The number of lives:

```
Debug.Log(SimpleGameUI.instance.playerSettings.lives)
```

Go to next level:

```
SimpleGameUI.instance.Levelup()
```

Show Game Over page:

```
SimpleGameUI.instance.GameOver()
```