UFE Coding (Advanced)

Besides visual tools, UFE also comes with several easy to use classes, events and methods all done in C#.
You can find usage examples under `UFE\Engine\Scripts\UI\Base\BattleGUI.cs` (Source) and `UFE\Engine\Scripts\UI\Templates\DefaultBattleGUI.cs`.

Example

```
void OnAwake(){
    /* Subscribe to UFE events:
     * Possible Events:
     * OnLifePointsChange(float newLifePoints, CharacterInfo player)
     * OnNewAlert(string alertMessage, CharacterInfo player)
     * OnHit(MoveInfo move, CharacterInfo hitter)
     * OnBlock(MoveInfo move, CharacterInfo blocker)
     * OnParry(MoveInfo move, CharacterInfo blocker)
     * OnMove(MoveInfo move, ControlsScript player)
     * OnRoundEnds(CharacterInfo winner, CharacterInfo loser)
     * OnRoundBegins(int roundNumber)
     * OnGameEnds(CharacterInfo winner, CharacterInfo loser)
     * OnGameBegins(CharacterInfo player1, CharacterInfo player2,
StageOptions stage)
     *
     * usage:
     * UFE.OnMove += YourFunctionHere;
     * .
     * .
     * void YourFunctionHere(T param1, T param2){...}
     *
     * The following code shows more usage examples
     */
    UFE.OnGameBegin += this.OnGameBegin;
    UFE.OnGameEnds += this.OnGameEnd;
    UFE.OnGamePaused += this.OnGamePaused;
    UFE.OnRoundBegins += this.OnRoundBegin;
    UFE.OnRoundEnds += this.OnRoundEnd;
    UFE.OnLifePointsChange += this.OnLifePointsChange;
    UFE.OnNewAlert += this.OnNewAlert;
    UFE.OnHit += this.OnHit;
    UFE.OnBlock += this.OnBlock;
    UFE.OnParry += this.OnParry;
    UFE.OnMove += this.OnMove;
    UFE.OnTimer += this.OnTimer;
    UFE.OnTimeOver += this.OnTimeOver;
    UFE.OnInput += this.OnInput;
```

```
}

void OnStart(){
    UFE.StartIntroScreen(1f);
}

public void OnGameBegin(CharacterInfo player1, CharacterInfo player2,
StageOptions stage){
    Debug.Log(player1.character.characterName + " - " player1.playerNum);
    Debug.Log(player2.character.characterName + " - " player2.playerNum);

    UFE.PlayMusic(stage.music);
}
```

Global Events

Use these to listen to several in-game events.

```
void OnLifePointsChange(float newLifePoints, CharacterInfo player)
```

> Triggered when the a character's life points change.

```
void OnNewAlert(string alertMessage, CharacterInfo player)
```

> Triggered when the game fires a new alert (first hit, combo, round 1, etc.).

```
void OnHit(HitBox strokeHitBox, MoveInfo move, CharacterInfo hitter)
```

> Triggered whenever a character gets hit.

```
void OnBlock(HitBox strokeHitBox, MoveInfo move, CharacterInfo blocker)
```

> Triggered whenever a character blocks.

```
void OnParry(HitBox strokeHitBox, MoveInfo move, CharacterInfo blocker)
```

> Triggered whenever a character parries.

```
void OnMove(MoveInfo move, ControlsScript player)
```

> Triggered whenever a new move is cast by a character.

```
void OnBasicMove(BasicMoveReference basicMove, CharacterInfo player)
```

> Triggered whenever a basic move is cast by a character.

```
void OnButtonPress(ButtonPress buttonPress, CharacterInfo player)
```

> Triggered whenever a button is pressed.


```
void OnButtonPress(ButtonPress buttonPress, CharacterInfo player)
```

> Triggered whenever a button is pressed.


```
void OnBodyVisibilityChange(MoveInfo move, CharacterInfo player,
BodyPartVisibilityChange bodyPartVisibilityChange, HitBox hitBox)
```

> Triggered whenever Body Parts Visibility Changes is casted.

```
void OnParticleEffects(MoveInfo move, CharacterInfo player,
MoveParticleEffect particleEffects)
```

> Triggered whenever Particle Effects is casted in a move.

```
void OnRoundBegins(int roundNumber)
```

> Triggered when the round begins.

```
void OnGameBegins(CharacterInfo player1, CharacterInfo player2, StageOptions
stage)
```

> Triggered when the game begins.

```
void OnGameEnds(CharacterInfo winner, CharacterInfo loser)
```

> Triggered when the game ends. Use `winner.playerNum` or `loser.playerNum` to know which player is which.

```
void OnTimer(float timer)
```

> Triggered whenever the timer ticks and returns the current timer value (float).

```
void OnTimeOver()
```

> Triggered when timer runs out.

Public Methods

You can find examples on all these under the following folders:

`.\Scripts\UI\Base\` (UFE Source and Bundle)
`.\Scripts\UI\Templates\` (Any version)

Note: All screen transition methods use the *Default Fade Duration* value under Global Editor -> GUI Options. You can override this value by typing the time (in seconds) as a parameter. Example:

```
void OnStart(){
    //Fades for 3 seconds before reaching the Main Menu:
    UFE.StartMainMenuScreen(3f);
}
```

Main

```
UFE.StartMainMenuScreen()
```

> Fades from the current loaded screen (if any) to the main menu screen.

```
UFE.StartOptionsScreen()
```

> Fades from the current loaded screen to the options screen.

```
UFE.StartCharacterSelectionScreen()
```

> Fades from the current loaded to the selected character selection screen.

```
UFE.StartStageSelectionScreen()
```

> Fades from the current loaded screen to the stage selection screen.

```
UFE.StartTrainingMode()
```

> Starts the character selection screen and sets `UFE.gameMode` to `GameMode.TrainingMode`. See Training Mode Options for more information.

```
UFE.StartLoadingBattleScreen()
```

> Fades to loading screen before starting the game. Its recommended you load the game this way, specially if you are using the Preload Options. Calling this function will use the current defined `GameMode` and characters selected.

```
UFE.StartGame()
```

> Skips the loading screens and fades to the game screen instantly. Calling this function will use the current defined `GameMode` and characters selected.

Extras

```
UFE.StartIntroScreen()
```

> Fades from the current loaded screen (if any) to the selected intro screen.

```
UFE.StartCreditsScreen()
```

> Fades from the current loaded screen to the selected credits screen.

```
UFE.Quit()
```

> Quits the application.

Versus

```
UFE.StartPlayerVersusPlayer()
```

> Starts the character selection screen and sets `UFE.gameMode` to `GameMode.VersusMode`.

```
UFE.StartCpuVersusCpu()
```

> Starts the character selection screen with UFE.SetCPU set to `true` for both players. It also sets `UFE.gameMode` to `GameMode.VersusMode`.

```
UFE.StartPlayerVersusCpu()
```

> Starts the character selection screen with UFE.SetCPU set to `true` for player 2. It also sets `UFE.gameMode` to `GameMode.VersusMode`.

```
UFE.StartVersusModeAfterBattleScreen()
```

> Starts the option menu that shows after a match on *Versus Mode*.

Story

```
UFE.StartStoryMode()
```

> Starts the character selection screen and sets `UFE.gameMode` to `GameMode.StoryMode`. During *Story Mode* `UFE.SetCPU` is set to `true` for player 2 and it runs a series of battles listed under Story Mode options while following the intro and outro instructions. All subsequent functions are executed automatically after this.

```
UFE.StartStoryModeOpeningScreen()
```

> (*) Starts the opening screen. This function is automatically called after player selects the character on GameMode.StoryMode.

```
UFE.StartStoryModeBattle()
```

> (*) Starts the next battle listed on the story of this character. This function is automatically called after every conversation on GameMode.StoryMode.

```
UFE.StartStoryModeContinueScreen()
```

> (*) Starts the continue screen. This function is automatically called after the player loses a match on GameMode.StoryMode.

```
UFE.StartStoryModeConversationBeforeBattleScreen(UFEScreen
conversationScreen)
```

> (*) Starts the conversation before battle listed on the story for this match. This function is automatically called after each battle on GameMode.StoryMode.

```
UFE.StartStoryModeConversationAfterBattleScreen(UFEScreen
conversationScreen)
```

> (*) Starts the conversation after battle listed on the story for this match. This function is automatically called after each battle on GameMode.StoryMode.

```
UFE.StartStoryModeCongratulationsScreen()
```

> (*) Starts the congratulation screen. This function is automatically called after the player defeats all the opponents on GameMode.StoryMode.

```
UFE.StartStoryModeEndingScreen()
```

> (*) Starts the ending screen. This function is automatically called after the congratulations screen on GameMode.StoryMode.

```
UFE.StartStoryModeGameOverScreen()
```

> (*) Starts the game over screen. This function is automatically called after the continue screen timer runs out on GameMode.StoryMode.

* These methods are used mostly internally. Make sure you know what you are doing before calling them!

Core Control

```
UFE.SetPlayer1(CharacterInfo player1)
```

> Sets the player 1 character (CharacterInfo class).

```
UFE.GetPlayer1()
```

> Returns the player 1 selected character (CharacterInfo class).

```
UFE.SetPlayer2(CharacterInfo player2)
```

> Sets the player 2 character (CharacterInfo class).

```
UFE.GetPlayer2()
```

> Returns the player 2 selected character (CharacterInfo class).

```
UFE.SetCPU(int player, bool cpuEnabled)
```

> Set if `player` (1 or 2) is controlled by the A.I. engine or a human player. Make sure you set this method *before* the battle begins.

```
UFE.SetLanguage(string language)
```

> Set the language to one of the listed languages.

```
UFE.SetStage(string stageName)
```

> Set the selected stage to one of the listed stages.

```
UFE.GetStage()
```

> Set the selected stage (StageOptions class).

```
UFE.GetFont(FontId fontId)
```

> (*Deprecated*) Returns the font (FontOptions class) based on the provided id (FontId enum) (as defined under global editor's fonts).

```
UFE.GetCurrentMoveSet(CharacterInfo character)
```

> Returns the current selected move set (MoveSetData class) from the provided character.

```
UFE.SetLifePoints(float newValue, CharacterInfo character)
```

> Set player's life points.

```
UFE.FireAlert(string alertMessage, CharacterInfo character)
```

> Fire a new game alert. If you are using `GUIScript.cs`, the character provided will represent the side it will come out.

```
UFE.GetMusic()
```

> Is the music on or off.

```
UFE.SetMusic(bool on)
```

> Sets music on/off.

```
UFE.LoopMusic(bool loop)
```

> Sets music looping mode on/off.

```
UFE.GetSoundFX()
```

> Is the sound fx on/off.

```
UFE.SetSoundFX(bool on)
```

> Sets sound fx on/off.

```
UFE.GetVolume()
```

> Gets audio volume.

```
UFE.SetVolume(float volume)
```

> Sets audio volume.

```
UFE.GetInputReference(ButtonPress button, InputReferences[] inputReferences)
```

> Get Unity Input Manager's button reference based on an UFE assigned Key.

```
UFE.PlaySound(AudioClip sound)
```

> Play a sound FX once (sound fx must be toggled on)

```
UFE.PauseGame(bool pause)
```

> Pause Game.

```
UFE.IsPaused()
```

> Is the game paused?

Public Variables

All public variables defined on the editors can be found and manipulated live using the following classes:

- Global data: `UFE.config`
- Character data: `UFE.config.player1Character` or `UFE.config.player2Character`
- Move data: `UFE.config.player1Character.moves` or `UFE.config.player2Character.moves`

With the project open, use IntelliSense to reach the desired variable.

< Back to Universal Fighting Engine - Introduction

From:
http://ufe3d.com/ - **Universal Fighting Engine**

Permanent link:
http://ufe3d.com/doku.php/code?rev=1614892929

Last update: **2021/03/04 16:22**