FORK PARTICLE

Getting Started with Fork Particle for Unity Integration

Compatible with Unity 5.6.1

Installed Package Components

Fork Particle package will add the following files and folders:

- Editor:
  - Contains scripts to convert Fork PSB to use able objects.
- Plugins:
  - Contains bundle and dll for MAC (OpenGL) and PC (DX11).
- Resources/ForkFX:
  - Contains PSB files (effect files exported from Fork Particle Studio) and their associated assets (textures and prefabs).
- Scene:
  - ForkScene, demonstrate how you can set up Fork Effects in your game.
- Scripts:
  - ForkAPI, contains scripts required to run Fork Effects.

Work Flow:

1) Create a global object and attach ForkParticlePlugin script to it.
2) Export an effect PSB from Fork Particle Studio.
3) Drag and drop OR Copy the PSB into Resources/ForkFX Folder along with its textures. Mark the textures as Read/Write Enabled from Inspector.
4) Instantiate the prefabs created against the imported PSBs.
5) That's it!!

NOTE:

We recommend that you place the Fork particle exported PSB in the Unity Resources/ForkFX folder along with its texture. The textures should be marked as "Read/Write Enabled". The PSB is also required in the standalone build. You can use post build commands in Unity to add them to your build.

ForkParticleEffect: API and FUNCTIONS

Attached to the prefabs that are created from PSB files. It creates the effect by passing its PSB data to dll through ForkEffectCreate function. The dll after successful creation of the effects passes the effects pointer back. The prefab uses its effect pointer to pass its color, scale and rotation to the dll. It sets a call back function for the dll to pass the textures name to it, the prefabs loads the texture pointer from Resources/ForkFX folder and passes it back to dll so that it could be applied to the effect.

- **Start**
  Checks if the Fork SDK has been initialized. If the SDK has been initialized it ask the dll to create the effect. The effects are created in stop state. You will have to manually call the PlayEffect (refer below) to start the effect.

- **Update**
  Checks if the SDK has been initialized. If the SDK check fails in the Start function it checks for the SDK initialization again. Once an effect is crated after SDK initialization it is marked as valid. Effects marked as valid will start setting their translation, rotation and scale in the dll. Fork uses uniform scale, the value of “x” component of the scale will be applied to x, y and z components of the effects.

- **ClearForkEffect**
  Kills all the particles instantly. Once killed the particles will be born again. You can simply call this public function on prefab.

- **IsEffectAlive**
  Returns an integer value depending upon the effect alive state. Returns 1, if the effect is alive 0 otherwise.

- **EffectFreeze**
  Freezes the effect in its current state. New particles are not born and currently emitted particles are not simulated. The frozen effect will remain in its current state unless PlayEffect is called.

- **PlayEffect**
  Un-Freezes the effect.

- **SetEffectColor**
  Change the color of the effect in R, G, B, A format.

- **RestartEffect**
  Kills all the particles instantly and plays the effect

- **OnDestroy**
  Unity's function, called when an object is destroyed. Destroys the effect and free its resources in the dll. Unity automatically calls OnDestroy. You can simple destroy the prefab you have instantiated.

ForkParticlePlugin: API and FUNCTIONS

Initializes the Fork SDK and starts the render co routine. The SDK should be initialized before all. If the SDK is not initialized the effects will not be created.

This section provides descriptions of function implemented in the ForkParticlePlugin script. You need not call any of these functions yourself.

- **Start**
  Initializes the Fork SDK. Should be called before any other SDK and effect call. The dll call returns a Boolean value, true if SDK initialization is successful false otherwise.

- **ShutDownForkSDK**
  Shuts down the Fork SDK. Shutdown is called from OnApplicationQuit. The shutdown function sets the validation of all the created effects to false, due to which all the effects stops setting their translation and rotation. The main dll call that shuts down the returns a Boolean value, true if SDK shutdown is successful, false otherwise. If you are switching to a new scene and initializing the SDK again in the new scene we recommend that you shut down the SDK beforehand. The ideal state would be to mark the SDK object as do not destroy on load, so that you don't have to initialize the SDK again in new scene.

Plugin Architecture:

The Editor script PSBImporter converts the PSB exported from Fork Particle Studio to a useable prefab. It attaches a ForkParticleEffect script to it so that the effect could call dll functions on its self. It also attaches Mesh Renderer and Mesh Filter to it so that the particles could render. You can simply drag and drop OR copy the PSB files into the Resources/ForkFX folder.

The second Editor script PSBPostProcessor provides facility to update the prefab assets. You can move, delete and add the PSB files and it will take care of the prefab assets. PSBAsset script reads and loads the PSB data for the effect.

Plugins folder contains the MAC bundle that can be used for Unity MAC for both editor and standalone version. It also contains x64 and x86 folders with PC dlls.

Resources/ForkFX folder contains sample effects PSBs (exported to run on MAC, PC and PCx64) their converted prefabs and their relevant textures. The prefabs can simply be used as they are without any changes required in them. The textures are marked as Read/Write Enabled.

Scene folder contains a sample scene that can be loaded to see how you can setup Fork Effects in you game. The scene contains _ForkPlugin object which has a ForkParticlePlugin script attached to it.

NOTE:

The Editor scripts in the package should be placed under your main Editor folder. We recommend that you places the ForkFX folder under your main Resources folder and place all the PSBs exported from Fork Particle Studio and their related texture in Resources/ForkFX folder. The plugins folders contains the dlls for both x64 (for editor) and x86 (for standalone build). You can copy the whole Plugin folder if you don't have one already. The scripts related to Fork Particle plugin are under Scripts/ForkAPI folder, you can simply copy the ForkAPI folder and place it under your Scripts folder. The package also contains a sample scene ForkScene which can be loaded to get the idea how Fork Effects can be used in the game.