Fabric Engine JavaScript Scene Graph Developer Install Guide

Fabric Engine Version 1.2.0-beta

Table of Contents

1. Introduction 5
    1.1. Installing the Plugin 5
2. Core Setup 7
    2.1. Prerequisites 7
        2.1.1. Windows 7
            2.1.1.1. Install Git and Configure SSH 7
        2.1.2. Mac OS X 7
        2.1.3. Linux 7
    2.2. Clone the PublicDev Repository 7
3. SceneGraph Setup 9
    3.1. Prerequisites 9
    3.2. Install SceneGraph Resources 9
        3.2.1. Mac OS X and Linux 9
        3.2.2. Windows 9
            3.2.2.1. Install Tortoise SVN 9
            3.2.2.2. (Optional) Setup SSH for write access 9
            3.2.2.3. Checking out the JSSceneGraphResources repository 10
            3.2.2.4. References 10
    3.3. Setting up a local Fabric web server 11
        3.3.1. Windows 11
        3.3.2. Mac OS X 12
        3.3.3. Linux 13
    3.4. Run Sample Fabric Applications 13
4. Installing optional extensions 15
    4.1. Installing the Kinect Extension (Windows only) 15
    4.2. Installing the filesystem extension 15
        4.2.1. Installation 15
        4.2.2. Demos 15
        4.2.3. General overview of File IO in Fabric Engine 16
        4.2.4. KL Usage 16
        4.2.5. Creating a FabricFileHandle for use in other extensions 17
        4.2.6. Use of the FabricFILESTREAM 17
5. Debugging 19
    5.1. Windows: Attaching the VisualStudio debugger 19

Chapter 1. Introduction

This document will outline how to set up your development environment in order work with the Fabric Engine JavaScript Scene Graph under Mac OS X, Linux, and Windows.

Fabric supports the following platforms:

- *Windows*: Vista or Windows 7.
- *Mac OS X*: 10.6 (Snow Leopard) or above
- *Linux*: Tested on Ubuntu 10.04 and above (32-bit or 64-bit). Should work with any modern distro.

1.1. Installing the Plugin

Developers building client side web applications do not need to build Fabric Engine from souce and can use the plugin installed when visiting any of our online demos. Note that once the plugin is installed, developers can immedeiately start working with Fabric as the entire toolchain is distributed as part of the plugin. The remaining portion of the document covers retrieving the Fabric Engine Java Script Scene Graph source code and working with extensions.

Fabric Engine Web Demos [http://demos.fabric-engine.com]

Chapter 2. Core Setup

2.1. Prerequisites

Before getting started you will need several packages installed in order to be able to retrieve and compile the Fabric source code. Packages vary depending on build operating system.

2.1.1. Windows

The Fabric Engine source code is hosted in a GitHub repositroy, which uses SSH for security. You will need to set up Git and SSH on your windows machine to be able to clone the repository.

2.1.1.1. Install Git and Configure SSH

Install Git for Windows and (optionally) set up your github ssh key by following the instructions found here: `Set Up Git` [http://help.github.com/win-set-up-git/]

If you want the system to remember your git passphrase, you can follow the instructions found here: `Why git can't remember my passphrase under Windows` [http://stackoverflow.com/questions/370030/why-git-cant-remember-my-passphrase-under-windows]

2.1.2. Mac OS X

Follow the instructions on GitHub for setting up Git on OsX. `mac-set-up-git` [http://help.github.com/mac-set-up-git/]

2.1.3. Linux

Follow the instructions on GitHub for setting up Git on Linux. `linux-set-up-git` [http://help.github.com/linux-set-up-git/]

2.2. Clone the PublicDev Repository

If read-only access is enough, you can clone the PublicDev repository from the command line without setting up an ssh key by doing:

```
git clone https://github.com/fabric-engine/PublicDev.git $FABRIC_CORE_PATH
```

Where `$FABRIC_CORE_PATH` is the folder set up above.

If you will require write access then you will need to:

- Ensure that you have uploaded your public SSH key to your github.com account.
- Test that your SSH keys are properly set up for github.com by running: `ssh -T git@github.com`. You should see a message like the following: `Hi myuser! You've successfully authenticated, but GitHub does not provide shell access.`
- If your keys have been set up and you have been granted write access, you can check out the repository by running: `git clone ssh://github.com/fabric-engine/PublicDev.git $FABRIC_CORE_PATH`

Chapter 3. SceneGraph Setup

If you will be working with the Fabric Javascript SceneGraph you'll need to install some additional resources in addition to a web server for local testing.

3.1. Prerequisites

The following software is required to use the NPAPI plugin client:

- Google Chrome [http://www.google.com/chrome/]. You can also use Firefox, but Chrome is recommended for now. Use standard settings.

3.2. Install SceneGraph Resources

3.2.1. Mac OS X and Linux

You'll need to install Subversion via MacPorts or your distributions package manager. Once installed, you will need to check them out into your `$FABRIC_CORE_PATH/Web` folder by running:

```
svn co --force http://svn.fabric-engine.com/JSSceneGraphResources/stable
$FABRIC_CORE_PATH/Web
```

If you have been granted write access to the resources repository via ssh, you'll only need to change your checkout command to use ssh rather than http:

```
svn co --force svn+ssh://svn.fabric-engine.com/JSSceneGraphRe-
sources/stable $FABRIC_CORE_PATH/Web
```

3.2.2. Windows

These instructions are specific to Tortoise SVN; other clients might be as good or better (since Tortoise SVN has command line issues). You may use any SVN client you prefer but these instructions will outline how to configure Tortoise.

3.2.2.1. Install Tortoise SVN

Tortoise SVN Windows installers are available from their Sourceforge files page [http://sourceforge.net/projects/tortoisesvn/files].

Note: In TortoiseSVN's installation steps, enable the installation of the "Command Line Tools" component.

3.2.2.2. (Optional) Setup SSH for write access

Note: This only applies if you have been granted write access to the resources repository by Fabric.

3.2.2.2.1. Configure Tortoise SVN for OpenSSH (not required if you use PuTTY for SSH)

If you are using OpenSSH for git SSH:

- Install Putty [http://tartarus.org/~simon/putty-snapshots/x86/putty-installer.exe]
- Convert your OpenSSH key to the PuTTy format:
  - Open puttygen.exe from the PuTTY install directory
  - Do Conversion->Import key, and select your .ssh private key (eg: C:\Users\myname.ssh\id_rsa)
  - Do File->Save private key, and save it as [keyname].ppk (eg: id_rsa.ppk)

3.2.2.2.2. Setup a PuTTY session for svn+ssh

- Open PuTTY.exe
- In the Connection->SSH->Auto item, set Private key file for authentication to you .ppk private key path (eg: C:\Users \myname.ssh\id_rsa.ppk)
- In the Session tab, set the Host Name to "[svnuser]@svn.fabric-engine.com" (eg: myname@svn.fabric-engine.com), set a session name (eg: SVNConnection), and press Save (should appear in the Saved Sessions list)
- Start pageant.exe (agent from the PuTTY folder). On the Windows taskbar icons, find the PuTTY agent, right-click and choose "Add Key", and select your .ppk PuTTY private SSH key.

Note: This agent needs to be started before any SVN operation is done; you might set it as a startup program.

3.2.2.3. Checking out the JSSceneGraphResources repository

For some reason, operations over the SSH tunnel don't work from the command line. Until someone finds how to fix that, all SVN operations are done with Turtoise's customized Windows Explorer menu.

- Open the Fabric folder in an Explorer. Right-click on the Web folder, and select "SVN Checkout" in the contextual menu.
- Set the URL of the repository to "svn+http://SVNConnection/JSSceneGraphResources/stable" (where "SVNConnection" is the name of your saved PuTTY session), and set the Checkout directory to "[FABRICPATH]\Web" (eg: C: \Users\myuser\Fabric\Web).
- If you have been granted write access, instead set the URL of the repository ="svn+ssh://SVNConnection/JSSceneGraphResources/stable".
- You should now be able to run SVN commands from the Windows Explorer on files or folders located in the Web sub-directories, using the contextual menu (right-click).

3.2.2.4. References

If you are unfamiliar with SVN or run into trouble with the above, the following resources may be useful:

- Securing Svnserve using SSH [http://tortoisesvn.net/docs/nightly/TortoiseSVN_en/tsvn-ssh-howto.html]
- Tortoise SVN + Putty SSH integration tutorial [http://vimeo.com/5378553]

3.3. Setting up a local Fabric web server

In order to the SceneGraph locally you will need to install a local web server. The web server used will depend on your operating system.

3.3.1. Windows

The following describes how to setup a local web server using Windows's Internet Information Services (IIS). Other web server services, such as Apache, work too however we don't have precise steps for these.

- Activating IIS:
  - Click Start and then click Control Panel
  - In Control Panel, click Programs And Features and then click Turn Windows features on or off
  - In the Windows Features dialog box, click Internet Information Services and then click OK (you can use the default sub-feature selection)
- Configuring IIS:
  - From the Windows Start Menu search field, type 'inetmgr' and run it, this will open the IIS configuration tool
  - On the left pane, expand to -+Sites-+Default Web Site
    - Right-click over the "Default Web Site" item, and select 'Add Virtual Directory...'
      - Alias = Fabric
      - Physical path = FABRICDIR\Web (eg: C:\Fabric\Web)
    - Click Ok
  - On the left pane, select Fabric, under 'Default Web Site'
  - On the center pane, open the 'Directory Browsing' item
    - On the right pane, click 'Enable'
  - On the left pane, select Fabric, under 'Default Web Site'
  - On the center pane, open the 'MIME Types' item
    - On the right pane, click 'Add...' to add all of these
      - File name extension: .kl MIME Type: text/plain
      - File name extension: .obj MIME Type: text/plain
      - File name extension: .dae MIME Type: text/plain
      - File name extension: .cl MIME Type: text/plain
      - File name extension: .glsl MIME Type: text/plain

  - File name extension: .ply MIME Type: text/plain
  - File name extension: .abc MIME Type: application/octet-stream
  - File name extension: .laz MIME Type: application/octet-stream
  - File name extension: .nrrd MIME Type: application/octet-stream
- In order to avoid clearing browser cache when modifying files, caching should be disabled. On the left pane, select Fabric, under 'Default Web Site'
  - On the center pane, open the 'Output Caching' item
  - On the right pane, click 'Edit Feature Settings...'
  - Uncheck 'Enable cache'

Note: In some cases, Skype can conflict with localhost. Localhost will not be available and clicking 'Start' for localhost in the inetmgr.exe will give an error about a file being used by another process. To avoid the problem, turn off: Skype -> Tools -> Options -> Advanced -> Connection -> Use port 80 and 443 as alternative for incoming connections.

3.3.2. Mac OS X

Note that in the follow steps, `$USER` should be replaced with your UNIX username on your Mac OS X system. If you're unsure what your UNIX username is, in Terminal run `echo $USER`.

- Ensure that "Web Sharing" is enabled in the System Settings
- As root (ie. using sudo), edit the text file `/private/etc/apache2/users/$USER.conf`
- Change the `AllowOverride` directive to `All` (it defaults to `None`). The file will look something like this:

```
<Directory "/Users/$USER/Sites/">
    Options Indexes MultiViews
    AllowOverride All
    Order allow,deny
    Allow from all
</Directory>
```

- Create a new text file `~/Sites/.htaccess` with the following content:

```
Options +FollowSymLinks
Header add Cache-Control "no-cache, no-store, max-age=0, must-revalidate"
Header add Pragma "no-cache"
Header add Expires "Fri, 01 Jan 1990 00:00:00 GMT"
```

- Restart the web server by running: `sudo apachectl graceful`
- the URL http://localhost/~$USER/Fabric direct to `~/Fabric/Web` by running `ln -s ~/Fabric/Web ~/Sites/Fabric`

3.3.3. Linux

For the following instructions, replace `$USER` with your username.

- Ensure that Apache is installed by running:

```
sudo apt-get install apache2
```

- Enable user directories and header control on Apache2 by running:

```
sudo ln -s /etc/apache2/mods-available/userdir.load /etc/apache2/mods-enabled/
sudo ln -s /etc/apache2/mods-available/userdir.conf /etc/apache2/mods-enabled/
sudo ln -s /etc/apache2/mods-available/headers.load /etc/apache2/mods-enabled/
sudo /etc/init.d/apache2 restart
```

- Create a new text file `~/public_html/.htaccess` with the following content:

```
Header add Cache-Control "no-cache, no-store, max-age=0, must-revalidate"
Header add Pragma "no-cache"
Header add Expires "Fri, 01 Jan 1990 00:00:00 GMT"
```

- Make the URL http://localhost/~$USER/Fabric serve `~/Fabric/Web` by running:

```
ln -s ~/Fabric/Web ~/public_html/Fabric
```

3.4. Run Sample Fabric Applications

- In Google Chrome, navigate to http://localhost/Fabric.
- Run any of the demos in your browser.

Chapter 4. Installing optional extensions

4.1. Installing the Kinect Extension (Windows only)

To install the Fabric Engine extension for supporting the Microsoft Kinect hardware, you first need to install the Kinect SDK. You can download the SDK on the official site: Kinect for Windows [http://kinectforwindows.org/].

Once the SDK is installed, please connect up your Kinect and try it with one of the sample applications provided by the Kinect SDK, for example the Skeletal Viewer.

Once the camera has proven to work, you need to get the extension files from the Fabric distributions page [http://dist.fabric-engine.com/latest/]. The file is named `FabricEngine-KinectExt-Windows-x86-VERSION.zip`, where VERSION is the latest Fabric version.

Unzip the files into the Extensions folder, based in `%APPDATA%/Fabric/Exts`.

Once the extension files are put in the right place, along with the other Fabric Engine extensions, you should restart your browser, and try one of the Fabric Engine based applications utilizing the kinect, for example this one: Kinect Depth Points [http://demos.fabric-engine.com/Apps/Sample/Kinect/DepthPoints.html]

Other applications available for reference are:

- Kinect Tilt [http://demos.fabric-engine.com/Apps/Sample/Kinect/Tilt.html]
- Kinect Depth [http://demos.fabric-engine.com/Apps/Sample/Kinect/Depth.html]
- Kinect Skeletons [http://demos.fabric-engine.com/Apps/Sample/Kinect/Skeletons.html]

4.2. Installing the filesystem extension

The Fabric Engine extension for accessing the local filesystem, called FabricFILESYSTEM, is a so called 'unsafe' extension. Since it allows Fabric to access your local drives, be sure to understand the risks of installing this extension. It provides high performance local file access, but introduces security risks for unknown or malicious Fabric applications. This extension works both for Fabric Engine client as well as the Fabric Engine Node.js module.

You can download the FabricFILESYSTEM extension for all supported platforms from the Fabric distributions page [http://demos.fabric-engine.com/latest/]. The file is named `FabricEngine-FileSystemExt-OS-VERSION.zip` or `FabricEngine-FileSystemExt-OS-VERSION.tar.bz2` where OS is your operating system and VERSION is the latest Fabric version.

4.2.1. Installation

- Windows: Unzip the archive into the Extensions folder, based in `%APPDATA%/Fabric/Exts`.
- Mac/Linux: Untar the archive into the Extensions folder, based in `~/.fabric/Exts`. You might have to create the directory if it doesn't exist.

4.2.2. Demos

- Accessing the local filesystem and browsing folders: FileSystem [http://demos.fabric-engine.com/Apps/Sample/BasicDemos/FileSystem.html]

- Reading and writing local files using the FileSTREAM: FileStream [http://demos.fabric-engine.com/Apps/Sample/BasicDemos/FileStream.html]

4.2.3. General overview of File IO in Fabric Engine

Fabric Engine supports access to the local harddrive(s) through a special kind of string, which is called the FabricFileHandle. It is a secure unique key, which represents a path to folder or a file on disk, but it is encrypted. That way you can provide a FabricFileHandle to extensions for example, without knowing which file you are reading or writing. This provides a secure model for allowing the application binary access to local files while Fabric Engine cannot access files without the user's action. The process is as follows:

- User can be asked to provide a FabricFileHandle (through an open or save dialog).
- The chosen path will be encoded to a secret unique key by the Fabric Engine core.
- The FabricFileHandle can be passed around and provided to extensions.
- Extensions can create a FILE * in C++ for example out of the FabricFileHandle using the EDK.

This means that every FabricFileHandle will have to be initiated by the user and makes Fabric Engine's IO secure.

The FabricFILESYSTEM extension adds the ability to create FabricFileHandles from fullpaths, providing full access to the harddrive without any limitations, therefore it is not secure. We don't distribute the extension together with Fabric Engine, but it is available as a separate download (on this page).

4.2.4. KL Usage

The extension provides two new KL types to Fabric Engine: The FabricFolderHandle and the FabricFileHandleWrapper. The FabricFolderHandle provides access to folders on the local disk, as well as the contained subfolders and files. Its API is as follows:

```
FabricFolderHandle.setAbsolutePath(String path)
String FabricFolderHandle.getAbsolutePath()
String FabricFolderHandle.getBaseName()
Boolean FabricFolderHandle.isValid()
Boolean FabricFolderHandle.exists()
FabricFolderHandle FabricFolderHandle.getParentFolder()
FabricFolderHandle.getSubFolders(io FabricFolderHandle subfolders[])
FabricFolderHandle.getFiles(io FabricFileHandleWrapper files[])
```

The methods are self-explanatory and allow to walk the filesystem as well as access all FabricFileHandleWrappers of a given folder.

The FabricFileHandleWrapper provides functionality for using the FabricFileHandle, creating new handles based on file paths etc. The API looks as follows:

```
FabricFileHandleWrapper.setHandle(io String handle)
String FabricFileHandleWrapper.getHandle()
FabricFileHandleWrapper.setAbsolutePath(io String path)
String FabricFileHandleWrapper.getAbsolutePath()
FabricFolderHandle FabricFileHandleWrapper.getParentFolder()
String FabricFileHandleWrapper.getName()
String FabricFileHandleWrapper.getBaseName()
String FabricFileHandleWrapper.getExtension()
String FabricFileHandleWrapper.getExtensionLower()
Boolean FabricFileHandleWrapper.isValid()
Boolean FabricFileHandleWrapper.exists()
Boolean FabricFileHandleWrapper.isReadOnly()
Size FabricFileHandleWrapper.getSize()
```

4.2.5. Creating a FabricFileHandle for use in other extensions

You will need to convert an absolute path string to a FileHandle using the FabricFileHandleWrapper. Like so:

```
use FabricFILESYSTEM;

String path;
FabricFileHandleWrapper wrapper;
wrapper.setAbsolutePath(path);
myExtensionFunction(wrapper.getHandle())
```

This will allow you to implement a safe extension, that can work securely on the internet, but also perform on the local drive, if the user installs the options FabricFILESYSTEM extension.

4.2.6. Use of the FabricFILESTREAM

Along with the client resp. the node.js module you will find another File IO related extension, called FabricFILESTREAM. It is a wrapper for a std::ifstream resp. std::ofstream in KL, and can be created using a FabricFileHandle string, to allow to provide secure IO with stream functionality. The extension provides another KL type, called FabricFileStream. Its API looks like this:

```
FabricFileStream(String handle, String mode) // Mode can be 'r', 'w' or 'a' (append).
FabricFileStream.close()
FabricFileStream.closeOnFullyRead(Boolean close) // If set to true, the stream will close once
 all of its contents are read.
Boolean FabricFileStream.isValid()
Boolean FabricFileStream.isWritable()
Size FabricFileStream.getSize()
Size FabricFileStrean.getSizeRead()
Size FabricFileStream.getSeek()
FabricFileStream.setSeek(Size seek)
FabricFileStream.setSeekStart()
FabricFileStream.setSeekEnd()
FabricFileStream.writeData(Data data, Size size)
FabricFileStream.readData(io Data data, Size size)
FabricFileStream.writeString(String string)
FabricFileStream.readString(io String string)
FabricFileStream.writeStringArray(String strings[])
FabricFileStream.readStringArray(io String strings[])
FabricFileStream.writeSize(Size size)
FabricFileStream.readSize(io Size size)
```

As a sample, you could do something like this:

```
use FabricFILESYSTEM;
use FabricFILESTREAM;

String path;
FabricFileHandleWrapper wrapper;
wrapper.setAbsolutePath(path);

FabricFileStream outputStream;
outputStream.open(wrapper.getHandle(),"w")
Scalar scalars[];
scalars.push(1.0);
scalars.push(2.0);
scalars.push(3.0);
scalars.push(4.0);
scalars.push(5.0);
outputStream.writeSize(scalar.size())
outputStream.writeData(scalars.data(),scalars.dataSize())
outputStream.close();

FabricFileStream inputStream;
inputStream.open(wrapper.getHandle(),"r")
Size nbOfScalars;
inputStream.readSize(nbOfScalars);
scalars.resize(nbOfScalars);
inputStream.readData(scalars.data(),scalars.dataSize());
inputStream.close();

report(scalars);
```

Chapter 5. Debugging

If you have built the Fabric Engine plugin from source according to the Developer Install Guide, you can debug the build dlls that

5.1. Windows: Attaching the VisualStudio debugger

With Chrome:

1. Start Fabric (eg: open any Sample in Chrome)
2. Open Visual Studio
3. Choose Debug -> Attach To Process
4. Select the upper one which is Chrome, but which has no "[Low Rights]" in the user name column

With Firefox:

1. Do the same as above, but select "PluginContainer"
2. You need to turn off some obscure preferrence which will automatically kill an unresponsive plugin after a few seconds. I don't remember what it is; please update this page if you find it.