

Feydra: Getting Started

About

When creating a Sitecore web application, there are many dependencies between the tasks of front-end and back-end developers. Feydra tries to break these dependencies by allowing front-end developers to work on the Sitecore web application without needing to install and maintain an entire Sitecore development environment.

The Virtual Sandbox - How it works

Feydra allows any MVC web application to host front-end development by creating virtual sandboxes for each front-end developer. A virtual sandbox is a virtualized environment in which front-end assets (css, js, cshtml, etc.) can be selectively replaced without disturbing the back-end functionality of the web application. This virtualization allows back-end developers to create **stubs** for front-end functionality without having the actual front-end assets available at development time. The front-end developers can hook-up the components as they become available without impacting back-end development.

Feydra also greatly improves front-end developers' ability to maintain and update their portion of the web application. If there are issues with the front-end functionality of the site, a front-end developer can easily update the front-end assets in their virtual sandbox and test fixes against a working web application; all without the overhead of maintaining a full Sitecore development environment or MVC application.

Ultimately, Feydra is designed to allow front-end developers to use their preferred development environment; make their changes in a virtual sandbox and test by pushing their changes to a shared location (File share, FTP, etc.) using standard tools.

Installation with NuGet

Feydra is distributed as a .zip of NuGet packages. These packages can be added to a local NuGet feed to make them accessible to developers and the build process.

To install Feydra, simply add the NuGet package **Hedgehog.Feydra** to a web application and the Feydra assembly is automatically installed. If the developer is using Sitecore, add the NuGet package **Hedgehog.Feydra.Sitecore** to the web application.

The MVC core of Feydra consists of a single assembly called **Hedgehog.Feydra.dll**. If this file is present in the /bin folder of a website, Feydra will be available on that server. Deleting this file will remove Feydra.

The Sitecore components of Feydra consist of two additional files. These are **Hedgehog.Feydra.SC.dll** and **Z_Feydra.config**. These are installed in the **/bin** and **/App_Config/Include** folders, respectively.

Installation without NuGet

To install Feydra without using NuGet, extract the files from the NuGet packages and place the files in the folders specified in the table below:

File	Folder
Hedgehog.Feydra.dll	/bin
Hedgehog.Feydra.SC.dll	/bin
Z_Feydra.config	/App_Config/Include

Deployment

Feydra needs to be deployed on the servers where the front-end developers are going to be updating front-end assets. This can be any server capable of running the web application. We recommend using a server that is not used for CI builds, since the build/deployment process will interfere with the front-end developer testing.

Feydra doesn't support load balanced servers. Load balanced servers are typically not needed in a development environment, so this should not be a major problem.

The Feydra assemblies and configuration files should not be deployed to a production server.

Setup

Once Feydra is installed on a server, Feydra will verify the environment and display error messages indicating any problems found on the server that would prevent Feydra from functioning correctly. The following is a list of requirements Feydra will check at startup:

- There is a folder called **FeydraRoot** in the root of the website. If the folder doesn't exist, Feydra will try to add it. Feydra will report a problem if the folder cannot be created.
- The Web Server needs write rights to **FeydraRoot**.
- There must be a folder called **/App_Data/Feydra**. If the folder doesn't exist, Feydra will try to add it. Feydra will report an error if the folder cannot be created.
- The Web Server needs write rights to **/App_Data/Feydra** and all files in that folder.
- The Web Server needs write rights to a folder called **/Areas/Feydra**.
- The property **runAllManagedModulesForAllRequests** must be set on the **/configuration/modules** element in the **web.config**.

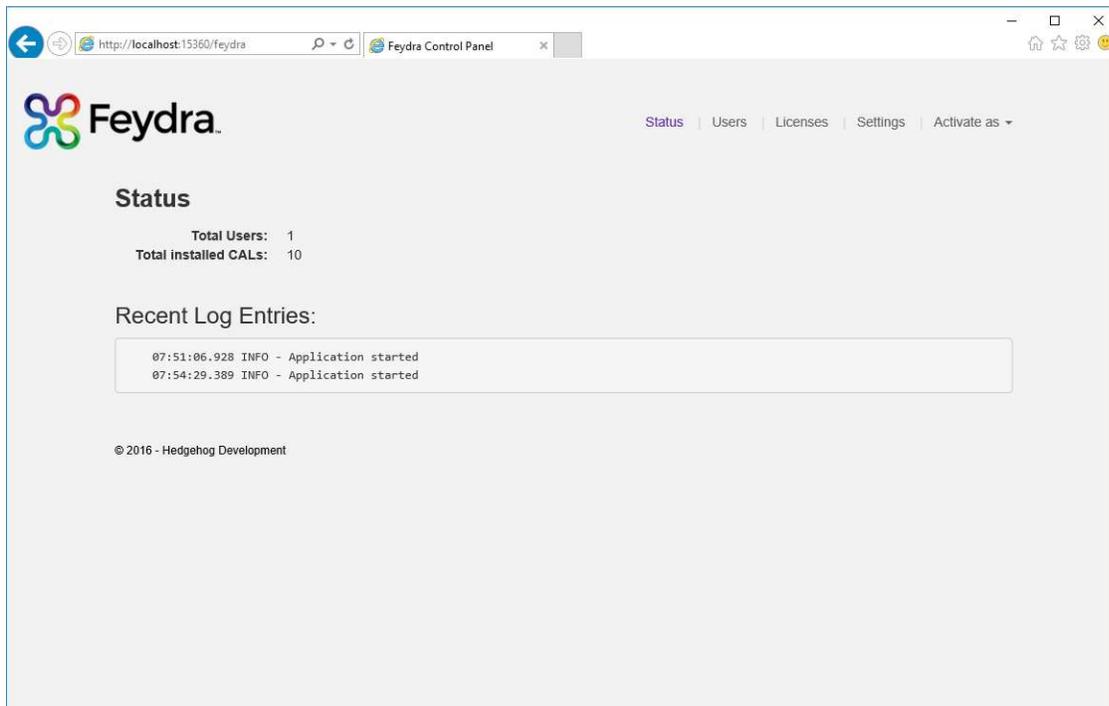
Using Feydra

A front-end developer needs to be able to copy files to the web server to work with the website. There are a number of ways this can happen (Unc path, remote access, FTP, etc.). The choice of the technology for copying files to the server is left up to the end user.

The Feydra Dashboard

Feydra is controlled through the Feydra dashboard. The Feydra dashboard is accessed by the url [http://\[WebServer\]/Feydra](http://[WebServer]/Feydra) where [WebServer] is the domain name of the web server. The dashboard has the following functions:

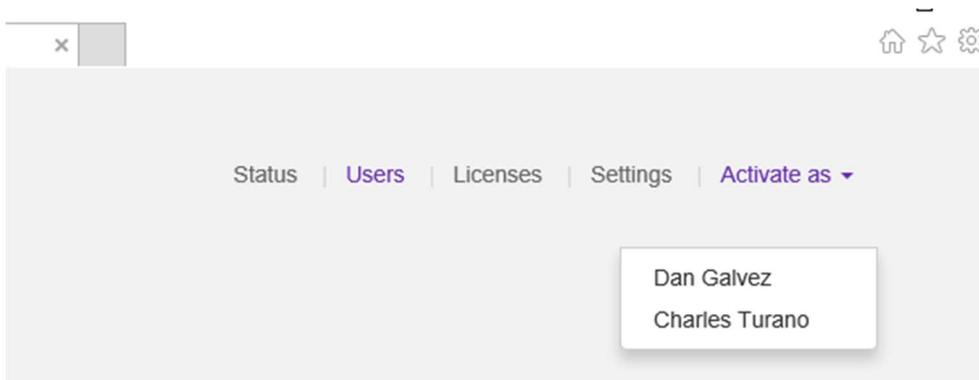
- **Status** – Shows the number of users, licenses and the latest log entries.
- **Users** – Allows users to be added and removed. Users can also obtain a link to activate Feydra.
- **Licenses** – Allows the licenses for Feydra to be updated. Viewing and adding licenses can only be performed on the local server. Remote viewing of the license screen is not permitted.
- **Settings** – Allows some of the settings controlling Feydra to be updated. The initial files a user sees when their username is created can be updated here.



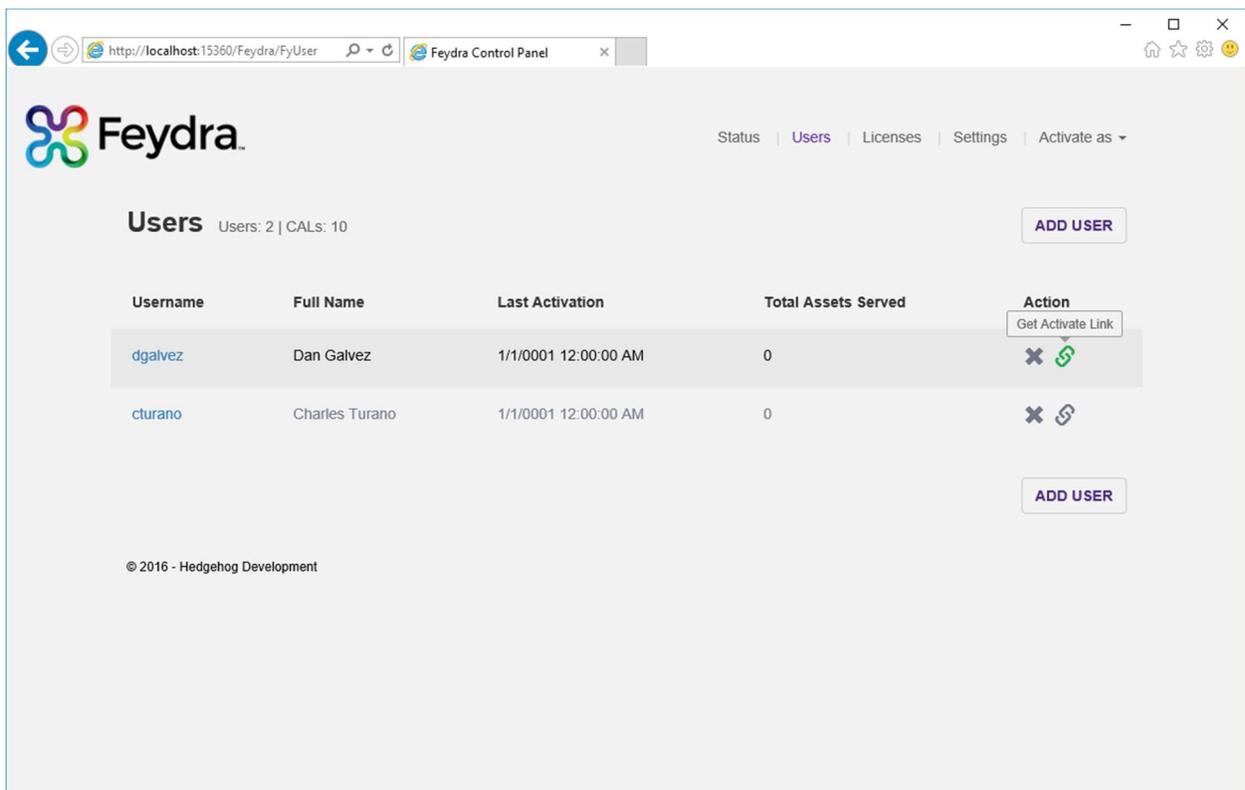
There is a dropdown link in the menu bar of the website that will allow a user to activate Feydra for any of the registered users.

Activating Feydra

The web application behaves normally when the front-end developer has not activated Feydra. To activate Feydra, the user can open the dashboard and use the dropdown link in the menu bar to activate Feydra by selecting their username.



The user can also obtain an activation link from the users page by clicking on the “🔗” icon in the row for the user.



Updating the front-end assets

When the front-end developer activates Feydra, they will see a copy of the application running application. We call this the users’ virtual sandbox. This virtual sandbox uses all of the deployed front-end assets (cshtml, js, css, etc...) to run the application. If the user copies a front-end asset into their personal folder under **/FeydraRoot**, the front-end asset from the personal folder replaces the file of the same name and relative path in the deployed web application.

An example of this is a web application with the following folder structure:

```
[WebRoot]
├── Areas
│   └── Products
│       ├── Index.cshtml
│       └── View.cshtml
├── Include
│   ├── css
│   │   └── site.css
│   └── js
│       └── site.js
```

When User1 is created, a personalized sandbox folder for that user will be created under the **FeydraRoot** folder:

```
[WebRoot]
├── FeydraRoot
│   └── User1
```

Now, User1 can use their personal Feydra activation link to activate Feydra. The web application will look exactly as it did if Feydra isn't activated since they have no files in their sandbox.

If the user creates a file called **Index.cshtml** in **/FeydraRoot/User1/Area/Products**, the file will override the **Index.cshtml** file used to render the products page on the website. This allows the front-end developer to update the front-end functionality of the web application, while allowing other parts of the application to function with existing front-end assets.

The user may also update the **Site.js** file in the folder **/Include/js** by placing a new Site.js file in **/FeydraRoot/User1/Include/js**. The new Site.js will be sent to the browser in place of the existing one when the page is refreshed.

If there are other users of the site who have not activated as User1, they will not see any of the changes made by User1 until the changes are deployed as part of the normal deployment process. Additional users may be created to allow multiple developers to work with the website. Each developer will only see the changes in their virtual sandbox while Feydra is activated.

If the front-end developer views the page source when Feydra is activated, they will see comments around various parts of the code indicating where the .cshtml files exist on the file system.

```

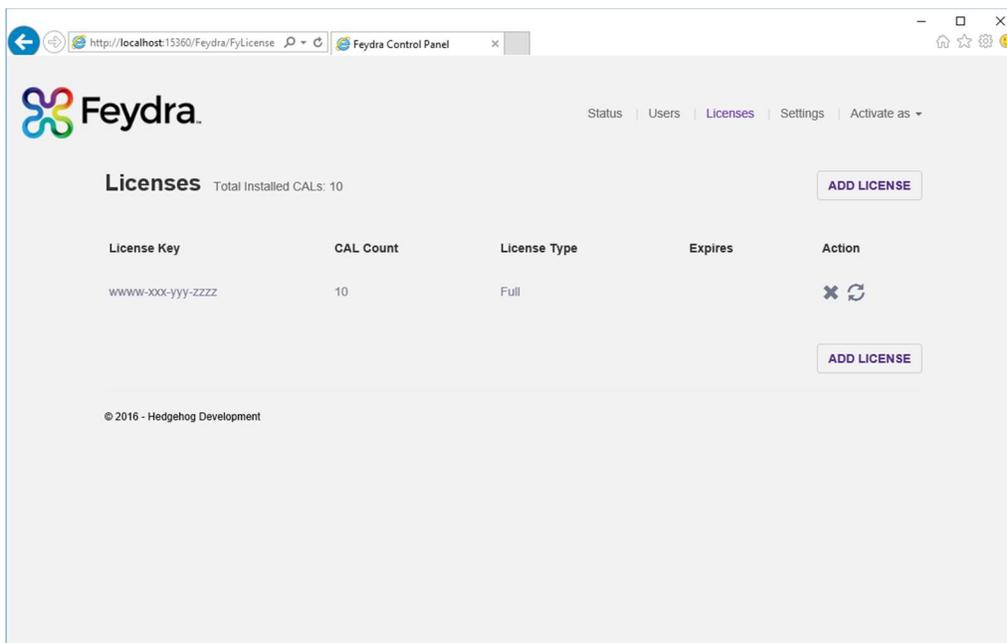
562 <!-- Feydra View End ~/Areas/TDS/Views/Content/VideoCarousel.cshtml -->
563 <!-- Feydra View Begin /Areas/TDS/Views/Callouts/VerticalCalloutContainer.cshtml -->
564 <div class="row">
565     <section class="small-12 columns vertical-container">
566         <div class="body-content">
567             <div class="row">
568                 <div class="small-12 columns">
569                     <h4>Licensing</h4>
570
571                 <div class="row" data-equalizer="description">
572                     <!-- Feydra View Begin ~/Areas/TDS/Views/Callouts/VerticalCallout.cshtml -->
573
574 <div class="small-12 medium-4 large-4 columns vertical-callout">

```

This will provide hints to the front-end developer for manipulating various components on the page.

Adding Licenses

Licenses are added to Feydra using the license management screen. This screen may only be accessed when logged into the console of the local server.



Feydra will not allow users to activate their virtual sandbox if there aren't enough CAL's for all of the created users.

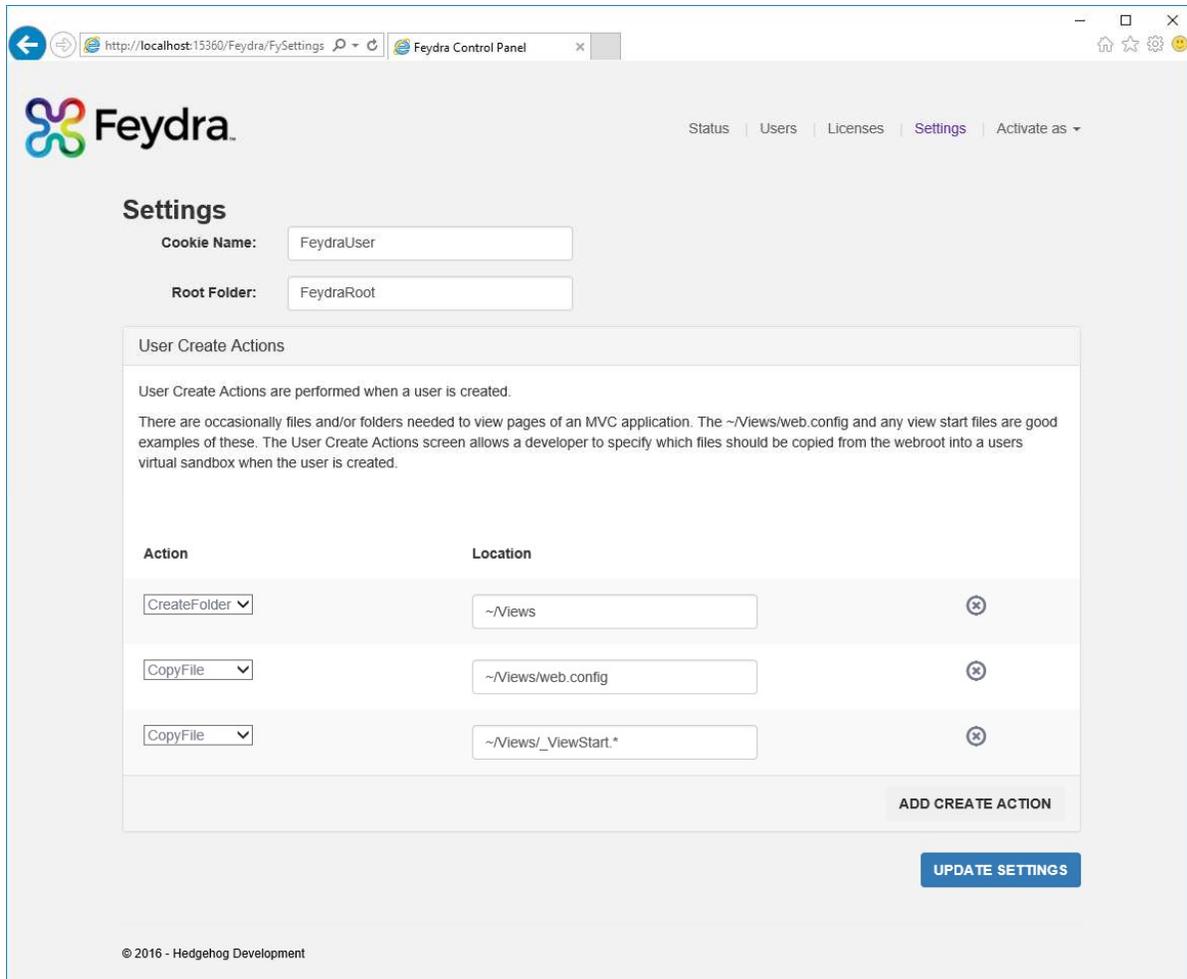
The user can add new license keys by selecting **Add License**.

The “” (Refresh) action will contact the Hedgehog License Server and update the number of CAL's (Client Access License) or license type if it has changed.

The “” (Delete) action will remove a license and its associated CAL's from the server.

Feydra Settings

The Feydra settings screen allows the developer to view and/or change the settings Feydra uses on the server.



The screenshot shows the Feydra Settings web interface. At the top, there is a navigation bar with the Feydra logo and links for Status, Users, Licenses, Settings (highlighted), and Activate as. Below the navigation bar, the Settings section is displayed. It includes two input fields: 'Cookie Name' with the value 'FeydraUser' and 'Root Folder' with the value 'FeydraRoot'. Below these is a section titled 'User Create Actions' which contains a text block explaining that these actions are performed when a user is created and lists examples of files and folders needed for an MVC application. Below the text is a table with two columns: 'Action' and 'Location'. The table contains three rows: 'CreateFolder' with location '~Views', 'CopyFile' with location '~Views/web.config', and 'CopyFile' with location '~Views/_ViewStart.*'. At the bottom right of the table is an 'ADD CREATE ACTION' button. Below the table is an 'UPDATE SETTINGS' button. At the bottom left of the page, there is a copyright notice: '© 2016 - Hedgehog Development'.

Action	Location
CreateFolder	~/Views
CopyFile	~/Views/web.config
CopyFile	~/Views/_ViewStart.*

To use this screen, simply make the changes needed to the settings and click “Update Settings”

The settings the user can update are as follows:

- Cookie Name – The name of the cookie Feydra uses to maintain the current user activation.
- Root Folder – The name of the Feydra root folder. If this is changed after users are created, the developer is responsible for moving the users files into the new location.
- User Create Actions – This allows the developer to specify files to copy (relative to the web root) into each virtual sandbox when the user is created. This is useful for setting up files like the **/Views/web.config**, since these files are needed by MVC to build the views correctly.

Conclusion

Feydra supports multiple user sandboxes, allowing multiple front-end developers to work with a single web application. This will dramatically reduce the costs of maintaining development and test environments for front-end developers while allowing them to complete their task more efficiently.