CFDSource and GITHub Integration

This document describes how to integrate CFDSource with GITHub.

GIT, Caché and Users

The important this to remember is that integration with GIT is done from the machine where Caché/Ensemble are installed/running from, and further done under the user that is running Caché/Ensemble (which by default is 'cacheusr' on linux, and the default system account on Windows).

> **Note:** It is irrelevant where you are connecting from for GIT integration. Your PC and your OS user are not known, seen or used when communicating with GIT.

You work in a namespace, and it is the namespace that is connected to source control. So you could have several people working in the same namespace, but all their changes would be together in each feature.

> **Note:** If this seems like a limitation, it is not. It actually is a potential source of great power! Pair programming over long distances is possible because the code is not on files, but in a database. You want your own repository, create your own namespace. You could even use Caché security to lock others out of it!

Preparing for GIT

In order to access GIT via SSH we need to have a public/private key pair setup for the OS user Caché/Ensemble are running as. Let's say this is cacheusr, here is how this could be generated:

> $>su –cacheusr
> $>ssh-keygen –b 2048 –C {Your GIT Email ID}
> Note: Accept defaults and enter no passphrase

You then need to install the public key on GITHub under your account (or the account under which GIT integration will be done). Do not worry about the email address, when a project is created or cloned via CFDSource it will setup the user/email and that can be specific to each namespace.

The private key should be protected. It is in ~/.ssh/id_rsa and should not leave the server. If it is someone can communicate with GITHub as you. See permissions on the file as you deem appropriate.

> Note: On windows it is similar and ssh-keygen is also available. If you have installed git bash then you will have it. Just make sure you install the certificates in the .ssh directory under the user, who is running Caché/Ensemble, home directory. I have tested this on Linux.

You can test the SSH key and communication with GITHub via:

> $>ssh -T git@github.com

There is no point is using GIT integration via CFDSource if this does not work!

Adding SSL Configuration

The final step is we need to add a SSL configuration to our Caché/Ensemble server. To do this go to System Administration > Security > SSL/TLS Configurations in the Management Portal. In here create a new Client configuration called “GITHub” (the name, minus the “ is important and hard coded into CFDSource at present). Leave all settings default/blank:

- Enabled: true
- Type: client
- Peer Level: None
- CA: <blank>
- Client Certificate: <blank>
- Client Key: <blank>
- Type: RSA
- Protocols: SSLv3, TLS
- Cipher: <default>

You can of course make this more specific if you wish. I recommend testing this configuration against api.github.com on port 443.

Using with GITHub Repositories

The process for actually using a project with GITHub is exactly the same as without. The only real different is on creating or importing the project in the first instance.

On Import (clone)

The repository URL should be a ssh based GIT url. For instance git@github.com:thegaffer/CFDSource.git for CFDSource itself.

On Create

When creating the project CFDSource will create the GITHub repository and check into it. It does this using the public GITHub restful API (and once created GIT). To do this it needs 3 pieces of information:

1. Your GIT user, i.e. thegaffer in the above example
2. Your GIT email address, i.e. the email associated with thegaffer (also in the comments of the ssh key)
3. Your GIT password

Note that these details are not stored in the Caché Database and are only transitively held whilst filling in the form. They are sent between your client and the Caché server (CSP), so it is best not to do this creation when connected to Caché over a public network.

Once cloned or created, as long as the SSH private key is setup as described for the (OS) user running Caché then pushes and pulls will be done as with file base repositories.