WilHall / PlunkerInstallationGuide.md

Last active a month ago

Plunker Installation Guide

apache-vhost-example.conf

```
<VirtualHost *:80>
    ServerAdmin admin@example.com
    ServerName plunk.example.com
    ProxyPreserveHost On

    <Proxy *>
        Order allow,deny
        Allow from all
    </Proxy>
    ProxyPass / http://localhost:8000/
    ProxyPassReverse / http://localhost:8000/
</VirtualHost>
```

config.api.example.json

```
{
  "host": "plunk.example.com",
  "url": {
    "www": "http://plunk.example.com",
    "collab": "http://collab.example.com:8001",
    "api": "http://api.example.com:8002",
    "embed": "http://embed.example.com:8003",
    "run": "http://run.example.com:8004",
    "carbonadsH": "You can probably ignore this",
    "carbonadsV": "...and this"
  },
  "PORT": 8002,
  "oauth": {
    "github": {
      "id": "YOUR_CLIENT_ID",
      "secret": "YOUR_CLIENT_SECRET"
    }
  },
  "mongodb": {
    "host": "localhost",
    "port": "27017",
    "auth": "DB_USERNAME:DB_PASSWORD",
    "pathname": "/plunker"
  }
}
```

config.collab.example.json

```
{
  "host": "plunk.example.com",
  "url": {
    "www": "http://plunk.example.com",
    "collab": "http://collab.example.com:8001",
    "api": "http://api.example.com:8002",
    "embed": "http://embed.example.com:8003",
    "run": "http://run.example.com:8004",
    "carbonadsH": "You can probably ignore this",
    "carbonadsV": "...and this"
  },
  "PORT": 8001,
```

```
  "oauth": {
    "github": {
      "id": "YOUR_CLIENT_ID",
      "secret": "YOUR_CLIENT_SECRET"
    }
  }
}
```

config.embed.example.json

```
{
  "host": "plunk.example.com",
  "url": {
    "www": "http://plunk.example.com",
    "collab": "http://collab.example.com:8001",
    "api": "http://api.example.com:8002",
    "embed": "http://embed.example.com:8003",
    "run": "http://run.example.com:8004",
    "carbonadsH": "You can probably ignore this",
    "carbonadsV": "...and this"
  },
  "PORT": 8003,
  "oauth": {
    "github": {
      "id": "YOUR_CLIENT_ID",
      "secret": "YOUR_CLIENT_SECRET"
    }
  }
}
```

config.run.example.json

```
{
  "host": "plunk.example.com",
  "url": {
    "www": "http://plunk.example.com",
    "collab": "http://collab.example.com:8001",
    "api": "http://api.example.com:8002",
    "embed": "http://embed.example.com:8003",
    "run": "http://run.example.com:8004",
    "carbonadsH": "You can probably ignore this",
    "carbonadsV": "...and this"
  },
  "PORT": 8004,
  "oauth": {
    "github": {
      "id": "YOUR_CLIENT_ID",
      "secret": "YOUR_CLIENT_SECRET"
    }
  }
}
```

config.www.example.json

```
{
  "host": "plunk.example.com",
  "url": {
    "www": "http://plunk.example.com",
    "collab": "http://collab.example.com:8001",
    "api": "http://api.example.com:8002",
    "embed": "http://embed.example.com:8003",
    "run": "http://run.example.com:8004",
    "carbonadsH": "You can probably ignore this",
    "carbonadsV": "...and this"
  },
  "PORT": 8000,
  "oauth": {
```

```
    "github": {
      "id": "YOUR_CLIENT_ID",
      "secret": "YOUR_CLIENT_SECRET"
    }
  }
}
```

PlunkerInstallationGuide.md

Plunker Installation Guide

This installation guide walks through the process of installing Plunker and all its components, assuming they are all running on the same server. Tested on Ubuntu 12 LTS.

Plunker Components

Plunker is made up of several component repositories (taken from the plunker documentation):

- plunker_api The server that connects to a mongodb database and serves requests over a restful api.
- plunker_www The server that is responsible for hosting and running the front-end that users see and touch everyday.
- plunker_run The server that allows for previewing of plunks and temporary previews and also does the dynamic transpilation.
- plunker_collab The server that serves the code necessary for collaborative coding as well as doing the actual operational transformation over a browserchannel connection.
- plunker_embed The server that hosts the embedded views of plunks.

Some components require special attention; this guide will walk you through a successfull install of all components, assuming the following versions:

- plunker_api@9370605fa6d73db18a450cb11e02a730633d9b02
- plunker_www@4785ffd0f95450effd0ce8062bce7f22a6f3318b
- plunker_run@fd72c304eff55a7b9726803883bd9f9eb0798409
- plunker_collab@22d82620fa7541858533f96394ac521b00043229
- plunker_embed@c56e3b4aea283d6bed9314dc40c33e869c52f674

##Installation

Ubuntu Prerequisites

Install NodeJS v0.10.22 and NPM

```
sudo apt-get install python-software-properties python g++ make
sudo add-apt-repository ppa:chris-lea/node.js
sudo apt-get update
sudo apt-get install nodejs
```

###Plunker Configuration Files Each plunker component uses its own configuration file, usually `config.production.json` or `config.development.json` depending on the environment. The environment is defined by the `process.env.NODE_ENV` towards the top of the `server.js` file of each component.

Below is a table of the configuration files required for each component, and reference to examples attached to this gist:

Component	Config Filename	Example Filename
plunker_api	`config.<env>.json`	`config.api.example.json`
plunker_www	`config.<env>.json`	`config.www.example.json`
plunker_run	`config.<env>.json`	`config.run.example.json`
plunker_collab	`config.<env>.json`	`config.collab.example.json`
plunker_embed	`config.json`	`config.embed.example.json`

Currently,there is a problem where the `"port"` configuration value is lowercase in the example configuration files, but should be uppercase `"PORT"` . Be sure to change this, or the port value set in the configuration files will not take effect (See Issue #58).

The configuration file values for the URLs for each Plunker component should be duplicated (and identical) between all the configuration files, but the `"PORT"` should be changed to reflect each component.

###Installing plunker_api

First, clone the repository to the specified version:

```
git clone https://github.com/filearts/plunker_api.git
cd plunker_api
git checkout 9370605fa6d73db18a450cb11e02a730633d9b02
```

Next, install node dependencies:

```
npm install
```

Install and configure mongodb:

```
sudo apt-get install mongodb
mongo
> use plunker
> db.addUser("DB_USERNAME", "DB_PASSWORD")
> exit
```

Be sure to modify the configuration file settings for the plunker_api configuration file to match your mongodb settings.

###Installing plunker_www First, clone the repository to the specified version:

```
git clone https://github.com/filearts/plunker_www.git
cd plunker_www
git checkout 4785ffd0f95450effd0ce8062bce7f22a6f3318b
```

Next, install node dependencies:

```
npm install
```

###Installing plunker_run First, clone the repository to the specified version:

```
git clone https://github.com/filearts/plunker_run.git
cd plunker_run
git checkout fd72c304eff55a7b9726803883bd9f9eb0798409
```

Next, install node dependencies:

```
npm install
```

###Installing plunker_collab First, clone the repository to the specified version:

```
git clone https://github.com/filearts/plunker_collab.git
cd plunker_collab
git checkout 22d82620fa7541858533f96394ac521b00043229
```

Edit `package.json` and change the ShareJS dependency tarball to point to version 0.6:
https://github.com/share/ShareJS/archive/v0.6.0.tar.gz

Install node dependencies:

```
npm install
```

###Installing plunker_embed First, clone the repository to the specified version:

```
git clone https://github.com/filearts/plunker_embed.git
cd plunker_embed
git checkout c56e3b4aea283d6bed9314dc40c33e869c52f674
```

Install node dependencies:

```
npm install
```

Note: plunker_embed looks for `config.json` NOT `config.<env>.json`

##Deployment ###Enabling Github Authentication

Visit the Github Application Settings page and create a new application to represent your installation of Plunker. Your callback URL is `/auth/github` ; the full value should look something like: `http://plunk.example.com/auth/github`

Take note of your `Client ID` and `Client Secret` , and distribute these values into all of your Plunker configuration files.

Note: All Set; but you may need to restart Plunker.

###Deploying under Apache2 using mod-proxy

This is my preferred method of deployment, and the only one I have tested with.

First, enable mod_proxy:

```
sudo a2enmod proxy_http
```

Next, create a new apache `VirtualHost` configuration for your Plunker installation (use the attached `apache-vhost-example.conf` as a template).

Then restart apache:

```
sudo service apache2 restart
```

Plunker should now be accessible from behind apache.

###Running Plunker On Startup The most straightforward way to run Plunker on startup is using init.d scripts. For node.js applications, I highly recommend using `initd-forever`:

```
sudo npm install forever -g
sudo npm install initd-forever -g
```

And to generate an `init.d` script for each plunker component, just run:

```
initd-forever -a /path/to/your/plunker_www/server.js
. . .
initd-forever -a /path/to/your/plunker_embed/server.js
```

Depending on your environment, this may give you some trouble. You may need to reference the full path to the `forever` binary (see output of `which forever`), and/or `cd` into the `node.js` application path.

For an example of my configuration, see the attached `upstart_example.sh`.

###Unknown Server Error for Thumbnails Plunker generates webpage thumbnails using the immediatenet thumbnail API, which doesn't support URLs containing port numbers (Ex. `run.example.com:8004`). You may see this error:

Thumbnails are generated from the `plunker_run` component, which in this tutorial is running at `run.example.com:8004`. In order to make use of the immediatenet thumbnail API, another Apache `VirtualHost` can be added so that `ProxyPass` directs all requests to `run.example.com` to `run.example.com:8004`.

upstart_example.sh

```
#!/bin/bash
#
# initd a node app
# Based on a script posted by https://gist.github.com/jinze at https://gist.github.com/3748766
#

# Source function library.
. /lib/lsb/init-functions

pidFile=/var/run/plunker-www.pid
logFile=/var/run/plunker-www.log
```

```
12
13  nodePath=/path/to/plunker/plunker_www
14  nodeApp=$nodePath/server.js
15
16  start() {
17          echo "Starting $nodeApp"
18          cd $nodePath
19          # This is found in the library referenced at the top of the script
20          start_daemon
21
22          # Notice that we change the PATH because on reboot
23     # the PATH does not include the path to node.
24     # Launching forever with a full path
25     # does not work unless we set the PATH.
26     PATH=/usr/local/bin:$PATH
27          export NODE_ENV=development
28     #PORT=80
29     /usr/bin/forever start --pidFile $pidFile -l $logFile -a -d $nodeApp
30     RETVAL=$?
31  }
32
33  restart() {
34          echo -n "Restarting $nodeApp"
35          /usr/bin/forever restart $nodeApp
36          RETVAL=$?
37  }
38
39  stop() {
40          echo -n "Shutting down $nodeApp"
41     /usr/bin/forever stop $nodeApp
42     RETVAL=$?
43  }
44
45  status() {
46     echo -n "Status $nodeApp"
47     /usr/bin/forever list
48     RETVAL=$?
49  }
50
51  case "$1" in
52     start)
53          start
54          ;;
55       stop)
56          stop
57          ;;
58     status)
59          status
60         ;;
61     restart)
62          restart
63          ;;
64          *)
65         echo "Usage:  {start|stop|status|restart}"
66         exit 1
67          ;;
68  esac
69  exit $RETVAL
```

ggoodman commented on Nov 28, 2013

I've recently changed the structure a little bit with the latest commits in plunker_www. I've decided to remove the separation of plunker_embed from plunker_www and now the embed server is a hostname remap from embed.plnkr.co/ to plnkr.co/embed/. This allows me to rationalize one server (and a server that doesn't suffer nearly the same load as either the api or run servers).

How and where have you deployed plunker? How are you using it and how well is it meeting your needs?