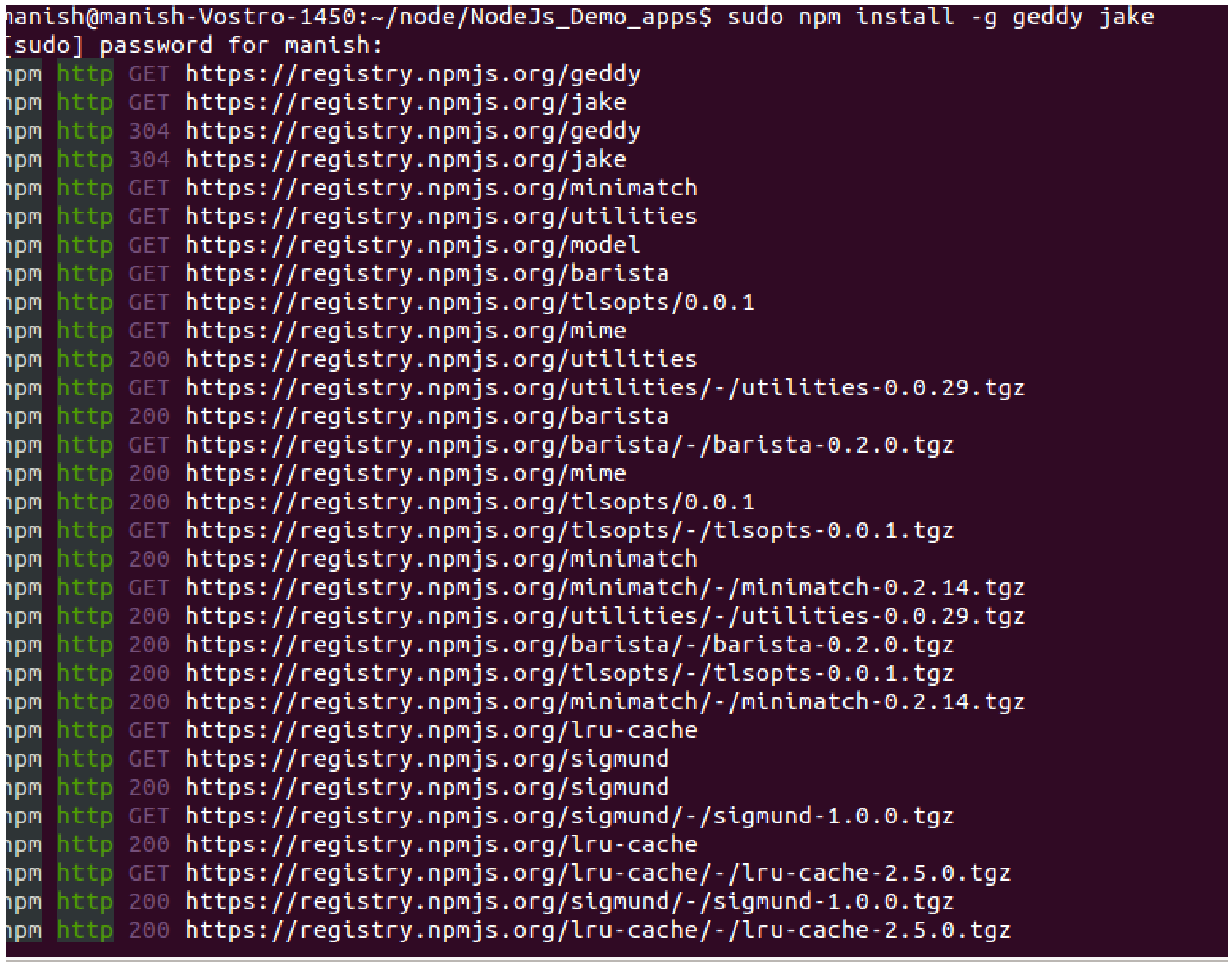
Installing Geddy with npm

Jake is a JavaScript build program for Node.js.

- Open up your terminal
- type `npm install -g geddy jake`



```
anish@manish-Vostro-1450:~/node/NodeJs_Demo_apps$ sudo npm install -g geddy jake
[sudo] password for manish:
npm http GET https://registry.npmjs.org/geddy
npm http GET https://registry.npmjs.org/jake
npm http 304 https://registry.npmjs.org/geddy
npm http 304 https://registry.npmjs.org/jake
npm http GET https://registry.npmjs.org/minimatch
npm http GET https://registry.npmjs.org/utilities
npm http GET https://registry.npmjs.org/model
npm http GET https://registry.npmjs.org/barista
npm http GET https://registry.npmjs.org/tlsopts/0.0.1
npm http GET https://registry.npmjs.org/mime
npm http 200 https://registry.npmjs.org/utilities
npm http GET https://registry.npmjs.org/utilities/-/utilities-0.0.29.tgz
npm http 200 https://registry.npmjs.org/barista
npm http GET https://registry.npmjs.org/barista/-/barista-0.2.0.tgz
npm http 200 https://registry.npmjs.org/mime
npm http 200 https://registry.npmjs.org/tlsopts/0.0.1
npm http GET https://registry.npmjs.org/tlsopts/-/tlsopts-0.0.1.tgz
npm http 200 https://registry.npmjs.org/minimatch
npm http GET https://registry.npmjs.org/minimatch/-/minimatch-0.2.14.tgz
npm http 200 https://registry.npmjs.org/utilities/-/utilities-0.0.29.tgz
npm http 200 https://registry.npmjs.org/barista/-/barista-0.2.0.tgz
npm http 200 https://registry.npmjs.org/tlsopts/-/tlsopts-0.0.1.tgz
npm http 200 https://registry.npmjs.org/minimatch/-/minimatch-0.2.14.tgz
npm http GET https://registry.npmjs.org/lru-cache
npm http GET https://registry.npmjs.org/sigmund
npm http 200 https://registry.npmjs.org/sigmund
npm http GET https://registry.npmjs.org/sigmund/-/sigmund-1.0.0.tgz
npm http 200 https://registry.npmjs.org/lru-cache
npm http GET https://registry.npmjs.org/lru-cache/-/lru-cache-2.5.0.tgz
npm http 200 https://registry.npmjs.org/sigmund/-/sigmund-1.0.0.tgz
npm http 200 https://registry.npmjs.org/lru-cache/-/lru-cache-2.5.0.tgz
```

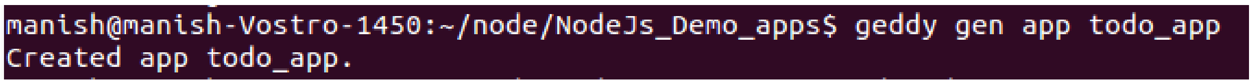
Generating a Geddy App

Geddy uses a global executable to generate apps/resources, and to start up your app server. This will all take place on the command line, so open up your terminal again. Before we generate our app, let's `cd` to a good location to store your app. This can be anywhere on your machine, though, I prefer to do my development in my `~/dev/` directory.

```
cd path/to/the/place/you/code
```

Next, we'll use `geddy` to generate our app structure. We'll be creating a to-do application, so we'll call ours,`todo_app`

```
geddy gen app todo_app
```



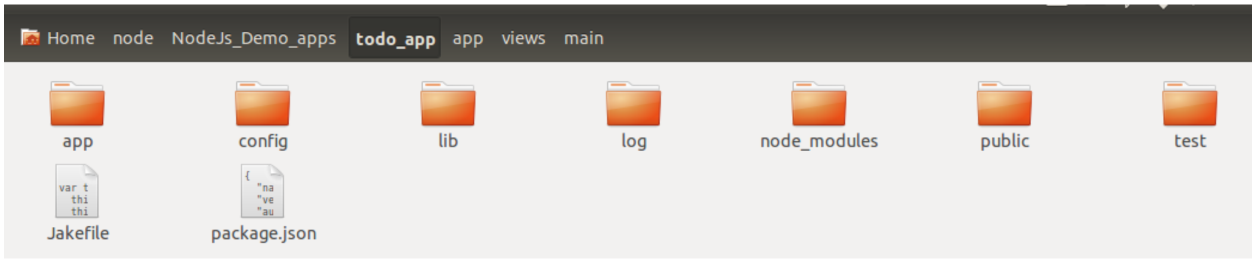
```
manish@manish-Vostro-1450:~/node/NodeJs_Demo_apps$ geddy gen app todo_app
Created app todo_app.
```

All done. Now what did that do for us?

An Overview of Our Generated App

If you take a look within the newly created `todo_app` directory, you'll see that Geddy has generated a fair bit of code for you. Your directory structure should look a bit like this:

- app/
    - controllers/
    - models/
    - views/
- config/
- lib/
- log/
- node_modules/
- public/



Let's step through these one by one:

app: Here's where most of the magic happens. Most of your app's logic will be located in one of the three directories contained in this one.

app/controllers: All of your app's controllers (the part that ties your models to your views) go here. You'll also notice that there's already two controller files in there: `application.js` (which all controllers inherit from) and `main.js` (the controller that ties your `/` route to your `app/views/main/index.html.ejs`template).

app/models: Here's where you'll be storing your models – there's nothing in there yet, but we'll adding one in during the next tutorial.

app/views: All of your app's templates are stored here. For now, you'll see that you have an`application.html.ejs` file in the `layouts` directory – this file is the main template for your app, all of your front-end wrapper code should go in here. You should also have an `index.html.ejs` file in the `main`directory. This is what get's rendered by the main controller's `index` action when you hit the `/` route.

config: The configuration files for your app goes here. You should have the `development.js`,`production.js`, `environment.js`, `router.js` and `init.js` files in there. The `init.js` file is a file that runs just after the app gets started, before any requests come in. This can be used to add functions and properties that need to be app-wide. The `router.js` file is used to create routes for your application. Routes tie URLs to controller actions. For global settings, you'll want to edit the `environment.js` file. For production and development settings, edit the corresponding config files.

lib: This is the place where you can put any file's that you'd like to use all over your app.

log: All of your logs will be stored here. You should get an `access.log`, a `stdout.log`, and a `stderr.log`after you run your app.

node_modules: This is where the modules that you install will be stored. Think of it as a lib for other people's code.

public: Finally, here's where all of your front end specific stuff will live. All you css files, images, and front-end js files will be in here. You’ll notice that Twitter’s bootstrap and jQuery come pre-packaged with all Geddy apps.

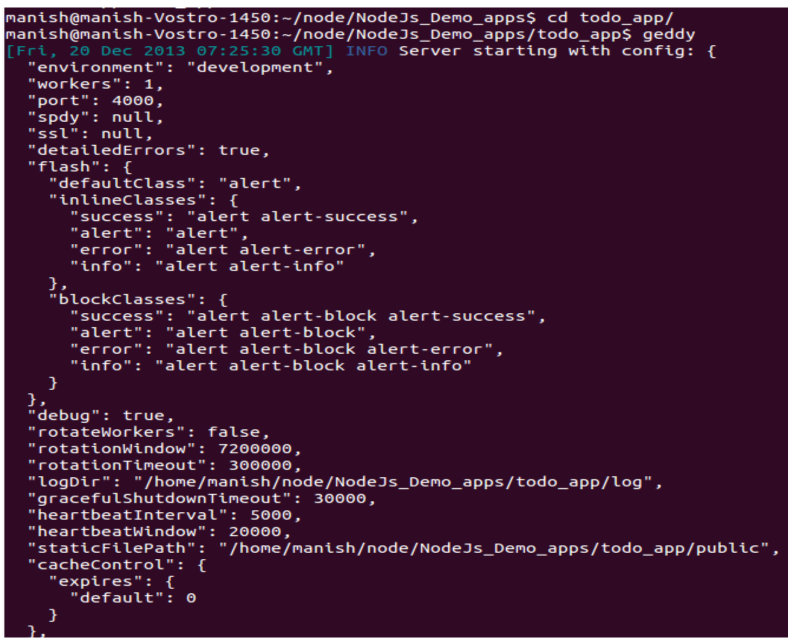
Starting Up Your New Geddy App

Now that we have an app generated, I’ll demonstrate how to start it up. First, open the terminal again, and navigate to your app’s directory:

```
cd ~/path/to/code/todo_app
```

Once you’re there, start the app up by using the `geddy` command:

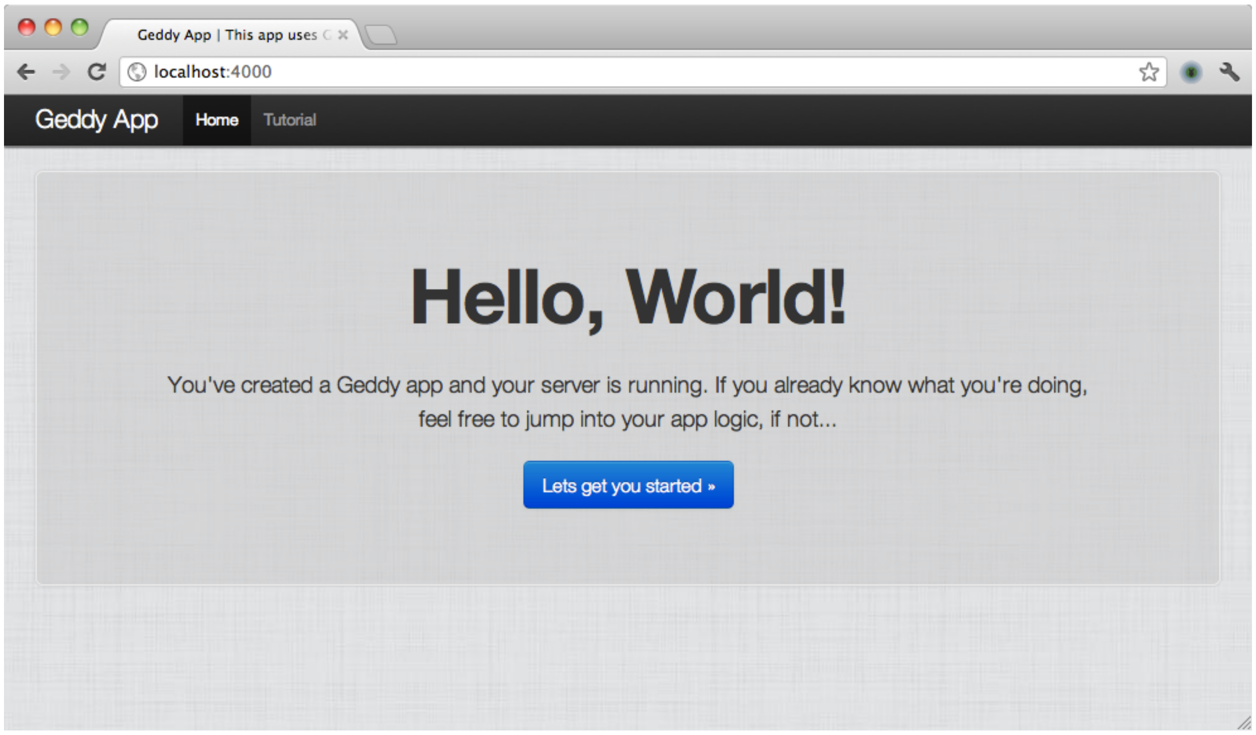
```
geddy
```



```
manish@manish-Vostro-1450:~/node/NodeJs_Demo_apps$ cd todo_app/
manish@manish-Vostro-1450:~/node/NodeJs_Demo_apps/todo_app$ geddy
[Fri, 20 Dec 2013 07:25:30 GMT] INFO Server starting with config: {
  "environment": "development",
  "workers": 1,
  "port": 4000,
  "spdy": null,
  "ssl": null,
  "detailedErrors": true,
  "flash": {
    "defaultClass": "alert",
    "inlineClasses": {
      "success": "alert alert-success",
      "alert": "alert",
      "error": "alert alert-error",
      "info": "alert alert-info"
    },
    "blockClasses": {
      "success": "alert alert-block alert-success",
      "alert": "alert alert-block",
      "error": "alert alert-block alert-error",
      "info": "alert alert-block alert-info"
    }
  },
  "debug": true,
  "rotateWorkers": false,
  "rotationWindow": 7200000,
  "rotationTimeout": 300000,
  "logDir": "/home/manish/node/NodeJs_Demo_apps/todo_app/log",
  "gracefulShutdownTimeout": 30000,
  "heartbeatInterval": 5000,
  "heartbeatWindow": 20000,
  "staticFilePath": "/home/manish/node/NodeJs_Demo_apps/todo_app/public",
  "cacheControl": {
    "expires": {
      "default": 0
    }
  },
```

You should see some output that looks a bit like this:

Now that we’ve started up the server, go ahead and check it out in browser. Visit`http://localhost:4000`, and take a look!

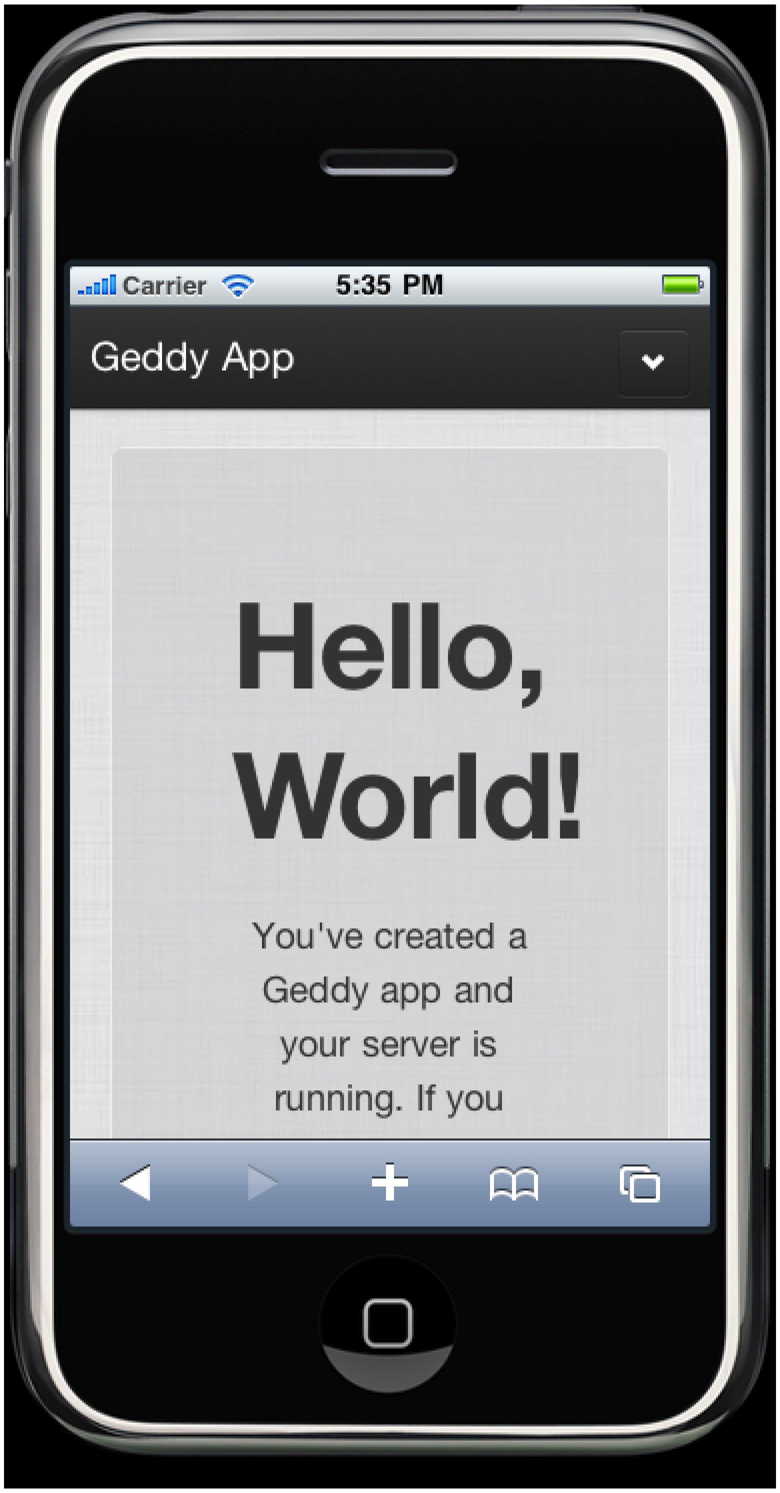


Hello, World!

You've created a Geddy app and your server is running. If you already know what you're doing, feel free to jump into your app logic, if not...

***Bonus**: Because Geddy uses Bootstrap out of the box, with it's responsive layout enabled, our app will immediately display nicely in a mobile browser. Resize your browser window to verify this.*

This concludes the first part of our tutorial series on Node.js and Geddy. :)



Hello, World!

You've created a Geddy app and your server is running. If you