Commandline interface to use puer https://github.com/leeluolee/puer with Freemarker templates http://freemarker.org/.

Usage

Install it either as a global command line tool or as a local development dependency:

```
npm install -g puer-freemarker-cli
or
npm install --save-dev puer-freemarker-cli
```

Move into your working directory and run it:

```
cd your/working/directory
puerf
```

puerF requires that you have Java installed as it is needed to render Freemarker templates.

Command reference

```
Usage: puerf [cmd] [options]

Commands:

    init [options]   Set up basic folders and files tow ork with puerf

Start a puer Server, easily mock routes and render FreeMarker templates

Options:

    -h, --help                output usage information
    -V, --version             output the version number
    -f, --freemarker <file>   Mock file for Freemarker routes
    -m, --mock <file>         Your standard puer mock file
    -c, --combined <file>     Where to save the combined file, defaults to "mock/allRoutes.js"
    -t, --templates <path>    Path to folder in which Freemarker templates are stored
    -r, --root <folder>       The root folder that files should be served from
    -p, --port <number>       Specific port to use
    -w, --watch <files>       Filetypes to watch, defaults to js|css|html|xhtml
    -x, --exclude <files>     Exclude files from being watched for updates
    -l, --localhost           Use "localhost" instead of "127.0.0.1"
    --no-browser              Do not autimatically open a brwoser
    --debug                   Display debug messages
```

Mocking requests

This is what puerF is really all about. Making it as easy as possible for you to "fake" a backend. To achieve this puerF builds upon puers mocking of request https://github.com/leeluolee/puer#mock-request. And simplefies the use of Freemarker templates for those requests.

puerF will automatically look for two route files. `mock/routes.js` and `mock/ftlRoutes.js`. While `routes.js` should follow the puer documentation https://github.com/leeluolee/puer#mock-request and can mock any kind of route, the `ftlRoutes.js` file can only contain Freemarker routes.

Should you wish to use files from a different location you can do so useing the `-m` and `-f` options.

Working with query parameters

Real world applications might use query parameters to get specific results from a URL. You can easily mock those requests as well. To mock a route like `/example/some?user=name`, inside your regular routes file, you can simply access `req.query.user` to get the users name.

```
"GET /example/some": function(req, res, next) {
    var name = req.query.user;

    //Do something with the name, like sending it back.
    res.status(200).send(name).end();
}
```

Freemarker routes

The file containing routes for Freemarker should export a single object containing key like a standard puer routes mock file but provide objects as values for those keys. These 'route configurations' should have two properties:

- template: The template to use.
- data: Data that should be handed to the template.

Note that if `data` has an attribute called `user` the template will get a variable called `user` passed to it.

```
module.exports = {
    'GET /test': {
        template: 'test.ftl',
        data: {
            name: 'value',
            objName: {
                property: 'someValue',
                number: 13
            }
        }
    },
    //....
}
```

Freemarker templates

By default puerF will look for Freemarker templates in `./templates`. To specify another folder you can use `-t`. Read the guid to author Freemarker templates http://freemarker.org/docs/dgui.html.