Templates

Agenda

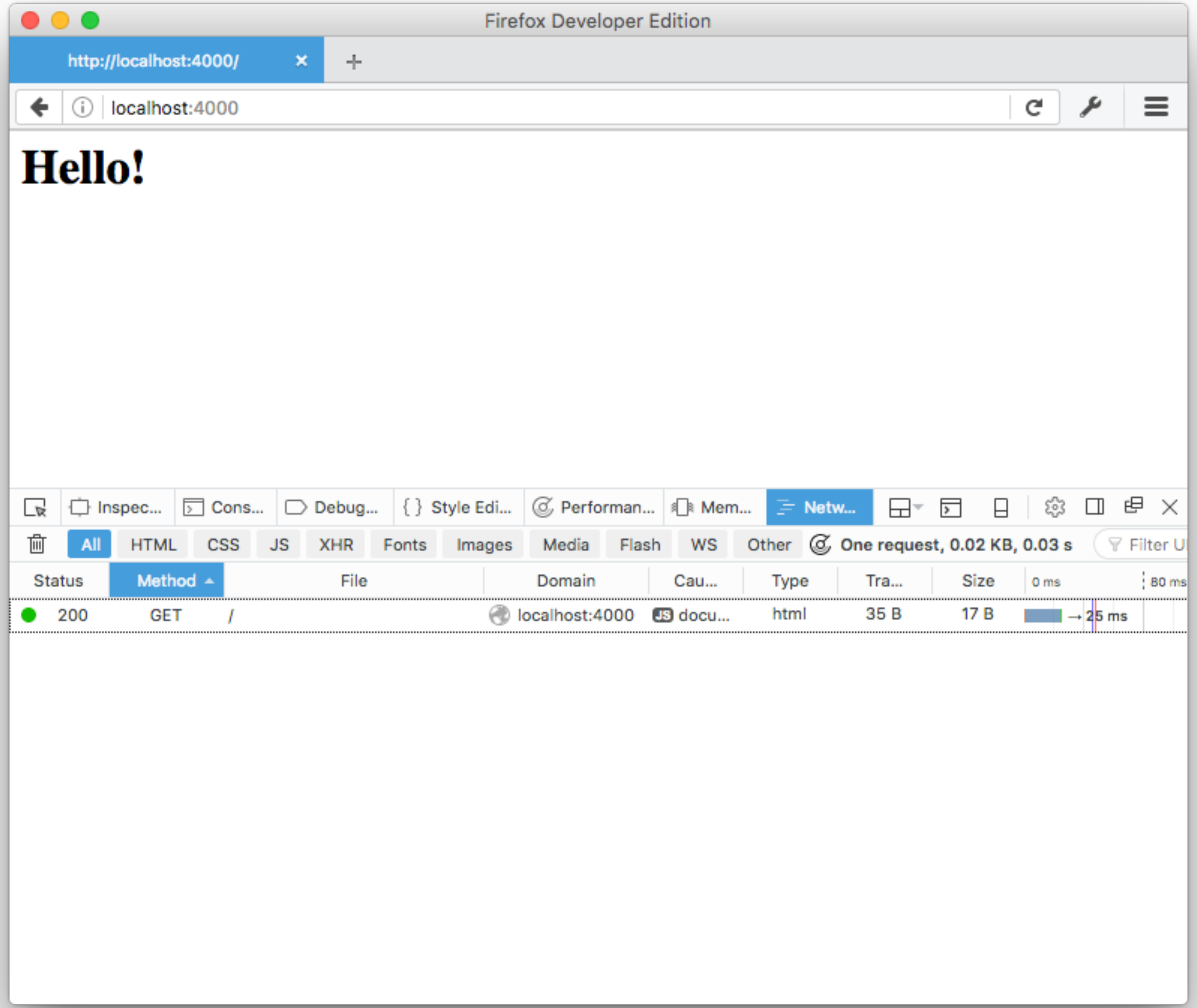
- Need for a tempting engine
- Handlebars
- Handlebars in Hapi

reply

- In order to render web pages we could pass html content
- This would become very unwieldy and unmaintainable

controller.js

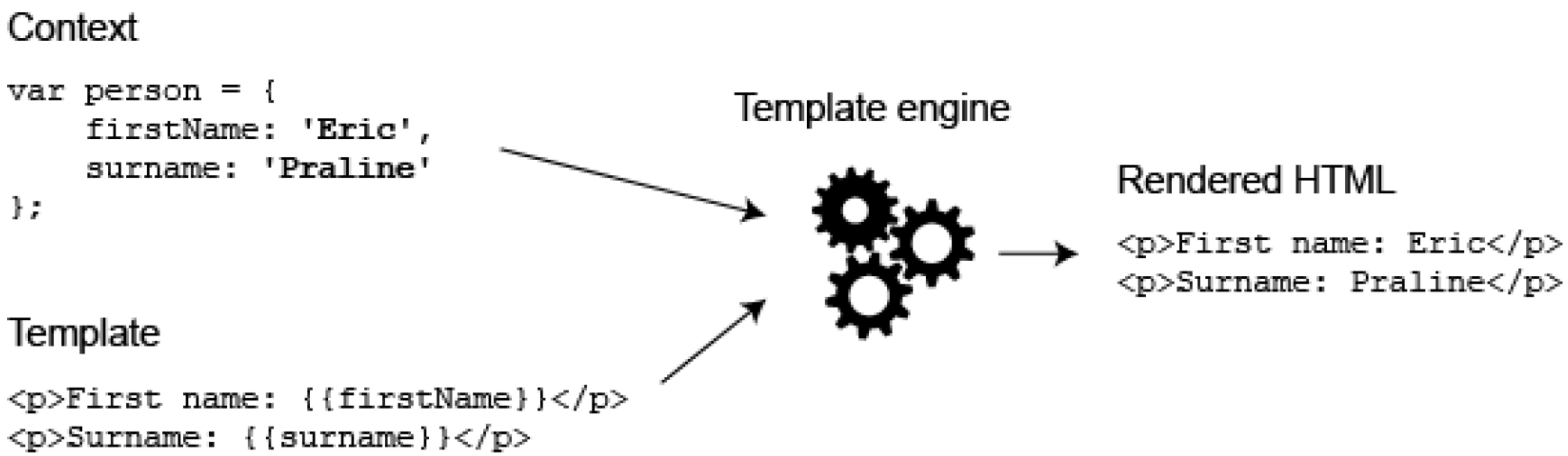
```
exports.index = {

  handler: function (request, reply) {
    reply('<h1> Hello! </h1>');
  }

};
```


Firefox Developer Edition
http://localhost:4000/
localhost:4000
Inspec... Cons... Debug... Style Edi... Performan... Mem... Netw...
All HTML CSS JS XHR Fonts Images Media Flash WS Other
One request, 0.02 KB, 0.03 s
Filter U

Hello!

Status	Method	File	Domain	Cau...	Type	Tra...	Size	0 ms
200	GET	/	localhost:4000	docu...	html	35 B	17 B	→ 25 ms

Tempting Engine



```
var person = {
    firstName: 'Eric',
    surname: 'Praline'
};
```

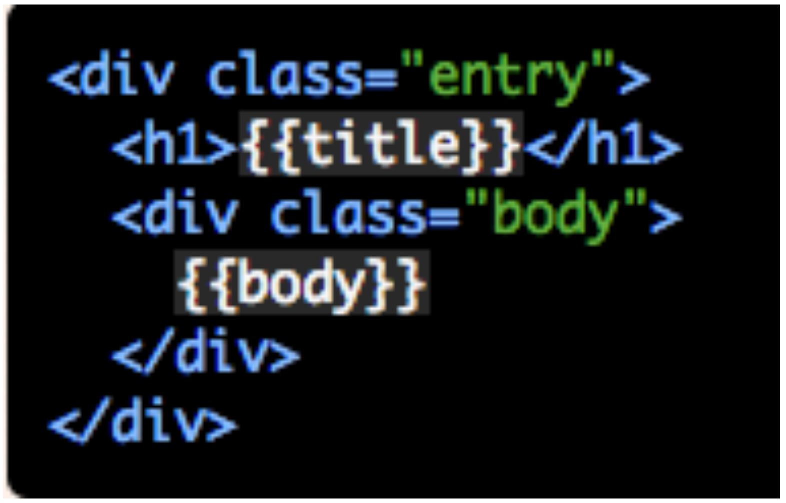
```
<p>First name: {{firstName}}</p>
<p>Surname: {{surname}}</p>
```

```
<p>First name: Eric</p>
<p>Surname: Praline</p>
```

Template Engines: Handlebars

“Handlebars provides the power necessary to let you build semantic templates effectively with no frustration.

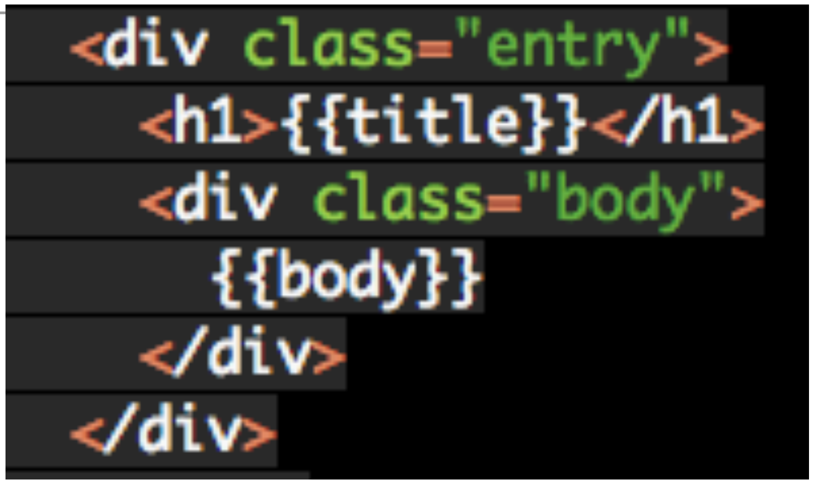
Handlebars is largely compatible with Mustache templates. In most cases it is possible to swap out Mustache with Handlebars and continue using your current templates..”



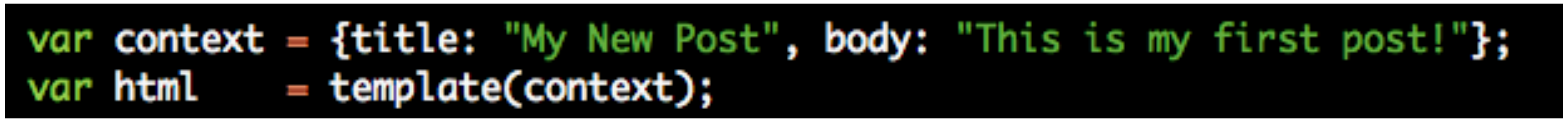
```
<div class="entry">
  <h1>{{title}}</h1>
  <div class="body">
    {{body}}
  </div>
</div>
```

Template Expressions

- A handlebars expression is a {{, some contents, followed by a }}



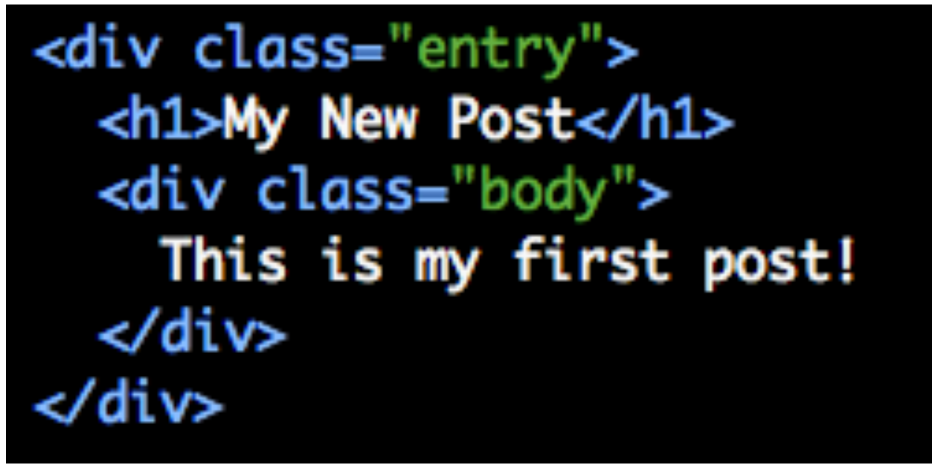
```
<div class="entry">
  <h1>{{title}}</h1>
  <div class="body">
    {{body}}
  </div>
</div>
```



```
var context = {title: "My New Post", body: "This is my first post!"};
var html    = template(context);
```

- In Javascript, create an object literal with matching properties

- When rendered, the properties replace the handlebars expressions



```
<div class="entry">
  <h1>My New Post</h1>
  <div class="body">
    This is my first post!
  </div>
</div>
```

Handlebars Features

http://handlebarsjs.com/expressions
http://handlebarsjs.com/builtin_helpers
http://handlebarsjs.com/helpers

- Expressions
- Helpers
- Partials

} must be mastered by developer

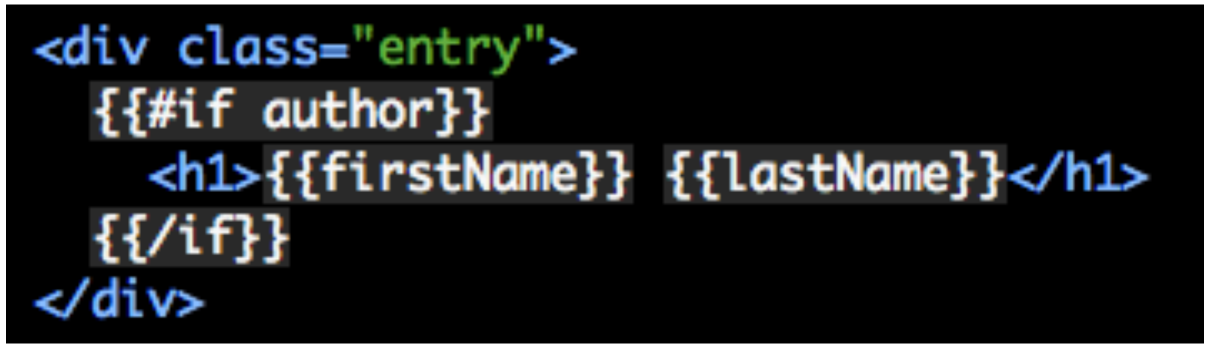
- Precompilation
- Execution

} integrated into Hapi by ‘views’ plugin http://handlebarsjs.com/builtin_helpers

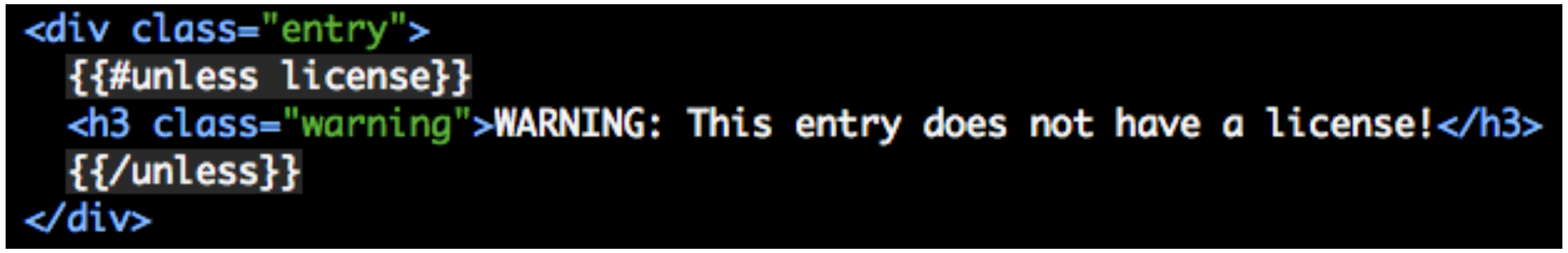
Helpers

- Block expressions allow you to define helpers that will invoke a section of your template with a different context than the current.

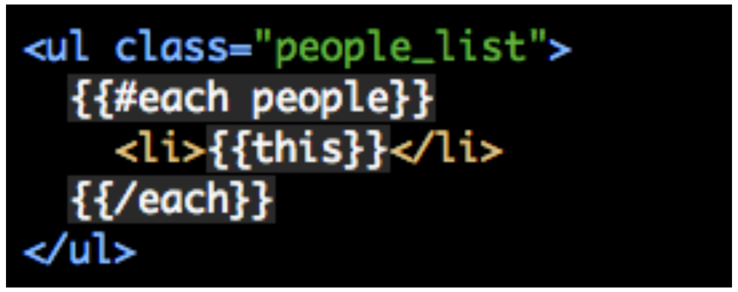
- These block helpers are identified by a # preceeding the helper name and require a matching closing mustache, /, of the same name.



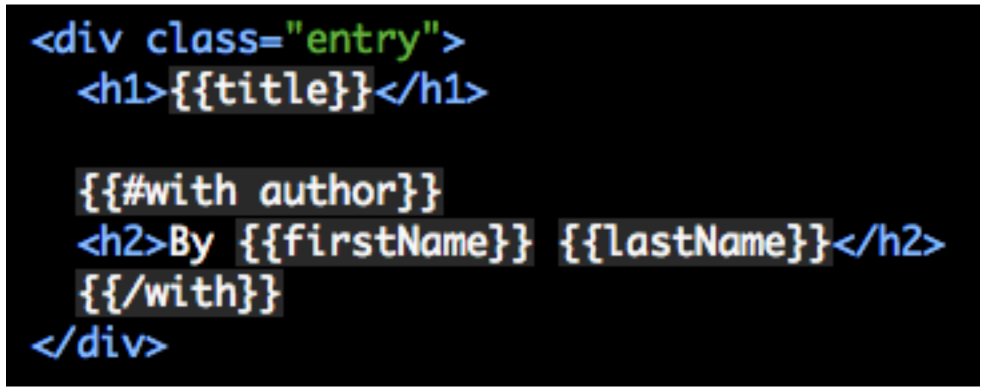
```
<div class="entry">
  {{#if author}}
    <h1>{{firstName}} {{lastName}}</h1>
  {{/if}}
</div>
```



```
<div class="entry">
  {{#unless license}}
  <h3 class="warning">WARNING: This entry does not have a license!</h3>
  {{/unless}}
</div>
```



```
<ul class="people_list">
  {{#each people}}
    <li>{{this}}</li>
  {{/each}}
</ul>
```

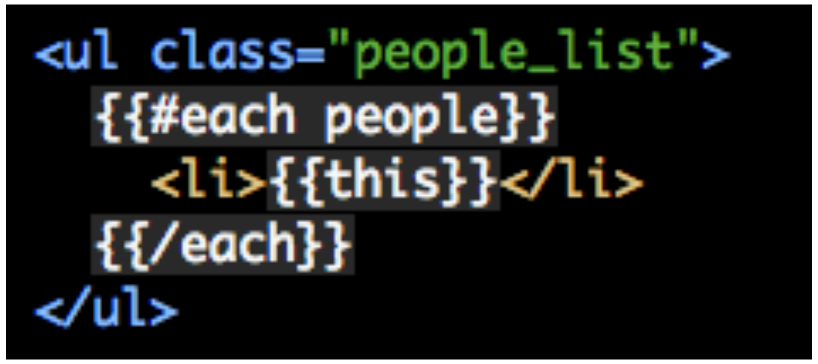


```
<div class="entry">
  <h1>{{title}}</h1>

  {{#with author}}
  <h2>By {{firstName}} {{lastName}}</h2>
  {{/with}}
</div>
```

- if
- unless
- each
- with
- lookup
- log

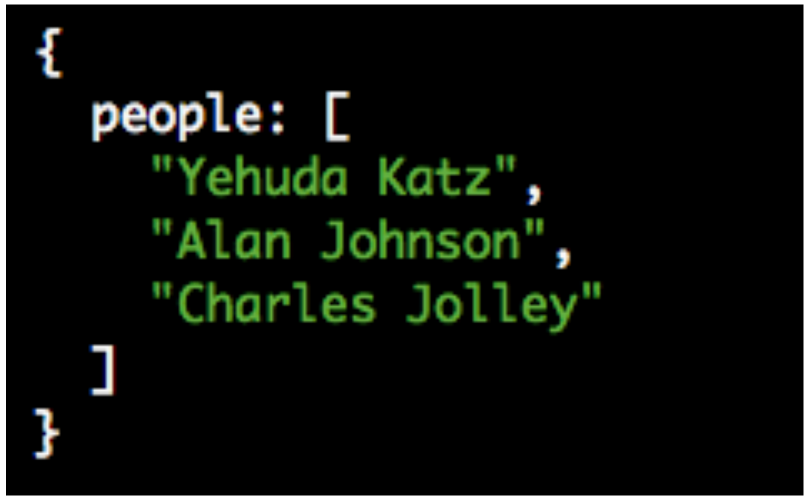
each helper

You can iterate over a list using the built-in each helper. Inside the block, you can use this to reference the element being iterated over.



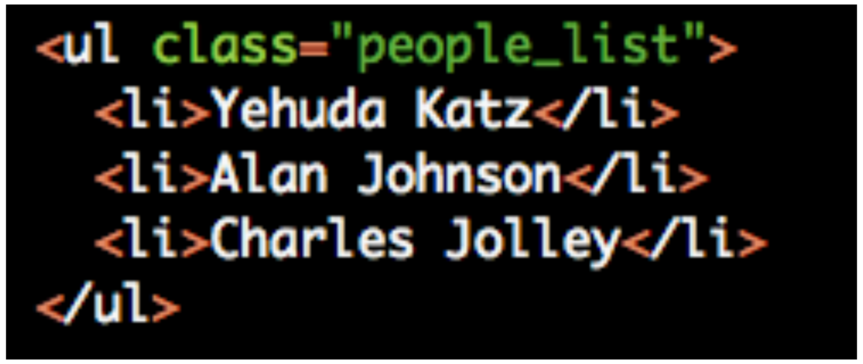
```
<ul class="people_list">
  {{#each people}}
    <li>{{this}}</li>
  {{/each}}
</ul>
```

when used with this context:



```
{
  people: [
    "Yehuda Katz",
    "Alan Johnson",
    "Charles Jolley"
  ]
}
```

will result in:



```
<ul class="people_list">
  <li>Yehuda Katz</li>
  <li>Alan Johnson</li>
  <li>Charles Jolley</li>
</ul>
```

Partials

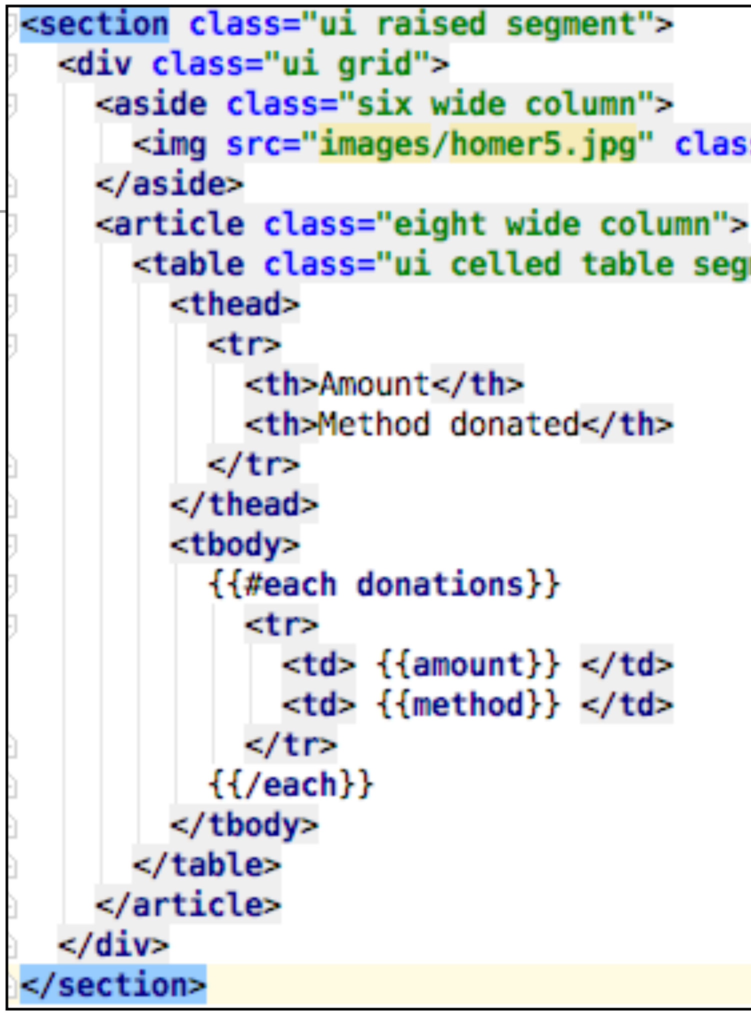
- Handlebars partials allow for code reuse by creating shared templates.

- Calling the partial is done through the partial call syntax:

```
{{> myPartial }}
```

- Will render the partial named myPartial. When the partial executes, it will be run under the current execution context.

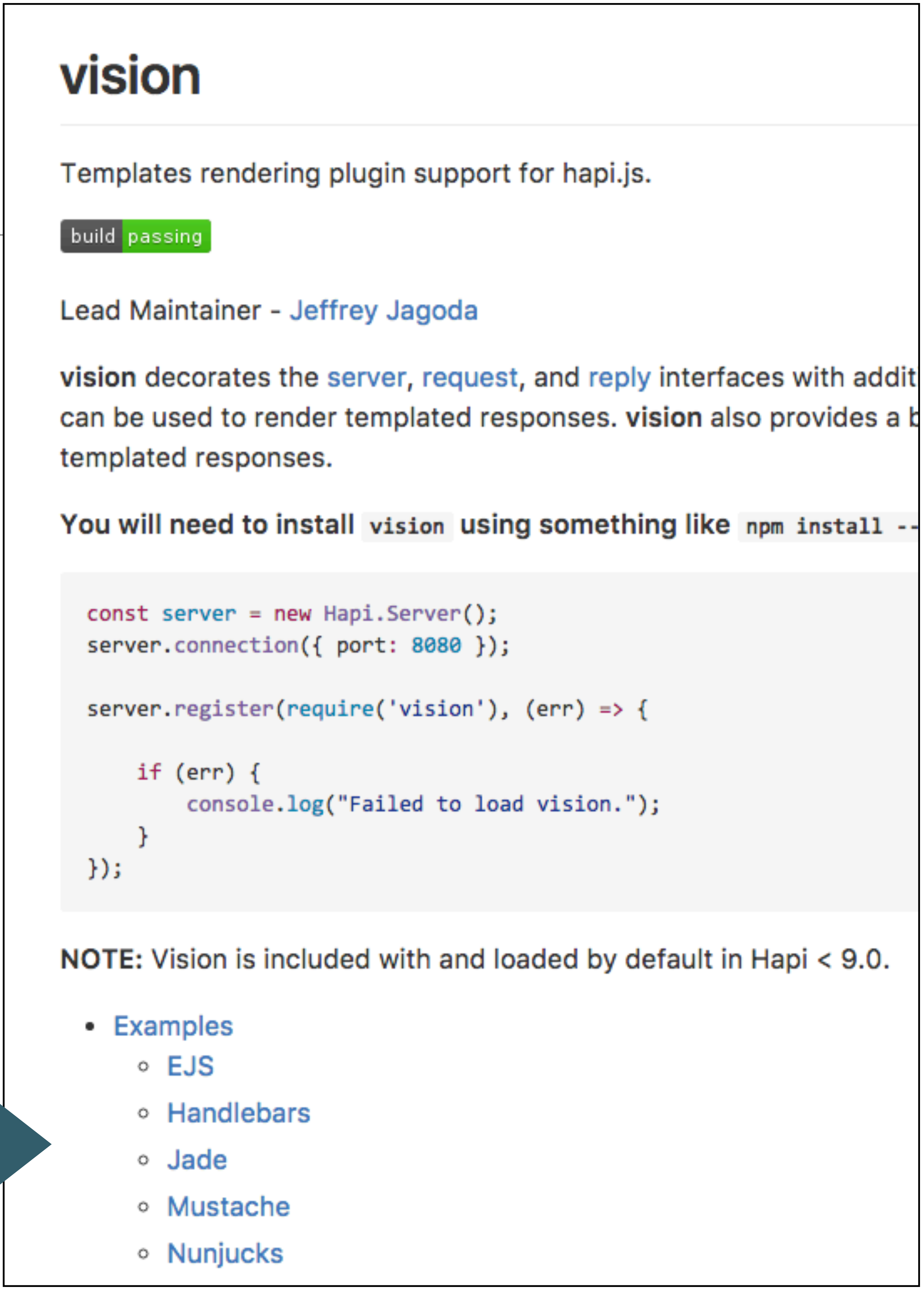
myPartial.hbs



```
<section class="ui raised segment">
  <div class="ui grid">
    <aside class="six wide column">
      <img src="images/homer5.jpg" clas
    </aside>
    <article class="eight wide column">
      <table class="ui celled table seg
        <thead>
          <tr>
            <th>Amount</th>
            <th>Method donated</th>
          </tr>
        </thead>
        <tbody>
          {{#each donations}}
            <tr>
              <td> {{amount}} </td>
              <td> {{method}} </td>
            </tr>
          {{/each}}
        </tbody>
      </table>
    </article>
  </div>
</section>
```

Handlebars in Hapi

- Vison Plugin loads and manages a templating engine
- Supports a range of tempting languages



vision

Templates rendering plugin support for hapi.js.

build passing

Lead Maintainer - Jeffrey Jagoda

vision decorates the server, request, and reply interfaces with addit
can be used to render templated responses. **vision** also provides a b
templated responses.

You will need to install `vision` using something like `npm install --`

```
const server = new Hapi.Server();
server.connection({ port: 8080 });

server.register(require('vision'), (err) => {

    if (err) {
        console.log("Failed to load vision.");
    }
});
```

NOTE: Vision is included with and loaded by default in Hapi < 9.0.

- Examples
  - EJS
  - Handlebars
  - Jade
  - Mustache
  - Nunjucks

Plugin Install

- Install the Vision plugin + the specific tempting engine you wish to use

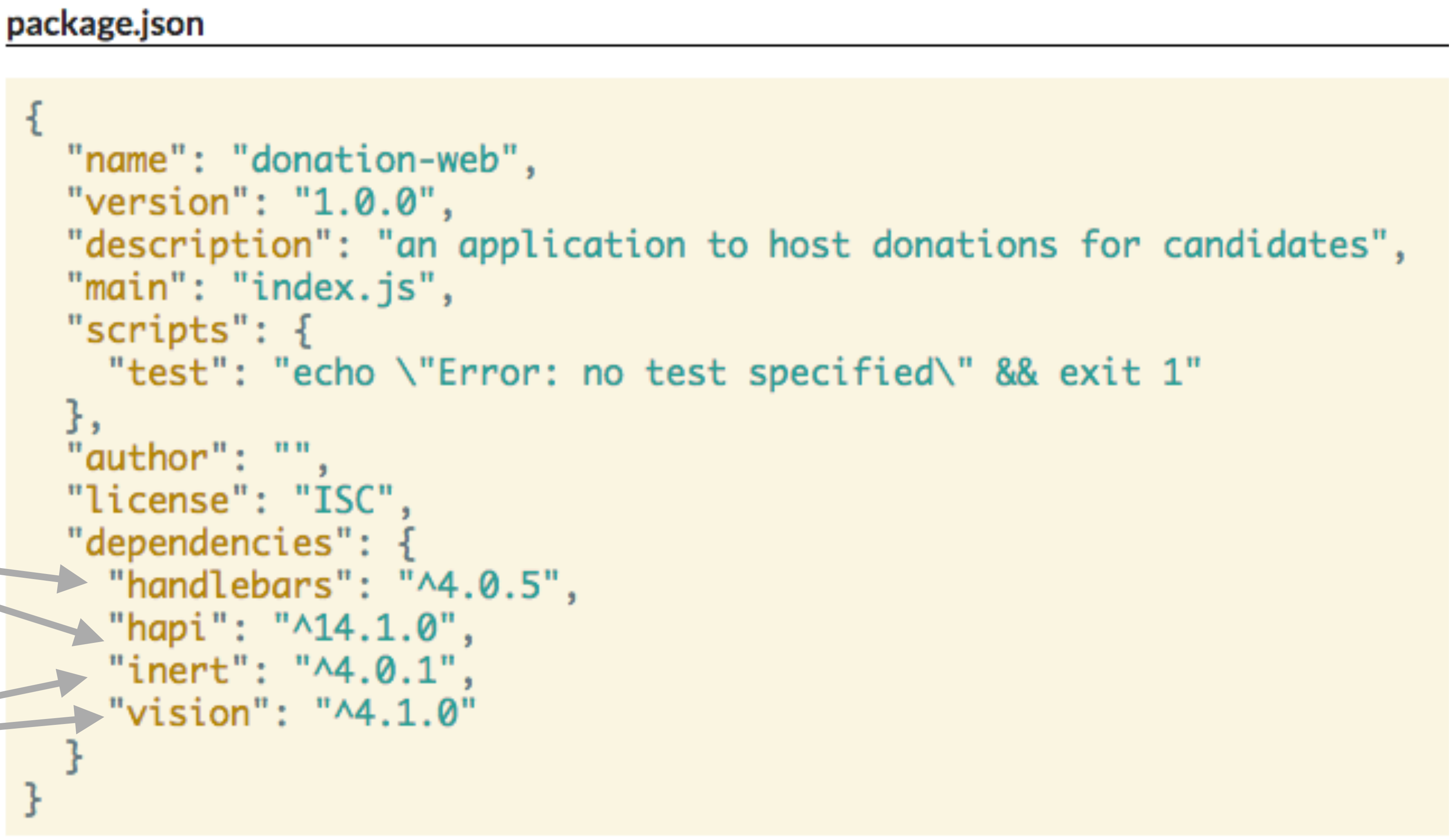
```
npm install vision -save
```

```
npm install handlebars -save
```

A generic node module descriptor

general purpose node modules

Hapi plugin node modules



package.json

```
{
  "name": "donation-web",
  "version": "1.0.0",
  "description": "an application to host donations for candidates",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "handlebars": "^4.0.5",
    "hapi": "^14.1.0",
    "inert": "^4.0.1",
    "vision": "^4.1.0"
  }
}
```

Register the Plugin

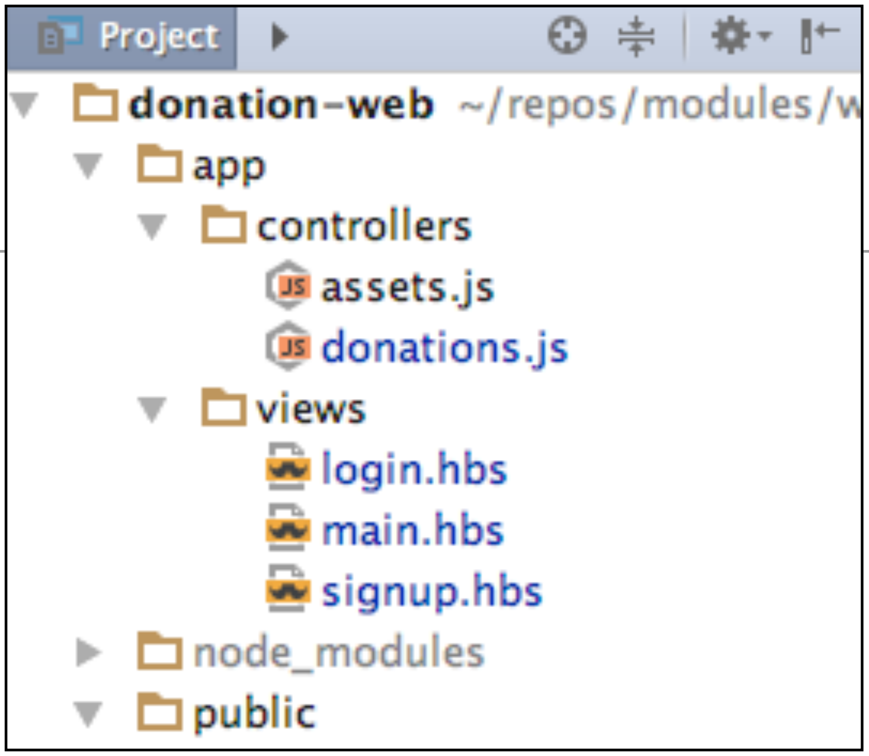
Import & register the plugin

Initialise to use Handlebars engine

Define template locations and cache settings

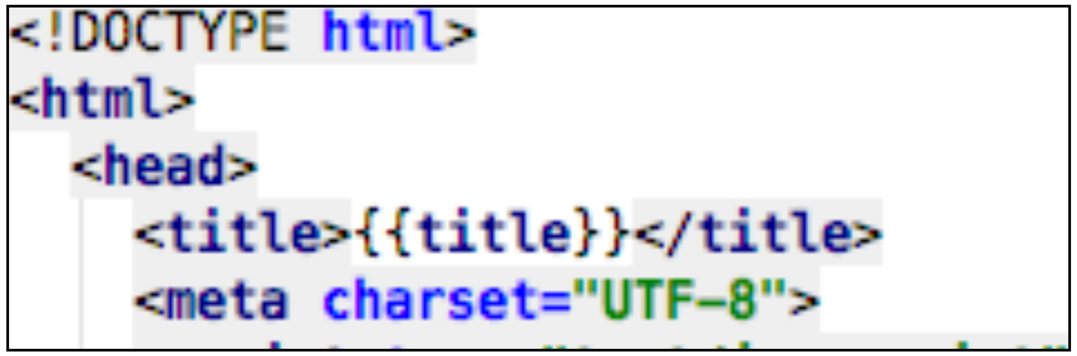
```
...

server.register([require('inert'), require('vision')], err => {

  if (err) {
    throw err;
  }

  server.views({
    engines: {
      hbs: require('handlebars'),
    },
    relativeTo: __dirname,
    path: './app/views',
    isCached: false,
  });

...

});
```

Rename views to ‘.hbs’



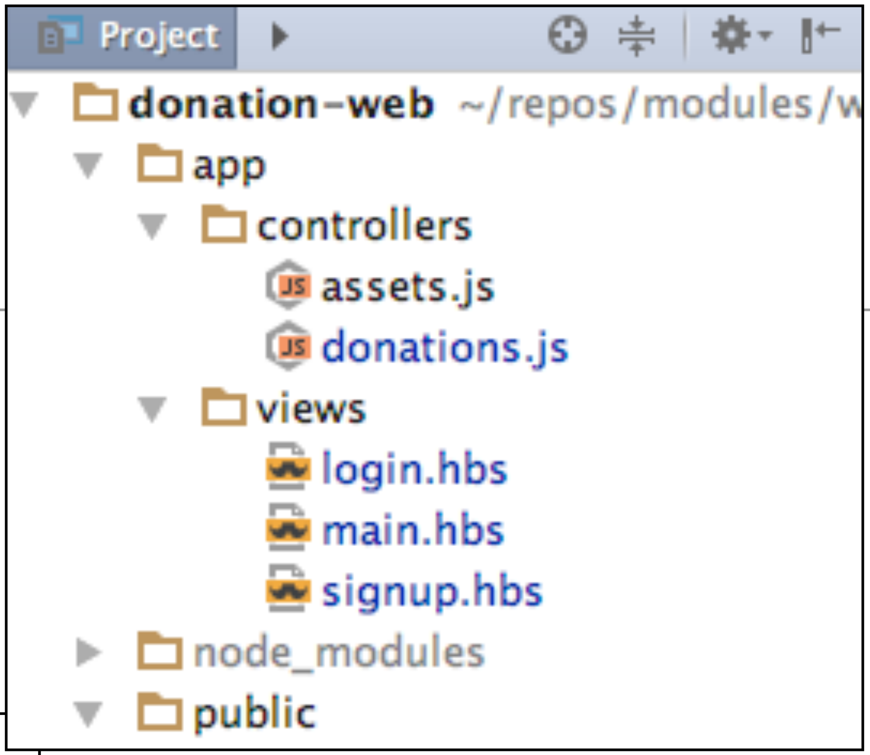
```
Project
▼ donation-web ~/repos/modules/w
  ▼ app
    ▼ controllers
        assets.js
        donations.js
    ▼ views
        login.hbs
        main.hbs
        signup.hbs
  ▶ node_modules
  ▼ public
```

- View can now new handlebars expressions, which will be interpolated if the correct context is provided



```
<!DOCTYPE html>
<html>
  <head>
    <title>{{title}}</title>
    <meta charset="UTF-8">
```

Rendering the Context



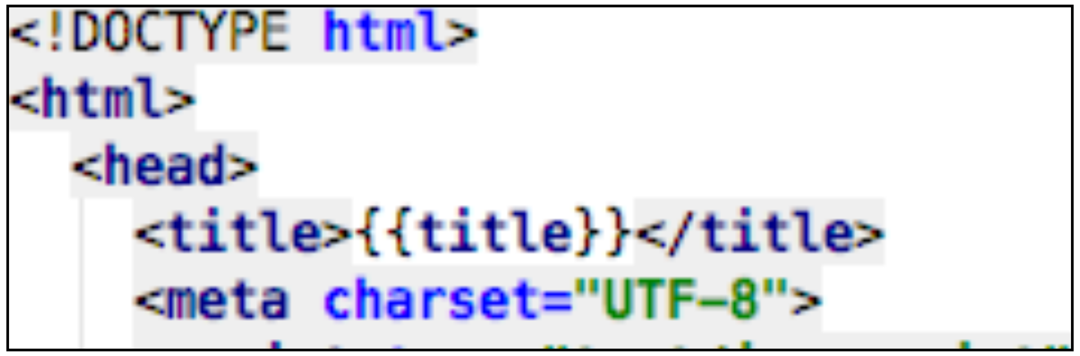
```
donation-web ~/repos/modules/w
  app
    controllers
      assets.js
      donations.js
    views
      login.hbs
      main.hbs
      signup.hbs
  node_modules
  public
```

- view function accepts a ‘context’ object

```
'use strict';

exports.home = {

  handler: (request, reply) => {
    reply.view('main', { title: 'Welcome to Donations' });
  },

};
```

- ‘main’ template loaded

- handlebars expressions retrieve information from the ‘context’



```
<!DOCTYPE html>
<html>
  <head>
    <title>{{title}}</title>
    <meta charset="UTF-8">
```

Partials & Layouts

- Partials & Layouts play a prominent role in enabling DRY (Dont Repeat Yourself) principles
  - Partials: Reusable templates
  - Layouts: Reusable Page Structure
- These features must be explicitly enabled

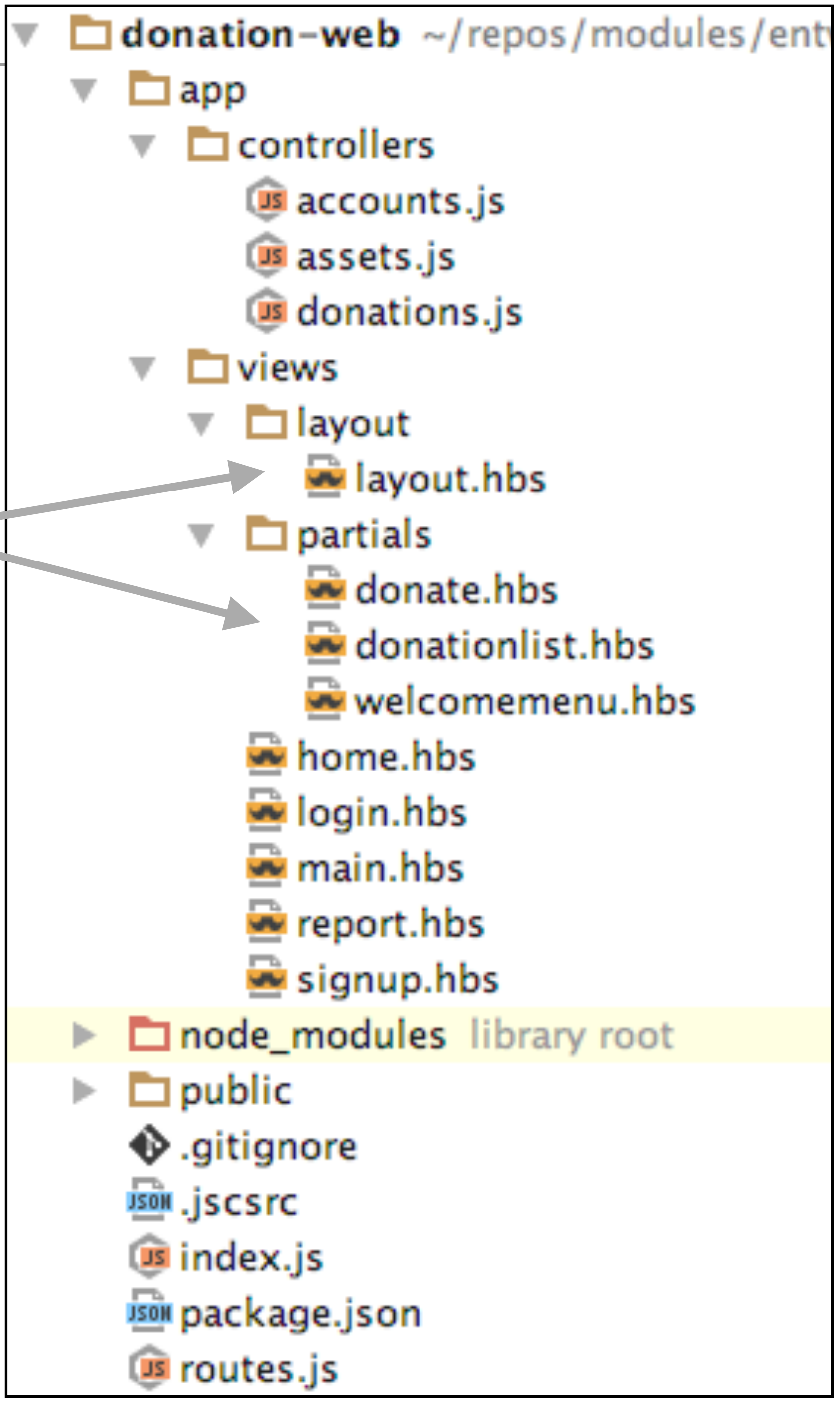
```
...
server.views({
  engines: {
    hbs: require('handlebars'),
  },
  relativeTo: __dirname,
  path: './app/views',
  layoutPath: './app/views/layout',
  partialsPath: './app/views/partials',
  layout: true,
  isCached: false,
});
...
```

partials & layouts directories in project

Revised Project Layout

```
...
server.views({
  engines: {
    hbs: require('handlebars'),
  },
  relativeTo: __dirname,
  path: './app/views',
  layoutPath: './app/views/layout',
  partialsPath: './app/views/partials',
  layout: true,
  isCached: false,
});
...
```

partials & layouts directories in project



```
donation-web ~/repos/modules/ent
  app
    controllers
      accounts.js
      assets.js
      donations.js
    views
      layout
        layout.hbs
      partials
        donate.hbs
        donationlist.hbs
        welcomemenu.hbs
      home.hbs
      login.hbs
      main.hbs
      report.hbs
      signup.hbs
  node_modules library root
  public
  .gitignore
  .jscsrc
  index.js
  package.json
  routes.js
```

- Templates can now assume these folders part of the rendering pipeline

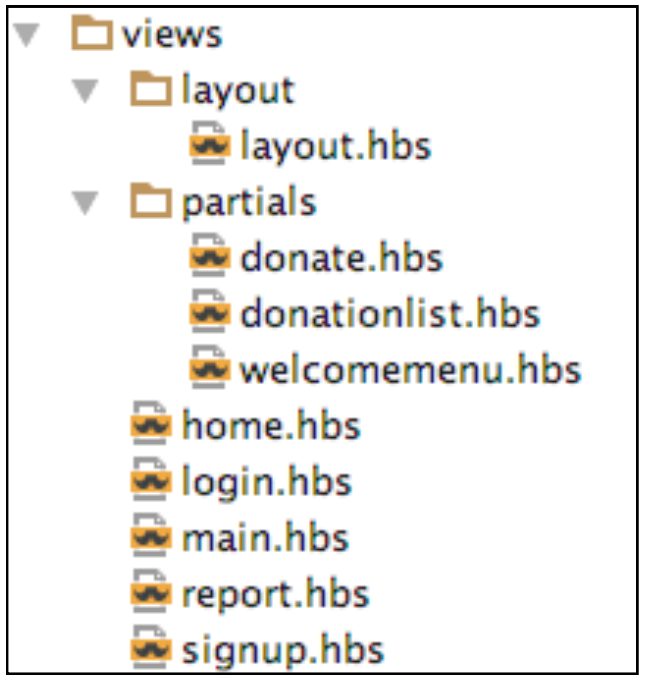
Layouts & Partials in Action

layout.hbs

```
<!DOCTYPE html>
<html>
  <head>
    <title>{{title}}</title>
    <meta charset="UTF-8">
    ...include stylesheets
  </head>
  <body>
    <section class="ui container">
      {{{content}}}
    </section>
    <script>
  </body>
</html>
```

welcomemenu.hbs

```
<nav class="ui inverted menu">
  <header class="header item"> <a href="/"> Donation </a> </header>
  <div class="right menu">
    <a class="item" href="/signup"> Signup</a>
    <a class="item" href="/login">  Login</a>
  </div>
</nav>
```



```
views
  layout
    layout.hbs
  partials
    donate.hbs
    donationlist.hbs
    welcomemenu.hbs
  home.hbs
  login.hbs
  main.hbs
  report.hbs
  signup.hbs
```

main.hbs

```
{{> welcomemenu }}

<section class="ui stacked segment">
  <div class="ui grid">
    <aside class="six wide column">
      <img src="images/homer.png" class="ui medium image">
    </aside>
    <article class="ten wide column">
      <header class="ui  header"> Help Me Run Springfield</header>
      <p> Donate what you can now - No Bitcoins accepted! </p>
    </article>
  </div>
</section>
```

- main content is based on layout - replacing {{content}} expression
- welcomemenu is injected into main to provide menu

```
<!DOCTYPE html>
<html>
  <head>
    <title>{{title}}</title>
    <meta charset="UTF-8">
    ...include stylesheets
  </head>
  <body>
    <section class="ui container">
      {{{content}}}
    </section>
    <script>
  </body>
</html>
```

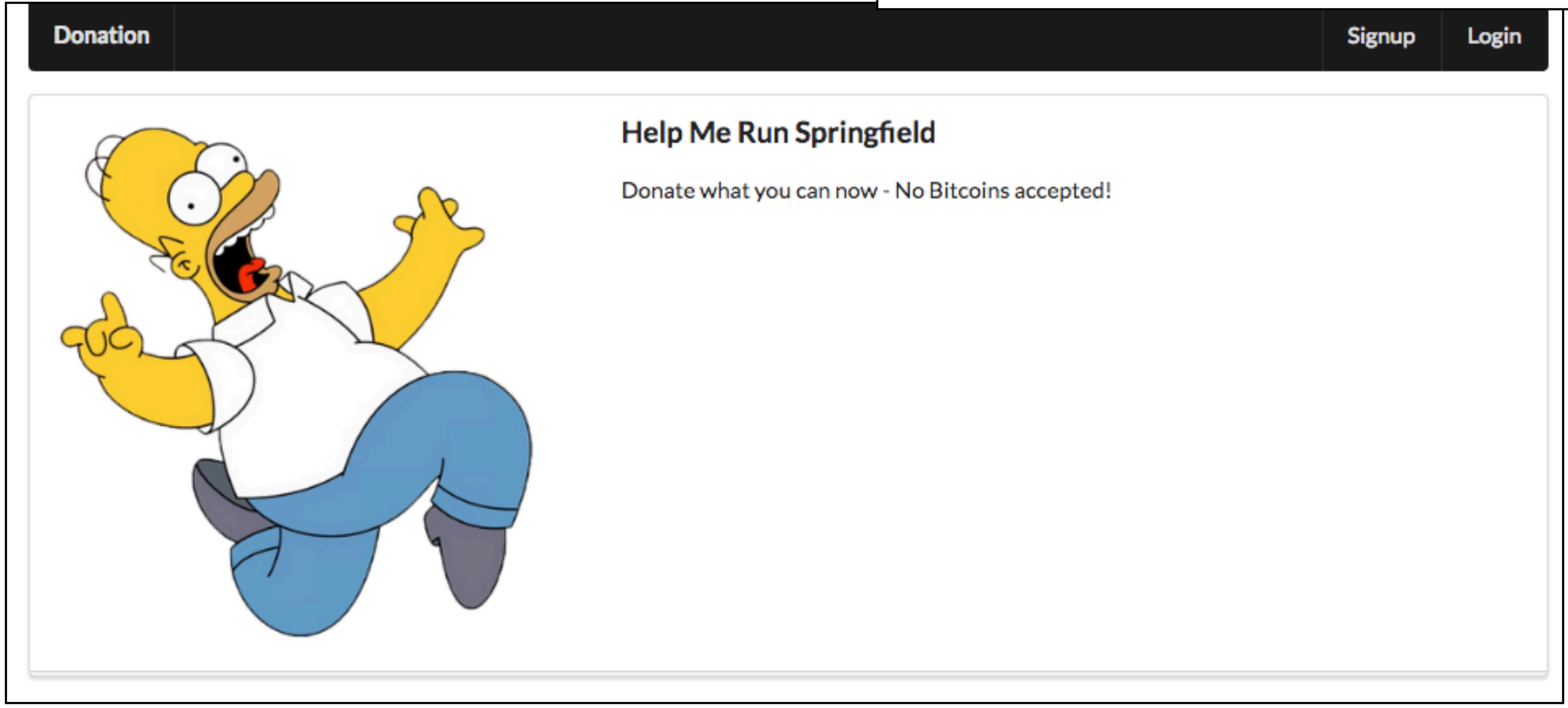
layout.hbs

```
<nav class="ui inverted menu">
  <header class="header item"> <a href="/"> Donation </a> </header
  <div class="right menu">
    <a class="item" href="/signup"> Signup</a>
    <a class="item" href="/login">  Login</a>
  </div>
</nav>
```

welcomemenu.hbs

```
{{> welcomemenu }}

<section class="ui stacked segment">
  <div class="ui grid">
    <aside class="six wide column">
      <img src="images/homer.png" class="ui medium image">
    </aside>
    <article class="ten wide column">
      <header class="ui  header"> Help Me Run Springfield</header>
      <p> Donate what you can now - No Bitcoins accepted! </p>
    </article>
  </div>
</section>
```

main.hbs



Help Me Run Springfield

Donate what you can now - No Bitcoins accepted!