Web Development

Produced by

Eamonn de Leastar (edeleastar@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

http://www.wit.ie

http://elearning.wit.ie

Play Review

Web Development with Play

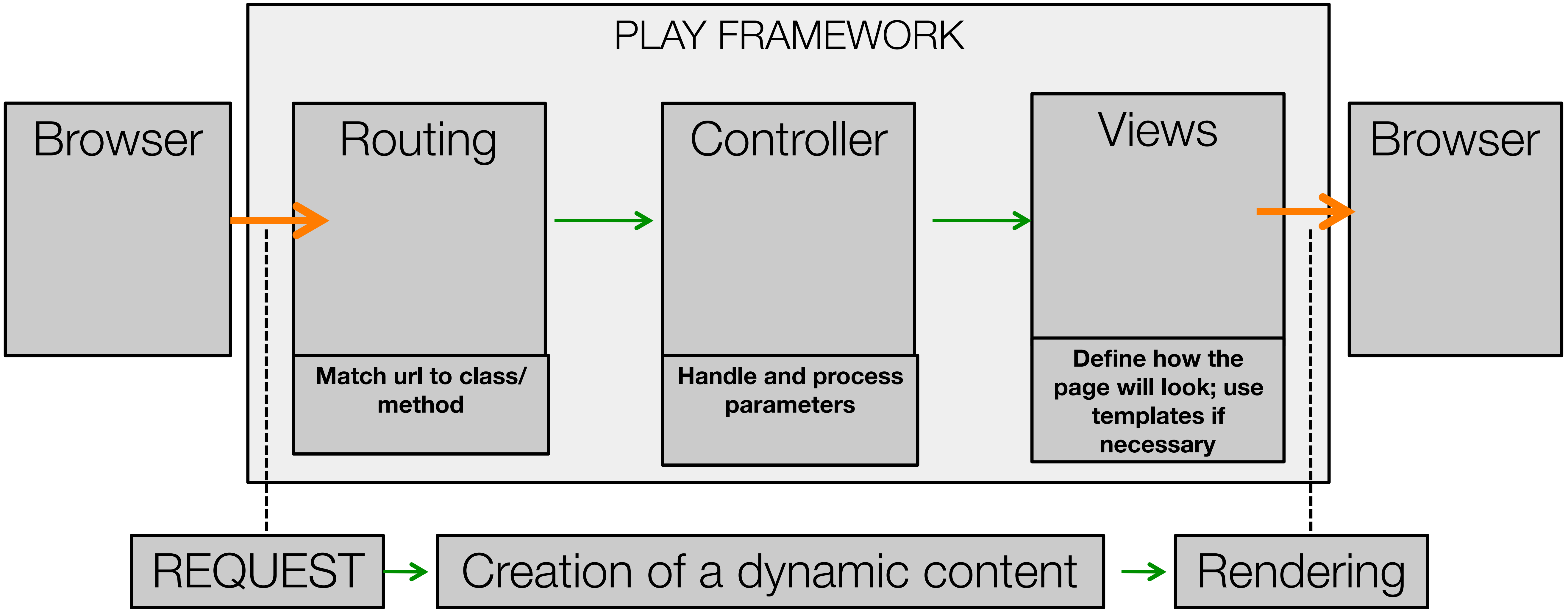
Agenda

- A summary of the process involved in using the Play Framework

Templates

- When creating a website pages will have similarities and differences
- We can base similar pages on the same template (common code) to group all similar features
- We can also add individualised elements dynamically (e.g., title) and display different content based on the url requested (e.g., image)

Overview


PLAY FRAMEWORK
Browser
Routing
Match url to class/ method
Controller
Handle and process parameters
Views
Define how the page will look; use templates if necessary
Browser
REQUEST
Creation of a dynamic content
Rendering

Request/Response

- A Page is requested using a url/path
- The request is routed to a controller and the corresponding method is called
- The method evaluates the parameter(s) and renders the page accordingly
- The corresponding page is called with the parameters passed in the method
- The page is rendered

Template Structure

- The page may:
  - include predefined HTML (includes)
  - be based on a template (extends)
  - use parameters passed from the Java class (initially included in the request)

Routing

- Routing effectively links the http request to the final file
- Routing is based on the url requested and its structure/syntax
- Routing instructions are case-sensitive
- Using routing we match a url or path to a specific rendering (view)
- Syntax: **GET…path…class.method**

```
# Routes
# This file defines all application routes (Higher priority routes first)
# ~~~~

# Landing page
GET     /                                       Home.index

# Home page
GET     /home                                   Home.index
GET     /home/drop/{name}                       Home.drop

# Members page
GET     /members                                Members.index
GET     /members/follow/{name}                  Members.follow

# Profile page
GET     /profile                                Profile.index

# Public Profiles
GET     /publicprofile/{name}                   PublicProfile.visit

# Ignore favicon requests
GET     /favicon.ico                            404

# Map static resources from the /app/public folder to the /public path
GET     /public/                                staticDir:public

# Catch all
*       /{controller}/{action}                  {controller}.{action}
```

Views

- Views are saved as html files
- Views can extend predefined template
- Views can include previously created html files (include)
- Page properties (e.g., set title) can be set
- Views use parameters passed through the corresponding method

```
#{extends 'main.html' /}
#{set title:'Members' /}

<nav class="ui menu">
  <a class="ui item" href="/home">Home</a>
  <a class="ui active item" href="/members">Members</a>
  <a class="ui item" href="/profile">Profile</a>
  <a class="ui item" href="/login">Logout</a>
</nav>

<section class="ui segment">
  <h2 class="ui header">SpaceBook's Members</h2>
  <div class="ui list">
    <div class="item">
      <i class="right triangle icon"></i> marge [<a href="home.html">follow</a>]
    </div>
    <div class="item">
      <i class="right triangle icon"></i> bart [<a href="home.html">follow</a>]
    </div>
    <div class="item">
      <i class="right triangle icon"></i> lisa [<a href="home.html">follow</a>]
    </div>
    <div class="item">
      <i class="right triangle icon"></i> maggie [<a href="home.html">follow</a>]
    </div>
  </div>
</section>
```

Controllers

- Include predefined and customized methods

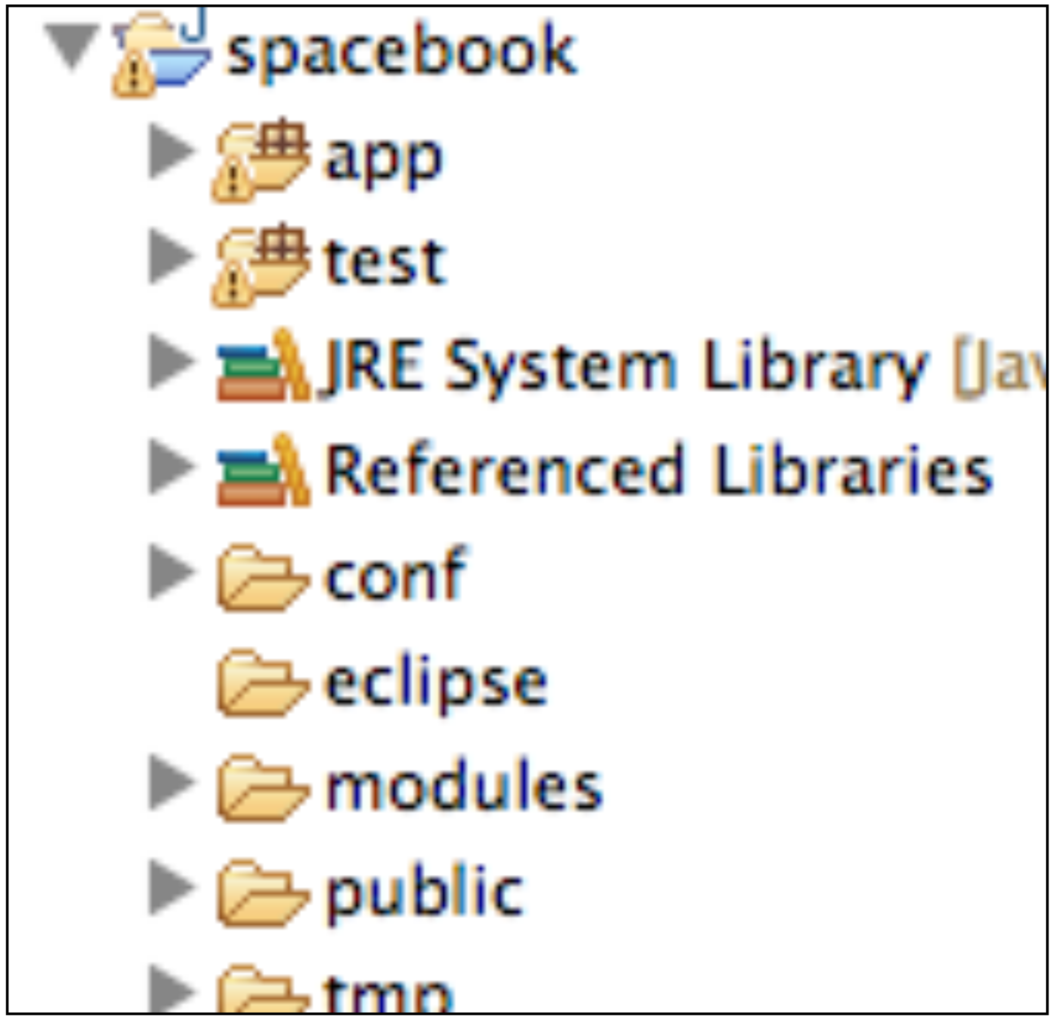
- The name of each method has a matching file name in the view (index method for members will have a corresponding index.html file in the view

- Creating different methods allows us to create individualized rendering (and views)

- Render is called with or without parameters

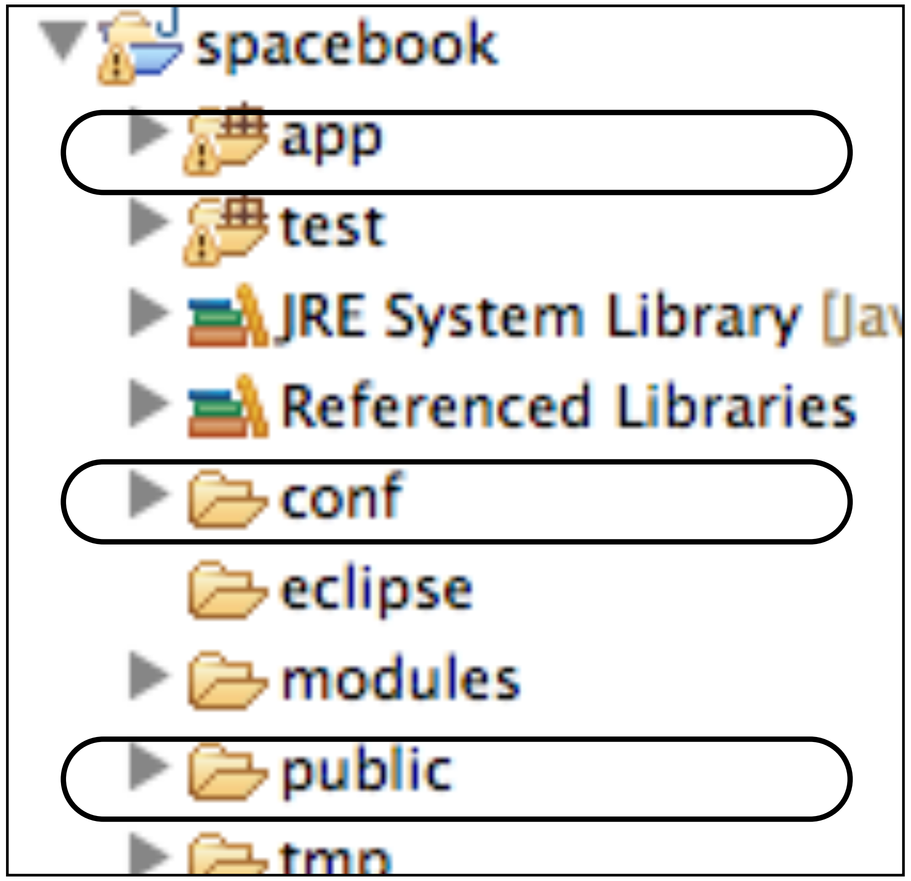
```java
public class Members extends Controller
{
  public static void index()
  {
    render();
  }

  public static void follow (String name)
  {
    Logger.info("Following " + name);
    index();
  }
}
```

Project Structure



- spacebook
  - app
  - test
  - JRE System Library [Ja
  - Referenced Libraries
  - conf
  - eclipse
  - modules
  - public
  - tmp

Project Structure



app

- Application Source

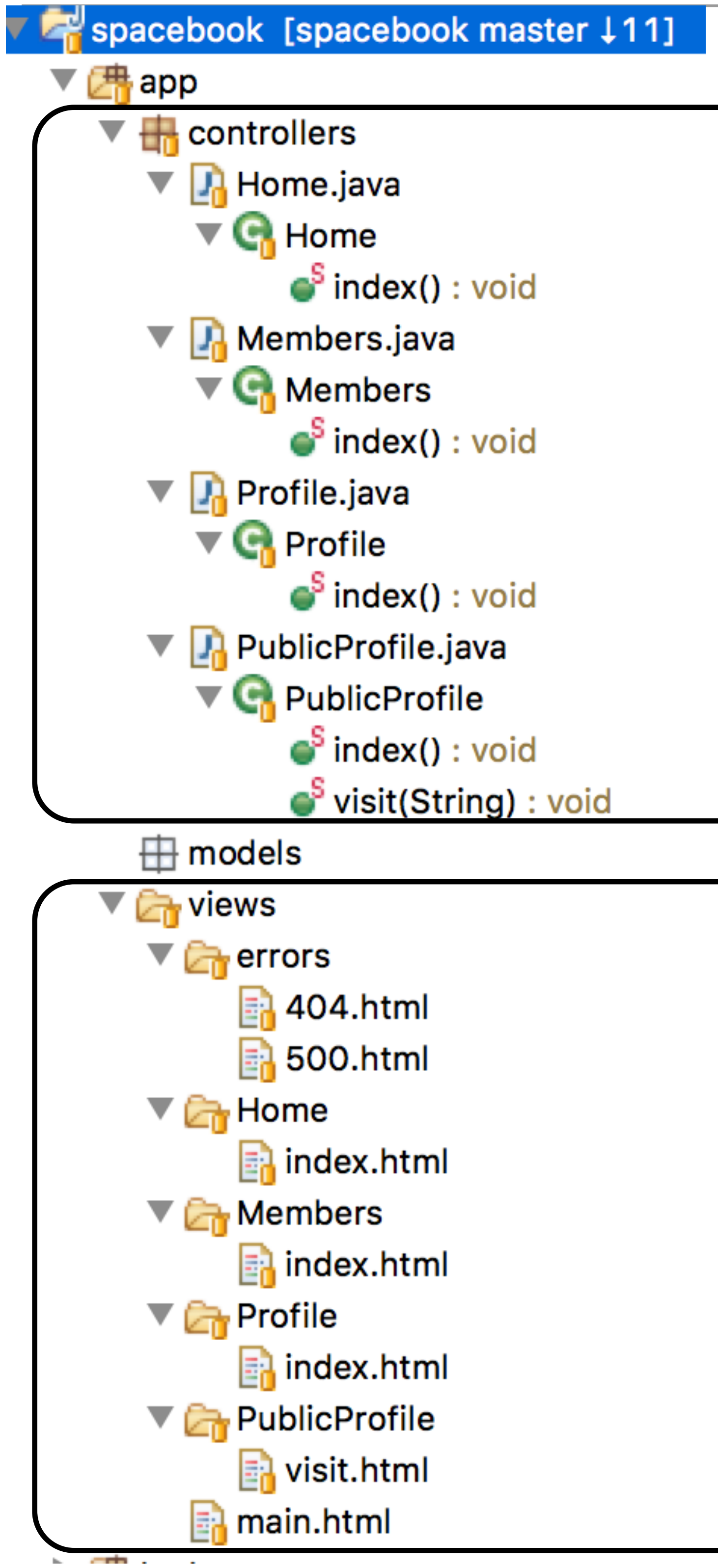
conf

- Configuration

public

- Static web resources

App



```
▼ spacebook  [spacebook master ↓11]
  ▼ app
    ▼ controllers
      ▼ Home.java
        ▼ Home
            index() : void
      ▼ Members.java
        ▼ Members
            index() : void
      ▼ Profile.java
        ▼ Profile
            index() : void
      ▼ PublicProfile.java
        ▼ PublicProfile
            index() : void
            visit(String) : void
      models
    ▼ views
      ▼ errors
          404.html
          500.html
      ▼ Home
          index.html
      ▼ Members
          index.html
      ▼ Profile
          index.html
      ▼ PublicProfile
          visit.html
        main.html
```

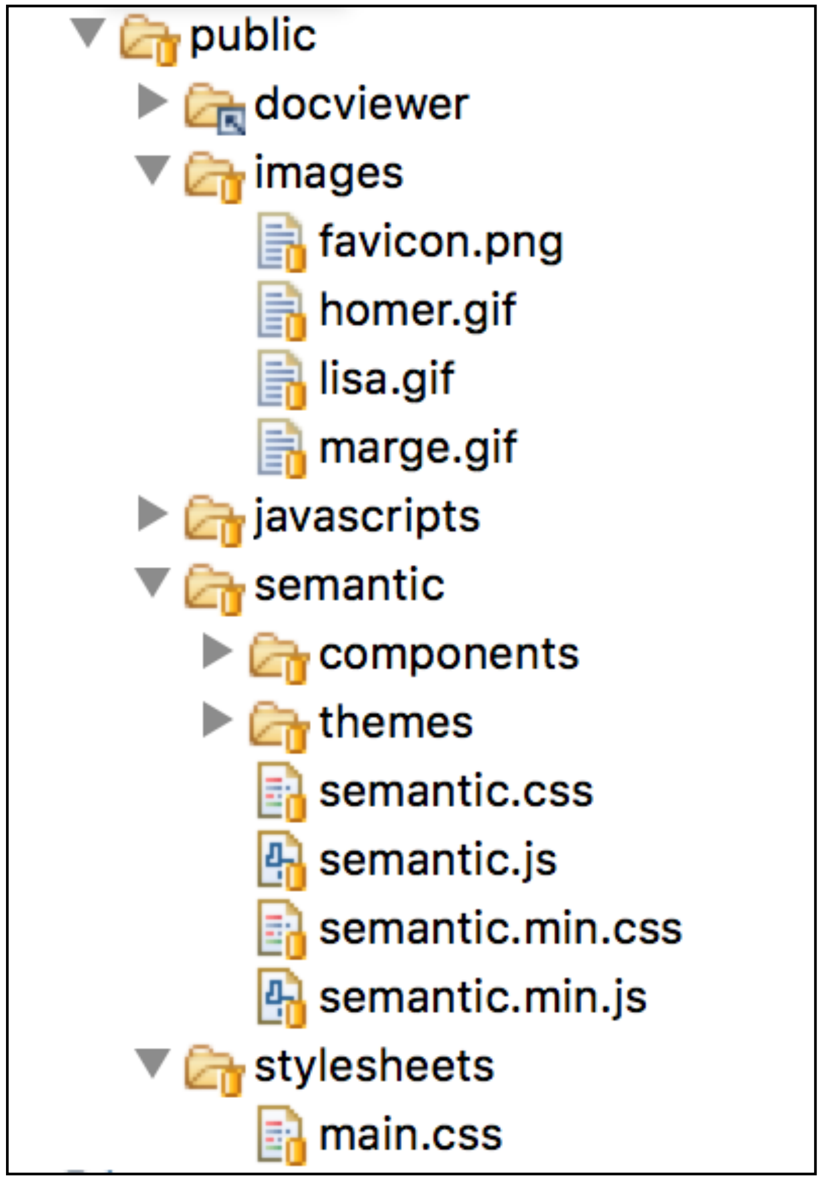
Controllers (java):

- One class for each ‘View’
- Consisting of methods - called ‘Actions’
- One Action for each ‘link’ or ‘form’ on a View

Views (html)

- main.html: structure for all pages served by this app
- + one index.html for each controller representing a logical ‘view’ to be rendered by that controller.

public



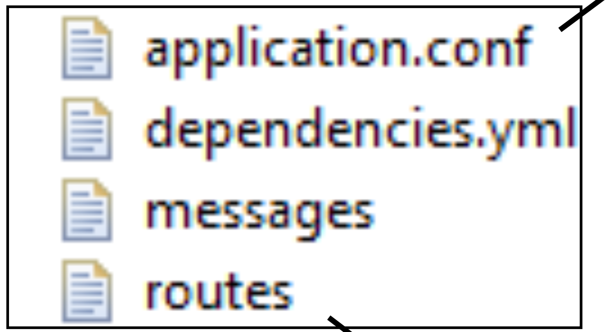
```
public
  docviewer
  images
    favicon.png
    homer.gif
    lisa.gif
    marge.gif
  javascripts
  semantic
    components
    themes
    semantic.css
    semantic.js
    semantic.min.css
    semantic.min.js
  stylesheets
    main.css
```

- Files placed in the ‘public’ folder on the web app when it is deployed.
- Can contain any ‘static’ resource - images, complete html files, css, javascript etc...
- View can refer to these resource via:
  - "@{'/public/'}"

```
<head>
  <title>#{get 'title' /}</title>
  <meta charset="utf-8">
  <link rel="stylesheet" media="screen" href="@{'/public/semantic/css/semantic.min.css'}">
</head>
```

Conf

- Configuration parameters for the entire application
- Not Java, Not HTML, Not template language


application.conf
dependencies.yml
messages
routes

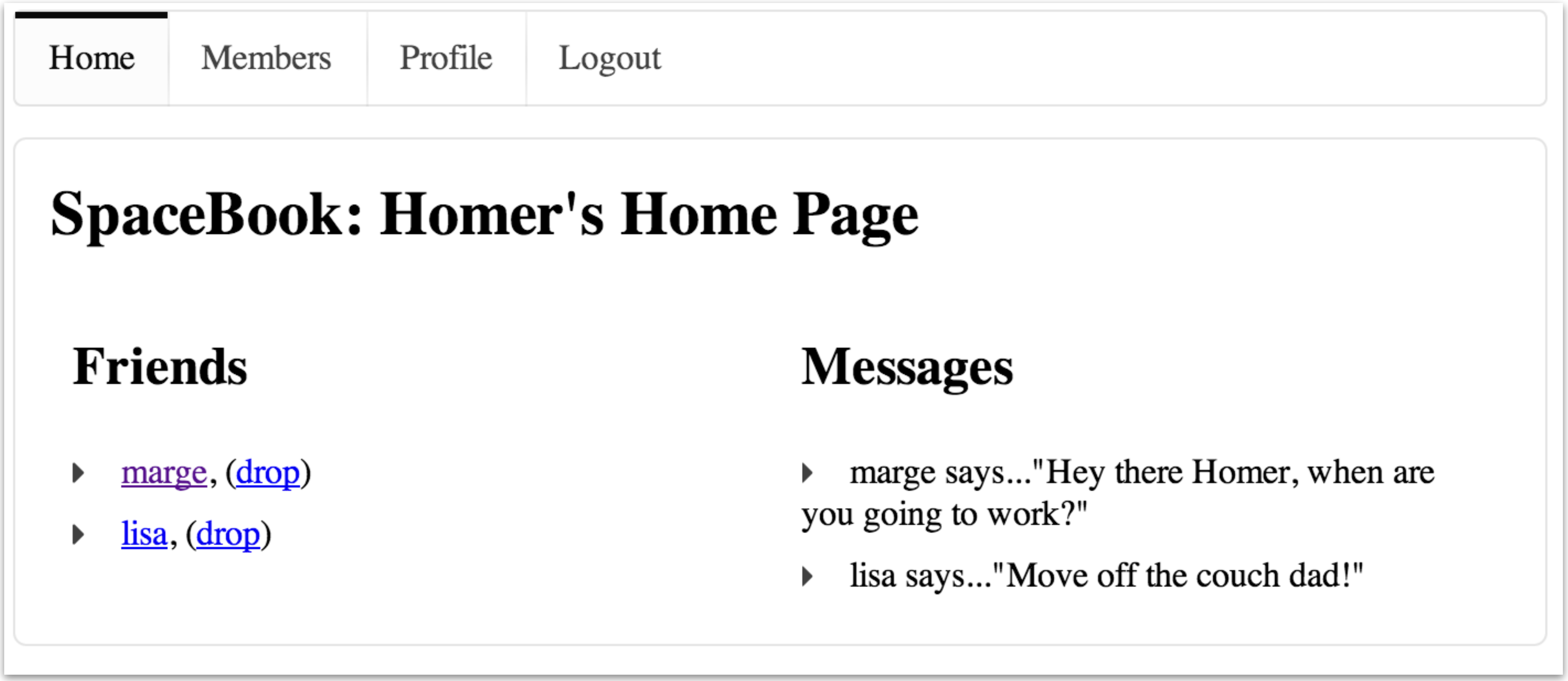
- Various Application Settings, including
  - application name
  - Java version
  - Database configuration
  - etc…

- Application Routes
  - Route Pattern -> Java Action

Home

```
public class Home extends Controller
{
  public static void index()
  {
    render();
  }

  public static void drop (String name)
  {
    Logger.info("Dropping " + name);
    index();
  }
}
```

```
#{extends 'main.html' /}
#{set title:'Home' /}

<nav class="ui menu">
  <a class="ui active item" href="/home">Home</a>
  <a class="ui item" href="/members">Members</a>
  <a class="ui item" href="/profile">Profile</a>
  <a class="ui item" href="/login">Logout</a>
</nav>

<section class="ui segment">
  <h2 class="ui header">SpaceBook: Homer's Home Page</h2>
  <div class="ui two column grid segment">
    <div class="ui row">
      <div class="ui column">
        <h2>Friends</h2>
        <div class="ui list">
          <div class="item">
            <i class="right triangle icon"></i> <a href="/publicprofile/marge">marge</a>, (<a href="drop/marge">drop</a>)
          </div>
          <div class="item">
            <i class="right triangle icon"></i> <a href="/publicprofile/lisa">lisa</a>, (<a href="drop/lisa">drop</a>)
          </div>
        </div>
      </div>
      <div class="ui column">
        <h2>Messages</h2>
        <div class="ui list">
          <div class="item">
            <i class="right triangle icon"></i> marge says..."Hey there Homer, when are you going to work?"
          </div>
          <div class="item">
            <i class="right triangle icon"></i> lisa says..."Move off the couch dad!"
          </div>
        </div>
      </div>
    </div>
  </div>
</section>
```


Home | Members | Profile | Logout

SpaceBook: Homer's Home Page

Friends

- marge, (drop)
- lisa, (drop)

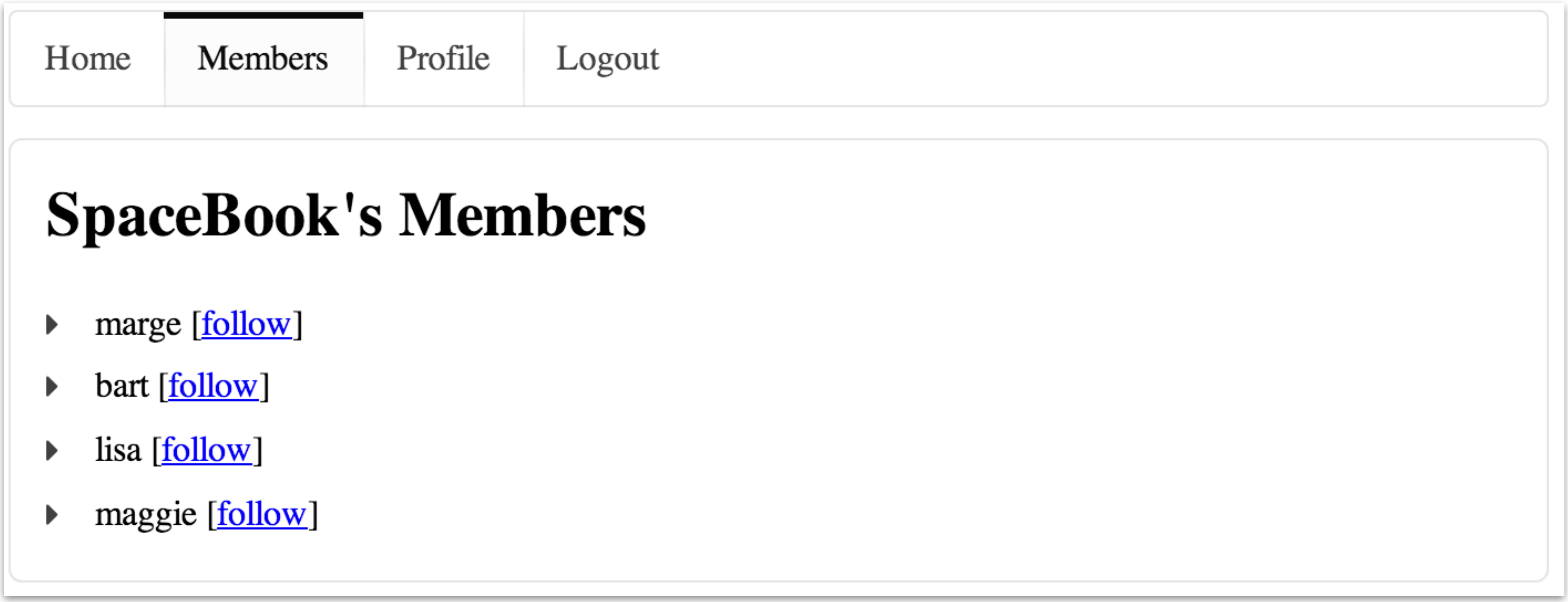
Messages

- marge says..."Hey there Homer, when are you going to work?"
- lisa says..."Move off the couch dad!"

Members

```java
public class Members extends Controller
{
  public static void index()
  {
    render();
  }

  public static void follow (String name)
  {
    Logger.info("Following " + name);
    index();
  }
}
```

```html
#{extends 'main.html' /}
#{set title:'Members' /}

<nav class="ui menu">
  <a class="ui item" href="/home">Home</a>
  <a class="ui active item" href="/members">Members</a>
  <a class="ui item" href="/profile">Profile</a>
  <a class="ui item" href="/login">Logout</a>
</nav>

<section class="ui segment">
  <h2 class="ui header">SpaceBook's Members</h2>
  <div class="ui list">
    <div class="item">
      <i class="right triangle icon"></i> marge [<a href="home.html">follow</a>]
    </div>
    <div class="item">
      <i class="right triangle icon"></i> bart [<a href="home.html">follow</a>]
    </div>
    <div class="item">
      <i class="right triangle icon"></i> lisa [<a href="home.html">follow</a>]
    </div>
    <div class="item">
      <i class="right triangle icon"></i> maggie [<a href="home.html">follow</a>]
    </div>
  </div>
</section>
```



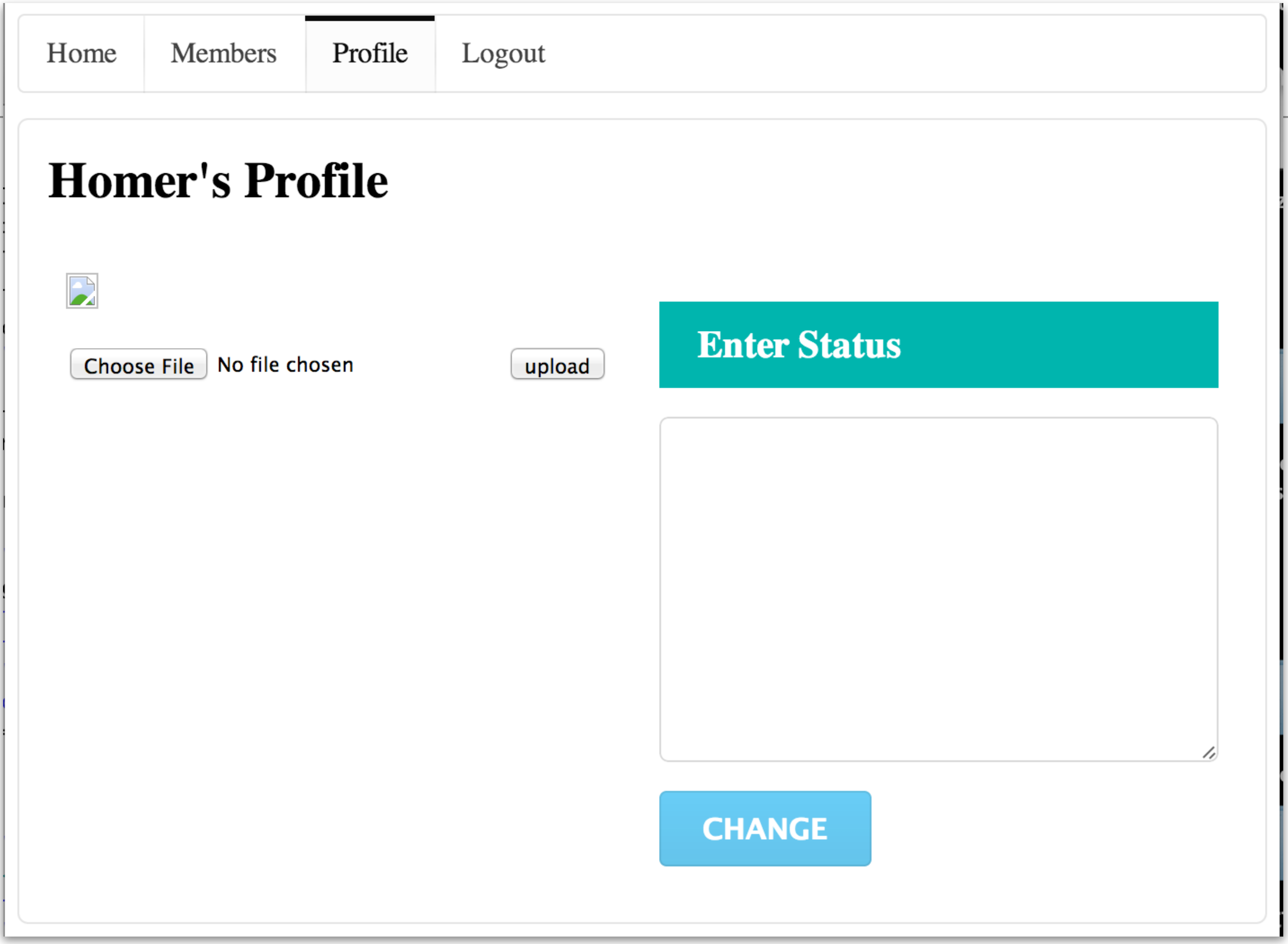
SpaceBook's Members

- marge [follow]
- bart [follow]
- lisa [follow]
- maggie [follow]

Profile

```
public class Profile extends Controller
{
  public static void index()
  {
    render();
  }
}
```

```
#{extends 'main.html' /}
#{set title:'Profile' /}

<nav class="ui menu">
  <a class="ui item" href="/home">Home</a>
  <a class="ui item" href="/members">Members</a>
  <a class="ui active item" href="/profile">Profile</a>
  <a class="ui item" href="/login">Logout</a>
</nav>

<section class="ui segment">
  <h2 class="ui header">Homer's Profile</h2>
  <div class="ui two column grid segment">
    <div class="ui row">
      <div class="ui column">
        <p>
          <img src="images/homer.gif" />
        </p>
        <form action="homeprofile/upload" method="post" enctype="multipart/form-data">
          <input type="file" name="userfile" value="" /> <input type="submit" name="submit" value="upload" />
        </form>
      </div>
      <div class="ui column form segment">
        <form action="homeprofile/changetext" method="post">
          <h3 class="ui inverted teal block header">Enter Status</h3>
          <textarea class="ui field" name="profiletext"> </textarea>
          <input class="ui blue button" type="submit" name="submit" value="Change" />
        </form>
      </div>
    </div>
  </div>
</section>
```

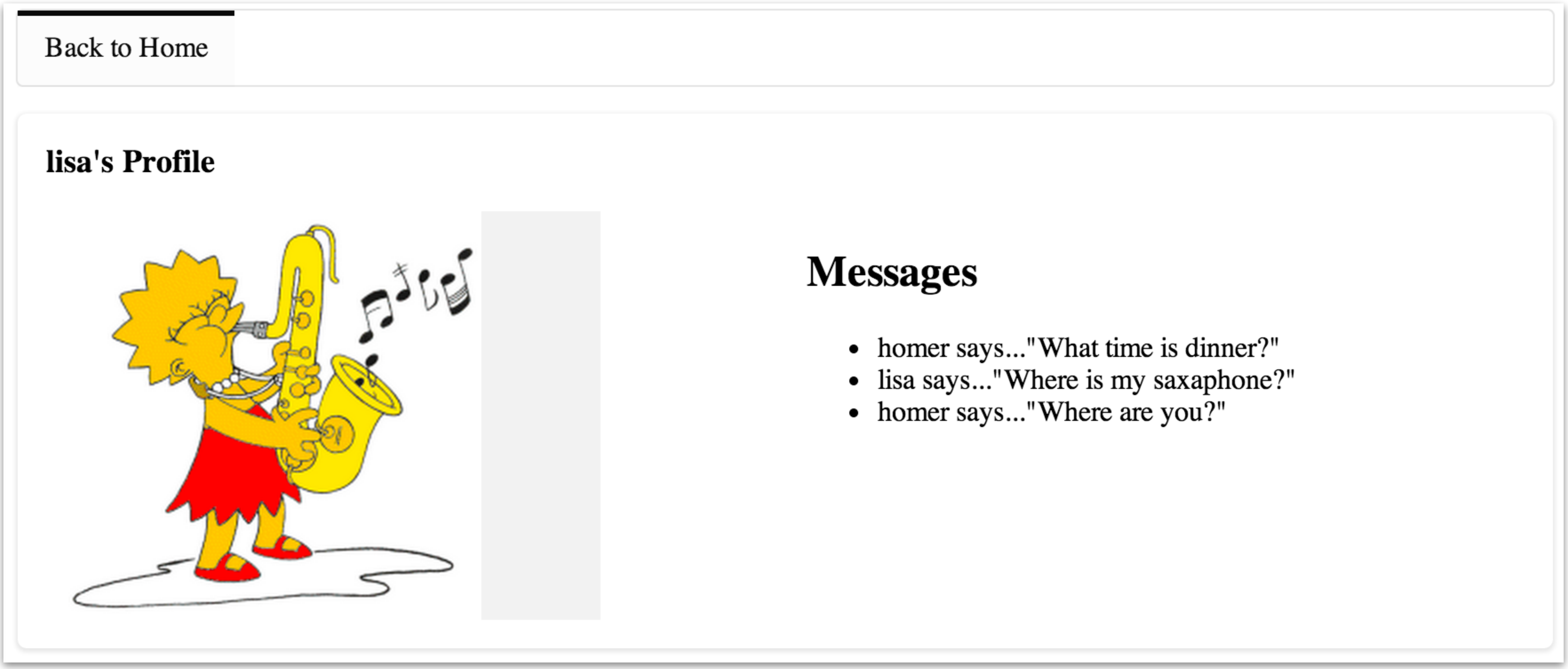


Homer's Profile

PublicProfile

```
public class PublicProfile extends Controller
{
  public static void index()
  {
    render();
  }

  public static void visit(String name)
  {
    Logger.info("Just visiting the page for " + name);
    render(name);
  }
}
```

```
#{extends 'main.html' /}
#{set title:'Spacebook' /}

<nav class="ui menu">
  <a class="ui active item" href="/home">Back to Home</a>
</nav>

<section class="ui raised segment">
  <div class="ui small header"> ${name}'s Profile</div>
  <section class="ui  two column grid segment">
    <div class="row">
      <div class="column">
        <div class="ui medium image">
          <img src="/public/images/${name}.gif"/>
        </div>
      </div>
      <div class="column">
        <h2> Messages </h2>
        <ul>
          <li>homer says..."What time is dinner?"</li>
          <li>lisa says..."Where is my saxaphone?"</li>
          <li>homer says..."Where are you?"</li>
        </ul>
      </div>
    </div>
  </section>
</section>
```



Back to Home

lisa's Profile

Messages

- homer says..."What time is dinner?"
- lisa says..."Where is my saxaphone?"
- homer says..."Where are you?"

Routes

```
# Routes
# This file defines all application routes (Higher priority routes
first)
# ~~~~

# Landing page
GET /                       Home.index

# Home page
GET /home                   Home.index
GET /home/drop/{name}       Home.drop

# Members page
GET /members                Members.index
GET /members/follow/{name}  Members.follow

# Profile page
GET /profile                Profile.index

# Public Profiles
GET /publicprofile/{name}   PublicProfile.visit

# Ignore favicon requests
GET /favicon.ico            404

# Map static resources from the /app/public folder to the /public path
GET /public/                staticDir:public

# Catch all
*   /{controller}/{action}  {controller}.{action}
```

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit