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

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit

Messages

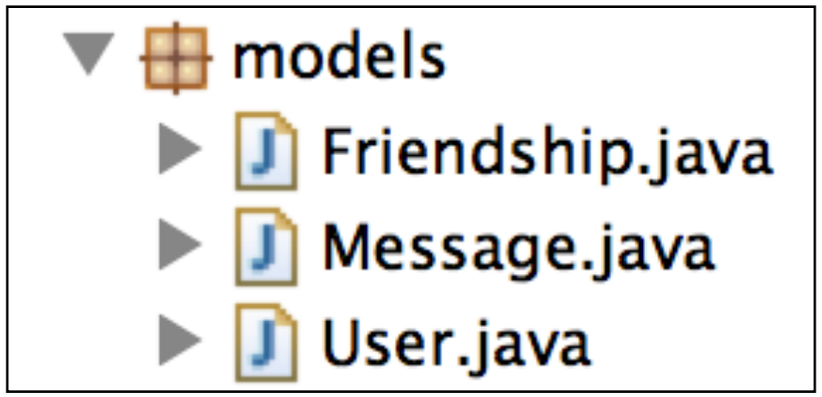
Lab12

PublicProfile

```
public class PublicProfile extends Controller
{
  public static void visit(Long id)
  {
    User user = User.findById(id);
    Logger.info("Just visiting the page for " + user.firstName + ' ' + user.lastName );
    render(user);
  }

  public static void sendMessage()
  {
    Logger.info("Just received a message from someone!");
    index();
  }
}
```

Messages Model




```
public class Message extends Model
{
  public String messageText;

  @ManyToOne
  public User from;

  @ManyToOne
  public User to;

  public Message(User from, User to, String messageText)
  {
    this.from = from;
    this.to = to;
    this.messageText = messageText;
  }
}
```

sendMessage Action - recipient

```
<form action="/publicprofile/sendmessage/${user.id}" class="ui form" method="post">
  <div class="field">
    <textarea name="messageText"></textarea>
  </div>
  <button  class="ui fluid blue labeled submit icon button">
    <i class="icon edit"></i> Send
  </button>
</form>
```

```
public static void sendMessage(Long id, String messageText)
{
  String userId = session.get("logged_in_userid");
  User fromUser = User.findById(Long.parseLong(userId));

  Logger.info("Message from user " +
      fromUser.firstName + ' ' + fromUser.lastName +" to " +
      messageText);

  visit(id);
}
```

- Discover who is sending the message - *fromUser*
- Log the name of the sender + the message to the console

sendMessage Action - recipient

```
public static void sendMessage(Long id, String messageText)
{
  String userId = session.get("logged_in_userid");
  User fromUser = User.findById(Long.parseLong(userId));

  User toUser = User.findById(id);

  Logger.info("Message from user " +
      fromUser.firstName + ' ' + fromUser.lastName +" to " +
      toUser.firstName + ' ' + toUser.lastName +": " +
      messageText);

  visit(id);
}
```

- Discover who the message is for - *toUser*

sendMessage Action - send the message!

```
public static void sendMessage(Long id, String messageText)
{
  String userId = session.get("logged_in_userid");
  User fromUser = User.findById(Long.parseLong(userId));
  User toUser = User.findById(id);

  Logger.info("Message from user " +
      fromUser.firstName + ' ' + fromUser.lastName +" to " +
      toUser.firstName + ' ' + toUser.lastName +": " +
      messageText);

  fromUser.sendMessage(toUser, messageText);
  visit(id);
}
```

- Invoke ‘sendMessage()’ method on User object

Extend User Model

- User model extended to incorporate ‘outbox’ and ‘inbox’ collections

- These store all the messages the user has sent, and also all those messages the user has received.

```
public class User extends Model
{
  ...

  @OneToMany(mappedBy = "to")
  public List<Message> inbox = new ArrayList<Message>();

  @OneToMany(mappedBy = "from")
  public List<Message> outbox = new ArrayList<Message>();

  ...
}
```

User Model - sendMesage()

- sendMessage method implement the messaging feature:
  - Create Message Object
  - Add it to the outbox of the sender
  - Add to inbox of recipient

```
public class User extends Model
{
  ...

  @OneToMany(mappedBy = "to")
  public List<Message> inbox = new ArrayList<Message>();

  @OneToMany(mappedBy = "from")
  public List<Message> outbox = new ArrayList<Message>();

  public void sendMessage (User to, String messageText)
  {
    Message message = new Message (this, to, messageText);
    outbox.add(message);
    to.inbox.add(message);
    message.save();
  }
  ...
}
```

```
public static void sendMessage(Long id, String messageText)
{
  String userId = session.get("logged_in_userid");
  User fromUser = User.findById(Long.parseLong(userId));
  User toUser = User.findById(id);

  Logger.info("Message from user " +
      fromUser.firstName + ' ' + fromUser.lastName +" to " +
      toUser.firstName + ' ' + toUser.lastName +": " +
      messageText);

  fromUser.sendMessage(toUser, messageText);
  visit(id);
}
```

```
public class User extends Model
{
  ...

  @OneToMany(mappedBy = "to")
  public List<Message> inbox = new ArrayList<Message>();

  @OneToMany(mappedBy = "from")
  public List<Message> outbox = new ArrayList<Message>();

  public void sendMessage (User to, String messageText)
  {
    Message message = new Message (this, to, messageText);
    outbox.add(message);
    to.inbox.add(message);
    message.save();
  }
  ...
}
```

Rendering the UserProfile

```
public class PublicProfile extends Controller
{
  ...
 public static void visit(Long id)
 {
    User user = User.findById(id);
    render(user);
 }
 ...
}
```

- Currently object for logged in user passed to view

```
<h3>Your Friend ${user.firstName} ${user.lastName}'s Home Page</h3>
```

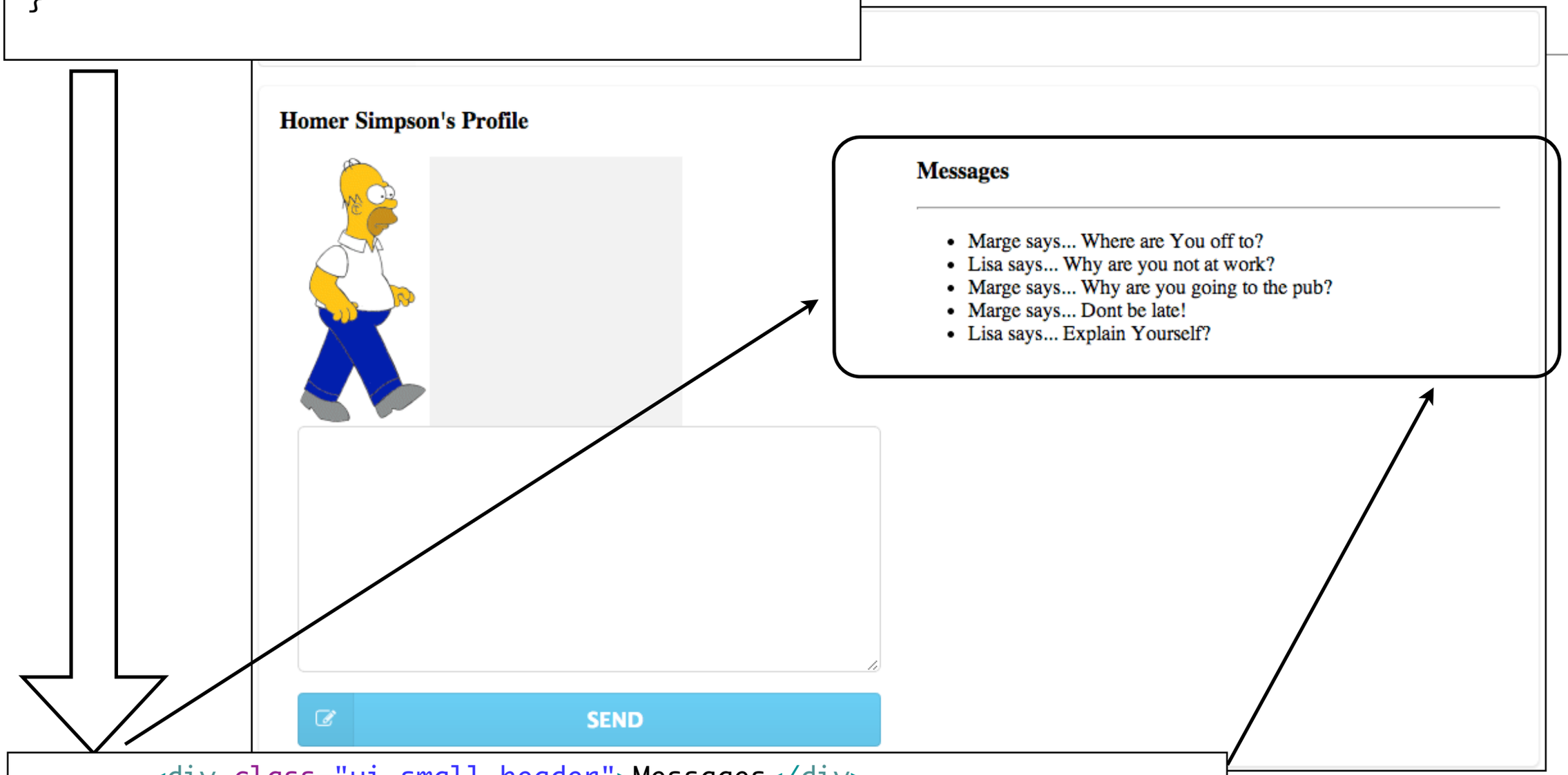
Rendering the UserProfile + Messages

```
<div class="ui small header">Messages</div>
<hr>
<ul>
  #{list items:user.inbox, as:'message'}
    <li>
      ${message.from.firstName} says...  ${message.messageText}
    </li>
  #{/list}
</ul>
```

Loop through the list and get each message into a variable called ‘message’

In ‘message’, display the first name of the sender + the message text in a <li> element

```
public static void visit(Long id)
{
    User user = User.findById(id);
    render(user);
}
```




```
<div class="ui small header">Messages</div>
<hr>
<ul>
  #{list items:user.inbox, as:'message'}
    <li>
      ${message.from.firstName} says...  ${message.messageText}
    </li>
  #{/list}
</ul>
```

Rendering the Messages on the Home Page

views/Home/index.html

```
<h2>Messages</h2>
<div class="ui list">
    #{list items:inbox, as:'message'}
      <div class="item">
        <i class="right triangle icon"></i> ${message.from.firstName} says...  ${message.messageText}
      </div>
    #{/list}
</div>
```