Web Development

Produced by

Eamonn de Leastar (edeleastar@wit.ie)
Dr. Brenda Mullally (bmullally@wit.ie)
Siobhan Drohan (sdrohan@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

Pictures

Pictures

- Profile controller can be equipped with two new actions:
    - upload a picture:
        - send picture to databased for the current user
    - ‘get’ a picture:
        - read back a picture from the database for some given id
- In both cases, the picture will be associated with the ‘id’ of the user
- In a Play model - a picture is represented by a ‘Blob’ class
    - BLOB = Binary Large Object

Four Steps:

1: Equip User Model with a Blob

2: Provide ‘Upload Picture’ route + action in HomeProfile

3: Provide ‘Get Picture’ route + action in HomeProfile

4: Invoke ‘Get Picture’ action in views/HomeProfile

5: Invoke ‘Upload Picture’ action in views/HomeProfile

6: Invoke ‘Get Picture’ action in views/UserProfile

1: Equip User Model with a Blob

- The Blob will be the field that holds the picture in the database

```
public class User extends Model
{
  public String firstName;
  public String lastName;
  public String email;
  public String password;
  public String statusMessage;
  public Blob   profilePicture;

  // as before...
}
```

2: Provide ‘Upload Picture’ route + Action in HomeProfile

```
POST    /profile/uploadpicture/{id}                 Profile.uploadPicture
```

- Must provide ID of user + the image data.
  - Locate the user in the database
  - Insert the picture
  - Save the changes

```
public static void uploadPicture(Long id, Blob picture)
{
  User user = User.findById(id);
  user.profilePicture = picture;
  user.save();
  index();
}
```

3: Provide ‘getPicture’ route + action in HomeProfile

```
GET     /profile/getpicture/{id}                    Profile.getPicture
```

- Look up the user in the database
- Read the picture from the correct field
- If there is actually a picture there
- render the image to the view as binary data

```
public static void getPicture(Long id)
{
  User user = User.findById(id);
  Blob picture = user.profilePicture;
  if (picture.exists())
  {
    response.setContentTypeIfNotSet(picture.type());
    renderBinary(picture.get());
  }
}
```

4: Invoke ‘Get Picture’ action in views/Profile

```
<h3>Profile Image</h3>
<img src="/profile/getpicture/${user.id}">
```

```
public static void getPicture(Long id)
{
  User user = User.findById(id);
  Blob picture = user.profilePicture;
  if (picture.exists())
  {
    response.setContentTypeIfNotSet(picture.type());
    renderBinary(picture.get());
  }
}
```

5: Invoke ‘Upload Picture’ action in views/HomeProfile

```
<form action="/profile/uploadpicture/${user.id}" method="post" enctype="multipart/form-data">
  <input type="file" name="picture" />
  <input type="submit" name="submit" value="upload" />
</form>
```

- Input type is ‘file’ - which will trigger browse of local file system and enable selection of any image file on disk

```
public static void uploadPicture(Long id, Blob picture)
{
  User user = User.findById(id);
  user.profilePicture = picture;
  user.save();
  index();
}
```

6: Invoke ‘Get Picture’ action in views/PublicProfile

```
<img src="/profile/getpicture/${user.id}"/>
```

THE END