■続・Play で遊んでみる

今日は，Play 2.x の Scala Templates でビュー＆フォーム操作から.

Template

テンプレートからテンプレートもよべる，ということらしい.
も少し慣れが必要そうだなあ ...

ユーザ入力フォームを作ってみる

models に追加した User クラスがに age があるのにサンプルでは age の分がなくて
apply/unapply がエラーになったので，"age" を追加してみた "-> text" じゃだめだったので,
"-> number" にしてみたけど，これって何だろう ...

```
object UserController extends Controller{
  val userForm = Form(
      mapping("name" -> text, "email" -> text, "age" -> number)
      (User.apply) (User.unapply)
      );
  def entryInit = Action {
    val filledForm = userForm.fill(User("user name", "email address", 0))
       Ok(views.html.user.entry(filledForm))
  }
  def entrySubmit = Action { implicit request =>
    val user = userForm.bindFromRequest.get
    println(user)
    Ok(views.html.user.entrySubmit())
  }
}
```

entry.scala.html も

```
    <fieldset>
        <legend>input user info.</legend>
         @helper.inputText(userForm("name"))
         @helper.inputText(userForm("email"))
         @helper.inputText(userForm("age"))
    </fieldset>
```

のように変更.

バリデート

バリデート処理がかけるらしい．number もバリデートなのだな.
Forms のところにバリデートはのってる
http://www.playframework.com/documentation/2.2.3/api/scala/index.html#play.api.data.Forms$
number には最大値 / 最小値も指定できるようだ.

```
 val userForm = Form(
     mapping("name" -> nonEmptyText, "email" -> email, "age" -> number(min=10, max=90))
     (User.apply) (User.unapply)
     );
```

としてみた.

```
def entryInit = Action {
  val filledForm = userForm.fill(User("user name", "email address", 30))
      Ok(views.html.user.entry(filledForm))
}
```

デフォルト値のまま送信すると "email address" がメールアドレスとして不適切なのでエラーになって
デバッグ的な表示が.
アプリとしては，エラー処理が必要だよね.
と，思ったら直後にエラー処理の書き方が説明してあった.

```
def entrySubmit = Action { implicit request =>
  userForm.bindFromRequest.fold(
    errors => {
      println("error!")
      BadRequest(views.html.user.entry(errors))
    },
    success => {
      println("entry success!")
      Ok(views.html.user.entrySubmit())
    })
}
```

次は，データベースとの連携 ... らしいので，まず MySQL をインストール

ちなみに，なぜか xcodebuild が実行できないとかいうので，xcodebuild を単独で実行してみると

```
xcode-select: error: tool 'xcodebuild' requires Xcode, \
 but active developer directory '/Library/Developer/CommandLineTools' is a command line tools
instance
```

とエラーをはいていた．というわけで,

```
sudo xcode-select -switch /Applications/Xcode.app/Contents/Developer
```

としてから MySQL をインストール.

```
sudo port install mysql56 +openssl +system_readline
sudo port install mysql56-server
sudo /opt/local/lib/mysql56/bin/mysql_install_db --user _mysql
sudo port load mysql56-server
sudo port select mysql mysql56
```

MySQL で DB を用意

まずは，ユーザを作る.

```
mysql -u root mysql
```

で MySQL にアクセスして

```
create user 'miyo'@'localhost' identified by PASSWORD
```

で，まあ，ローカルのマシンだし，ということで ...

```
grant all on *.* to 'miyo'@'localhost';
```

とか.

```
show grants for miyo@localhost;
```

で確認．MySQL から切断して,

```
mysql -u miyo -p
```

で作成した miyo でログイン.

```
create database play_test;
```

でデータベース作った.

```
show databases;
```

で，作ったデータベースを確認.

Play に MySQL の設定をする

conf/application.conf の database 関連のパラメタを設定

```
db.default.driver=com.mysql.jdbc.Driver
db.default.url="jdbc:mysql://localhost/play_test"
db.default.user=miyo
db.default.password=miyo
db.default.logStatements=true
```

projects/Build.scala を設定 ... とあるけど，build.sbt に置き換えられたらしい.
mysql を追加

```
libraryDependencies ++= Seq(
  "mysql" % "mysql-connector-java" % "5.1.30",
  jdbc,
  anorm,
  cache
)
```

play コンソールで

```
reload
update
```

とする．play コンソールで

```
run
```

として Web ブラウザから http://localhost:9000/sample1 にアクセスすると,
DB にアクセスできないというエラーが.
MySQL の skip-network が有効になっているのが問題なようなので,
/opt/local/etc/mysql56/macports-default.cnf の skip-network をコメントアウト.

```
sudo port unload mysql56-server
sudo port load mysql56-server
```

として，アクセスできた．よかった，よかった.

■メモ

いつか使うかも - 高クオリティ！商用利用 OK のアイコン、イラスト無料 (フリー) 素材まとめ