Play Framework

One Web Framework to rule them all

Felix Müller

Agenda

- Yet another web framework?
- Introduction for Java devs
- Demo
- Summary

Yet another web framework?

Yet another web framework?

- Why do we need another web framework?
- Existing solutions: Servlets, Ruby on Rails, Grails, Django, node.js

Yet another web framework?

Threaded

- 1 thread per request
- threads may block during request processing

Evented

- 1 thread per cpu core
- threads shall never block

Yet another web framework?

Threaded

- Servlets
- Ruby on Rails
- Grails
- Django

Evented

- Node.js
- Play Framework

Yet another web framework?

- Play is completely asynchronous
- horizontally scalable out of the box

and a lot other goodies...

Introduction for Java devs

Play Framework

- MVC pattern
- Scala and Java API
- asset compiler for CoffeeScript and LESS
  + Google Closure Compiler
  + require.js

Play Console

- `play new <appName>`
- `play compile|test|run|debug`
- `play ~compile|test|run`

Application structure

- app directory contains source code, templates and assets
- standard packages based on MVC:
  - app/controllers
  - app/models
  - app/views

Application structure

- public directory is default for assets as css and javascript files
  - public/stylesheets
  - public/javascripts
  - public/images
- served directly by web server

Routes configuration

- contains url to controller mapping
- statically typed
- pattern: <HTTP method> <url> <controller>

routes

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

# The home page
GET     /                                   controllers.Projects.index()

# Authentication
GET     /login                              controllers.Application.login()
POST    /login                              controllers.Application.authenticate()
GET     /logout                             controllers.Application.logout()

# Projects
POST    /projects                           controllers.Projects.add()

POST    /projects/groups                    controllers.Projects.addGroup()
DELETE  /projects/groups                    controllers.Projects.deleteGroup(group: String)
PUT     /projects/groups                    controllers.Projects.renameGroup(group: String)

DELETE  /projects/:project                  controllers.Projects.delete(project: Long)
PUT     /projects/:project                  controllers.Projects.rename(project: Long)

POST    /projects/:project/team             controllers.Projects.addUser(project: Long)
DELETE  /projects/:project/team             controllers.Projects.removeUser(project: Long)

# Tasks
GET     /projects/:project/tasks            controllers.Tasks.index(project: Long)
POST    /projects/:project/tasks            controllers.Tasks.add(project: Long, folder: String)
PUT     /tasks/:task                        controllers.Tasks.update(task: Long)
DELETE  /tasks/:task                        controllers.Tasks.delete(task: Long)

POST    /tasks/folder                       controllers.Tasks.addFolder()
DELETE  /projects/:project/tasks/folder     controllers.Tasks.deleteFolder(project: Long, folder: String)
PUT     /project/:project/tasks/folder      controllers.Tasks.renameFolder(project: Long, folder: String)

# Javascript routing
GET     /assets/javascripts/routes          controllers.Application.javascriptRoutes()

# Map static resources from the /public folder to the /public path
GET     /assets/*file                       controllers.Assets.at(path="/public", file)
```

Controllers

```
import play.mvc.*;

public class Application extends Controller {

  public static Result index() {
    return ok("It works!");
  }
}
```

Templates

- built-in Scala based template engine
- templates are type safe
- templates are just functions

Templates

```
@(customer: Customer, orders: List[Order])

<h1>Welcome @customer.name!</h1>

<ul>
@for(order <- orders) {
  <li>@order.getTitle()</li>
}
</ul>
```

Models

- are just POJOs
- getters and setters are generated by Play
- often Active Record pattern
- Ebean as default ORM

State in Play

- session state is stored only client-side
- cookie is used for this (4KB per user)
- cookie is signed with secret key

State in Play

- session is added to each http request
- session is not invalidated automatically
- Flash scope: state for the next request, cookie is not signed

Testing Support

- Play comes with decent testing support
- Helper classes for mocking and faking everything
- Selenium and FluentLenium for browser testing

Testing Support

```
@Test
public void callAction() {
  Result result = callAction(
    controllers.routes.ref.Application.index()
  );

  assertThat(status(result)).isEqualTo(OK);
  assertThat(contentType(result))
    .isEqualTo("text/html");
  assertThat(contentAsString(result))
    .contains("It works");
}
```

Testing Support

```
running(
  testServer(3333,
    fakeApplication(inMemoryDatabase())),
  HTMLUNIT,
  new Callback<TestBrowser>() {
    public void invoke(TestBrowser browser) {
      browser.goTo("http://localhost:3333");
      assertThat(browser.pageSource())
        .contains("application is ready");
    }
  }
);
```

Deployment

- `play start` for interactive mode
- `play stage` for automated deployments
- `play dist` for standalone distributions

Deployment

- or deploy in Java EE web container:

  github.com/dlecan/play2-war-plugin

Async Features

- never block, especially when dealing with long computations or legacy 3rd party web services
- return the promise of a result: Promise<Result>
- only the client will be blocked while waiting for response

Async Features

```
public static Promise<Result> asyncComp() {
  Promise<Double> promiseOfPi = Promise.promise(
    new Function0<Double>() {
      public Double apply() { return Math.PI; }
    });

  return promiseOfPi.map(
    new Function<Double, Result>() {
      public Result apply(Double pi) {
        return ok("Got PI! " + pi);
      }
    });
}
```

Demo

Summary

Summary

- Play is a modern web framework on the JVM
- has great developer experience
- facilitates the right patterns for the cloud and enterprise environments

Thank you! Questions?

Felix Müller
@fmueller_bln