УДК 004.4'23

В.В. Герасимов, І.В. Пономарьов, І.О. Щетінін

ОПТИМІЗАЦІЯ ПРОЦЕСУ РОЗРОБКИ ІНТЕРНЕТ-ДОДАТКІВ НА ПЛАТФОРМІ JAVA

Анотація. Розглянуто Play Framework, який значно спрощує та пришвидшує процес розробки Інтернет-додатків на платформі Java. Описано особливості використання та переваги цієї технології. Представлено результати порівняння Play Framework з альтернативними технологіями.

Ключові слова: Play Framework, MVC, HTTP, View, Controller, JPA, log4j, Groovy, Selenium, JSP, Spring Framework, HTML, JSON, XML, MySQL.

Постановка проблеми. Активний розвиток глобальної мережі Internet призвів до виникнення та популярності web-додатків. Існує безліч технологій для їх створення, проте не завжди процес розробки є простим та швидким. Наприклад, часто вирішення таких типових задач як взаємодія з базами даних або тестування додатків вимагає значних зусиль від розробника та є причиною написання «надлишкового» коду. Розробники нової технології Play Framework спробували вирішити цю проблему.

Основна частина. Play framework є альтернативою «роздутим» Java EE технологіям. Він фокусується на продуктивності розробки та цілях RESTfull архітектури. Play є дуже ефективним засобом при гнучкій розробці програмного забезпечення. Мета Play framework — дотримуючись Java, зробити розробку web додатків простішою. За думкою його розробників простішими речами можна зробити складніші речі.

Play — це чистий Java фреймворк, який дозволяє користуватись улюбленими засобами розробки та бібліотеками. При наявності досвіду користування платформою розробки Java не має необхідності переходити на іншу мову, інше IDE та бібліотеки. Достатньо просто перейти на продуктивніші технології Java.

Розглянемо більш прискіпливо головні особливості Play framework'у.

HTTP маршрутизація (HTTP-to-code mapping). Інші Java Web фреймворки такі як Servlet API або Struts framework змушують мати справу з абстрактним відображенням HTTP протоколу та дивними API концепціями.

Play побудований інакше: фреймворк web-додатків має повний доступ до HTTP та його концепцій. Це одна з фундаментальних різниць між Play та іншими Java web-фреймворками.

HTTP, модель запит/відповідь, архітектурний стиль REST, URI — це головні ідеї Play framework.

Наприклад, зв'язування URI з викликом Java виконується одним рядком:

GET /clients/{id} Clients.show

Ефективна шаблонна система. Ідея JSP та мови виразів JSTL досить непогана. Проте необхідно багато конфігураційних файлів для створення бібліотеки тегів. Також не має можливості мати доступ до базової моделі об'єкта. JSP має багато обмежень і це дійсно розчаровує. Play використовує шаблонну систему на кшталт JSP, яка позбавлена цих недоліків.

Наприклад, в JSP та JSTL маємо такий програмний код:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"
%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" pre-
fix="fn" %>
<c:choose>
 <c:when test="${emails.unread != null &&
fn:size(emails.unread)}">
 You have ${fn:size(emails.unread)} unread email(s)!
 </c:when>
 <c:otherwise>
 You have no unread emails!
 </c:otherwise>
</c:choose>
```

Еквівалент в шаблонній системі Play буде виглядати наступним чином:

```
You have ${emails.unread ?: 'no'}
${emails.unread?.pluralize('email')} !
```

Мова виразів, що використовується шаблонною системою Play, називається Groovy, та є послідовником мови Java. В той час поки Play головним чином використовує шаблонну систему для побудови HTML відповідей, розробники мають можливість використовувати її для генерації будь-яких інших документів, таких як e-mail повідомлення, сторінки JSON формату та ін.

Більш потужне JPA. JPA — це в чистому вигляді засоби об'єктно-реляційного представлення даних (object-relational mapping — ORM), доступні для мови Java. Робота з JPA в Play стала ще простішою та ефективнішою. Без спеціального налаштування Play автоматично запускає менеджер сутностей (JPA Entity Manager) та чарівним образом синхронізується, коли перезавантажується програмний код.

Більш того, якщо використовувати суперкласс play.db.jpa.Model, він стане у нагоді в прагненні зробити програмний код правильнішим та кращим. Наприклад:

```
public static void messages(int page) {
User connectedUser = User.find("byEmail", connected()).first();
List<Message> messages = Message.find("user = ? and read = 
false order by date desc", connectedUser).from(page * 10).fetch(10);
render(connectedUser, messages);
}
```

Розробка через тестування. Інтегрований запуск тестів в Play значно полегшує їх розробку. Є можливість писати всі види тестів, від простих до функціональних, та запускати їх спеціальними play командами в командному рядку за допомогою середовища Selenium. Керувати виконанням тестів можна через браузер. Підтримуються розрахунки тестового покриття. Якщо підключити спеціальний модуль, наприклад Cobertura, то можна отримати детальніші результати тестування.

Повноцінний фреймворк додатків. Play framework має всі засоби для створення сучасного web додатку:

- Підтримка реляційних баз даних через JDBC.
- Об'єктно-реляційне представлення даних через Hibernate (з JPA API).
- Інтегрована підтримка кешу з легким використанням системи розподіленого кешу в пам'яті.

- Пряме використання web-сервісів або в JSON, або в XML (маються на увазі реальні web-сервіси, не SOAP).
- Підтримка OpenID для розширеної аутентифікації.
- Web додаток може бути розгорнутим де завгодно (серверах додатків, сервісах Google App Engine, хмарах тощо).
- API для роботи із зображеннями.
- Модульна архітектура дозволяє комбінувати додаток з іншими додатками та технологіями.

Чому саме Play Framework. Фреймворк розроблений для того, щоб якомога швидше почати розробляти і при цьому відразу бачити проміжний результат.

Фреймворк має вбудований (embeded) web-сервер і настроюваний classloader. Які можливості це дає? У першу чергу запуск проекту без попереднього налаштування серверів Tomcat або подібних та перекомпіляція вихідних кодів «на льоту». Тобто при додаванні нового обробника запиту достатньо перезапустити сторінку в браузері. Винятки треба зробити для прекомпільованних ресурсів: плагінів, бібліотек тощо. При додаванні бібліотеки додаток необхідно перезапускати.

- Фреймворк вже включає в себе інструменти першої необхідності і дозволяє не витрачати часу на їх початкову настройку.

Наприклад, в Play Framework реалізовані власні бібліотеки для спрощення роботи з JPA та log4j. Не потрібно реалізовувати підключення до бази даних в програмному коді за допомогою JDBC, це можна легко зробити в файлі конфігурації додатку application.conf. Для того, щоб Java клас став моделлю JPA в Play Framework він має бути нащадком класу play.db.jpa.Model, його налаштування здійснюється за допомогою анотацій визначених в пакеті play.db.jpa. Для логування Play надає стандартний клас play.Logger, налаштувати рівні логування можна в файлі conf/log4j.properties. Розробниками Play створено безліч модулів для вирішення типових задач. Наприклад, модуль Secure реалізовує авторизацію та реєстрацію користувачів. Модулі легко підключаються, налаштовуються та мають відкритий код.

- Зручний template-engine.

За умовчанням використовується Groovy, проте можна використовувати і іншу технологію, наприклад Japid. Groovy надає багато

можливостей. Це і побудова логічних умов в шаблонах, робота зі списками та масивами, оголошення змінних, шаблонне наслідування, багато зручних тегів, наприклад, #{form}, та ін.

- Фреймворк має відкритий код та підтримує сторонні плагіни.

Вже існує безліч сторонніх плагінів, що дозволяють, наприклад, інтегруватися з MongoDB, GAE тощо.

- TDD (керування запуском тестів).

За допомогою команди play test додаток запускається в режимі виконання всіх unit, функціональних та selenium тестів.

MVC в Play Framework. Модель MVC (рис. 1) розділяє додаток на окремі прошарки: прошарок презентації та прошарок моделей (model). Прошарок презентації далі розділяється на представлення (view) та прошарок контролерів (controller).

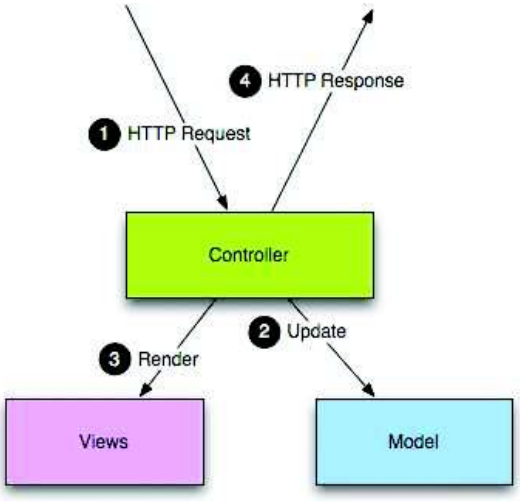

Рисунок 1 - Модель MVC в Play framework

Модель (model) є предметно-орієнтоване подання інформації, з якою працює додаток. Предметна логіка додає сенс даним — наприклад, розрахунки днів народжень користувачів, кошиків покупців тощо. Більшість додатків використовують постійний механізм зберігання даних, такий як база даних.

Представлення (view) надає моделі форми, придатні для взаємодії. Зазвичай це інтерфейс користувача. Кілька уявлень можуть існувати для однією моделі, але для різних цілей. В web додатках представлення зазвичай надаються у web форматах, таких як HTML, XML або JSON. Однак є деякі випадки, коли подання може бути виражено в двійковому вигляді, наприклад, динамічні графіки діаграм.

Контролер (controller) відповідає на події (зазвичай дії користувача), обробляє їх та може також змінювати логіку моделей. В web-додатках подіями зазвичай є HTTP запити: контролер приймає HTTP запит, витягує необхідні дані з події, такі як рядок параметрів запиту, заголовок запиту тощо. За допомогою логіки модель обробляє подію та надсилає результат роботи в представлення (view), тобто в інтерфейс користувача.

В Play ці три прошарки визначені в директорії проекту, кожен в окремому пакеті.

app/controllers. Контролер — це Java клас, де кожен статичний, відкритий метод є дією (action). Управління передається в такі методи, коли надходить відповідний запит. Код в контролерах нагадує швидше процедурно-орієнтоване програмування, ніж об'єктно-орієнтоване. Бо складається з набору методів, кожен з яких відповідає за обробку певного запиту. Ці методи витягують необхідні дані з HTTP запитів, працюють з об'єктами моделей та відправляють назад результати, які упаковуються в HTTP відповідь.

app/models. Прошарок моделей являє собою набір Java класів, який використовує всі об'єктно-орієнтовані особливості, доступні мові Java. Вони містять структуру даних та операції, на яких функціонує додаток. Кожного разу, коли об'єкти моделей повинні бути збережені в постійному сховищі, вони можуть включати деякі артефакти, такі як JPA анотації або SQL вирази.

app/views. Більшість представлень (або шаблонів) додатку генеруються ефективним шаблонним движком, який надається Play. Щоб відобразити необхідну інформацію контролер використовує шаблон. Цей пакет підтримує HTML, XML, JSON та інші формати шаблонів зі спеціальними директивами для динамічного створення моделі подання.

Результати порівняння Play framework з популярними технологіями розробки Інтернет-додатків такими як JSP (Java Server Pages) та Spring Framework наведено нижче. Кожна характеристика оцінювалась за п'ятибальною шкалою.

 ISSN 1562-9945

Таблиця 1

Порівняння Play Framework з альтернативними технологіями

	Java Server Pages	Spring Framework	Play Framework
Тестування	++	++++	+++++
	JSP не має вбудованих засобів тестування. Для цього використовуються сторонні програмні продукти. В Spring Framework є спеціальні бібліотеки для тестування, за основу яких взято JUnit, проте запускаються тести за допомогою програм збірки проектів Java (наприклад maven). Play Framework може запускати додаток в режимі виконання всіх тестів. За допомогою бібліотеки play.test створювати unit, функціональні та selenium тести набагато простіше.		
Сервер web додатків	+++	+++	++++
	Додатки Java Server Pages та Spring Framework розгортаються та функціонують на серверах web додатків Java від сторонніх виробників. Наприклад Tomcat, GlassFish, JBoss, Jetty, WebLogic і т.д. Play Framework містить вбудований web сервер. Також play додатки можуть бути розгорнуті на сторонніх серверах, як і Java Server Pages та Spring Framework додатки.		
Робота з базами даних.	+	++++	+++++
	Java Server Pages. Взаємодія з серверами баз даних є досить кропіткою роботою. Треба створювати джерело даних JDBC та пул підключень, пишучи при цьому програмний код. **Spring Framework.** Можна використовувати Spring JDBC, DAO та ORM. Spring забезпечує повне абстрагування питань підключення до баз даних та управління підключеннями. Також надає безліч допоміжних класів DAO, підтримує декілька технологій ORM, таких як Hibernate, JDO, iBatis. **Play Framework** має власний SQL сервер. В Play Framework визначені бібліотеки ORM JPA. За допомогою цих бібліотек можна реалізувати моделі баз даних та необхідну логіку роботи з ними, а тому немає потреби використовувати JDBC.		
Модульність додатків	++	+++++	+++++
	В додатку, побудованому суто на JSP та сервлетах-контролерах, напевно найважче використовувати допоміжні програмні модулі. З іншого боку існує велика кількість маленьких фреймворків, які входять до складу Spring Framework і можуть виконувати такі допоміжні функції як авторизація, аутентифікація, управління транзакціями, повідомленнями і т.д. Будь-яку складову частину цієї технології під час розробки можна замінити на іншу або створити власний аналог. Play Framework — модульно-орієнтовна технологія, розробниками якої вже створено досить багато готових модулів. Також досить просто написати власний модуль. Керування модулями в Play Framework дуже зручне.		

Висновки. В сучасному світі Internet технологій створено досить багато засобів розробки web-додатків, проте більшість з них є досить складними та незручними у використанні, що призводить до збільшення часу розробки та складності програмного коду. Play Framework — це молода технологія, проте вона вже набула популярності. Її розробники активно працюють над усуненням недоліків та випускають нові версії цієї технології.

Завдяки Play Framework процес створення web-додатків стає простим та швидким. Для функціонування play додатків не потрібне додаткове програмне забезпечення, на кшталт сервера web-додатків Tomcat і подібних, можна використовувати власні засоби даного фреймворку. Значно спрощена робота з базами даних завдяки підтримці JPA. Допоміжні модулі надають можливості для вирішення багатьох задач, з якими повсякчас зіштовхуються розробники. Модулі мають відкритий програмний код та по суті є звичайними play додатками, які встановлюються та налаштовуються досить просто.

ЛІТЕРАТУРА

1. James Ward. Tutorial: Play Framework, JPA, JSON, jQuery, & Heroku. — Режим доступу:
http://www.jamesward.com/2011/12/ 11/tutorial-play-framework-jpa-json-jquery-heroku
2. Play Framework. Manual, tutorials & references. — Режим доступу: http://www.playframework.org/documentation/1.2.4/home
3. Isa Goksu. Writing Testable Code // "Methods & Tools" Software Development Magazine — Programming, Software Testing, Project Management, Jobs — Режим доступу:
http://www.methodsand-tools.com/archive/archive.php?id=103
4. Alexander Reelsen. Play Framework: Data Validation Using Controllers // Packt Publishing — Режим доступу:
http://www.packtpub.com/article/play-framework-data-validation-controllers
5. Ray Gomez. Develop using the Play! Framework: The JPA Model // The Buzz Media — 2009 — Режим доступу:
http://www.thebuzzmedia.com/develop-using-the-play-framework-the-jpa-model/
6. Groovy. User Guide. — Режим доступу:
http://groovy.codehaus.org/User+Guide