Някои технологични изводи породени от Java Enterprise Edition 6

Адриан Митев, Петър Сигалов

Innovations in Java Enterprise Edition 6: *The paper justifies the necessity of information about the new and noteworthy features in the latest version of Java Enterprise Edition. New and updated features are specified for each of the most used technologies in the platform with additional information about how these new features impact the way of using and development with the platform.*

Key words: *Programming, Java, Java Enterprise Edition, Object-oriented programming.*

ВЪВЕДЕНИЕ

В днешно време значително нарастват нуждите от разпределени, транзакционни и преносими приложения, които осигуряват бързодействие, сигурност и надеждност. Корпоративните приложения носят значителни ползи на бизнеса увеличавайки производителността и намалявайки разходите. Приложенията трябва да бъдат централизирани (с централизирано управление) и да могат да комуникират с други приложения, трябва да бъдат планирани, построени и вкарани в употреба с по-малка цена, за по-бързи срокове и чрез използване на по-малко ресурси.

Java Enterprise Edition е платформа за разработка на корпоративни приложения и информационни системи обслужващи голям брой потребители базирана на Java технологията. Основната цел на платформата е да предостави на разработчиците мощен набор от готови програмни интерфейси, намалявайки времето за разработка и сложността и подобрявайки производителността и скалируемостта на приложенията.

Целта на настоящата публикация е представяне на нововъведенията в новата версия на Java Enterprise Edition 6 и предимствата на тези нововъведения спрямо предходните версии на платформата.

ИЗЛОЖЕНИЕ

1. Нуждата от следваща версия на Java Enterprise Edition

Възникването на Java Enterprise Edition 6 е породено от необходимостите и предложенията за подобрение очертали се по времето на тригодишното съществуване на Java Enterprise Edition 5. Въпреки, че в пета версия на платформата много от технологиите биват значително подобрени (а някои дори разработени от нула) се налагат допълнителни промени, които да спомогнат за по-лесната разработка на приложения с платформата.

2. Основни характеристики на платформата

2.1. Опростен конфигурационен модел

Средата използва значително опростен модел в сравнение с предишни версии. XML файловете използвани за конфигуриране вече са опционални, а вместо тях разработчиците могат да използват Java анотации директно в кода. При стартиране на приложението сървърът прочита тези дефиниции и конфигурира компонентите на база прочетените мета данни. Под мета данни се има в предвид текстови записи с описателна информация за други данни или източници на информация (данни за данните). Чрез използване на анотации, необходимите конфигурационни параметри вървят ръка за ръка с програмния елемент, който те конфигурират, а това значително подобрява подредеността и читаемостта на програмния код.

2.2. Поддръжка на „инжектиране на зависимости”

„Инжектиране на зависимости” (Dependency injection) е програмен подход, при който референция към външна зависимост (компонент) се инжектира в друг компонент, чиято работа е пряко свързана с външния компонент. Под външен

компонент се разбира софтуерен компонент, обикновено със затворен и вече компилиран код, който не може да бъде променян, а може единствено да бъде използван като спомагателно средство за извършване на определена работа. Целта на подхода е да се раздели мениджмънтът на външния компонент и жизнения му цикъл от компонента, който го използва. Този подход се реализира, като позволява да бъдат „инжектирани" ресурси почти от всякакъв тип и автоматично се грижи за жизнения цикъл на външните компоненти. Местата, където е необходим външен компонент се описват чрез Java анотации и преди извикване на дадения код, сървърът удовлетворява необходимите външни зависимости.

2.3. Профили

Профилите представляват вграден набор от конфигурации (модули и библиотеки) на платформата, даващи възможност за избор от страна на потребителя да включи само тези, които ще използва, намалявайки значително използваните от приложния сървър ресурси. Двата основни профила са "Web Profile" – профил за уеб приложения и "Full Profile" – профил за корпоративни приложения използващи пълната мощ на Java Enterprise Edition.

2.4. Стандартизиране на имената на компонентите

Вградените компоненти на платформата могат да бъдат достъпени чрез стандартизирани имена (JNDI имена) за разлика от пета версия, където всеки приложен сървър използва различни имена, което прави трудна и бавна миграцията от приложен сървър на един софтуерен доставчик към приложен сървър на друг доставчик.

2.5. Разширяемост

Новата версия е замислена така, че технологиите в нея да могат да бъдат лесно разширени от разработчиците, които я използват. Това включва добавяне на нови модули и функционалности.

3. Новости в технологиите

3.1. Servlet 3.0

Нововъведенията във версия 3.0 на Servlet спецификацията (JSR-315) значително улесняват настройките и преносимостта на приложенията и разработката на преизползваеми модули (написани веднъж те могат да се използват в множество приложения). Към новите функционалности влизат:

- Описанието на сървлета в web.xml става опционално – в предходните версии всеки сървлет трябва да бъде описан в конфигурационния файл на приложението (web.xml) и при добавяне на преизползваем модул е необходимо сървлетите и сървлет филтрите от модула да бъдат конфигурирани ръчно, което внася неудобства и забавя работата. Този проблем е решен, като сървлетите се описват с Java анотации директно в класа на сървлета.
- Асинхронно извикване на сърлети - ако сървлетът изпълнява времеемка операция, той може да бъде изпълнен в отделна нишка и да върне отговор към клиента още преди изпълняваната задача да завърши (без да е необходимо да се изчаква приключването на операцията) а след това в последващи заявки да се обработи резултата от операцията.
- Поддръжка на HttpOnly cookies – cookies (бисквитки или кукита) представляват малки фрагменти информация, които се изпращат от браузъра към сървъра и обратно при всяка потребителска заявка и носят специфична за текущия клиент информация. HttpOnly бисквитките не позволяват тази информация да бъде четена и променяна от клиентски скриптове (например JavaScript), с което да се предизвика хакерска атака.

3.2. JSF 2.0

JavaServer Faces 2.0 (JSR-314) е изцяло преобразена версия на технологията. Основната промяна е в езика и механизма за описание на потребителския интерфейс. Остарялата технология JSP вече не се използва а на нейно място се използва механизмът за шаблони наречен Facelets. Основните промени и допълнения в новата версия са:

- Project stage – етап на проекта. Конфигурира се в web.xml и чрез него се указва на програмния интерфейс в какъв етап на разработка е проекта. Възможните стойности са – *Production, UnitTest, SystemTest и Extension Development*. Въз основа на тази конфигурация технологията се държи по различен начин. Например, ако е в режим на *Production* се оптимизира работата така, че да се получи максимална производителност, а ако е в режим на *Development* се визуализира дебъг информация, която при реална работа на приложението би намалила бързодействието на системата.
- Вградена функционалност за мениджмънт на ресурсите използвани от приложението и компонентните библиотеки – различните уеб компоненти и приложения се нуждаят от статични ресурси като JavaScript и CSS. JSF2 позволява всеки уеб компонент да укаже от кои статични ресурси се нуждае и къде се намират те. При използването на уеб компонент JSF изпраща към браузъра и мета данни заявяващи необходимите ресурси.
- Подобрен механизъм за обработка на събития - към почти всяко събитие от жизнения цикъл на потребителската заявка до изпращане на отговор до потребителя може да се извърши обработка чрез добавяне на съответен метод/клас, който бива извикан при настъпване на определеното събитие. Пример за такива събития са: промяна на стойност от потребителя, зареждане на страница, натискане на бутон от потребителя, извикване на валидация и други.
- Вградена поддръжка на AJAX - вградена поддръжка на AJAX, която позволява да се изпращат асинхронни заявки и да се обработва резултатът им, без да се използва JavaScript, единствено декларативно с XHTML тагове.
- Нов контекст за запазване на данни – разработчикът може вече да запазва обекти в нов контекст отнесен към текущата страница и когато потребителят напусне страницата, този контекст се унищожава заедно с обектите в него и заетата памет се изчиства. Другият нов контекст е така нареченият Flash контекст. Неговата цел е да се запазват данни, които да „преживеят” сървърно пренасочване. С помощта на този контекст може да се реализира Post-Redirect-Get алгоритъм.
- Поддръжка на позволяващи запазване (bookmark) на уеб адреси – в предишни версии на технологията е много трудно и изискващо много усилия, за да се промени приложението, така че да поддържа адреси позволяващи запазване. Такива адреси са адреси които изпращат GET заявка към приложението и изпращат параметри, които биват обработвани и на тяхна база се визуализира определено съдържание на потребителя.
- Оптимизиране на паметта и производителността – значително е намалена паметта използвана от UI компонентите, което прави приложенията по-производителни.

3.3. EJB 3.1

Промените във версия 3.1 на Enterprise JavaBean спецификацията (JSR-318) са по-скоро козметични и не променят съществено досегашния начин на тяхната употреба. Промените са:

- Единствена инстанция (Singleton) – даден bean може да се обозначи като единствена инстанция, при което контейнерът ще създаде само един обект от класа на този bean и обектът ще бъде споделен между всички други обекти използващи по

някакъв начин този клас. Този подход спестява системни ресурси, но има основен недостатък и той е: проблеми със синхронизацията на състоянието на конкретния bean. За да не се допускат такива проблеми е целесъобразно в такива beans да не се пази информация, която може да бъде променяна и четена конкурентно.

- Опционално използване на локални и дистанционни интерфейси – проблем в предишни реализации е налагането на задължителен интерфейс, който bean-ът трябва да имплементира. В много случаи тази абстракция е ненужна и на моменти дори представлява дразнеща механична работа, която трябва да се извърши от програмиста само за да се удовлетворят изискванията на програмния интерфейс.
- Поддръжка на асинхронни бизнес методи – предоставя се възможност определени методи да бъдат изпълнени в отделна нишка, като резултатът от изпълнението им е достъпен чрез Java Future програмен интерфейс. Този механизъм може да бъде използван за времеемки задачи, чийто отговор не е необходимо да бъде наличен незабавно след извикване на метода.
- EJB Lite – олекотена версия на Enterprise JavaBeans предоставяща минимален набор от функционалности необходими за уеб приложения с малък набор от изисквания.

3.4. JPA 2

Спецификацията на Java Persistence API версия 2 (JSR-317) внася подобрения от към позлваемостта на технологията, които улесняват в голяма степен разработката. Към подобренията спадат:

- Поддръжка на едностранни асоциации от типа едно към много с вторичен ключ – в предишната версия такива асоциации се поддържат единствено с присъединителна таблица. В много голям процент от случаите двустранната асоциация е излишна а наличието на присъединителна таблица увеличава броя на заявките и намалява производителността на приложенията.
- Java програмен интерфейс за сглобяване на динамични заявки – дава се възможност чрез Java код и без употребата на SQL да се сглобяват динамични заявки за извличане и корекция на данни. Тази функционалност практически елиминира нуждата от дълбоки познания по SQL и значително увеличава читаемостта и поддръжката на кода.
- Изтриване на същности без родител – в предишната версия на спецификацията, когато една същност бъде премахната от даден списък (едно към много или много към много асоциация) тя не бива изтрита от базата докато ръчно не се укаже, че трябва да бъде изтрита. В новата версия това става автоматично, намалявайки вероятността от програмни грешки.

3.5. CDI

Context and Dependency Injection (JSR-299) е най-иновативната технология в Java Enterprise Edition 6. Тя е резултат от дългогодишен опит с технологии способстващи метода „инжектиране на зависимости”. Основната цел на технологията е да създаде свързващ слой между всички технологии на Java Enterprise Edition, защото до момента на създаването такъв стандартен слой липсва, а е от голямо значение. Основните върху които стъпва технологията са:

- Мениджмънт на жизнения цикъл на обекти следващи JavaBeans конвенцията свързвайки ги към ясно дефинирани контексти (обхват на обектите).
- Типизиран достъп до обектите чрез „инжектиране на зависимости” – използват се Java анотации като мета данни, указващи какъв обект трябва да се инжектира. Тези анотации се наричат „определения” (qualifiers).
- Стандартен механизъм за създаване и обработка на събития – поддържа се вграден механизъм за генериране на събития и обработката им, по такъв начин,

че напълно се развързва кода за генерирането от кода на разработката. Дори е възможно код в една Java библиотека да генерира събитие, а код в друга Java библиотека да обработи това събитие, без библиотеките да са компилационно-свързани.

- Редуцирана обвързаност между обектите – това е резултат от „инжектиране на зависимости".
- Стандартна поддръжка на „прихващачи на извикване" (интерцептори).
- Интеграция с унифицирания език за изрази (EL).

ЗАКЛЮЧЕНИЕ

Практическото използване на Java Enterprise Edition 6 и на предходни версии на платформата позволява да се направят следните изводи относно това, което новата версия предлага:

- Значително повишена ефективност при разработка на приложения.
- Увеличена производителност на крайния продукт.
- По-удобен за поддръжка и тестване код.
- Преносимост между приложни сървъри на различни софтуерни доставчици.
- По-ниска вероятност от конфигурационни грешки от страна на програмиста.

ЛИТЕРАТУРА

[1] Java Community Process - http://jcp.org Последно ползване: м. август 2010
[2] Oracle "The Java EE 6 Tutorial" - http://download.oracle.com/javaee/6/tutorial/doc/javaeetutorial6.pdf Последно ползване: м. август 2010
[3] Rahman R., "Java EE 6 Overview", 02.2009 - http://www.theserverside.com/news/1363662/Java-EE-6-Overview Последно ползване: м. август 2010

За контакти:

маг.-инж. Адриан Митев, Сирма Ай Ти Ти, София бул. Цариградско шосе 135, email: adrian.mitev@sirma.bg

доц. д-р Петър Сигалов, Русенски университет "Ангел Кънчев", кат. Информатика и информационни технологии, тел.: (082) 888754, e-mail: sigalov@ami.uni-ruse.bg

Докладът е рецензиран.