JDK 7 Update

Dalibor Topić
ORACLE Deutschland B.V. & Co KG
Hamburg

Keywords:

Java, JVM, JDK 7, OpenJDK, Open Source, Dynamische Programmiersprachen, Project Coin

Einleitung & Agenda

Auf der JavaOne 2010 Konferenz in San Francisco im September wurde der aktuelle Status zur Planung von JDK 7 vorgestellt. Die Entwicklung der Java Plattform geht mit starkem Engagement der Java Community in großen Schritten voran in einem erfolgreichen Open Source Projekt im Rahmen der OpenJDK Community. Neben einer Übersicht der geplanten Features, gibt dieses Paper auch einen Überblick über die Geschichte der Entwicklung von JDK 7, die Partizipation der Java Community, und zeigt Wege auf, wie man die Entwicklung aktuell und zukünftig geplanter Features verfolgen und, bei Interesse, auch an ihr direkt teilnehmen kann.

Wie alles mal klein anfing

Am 13. November 2006 wurden die ersten Teile des JDK von Sun Microsystems im Rahmen der OpenJDK unter der GNU General Public License v2 freigegeben - einem in der Open Source Community bereits bestehenden und weit akzeptierten Lizenzmodell. Das waren am Anfang der Compiler javac, der in der Programmiersprache Java geschr iebene Quelltext in den Java Bytecode übersetzt, und die HotSpot JVM, die diesen Bytecode ausführen kann.

Die Freigabe der Klassenbibliotheken erforderte ein wenig mehr Zeit. Sie wurden von Sun Microsystems zur JavaOne 2007 im Rahmen der OpenJDK Community freigegeben. Mit diesem Schritt war die Freigabe der Quelltexte des JDK abgeschlossen, und die Arbeit am JDK 7 konnte im Rahmen des entsprechenen JDK 7 Projects (http://openjdk.java.net/projects/jdk7) in der OpenJDK Community beginnen.

OpenJDK Community

Die OpenJDK Community, zu finden unter http://openjdk.java.net , ist auf den ersten Blick etwas ungewöhnlich. Sie ist der Ort an dem Oracle und die Java & Open Source Community an der Entwicklung der Open Source Implementierung der Java SE Plattform arbeiten, und damit verwandter Projekte. Neben dem Hauptprojekt JDK 7 beinhaltet sie auch kleinere, auf bestimmte Aspekte zugeschnittene Projekte wie Project Coin (http://openjdk.java.net/projects/coin), die sich mit der Entwicklung einzelner für JDK 7 geplanter oder noch experimenteller Features befassen, sowie weitere Projekte, die beispielsweise Portierungen des JDK 7 auf andere Betriebssysteme oder CPU Architekturen sind.

Ein Teil der Entwicklungsarbeit an geplanten Features findet in Unterprojekten wie dem eingangs erwähnten Project Coin statt, das sich mit kleinen Sprachänderungen zur Steigerung der Produktivität von Java Entwicklern befasst. Wenn die Implementierung einzelner geplanter Features in den für ihre

Entwicklung zuständigen Unterprojekten ausgereift ist, werden sie in das JDK 7 Projekt integriert. Die verteilte Art der Entwicklung über Unterprojekte wird auf der technischen Seite durch die Nutzung des verteilten Versionskontrollsystems Mercurial unterstützt, sowie auf der organisatorischen Seite durch einen strikten Entwicklungsprozess, der Reviews an verschiedenen Punkten der Entwicklung vorsieht.

Die geplanten Features

Die für JDK 7 geplanten Features lassen sich grob in vier Kategorien unterteilen: VM, Programmiersprache, Core Libraries, Client. Die Core Libraries umfassen dabei die Kernbibliotheken der Plattform, von I/O, über Internationalisierung bis zur XML Unterstützung. Unter Client sind die geplanten Verbesserungen im Bereich der GUI-Bibliotheken zu finden. Eine ausführliche Liste der Features findet sich auf der JDK 7 Projektseite (http://openjdk.java.net/projects/jdk7/features), mit weiterführenden Verweisen zu mehr Informationen zu den einzelnen Features, den Entwicklern, die für die Implementierung federführend zuständig sind, und auch zu den einzelnen Unterprojekten, in denen die Features entwickelt werden.

Die geplanten VM-Features

Im VM Bereich ist nur ein Feature geplant, aber das hat es in sich. Die Implementierung des JSR 292, und den dort spezifizierten Invokedynamic Bytecode im vom John Rose geführtem Multi-language VM Project geht einher mit einer verbesserten Unterstützung für dynamische Programmiersprachen wie Ruby, Python oder PHP auf der JVM. Der neue Bytecode ergänzt die in der JVM bereits vorhandenen Invokevirtual, Invokestatic, Invokespecial und Invokeinterface Bytecodes, die den Aufruf von Methoden in Java Klassen auf der JVM Ebene steuern.

Die bestehenden Bytecodes orientieren sich in ihrem Verhalten an der Programmiersprache Java, und sind daher nicht unbedingt ohne Umwege zur Implementierung anderer Programmiersprachen geeignet, die sich ihrerseits von Java mehr oder weniger radikal unterscheiden. Solche Unterschiede treten oft hervor in den Regeln wie Methoden in Klassen aufgerufen werden können, z.B. darin wie Namen von Methoden beim Methodenaufruf auf konkrete Implementierungen abgebildet, und wie Konflikte dabei gelöst werden. Der Invokedynamic Bytecode und die dazugehörige Infrastruktur im Paket java.dyn wie CallSites und MethodHandles stellen ein mächtiges Werkzeug dar, welches dem Entwickler von dynamischen Programmiersprachen für die JVM aus dem Blickwinkel der Performanz wesentlich teuerere Umwege über Reflection oder Bytecode-Generierung erspart, und ihm die Möglichkeit bietet für einzelnen Aufrufe des Invokedynamic Bytecodes die Bindung der Methodennamen an Implementierungen selbst in die Hand zu nehmen, und sie gegebenenfalls auch dynamisch wieder zu ändern. Damit kann das Potenzial der JVM als leistungsfähige Laufzeitumgebung für andere Programmiersprachen besser und einfacher ausgeschöpft werden.

Die geplanten Sprachfeatures

Für JDK 7 sind das erste mal seit Java SE 5.0 Verbesserungen an der Programmiersprache selbst geplant. Im Project Coin sammelte 2008 Joe Darcy einen Monat lang die Ideen aus der Java Entwickler-Community für Sprach-Features. Die Vorschläge sollten sich auf kleine Verbesserungen konzentrieren. Aus den eingesandten und auf der Projektmailingliste coin-dev rege diskutierten Vorschlägen wurden von Joe Darcy und seinen Mitstreitern etwa ein Dutzend zur konkreten Implementierung innerhalb des Project Coin ausgewählt. Davon sind für JDK 7 geplant: Verwendung von Strings in switch-Anweisungen („strings in switch“), automatisches Ressourcenmanagement für try-Anweisungen („try-with-resources“), verbesserte Typinferenz für new-Ausdrücke mit generischen

Typen („diamond“), vereinfachter Aufruf von Methoden mit Varargs, verbesserte Unterstützung für Zahlenkonstanten und Unterstützung für die Verarbeitung von mehreren Exceptions in einem einzelnen Catch-Block („multi-catch“).

Die geplanten Features der Klassenbibliotheken

In den Kernbibliotheken wurde die ClassLoader API und Implementierung angepasst, um Deadlocks in komplexen Nutzungsfällen mit Nebenläufigkeit zu vermeiden. Die URLClassLoader Klasse bekam eine close() Methode spendiert, um die von ihr verwendeten Ressourcen explizit freizugeben. Und schliesslich wurde ein Update des JSR 166 von Doug Lea integriert. Dieses Update bringt unter anderem ein Fork/Join Framework mit, wodurch die Entwicklung von Software nach dem „Teile-und-Herrsche“ Prinzip vereinfacht wird.

Im Bereich der Internationalisierung sind ein Update auf Unicode 6.0 geplant, sowie ein Upgrade der Locale API und Implementierung um neue Standards in dem Bereich zu unterstützen.

Bei der Unterstützung für Netzwerkprotokolle sind im JDK 7 zwei wesentliche Neuerungen geplant. Einerseits die Unterstützung für Sockets Direct Protocol unter sowohl Solaris als auch unter Linux, sowie für das Stream Control Transmission Protocol unter Solaris. Unter weiteren Netzwerkfeatures finden sich die geplante Unterstützung für TLS 1.2 und die Verwendung von nativen IPv6 Stack unter Windows Vista, sofern vorhanden.

Im I/O Bereich der Klassenbibliotheken kommt dazu noch die Implementierung des von Alan Bateman geführten JSR 203, der sich der Erweiterung und Verbesserung der NIO APIs verschrieben hat. Dort finden sich neue APIs für den einfacheren Umgang mit Dateien, Dateipfaden, Dateiattributen auf verschiedenen Dateisystemen, Verzeichnisstrukturen, virtuelle Dateisysteme und Dateisysteme im Allgemeinen, sowie Verbesserungen im Bereich des asynchronen I/O und diverser Netzwerkoperationen. Als ein Anwendungsfall der Unterstützung für eigene, virtuelle Dateisysteme ist eine Sicht auf ZIP und JAR Dateien als Dateisystem als JDK 7 Feature geplant.

Mit jedem Release der Plattform geht auch ein Update der in ihr enthaltenen Web und JDBC APIs einher. Die Implementierung und die APIs fpr JAXB, JAXP und JAX-WS sollen auf den aktuellen Stand gebracht werden. Es ist weiterhin für JDK 7 geplant, JDBC 4.1 zu unterstützen.

Die geplanten Client-Features

Es ist für JDK 7 geplant die GUI-Bibliotheken der Plattform ebenfalls in einigen Bereichen zu verbessern. Ein Feature, welches aus der OpenJDK Community hervorgegangen ist, ist die Implementierung einer Java2D Pipeline, die auf der XRender Erweiterung von X 11 basiert, und damit die Grafikwiedergabe unter Linux und Solaris mit modernen, von XRender unterstützen GPUs beschleunigt. Des weiteren kommen im Client Bereich noch das Nimbus Look and Feel & JLayer dazu, sowie öffentliche APIs für in Java SE 6u10 dazugekommene Funktionalität wie z.B. durchscheinende und verformte Fenster.

Nach JDK 7

In OpenJDK Projekten finden sich auch Features, die für ein Release der Plattform nach JDK 7, d.h. für JDK 8 oder ein Release danach geplant sind. Das sind unter anderem die Unterstützung für Closures, die im Project Lambda (http://openjdk.java.net/projects/lambda) vorangetrieben wird, sowie

die Modularisierung der Plattform mit den dazugehörigen Werkzeugen, die im Project Jigsaw (http://openjdk.java.net/projects/jigsaw) angegangen wird.

Höhere Transparenz bei der Entwicklungsarbeit

Bei der Vielzahl von geplanten Features an denen für JDK 7 gearbeitet wurde und noch gearbeitet wird, kann es für den einzelnen Java Entwickler, der nicht die Arbeit an verschiedenen OpenJDK Projekten minutiös verfolgt, durchaus eine kleine Herausforderung sein, sich das richtige Mittel der Wahl auszusuchen, mit dem man auf dem Laufenden bleibt. Für Java Entwickler, die sich für grundlegende Aspekte der JDK Entwicklungsarbeit interessieren, bieten sich jährlich stattfindende Konferenzen wie JavaOne, Devoxx oder die DOAG Konferenz an, um eine Übersicht über den aktuellen Status der Entwicklung zu bekommen. Ergänzend dazu bietet der OpenJDK News Blog (http://blogs.sun.com/openjdk) einen regelmäßigen Nachrichtenüberblick aus dem JDK 7 Projekt, und den populärsten Unterprojekten der OpenJDK Community.

Für Entwickler mit einem tieferen Interesse an einzelnen Features sind die Oracle TechCast Live Videopodcasts zu empfehlen – in Live-Interviews mit einer anschliessenden Q&A Runde erläutern dort führende Köpfe der Java Entwicklung, wie Mark Reinhold, Chief Architect der Java Platform Group bei Oracle, oder Alex Buckley, der bei Oracle für die Spezifikationen der Programmiersprache Java und der Java Virtual Machine zuständig ist, was es mit JDK 7 auf sich hat, oder wie sehr sich die JVM als universelle Laufzeitumgebung für beliebige Programmiersprachen eignet. Zum letzteren Thema finden sich im Videoarchiv des JVM Language Summit (http://www.jvmlangsummit.com/) Aufnahmen der verschiedenen Vorträge zu tiefgreifenden JVM Themen von Oracle Mitarbeitern und anderen OpenJDK Community Mitgliedern.

Für noch mehr Einsicht sorgen die Blogs der federführenden Entwickler. Die Blogs der Entwickler aus der OpenJDK Community werden durch Planet JDK (http://planetjdk.org) zusammengetragen und zentral verfügbar gemacht.

Schließlich bleiben noch die Mailinglisten der einzelnen (Unter)projekte, die auf den jeweiligen Projektseiten verzeichnet sind, und die von ihnen verfügbar gemachten Quellen, die im Mercurial Repository der OpenJDK Community (http://hg.openjdk.java.net) zu finden sind.

Feedback erwünscht

Eine einfache Möglichkeit Feedback zur laufenden Entwicklung zu nehmen, ist es einen aktuellen Preview Build des JDK 7 mit eigenem Code auszuprobieren. Für den jeweils aktuellen wöchentlichen Build des JDK 7 sind auf dem Downloadserver (http://download.java.net/jdk7/binaries) Binaries für Windows, Solaris und Linux verfügbar. Sollten beim Ausprobieren Fehler auftreten, so sollte diese an die BugParade (http://bugs.sun.com) gemeldet werden. Das gleiche gilt für Features, die man im JDK vermisst. Bei sich mitten in der Entwicklung befindlichen Features sollten Kommentare zum aktuellen Stand direkt auf die Mailinglisten der dafür zuständigen OpenJDK Projekte fliessen.

Wer mehr als Feedback zur Implementierung von Features beitragen möchte, kann das über den OpenJDK Contribution Process (http://openjdk.java.net/contribute) tun. Mehrere Dutzend Mitglieder der OpenJDK Community, die nicht bei Oracle angestellt sind, tragen durch ihre Mitwirkung an der Entwicklung von OpenJDK Projekten mit, die beste und beliebteste Open Source Implementierung der Java SE Plattform in verschiedenen Aspekten weiter zu verbessern. Informationen zu einzelnen von den OpenJDK Community Mitgliedern geführten Projekten, wie z.B. der Portierung für BSD Betriebssyteme finden sich auf den jeweiligen Projektseiten.

Kontaktadresse:

Dalibor Topić
ORACLE Deutschland B.V. & Co KG
Nagelsweg 55
20097 Hamburg

Phone: +49(0)40-23646738
Email dalibor.topic@oracle.com
Internet: http://twitter.com/robilad

Oracle is committed to developing practices and products that help protect the environment

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. UNIX is a registered trademark licensed through X/Open Company, Ltd. 0410

SOFTWARE. HARDWARE. COMPLETE.