Enterprise

'Never a dull moment' in de wereld van Java. Op die regel zal 2009 ook zeker geen uitzondering zijn. Het jaar van de waarheid voor JavaFx, de vermoedelijke afronding van Java 7 (Dolphin), de doorbraak van OSGi in enterprise infrastructuur, verdere omarming binnen Java en de Java-gemeenschap van dynamische talen, nog rijkere gebruikerinterfaces, opbloei van Cloud Computing en de volwassenwording van SOA standaarden als SCA en SDO. Eén van de ontwikkelingen die al enige tijd zijn schaduw vooruit werpt en in de loop van 2009 tot voorlopig hoogtepunt komt is JEE 6, de nieuwste release van de Java Enterprise Edition (JEE).

Het jaar van JEE 6

Aftrap voor een avontuurlijke verkenning

Lucas Jellema,
AMIS
(Lucas.jellema@amis.nl)

Misschien is dit niet eens het hipste of meest aansprekende onderwerp voor de gemiddelde Java-ontwikkelaar, maar voor de positie van het Java-platform in het hart van organisaties van groot belang. Oorspronkelijk was de release van JEE 6 voorzien voor het najaar van 2008; nu wordt gemikt op mei 2009 - vlak voor JavaOne 2009.
JEE is een etiket dat wordt geplakt op een verzameling specificaties (JSRs) die door het Java Community Process zijn vastgesteld. In dat proces zijn individuele ontwikkelaars betrokken, naast vertegenwoordigers van open source communities en commerciële leveranciers. Het proces dat de specificaties doorlopen is openbaar en voor iedereen via internet te volgen. Via drafts en proposals, inspraakrondes en stemrondes wordt de uiteindelijke specificatie vastgesteld. Bij een specificatie hoort overigens altijd een Referentie Implementatie - werkende code die de specificatie implementeert en demonstreert.
Voor de JEE 5 en 6 is Glassfish, de open source applicatie server, de referentie-implementatie (zie ook https://glassfish.dev.java.net).

Op basis van de specificaties gaan zowel open source gemeenschappen als commerciële leveranciers aan de slag om hun producten aan te passen. Het binnenhalen van het JEE certificaat - het etiket dat zekerheid geeft aan klanten dat volgens de specificaties gebouwde applicaties succesvol in productie kunnen worden genomen - is bijna de heilige graal voor leveranciers van applicatieservers, in elk geval uit oogpunt van marketing. De komende periode zullen we een race zien tussen bijvoorbeeld Oracle, IBM en RedHat/JBoss om het JEE6 gehalte van hun applicatieserver op peil te brengen.
Over de jaren en releases van J(2)EE zien we de aanpak in het tot stand brengen van de specificaties wat verschuiven. In het verleden werden de specificaties sterk gestuurd door ervaren Java-rotten vanuit theorie en laboratorium, uitmondend in correcte maar soms zeer complexe resultaten. EJB 2.x wordt veelal gezien als summum van die complexiteit en stond symbool voor de dreigende ondergang van J2EE (en de opkomst van het Spring Framework).

Ommekeer

JEE 5 zorgde in 2006 voor een ommekeer. De specificaties werden op een veel pragmatischer manier ingericht, werden de vriend van de ontwikkelaar in plaats van haar vijand. Het simpele adagium 'configuration by exception' - alleen configureren wat afwijkt van de default én zorgen voor simpele logische defaults - maakte alleen al een wereld van verschil. De inzet van Java 5 Annotaties maakt de nog resterende configuratie nog eenvoudiger. De insteek voor JEE 5 was ook sterk pragmatisch; ervaring en best practices met allerlei frameworks werd nadrukkelijk ingebracht bij de ontwikkeling van de nieuwe specificaties. Een goed voorbeeld is de betrokkenheid van Craig McClanahan- de vader van Struts - bij de ontwikkeling van JavaServer Faces (JSF), maar ook de inbreng van de teams achter Hibernate en TopLink bij de totstandkoming van Java Persistence API

(JPA). JEE 5 introduceerde met JSF en vooral JPA en EJB 3.0 nieuwe specificaties met heel concrete toegevoegde waarde.

Basis

Het originele Java Specification Request (JSR-316) dat de basis vormt voor het Java Community proces waarin de JEE 6 specificatie wordt ontwikkeld beschrijft twee hoofddoelstellingen voor de nieuwe release van het platform: *extensibility* en *profiles*. Daarnaast zijn verbetering van bestaande specificaties, reduceren van complexiteit en het aansluiten bij SOA infrastructuren met ondermeer verdere ondersteuning voor WebServices belangrijke doelen. Aangezien JEE 6 de enterprise editie is van versie 6 van het Java platform is het gebaseerd op Java SE 6 - en gebouwd op JEE 5.

Extensibility en profiles zijn de hoofddoelen voor de nieuwe release

Extensibility

Het JSR comité constateert dat er om het kern-JEE-platform heen allerlei complementaire technologieën, toepassingen en frameworks zijn ontstaan. Met extensibility wordt nagestreefd deze aanvullingen zo goed mogelijk in te passen in het Java EE landschap. Soms door formele specificaties voor deze technologieën te ontwikkelen en die in het platform op te nemen - JAX-RS, WebBeans - maar steeds vaker door aanknopingspunten te bieden: interfaces waar uitbreidingen op of specifieke implementaties van platform onderdelen op ingeplugd kunnen worden.

Profiles

Een veel besproken onderdeel van JEE 6 is het idee van *profiles*. Dit idee komt voort uit de constatering dat het JEEE platform te groot, complex is geworden en voor de meeste situaties te weinig specifiek focus heeft. Een profiel is een formeel gedefinieerde subset van specificaties uit JEE 6 - mogelijk aangevuld met componenten van buiten JEE 6. Een container kan nu JEE gecertificeerd worden voor een bepaald profiel, een soort JEE 6 Light Container. Op dit moment is er één profiel gedefinieerd binnen JEE 6: het Web Profile. Dit profiel beschrijft de verzameling van specificaties die nodig zijn voor het ontwikkelen van JEE 6-stijl Web Applicaties en zou kunnen worden gebruikt om containers als Tomcat ook van een JEE stempel te voorzien. De samenstelling van dit profiel is overigens nog onderwerp van discussie: moeten JSF en WebBeans daar bijvoorbeeld ook in?
De gedachte achter Profiles is dat er eenvoudiger toegang komt tot de JEE wereld - voor zowel leveranciers als ontwikkelaars -via goed samenhangende, formeel gecertificeerde lichtgewicht deelverzamelingen bestemd voor specifieke toepassingen. De *portability* tussen verschillende containers die voor hetzelfde profiel zijn gecertificeerd zou door de certificering gewaarborgd moeten zijn.

Snoeien

Een andere manier om de omvang en de complexiteit van het JEE platform te beperken is 'snoeien' (pruning): het uitfaseren van specificaties die min of meer ballast zijn geworden, bijvoorbeeld omdat ze achterhaald zijn door nieuwe specificaties. Dat laatste geldt in de ogen van het specificatie comité bijvoorbeeld voor JAX-R(PC) - ingehaald door JAX-WS - en EJB CMP (Container Managed Persistence) - feitelijk vervangen door JPA. Het proces van uitfaseren van een specificatie is uiteraard formeel en zorgvuldig maar kan op brede steun rekenen. Meer kandidaten voor 'snoeien' zullen de komende periode voorgedragen worden.

Oogsten

Van het snoeien naar het oogsten. De volgende generatie van JEE is ook wel een 'oogst' release genoemd: van de nieuw in JEE 5 geïntroduceerde specificaties komen belangrijke nieuwe versies, naar aanleiding van de praktijkervaring in de afgelopen jaren van zowel de ontwikkelaars als de leveranciers. Het gaat ondermeer om: JPA 2.0, EJB 3.1 en JSF 2.0. Daarnaast wordt het JEE

platform uitgebreid met nieuwe specificaties die sterk geïnspireerd zijn op bestaande, bewezen frameworks als JBoss Seam en Google Guice (WebBeans) of verschillende implementaties voor RESTful WebService (JAX-RS).

Een gouwe ouwe binnen JEE is de Servlet specificatie, die in JEE 6 met de 3.0 versie flink nieuw leven ingeblazen krijgen. De JavaServer Pages (JSP) specificatie wordt overigens niet geupgrade. Een ander "ouwetje" dat wel vernieuwing zal ondergaan is JCA (Java EE Connector Architecture).

Hoogtepunten in JEE 6

Heel in het kort volgt hier een overzicht van de meest opvallende elementen van enkele van de JEE 6 specificaties:

Web Beans (JSR-299): het doel is om EJB 3.0 componenten als JSF managed beans beschikbaar te stellen en de twee modellen - EJB 3.0 en JSF - te combineren. Daarnaast biedt WebBeans een fijnmazig model voor *state management* en ondersteuning voor werkeenheden (conversations) die meerdere web transacties omvatten. Web Beans is sterk geïnspireerd op JBoss Seam.

Java Persistence API (JPA) 2.0 (JSR-317): meer flexibiliteit in de O/R mapping, ondersteuning voor collecties van (geneste) embedded objects, gesorteerde en geaggregeerde lijsten en extra metadata om de generatie van DDL scripts - om database objects te creëren - te sturen. Uitbreiding van de query faciliteiten van JPA met een Criteria API, gebruik van hints voor het configureren van query's en entity managers, ondersteuning voor pessimistic locking en voor cache APIs.

Enterprise Java Beans (EJB) 3.1 (JSR-318): verdere vereenvoudiging van het EJB model, uitbreiding van de Timer Service, mogelijkheid om zonder lokale business interface een EJB component - dus met alleen een bean Class - te gebruiken. Daarnaast kunnen EJB componenten in een WAR gedeployed worden, zonder dat een ejb-jar nodig is. Een zeer interessante optie is de asynchrone aanroep van session bean methodes, met of zonder return type.
De optie om 'embeddable' EJB containers te draaien buiten de context van échte, volwaardige JEE servers - zoals voor ondermeer unittesten al wordt gedaan met bijvoorbeeld OpenEJB en Embedded JBoss - wordt gestandaardiseerd. Daarmee wordt het lokaal toepassen van EJBs, met alle faciliteiten als gedistribueerde transacties, messaging, scheduling en asynchrone aanroepen mogelijk in desktop of andere Standard Edition applicaties.

Servlet 3.0 (JSR-315): Servlets gaan een tweede jeugd doormaken op basis van hun nieuwe specificatie. De ondersteuning van asynchrone afhandeling van requests geeft ons een schaalbare basis voor 'server push' architecturen die met AJAX gebaseerde achtergrondcommunicatie en DHTML partiële pagina verversing zorgt voor actieve, dynamische Web User Interfaces met geïntegreerde voorzieningen voor communicatie en event registratie.

JavaServer Faces (JSF) 2.0 (JSR-314): JSF bouwt op het fundament dat de Servlet API definieert en kan dus van de asynchrone ondersteuning en de web.xml integratie faciliteiten profiteren. Daarnaast is een belangrijk doel voor deze release het verder vereenvoudigen van het gebruik van JSF, ondermeer door eenvoudiger configuratie en het gebruik van annotaties
AJAX wordt een integraal onderdeel van de life cycle afhandeling, Facelets worden ondersteund als View implementatie naast JSPs, ontwikkeling van nieuwe customcomponenten wordt aanzienlijk vereenvoudigd en het beheer van resources zoals JavaScript wordt gestroomlijnd. Tenslotte integreert JSF met diverse andere specificaties waarvan JAX-RS (REST), WebBeans en Bean Validation de meest opvallende zijn.

Volgende stappen richting JEE 6

Een voordehandliggende manier om thuis te raken in JEE 6 en de onderliggende specificaties is uiteraard door ze te lezen. Het valt mij iedere keer weer mee hoe leesbaar de meeste JSR-documenten eigenlijk zijn. Vaak is er aan geschreven door auteurs van boeken over Java en JEE technologie en lezen ze heel aardig weg - bijna als een boek.
Daarnaast is er de afgelopen anderhalf jaar al het nodige geblogd en zeker ook gediscussieerd over JEE 6. Ook een prima manier om gevoel te krijgen voor de belangrijkste JEE 6 onderdelen en meest pikante discussie-onderwerpen.
De eerste implementaties van de verschillende JEE 6 specificaties zijn al geruime tijd in ontwikkeling, zeker de Referentie Implementaties die als onderdeel van de specificatie moeten worden opgeleverd. Je kunt met een aantal van deze implementaties aan de slag om vast een eerste gevoel te krijgen voor wat er op ons afkomt.

In de volgende nummers van Java Magazine zullen we in meer gedetailleerd ingaan op de belangrijkste onderdelen van JEE 6, uiteraard met voorbeelden van praktische toepassingen en code die je zelf ook kan gaan uitvoeren. «

Als je al aan de slag wilt met eerste implementaties van de JEE 6 specificaties - bijvoorbeeld Servlet 3.0, EJB 3.1 en JSF 2.0 - dan zou je naar Glassfish kunnen kijken. Op https://glassfish.dev.java.net/downloads/v3-prelude.html kan je Glassfish V3 Prelude downloaden, een voorloper op de volledige JEE 6 ondersteuning in Glassfish. Apache MyFaces is bezig met de ondersteuning van JSF 2.0; een rechtstreekse code check out kan worden gedaan uit de Subversion Repository (https://svn.apache.org/repos/asf/myfaces/core/branches/2_0_0/). Vingeroefeningen met JPA 2.0 zou je kunnen gaan doen met een nightly build van EclipseLink, de referentie implementatie voor deze specificatie, te vinden op: http://www.eclipse.org/eclipselink/downloads/nightly.php.