DOAG
Deutsche ORACLE-Anwendergruppe e.V.

Java EE 6

.msg systems

.consulting .solutions .partnership

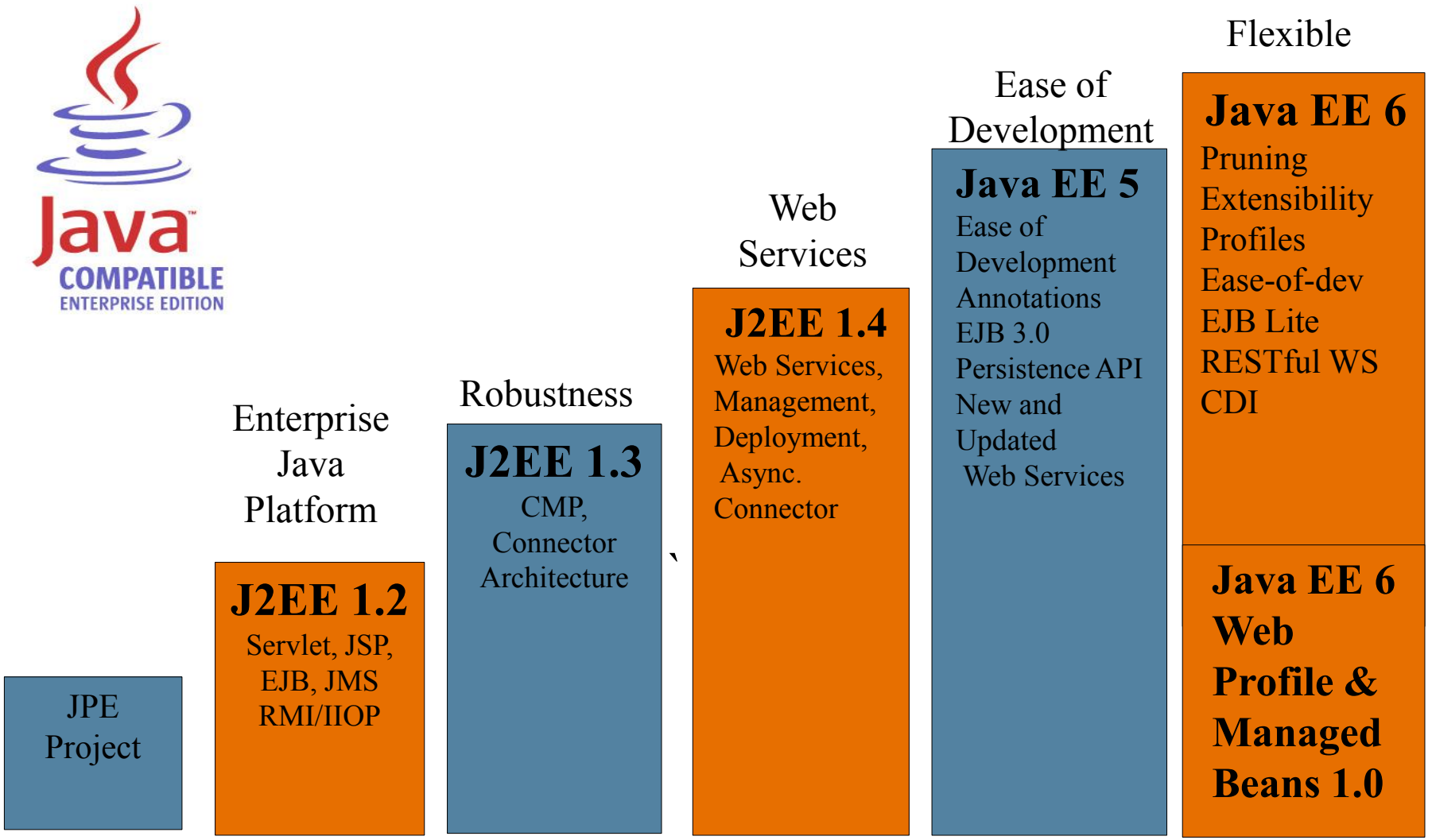
..ein Überblick.

Markus Eisele

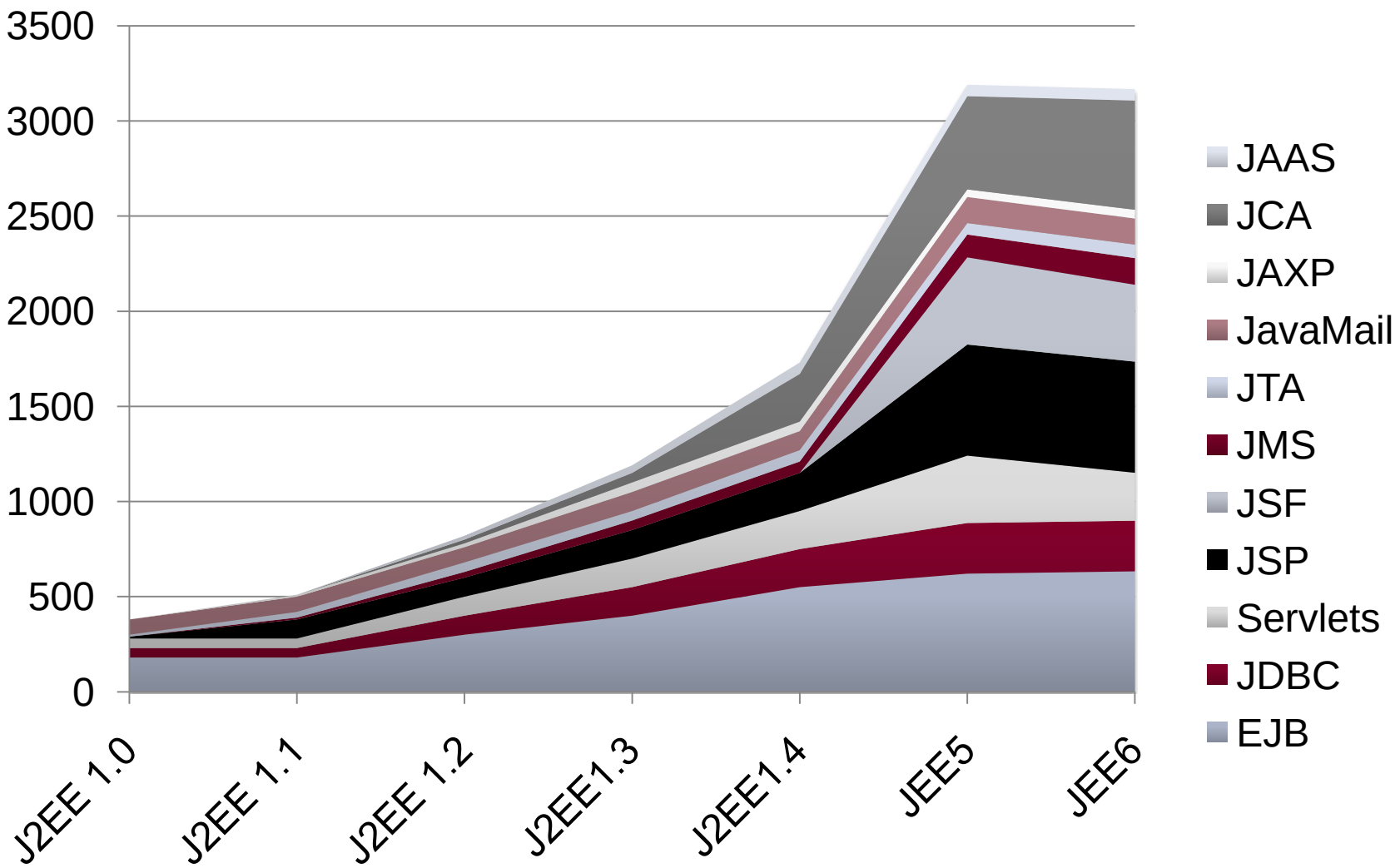
blog.eisele.net
@myfear

http://blog.eisele.net
http://twitter.com/myfear
markus.eisele@msg-systems.com

Java EE: Past and Present



Numbers for complexity?



Compatible Java EE 5 Implementations

http://java.sun.com/javaee/overview/compatibility-javaee5.jsp

Compatible Java EE 6 Implementations

Today

Coming

Goals for the Java EE 6 Platform

- Flexible & Light-weight
- Extensible
  - Embrace Open Source Frameworks
- Easier to use, develop on
  - Continue on path set by Java EE 5

Java EE 6 is Flexible

- Decouple specs to allow more combinations
- Expands potential licensee ecosystem
- Profiles
  - Targeted bundle of technologies
  - Defined through the JCP
  - Web Profile Defined
    - Defined by the Java EE 6 Expert Group

- Fully functional mid-sized profile
  - Actively discussed in the Java EE 6 Expert Group and outside it
  - Technologies
    - Servlets 3.0, JSP 2.2, EL 2.2, Debugging Support for Other Languages 1.0, JSTL 1.2, JSF 2.0, Common Annotations 1.1, EJB 3.1 Lite, JTA 1.1, JPA 2.0, Bean Validation 1.0, Managed Beans 1.0, Interceptors 1.1, Context & Dependency Injection 1.0, Dependency Injection for Java 1.0

Java EE 6 is Lightweight

- Pruning
  - Make some technologies optional
- Pruned today, means
  - Optional in the next release
  - Deleted in the subsequent releases
- Technologies marked in Javadocs
  - JAX-RPC, EJB 2.x Entity Beans, JAXR, JSR 88

Java EE 6 is Final

- Specifications approved by the JCP
- Reference Implementation is GlassFish v3
- TCK

Java EE 6 Specifications

- The Platform
- Java EE 6 Web Profile 1.0
- Managed Beans 1.0

REVIEW Java-Entwicklung

Java EE 6 nimmt Masse und Geschwindigkeit auf

Warp 6

Markus Eisele

Wegen Lizenzstreitigkeiten mit Sun war Java EE 6, im April 2007 als Java Specification Request 313 gestartet, zunächst fulminat gescheitert. Unter dem JSR 316 formierte sich die zuständige Expert Group neu. Nun liegen die Ergebnisse vor.

Seit der Verabschiedung von Java EE 5 dauerte es zweieinhalb Jahre, bis sich die Initiatoren des „Umbrella JSR" auf Art und Umfang von Java EE 6 einigen konnten. Beteiligt waren Sun, Oracle, IBM, SAP, Red Hat, SpringSource und andere. Die über 200 Seiten starke Spezifikation zum Java Specification Request 316 spannt einen Schirm über mehr als 30 eigenständige JSRs und koordiniert deren Zusammenarbeit. Neben den Verweisen auf die enthaltenen Techniken enthält er – wie die Vorgängerversionen – eine Beschreibung der Entwicklungs-, Deployment- und Lebenszyklen. Zum Leistungsumfang gehören eine Kompatibilitäts-Testsuite (CTS), die Referenzimplementierung Glass-Fish v3, sogenannte Profiles sowie eine aktuelle Ausgabe der Java-EE-Blueprints. Das Ganze präsentiert sich als Gemisch aus Text und Implementierungen.

Die mit JEE 5 betretenen Pfade hat die Expert Group konsequent ausgebaut. Das Entwicklungsmodell folgt spezifikationsübergreifend immer deutlicher dem Metadaten- beziehungsweise Annotationskonzept. Ergänzt wird es durch die entsprechenden Container (Web, EJB et cetera) mit ihren jeweiligen Standardverhalten. Die ungeliebten XML-Deployment-Beschreibungen erledigen sich damit in den meisten Fällen. Am Beispiel der Servlets lässt sich das veranschaulichen: Bei JEE 5 und früheren Versionen musste der Entwickler das gewünschte Verhalten im Webapplikations-Deployment-Deskriptor *web.xml* festlegen, nun kann er dazu ein JEE 6-konformes Servlet mit Annotationen konfigurieren:

```
@WebServlet("/ix")
public class NewsTickerServlet extends HttpServlet
{...}
```

Wenn bestimmte Anforderungen an die Ausführungsumgebung des Servlets nicht explizit im Code beziehungsweise in der Annotation formuliert sind, übernimmt der Container und führt Aufrufe an das Servlet entsprechend der hinterlegten Voreinstellungen aus. Ist etwa nur das URL-Pattern (wie im obigen Beispiel) angegeben, verwendet der Container standardmäßig den vollqualifizierten Klassennamen als Servlet-Namen. Der Entwickler kann immer noch über den nun optionalen Deployment-Deskriptor eingreifen, viele der beteiligten Spezifikationen haben diesen Wandel erfolgreich vollzogen.

Vom Komplizierten zum Einfachen

Abseits dieser Vereinfachungen enthält JEE 6 einige größere Versions- und damit Funktionssprünge. Die Enterprise-JavaBeans-Spezifikation (EJB) liegt nun in der Version 3.1 vor. Technischer Höhepunkt ist der *no-interface view*, der das Umsetzen von EJBs als Plain Old Java Objects (POJOs) ermöglicht. Bisher benötigte eine lokale Session Bean immer ein lokales Business Interface. Session Beans ohne Interface gelten jetzt automatisch als lokal, deren Methoden der Client vollständig aufrufen kann. Singleton Beans unterstützt die Spezifikation ebenfalls. Dabei handelt es sich um EJBs, die nur eine Instanz besitzen. Bisher ließen sich solche Konstrukte ausschließlich über herstellerspezifische Funktionen abbilden. Eine Singleton Bean bekommt als Annotation *@Singleton* zugewiesen. Der EJB-Container erzeugt und verwaltet auch diese Beans.

Angelehnt an die Concurrency API aus Java SE 5 gibt es nun asynchrone Methodenaufrufe. Bisher war das ein Fall für den vergleichsweise schwergewichtigen Java Messaging Service (JMS). Auch hier wird durch das Hinzufügen einer Annotation (*@Asynchronous*) eine Bean-Methode zum asynchronen Ausführen eingestellt.

Gänzlich neu ist die EJB 3.1 Lite, ein Subset, das EJB-Funktionen für das

66 iX 1/2010

http://www.heise.de/kiosk/archiv/ix/2010/1/66

Java EE 6 Specifications - NEW

- Context and Dependency Injection for Java EE (JSR 299)
- Bean Validation 1.0 (JSR 303)
- Java API for RESTful Web Services (JSR 311)
- Dependency Injection for Java (JSR 330)

- Java Server Faces 2.0 (JSR 314)
- Java Servlets 3.0 (JSR 315)
- Java Persistence 2.0 (JSR 317)
- Enterprise Java Beans 3.1 & Interceptors 1.1 (JSR 318)
- Java EE Connector Architecture 1.6 (JSR 322)

Java EE 6 Specifications - Updates

- Java API for XML-based Web Services 2.2 (JSR 224)
- Java API for XML Binding 2.2 (JSR 222)
- Web Services Metadata MR3 (JSR 181)
- JSP 2.2/EL 2.2 (JSR 245)
- Web Services for Java EE 1.3 (JSR 109)
- Common Annotations 1.1 (JSR 250)
- Java Authorization Contract for Containers 1.3 (JSR 115)
- Java Authentication Service Provider Interface for Containers 1.0 (JSR 196)

Java EE 6 Specifications – As is

- JDBC 4.0 API
- Java Naming and Directory Interface 1.2
- Java Message Service 1.1
- Java Transaction API 1.1
- Java Transaction Service 1.0
- JavaMail API Specification 1.4
- JavaBeans Activation Framework 1.1
- Java API for XML Processing 1.3
- Java API for XML-based RPC 1.1
- SOAP with Attachments API for Java 1.3
- Java API for XML Registries 1.0
- Java EE Management Specification 1.1 (JSR 77)
- Java EE Deployment Specification 1.2 (JSR 88)
- Java Management Extensions 1.2
- Java Authentication and Authorization Service 1.0
- Debugging Support for Other Languages (JSR 45)
- Standard Tag Library for JSP 1.2 (JSR 52)
- Streaming API for XML 1.0 (JSR 173)

- Continue advancements of Java EE 5
- Primary focus: Web Tier
- General principles
  - Annotation-based programming model
  - Reduce or eliminate need for DD
  - Traditional API for advanced users

Servlets in Java EE 5 – Two Files


```
<!--Deployment descriptor
   web.xml -->
<web-app>
   <servlet>
     <servlet-name>MyServlet
             </servlet-name>
       <servlet-class>
         com.sun.MyServlet
       </servlet-class>
   </servlet>
   <servlet-mapping>
     <servlet-name>MyServlet
       </servlet-name>
     <url-pattern>/myApp/*
       </url-pattern>
   </servlet-mapping>
    ...
</web-app>
```

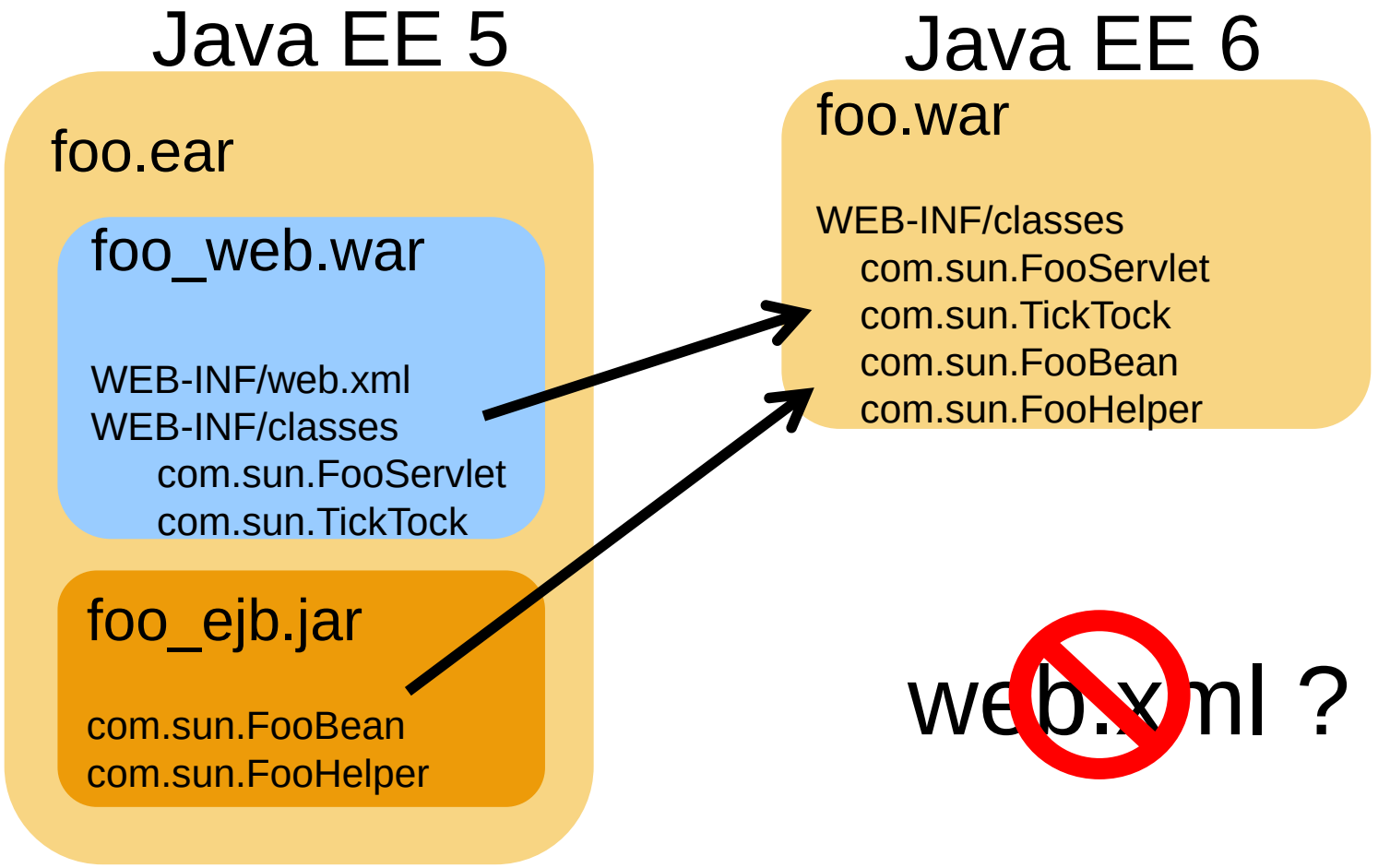
```
/* Code in Java Class */

package com.sun;
public class MyServlet extends
HttpServlet {
public                          void
doGet(HttpServletRequest
req,HttpServletResponse res)
{
…
}
…
}
```

Servlets in Java EE 6 – Annotation driven

```
package com.sun;
@WebServlet(name="MyServlet", urlPattern="/myurl/*")
public class MyServlet extends HttpServlet {
   public void doGet(HttpServletRequest req,
                     HttpServletResponse res)
   {
      ...
   }
```

- Annotations to declare Servlets, Filters, Init param, ...
  > “web.xml” is optional in most of the cases
- Plugin libraries using web fragments
  - modular web.xml
- Async support
  - @WebServlet(asyncSupported=true)
- Programmatic authentication and logout
- Default error page, File upload support
- Using new language features – e.g. Generics

EJB 3.1 – Simplified Packaging



- @Singleton beans – shared state per VM
- No interface view – one source file per bean
- Calendar timers – cron like semantics
  - @Schedule(dayOfWeek=”Mon,Wed”)
  - (hour=”14”, dayOfMonth=”Last Thu”, month=”Nov”)
  - (minute=”*/5”, hour=”*”)
- Application startup/shutdown callbacks
- EJB “Lite” - Small subset for Web profile

EJB 3.1 Simple Bean


```
@Stateless
public class App {
      public String sayHello(String name) {
            return "Hello " + name;
      }
}
```

EJB 3.1 Lite

A proper subset of the full EJB 3.1 API that includes a small, powerful selection of EJB features suitable for writing portable transactional business logic

. . .

suitable for inclusion in a wider range of Java products, many of which have much smaller installation and runtime footprints than a typical full Java EE implementation

TABLE 2. EJB LITE AND EJB 3.1 FEATURES COMPARISON

FEATURE	EJB LITE	EJB 3.1
Local session beans	X	X
Declarative security	X	X
Transactions (CMT/BMT)	X	X
Interceptors	X	X
Security	X	X
Message-driven beans		X
RMI/IIOP interoperability and remote view		X
EJB 2.x backward compatibility		X
Time service		X
Asynchronous method invocations		X

- Facelets as “templating language” for the page
  - Custom components much easier to develop
- Ajax support integrated
  - f:ajax
- “faces-config.xml” optional in common cases
- Mojarra is the Reference Implementation
-

Bean Validation – JSR-303

- Tier-independent mechanism to define constraints for data validation
  - Represented by annotations
  - javax.validation.* package
- Integrated with JSF and JPA
  - JSF: f:validateRequired, f:validateRegexp
  - JPA: pre-persist, pre-update, and pre-remove
- @NotNull(message=”...”), @Max, @Min, @Size
- Fully Extensible
  - @Email String recipient;

Managed Beans 1.0

- JavaBeans component model for Java EE
  - Simple and Universally useful
  - Advanced concepts in companion specs
- Basic Services
  - Resource Injection
  - Lifecycle Callbacks
  - Interceptors
- Available as
  - @Resource / @Inject
  - java:app/<module-name>/<bean-name>
  - java:module/<bean-name>

Contexts & Dependency Injection (JSR 299)

- Type-safe Dependency Injection
- Builds on @Inject API
- Context/Scope management
- Works with multiple bean types
- Includes ELResolver

Java-Entwicklung

Dependency Injection in Java EE 6

Direkteinspritzer

Markus Eisele

Die Java Specification Requests 330 und 299 haben es kurz vor Toreschluss in die Java-EE-6-Spezifikation geschafft. Gemeinsam bilden sie die Grundlage für die typsichere Injektion von Abhängigkeiten. Die nicht unumstrittenen JSRs haben den Anspruch, die besten Funktionen aus etablierten Frameworks wie Seam, Guice und Spring zu vereinen und als Standard zu etablieren.

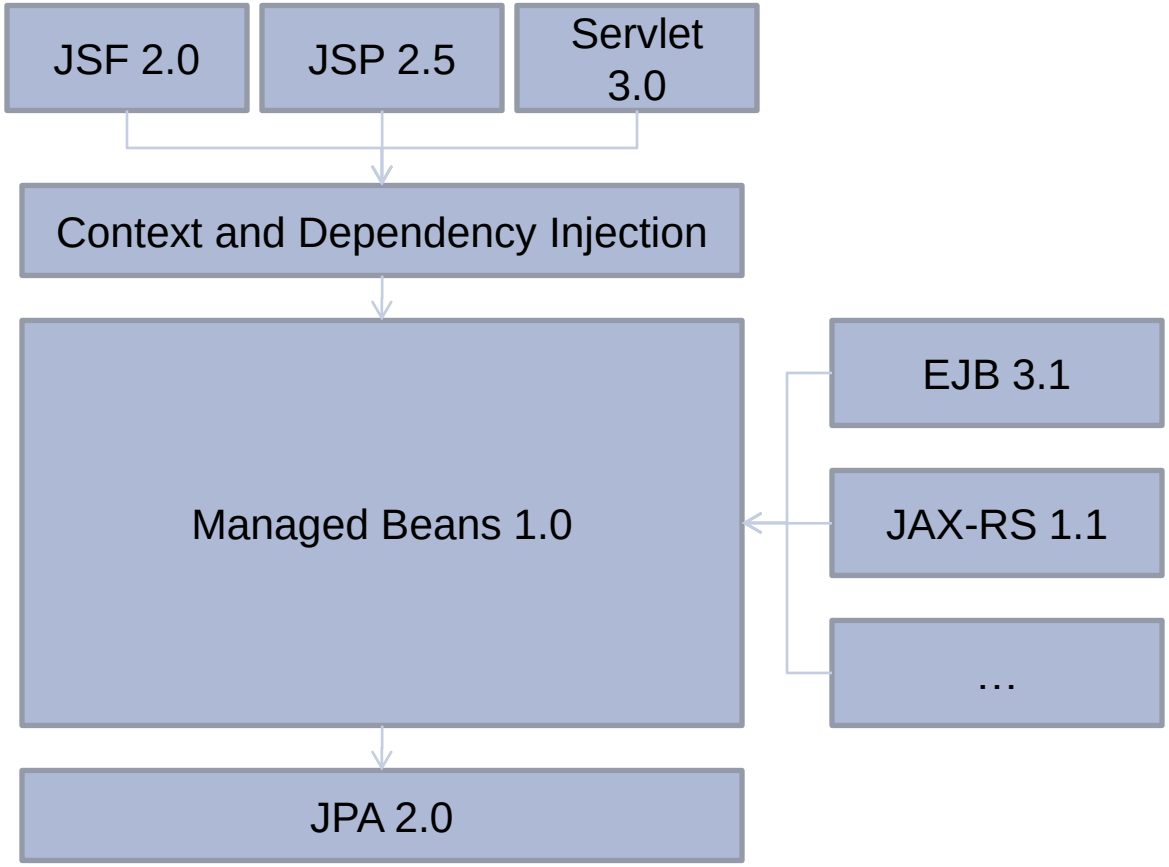
Angst vor dem Komplexen

134

IX 4/2010

http://www.heise.de/kiosk/archiv/ix/2010/4/134

CDI and Managed Beans

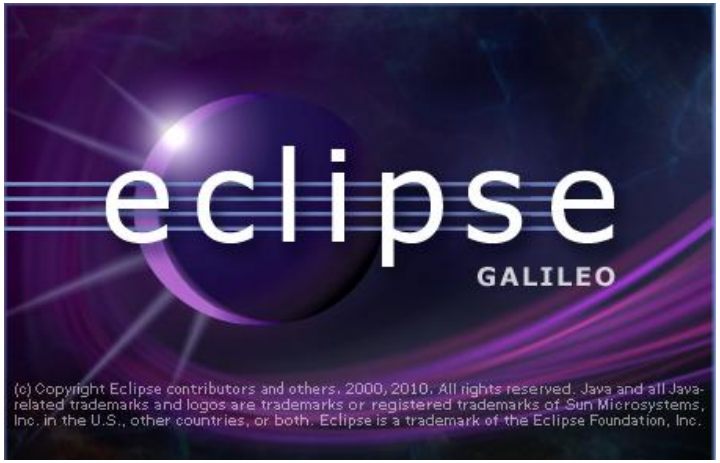


Java Persistence API 2.0 (JSR 317)

- OneToMany / @ManyToOne
- @ElementCollection
- @OrderColumn
- @MapKeyJoinColumn
- @Embeddable
- @Access
- Combined Primary Keys
- Type-safe Criteria API
- Metamodel API
- …

http://www.slideshare.net/myfear/new-features-of-jsr-317-jpa-20

(first) IDE Support for Java EE 6



Further Reading Java EE 6

- http://www.oracle.com/technetwork/java/javaee/overview/index.html
- http://download.oracle.com/javaee/6/tutorial/doc/
- https://www.packtpub.com/java-ee-6-applications-with-glassfish-3-application-server/book
- http://www.apress.com/book/view/9781430219569
- http://apress.com/book/view/9781590596715

References

- glassfish.org
- blogs.sun.com/theaquarium
- oracle.com/goto/glassfish
- Twitter: @glassfish

Vielen Dank für Ihre Aufmerksamkeit !

.msg systems

.consulting .solutions .partnership