Gleich geht’s los…

Moderne Architekturen mit EJB 3.1 und CDI

TONI BARTL

AKTUELLE TECHNOLOGIEN ZUR ENTWICKLUNG VERTEILTER JAVA-ANWENDUNGEN

IB6A, SOMMERSEMESTER 2013, 05522309

Agenda

- Einleitung
- Session Beans
- Context and Dependency Injection
- Asynchrone Methodenaufrufe
- Timer Service
- Fazit

Einführung

- Enterprise Java Beans -> Anfang (Version 1.0) -> 1998 (JPE)
  - standardisierte Komponenten -> weniger Aufwand, weniger infrastruktureller Code
  - EJB's sind, vom AS bzw. vom Container (EJB-Container) verwaltete Objekte.
  - Der AS/Container bietet Dienste und Funktionalitäten sowie bestimmte Protokolle die wir unter Einhaltung bestimmter Programmierparadigmen nutzen können. z.B.:
    Ressourcenverwaltung
    Lastverteilung
    Security Management
    Persistenzmanagement
    Automatische Transaktionsverwaltung
    Monitoring, etc...
- Es gibt 3 Bohnen Arten

Session Beans

Messag[illegible]n Beans

Persis[illegible]ntities

Session Beans

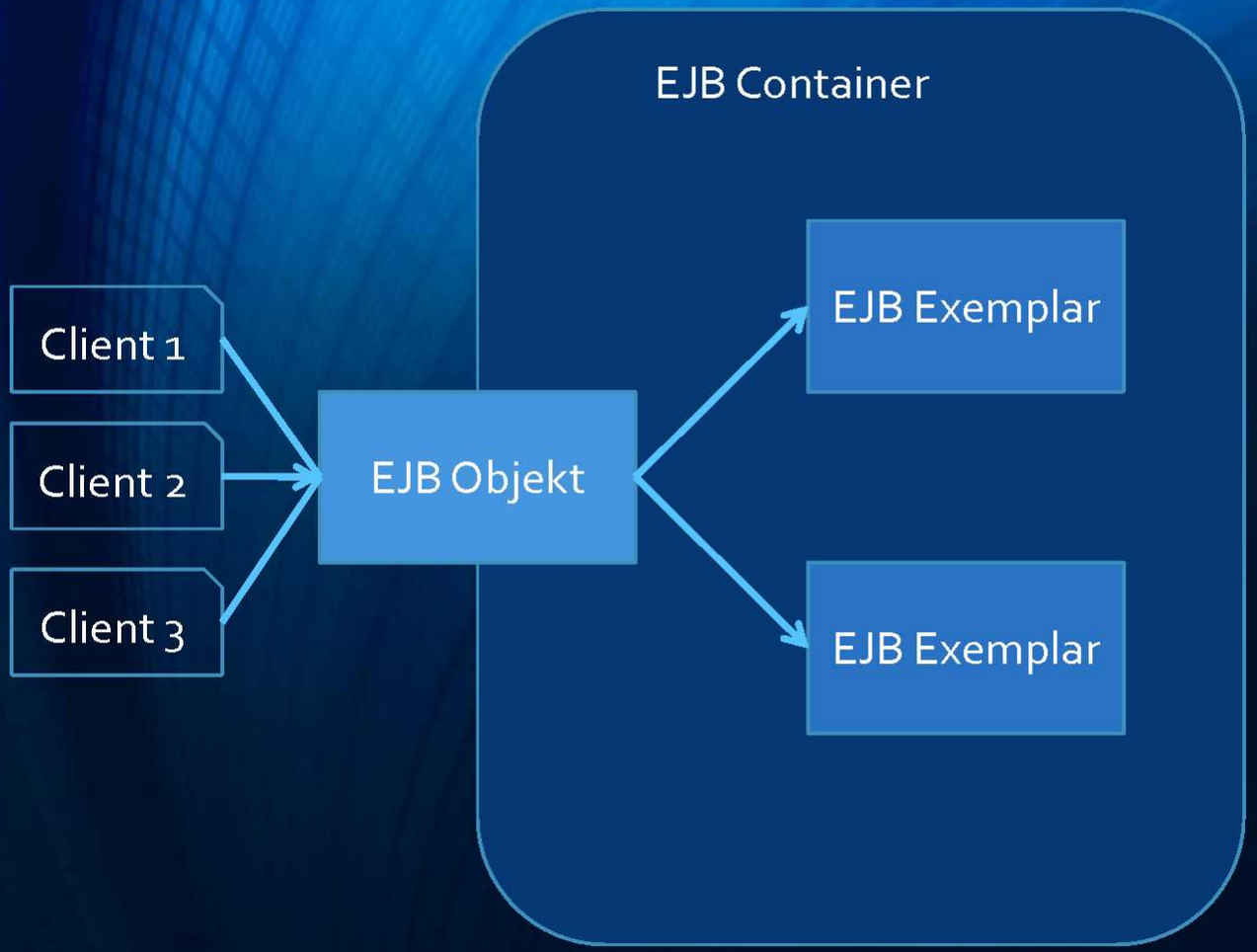
- Klarer Bestandteil der Geschäftslogik.
- Die Logik kann beliebig komplex sein.
- Session Beans sind nicht persistent.

  Liegen in drei Ausführungen vor:

  - Stateless
  - Stateful
  - Singleton
- Der Unterschied liegt in der Beziehung zum Client.

Stateless Session Bean

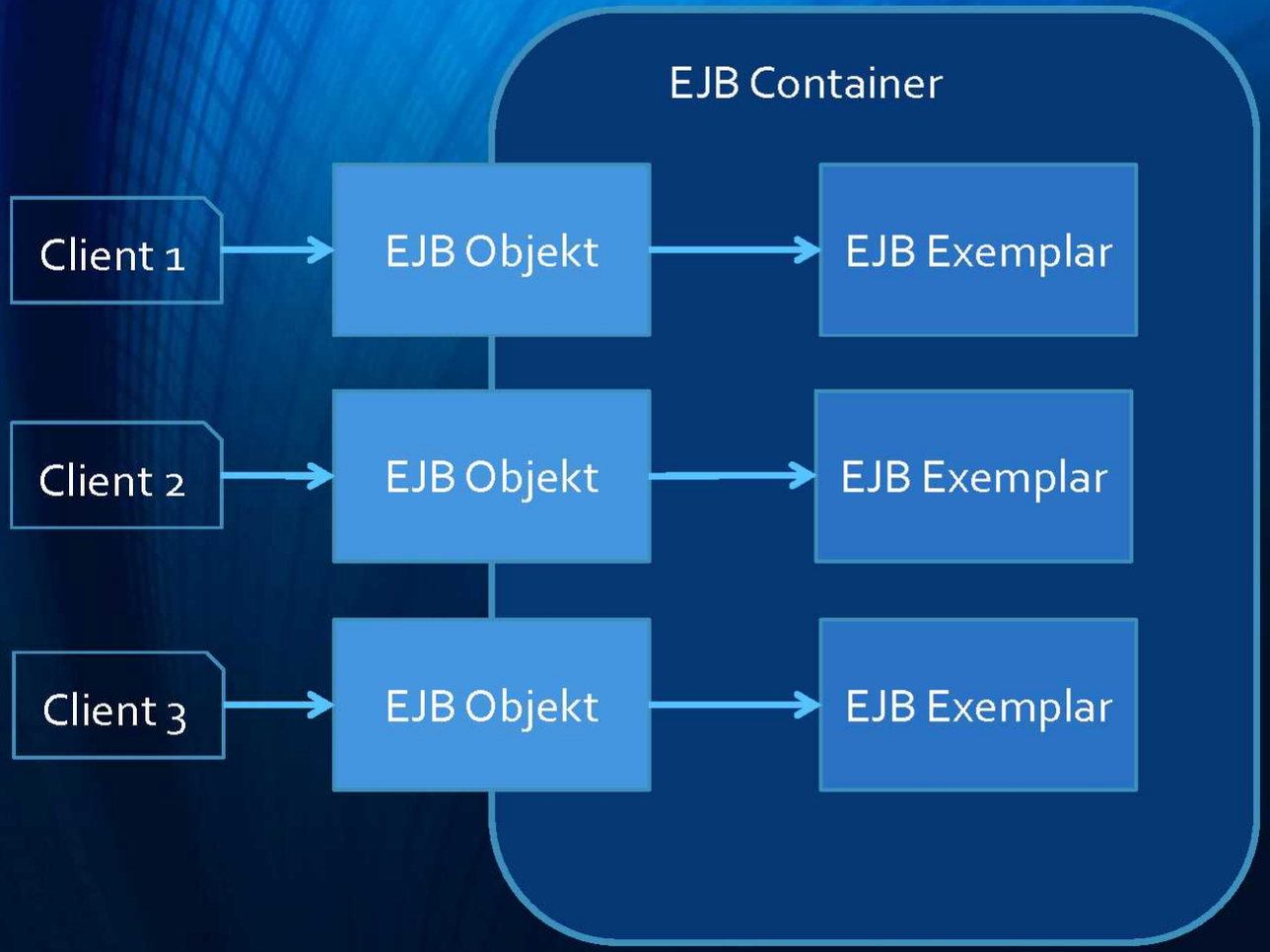
- Werden häufig als zustandslose Beans bezeichnet.
- Speichern keine Informationen über den Client.
- Verbindung für die Dauer eines Methodenaufrufs
- Instanz Pooling
  - Beans werden in einem Pool vorgehalten.
  - Anfrage des Clients immer an freie Instanz.
  - Nach Bearbeitung, wird die Bean wieder in den Pool entlassen.
- Dadurch lassen sich viele Clientanfragen ressourcenschonend bedienen.
- @Stateless



Quelle: In Anlehnung an (tversu)

Statelful Session Bean

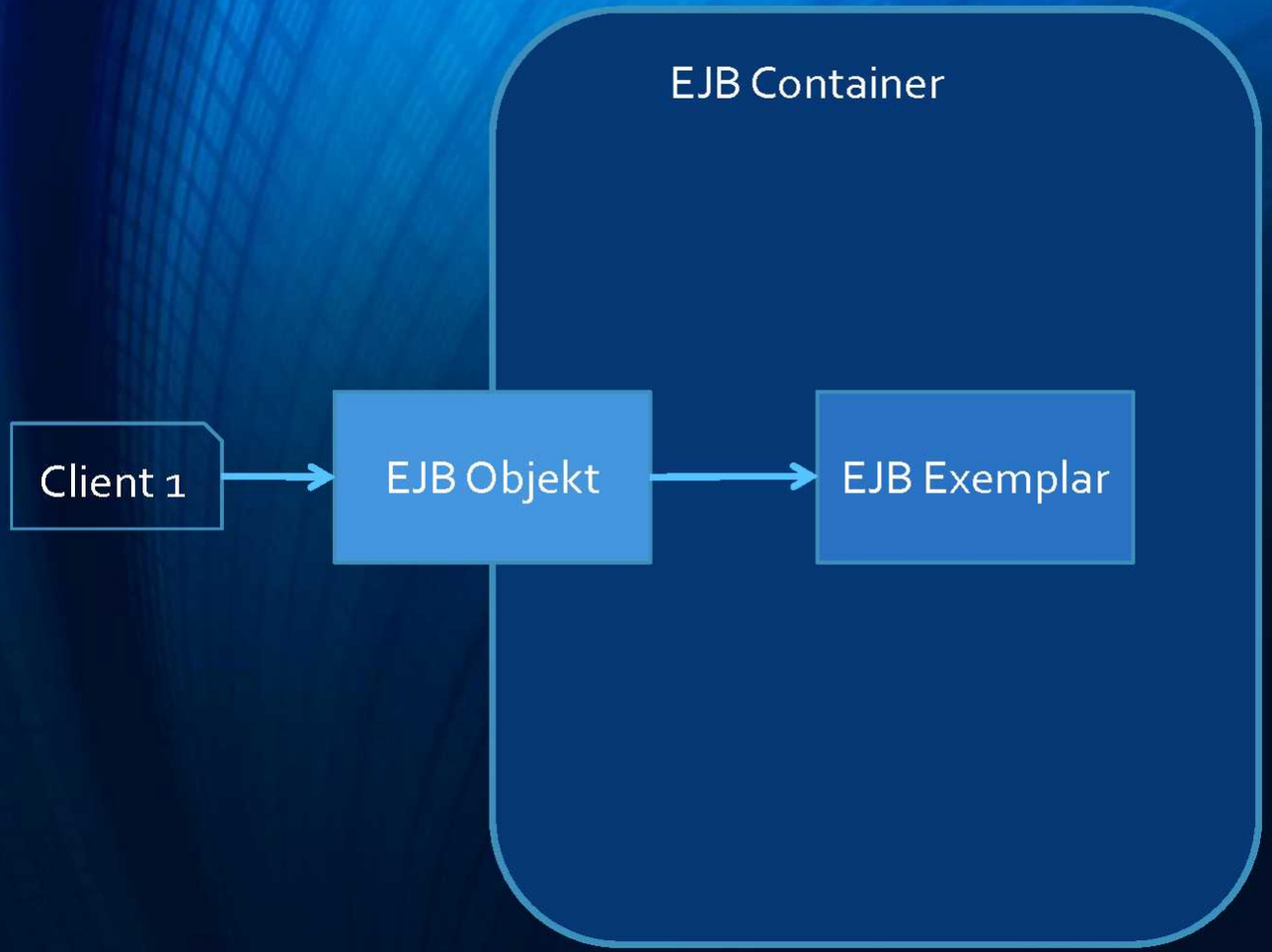
- Speichert Clientspezifischen Zustand.
- Jedem Client ist einem Stateful SB zugeordnet.
- Der Client bestimmt hierbei die Sitzungsdauer
- Sitzung endet mit:
  - @Remove annotierten Klasse
  - @Timeout definierten Zeitpunkt
  - System Exception
- Stateful SB haben ein Conversational State
- Aktivierung <-> Passivierung
- @Stateful, (@Serializable)



Quelle: In Anlehnung an (tversu)

Singleton Session Bean

- Singleton eigentlich ein Entwurfsmuster.
  - Von einer Klasse höchstens eine Instanz.
  - Proprietäre Lösungen oder eigen Impl.
  - Verlust von der vom Container bereitgestellten Funktionalität.
- Verwendung:
  - Applikationsweiter Zustand
  - Speichern von Konstanten / Caches
- Verbindung zum Client wie Stateless SB. Dauer eines Methodenaufrufs.
- Lässt sich direkt beim Starten der Applikation instanziieren. @Startup
- @Singleton



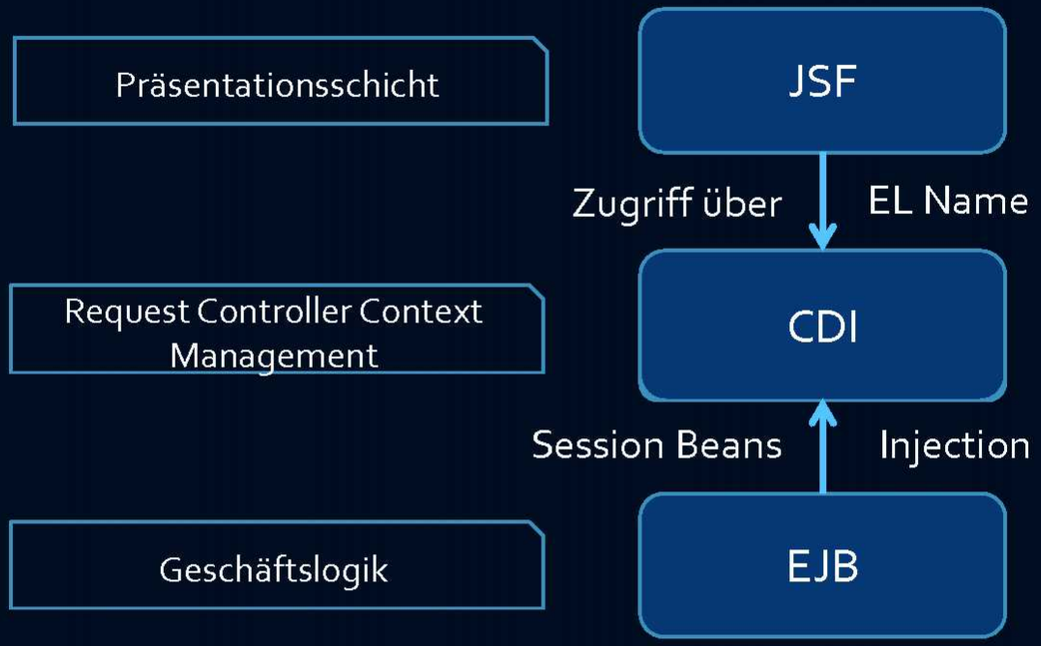
Quelle: In Anlehnung an (tversu)

Context and Dependency Injection

- Java EE Anwendungen sind nicht monolithisch aufgebaut.
- Komponenten die sich gegenseitig nutzen.

Verwaltete Komponenten ⟷ POJO's

new Operator ger Kopplung



Container -> Mach du mal !

Quelle: (Oliver Ihns)

Context and Dependency Injection

- Die Zentrale Komponente von CDI ist der Container.
- Dependency Injection ermöglich es dem Container, Objekte zu erzeugen und zu zerstören.
- Der Kontext beschreibt die Beziehung der Objekte untereinander die im Container verwaltet werden.

- CDI bietet mehr als andere Dependency Injection Frameworks.
- typsichere Auflösung, frühe Information wenn ein Injektionspunkt nicht aufgelöst werden kann.
- grenzübergreifend zwischen Frontend und Backend.

Scopes (= Sichtbarkeit und Lebensdauer)

Request Scope	Dauer eines HTTP Requests.
Conversation Scope	Selbst festgelegte Gesprächsdauer.
Session Scope	Benutzer zugeordneter HTTP Session.
Application Scope	Solange die (Web)-Applikation läuft.

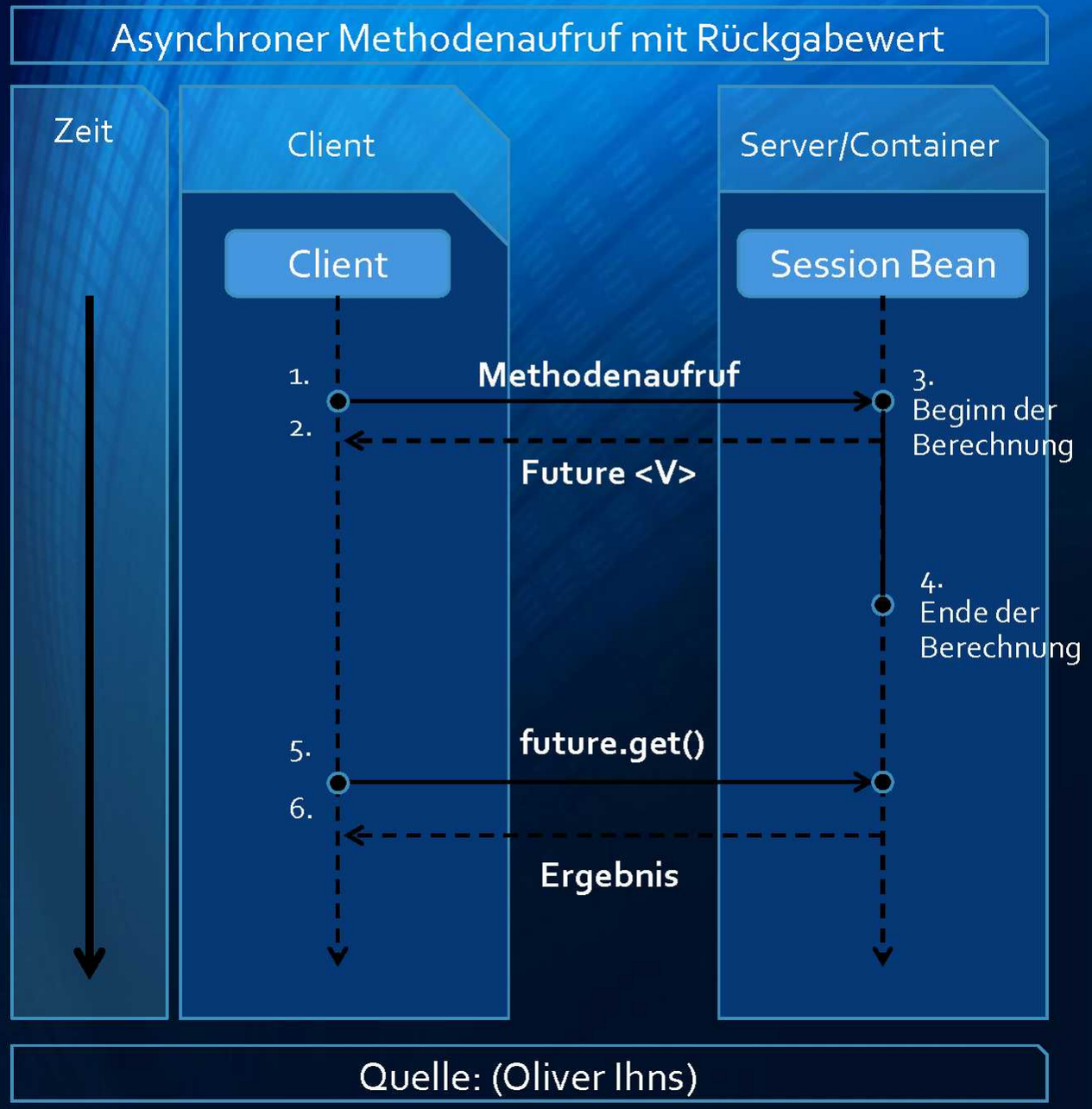
Context and Dependency Injection

- Bean Typen:
  - Mithilfe von BeanTypen kann man Klassen einschränken die an einem bestimmten Injektionspunkt injiziert werden können.
  - @Typed
- Qualifier:
  - Werden benötigt wenn es zu einem verwendeten Interface mehrere Implementierungen gibt.
  - Selbst definierte Annotationen. Annotiert wird die entsprechende Klasse und der Injektionspunkt .
  - @Qualifier
- Alternativen:
  - Abhängig von der Laufzeitumgebung lassen verschiedene Implementierungen eines Interfaces nutzen.
  - Per Default ist diese Funktion Deaktiviert. CDI-Deployment-Desktiptor (beans.xml).
  - @Alternative

Praktisches Beispiel
CDI

Asynchrone Methodenaufrufe

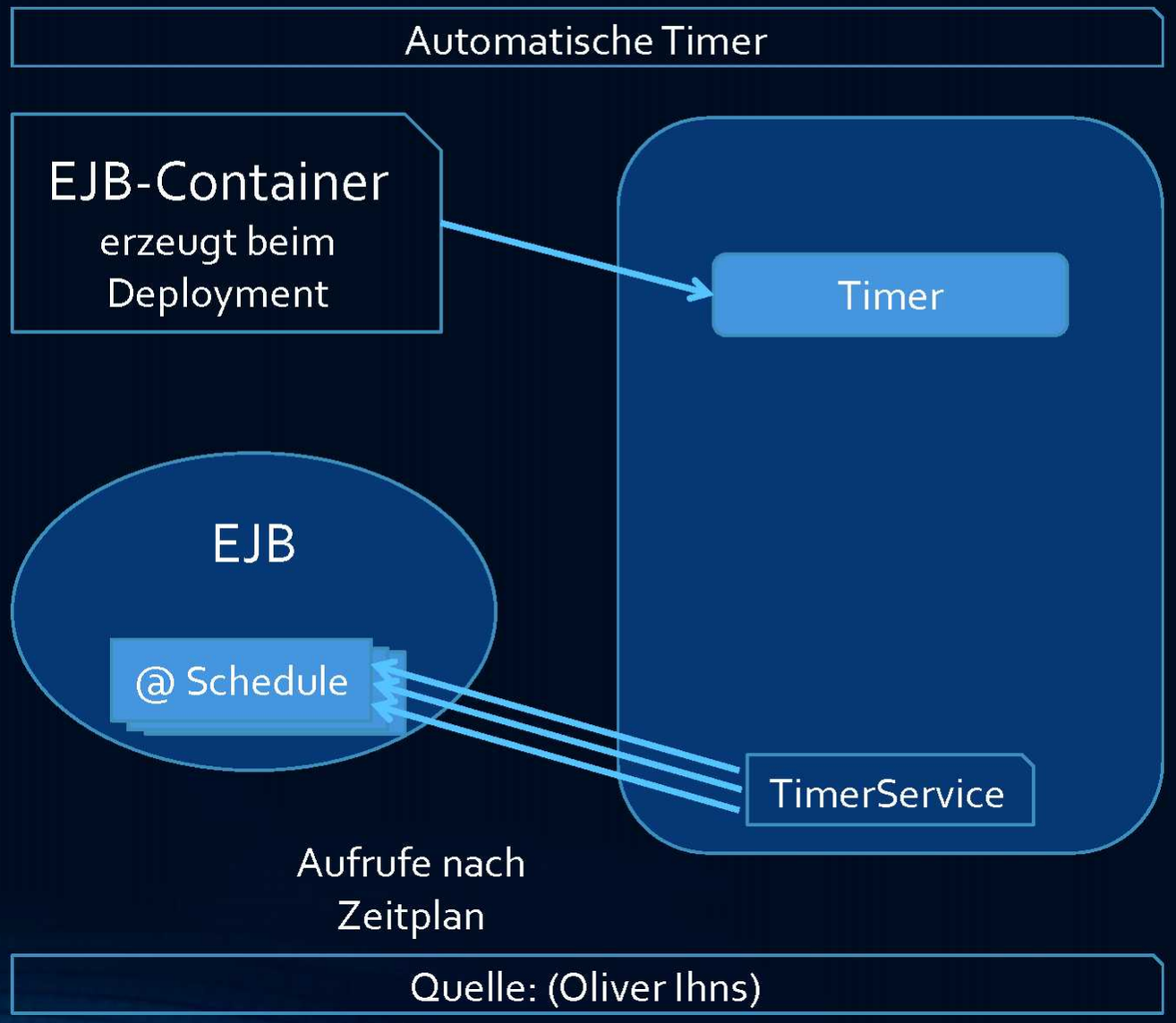
- Anstoßen von Hintergrundprozessen
- Ausführen komplexerer Aufgaben.
- Zwei Arten von Asynchronen Methodenaufrufen:
  - Void
  - Future <V>
- Innerhalb einer EJB
  - @Asynchronus
- Wichtige Methoden:
    - get()
    - isdone()
    - cancel()
    - getCause()



PRAKTISCHES BEISPIEL ASYNCHRONE METHODENAUFRUFE

Timer Service

- Regelmäßige, wiederkehrende Ausführung von Geschäftsprozessen. (Reports, Archivierungsproz.)
- programmgesteuerte und automatische Timer
  - weder static noch final
  - keinen Rückgabewert -> „void“
  - @Schedule (Ausführungszeitplan)
  - Parameter vom Typ Timer übergeben.



Praktisches Beispiel Timer Service

Fazit

- Mit Java EE 6 bzw. EJB 3.1 und CDI ist endlich die Leichtgewichtigkeit die schon lange angestrebt war.
- Einfachheit/Leichtgewichtigkeit kommt der Anwendung und dem Entwickler zu Gute.
- Oracle bemüht sich sehr den Anforderungen der Community gerecht zu werden.
- Es lassen sich mit Java EE 6 und EJB 3.1 sehr schnell und einfach komplexe Unternehmensapplikationen programmieren.
- Jedenfalls schneller als bisher. ☺
- Für ein gekonntes Einsetzen von Java EE 6 benötigt man einiges an Einarbeitungszeit.
- Für Entwickler die im Java Umfeld tätig, lohnt sich eine Einarbeitung definitiv.

Vielen Dank für Ihre Aufmerksamkeit!

Literaturverzeichnis

- Oliver Ihns, Stefan M.Heldt, Holger Koschek, Joachim Ehm, Casten Sahling, Roman Schlömmer. EJB 3.1 proffessional - Grundlagen- und Expertenwissen zu Enterprise JavaBeans 3.1. Heidelberg: dpunkt.verlag, 2011.

- Oliver Ihns, Stefan M.Heldt, Holger Koschek, Joachim Ehm, Casten Sahling, Roman Schlömmer. EJB 3.1 proffessional - Grundlagen- und Expertenwissen zu Enterprise JavaBeans 3.1. Heidelberg: dpunkt.verlag, 2011.

- Werner Eberling, Jan Lessner. Enterprise Java Beans 3 - Das EJB 3 Praxisbuch. München: Hanser, 2007.

- http://edc.tversu.ru/elib/inf/0052/0201914662_ch04lev1sec2.html, aufgerufen am 19.05.2013

- http://jcp.org/aboutJava/communityprocess/final/jsr318/, aufgerufen am 06.05.2013

Fragen ?