exensio
intelligente informationssysteme

Kann ich mit Grails Enterprise Applikationen umsetzen?

DOAG Nürnberg

21.11.2013

Tobias Kraft, exensio GmbH

Agenda

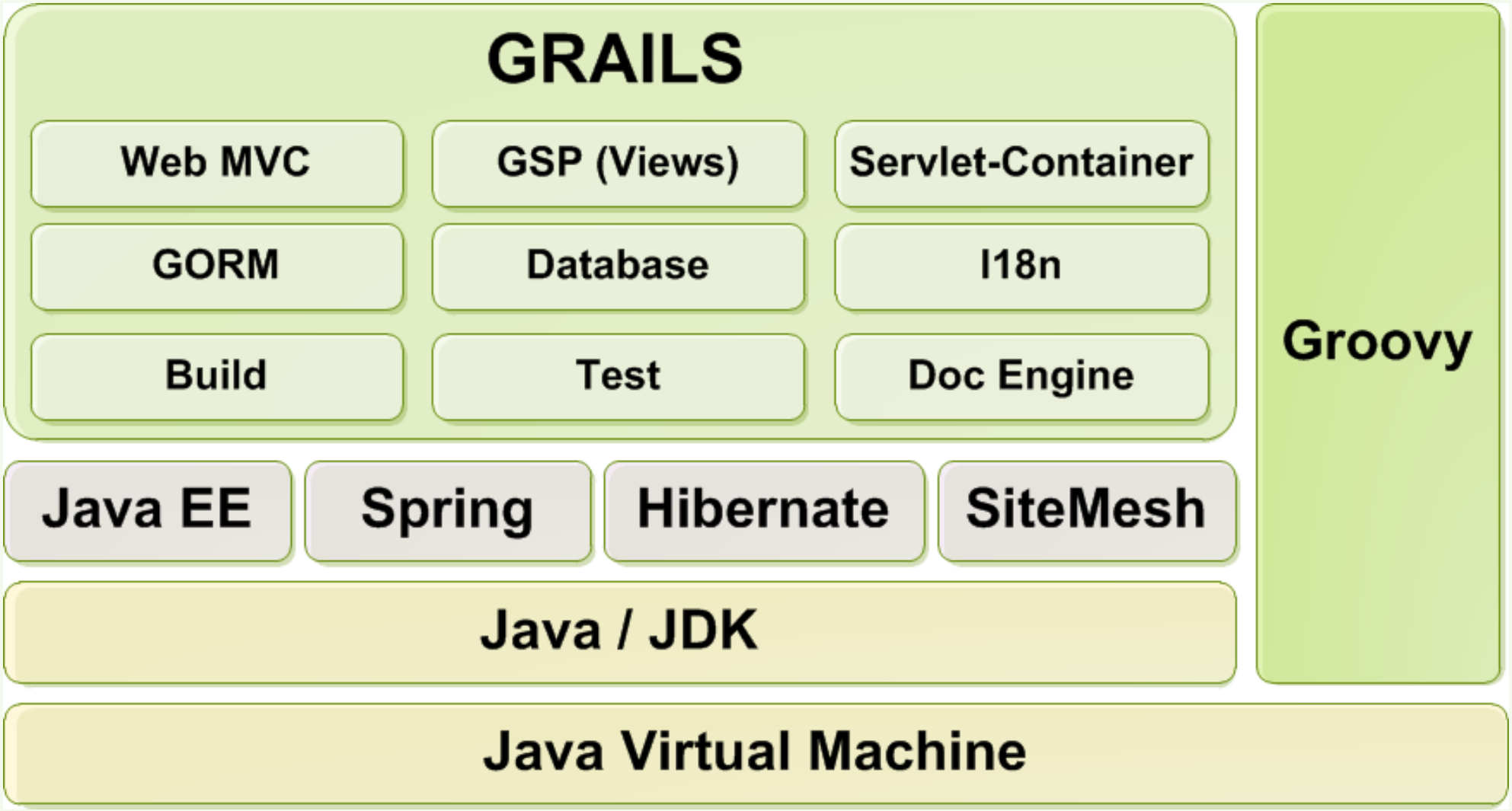
- **Grails im Überblick**
- Anforderungen an Enterprise Applikationen
- Enterprise Applikationen mit Grails

Was ist Grails?

A Full-Stack Web (MVC) Framework

- Install & GO!
- Nahtlose Java Integration, wird auf der Java Virtual Machine (JVM) ausgeführt
- Ursprünglich inspiriert von Ruby on Rails
- Open Source
- Grails Historie
  - Projekt-Beginn 2005
  - Grails 1.0 (02/2008)
  - Grails 2.3.2 (11/2013, aktuelles Release)
- Basiert auf Groovy

Aufbau von Grails

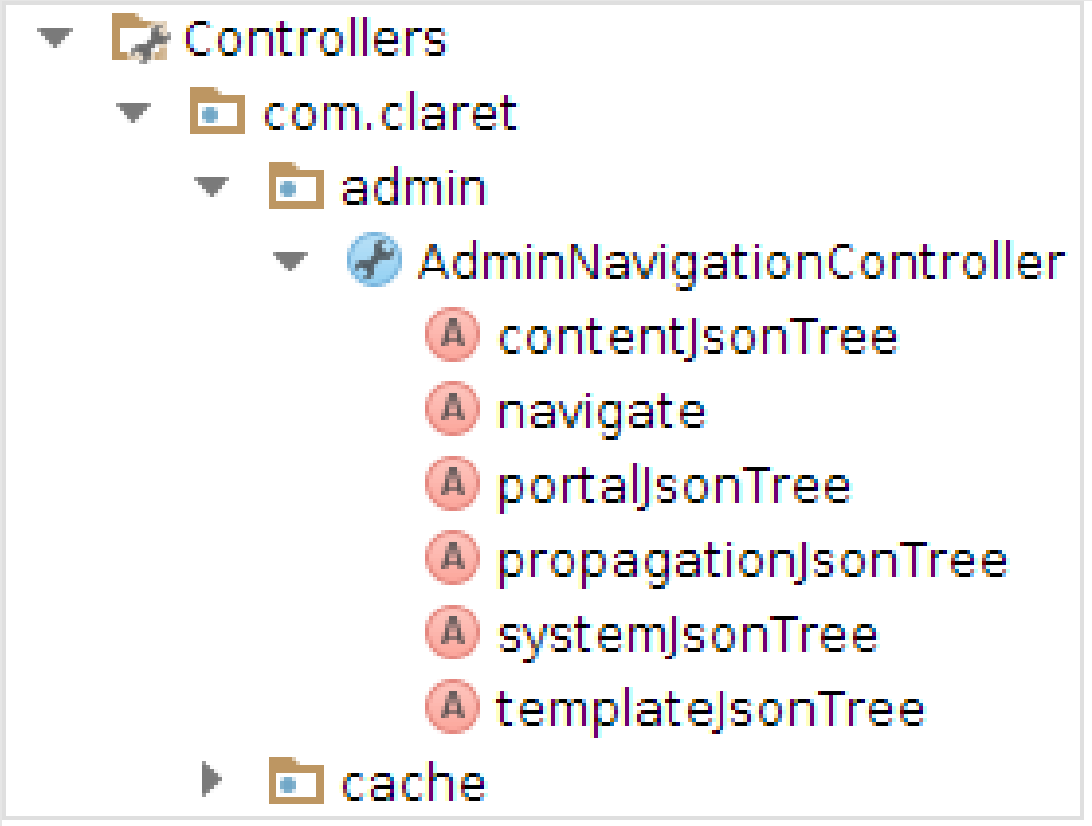


Schlüsselkonzepte

Der Standard-Weg ist konfiguriert, aber Änderungen sind möglich

- Convention over Configuration (CoC)
- Don’t repeat Yourself (DRY)

- Beispiele
  - Aufbau Projekt-Struktur
  - DB-Mapping
  - Klassennamen
  - Logging
  - Autowiring



Groovy als Basis

- Dynamische typisierte Sprache
- Basiert auf JVM
- Bietet zusätzliche Features zu Java

Beispiele

```
def names = ['DOAG SIG Security', 'SIG MySQL', 'OpenWorld', 'SIG Database: Migration']
println names

def sigEvents = names.findAll{ it ==~ /.*SIG.*/ }.sort()
println "number of SIG events: ${sigEvents.size()}"
sigEvents.each { println it }
```

Einfaches Arbeiten mit Grails am Beispiel von GORM

- Hibernate ist der Standard
- Austausch Persistenzschicht ist möglich
- DB-Operationen erfolgen direkt am Domain-Objekt

```
Portlet.delete()
Portlet.save(flush:true)
```

- Queries einfach und intuitiv

Dynamic Finders

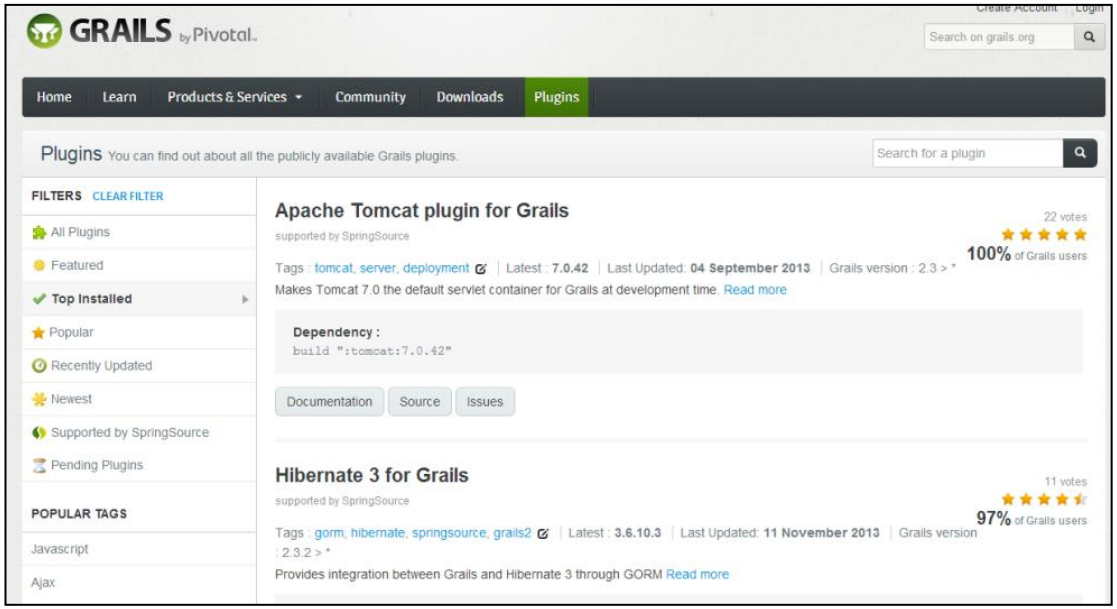
```
Portlet.findAllByPageAndPortal('Home', portalname)
```

Where-Queries

```
PortletInst.where{
        year(published) == 2013 && right != 'public'
}
```

Plugins

- In sich abgeschlossenes Bundle von Funktionalitäten
  - Wiederverwendbarkeit
  - Modularisierung / Kapselung
- Kern-Plugins: hibernate, resources, jquery, …
- Plugins können selbst entwickelt werden
- Grails Plugin-Repository mit ca. 1000 Plugins für
  - Frontend
  - Integration von Drittsystemen
  - Fachliche Funktionalitäten
  - Persistenzschicht
  - Nichtfunktionale Erweiterungen



Wo geht der Weg hin?

- Grails 2.4 (Anfang 2014)
  - @CompileStatic Integration
  - Spring 4.0 Support
  - Groovy 2.2 Support
- Grails 3.0 (2014)
  - Gradle als Build System
  - Unabhängigkeit vom Servlet Container
  - Application Profiles
  - Modularisierung

http://www.morguefile.com/archive/display/58914

Was ist cool an Grails?

- Herunterladen und loslegen
- Groovy macht vieles kürzer und schneller
- Zahlreiche Plugins steigern die Produktivität
- Full-Stack Unterstützung / Continuous Integration
  - Testing
  - Coverage
  - Build

http://www.morguefile.com/archive/display/196579

Agenda

Grails im Überblick

Anforderungen an Enterprise Applikationen

Enterprise Applikationen mit Grails

Was sind Enterprise Applikationen?

Enterprise software, also known as **enterprise application software** (EAS), is purposed-designed computer software used in the furtherance of the needs and objectives of the organizations; such purposes can vary widely as in a business, schools, interest-based user groups[1] and clubs, retailers, or government,[2] as opposed to software used by individuals. Enterprise software is an integral part of a (computer based) Information System, and as such includes web site software production.

http://en.wikipedia.org/wiki/Enterprise_application

Eigenschaften von Enterprise Applikationen sind oftmals

- Komplexe Geschäftslogik
- Zugriff auf relationale DB's
- Distributed Computing
- Verteilte Transaktionen
- Datenaustausch zwischen heterogenen Systemen
- Message-Oriented Middleware
- Directory und Naming Services
- Inter-Kommunikation (Mail, Dokumente, ..)
- Security
- Integration mit Legacy-Systemen
- Integration mit anderer Geschäftsbereichen / Organisationen
- Zentrale Administration und Wartung

Anforderungen an Enterprise Applikationen

Eine Auswahl relevanter Kriterien

- Verfügbarkeit
- Ausfallsicherheit
- Wartbarkeit, Erweiterbarkeit, Strukturierungsmöglichkeit
- Integration
- Security
- Kein Daten-Verlust bzw. Korruption bei Fehlern
- Testbarkeit
- Betrieb und Monitoring

Agenda

Grails im Überblick

Anforderungen an Enterprise Applikationen

Enterprise Applikationen mit Grails

Externe Konfigurationsmöglichkeiten

- Auslesen aus der DB

ID	ENCRYPTED	PROP_KEY	SUBKEY	PROP_VALUE
2	0	EXCLUDEFROMSOLR	(null)	None
3	0	MIGRATION_DIR_PM	FILES	/mnt/exensiofs01/User...
4	0	USE_LDAP	(null)	false

- Auslagerung in externe Dateien (Java-Property- oder Groovy-Datei)

deployment.groovy

```
// Local search Configuration
elasticSearch{
    client.hosts = [
            [host:'localhost', port:9300]
    ]
    client.mode = 'transport'
    cluster.name='mrs'
    indexName='mrs'
    client.transport.sniff = false
}
// LDAP Settings
ldap {
```

- JNDI (Java Naming and Directory Interface)

Nutzung von JNDI-Resourcen

- DB Anbindung

```
dataSource {
   jndiName = 'TW_DEMO_DS'
   dialect = 'org.hibernate.dialect.Oracle10gDialect'
   driverClassName = 'oracle.jdbc.OracleDriver'
   dbCreate = 'create-drop'
}
```

- Mail-Service

```
grails {
    mail {
        jndiName = 'TW_DEMO_MAIL_SESSION'
        disabled = false
    }
    mail.default.from = 'noReply@twitter-demo.de'
}
```

Security

- Spring-Security für Authentifizierung und Autorisierung

Features

- Authentifizierungsketten
- Windows-Authentifizierung
- Authentifizierungsmechanismen für Twitter, Facebook und diverse andere Provider
- AD- / LDAP-Anbindungen
- Hierarchische Rollen
- Einhängen von kundenspezifischen Mechanismen / Erweiterungen
- Switch-User Mechanismus

Fehlerbehandlung

- Exception-Handling: Verwendung des Java-Standard Mechanismus
- URL-Mapping / Error-Controller

```
"/admin/$action?" (controller:"user")

"500"(controller:"error", action:"show")
"404"(controller:"error", action:"show")
```

- Action Error Handlers für Controller
  - Actions mit einer Exception als erstes Argument werden aufgerufen falls Exception durch andere Action geworfen wird

```
class ListController {
    ...
    def syntax(SearchSyntaxException e) {
        render view: ' syntaxError'
    }
}
```

Testbarkeit

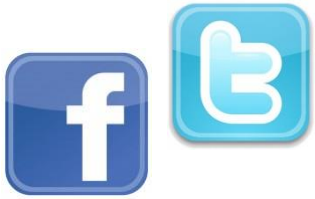
- „Out of the box“ Testing
- Tests-Klassen werden bei create-* Kommandos miterzeugt
- Unterstützte Testarten
  - Unit
  - Integration
  - Functional
- Viele Plugins für Testframeworks und Testdatengenerierung
  - Fixtures, Build-Testdata
  - Jasmine, jsUnit, Webtest
- Continuous Integration

Toolunterstützung

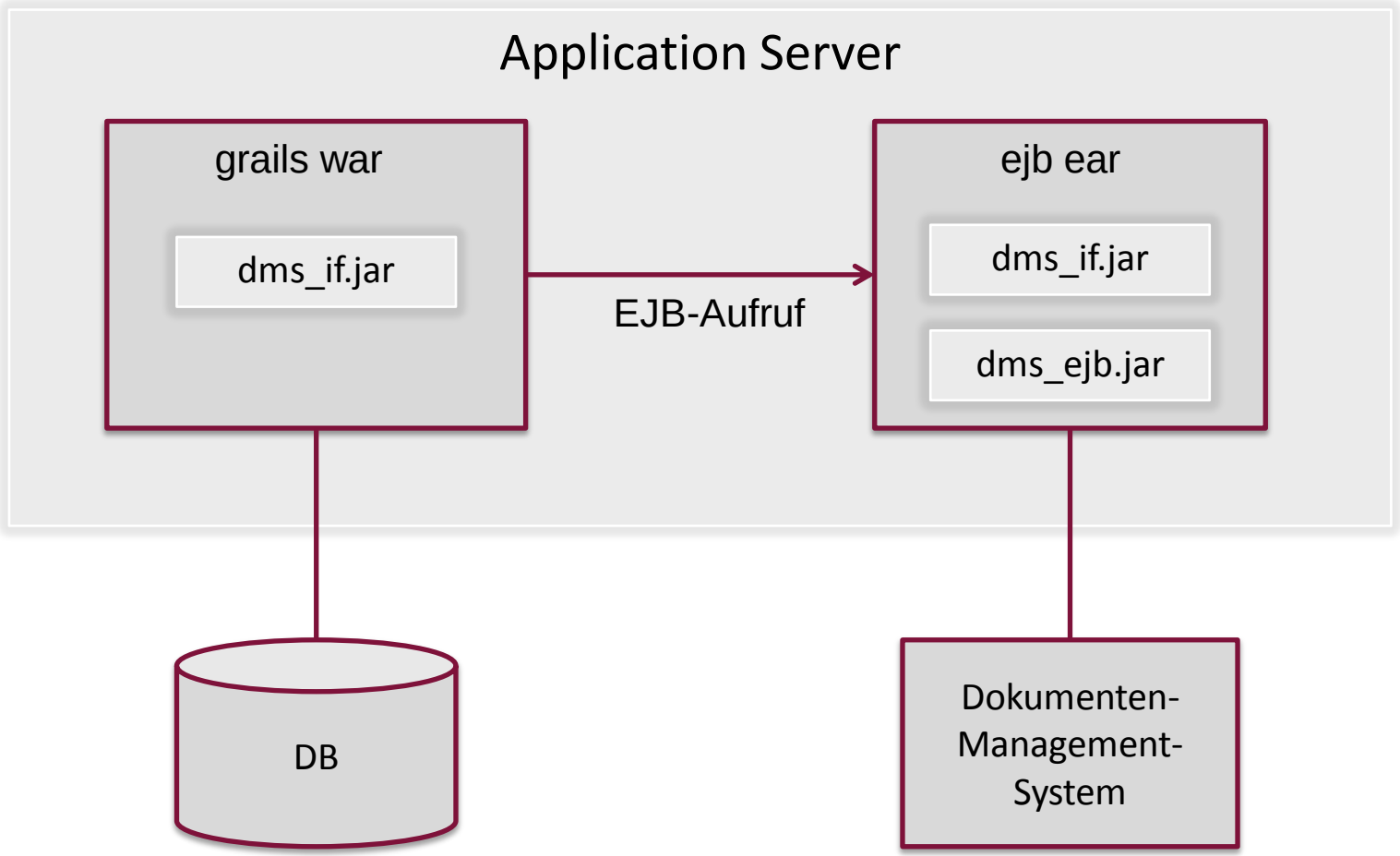
- Entwicklungsumgebung
- Testing
- Build
- Continuous Integration

Integration von Drittsystemen

eine Auswahl



Integrations-Szenario für JEE



Integrations-Szenario für JEE

resources.groovy

```
beans = {
    ejbJndi(org.springframework.jndi.JndiTemplate) {
        environment = [
        "java.naming.factory.initial":"weblogic.jndi.WLInitialContextFactory",
        "java.naming.provider.url":"t3://localhost:7001"
        ]
    }

    dctmService(org.springframework.jndi.JndiObjectFactoryBean) {
        lookupOnStartup = false
        jndiName = "DCTMService#de.exensio.dctm.facade.DCTMService"
        jndiTemplate = ref("ejbJndi")
        proxyInterface = de.exensio.dctm.facade.DCTMService.class as Class
    }
```

Aufruf im Service

```
class DocumentumTransferBatchService extends BatchService {
    def dctmService

    def testCall() {
        def serviceName = dctmService.getServiceName()
        log.info("serviceName: ${serviceName}")
        return serviceName
    }
```

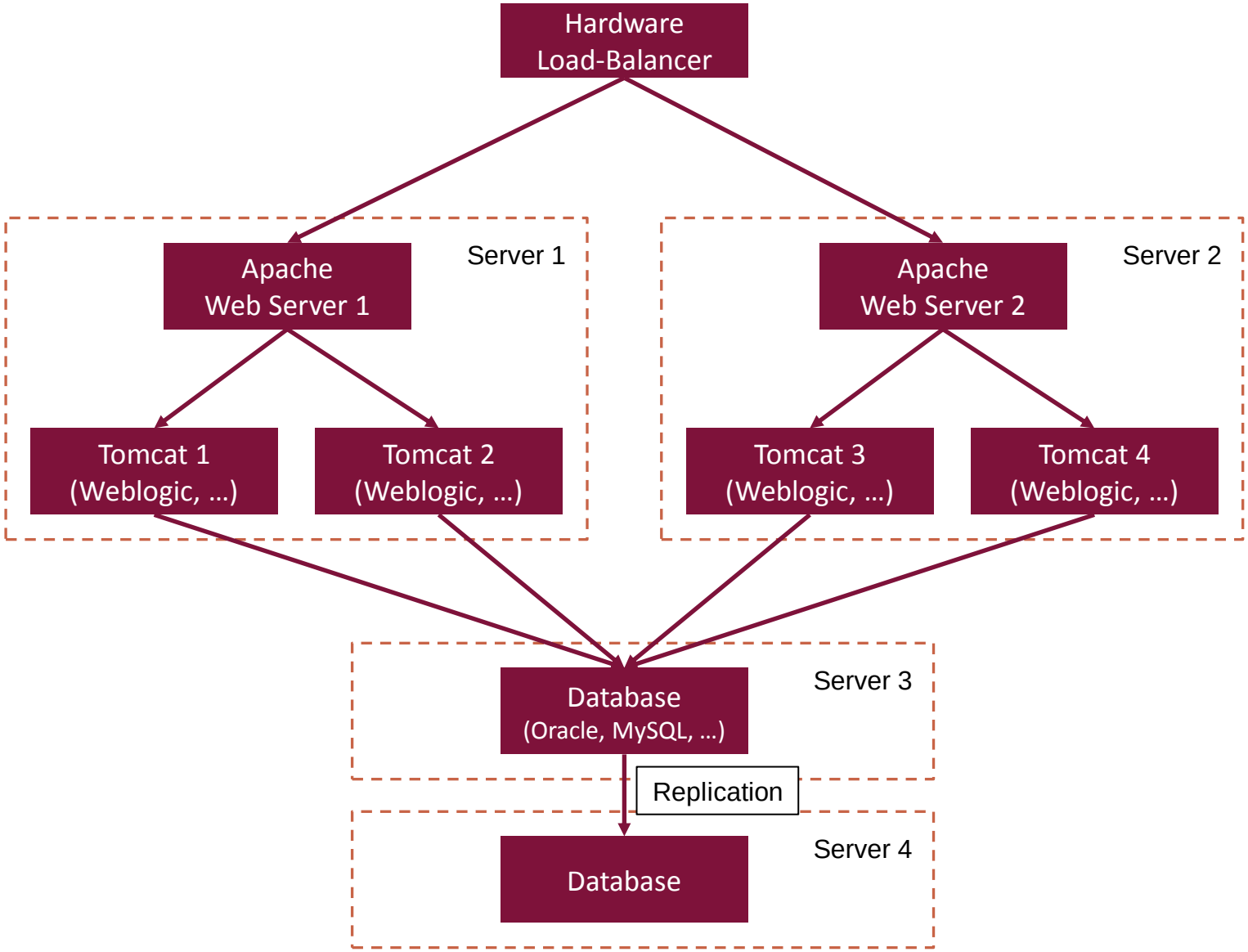
Produktivbetrieb

- Grails Deployment erfolgt über WAR-Datei

  → **Funktionalitäten der Java-Laufzeit-Umgebung sind verfügbar**

- Vorhandene JEE Infrastruktur und KnowHow kann verwendet werden
- Clustering
  - Load-Balancing
  - Session Replication
- Logging
- Monitoring
  - JMX
  - App-Server Überwachungs-Tools
  - Nagios

Clusterszenario für Produktivbetrieb



Verhalten von Grails in „großen“ Projekten

Keine negativen Erfahrungen bei Applikationen mit

- 200 Domain-Klassen
- 20 GB großen Datenbanken
- über 400 PT Aufwand
- dem Anbinden mehrerer DB‘s in einer Applikation
- der Verwendung von mehreren eigenen Plugins

→ Größere Vorhaben nach fachlichen Gesichtspunkten schneiden und Teile über den Plugin-Mechanismus auszulagern.

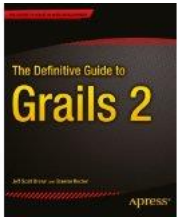
Performance Groovy

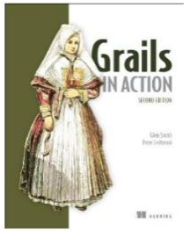
	Fibonacci	Pi (π) quadrature	Binary trees
Java	191 ms	97 ms	3.6 s
2.x Static compilation	**197 ms**	**101 ms**	**4.3 s**
1.8 Primitive optimizations	360 ms	111 ms	23.7 s
1.7 No prim. optimizations	2590 ms	3220 ms	50.0 s

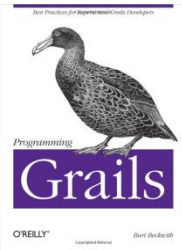
What makes Groovy groovy? — JAX London 2013 by Guillaume Laforge

Support und Verbreitung

- Regelmäßige Releases
- Patches
- Community
  - Mailinglisten
  - Zahlreiche Blogs
- Dokumentation
  - Bücher
  - http://grails.org/doc/latest/
- Konferenzen
  - Jährlich ca. 4 Konferenzen des Groovy/Grails Ökosystems

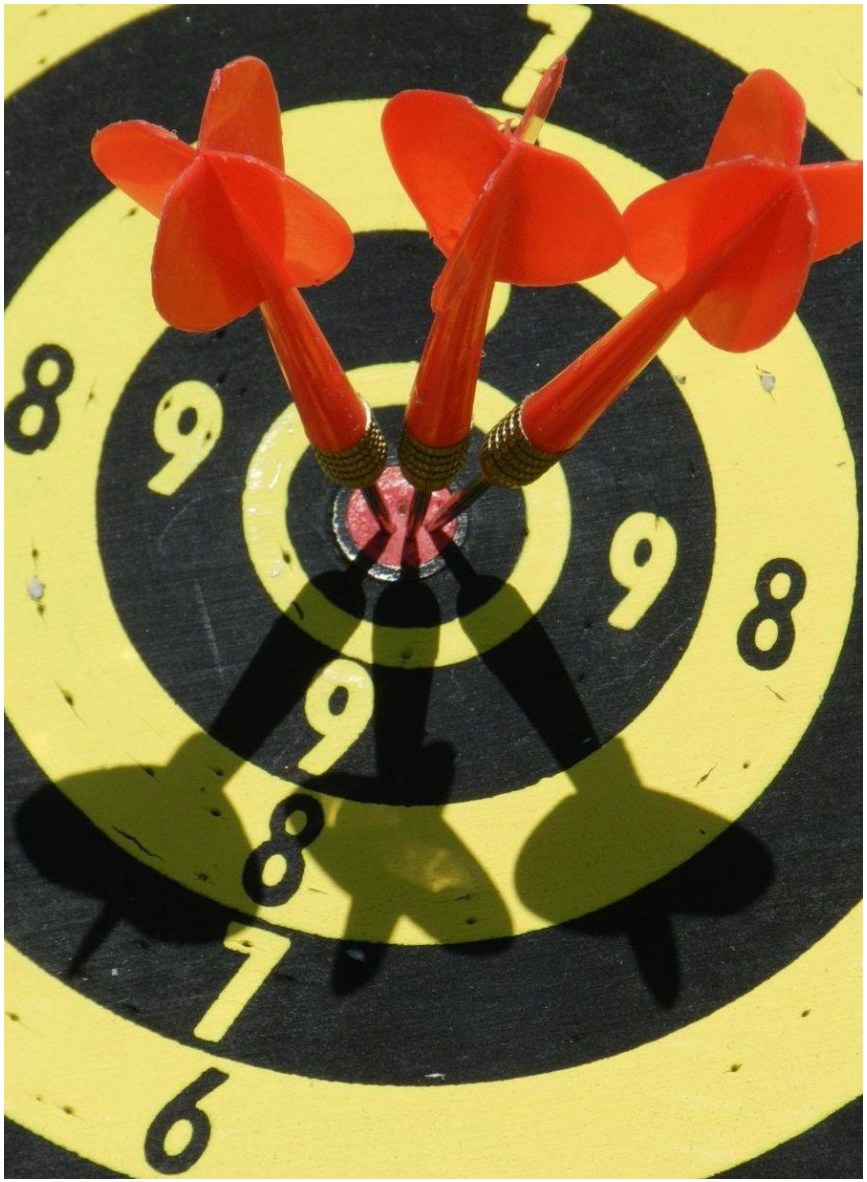






Warum Grails für Enterprise Applikationen einsetzen?

- Kompletter „Java-Stack“ ist verfügbar
- Einfache Integration von Drittsystemen
- Schnelle Umsetzung neuer Anforderungen
- Einfaches Testing mit Continuous Integration Unterstützung



http://www.morguefile.com/archive/display/865864

Was sollte beachtet werden?

- Strukturierung Code
  - Schichtentrennung
  - Keine schwergewichtige Taglibs
  - Logik in Services
- Vernachlässigung der Tests
- Zielplattform
- Version-Upgrades
- Vorsicht bei „0“-er Versionen (2.3.0, …)

http://www.morguefile.com/archive/display/28600

Fazit

- Grails bringt Basis zur Umsetzung von Enterprise Applikationen mit
- Tragfähigkeit für große Applikationen ist in Praxis erprobt
- Fehlende Funktionalitäten können bei Bedarf einfach selbst umgesetzt werden

exensio
intelligente informationssysteme

Vielen Dank!

exensio Gesellschaft für Informationstechnologie mbH
Am Rüppurrer Schloß 12
76199 Karlsruhe

http://www.exensio.de
http://blog.exensio.de
http://twitter.com/exensio

Partner:

elasticsearch.

ORACLE PARTNERNETWORK