

第一章

课程概述

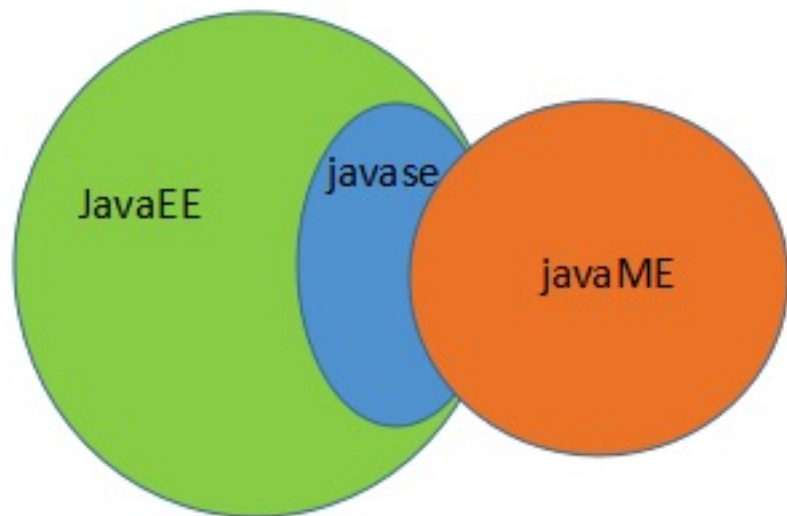
- Java企业版
- Spring家族

内容提要

- 什么是Java企业版应用
- Web应用的进化
- Java EE与MVC
- 容器 和 组件
- 初识Spring
- 构建开发环境

什么是Java EE? -----Java 平台

- 适用小型设备和智能卡Java 2平台Micro版（Java 2 Platform Micro Edition, J2ME）
- 适用桌面系统的Java 2平台标准版（Java 2 Platform Standard Edition, J2SE）
- 适用于创建服务器应用程序和服务的Java 2平台企业版（Java 2 Platform Enterprise Edition, J2EE）。



Java EE正式更名Jakarta EE

- 最终版本Java EE 8, [官网](#)
- Jakarta EE, [官网](#)
- [Java版权案](#)

企业级应用（大规模应用）

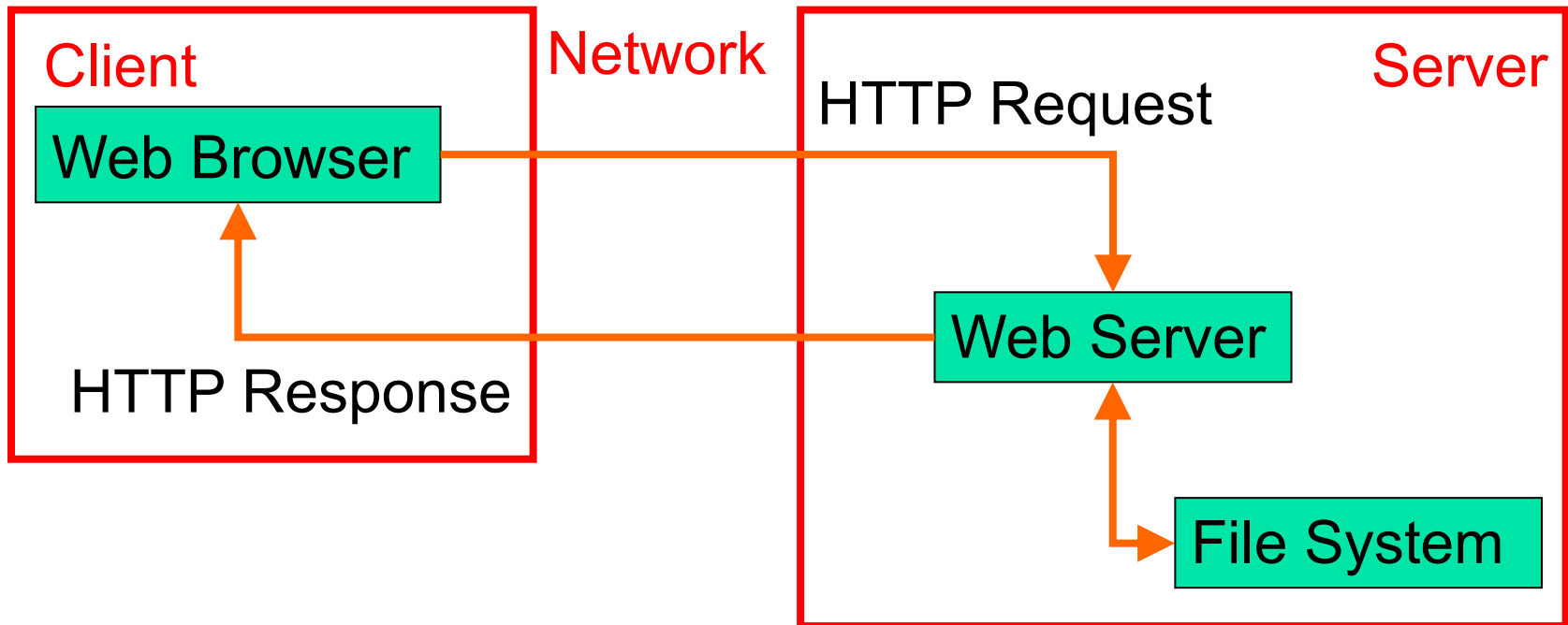
- 一般有众多的使用者，要有很长的生命周期，所以应用系统必须要稳定可靠
- 组件往往分布在异构的计算环境中，所以应用系统必须可以跨平台
- 对系统的可维护性、可扩展性与可重用性有很高的要求
- 需要有事务管理、安全管理、线程管理等等

内容提要

- 什么是Java企业版应用
- Web应用的进化
- Java EE与MVC
- 容器 和 组件
- 初识Spring
- 构建开发环境

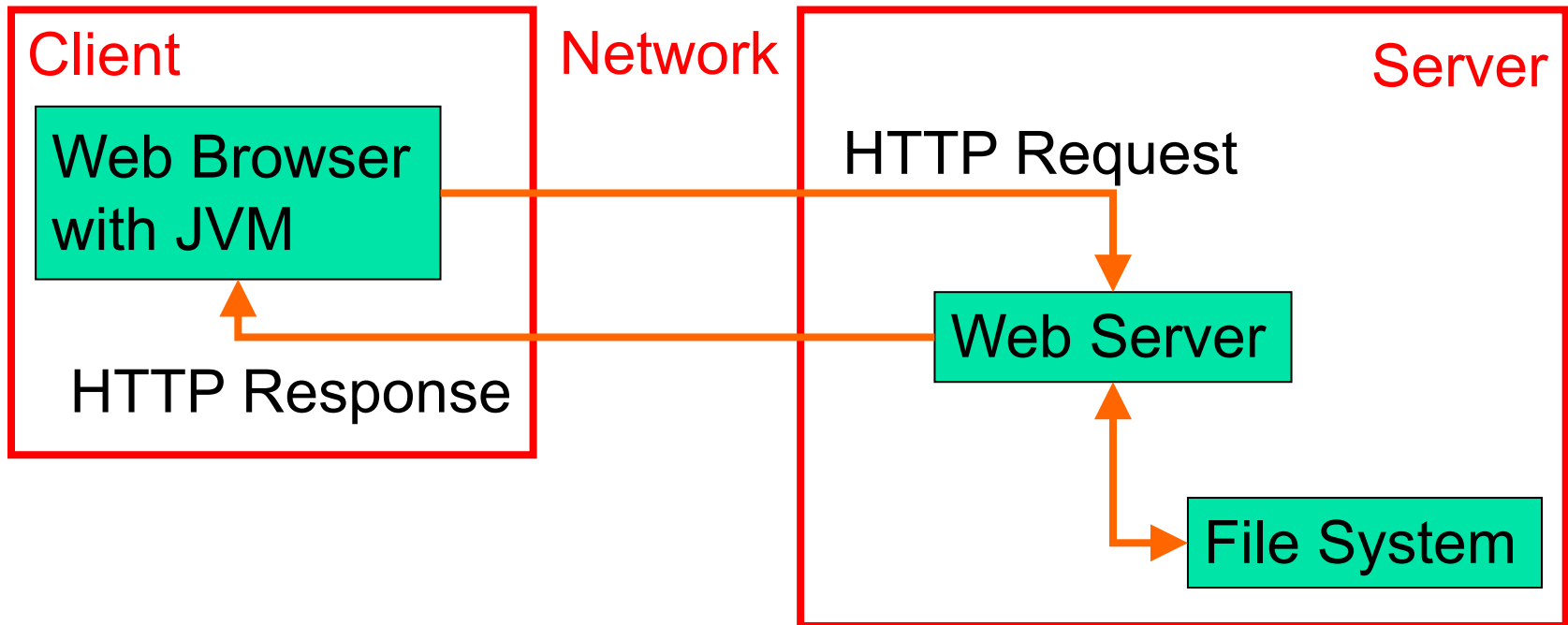
Web应用进化 – 静态

- Organizations want to make their information available to as many people in the world as possible
- This can be achieved by using the Web, delivering the information as static HTML pages



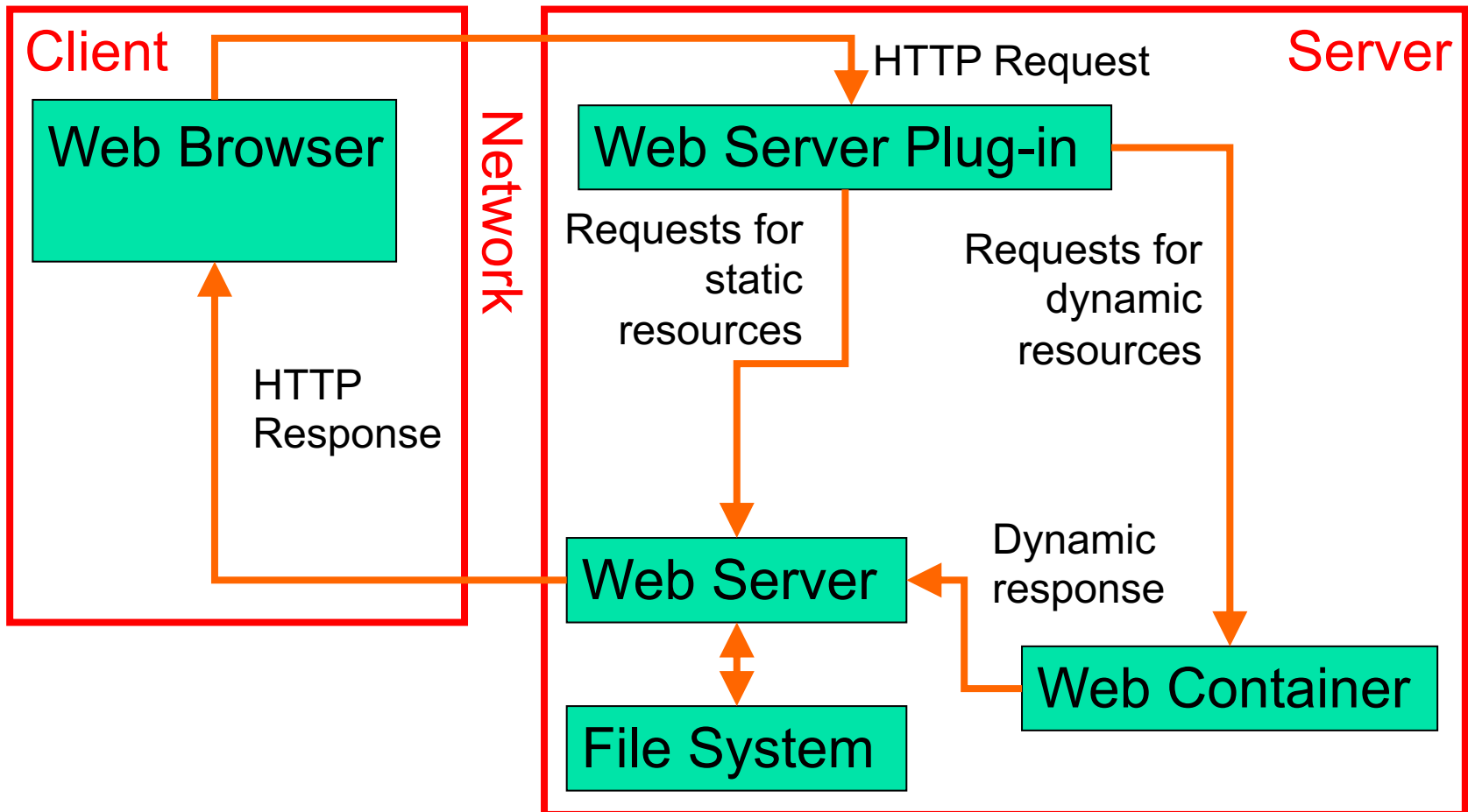
Web应用进化 - Applet

- With static HTML, users see passive page presentations which are always the same
- Presentation can be improved with Java applets or other client-side programs



Web应用进化 - Servlets

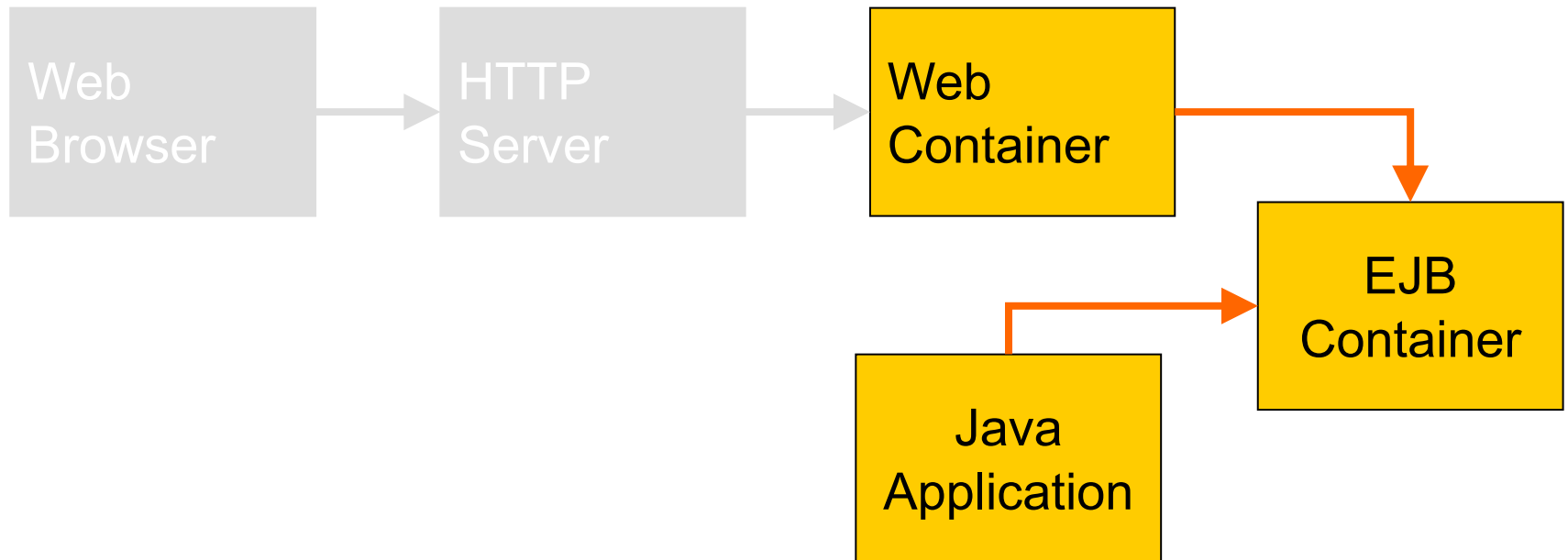
- Applets cannot access data on back-end systems
- A Web container can provide server-side components (such as servlets) to generate dynamic content



Web应用进化 - EJBs

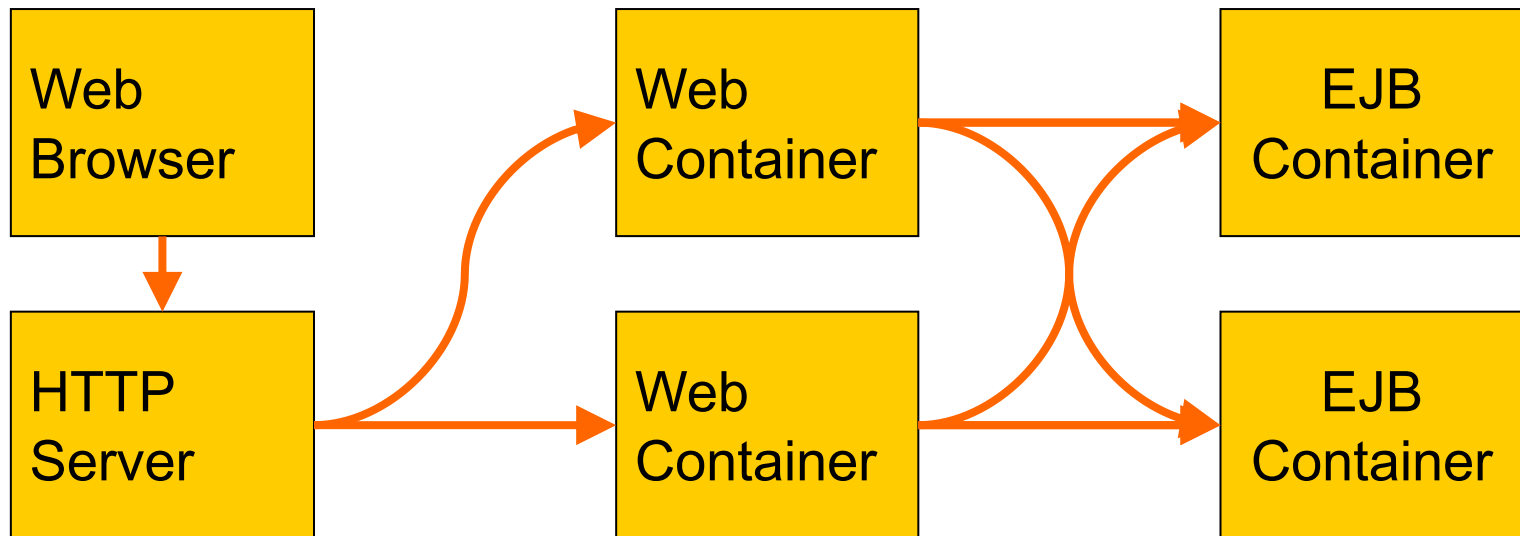
■ EJBs:

- Are available remotely over the network
- Encapsulate business rules, application-specific logic, and access to data
- Can be used by many different types of application concurrently
- Represent a central repository of business logic



Web应用进化 - Scalability

- Business requirements often involve high availability
- Improved performance may be required as business grows
- Both these requirements can be achieved through scaling
 - Servers can provide redundancy in the system
 - By sharing the load between servers, performance can be enhanced

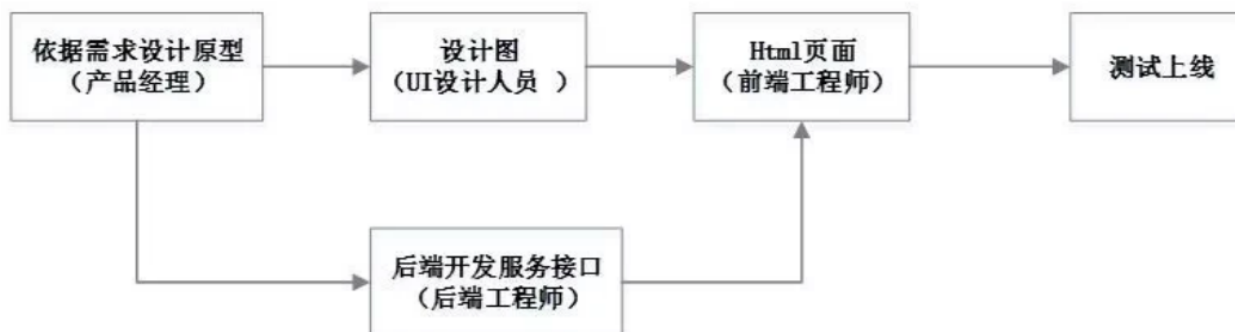


Web应用进化 – 前后端分离

初期

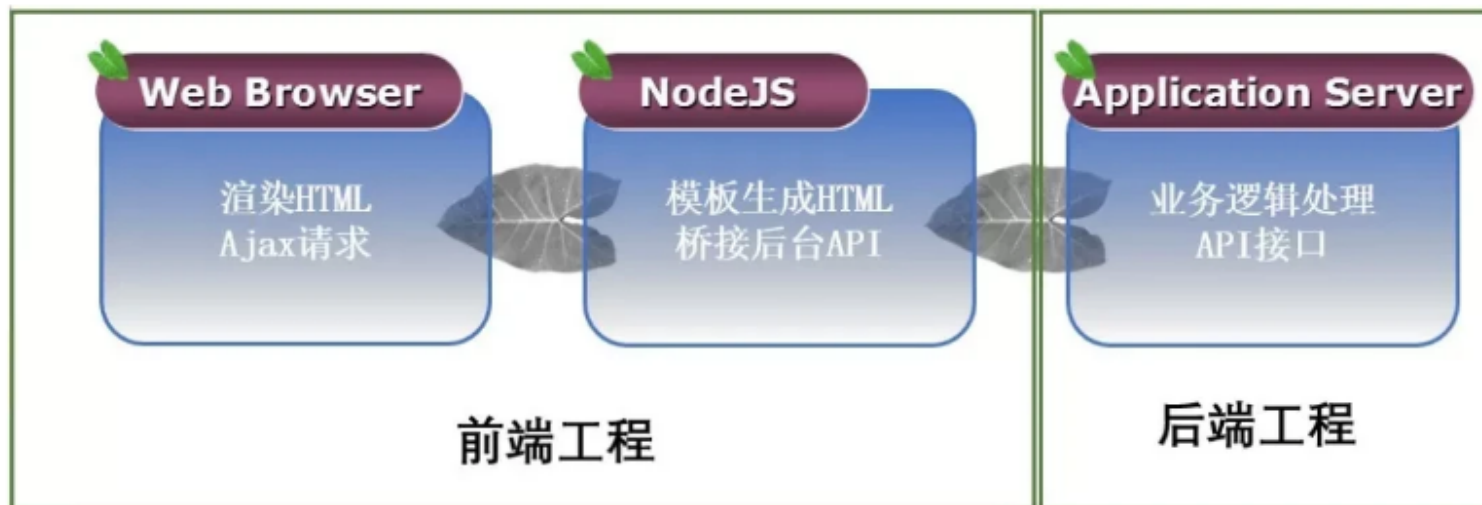
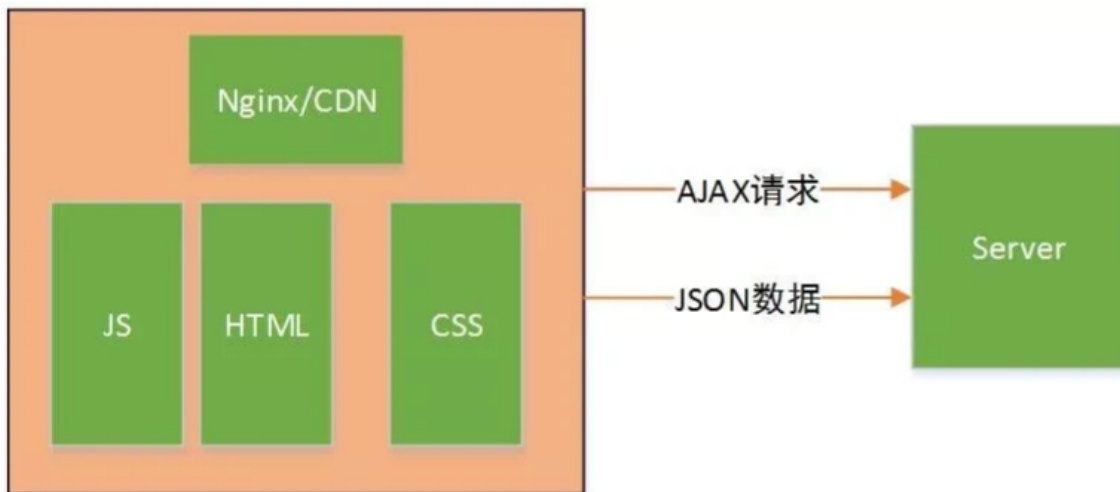


现在

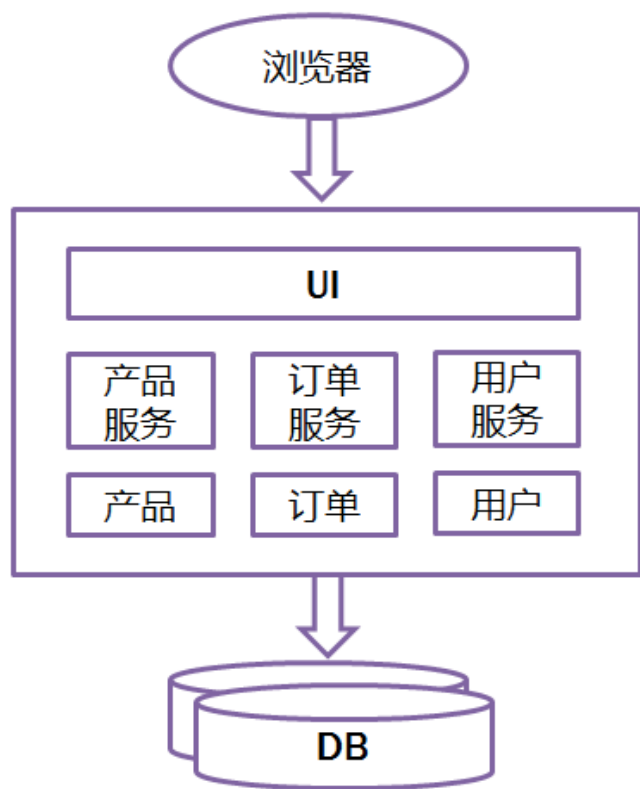


Serverless

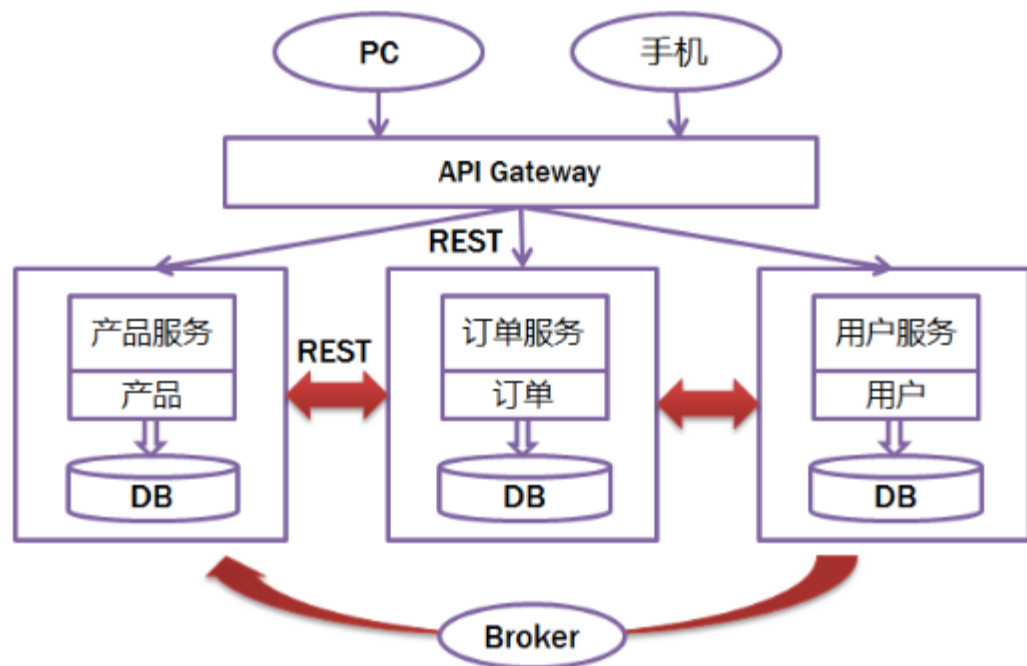
Web应用进化 – 前端工程



Web应用进化 – 微服务



单体架构

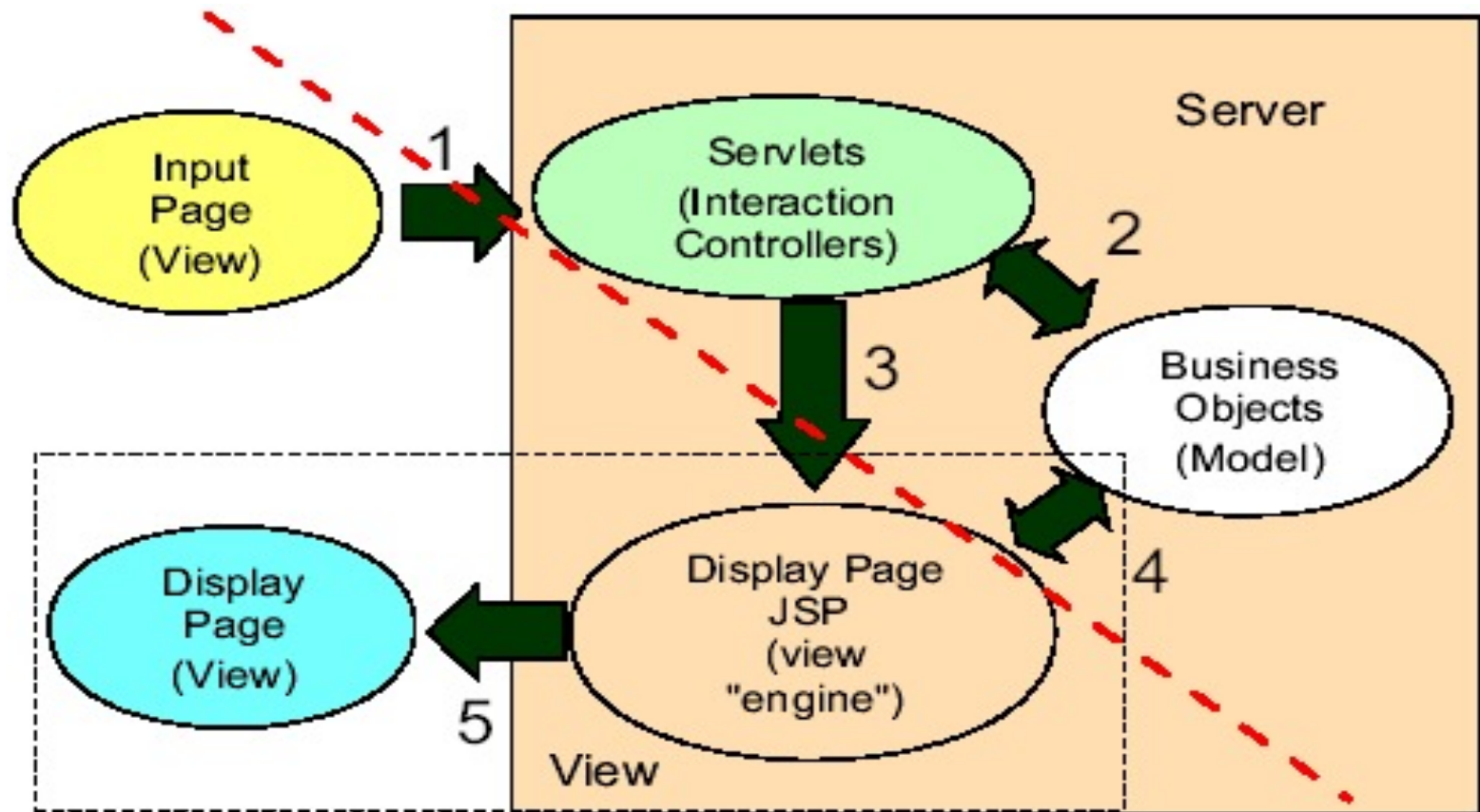


微服务架构

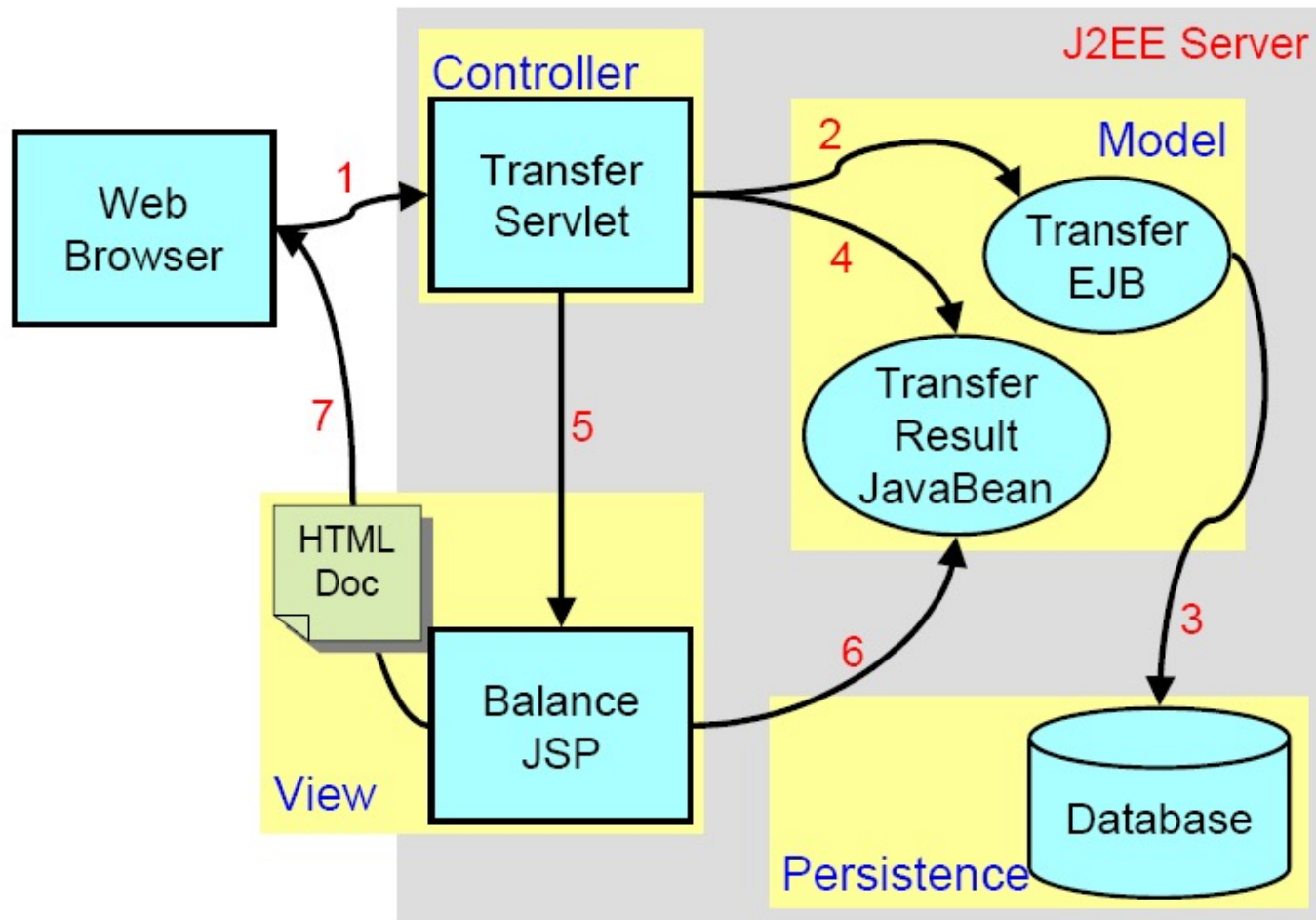
内容提要

- 什么是Java企业版应用
- Web应用的进化
- Java EE与MVC
- 容器 和 组件
- 初识Spring
- 构建开发环境

Model-View-Controller



Java EE 与 MVC



内容提要

- 什么是Java企业版应用
- Web应用的进化
- Java EE与MVC
- 容器 和 组件
- 初识Spring
- 构建开发环境

Java EE平台技术

■ 组件(Components)

--客户端组件

Applets

Application Clients

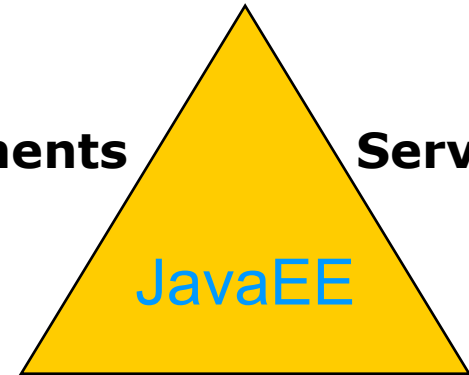
--服务器端组件

EJBS

Web Components(Servlets, JSP)

Components

Services



Communication

■ 服务(Services)

--Java EE组件利用的功能

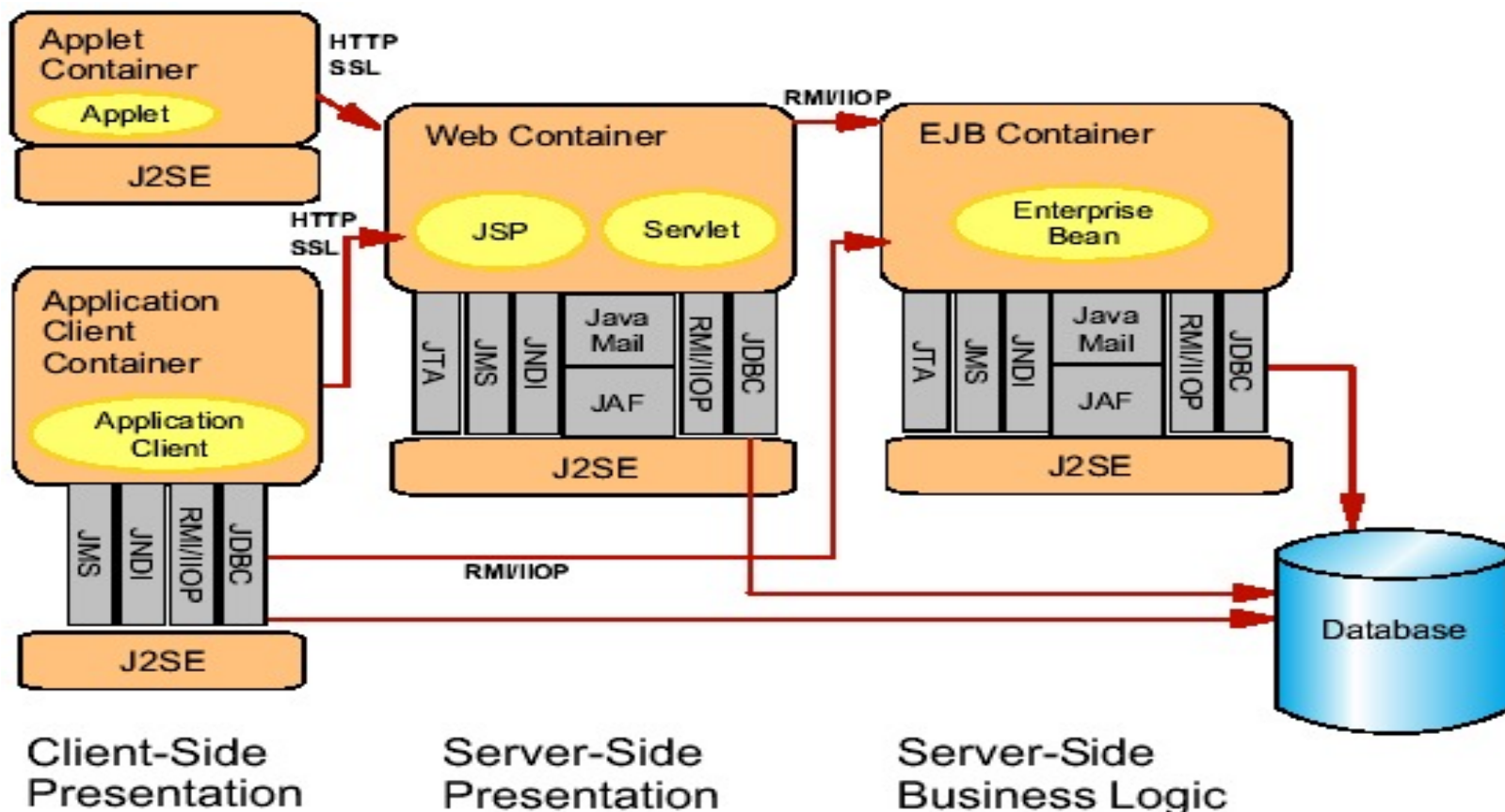
--由Java EE平台提供者(如WebSphere)执行的API

■ 通讯(Communication)

--使合作的对象间能够通信

--由容器提供

Java EE对象模型



What is Servlet?

- 扩展了HTTP服务器功能的Java objects
- 能够动态产生(页面)内容
- Better alternative to CGI, NSAPI, ISAPI, etc.
 - **Efficient**
 - **Platform and server independent**
 - **Session management**

What is EJB Technology?

■Java EE的基石

- **Java**服务器端服务框架的规范，软件厂商根据它来实现**EJB**服务器。应用程序开发者可以专注于支持应用所需的业务逻辑，而不用担心周围框架的实现问题。
- 事务管理，分布式，多层，可移植性，可扩展性，安全等等

■服务器端的组件

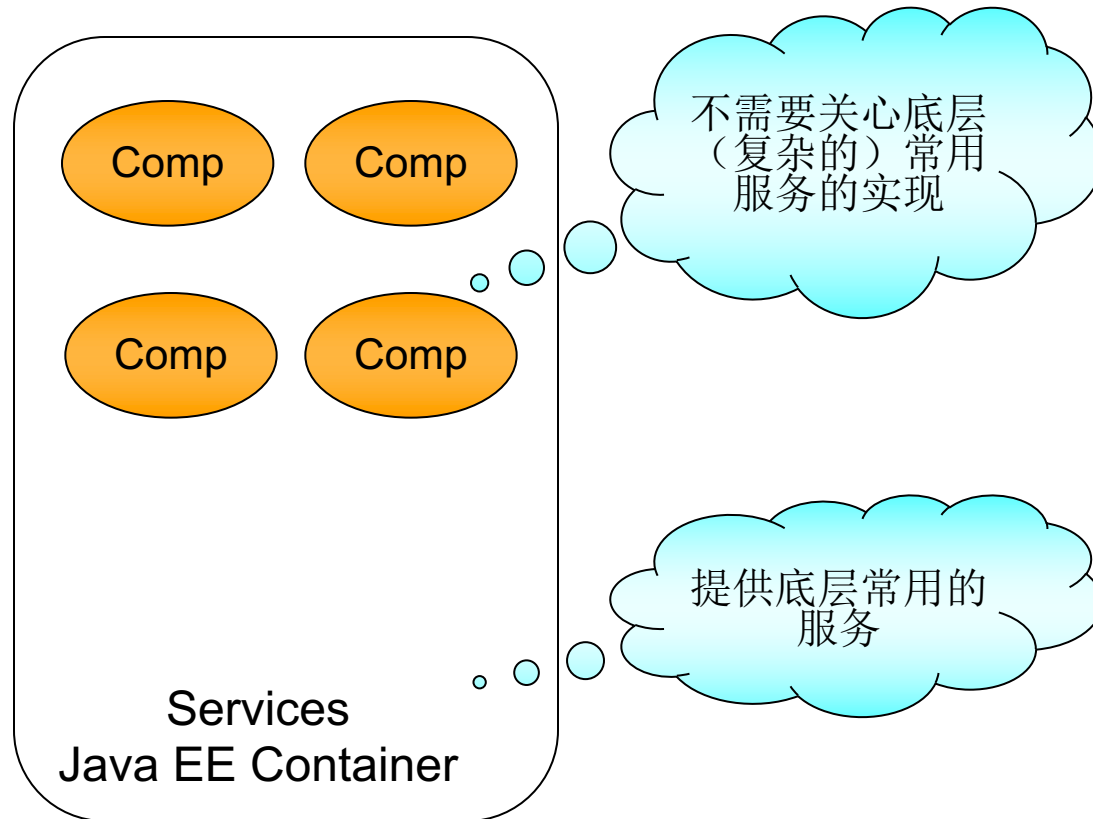
■共有三种EJB：

- 会话 (**session**) **bean**：代表短暂的与客户的会话，当客户结束执行时，会话**bean**及它的数据就消失了
- 实体 (**entity**) **bean**：实体**bean**代表存储在数据库的表，如果客户结束程序或服务器关闭，潜在的服务方法会将数据存储。
- 消息驱动(**message-driven**) **bean**。

Container in Java EE

- **EJB容器：**管理Java EE应用程序中企业Bean的运行。企业Bean和它们的容器在Java EE服务器中运行， e.g. WebLogic
- **Web容器：**管理Java EE应用程序中JSP页面和Servlet组件的运行。Web组件和容器也在Java EE服务器中运行， e.g. Tomcat
- **Application client container（应用程序客户端容器）：**管理应用程序客户端组件的运行。应用程序客户端和它的容器运行在客户机
- **Applet container（Applet容器）：**管理Applet的运行。由在客户端运行的浏览器和Java插件组成

Container & Component



内容提要

- 什么是Java企业版应用
- Web应用的进化
- Java EE与MVC
- 容器 和 组件
- 初识Spring
- 构建开发环境

Spring Framework是什么

- Spring是一个开源的控制反转(Inversion of Control ,IoC)和面向切面(AOP)的容器框架.它的主要目的是简化企业开发.

IOC (Inversion of Control) 控制反转

```
public class PersonServiceBean {  
    private PersonDao personDao = new PersonDaoBean();  
  
    public void save(Person person){  
        personDao.save(person);  
    }  
}
```

PersonDaoBean 是在应用内部创建及维护的。所谓控制反转就是应用本身不负责依赖对象的创建及维护，依赖对象的创建及维护是由外部容器负责的。这样控制权就由应用转移到了外部容器，控制权的转移就是所谓反转。

依赖注入(Dependency Injection)

当我们把依赖对象交给外部容器负责创建，那么PersonServiceBean类可以改成如下：

```
public class PersonServiceBean {  
    private PersonDao personDao ;  
    //通过构造器参数，让容器把创建好的依赖对象注入进PersonServiceBean，当然也可以使用setter方法进行注入。  
    public PersonServiceBean(PersonDao personDao){  
        this.personDao=personDao;  
    }  
    public void save(Person person){  
        personDao.save(person);  
    }  
}
```

所谓依赖注入就是指：在运行期，由外部容器动态地将依赖对象注入到组件中。

为何要使用Spring Framework

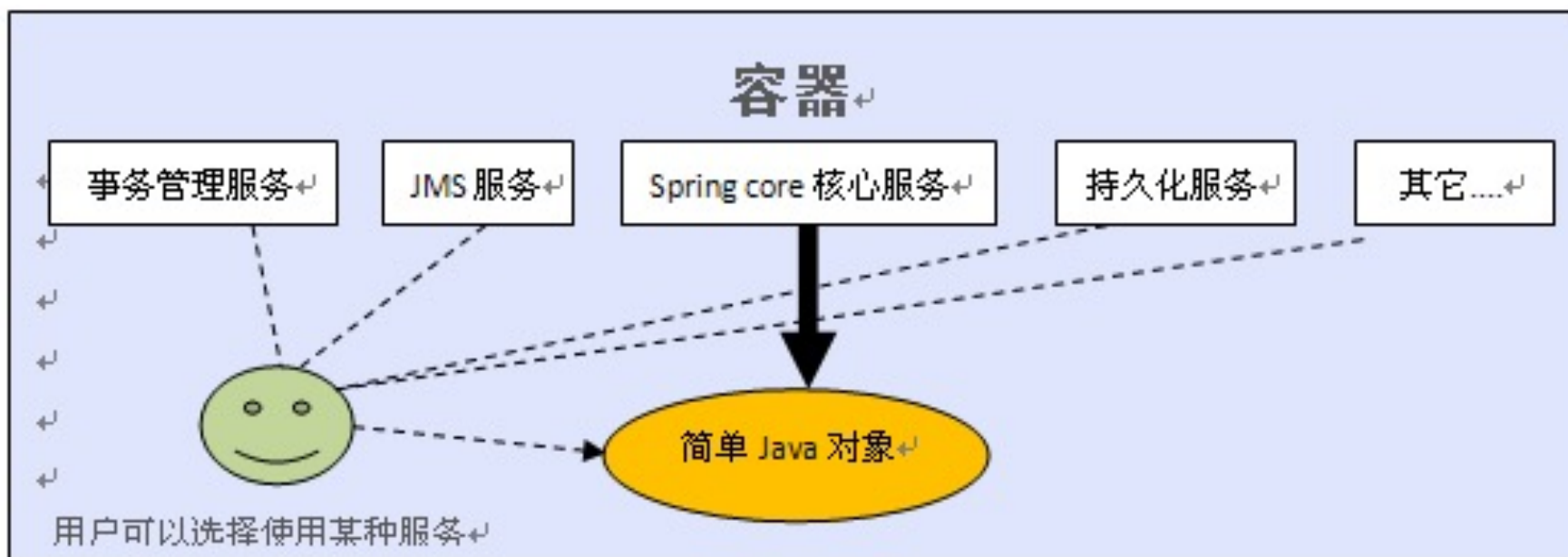
- 降低组件之间的耦合度,实现软件各层之间的解耦。



- 可以使用容器提供的众多服务,如:事务管理服务、消息服务等等。当我们使用容器管理事务时,开发人员就不再需要手工控制事务.也不需处理复杂的事务传播。
- 容器提供单例模式支持,开发人员不再需要自己编写实现代码。
- 容器提供了AOP技术,利用它很容易实现如权限拦截、运行期监控等功能。
- 容器提供的众多辅助类,使用这些类能够加快应用的开发,如:JdbcTemplate、HibernateTemplate。
- Spring对于主流的应用框架提供了集成支持,如:集成Hibernate、JPA、Struts等,这样更便于应用的开发。

使用Spring Framework的好处

- 当使用spring Framework时，我们可以选择使用容器提供的众多服务



Spring Framework的历史

- 诞生于 2002 年，成型于 2003 年，最早的作者为 Rod Johnson
 - 《Expert One-on-One J2EE Design and Development》
 - 《Expert One-on-One J2EE Development without EJB》
- 目前已经发展到了 Spring 5.x 版本，支持 JDK 8-11 及 Java EE 8



Spring, 始于框架, 但不不限于框架

Spring: the source for modern java



Spring, 始于框架, 但不限于框架

(1) Spring Framework (2) Spring相关项目 (3) 整个Spring家族

Spring Framework

- **Spring Framework**

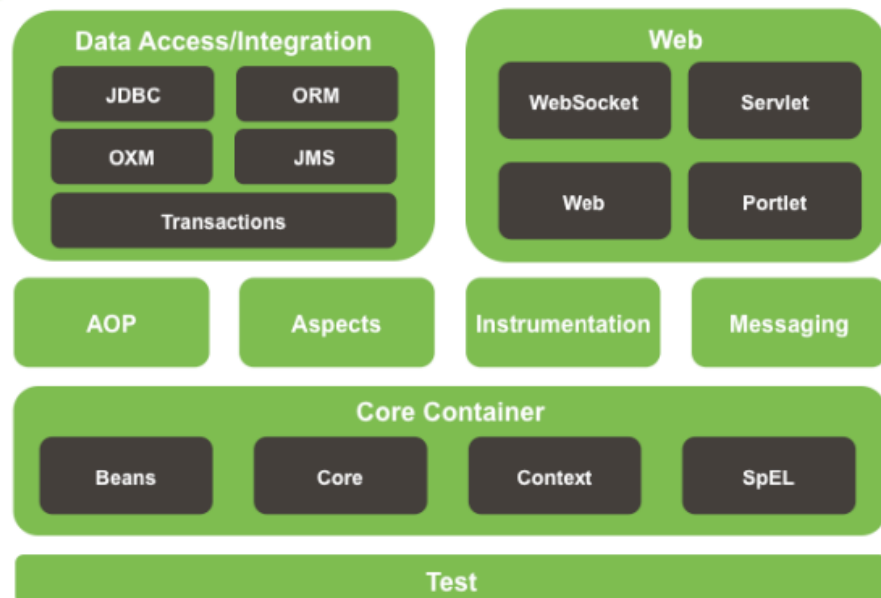
- 用于构建企业级应用的轻量级一站式解决方案



Spring Framework Runtime

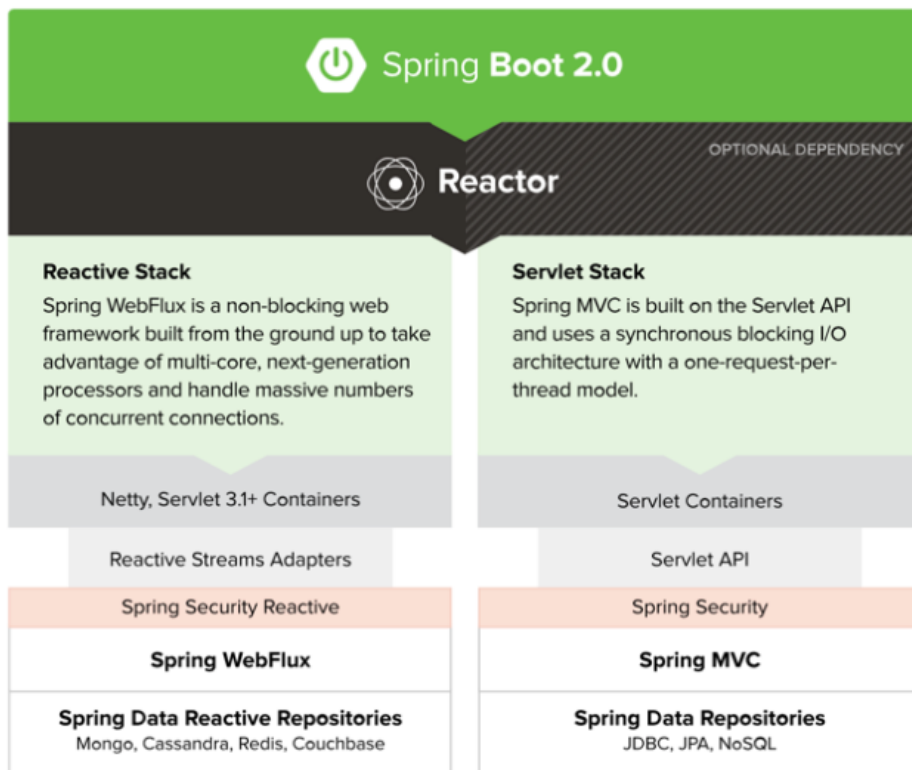
- **设计理念**

- 力争让选择无处不在
- 体现海纳百川的精神
- 保持向后兼容性
- 专注 API 设计
- 追求严苛的代码质量



Spring Boot

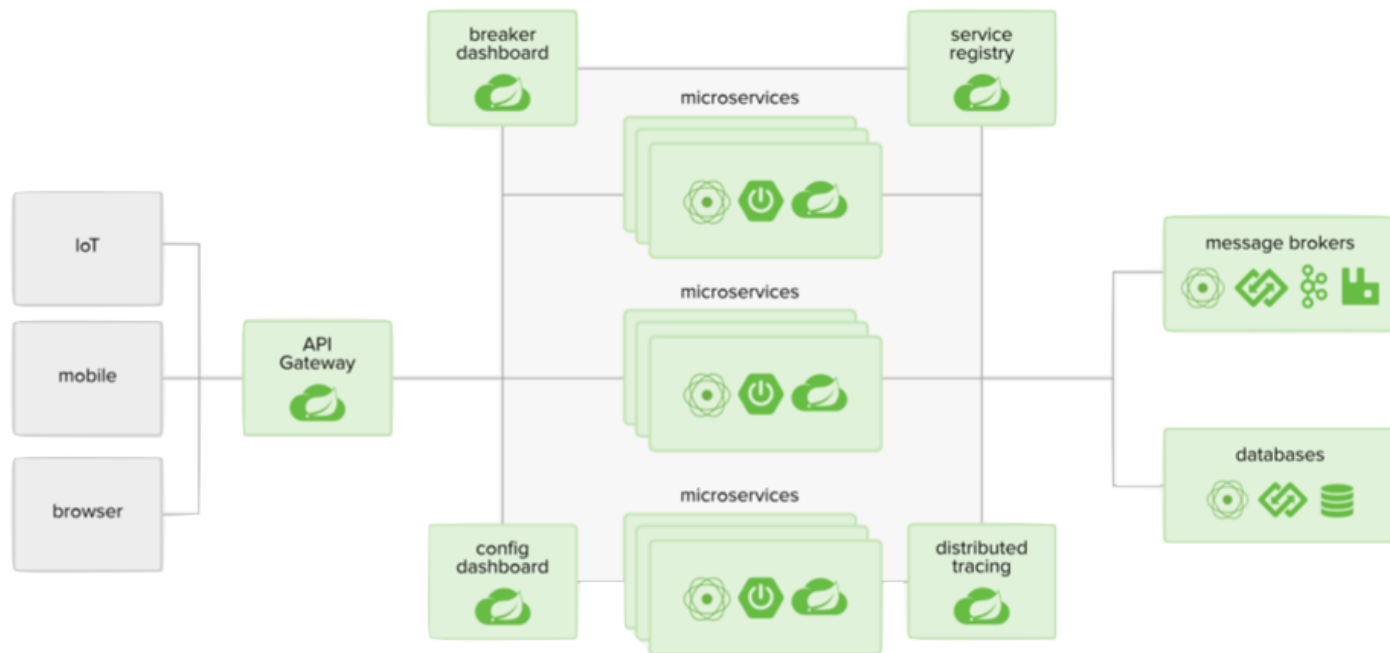
- 快速构建基于Spring的应用程序
 - 快、很快、非常快
 - 进可开箱即用，退可按需改动
 - 提供各种非功能特性
 - 不用生成代码，没有 XML 配置
- 在本课程中，你还会看到
 - Spring Data、Spring MVC、Spring WebFlux.....



Spring Cloud

- 简化分布式系统的开发

- 配置管理
- 服务注册与发现
- 熔断
- 服务追踪
-



Spring 5.x 的改变

改动点

改变的意义

一些思考

Java 8+、Kotlin

语言车轮滚滚向前

还在用低版本的 Java 我该怎么办

WebFlux

异步编程模式的崛起

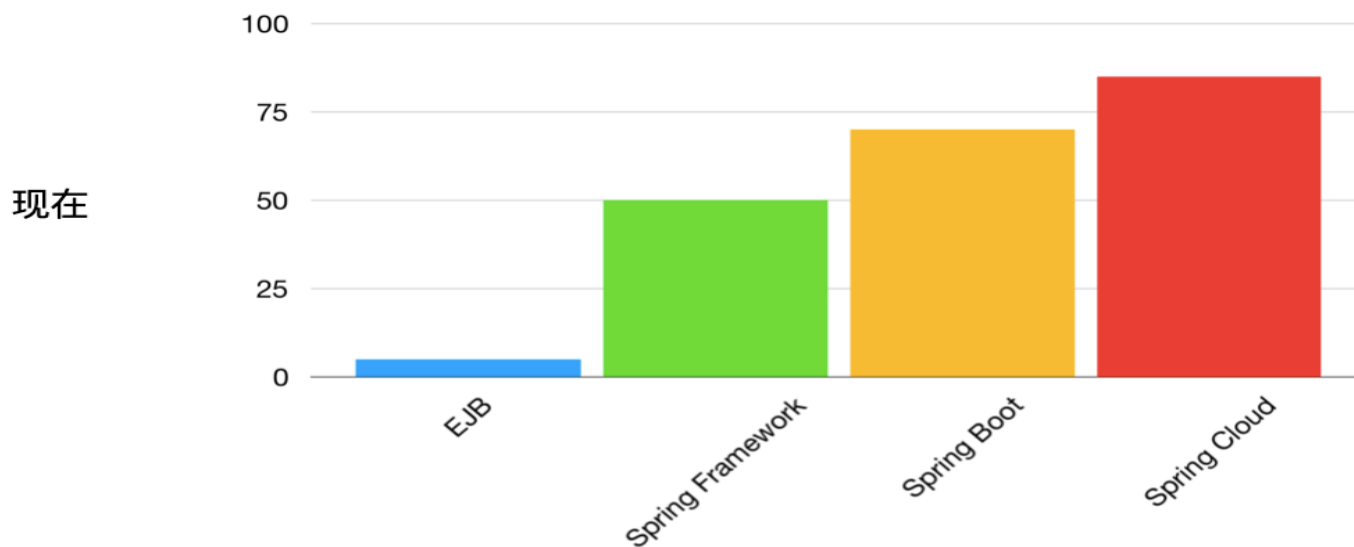
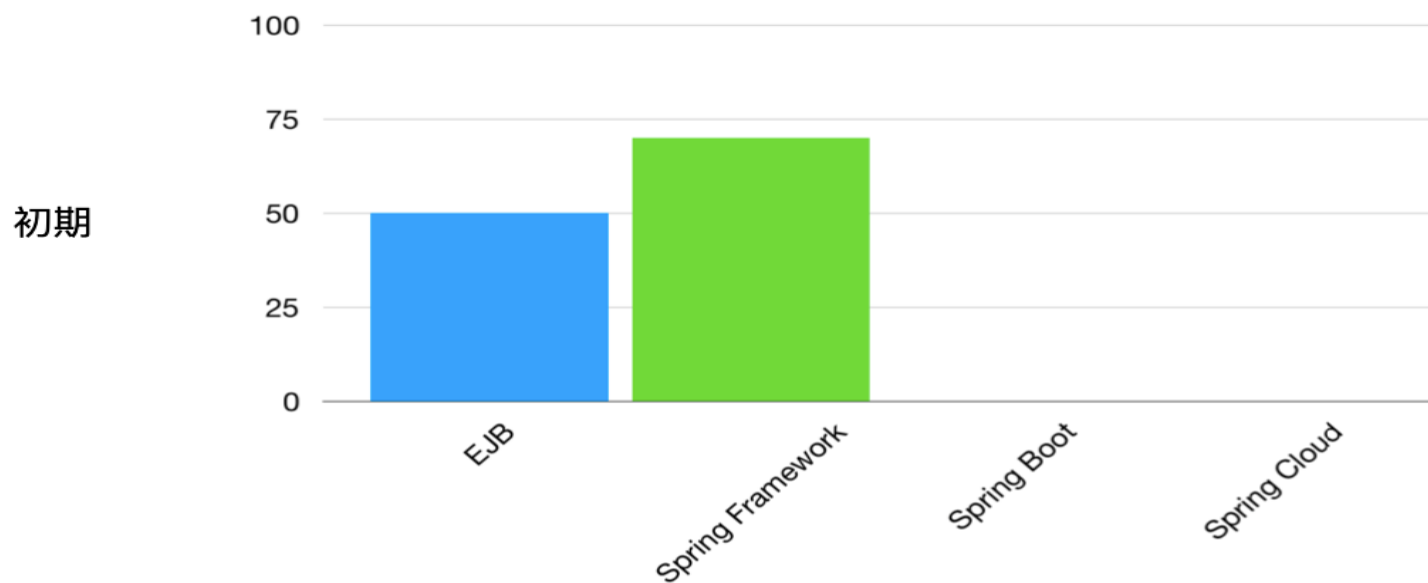
全面落地尚需时日

去掉了很多支持

Portlet 过时了、Velocity 不维护了、
JasperReport 不流行了

库有千千万，我该怎么选？

技术趋势



第一个Spring程序，Hello World

- 通过 Spring Initializr 生成骨架
- 编写第一段代码
- 运行你的程序
- 分析项目结构

SPRING INITIALIZR bootstrap your application now

Generate a Maven Project with Java and Spring Boot 2.1.1

Project Metadata

Artifact coordinates

Group

Artifact

Dependencies

Add Spring Boot Starters and dependencies to your application

Search for dependencies

Selected Dependencies

Web

[Generate Project](#)

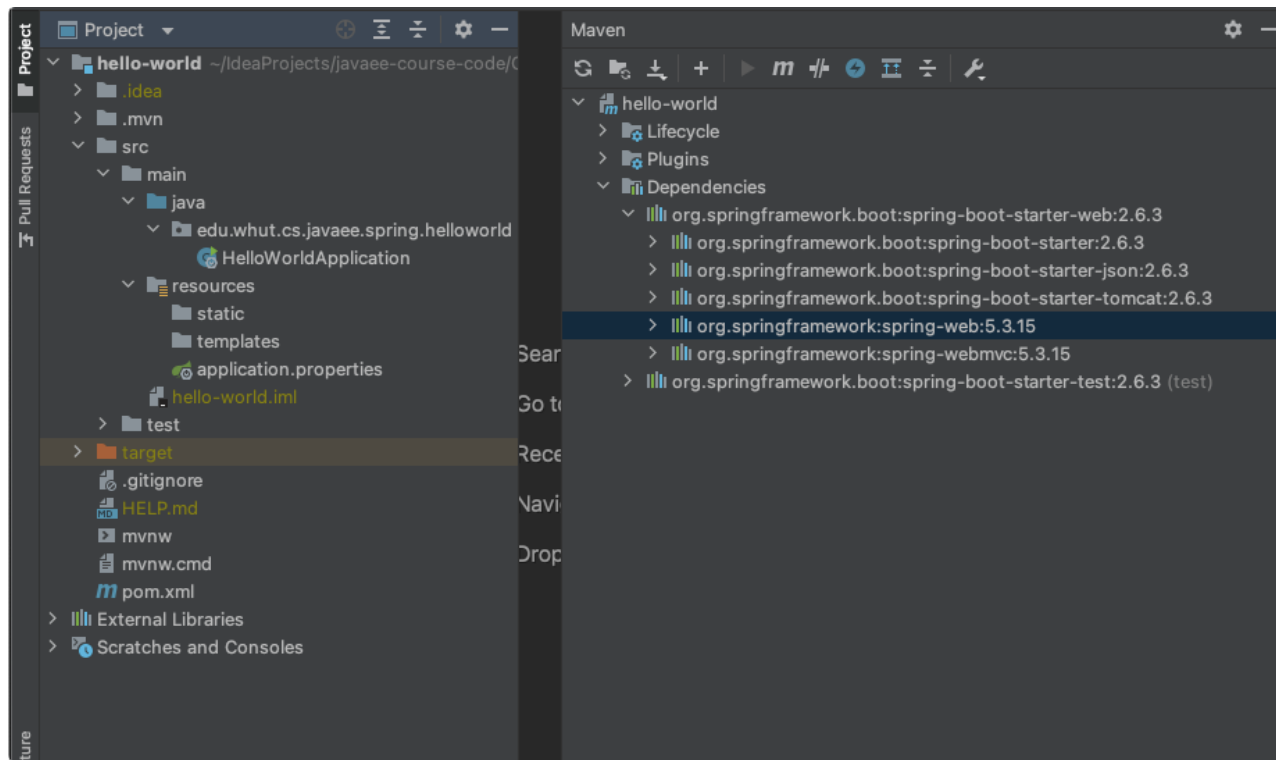
Don't know what to look for? Want more options? [Switch to the full version.](#)

start.spring.io is powered by [Spring Initializr](#) and [Pivotal Web Services](#)

Hello World 项目结构

hello-world

- 自动生成的 **Maven** 工程
 - pom.xml
 - 包含 main 方法的 Java 程序
 - 测试类
 - 配置文件



pom.xml 文件

依赖 spring-boot-starter-parent

- 方便快捷
- 自动引入 spring-boot-dependencies
- 自动配置 spring-boot-maven-plugin

```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.6.3</version>
  <relativePath/> <!-- lookup parent from repository -->
</parent>
<groupId>edu.whut.cs.javaee.spring</groupId>
<artifactId>hello-world</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>hello-world</name>
<description>hello-world</description>
<properties>
  <java.version>11</java.version>
</properties>
```


内容提要

- 什么是Java企业版应用
- Web应用的进化
- Java EE与MVC
- 容器 和 组件
- 初识Spring
- 构建开发环境

开发环境

- Java 8 / Java 11 (由于会用到Lambda, 必须是Java 8+), [下载](#)
- IntelliJ IDEA (安装 Lombok 插件), [下载](#)
- Apache Maven(如果不用命令行编译, 也可以用IDE自带的), [下载](#)
- Docker Desktop(用于在本地启动一些演示用的依赖设施), [下载](#)