Web应用开发

初稿：2015-10-9

最新：2015-12-7

张彦

MEIZU
魅族互联网新兵营

大纲

- Web应用概述
  - 什么是web应用
  - Web应用的发展
  - Java开发web应用
  - 接口类和网页类web应用
  - 优缺点
  - 前端优化
  - 技能要求
- Java web应用
  - 典型目录结构
  - servlet/filter/listener
  - request/response/session/application
  - MVC模式
  - 运行环境
  - 常用技术
  - 单元测试
- 学习资源
- 一个例子
- 附录：git和jenkins

Web应用概述

- 什么是web应用
- Web应用的发展
- Java开发web应用
- 接口类和网页类web应用
- Web应用的优缺点
- 前端优化
- 技能要求

Web应用概述

- 什么是web应用程序
  - 基于浏览器运行
    - 随着移动互联网的发展，web应用也大量用于移动app的开发
  - 使用http/https协议
  - 轻客户端，业务逻辑由服务侧实现
    - 随着移动互联网发展，越来越多的移动应用app采用了web应用作为后台，其业务逻辑也部分的前移到了客户端

Web应用概述

- Web应用的发展
  - 静态页面
  - 动态页面
    - Cgi
    - Asp/php/perl/…
    - Java servlet/jsp
    - Python
    - ……

Web应用概述

- Java开发web应用
  - Servlet技术
    - Servlet
    - Filter
    - Listener
    - Jsp
    - Tag
    - ……

Web应用概述

- 接口类和网页类web应用

	接口类	网页类
展示效果	不关注	关注
Session	基本不使用	使用
适用场合	移动app/ajax调用	浏览器访问

Web应用概述

- 优缺点
  - 优点
    - 协议简单，开发效率高
    - 易于横向扩展
    - 用户不需要安装专用软件
  - 缺点
    - 性能不高
    - 特定的应用场合不适合web应用，例如即时通讯，游戏等

Web应用概述

- 前端优化
  - Html
    - 善用html标签，对seo友好
    - 避免html语言格式错误
    - 不同浏览器兼容，坚持使用标准html
  - Css
    - 多个样式表合并
    - 代码简洁
  - Javascript
    - Ajax
    - 友好的提示，少弹窗
    - Jquery库

Web应用概述

- 技能要求
  - 前端
    - Web前端
      - Html语言
      - Css
      - Javascript
      - Html5
    - 移动前端
      - android/objective c
      - 其他
  - 后台
    - Java
    - Servlet/jsp
    - 数据库
    - 其他

Java web应用

- 典型目录结构
- servlet/filter/listener
- request/response/session/application
- MVC模式
- 运行环境
- 常用技术

Java web应用

- Java web应用典型目录结构

  - css/images/js一般是存放静态资源
  - WEB-INF目录是必须滴
    - classes：类文件
    - lib：第三方库
    - web.xml：web应用配置文件

Java web应用

- Servlet
  - 接收请求，进行处理，输出结果
- Filter
  - 过滤器，在servlet接收请求之前/响应请求之后被容器调用
  - Filter chain 链式处理
- Listener
  - 特定事件发生时，被容器调用。例如
    - web应用启动或停止
    - session创建或销毁
    - ……

Java web应用

- Request
  - 获取客户端信息
- Response
  - 向客户端返回响应
- Session
  - 可识别特定的客户端
- Application
  - 一般是表示web应用本身

Java web应用

- MVC模式：流行的web应用设计模式
  - M：模型，业务建模
    - 领域驱动设计
    - 测试驱动开发
    - 面向接口编程
  - V：视图
    - 常用技术：jsp，freemarker，……
  - C：控制
    - 控制层应该是很薄的一层，不应该有业务逻辑
      - 请求合法性校验
      - 参数验证
      - 请求参数，响应结果的封装
    - 常用技术：struts，springmvc，……

Java web应用

- 运行环境
  - Servlet容器
    - Tomcat
    - Jetty
    - ……

Java web应用

- 常用技术
  - spring
  - springmvc
  - mybatis
  - freemarker

Java web应用

- Spring
  - IOC
    - 基于注解自动装配：<context:component-scan base-package="com.meizu" />
  - AOP
  - 事务管理
  - 几个常用注解
    - @Compont
    - @Controller
    - @Repository
    - @Service
    - @Resource
    - @Transactional
    - @PostConstruct
    - @PreDestroy

Java web应用

- springmvc
  - web.xml配置

```
<servlet>
    <servlet-name>springmvc</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>classpath:config/spring-mvc.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>springmvc</servlet-name>
    <url-pattern>*.do</url-pattern>
</servlet-mapping>
```

  - 使用@Controller注解来声明action
  - 使用@RequestMapping注解来声明url
  - 使用@RequestParam注解来声明参数

Java web应用

- springmvc
  - 使用@ResponseBody来输出json
  - 输出json时的配置

```
<bean class="org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter">
    <property name="messageConverters">
        <list>
            <bean class="com.alibaba.fastjson.support.spring.FastJsonHttpMessageConverter">
                <property name="charset" value="UTF-8" />
                <property name="supportedMediaTypes">
                    <list><value>application/json; charset=UTF-8</value></list>
                </property>
                <property name="features">
                    <array>
                        <value>BrowserCompatible</value>
                        <value>WriteEnumUsingToString</value>
                        <value>DisableCircularReferenceDetect</value>
                    </array>
                </property>
            </bean>
        </list>
    </property>
</bean>
```

Java web应用

- springmvc
  - 返回网页时，使用freemarker模版

```
<bean id="viewResolver"
    class="org.springframework.web.servlet.view.freemarker.FreeMarkerViewResolver">
    <property name="viewClass" value="org.springframework.web.servlet.view.freemarker.FreeMarkerView"></property>
    <property name="cache" value="true" />
    <property name="order" value="1" />
    <property name="suffix" value=".ftl" />
    <property name="contentType" value="text/html;charset=UTF-8"></property>
</bean>

<bean id="freemarkerConfig" class="org.springframework.web.servlet.view.freemarker.FreeMarkerConfigurer">
    <property name="templateLoaderPath" value="/template/" />
    <property name="freemarkerSettings">
        <props>
            <prop key="default_encoding">UTF-8</prop>
            <prop key="number_format">0.##</prop>
            <prop key="datetime_format">yyyy-MM-dd HH:mm:ss</prop>
        </props>
    </property>
</bean>
```

Java web应用

- mybatis
  - 实体-关系映射
  - 定义接口
  - 编写sql语句
  - 接口方法和sql语句关联

Java web应用

- freemarker
  - 插值
  - ognl
  - 指令
  - 内置函数
  - ……

Java web应用

- 单元测试
  - Junit测试框架
    - 测试一个方法的输出是否符合预期
    - 使用断言而不是打印执行结果来判断测试的结果
    - Mock技术
  - Testng
    - 新一代的测试框架

学习资源

- 深入分析Java Web技术内幕
  - http://item.jd.com/11520670.html
- 领域驱动测试
  - http://item.jd.com/11423256.html
- Java测试新技术TestNG和高级概念
  - http://item.jd.com/10058667.html

- 前端技术
  - http://www.zhangxinxu.com/wordpress/

一个例子

- 一个简单例子，可以作为新项目的起始模版
  - 应用的搭建
  - Spring，springmvc，mybatis，freemarker整合
  - 单元测试

web应用开发

附录

- Git和jenkins平台
  - 管理员已经为新兵训练营创建了项目的仓库（前提）
  - 从git仓库下载项目工程，这时下载的应该是个空的工程
  - 在空的工程里创建自己的目录，及项目文件
  - 提交到git仓库，邀请有权限的成员进行评审和代码合入
  - 在jenkins平台创建job，构建已合入的工程
  - 构建成功
    - 在artifactory下载构建好的组件
    - 在sonar查看单元测试覆盖率和代码规范遵循度

The end

谢谢