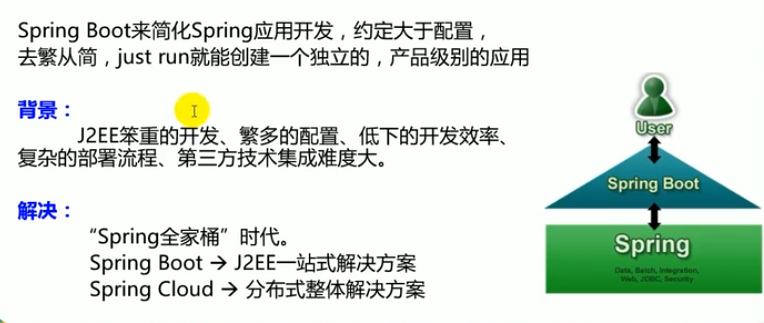
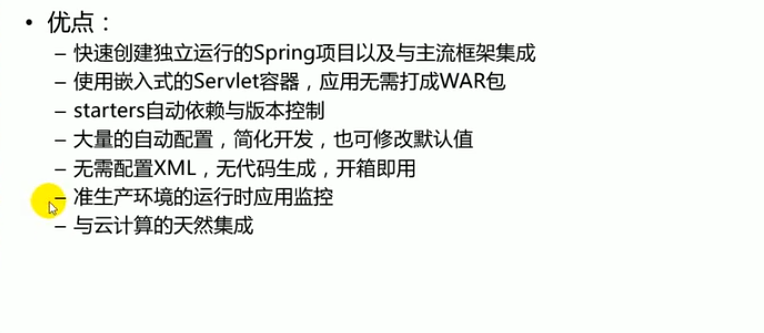
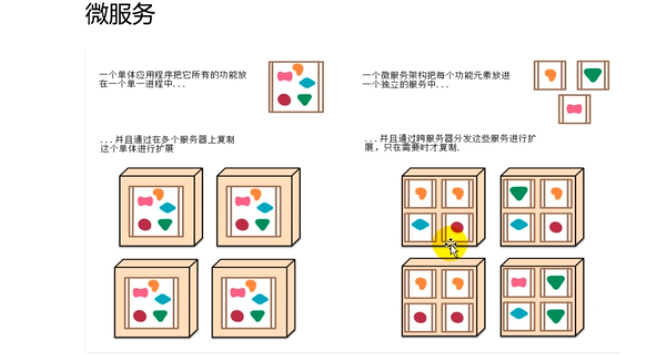
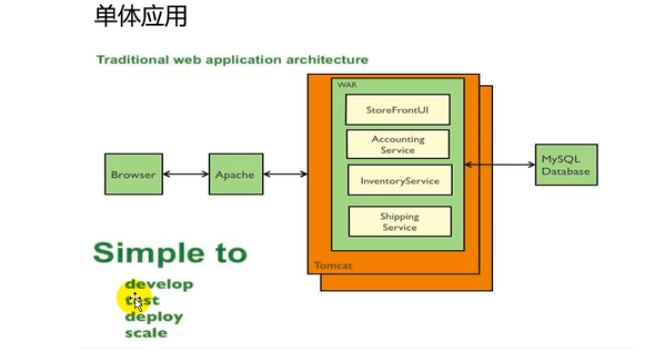
# Spring boot知识点

## 简介

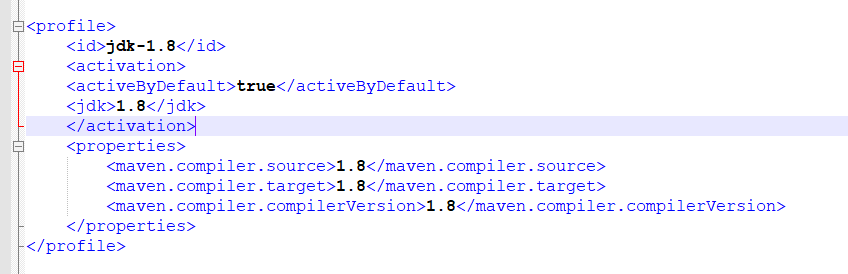


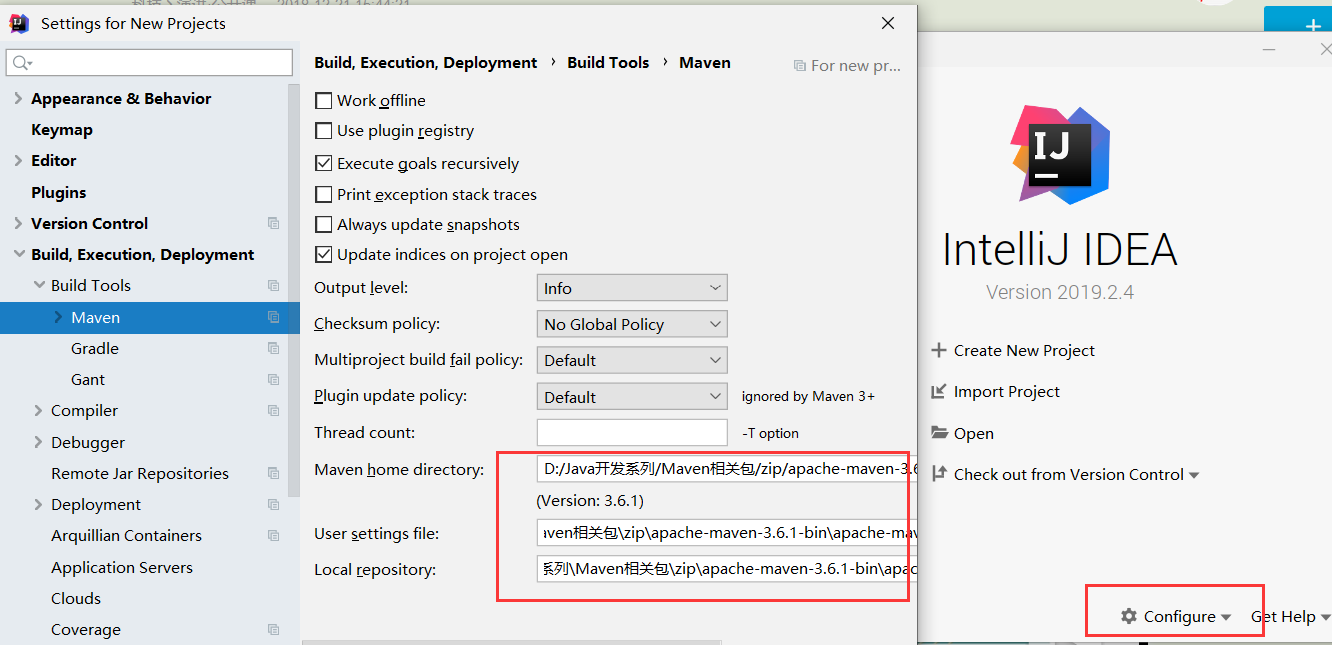




## 开发环境

在maven的setting配置文件profiles中添加





## Spring boot 创建hello word

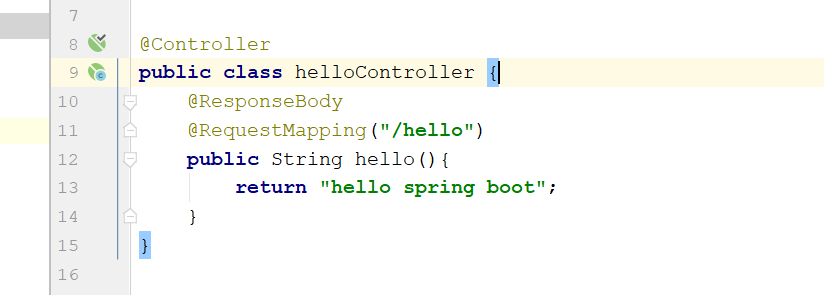
### 创建一个maven工程

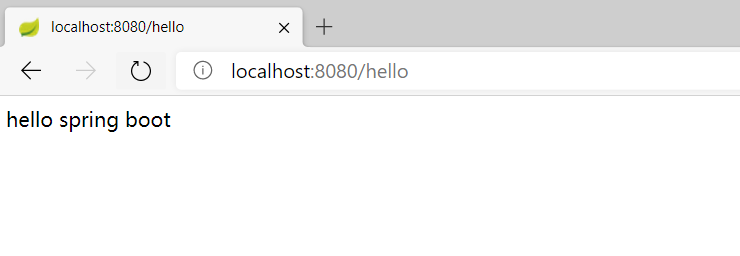
### 导入Spring boot相关依赖

<**parent**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-parent</**artifactId**>  
 <**version**>2.1.0.RELEASE</**version**>  
 </**parent**>  
<**dependencies**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-web</**artifactId**>  
 </**dependency**>  
</**dependencies**>

### 编写一个主程序：启动spring boot应用





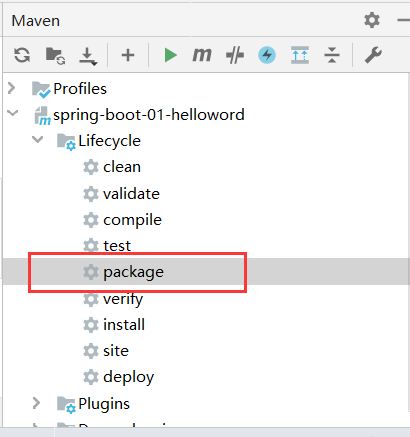


### 简化部署

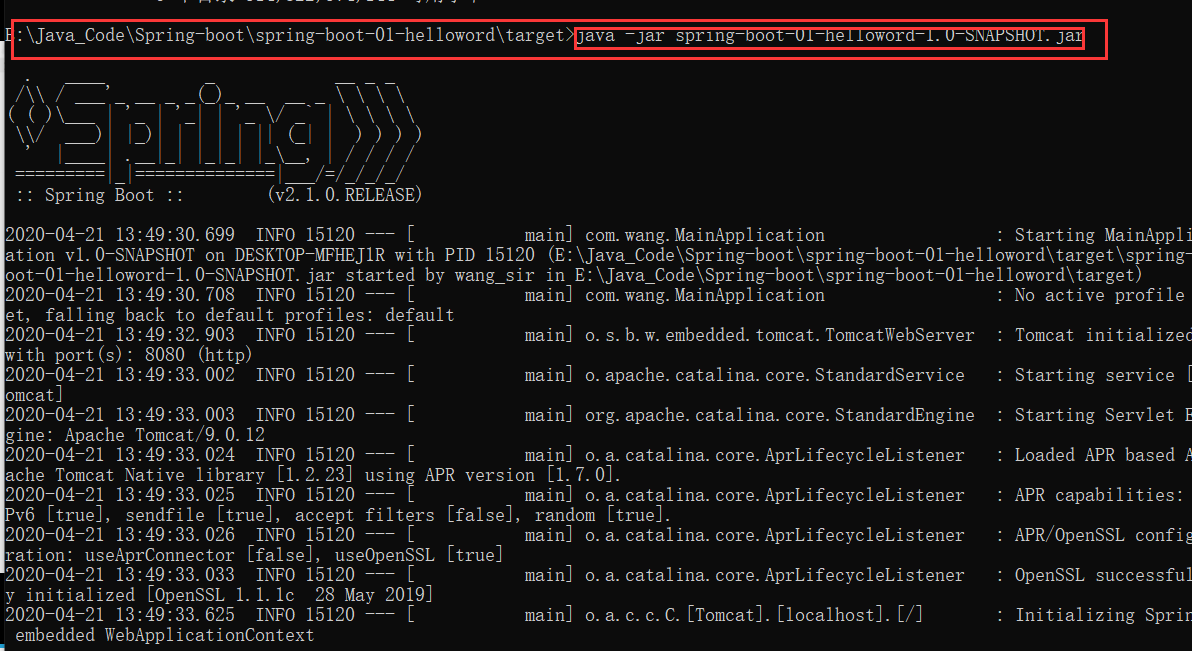
在pom.xml中导入spring boot 插件

*<!--简化spring boot部署-->*<**build**>  
 <**plugins**>  
 <**plugin**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-maven-plugin</**artifactId**>  
 <**configuration**>  
 *<!-- 没有该配置，devtools 不生效 -->* <**fork**>true</**fork**>  
 </**configuration**>  
 </**plugin**>  
 </**plugins**>  
</**build**>

### 打包程序



### 执行程序



## Hello Word探究

### POM文件

#### 父项目

<**parent**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-parent</**artifactId**>  
 <**version**>2.1.0.RELEASE</**version**>  
</**parent**>

此父项目的依赖为：

<**parent**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-dependencies</**artifactId**>  
 <**version**>2.1.0.RELEASE</**version**>  
 <**relativePath**>../../spring-boot-dependencies</**relativePath**>  
</**parent**>

它是真正管理spring boot应用里面的所有依赖版本；相当于Spring Boot的版本仲裁中心，以后我们导入依赖默认是不用写版本号

#### 启动器

<**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-web</**artifactId**>  
</**dependency**>



### 主程序类，主入口类

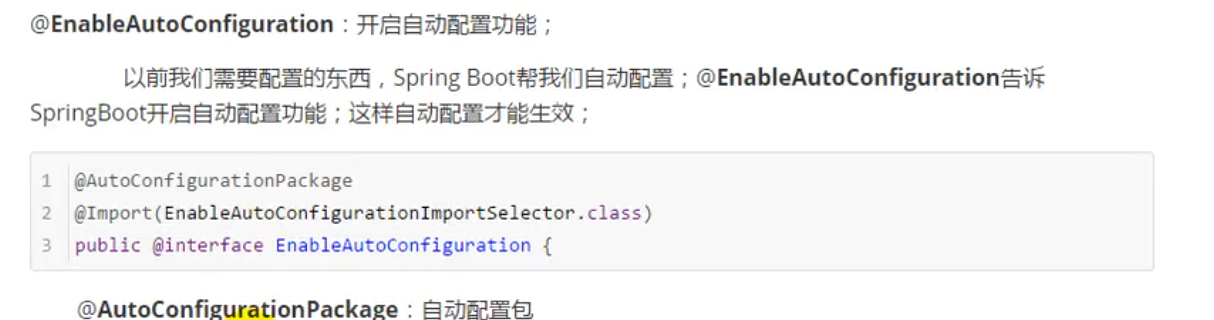
***/\*\*  
 \* @SpringBootApplication  
 \* 标注一个主程序类，说明这是一个springboot项目  
 \*/***@SpringBootApplication  
**public class** MainApplication {  
 **public static void** main(String[] args) {  
 *//spring应用启动器* SpringApplication.*run*(MainApplication.**class**, args);  
 }  
}

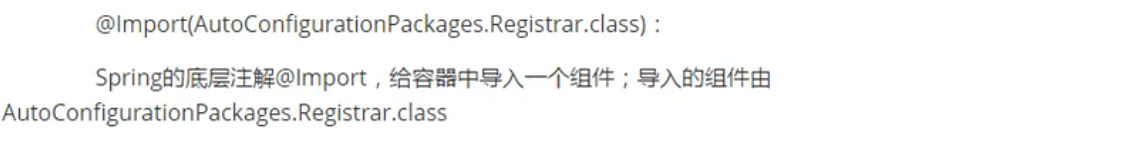
***@SpringBootApplication：spring boot应用标注在某个类上说明这个类是spring boot的主配置类，spring boot就应该运行这个类的main***

***方法来启动spring boot应用。***

@Target(ElementType.***TYPE***)  
@Retention(RetentionPolicy.***RUNTIME***)  
@Documented  
@Inherited  
@SpringBootConfiguration  
@EnableAutoConfiguration  
@ComponentScan(excludeFilters = {  
 @Filter(type = FilterType.***CUSTOM***, classes = TypeExcludeFilter.**class**),  
 @Filter(type = FilterType.***CUSTOM***, classes = AutoConfigurationExcludeFilter.**class**) })  
**public** @**interface** SpringBootApplication {

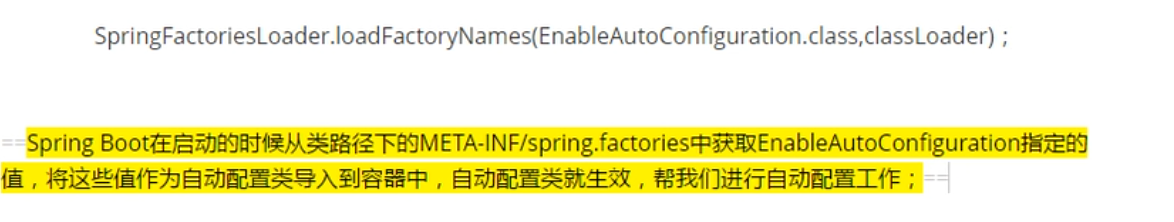
@SpringBootConfiguration：spring boot的配置类

标注在那个类上，就表明这个类是Spring Boot的配置类









## 快速创建spring boot

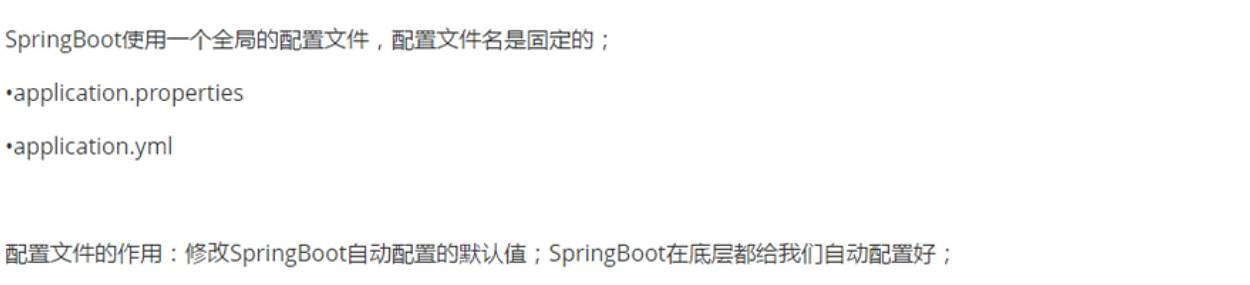
使用Spring Initializr



## Spring Boot配置

### 配置文件







### YAML语法

#### 基本语法

K:（空格）V 表示一对键值对（空格必须有）

以空格的缩进来表示控制层级关系：只要是左对齐的一列数据，都是同一个层级的。

例如：

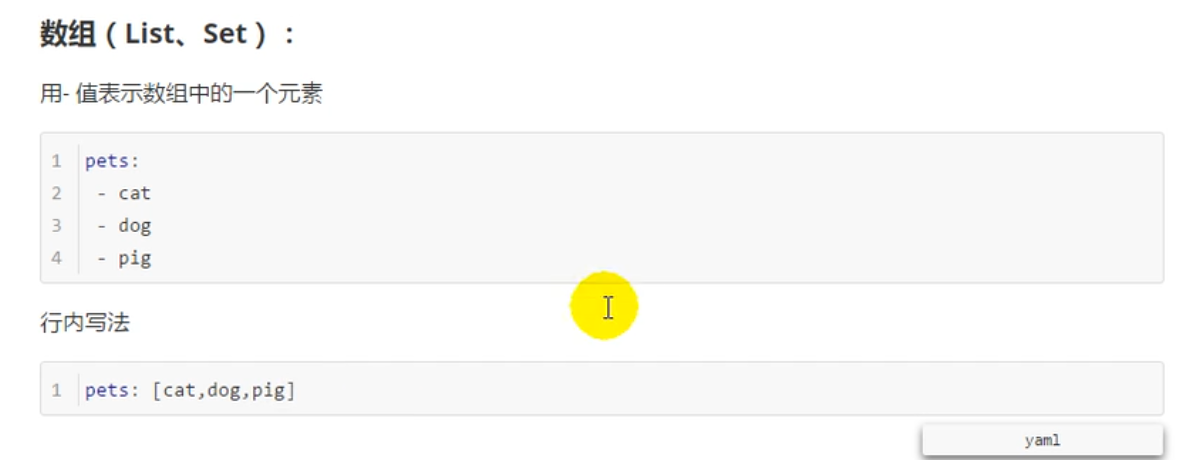
**server**:  
 **port**: 8081  
 **path**: /hello

属性和值也是大小写敏感。

#### 值的写法







### 配置文件值的注入







#### @PropertySource&@ImportResource









### 配置文件的占位符



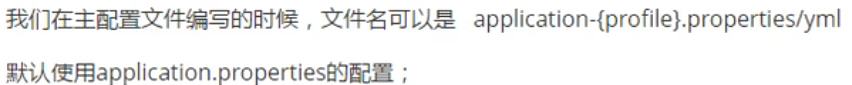




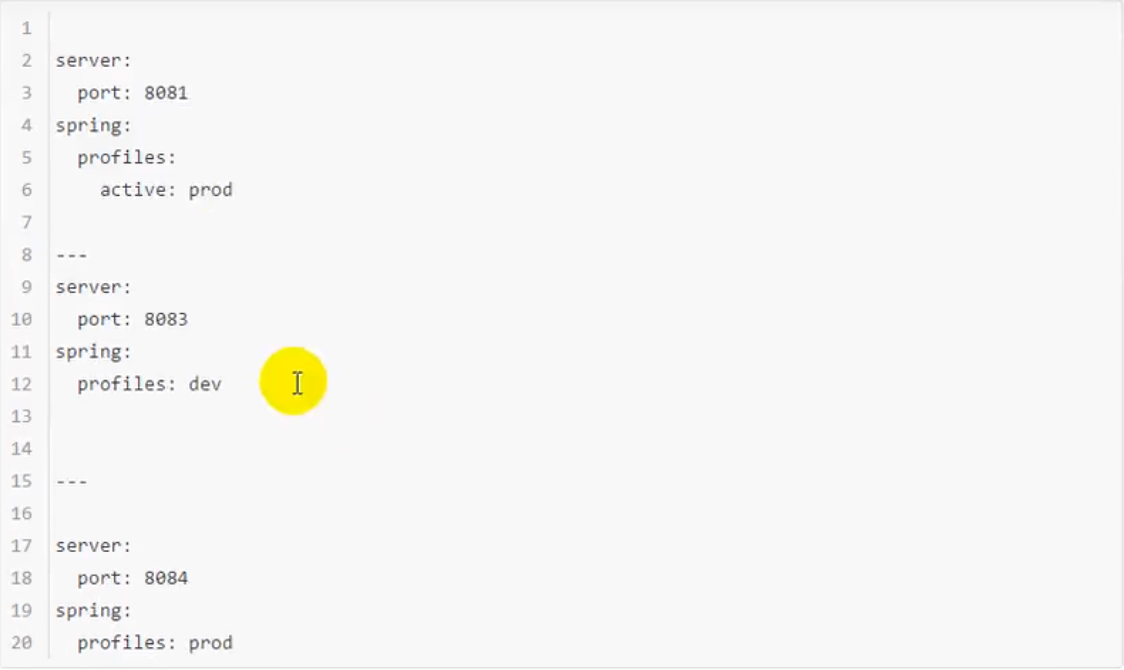
### Profile



#### 多profile文件



#### YML支持多文档方式



#### 激活指定profile



### 配置文件加载位置

