Spring Boot和Spring Cloud

1、什么是Srping Boot

SpringBoot是一款基于JAVA的开源框架。目的是为了简化Spring应用搭建和开发流程。是目前比较流行，大中小型企业常用的框架。正因为极大简化了开发流程，才收到绝大开发人员的喜爱。

SpringBoot核心原理是自动装配（自动配置）。

在这之前，开发一个JavaWeb，Spring等项目要进行很多配置，使用了SpringBoot就不用在过多考虑这些方面。

并且在SpringBoot中还内置了Tomcat。

官方介绍：

```
https://spring.io/projects/spring-boot
```

2、什么是Spring Cloud

Spring Cloud是一系列框架的有序集合。

它利用Spring Boot的开发便利性巧妙地简化了分布式系统基础设施的开发，如服务发现注册、配置中心、消息总线、负载均衡、断路器、数据监控等，都可以用Spring Boot的开发风格做到一键启动和部署。

Spring Cloud并没有重复制造轮子，它只是将各家公司开发的比较成熟、经得起实际考验的服务框架组合起来，通过Spring Boot风格进行再封装屏蔽掉了复杂的配置和实现原理，最终给开发者留出了一套简单易懂、易部署和易维护的分布式系统开发工具包。

大家也听过微服务吧。那它和Spring Cloud什么关系呢?

首先，什么是微服务?

微服务（英语：Microservices）是一种软件架构风格，它是以专注于单一责任与功能的小型功能区块 (Small Building Blocks) 为基础，利用模块化的方式组合出复杂的大型应用程序，各功能区块使用与语言无关的API集相互通信。

简单来说，微服务就是将一个大型的应用拆分成很多个小的应用，这些应用之间一般通过http协议进行通信，并且能够各自进行独立部署以及伸缩。由于微服务独立部署，可伸缩的特性，它能够迅速地大规模部署到云服务器上。

而使用Spring Cloud能够快速实现微服务架构。

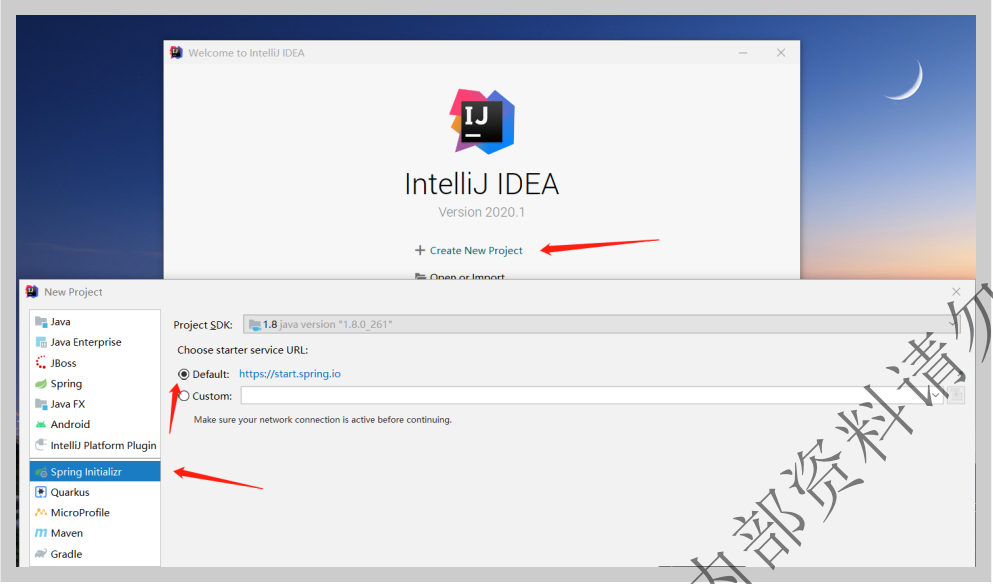
官方介绍：

```
https://spring.io/projects/spring-cloud
```

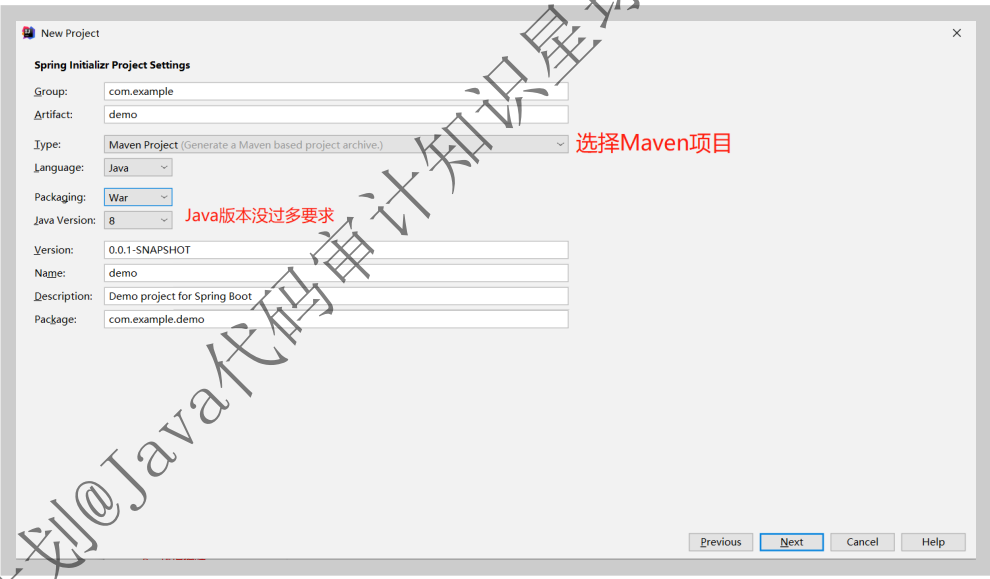
3、Spring Boot之HelloWorld

通过经典HelloWorld程序，来看看Springboot项目搭建多么简便。

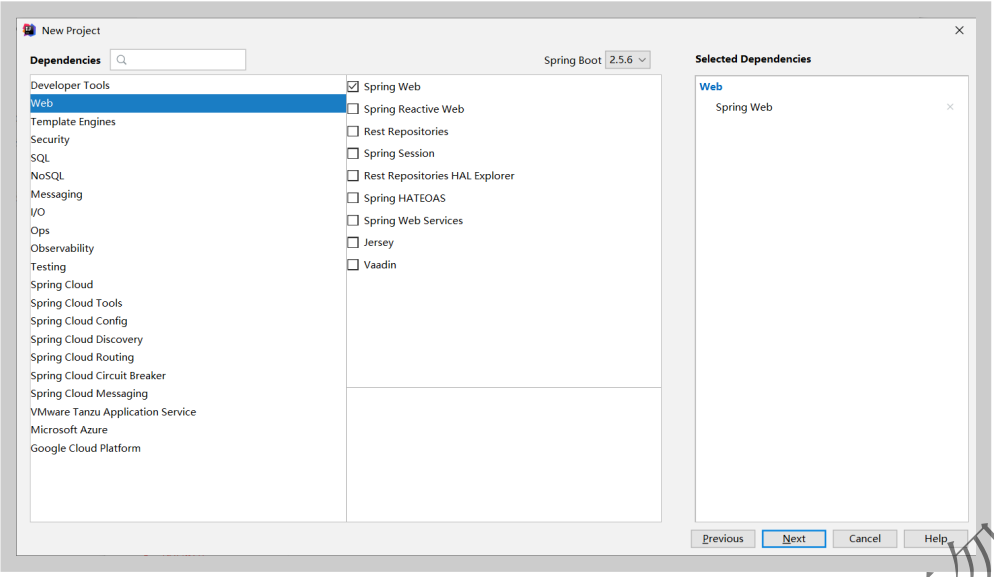
①、打开IDEA，选择 `Create New Project` ，选择 `Spring Initializer` ，右侧勾选 `Default` ，如下图所示：



②、点击Next， `Srping Initializr Project Settings` 配置内容默认就好，我们不做实际项目开发，如下图所示：



③、点击Next，进入依赖项选择页面，我们选择 `Web -> Spring Web` 这一个即可，如下图所示：

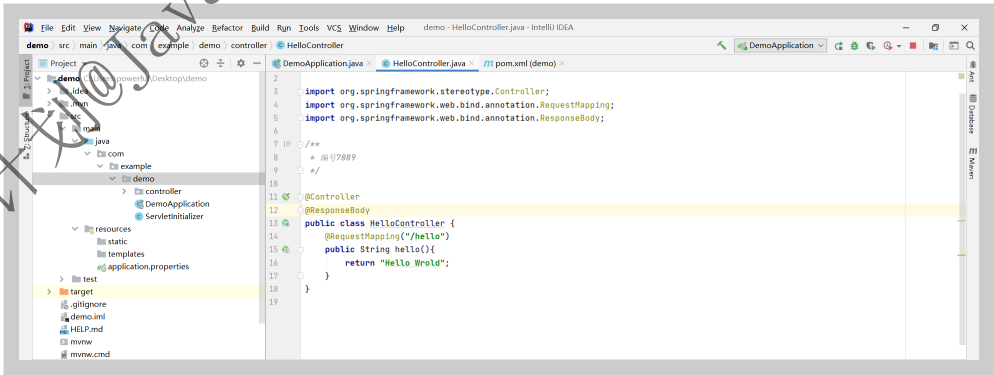


④、点击Next，填写项目名称和存放地址。练习项目，默认就可以，点击 `Finish` 完成创建。

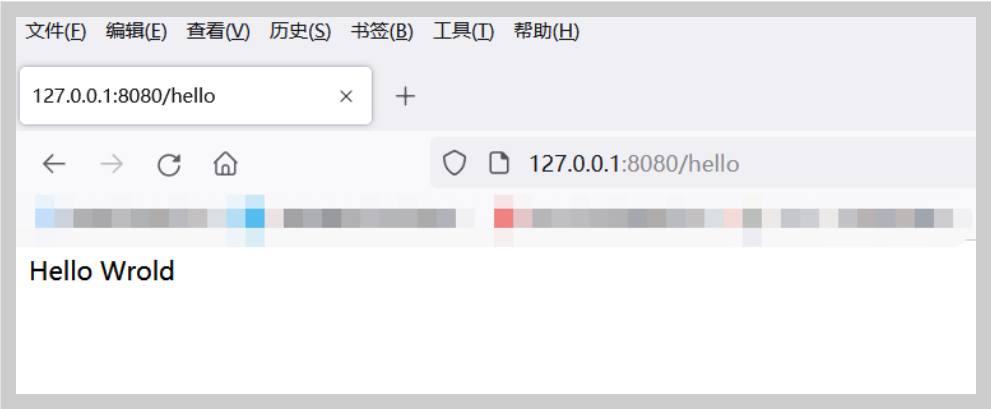
⑤、Maven自动加载完所需依赖后，整体项目结构如下图所示：

`@SpringBootApplication` 注解表示这个类为SpringBoot的主配置类，SpringBoot项目应运行这个类下面的main方法来启动SpringBoot应用。

⑥、创建 `HelloController` ，创建一个 `controller` 包，下面创建一个HelloController，在该controller中编写代码，如下图所示：



⑦、点击右上方运行，打开浏览器输入 `http://127.0.0.1:8080/hello` ，即可看到helloworld，如下图所示：



其他一些注解含义：

`@Controller` 注解：标注该类为controller类，可以处理http请求。@Controller一般要配合模版来使用。现在项目大多是前后端分离，后端处理请求，然后返回JSON格式数据即可，这样也就不需要模板了。

`@ResponseBody` 注解：将该注解写在类的外面，表示这个类所有方法的返回的数据直接给浏览器。 `@RestController 相当于 @ResponseBody 加上 @Controller`

`@RequestMapping` 注解：配置 `URL映射` ，可以作用于某个Controller类上，也可以作用于某Controller类下的具体方法中，说白了就是URL中请求路径会直接映射到具体方法中执行代码逻辑。

`@PathVariable` 注解：接受请求URL路径中占位符的值，示例代码如下图所示：

```
@Controller
@ResponseBody
@RequestMapping("/hello")
public class HelloController {
    @RequestMapping("/whoami/{name}/{sex}")
    public String  Hello(@PathVariable("name") String name,
@PathVariable("sex") String sex){
        return "Hello" + name + sex;
    }
}
```

`@RequestParam` 注解：将请求参数绑定到你控制器的方法参数上（是springmvc中接收普通参数的注解），常用于POST请求处理表单。

综上演示，可以看到Spring Boot部署非常方便，并且在开发后端服务时也极大简化了各种配置，可以更专注于编写代码。这对于我们审计Spring Boot架构的系统也极为的便利。