1. **springboot简单介绍**

Sping开源下的子项目，是spring组件一站式解决方案，降低了sping开发难度，简化了配置，提供了很多的 启动器，便于程序员开发

1. **什么是spring boot starters**

Starter可以理解为启动器，它包含了一系列可以继承到项目的依赖包，一站式集成spring组件和其他三方技术，首先他会提供一个自动化配置类，一般命名为xxxAutoConfiguration。然后通过条件注解@Conditional决定是否生效，同时还会提供一系列的默认配置，因此对于很多第三方框架，我们只要引入依赖就可以使用

SpringBoot官方启动器都是以Sping-Boot-Starter-命名的

1. **配置文件相关**

核心配置文件：application和bootstrap，前者用于springboot项目自动化配置，后者一般结合springCloud使用

Bootstrap由父ApplicationContext加载，通常要比application优先加载，且属性不能被覆盖

1. **介绍下核心配置注解@SpringBootApplication（自动配置原理）**

它包含了@SpringBootConfiguartion（组合@Configuration实现配置文件功能）。@EnableAutoConfiguration（将spring.factories中的自动配置类通过importSelector收集器收集，再加载到SpringFactoresLoader加载组件工厂，最后实例化加入到容器中，允许SpringBoot自动配置注解，开启后能根据当前类路径下的类包配置Spring bean）。@ConponentScan（开启组件扫描，自动扫描包路径下的@Component注解并注册bean到context中）

1. **SpingBoot注入依赖**

自动将一些配置的bean注册进ioc容器中，使用到的注解有@autowired或者@resource等

1. **开发RESTful web服务常用的注解有哪些**

RESTful风格看url和http请求方法就知道针对资源干什么，看status code知道结果如何

http协议的四种传参方式

|  |  |  |  |
| --- | --- | --- | --- |
| GET /api/dogs/{id} | 获取一个狗 | Path /books/12 | @PathVariable |
| GET /api/dogs | 获取所有狗 | Url /books?id=12 | @RequestParam |
| POST /api/dogs | 添加一个狗 | Body 传参 | @RequestBody |
| PUT /api/dogs/{id} | 修改一个狗 | Headers 传参 | @RequestHeader |
| DELETE /api/dogs/{id} | 删除一个狗 | PATCH 修改哪个 |  |

@RequestBody 修饰请求参数，注解用于接收HTTP的body，默认使用JSON格式，可用于复杂数据的传参

@ResponseBody 修饰返回值，说明携带响应数据，加上这个注解就说明为一个数据接口，不会走视图解析

@RequestMapping 标注服务端点，value：请求地址，method：请求方法等

也有@GetMapping和@PostMapping，@PutMapping等制定指定方法的注解

@RestController=@Controller+@ResponseBody

1. **读取配置文件的几种方式**

@Value(value=”${...}”)+@PropertySource(“classpath:..properties”)

也可以通过环境变量.getEnvironment.getProperty(“属性名”)

也可以使用 @ConfigurationProperties

1. **什么是JavaConfig，如何使用xml配置**

是spring社区的一个产品，提供了SpringIOC容器纯java方式，避免使用xml方式配置

1. **支持哪些嵌入式web容器**

支持使用jetty、netty、tomcat、undertow作为嵌入式web容器

1. **如何监控系统实际运行状况**

可以通过springboot actuator，监视器可以访问生产环境中正在运行的项目的当前状况，监视器提供了一组直接http url访问rest端点来检查状态

1. **如何做请求参数校验**

Web模块中有hibernate-validation使用注解@Validated+@Valid注解在控制层进行参数校验（请求参数），请求体使用@Valid，在传递过来的实体类上添加各种校验注解如@Null，@Email，@Range等。在server层使用@Validated+@Valid组合注解来实现。还可以自定义注解，然后编写校验类实现ConstraintValidator接口，重写方法

@NotNull和@Column（nullable=false）的区别？前者是jsr提供的不为null，后者声明列为非空的方法

1. **异常的常用处理方式**

使用@ControllerAdvice和ExceptionHandler处理全局异常，在异常处理类上添加@ControllerAdvice就可以使之变为全局异常处理类，也可以指定处理特定的Controller类。而@ExceptionHandler注解在异常处理类的方法上拦截指定的异常。还有可以使用@ResponseStatus简单处理异常，将异常映射为状态码

1. **如何实现定时任务**

创建一个scheduled task，使用@Scheduled注解就可以简单的实现。配合上Cron表达式完成复杂操作。在启动类上添加@EnableScheduling注解就可以启动定时任务。还有可以自定义线程池执行scheduled task，实现SchedulingConfigurer接口重写task方法就行

SpringBoot文档合集

https://mp.weixin.qq.com/mp/appmsgalbum?\_\_biz=Mzg2OTA0Njk0OA==&action=getalbum&album\_id=1322577180722872320&scene=173&from\_msgid=2247485563&from\_itemidx=1&count=3&nolastread=1#wechat\_redirect

# 常用注解

@Service用于标注业务层组件  
@Controller用于标注控制层组件  
@Repository用于标注数据库访问Dao组件  
@Component泛指组件，当组件不好归类的时候，我们可以使用这个注解进行标注  
@Autowired，自动注入，自动从spring的上下文找到合适的bean来注入  
@RestController，Spring4之后新加入的注解，原来返回json需要@ResponseBody和@Controller配合，将调用的结果直接返回给调用者。  
@Value：注入Spring boot [application.properties](http://application.properties" \t "_blank)配置的属性的值。  
@RequestMapping：提供路由信息，负责URL到Controller中的具体函数的映射。@RequestMapping(“/path”)表示该控制器处理所有“/path”的UR L请求。RequestMapping是一个用来处理请求地址映射的注解，可用于类或方法上。  
@GetMapping是一个组合注解，是@RequestMapping(method = RequestMethod.GET)的缩写。该注解将HTTP Get 映射到 特定的处理方法上。  
同理PostMapping也是一个组合注解，是@RequestMapping(method = [RequestMethod.POST](http://RequestMethod.POST" \t "_blank))的缩写。  
@PathVariable:获取url中的数据。  
@ComponentScan 组件扫描，发现和组装一些Bean。  
@EnableAutoConfiguration自动配置。  
@SpringBootApplication：申明让spring boot自动给程序进行必要的配置，这个配置等同于：@Configuration ，@EnableAutoConfiguration 和 @ComponentScan 三个配置。  
@Data 自动生成setter、getter方法  
@Import：用来导入其他配置类。  
@ImportResource：用来加载xml配置文件。  
@Bean:放在方法的上面，而不是类，意思是产生一个bean,并交给spring管理。  
@Inject：等价于默认的@Autowired，只是没有required属性；