# springboot作用

微服务框架

简化xml配置（mybatis的sql文件除外），提高开发效率

# pom.xml

**<parent>**

**<groupId>org.springframework.boot</groupId>**

**<artifactId>spring-boot-starter-parent</artifactId>**

**<version>1.3.3.RELEASE</version>**

**</parent>**

**spring-boot-starter-parent**

提供dependency management,也就是说依赖管理，引入以后在申明其它dependency的时候就不需要version（即管理springboot所需要的依赖，统一各种jar的版本号）

<!-- SpringBoot web 组件 -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

**spring-boot-starter-web**

springweb 核心组件（自动装配tomcat，自动从application.properties中读取web应用的配置）

# springboot注解

@RestController

在Controller类上加上@RestController 表示修饰该Controller所有的方法返回JSON格式,直接可以编写Restful接口

@RestController===Controller的每个方法加上@ResponseBody

@EnableAutoConfiguration

让Spring Boot 根据应用所声明的依赖来对 Spring 框架进行自动配置。如：spring-boot-starter-web添加了Tomcat和Spring MVC，所以auto-configuration将假定你正在开发一个web应用并相应地对Spring进行设置

@ControllerAdvice

是 controller 的一个辅助类，最常用的就是作为全局异常处理的切面类

# springboot启动方式

①启动方式一

Controller类写main方法入口，使用注解@**EnableAutoConfiguration启动**

**@RestController**

**@EnableAutoConfiguration**

**public class HelloController {**

**@RequestMapping("/hello")**

**public String index() {**

**return "Hello World";**

**}**

**public static void main(String[] args) {**

**SpringApplication.*run*(HelloController.class, args);**

**}**

**}**

②启动方式二

单独一个启动类，要扫描Controller包

@ComponentScan(basePackages = "Controller类扫描包")

@EnableAutoConfiguration

**public** **class** App {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(App.**class**, args);

}

}

# 静态资源访问

开发Web应用的时候，需要引用大量的js、css、图片等静态资源

Spring Boot默认提供静态资源目录位置需置于classpath下，目录名需符合如下规则：

/static

/public

/resources

/META-INF/resources

举例：我们可以在src/main/resources/目录下创建static，在该位置放置一个图片文件。启动程序后，尝试访问http://localhost:8080/D.jpg

# 全局捕获异常

@ControllerAdvice

**public** **class** GlobalExceptionHandler {

@ExceptionHandler(RuntimeException.**class**)

@ResponseBody

**public** Map<String, Object> exceptionHandler() {

Map<String, Object> map = **new** HashMap<String, Object>();

map.put("errorCode", "101");

map.put("errorMsg", "系統错误!");

**return** map;

}

}

# [使用@Scheduled创建定时任务](http://blog.didispace.com/springbootscheduled/" \t "_blank)

①在Spring Boot的主类中加入@EnableScheduling注解，启用定时任务的配置

②定时任务类

@Component

public class ScheduledTasks {

private static final SimpleDateFormat dateFormat = new SimpleDateFormat("HH:mm:ss");

@Scheduled(fixedRate = 5000)

public void reportCurrentTime() {

System.out.println("现在时间：" + dateFormat.format(new Date()));

}

}