## IDEA创建Spring Boot项目

file->new project->选择Spring Initializr，配置jdk1.8->选择maven project，并创建名称->选择spring boot版本，选择web项目

## 启动spring boot项目方法

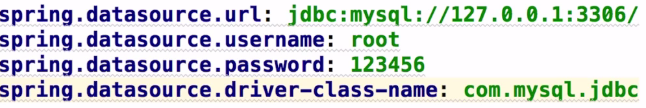
1 在包下会默认生成一个Application类，该类中有@SpringBootApplication，运行这个类就可以启动整个项目

2 从命令行中进入项目girl目录下，命令mvn spring-boot:run

3 在girl目录下运行mvn install（将当前目录下的程序打包成jar包）

然后进入target目录下，运行java -jar girl-0.0.1-SNAPSHOT.jar

## 属性配置



在application.properties中配置

**server.port**=**8081  
server.context-path**=**/girl**

访问的时候使用8081端口，而且必须加/girl

<http://localhost:8081/girl/hello>

（默认访问的时候使用<http://localhost:8081/hello>）

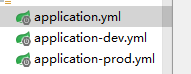
默认使用的application.properties文件，建议使用application.yml文件，语法更简洁

**server:  
 port:** 8082  
 **context-path:** /girl

通过@Value注解可以将配置文件中内容注入到属性中

也可以创建一个实体类，对应配置文件中的配置，该实体类需要加@ConfigurationProperties

不同环境下使用不同的注解：

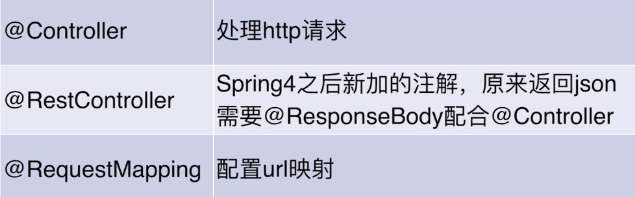


默认加载application.yml文件，application.yml文件内容：

**spring:  
 profiles:  
 active:** prod

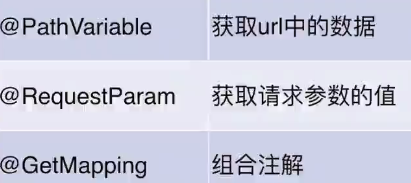
表示使用哪个配置文件

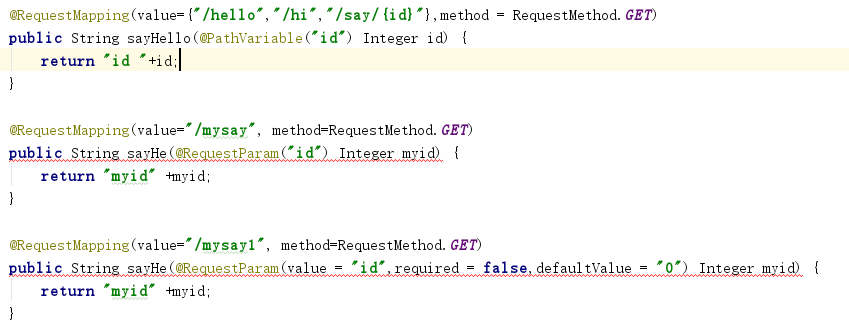
## Controller讲解



@RequestMapping可以配置在类上或者方法上

如果配置在类上，那么在访问该类下的每个方法时都需要加上对应的前缀

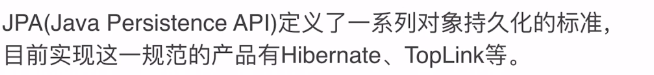




@RequestParam是获取url中?后面的参数值

@PathVariable是直接获取url中的数据/../{id}/..

## 数据库操作





Pom文件配置：

<**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-data-jpa</**artifactId**>  
</**dependency**>  
  
<**dependency**>  
 <**groupId**>mysql</**groupId**>  
 <**artifactId**>mysql-connector-java</**artifactId**>  
</**dependency**>

配置Application.yml文件：

**spring:  
 profiles:  
 active:** dev  
 **datasource:  
 driver-class-name:** com.mysql.jdbc.Driver  
 **url:** jdbc:msyql://127.0.0.1:3306/dbgirl  
 **username:** root  
 **data-password:** 0000  
 **jpa:  
 hibernate:  
 ddl-auto:** create  
 **show-sql:** true

定义一个GirlRepository接口，实现JpaRepository接口。

定义一个GirlController类

@RestController  
**public class** GirlController {  
  
 @Autowired  
 **private** GirlRepository **girlRepository**;  
  
 */\*\*  
 \* 查询女生列表  
 \** ***@return*** *\*/* @GetMapping(**"/girls"**)  
 **public** List<Girl> girlList() {  
 **return girlRepository**.findAll();  
  
 }  
  
 */\*\*  
 \* 添加一个女生  
 \** ***@param cupSize*** *\** ***@param age*** *\** ***@return*** *\*/* @PostMapping(**"/girls"**)  
 **public** Girl addGirl(@RequestParam(**"cupSize"**) String cupSize,  
 @RequestParam(**"age"**) Integer age){  
 Girl girl = **new** Girl();  
 girl.setAge(age);  
 girl.setCupSize(cupSize);  
 **return girlRepository**.save(girl);  
 }  
  
 */\*\*  
 \* 根据ID查询女生  
 \** ***@param id*** *\** ***@return*** *\*/* @GetMapping(**"/girls/{id}"**)  
 **public** Girl getGirlByID(@PathVariable(**"id"**)Integer id) {  
 **return girlRepository**.findOne(id);  
 }  
  
 */\*\*  
 \* 根据id更新一个女生  
 \** ***@param id*** *\** ***@param age*** *\** ***@param cupSize*** *\** ***@return*** *\*/* @PutMapping(**"/girls/{id}"**)  
 **public** Girl updateGirl(@PathVariable(**"id"**) Integer id,  
 @RequestParam(**"age"**) Integer age,  
 @RequestParam(**"cupSize"**) String cupSize) {  
 Girl girl = **new** Girl();  
 girl.setId(id);  
 girl.setCupSize(cupSize);  
 girl.setAge(age);  
  
 **return girlRepository**.save(girl);  
 }  
  
 */\*\*  
 \* 根据id删除一个女生  
 \** ***@param id*** *\*/* @DeleteMapping(**"/girls/{id}"**)  
 **public void** deleteGirl(@PathVariable(**"id"**)Integer id) {  
 **girlRepository**.delete(id);  
 }  
  
 @GetMapping(**"/girls/age/{age}"**)  
 **public** List<Girl> girlListByAge(@PathVariable(**"age"**) Integer age) {  
 **return girlRepository**.findByAge(age);  
 }  
}

## 事务管理