> 学习自http://www.spring4all.com/article/248

配置文件application.properties

自定义属性与加载

自定义

```
com.feng.blog.name=feng
com.feng.blog.title=Spring Boot教程
```

通过@Value("${属性名}")注解来加载对应的配置属性。

```
@Component
public class BlogProperties{
 @Value("${com.feng.blog.name}")
 private String name;
 @Value("${com.feng.blog.title}")
 private String title;

 //省略Getter和Setter
}
```

单元测试验证BolgProperties中的属性是否已经根据配置文件加载了

```
@Autowired
private BlogProperties blogProperties;

@Test
public void getHello() throws Exception {
 Assert.assertEquals(blogProperties.getName(), "feng");
 Assert.assertEquals(blogProperties.getTitle(), "Spring学习");
}
```

参数间的引用

在application.properties中的各个参数之间也可以相互引用来使用

```
com.feng.blog.name=feng
com.feng.blog.title=Spring学习
com.feng.blog.desc=来自${com.feng.blog.name}的${com.feng.blog.title}
```

单元测试验证BolgProperties中的属性是否已经根据配置文件加载了

```
Assert.assertEquals(blogProperties.getDesc(),"来
自"+blogProperties.getName()+"的"+blogProperties.getTitle());
```

使用随机数

Spring Boot的配置文件中可以通过${random)来产生int值、long值或String字符串，来支持属性的随机值。

```
#随机字符串
```

```
com。feng.blog.value=${random.value}
#随机int
com。feng.blog.number=${random.int}
#随机long
com。feng.blog.bignumber=${random.long}
#10以内的随机数
com。feng.blog.test1=${random.int(10)}
#10-20的随机数
 com。feng.blog.test2=${random.int[10,20]}
```

使用SpringApplication.setAddCommandLineProperties(false)来屏蔽通过命令行更改运行参数

多环境配置

在SpringBoot中多环境配置文件名需满足application-{profile}.properties的格式。其中{profile}对应环境标识。

```
application-dev.properties：开发环境
application-test.properties：测试环境
application-prod.properties：生产环境
```

自动配置

SpringBoot获取配置的优先级顺序

1.命令行参数

2.java:comp/env里的JNDi属性

3。JVM系统属性

4.操作系统环境变量

5.RandomValuePropertySource属性类生产的random.*属性

6.应用以外的application.properties(或yml)文件

7.打包在应用内的application.properties(或yml)文件

8.在应用 @Configuration 配置类中，用 @PropertySource 注解声明的属性文件

9..SpringApplication.setDefaultProperties 声明的默认属性