# 概述

配置文件的作用：修改SpringBoot自动配置的默认值；

SpringBoot在底层都给我们自动配置好了；

# 两种配置文件

有两种配置文件，文件名是固定的

* application.properties
* application.yml

## 区别

Yml：

**server:  
 port:** 80

properties：

**server.port**=**80**

# Yaml基本语法

## 基本语法

K**:**(**空格**)v**:** (**空格**)表示一对键值对（空格必须有）；

以**空格**的缩进来控制层级关系，只要是左对齐的一列数据都是同一层级的

例如

**server:  
 port:** 80  
 **path:** /hello

属性和值是大小写敏感的

## 值的写法

### 字面量

普通值（数字，字符串，布尔）

K: v: 值，字面量直接来写

字符串默认不用加上单引号或者双引号；

“”：双引号，不会转义字符串里面的特殊字符；特殊字符会作为本身想表示的意思

name: “zhangsan \n lisi” 输出zhangsan 换行 lisi

‘’：单引号，会转移特殊字符，特殊字符最终只是一个普通的字符串数据

### 对象，Map，直接键值对

K: v: 注意缩进

1. 第一种，多行写法

friends:

lastName: zhangsan

age: 20

1. 第二种，行内写法

friend: {lastName: zhangsan,age: 18}

### 数组或集合（list）

用-(空格)值表示数组中的一个元素

1. 第一种，多行写法

pets:

* cat
* dog
* pig

1. 第二种，行内写法

pets: [cat,dog,pat]

# 值获取

## 配置文件提示

导入一个依赖

*<!-- 导入配置文件处理器，配置文件绑定类后就会有提示 -->*<**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-configuration-processor</**artifactId**>  
 <**optional**>true</**optional**>  
</**dependency**>

## 在配置Bean上加入注解

*/\*\*  
 \** ***@ConfigurationProperties*** 告诉SpringBoot将本类中所有属性和配置文件中相关的配置进行绑定；  
 *\* prefix = "person"* 配置文件中那个下面的所有属性进行一一映射 *\*/*@Component@ConfigurationProperties(prefix = **"person"**)  
**public class** Person {

## 编写yml文件

**person:  
 name:** zhangsan  
 **age:** 18  
 **boss: false  
 maps:** {**k1:** v1,**k2:** v2}  
 **lists:** -1  
 -2  
 -3  
 **dog:  
 age:** 2  
 **name:** mimi

## 单元测试

@Autowired  
**private** Person **person**;  
@Test  
**public void** contextLoads() {  
 System.***out***.println(**person**);  
}

# 对应Properties的写法

**person.name**=**李白  
person.age**=**18  
person.boss**=**false  
person.maps.k1**=**v1  
person.maps.k2**=**v2  
person.lists**=**1,2,3,4  
person.dog.name**=**小狗  
person.dog.age**=**5**