1、1 关于SpringBoot配置文件的说明

1.1.1 properties说明

1.语法：k-v结构 key=value
2.数据类型：默认是String数据类型 不要添加多余的""引号
3.字符数据类型：properties的默认的加载的编码格式为ISO-8859-1 所以添加中文是需要字符转义
4.缺点：所有的key都必须手动的编译 没有办法复用 所以引入了yml配置

1.1.2 YML

1.语法 K-V结构 写法上 key:value 实质上key=value

key:value中间使用（:+空格）分隔
key与key之间有父子关系的，所以写的时候注意缩进
YML配置文件默认的都是UTF-8编码 所以可以直接编辑中文

1.2关于SpringMVC调用流程（了解）

1.知识扫盲

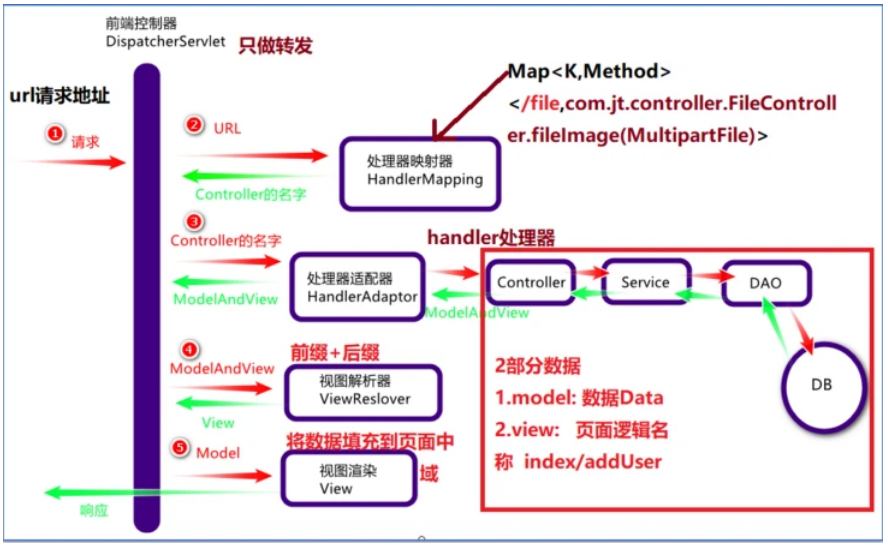
1.协议支持 http协议、https协议(OSI 7层网络模型 物数网传会表应)
2.java针对于服务器端开发了一种传输机制 Servlet机制(TCP-IP协议规范)用户使用servlet进行数据的传输的速度是较快的


2.SpringMVC调用流程

问题：用户发起请求/addUser是任何匹配到真实的业务方法

组件：

1.前端控制器 所有请求的中转站
2.处理器映射器 将用户的请求与执行的业务方法进行映射(绑定)
3.处理器适配器
4.视图解析器




1.2 关于配置文件赋值操作

1.2.1 入门案例

```
package com.jt.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
```

```
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

@RestController    //@ResponseBody 将返回值转化为json串使用 程序将不会执行视图解析器 直接返回
//@Controller         //String类型/moduleAndView
public class RedisController {

    private String host = "127.0.0.1";
    private Integer port = 6379;

    //如果使用RestController 返回值为String类型则返回字符串本身
    //如果返回的是一个对象 则结果必然是该对象的JSON数据.
    @RequestMapping("/getMsg")
    public String getMsg(){

        return host + ":" + port;
    }
}
```

1.2.2 @Value属性赋值操作

需求:有时对象中的属性的值可能会发生变化,如果直接写死到代码中可能导致耦合性高,能否利用配置文件方式动态为属性赋值

```
#配置redis节点信息
redis:
  host: 192.168.8.13
  port: 6666
```

```
 package com.jt.controller;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

@RestController        //@ResponseBody 将返回值转化为json串使用 程序将不会执行视图解析器 直接返回
//@Controller         //String类型/moduleAndView
public class RedisController {

    /**
     * 实现思路:
     *   如果可以从容器中获取数据的化,直接赋值给属性.则可以实现解耦
     * 如何实现:
     *      注解实现: @Value("${配置文件的key}")
     * 表达式:  spel 表达式
     */

    @Value("${redis.host}")
    private String host;
    @Value("${redis.port}")
    private Integer port;

    //如果使用RestController 返回值为String类型则返回字符串本身
    //如果返回的是一个对象 则结果必然是该对象的JSON数据.
    @RequestMapping("/getMsg")
    public String getMsg(){

        return host + ":" + port;
    }
}
```

1.2.3 利用properties文件为属性赋值

说明:由于YML配置文件一般都是配置第三方的整合信息，如果将业务的数据添加到YML中则不规范,最好将业务的操作添加到porperties文件中

1、创建redis.properties文件

```
#key必须唯一
redis.pro.host=直接了当
redis.pro.port=3333
```

2、

```
package com.jt.controller;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.PropertySource;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
@RestController
//需要通过Spring容器加载配置文件,并且以utf-8的格式进行加载
@PropertySource(value="classpath:/properties/redis.properties",encoding = "utf-8")
public class RedisProperties {
    @Value("${redis.pro.host}")
    private String proHost;

    @Value("${redis.pro.port}")
    private Integer proPort;

    @RequestMapping("/getMsg")
    public String getMsg2(){
        return proHost+" : "+proPort;
    }
}
```

1.3 SpringBoot环境切换问题

1.3.1 业务需求

业务场景:

员工是外包人员,经常性的需要往返 公司和甲方,进行代码调试时由于位置不同所以服务器IP地址必然不同,如果每次换环境都必须重新编辑IP地址和端口数据,必定繁琐能否优化?

1.3.2 业务实现-指定多个环境

注意事项:无论是什么样的环境,配置的个数都是相同的,只有值不同

```
#该配置文件，当Spring容器启动时加载 active加载的环境
spring:
  profiles:
    active: dev
#如果需要多环境配置则需要将YML环境分割---
---
#定义开发的环境
spring:
  profiles: dev
server:
  port: 8060
#配置redis节点信息
redis:
  host: 192.168.8.13
  port: 3333
#如果需要多环境配置则需要将YML环境分割---
---
spring:
  profiles: prod
server:
  port: 8080
#配置redis节点信息
redis:
```

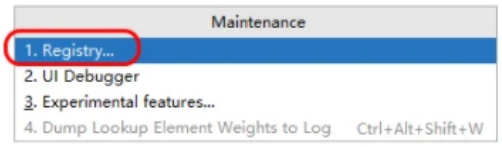
```
  host: 192.168.8.13
  port: 6666
```

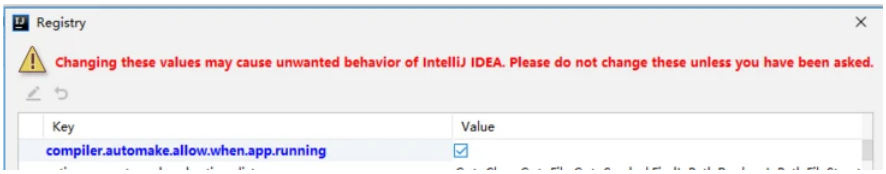
1.4 添加热部署配置

```
<!--添加热部署依赖-->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <scope>runtime</scope>
</dependency>
```

2、配置热部署

组合键:Ctrl+Shift+Alt+/




开启自动编译

1.5.4 关于SQL连接说明

1.serverTimezone=GMT%2B8 %2B代表"+"号 表示时区

2.&useUnicode=true&characterEncoding=utf8 指定编码为utf-8

3.&autoReconnect=true& 如果程序连接数据库中途断掉时是否重连

4.allowMultiQueries=true 是否允许批量操作

1.5.5 编辑YML配置文件

```
server:
  port: 80
 servlet:
    context-path: /
spring:
  datasource:
    #驱动版本问题 高版本需要添加cj关键字   一般可以省略
 #driver-class-name: com.mysql.cj.jdbc.Driver
 url: jdbc:mysql://127.0.0.1:3306/jtdb?
serverTimezone=GMT%2B8&useUnicode=true&characterEncoding=utf8&autoReconnect=true&allowMultiQueries=true

    username: root
    password: tarena
mybatis:
  #别名包定义 Mapper的resultType中只需要写类名 之后自动拼接
 type-aliases-package: com.jt.pojo
  #加载指定的XML映射文件
 mapper-locations: classpath:/mtbatis/mappers/*.xml
  #开启驼峰映射
 configuration:
    map-underscore-to-camel-case: true
```