1. 在配置文件[servlet-name]-servlet.xml中

<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xmlns:websocket="http://www.springframework.org/schema/websocket"  
 xmlns:mvc="http://www.springframework.org/schema/mvc"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.1.xsd  
 http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-4.1.xsd  
 http://www.springframework.org/schema/websocket http://www.springframework.org/schema/websocket/spring-websocket-4.1.xsd" >

xsi:schemaLocation中，两两对应

<http://www.springframework.org/schema/websocket>

与<http://www.springframework.org/schema/websocket/spring-websocket-4.1.xsd> 对应

1. 问题处理

* java.lang.IllegalArgumentException: 'dataSource' or 'jdbcTemplate' is required
* 没有找到dataSource 或者 jdbcTemplate bean，可能配置文件出错

1. Spring boot 默认使用的是logback，通过在classpath中添加一个logback.xml可以配置

下面的例子将spring的日志信息调整到WARN级别

<?xml version="1.0" encoding="UTF-8"?>  
<configuration>  
 <include resource="org/springframework/boot/logging/logback/base.xml"/>  
 <logger name="org.springframework.web" level="WARN"/>  
</configuration>

如果想使用log4j，可以参考<http://www.dutycode.com/spring_boot_log4j_shiyong.html>

1. @Autowire 可以用 @Resource(name=””)代替。配置不好，autowire可能会出现各种错误
2. Dispatcher 的url-pattern 配置一定要注意，写成 /, 否则会不止匹配一次

<servlet-mapping>  
 <servlet-name>dispatcherXML</servlet-name>  
 <url-pattern>/</url-pattern>  
</servlet-mapping>

1. 中文乱码问题

参考网址：http://josh-persistence.iteye.com/blog/2085015

1. jsp中解决方法

<%@ **page** contentType="text/html;charset=UTF-8" language="java" pageEncoding="utf-8" %>

1. 前后端数据传送过程中，配置filter

<filter>  
 <filter-name>encoding</filter-name>  
 <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>  
 <init-param>  
 <param-name>encoding</param-name>  
 <param-value>utf-8</param-value>  
 </init-param>  
 <init-param>  
 <param-name>forceEncoding</param-name>  
 <param-value>true</param-value>  
 </init-param>  
</filter>  
<filter-mapping>  
 <filter-name>encoding</filter-name>  
 <url-pattern>/\*</url-pattern>  
</filter-mapping>

1. @ResponseBody 直接返回数据时

@PostMapping(value ="/index", produces="text/html;charset=UTF-8")  
@ResponseBody  
public String handlePutIndex(HttpServletResponse response, Person person){  
 return person.getName() +" "+person.getAge();  
}

1. 如果jsp中的EL表达式不能够正确执行

① 查看以下依赖是否导入

<dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>jstl</artifactId>  
 <version>1.2</version>  
</dependency>  
<!-- https://mvnrepository.com/artifact/taglibs/standard -->  
<dependency>  
 <groupId>taglibs</groupId>  
 <artifactId>standard</artifactId>  
 <version>1.1.2</version>  
</dependency>

② 在JSP头部添加

<%@ **page** isELIgnored="false" %>

1. 出现错误：
   * The request sent by the client was syntactically incorrect.
   * 因为前台form表单中的某些参数与后台接受的参数类型不一致导致的，将日志级别调为DEBUG，可以看到spring抛出的异常信息
2. 如果是日期类型不能自动转换，可以在日期上加上注解来解决：@DateTimeFormat(“yyyy-MM-dd”)
3. 使用spring向其他接口请求数据，可以使用RestTemplate发起http请求：

HttpHeaders requestHeaders = new HttpHeaders();  
// 设置请求头为 application/json  
requestHeaders.setContentType(MediaType.*APPLICATION\_JSON\_UTF8*);  
HttpEntity<String> requestEntity = new HttpEntity<>(message,requestHeaders);  
RestTemplate rt = new RestTemplate();  
List<HttpMessageConverter<?>> messageConverters = rt.getMessageConverters();  
messageConverters.add(new MappingJackson2HttpMessageConverter());  
messageConverters.add(new StringHttpMessageConverter());  
messageConverters.add(new StringHttpMessageConverter(Charset.*forName*("UTF-8")));  
try {  
 log.info("[REQUEST BODY]: " + requestEntity.getBody());  
 ResponseEntity<String> response = rt.exchange(serverUrl, HttpMethod.*POST*, requestEntity, String.class);  
  
 HttpStatus status = response.getStatusCode();  
  
 if (status == HttpStatus.*ACCEPTED*){  
 String responseBody = response.getBody();  
 log.info(responseBody);  
 log.info("[REQUEST SUCCESS]:" + responseBody);  
 return responseBody;  
 } else {  
 log.info("[REQUEST ERROR]:" + status.toString());  
 return status.toString() + " : " + status.getReasonPhrase();  
 }  
} catch (Exception e) {  
 log.error(e.getCause());  
 return "{error : " + e.getMessage() + "}";  
}