# 自定义参数绑定—日期格式

## 案列

### 需求

在商品修改页面可以修改商品的生产日期，并且根据业务需求自定义日期格式。

### 需求分析

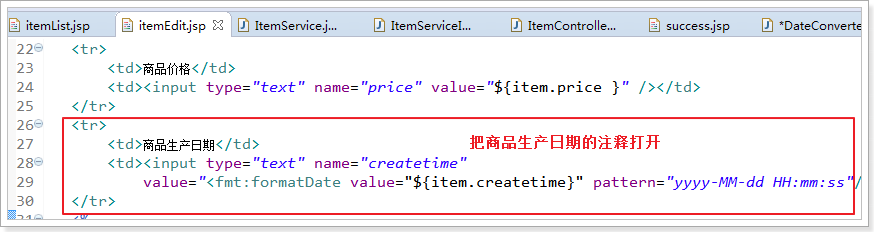
由于日期数据有很多种格式，springmvc没办法把字符串转换成日期类型。所以需要自定义参数绑定。

前端控制器接收到请求后，找到注解形式的处理器适配器，对RequestMapping标记的方法进行适配，并对方法中的形参进行参数绑定。可以在springmvc处理器适配器上自定义转换器Converter进行参数绑定。

一般使用<mvc:annotation-driven/>注解驱动加载处理器适配器，可以在此标签上进行配置。

### 修改jsp页面

如下图修改itemEdit.jsp页面，显示时间



### 自定义Converter

//Converter<S, T>

//S:source,需要转换的源的类型

//T:target,需要转换的目标类型

**public** **class** DateConverter **implements** Converter<String, Date> {

@Override

**public** Date convert(String source) {

**try** {

// 把字符串转换为日期类型

SimpleDateFormat simpleDateFormat = **new** SimpleDateFormat("yyy-MM-dd HH:mm:ss");

Date date = simpleDateFormat.parse(source);

**return** date;

} **catch** (ParseException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

// 如果转换异常则返回空

**return** **null**;

}

}

### 配置Converter

我们同时可以配置多个的转换器。

类似下图的usb设备，可以接入多个usb设备



<!-- 配置注解驱动 -->

<!-- 如果配置此标签,可以不用配置... -->

<mvc:annotation-driven conversion-service=*"conversionService"* />

<!-- 转换器配置 -->

<bean id=*"conversionService"* class=*"org.springframework.format.support.FormattingConversionServiceFactoryBean"*>

<property name=*"converters"*>

<set>

<bean class=*"cn.itcast.springmvc.converter.DateConverter"* />

</set>

</property>

</bean>

### 配置方式2（了解）

<!--注解适配器 -->

<bean class=*"org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter"*>

<property name=*"webBindingInitializer"* ref=*"customBinder"*></property>

</bean>

<!-- 自定义webBinder -->

<bean id=*"customBinder"* class=*"org.springframework.web.bind.support.ConfigurableWebBindingInitializer"*>

<property name=*"conversionService"* ref=*"conversionService"* />

</bean>

<!-- 转换器配置 -->

<bean id=*"conversionService"* class=*"org.springframework.format.support.FormattingConversionServiceFactoryBean"*>

<property name=*"converters"*>

<set>

<bean class=*"cn.itcast.springmvc.convert.DateConverter"* />

</set>

</property>

</bean>

注意：此方法需要独立配置处理器映射器、适配器，

不再使用<mvc:annotation-driven/>

## 日期格式化

### 需求

在商品修改页面可以修改商品的生产日期，并且根据业务需求自定义日期格式。

### 需求分析

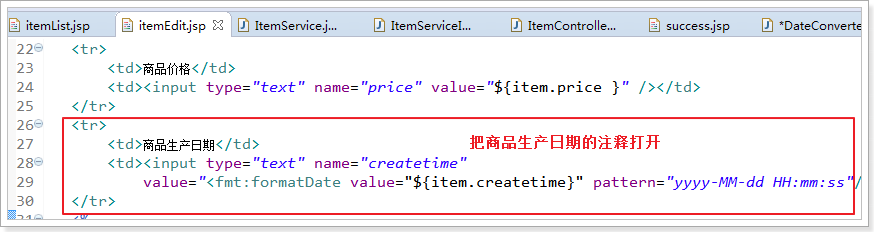
由于日期数据有很多种格式，springmvc没办法把字符串转换成日期类型。所以需要自定义参数绑定。

前端控制器接收到请求后，找到注解形式的处理器适配器，对RequestMapping标记的方法进行适配，并对方法中的形参进行参数绑定。可以在springmvc处理器适配器上自定义转换器Converter进行参数绑定。

一般使用<mvc:annotation-driven/>注解驱动加载处理器适配器，可以在此标签上进行配置。

### 修改jsp页面

如下图修改itemEdit.jsp页面，显示时间



### 自定义Converter

//Converter<S, T>

//S:source,需要转换的源的类型

//T:target,需要转换的目标类型

**public** **class** DateConverter **implements** Converter<String, Date> {

@Override

**public** Date convert(String source) {

**try** {

// 把字符串转换为日期类型

SimpleDateFormat simpleDateFormat = **new** SimpleDateFormat("yyy-MM-dd HH:mm:ss");

Date date = simpleDateFormat.parse(source);

**return** date;

} **catch** (ParseException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

// 如果转换异常则返回空

**return** **null**;

}

}

### 配置Converter

我们同时可以配置多个的转换器。

类似下图的usb设备，可以接入多个usb设备



<!-- 配置注解驱动 -->

<!-- 如果配置此标签,可以不用配置... -->

<mvc:annotation-driven conversion-service=*"conversionService"* />

<!-- 转换器配置 -->

<bean id=*"conversionService"* class=*"org.springframework.format.support.FormattingConversionServiceFactoryBean"*>

<property name=*"converters"*>

<set>

<bean class=*"cn.itcast.springmvc.converter.DateConverter"* />

</set>

</property>

</bean>

### 配置方式2（了解）

<!--注解适配器 -->

<bean class=*"org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter"*>

<property name=*"webBindingInitializer"* ref=*"customBinder"*></property>

</bean>

<!-- 自定义webBinder -->

<bean id=*"customBinder"* class=*"org.springframework.web.bind.support.ConfigurableWebBindingInitializer"*>

<property name=*"conversionService"* ref=*"conversionService"* />

</bean>

<!-- 转换器配置 -->

<bean id=*"conversionService"* class=*"org.springframework.format.support.FormattingConversionServiceFactoryBean"*>

<property name=*"converters"*>

<set>

<bean class=*"cn.itcast.springmvc.convert.DateConverter"* />

</set>

</property>

</bean>

注意：此方法需要独立配置处理器映射器、适配器，

不再使用<mvc:annotation-driven/>