31 Converter还是Formatter?

更新时间：2020-07-28 08:49:38

> 书是人类进步的阶梯。——高尔基

Converter

Spring 的 Converter 可以将一种类型转换成另一种类型。

在使用时，必须编写一个实现 org.springframework.core.convert.converter.Converter 接口的 java 类。这个接口的声明如下

```
public interface Converter<S, T> {
    T convert(S var1);
}
```

这里的 S 表示源类型，T 表示目标类型。

下面展示了一个将 String 类型转换成 Date 类型的 Converter：

```
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.springframework.core.convert.converter.Converter;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;

public class StringToDateConverter implements Converter<String, Date>{
   private static final Log logger = LogFactory.getLog(StringToDateConverter.class);
   private String datePattern;

   public StringToDateConverter(String datePattern) {
      this.datePattern = datePattern;
   }

   public Date convert(String s) {
      try {
         SimpleDateFormat dateFormat = new SimpleDateFormat(datePattern);
         dateFormat.setLenient(false);
         return dateFormat.parse(s);
      } catch (ParseException e) {
         throw new IllegalArgumentException("invalid date format. Please use this pattern\"" + datePattern + "\"");
      }
   }
}
```

为了使用 Spring MVC 应用程序定制的 Converter，需要在配置文件中添加一个 conversionService bean。Bean 的类名称必须为 org.springframework.context.support.ConversionServiceFactoryBean。这个 bean 必须包含一个 converters 属性，它列出要在应用程序中使用的所有定制 Converter。下面 bean 声明注册了上面 StringToDateConverter。

```
<bean id="conversionService" class="org.springframework.context.support.ConversionServiceFactoryBean">
   <property name="converters">
      <set>
         <bean class="app06a.converter.StringToDateConverter">
            <constructor-arg name="datePattern" value="yyyy-MM-dd"/>
         </bean>
      </set>
   </property>
</bean>
```

之后，还需要给 annotation-driven 元素的 conversion-service 属性赋上 bean 名称，如下

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

Formatter

Formatter 和 Converter 一样，也是将一种类型转换成另一种类型。但是，Formatter 的源类型必须是一个 String。在使用时，必须编写一个实现 org.springframework.format.Formatter 接口的 java 类。这个接口的声明如下

```
public interface Formatter<T> extends Printer<T>, Parser<T> {
}
public interface Printer<T> {
   String print(T var1, Locale var2);
}
public interface Parser<T> {
   T parse(String var1, Locale var2) throws ParseException;
}
```

这里的 T 表示输入字符串要转换的目标类型。

parse 方法利用指定的 Locale 将一个 String 解析成目标类型。print 方法相反，它是返回目标对象的字符串表示法。

下面展示了一个将 String 类型转换成 Date 类型的 Formatter

```
import org.springframework.format.Formatter;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;

public class DateFormatter implements Formatter<Date>{
  private String datePattern;
  private SimpleDateFormat dateFormat;

  public DateFormatter(String datePattern) {
    this.dateFormat = dateFormat;
    dateFormat = new SimpleDateFormat(datePattern);
    dateFormat.setLenient(false);
  }
  public Date parse(String s, Locale locale) throws ParseException {
    try {
      SimpleDateFormat dateFormat = new SimpleDateFormat(datePattern);
      dateFormat.setLenient(false);
      return dateFormat.parse(s);
    } catch (ParseException e) {
      throw new IllegalArgumentException("invalid date format. Please use this pattern\"" + datePattern + "\"");
    }
  }

  public String print(Date date, Locale locale) {
    return dateFormat.format(date);
  }
}
```

为了使用 Spring MVC 应用程序定制的 Formatter，需要在配置文件中添加一个 conversionService bean。Bean 的类名称必须为 org.springframework.format.support.FormattingConversionServiceFactoryBean。这个 bean 可以用一个 formatters 属性注册 Formatter，用一个 converters 属性注册 Converter。下面 bean 声明注册了上面 DateFormatter。

```
<bean id="conversionService" class="org.springframework.format.support.FormattingConversionServiceFactoryBean">
  <property name="formatters">
    <set>
      <bean class="app06a.formatter.DateFormatter">
        <constructor-arg name="datePattern" value="yyyy-MM-dd"/>
      </bean>
    </set>
  </property>
</bean>
```

之后，还需要给 annotation-driven 元素的 conversion-service 属性赋上 bean 名称，如下

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

用 Registrar 注册 Formatter

注册 Formatter 的另一种方法是使用 Registrar。

下面就用 Registrar 来注册前面的 DateFormatter。

先需要实现 org.springframework.format.FormatterRegistrar 接口，如下：

```
import org.springframework.format.FormatterRegistrar;
import org.springframework.format.FormatterRegistry;

public class MyFormatterRegistrar implements FormatterRegistrar {
  private String datePattern;

  public MyFormatterRegistrar(String datePattern) {
    this.datePattern = datePattern;
  }

  public void registerFormatters(FormatterRegistry formatterRegistry) {
    formatterRegistry.addFormatter(new DateFormatter(datePattern));
    //register more formatters here
  }
}
```

有了 Registrar，就不需要在 Spring MVC 配置文件中注册 Formatter，只要在配置文件中注册 Registrar 就行，如下：

```
<bean id="conversionService" class="org.springframework.format.support.FormattingConversionServiceFactoryBean">
  <property name="formatterRegistrars">
    <set>
      <bean class="app06a.formatter.MyFormatterRegistrar">
        <constructor-arg name="datePattern" value="yyyy-MM-dd"/>
      </bean>
    </set>
  </property>
</bean>
```

选择 Converter 还是 Formatter

Converter 是一般工具，可以将一种类型转换成另一种类型。例如，将 String 转换成 Date，或者将 Long 转换成 Date。Converter 既可以用在 Web 层，也可以用在其它层中。

Formatter 只能将 String 转成成另一种 Java 类型。例如，将 String 转换成 Date，但它不能将 Long 转换成 Date。所以，Formatter 适用于 Web 层。为此，在 Spring MVC 应用程序中，选择 Formatter 比选择 Converter 更合适。

}

30 Spring Core格式化Formatter 实现示例及背后原理探究

32 数据转换总结及常见面试题目解析