# 11. Validation, Formatting, and Type Conversion

-org.springframework.core.convert: convert any Java types within Spring application.

## 11.1 Converting String Values using PropertyEditors

-Record:

A close up of words

AI-generated content may be incorrect.

-Spring Configuration with Error:

A computer screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

-To fix this, We need to tell Spring how to do the conversion of text representations of calendar dates to LocalDate. We can do by an extension of PropertyEditorSupport:

A screen shot of a computer code

AI-generated content may be incorrect.

-Registry custom property editor:

A computer code with black text

AI-generated content may be incorrect.

A computer code with black text

AI-generated content may be incorrect.

## 11.2 Introduce Spring Type Conversion

### 11.2.1 Implement a Custom Converter

-Implement Converter<S,T>



A black text on a white background

AI-generated content may be incorrect.

-To use this converter instead of PropertyEditor, we need to configure ConversionService interface:

A computer screen shot of a program

AI-generated content may be incorrect.

+Declare conversionService bean with class ConversionServiceFactoryBean. This type of bean groups multiple conversion services: if no conversions service bean is defined, Spring will use PropertyEditor-based system.

-By default, type conversion service supports conversions between common types: string, number, enum, collection, map.

### 11.2.2 Converting between Arbitrary Types

A screenshot of a computer program

AI-generated content may be incorrect.

A computer screen shot of a program

AI-generated content may be incorrect.

-Spring MVC makes heavy use of conversion service (and Formatter SPI):

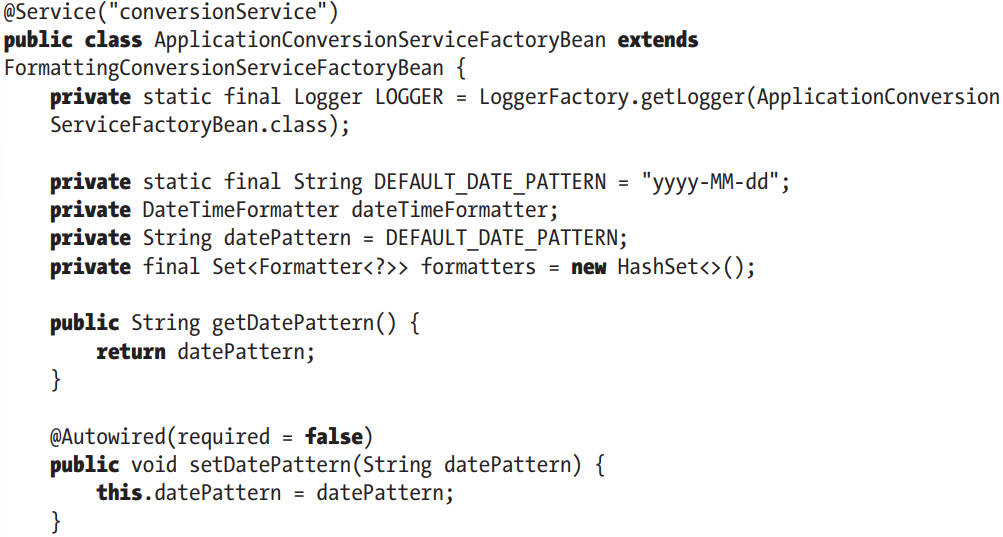
+@EnableWebMvc automatically register all default converters (StringToArrayConverter, StringToBooleanConverter, StringToLocaleConverter... in org.springframework.core.convert.support) and formatter (CurrencyStyleFormatter, DataFormatter, AbstractNumberFormatter in org.springframework.format)

## 11.3 Field Formatting in Spring

-Formatter SPI: configure filed-formatting aspects

-The main interface is Formatter<T>. It provides implementations: CurrencyStyleFormatter, DateFormatter, AbstractNumberFormatter, PercentStyleFormatter

-We extend FormattingConversionServiceFactoryBean and provide custom formatter. This is a factory class that provides access to FormattingConversionService



A screenshot of a computer code

AI-generated content may be incorrect.

## 11.4 Validation in Spring

### 11.4.1 Use Spring Validator in Spring

### 11.4.2 Use JSR-349/Jakarta Bean Validation

### 11.4.3 Dependencies

### 11.4.4 Define Validation Constraints on Domain Object Properties

### 11.4.5 Configure Bean Validation Support in Spring

### 11.4.6 Create a Custom Validator

### 11.4.7 Use AssertTrue for Custom Validation

### 11.4.8 Decide which Validation API to use

### 11.4.9 Configure Validation in Spring Boot App

# 12. Task Execution and Scheduling

## 12.1 Task Executing in Java

## 12.2 Task Executing in Spring

## 12.3 Task Scheduling in Spring

### 12.3.1 Introduce Spring TaskScheduler Abstraction

### 12.3.2 Explore a Sample Task

### 12.3.3 Asynchronous Task Execution in Spring

# 13. Spring Remoting

## 13.1 Communication via HTTP using Spring REST

## 13.2 Use JMS in Spring

### 13.2.1 Working with Apache ActiveMQ Artemis

## 13.3 Use Spring for Apache Kafka