Spring Expression Language (SpEL)

GETTING TO KNOW SPEL: SIMPLE SPEL EXPRESSIONS

Buddhini Samarakkody
JAVA DEVELOPER/INDEPENDENT CONSULTANT
www.buddhini-samarakkody.mystrikingly.com

Overview

Not a new feature – since Spring v 3.0

Highlights

- What is SpEL?
- Why use SpEL?
- How to use SpEL?
  - Writing & parsing basic expressions
  - Using an Evaluation Context

Case Study – Applying to an E-Commerce application

Dynamic Bean Wiring

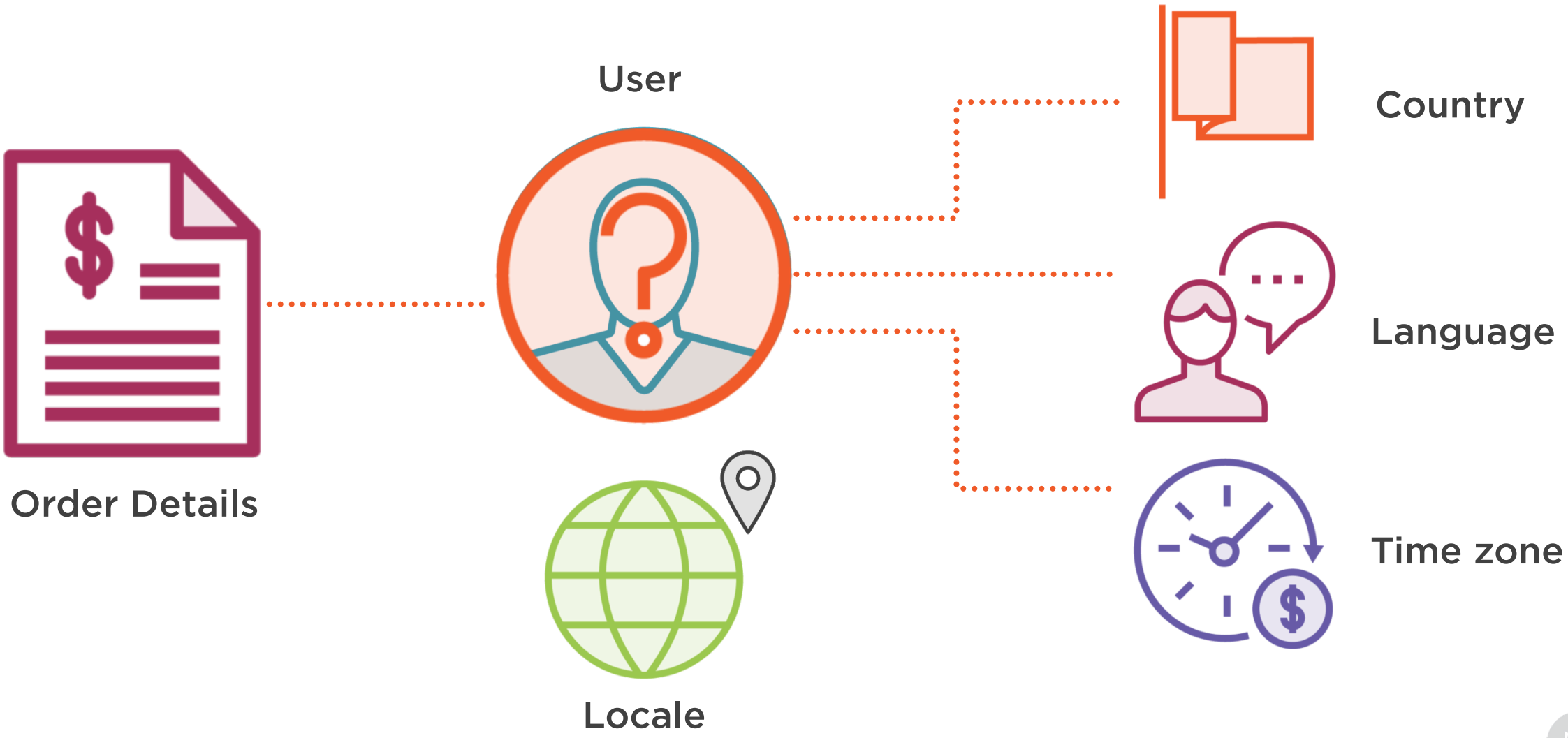
Pick a bean or assign default value to bean property

At runtime

Based on

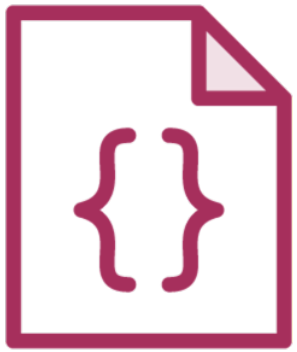
A condition

The E-commerce Application

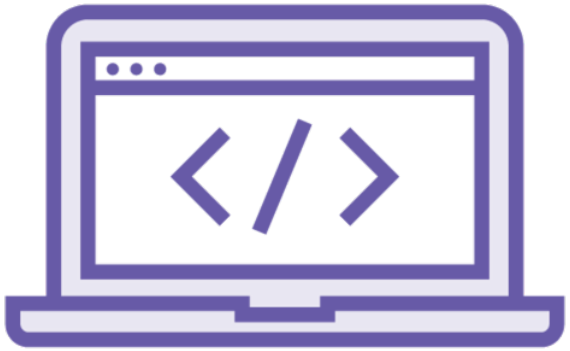


SpEL Overview

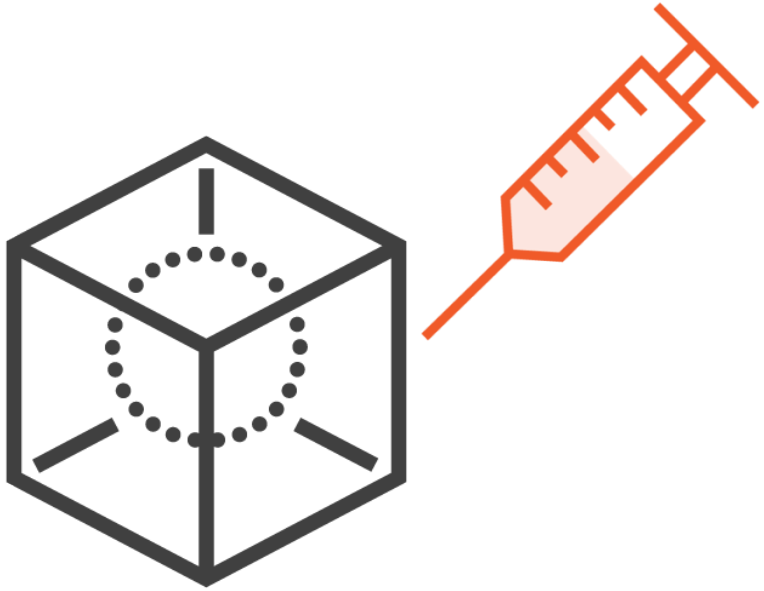
Spring Expression Language (SpEL)



"String" expression



Evaluate



Inject beans/values to beans

Syntax of SpEL

Strings

“ ”

“expression”

“#variableName”

“#{expression}”

Some Examples: Using with Plain Java Code

Literal String expression

"'Hello World'"

Accessing a variable

"#greeting"

Method call

"#greeting.length()"

Mathematical operation

"#greeting.length()*10"

Relational operator

"#greeting.length()>10"

Logical operator

“#greeting.length()>10 and #greeting.length()<20"

Some Examples: Using with Metadata (Annotations & XML)

Literal expressions

"#{'John Doe'}"

"#{30}"

Call bean property

"#{user.country}"

Mathematical operation

"#{100.55 + 500.75 + 400.66}"

Using SpEL with Plain Java vs. Metadata

Plain Java Code

Metadata

Inside of Spring, a collection of classes are used to parse and evaluate Spring expressions!

Inside Spring: SpEL

```
SpelExpressionParser parser = new SpelExpressionParser();

Expression exp1 = parser.parseExpression("'Hello World'");

String message = (String) exp1.getValue();
```

Demo

Basic SpEL expressions in plain Java code

Steps

- Create Spring Boot project and set it up on IntelliJ IDEA
- Check the dependencies
- Write an Expression Parser class to demonstrate some basic expressions
- Run the application

Evaluation Context

What is it?

Interface in the SpEL API

Used when evaluating an expression

- to resolve fields
- to resolve properties
- to resolve methods
- to perform type conversions

StandardEvaluationContext

EvaluationContext in Use

Resolving the field greeting in EvaluationContext ec1

AppExpressionParser.java

```
StandardEvaluationContext ec1= new StandardEvaluationContext();

ec1.setVariable("greeting", "Hello USA");

String msg = (String) parser.parseExpression("#greeting.substring(6)").getValue(ec1);
```

USA

EvaluationContext in Use

Resolving the field greeting in EvaluationContext ec2

AppExpressionParser.java

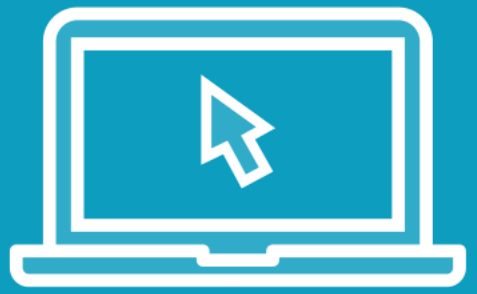
```
StandardEvaluationContext ec2 = new StandardEvaluationContext();

ec2.setVariable("greeting", "Hello UK");

String msg2 = (String) parser.parseExpression("#greeting.substring(6)").getValue(ec2);
```

UK

Demo



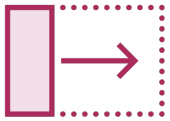
Using an Evaluation Context

Steps

- Create a StandardEvaluationContext
- Write code to resolve a field using it
- Create a bean: User
- Set it as root object for resolving bean properties

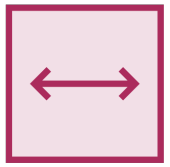
The systemProperties Predefined Variable

systemProperties: What and Why?



Storing current run-time information as key/value pairs

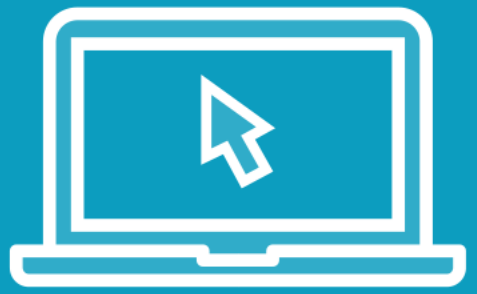
Information: OS, user name, country, language, time zone, etc.



Conditionalization of applications

SpEL provides pre-defined systemProperties variable

Demo



Setting the user's country, language and time zone based on the locale

Steps

- Write SpEL to access user information from systemProperties
- Wire the User bean's properties based on Locale specific information from systemProperties

Summary

We learned:

What is SpEL?

Why use it?

Write & parse simple expressions

Evaluation Context

Using the systemProperties pre-defined variable