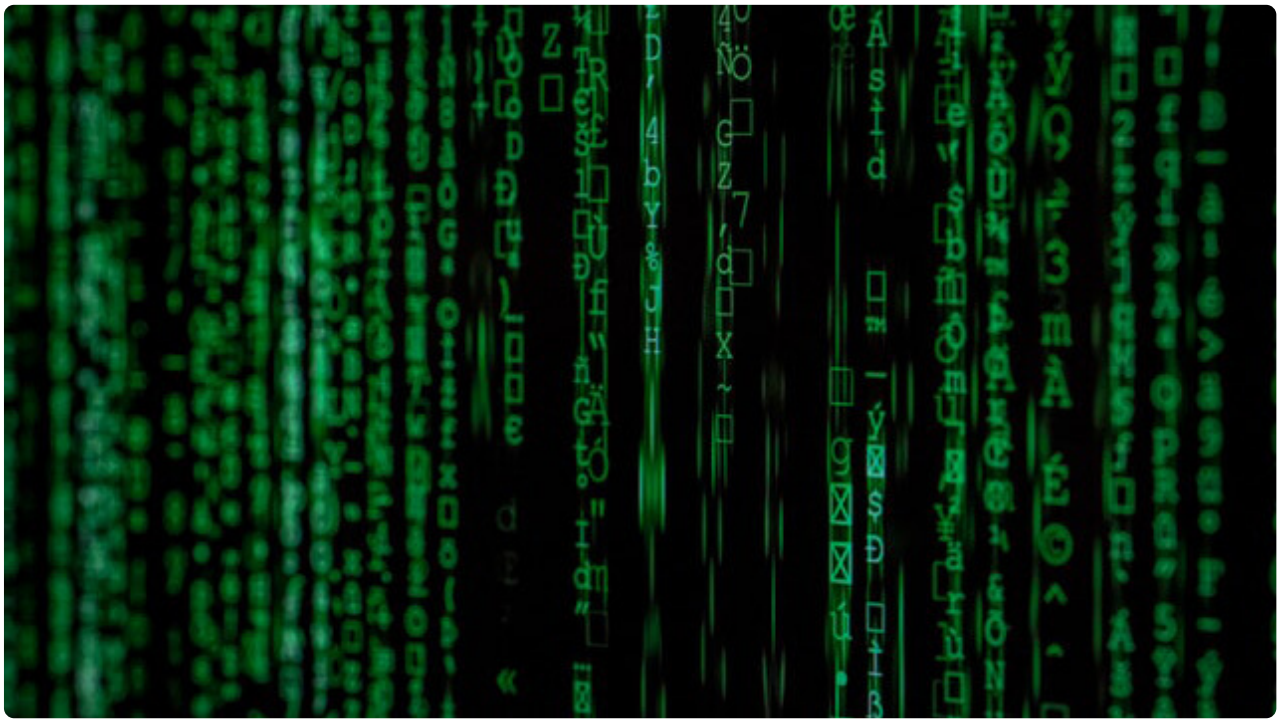
38 SpEL总结及常见面试题解析

更新时间：2020-08-17 14:48:54



> 我们有力的道德就是通过奋斗取得物质上的成功；这种道德既适用于国家，也适用于个人。——罗素

SpEL 总结

Spring Expression Language（简称 SpEL）是一种功能强大的表达式语言、用于在运行时查询和操作对象图；语法上类似于 Unified EL，但提供了更多的特性，特别是方法调用和基本字符串模板函数。

虽然目前已经有许多其他的 Java 表达式语言，例如 OGNL，MVEL 和 Jboss EL，SpEL 的诞生是为了给 Spring 社区提供一种能够与 Spring 生态系统所有产品无缝对接，能提供一站式支持的表达式语言。它的语言特性由 Spring 生态系统的实际项目需求驱动而来，比如基于 Eclipse 的 Spring Tool Suite（Spring 开发工具集）中的代码补全工具需求。尽管如此、SpEL 本身基于一套与具体实现技术无关的 API，在需要的时候允许其他的表达式语言实现集成进来。

尽管 SpEL 在 Spring 产品中是作为表达式求值的核心基础模块，本身可以脱离 Spring 独立使用。

SpEL 支持以下的一些特性：

- 字符表达式
- 布尔和关系操作符
- 正则表达式
- 类表达式
- 访问 properties，arrays，lists，maps 等集合
- 方法调用
- 关系操作符

- 赋值
- 调用构造器
- Bean 对象引用
- 创建数组
- 内联 lists
- 内联 maps
- 三元操作符
- 变量
- 用户自定义函数
- 集合投影
- 集合选择
- 模板表达式

SpEL 热点面试题集锦

Q-1：Spring 表达式语言的特点是什么？

A: SpEL 是一种强大的表达式语言，支持在 bean 创建时或运行时查询和操作对象。它类似于其他表达式语言，如 JSP EL、OGNL、MVEL 和 JBoss EL 等，还有一些附加特性，如方法调用和基本的字符串模板功能。

语法形式：

```
#{ expression }
```

它可以使用：

- 文字表达式（Literal expressions）；
- 布尔和关系运算（Boolean and relational operators）；
- 正则表达式（Regular expressions）；
- 类表达式（Class expressions）；
- 获取属性，数组，列表，键值对（Accessing properties, arrays, lists, maps）；
- 方法触发（Method invocation）；
- 关系操作符（Relational operators）；
- 赋值（Assignment）；
- 调用构造器（Calling constructors）；
- ……

Q-2：在 spring 的表达式计算中，类是如何使用的？

A: 下面的表达式计算代码片段可以看出：

```
ExpressionParser expPar = new SpelExpressionParser();
Expression exp = expPar.parseExpression("'Hi, My Name is Ram'");
String message = (String) exp.getValue();
```

Q-3: 在 spring 中 EvaluationContext 起到什么作用？

A: EvaluationContext 可以解析表达式中的属性，方法，成员变量，也可以做类型转换的工作。

Q-4: SpEL 支持的 Bean 定义方法有哪些？

A: SpEL 支持两种方式：

1. 基于 XML 配置；
2. 基于注解配置。

Q-5: 在 Spring 表达式中 #this 和 #root 变量分别是什么意思？

A: #this 和 #root 变量：

- #this 是当前计算对象的引用；
- #root 是根上下文对象的引用。

#this 变量永远指向当前表达式正在求值的对象（这时不需要限定全名）。变量#root 总是指向根上下文对象。#this 在表达式不同部分解析过程中可能会改变，但是#root 总是指向根。示例：

```
import java.util.ArrayList;
import java.util.List;
import org.springframework.expression.ExpressionParser;
import org.springframework.expression.spel.standard.SpelExpressionParser;
import org.springframework.expression.spel.support.StandardEvaluationContext;
public class SpELTest {
        public static void main(String[] args) {
                ExpressionParser parser = new SpelExpressionParser();
                List<Integer> even = new ArrayList<Integer>();
                even.add(2);
                even.add(4);
                even.add(6);
                even.add(8);
                even.add(10);

                StandardEvaluationContext seContext = new StandardEvaluationContext();
                seContext.setVariable("even",even);

                List<Integer> evenGtfour =
            (List<Integer>) parser.parseExpression("#even.?[#this>4]").getValue(seContext);

                for(Integer i: evenGtfour){
                        System.out.println(i);
                }
        }
}
```

输出结果

```
6
8
10
```

}

37 SpEL Bean方法属性引用原理

39 图说AOP—妈妈再也不担心我概念混淆了