尚硅谷大数据技术之 Scala
函数式编程

官网：www.atguigu.com

ShangGuigu Technologies Co., Ltd.

尚硅谷技术有限公司

一　函数式编程

"函数式编程"是一种"编程范式"（programming paradigm），也就是如何编写程序的方法论。

它属于"结构化编程"的一种，主要思想是把运算过程尽量写成一系列嵌套的函数调用。

函数编程语言最重要的基础是λ演算（lambda calculus），而且λ演算的函数可以接受函数当作输入（参数）和输出（返回值）。

函数式编程中，最重要的就是函数。那什么是函数？编程语言中所谓的函数就是为了**实现特定逻辑而封装的功能代码**，如果逻辑中需要外置的条件进行判断或处理，那么函数可以增加参数,如下面的代码

```
function print() {
    console.log(“Hello World”);
}
function sum( num1, num2 ) {
    return num1 + num2;
}
```

上面的代码其实就是 JavaScript 的代码，也就意味着，JavaScript 也是一种函数式编程语言。那么在学习 Scala 时，函数式编程的部分可以对比着 JavaScript 进行学习。

二　基础语法

2.1　函数声明

```
// JavaScript
// function 函数名（参数列表）{函数体}
// 无论函数是否有返回值，都无需声明无返回值类型（弱类型语言）
// 如果需要使用函数返回值，JavaScript 可以根据结果自动推断类型（动态类型语言）
// 函数调用
// 函数名()
// 小括号如果省略，那么方法不会执行，也不会报错
```

```
// Scala
// def 函数名（参数名[：参数类型]...）[: 返回值类型 = ] {函数体}
// 函数声明关键字为 def（definition）
// 如果函数中无返回值，那么返回值类型无需声明
// 如果函数中有返回值，且使用 return 关键字声明，那么返回值类型需要声明（即使有 unit，矛盾了，但是也不会报错，但是 return 不生效，函数的结果就是括号）
// 如果函数有返回值，但是返回值类型没有声明，那么方法返回值由 Scala 自行推断（scala 中函数的返回值可以不用 return 声明，函数会使用最后一行的结果作为函数的返回值）
Println 函数的返回值是 unit
// 函数调用
// 函数名()
// 如果函数无参数列表，那么小括号可以省略
```

2.2　变量声明

```
// JavaScript
// var 变量名 = 变量值
// 声明变量无需声明类型（弱类型语言）（在执行的时候判断出类型）
```

```
// Scala
```

```
// var | val 变量名 [：变量类型] = 变量值
// 声明变量时，类型可以省略（编程语言自动推断）
// var 关键字表示声明的值可以改变，而 val 关键字声明的变量表示值无法改变
```

2.3 类型声明

```
// JavaScript
// JavaScript 是弱类型语言，这里的弱类性指的是在编程时可以不需要类型的声明，但并不是说 JavaScript 就没有类型
// javascript 中有 2 种类型
// 原始类型（基本类型）：string, number,boolean,null,undefined
// 引用类型：object(Function, Date, Array)
```

// Scala

// Scala 常用类型中包含有 7 种数值类型：**Byte、Char、Short、Int、Long、Float、Double** 及 **Boolean** 类型，还有 **String** 类型。(**具体类型在后面的 JAVA 中介绍**)

Boolean	true 或者 false
Byte	8 位，有符号
Short	16 位，有符号
Int	32 位，有符号
Long	64 位，有符号
Char	16 位，无符号
Float	32 位，单精度浮点数
Double	64 位，双精度浮点数
String	其实就是由 Char 数组组成

2.4 语义推断

```
// JavaScript

// 一行逻辑代码的最后，无需使用分号结束，编程语言自动推断
```

```
// var sum = 1 + 1; (OK)
// var sum = 1 + 1 (OK)
// 但是一行代码中如果存在多个逻辑代码，那么不同的逻辑代码必须使用分号分隔
// var sum = 1 + 1    var avg = 20    (X)
// var sum = 1 + 1;    var avg = 20    (OK)
```

```
// Scala
// 一行逻辑代码的最后，无需使用分号结束，编程语言自动推断
// var sum = 1 + 1; (OK)
// var sum = 1 + 1 (OK)
// 但是一行代码中如果存在多个逻辑代码，那么不同的逻辑代码必须使用分号分隔
// var sum = 1 + 1    var avg = 20    (X)
// var sum = 1 + 1;    var avg = 20    (OK)
```

三　基础特性

3.1　函数是第一等公民

这里所谓的第一等公民，表示在函数式编程语言中，函数与其他数据类型一样，处于平等地位，可以赋值给其他变量，也可以作为参数，传入另一个函数，或者作为别的函数的返回值。**JavaScript 中的回调函数就是典型应用**。

```
// JavaScript
// 声明方法
function f1() {
    console.log(“Hello World”) // 控制台打印 Hello World
}
// 将方法赋值给变量使用，那么变量也就变成了一个方法
var v1 = f1;
// 调用方法
```

```
v1();
function f2() {
    return f1;
}
var v2 = f2
v2()()
```

```
// Scala
// 声明方法
def f1() {
    println("Hello World") // 控制台打印 Hello World
}
// 将方法赋值给变量使用
var v1 = f1;
// 调用方法
v1();
def f2() {
    f1
}
var v2 = f2
v2()()
```

2.2 纯函数

就是没有副作用的函数，这里所谓的没有副作用，指的是函数要保持独立，所有功能就是返回一个新的值，没有其他行为，尤其是不得修改外部变量的值。

```
// 纯函数
function sum( num1, num2 ) {
    return num1 + num2
```

```
}
// 非纯函数
var num1 = 10; // 函数外变量
function sum( num2 ) {
    return num1 + num2;
}
```

2.3 引用透明

在函数式编程中，引用透明指的是运行函数的时候，函数的每一个步骤都不会牵连到函数的外部变量或状态，而是只依赖于函数输入的参数，**相同的参数输入总会得到相同的函数返回值（如 MD5 加密算法）**。而在其他类型的语言中，函数的返回值不仅仅与函数的参数传入有关，也与当前的系统状态有关。在不同的系统状态的情况下，函数的返回值不同。