Scala03

- 样例类
  - 定义样例类
    - 语法：case class 样例类名(val/var 成员变量名：类型，成员变量名：类型；...)
    - 注意：如果要实现某个成员变量可以被修改，可以添加var
    - 默认为val，可以省略
    - 示例
      - 定义一个样例类
      - 可变成员变量
      - 当我们定义一个样例类，编译器会帮我们实现了以下几个有用的方法：
      - apply：apply方法可以让我们快速地使用类名来创建对象。
      - toString：toString返回样例类名称加成员变量1，成员变量2，成员变量3，...）。我们可以更方便地看到样例类的成员
      - equals：可以直接使用==比较两个样例类是否相等，即所有的成员变量是否相等
      - hashCode：如果所有成员变量的值相同，则hash值相同，只要有一个不一样，则hash值不一样
      - copy：可以快速创建一个相同的实例对象，可以使用带有参数指定成员的计算新赋值
  - 样例类的方法
- 样例对象
  - 主要作用：1.定义枚举
  - 2.作为没有任何参数的消息传递(Akka编程)
  - 定义：使用case object可以创建样例对象，样例对象是单例，而且它没有主构造器
  - 语法：case object 样例对象名
  - 示例：定义枚举
- 模式匹配
  - switch语句
  - 使用模式匹配快速获取数据
  - scala中有一个非常强大的模式匹配机制，可以应用在很多场景：
  - 简单匹配
    - 语法：
    - 示例：
  - 匹配类型
    - 语法：match {
      - case 类型变量名：类型1 => 表达式1
      - case 类型变量名：类型2 => 表达式2
      - case 类型变量名：类型3 => 表达式3
      - ...
      - case _ => 表达式n
    - }
    - 示例：
    - 如果case表达式中无需使用匹配到的变量，可以使用下划线代代替
  - 守卫
    - 定义：在case语句中添加守卫条件即可。
    - 示例：
  - 匹配样例类
    - 定义：scala中，Option类型本质可以理解为，这种类型的数据有两种形式：
    - Some(x)：表示实际的值
    - None：表示没有值
    - 使用getOrElse方法，当值为None情况以指定一个默认值
    - 示例：
  - 匹配集合
    - scala中的模式匹配，还能将不同的集合，进行模式匹配。
    - 匹配数组
    - 匹配列表：
    - 匹配元组：
  - 变量声明
    - 在定义变量的时候，可以使用模式匹配来进行取数据。
    - 获取数组中的元素
    - 获取List中的数据
- Option类型
  - 使用Option类型，可以很有效的避免空指针异常，也就是说，将来我们编写某些数据时，可以直接一个Option类型来表示。
  - 定义：
  - 示例：
- 偏函数
  - 偏函数是指花括号内没有match的一组case语句是一个偏函数
  - 定义：偏函数PartialFunction[-A, B]的一个实例
  - 对可能输入参数类型时(类型擦除)
  - 示例：
- 正则表达式
  - 在scala中，可以很方便地使用正则表达式来匹配数据。
  - 定义：
    - scala中提供了Regex来表示定义正则表达式。
    - 要构造一个Regex对象，直接使用String类的方法即可。
    - Regex类
    - 建议使用三个双引号来表示正则表达式，不然需要对正则中的反斜杠来进行转义
    - 示例：
    - 使用findAllMatchIn方法可以获取到所有正则匹配到的字符串
    - findAllMatchIn方法
  - 示例：
- 异常处理
  - 捕获异常
    - 语法：try {
      - // 代码
    - }
    - catch {
      - case ex:异常类型1 => // 代码
      - case ex:异常类型2 => // 代码
    - }
    - finally {
      - // 代码
    - }
    - 注意：finally代码是我们编写的业务处理代码
    - 在catch中要尽当出现某个异常时，需要执行的代码
    - 在finally中，是不管是否出现异常都会执行的代码
    - 示例：
  - 抛出异常
    - 语法抛出Throwable类型，使用throw new Exception。
    - 示例：
    - 抛出异常情况也方法上声明要抛出的异常，它的目的是为了其Java中被认为是设计失败的检查型异常。
    - 注意：
- 提取器
  - 要当模式匹配，必须要实现一个 提取器
  - 样例类自动实现了apply、unapply方法
  - 定义：
  - 要实现一个类的提取器，只需要实现类的伴生对象中实现一个 unapply 方法即可。
  - 语法：
  - 示例：
- 泛型
  - 定义泛型方法
    - 语法：def 方法名[泛型名称](_): _ = {
      - //...
    - }
    - 示例：
    - 不考虑泛型
    - 加入泛型
  - 定义一个泛型类
    - 语法：class 类[T](val 变量名: T)
    - 定义一个泛型类，直接在类名后面加上方括号，指定要使用的泛型参数
    - 指定类的泛型参数后，就使用泛型名称参数来定义变量了。
    - 示例：
    - 使用：类型名称后给类型加一个 上界，表示泛型参数必须要从该类（或本身）继承。
    - 上界
    - 示例：
    - 上界是要求当前类型的父类型，或者必须从某个类继承，而下界是必须是某个类的子类型。
    - 下界
    - 注意：实现类型上界、又有下界，下界写在前面，上界写在后面。
    - 示例：
    - 协变、逆变、非变
    - 语法：class Pair[T]{}
    - 默认是要非变的
    - 协变
    - 注意：类型形参的子类型，Pair[A]是Pair[B]的子类型，A是B的子类
    - 语法：class Pair+[T]{}
    - 逆变
    - 注意：类型形参的子类型，Pair[B]是Pair[A]的子类型，A是B的子类
    - 语法：class Pair-[T]{}
    - 非变
    - 注意：参数化类型的方向和类型的方向是一致的。
    - 语法：class Pair[-T]{}
    - 逆变
    - 注意：类型形参的子类型，Pair[A]是Pair[B]的子类型，A是B的子类
    - 语法：class Pair[+T]{}
    - 协变
    - 注意：参数化类型的方向和类型的方向是相反的。
    - 示例：

专注大数据
大数据 | 数科金融 | 人工智能
回复关注领取海量资源