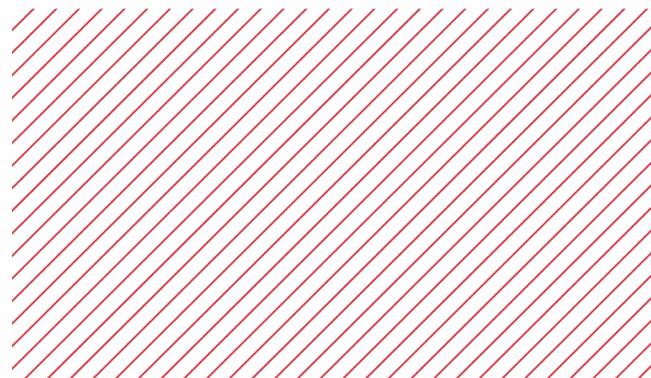


академия
больших
данных



Advanced Scala

Корольков Алексей



Структура курса

1. Введение в Scala. Основные конструкции языка



2. ООП в Scala. Pattern matching. Функциональные конструкции. Adt

3. Библиотека коллекций в Scala



4. Асинхронные операции, обработка исключений, неявные параметры

5. Параметрический полиморфизм. Имплиситы

6. Основы функционального программирования. Часть 1



7. Основы функционального программирования. Часть 2

8. ZIO, TF и основной ФП-стэк в Scala

9. Функциональные стримы на примере fs2, работа с бд

10. Акторы в Scala



Функциональный
эффект



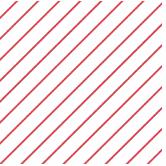
Функциональный эффект

- Шаблон, описание некоторого вычисления, потенциально с побочными эффектами, возможно асинхронного
- Позволяет отделить описание от исполнения



Функциональный эффект

- Иммутабельное значение (Объект)
- Ничего не делает, лишь описывает, что нужно сделать
- Как правило требует выполнения



Функциональный эффект

- Контроль
- Композиция
- Referential transparency



Функциональный эффект

```
val greet = {  
    println("Как тебя зовут?")  
    val name = StdIn.readLine()  
    println(s"Привет, $name")  
}
```

```
//greet : ???
```

Функциональный эффект

```
val greet = {  
    println("Как тебя зовут?")  
    val name = StdIn.readLine()  
    println(s"Привет, $name")  
}
```

```
val askForAge = {  
    println("Сколько тебе лет?")  
    val age = StdIn.readInt()  
    if (age > 18) println("Можешь проходить")  
    else println("Ты еще не можешь пройти")  
}
```



Функциональный
дизайн



Функциональный дизайн

- Иммутабельные структуры данных
- Модель
- Конструкторы
- Операторы



Функциональный дизайн

- Функциональные эффекты
- Парсер-комбинаторы
- Планировщики
- Стимы
- REST



Функциональный дизайн. Применение

- Функциональные эффекты
- Парсер-комбинаторы
- Планировщики
- Стимы
- REST



Функциональный дизайн. Модели

- **Исполняемая (Executable encoding)**

Конструкторы и операторы мы реализуем в терминах выполнения нашей модели

- **Декларативная (Declarative encoding)**

Конструкторы и операторы реализовываем, как данные, на выходе получаем древовидную рекурсивную структуру

Спасибо за внимание