

Java VM

09.09.2014

Code Style

- Набор правил и соглашений, используемых при написании кода на некотором языке программирования

Formatting

- Как в Java
- Здравый смысл

Имена

- Используйте () для функций с побочными эффектами
- Старайтесь не использовать `
- Не используйте префикс get в именах геттеров

Скобки

- Не используйте скобки для простых выражений:

```
def sqr(x: Int) = x * x
```

ВМЕСТО

```
def sqr(x: Int) = {  
    x * x  
}
```

- Не стоит опускать скобки везде, где можно:
`try` someMethod() `finally` dispose()

Import statements

- Следует сортировать в алфавитном порядке
- Используйте скобки и wildcard импорты:
`import com.foo.{A, B, C}` и `import com.foo._`
- Используйте имена с префиксом для коллекций:
`import scala.collection.immutable`
`val m = immutable.Map(1 -> 2)`
- Не используйте относительные импорты

Прочее

- Не злоупотребляйте системой типов

```
C:\work\project_scala\plugin\scala-plugin-for-ultimate\ScalaCommunity\scala-plugin\src\org\jetbrains\plugins\scala\util\macroDebug\GoToExpandedMacroCallProviderExt.scala
Error:(43, 8) error: inferred type arguments [scala.collection.immutable.LinearSeq[(Int, Int, Int)] with scala.collection.AbstractSeq[(Int, Int, Int)] with scala.collection.LinearSeqOptimized[(Int, Int, Int), scala.collection.immutable.LinearSeq[(Int, Int, Int)] with scala.collection.AbstractSeq[(Int, Int, Int)] with scala.collection.LinearSeqOptimized[(Int, Int, Int), scala.collection.immutable.LinearSeq[(Int, Int, Int)] with scala.collection.AbstractSeq[(Int, Int, Int)] with scala.collection.LinearSeqOptimized[scala.collection.LinearSeqOptimized[scala.collection.LinearSeqOptimized[(Int, Int, Int), List[(Int, Int, Int)]], scala.collection.LinearSeqOptimized[(Int, Int, Int), List[(Int, Int, Int)]], scala.collection.LinearSeqOptimized[scala.collection.LinearSeqOptimized[(Int, Int, Int), List[(Int, Int, Int)]], scala.collection.LinearSeqOptimized[(Int, Int, Int), List[(Int, Int, Int)]]]]]]] do not conform to method getOrElse's type parameter bounds [B >: scala.collection.immutable.LinearSeq[(Int, Int, Int)] with scala.collection.AbstractSeq[(Int, Int, Int)] with scala.collection.LinearSeqOptimized[(Int, Int, Int), scala.collection.immutable.LinearSeq[(Int, Int, Int)] with scala.collection.AbstractSeq[(Int, Int, Int)] with scala.collection.LinearSeqOptimized[(Int, Int, Int), scala.collection.immutable.LinearSeq[(Int, Int, Int)] with scala.collection.AbstractSeq[(Int, Int, Int)] with scala.collection.LinearSeqOptimized[(Int, Int, Int), scala.collection.immutable.LinearSeq[(Int, Int, Int)] with scala.collection.AbstractSeq[(Int, Int, Int)] with scala.collection.LinearSeqOptimized[scala.collection.LinearSeqOptimized[(Int, Int, Int), List[(Int, Int, Int)]], scala.collection.LinearSeqOptimized[(Int, Int, Int), List[(Int, Int, Int)]]]]]]] getOrElse nullOffsets
```

- Указывайте возвращаемый тип для публичных методов

```
def testUser = new User { def getId = 0 }
```

Задачки (1)

- GCD
- Fibonacci
- QSort

Задачки (2)

- Опишите для типа "электрическая схема"

```
abstract class Scheme
case class Res(v: Int)
case class Par(s1: Scheme, s2: Scheme)
case class Seqq(s1: Scheme, s2: Scheme)
```

функцию `show` так, чтобы она наглядно изображала схему в виде резисторов, соединенных проводами, с помощью символов псевдографики (например, с помощью '-', '|', '+')

Контакты

- dmitry.naydanov@jetbrains.com