



ТИПЫ: ПСЕВДОНИМЫ И КОМПОНЕНТЫ

ПСЕВДОНИМЫ

```
type IntList = List[Int]
```

- Для типов можно объявлять псевдонимы

```
type DenseMatrix[A] = Vector[Vector[A]]
```

- Они также могут иметь параметры

```
type SparseMatrix[+A] = Map[(Int, Int), A]
```

- И передавать варианты

ПСЕВДОНИМЫ

```
def evens(xs: IntList): IntList =  
  xs.filter(_ % 2 == 0)
```

- Псевдонимы полностью прозрачны, их использование эквивалентно использованию оригинального типа

ПСЕВДОНИМЫ

```
type Matrix[F[X] <: Iterable[X], A] = F[F[A]]
```

- Они могут накладывать ограничения на свои параметры

```
type IntMap[A] = Map[Int, A]
```

```
type DenseMatrix[A] = Matrix[IntMap, A]
```

- С помощью них можно создавать типы желаемого рода

ТИПЫ-КОМПОНЕНТЫ

```
trait Item {  
  type Key  
  type Value  
  def key: Key  
  def value: Value  
}
```

- Типы-псевдонимы без конкретного определения называются типами-компонентами, они могут различаться у различных экземпляров родительского типа

ТИПЫ-КОМПОНЕНТЫ

```
trait Coyoneda[A] { self =>
  type Pivot
  val start: Pivot
  val run: Pivot => A

  def value: A = run(start)
}
```

- Мы можем «забыть» конкретный тип-компонент у объекта, но удостовериться, что типы сходятся в реализациях других компонент

УТОЧНЕНИЕ

```
trait Container {  
  type Item  
  def values: List[Item]  
}  
  
val ints = new Container{  
  type Item = Int  
  val values = List(1, 2, 3)  
}  
  
ints.values // List[Int]
```

- Компилятор будет стараться запомнить как можно больше информации о типах

УТОЧНЕНИЕ

```
trait Container {  
  type Item  
  def values: List[Item]  
}  
  
val ints: Container {type Item = Int} =  
  new Container {  
    type Item = Int  
    val values = List(1, 2, 3)  
  }  
  
val xs: List[Int] = ints.values // List[Int]
```

- Это происходит благодаря структурным типам и уточнениям

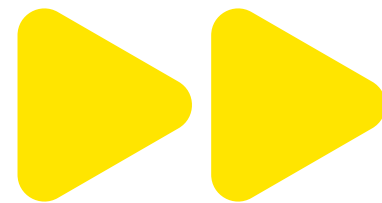
УТОЧНЕНИЕ

```
trait Container {  
  type Item  
  def values: List[Item]  
}
```

```
object Container{  
  type Aux[I] = Container{type Item = I}  
}
```

- Мы можем поставить в соответствие уточнённому типу параметрический псевдоним. Этот приём называется Aux-паттерн

**В этом разделе мы изучили
типы - псевдонимы
и типы - компоненты**



**В следующем
неявные параметры**