Dark

Syntax Sugars

Syntax Sugars

Single argument methods

- example: apply methods of Try, Option , Future, etc

```scala
def singleArgMethod(arg: Int): String =
  s"$arg little ducks..."

val description = singleArgMethod {
  // write some complex code
  42
}
```

```scala
val aTryInstance = Try {  // Try.apply
  throw new RuntimeException
}

List(1,2,3).map { x =>
  // code block for the lambda
  x + 1
}
```

Single abstract method pattern

- example: Java Runnables

```scala
trait Action {
  def act(x: Int): Int
}

val aFunkyInstance: Action = (x: Int) => x + 1
```

Syntax Sugars

Methods ending in : are special

- example: lists

```
1 :: 2 :: 3 :: List(4, 5)
List(4,5).::(3).::(2).::(1) // equivalent
```

Multi-word identifiers

- example: HTTP libraries

```
object `Content-Type` {
  val `application/json` = "application/json"
}
```

Syntax Sugars

Infix types

```scala
infix trait -->[A, B]
val towards: Int --> String = ???
```

Update

- example: arrays

```scala
val anArray = Array(1,2,3)
anArray(2) = 7 // rewritten to anArray.update(2, 7)
```

Mutable fields

```scala
class Mutable {
  private var internalMember: Int = 0
  def member = internalMember // "getter"
  def member_=(value: Int): Unit =
    internalMember = value // "setter"
}

val aMutableContainer = new Mutable
aMutableContainer.member = 42
```

Scala rocks