МЕТОДЫ

МЕТОДЫ

```
def plusOne(number: Int): Int = number + 1
```

Имя метода	Имя параметра	Тип параметра	Возвращаемый тип	Тело метода

```
plusOne(4) // 5
```

МЕТОДЫ

```
def plusOne(number: Int) = number + 1
```

- Тип результата иногда можно опускать
- Scala попробует вывести тип сама

```
plusOne(4) // 5
```

МЕТОДЫ

```
def plusOne(x: Int, y: Int, z: Int): Int = x + y + z
```

- У метода может быть несколько параметров

```
plusOne(1, 2, 3) // 6
```

МЕТОДЫ

```
def plusOne(x: Int, y: Int, z: Int): Int = x + y + z
```

- Параметры можно группировать в список

```
plusOne(1, 2)(3) // 6
```

МЕТОДЫ

```
def sixty: Int = 10 * 6
```

- У метода может вообще не быть параметров
- Это можно воспринимать как переменную, которая будет вычисляться каждый раз

```
sixty // 60
```

МЕТОДЫ

```
def plusAndPrint(x: Int, y: Int): Int = {
  val result = x + y
  println(s"$x + $y = $result")
  result
}
```

- Блок вычислений можно взять в фигурные скобки

```
plusAndPrint(2, 3) //prints 2 + 3 = 5 , returns 5
```

МЕТОДЫ

```
def plusAndPrint(x: Int, y: Int): Unit = {
  val result = x + y
  println(s"$x + $y = $result")
}
```

- Методы, не возвращающие ничего, должны возвращать тип **Unit**

```
plusAndPrint(2, 3) //prints 2 + 3 = 5
```

ВЛОЖЕННЫЕ МЕТОДЫ

```
def plusMul(q: Int, x: Int, y: Int): Int = {
  def mul(u: Int) = q * u
  mul(x) + mul(y)
}
```

- Внутри любых блоков можно объявлять методы
- Внутри тела метода вы можете ссылаться на его параметры

```
plusMul(10, 2, 3) // 50
```

Мы изучили методы

В следующем разделе продолжим их изучение