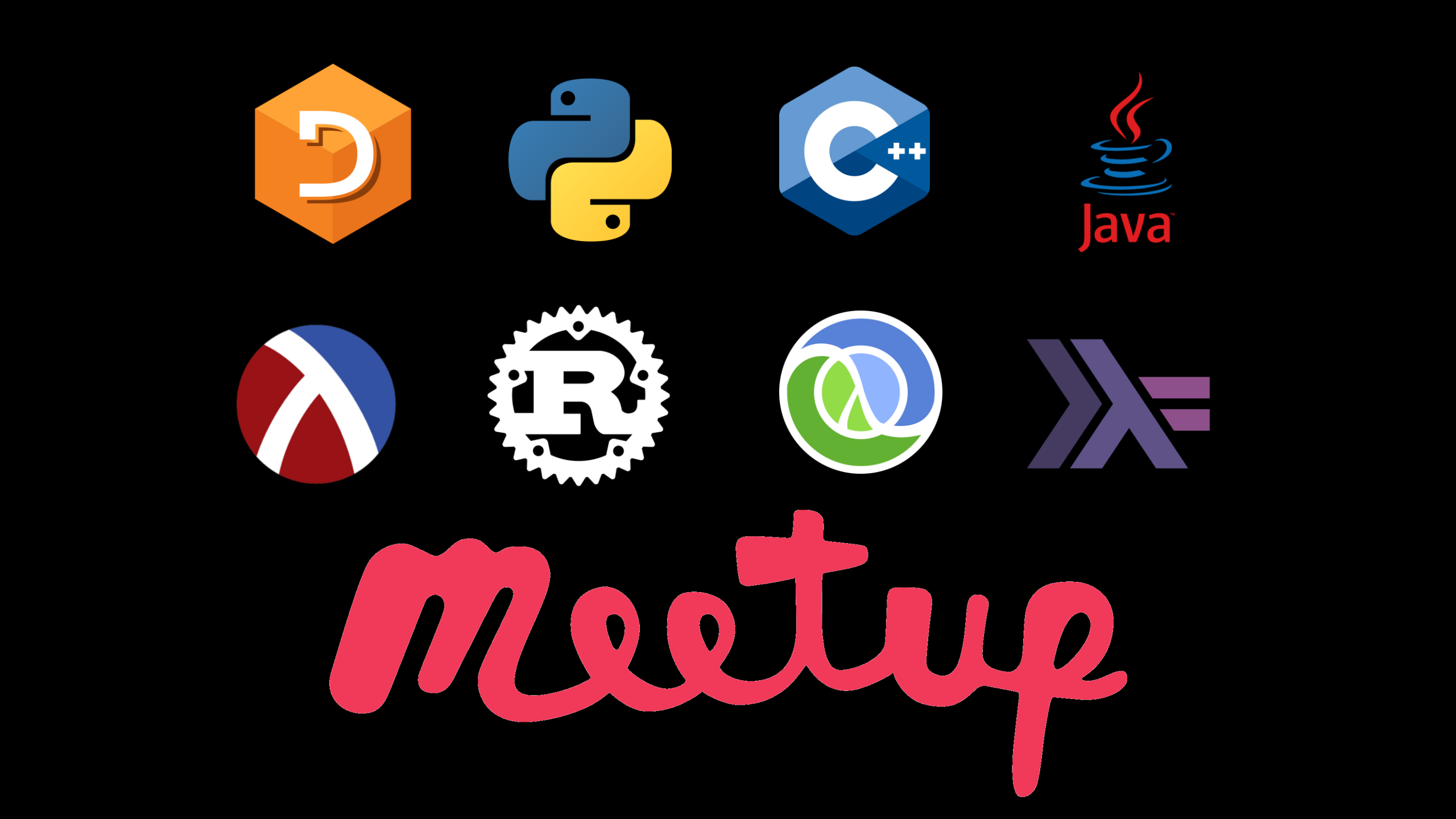
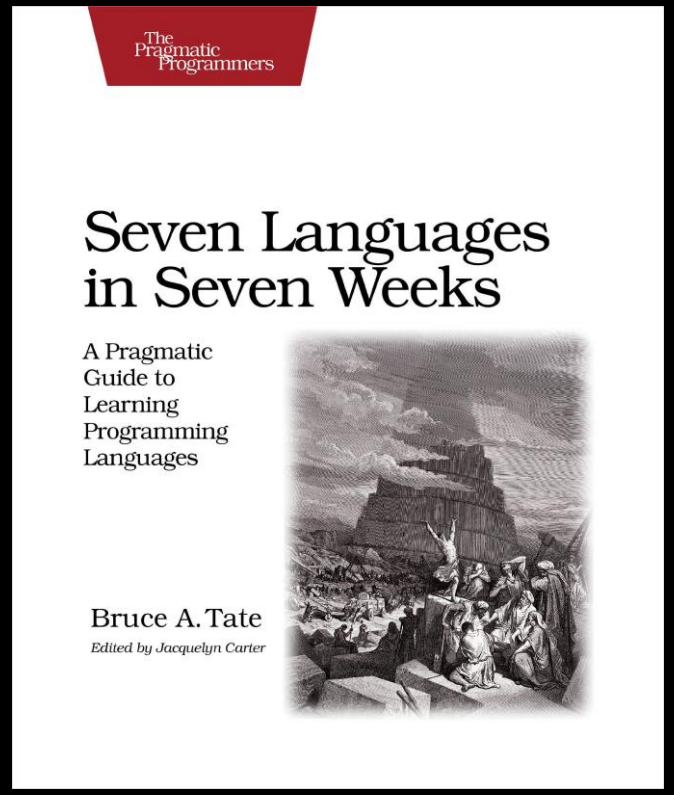

C++
Java
meetup

Berlin Code of Conduct



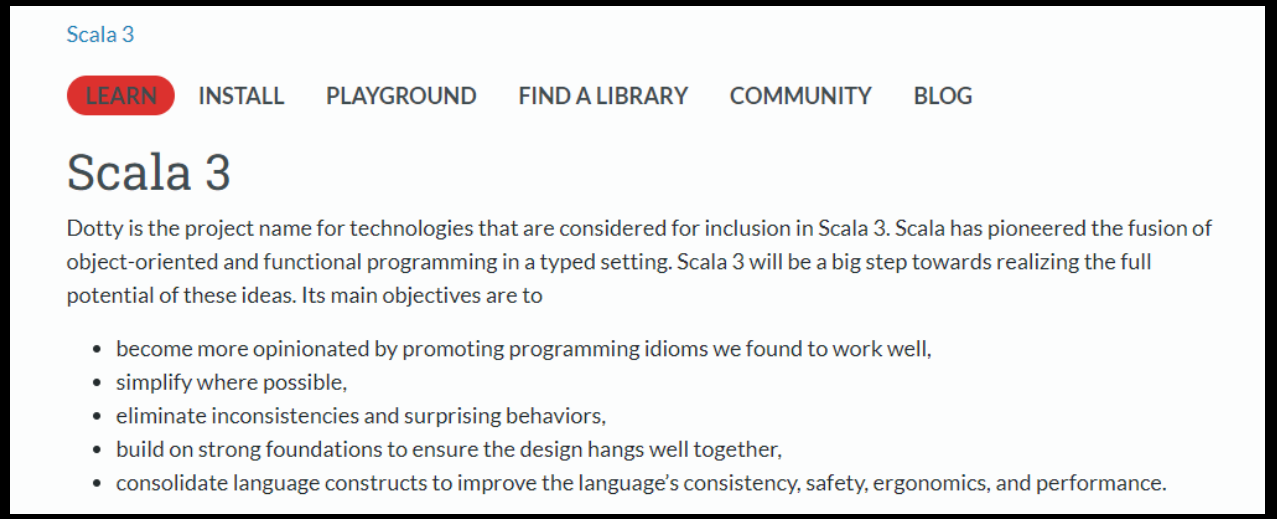
7 Languages in 7 Weeks

Scala

Overall Thoughts

Overall Thoughts

<

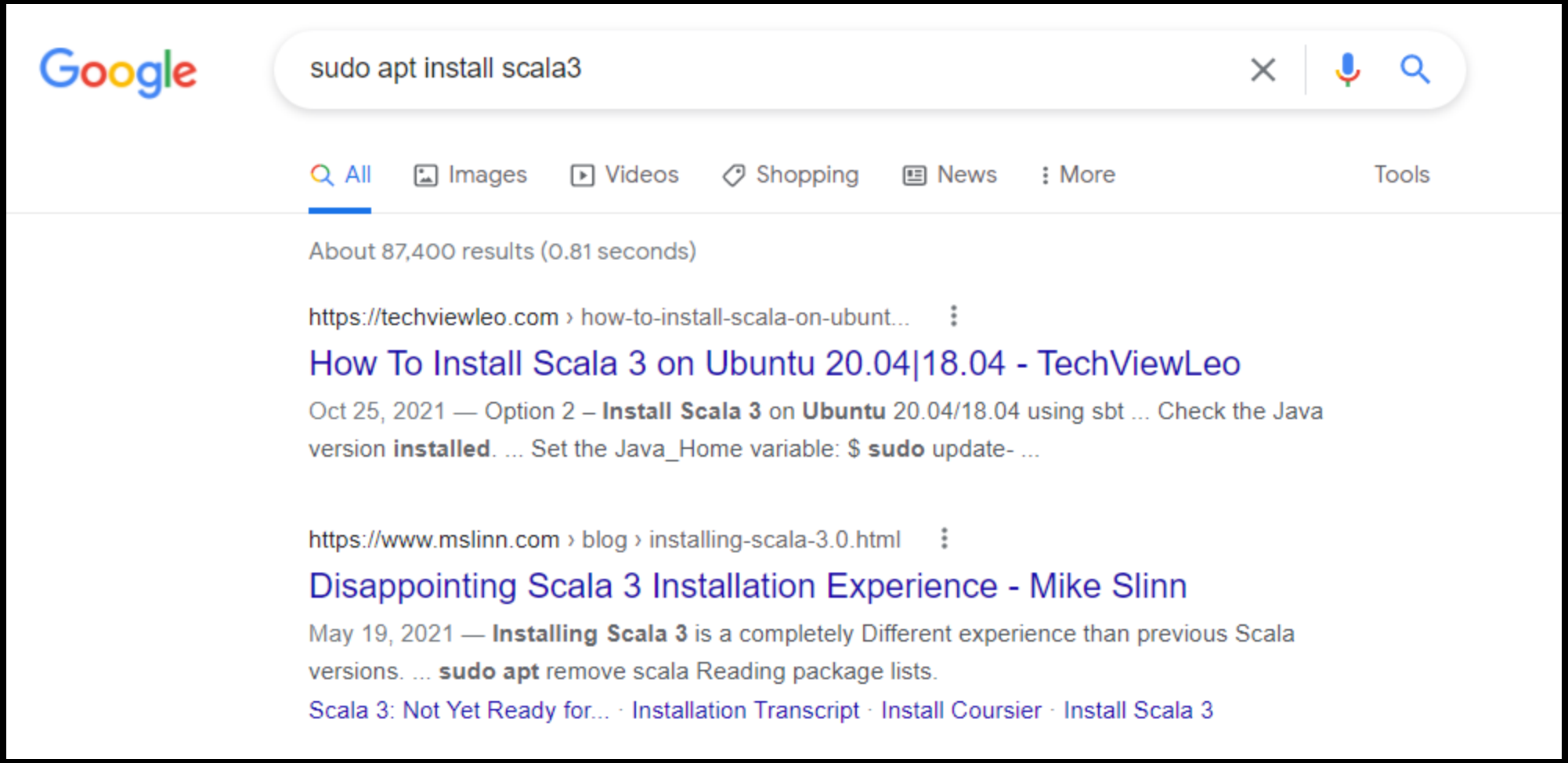

Scala 3
LEARN
INSTALL
PLAYGROUND
FIND A LIBRARY
COMMUNITY
BLOG
Scala 3
Dotty is the project name for technologies that are considered for inclusion in Scala 3. Scala has pioneered the fusion of object-oriented and functional programming in a typed setting. Scala 3 will be a big step towards realizing the full potential of these ideas. Its main objectives are to
become more opinionated by promoting programming idioms we found to work well,
simplify where possible,
eliminate inconsistencies and surprising behaviors,
build on strong foundations to ensure the design hangs well together,
consolidate language constructs to improve the language's consistency, safety, ergonomics, and performance.


Scala
docs – Scala 3
DOCUMENTATION FOR SCALA 3
First Steps
New in Scala 3
An overview of the exciting new features in Scala 3.

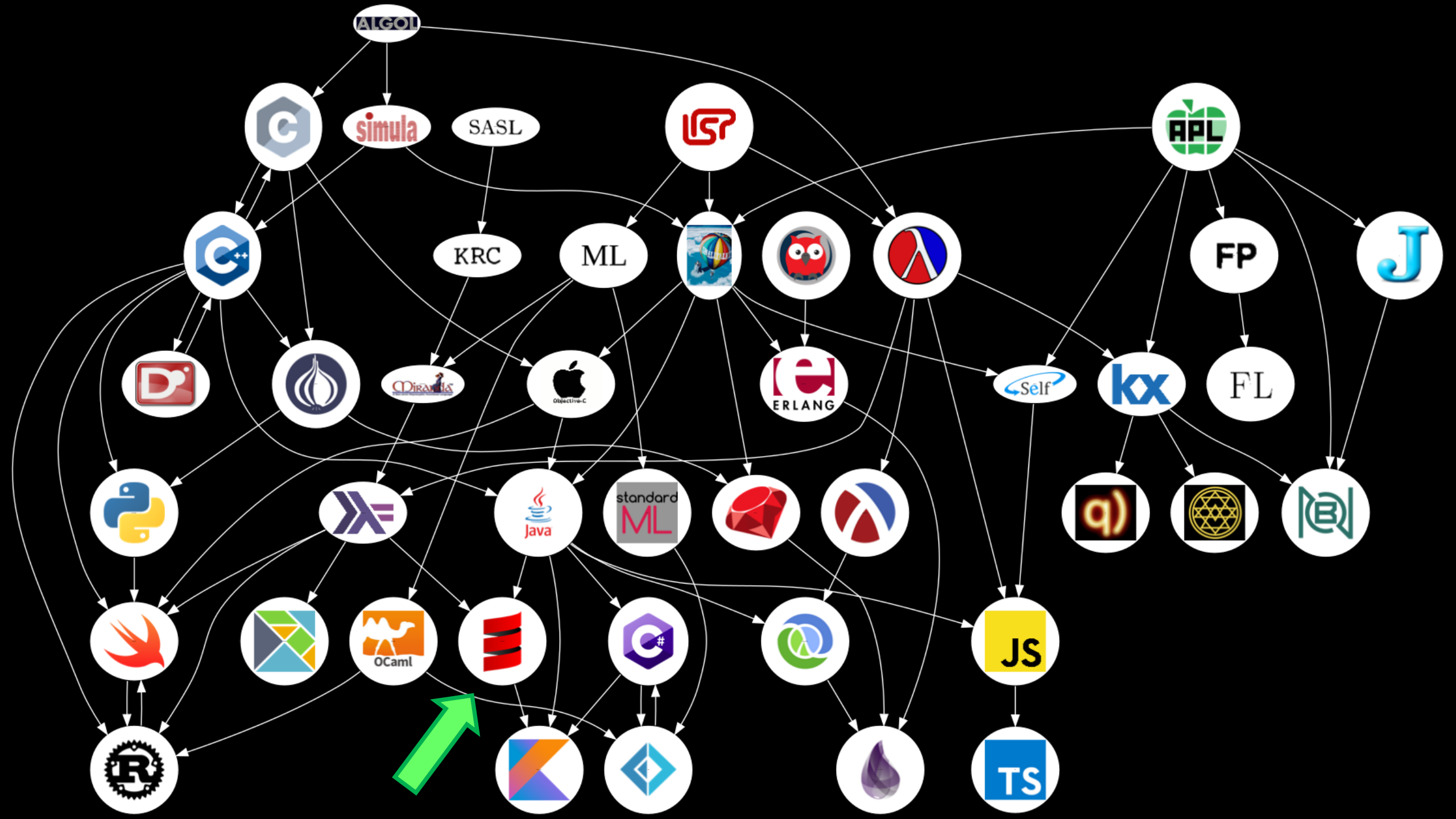

Scala
SCALA 3 IS HERE!
FRIDAY 14 MAY 2021
Anatolii Kmetiuk, LAMP

2.7.7

sudo apt install scala
2.11.12


Google
sudo apt install scala3
All
Images
Videos
Shopping
News
More
Tools
About 87,400 results (0.81 seconds)
https://techviewleo.com › how-to-install-scala-on-ubunt...
How To Install Scala 3 on Ubuntu 20.04|18.04 - TechViewLeo
Oct 25, 2021 — Option 2 – Install Scala 3 on Ubuntu 20.04/18.04 using sbt ... Check the Java version installed. ... Set the Java_Home variable: $ sudo update- ...
https://www.mslinn.com › blog › installing-scala-3.0.html
Disappointing Scala 3 Installation Experience - Mike Slinn
May 19, 2021 — Installing Scala 3 is a completely Different experience than previous Scala versions. ... sudo apt remove scala Reading package lists.
Scala 3: Not Yet Ready for... · Installation Transcript · Install Coursier · Install Scala 3

2.11.12


ALGOL
simula
SASL
KRC
ML
APL
FP
J
FL
Self
kx
ERLANG
Java
standard ML
OCaml
JS
TS

Affinity with Java...

Scala is at least a bridge and maybe more. It offers tight integration into Java, offering a chance for people to protect their investment in many ways:

- Scala runs on the Java virtual machine, so Scala can run side-by-side with existing deployments.
- Scala can use Java libraries directly, so developers can leverage existing frameworks and legacy code.
- Like Java, Scala is statically typed, so the languages share a philosophical bond.
- Scala's syntax is relatively close to Java's, so developers can learn the basics quickly.
- Scala supports both object-oriented and functional programming paradigms, so programmers can gradually learn to apply functional programming ideas to their code.

The buzz around Scala is growing, because Twitter has switched its core message processing from Ruby to Scala. The object-oriented features allow a pretty smooth transition from the Java language, but the ideas that are drawing attention to Scala are the functional programming features. Pure functional languages allow a style of programming that has strong mathematical foundations. A functional language has these characteristics:

- Functional programs are made up of functions.
- A function always returns a value.
- A function, given the same inputs, will return the same values.
- Functional programs avoid changing state or mutating data. Once you've set a value, you have to leave it alone.

Next, notice that these Scala variable declarations start with the val keyword. You can also use the var keyword. val is immutable; var is not. We'll talk more about this later.

```
scala> println("Hello World!")
Hello World!

scala> 1 + 1
res1: Int = 2

scala> (1).+(1)
res2: Int = 2

scala> 5 + 4 * 3
res3: Int = 17

scala> "abc".size
res4: Int = 3

scala> "abc" + 4
res5: String = abc4
```

```
scala> println("Hello World!")
Hello World!

scala> 1 + 1
res1: Int = 2

scala> (1).+(1)
res2: Int = 2

scala> 5 + 4 * 3
res3: Int = 17

scala> "abc".size
res4: Int = 3

scala> "abc" + 4
res5: String = abc4

scala> 4 + "abc"
res6: String = 4abc

scala> 4 + "1.0"
res7: String = 41.0

scala> 4 * "abc"
<console>:12: error: overloaded method value * with alternatives:
  (x: Double)Double <and>
  (x: Float)Float <and>
  (x: Long)Long <and>
  (x: Int)Int <and>
  (x: Char)Int <and>
  (x: Short)Int <and>
  (x: Byte)Int
 cannot be applied to (String)
       4 * "abc"
         ^
```

```
scala> 5 < 6
res9: Boolean = true

scala> 5 != 2
res10: Boolean = true

scala> val a = 1
a: Int = 1

scala> val r = 0 until 10
r: scala.collection.immutable.Range = Range(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)

scala> r.start
res11: Int = 0

scala> r.end
res12: Int = 10
```

```
scala> val r2 = 0 to 10
r2: scala.collection.immutable.Range.Inclusive = Range(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

scala> val me = ("Conor", "Hoekstra")
me: (String, String) = (Conor,Hoekstra)

scala> me._1
res13: String = Conor

scala> me._2
res14: String = Hoekstra
```

Do:

- Write a game that will take a tic-tac-toe board with X, O, and blank characters and detect the winner or whether there is a tie or no winner yet. Use classes where appropriate.

```
def any(arr : Array[Boolean]) : Boolean =
    arr.fold(false) { _ || _ }

def all_equal_to(arr : Array[Char], xo: Char) : Boolean =
    arr.map { _ == xo }.fold(true) { _ && _ }

def winner(board: String, xo: Char) : Boolean = {
    val indices = Array(Array(0,1,2), Array(3,4,5), Array(6,7,8), // rows
                        Array(0,3,6), Array(1,4,7), Array(2,5,8), // columns
                        Array(0,4,8), Array(2,4,6))               // diagonals

    return any(indices.map { is => all_equal_to(is.collect(board), xo) })
}

def tic_tac_toe_status(board: String) : String = {
    if      (winner(board, 'X')) { return "X wins" }
    else if (winner(board, 'O')) { return "O wins" }
    return "No winner yet"
}
```

```haskell
indices = [[0,1,2],[3,4,5],[6,7,8], -- rows
           [0,3,6],[1,4,7],[2,5,8], -- columns
           [0,4,8],[2,4,6]]         -- diagonals

allEqualTo e = and . map (==e)

collect is = map snd
           . filter ((flip elem) is . fst)
           . zipWith (,) [0..]

winner board xo = or
                . map (allEqualTo xo . (flip collect) board)
                $ indices

ticTacToeStatus board
    | winner board 'X' = "X wins"
    | winner board 'O' = "O wins"
    | otherwise        = "No winner yet"
```


```
ticTacToeStatus ← {
      i ← 1 5 9 3 5 7,(,⍉3 3⍴⍳9),⍳9
      m ← 1,⍤⍤1⌿∧/'XO'∘.=8 3⍴⍵[i]
      ' wins',⍤⍤1⌿⊃m/'X' 'O' 'No one'
}
```

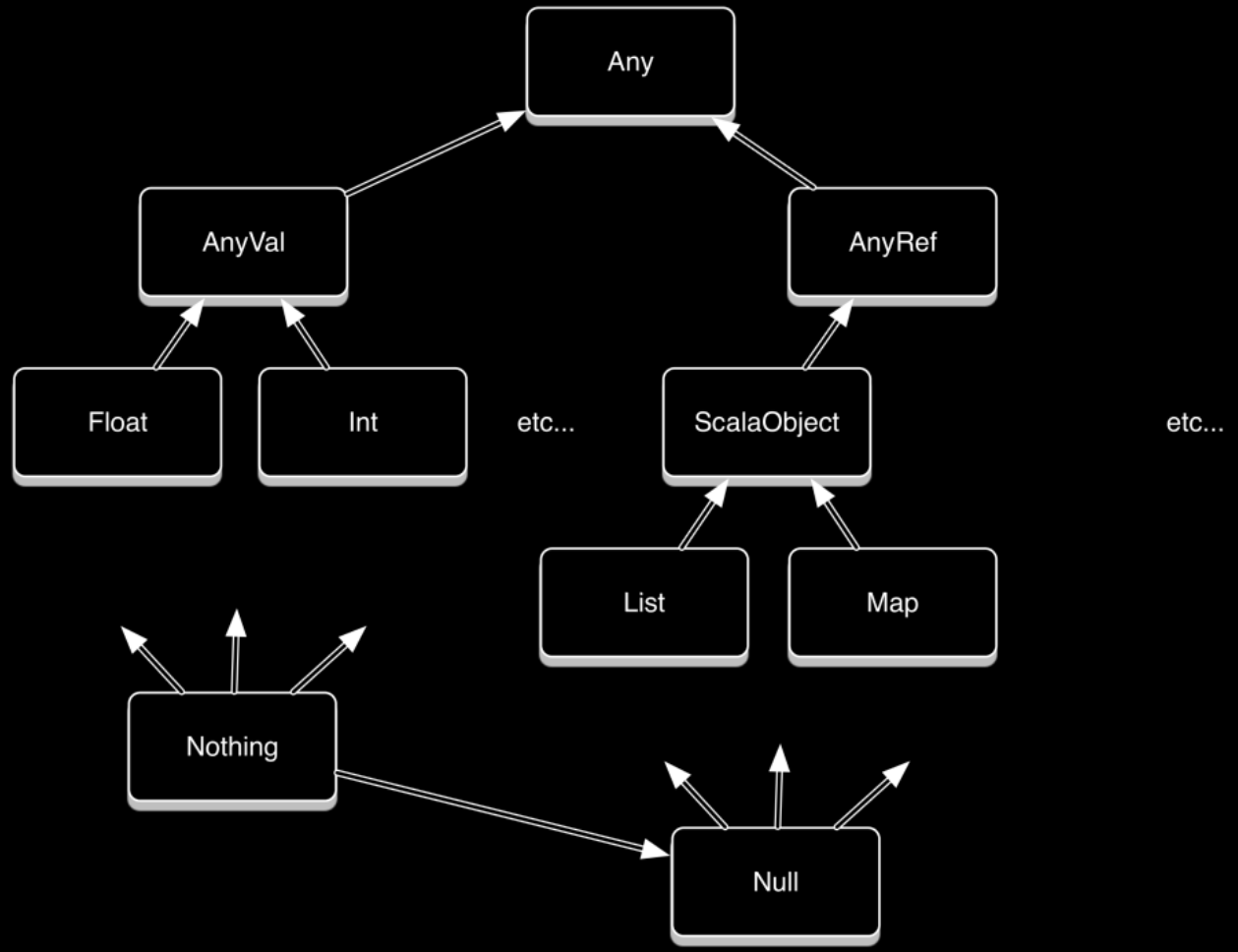


Figure 5.1: Any and Nothing

```
scala> Set(1,1,1)
res0: scala.collection.immutable.Set[Int] = Set(1)

scala> val langs = List("apl", "bqn", "j", "k")
langs: List[String] = List(apl, bqn, j, k)

scala> langs.foreach { l => println(l) }
apl
bqn
j
k
```

```
scala> langs.map { _.size }
res2: List[Int] = List(3, 3, 1, 1)

scala> langs.count { _.size == 1 }
res3: Int = 2

scala> (0 /: langs.map { _.size }) { _ + _ }
res4: Int = 8
```

Do:

- Use foldLeft to compute the total size of a list of strings.
- Write a Censor trait with a method that will replace the curse words *Shoot* and *Darn* with *Pucky* and *Beans* alternatives. Use a map to store the curse words and their alternatives.

```
def total_size_of_strings(lst: List[String]) : Int =
    lst.map { _.size }.sum
```

```
totalSize = sum . map length
```


APL

+/≠¨

```
trait Bleep {
    val swear_words = Map("shoot" -> "pucky",
                          "darn"  -> "beans")
    def clean() : Sentence
}

class Sentence(var words: List[String]) extends Bleep {
    def clean() : Sentence =
        new Sentence(words.map { w => swear_words.getOrElse(w, w) })
}
```

```
scala> val langs = List("apl", "bqn", "j", "k", "q")
langs: List[String] = List(apl, bqn, j, k, q)

scala> val hasQ = """.*q.*""".r
hasQ: scala.util.matching.Regex = .*q.*

scala> langs.map { hasQ.findFirstIn(_) }
res0: List[Option[String]] = List(None, Some(bqn), None, None, Some(q))
```

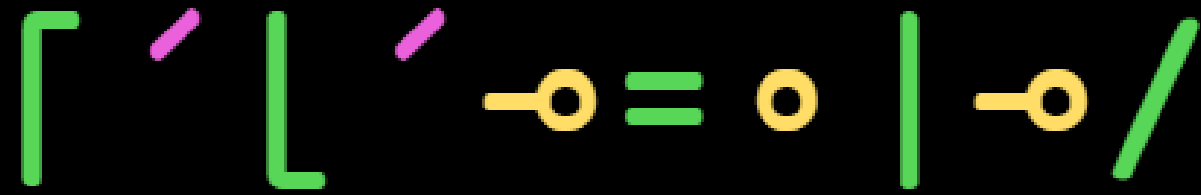
2239. Find Closest Number to Zero

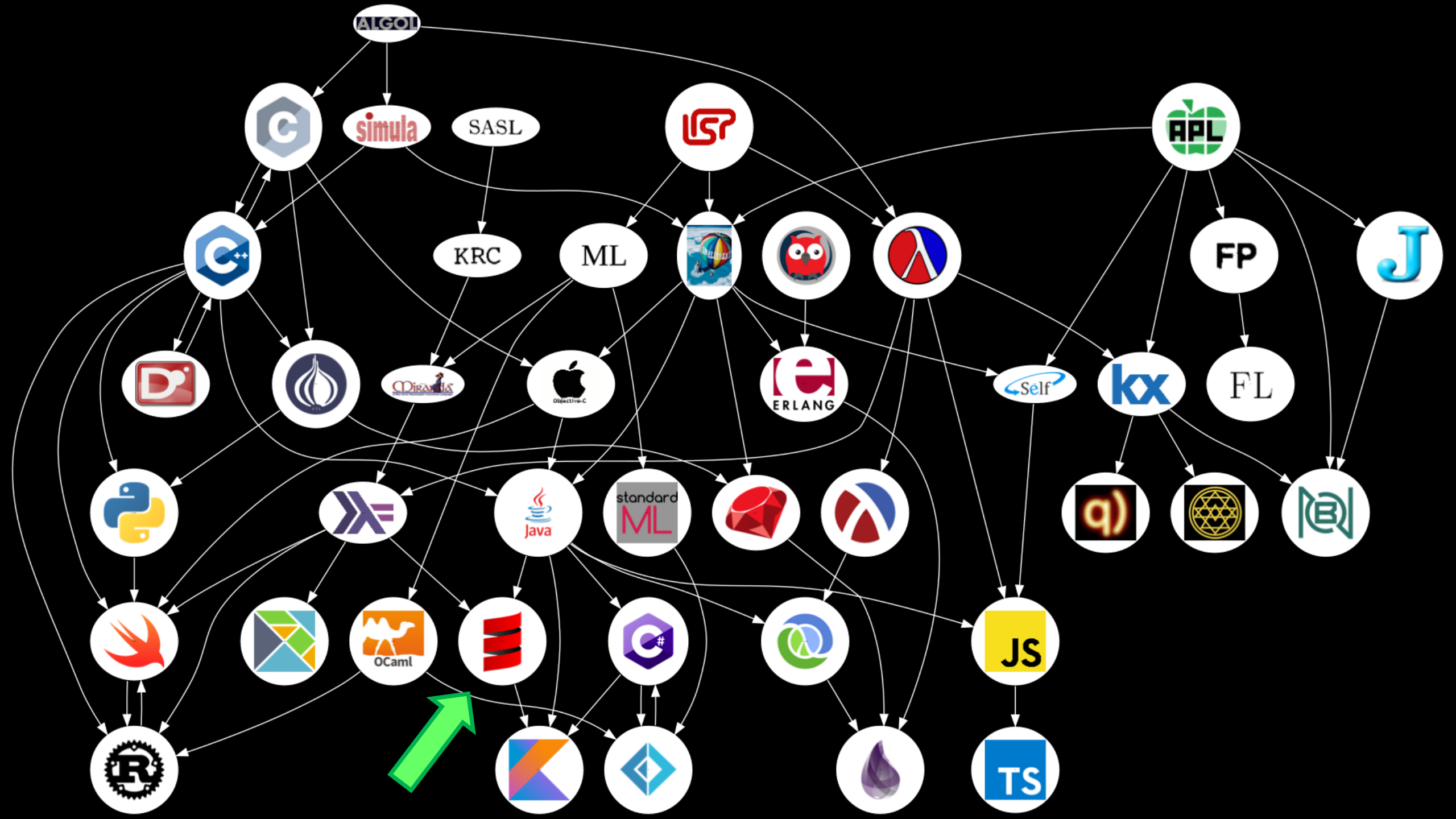
Given an integer array `nums` of size `n`, return *the number with the value* ***closest*** *to* `0` *in* `nums`. If there are multiple answers, return *the number with the* ***largest*** *value*.

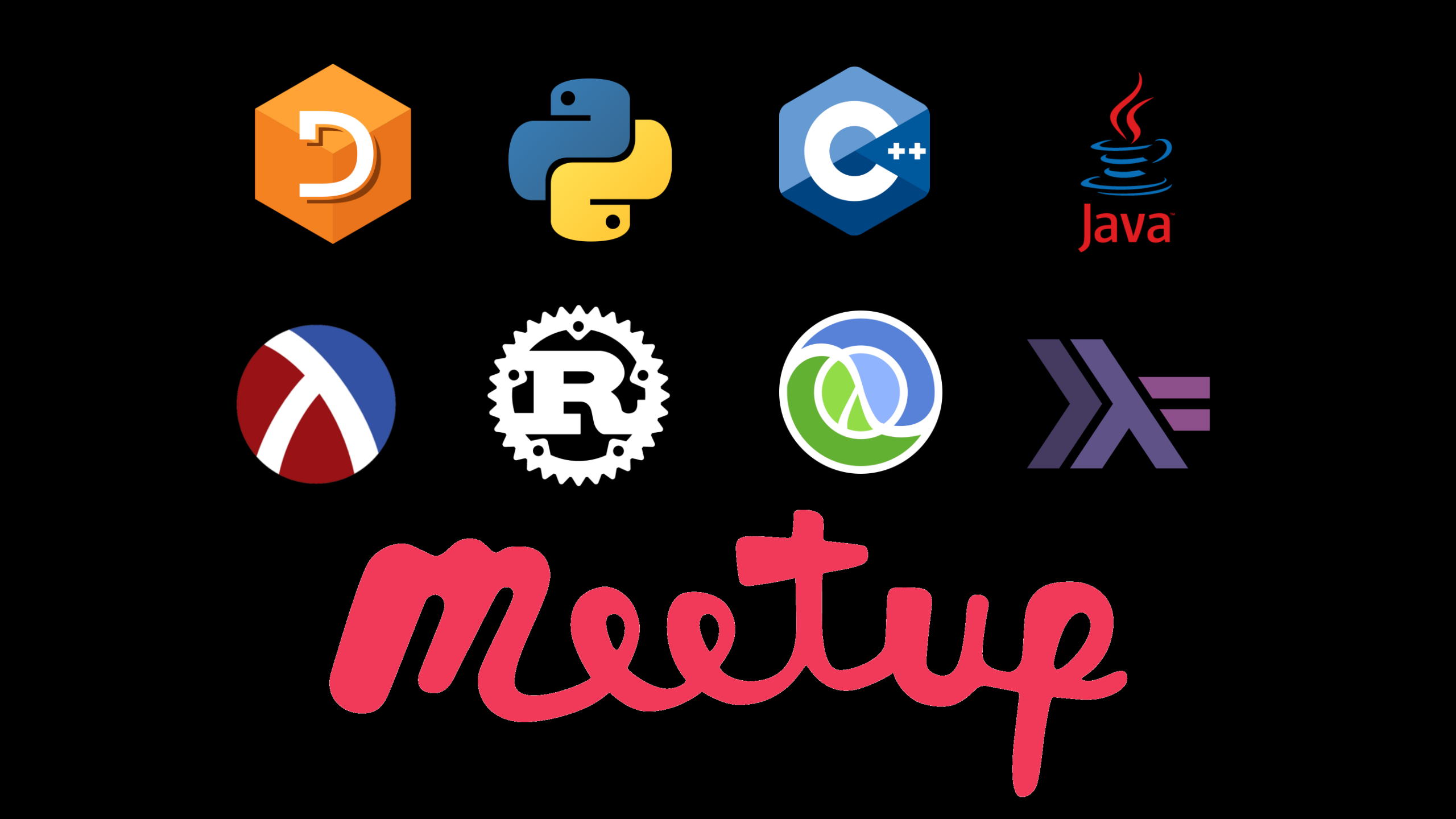
Input: nums = [2,-1,1]
Output: 1
Explanation: 1 and -1 are both the closest numbers to 0, so 1 being larger is returned.

```scala
// Bad: ITM
def findClosestNumber_(nums: Array[Int]): Int = {
    var res = nums(0)
    for (num <- nums) {
        if ((num.abs < res.abs) ||
            (num.abs == res.abs && num > res)) {
            res = num
        }
    }
    return res
}
```

```scala
// Good :)
def findClosestNumber(nums: Array[Int]): Int = {
    val lo = nums.map { _.abs }.min
    return nums.filter { _.abs == lo }.max
}
```




ALGOL
simula
SASL
KRC
ML
APL
FP
J
FL
Self
kx
ERLANG
Objective-C
Java
standard ML
OCaml
JS
TS


C++
Java
meetup