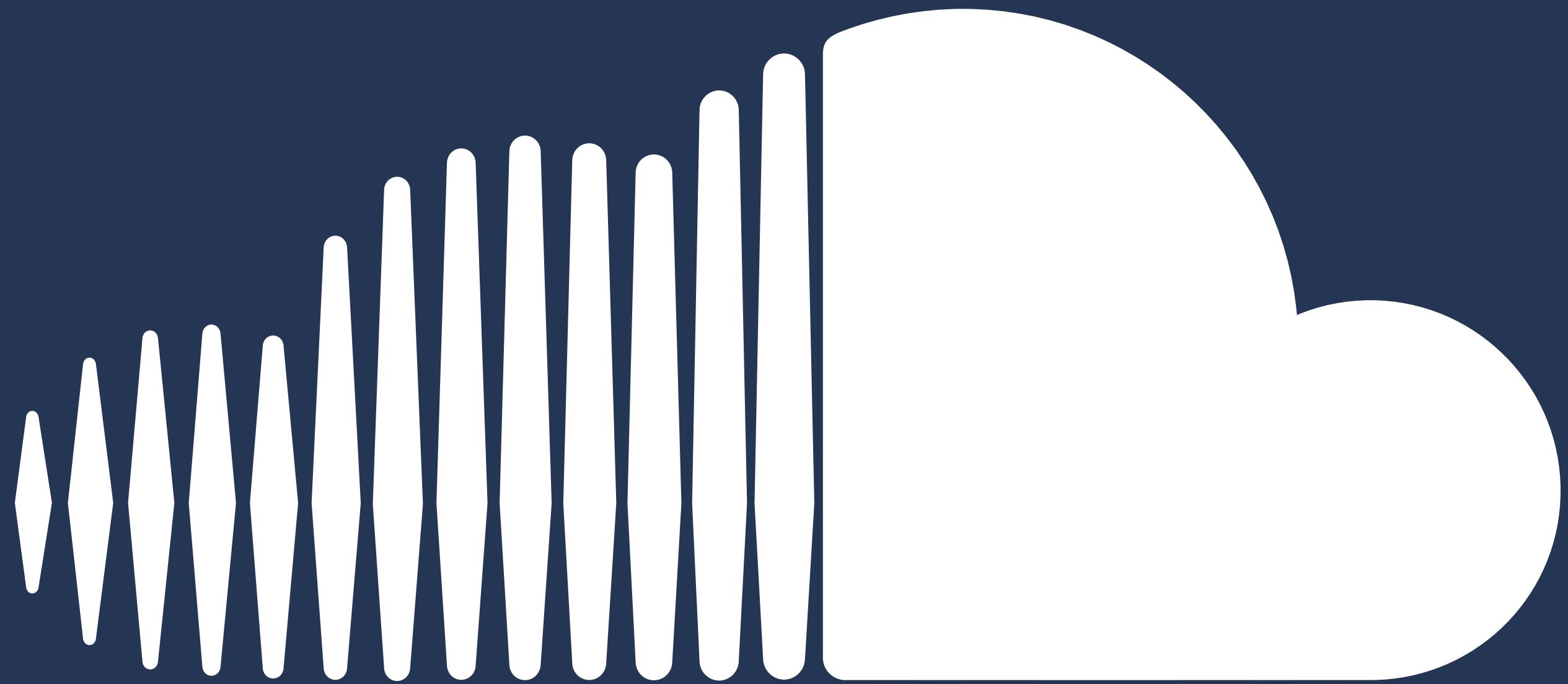


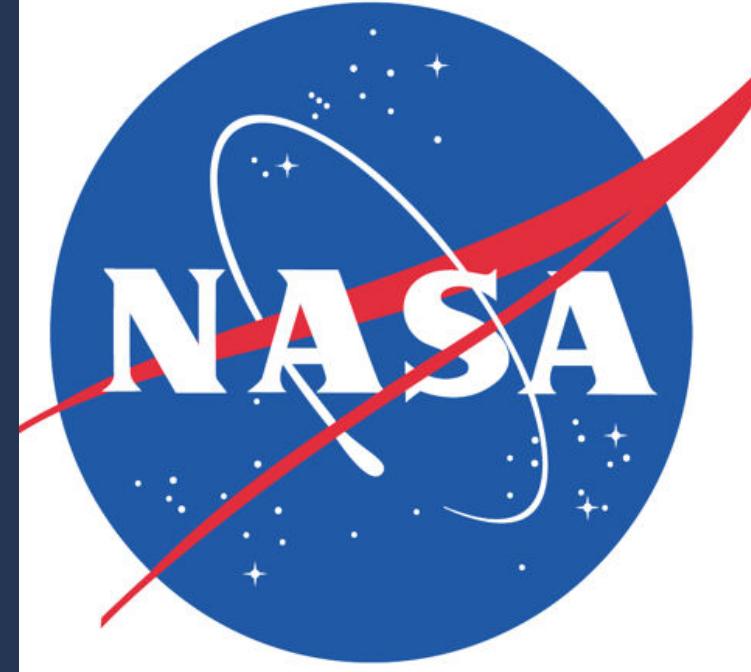
Diecio

My name is @folone

<https://github.com/folone/scalaworld>



SOUND CLOUD





- 12 hours uploaded every minute
- ~35k listening years every month
- >100M tracks
- ~300M monthly active users



Search for topics

Groups

unexpected challenges

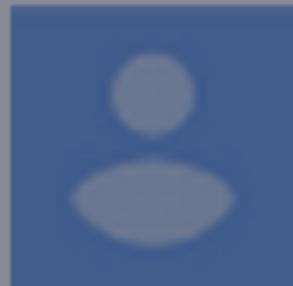
POST REPLY

Data Center & Network Engineering ›

Maintenance postponed by rapper diss war. :-/

3 posts by 1 author

8+1



ryan



Other recipients: tkau...@batblue.com, bpa...@batblue.co

hands on



TL;what now?^o



^o <http://typelevel.org/blog/2014/09/02/typelevel-scala.html>

why we love



Roll your own Scala

A scenic view of a rocky mountain slope with sparse vegetation. The foreground shows dark, jagged rocks and patches of dry grass. In the middle ground, the slope rises with more rocks and some green shrubs. The background consists of rolling hills under a clear blue sky.

I think that's pretty neat,
but I can also understand why
almost everyone else would find it
horrifying.

-- Travis Brown



```
scalaVersion := "2.11.7"
scalaOrganization := "org.typelevel"
```

- git clone git@github.com:folone/scalaworld.git && \
cd scalaworld && \
sbt -Dsbt.boot.properties=sbt.boot.properties

```
[repositories]
maven-central
```



[some] Features

- Type lambdas
- @implicitAmbiguous (coming to 2.12 #4673)
- Singleton types
- -Zirrefutable-generator-patterns
- Nifties

Type lambdas

```
trait Functor[F[_]] {  
    def map[A, B](fa: F[A])(fn: A => B): F[B]  
}
```

```
trait LeftFunctor[R] extends  
    Functor[({type U[x] = Either[x, R]})#U]
```

```
trait RightFunctor[L] extends  
    Functor[[y] => Either[L, y]]
```

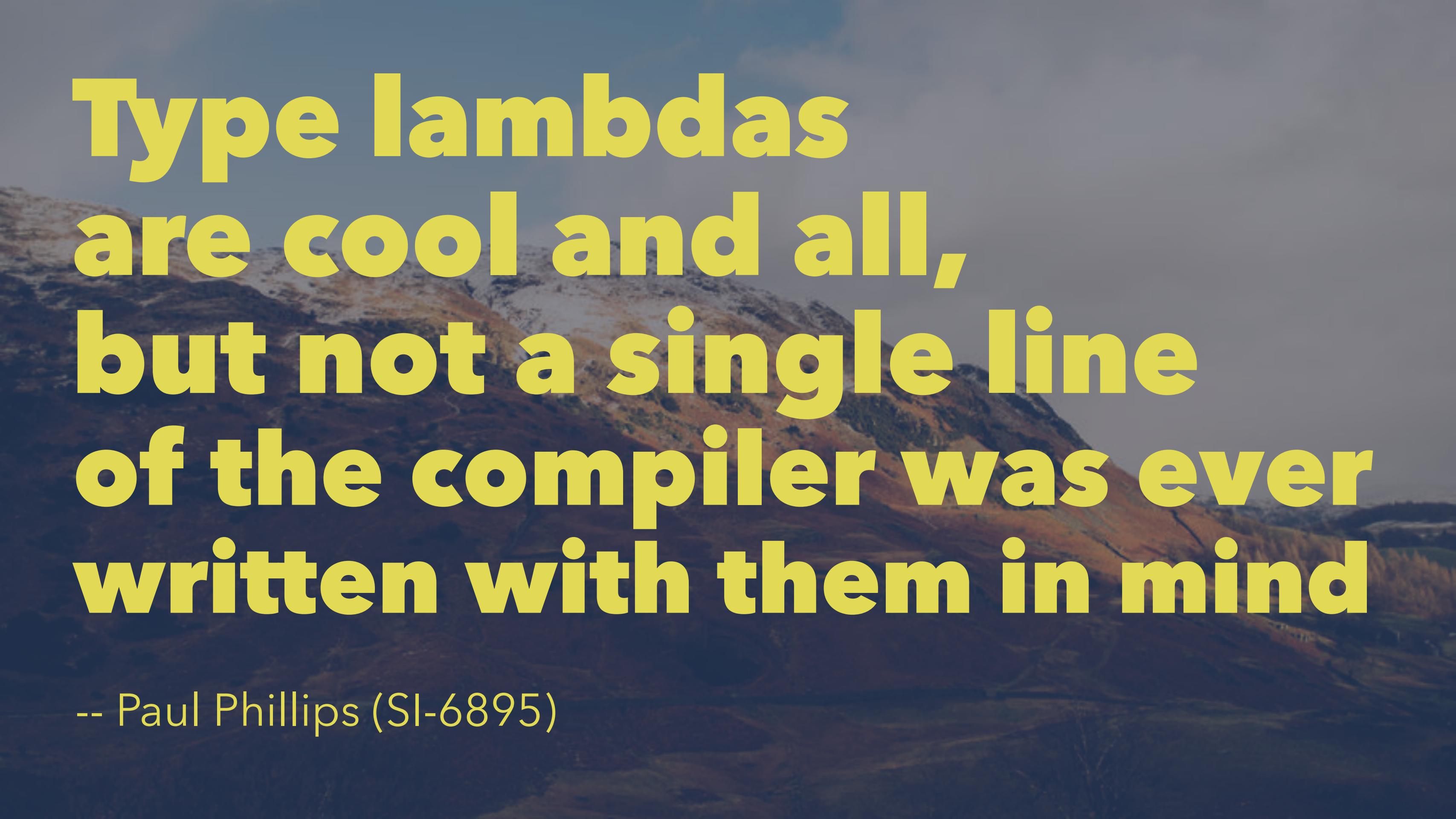
Type lambdas

[x] => (x, x)

[x, y] => (x, Int) => y

[x[_]] => x[Double]

[+x, -y] => Function1[y, x]



Type lambdas
are cool and all,
but not a single line
of the compiler was ever
written with them in mind

-- Paul Phillips (SI-6895)

@implicitAmbiguous¹

```
// Encoding for "A is not a subtype of B"
trait <:!<[A, B]

// Uses ambiguity to rule out the cases we're trying to exclude
implicit def nsub[A, B] : A <:!< B = null
@typelevel.annotation.implicitAmbiguous("Returning ${B} is forbidden.")
implicit def nsubAmbig1[A, B >: A] : A <:!< B = null
implicit def nsubAmbig2[A, B >: A] : A <:!< B = null

// Type alias for context bound
type |¬|[T] = {
  type λ[U] = U <:!< T
}
```

¹ <https://gist.github.com/milessabin/c9f8befa932d98dcc7a4>

@implicitAmbiguous

```
def foo[T, R : |~| [Unit]#λ](t: T)(f: T => R) = f(t)
```

```
foo(23)(_ + 1) // OK
```

```
foo(23)(println) // Doesn't compile: "Returning Unit is forbidden."
```

Singleton types²

```
trait Assoc[K] { type V ; val v: V }
```

```
def mkAssoc[K, V0](k: K, v0: V0): Assoc[k.type] { type V = V0 } =  
  new Assoc[k.type] { type V = V0 ; val v = v0 }  
def lookup[K](k: K)(implicit a: Assoc[k.type]): a.V = a.v
```

² Requires –Xexperimental flag.

Singleton types

```
implicit def firstAssoc = mkAssoc(1, "Panda!")
//> firstAssoc : Assoc[Int(1)]{type V = String}
implicit def secondAssoc = mkAssoc(2, 2.0)
//> secondAssoc : Assoc[Int(2)]{type V = Double}
```

```
implicit def ageAssoc = mkAssoc("Age", 3)
//> ageAssoc : Assoc[String("Age")]{type V = Int}
implicit def nmAssoc = mkAssoc("Name", "Jane")
//> nmAssoc : Assoc[String("Name")]{type V = String}
```

Singleton types

```
lookup(1)
// > res1: String = Panda!
lookup(2)
// > res2: Double = 2.0
lookup("Age")
// > res3: Int = 3
lookup("Name")
// > res4: String = Jane
```

Irrefutable generator patterns

```
for {  
    (x, _) <- Option((1, 2))  
} yield x
```

Desugars to

```
Some(scala.Tuple2(1, 2))
  .withFilter(((check$ifrefutable$1) =>
    check$ifrefutable$1: @scala.unchecked match {
      case scala.Tuple2((x @ _), _) => true
      case _ => false
    })).map(((x$1) => x$1: @scala.unchecked match {
      case scala.Tuple2((x @ _), _) => x
    }))
```

With irrefutable patterns

```
Some(scala.Tuple2(1, 2)).map(((x$1) => x$1 match {  
    case scala.Tuple2((x @ _), _) => x  
} ))
```

Nifties

```
def fib(n: Int) = {  
  def fib'(n: Int, next: Int, £: Int): Int =  
    n match {  
      case 0 => £  
      case _ => fib'(n - 1, next + £, next)  
    }  
  fib'(n, 1, 0)  
}  
  
val £': Byte = 127z
```



vision

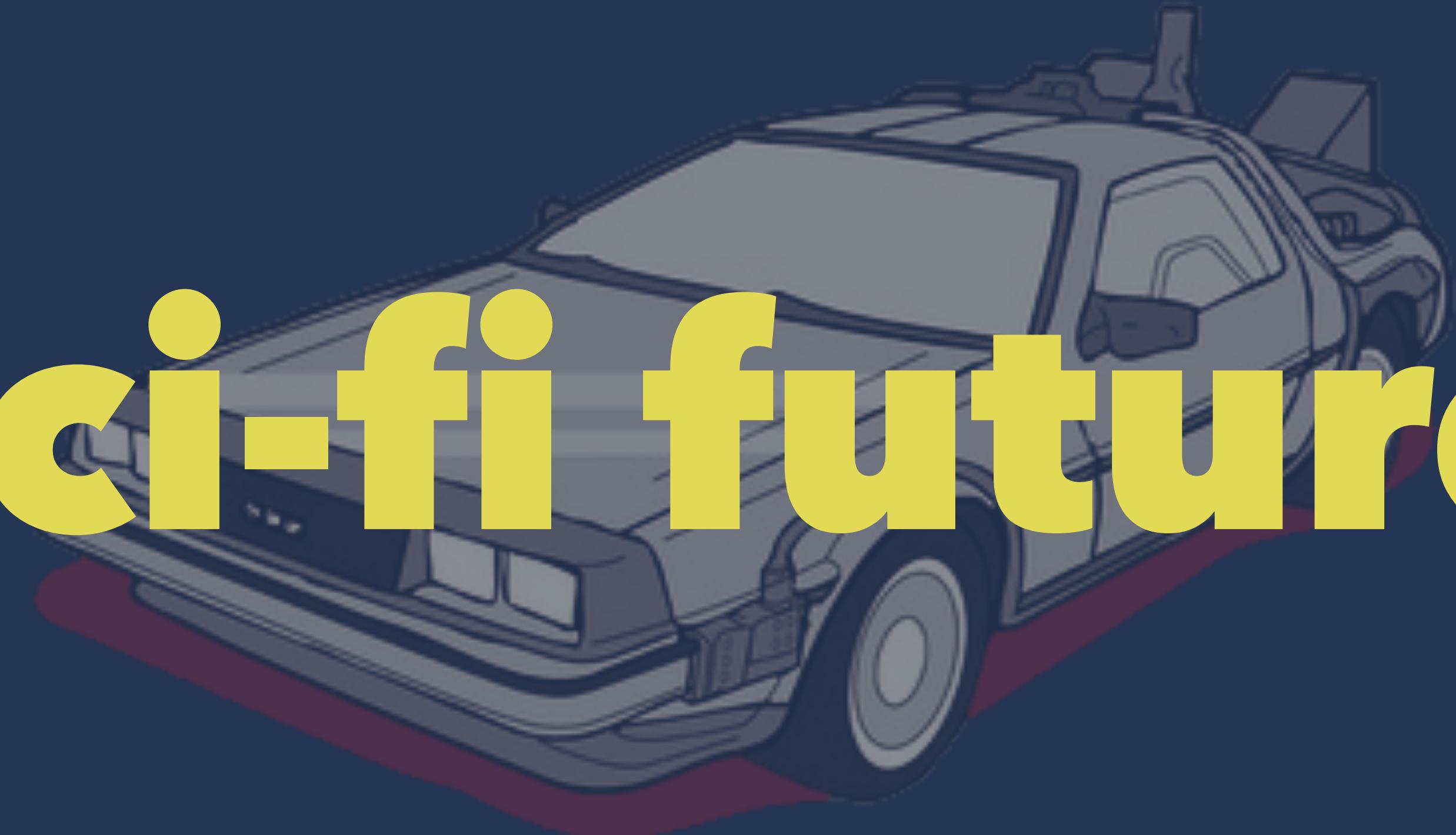


Low-hanging fruits

- converting partest tests to junit
- documentation
- Reporting bugs
- Fixing bugs
- Backporting changes from typesafe scala



sci-fi future

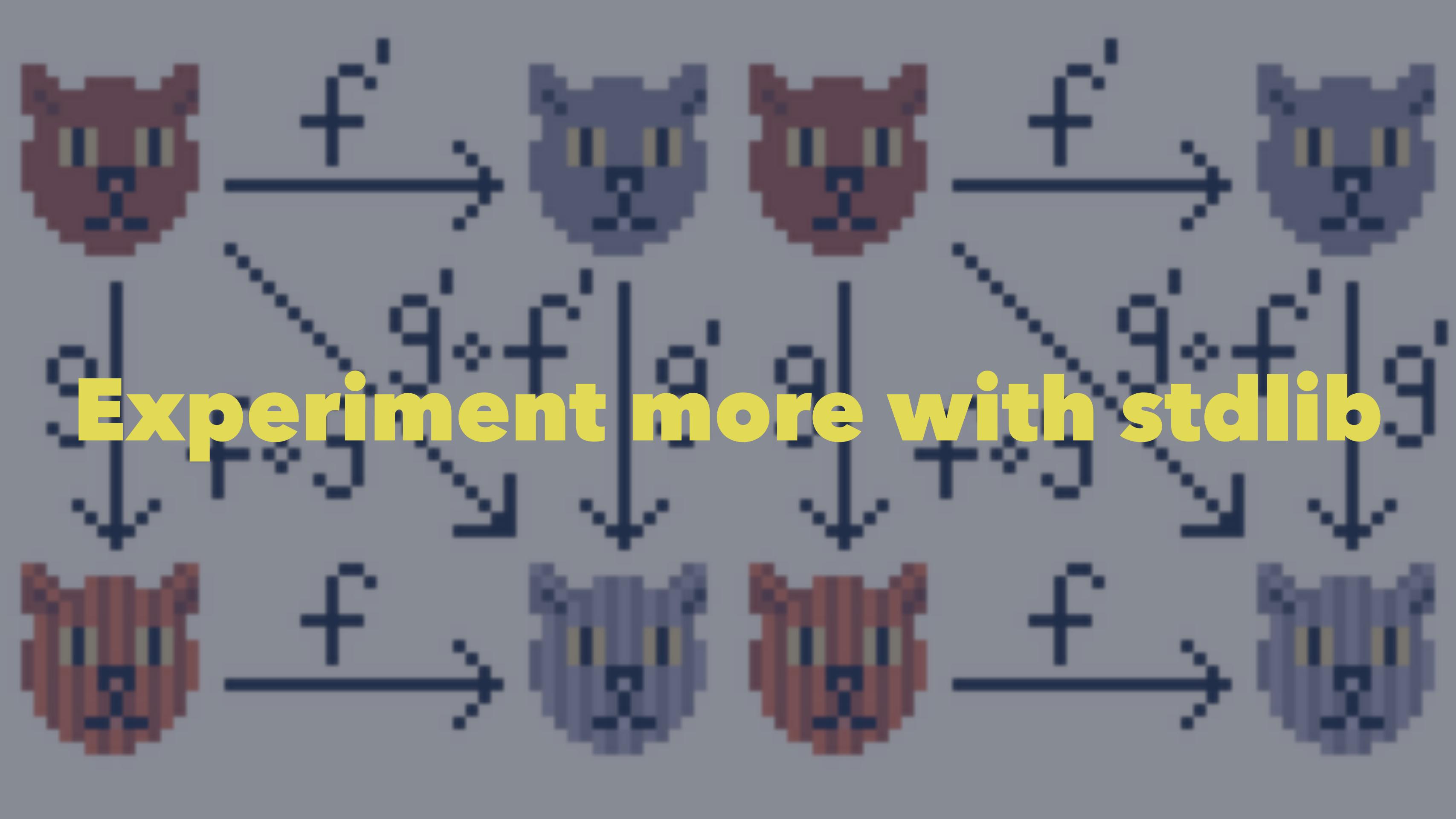


$$x_0 : A_0, x_1 : A_1, \dots, x_{n-1} : A_{n-1} \vdash t : B$$

Refinement types

$$\frac{\Gamma, x:A \vdash B : \text{Set}}{\Gamma \vdash \sum x:A, B : \text{Set}}$$

$$\sum_{\substack{t, T : \text{Set} \rightarrow \text{Set} \\ x, y : \text{Set}}} T(x \rightarrow y) \rightarrow (t_0 x \rightarrow t_0 y)$$



Experiment more with stdlib

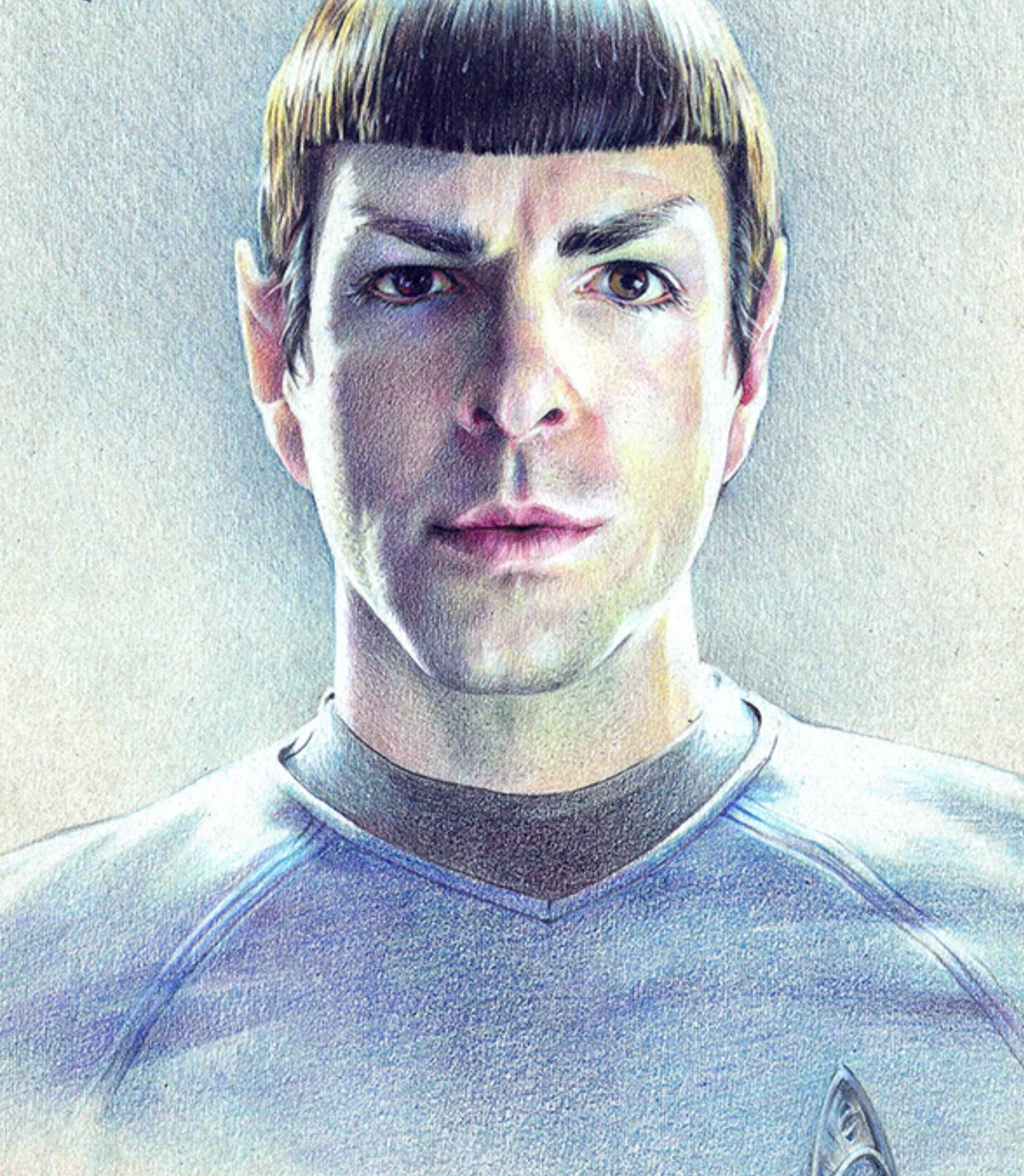


3. fish /Users/haoyi (java)

```
haoyi-mbp:~ haoyi$ ~/amm
Loading...
Welcome to the Ammonite Repl 0.4.6
(Scala 2.11.7 Java 1.8.0_25)
@ List(Seq(Seq("mgg", "mgg", "lols"), Seq("mgg", "mgg")), Seq(Seq("ggx", "ggx"), Seq("ggx", "ggx", "wt
  fx"))) // pretty printing
res0: List[Seq[Seq[String]]] = List(
  List(List("mgg", "mgg", "lols"), List("mgg", "mgg")),
  List(List("ggx", "ggx"), List("ggx", "ggx", "wtfx")))
@
```

Integrating alternative repl

```
@ import scalatags.Text.all._
import scalatags.Text.all._
@ a("omg", href:="www.google.com").render
res3: String = """
<a href="www.google.com">omg</a>
"""
@
```



Rethinking the role of implicits³

³ <https://github.com/typelevel/scala/issues/28 -- @implicitWeight>

hands-on

```
scalaVersion      := "2.11.7"  
scalaOrganization := "org.typelevel"
```

hands-on

- sbt -Dsbt.boot.properties=sbt.boot.properties

[repositories]

maven-central

A wide-angle photograph of a rural landscape. In the foreground, there's a field of dry, golden-brown grass. A small, dark, irregular patch of ground with some low-lying red and orange vegetation is visible in the lower-left corner. The middle ground consists of rolling hills covered in similar dry grass. In the background, a range of mountains is visible under a sky filled with heavy, grey clouds.

What do I do once I check out the repo?



sbt build vs ant
build

The background of the image is a photograph of a rural landscape. In the foreground, there are rolling hills covered in dry, golden-brown grass. Some patches of reddish-orange vegetation are visible on the lower slopes. In the middle ground, a small stream or river winds its way through the terrain. The background features a range of mountains under a dark, overcast sky.

sbt test vs ant test
./tools/partest-ack



sbt publish vs ant
publish-local-opt

A scenic landscape featuring rolling hills covered in dry, golden-brown grass. In the background, there are more hills and mountains under a dark, overcast sky.

If all the things have
failed, ant all. clean

A scenic view of a coastal town. In the foreground, there's a dark stone wall and a black metal fence. Beyond the fence, a large evergreen tree stands prominently. In the background, there's a building with a tiled roof and several chimneys. The sky is blue with some white clouds.

To summarize

A photograph of a person sitting on a rocky, grassy hillside. They are wearing a grey hoodie, blue jeans, and grey gloves. A camouflage-patterned backpack is strapped to their back. They are looking down at a small device in their hands. The background consists of a steep, rocky hillside covered in sparse vegetation.

Danke!