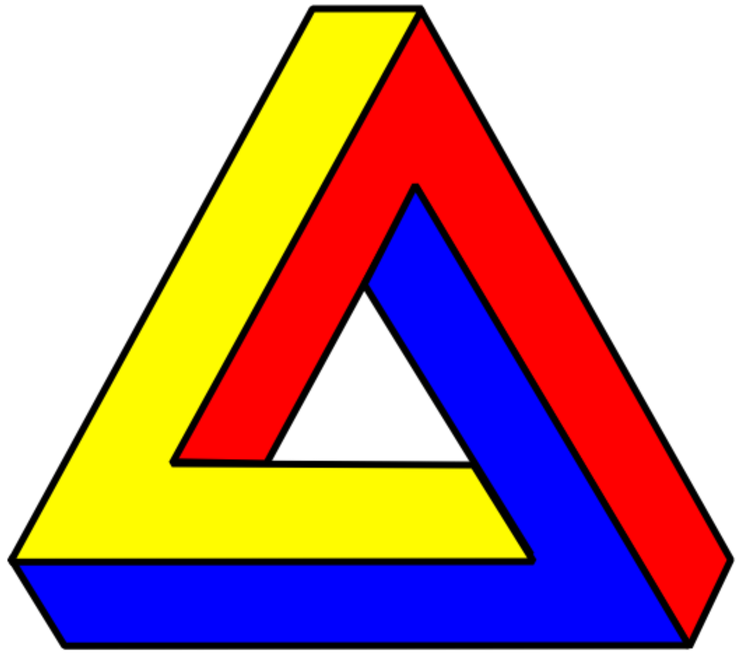
Rx
from first principles



emeijer@applied-duality.com

Getters

$$() => A$$

Covariant

$$A <: B$$

$$() => A <: () => B$$

Functor

```
val map: (A=>B)
         =>(()=>A)=>()=>B

map f a =()=>f(a())
```

“Monad”

```
val flatMap:(()=>A)
  =>(A=>(()=>()=>B))=>(()=>B)

flatMap a f = ()=>f()(a())()
```

Side Effects

```
val steveb: ()=>String

steveb() // “developer”
steveb() // “blah”
steveb() // “Windows 8”
steveb() // ☰
```

Side Effects

```
() => Try[A]
```

Side Effects

```
val sjobs: ()=>String

sjobs() // “iPhone”
sjobs() // “iPad”
sjobs() // “iCloud”
sjobs() //  †
```

Side Effects

```
()=>Try[Option[A]]
```

Getter Getter

```
() =>
  (() =>
    Try[Option[A]]
  )
```

Interfaces

```
trait Enumerable[+T] {
    def getEnumerator(): Enumerator[T]
}

trait Enumerator[+T] {
    def moveNext(): Boolean
    def current: T
}
```

Lifting

```
trait Enumerable[+T] {
    def getEnumerator(): Enumerator[T]

    def lift(f: Enumerator[T]=>Enumerator[S]):
        val that = this
        Enumerable[S] = {
            new Enumerable[S] {
                def getEnumerator() =
                    f(that.GetEnumerator())
            }
    }
}
```

Functor

```
val map: (A=>B)=>
Enumerable[A]=>Enumerable[B]

map f as = as.lift(_.map)
```

Monad

```
val flatMap: (A=>Enumerable[B])=>
Enumerable[A]=>Enumerable[B]

flatmap f as = as.lift(_.flatmap)
```

Reverse
All
Those
=>

Setters

$$\mathbb{A} => ()$$

Contravariant

$$A <: B$$

$$B => () <: A => ()$$

coFunctor

```
val map: (A=>B)
         =>((B=>())=>(A=>()))

map f b = a=>(b(f a))
```

“Monad”

```
val flatMap:(A=>())
   =>(B=>(()=>A)=>())=>(B=>()

flatMap a f = b=>f(b)(a)
```

Side Effects

```
val emeijer: String=>()

emeijer(“Comega”)
emeijer(“LINQ”)
emeijer(“Rx”)
emeijer(☲)
```

Side Effects

```
Try[A] => ()
```

Side Effects

```
val kubric: String=>()

kubric(“Spartacus”)
kubric(“Lolita”)
kubric(“Eyes Wide Shut”)
kubric(†)
```

Side Effects

```
Try[Option[A]]=>()
```

Setter Setter

```
(Try[Option[A]]
    => ()
)=>()
```

Interfaces

```
trait Observable[+T] {
    def Subscribe(o: Observer[T]): Unit
}

trait Observer[-T] {
    def onCompleted(): Unit
    def onError(error: Throwable): Unit
    def onNext(value: T): Unit
}
```

Lifting

```
trait Observable[+T] {
    def subscribe(o: Observer[T])

    def lift(f: Observer[S]=>Observer[T]):
        val that = this
        Observable[S] = {
            new Observable[S] {
                def subscribe(o: Observer[S]) =
                    that.Subscribe(f(o))
            }
    }
}
```

Functor

```
val map: (A=>B)=>
Observable[A]=>Observable[B]

map f as = as.lift(_.map)
```

Monad

```
val flatMap:(A=>Observable[B])=>
Observable[A]=>Observable[B]

flatmap f as = as.lift(_.flatmap)
```

Real World

NETFLIX

The Netflix Tech Blog

Thursday, January 16, 2014

Improving Netflix's Operational Visibility with Real-Time Insight Tools

By Ranjit Mavinkurve, Justin Becker and Ben Christensen

For Netflix to be successful, we have to be vigilant in supporting the tens of millions of connected devices that are used by our 40+ million members throughout 40+ countries. These members consume more than one billion hours of content every month and account for nearly a third of the downstream Internet traffic in North America during peak hours.

Links

- Netflix US & Canada Blog
- Netflix America Latina Blog
- Netflix Brasil Blog
- Netflix UK & Ireland Blog
- Open positions at Netflix
- Netflix Website
- Facebook Netflix Page
- RSS Feed

Real World


Project
.idea
src
rx
firebase.scala
events
javafx.scala
events
Main.scala
LanguageDe
Main
Ping
Pong
Scribble.scala
Scribble
Scribble
OperatorRepeat.java
Schedulers.java
Main.scala
firebase.scala
Test.java
cancelled: Throwable => Unit = error => {}
): ValueEventListener = new ValueEventListener {
def onDataChange(data: DataSnapshot) {
Rx Scribble
Rx Scribble Spy
Run:
Scribble
/Library/Java/

Real World


USGS Earthquake Stream
Map
Satellite
4.3
Iceland
Sweden
Norway
Russia
United Kingdom
Canada
Poland
Germany
Ukraine
Kazakhstan
Mongolia
France
Italy
Spain
Turkey
China
South Korea
United States
North Atlantic Ocean
Afghanistan
Iraq
Iran
Pakistan
Algeria
Libya
Egypt
Saudi Arabia
India
Thailand
Mexico
Mali
Niger
Sudan
Chad
Nigeria
Ethiopia
Venezuela
Colombia
Kenya
DR Congo
Indonesia
Tanzania
Brazil
Peru
Angola
Bolivia
Namibia
Madagascar
Botswana
Indian Ocean
South Atlantic Ocean
Chile
South Africa
Argentina