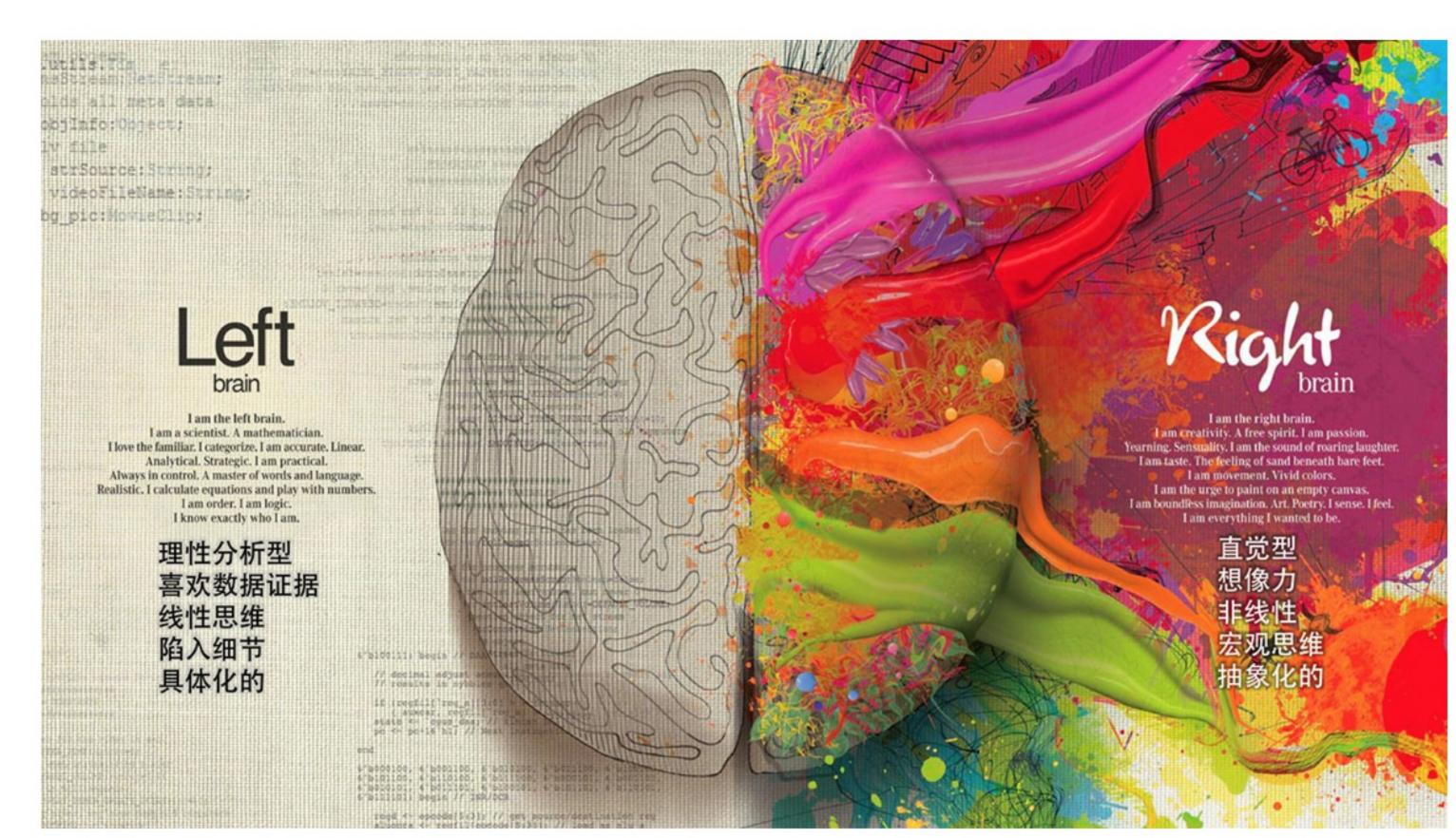
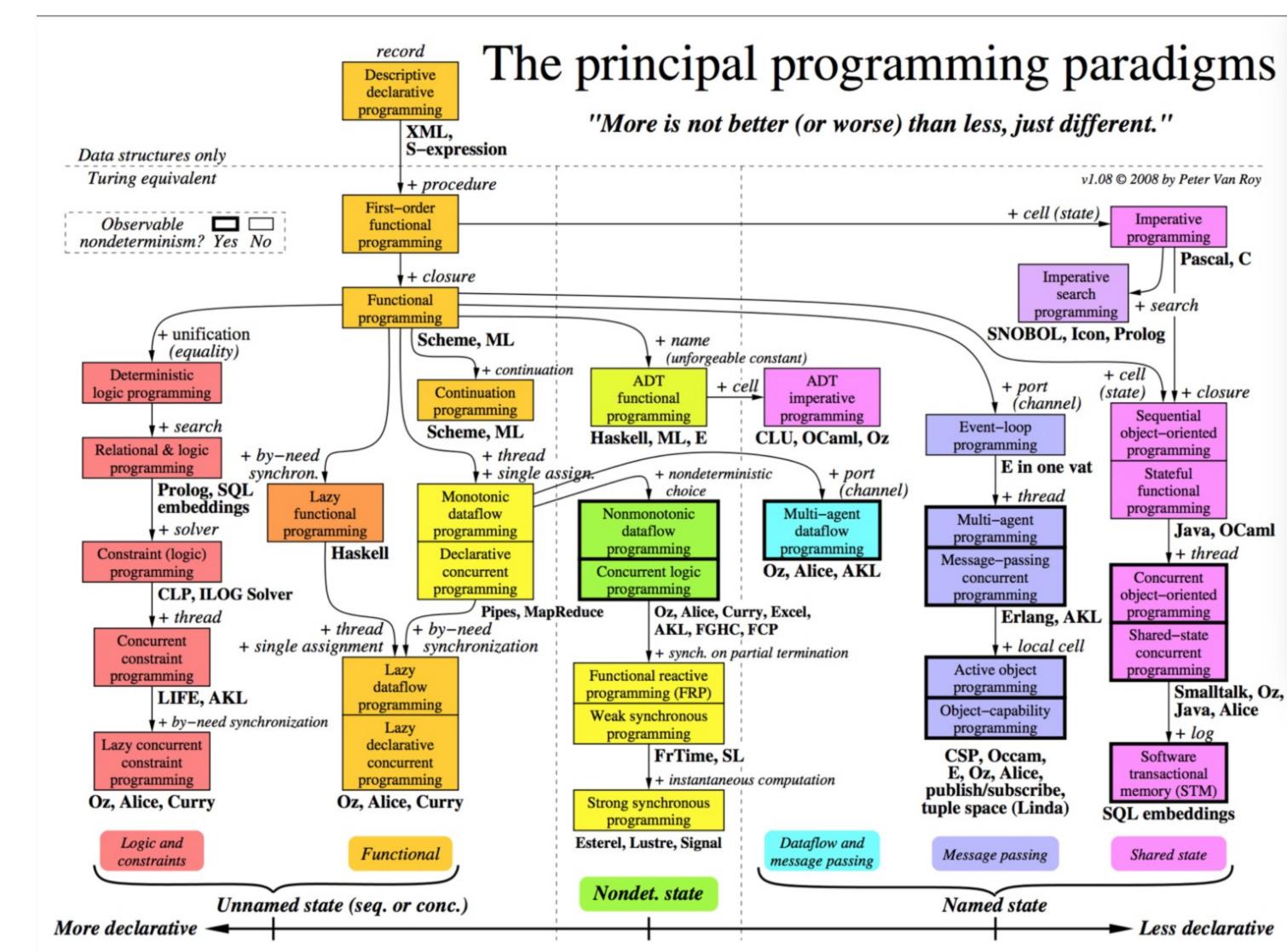
2018-02-15 陈皓



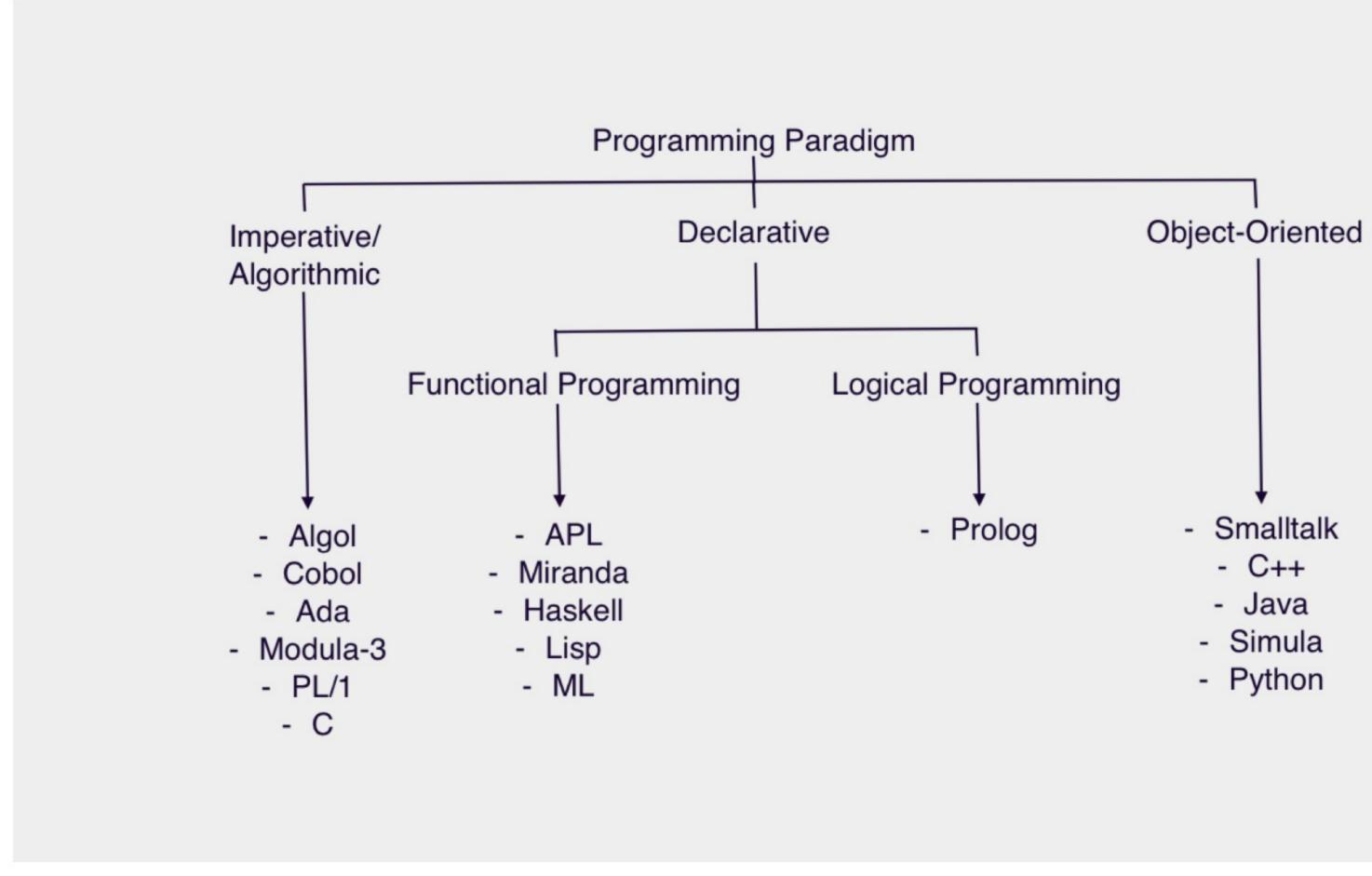
这个世界到今天有很多很多的编程范式,相当复杂。下面这个图比较好地表现了这些各式各样的编程范 式。这个图越往左边就越是"声明式的",越往右边就越不是"声明式的"(指令式的),我们可以看到,函 数式编程、逻辑编程,都在左边,而右边是指令式的,有状态的,有类型的。

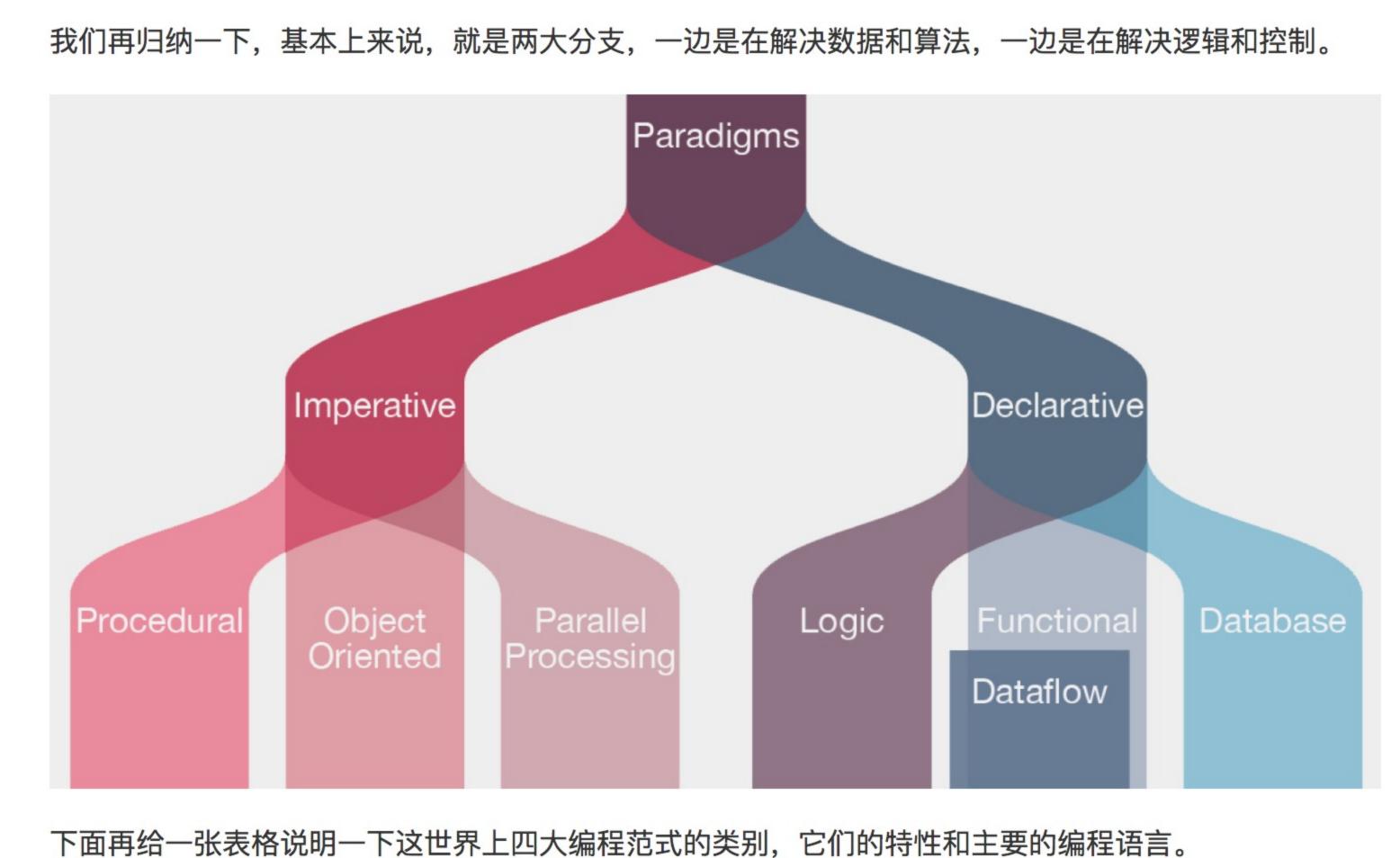


上面这个图有点乱,不过总体说来,我们可以简单地把这世界上纷乱的编程范式,分成这几类:声明式、 命名式、逻辑的、函数式、面向对象的、面向过程的。

于是我们归纳一下,就可以得到下面这个简单的图。简单描述一下,

- 中间两个声明式编程范式(函数式和逻辑式)偏向于你定义要什么,而不是去怎么做。
- 而两边的命令式编程范式和面向对象编程范式,偏向于怎么做,而不是要做什么。





编程范式	描述	主要的特性	相关的编程语言
Imperative 命令式	使用流程化的 语句和过程直 接控制程序的 运行和数据状 态。	直接赋值 常用的数据结构 全局变量、局部变量 Goto语句 顺序化数据的操作和迭代 以功能为主的模块化	C, C++, Java, PHP, Python, Ruby
Functional 函数式	通过数学函数 表达式的方式 来避免状态和 可变的数据。	代码公式化 Lambda 表达式 函数的包装和嵌套(高阶函数、Pipeline、Currying、 Map/Reduce/Filter) 递归(尾递归) 无数据共享或依赖 无副作用(并行、重构、组合)	C++, Clojure, CoffeeScrip Elixir, Erlang, F#, Haskell, Lisp, Python, Ruby, Scal SequenceL, SML
Object- Oriented 面向对象	把一组字段和 作用在其上面 的方法抽象成 一个个对象。	对象封装 消息传递 隐藏细节 数据和接口抽象 多态 继承 对象的序列化和反序列	Common Lisp, C++, C#, Eiffel, Java, PHP, Python Ruby, Scala
Declarative 声明式	定义计算的逻辑而不是定义 具体的流程控制。	4GLs, spreadsheets, report program generators	SQL, 正则表达式, CSS, Prolog, OWL, SPARQL

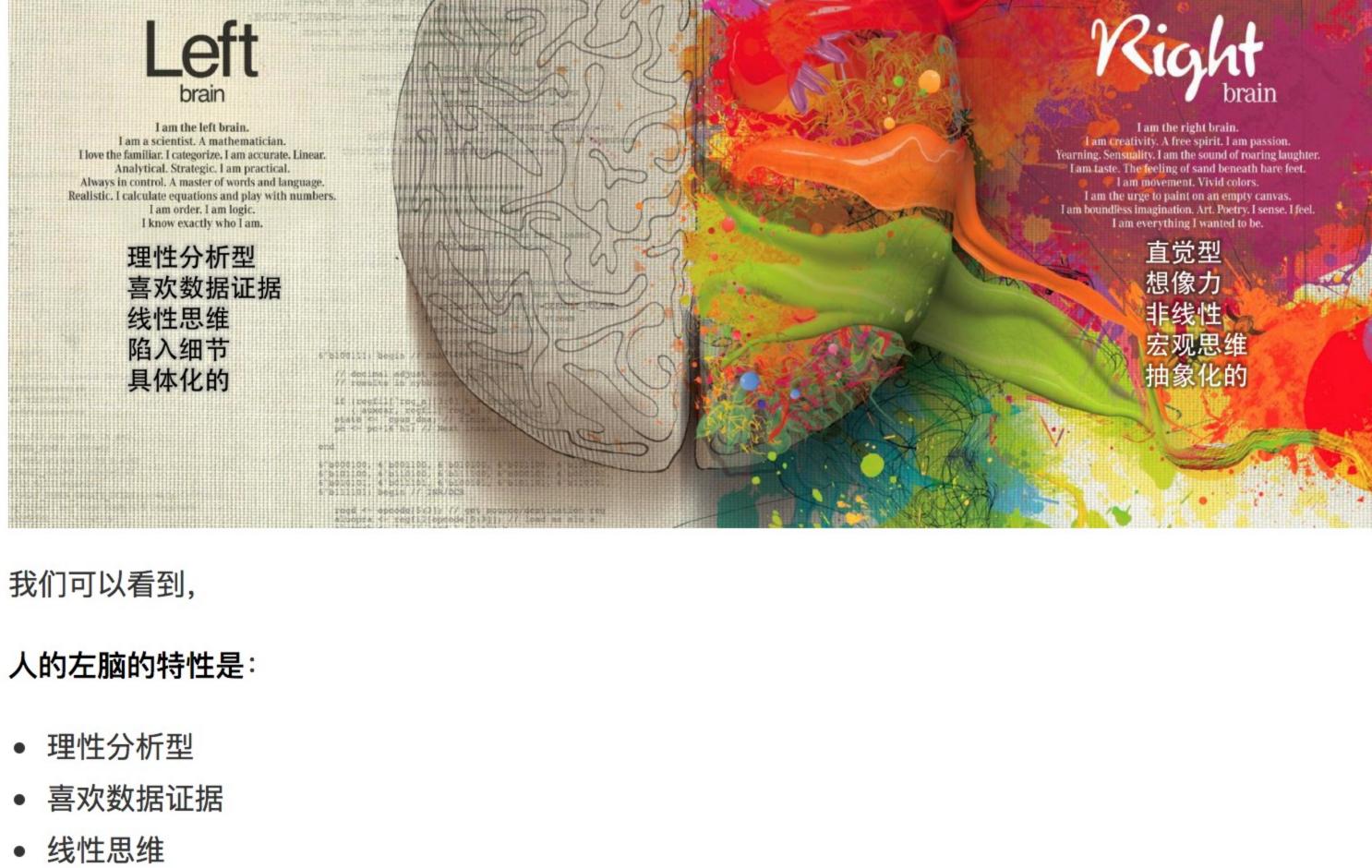
了,我们只剩左边。我们陷入细节,我一说 Java 是最好的程序设计语言,一堆人就来了,找各种各样的 细节问题跟你纠缠。 离我们最近的是函数式编程,但既然函数式编程这么好,为什么函数式编程火不起来呢?首先,这里有个 逻辑上的问题,并不是用的人越多的东西就越好。因为还要看是不是大多数人都能理解的东西。函数式编 程或是声明式编程,需要的是用我们的右脑,而指令式的则需要用我们的左脑。

程序编程范式。一个是左脑,一个右脑。我们程序员基本上是在用左脑,左脑是理性分析,喜欢数据证

据,线性思维,陷入细节,具体化的,不抽象。但是实际上玩儿出这些东西的都在右脑,函数式,还有像

逻辑式的抽象能力都在右脑。所以我们非线性的想象力都在这边,而标准化教育把我们这边已经全部干掉

参看下图: lutiis.Tim e NasizamijetSiram; ilds all meta data objimic:Object; n file



• 陷入细节

atmSource: String;

bg_pic:MowieClups

videofileWame:5%:01nm;

- 具体化的
- 人的右脑的特性是: 直觉型
- 想象力 非线性 • 宏观思维
- 人类社会中,绝大多数人都是左脑型的人,而只有少数人是右脑型的人,比如那些哲学家、艺术家,以及 能够创造理论知识的人。这些人在这个世界上太少了。 这是为什么很多人理解和使用声明式的编程范式比较有困难,因为这要用你的右脑,但是我们习惯于用我

• 抽象化的

说到人类的大脑了,已经到了不是我专长的地方了,这个话题太大了,所以,也是时候结束《编程范式游 记》这一系列文章了。希望你能从这一系列文章中有所收获。如果有什么疑问或是我有什么没有讲对的,

以下是《编程范式游记》系列文章的目录,方便你了解这一系列内容的全貌。这一系列文章中代码量很 大,很难用音频体现出来,所以没有录制音频,还望谅解。

• 编程范式游记(2) – 泛型编程 • 编程范式游记(3) – 类型系统和泛型的本质 • 编程范式游记(4) – 函数式编程

• 编程范式游记(6) – 面向对象编程 • 编程范式游记(7) – 基于原型的编程范式 • 编程范式游记(8) – Go 语言的委托模式

2018-02-17

不记年

2018-02-16

直接描述问题的逻辑

• 编程范式游记(1) – 起源

• 编程范式游记(9) - 编程的本质 • 编程范式游记(10) - 逻辑编程范式 • 编程范式游记(11) – 程序世界里的编程范式

• 编程范式游记(5) – 修饰器模式

们的左脑,左脑用多了以后右脑就有点跟不上了。

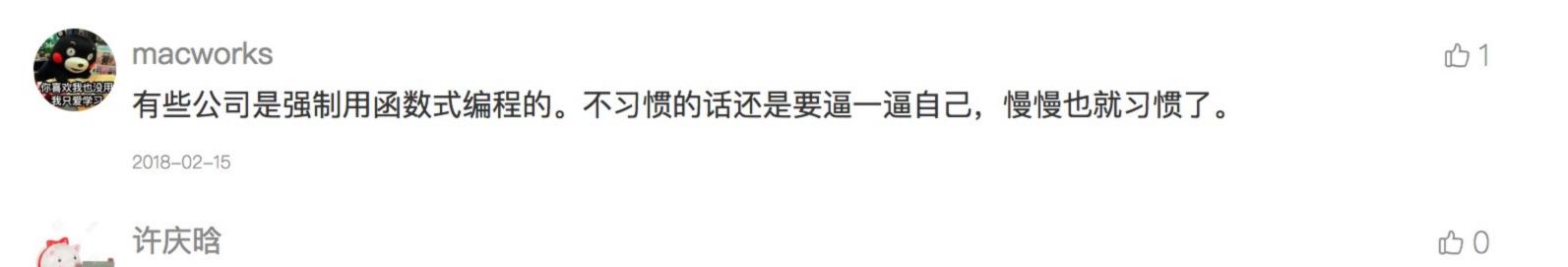
还希望得到你的批评和指正。先谢谢了。





4 极客时间

60



文章都同意,唯独左右脑的图文解释不服。左脑图上明明写了mathematician和logic control order,

册子,用三行写出来了个quicksort,感觉对recursion有了更加深入的理解 😁

这才更符合本系列讲的函数式和logic control分离。从图像上看,左脑才像是函数式,声明式的程序 嘛。右脑的图更像混杂了logic和control,充满了side effects的代码呀,或者,甲方,投资方脑子里的 项目(=゚Д゚=)。。。我觉得这个世界上,还是左脑函数式思维的人更少吧。也许国外高端圈子不是这 样? 2018-04-07