

属兔的处子

Clojure太灵活，臣妾驾驭不住啊！



促进软件开发领域知识与创新的传播



关注InfoQ官方信息
及时获取QCon软件开发者
大会演讲视频信息



扫码，获取限时优惠

ArchSummit

全球架构师峰会 2017 [深圳站]

2017年7月7-8日 深圳·华侨城洲际酒店

咨询热线：010-89880682

QCon

全球软件开发大会 [上海站]

2017年10月19-21日

咨询热线：010-64738142

何婧誉 (Loretta)

- 剑桥大学
- 高盛……花旗……SwiftKey……微软……
- Morgan Stanley!
- 这是一个Clojure的分享，但我现在在写Scala
- 为啥？摩根士丹利的优点……
- We are hiring!

Morgan Stanley

Clojure辣么灵活，
不就是因为动态类型吗？

不信你看！

咱们写个读json的，一行妥妥的！

```
(defn read-json  
  [file-name]  
  (json/read (clojure.java.io/reader file-name)))  
  
(read-json "data.json")
```

然而……

读了之后数据是什么样儿的？

```
(defn manipulate-json
  [json-data]
  (->> json-data
    (mapv #(select-keys % [ :age :name :job :address ])))
    (mapv #(update % :age inc))
    (zipmap (range 0 (count json-data)))))
```

刚才还是简单的……

你们觉得:name的值应该是什么样儿的？

String? Vector? List?



```
(defn get-names  
  [json-data]  
  (mapv #(clojure.string/join " " (:name %)) json-data))
```

看到代码之前完全不明白要怎么用

来看个实例

感谢链家！ ^_^

-
- 来北京嘛，不得不关注一下大家都关注的房价
 - 去链家网上看了看二手房
 - 一页一页的数据太难整理啦！
 - 写个程序抓取一下d(`·▽·)b

注：此次实验中没有链家服务器受到伤害，请大（鸟）家（哥）放心！

Intellij 中 vanilla 项目 demo



灵活的胖子

core.typed

加个类型系统

core.typed

- 通过一个库给动态语言加上类型系统——即插即用
- 可以给已经写好的函数或者是用的无类型库标注类型
- 可以选择性地加上类型
- 加上了类型也并非一定要type check

core.typed

- 支持Option Type, Ordered Intersection Types, Union Types
- 支持Heterogenous Maps和Sequentials
- 支持Polymorphism (All, Context Bounds), Higher-Kinds
- 支持Occurrence Typing! (通过检查control flow进行类型推导)
- 宏也会被展开后再推导类型

It's *Amazing!*

Demo the typed project

但是并不完美

- 宏一复杂了还是不行
- 要不停地给用的库加类型——自己加上了类型的Clojure库实在太少了……
- 不少核心函数根本没法儿加类型
- 于是你开始到处加`^:no-check`……

没有别的办法了吗？



给兔子加上牵引绳

core.spec

Contract式限制

core.spec

- Runtime性能基本不会受到影响（缺省spec验证关闭）
- Map的类型应该就是key及其对应的值的类型！（keys）
- Sequence可以多方面限制（cat, alt, regex style matching, coll-of）
- 只有一个参数的返回boolean值的函数通通都自动成为predicate
- 各种验证方式，满足你的需求（conform, explain, valid?）
- multi-spec支持更复杂的数据结构

spec项目demo

生成性测试

测试复杂数据结构是麻烦的

- Haskell的quickcheck是生成性测试的鼻祖
- 顾名思义，生成性测试就是自己生成测试数据的测试
- 生成性测试可以省去模拟数据的麻烦
- 生成性测试还会自动缩小至失败的最小案例以供debug
- 用core.spec定义了函数的spec之后，即可用check自动进行生成型测试
- 除非数据结构范围宽广，自带的generator不够用
- 但可以用自定义的generator补充

core.typed vs core.spec - pro

core.typed

- Require时检查
- 支持HMap/HVec/HSeq
- 支持I/U types
- 支持宏展开type check
- Occurrence typing

core.spec

- 基本通过测试检查
- Global spec
- 支持coercion
- 可选的运行期检查
- 灵活性强, 可以覆盖所有函数

core.typed vs core.spec - con

- core.typed
 - 一个人在维护
 - 有些慢
 - 仍有函数无法标注
 - 绝大部分库没有类型标注过
- core.spec
 - 仍在Alpha
 - 文档仍需补全
 - 和clojure.test仍未完全紧密结合

Q \rightarrow (Option A)

我猜你们会问我的问题

- Clojure for the Brave and True还译不译？
译！有出版社拿版权我就译！
- 会出书吗？出的话什么时候？
有家出版社有和我联系，但等我有力气了再说吧……
- core.typed和core.spec你推荐哪个？
我的脑子喜欢core.spec，因为有前景。我的心喜欢core.typed，因为给东西加类型写起来真得很过瘾。