### Node 地下铁沙龙 #8 Let's Go Serverless!

GitHub @hax 知乎@贺师俊 微博@johnhax Twitter @haxy

### FaaS与FP 之初步思考

# 





# 63

#### Function as a Service



#### Functional Programming

### FaaS — Computing Architecture FP — Programming Paradigm

# 相同的下不同层面

#### Fin FaaS(计算架构)

- 部署和伸缩单位(VM/OS ➡ 应用 ➡ 函数)
- 资源限制(内存、运行时间) 计费方式(不为Idle付费)

#### Fin FP(编程模型)

- 高阶函数
- 纯、引用透明性
- 递归、求值策略

# 所以 FaaS 和 FP 并没有什么关系!

### The End

# 硬要说的话····· Stateless — Pure

### 安利一下FP

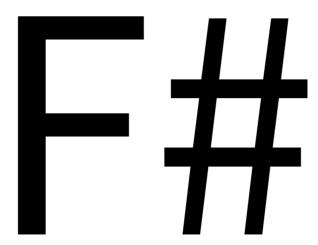
## FPH 등

- Haskell
- Lisp
- ML
- Erlang

### FaaS平台

- Node.js
- Java
- .NET
- Python, Go, PHP

#### 没有一个原生支持的FP?



- 选择可以在FaaS上跑的FP语言
- 选择现有的多范式语言

- NET -> F#
- Java -> Clojure
- Node.js -> \*

### 暗度陈仓

#### Node.js: child\_process.exec

- .NET -> C#
- Java -> Java, Scala, Kotlin, ...
- Node.js -> JavaScript

#### FaaS 语言平台的选择

- Package Size
- RAM
- Cold Start
- Performance
- Programming language

# Node.js

### 为 用 注

### • 实时编译

### • 预先编译

- runtime
- 调试
- 性能开销

### OCaml (BuckleScript)

## MASM

### Use JavaScript as FP?

- Tail call (STC/PTC)
- Syntax (pipeline op?)
- Data structure (immutable library?)

### FP in JS

### Garbage

#### 以不是那么FP的方式写JS

- 不用递归
- 不用特殊的数据结构

#### 如何实施 immutable 约束?

- TypeScript (ReadOnly<X>)
- Proxy (X.readonly)

# 

- Node is Wrong for severless? (async)
- FaaS Functional Patterns? (函数编排)
- FaaS 在特定语言的优化问题(JIT、GC)

