Go on **AppEngine**

2013.04.13 - GoCon

AppEngine

PaaS

自分のアプリをGoogleのインフラ上で動かせる

容易・効率

- Go:
  構文、型システム、コンパイルが速い、
  テスト、標準パッケージ充実

- AppEngine:
  PaaS、テスト、開発環境

安全性

- Go:
  静的型、ポインタ演算なし、GC

- AppEngine:
  Googleのセキュリティ、サンドボックス、
  メンテフリー

パフォーマンス

- Go:
  Nativeで速い、省メモリ

- AppEngine:
  大規模、伸縮

モダン

- Go:
  並列、型推論、
  Structual subtyping（部分型）

- AppEngine:
  TQ、Memcache、自動スケール

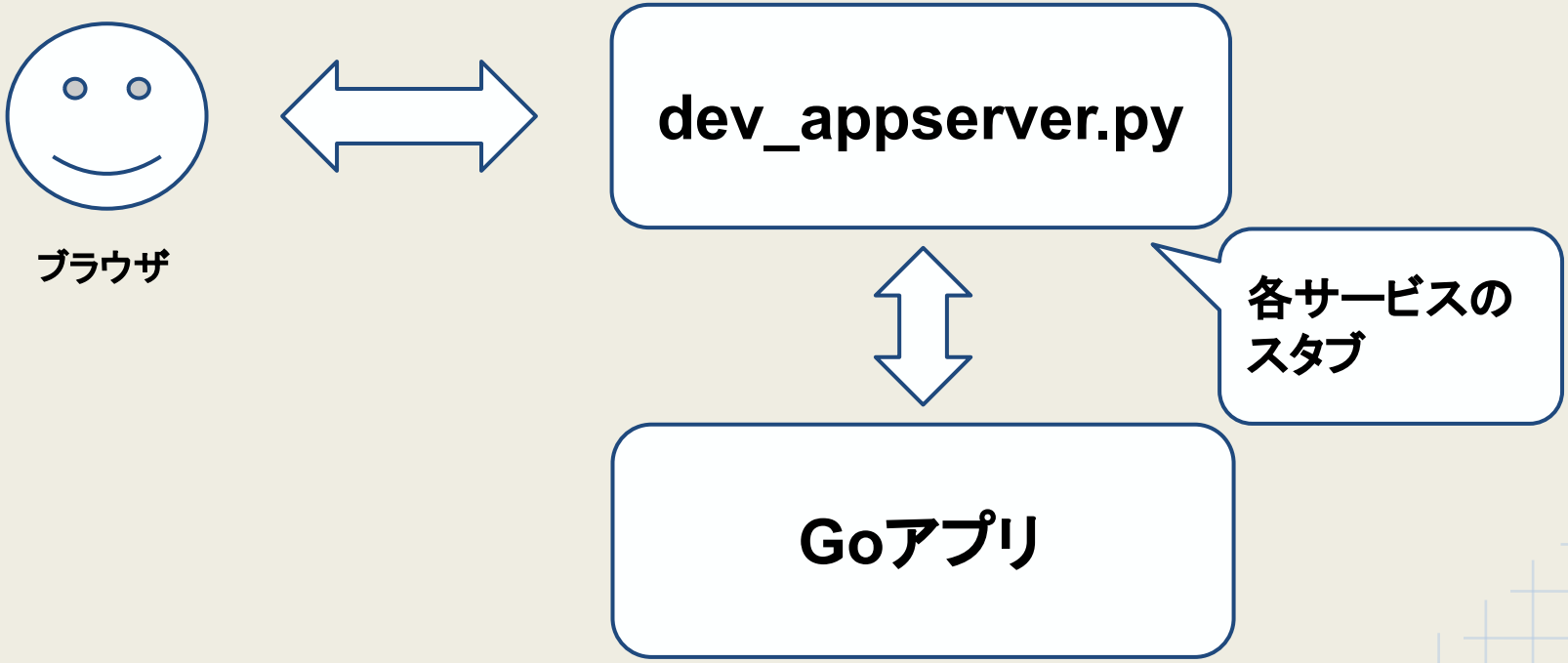
Getting Started

```
package hello

import ("net/http"; "fmt")

func init() {
    http.HandleFunc("/", welcome)
}

func welcome(w http.ResponseWriter, r *http.Request) {
    fmt.Fprint(w, "こんにちは")
}
```

Datastore

```
type TestData struct {
    Content string  `datastore:"c,noindex"`
}

func welcome(w http.ResponseWriter, r *http.Request) {
    c := appengine.NewContext(r)
    td := TestData{
        Content: "これはテストです",
    }
    key := datastore.NewIncompleteKey(c, "TestData", nil)
    _, err := datastore.Put(c, key, &td)
}
```

SDK



Testing

- Go言語には
  テスト用パッケージ **testing** がある
- AppEngineのスタブが**Python**なので
  テストからAPIを使うのが**難しい**

testbed

PythonのAppEngineテスト用モジュール
testbed

Goから使うためのプロジェクト
https://github.com/najeira/testbed

※ 1.7.7 では動きません！
SDKにパッチをあてる

Spin-up

Python :

約 386 ms

Go :

約 48 ms

※ミニマムなアプリで計測。Pythonはwebapp2使用

Memory usage

Python :

約 38 MB

Go :

約 4 MB

※ミニマムなアプリで計測。Pythonはwebapp2使用

Goの優位

Spin-upが速い
　急激なスパイクに強い

省メモリ
　大きなデータを使った処理がしやすい

※本当に省メモリかどうかは、コードに依存するが、
少なくともランタイムは小さい

速度について補足

Datastoreなど I/O待ちの時間が支配的なケースも多い

Goが常に速いというわけではない

まとめ

新規プロジェクトの場合は、選択肢に入る

無理にPythonから乗り換えるほどではない（と思う）

一部の処理をGoで書いてBackground処理をするのも良さそう