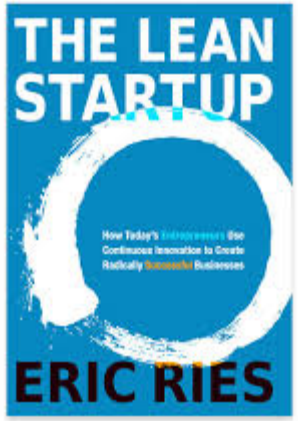
Tiny Web Apps

Google App Engine

Matt Messinger

What are “tiny web apps”?

The Lean Startup by Eric Ries



“A core component of Lean Startup methodology is the build-measure-learn feedback loop. The first step is figuring out the problem that needs to be solved and then **developing a minimum viable product (MVP) to begin the process of learning as quickly as possible**. Once the MVP is established, a startup can work on tuning the engine. This will involve measurement and learning and must include actionable metrics that can demonstrate cause and effect question.”

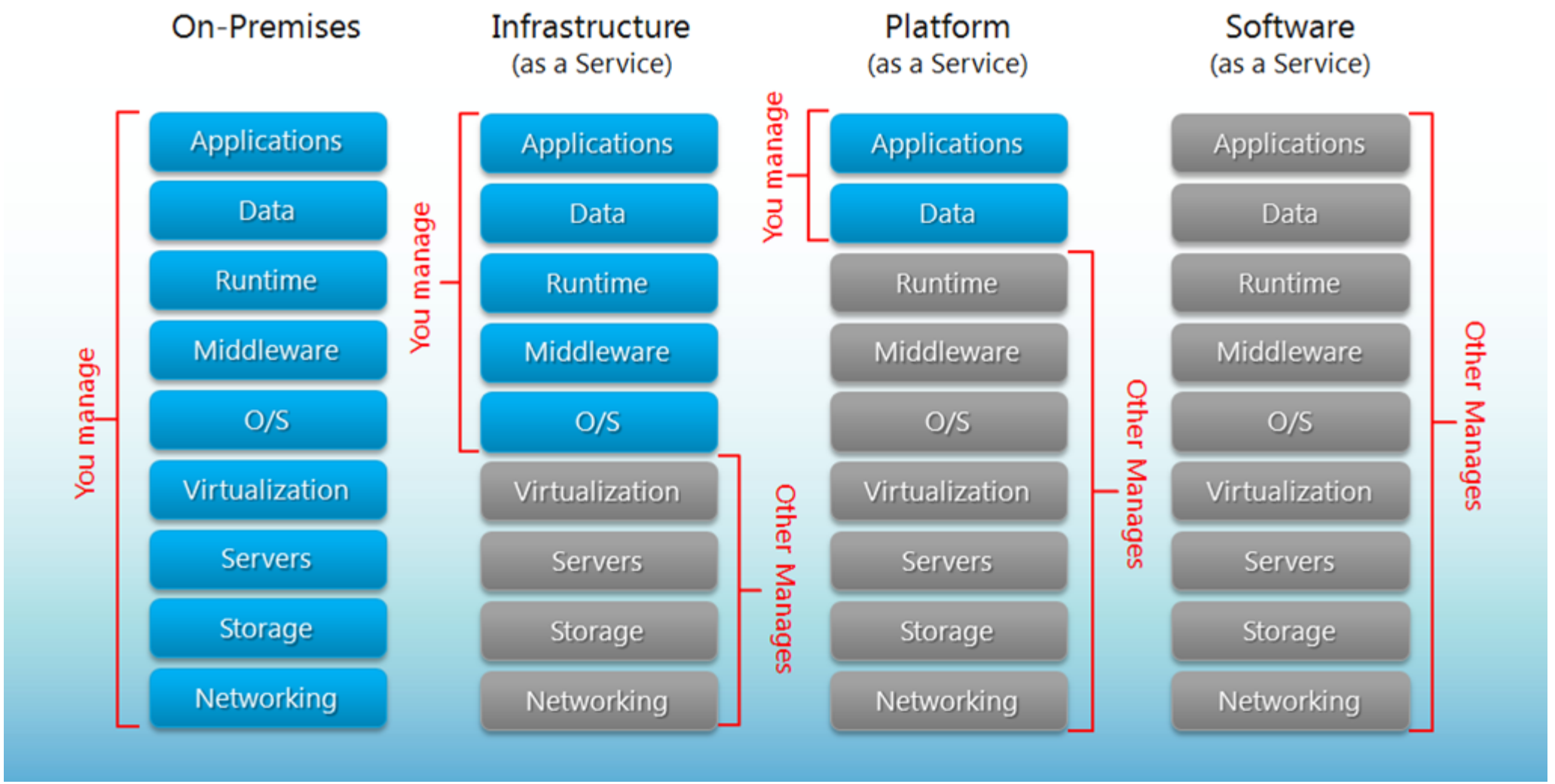
In the web world, I call this MVP a “tiny web app”

or - just because it’s fun

Goals

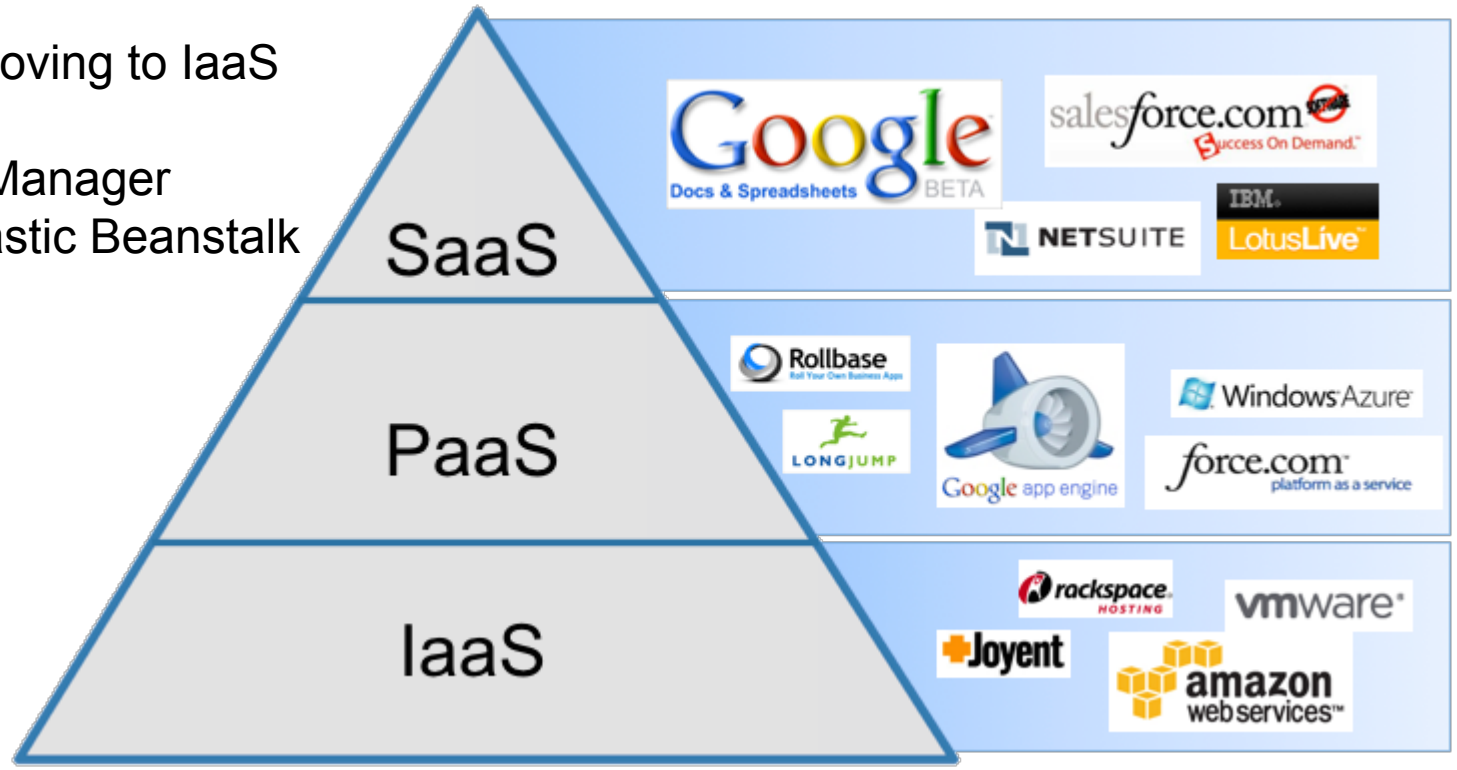
- Build something quickly
- Minimize development and hosting costs
- Minimize issues that would hinder early adoption
- Be able to scale with demand quickly

Separation of Responsibilities



*aaS?

- Google started in PaaS and is moving to IaaS
  - Google Compute Engine
  - Google Cloud Deployment Manager
- AWS has PaaS offerings like Elastic Beanstalk



Google App Engine

- Fully managed platform
- Web Management Console
- Supports Python, Java, PHP, Go and has language specific SDK's
- SDK to run local development just like production
- Command line utility to upload app to production
- Multiple storage options:
  - Cloud Datastore - NoSQL
  - Cloud SQL - fully managed MySQL database
  - Cloud Storage - object storage

Google App Engine

- Built in services:
  - Channel - bidirectional channel with client
  - Images - manipulate image data
  - Mail - send and receive
  - Memcache - explicit and automatic
  - Task Queues
  - Users - google accounts, google apps, OpenID
  - XMPP
- Scheduled tasks, DoS protection, Auto-indexing

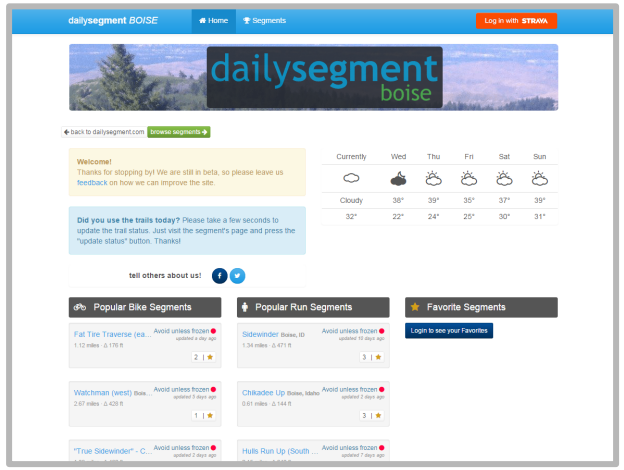
dailysegment.com

app.yaml

```yaml
application: XXX-XXX-XXX
version: 1
runtime: python27
threadsafe: true
api_version: 1


handlers:
# api handler
- url: /api/.*
  script: api.app.application
  secure: always

# admin handler
- url: /admin/.*
  script: admin.app.application
  login: admin
  secure: always

# static handler
- url: /styles
  static_dir: web/dist/styles
- url: /fonts
  static_dir: web/dist/fonts
- url: /scripts
  static_dir: web/dist/scripts
- url: /bower_components
  static_dir: web/dist/bower_components
- url: /images
  static_dir: web/dist/images
- url: /favicon\.ico
  static_files: web/dist/favicon.ico
  upload: web/dist/favicon\.ico

# Auth handler
- url: /auth/.*
  script: web.auth.application
  secure: always

# Web handler
- url: /.*
  script: web.app.application
  secure: always


libraries:
- name: jinja2
  version: latest
```

dailysegment

trails · usage · leaderboards



AngularJS *(frontend)*

Bootstrap/Bootswatch *(style)*

Web App / webapp2+jinja

(REST api / html template)

Python Runtime

App Engine Web Server

Mail

Memcache

URL Fetch

Google App Engine

Google Cloud Storage

(static images for FB)

Google Cloud Datastore NDB

(schemaless DB)

Google Apps

(DNS and SSL Certificate)

Demo

- Simple Python Web App
  - Angular js/css served statically out of app
  - Bootstrap “self” with template & grunt-replace-string
- Leverages GAE User, Email, Memcache, NDB services

https://github.com/mattmessinger/boise-angularjs-gae-demo

Obligatory AngularJS Slide


```
.run(function($rootScope, $window) {
  $rootScope.$on('$routeChangeSuccess', function () {
    $window.scrollTo(0, 0);    //scroll to top of page after each route change
  });
})
```