
Lakshman Prasad (@becomingGuru)

September 18, 2011
Development is but, one part of the Application "Growth" Cycle
Common non-Development Tasks

- Distributed log storage and analysis
- Backups and Snapshots
- Graphing, Instrumentation and Monitoring
- HTTP Caching, Memory Caching
- Failover, Node addition/removal
- Auto-scaling for cloud resources
- Data Retention/Archival
- Data Model Changes, Database sharding
- CDN Management
- API Metering, Rate Limiting
- Handling Multiple Environments, Multiple Versions, Rollbacks
The Application "Growth" Cycle

- Develop
- Test
- Design Production Environment
- Deploy
- Monitor
- Tune
17444 Pypi packages
<table>
<thead>
<tr>
<th>Repository</th>
<th>Description</th>
<th>Size</th>
<th>Forks</th>
<th>Watchers</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>django / django</td>
<td>Official clone of the Subversion repository.</td>
<td>7.3 MB</td>
<td>341</td>
<td>1803</td>
<td>about 9 hours ago</td>
</tr>
<tr>
<td>robhudson / django-debug-toolbar</td>
<td>A configurable set of panels that display various debug information about the current request/response.</td>
<td>617.2 KB</td>
<td>168</td>
<td>1283</td>
<td>8 days ago</td>
</tr>
<tr>
<td>pinax / pinax</td>
<td>A Django-based platform for rapidly developing websites</td>
<td>4.5 MB</td>
<td>206</td>
<td>1176</td>
<td>4 days ago</td>
</tr>
</tbody>
</table>
... one for every size and style
<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
<th>message</th>
<th>history</th>
</tr>
</thead>
<tbody>
<tr>
<td>appengine</td>
<td>November 07, 2010</td>
<td>REMoved PyCharm thingies, moved appengine stuff to ... [shabda]</td>
<td></td>
</tr>
<tr>
<td>bottle</td>
<td>November 08, 2010</td>
<td>Pep-8. [shabda]</td>
<td></td>
</tr>
<tr>
<td>django</td>
<td>November 08, 2010</td>
<td>Created a module to expose a WSGI app called appli... [ericmoritz]</td>
<td></td>
</tr>
<tr>
<td>flask</td>
<td>November 08, 2010</td>
<td>Pep-8. [shabda]</td>
<td></td>
</tr>
<tr>
<td>itty</td>
<td>November 08, 2010</td>
<td>Pep-8. [shabda]</td>
<td></td>
</tr>
<tr>
<td>juno</td>
<td>November 08, 2010</td>
<td>Pep-8. [shabda]</td>
<td></td>
</tr>
<tr>
<td>nagare</td>
<td>November 09, 2010</td>
<td>Remove unnecessary files [Vincent]</td>
<td></td>
</tr>
<tr>
<td>nodejs</td>
<td>November 26, 2010</td>
<td>added Ruby frameworks, sinatra and rails [ashok-raavi]</td>
<td></td>
</tr>
<tr>
<td>php</td>
<td>March 14, 2011</td>
<td>PHP version (no framework) [elcio]</td>
<td></td>
</tr>
<tr>
<td>pyramid</td>
<td>November 08, 2010</td>
<td>Merge branch 'master' of <a href="https://github.com/agiliq">https://github.com/agiliq</a>... [ericmoritz]</td>
<td></td>
</tr>
<tr>
<td>pyroutes</td>
<td>November 08, 2010</td>
<td>Added pyroutes version [klette]</td>
<td></td>
</tr>
<tr>
<td>rails</td>
<td>November 19, 2010</td>
<td>added ruby frameworks, sinatra and rails [ashok-raavi]</td>
<td></td>
</tr>
<tr>
<td>sinatra</td>
<td>November 19, 2010</td>
<td>added ruby frameworks, sinatra and rails [ashok-raavi]</td>
<td></td>
</tr>
<tr>
<td>test</td>
<td>November 08, 2010</td>
<td>Fixed the table headers [ericmoritz]</td>
<td></td>
</tr>
<tr>
<td>tornado</td>
<td>November 09, 2010</td>
<td>Merge branch 'didip-master' [shabda]</td>
<td></td>
</tr>
<tr>
<td>twisted</td>
<td>November 08, 2010</td>
<td>added app based on twisted.web [zed]</td>
<td></td>
</tr>
<tr>
<td>web2py</td>
<td>November 08, 2010</td>
<td>Removed unnneeded files, per mdipierro's suggestion... [shabda]</td>
<td></td>
</tr>
<tr>
<td>webob</td>
<td>November 08, 2010</td>
<td>Pep-8. [shabda]</td>
<td></td>
</tr>
<tr>
<td>webpy</td>
<td>November 08, 2010</td>
<td>Pep-8. [shabda]</td>
<td></td>
</tr>
<tr>
<td>.gitignore</td>
<td>November 07, 2010</td>
<td>gitignore [shabda]</td>
<td></td>
</tr>
<tr>
<td>AUTHORS</td>
<td>March 15, 2011</td>
<td>Added Elcio Ferreira for the PHP version. [theju]</td>
<td></td>
</tr>
</tbody>
</table>
So you wrote a web app.
"Code without tests is broken by design"
Unit Testing

```python
from django.utils import unittest
from myapp.models import Animal

class AnimalTestCase(unittest.TestCase):
    def setUp(self):
        self.lion = Animal.objects.create(name="lion",
                                           sound="roar")
        self.cat = Animal.objects.create(name="cat",
                                           sound="meow")

    def testSpeaking(self):
        self.assertEqual(self.lion.speak(),
                         'The lion says "roar" ')
        self.assertEqual(self.cat.speak(),
                         'The cat says "meow" ')
```
Titus Brown on TDD and BDD

"I don't do test-driven development; I do stupidity-driven testing. When I do something stupid, I write a test to make sure I don’t do it again."
Feature Testing

```python
>>> from django.test.client import Client
>>> c = Client()
>>> response = c.post('/login/',
                  {'username': 'john',
                   'password': 'smith'})
>>> response.status_code
200
>>> response = c.get('/customer/details/')
>>> response.content
'<!DOCTYPE html...'
```
In Browser Testing

- Selenium
- Twill
An idempotent deployment

- Automated - Fabric, Puppet, Capistrano, Buildout
- Isolated - virtualenv, Buildout
- Repeatable - pip, easyinstall
- Dependency Management - Yum, Deb, pip
LiveJournal Backend: Today

Roughly.

Real World Production Environment
To start with
Seperate DB machine (with pooling)
Seperate static server
Together called a web server
Then, a LB and all set to Scale
Then, scale into clusters
Many click-hosting providers

Various Heroku-like services for Python web apps (all of which support Django, and several of which focus on Django exclusively) are currently available in public or private beta:

- [http://30loops.net/](http://30loops.net/)
- [https://apphosted.com](https://apphosted.com)
- [http://www.deployfu.com/](http://www.deployfu.com/)
- [http://djangozoom.com/](http://djangozoom.com/)
- [http://www.dotcloud.com/](http://www.dotcloud.com/)
- [http://op.io/](http://op.io/)
- [http://genforma.com/](http://genforma.com/)
- [http://gerbarista.com/](http://gerbarista.com/)
- [http://gondor.io/](http://gondor.io/)
- [http://openshift.redhat.com/app/](http://openshift.redhat.com/app/)
- [http://pydra.com](http://pydra.com)
- [http://tinyflock.com/](http://tinyflock.com/)

There is also a roll-your-own solution: [http://cloudsilverlining.org/](http://cloudsilverlining.org/)
Google App Engine enables you to build and host web apps on the same systems that power Google applications. App Engine offers fast development and deployment; simple administration, with no need to worry about hardware, patches or backups; and effortless scalability.

Discover why developers are choosing Google App Engine for their projects.

Focus on your app, leave the rest to us

All the power of Google in one, simple platform.

- **Zero to sixty**: App Engine enables your application to scale automatically without worrying about managing machines.
- **Supercharged APIs**: The App Engine platform provides amazing services such as Task Queue, XMPP, and Prospective search, all powered by the same infrastructure that powers Google’s applications.
- **You're in control**: The simple, web-based dashboard makes it easy to manage your application without having to babysit it.

**Download**

Download the App Engine SDKs for Python, Java, or Go.

- [Google App Engine SDK for Go](https://developers.google.com/appengine/docs/python/quickstart) **Experimental**
- [Google App Engine SDK for Java](https://developers.google.com/appengine/docs/java/)
- [Google App Engine SDK for Python](https://developers.google.com/appengine/docs/python/)
- [Google Plugin for Eclipse](https://developers.google.com/appengine/docs/eclipse/)

**Dive Deeper**

Everything you need to know to grok App Engine.

- [Learn more about Google App Engine](https://developers.google.com/appengine/)
- [App Engine Blog](https://appengine.googleblog.com/)
- [Go Documentation](https://golang.org/doc/)
- [Java Documentation](https://docs.oracle.com/javase/8/docs/api/)
- [Python Documentation](https://docs.python.org/3/)
Custom APIs

Python

- Overview
- CGI Environment
- Backends
- Storing Data
- Services
  - App Identity
  - Blobstore
  - Capabilities
  - Channel
  - Images
  - Mail
  - Memcache
  - Multitenancy
  - OAuth
  - Prospective Search
  - Task Queues
  - URL Fetch
  - Users
  - XMPP
  - Configuration
Interactively analyze large datasets

BigQuery is a web service that enables you to do interactive analysis of massively large datasets. Scalable and easy to use, BigQuery lets developers and businesses tap into powerful data analytics on demand.

Features

- Speed - Analyze billions of rows in seconds
- Scale - Terabytes of data, trillions of records
- Simplicity - SQL-like query language, hosted on Google infrastructure
- Sharing - Powerful group-and user-based permissions
- Security - Secure SSL access
- Flexibility - REST APIs, JSON RPC, Google Apps Script

Uses

- Ad-hoc analysis
- Standardized reporting
- Data exploration
- App prototyping

BigQuery service is currently in preview and open to a limited number of enterprises and developers. Please sign up to get on the waitlist and be notified when you can start using BigQuery. For more information, take a look at the Getting Started document.
What is the Google Prediction API?

The Prediction API enables you to make your smart apps even smarter. The API accesses Google's machine learning algorithms to analyze your historic data and predict likely future outcomes. Using the Google Prediction API, you can build the following intelligence into your applications:

- Recommendation systems ([demo code](https://developers.google.com/prediction/labs/tutorial/recommendations#predicting_products))
- Spam detection ([demo code](https://developers.google.com/prediction/labs/tutorial/spam#detect_spam_email))
- Customer sentiment analysis ([demo code](https://developers.google.com/prediction/labs/tutorial/sentiment#analyzing_sentiments))
- Upsell opportunity analysis
- Message routing decisions
- Diagnostics
- Document and email classification
- Suspicious activity identification
- Churn analysis
- Language identification
- And much more...

Features

- Lightweight RESTful API
- Asynchronous training
- Automatically selects from several available machine learning techniques
- Supported inputs: numeric data and unstructured text
- Outputs hundreds of discrete categories, or continuous values
- Gallery of pre-trained prediction models
- Ability to add new training data on the fly
- Accessible from many platforms: Google App Engine, Apps Script (Google Spreadsheets), web & desktop apps, and command line

©2011 Google - Code Home - Site Terms of Service - Privacy Policy - Site Directory

Google Code offered in: English - Español - 日本語 - 한국어 - Português - Русский - 中文(简体) - 中文(繁體)
Effortless deployment for Django

Deploying a release of your Django web application should not be a time-consuming process.

With DjangoZoom

- no more tedious server setup
- one-click deployment process
- see your live app running in seconds
- get back to writing code

By far, the simplest possible Django Hosting

Sign up for the beta

Recent blog posts

- Mayan EDMS will help you OCR and manage your documents
- DjangoZoom now supports Mercurial and Subversion

Testimonials

"DjangoZoom is by far the easiest way I've found to deploy my Django projects. With other services, I spent a ton of time setting up and maintaining servers. With DjangoZoom, I get to spend my time on the things that are more important to me."
Enterprise level SLAs

AppHosted provides the tools to host your Python® WSGI compliant applications. Its Code - Deploy - Maintain workflow simplifies and expedites deployment and use of your applications. With features including auto configuration for Django, Flask, and Pylons, application performance monitoring tools, and rapid scalability, AppHosted frees you from the tasks of configuring and managing servers.

No vendor lockin. No more server provisioning delays. No more webserver management. No scalability worries. AppHosted empowers you to "Code and Go" - immediately releasing your applications to your end-users through AppHosted's distributed application hosting environment. With AppHosted, you pay only for what you use.

Signup
Gondor was designed for people who want to deploy their Django sites early and often.

Whether it's feature branches in development being deployed for review and testing, or a multi-server dedicated production stack, Gondor frees you up to focus on your site, not your infrastructure.

Gondor supports:

- command-line deployment
- unlimited domains
- revision control via git or mercurial
- dependency management using pip
- database migrations via South or nashvegas
- full backups of your entire application
- asynchronous and scheduled task execution
- full-text search using Solr and django-haystack
- caching via redis

$ pip install gondor
$ gondor create primary
deploy primary master
$ gondor list
$ gondor sqldump primary
$ gondor run primary createsuperuser
One Platform, Any Stack

Build and deploy any application to the cloud. Manage it all in one place.

Kitchen Sink included, on EC2

Sign Up Free

or  Take a Tour

Trusted by thousands of developers.

Now Supporting Every PHP Framework
We deploy and host your Python apps.

Fully managed
You don't need to be a sysadmin. We look after the servers so you can concentrate on writing code.

Works with everything
We support Django, Pylons, Pyramid, Flask, Trac and any other WSGI compatible application.

Room to grow
Our grid will intelligently assign you more servers and load balance between them when you need it.

Only pay for what you use
We give you a generous free quota, then you only pay for your bandwidth and CPU usage.

Balance of Features and Flexibility
Epio is currently invite-only, but we invite more people every week. If you’d like an invite, fill out the form below; we’ll send you an invitation when we’re ready (usually only a few weeks).

Look through our [documentation](https://example.com/documentation) and our [prices](https://example.com/prices).

Email:

Join invite list

(Got an invite? [Sign up here.](https://example.com/invite))
Monitoring

- Nagios
- Monit
- PingDom
- PagerDuty, ServiceUptime, ...
Monitor Resources, generate pretty graphs
Fire UDP Packets per each action
One more thing...
For PyCon India attendees

- epio invite code: pyconindia82731
- django-zoom priority invites!
About Me

- Active Djangonaut and active in Python world
- Part of a few popular open source django applications
  github.com/becomingGuru, stackoverflow.com/users/55562
- Co-Authored an ebook "django-design-patterns"
- Architect and develop django applications at InMobi
- Earlier, Consulting and Development via Agiliq Solutions
- Developed several custom proprietary django applications

- twitter.com/becomingGuru http://becomingguru.com
Resources

So starving:
https://github.com/agiliq/so-starving

Scaling:
"Cal_Henderson: Building Scalable Websites"
Highscalability.com, Kitchensoap.com

Performance:
"Steve_Souders: Even Faster Websites"

Cloud Hosting:
"Ken_Cochrane http://kencocharger.net/blog/"

"Jacob_Kaplan_Moss: Django in the Real World"
Image Attributions

http://www.flickr.com/photos/tejedoro_de_luz/3157690060/
http://www.flickr.com/photos/23820645@N05/4287681570/
http://www.flickr.com/photos/aidan_jones/3575000735/
http://jacobian.org/
http://sanjuancollege.edu/lib/images/philosophy_brain.jpg
http://www.flickr.com/photos/uhop/105062059/
http://geekandpoke.typepad.com/geekandpoke/images/2008/06/03/sexpl18.jpg
http://www.flickr.com/photos/go/253819/
http://aroundthesphere.files.wordpress.com/2009/05/swiss-army-knife.jpg
http://www.freefoto.com/images/41/04/41_04_9——Keep—Left_web.jpg
http://www.flickr.com/photos/orinrobertjohn/114430223/