Serverlessness

Django & Functions as a Service

Tom Dyson, DjangoCon Florence, April 2017
Me

→ Torchbox
→ Wagtail
→ @tomd, @torchbox, @wagtailcms
MailBucket

Instructions // To do // Colophon

MailBucket is an experiment in alternative methods of email management. For now its only feature is a public email-to-RSS gateway: forward your email to slurp@mailbucket.org and have your news reader pick it up at mailbucket.org/slurp.xml (where you choose slurp, having checked that it's not already in use).

The service is probably most useful to those who lurk on high-traffic mailing lists, but it could also be used as a rudimentary bridge between applications, with email as the transfer protocol.

The generated RSS feeds are valid and compressed (if your client supports compressed content); they are also public and impermanent, which may be a disadvantage for you.

Existing lists include:
xml-dev
syncato-general
cf-talk
ruby-talk
cocoa-dev
go problems
mtraffic.org

Free UK traffic reports for your mobile phone.

Road: [input]  [check traffic]

About mtraffic

Recent queries: A30 | A19 | A1079 | A64 | B1228 | A163
Yes

The frequency of our electricity supply is 50.092Hz, which means that the grid has some capacity.

© tom dyson | background
Making a web app #1

➔ Have an idea
➔ Write some code
➔ Buy a server
➔ Install the OS
➔ Install a web server
Making a web app #2

➔ Install gunicorn
➔ Read a blog article
➔ Install uwsgi
➔ Set up supervisor
➔ Set up a firewall
Making a web app #3

➡ Set up backups
➡ Configure DNS
➡ Obtain an SSL certificate
➡ Set up monitoring
➡ Launch!
→ Have an idea
→ Write some code
→ Buy a server
→ Install the OS
→ Install a web server
→ Install gunicorn
→ Read a blog article
→ Install uwsgi
→ Set up supervisor

→ Set up a firewall
→ Set up backups
→ Configure DNS
→ SSL certificate
→ Set up monitoring
→ Launch!
Have an idea
Write some code
Buy Rent a server
Install the OS
Install a web server
Install gunicorn
Read a blog article
Install uwsgi
Set up supervisor
Set up a firewall
Set up backups
Configure DNS
SSL certificate
Set up monitoring
Launch!
Have an idea ➔ Write some code ➔ Buy a server ➔ Install the OS ➔ Install a web server ➔ Install gunicorn ➔ Read a blog article ➔ Install uwsgi ➔ Set up supervisor ➔ Set up a firewall ➔ Set up backups ➔ Configure DNS ➔ SSL certificate ➔ Set up monitoring ➔ Set up CI & CD ➔ Ansible ➔ Webpack ➔ Launch!
“Serverless”
data centre → shared storage → computer →
operating system → hypervisor → container → proxy →
router → firewall →

“Serverless”
“Serverless” “FaaS”

Functions as a Service
def my_lovely_function(input):
    output = something_clever(input)
    return output
$ deploy my_lovely_function.py
$ curl https://faasbender.com/x
42
Abstract!

➔ Owned servers
➔ ‘Dedicated’ servers
➔ VMs in the cloud
➔ Containers in the cloud
➔ Functions in the cloud
Abstract!

- Owned servers
- ‘Dedicated’ servers
- VMs in the cloud
- Containers in the cloud
- Functions in the cloud / cgi?
Scale up

➔ Auto scaling
➔ Parallelisation
Scale down

→ Turn the lights off when you leave the room
HTTP events

- Logs
- S3 uploads
- Pub/sub
- Email
- Cron
Encourages better design

➔ ‘Small pieces, loosely joined’
➔ ‘Write less code’
Toolkits

➔ AWS Lambda
➔ Google Cloud Functions
➔ Azure Functions
➔ Serverless Framework
➔ Chalice
➔ Zappa
Toolkits

➔ zeit.co/now
Use cases

➔ Thumbnailing
➔ Document processing
➔ Augmenting static sites
➔ Mobile & JS apps
➔ Creating APIs
➔ Bots
Awesome CMS

A collection of 103 open and closed source Content Management Systems (CMS) for your perusal. Check out the [post](https://example.com) on the creation of Awesome CMS.

Last generated on March 31st, 2017. See [CONTRIBUTING.md](https://example.com) for details on generation and contribution.

Contents

- .NET
- Java
- JavaScript
- PHP
- Python
- Ruby
- Closed Source

Key

<table>
<thead>
<tr>
<th>Emoji</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
def get_cmss():
    page = urllib2.urlopen("https://github.com/postlight/awesome-cms")
    soup = BeautifulSoup(page.read(), "html.parser")
    p = re.compile('\\x85[[0-9,]+')
    cmss = {}
    sorted_cmss_list = []
    for cms in soup.find_all('b'):
        title = cms.text
        parent = cms.parent.parent
        try:
            stars = p.findall(str(parent))[0][1:]
Top CMSs on Github

1. Ghost ☆ 22362 stars
2. KeystoneJS ☆ 9495 stars
3. WordPress ☆ 8945 stars
4. Relax ☆ 7039 stars
5. Grav ☆ 6462 stars
6. October ☆ 5773 stars
7. django CMS ☆ 4801 stars
8. Magento ☆ 4231 stars
9. Wagtail CMS ☆ 4209 stars
10. Pagekit ☆ 4095 stars
11. Prose ☆ 3640 stars
12. Reaction ☆ 3552 stars
13. Refinery CMS ☆ 3440 stars
# Schedule

## Talks

**April 3, 2017 @ Cinema Odeon**

<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:00 a.m.</td>
<td>Registration</td>
</tr>
<tr>
<td>9:00 a.m.</td>
<td>Welcome!</td>
</tr>
</tbody>
</table>
$ pip install chalice
$ chalice new-project sched-scrape
@app.route('/

def index():
    events = scrape_schedule()
    return {‘events’: events}
$ chalice local
Serving on localhost:8000
$ chalice deploy
Deploying to: dev
https://2v15sa6qh9.execute-api.eu-west-1.amazonaws.com/dev/
DjangoCon Florence 2017

April 3, 2017 at Cinema Odeon

- 9:00 a.m. Welcome!
- 9:15 a.m. Setting up a Python Community in Zimbabwe Anna Makazude & Humphrey Butau
- 10:00 a.m. The road to unempathic communities is paved with good intentions Erik Romijn
- 10:45 a.m. Services, Architecture & Channels Andrew Godwin
- 11:15 a.m. Coffee Break
- 11:45 a.m. The Struggle of Tech: Feeling Better as a Learner Gloria Dwonoh
- 12:15 p.m. Level up! Rethinking the Web API framework. Tom Christie
- 12:45 p.m. Planet friendly web development with Django Chris Adams
- 1:15 p.m. Lunch
- 2:30 p.m. The art of interacting with an autistic software developer Sara Peeters
- 3:00 p.m. Django’s watching my back(end) Carlos de las Heras
- 3:30 p.m. Staying DRY(ER) when working with Django and frontend frameworks/libraries Emma Decescolle
- 4:00 p.m. Coffee break
- 4:30 p.m. Radio Free Django - Building a radio station on Django Mark Steadman
/schedule
/schedule tuesday
Serverless Django
Serverless & Django

➔ Shrink your Django project
➔ Use FaaS to flatten the bumps
➔ Move, cautiously, to microservices
➔ Consider JAMStack for CMS projects
➔ Handle more varied events
Caveats

➔ Warm up time
➔ Platform maturity
➔ No cost cap
➔ Persistence
“This is, perhaps, the final state of computing as we know it, with massive clouds abstracting their infrastructure services at such a high level that no one even knows there is an operating system or such things as middleware, databases, or web servers.”

Timothy Prickett Morgan
Thank you