

What makes Python AWESOME?

Daniel Greenfeld



[ABOUT US](#) | [PROJECTS](#) | [CAREERS](#) | [TECHNOLOGY](#)



Disney!

Portability

Command-line
Web App
Desktop App

- Windows
- Linux/Unix
- OS X
- Google App Engine
- JVM
- .Net

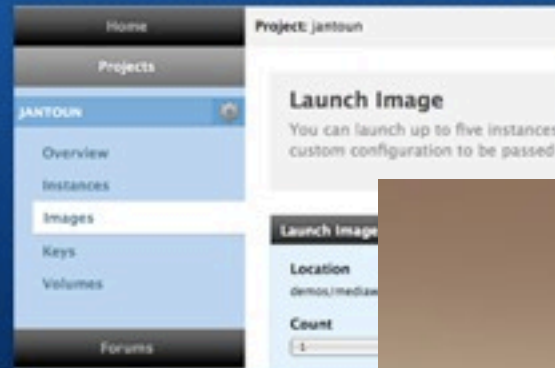


NEBULA Cloud Computing Platform

[HOME](#)[ABOUT](#)[SERVICES](#)[BLOG](#)

Announcing Nebula IaaS Launch

The pre-release of Nebula IaaS is available to all Agency personnel. The pre-release will be seamlessly transitioned to production after the Operational Readiness Review (ORR) is completed in the coming weeks.



Getting Started

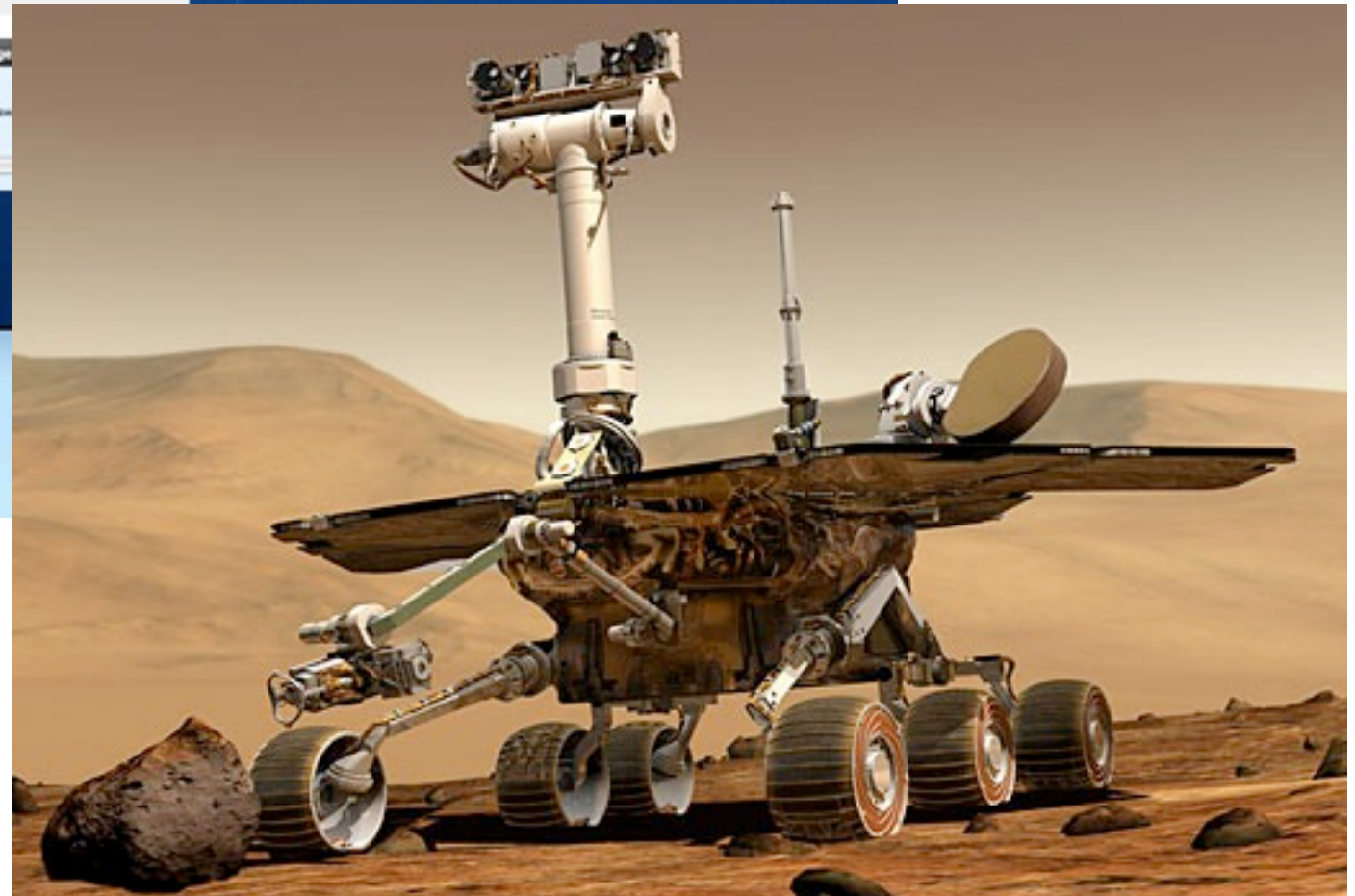
Learn how to harness the power of cloud computing to reach science goals faster.



Rapid Deployment



Scalable



NASA

Science and websites

Batteries Included*

Strings

Data Types

Math

File and Directory Access

Data Persistence

Data Compression

File Formats

Logging

Specific OS Services

Interprocess Communication

Internet Data

Internet Protocols

Structured Markup

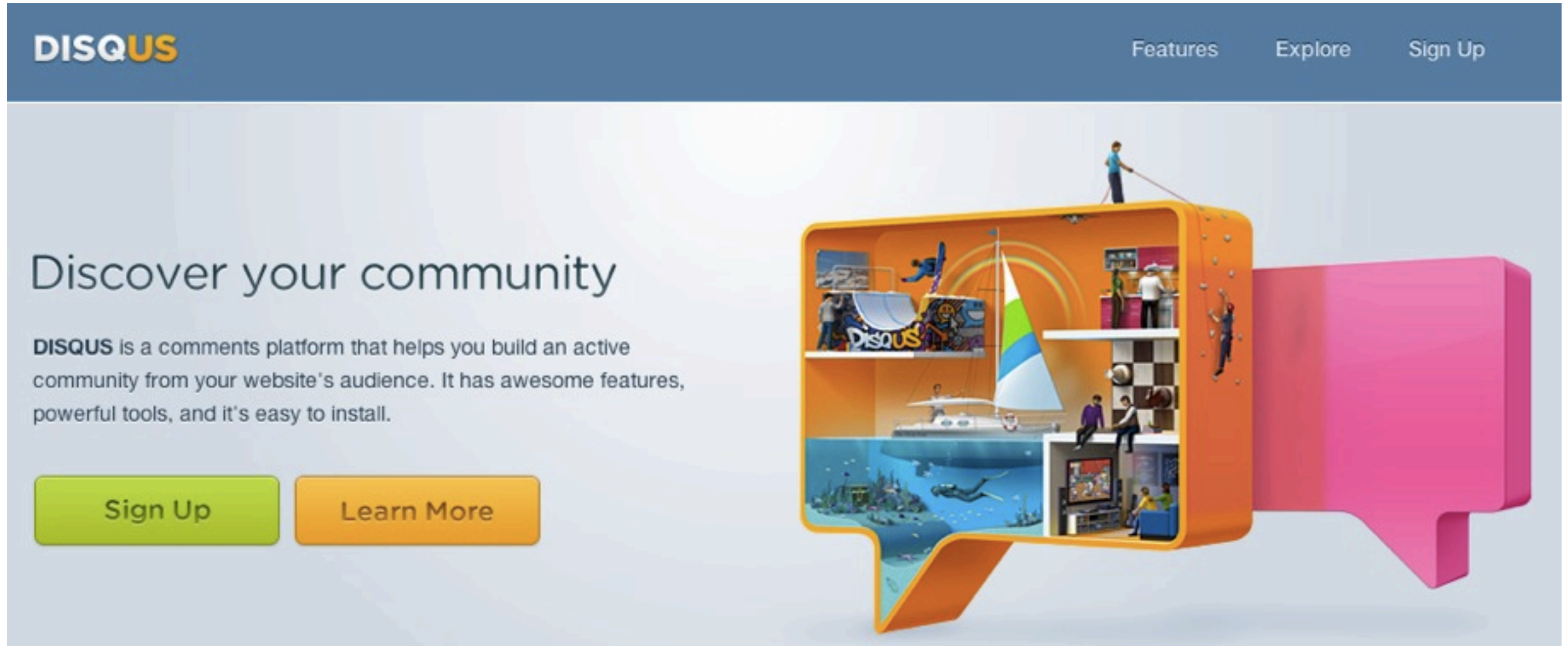
Multimedia

Internationalization

Testing

*These are categories - not libraries!

Your comments...



Python can handle gigantic scope efforts

Easy to learn

```
from datetime import datetime
```

```
name = “Daniel Greenfeld”
```

```
started_python = datetime(2005, 11, 1)
```

```
if name.startswith(“Daniel”):
```

```
    print(name)
```

```
for i in range(10):
```

```
    print(i)
```

Easy to learn

Daniel Greenfeld

0

1

2

3

4

5


6

7



8



9

Powers the high profile




SEARCH

SHOP WITH US 

TRAVEL WITH US 


TV SHOWS ▾ TV SCHEDULES ▾ NEWS ▾ VIDEO ▾ PHOTOS ▾ GAMES ▾ BLOGS ▾ TOPICS ▾ NEWSLETTERS ▾ KIDS ▾ SHOP ▾





Torpedo Tastic!

The MythBusters reveal why they did what they did in this week's episode.

NEW! Video: [MythBusters Aftershow](#)
Video: [Behind-the-Scenes Interviews](#)
Photos: [Kari Byron](#)
[More »](#)





Buzzword Compliant

- Object oriented
- Functional
- Procedural
- Multiple Inheritance
- Interfaces
- Duck-typing

In our neighborhood

The screenshot shows the homepage of the EveryBlock Los Angeles website. At the top, the browser address bar displays 'la.everyblock.com'. The main header features the 'EveryBlock' logo in white and green, followed by 'Los Angeles' in green. A search bar below the header contains the placeholder text 'e.g., 200 N. Spring St, 90064, Downtown' and a 'Search' button. In the top right corner, there is a 'Log in' button and a star icon. The background of the header section features a dark city skyline silhouette with various icons: a yellow flag with a grid pattern, a blue flag with a grid pattern, a newspaper with 'EXTRA! EXTRA!' written on it, two silver cans connected by a string, and a blue starburst with a yellow '#1' inside. Below the header, there are four main content blocks arranged horizontally. On the far left, a vertical orange banner reads 'HOW IT WORKS'. The first block is titled 'Follow your favorite places' and describes picking a neighborhood, block, or ZIP to create a personalized area. The second block is titled 'Learn what's happening' and describes reading nearby news and getting updates via email or a custom homepage. The third block is titled 'Share with neighbors' and describes starting a discussion or sharing an announcement. The fourth block is titled 'Your block gets better' and describes exchanging ideas, gaining recognition, and solving problems to make the block a better place.

la.everyblock.com

Log in

EveryBlock Los Angeles

e.g., 200 N. Spring St, 90064, Downtown Search

HOW IT WORKS

Follow your favorite places
Pick a neighborhood, block or ZIP — or create a personalized area. Sign up for one or many.

Learn what's happening
Read nearby news from and hundreds of sources. Get updates via e-mail or your custom homepage.

Share with neighbors
Start a discussion, share an announcement, ask your neighbors a question, or answer one of theirs.

Your block gets better
Exchange ideas. Gain recognition. Solve problems. Make your block a better place.

Speed

- Development?
- Performance?

Anecdotal Development speed

“I code about 5x faster in Python than in Java.”

-me

YMMV

Development speed

- Less boilerplate
- Straightforward syntax
- Whitespace defines blocks

Performance Metrics

Benchmarks are as
accurate as elections...

but...

Python is one of the faster
dynamic languages. Comparable
to Perl and Ruby.

Type to search

Ubuntu is a fast, secure and easy-to-use operating system used by millions of people around the world.

Get Ubuntu

It works with your favourite apps:



Explore features ›

11.04
is here!



Ubuntu

Canonical

Package management

- <http://pypi.python.org>
- <http://djangopackages.com>

Web Frameworks

Django
Pyramid
Flask


... or write your own!

mozilla
Firefox

FEATURES MOBILE ADD-ONS SUPPORT ABOUT

visit mozilla.org

Register or Log in Other Applications ▾


 **ADD-ONS**
2,457,356,018 add-ons downloaded

search for add-ons →


Categories

- Alerts & Updates
- Appearance
- Bookmarks
- Download Management
- Feeds, News & Blogging
- Games & Entertainment
- Language Support
- Photos, Music & Videos
- Privacy & Security
- Shopping
- Social & Communication


Discover great add-ons for everyone

 **Echofon for Twitter**
Echofon (formerly TwitterFox) notifies you of your friends' tweets on Twitter. ...

Continue to Download →
Featured

 **ColorfulTabs**
The most beautiful yet the simplest add-on that makes a strong colorful appeal. ...

↓ Download Now
Featured

 **Flagfox**
Displays a country flag depicting the location of the current website's server ...

↓ Download Now
Featured

Like these? Find more add-ons in the [Rock Your Firefox - September 2010](#) collection.

◀ Introduction Rock Your Firefox Privacy Web Development Travel ▶

Mozilla.org from PHP to Django
cut the source from 44K to 12K loc

More features

- Simplified memory management
- Name Spaces
- Monty Python humor

Science? Math?

SciPy



Download



Getting Started



Documentation



Report Bugs



Read the Blog

**ENTHOUGHT**
SCIENTIFIC COMPUTING SOLUTIONS

EPD repository | code.enthought.com | www.enthought.com

PRODUCTS | CONSULTING | TRAINING | SECTORS

**Enthought Python Distribution 7.0**
DOWNLOAD | PURCHASE

**Software Developer | Web Services**
MULTIPLE OPENINGS

**EuroPython 2011**
JUNE 20-26 | FLORENCE, ITALY

**SciPy 2011**
JULY 11-16 | AUSTIN, TX

**PyCon Au 2011**
8.20-21 | SYDNEY, AU

**Kiwi PyCon 2011**
8.27-28 | WELLINGTON, NZ

**ENTHOUGHT TRAINING**
ENTHOUGHT OPEN COURSE: 75+ Libraries
SciPy, NumPy, MKL
3 MODULES INCLUDED:
Python for Scientists & Engineers 3-DAY
Interfacing with C/C++/Fortran 1-DAY
Introduction to UIs & Visualization 1-DAY
EPD 7.0
REGISTER NOW: 512.536.1050

**PRODUCTS**
The Enthought Python Distribution provides a platform environment for scientific computing programming language. A single-click installer allows over 75 libraries and tools. Our open source initiative is the Enthought Tool Suite.

**CONSULTING**
We work with you to develop software for interactive data manipulation and visualization. In addition to full application development, we also offer short-term consultation to address critical technical computing needs. Our areas of expertise include geoscience, financial analysis, 3D modeling, fluid dynamics and microrheology.

 **matplotlib**

[home](#) | [search](#) | [examples](#) | [gallery](#) | [docs](#) »

intro

matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. matplotlib can be used in python scripts, the python and [ipython](#) shell (ala MATLAB® or Mathematica®), web application servers, and six graphical user interface toolkits.

matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc, with just a few lines of code. For a sampling, see the [screenshots](#), [thumbnail gallery](#), and [examples](#) directory

Culture of Documentation

Python v2.7.1 documentation »

modules index

Project Versions
latest

Download
Download these documents

Docs for other versions
Python 2.6 (stable)
Python 3.2 (stable)
Python 3.3 (in development)
Old versions

Other resources
FAQs
Guido's Essays
New-style Classes
PEP Index
Beginner's Guide
Book List
Audio/Visual Talks
Other Doc Collections
Report a Bug

Python v2.7.1 documentation

Welcome! This is the documentation for Python 2.7.1, last updated Apr 20, 2011.

Parts of the documentation:

What's new in Python 2.7?
or all "What's new" documents since 2.0

Tutorial
start here

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs
in-depth documents on specific topics

Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

Installing Python Modules
information for installers & sys-admins

Distributing Python Modules
sharing modules with others

Documenting Python
guide for documentation authors


FAQs
frequently asked questions (with answers!)

Indices and tables:

Quick search

Brought to you by
Read the Docs

Python gave us a win



confidox

About Confidox | [FAQ](#) | [Daniel Greenfeld](#) | [Logout](#)

Securely connecting legal talent with employers


- Attorneys and Paralegals: Find high-quality legal talent
- Law firms: Find high-quality legal talent

Confidox is a secure, confidential platform for finding legal talent. It connects law firms with attorneys, paralegals, and other legal professionals. You can find opportunities or law office staff with the confidence that your information is protected.

[Sign Up](#)

For Law Firms

Looking to hire? Then create a job list. We'll help you find lawyers, paralegals, assistants, secretaries, expert witnesses, and other staff members that you need.




confidox

About Confidox | [FAQ](#) | [Daniel Greenfeld](#) | [Logout](#)

[All Profiles](#)

Daniel Greenfeld's Profile



Law School: Harvard Yale





Years of experience: 3-5 years

Areas of Law:

[Daniel Greenfeld](#)

(Already notified)

Looking for work as a **None** in or near Los Angeles

- [Resume](#) 
- [Samples](#) 
- [Transcript](#) 
- [Salary](#) 

[Who contacted me?](#)

[Edit basic info](#)

[Edit Resume information](#)

[Edit Avatar and Payment](#)

Python gave us a win

 **confidox**

[About Confidox](#) | [FAQ](#) | [Daniel Greenfeld](#) | [Logout](#)

[All Profiles](#)

Daniel Greenfeld's Profile



Law School: Harvard Yale

Years of experience: 3-5 years

Areas of Law:

- [Resume](#)
- [Samples](#)
- [Transcript](#)
- [Salary](#)

[Daniel Greenfeld](#)

(Already notified)

Looking for work as a **None** in or near Los Angeles

```
url(regex=r'^(?P<slug>[-\w]+)/$',
    view=login_required(DetailView.as_view(
        model=Profile,
        template_name='profiles/profile_detail.html')),
    name='profile_detail',
),
```

```
88
89 <div class="span-5 push-1 append-1">
90     {% if request.user == profile.user %}
91         {% with profile.contact.all as contacts %}
92             {% if contacts|length %}
93                 {% if profile.role == "resource" %}
94                     <div class="box">
95                         <h4><a href="#" id="contacted-me-link">Who contacted me?</a></h4>
96                         <div
97                             style="display: none;"
98                             id="contacted-me-popup">
99                             <p><i>(click name to ping back)</i></p>
100                             <ul>
101                                 {% for contact in contacts %}
102                                     <li><a href="{% url profile_pingback contact.username %}">{{
103                                     {% endfor %}
104                                 </ul>
105                             </div>
106                         </div>
107                     {% endif %}
108                 {% endif %}
109             {% endwith %}
110             <div class="push-1">
111                 {% if profile.role == "resource" %}
112                     <p><a href="{% url profile_edit_page_1 %}">Edit basic info</a></p>
113                     <p><a href="{% url profile_edit_page_2 %}">Edit Resume information</a></p>
114                     <p><a href="{% url profile_edit_page_3 %}">Edit Avatar and Payment</a></p>
115                 {% else %}
116                     <p><a href="{% url company_edit_page_3 %}">Edit Profile and Payment</a></p>
117                 {% endif %}
118             </div>
119         {% endif %}
120     </div>
121
122
123 {% endblock %}
```

Zen of Python

```
>>> import this  
The Zen of Python, by Tim Peters
```

```
Beautiful is better than ugly.  
Explicit is better than implicit.  
Simple is better than complex.  
Complex is better than complicated.  
Flat is better than nested.  
Sparse is better than dense.  
Readability counts.  
Special cases aren't special enough to break the rules.  
Although practicality beats purity.  
Errors should never pass silently.  
Unless explicitly silenced.  
In the face of ambiguity, refuse the temptation to guess.  
There should be one-- and preferably only one --obvious way to do it.  
Although that way may not be obvious at first unless you're Dutch.  
Now is better than never.  
Although never is often better than *right* now.  
If the implementation is hard to explain, it's a bad idea.  
If the implementation is easy to explain, it may be a good idea.  
Namespaces are one honking great idea -- let's do more of those!
```

Zen of Python

```
>>> import this  
The Zen of Python, by Tim Peters
```

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Flat is better than nested.

Sparse is better than dense.

Readability counts.

Special cases aren't special enough to break the rules.

Although practicality beats purity.

Errors should never pass silently.

Unless explicitly silenced.

In the face of ambiguity, refuse the temptation to guess.

There should be one-- and preferably only one --obvious way to do it.

Although that way may not be obvious at first unless you're Dutch.

Now is better than never.

Although never is often better than *right* now.

If the implementation is hard to explain, it's a bad idea.

If the implementation is easy to explain, it may be a good idea.

Namespaces are one honking great idea -- let's do more of those!

Python Jobs are
plentiful!