Why Web Apps?

- Cross Platform
- New Features in HTML 5
- Easy to share and make social

Why Python?

- Dynamic languages are productive
- Performance is more than fast enough
- Alternatives? PHP, Perl, Ruby

Python's Benefits

- Clean syntax 
- Object oriented
- Extensive libraries

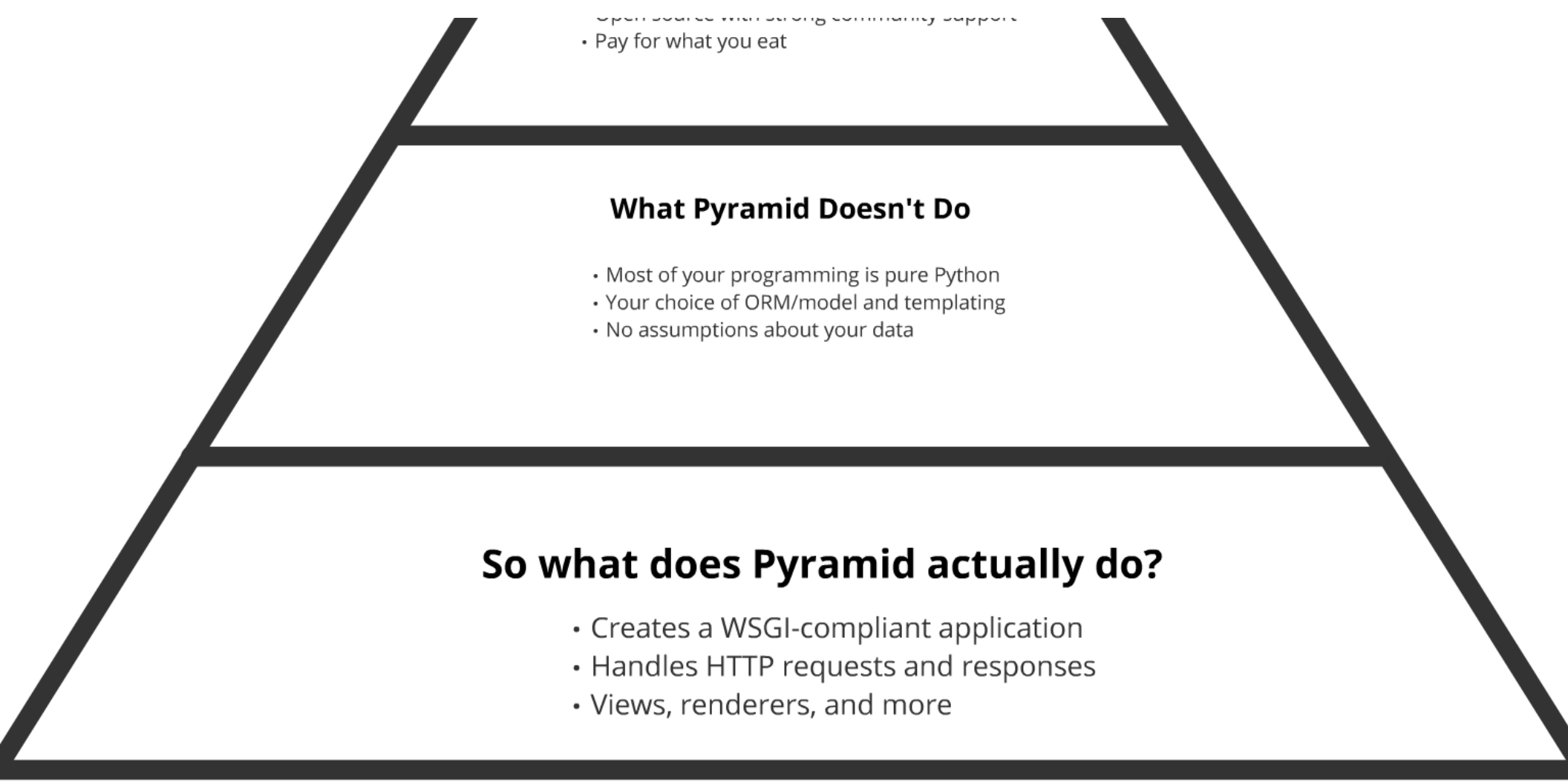
```
# prints lines from a file in reverse order
def printlines(filepath):
    rawdata = open(filepath).read()
    lines = rawdata.splitlines()
    lines.reverse()
    for line in lines:
        print line
```

Why Pyramid?

- Fast, 100% test coverage, well-documented
- Open source with strong community support
- Pay for what you eat

What Pyramid Doesn't Do

- Most of your programming is pure Python
- Your choice of ORM/model and templating
- No assumptions about your data


• Pay for what you eat
What Pyramid Doesn't Do
• Most of your programming is pure Python
• Your choice of ORM/model and templating
• No assumptions about your data
So what does Pyramid actually do?
• Creates a WSGI-compliant application
• Handles HTTP requests and responses
• Views, renderers, and more

Need another reason?

We have an awesome t-shirt

Your First Pyramid App

Hello, World!

```
from paste.httpserver import serve
from pyramid.config import Configurator
from pyramid.response import Response

def hello_world(request):
    return Response('Hello world!')

if __name__ == '__main__':
    config = Configurator()
    config.add_view(hello_world)
    app = config.make_wsgi_app()
    serve(app, host='0.0.0.0')
```

File Layout

Simple

~/myapp/__init__.py
~/myapp/models.py
~/myapp/views.py
~/myapp/templates/
~/myapp/static/

More Advanced

~/myapp/__init__.py
~/myapp/models/__init__.py
~/myapp/models/security.py
~/myapp/models/blog.py
~/myapp/views.py
~/myapp/routes.py
~/myapp/templates/
~/myapp/static/

Code Examples

Views

```
from myapp.models import Users
from pyramid.security import authenticated_userid

@view_config(renderer='mytemplate.html', route_name='sayhello')
def welcome(request):
    userid = authenticated_userid(request)
    user = Users.retrieve(userid)
    return {'name': user.name}
```

Template Before Rendering

```
<html>
  <body>
    Welcome back, ${name}!
  </body>
</html>
```

Template After Rendering

```
<html>
  <body>
    Welcome back, Arthur!
  </body>
</html>
```

Make Route Patterns with URLs

/myblog/{post_title}
/data/{author}/bio

Add the route to config

```
config.add_route('blogpost', '/myblog/{post_title}')
```

config.add_route('blogpos

Associate the Route with a Function

```
@view_config(renderer='blogpost.html',
                        route_name='blogpost')
def myfunction(request):
    ...
```

The Model

- Relational database (PostgreSQL, MySQL, SQLite)
- NoSQL (MongoDB, CouchDB, Riak)
- Flat-files (XML, JSON, ZODB)

Introducing:

SQLAlchemy

SQLAlchemy Lets You...

- Model your data
- Interact with a relational database
- Use Python code and objects

```
# models.py

class BlogPost(Base):
    id = Column(Integer, 'id')
    title = Column(Unicode(100), 'title')
    body = Column(UnicodeText(), 'body')

    def __init__(self, title, body):
        self.title = title
        self.body = body

    @classmethod
    def getpost(cls, title):
        post = Session.query(cls).filter(cls.title==title).first()
        return post
```

```
# Now your view can look up blog posts

@view_config(renderer='blogpost.html', route_name='blogpost')
def displaypost(request):
    title = request.matchdict['post_title']
    post = BlogPost.getpost(title)
    message = 'hi there'
    return {'post': post, 'message': message }
```

```
<!-- blogpost.html -->
<html>
  <body>
    <div>${post.title}</div>
    <div>${post.body}</div>
      <p>And the message is: ${message}</p>
  </body>
</html>
```

Where to go from here

Choose the right level of abstraction

- How much control do you need?
- "Low level" vs "High level"
- Frameworks that use language subsets
- Building a high level framework with Pyramid

Too much boilerplate?

- Libraries: Deform, repoze.tm, Beaker
- Paster Templates (scaffolds)
- Python modules

Transferable Skills

- Pyramid encourages good design
- MV or MVC architecture
- SQLAlchemy
- Unrestricted Python

Use your newfound Python skills for....

- System administration
- Commandline/GUI utilities
- Data analysis
- And of course: web development

A word on web hosts

- Learn more at pylonsproject.org
- Visit #pyramid on freenode