Freie Universität Berlin

Webframeworks für Python

Vorstellung und Vergleich

Andreas Knöpfle, Bastien Sachs und Tobias Schmid

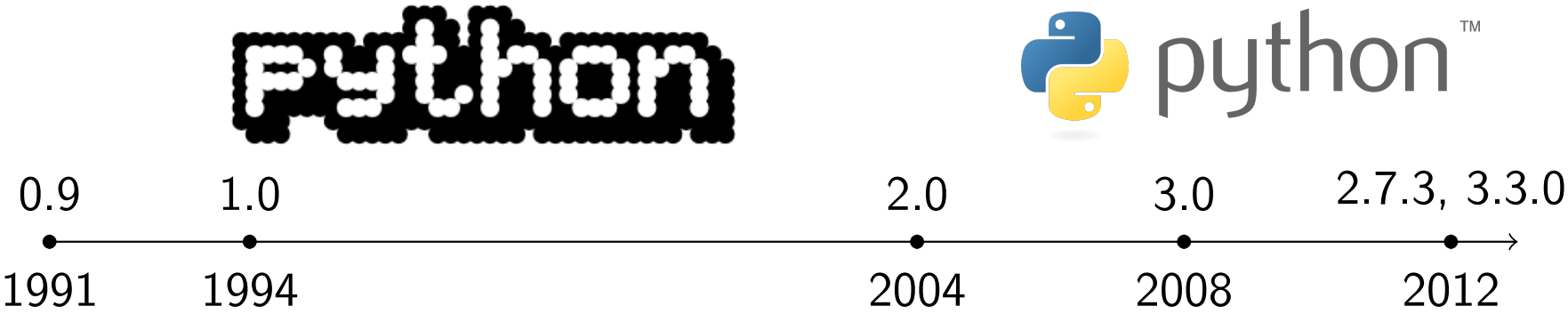
Institut für Informatik

28. November 2012

Einstiegs-Idee:
viele Logos der Webframeworks für Python
Stichpunkte:
- In Python geschriebene Webframework gibt es wie „Sand am Meer“
- Warum? Einblick gibt unser Vortrag

Gliederung

Python – Programmiersprache fürs Web?



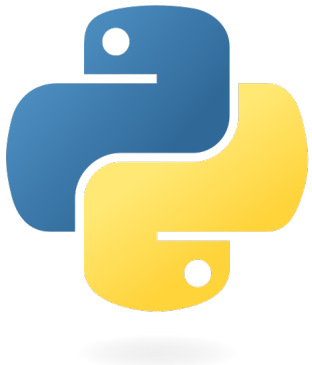
- Entwurfsphilosophie betont Programmlesbarkeit
  - Blöcke durch gleiche Einrückung begrenzt
  - relativ wenige Schlüsselwörter
- Multiparadigmen-Sprache (primär OOP, funktional, AOP)
- dynamische Datentypen, garbage collection
- große, umfangreiche Standardbibliothek “batteries included”

Syntaxvergleich zu Java

```java
int value = 0;
String title = String.valueOf(value);
if ("0".equals(title)) {
    throw new Exception("Error");
}
for (int i = 1; i < 10; i++) {
    System.out.printf("row %s", i);
}
```

Quelle: [??]

```python
value = 0
title = str(value)
if title == "0":
    raise Exception("Error")
for i in range(1, 10):
    print "row {}".format(i)
```



Quelle: [??]

Web Server Gateway Interface (WSGI)

* Apache (mod_wsgi), nginx (uWSGI), Gunicorn

Bild

Python – Programmiersprache fürs Web!

Zusammenfassung der genannten Punkte

Webframeworks für Python (Auswahl)

Full-Stack Frameworks

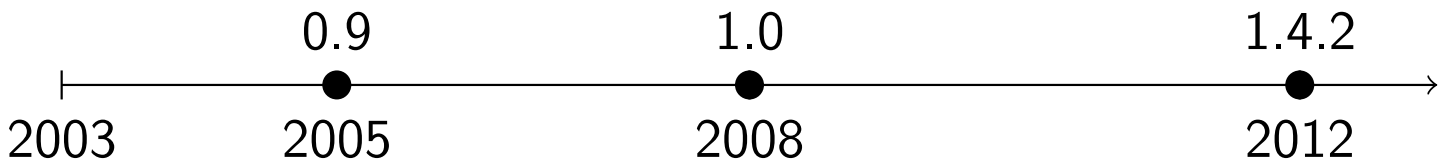
- Django
- TurboGears
- web2py
- Pylons/Pyramid
- ...

Microframeworks

- Bottle
- CherryPy
- ...

Django

> *"The web framework for perfectionists with deadlines"*



Farbdeutung bei Vergleichen

Hauptteil:
Vor- und Nachteile einiger weniger Frameworks aufzeigen
(konkrete) Lösungsansätze für bestimmte Probleme/Vergleichskriterien

Persistenz

SQL-ORMs

- Django built-in ORM (Django)
- SQLAlchemy (Pylons)
- SQLObject (Pylons,TurboGears)
- DAL (web2py)
- ...

MongoDB ORMs

- MongoDB Engine (Django)
- Ming (TurboGears)

Full-Stack Frameworks haben entweder eigenes ORM oder ermöglichen den Einsatz von *SQLAlchemy* oder *SQLObject*.

Persistenz Beispiel Django

```
from django.db import models

class Employee(models.Model):
    name = models.CharField(max_length=60)
    boss = models.BooleanField(default=False)
    concern = models.ForeignKey(Concern)

class Concern(models.Model):
    name = models.CharField(max_length=40)
```

```
concern = Concern.objects.create(name="Musterfirma")
Employee.objects.create(name="Mustermann", concern=concern)

# Alle Mitarbeiter des Konzerns zum Chef machen
Employee.objects.filter(concern=concern).update(boss=True)
```

Templates

Templates in Python

- Text-basiert:
  - Django template language
  - Cheetah
  - Mako (füher Myghty)
  - Jinja2
- XML-basiert:
  - Genshi
  - SimpleTAL

Es gibt noch sehr viel mehr Template-Engines
(siehe `http://wiki.python.org/moin/Templating`)

Templates

Django template language

- Template-Inheritance
- Unterstüzt Filter und Kontrollstrukturen

```
<body>
    <div id="content">
        {% block content %}{% endblock %}
    </div>
</body>
```

```
{% extends "base.html" %}

{% block content %}
    Inhalt: {{ content }}
{% endblock %}
```

Templates

Cheetah

- Wird von allen Full-Stack-Frameworks unterstützt
- Große und aktive Community
- Templates können von Python-Klassen abgeleitet werden

```
<table>
   #for $client in $clients
      <tr>
         <td>$client.surname, $client.firstname</td>
         <td><a
    href="mailto:$client.email">$client.email</a></td>
      </tr>
   #end for
</table>
```

I18N und L10N

- gettext

Konfiguration, Routing

Django

- URLconf (URL Konfiguration)
- einfaches Mapping zwischen URL-Patterns (Regex)
- Wenn der Ausdruck passt, ruft Django den View auf
- settings.py für alle weiteren Konfigurationen
  - Datenbankeinstellungen
  - E-Mail und Fehlermeldungseinstellungen
  - i18n und URL Einstellungen
  - Applikations and Middleware Einstellungen

Konfiguration, Routing

TurboGears

- Object Dispatch, und built in Routes Integration
- kann überschrieben werden

Routing Beispiel Django

```
urlpatterns = patterns('news.views',
    (r'^articles/2003/$', 'special_case_2003'),
    (r'^articles/(\d{4})/$', 'year_archive'),
    (r'^articles/(\d{4})/(\d{2})/$', 'month_archive'),
    (r'^articles/(\d{4})/(\d{2})/(\d+)/$', 'article_detail'),
)
```

```
/articles/2005/03/      => month_archive(request,'2005','03')
/articles/2005/3/       => no match
/articles/2003/         => special_case_2003(request,'2003')
/articles/2003          => no match
/articles/2003/03/03/   =>
    article_detail(request,'2003','03','03')
```

Routing Beispiel TurboGear 2.0

```
def setup_routes(self):
    map = Mapper(
        directory=config['pylons.paths']['controllers'],
        always_scan=config['debug'])

    # Setup a default route for the root of object dispatch
    map.connect('*url', controller='root',
        action='routes_placeholder')
    config['routes.map'] = map
```

Vergleich Konfiguration, Routing

	Django	TurboGear 2.0
Konfiguration		
Routing	mächtig, evtl. Overload	einfach,anpassbar

Formulare, Validierung

- django.forms
  - HTML form widget
  - Field validation
  - ...

Sicherheitmechanismen

- django: http://www.djangobook.com/en/2.0/chapter20.html

Bootstrapping, Scaffolding, Erweiterbarkeit

Extras: WebServices, Caching, Tests

Quelle: [2]

"...microframework for Python based on Werkzeug, Jinja 2 and good intentions..."

- Was kann Flask ?
  - Entwicklungsserver and Debugger
  - Unittesting wird unterstützt
  - RESTful
  - Benutzt Jinja2 als Template-Sprache
  - Secure-Cookies
  - WSGI 1.0 konform

Flask

hello.py:

```
  from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello World!'

if __name__ == '__main__':
    app.run()
```

```
    $ python hello.py
  * Running on http://127.0.0.1:5000/
```

Tornado

bottle

web.py

Kriterienübersicht

* Vergleichstabellen (Django vs. ...)

Fazit

Je Anforderungen an das Webframework ("Taste")

...

Quellen

- http://www.infoworld.com/d/application-development/pillars-python-six-python-web-frameworks-compared-169442
- http://wiki.python.org/moin/WebFrameworks
- http://wiki.python-forum.de/Web-Frameworks
- http://blog.ianbicking.org/turbogears-and-pylons.html

alle URLs aufgerufen am 20. November 2012.

Quellen der Abbildungen

1. Innenhof Informatik `http://www.flickr.com/photos/bennybenny/3597853896/`
2. Fask Logo `http://flask.pocoo.org/static/logo.png`

alle URLs aufgerufen am 14. November 2012.

Ende der Präsentation