

# *Web 2. Gang*

## *Praktische Einführung in die Entwicklung von Webanwendungen mit TurboGears*

Python User Group Köln

13.12.2006

Christopher Arndt <chris@chrisarndt.de>

---

---

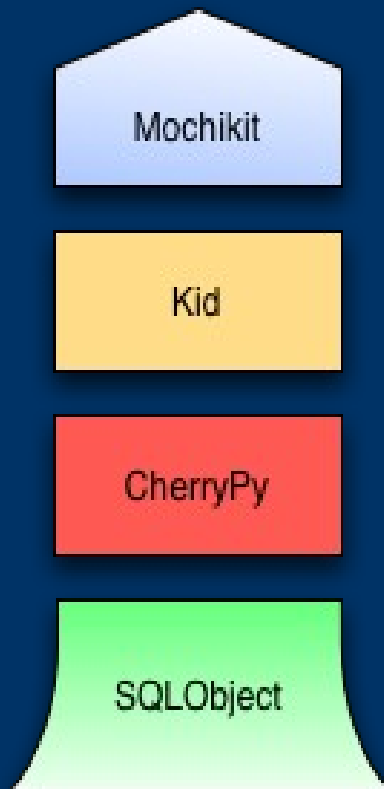
# *Was ist TurboGears?*

- Ein Python Web-Meta-Framework!
  - Vergleichbar mit Django und Ruby-on-Rails
  - Open Source (MIT Lizenz)
  - Relativ jung (1. öffentliche Version Herbst 2005)
  - Buzzword-kompatibel: MVC, AJAX(J), REST etc.
- 
-

# *Was kann ich mit TurboGears machen?*

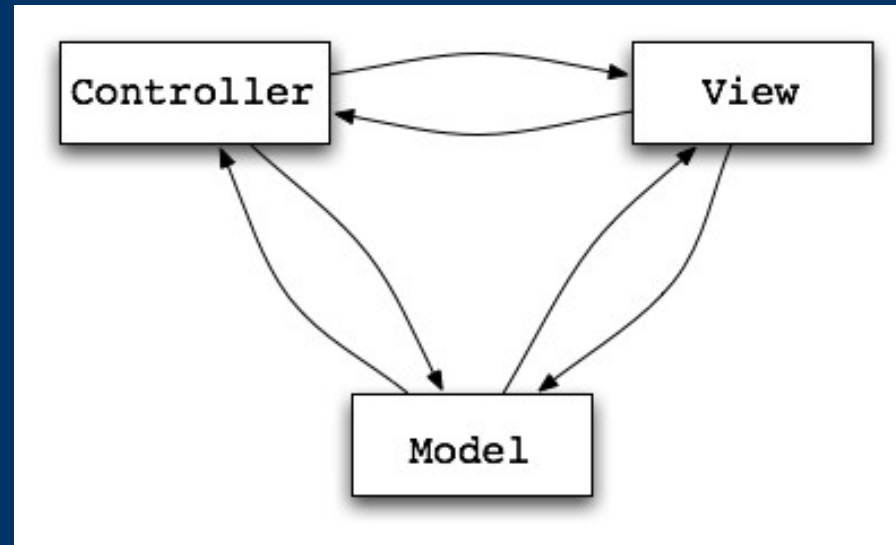
- „Klassische“ Webanwendungen, z.B. Blogs, Wikis, CM Systeme
  - Intranetanwendungen, z.B. *WhatWhat Status*
  - Webadministrations-Werkzeuge, z.B. *WebFaction.com Control Panel*
  - „Microapps“ (<http://microapps.org>) à la *Gravatar.com, Websnapr.com, etc.*
- 
-

# Welche Bestandteile hat TurboGears?



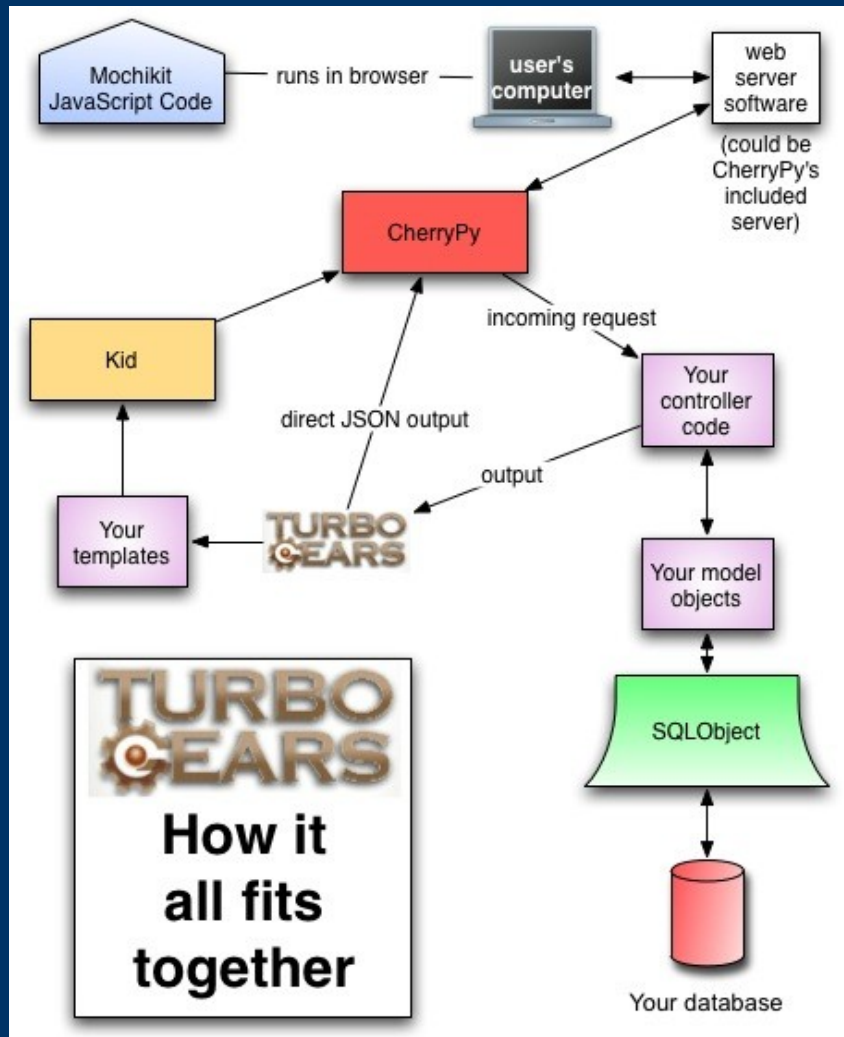
- Datenbankabstraktion: *SQLObject*
- Applikations-Server: *CherryPy*
- Template-Engine: *Kid*
- Client-side JavaScript: *MochiKit*
- weitere Komponenten:  
*FormEncode* (Validierung),  
*Nose* (Unit tests),  
*simplejson* (JSON) u.v.m.

# Was hat es mit MVC auf sich?



- MVC = Model, View, Controller
- deutsch: Modell, Repräsentation, Steuerung
- Trennung der Komponenten der (Web)Anwendung u.a. zum leichteren Austausch

# MVC auf TurboGears-Art



- controllers.py
- model.py
- templates/\*.kid

# 10 Schritte zur TG-Anwendung

1) Schnellstarte dein Projekt:

```
tg-admin quickstart
```

2) Code dein Datenmodell:

```
editor model.py
```

3) Erzeuge die Datenbank:

```
tg-admin sql create
```

---

---

## *10 Schritte zur TG-Anwendung (II)*

- 4) Starte die toolbox und füge einen Benutzer und einige Daten hinzu:

```
tg-admin toolbox
```

- 5) Designe deine URLs.
  - 6) Schreibe controller-Klassen für jedes Objekt deines Datenmodells mit Methoden zum Auflisten, Ansehen und Erzeugen/Editieren.
- 
-

## 10 Schritte zur TG-Anwendung (III)

- 7) Schreibe eine `index()` Methode um die Objekte aufzulisten.  
(`controller.py`, `templates/<object>/list.kid`)
  - 8) Schreibe eine `edit()` Methode um das Editier-Formular anzuzeigen.  
(`controller.py`, `widgets/<object>form.py`,  
`templates/<object>/edit.kid`)
  - 9) Schreibe `<action>()` Methoden um Objekte hinzuzufügen, zu löschen und zu ändern.  
(`controller.py`)
- 
-

# *10 Schritte zur TG-Anwendung (IV)*

10) Schreibe deine Templates.

**Kleinarbeit: ;-)**

- CSS
- JavaScript
- Deployment
- ...



# Beispiel: eine Lesezeichen-Datenbank

Lesezeichen-Modell: *Titel, URL, Beschreibung, Tags (Schlagwörter)*

```
class Bookmark(SQLObject):  
  
    title = UnicodeCol(length=255, notNull=True)  
    url = UnicodeCol(length=255, notNull=True)  
    description = UnicodeCol()  
  
    tags = RelatedJoin('Tag', orderBy='name')  
  
    # meta data  
    created = DateTimeCol(default=datetime.now)  
    owner = ForeignKey('User', notNull=True)
```

---

---

# Lesezeichen-Modell (Tags)

```
class Tag(SQLObject):  
  
    name = UnicodeCol(length=100, notNull=True)  
  
    bookmarks = RelatedJoin('Bookmark',  
                             orderBy='-created')  
  
    # meta data  
    owner = ForeignKey('User', notNull=True)  
    created = DateTimeCol(default=datetime.now)
```

---

---

# URLs

- <http://mysite/bookmarks/> → Liste der Lesezeichen

<http://mysite/bookmarks/list>

- [/bookmarks/<id>](#)

[/bookmarks/<id>/view](#)

- [/bookmarks/<id>/edit](#)

[/bookmarks/<id>/add](#)

Anzeige des Lesezeichens /  
Anzeige des Edit-Formulars

- [/bookmarks/<id>/delete](#) → Löschen des Lesezeichens

- [/bookmarks/<id>/update](#) → Ändern des Lesezeichens

# Controller Methoden (I)

Liste der Lesezeichen:

```
class BookmarksController(Controller)

    @expose(template=
        'bookmarker.templates.bookmarks.list')
    def index(self):
        bookmarks = Bookmark.select()
        return dict(entries=bookmarks)

list = index
```

---

---

# Templates (I)

Liste der Lesezeichen:

```
<?python item = tg.ipeek(entries) ?>

<div py:if="item" class="bookmarks">
  <dl py:for="bookmark in entries">
    <dt><a href="{bookmark.url}"
      py:content="bookmark.title" /></dt>

    <dd><p py:content="bookmark.description" />
      <p><a href="{tg.url('/bookmarks/%i/edit' %
        bookmark.id)}">Edit</a></p></dd>
  </dl>
</div>

<div py:if="not item" class="bookmarks">
  No bookmarks found
</div>
```

---

---

## Controller Methoden (II)

Anzeige eines Lesezeichens (I), REST-Dispatcher:

```
...
@expose()
def default(self, *params, **kw):
    if len(params) == 1:
        id = params[0]
        redirect(url('%s/view' % id))
    elif len(params) >= 2:
        id, verb = params[:2]
        action = getattr(self, verb, None)
        if not action or not \
            getattr(action, 'exposed'):
            raise cherrypy.NotFound
        action(item, *params[2:], **kw)
```

---

---

# Controller Methoden (III)

## Anzeige eines Lesezeichens (II):

...

```
@expose(template=  
    'bookmarker.templates.bookmarks.edit')  
def view(self, id, *params, **kw):  
    try:  
        bookmark = Bookmark.get(id)  
    except SQLAlchemyObjectNotFound:  
        flash('Bookmark not found.')        redirect('/')  
    return dict(entry=bookmark)
```

## Templates (II)

Anzeige/Editieren eines Lesezeichen:

```
<form action="update" method="POST">
  <input type="text" name="title" value="{entry.title}" />

  <input type="text" name="url" value="{entry.url}" />

  <textarea name="description">
    {entry.description}
  </textarea>

  <input type="text" name="tags"
    value="{', '.join([tag.name for tag in entry.tags])}"
  />

  <input type="submit" value="Save">
</form>
```

---

---

# Controller Methoden (IV)

Ändern eines Lesezeichens:

```
@expose()  
def update(self, id, *params, **kw):  
    try:  
        bookmark = Bookmark.get(id)  
    except SQLAlchemyObjectNotFound:  
        bookmark = Bookmark(  
            title=kw.get('title'),  
            url=kw.get('url'),  
            description=kw.get('description')  
        )  
    else:  
        bookmark.title = kw.get('title')  
        bookmark.url = kw.get('url')  
        bookmark.description=kw.get('description')  
    # TODO: handle tags specially  
    redirect('/bookmarks/')
```

# Controller Methoden (V)

Löschen eines Lesezeichens:

```
@expose()  
def delete(self, id, *params, **kw):  
    try:  
        Bookmark.delete(id)  
    except SQLAlchemyObjectNotFound:  
        flash('Bookmark not found.')    else:  
        flash('Bookmark deleted.')    redirect('/bookmarks')
```

***Danke fürs Zuhören!***

<http://chrisarndt.de/talks/tg-tutorial/>

(ab 14.12.2006)

---

---