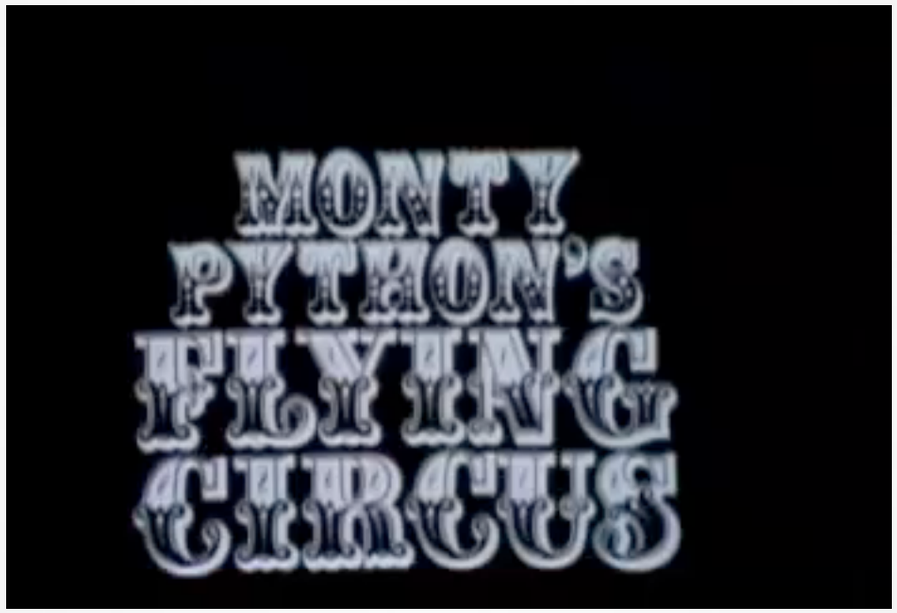
Tutorium Skriptsprachen

2009 - Max Maurer

Monty Python

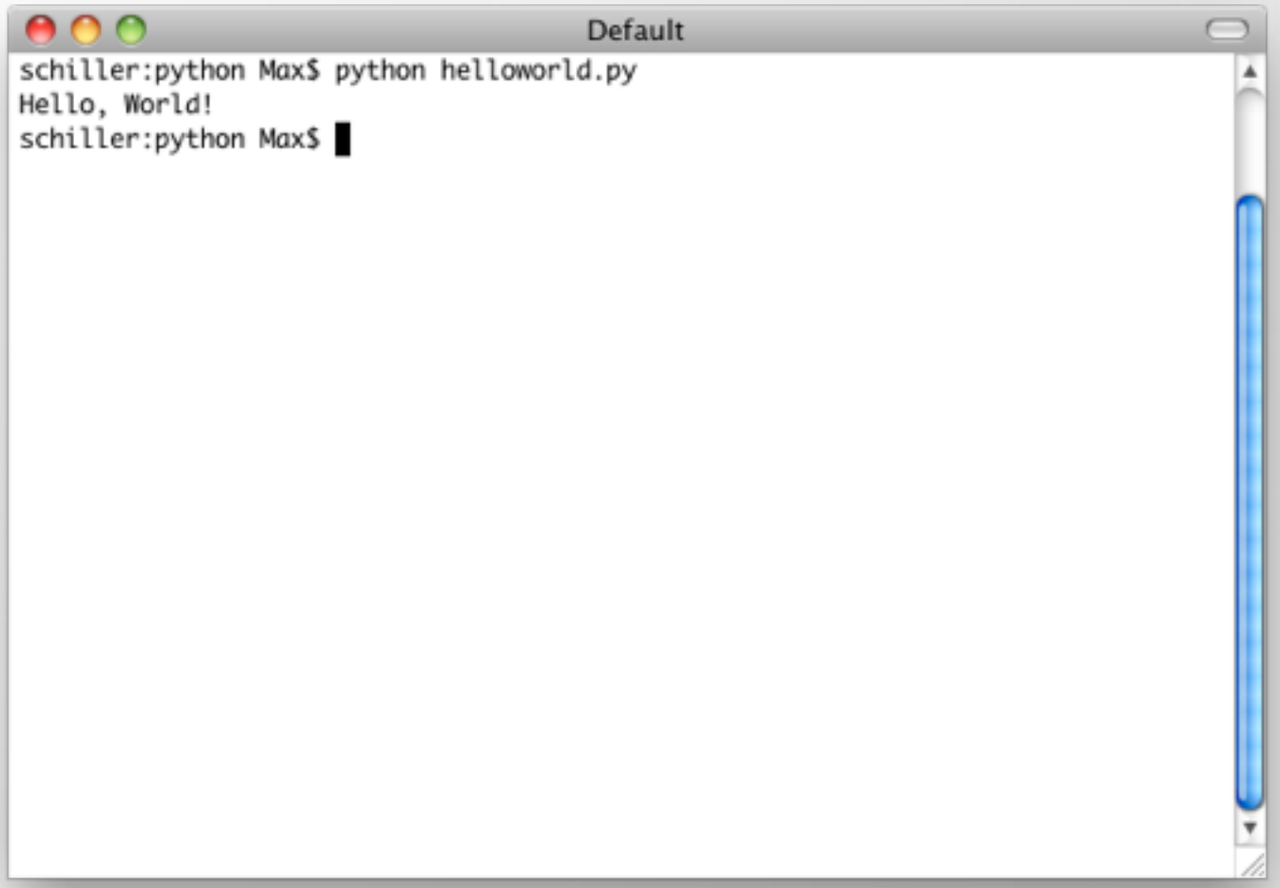


© Youtube.com

Hello World

Hello World!

```
#!/path/to/python
print "Hello, World!"
```



Allgemeines

Fakten

Entstanden:	1991
Erfinder:	Gudio van Rossum
Firma:	Python Foundation
Lizenz:	PSF-Lizenz
Stärken:	Flexibilität
Anwendungsgebiete:	Anwedungsskriptsprache Internetskriptsprache Skripting für C & C++

Guido van Rossum

Guido van Rossum
(c) Wikimedia Commons

“Vor über sechs Jahren, im Dezember 1989, suchte ich nach einem Programmierprojekt, das mich über die Weihnachtswoche beschäftigen würde. Mein Büro würde geschlossen bleiben, aber ich hatte auch zu Hause einen PC und sonst nicht viel zu tun. Ich entschied mich, einen Interpreter für die Scriptsprache zu schreiben, über die ich kürzlich nachdachte: Ein Nachfolger von ABC, der auch Unix- und C-Hacker ansprechen würde. Ich wählte Python als Arbeitstitel für das Projekt, weil ich in einer leicht respektlosen Stimmung (und ein großer Fan des Monty Python’s Flying Circus) war.”

Wikipedia.de

Einsatzgebiete

- Python immer beliebter
- Große Firmen benutzen Python:
  - Google (YouTube ist größtenteils pythonbasiert)
  - Industrial Light & Magic (Spezialeffekte)
  - One Laptop per Child

Characteristika

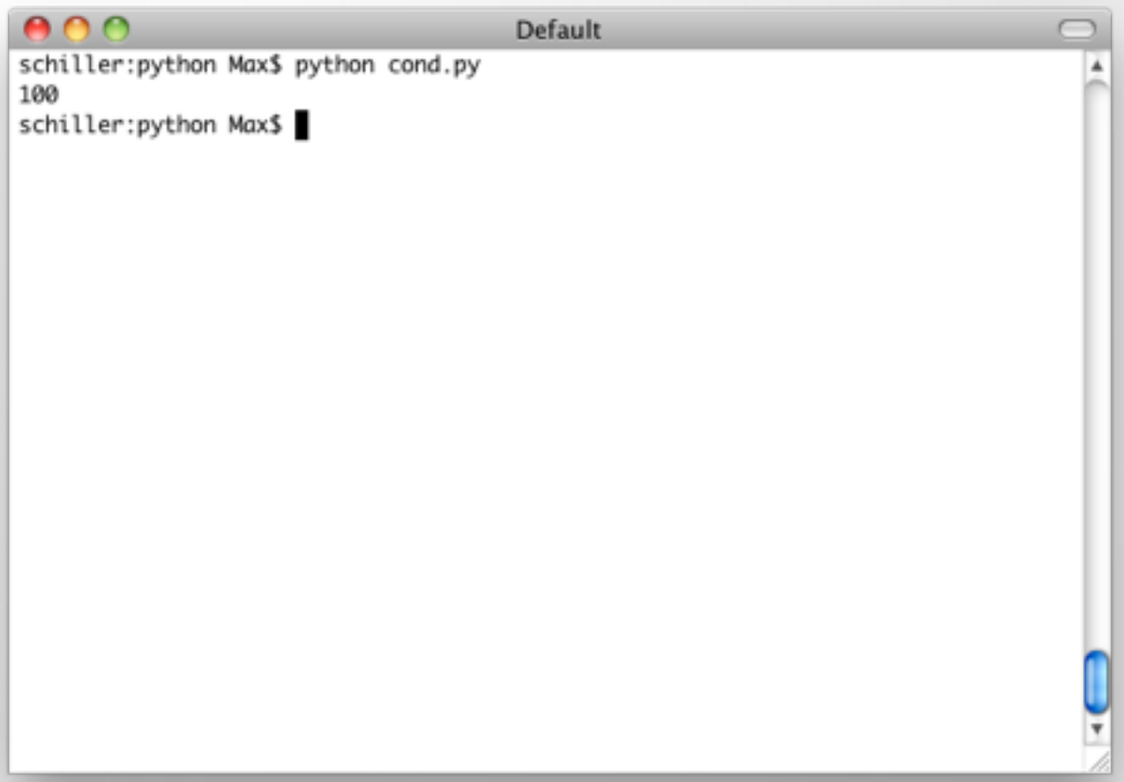
- Interpreterbasierte Sprache
- kann zu intermediate byte-code kompiliert werden (ähnlich zu Java)
- vereint verschiedene Programmierparadigmen (imperativ, objektorientiert, funktional, aspektorientiert)
- dynamische Typbindung
- Garbage Collection
- **keine Klammerung!** Semantik über Einrückung (Tabulatoren)
- ermöglicht GUI-Erzeugung (wxPython)

Code-Beispiele

Code-Einrückung

- Indentation sorgt für Codebedeutung
- es entsteht automatisch ordentlicher Programmcode
- man spart sich viele Zeichen, die oftmals gerade auf deutschen Tastaturen Schwierig zu tippen sind

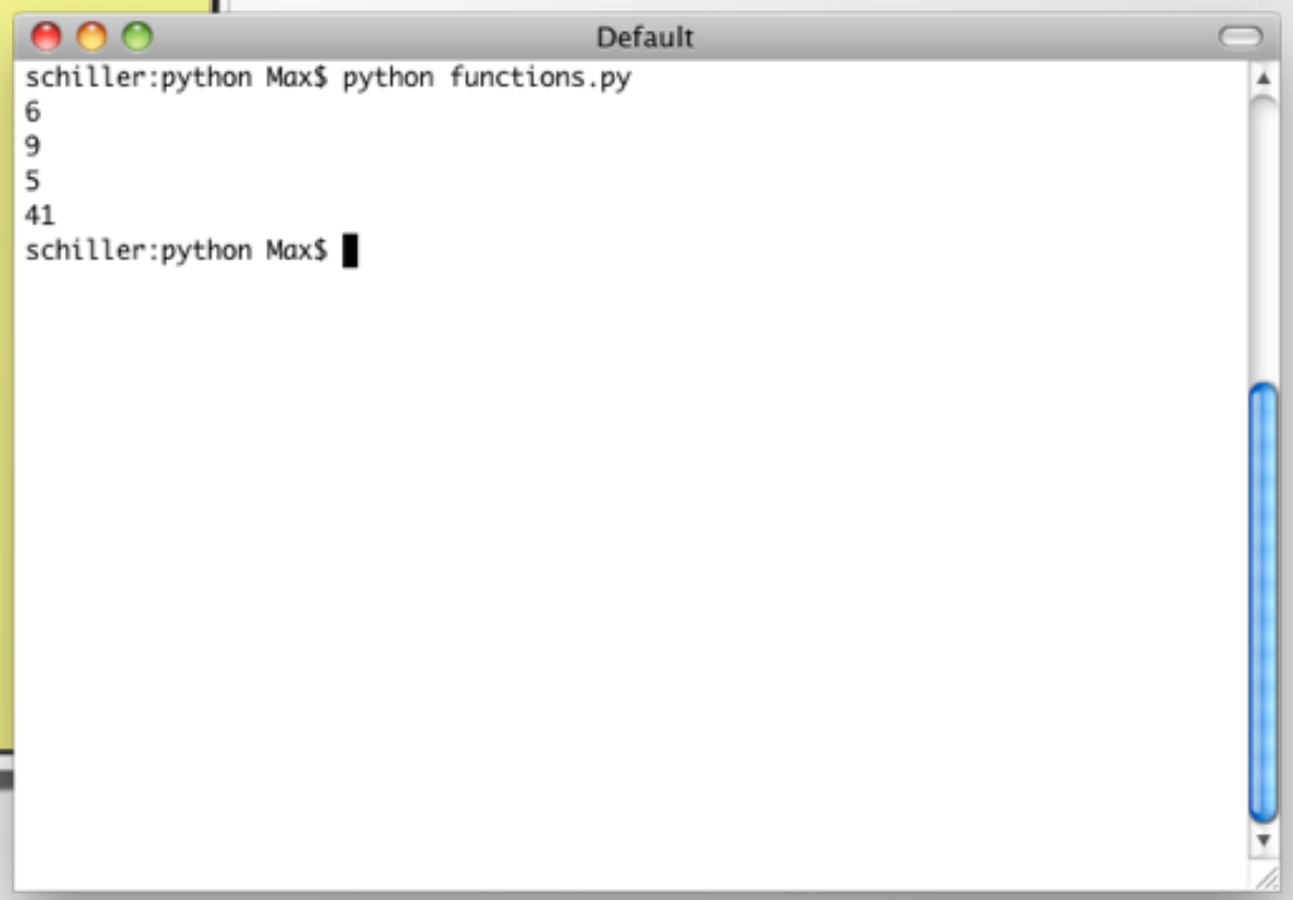
```
a = 1
b = 2

if a > b:
    a = 10
    print a
else:
    a = 100
    print a
```



Funktionen und Parameter

- Funktionsdefinitionen mit Schlüsselwort def
- Parameter erhalten Namen und optional einen Standardwert
- Parameterwerte können auch direkt gesetzt werden.

```
#!/usr/bin/env python
def test(a=1,b=2,c=3):
    def quadrat(x):
        return x*x
    return quadrat(a)+b+c

print test(1)
print test(2,2)
print test(c=2)
print test(test())
```



Globale und lokale Variablen

- lokale Variablen gelten nur in Funktionen in denen Sie deklariert wurden und überschatten bisherige Deklarationen
- globals() und locals() zeigt an welche globalen und lokalen Variablen es gibt

```
#!/usr/bin/env python
def funcLocal():
    x = 0
    print "X-wert in local: %d" % (x,)
    print locals()
    print globals()

def funcGlobal():
    global x
    x=0
    print "X-Wert in global: %d" % (x,)
    print locals()
    print globals()
```

```
Default
schiller:python Max$ python globalsandlocals.py
X-wert in local: 0
{'x': 0}
{'funcLocal': <function funcLocal at 0x3796b0>, '__builtins__': <module '__built
in__' (built-in)>, '__file__': 'globalsandlocals.py', '__package__': None, 'func
Global': <function funcGlobal at 0x379670>, 'x': 2, '__name__': '__main__', '__d
oc__': None}
X-Wert nach funcLocal: 2
X-Wert in global: 0
{}
{'funcLocal': <function funcLocal at 0x3796b0>, '__builtins__': <module '__built
in__' (built-in)>, '__file__': 'globalsandlocals.py', '__package__': None, 'func
Global': <function funcGlobal at 0x379670>, 'x': 0, '__name__': '__main__', '__d
oc__': None}
X-Wert nach funcGlobal: 0
schiller:python Max$
```

Error Handling

- Fehlerhandlung mit try und except
- Fehler werden mit raise geworfen und müssen von Exception erben

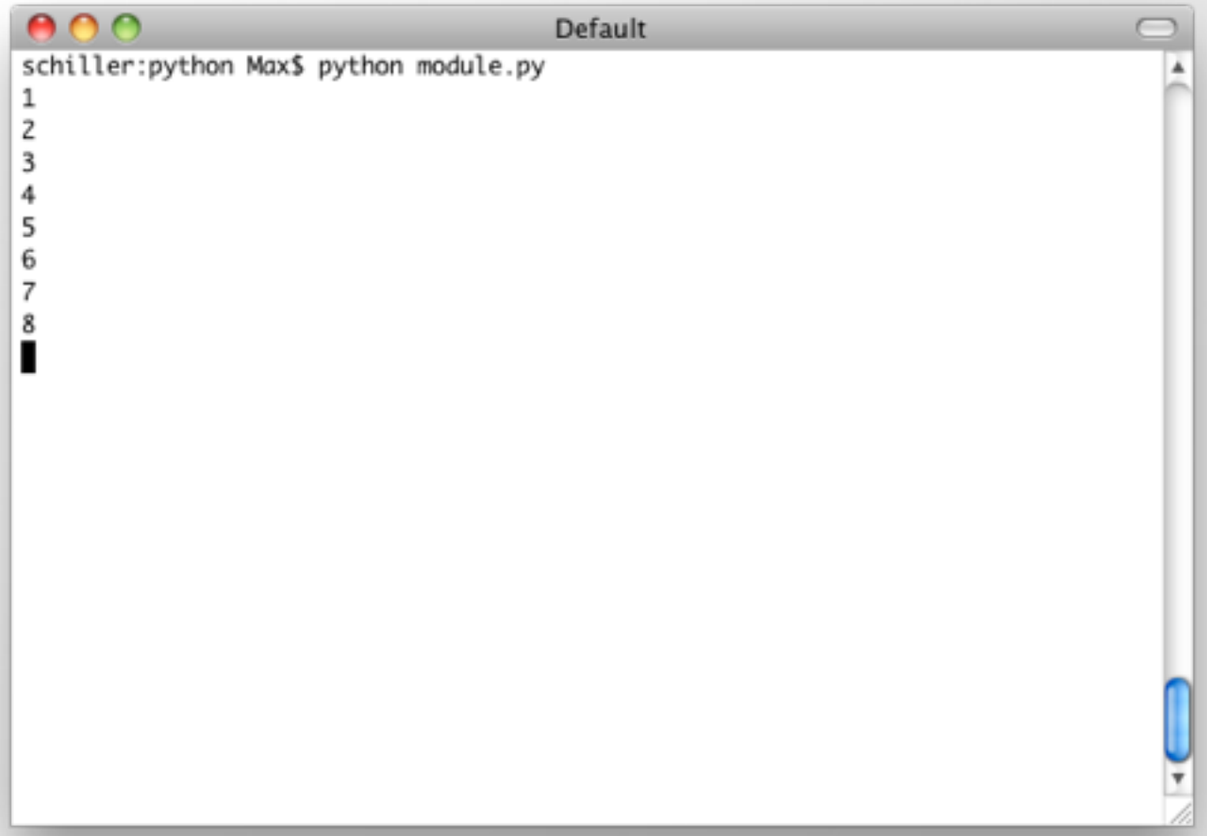
```
#!/usr/bin/env python
class MyError(Exception):
    def __init__(self, value):
        self.value = value
    def __str__(self):
        return repr(self.value)

def errorFunc():
    print "Watch out an error will occur"
    raise MyError("Whoops!")

try:
    errorFunc()
except MyError as e:
    print 'My exception occurred, value:', e.value
finally:
    print "Cleanup actions go here"
```

```
Default
schiller:python Max$ python error.py
Watch out an error will occur
My exception occurred, value: Whoops!
Cleanup actions go here
schiller:python Max$
```

Module

- viele Module zur Erweiterung von Python
- grundlegende Funktionen (Threads)
- auch komplexe Frameworks (wxPython für GUIs, pygame für die Spieleentwicklung)

```
#!/usr/bin/python
import thread
import time
def test():
        i=0
        while True:
            i=i+1
            print i
            time.sleep(1)
thread.start_new_thread(test,())
while True:
    pass
```

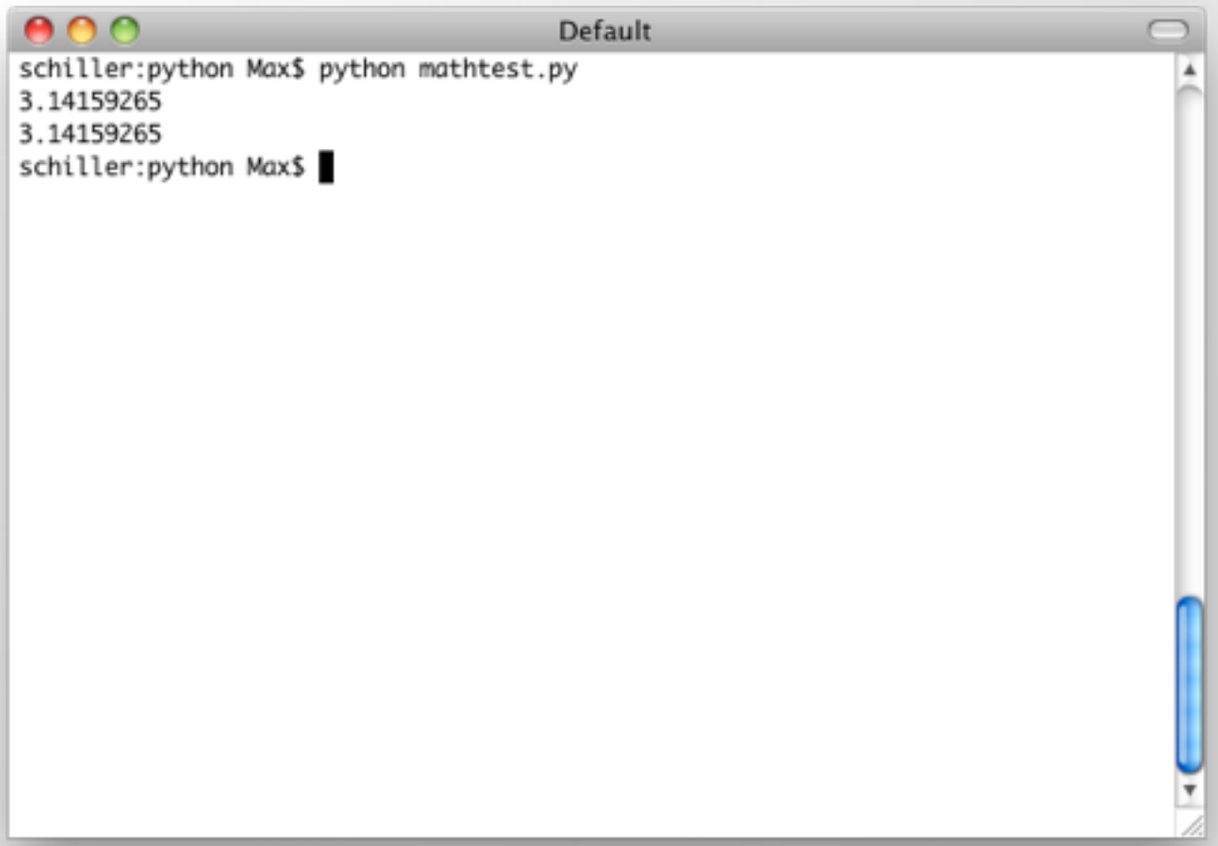


Eigene Module

- Module sind normale python files
- Import auf zwei verschiedene Arten möglich

```
# math.py
#!/usr/bin/env python
def PI():
    return 3.14159265
```

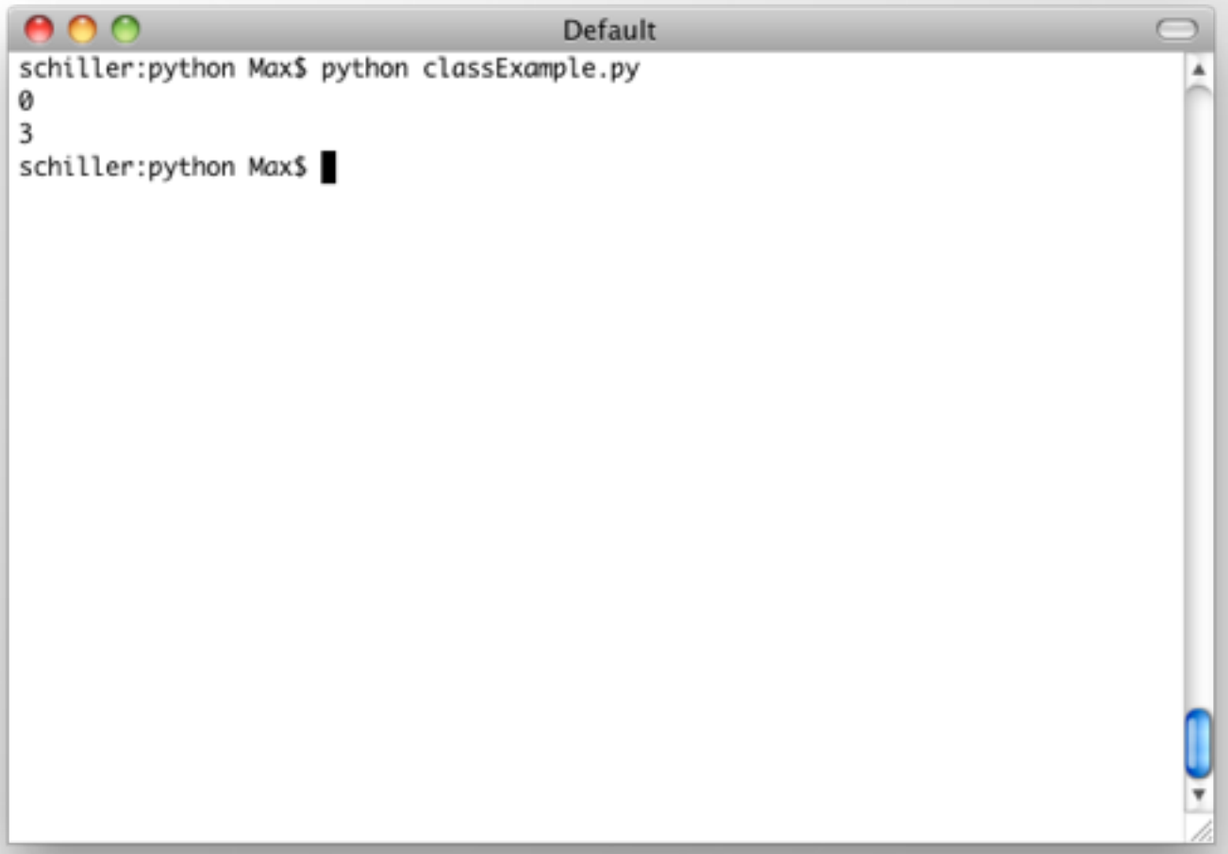
```
# mathtest.py
#!/usr/bin/env python
import math
print math.PI()

from math import PI
print PI()
```



Klassen

- in Python sind alle Daten Instanzen von Objekten (auch primitive Datentypen)
- eigene Klassen lassen ähnlich zu Funktionen definieren, lediglich wird das Schlüsselwort class benutzt

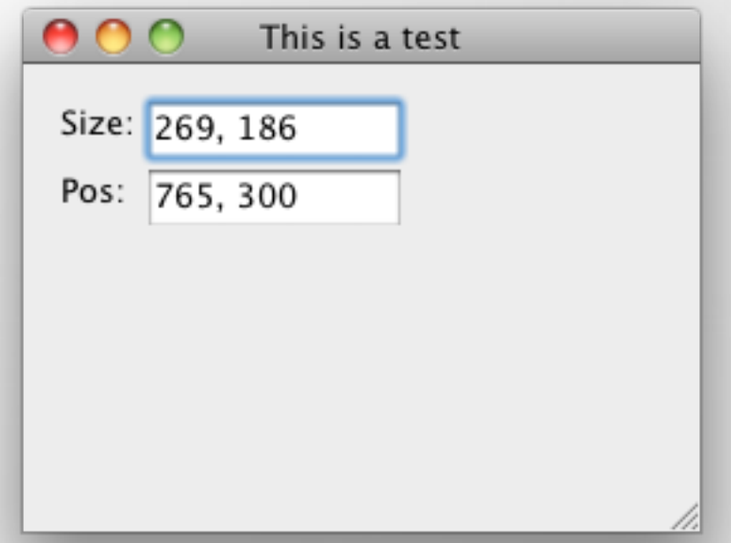
```
#!/usr/bin/env python
class Counter:
    def __init__(self):
        self.k=0
    def count(self):
        self.k+=1
    def reset(self):
        self.k=0
    def getValue(self):
        return self.k

c = Counter()
print c.getValue()
c.count()
c.count()
c.count()
print c.getValue()
```



GUI mit wxPython

- mit wxPython kann man grafische Benutzeroberflächen erstellen

```
#!/usr/bin/env python
import wx

class MyFrame(wx.Frame):
    def __init__(self, parent, id, title):
        wx.Frame.__init__(self, parent, id, title)
        self.Bind(wx.EVT_SIZE, self.OnSize)
        self.Bind(wx.EVT_MOVE, self.OnMove)
        panel = wx.Panel(self, -1)
        label1 = wx.StaticText(panel, -1, "Size:")
        label2 = wx.StaticText(panel, -1, "Pos:")
        self.sizeCtrl = wx.TextCtrl(panel, -1, "", style=wx.TE_READONLY)
        self.posCtrl = wx.TextCtrl(panel, -1, "", style=wx.TE_READONLY)
        self.panel = panel
        sizer = wx.FlexGridSizer(2, 2, 5, 5)
        sizer.Add(label1)
        sizer.Add(self.sizeCtrl)
        sizer.Add(label2)
        sizer.Add(self.posCtrl)

        border = wx.BoxSizer()
        border.Add(sizer, 0, wx.ALL, 15)
        panel.SetSizerAndFit(border)
        self.Fit()

    def OnSize(self, event):
        size = event.GetSize()
        self.sizeCtrl.SetValue("%s, %s" % (size.width, size.height))
```



Spiele mit pygame

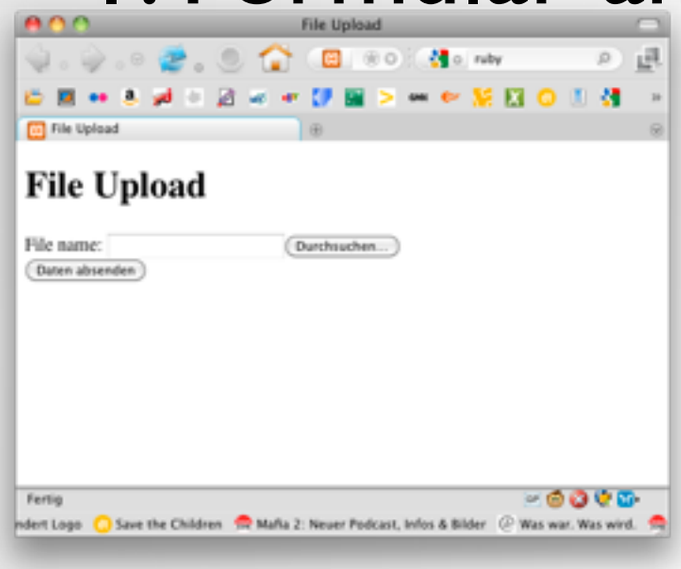
```
#!/usr/bin/env python
if __name__ == '__main__':
    pass

import os, pygame
from pygame.locals import *

class GameObject:
    def __init__(self, image, height, speed):
        self.speed = speed
        self.image = image
        self.pos = image.get_rect().move(0, height)
    def move(self):
        self.pos = self.pos.move(self.speed, 0)
        if self.pos.right > 600:
            self.pos.left = 0

pygame.init()
screen = pygame.display.set_mode((640, 480))
player = pygame.image.load('data/player1.gif').convert()
background = pygame.image.load('data/liquid.bmp').convert()
objects = []
for x in range(10):
        o = GameObject(player, x*40, x)
        objects.append(o)
while 1:
    for event in pygame.event.get():
        if event.type in (QUIT, KEYDOWN):
            quit()
    screen.blit(background, (0,0))
    for o in objects:
            o.move()
            screen.blit(o.image, o.pos)
    pygame.display.update()
```

- pygame hat verschiedene Module zur schnellen und einfachen Erstellung von Spielen

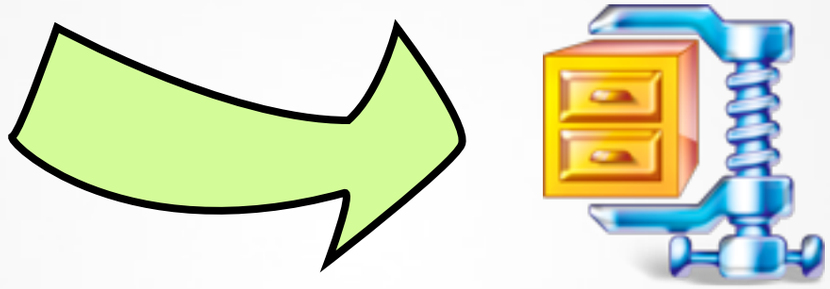
Die Aufgabe

Galerie Baukasten

1. Formular anzeigen

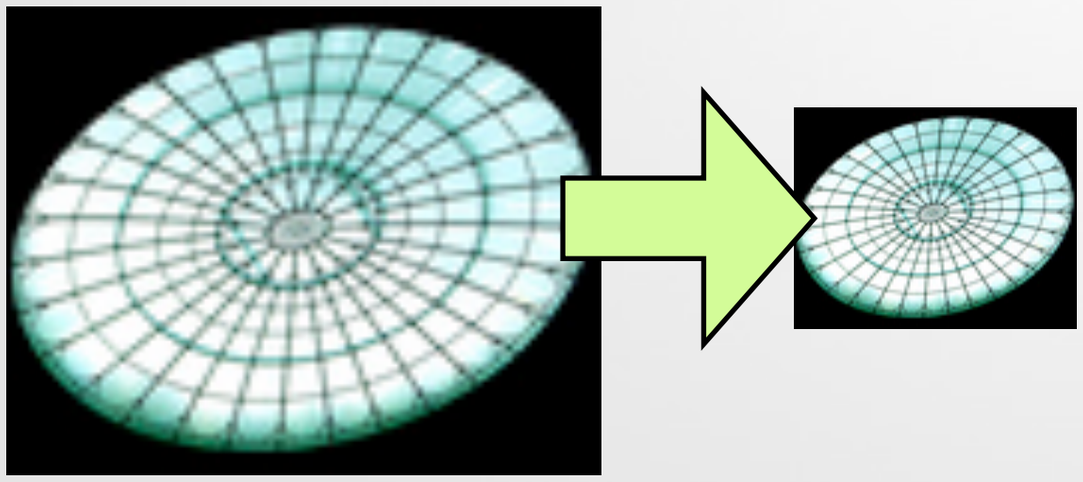


2. Zip-Datei hochladen



3. Entpacken

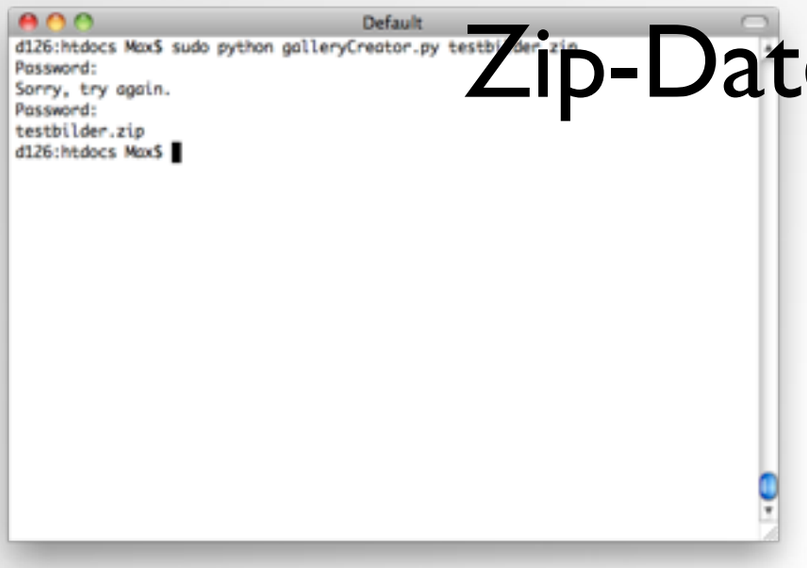
4. Thumbnails rendern



5. Gallery Seite anzeigen

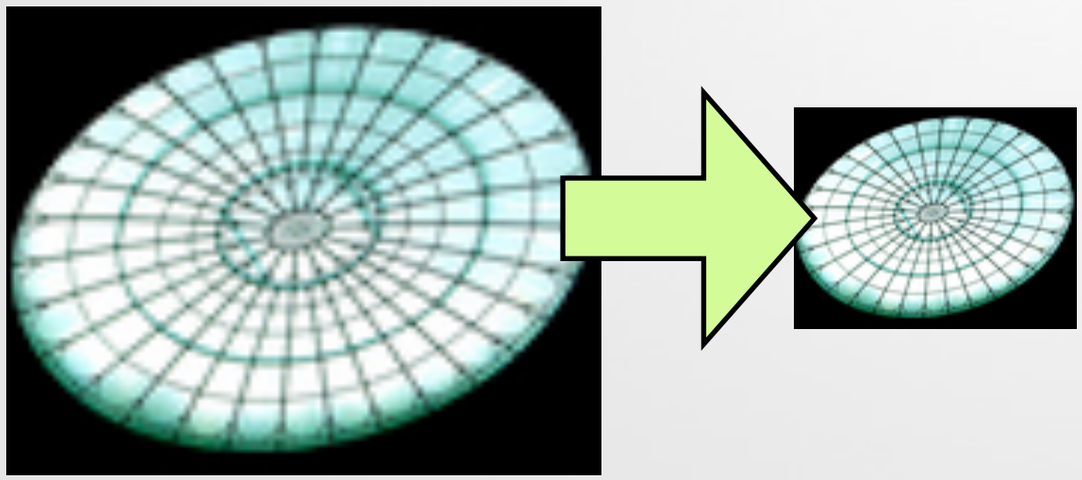
Kommandozeilenversion

1. Kommandozeilenaufruf mit Zip-Datei



2. Entpacken

3. Thumbnails rendern



4. Gallerie Seite erzeugen

Grundskript und Kommandozeile

Grundskript

```
#!/usr/bin/env python
import sys

HEADER = "content-type: text/html\n\n"

def error(text):
    print HEADER
    print "<html><body><h1>"+text+"</h1></body></html>"
    quit()

if len(sys.argv)>=2:
    # wir haben ein command line argument
else:
    print HEADER
    error("HTML-Seite")
```

Das ‚sys'-Modul

- Systemspezifische Parameter und Funktionen
- Wichtige Parameter und Funktionen
  - sys.argv: Kommandozeilenargumente (Element 0 ist der Name des Skripts)
  - sys.getDefaultEncoding(): Standard-Kodierung für Strings
  - sys.maxint: Maximaler Integerwert
  - sys.path: Modulpfade
  - sys.platform: Welches Betriebssystem läuft (z.B. win32, cygwin, darwin, os2, linux2)
  - sys.version: Python Version

Grundskript

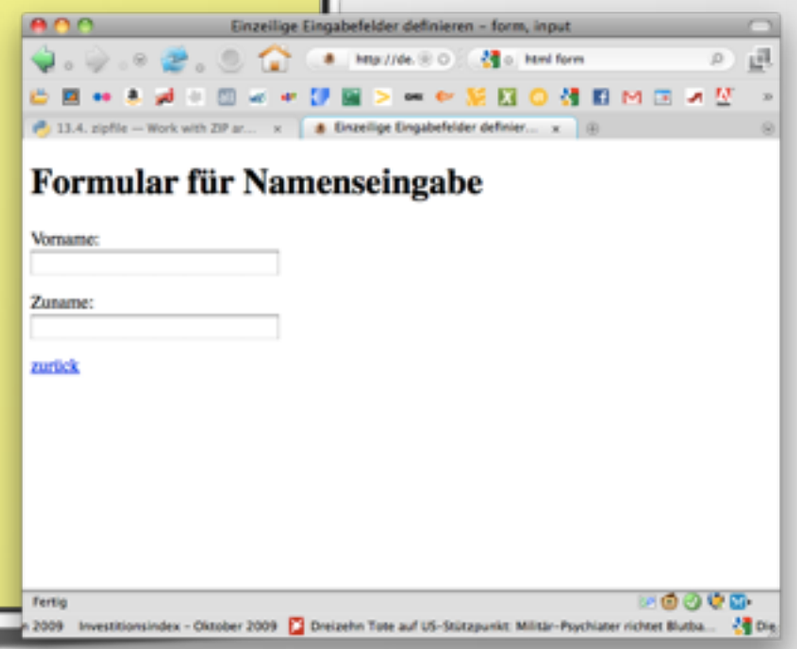
```
#!/usr/bin/env python
import sys

HEADER = "content-type: text/html\n\n"

def error(text):
    print HEADER
    print "<html><body><h1>"+text+"</h1></body></html>"
    quit()

if len(sys.argv)>=2:
    # wir haben ein command line argument
else:
    print HEADER
    error("HTML-Seite")
```

HTML-Formulare

HTML-Formulare

- Darstellung verschiedener Eingabemöglichkeiten
  - Eingabefelder, Dropdown, Checkbox, Radio Button, Datei Upload, TextArea, Button

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
       "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Einzeilige Eingabefelder definieren</title>
</head>
<body>

<h1>Formular f&uuml;r Namenseingabe</h1>

<form action="input_text.htm" method="post">
  <p>Vorname:<br><input name="vorname" type="text" size="30"
maxlength="30"></p>
  <p>Zuname:<br><input name="zuname" type="text" size="30"
maxlength="40"></p>
</form>
```



Formulardaten allgemein

- Zwei Methoden: GET oder POST
- GET
  - Übergabe über die URL: „test.py?action=hallo&var1=wert1“
  - Direkt sichtbar und manipulierbar. Variablen bleiben bei Copy&Paste in E-Mails z.B. erhalten (z.B. Google Maps)
- POST
  - Nicht im Browser sichtbar auch nicht im Browser Cache gespeichert, werden im Anfrage-Header von HTTP übergeben

Formulardaten in Python

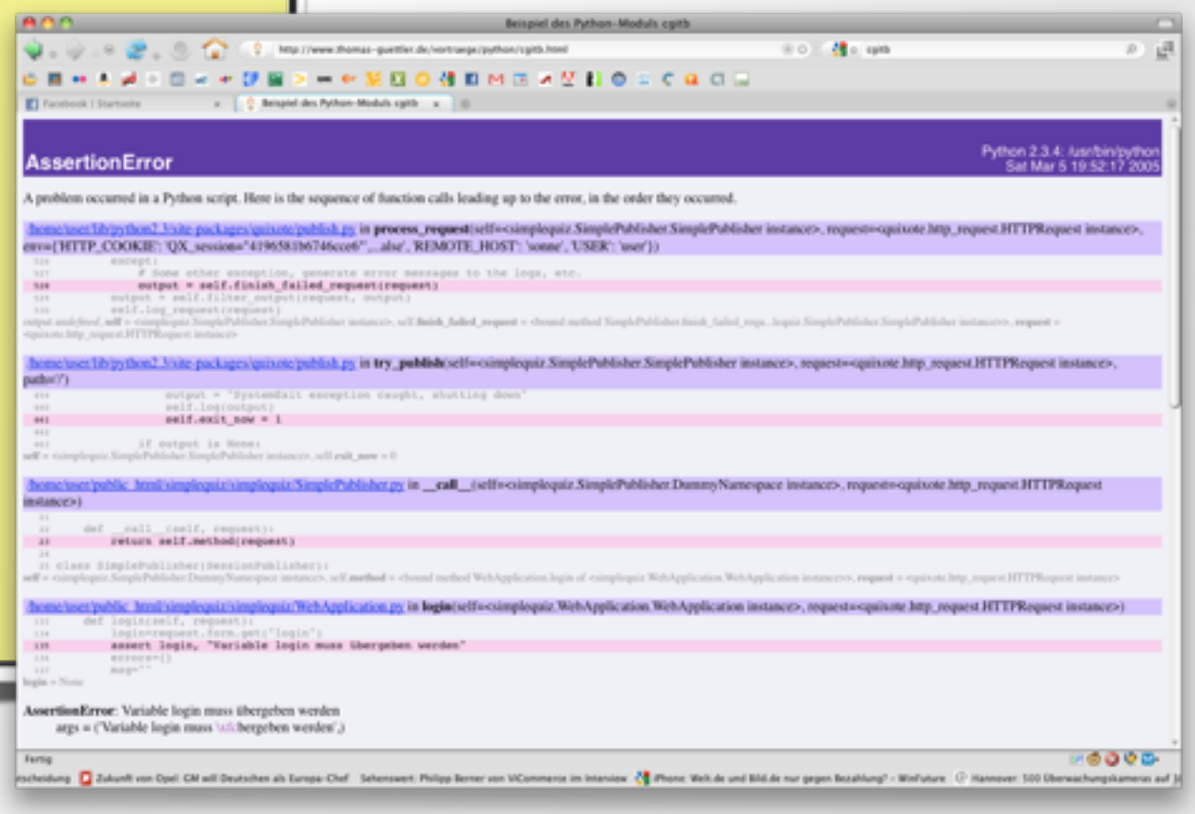
- Keine direkte Unterscheidung zwischen GET und POST
- Modul zum Benutzen von Daten: cgi

```
form = cgi.FieldStorage()
if "name" not in form or "addr" not in form:
    print "<H1>Error</H1>"
    print "Please fill in the name and addr fields."
    return
print "<p>name:", form["name"].value
print "<p>addr:", form["addr"].value
```

Modul „cgitb“

- Ausgabe von Fehlermeldungen und Stacktrace im Browser
- Normalerweise Fehler ohne „content-type“ und HTML Code
  - Dadurch Debugging einfacher und im Browser möglich

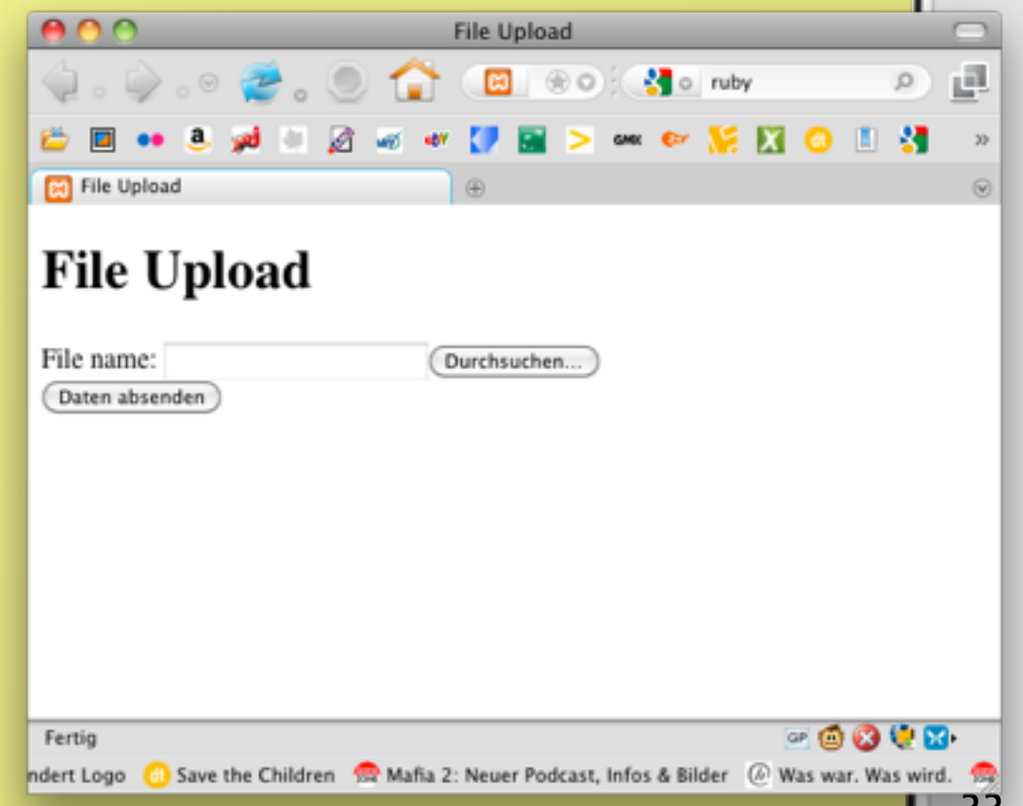
```
import cgi
cgitb.enable()
```



Datei-Upload

Formular ausgeben

```
HTML_TEMPLATE = """<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
       "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<html><head><title>File Upload</title>
</head><body><h1>File Upload</h1>
<form action="?action=upload" method="POST" enctype="multipart/form-
data">
File name: <input name="file" type="file"/><br/>
<input name="submit" type="submit"/>
</form>
</body>
</html>"""

HEADER = "content-type: text/html\n\n"

print HEADER
print HTML_TEMPLATE
```



Upload auslesen

- Gelesene Datei temporär hinterlegt
- Muss nun gespeichert werden

```
import cgi
import os
form = cgi.FieldStorage()

def processFile (form_field):
    """This saves a file uploaded by an HTML form.
    """
    if not form.has_key(form_field): return
    fileitem = form[form_field]
    if not fileitem.file: return
    filepath = os.path.join(UPLOAD_DIR, fileitem.filename)
    fout = file (filepath, 'wb')
    while 1:
        chunk = fileitem.file.read(100000)
        if not chunk: break
        fout.write (chunk)
    fout.close()
    return filepath
```

Das ‚os'-Modul

- Betriebssystemabhängige Funktionen
- Wichtige Parameter und Funktionen
  - os.environ Umgebungsvariablen (z.B. HOME)
  - os.getlogin() gibt aktuellen Benutzernamen zurück
  - os.chdir(path) ändert das Arbeitsverzeichnis des Skripts
  - os.mkdir() und os.rmdir(): Verzeichnis anlegen oder löschen
  - os.rename(src, dst): Datei oder Verzeichnis umbenennen
  - os.unlink(file): Datei löschen
- Submodul ‚os.path' für Pfadoperationen
  - os.path.join(p1, p2): Verkettet Pfade betriebssystemspezifisch

ZIP-Datei entpacken

Zugriff auf ZIP-Dateien

```
#!/usr/bin/env python
import zipfile

fh = open('test.zip', 'rb')
z = zipfile.ZipFile(fh)
for name in z.namelist()
  print name
```

Das ‚zipfile'-Modul

- Umgang mit ZIP-Dateien
- Wichtige Parameter und Funktionen
  - zipfile.ZipFile(filehandle) erzeugt ein neues Objekt für diese Zip-Datei
  - zipfile.namelist() gibt einen Liste mit allen Dateien im Archiv zurück
  - zipfile.open(name) benutze ein Objekt als wäre es eine Datei im Dateisystem
  - zipfile.extract(member[, path]) oder zipfile.extractAll([path]): entpackt Dateien (Achtung: unsicher bei relativen Pfaden)
  - zipfile.read(name): Element als bytes einlesen
  - zipfile.write(name) oder zupfile.writestr(name,bytes) schreibt neue Datei in die ZipFile

Zugriff auf ZIP-Dateien

- Zip-Datei ‚flat' entpacken

```
#!/usr/bin/env python
import zipfile

def unzip(file, path):
    fh = open(file, 'rb')
    z = zipfile.ZipFile(fh)
    for name in z.namelist():
        m = re.search('/([^/]*)$', name)
        if m == None: continue
        if not m.group(1): continue
        if (m.group(1).endswith('/')): continue
        outfile = open(os.path.join(path,m.group(1)), 'wb')
        outfile.write(z.read(name))
        outfile.close()
    fh.close()
    #os.remove(file) # remove zip file
```

Regular Expressions

- Zeichenketten prüfen und ändern
  - „/test/“ testet ob die Zeichenkette „test“ im Text vorkommt
  - „/[a-z]/“ testet ob ein kleiner Buchstabe im Text vorkommt
  - „/^[a-z]/“ oder „/[a-z]$“ kleiner Buchstabe am Anfang und am Ende
  - „/^[a-z]*$/“ Quantifikatoren (*,+,?): Test besteht aus keinem Zeichen oder nur aus Kleinbuchstaben
  - „/Anfang(.*)Ende/“: Group matching sucht alle Zeichen zwischen Anfang und Ende und speichert diese
  - „/^...$/“: Was ist das?
  - „/([^/]*)$“: und das?

Zugriff auf ZIP-Dateien

- Zip-Datei ‚flat' entpacken

```
#!/usr/bin/env python
import zipfile

def unzip(file, path):
    fh = open(file, 'rb')
    z = zipfile.ZipFile(fh)
    for name in z.namelist():
        m = re.search('/([^/]*)$', name)
        if m == None: continue
        if not m.group(1): continue
        if (m.group(1).endswith('/')): continue
        outfile = open(os.path.join(path,m.group(1)), 'wb')
        outfile.write(z.read(name))
        outfile.close()
    fh.close()
    #os.remove(file) # remove zip file
```

Thumbnails schreiben

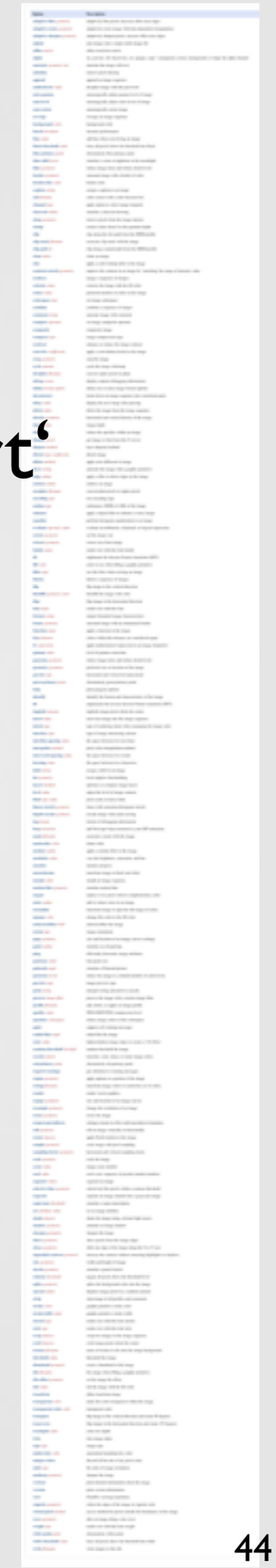
Thumbnails schreiben

- Intern per Modul
- Extern per Kommandozeilenaufruf (‚convert‘)

```
import os
import subprocess
import re

def createThumbnails(path):
    for f in getFilesWithSuffix(path, '.jpg','_T.jpg'):
        inputfile = os.path.join(path,f)
        imageName = re.search('(.*)\.jpg$',f).group(1)
        outputfile = os.path.join(path,imageName+'_T.jpg')
        cmd = "/usr/local/bin/convert -resize 100x100 "+inputfile+" "+outputfile
        proc = subprocess.Popen(cmd, shell=True,stdout=subprocess.PIPE, stderr=subprocess.PIPE);
        returnValues = proc.communicate()
        if (proc.returncode==1):
            os.remove(inputfile)
            continue

def getFilesWithSuffix(directory, suffix, notsuffix):
  fileList=os.listdir(directory)
  return [f for f in fileList if f.endswith(suffix) and not f.endswith(notsuffix)]
```

ImageMagick

- Kommandozeilen Bildverarbeitung mit umfangreichem Bildumwandlungstool ‚convert‘
- Dateiformate (> 100!)
  - Beispiele: AVI, BMP, JPEG, MPEG, PCX, PNG, PSD, SVG, TTF, WMF
- Optionen (s. rechts)
  - nur Einige: fill, rotate, resize, white-point
- Mehr Infos: www.imagemagick.org

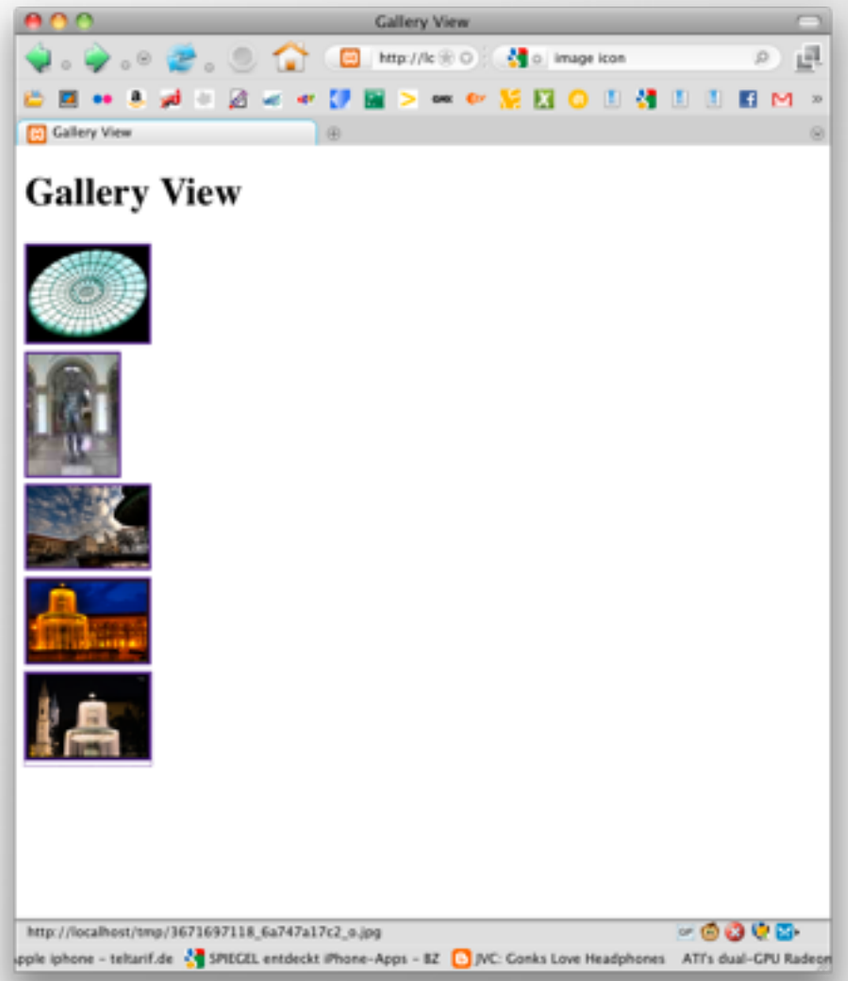


Gallery Seite erzeugen

Thumbnails schreiben

- Seite mit Template bauen und alle gefunden Bilder einfügen

```
UPLOAD_TEMPLATE = """<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
       "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<html><head><title>Gallery View</title>
</head><body><h1>Gallery View</h1>
"""

def generateWebsite(path):
     html = UPLOAD_TEMPLATE
     for f in getFilesWithSuffix(path, '.jpg','_T.jpg'):
          inputfile = os.path.join(path,f)
          imageName = re.search('(.*)\.jpg$',f).group(1)
          outputfile = os.path.join(path,imageName+'_T.jpg')
          html += '<div class="image"><a href="'
          html += inputfile
          html += '"><img src="'
          html += outputfile
          html += '"/></a></div>'
     html += """</body></html>"""
     return html

print HEADER
print generateWebsite(UPLOAD_DIR)
```



Zeit messen

Zeitmessung

- Nicht zwangsläufig notwendig für die Aufgabenstellung aber interessant zum Vergleichen
- Möglich mit dem Modul ‚time‘

```
import time
startzeit = time.time()
# Aufwaendige Berechnung
zeitpunkt1 = time.time()-startzeit
# Aufwaendige Berechnung
zeitpunkt2 = time.time()-startzeit
print „Gebrauchte Zeit (1): %f Sekunden“ % zeitpunkt1
print „Gebrauchte Zeit (2): %f Sekunden“ % zeitpunkt2
```

Kompletter Code

Gallery Uploader Code

```
#!/usr/bin/env python
import time
start = time.time()
import cgi
import cgitb
import zipfile
import re
import string
import subprocess
import os, sys
cgitb.enable()
form = cgi.FieldStorage()

UPLOAD_DIR = "./tmp"

HTML_TEMPLATE = """<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
       "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<html><head><title>File Upload</title>
</head><body><h1>File Upload</h1>
<form action="?action=upload" method="POST" enctype="multipart/form-data">
File name: <input name="file" type="file"/><br/>
<input name="submit" type="submit"/>
</form>
</body>
</html>"""

UPLOAD_TEMPLATE = """<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
       "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<html><head><title>Gallery View</title>
</head><body><h1>Gallery View</h1>
"""

HEADER = "content-type: text/html\n\n"
```

Gallery Uploader Code

```
def error(text):
     print HEADER
     print "<html><body><h1>"+text+"</h1></body></html>"
     quit()

def getFilesWithSuffix(directory, suffix, notsuffix):
  fileList=os.listdir(directory)
  return [f for f in fileList if f.endswith(suffix) and not f.endswith(notsuffix)]

def processFile (form_field):
    """This saves a file uploaded by an HTML form.
    """
    if not form.has_key(form_field): return
    fileitem = form[form_field]
    if not fileitem.file: return
    filepath = os.path.join(UPLOAD_DIR, fileitem.filename)
    fout = file (filepath, 'wb')
    while 1:
        chunk = fileitem.file.read(100000)
        if not chunk: break
        fout.write (chunk)
    fout.close()
    return filepath

def unzip(file, path):
     fh = open(file, 'rb')
     z = zipfile.ZipFile(fh)
     for name in z.namelist():
               m = re.search('/([^/]*)$', name)
               if m == None: continue
               if not m.group(1): continue
               if (m.group(1).endswith('/')): continue
               outfile = open(os.path.join(path,m.group(1)), 'wb')
               outfile.write(z.read(name))
               outfile.close()
     fh.close()
     #os.remove(file) # remove zip file
```

Gallery Uploader Code

```
def createThumbnails(path):
     for f in getFilesWithSuffix(path, '.jpg','_T.jpg'):
          inputfile = os.path.join(path,f)
          imageName = re.search('(.*)\.jpg$',f).group(1)
          outputfile = os.path.join(path,imageName+'_T.jpg')
          cmd = "/usr/local/bin/convert -resize 100x100 "+inputfile+" "+outputfile
          proc = subprocess.Popen(cmd, shell=True,stdout=subprocess.PIPE, stderr=subprocess.PIPE);
          returnValues = proc.communicate()
          if (proc.returncode==1):
               os.remove(inputfile)
               continue

def generateWebsite(path):
     html = UPLOAD_TEMPLATE
     for f in getFilesWithSuffix(path, '.jpg','_T.jpg'):
          inputfile = os.path.join(path,f)
          imageName = re.search('(.*)\.jpg$',f).group(1)
          outputfile = os.path.join(path,imageName+'_T.jpg')
          html += '<div class="image"><a href="'
          html += inputfile
          html += '"><img src="'
          html += outputfile
          html += '"/></a></div>'
     html += "Store file: %f seconds<br/>" % timeUpload;
     html += "Unzip file: %f seconds<br/>" % (timeUnzip-timeUpload);
     html += "Create thumbnails: %f seconds<br/>" % (timeThumbnails-timeUnzip);
     html += "Overall: %f seconds<br/>" % timeThumbnails;
     html += """</body></html>"""
     return html
```

Gallery Uploader Code

```
if len(sys.argv)>=2:
     # wir haben ein command line argument!
     print sys.argv[1]
     filepath = sys.argv[1]
     unzip(filepath, UPLOAD_DIR)
     createThumbnails(UPLOAD_DIR)
     html = generateWebsite(UPLOAD_DIR)
     outfile = open('gallery.html', 'wb')
     outfile.write(html)
     outfile.close()
     quit()
else:
     if form.has_key("action") and form["action"].value == "upload":
          filepath = processFile ("file")
          timeUpload = time.time()-start;
          unzip(filepath, UPLOAD_DIR)
          timeUnzip = time.time()-start;
          createThumbnails(UPLOAD_DIR)
          timeThumbnails = time.time()-start;
          print HEADER
          print generateWebsite(UPLOAD_DIR)
          quit()

     print HEADER
     print HTML_TEMPLATE
```

Nächste Woche: Perl

Literatur

- Peter Kaiser, Johannes Ernesti: Python - Das umfassende Handbuch (Openbook)
- Gudio van Rossum: An introduction to python
- Alex Martelli, Anna Martelli Ravenscroft, David Ascher: Python Cookbook

Bildnachweis

- http://www.creativeuncut.com/gallery-07/art/mlpit-mario-baby-hammer.jpg
- http://www.dotolearn.com/picturecards/images/imageschedule/proud_1.gif
- http://www.hakstpoelten.ac.at/buchoase/Buch.gif
- http://school.discoveryeducation.com/clipart/images/arteasel4c.gif
- http://www.gg-schnitt.at/wp-content/uploads/2009/01/lego_brick.png
- http://www.helliot.com/cms/images/stories/logo/logo_school.gif
- http://www.clker.com/clipart-window-icon.html
- http://globaleuropeans.com/uploads/images/GE_global%20projects%202.jpg
- http://www.sbac.edu/~tpl/clipart/Animals%20and%20Insects/bug%20cartoon%2002.jpg
- http://jasoncirillo.files.wordpress.com/2009/03/pong.jpg