Some Experiences With Python For Android (Py4A)

Nik Klever
University of Applied Sciences Augsburg
Content

- Introduction and General Aspects
  - Motivation and Background / Changing to Android / Advantages and Usability
- SL4A Basics
  - Architecture / Features / API
- SL4A API
  - Sensors / Location / Camera
- Examples
  - Web2py / GeoTracking
- What's missing?
- The Future
Motivation and Background

• Teaching since 2001 in our degree program "Interactive Media"
  • "Interactive Media" is a program based on 50% computer science and 50% art design
  • the students are starting to learn programming by a one year java course
  • I try to motivate them using Python in web programming using web2py since 2008
• Research projects http://www.hs-augsburg.de/~john/mobile-experience/
  • Java projects on Symbian mobile devices since 2003
  • some projects with Python on S60 in 2007 and 2008
  • but these projects are mostly a single thesis or a single project
  • and none of them could be used for teaching in beginners classes
• Teaching needs
  • ease of use and simplicity
  • fast success stories
  • Open Source and not restricted frameworks
Changing to Android

- Android as an Operating System for mobiles appeared first in 2008
- increasing mobile smartphone market since 2010 – already all students have now a device, no additional costs for the faculty
- based on Linux Android opens this devices to desktop quality

Source: http://en.wikipedia.org/wiki/Mobile_operating_system
Advantages and Usability

• **Device Driven Development vs. Emulator Driven Development**
  • only a Text Editor is needed
  • no complex Emulator Installation is necessary

• **Why?**
  • currently more for experienced programmers
  • not for students based on windows-clickable knowledge

• **How?**
Beamer Access

- **MicroUSB – HDMI Adapter**
  - using this adapter (with Android > 4.0) lets you easily output the user interface via modern beamers

- **HDMI – VGA Adapter**
  - for middle-aged beamers like in this room
  - be careful and **test it**!
  - there are so many Adapters available, and they are not compatible – even for each smartphone there is a new one!
WIFI Connection vs. USB Connection

- using the **SwiFTP** Server on the Android and **FileZilla** on the Desktop lets you access the whole root directory and not only the sdcard
- additionally, the sdcard device is not unmounted during transfers as it is using the USB connection
- editing on the PC an testing on the device can be done on-the-fly

**SwiFTP APK**
- [http://github.com/ppareit/swiftp](http://github.com/ppareit/swiftp)

**FileZilla** (or any other FTP-client)
- [http://filezilla-project.org/](http://filezilla-project.org/)
• SL4A is a wrapper for the Android API via so-called Android Facades using RPC calls
  - code.google.com/p/android-scripting

• Unofficial Release:
  sl4a_r5x (r5x21 – 4.6.2012)
  - code.google.com/p/android-scripting/wiki/Unofficial
• SL4A RPC Server

![SL4A architecture overview](image)

**Figure 5-21. SL4A architecture overview**

Source: Jordan/Greyling, Practical Android Projects

• SL4A uses JavaScript Object Notation (JSON) for the communication between the SL4A RPC Server and its clients.
SL4A Features

- SL4A makes it possible to **quickly prototype applications for Android on the device itself** using high-level scripting languages.

- SL4A provides Android facades which make the **Android APIs available via JSON RPC calls**.

- As for the Android facades, the **API is primarily focused on making it easier to write scripts** than on the performance of those scripts.

- Python is actually the cross compiled C binary version running natively.

Dividing the SL4A API into groups of facades:

- **Subdevices** (BatteryManager, Bluetooth, **Location**, SensorManager, WiFi, SignalStrength)
- **Media** (MediaPlayer, MediaRecorder, SpeedRecognition, TextToSpeech, Phone, Sms, **Camera**, WebCam, ToneGenerator)
- **UI**
- **Contacts**
- **Intents** (e.g. scanBarcode, search, viewHtml, viewMap, ...)
- **Others** (Android, **Events**, Preferences, Settings, WakeLock, ActivityResult, ApplicationManager)
List of all facades:

http://code.google.com/p/android-scripting/wiki/ApiReference

- enough description for the most methods
• PythonForAndroid-r7b1.apk (Python 2.6.2)
  • http://code.google.com/p/python-for-android/
  • or Python3ForAndroid-r6.apk (Python 3.3)

• Additional you need to install the modules and scripts from the above download area
  • http://python-for-android.googlecode.com/files/python Extras_r13.zip
  • r13 this worked for my devices
  • r14, r15, r16 doesn't – might be because of the r7b1?

• additional Documentation:
SL4A Example: Web2py

• What's Web2py?
  – modern Python Web Application Framework
  – built originally for teaching students by Massimo di Pierro
  – meanwhile an interesting alternative for Python based Web Application Frameworks

• running out-of-the-box
  – integrated SQLite 3
  – integrated Roxen Webserver
  – integrated jQuery

• nevertheless modular and easily adoptable to all well-known Databases and Webservers
Web2py Mobile

• using the mobile as a client via jQuery mobile
  • http://jquerymobile.com/
  • http://web2py.com/plugins/plugin_jqmobile/about

• using the mobile as a server via SL4A – Scripting Layer for Android
  • necessary adoption:
    Python module **shelve** doesn't work correct because of the fallback from **anydbm** to **dumbdbm**
    - Workaround: http://klever.hs-augsburg.de/aktuelles#web2pyAndroid
  • disadvantage:
    problems with disk-caching – but only necessary for optimization
code example for a simple sensors function within the web2py framework, running on my mobile and accessed via WLAN http://10.20.x.y:8000/Android/default/sensors:

```python
import android

droid = android.Android()

def sensors():
    result = None
    while not result:
        # 1 = all Sensors, 100 ms time between readings
        droid.startSensingTimed(1, 100)
        time.sleep(2)
        result = droid.readSensors().result
    return response.render('default/sensors.html', dict(data=result))
```
code example for a simple gps function within the web2py framework, running on my mobile and accessed via WLAN http://10.20.x.y:8000/Android/default/gps :

```python
import android
droid = android.Android()

def gps():
    result = None
    while not result:
        # 2 Minutes as minimum time between updates,
        # 200 m minimum distance between updates
        droid.startLocating(2*60*1000,200)
        time.sleep(2)
        result = droid.readLocation().result
    return response.render('default/sensors.html',dict(data=result))
```
Camera Picture

code example for a simple camera picture function like before [http://10.20.x.y:8000/Android/default/picture](http://10.20.x.y:8000/Android/default/picture):

```python
import android, datetime
droid = android.Android()

def picture():
    dt = datetime.datetime.now()
    path = '/mnt/sdcard/sl4a/scripts/web2py/applications/Android/static/'
    PictureName = 'pic_%s.jpg' % dt.strftime('%Y_%m_%d_%H_%M_%S')
    response = droid.cameraCapturePicture(path+PictureName)
    if response.result['takePicture']:
        return dict(data=PictureName)
    else:
        return dict(data=None)
```
SL4A Web2py Example

- accessing the Web2py Webserver via WLAN
  - Sensors Example
  - Location Example
  - Picture Example

- changing some layout
  - e.g. in `picture.html` in your preferred editor on the PC
  - transfer it to the device via `ftp`
  - reload the site
• Support of the Standard Android UI
  - XML Layout Schema as described in
  - LinearLayout, RelativeLayout
Full UI Example

• adopted from fulluitest.py

```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/background"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ff000000">
    <TextView android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:text="TextView"
              android:id="@+id/textView"
              android:textAppearance="?android:attr/textAppearanceLarge"
              android:gravity="center_vertical|center_horizontal|center"></TextView>
    <EditText android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:id="@+id/editText"
              android:tag="Tag Me"
              android:inputType="textCapWords|textPhonetic|number">
        <requestFocus/>
    </EditText>
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Ok"></Button>
    <CheckBox android:layout_height="wrap_content"
              android:id="@+id/checkBox1"
              android:layout_width="234dp"
              android:text="Howdy, neighbors."
              android:checked="true"></CheckBox>
</LinearLayout>
```
```python
import android
android = android.Android()

def eventloop():
    while True:
        event = droid.eventWait().result
        print event

        if event['name'] == 'click':
            id = event['data']['id']
            if id == 'button':
                droid.fullSetProperty("editText","text","OK has been pressed")
            elif id == 'checkBox1':
                droid.fullSetProperty("textView","text","Other stuff here")
            elif event['data']['checked'] == 'false':
                droid.fullSetProperty("background","backgroundColor","0xff7f0000")
            else:
                droid.fullSetProperty("background","backgroundColor","0xff000000")
        elif event['name'] == 'key':
            if event['data']['key'] == '4':
                return

    layout = '' ... ''

    print droid.fullShow(layout)
    eventloop()
    print droid.fullQuery()
    print "Data entered =", droid.fullQueryDetail("editText").result
    droid.fullDismiss()
```
SL4A Examples on the Mobile

- Accessing the Web2py Webserver on the Device – using the jQuery mobile plugin
  - Sensors Example
  - Location Example

- fulluitest.py

- Full Screen UI Wrapper
Summary and What's Missing?

- **Summary SL4A**
  - tty output and file operations (e.g. logging) as you would expect,
  - most Python standard library modules are integrated,
  - but no tty input
  - simple access to the cool subdevices and media ingredients like Bluetooth, Phone, SMS, Location, Camera, Accelerometer, Magnetic Sensor, ...

- basic UI – adopted to Android UI Design

- **What's missing?**
  - The normal touchable interface is not supported
    - caused by the RPC Architecture
  - What's to do?
    - googling for **Python-For-Android**?
Google for "Python for Android"

Search

About 52,300,000 results (0.20 seconds)

Web

- python-for-android - Py4A - Google Project Hosting
  code.google.com/p/python-for-android/
  This is Python built to run on Android devices. It is made to be used together with
  SL4A (Scripting Layer For Android). Nearly all the actual non-python specific ...
  ➔ Python3 - BuildingModules - Source - Modules

- kivy/python-for-android - GitHub
  github.com > kivy > python-for-android
  19 Jun 2012 – python-for-android - Turn your python application to an Android APK -
  Build your own python and extension.

- Python for Android | Linux Journal
  www.linuxjournal.com/article/10940
  30 Apr 2011 – Mobile app development for smartphones is hot. This is no more
  prevalent than in the Android space where the activity level at times is ...

- Python for Android — Python for Android 1.0 documentation
  python-for-android.rtfd.org/
  Python for android is a project to create your own Python distribution including the ...
  you want, and create an apk including python, ibis, and your application.

- Introducing "Python for Android" | Tuxzone
  txzone.net/2012/01/introducing-python-for-android/
  8 Jan 2012 – I’m glad to share a new project called Python for Android. The goal of
  this project is to package your python application into an APK.
Python for Android - Results

• **python-for-android** - Py4A - Google Project Hosting
code.google.com/p/python-for-android

„Python for Android“ as Py4A in addition to Scripting Layer for Android (SL4A), as described above

• **kivy/python-for-android** – GitHub
github.com/kivy/python-for-android

a second „Python for Android“ exists in addition and combination with Kivy, a Python framework for multi-touch devices

• **Others**

There are some other Python „on“ Android solutions ... (PySide, PyGame, PGS4A, Python-on-a-chip, ...) which are discussed by Thomas Perl and Andreas Schreiber earlier in this week
The Future: Kivy

Kivy - Open source library for rapid development of applications that make use of innovative user interfaces, such as multi-touch apps.
Cross platform

- running on Linux, Windows, MacOSX, Android and IOS
- can use natively most input protocols and devices

GPU Accelerated

- graphics engine is built over OpenGL ES 2
- toolkit with more than 20 widgets
- written in C and Cython

Source: www.kivy.org/#home
Kivy - Architecture

Source: www.kivy.org/docs/guide-index.html
Kivy API

- Kivy framework (..., App, ...)
- Core Abstraction (Audio, Camera, Image, Text, ...)
- Extension Support
- Graphics (Basics (..., Color, Ellipse, Line, ...), Canvas, ...)
- Input management (Providers, Motion Event, ...)
- External libraries (jinja2)
- Modules (Inspector, Monitor, ...)
- Network support (url)
- Widgets (..., Widget, Button, ...)

EuroPython 2012

Nik Klever, UAS Augsburg
Kivy on Android

- as stated here:
- the Android module of Kivy is written in
  - Cython, transformed to JNI via
  - C JNI, to access the Android Java API via the JNI Layer
- Recipes (additional Python modules) are already available
- all about Python For Android:
MyPaint Example

from random import random
from kivy.app import App
from kivy.uix.widget import Widget
from kivy.uix.button import Button
from kivy.graphics import Color, Ellipse, Line

class MyPaintWidget(Widget):
    def on_touch_down(self, touch):
        userdata = touch.ud
        userdata['color'] = c = (random(), 1, 1)
        with self.canvas:
            Color(*c, mode='hsv')
            d = 30
            Ellipse(pos=(touch.x - d/2, touch.y - d/2), size=(d, d))
            userdata['line'] = Line(points=(touch.x, touch.y))

    def on_touch_move(self, touch):
        touch.ud['line'].points += [touch.x, touch.y]

class MyPaintApp(App):
    def build(self):
        return MyPaintWidget()

if __name__ == '__main__':
    MyPaintApp().run()

Source: http://kivy.org/docs/guide/firstwidget.html
Kivy Examples

- MyPaint
- **Touchtracer** from the Kivy Team
- **Showcase** from the Kivy Team
Thanks

- especially thanks to
- **Massimo Di Pierro**, the creator and maintainer of web2py
- the **Kivy Team**
- **Wyn Williams** and **Sylvio Tomati** from the Wifi-Team here at EuroPython, because of helping to bring up a stable Wifi connection for this talk
- additionally all guys here at the EuroPython behind the scene, they did an excellent work!
Questions?

- http://klever.hs-augsburg.de/aktuelles#EuroPython2012
Literature and additional Links

• Lucas Jordan & Pieter Greyling „Practical Android Projects“, 2011, Apress


• Paul Ferrill „Pro Android Python with SL4A“, 2011, Apress