

Lecture 1

Symbian/Python Basics

Tim Bretl

Department of Aerospace Engineering
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign

AE498MPA
August 29, 2007

Some facts about the Nokia N95

- Symbian OS
- 332 MHz CPU
- 64 MB RAM
- 160 MB internal memory + microSD memory card
- 240x320 color screen
- GSM 850/900/1800/1900 MHz + GPRS
- Hardware accelerated 3D graphics (OpenGL ES)
- Two cameras (5 megapixel)
- Internal, fully-functional GPS (1 Hz)
- Bluetooth, wifi, infrared, fm radio, mp3 player, audio recorder, ... , and also a phone

A note about setting up your N95

- Please don't forget to charge it for 24 hours first, as usual.
- Don't forget to install the microSD memory card.
- Please upgrade your phone (requires a SIM card) by downloading [this application](#) before using it.

Why Python?

- It is an object-oriented programming language that supports integration with Java and C/C++, has extensive standard libraries, and is **easy to learn**.
- It is open-source (download for your PC from [here](#)) and cross-platform, although extensions are often specific to particular hardware or operating systems.
- The syntax is simple, so code is typically 1/3 to 1/5 the size of Java/C++.
- Any text editor can be used to create python scripts. No compiler is required (you only need a python interpreter, similar to MATLAB). Try to get a text editor that does syntax highlighting, like [ConTEXT](#), Emacs/XEmacs, or [TextWrangler](#) (a.k.a. a poor man's "BBEdit" on a mac).
- It is supported by Nokia with the port "Python for S60" (PyS60). This port provides extensions that access most of the phone's important features (camera, contacts, calendar, audio input/output, TCP/IP, bluetooth, telephony, GPS, etc.). **In fact, the folks at Nokia use python for their own research and product development.**

Why not Python?

- As an interpreted language, execution speed is not always so fast as compiled languages like Java or C/C++.
- MATLAB has the same drawback, by the way.
- But... it is much easier to use than C/C++ (which is very weird for Symbian OS) and, through extensions, has more access to the phone's features than Java.

How to install python on your phone

1. Download PythonForS60_1_4_0_3rdEd.sis and PythonScriptShell_1_4_0_3rdEd.sis from the [SourceForge project page](#).
2. Install these, in order, on the phone. On a windows machine, use [Nokia PC Suite](#). On a mac, use Bluetooth File Exchange (“Browse device...”) to copy both files to /C:/Data/Installs, then on the phone use Tools->Application Manager (option, then install).
3. Using either Nokia PC Suite or Bluetooth File Exchange, create the folders E:/Python and E:/Python/lib. Anything you put in these folders will be visible as a python script or library on the phone.
4. Test your installation --- run Applications -> Python, and you should start the interpreter.

(See the [open-source site at Nokia](#) for more information.)

A quick way to test your installation

- In the python interpreter (on the phone), select Options->Interactive Console.
- Turn off the dictionary (essentially word completion) by typing # twice.
- Then use your keypad to enter our favorite command:

```
>>> print "Hello, world!"
```

- You can do everything directly on the phone in this way, if you want. But it will be more productive to (1) use scripts, and (2) use a bluetooth console.

Using scripts

- In a text editor on your PC, create a file called “helloworld.py” with the following lines:

```
# import the user interface module
import appuifw

# create an alert (shows up in the UI)
appuifw.note(u"Hello, world!", "info")

# do some math (shows up in the console)
print (1.2+2.3+3.4)/5
```

- Transfer this file to E:/Python/.
- In the python interpreter, do Option->Run Script, and choose helloworld.py from the list. You should see the alert, then the result (should be 1.38) of math in the console.
- This is an example of running a script. (Like an m-file in MATLAB.)

Using a bluetooth console

- The idea is to pipe input/output over a bluetooth connection to a console on your PC. So everything runs on the phone, and you see the UI output on the phone, but you see stdin/stdout on the bluetooth console. This makes it easy to use the interactive python console (like the interactive MATLAB prompt), so you can test out syntax without a script.
- For a lightweight approach, follow [these instructions](#) (works on mac, pc, linux).
- For a heavyweight approach (which I recommend), use [PUTools](#). This is what I used in class. Note that with the following code, you can send a snapshot of the phone to your PC:

```
import sys
sys.stdout.snap()
```

- Also see [this site](#) for more information.

Some syntax examples (numbers, strings)

- Numbers

```
>>> (1+2+3)/5
1
>>> (1+2+3)/5.0
1.2
>>> int(2.6)
2
```

- Strings

```
>>> s='hello'
>>> s
'hello'
>>> print s
hello
>>> print s, 'world!'
hello world!
>>> print s+'world!'
helloworld

>>> s[1]
'e'
>>> s[-1]
'o'
>>> s*3
'hellohellohello'
```

Some syntax examples (lists)

- Lists (think of them as lists of **objects**)

```
>>> list=[1,2,3]
>>> list
[1, 2, 3]
>>> list[0]
1
>>> list.append(4)
>>> list
[1, 2, 3, 4]
>>> list.append('string')
>>> list.append(['firstitem','seconditem'])
>>> list
[1, 2, 3, 4, 'string', ['firstitem', 'seconditem']]
>>> list.pop()
['firstitem', 'seconditem']
>>> list
[1, 2, 3, 4, 'string']
>>> list.pop(0)
1
>>> list
[2, 3, 4, 'string']
```

An interesting thing about lists

- Look at what happens if add a variable, which we have already defined, to a list.

```
>>> q=['tim']
>>> p=['george','gracie',q]
>>> p,q
(['george', 'gracie', ['tim']], ['tim'])
>>> p
['george', 'gracie', ['tim']]
>>> q
['tim']
>>> q.append('andrea')
>>> p
['george', 'gracie', ['tim', 'andrea']]
>>> q
['tim', 'andrea']
```

- Note that `p[2]` and `q` point to the same object. Be careful, though...

```
>>> q='another person'
>>> p
['george', 'gracie', ['tim', 'andrea']]
>>> q
'another person'
```

Some syntax examples (if/else, for)

- If/else (note the colon and the indent, and note the lack of semi-colons)

```
>>> name='tim'
>>> if name=='andrea':
...     print 'Same name!'
... else:
...     print 'Different name!'
...
Different name!
```

- For (note we can iterate over elements of a list)

```
>>> for i in range(0,5):
...     print i
...
0
1
2
3
4

>>> list
[2, 3, 4, 'string']
>>> for i in list:
...     print i
...
2
3
4
string
```

Some syntax examples (while, functions)

- While

```
>>> i=1
>>> while i<4:
...     print i
...     i=i+1
...
1
2
3
```

- Functions (think of these, too, as objects)

```
>>> def f(n):
...     print 2*n
...
>>> f(5)
10
>>> f('tim')
timtim
```

```
>>> f
<function f at 0x733330>
>>> g=f
>>> g
<function f at 0x733330>
>>> g('andrea')
andreaandrea
```

Some syntax examples (on the phone)

- Remember our example script. Type this into your interactive bluetooth console, and you will see the alert on the phone. Note that the string given to `appuifw.note` is prefixed with a “u”. This is because for text to be displayed correctly on the phone, it must be in **unicode**.

```
# import the user interface module
import appuifw

# create an alert (shows up in the UI)
appuifw.note(u"Hello, world!", "info")

# do some math (shows up in the console)
print (1.2+2.3+3.4)/5
```

- Note that you can also typecast normal strings to unicode.

```
>>> s="Hello, world!"
>>> s
'Hello, world!'
>>> unicode(s)
u'Hello, world!'
```

More information

- For the standard library and basic syntax, see the [documentation from python.org](#). In particular, you should go through the [tutorial](#) (or at least parts of it) before next class. This should not take you more than about 2 hours.
- For the pyS60 phone API, download [PythonFoS60_1_4_0_doc.pdf](#) from the [sourceforge site](#). (Make sure you get the right version of the documentation.)
- Also see the [Nokia opensource site](#).
- Also see this [tutorial](#), which is filled with examples.
- If you want a book, try [Learning Python](#).
- The [python for s60 discussion board](#) may also be helpful. Yes, people do respond to your questions fairly quickly, in most cases.

Emulator

- It is possible to test and debug code on your PC only.
- I don't necessarily recommend this, since the setup process is not trivial, since what you get is really a "simulator" and not an "emulator," and since certain key elements of the phone (camera, GPS, etc.) are not simulated.
- Moreover, a lot of your syntax can be tested/debugged just by running python on your PC (eliminating calls to the phone API, of course).
- And of course, by using a bluetooth console, you have a lot of the benefits of PC testing while actually running things on the phone.
- But if you are using a windows machine and are gung-ho...

Setting up an emulator on a windows machine

- Download and install the S60 C++ SDK from [Forum Nokia](#).
- Download the latest pyS60 SDK zip package from [SourceForge](#) that matches the version of the C++ SDK you just installed.
- Unzip the package, which contains sdk_files.zip and an uninstaller.
- Unzip sdk_files.zip in the directory that contains the epoc32 directory of your C++ SDK. So for 3rd Edition, unpack sdk_files.zip in “C:\Symbian\9.2\S60_3rd_FP1”.
- Load your python script into the emulator by copying it to a folder in the epoc32 folder of your S60 SDK. For example, you might put it in “C:\Symbian\9.2\S60_3rd_FP1_4\Epoc32\release\wincsw\udeb\z\Python”.
- Launch the emulator by running “epoc -wincsw” in a terminal.
- Open the python application and run the script as usual.

An important note

- Symbian OS is fairly open to developers. It does this by using a special certification system, called “Symbian Signed”. The *.sis files that you install on the phone are all signed using this system. If you install something that is unsigned, or that is “self-signed”, you won’t have access to all of the phone’s features.
- Currently, we are unable to register new accounts with the Symbian Signed website, hence cannot certify the Python installations on our own phones.
- Consequently, we don’t have access yet to certain key modules, such as the GPS (positioning) module.
- Don’t worry, I am working to correct this problem.
- In the meantime, we have access to a LOT of other stuff, including the camera, bluetooth, file input/output, the user interface, etc.

Acknowledgments

- Some of the material for these notes was taken from the following sources:
 - Vidya Setlur, Nokia Research Center (Palo Alto)
 - Larry Rudolph, MIT
 - Nathan Eagle, MIT