Interfacing with Python

Ravi Chugh
April 19, 2006
Why?

- Python is still a young language
- Many more C/C++/Java libraries than Python
- Ability to communicate between languages is powerful
- Use languages that are suited for different problems, then combine them
Overview

• Many independent ways of achieving cross-language communication:

1) Extending Python using C
2) Embedding Python in C applications
3) Extending Python to support Java
4) Embedding Python in Java apps
1/2: Python/C API

• Can create your own Python modules in C, callable from Python
• Can call Python modules from C
• Both using Python/C API
• `#include <Python.h>`
• Writing modules – boilerplate wrapper for your C code
• See Python/C Tutorial for details
3: Using Java from Python

• Here are two ways:
  – Jython
    • An implementation of Python written completely in Java
    • Runs completely within JVM
  – JPype
    • Native Python interpreter communicates with JVM through JNI (Java Native Interface)
Jython

- Jython is an executable jar file

```python
>>> from java.lang import *
>>> System.out.println("Hello from Jython!")
Hello from Jython!
>>> s = String("No type declaration and no new!")
>>> System.out.println(s)
No type declaration and no new!
```
Jython

• Simple Swing example:

```python
>>> from javax.swing import *
>>> frame = JFrame()
>>> frame.setSize(300, 300)
>>> frame.setTitle("Hello!")
>>> frame.show()
```
Jython

dates.py:

from java.util import Date
from java.lang import System

day = 24 * 60 * 60 * 1000
d = Date()
now = d.getTime()
list = [(Date(now + i*day), now + i*day) for i in range(0, 7)]
for x in list:
    print x[0], "::", x[1]
Jython

jython dates.py:

Tue Apr 18 15:17:30 EDT 2006 : 1145387850579
Wed Apr 19 15:17:30 EDT 2006 : 1145474250579
Thu Apr 20 15:17:30 EDT 2006 : 1145560650579
Fri Apr 21 15:17:30 EDT 2006 : 1145647050579
Sat Apr 22 15:17:30 EDT 2006 : 1145733450579
Sun Apr 23 15:17:30 EDT 2006 : 1145819850579
Mon Apr 24 15:17:30 EDT 2006 : 1145906250579
JPype

• External communication with JVM
• A little more work to set up

```python
>>> from jpype import *
>>> startJVM("C:\Program Files\Java\jdk1.5.0\jre\bin\client\jvm.dll", "-ea")
>>> java.lang.System.out.println("Hello!")
Hello!
```

• Can access Java libraries, but need full package qualifiers
dates2.py:

```python
from jpytype import *
startJVM("C:\Program Files\Java\jdk1.5.0\jre\bin\client\jvm.dll", "-ea")
day = 24 * 60 * 60 * 1000
d = java.util.Date()
now = d.getTime()
list = [(java.util.Date(now + i*day), now + i*day) for i in range(0, 7)]
for x in list:
    print x[0], ":", x[1]
shutdownJVM()  # optional
```

python dates2.py:
<same output as from Jython>
4: Embed Python in Java

• Here are two ways:
  – Use Jython packages directly
    • Very easy to use
  – JEP
    • May be faster, since it runs native Python interpreter, not through JVM
    • Access to Python extensions
Embedded Jython

• Jython is fully implemented in Java
• Can directly use Jython packages
• Add jython.jar to classpath
• Import:
  – import org.python.core.*;
  – import org.python.util.PythonInterpreter;
• And then...
Embedded Jython

public static void example1() {
    PythonInterpreter interp = new PythonInterpreter();
    interp.exec("import sys");
    interp.exec("print 'From Jython inside Java!'"陂);
    System.out.println("Goodbye from Java!"陂);
}

>>>  
From Jython inside Java!  
Goodbye from Java!
Embedded Jython

```java
public static void example2() {
    PythonInterpreter interp = new PythonInterpreter();
    interp.exec("import sys");
    interp.exec("a = 5");
    interp.set("b", new PyInteger(10));
    interp.exec("x = a + b");
    interp.exec("print 'x =', x, 'from Jython'");
    PyObject x = interp.get("x");
    System.out.println("x = " + x + " from Java");
}
```

```shell
>>> 
x = 15 from Jython
x = 15 from Java
```
public static void example3() {
    PythonInterpreter interp = new PythonInterpreter();
    interp.execfile( "hi.py" );
}

>>> 
I am a saved Python program!
Embedded Jython

public static void example4() {
    PythonInterpreter interp = new PythonInterpreter();
    interp.execfile( "interleave.py" );
    interp.exec("print allinter([1,2], [3,4])" );
}

>>> 
[[1, 2, 3, 4], [1, 3, 2, 4], [1, 3, 4, 2], [3, 1, 2, 4], [3, 1, 4, 2], [3, 4, 1, 2]]
Incestuous Interpreter

```java
public static void main(String[] args) throws PyException, IOException {

    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    PythonInterpreter interp = new PythonInterpreter();
    String line = ""
    do {
        System.out.print("%%% ");
        line = br.readLine();
        try {
            interp.exec( line );
        } catch (PyException e) {
            e.printStackTrace();
        }
    } while ( line != null);
}

>>> java IncestuousInterpreter
%%% print "This is odd"
This is odd
%%%
JEP

• Java Embedded Python
• Alternative to embedding Jython
• http://jepp.sourceforge.net/
Resources

- [http://www.seas.upenn.edu/~rkc/JythonResources.htm](http://www.seas.upenn.edu/~rkc/JythonResources.htm)
- Instructions for setting up various techniques
- Links to tutorials and other resources