Integrating Java and Python
Python for Non-Traditional Programmers

Ease of Use

Access To Learning Resources

Fast, powerful libraries: numpy, scipy

IPython Notebook

Great Community
JAVA

Statically -Typed

Verbose

Scales well

Loved by tool developers and enterprise

Eclipse RCP, SWT, EMF

PYTHON

Dynamically-Type

Concise

Ideal for beginners

Loved by academics and researchers

Numpy, scipy, matplotlib, IPython notebook
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}

if __name__ == '__main__':
    print("Hello, World!")
Case Study: Diamond LightSource
List of all perspectives and tools available to perform data analysis.
Requirements

- Control the User Interface

- Python to Java
  - In Python, being able to access Java objects
  - Moving data around
  - Needed for Plotting arrays in the workbench
  - (while still running your existing scripts)

- Java to Python
  - In Java, code can access scripts in Python
  - Workflows, Extending Eclipse with Python
1. PyDev

- C Python
- Jython
- PyPy
- IronPython
- Cython
1. PyDev - Debug

```python
my_sample_function()  
```
1. PyDev - Integration
2. Jython

- Java implementation of Python
- Runs in JVM so can access all JAVA classes

- Jython cannot access C modules
  - numpy, scipy
    - JyNi could solve this,
    - but no solution today
- Separate environment
3. Py4J

- “A Bridge between Java and Python”
- Runs in JVM and allows Python to access JAVA classes
- Enables Python to Java
- Allows control of User Interface and ability to run your scripts

- Creates proxies rather than moving data
- Exposes everything
- No Java to Python support
3. Py4J

- Interactive autocompletion of Java code in Python
- Autogenerated pop-up help based on method signatures
4. AnalysisRPC

Custom solution to move data around

Enhanced Remote Procedure Call
- Datasets (ndarrays)
- Regions Of Interest (ROIs)
- Exceptions

Language Independent
Server and client written for Java and Python
Python

```python
x = numpy.ndarray(...) 
y = numpy.ndarray(...) 
rpcclient.Plotter.line(x, y)
```

Java

```java
import org.eclipse.dawnsci.IDataset
class Plotter {
    public void line(IDataset x, IDataset y) {
        // ...
    }
}
```

Transport

(XML-RPC + Disk/Memory)

Meta and simple data:
- Target → Plotter
- Method → line
  - x → /tmp/x
  - y → /tmp/y

Datasets:
- /tmp/x
- /tmp/y
Demo
DAWN Science Python Integration
Java

interface IFilter {
    IDataset filter(IDataset data);
}

//create client for PyFilter
client.newProxyInstance(IFilter.class)
IDataset result = proxy.filter(data)

Python

class PyFilter:
    def filter(data):
        # manipulate ndarray data
        return data

    # create server
    server.add_handler(PyFilter())

Transport

(XML-RPC + Disk/Memory)

Meta and simple data:
Target ➔ PyFilter
Method ➔ filter
data ➔ /tmp/data

Datasets:
/tmp/data
Python To Java

- Workflows – Python Actor
AnalysisRPC

- Allows moving data around efficiently
- Java to Python, Python to Java
- Generic Calling Infrastructure
- Way of Handling Exceptions

- Only exists in DAWNSci today – needs repackaging for use more widely.
- Support for custom datatypes
Java and Python Integration is possible thanks to a mix of different technologies:

- PyDev
- Jython
- AnalysisRPC
- Py4J

Going forward can we unify this technology?
Science Working Group

- Ongoing initiative

- Looking for a home for AnalysisRPC
  - Make more readily available for science
    - DAWN, EASE, Science Top level project?

- Requirements Capture
  - DAWNSci, ICE, Passerelle, Airbus, TraceCompass
  - Let us know your requirements too!
Questions?

- Tracy Miranda  tracy@kichwacoders.com
- Jonah Graham  jonah@kichwacoders.com

Acknowledgements
- Diamond Light Source, DAWNSci - www.dawnsci.org