

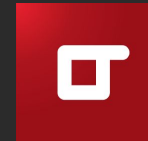
Polynote

A better notebook for Scala



Vincent Brule, Lunatech

Goals



Notebook

Concept

Benefits

Polynote

Introduction

Examples

Conclusion

Almond + Jupyter

Wrap up

Notebook

”Support **interactive** data science and scientific computing across **all programming languages.**”

Scala & its Ecosystem

Part 1: Polynote + request + uJson

Dependencies

Python:

- matplotlib (plotting) - <https://matplotlib.org/>
- pyFunctional (functional programming with collections in a data pipeline style) - <https://www.pyfunctional.org/>

Scala:

- request (request HTTP) - <https://github.com/lihaoyi/requests-scala>
- uJson (decode json response) - <https://www.lihaoyi.com/post/uJsonfastflexibleandintuitiveJSONforScala.html>

The goal of this notebook is to request weather data with Scala (request + uJson) and plot them with Python (matplotlib + pyfunctional). We will experiment the interaction from Python to Scala and vice versa using JEP through Polynote.

Scala

Imports

```
In(3): Scala {}
1 import java.util
2 import java.lang.{ Double => JDouble }
3 import java.lang.{ Integer => JInt }
4 import java.lang.{ Long => JLong }
5 import java.util.Locale
6 import java.time.Instant
7 import java.time.temporal.ChronoUnit
8 import ujson.Value
9 import java.time.format.{ DateTimeFormatter, FormatStyle }
10 import polynote.runtime.python.PythonObject
11 import scala.collection.JavaConverters._
```

Customisable parameters

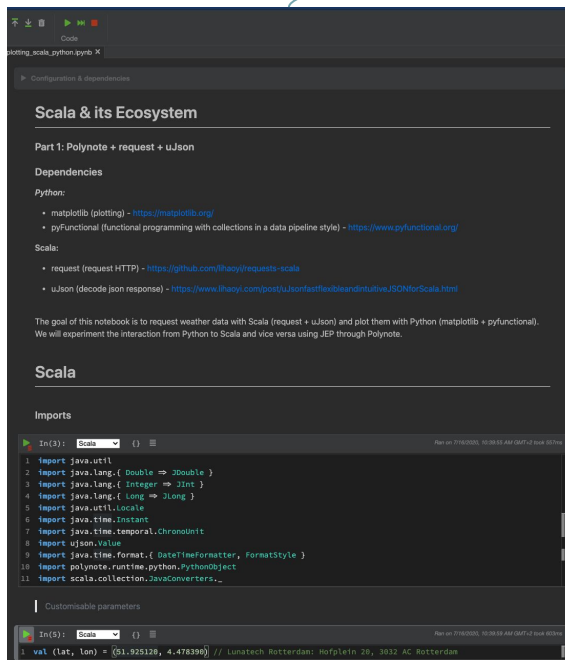
```
In(5): Scala {}
1 val (lat, lon) = (51.925120, 4.478390) // Lunatech Rotterdam: Hofplein 20, 3032 AC Rotterdam
```

Polynote Version: 0.3.10
Build Commit: b0b34a08462583f881a17..

Symbols

Name	Type
lon	Double
lat	Double
Tasks	

General concept



The screenshot shows a Polynote interface with a dark theme. At the top, there's a title bar with icons for file operations and a window title 'plotting_scala_python.pynb X'. Below the title bar is a section titled 'Configuration & dependencies'. The main content area is titled 'Scala & its Ecosystem'. Under this, there's a section 'Part 1: Polynote + request + ujson'. Below that is a 'Dependencies' section with two bullet points: 'matplotlib (plotting) - https://matplotlib.org/' and 'pyfunctional (functional programming with collections in a data pipeline style) - https://www.pyfunctional.org/'. Below the dependencies is a 'Python:' section with two bullet points: 'request (request HTTP) - https://github.com/psf/requests' and 'ujson (decode json response) - https://www.lhacy.com/post/using-off-the-beat-path-to-decode-json-in-scala.html'. Below the Python section is a 'Scala:' section with two bullet points: 'request (request HTTP) - https://github.com/psf/requests' and 'ujson (decode json response) - https://www.lhacy.com/post/using-off-the-beat-path-to-decode-json-in-scala.html'. Below the Scala section is a paragraph: 'The goal of this notebook is to request weather data with Scala (request + ujson) and plot them with Python (matplotlib + pyfunctional). We will experiment the interaction from Python to Scala and vice versa using JEP through Polynote.' Below the paragraph is a section titled 'Scala'. Below the 'Scala' section is an 'Imports' section with a dropdown menu set to 'Scala'. Below the imports is a code editor with the following code:

```
1 import java.util._
2 import java.lang.{ Double => Double }
3 import java.lang.{ Integer => Int }
4 import java.lang.{ Long => Long }
5 import java.util.Locale
6 import java.time.Instant
7 import java.time.temporal.ChronoUnit
8 import ujson.Value
9 import java.time.format.{ DateTimeFormatter, FormatStyle }
10 import polynote.runtime.python.PythonObject
11 import scala.collection.JavaConverters._
```

 Below the code editor is a section titled 'Customisable parameters'. Below the parameters is a code editor with the following code:

```
1 val (lat, lon) = (51.925128, 4.478390) // Lunstede Rotterdam: Hofplein 10, 3032 AC Rotterdam
```



My first notebook



My Use Cases

- **Internship** (interactive)
- **Workshop** (interactive and easy)
- ... **Almost everyday** (fast iteration = easy)

Installation

→ Docker

→ Locally

<https://polynote.org/docs/01-installation.html>

```
66 lines (47 sloc) | 1.94 KB
1
2 FROM jupyter/minimal-notebook:latest
3
4 # -----
5 # --- Constants
6 # -----
7
8 # Set the desired version of almond
9 ENV ALMOND_VERSION="0.9.1"
10
11 # Set the desired version of SBT
12 ENV SBT_VERSION="1.2.7"
13
14 # -----
15 # --- Install dependencies
16 # -----
17
18 USER root
19
20 # Install software-properties and curl
21 RUN \
22 apt-get update \
23 && apt-get install -y curl \
24 && apt-get install -y openjdk-8-jdk \
25 && rm -rf /var/lib/apt/lists/*
26
27 # Define JAVA_HOME environment variable
28 ENV JAVA_HOME /usr/lib/jvm/openjdk-8
29 ENV PATH=${PATH}:${JAVA_HOME}/bin
30
31 # -----
32 # --- Download and install Almond, Scala kernel for Jupyter / IPython 3.
33 # --- For details, see https://github.com/almond-sh/almond
34 # -----
35
36 # Download SBT
37 RUN curl -sL --retry 5 "https://github.com/sbt/sbt/releases/download/v${SBT_VERSION}/sbt-${SBT_VERSION}.tgz" \
38 | gunzip \
39 | tar -x -C "/tmp/" \
40 && mv "/tmp/sbt" "/opt/sbt-${SBT_VERSION}" \
41 && chmod +x "/opt/sbt-${SBT_VERSION}/bin/sbt"
42
43 ENV PATH=${PATH}:/opt/sbt-${SBT_VERSION}/bin/
44
45 RUN curl -L -o /usr/local/bin/coursier https://git.io/coursier && chmod +x /usr/local/bin/coursier
46
47 # Switch back to jovyan to avoid accidental container runs as root
48 USER $NB_USER
49
50 WORKDIR /tmp
51
52 ENV SCALA_VERSION="2.12.10"
53
54 RUN coursier bootstrap \
55 -r jitpack \
56 -i user -I user:sh.almond:scala-kernel-api_${SCALA_VERSION}:SALMOND_VERSION \
57 sh.almond:scala-kernel_${SCALA_VERSION}:SALMOND_VERSION \
58 -o almond && \
59 chmod +x almond && \
60 ./almond --install
61
62 RUN rm /tmp/almond
63
64 WORKDIR /home/$NB_USER/work
65
66 CMD jupyter notebook --port=8888 --no-browser --ip=0.0.0.0 --allow-root
```


Polynote

“Polynote is an experimental polyglot notebook environment.”



Demo



Comparison

Jupyter

- + Older
- + Big community
- + Customizable
- + More languages

Polynote

- + First class Scala Support
- + Code editing (autocomplete, jump to definition?)
- + Text editing (WYSIWYG editor)
- + Multi-Language Support
- + Runtime insight

Conclusion

- Your **favorite language** is supported
- Collaborative + Fast iteration

Thanks for listening!

vincent.brule@lunatech.nl

github.com/vincentbrule

twitter.com/BruleVincent



Vincent Brulé, Lunatech