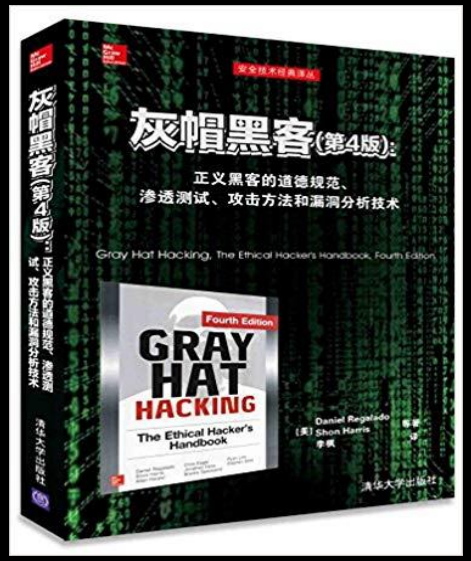
Scala Meetup China
(#8, 2023 Online)

for Interactive Notebook

Feng Li (李枫)
hkli2013@126.com
Jan 14, 2023

Who Am I

- An indie developer from China
- The main translator of the book «Gray Hat Hacking The Ethical Hacker's Handbook, Fourth Edition» (ISBN：9787302428671) & «Linux Hardening in Hostile Networks, First Edition» (ISBN: 9787115544384)



- Pure software developmement for ~15 years(~11 years on Mobile dev)
- Actively participating Open Source Communities
- https://github.com/XianBeiTuoBaFeng2015/MySlides
- Recently, focus on infrastructure of Cloud/Edge Computing, AI, IoT, Programming Languages & Runtimes, Network, Virtualization, RISC-V, EDA, 5G/6G…

Agenda

I. Background
- Tech Stack
- Testbed

II. Project Almond on ARM
- Overview
- RPi4

III. Project Polynote on ARM
- Overview
- RPi4

IV. New Exploration Technologies
- Kernel-free Notebook
- Wasm-based Notebook
- Data-oriented Notebook
- Ray.Graal
- Cross-platform integration for Polyglot Programming

V. Wrap-up

I. Background

1) Tech Stack

REPL

- **https://en.wikipedia.org/wiki/Read-eval-print_loop**

A **read–eval–print loop** (**REPL**), also termed an **interactive toplevel** or **language shell**, is a simple interactive computer programming environment that takes single user inputs, executes them, and returns the result to the user; a program written in a REPL environment is executed piecewise.[1] The term usually refers to programming interfaces similar to the classic Lisp machine interactive environment. Common examples include command-line shells and similar environments for programming languages, and the technique is very characteristic of scripting languages.[2]

…

REPLs facilitate exploratory programming and debugging because the programmer can inspect the printed result before deciding what expression to provide for the next read. The read–eval–print loop involves the programmer more frequently than the classic edit–compile–run–debug cycle.

Because the *print* function outputs in the same textual format that the *read* function uses for input, most results are printed in a form that could be copied and pasted back into the REPL. However, it is sometimes necessary to print representations of elements that cannot sensibly be read back in, such as a socket handle or a complex class instance. In these cases, there must exist a syntax for unreadable objects. In Python, it is the `<__module__.class instance>` notation, and in Common Lisp, the `#<whatever>` form. The REPL of CLIM, SLIME, and the Symbolics Lisp Machine can also read back unreadable objects. They record for each output which object was printed. Later when the code is read back, the object will be retrieved from the printed output.

REPLs can be created to support any text-based language. REPL support for compiled languages is usually achieved by implementing an interpreter on top of a virtual machine which provides an interface to the compiler. For example, starting with JDK 9, Java included JShell as a command-line interface to the language. Various other languages have third-party tools available for download that provide similar shell interaction with the language.

…

1.1 Project Jupyter

1.1.1 Overview

- https://en.wikipedia.org/wiki/Project_Jupyter

Project Jupyter (/ˈdʒuːpɪtər/ (listen)) is a project with goals to develop open-source software, open standards, and services for interactive computing across multiple programming languages. It was spun off from IPython in 2014 by Fernando Pérez and Brian Granger. Project Jupyter's name is a reference to the three core programming languages supported by Jupyter, which are Julia, Python and R. Its name and logo are an homage to Galileo's discovery of the moons of Jupiter, as documented in notebooks attributed to Galileo. Project Jupyter has developed and supported the interactive computing products Jupyter Notebook, JupyterHub, and JupyterLab. Jupyter is financially sponsored by NumFOCUS.[1]

History:

A manuscript ascribed to Galileo Galilei's observations of Jupiter (⊛) and four of its moons (✱), which inspired the Jupyter logo

The first version of Notebooks for IPython was released in 2011 by a team including Fernando Pérez, Brian Granger, and Min Ragan-Kelley.[2] In 2014, Pérez announced a spin-off project from IPython called Project Jupyter.[3] IPython continues to exist as a Python shell and a kernel for Jupyter, while the notebook and other language-agnostic parts of IPython moved under the Jupyter name.[4][5] Jupyter supports execution environments (called "kernels") in several dozen languages, including Julia, R, Haskell, Ruby, and Python (via the IPython kernel).

In 2015, about 200,000 Jupyter notebooks were available on GitHub. By 2018, about 2.5 million were available.[6] In January 2021, nearly 10 million were available, including notebooks about the first observation of gravitational waves[7] and about the 2019 discovery of a supermassive black hole.[8]

Major cloud computing providers have adopted the Jupyter Notebook or derivative tools as a frontend interface for cloud users. Examples include Amazon SageMaker Notebooks,[9] Google's Colaboratory,[10][11] and Microsoft's Azure Notebook.[12]

Visual Studio Code supports local development of Jupyter notebooks. As of July 2022, the Jupyter extension for VS Code has been downloaded over 40 million times, making it the second-most popular extension in the VS Code Marketplace.[13]

The Atlantic published an article entitled "The Scientific Paper Is Obsolete" in 2018, discussing the role of Jupyter Notebook and the Mathematica notebook in the future of scientific publishing.[14] Economist Paul Romer, in response, published a blog post in which he reflected on his experiences using Mathematica and Jupyter for research, concluding in part that Jupyter "does a better job of delivering what Theodore Gray had in mind when he designed the Mathematica notebook."[15]

In 2021, *Nature* named Jupyter as one of ten computing projects that transformed science.[8]

- ...

1.1.1.1 IPython

Overview

- **https://ipython.org/**

IPython provides a rich architecture for interactive computing with:

- A powerful interactive shell.
- A kernel for Jupyter.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into your own projects.
- Easy to use, high performance tools for parallel computing.

```
In [1]: print('Hello IPython')
Hello IPython

In [2]: 21 * 2
Out[2]: 42

In [3]: def say_hello(name):
   ...:     print('Hello {name}'.format(name=name))
   ...:
```

```
In [1]: %timeit range(1000)
100000 loops, best of 3: 7.76 us per loop

In [2]: %%timeit x = range(10000)
...: max(x)
...:
1000 loops, best of 3: 223 us per loop
```

- **Jupyter Notebook(formerly known as the IPython Notebook)**

Jupyter and the future of IPython

IPython is a growing project, with increasingly language-agnostic components. IPython 3.x was the last monolithic release of IPython, containing the notebook server, qtconsole, etc. As of IPython 4.0, the language-agnostic parts of the project: the notebook format, message protocol, qtconsole, notebook web application, etc. have moved to new projects under the name Jupyter. IPython itself is focused on interactive Python, part of which is providing a Python kernel for Jupyter.

- **https://github.com/ipython**
- **https://ipython.readthedocs.io/**
- **...**

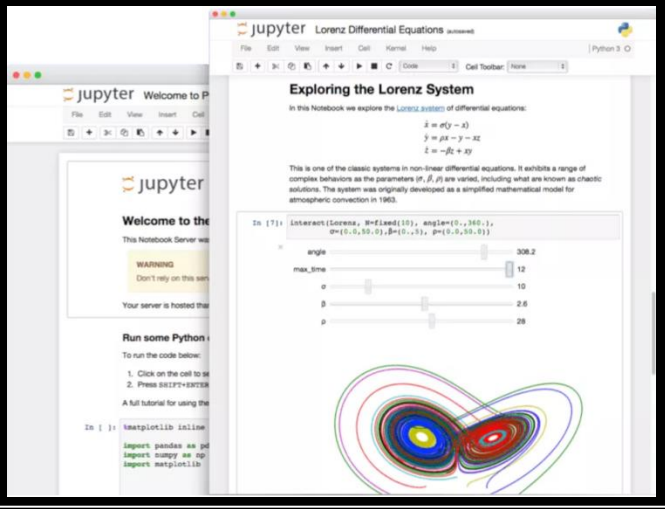
1.1.2 Jupyter

Overview

- **https://jupyter.org/**
  Free software, open standards, and web services for interactive computing across all programming languages.

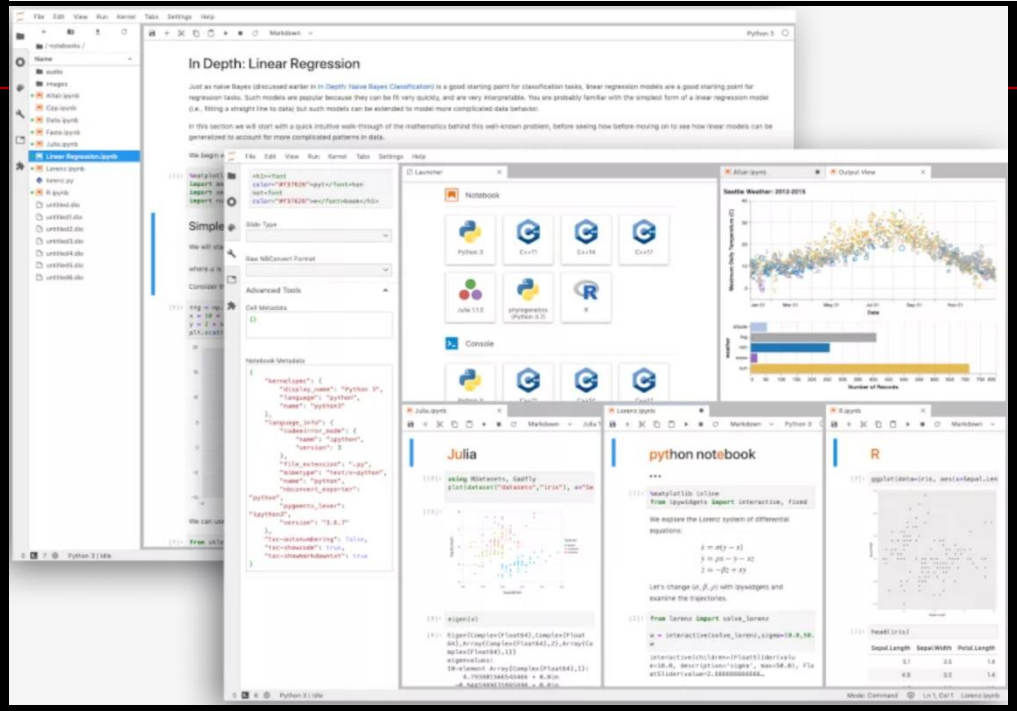
- **Jupyter Notebook: The Classic Notebook Interface**

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.



JupyterLab: A Next-Generation Notebook Interface

JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.



Jupyterhub

A multi-user version of the notebook designed for companies, classrooms and research labs

Pluggable authentication

Manage users and authentication with PAM, OAuth or integrate with your own directory service system.

Centralized deployment

Deploy the Jupyter Notebook to thousands of users in your organization on centralized infrastructure on- or off-site.

Container friendly

Use Docker and Kubernetes to scale your deployment, isolate user processes, and simplify software installation.

Code meets data

Deploy the Notebook next to your data to provide unified software management and data access within your organization.

Open Standards for Interactive Computing

Project Jupyter promotes open standards that third-party developers can leverage to build customized applications. Think HTML and CSS for interactive computing on the web.

Notebook Document Format

Jupyter Notebooks are an open document format based on JSON. They contain a complete record of the user's sessions and include code, narrative text, equations, and rich output.

Interactive Computing Protocol

The Notebook communicates with computational Kernels using the Interactive Computing Protocol, an open network protocol based on JSON data over ZMQ, and WebSockets.

The Kernel

Kernels are processes that run interactive code in a particular programming language and return output to the user. Kernels also respond to tab completion and introspection requests.

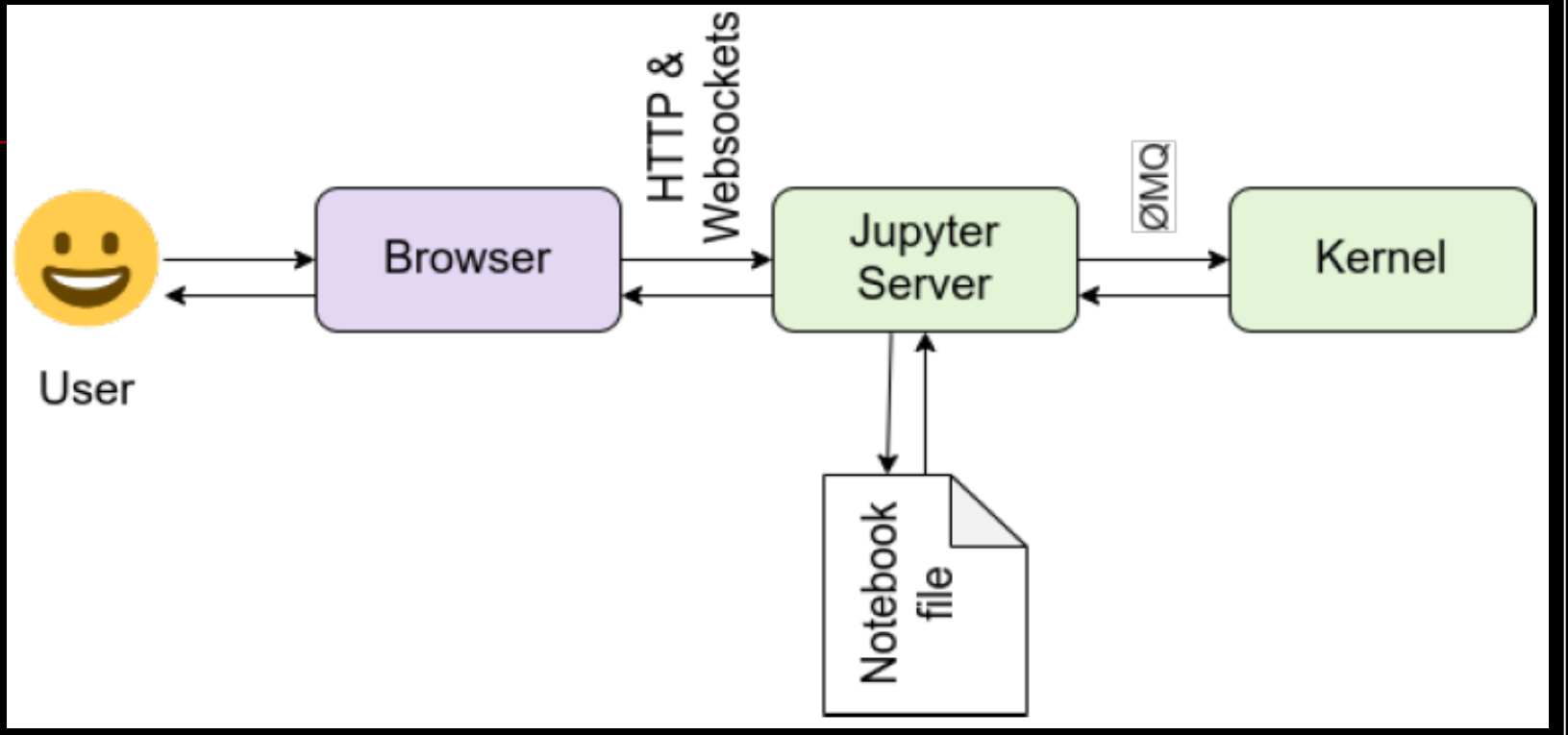
- **https://voila.readthedocs.io/en/stable/**

Voilà allows you to convert a Jupyter Notebook into an interactive dashboard that allows you to share your work with others. It is secure and customizable, giving you control over what your readers experience.

- **https://jupyter.org/widgets**
- **…**

1.1.2.1 Architecture & Design

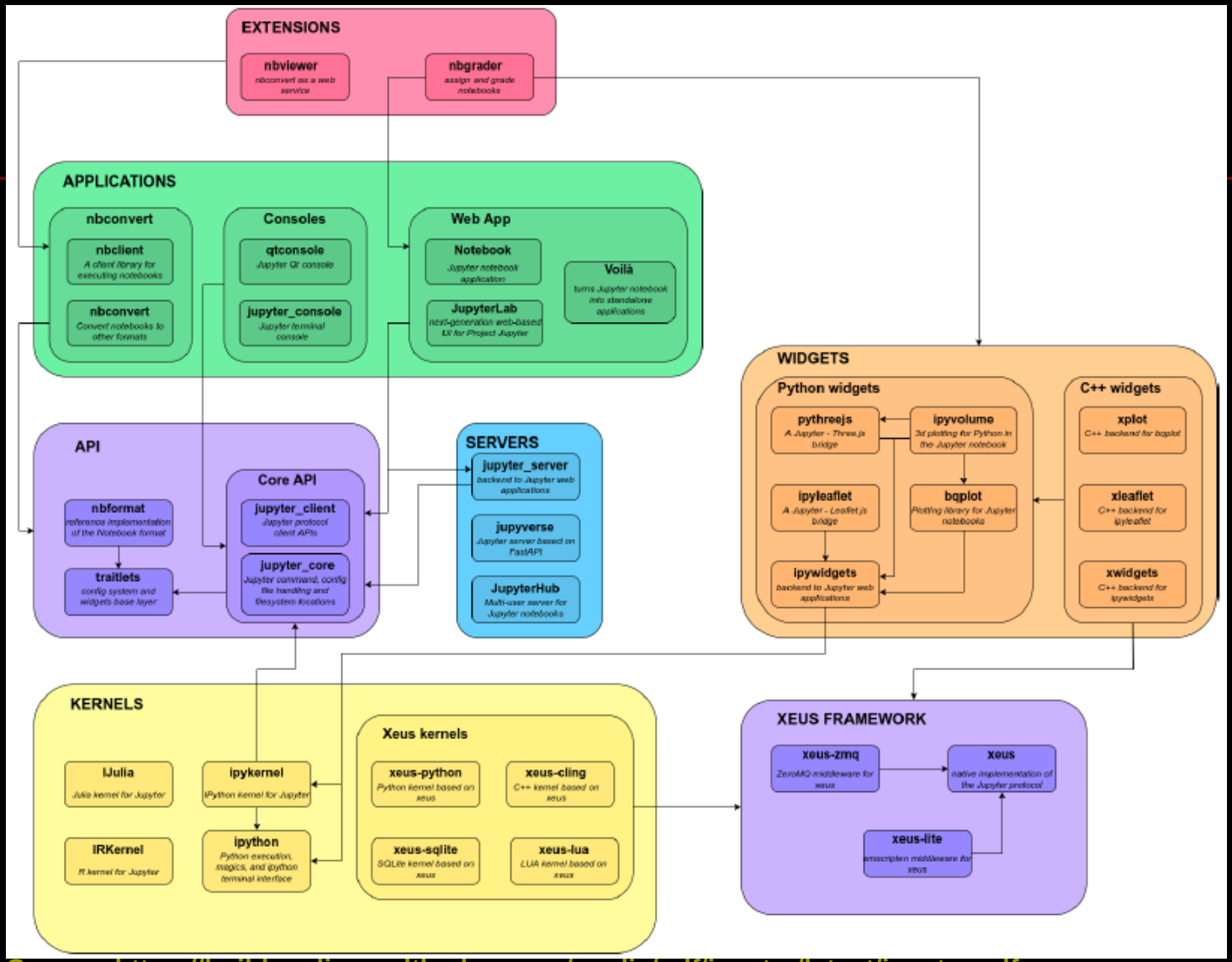
- Workflow



The Jupyter server is a communication hub. The browser, notebook file on disk, and kernel cannot talk to each other directly. They communicate through the Jupyter server. The Jupyter server, not the kernel, is responsible for saving and loading notebooks, so you can edit notebooks even if you don't have the kernel for that language—you just won't be able to run code. The kernel doesn't know anything about the notebook document: it just gets sent cells of code to execute when the user runs them.

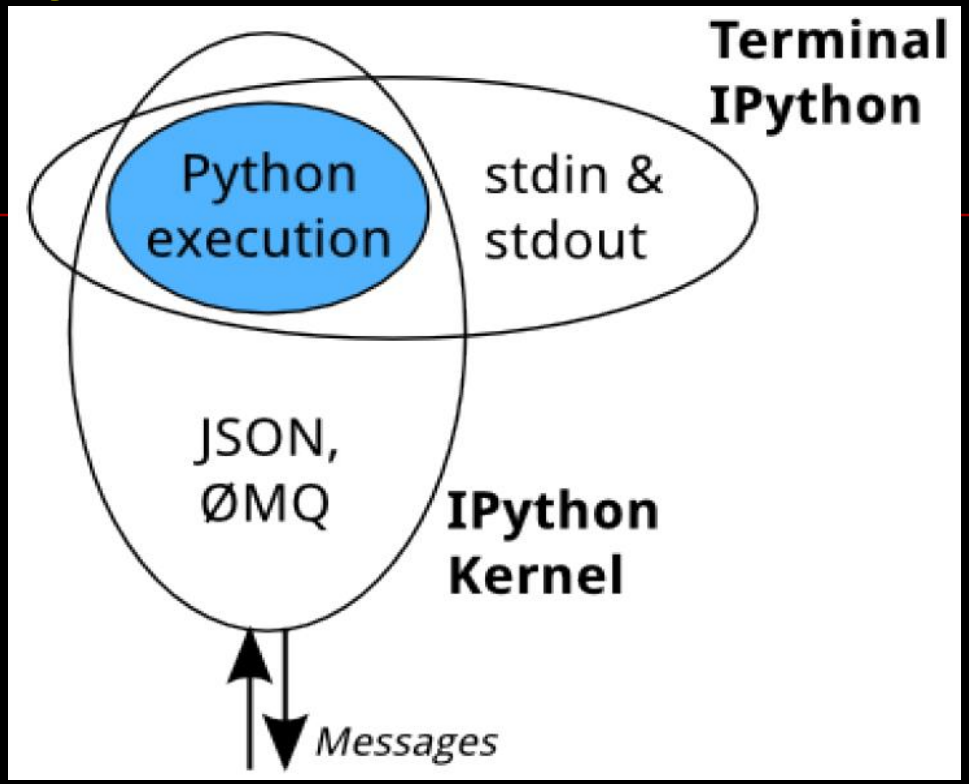
Source: https://buildmedia.readthedocs.org/media/pdf/jupyter/latest/jupyter.pdf

- A high level visual overview of project relationships as of 2022.



Source: https://buildmedia.readthedocs.org/media/pdf/jupyter/latest/jupyter.pdf

- **IPython Kernel**



All the other interfaces —- the Notebook, the Qt console, `ipython console` in the terminal, and third party interfaces —- use the IPython Kernel. IPykernel is a separate process which is responsible for running user code, and things like computing possible completions. Frontends, like the notebook or the Qt console, communicate with the IPython Kernel using JSON messages sent over ZeroMQ sockets; the protocol used between the frontends and the IPython Kernel is described in Messaging in Jupyter.

The core execution machinery for the kernel is shared with terminal IPython.

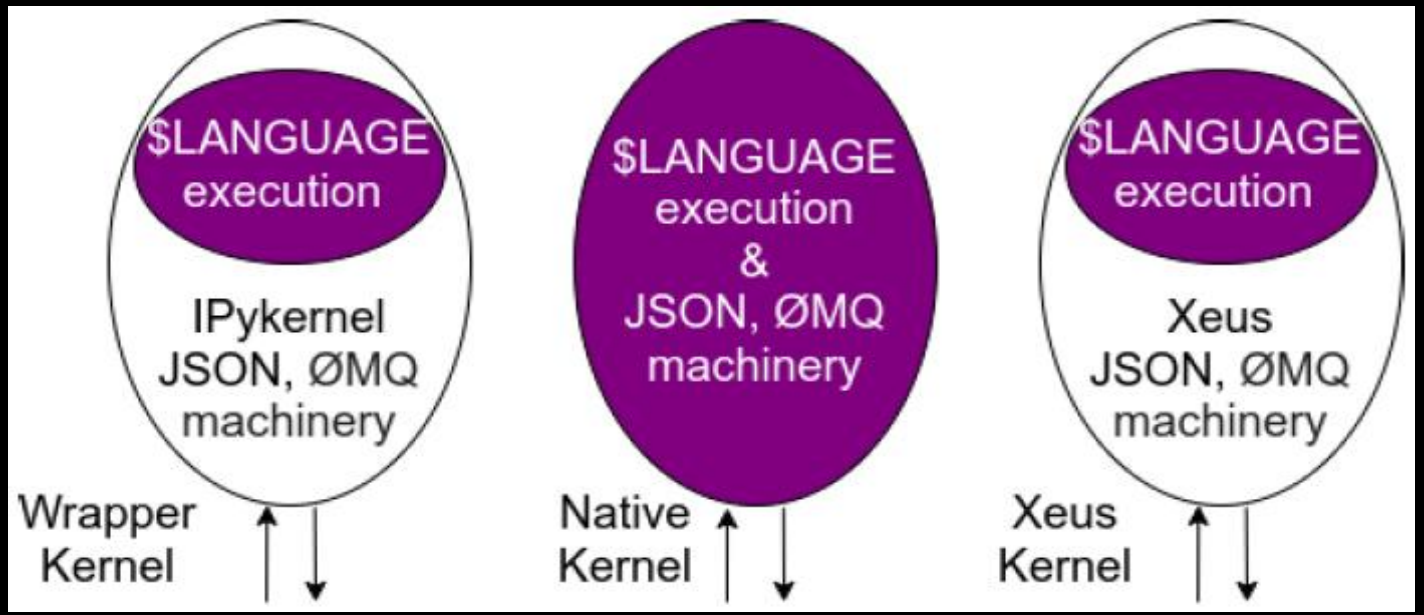
Source: https://buildmedia.readthedocs.org/media/pdf/jupyter/latest/jupyter.pdf

A kernel process can be connected to more than one frontend simultaneously. In this case, the different frontends will have access to the same variables.

This design was intended to allow easy development of different frontends based on the same kernel, but it also made it possible to support new languages in the same frontends, by developing kernels in those languages, and we are refining IPython to make that more practical.

Today, there are three ways to develop a kernel for another language:

- Wrapper kernels reuse the communications machinery from IPykernel, and implement only the core execution part.
- Native kernels implement execution and communications in the target language.
- Kernels based on xeus, a native implementation of the protocol, implement the language-specific part of the kernels. Contrary to the wrapper approach, ***xeus*** does not depend on a python runtime.



Source: https://buildmedia.readthedocs.org/media/pdf/jupyter/latest/jupyter.pdf

1.1.2.2 JupyterLab

Overview

- **https://github.com/jupyterlab/jupyterlab**

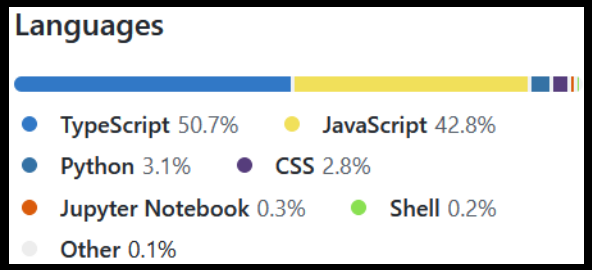
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture. Currently ready for users.

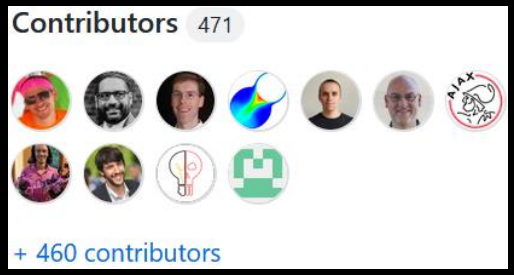
JupyterLab is the next-generation user interface for Project Jupyter offering all the familiar building blocks of the classic Jupyter Notebook (notebook, terminal, text editor, file browser, rich outputs, etc.) in a flexible and powerful user interface. JupyterLab will eventually replace the classic Jupyter Notebook.

JupyterLab can be extended using npm packages that use our public APIs. The *prebuilt* extensions can be distributed via PyPI, conda, and other package managers. The *source* extensions can be installed directly from npm (search for jupyterlab-extension) but require additional build step. You can also find JupyterLab extensions exploring GitHub topic jupyterlab-extension. To learn more about extensions, see the user documentation.

The current JupyterLab releases are suitable for general usage, and the extension APIs will continue to evolve for JupyterLab extension developers.

- **Src**





- **https://jupyterlab.readthedocs.io/**
- **…**

1.1.3 Kernels

Overview

- **https://github.com/jupyter/jupyter/wiki/Jupyter-kernels**

Kernel Zero is IPython, which you can get through ipykernel, and is still a dependency of jupyter. The IPython kernel can be thought of as a reference implementation, as CPython is for Python.

Name	Jupyter/IPython Version	Language(s) Version	3rd party dependencies	Example Notebooks
Micronaut		Python>=3.7.5, Groovy>3	Micronaut	https://github.com/stainlessai/micronaut-jupyter/blob/master/examples/basic-service/notebooks/use-library.ipynb
Agda kernel		2.6.0		https://mybinder.org/v2/gh/lclem/agda-kernel/master?filepath=example/LabImp.ipynb
Dyalog Jupyter Kernel		APL (Dyalog)	Dyalog >= 15.0	Notebooks
Coarray-Fortran	Jupyter 4.0	Fortran 2008/2015	GFortran >= 7.1, OpenCoarrays, MPICH >= 3.2	Demo, Binder demo
LFortran				Binder demo
Ansible Jupyter Kernel	Jupyter 5.6.0.dev0	Ansible 2.x		Hello World
sparkmagic	Jupyter >=4.0	Pyspark (Python 2 & 3), Spark (Scala), SparkR (R)	Livy	Notebooks, Docker Images

...

IScala		Scala		
almond (old name: Jupyter-scala)	IPython>=3.0	Scala>=2.10		examples

...

- https://jupyter-client.readthedocs.io/en/latest/kernels.html

...

A 'kernel' is a program that runs and introspects the user's code. IPython includes a kernel for Python code, and people have written kernels for several other languages.

At kernel startup, Jupyter passes the kernel a connection file. This specifies how to set up communications with the frontend.

There are three options for writing a kernel:

1. You can reuse the IPython kernel machinery to handle the communications, and just describe how to execute your code. This is much simpler if the target language can be driven from Python. See Making simple Python wrapper kernels for details.
2. You can implement the kernel machinery in your target language. This is more work initially, but the people using your kernel might be more likely to contribute to it if it's in the language they know.
3. You can use the xeus library that is a C++ implementation of the Jupyter kernel protocol. Kernel authors only need to implement the language-specific logic in their implementation (execute code, auto-completion...). This is the simplest solution if your target language can be driven from C or C++: e.g. if it has a C-API like most scripting languages. Check out the xeus documentation for more details. Examples of kernels based on xeus include:

    - xeus-cling
    - xeus-python
    - JuniperKernel

...

- https://github.com/Calysto/metakernel
- https://github.com/jupyter-xeus
- https://github.com/google/evcxr
- ...

1.2 Scala Projects
1.2.1 Overview

- ...

1.2.2 Mill

1.2.2.1 Overview

- https://com-lihaoyi.github.io/mill/mill/Intro_to_Mill.html

Mill is your shiny new Java/Scala build tool! Scared of SBT? Melancholy over Maven? Grumbling about Gradle? Baffled by Bazel? Give Mill a try!

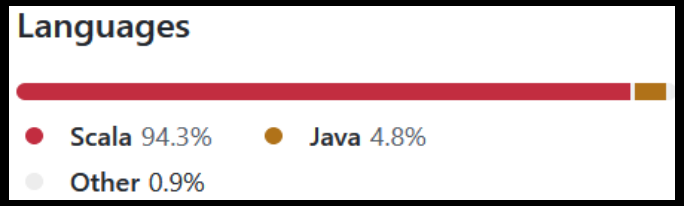
Mill aims for simplicity by reusing concepts you are already familiar with, borrowing ideas from modern tools like Bazel. It lets you build your projects in a way that's simple, fast, and predictable.

Mill has built-in support for the Scala programming language, and can serve as a replacement for SBT. It can be extended to support any other language or platform via modules (written in Java or Scala) or through external subprocesses.

If you are using Mill, you will find the following book by the Author useful in using Mill to the fullest:

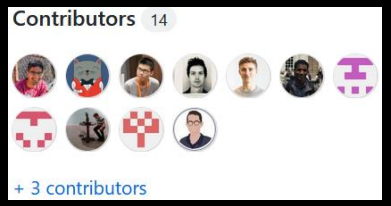
- https://handsonscala.com/

- https://github.com/com-lihaoyi/mill



- …

- https://github.com/com-lihaoyi/mill



- https://scalapy.dev/docs/
- https://scalapy.dev/docs/jupyter-notebooks/
- …

1.2.3 ScalaPy

1.2.3.1 Overview

- https://scalapy.dev/

  Use Python libraries from the comfort of Scala.

- Features

Complete Ecosystem

Use any Python library you can dream of. Want to train neural networks on GPUs with TensorFlow? ScalaPy supports it.

```
val tf = py.module("tensorflow")
val hello = tf.constant("Hello, TensorFlow!")
val sess = tf.Session()
println(sess.run(hello)) // Hello, TensorFlow!
```

```
val np = py.module("numpy").as[NumPy]
val xData = np.random.rand(100).astype(np.floate32)
// error: value floate32 is not a member of NumPy
```

Strong Typing

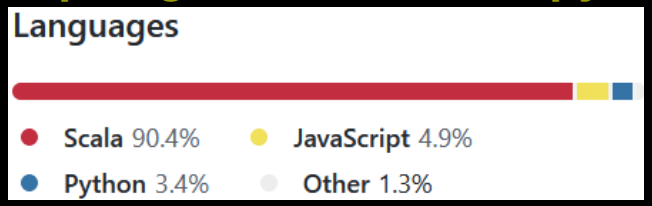
Add type definitions for Python libraries as you go to catch bugs before they happen in production.

Performant Interop

Compile to native binaries with Scala Native to unlock maximum performance with direct bindings to CPython.

```
$ time ./scalapy-out
Hello from native ScalaPy!
The length of the Python list is 100
        0.03 real         0.02 user         0.00 sys
```

- https://github.com/scalapy/scalapy



- https://scalapy.dev/docs/
- https://scalapy.dev/docs/jupyter-notebooks/
- …

1.2.4 Scala.js

1.2.4.1 Overview

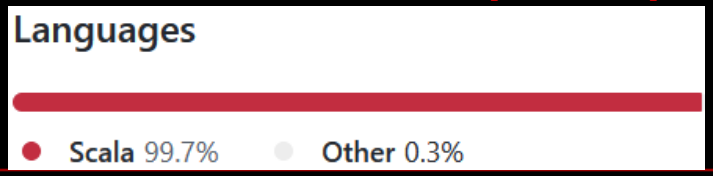
- https://www.scala-js.org/

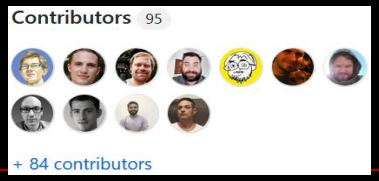
  A safer way to build robust front-end web applications!

- Features

Feature	JavaScript ES5	JavaScript ES6	TypeScript	Scala.js
Interoperability				
Fully EcmaScript5 compatible	●	●	●	●
No compilation required	●	●	○	○
Use existing JS libraries	●	●	●	●
Language features				
Classes	○	●	●	●
Modules	○	●	●	●
Support for types	○	○	●	●
Strong type system	○	○	○	●
Extensive standard libraries	○	○	○	●
Optimizing compiler	○	○	○	●
Macros, to extend the language	○	○	○	●
IDE support				
Catch most errors in IDE	○	○	●	●
Easy and reliable refactoring	○	○	●	●
Reliable code completion	○	○	●	●

- https://github.com/scala-js/scala-js
  The Scala to JavaScript compiler.





- https://www.scala-js.org/doc/sjs-for-js/
- …

1.2.5 Scala CLI

1.2.5.1 Overview

- https://scala-cli.virtuslab.org/

Scala CLI is a command-line tool to interact with the Scala language.

It lets you compile, run, test, and package your Scala code (and more!)

```
$ cat demo.scala
def niceArgs(args: String*): String =
  args.map(_.capitalize).mkString("Hello: ", ", ", "!")

@main def demo(args: String*) = println(niceArgs(args*))
$ scala-cli demo.scala -- Ala jake Mike
Compiling project (Scala 3.1.3, JVM)
Compiled project (Scala 3.1.3, JVM)
Hello: Ala, Jake, Mike!
```

- Features

Intuitive, simple

No complicated mechanisms, tasks, plugins or extensions: just a single-module. All our commands have multiple aliases and follow well-known conventions.

Fast

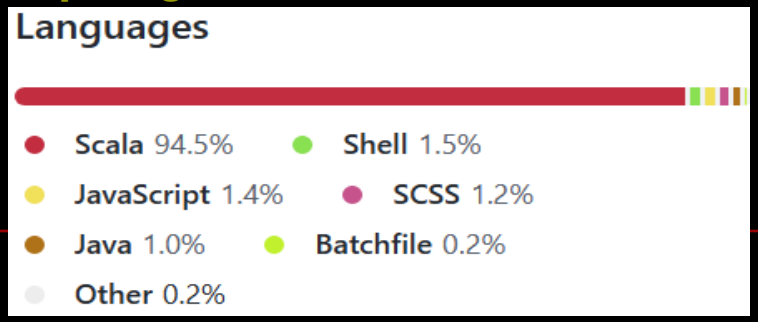
Scala CLI is optimized to be as fast as possible. CLI is compiled to native code and compilations are offloaded to bloop.

Command-line first

Scala CLI does not require a configuration file, and all in-file configurations can be overridden by command-line. No additional installation or setup of an environment (such as a specific working directory) are required.

...

- https://github.com/Virtuslab/scala-cli



- https://scala-cli.virtuslab.org/docs/cookbooks/intro
- …

1.2.6 Ammonite

1.2.6.1 Overview

- http://ammonite.io/

Ammonite lets you use the Scala language for scripting purposes: in the REPL or as scripts.

Ammonite-REPL

A Modernized Scala REPL. With syntax highlighting, multi-line editing, the ability to load maven artifacts directly in the REPL, and many other quality-of-life improvements missing in the default Scala REPL.

Scala Scripts

Lightweight Programming in Scala. Create scripts that you can run easily from the command line, without the overhead of setting up a "project" or waiting for SBT's slow startup times.

If you use Ammonite, you will probably find the following book by the Author helpful in using Ammonite to the fullest:

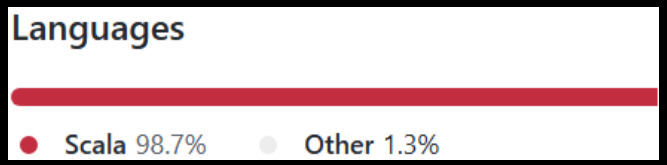
- https://handsonscala.com/

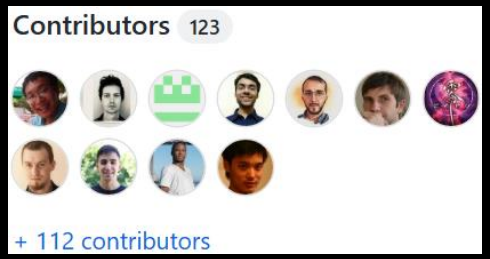
Ammonite is a project by Li Haoyi. If you use Ammonite and enjoyed it, please chip in to support our development at:

- https://www.patreon.com/lihaoyi

```
3. fish  /Users/haoyi (java)
haoyi-mbp:~ haoyi$ ~/amm
Loading...
Welcome to the Ammonite Repl 0.4.6
(Scala 2.11.7 Java 1.8.0_25)
@ List(Seq(Seq("mgg", "mgg", "lols"), Seq("mgg", "mgg")), Seq(Seq("ggx", "ggx"),Seq("ggx", "ggx", "wt
  fx"))) // pretty printing
res0: List[Seq[Seq[String]]] = List(
  List(List("mgg", "mgg", "lols"), List("mgg", "mgg")),
  List(List("ggx", "ggx"), List("ggx", "ggx", "wtfx"))
)
@ load.ivy("com.lihaoyi" %% "scalatags" % "0.4.5") // load a library

@ import scalatags.Text.all._
import scalatags.Text.all._
@ a("omg", href:="www.google.com").render
res3: String = """
<a href="www.google.com">omg</a>
"""
@
```

- https://github.com/com-lihaoyi/Ammonite





- http://ammonite.io/#Ammonite-REPL
- http://ammonite.io/#AmmoniteCookbook
- https://github.com/alexarchambault/ammonite-spark
  Run spark calculations from Ammonite.
- ...

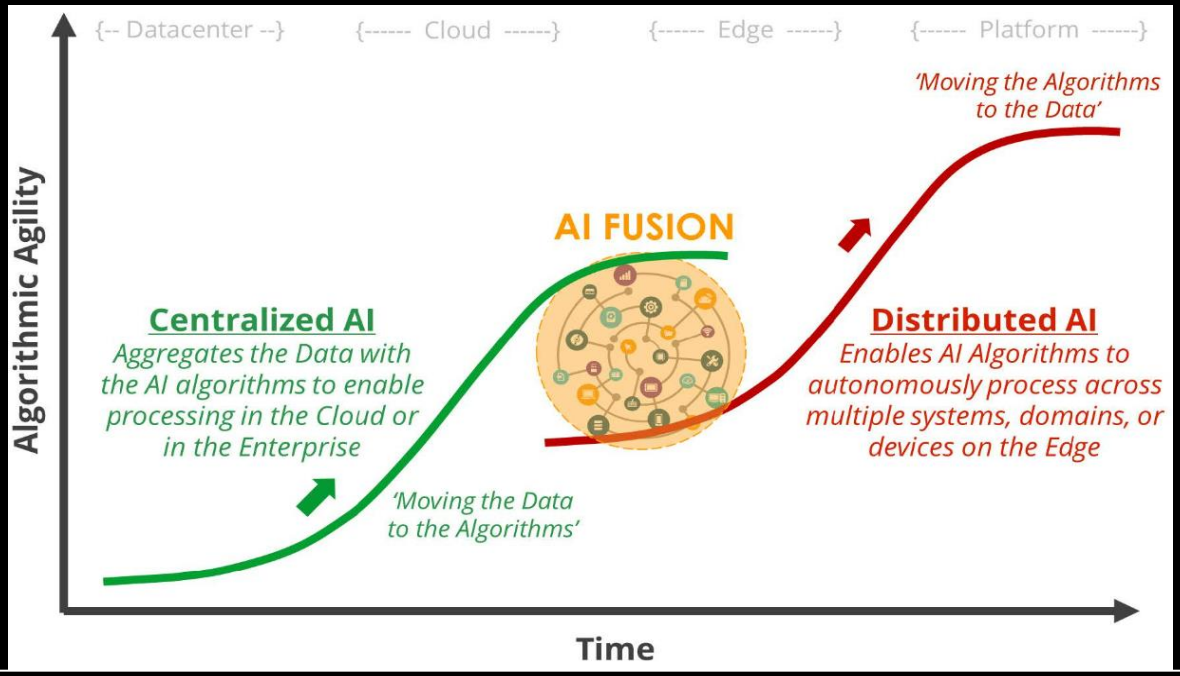
1.3 Distributed AI

1.3.1 Overview

- https://en.wikipedia.org/wiki/Distributed_artificial_intelligence

> **Distributed Artificial Intelligence (DAI)** also called Decentralized Artificial Intelligence[1] is a subfield of artificial intelligence research dedicated to the development of distributed solutions for problems. DAI is closely related to and a predecessor of the field of multi-agent systems.

- https://www.xenonstack.com/blog/distributed-ai-latest-trends
- https://developer.ibm.com/learningpaths/get-started-distributed-ai-apis/what-is-distributed-ai/
- https://engineering.cmu.edu/accelerator/news/2021/03/03-ai-fusion.html



AI Fusion

AI Frameworks & Algorithms

- AI Optimized Models
- Distributed Algorithmic Processing
- Autonomous Data Discovery
- Amorphous Learning & Prediction

AI Fabric & Abstraction Layer

- Dynamic AI Resource Management & Prioritization
- Algorithmic orchestration and ensembling between AI in the Enterprise and AI embedded in platforms and sensors at the Edge

AI Microelectronics & AI-infused Systems

- Edge AI Hardware
- AI Optimized Micro-Electronics
- Cognitive Sensors & Synthesis
- AI-Ready ICs supporting Algorithmic Multi-Threading

AI Engineering & Security

- AI Verification and Validation
- AI System Architecture and Integration
- AI Deployment and Monitoring *(AI DevOps)*
- Data / Model Management & Coherence
- AI Infrastructure Orchestration
- AI Policy Mechanisms *(Ethical Principles)*

1.3.2 Arrow

1.3.2.1 Overview

- https://en.wikipedia.org/wiki/Apache_Arrow

Apache Arrow is a language-agnostic software framework for developing data analytics applications that process columnar data. It contains a standardized column-oriented memory format that is able to represent flat and hierarchical data for efficient analytic operations on modern CPU and GPU hardware.[3][4][5][6][7] This reduces or eliminates factors that limit the feasibility of working with large sets of data, such as the cost, volatility, or physical constraints of dynamic random-access memory.[8]

Interoperability [edit]

Arrow can be used with Apache Parquet, Apache Spark, NumPy, PySpark, pandas and other data processing libraries. The project includes native software libraries written in C, C++, C#, Go, Java, JavaScript, Julia, MATLAB, Python, R, Ruby, and Rust. Arrow allows for zero-copy reads and fast data access and interchange without serialization overhead between these languages and systems.[3]

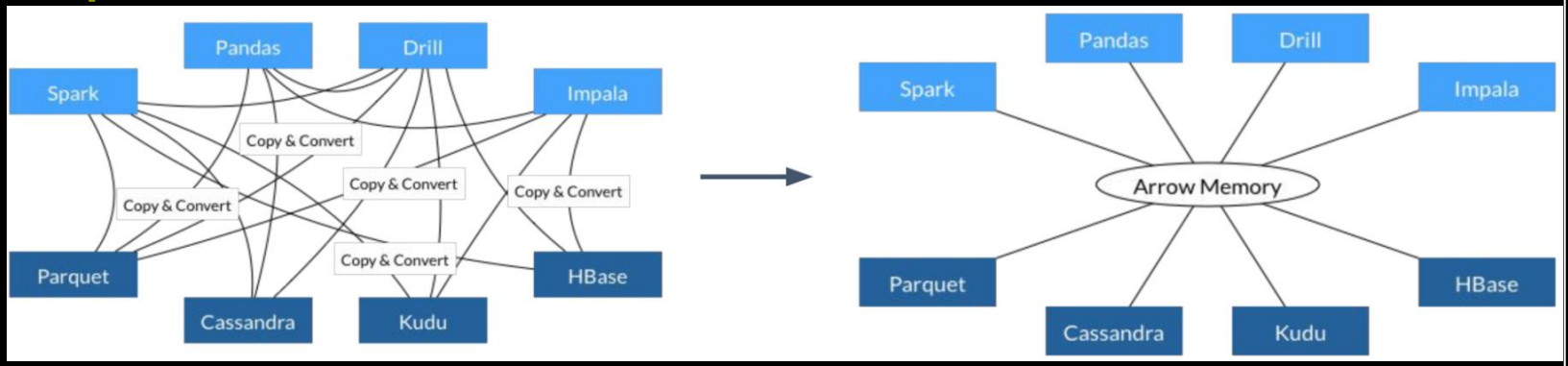
Applications [edit]

Arrow has been used in diverse domains, including analytics,[9] genomics,[10][8] and cloud computing.[11]

Comparison to Apache Parquet and ORC [edit]

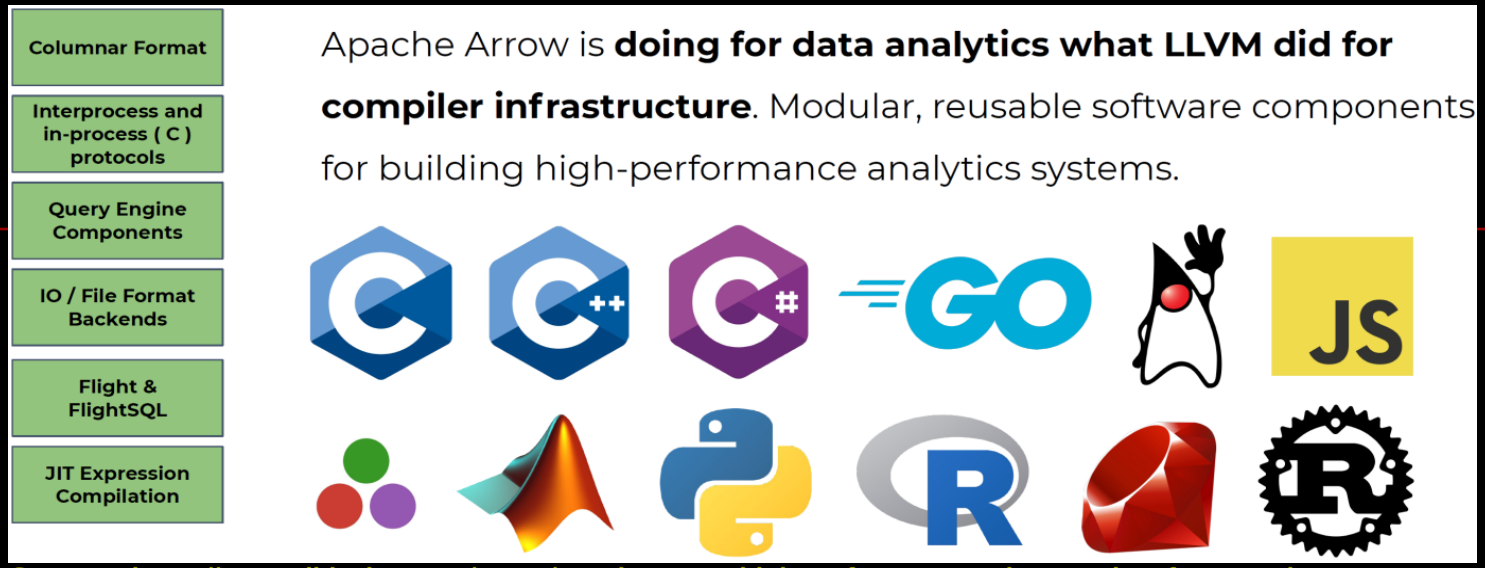
Apache Parquet and Apache ORC are popular examples of on-disk columnar data formats. Arrow is designed as a complement to these formats for processing data in-memory.[12] The hardware resource engineering trade-offs for in-memory processing vary from those associated with on-disk storage.[13] The Arrow and Parquet projects includes libraries that allow for reading and writing data between the two formats.[14]

- https://arrow.apache.org/
- Simplified data access with the Arrow columnar format



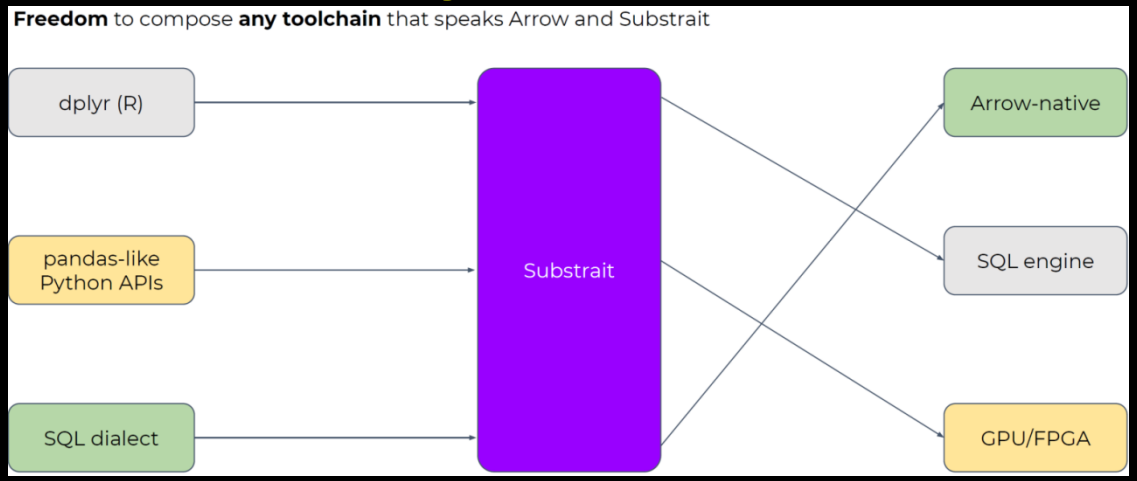
Source: https://www.slideshare.net/wesm/apache-arrow-high-performance-columnar-data-framework

- **Features**



Source: https://www.slideshare.net/wesm/apache-arrow-high-performance-columnar-data-framework

- **LEGO for data ecosystem**



Source: https://www.slideshare.net/wesm/apache-arrow-high-performance-columnar-data-framework

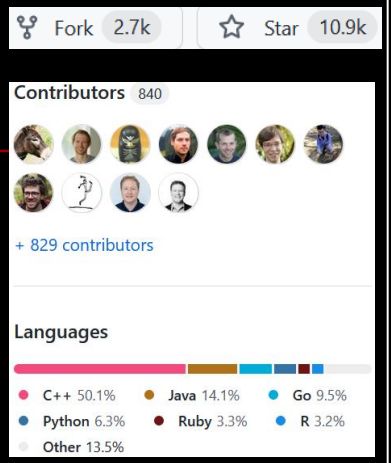
- ...

Src

- https://github.com/apache/arrow

Major components of the project include:

- The Arrow Columnar In-Memory Format: a standard and efficient in-memory representation of various datatypes, plain or nested
- The Arrow IPC Format: an efficient serialization of the Arrow format and associated metadata, for communication between processes and heterogeneous environments
- The Arrow Flight RPC protocol: based on the Arrow IPC format, a building block for remote services exchanging Arrow data with application-defined semantics (for example a storage server or a database)
- C++ libraries
- C bindings using GLib
- C# .NET libraries
- Gandiva: an LLVM-based Arrow expression compiler, part of the C++ codebase
- Go libraries
- Java libraries
- JavaScript libraries
- Plasma Object Store: a shared-memory blob store, part of the C++ codebase
- Python libraries
- R libraries
- Ruby libraries
- Rust libraries



1.3.3 Project Ray

1.3.3.1 What is it

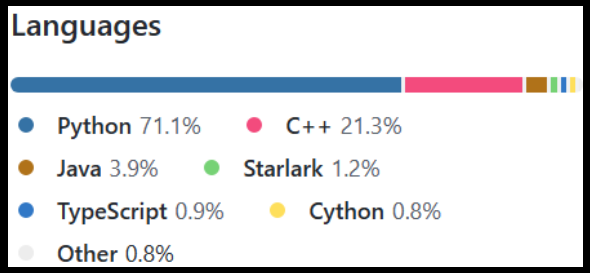
- **https://www.ray.io/**

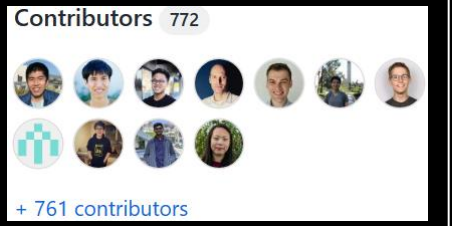
  Effortlessly scale your most complex workloads

  Ray is an open-source unified compute framework that makes it easy to scale AI and Python workloads — from reinforcement learning to deep learning to tuning, and model serving. Learn more about Ray's rich set of libraries and integrations.

- **https://github.com/ray-project/ray/**

  A unified framework for scaling AI and Python applications. Ray consists of a core distributed runtime and a toolkit of libraries (Ray AIR) for accelerating ML workloads.



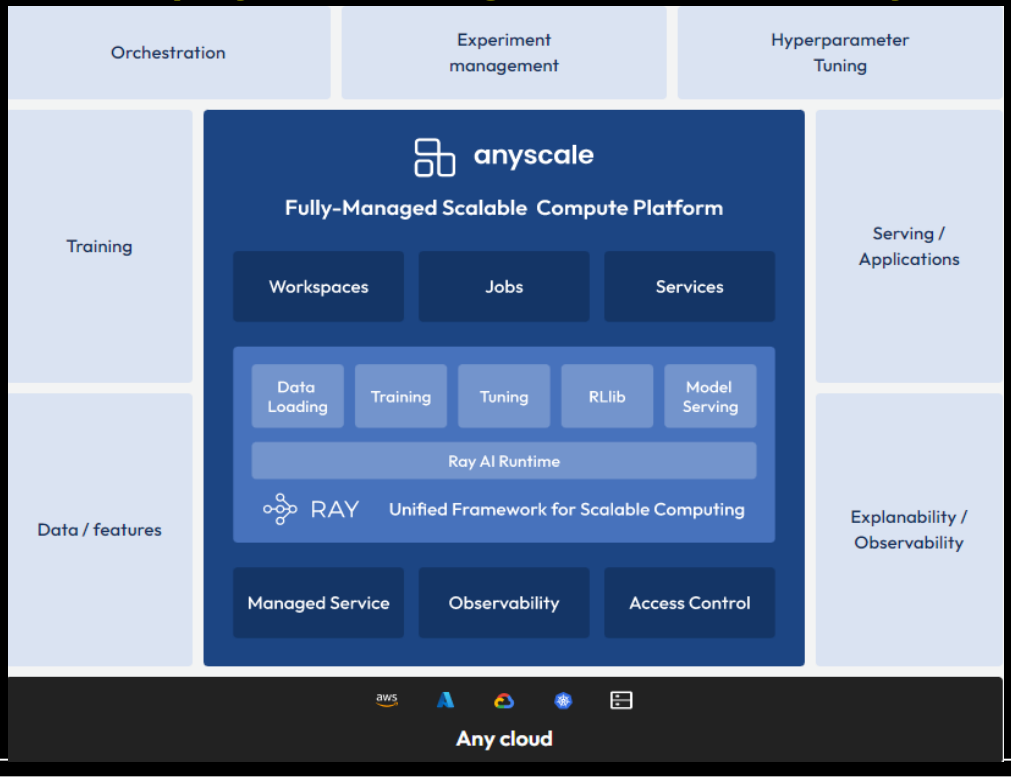


- **https://rise.cs.berkeley.edu/projects/ray/**

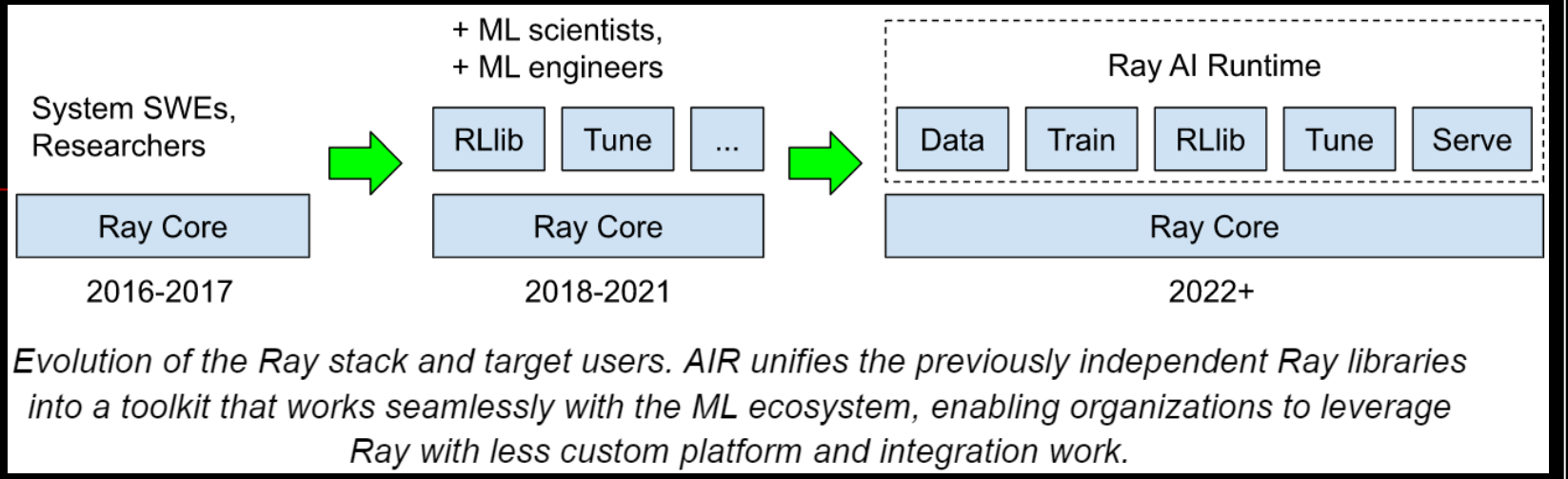
Ray is a high-performance distributed execution framework targeted at large-scale machine learning and reinforcement learning applications. It achieves scalability and fault tolerance by abstracting the control state of the system in a global control store and keeping all other components stateless. It uses a shared-memory distributed object store to efficiently handle large data through shared memory, and it uses a bottom-up hierarchical scheduling architecture to achieve low-latency and high-throughput scheduling. It uses a lightweight API based on dynamic task graphs and actors to express a wide range of applications in a flexible manner.

- **https://www.anyscale.com/**

Anyscale is a fully managed, unified compute platform that makes it easy to build, deploy, and manage scalable AI and Python applications on Ray.



History



Evolution of the Ray stack and target users. AIR unifies the previously independent Ray libraries into a toolkit that works seamlessly with the ML ecosystem, enabling organizations to leverage Ray with less custom platform and integration work.

Source: Ray AIR Technical Whitepaper

- **https://en.wikipedia.org/wiki/AMPLab**
- **https://news.berkeley.edu/story_jump/berkeley-launches-riselab-enabling-computers-to-make-intelligent-real-time-decisions/**
- **https://rise.cs.berkeley.edu/**

Architecture & design

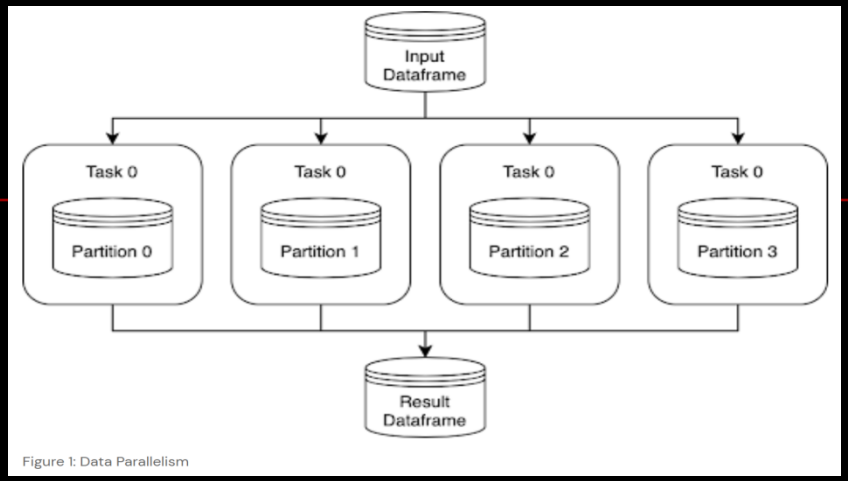


Figure 1: Data Parallelism

VS

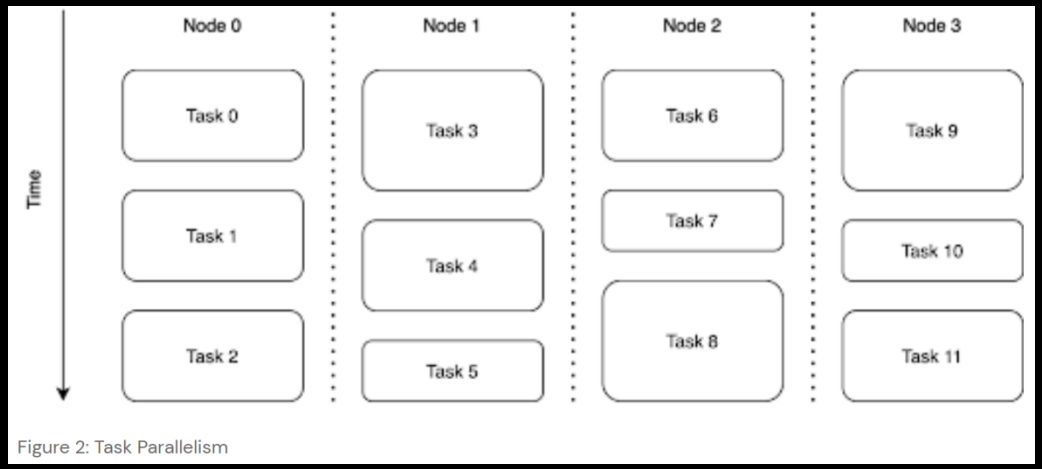
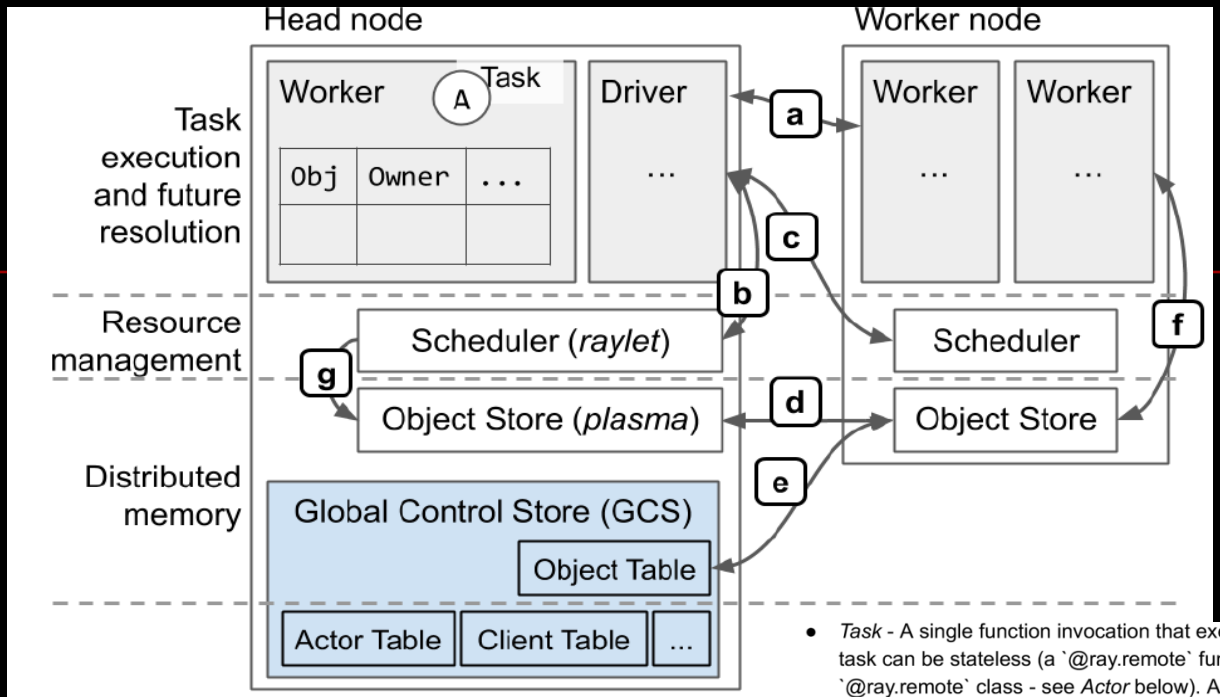


Figure 2: Task Parallelism

Source: https://www.databricks.com/blog/2021/11/19/ray-on-databricks.html



Architecture and protocols

Protocol overview (mostly over gRPC):

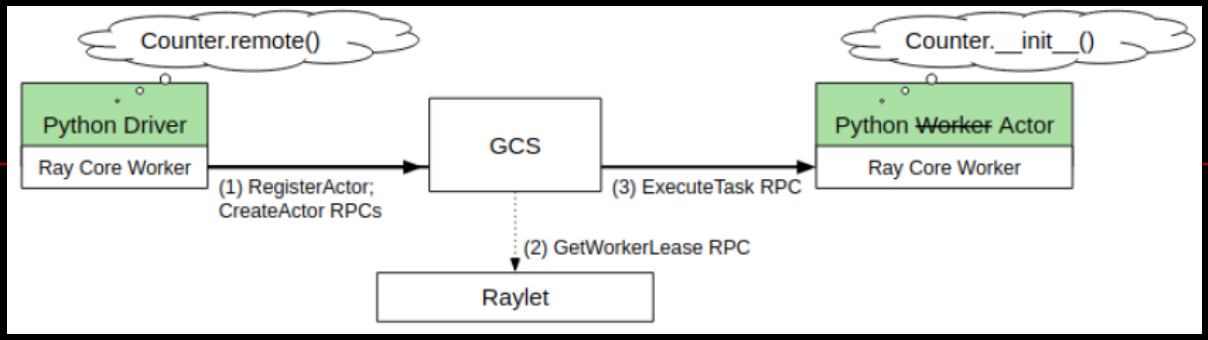
a. Task execution, object reference counting.
b. Local resource management.
c. Remote/distributed resource management.
d. Distributed object transfer.
e. Location lookup for objects stored in distributed object store.
f. Storage and retrieval of large objects. Retrieval is via `ray.get` or during task execution, when replacing a task's ObjectID argument with the object's value.
g. Scheduler fetches objects from remote nodes to fulfill dependencies of locally queued tasks.

- *Task* - A single function invocation that executes on a process different from the caller. A task can be stateless (a `@ray.remote` function) or stateful (a method of a `@ray.remote` class - see *Actor* below). A task is executed asynchronously with the caller: the `.remote()` call immediately returns an `ObjectRef` that can be used to retrieve the return value.
- *Object* - An application value. This may be returned by a task or created through `ray.put`. Objects are **immutable**: they cannot be modified once created. A worker can refer to an object using an `ObjectRef`.
- *Actor* - a stateful worker process (an instance of a `@ray.remote` class). Actor tasks must be submitted with a *handle*, or a Python reference to a specific instance of an actor.
- *Driver* - The program root. This is the code that runs `ray.init()`.
- *Job* - The collection of tasks, objects, and actors originating (recursively) from the same driver.

Source: Ray Architecture whitepaper 1.0

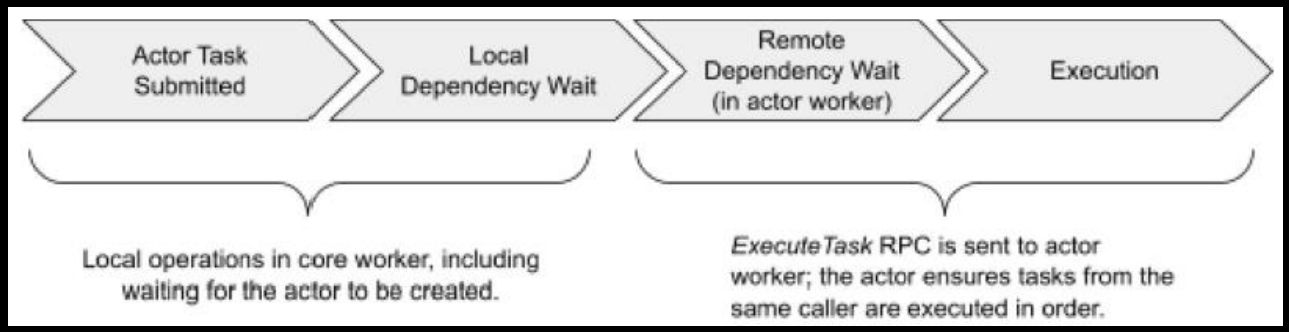
- Actor management

Actor creation



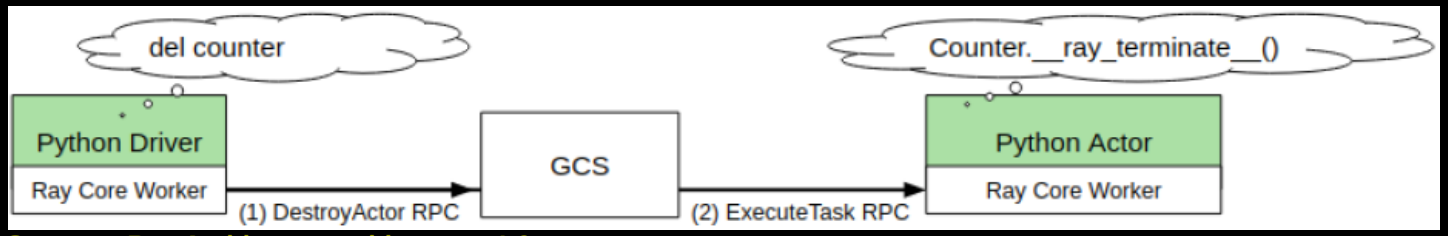
Source: Ray Architecture whitepaper 1.0

Actor task execution



Source: Ray Architecture whitepaper 1.0

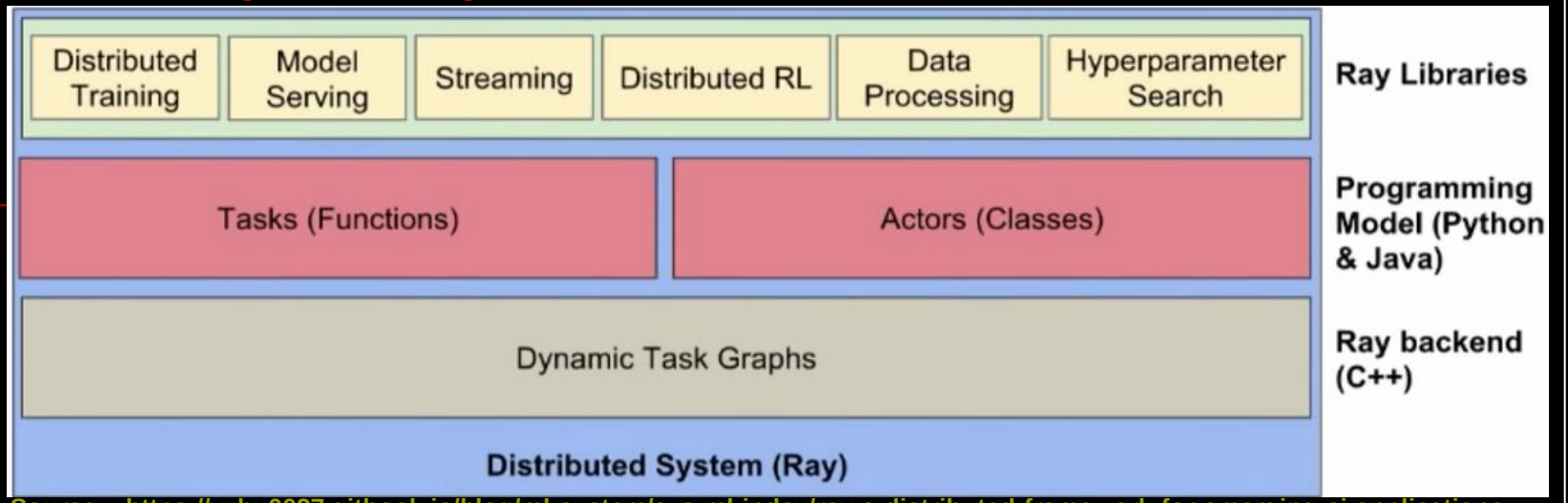
Actor death



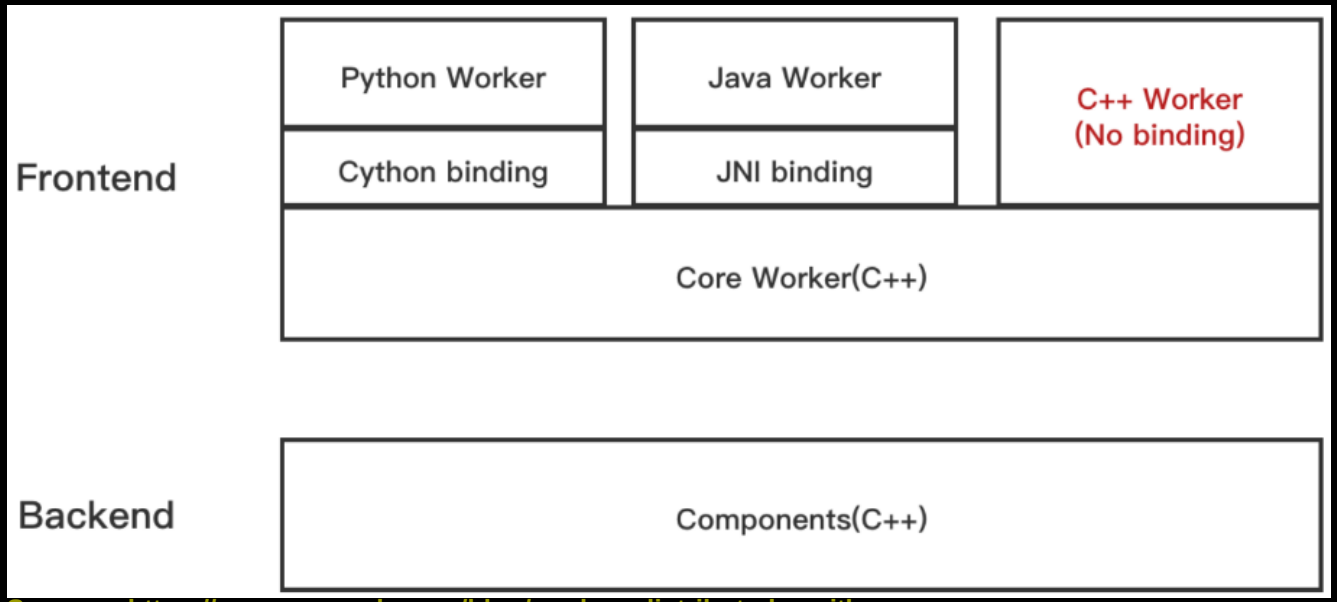
Source: Ray Architecture whitepaper 1.0

- Ray Design Patterns 1.0

Software layers of Ray



Source: https://xzhu0027.gitbook.io/blog/ml-system/sys-ml-index/ray-a-distributed-framework-for-emerging-ai-applications



Source: https://www.anyscale.com/blog/modern-distributed-c-with-ray

Ray Core in C++

```
[mydev@fedora ray-master]$ tree -d src/ray
src/ray
├── common
│   ├── asio
│   ├── ray_syncer
│   ├── task
│   └── test
├── core_worker
│   ├── lib
│   │   └── java
│   ├── store_provider
│   │   └── memory_store
│   ├── test
│   └── transport
├── design_docs
├── gcs
│   ├── gcs_client
│   │   └── test
│   ├── gcs_server
│   │   └── test
│   ├── pubsub
│   ├── store_client
│   │   └── test
│   └── test
├── internal
├── object_manager
│   ├── plasma
│   │   └── test
│   └── test
├── protobuf
├── pubsub
│   └── test
├── raylet
│   ├── format
│   ├── scheduling
│   │   └── policy
│   └── test
├── raylet_client
├── rpc
│   ├── agent_manager
│   ├── gcs_server
│   ├── node_manager
│   ├── object_manager
│   ├── runtime_env
│   ├── test
│   │   └── grpc_bench
│   └── worker
├── stats
├── thirdparty
└── util
```

```
[mydev@fedora ray-master]$ tokei src/ray
===============================================================================
 Language            Files        Lines         Code     Comments       Blanks
===============================================================================
 C                       3         6546         3828         2190          528
 C Header              233        44034        21320        16072         6642
 C++                   272        97857        75274        11561        11022
 Dockerfile              1            7            4            0            3
 FlatBuffers Schema      2          563          276          231           56
 LD Script               2           77           77            0            0
 Markdown                2          156            0          124           32
 Protocol Buffers       22         4718         2415         1743          560
 ReStructuredText        2          113           80            0           33
===============================================================================
 Total                 539       154071       103274        31921        18876
===============================================================================
```

Demo

- https://www.databricks.com/notebooks/raydemo.html

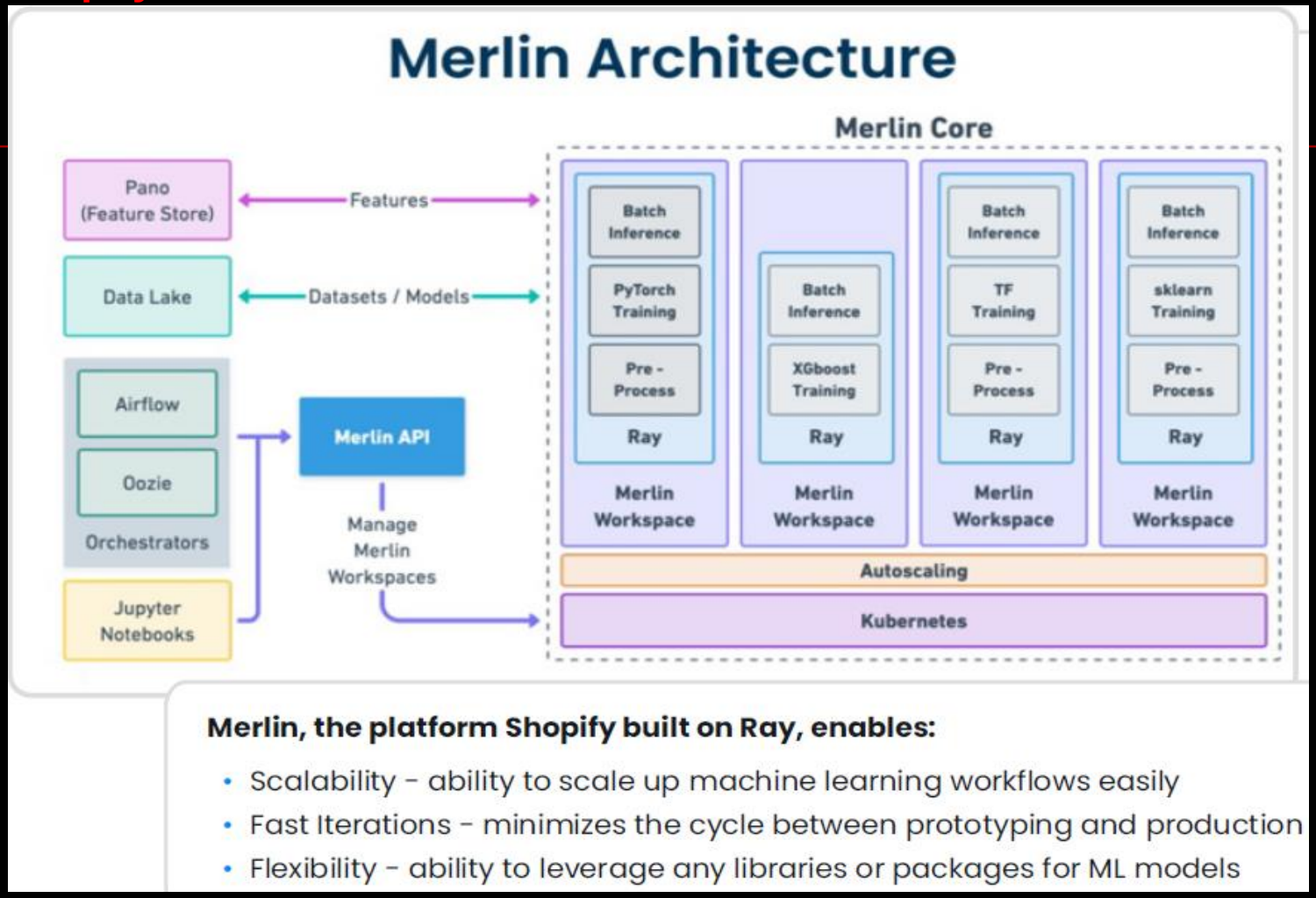
```
# First, decorate your function with @ray.remote to declare that you want to run this function remotely.
# Lastly, call that function with .remote() instead of calling it normally.
# This remote call yields a future, or ObjectRef that you can then fetch with ray.get.

@ray.remote
def f(x):
    return x * x

futures = [f.remote(i) for i in range(4)]
print(ray.get(futures)) # [0, 1, 4, 9]

[0, 1, 4, 9]
```

```
# Ray provides actors to allow you to parallelize an instance of a class in Python.
# When you instantiate a class that is a Ray actor, Ray will start a remote instance of that class in the cluster.
# This actor can then execute remote method calls and maintain its own internal state.

@ray.remote
class Counter(object):
    def __init__(self):
        self.n = 0

    def increment(self):
        self.n += 1

    def read(self):
        return self.n

counters = [Counter.remote() for i in range(4)]
[c.increment.remote() for c in counters]
futures = [c.read.remote() for c in counters]
print(ray.get(futures)) # [1, 1, 1, 1]

[1, 1, 1, 1]
```

...

Use Cases

- **Shopify Merlin**



Source: Ray OSS Datasheet.

- **https://docs.ray.io/en/latest/ray-overview/use-cases.html**

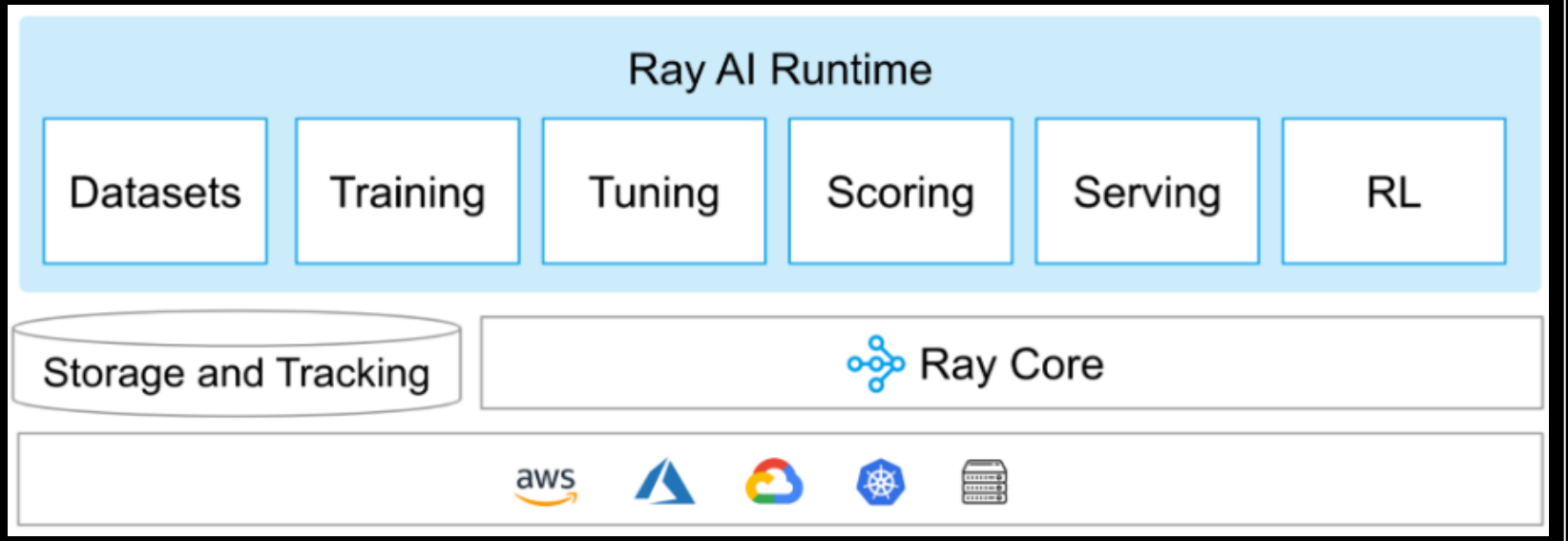
1.3.3.2 Ray 2.x

- https://www.anyscale.com/blog/announcing-ray-2-0

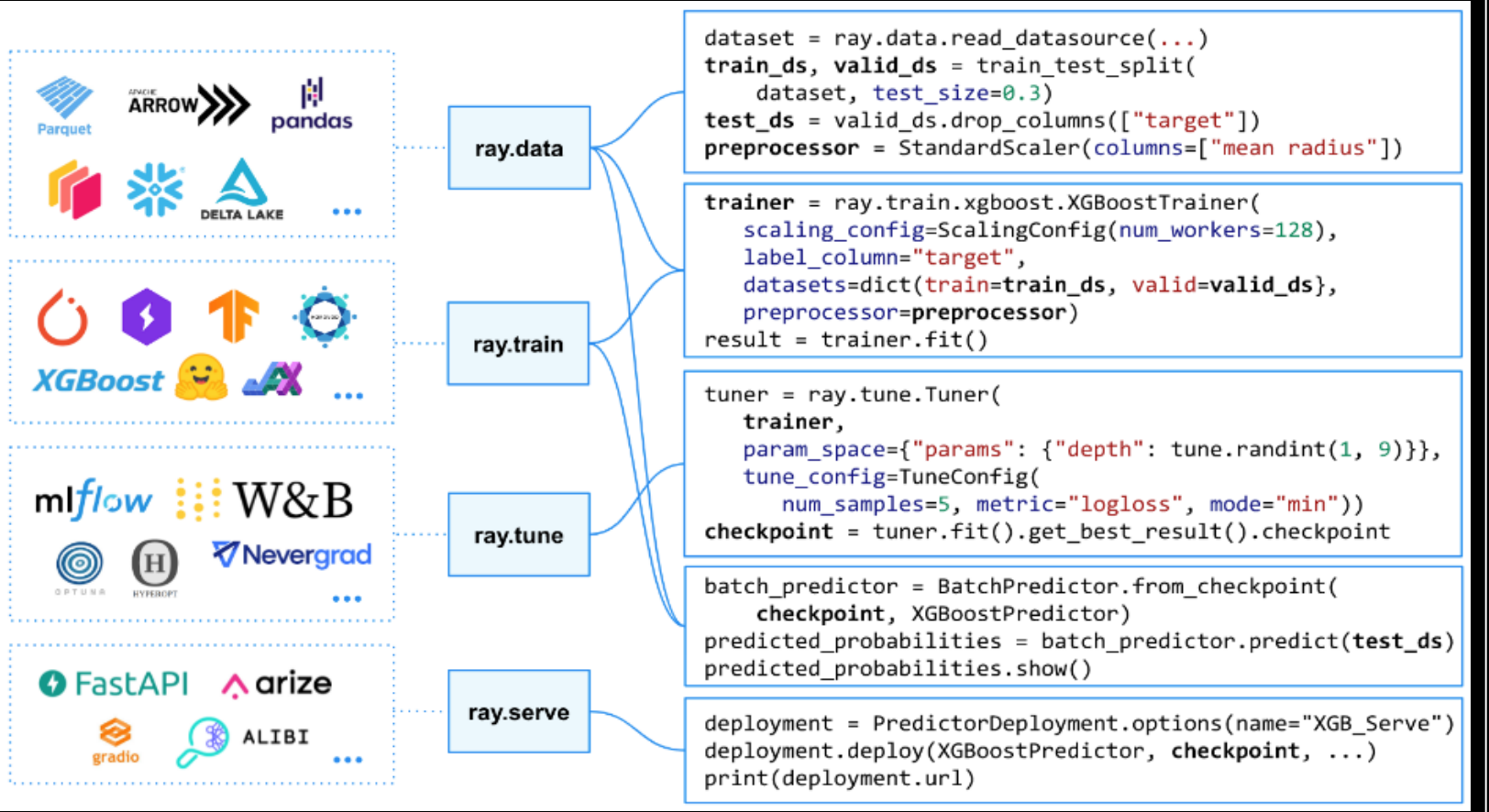
Unifying the ML Ecosystem

- Ray AI Runtime — a scalable and unified toolkit for ML applications

Since its inception, Ray has been used to accelerate and scale compute-heavy machine learning workloads. The Ray AI Runtime (AIR), released as beta, is an effort to incorporate and synthesize lessons learned: by talking to community users and taking into account their use cases both large and small. This deliberate effort has produced a simple, unified toolkit for the Ray community.



With Ray AIR, we're aligning Ray's existing native ML libraries to work seamlessly together and integrate easily with popular ML frameworks in the community. Our goal with Ray AIR is to make it easy to run any kind of ML workload in just a few lines of Python code, letting Ray do the heavy lifting of coordinating computations at scale.

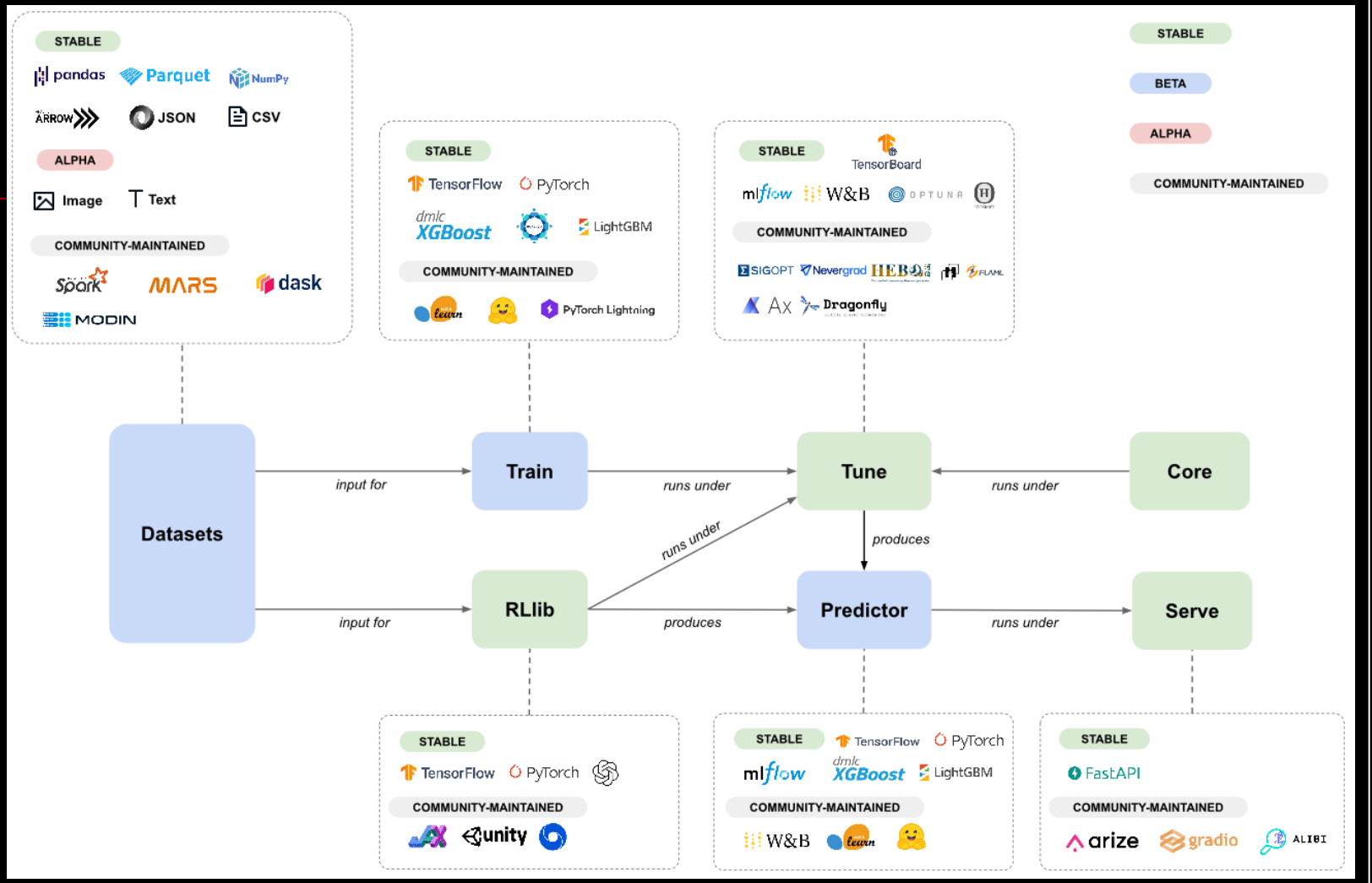


1.3.3.3 AIR Ecosystem

- https://docs.ray.io/en/master/ray-air/air-ecosystem.html#air-ecosystem-map

The following map visualizes the landscape and maturity of AIR components and their integrations. Solid lines denote integrations between AIR components; dotted lines denote integrations with the broader ML ecosystem.

- **Stable**: This component is stable.
- **Beta**: This component is under development and APIs may be subject to change.
- **Alpha**: This component is in early development.
- **Community-Maintained**: These integrations are community-maintained and may vary in quality.


STABLE
pandas
Parquet
NumPy
ARROW
JSON
CSV
ALPHA
Image
Text
COMMUNITY-MAINTAINED
Spark
MARS
dask
MODIN
STABLE
TensorFlow
PyTorch
dmlc XGBoost
LightGBM
COMMUNITY-MAINTAINED
PyTorch Lightning
STABLE
TensorBoard
mlflow
W&B
OPTUNA
COMMUNITY-MAINTAINED
SIGOPT
Nevergrad
HEBO
FLAML
Ax
Dragonfly
STABLE
BETA
ALPHA
COMMUNITY-MAINTAINED
Datasets
input for
Train
runs under
Tune
runs under
Core
runs under
produces
input for
RLlib
produces
Predictor
runs under
Serve
STABLE
TensorFlow
PyTorch
COMMUNITY-MAINTAINED
Unity
STABLE
TensorFlow
PyTorch
mlflow
dmlc XGBoost
LightGBM
COMMUNITY-MAINTAINED
W&B
STABLE
FastAPI
COMMUNITY-MAINTAINED
arize
gradio
ALIBI

1.3.3.4 Data Processing

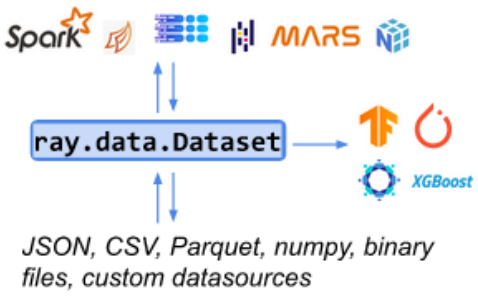
- https://docs.ray.io/en/master/data/dataset.html

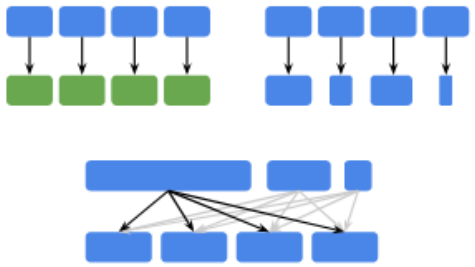
Ray Datasets: Distributed Data Preprocessing.

Ray Datasets are the standard way to load and exchange data in Ray libraries and applications. They provide basic distributed data transformations such as maps (`map_batches`), global and grouped aggregations (`GroupedDataset`), and shuffling operations (`random_shuffle`, `sort`, `repartition`), and are compatible with a variety of file formats, data sources, and distributed frameworks.

Here's an overview of the integrations with other processing frameworks, file formats, and supported operations, as well as a glimpse at the Ray Datasets API.

Check our compatibility matrix to see if your favorite format is already supported.





```
ds = ray.data.read_parquet(...)
# Pass datasets to tasks and actors
func.remote(ds)
# Split datasets into shards
shards = ds.split(xgboost.num_workers,
    locality_hints=xgboost.workers)
# Distributed training on datasets
for i, s in enumerate(shards):
   xgboost.workers[i].train.remote(s)
```

Standard way to load and exchange data in Ray

Basic distributed ops: map, filter, and repartition

Seamless interop with Ray tasks, actors, and libraries

- …

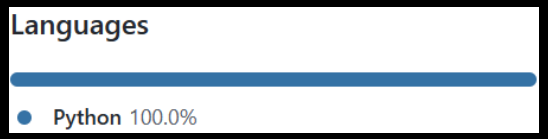
DeltaCAT

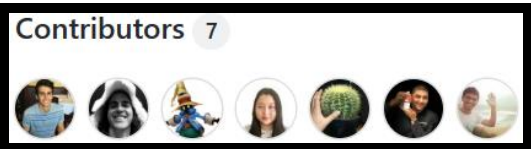
- https://github.com/ray-project/deltacat

DeltaCAT is a Pythonic Data Catalog powered by Ray.

Its data storage model allows you to define and manage fast, scalable, ACID-compliant data catalogs through git-like stage/commit APIs, and has been used to successfully host exabyte-scale enterprise data lakes.

DeltaCAT uses the Ray distributed compute framework together with Apache Arrow for common table management tasks, including petabyte-scale change-data-capture, data consistency checks, and table repair.



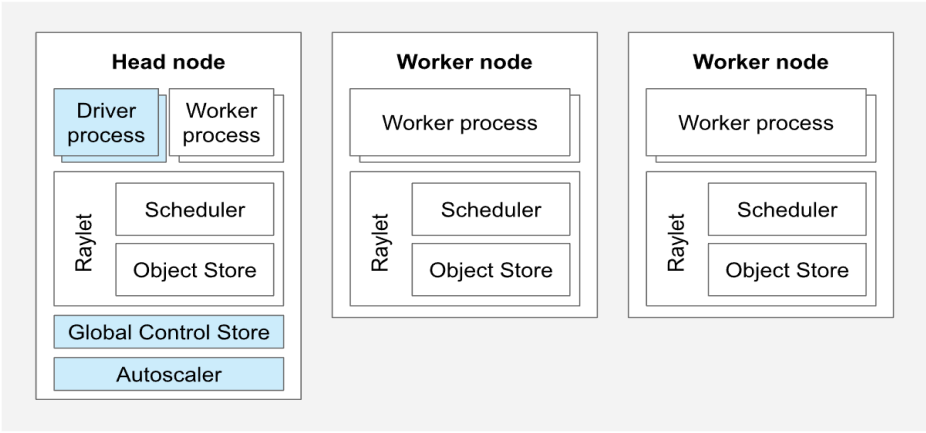


1.3.3.5 Clustering

Overview

- https://docs.ray.io/en/latest/cluster/vms/user-guides/launching-clusters/on-premises.html#manually-set-up-a-ray-cluster

A Ray cluster consists of a single head node and any number of connected worker nodes:



A Ray cluster with two worker nodes. Each node runs Ray helper processes to facilitate distributed scheduling and memory management. The head node runs additional control processes (highlighted in blue).

The number of worker nodes may be *autoscaled* with application demand as specified by your Ray cluster configuration. The head node runs the autoscaler.

Note

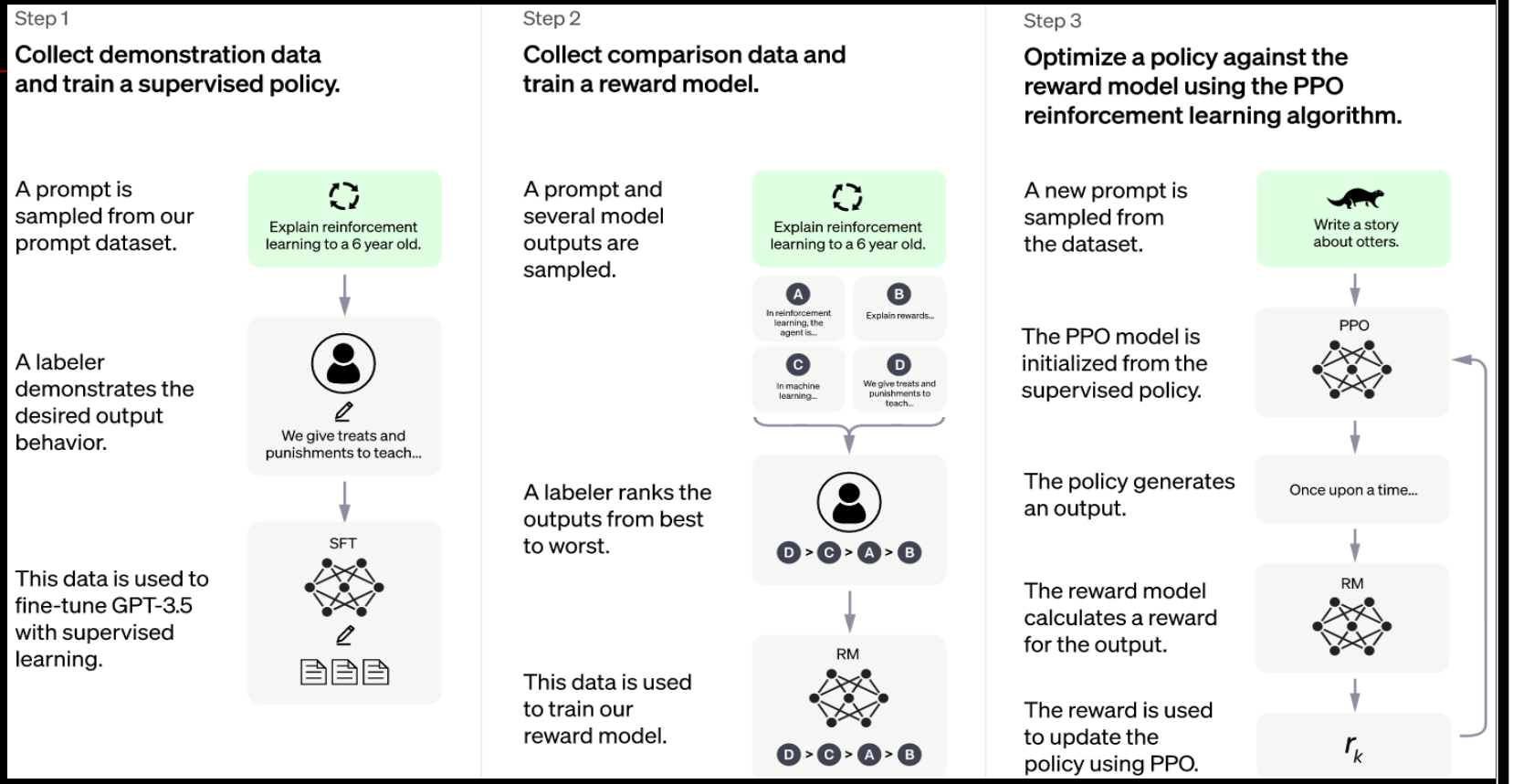
Ray nodes are implemented as pods when running on Kubernetes.

Users can submit jobs for execution on the Ray cluster, or can interactively use the cluster by connecting to the head node and running `ray.init`. See Ray Jobs for more information.

1.3.3.6 ChatGPT

Overview

- https://openai.com/blog/chatgpt/



- https://headtopics.com/us/how-openai-uses-a16z-backed-anyscale-s-ray-to-train-tools-like-chatgpt-32945289
- ...

1.3.3.7 Summary

- For more details, you may refer to our previous talk "Ray -- A Swiss Army Knife for Distributed Computing & AI" at COSCon 2022(Online).

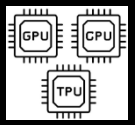
Pros & Cons of Ray from our point of view:

Pros

- A high-performance distributed execution framework targeted at large-scale ML and RL applications;
- A unified framework for scalable Distributed Computing;
- A fast-growing ecosystem;
- Python-first;
- …

Cons

- Mainly targets X86, and is not well-tested on ARM and RISC-V;
- Focus on Cloud-side, while its potential for Edge-side need to be further released;
- Currently can only be accelerated by GPU, and is not all the XPU-aware;

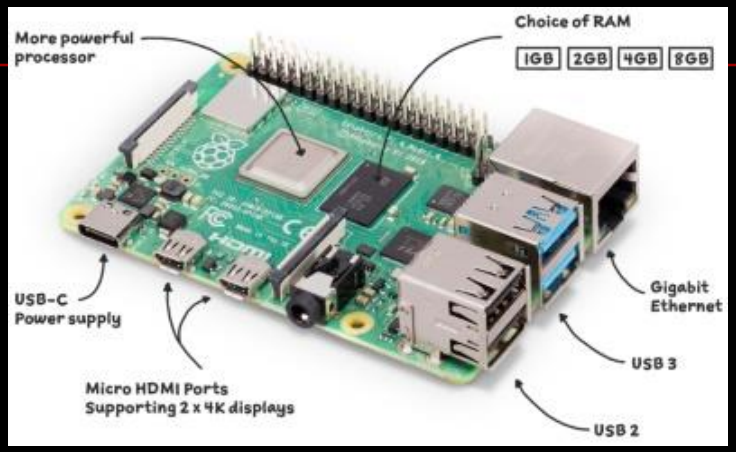


- A lightweight solution when compared with Spark-based projects, but is getting heavier;
- …

2) Testbed

2.1 Raspberry Pi 4(8GB LPDDR4) with Fedora 37

- HW env



SW env

```
[mydev@fedora /]$ uname -a
Linux fedora 6.0.18-300.fc37.aarch64 #1 SMP PREEMPT_DYNAMIC Sat Jan 7 16:48:26 UTC 2023 aarch64 aarch64 aarch64 GNU/Linux
[mydev@fedora /]$
[mydev@fedora /]$ free -m
               total        used        free      shared  buff/cache   available
Mem:            7826         478        3749          16        3597        7072
Swap:           7825           0        7825
[mydev@fedora /]$
```

```
[mydev@fedora /]$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/aarch64-redhat-linux/12/lto-wrapper
Target: aarch64-redhat-linux
Configured with: ../configure --enable-bootstrap --enable-languages=c,c++,fortran,objc,obj-c++,ada,go,d,lto --prefix=/usr -
mandir=/usr/share/man --infodir=/usr/share/info --with-bugurl=http://bugzilla.redhat.com/bugzilla --enable-shared --enable-
hreads=posix --enable-checking=release --enable-multilib --with-system-zlib --enable-__cxa_atexit --disable-libunwind-excep
ions --enable-gnu-unique-object --enable-linker-build-id --with-gcc-major-version-only --enable-libstdcxx-backtrace --with-
inker-hash-style=gnu --enable-plugin --enable-initfini-array --with-isl=/builddir/build/BUILD/gcc-12.2.1-20221121/obj-aarch
4-redhat-linux/isl-install --enable-gnu-indirect-function --build=aarch64-redhat-linux --with-build-config=bootstrap-lto --
nable-link-serialization=1
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 12.2.1 20221121 (Red Hat 12.2.1-4) (GCC)
[mydev@fedora /]$
[mydev@fedora /]$ clang -v
clang version 15.0.6 (Fedora 15.0.6-3.fc37)
Target: aarch64-redhat-linux-gnu
Thread model: posix
InstalledDir: /usr/bin
Found candidate GCC installation: /usr/bin/../lib/gcc/aarch64-redhat-linux/12
Selected GCC installation: /usr/bin/../lib/gcc/aarch64-redhat-linux/12
Candidate multilib: .;@m64
Selected multilib: .;@m64
[mydev@fedora /]$
```

```
[mydev@fedora /]$ python -V
Python 3.11.1
[mydev@fedora /]$
```

```
[mydev@fedora /]$ graalpy -V
GraalPy 3.10.8 (GraalVM CE Native 23.0.0-dev)
[mydev@fedora /]$
```

```
[mydev@fedora /]$ java --version
openjdk 19.0.1 2022-10-18
OpenJDK Runtime Environment GraalVM CE 23.0.0-dev (build 19.0.1+10-jvmci-23.0-b04)
OpenJDK 64-Bit Server VM GraalVM CE 23.0.0-dev (build 19.0.1+10-jvmci-23.0-b04, mixed mode, sharing)
[mydev@fedora /]$
[mydev@fedora /]$ gu list
ComponentId              Version             Component name                Stability
------------------------------------------------------------------------------------------
------
graalvm                  23.0.0-dev          GraalVM Core                  Experimental
js                       23.0.0-dev          Graal.js                      Experimental
llvm                     23.0.0-dev          LLVM Runtime Core             Experimental
llvm-toolchain           23.0.0-dev          LLVM.org toolchain            Experimental
native-image             23.0.0-dev          Native Image                  Experimental
native-image-llvm-backend23.0.0-dev          Native Image LLVM Backend     Experimental
nodejs                   23.0.0-dev          Graal.nodejs                  Experimental
python                   23.0.0-dev          GraalVM Python                Experimental
visualvm                 23.0.0-dev          VisualVM                      Experimental
wasm                     23.0.0-dev          GraalWasm                     Experimental
[mydev@fedora /]$
```

```
[mydev@fedora .ivy2]$ sbt -V
downloading sbt launcher 1.8.0
[info] [launcher] getting org.scala-sbt sbt 1.8.0  (this may take some time)...
[info] [launcher] getting Scala 2.12.17 (for sbt)...
sbt version in this project: 1.8.0
sbt script version: 1.8.0
[mydev@fedora .ivy2]$
[mydev@fedora .ivy2]$ scalac -version
Scala compiler version 2.13.10 -- Copyright 2002-2021, LAMP/EPFL and Lightbend, Inc.
[mydev@fedora .ivy2]$
```

2.1.1 Fedora

- A Linux distribution developed by the community-supported Fedora Project which is sponsored primarily by Red Hat.
- https://en.wikipedia.org/wiki/Fedora_(operating_system)
- https://getfedora.org/
- https://alt.fedoraproject.org/alt/
- https://spins.fedoraproject.org/
- https://fedoraproject.org/wiki/Architectures/ARM
- https://fedoramagazine.org/
- https://silverblue.fedoraproject.org/
- https://fedoraproject.org/wiki/Changes/RaspberryPi4
- Developer friendly!

2.1.1.2 Fedora 37

- https://fedoraproject.org/wiki/Releases/37/ChangeSet
- https://www.phoronix.com/news/Fedora-37-Released

Raspberry Pi 4

- **Fedora 37 To Offer Official Support On Raspberry Pi 4 Devices**
  https://www.phoronix.com/news/Raspberry-Pi-4-Fedora-37

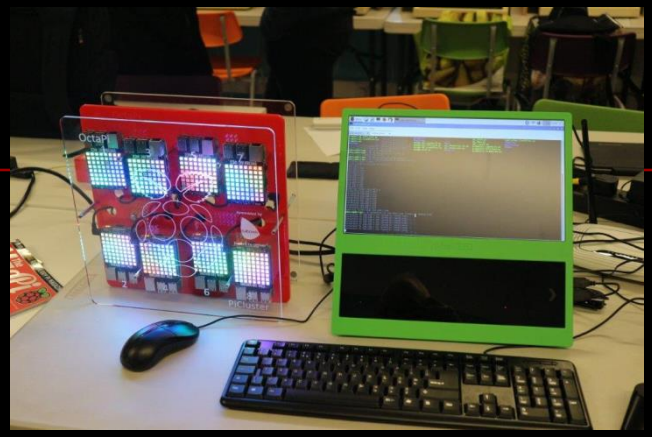
A month ago there was the Fedora 37 change proposal for Fedora to officially support the Raspberry Pi 4, including its accelerated Broadcom graphics and to better advertise Fedora for the Raspberry Pi. The Fedora Engineering and Steering Committee (FESCo) has now signed off on this "official" support for the Raspberry Pi 4.

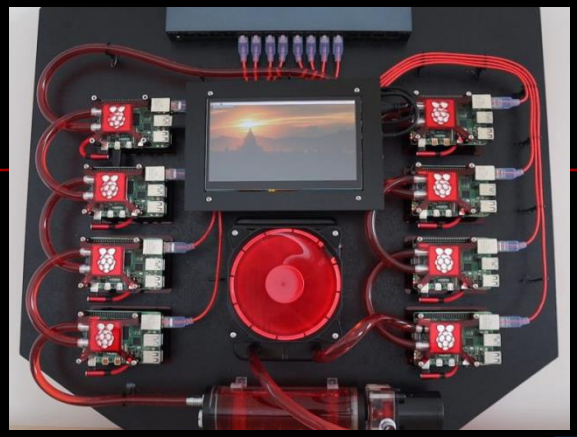
The Raspberry Pi 4 to date hasn't been a significant focus for Fedora Workstation due to various patches not being upstreamed-- most notably, waiting on the open-source 3D graphics bits to be upstreamed in the kernel. Now though that those upstream bits are coming together, Fedora 37 will be focusing on advertising its support for the Raspberry Pi 4 Model B as well as the Raspberry Pi 400 and Raspberry Pi CM4 compute module.

With the open-source OpenGL and Vulkan support, these latest Raspberry Pi boards are suitable for Fedora Workstation usage. The latest milestones there are V3DV just having crossed Vulkan 1.2 and Raspberry Pi 4 V3D DRM/KMS driver support in Linux 6.0. However, there still are upstream issues surrounding WiFi support for the Raspberry Pi 400, the Raspberry Pi CM4 not necessarily being very suitable for desktop use but more edge/IoT/embedded, and some device support such as around audio may be problematic.

> ***The Raspberry Pi 4 is a widely available, reasonably priced device. It has worked well in Fedora for some time in IoT and Server use cases, and now with a fully accelerated graphics stack available it's a great device from a price-per-performance perspective, and it has a wide ecosystem, so fully supporting this in Fedora makes a compelling case.***

2.1.2 RPi Clusters





Pico 20 Raspberry PI4 8GB

~~$ 758.00~~ $ 689.00 Sale

II. Project Almond on ARM

1) Overview

- **https://almond.sh/**

  A Scala kernel for Jupyter.

- **Features**

  - all the Ammonite niceties,
  - an API that libraries can rely on to interact with Jupyter front-ends,
  - extensible plotting support,
  - extensible support for big data libraries, with in particular
  - Spark support, relying on ammonite-spark, extended to get progress bars among others.

  It also provides libraries allowing one to write custom Jupyter kernels in Scala.

  Source: https://almond.sh/docs/intro

All the Ammonite niceties

Ammonite is a modern and user-friendly Scala shell. Almond wraps it in a Jupyter kernel, giving you all its features and niceties, including customizable pretty-printing, magic imports, advanced dependency handling, its API, right from Jupyter.

This also makes it easy to copy some code from notebooks to Ammonite scripts, and vice versa.

```
[25] import $ivy.`io.circe::circe-generic:0.11.1`
     import io.circe._, io.circe.generic.auto._, io.circe.syntax._

     import $ivy.$
     import io.circe._, io.circe.generic.auto._, io.circe.syntax._

[26] case class Message(content: String, version: Int = 2)

     defined class Message

[27] val json = Message("Hello").asJson

     json: Json = JObject(object[content -> "Hello",version -> 2])

[ ]
```

```
[8]  def progress(pct: Int) =
       s"""
         <div class="progress">
           <div class="progress-bar" style="height: 1.3em; background-c
           </div>
         </div>
       """

     defined function progress

[9]  val handle = html(progress(0))

     for (n <- 1 to 100 by 2) {
       Thread.sleep(20L)
       handle.update(progress(n))
     }

     handle: almond.api.helpers.Display = text/html #57c58a9a-bcaa-44fc-
     8a97-7c83c324493d

[ ]
```

APIs to interact with Jupyter front-ends

Almond exposes APIs to interact with Jupyter front-ends. Call them from notebooks... or from your own libraries.

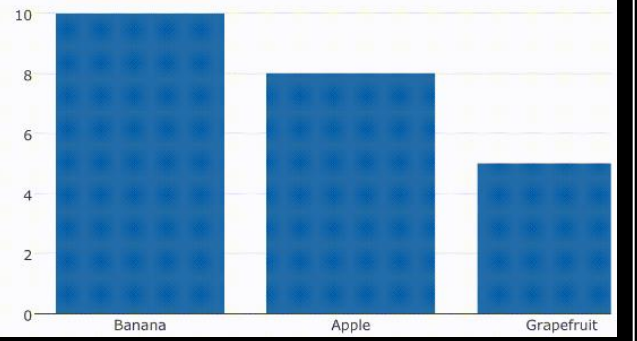
Source: https://almond.sh/

Plotting libraries

Several plotting libraries are already available to plot things from notebooks, such as plotly-scala or Vegas.

```
[12] import $ivy.`org.plotly-scala::plotly-almond:0.5.2`

     import plotly._, plotly.element._, plotly.layout._
     import plotly.Almond._

     import $ivy.$

     import plotly._, plotly.element._, plotly.layout._

     import plotly.Almond._
```

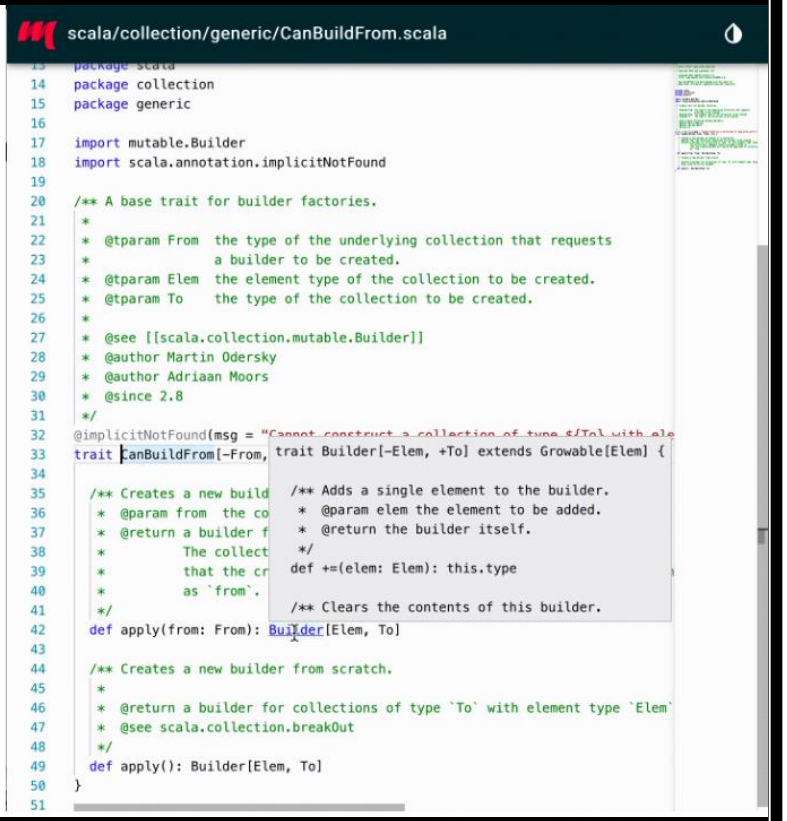
```
[13] Bar(Seq("Banana", "Apple", "Grapefruit"), Seq(10, 8, 5)).plot()
```



Source: https://almond.sh/

IDE-like features

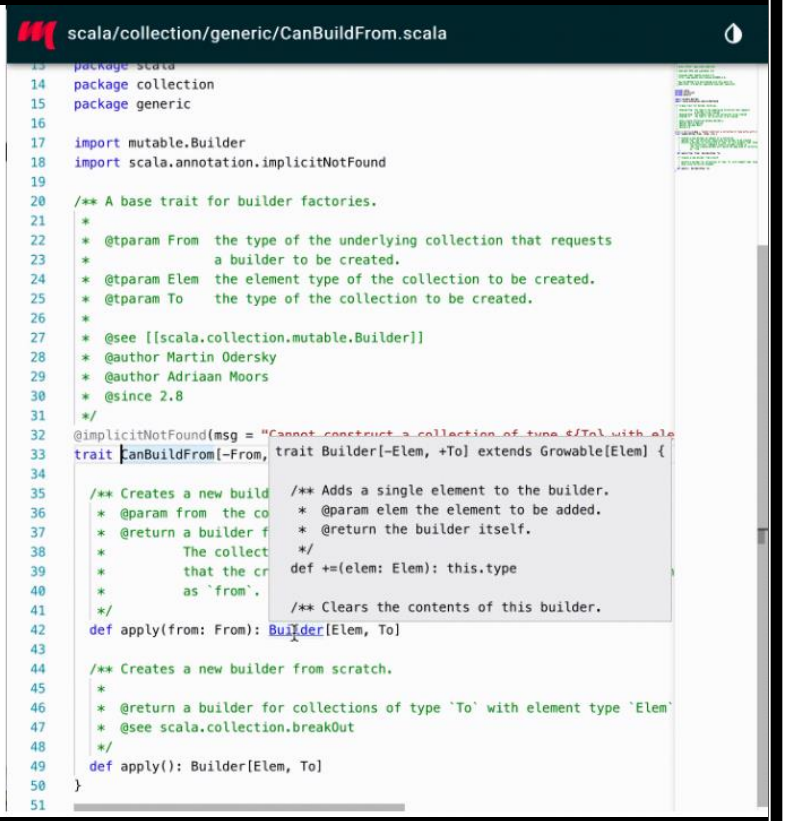
Almond already supports code navigation in dependencies via metabrowse, paving the way for more IDE-like features and a closer integration with the scalameta ecosystem.



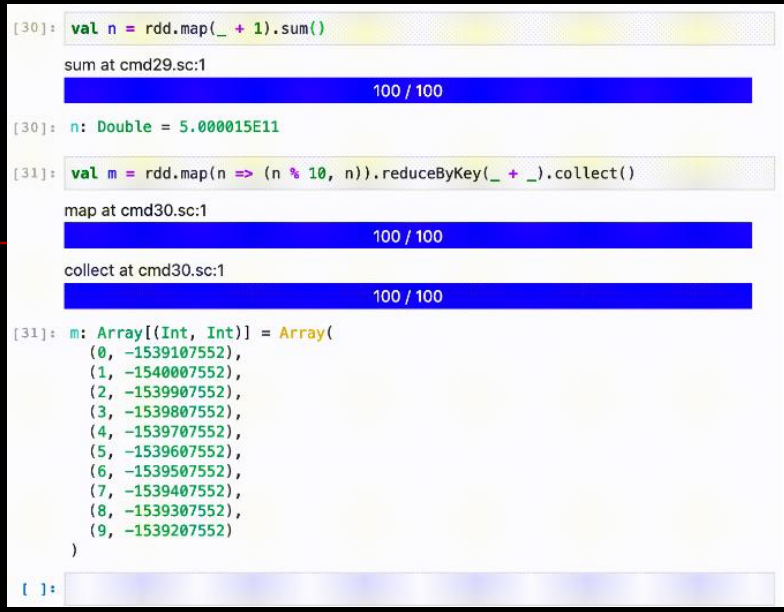
Source: https://almond.sh/

IDE-like features

Almond already supports code navigation in dependencies via metabrowse, paving the way for more IDE-like features and a closer integration with the scalameta ecosystem.

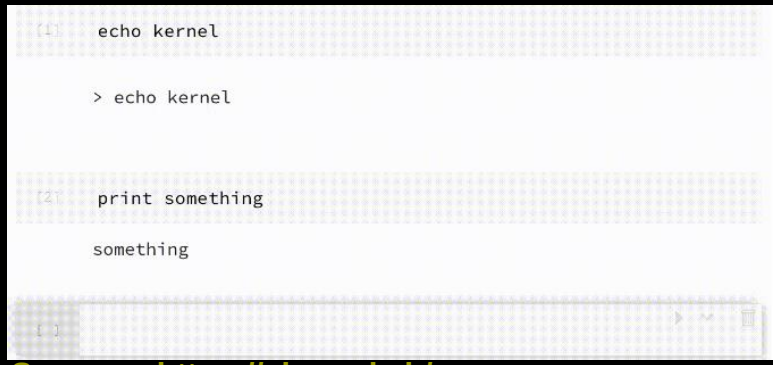


Source: https://almond.sh/



Spark support

Load the spark version of your choice, create a Spark session, and start using it from your notebooks.



Custom kernels

Write Jupyter kernels for the language of your choice, in Scala, by relying on the exact same libraries as Almond.

Source: https://almond.sh/

2) RPi4
2.1 Installation
2.1.1 Installing from sources

Official Guide

- **https://almond.sh/docs/dev-from-sources**

...

Almond is built with mill. A mill launcher ships with almond, so that you don't need to install mill yourself. We list below useful commands, to help get you started using mill to build almond.

```
[mydev@fedora almond-main]$ tree -L 1
.
├── build.sc
├── CONTRIBUTING.md
├── Dockerfile
├── docs
├── examples
├── LICENSE
├── mill
├── mill.98e8pj
├── mill.bat
├── modules
├── project
├── README.md
└── scripts
```

Our test:

```
[mydev@fedora almond-main]$ ./mill -i jupyter
```

...

```
https://repo1.maven.org/maven2/org/apache/commons/commons-compress/1.21/commons-compress-1.21.jar
  100.0% [##########] 994.4 KiB (4.6 KiB / s)
https://repo1.maven.org/maven2/com/google/guava/guava/21.0/guava-21.0-sources.jar
  100.0% [##########] 1.4 MiB (4.6 KiB / s)
Compiling /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/project/settings.sc
Compiling /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/build.sc
[35/549] shared.interpreter-api[2.13.10].scalacOptions.overridden.ammonite.$file.project.settings.AlmondScala2Or3Module.sc
[40/549] shared.interpreter-api[2.13.10].compile
Compiling compiler interface ...
[info] compiling 4 Scala sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/shared/int
rpreter-api/2.13.10/compile.dest/classes  ...
[info] done compiling
[71/549] shared.logger-scala2-macros[2.13.10].scalacOptions.overridden.ammonite.$file.project.settings.AlmondScala2Or3Modu
[76/549] shared.logger-scala2-macros[2.13.10].compile
[info] compiling 1 Scala source to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/shared/logg
r-scala2-macros/2.13.10/compile.dest/classes  ...
[info] done compiling
```

...

```
[warn] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/modules/shared/interpreter/src/main/scala/al
mond/interpreter/util/CancellableFuturePool.scala:70:11: method stop in class Thread is deprecated
[warn]         t.stop()
[warn]           ^
[warn] one warning found
[info] done compiling
[237/549] shared.kernel[2.13.10].scalacOptions.overridden.ammonite.$file.project.settings.AlmondScala2Or3Module.scalacOptio
[242/549] shared.kernel[2.13.10].compile
[info] compiling 8 Scala sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/shared/kern
el/2.13.10/compile.dest/classes  ...
[info] done compiling
[272/549] scala.jupyter-api[2.13.10].scalacOptions.overridden.ammonite.$file.project.settings.AlmondScala2Or3Module.scalacO
[277/549] scala.jupyter-api[2.13.10].compile
[info] compiling 19 Scala sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/scala/jupy
ter-api/2.13.10/compile.dest/classes  ...
[warn] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/modules/scala/jupyter-api/src/main/scala/alm
ond/api/JupyterApi.scala:48:30: object JavaConverters in package collection is deprecated (since 2.13.0): Use `scala.jdk.Co
llectionConverters` instead
[warn]         def display(t: T) = f(t).asJava
[warn]                             ^
[warn] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/modules/scala/jupyter-api/src/main/scala/alm
ond/display/Display.scala:33:17: object JavaConverters in package collection is deprecated (since 2.13.0): Use `scala.jdk.C
ollectionConverters` instead
[warn]           d.data().asJava
[warn]                   ^
[warn] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/modules/scala/jupyter-api/src/main/scala/alm
ond/display/UpdatableDisplay.scala:14:32: method mapValues in trait MapOps is deprecated (since 2.13.0): Use .view.mapValue
s(f). A future version will include a strict version of this method (for now, .view.mapValues(f).toMap).
[warn]     data.copy(data = data.data.mapValues(_ => "").toMap)
[warn]                                ^
[warn] three warnings found
[info] done compiling
```

```
...
[377/549] scala.scala-kernel[2.13.10].scalacOptions.overridden.ammonite.$file.project.settings.AlmondScala2Or3Module.scala
[382/549] scala.scala-kernel[2.13.10].compile
[info] compiling 3 Scala sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/scala/scal
-kernel/2.13.10/compile.dest/classes ...
[info] done compiling
[386/549] scala.scala-kernel-api[2.13.10].artifactName.overridden.ammonite.$file.project.settings.AlmondArtifactName.artif
[396/549] shared.interpreter-api[2.13.10].artifactName.overridden.ammonite.$file.project.settings.AlmondArtifactName.artif
[408/549] scala.jupyter-api[2.13.10].artifactName.overridden.ammonite.$file.project.settings.AlmondArtifactName.artifactNa
[434/549] scala.scala-kernel[2.13.10].artifactName.overridden.ammonite.$file.project.settings.AlmondArtifactName.artifactN
[454/549] shared.interpreter[2.13.10].artifactName.overridden.ammonite.$file.project.settings.AlmondArtifactName.artifactN
[498/549] shared.logger-scala2-macros[2.13.10].artifactName.overridden.ammonite.$file.project.settings.AlmondArtifactName.
[515/549] scala.scala-interpreter[2.13.10].artifactName.overridden.ammonite.$file.project.settings.AlmondArtifactName.arti
[549/549] jupyter0
JUPYTER_PATH=/opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/jupyter0.dest/jupyter
Traceback (most recent call last):
  File "/home/mydev/.local/bin/jupyter-lab", line 5, in <module>
    from jupyterlab.labapp import main
ModuleNotFoundError: No module named 'jupyterlab'
Jupyter command exited with code 1
[mydev@fedora almond-main]$
```

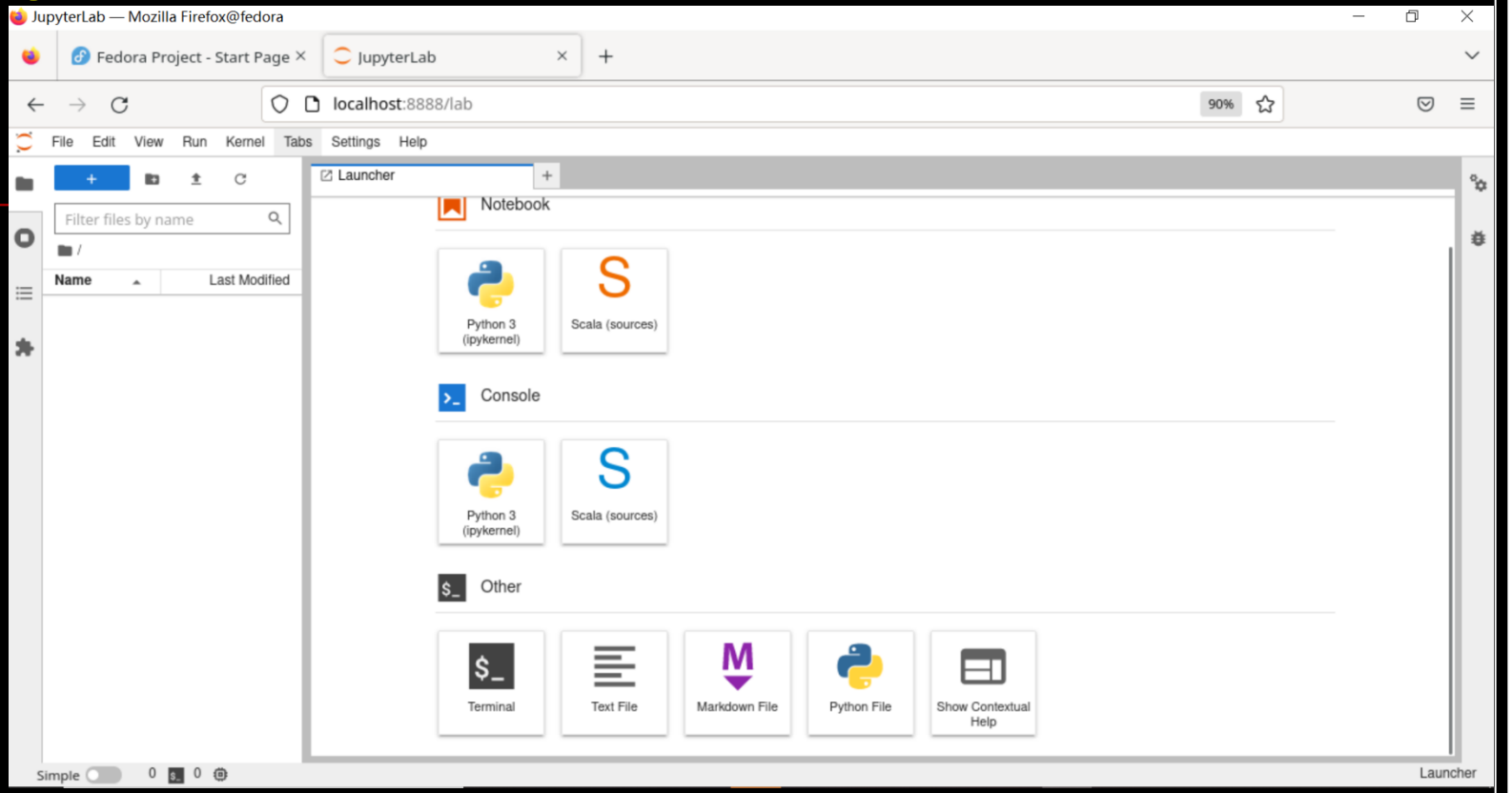
install juyterlab

```
[mydev@fedora /]$ pip install --user jupyterlab
Collecting jupyterlab
  Downloading jupyterlab-3.5.2-py3-none-any.whl (8.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.8/8.8 MB 159.1 kB/s eta 0:00:00
Requirement already satisfied: ipython in /usr/lib/python3.11/site-packages (from jupyterlab) (8.5.0)
Requirement already satisfied: packaging in /usr/lib/python3.11/site-packages (from jupyterlab) (21.3)
Requirement already satisfied: tornado≥6.1.0 in /usr/lib64/python3.11/site-packages (from jupyterlab) (6.2)
Requirement already satisfied: jupyter-core in /usr/lib/python3.11/site-packages (from jupyterlab) (4.10.0)
Collecting jupyterlab-server~=2.10
  Downloading jupyterlab_server-2.18.0-py3-none-any.whl (56 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.4/56.4 kB 152.0 kB/s eta 0:00:00
Collecting jupyter-server<3,≥1.16.0
  Downloading jupyter_server-2.0.6-py3-none-any.whl (363 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 364.0/364.0 kB 120.3 kB/s eta 0:00:00
Collecting nbclassic
  Using cached nbclassic-0.4.8-py3-none-any.whl (9.8 MB)
Requirement already satisfied: notebook<7 in /usr/lib/python3.11/site-packages (from jupyterlab) (6.4.12)
Requirement already satisfied: jinja2≥2.1 in /home/mydev/.local/lib/python3.11/site-packages (from jupyterlab) (3.1.2)
Requirement already satisfied: tomli in /usr/lib/python3.11/site-packages (from jupyterlab) (2.0.1)
Requirement already satisfied: MarkupSafe≥2.0 in /usr/lib64/python3.11/site-packages (from jinja2≥2.1→jupyterlab) (2.1.
)
Collecting anyio<4,≥3.1.0
  Using cached anyio-3.6.2-py3-none-any.whl (80 kB)
Requirement already satisfied: argon2-cffi in /usr/lib64/python3.11/site-packages (from jupyter-server<3,≥1.16.0→jupyter
ab) (21.1.0)
...
```

Retest:

```
[mydev@fedora almond-main]$ ./mill -i jupyter
[297/549] scala.scala-kernel-api[2.13.10].resources.overridden.ammonite.$file.project.settings.Depen
[549/549] jupyter0
JUPYTER_PATH=/opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/jupyter0.de
st/jupyter
[I 2023-01-04 07:44:50.904 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2023-01-04 07:44:50.937 ServerApp] jupyterlab | extension was successfully linked.
[I 2023-01-04 07:44:50.978 ServerApp] nbclassic | extension was successfully linked.
[I 2023-01-04 07:44:54.293 ServerApp] jupyter_nbextensions_configurator | extension was found and en
abled by notebook_shim. Consider moving the extension to Jupyter Server's extension paths.
[I 2023-01-04 07:44:54.294 ServerApp] jupyter_nbextensions_configurator | extension was successfully
 linked.
[I 2023-01-04 07:44:54.294 ServerApp] notebook_shim | extension was successfully linked.
[I 2023-01-04 07:44:54.379 ServerApp] notebook_shim | extension was successfully loaded.
[I 2023-01-04 07:44:54.382 ServerApp] [jupyter_nbextensions_configurator] enabled 0.6.1
[I 2023-01-04 07:44:54.383 ServerApp] jupyter_nbextensions_configurator | extension was successfully
 loaded.
[I 2023-01-04 07:44:54.387 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2023-01-04 07:44:54.390 LabApp] JupyterLab extension loaded from /home/mydev/.local/lib/python3.1
1/site-packages/jupyterlab
[I 2023-01-04 07:44:54.390 LabApp] JupyterLab application directory is /home/mydev/.local/share/jupy
ter/lab
[I 2023-01-04 07:44:54.402 ServerApp] jupyterlab | extension was successfully loaded.
[I 2023-01-04 07:44:54.416 ServerApp] nbclassic | extension was successfully loaded.
[I 2023-01-04 07:44:54.418 ServerApp] Serving notebooks from local directory: /opt/MyWorkSpace/MyPro
js/Dev/Notebook/Scala/Almond/Official/almond-main/notebooks
[I 2023-01-04 07:44:54.418 ServerApp] Jupyter Server 2.0.6 is running at:
[I 2023-01-04 07:44:54.418 ServerApp] http://localhost:8888/lab?token=fc8ebcdbc9f176f5ccfe993d051d25
2d7a62a3da92a67a19
[I 2023-01-04 07:44:54.418 ServerApp]  or http://127.0.0.1:8888/lab?token=fc8ebcdbc9f176f5ccfe993d05
1d252d7a62a3da92a67a19
[I 2023-01-04 07:44:54.418 ServerApp] Use Control-C to stop this server and shut down all kernels (t
wice to skip confirmation).
[C 2023-01-04 07:44:55.428 ServerApp]

    To access the server, open this file in a browser:
        file:///home/mydev/.local/share/jupyter/runtime/jpserver-8072-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/lab?token=fc8ebcdbc9f176f5ccfe993d051d252d7a62a3da92a67a19
     or http://127.0.0.1:8888/lab?token=fc8ebcdbc9f176f5ccfe993d051d252d7a62a3da92a67a19
[I 2023-01-04 07:46:11.473 LabApp] Generating new user for token-authenticated request: e7dc3220acce44c0bd37c9eca6
f7a7d1
[I 2023-01-04 07:48:42.261 LabApp] Build is up to date
```

try it on browser



Validate the example notebooks:

```
[mydev@fedora almond-main]$ ./mill -i validateExamples
[35/607] shared.interpreter-api[2.12.17].scalacOptions.overridden.ammonite.$file.project.settings.AlmondScala2Or3
[40/607] shared.interpreter-api[2.12.17].compile
Compiling compiler interface...
[info] compiling 4 Scala sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/sh
ared/interpreter-api/2.12.17/compile.dest/classes ...
[info] done compiling
[58/607] shared.logger-scala2-macros[2.12.17].mandatoryIvyDeps.overridden.mill.scalalib.JavaModule.mandatoryIvyDep
[61/607] shared.logger-scala2-macros[2.12.17].mandatoryIvyDeps.overridden.mill.scalalib.ScalaModule.mandatoryIvyD
[63/607] shared.logger-scala2-macros[2.12.17].transitiveIvyDeps.overridden.mill.scalalib.JavaModule.transitiveIvy
[71/607] shared.logger-scala2-macros[2.12.17].scalacOptions.overridden.ammonite.$file.project.settings.AlmondScal
[76/607] shared.logger-scala2-macros[2.12.17].compile
[info] compiling 1 Scala source to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/sha
red/logger-scala2-macros/2.12.17/compile.dest/classes ...
...
[335/607] scala.scala-interpreter[2.12.12].transitiveIvyDeps.overridden.mill.scalalib.JavaModule.transitiveIvyDep
[343/607] scala.scala-interpreter[2.12.12].scalacOptions.overridden.ammonite.$file.project.settings.AlmondScala2O
[348/607] scala.scala-interpreter[2.12.12].compile
[info] compiling 19 Scala sources and 2 Java sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Offici
l/almond-main/out/scala/scala-interpreter/2.12.12/compile.dest/classes ...
[warn] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/modules/scala/scala-interpreter/sr
/main/scala/almond/Execute.scala:186:17: method stop in class Thread is deprecated
[warn]               t.stop()
[warn]                 ^
...
[607/607] validateExamples
[NbConvertApp] Converting notebook /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/exampl
s/colors.ipynb to notebook
[NbConvertApp] Writing 2111 bytes to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/
alidateExamples.dest/output/colors.ipynb
[NbConvertApp] Converting notebook /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/exampl
s/displays.ipynb to notebook
[NbConvertApp] Writing 5226 bytes to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/
alidateExamples.dest/output/displays.ipynb
[NbConvertApp] Converting notebook /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/exampl
s/plotly-scala.ipynb to notebook
[NbConvertApp] Writing 63902 bytes to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out
validateExamples.dest/output/plotly-scala.ipynb
[NbConvertApp] Converting notebook /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/exampl
s/scalapy-displays.ipynb to notebook
[NbConvertApp] Writing 7314 bytes to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/
alidateExamples.dest/output/scalapy-displays.ipynb
[NbConvertApp] Converting notebook /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/exampl
s/test.ipynb to notebook
[NbConvertApp] Writing 1101 bytes to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/
alidateExamples.dest/output/test.ipynb
[mydev@fedora almond-main]$
```

Build a kernel launcher and install it:

```
[mydev@fedora almond-main]$ ./mill launcher
[297/549] scala.scala-kernel-api[2.13.10].resources.overridden.ammonite.$file.project.settings.DependencyListReso
[549/549] launcher
/opt/MyWorkSpace/MyProjs/Dev/Notebook/Scala/Almond/Official/almond-main/out/scala/scala-kernel/2.13.10/unixLaunch
r.dest/launcher
[mydev@fedora almond-main]$

[mydev@fedora almond-main]$ out/scala/scala-kernel/2.13.10/unixLauncher.dest/launcher --install
Installed scala kernel under /home/mydev/.local/share/jupyter/kernels/scala
[mydev@fedora almond-main]$
```

III. Project Polynote on ARM

1) Overview

- https://polynote.org/

  The polyglot notebook with first-class Scala support..

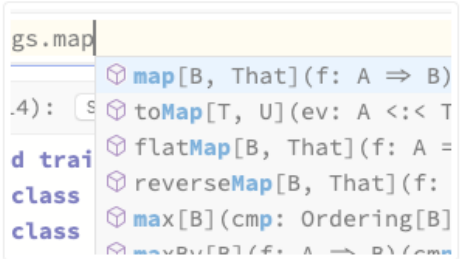
- Features

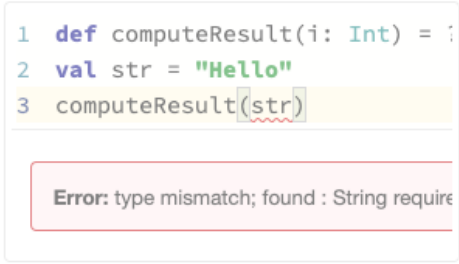
Polynote is a different kind of notebook. It supports **mixing multiple languages** in one notebook, and sharing data between them seamlessly. It encourages **reproducible notebooks** with its immutable data model. (Learn more)

Here are some of its *unique* features:

Editing

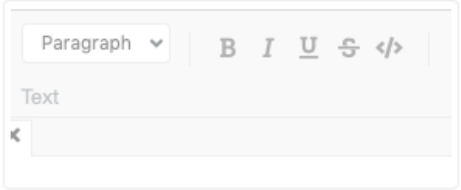
Interactive auto-complete helps you find what you're looking for without switching to the documentation.

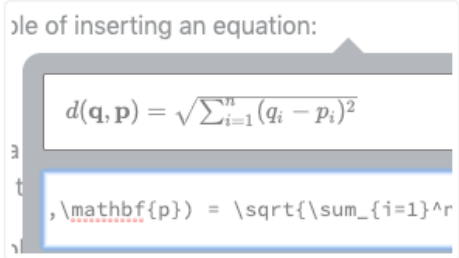




Highlighting your errors helps you quickly figure out what's gone wrong.

Rich text editing lets you see your formatted text cells as you write them, and edit them like a document.

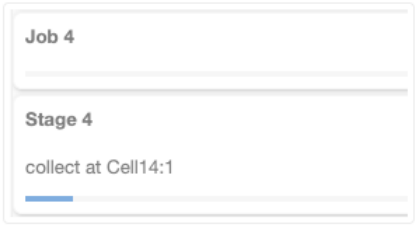


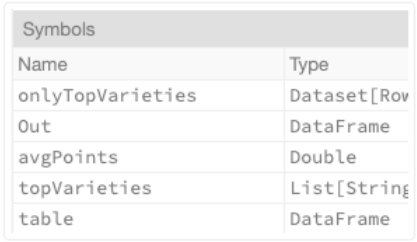


Easily insert **LaTeX equations** into your text cells.

Executing

Always know what the kernel is doing with **individual task tracking.**





Symbols	
Name	Type
onlyTopVarieties	Dataset[Rov
Out	DataFrame
avgPoints	Double
topVarieties	List[String
table	DataFrame

The **symbol table** keeps track of what you've defined, and what's available in the current cell. **Inspect any value** for detailed information and rich visualizations.

See exactly what code is executing right now, as Polynote **highlights the running statement in real-time.**

Ordered cell semantics ensures your notebook can be reproducibly executed.

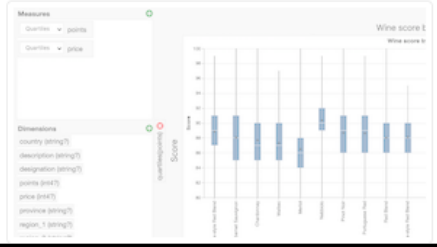
Data and visualization

First-class integration with **Apache Spark™** for exploring, analyzing, and visualizing big data.

region_2: string?	variety: string?
Napa	Cabernet Sauvignc
Sonoma	Sauvignon Blanc
Willamette Valley	Pinot Noir
Willamette Valley	Pinot Noir
Sonoma	Pinot Noir

Browse **table-structured data**, including collections of data structures and Spark data sets.

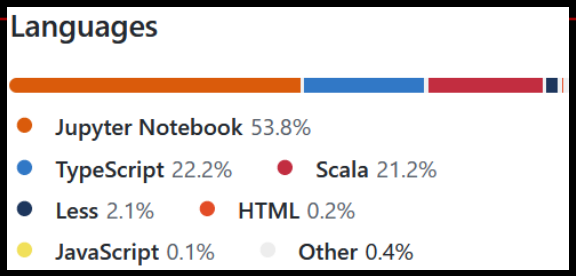
See **rich representations** of your data, automatically derived *or* supplied by a typeclass.



Easily create visualizations with the built-in **plot editor**, or define your own with the Vega visualization language.

Src

- **https://github.com/polynote/polynote**
  Polynote is an experimental polyglot notebook environment. Currently, it supports Scala and Python(with or without Spark), SQL, and Vega.

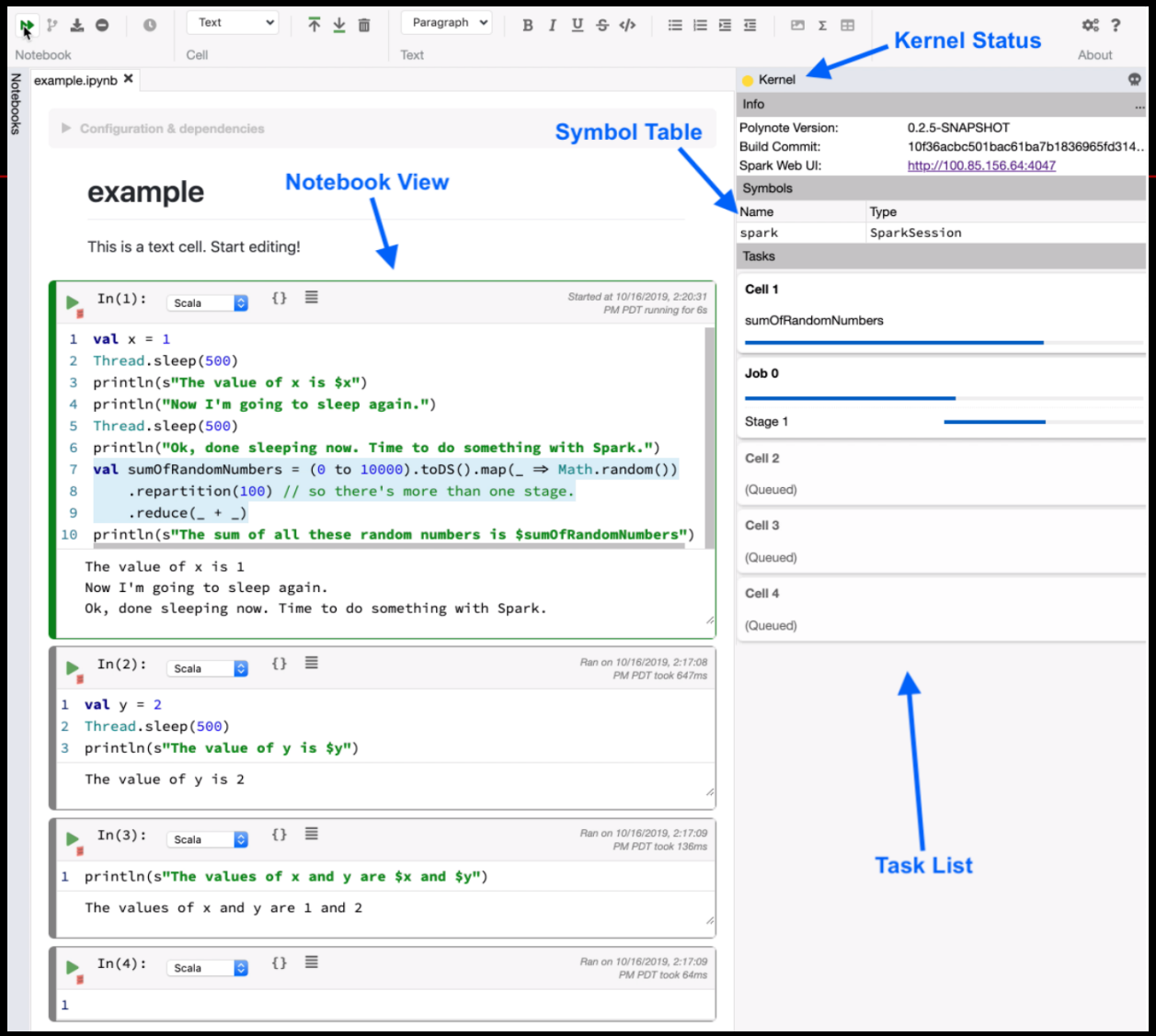


- **Why is it**

Current notebook solutions, like Jupyter and Zeppelin, are lacking in some fundamental features:

- *Code editing* – the code editing capabilities in most notebook tools leave plenty to be desired. Why can't a notebook tool have modern editing capabilities like those you'd find in an IDE? Polynote provides useful autocomplete, parameter hints, and more – we're planning to add even more features, like jump-to-definition.
- *Text editing* – you can use the WYSIWYG editor for composing text cells, so you'll know what the text will look like as you're writing. TeX equations are also supported.
- *Multi-language support* – Polynote allows you to mix multiple languages in one notebook, while sharing definitions seamlessly between them.
- *Runtime insight* – Polynote tries to keep you informed of what's going on at runtime:
  - The tasks area shows you what the kernel is doing at any given time.
  - The symbol table shows you what variables and functions you've defined, so you don't have to scroll around to remind yourself.
  - Compile failures and runtime exceptions are highlighted in the editor (for supported languages), so you can see exactly what's going wrong.

Example



Source: https://netflixtechblog.com/open-sourcing-polynote-an-ide-inspired-polyglot-notebook-7f929d3f447

2) RPi4

1.1 Quick start

Official Guide

- https://polynote.org/latest/docs/installation/

Danger

Polynote allows **arbitrary remote code execution**, which is necessary for a notebook tool to function. While we'll try to improve safety by adding security measures, it will never be completely safe to run Polynote on your personal computer. For example:

- It's possible that other websites you visit could use Polynote as an attack vector. Browsing the web while running Polynote is unsafe.
- It's possible that remote attackers could use Polynote as an attack vector. Running Polynote on a computer that's accessible from the internet is unsafe.
- Even running Polynote inside a container doesn't guarantee safety, as there will always be privilege escalation and container escape vulnerabilities which an attacker could leverage.

Please be diligent about checking for new releases, as they could contain fixes for critical security flaws.

Please be mindful of the security issues that Polynote causes; consult your company's security team before running Polynote. You are solely responsible for any breach, loss, or damage caused by running this software insecurely.

…

Prerequisites:

1) Spark support

   Omitted currently…

2) Python support

pip install -r $SRC_POLYNOTE/requirements.txt

```
virtualenv
ipython
nbconvert
numpy
pandas==1.2.5
jedi>=0.18.1
jep
```

sudo dnf install python3-jep…

Download:

Polynote consists of a JVM-based server application, which serves a web-based client. To try it locally, find the latest release on the releases page and download the attached `polynote-dist.tar.gz` file (you'll find it under `Assets`). Unpack the archive:

```
tar -zxvpf polynote-dist.tar.gz
cd polynote
```

Configure:

You won't need to change any of the default configuration in order to get Polynote up and running locally.

https://polynote.org/latest/docs/server-configuration/

Run:

To start the server, run the included python script:

```
./polynote.py
```

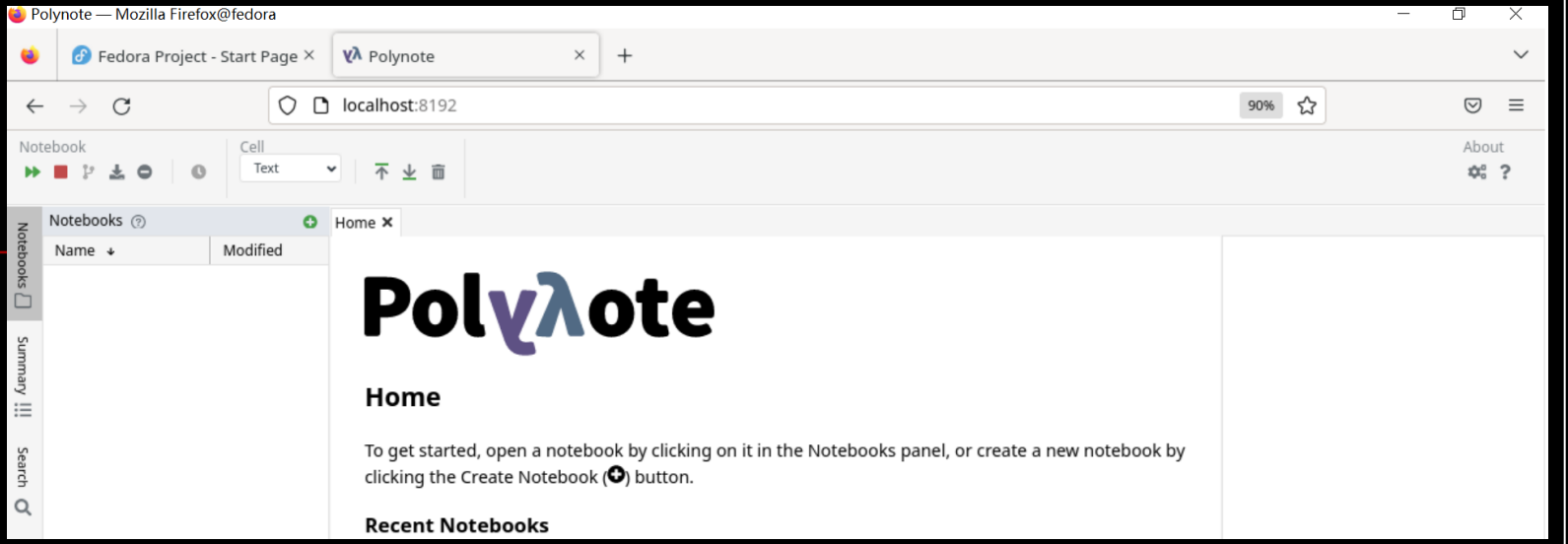
Once the server has started, navigate your browser to `http://localhost:8192` (if using the default network configuration).

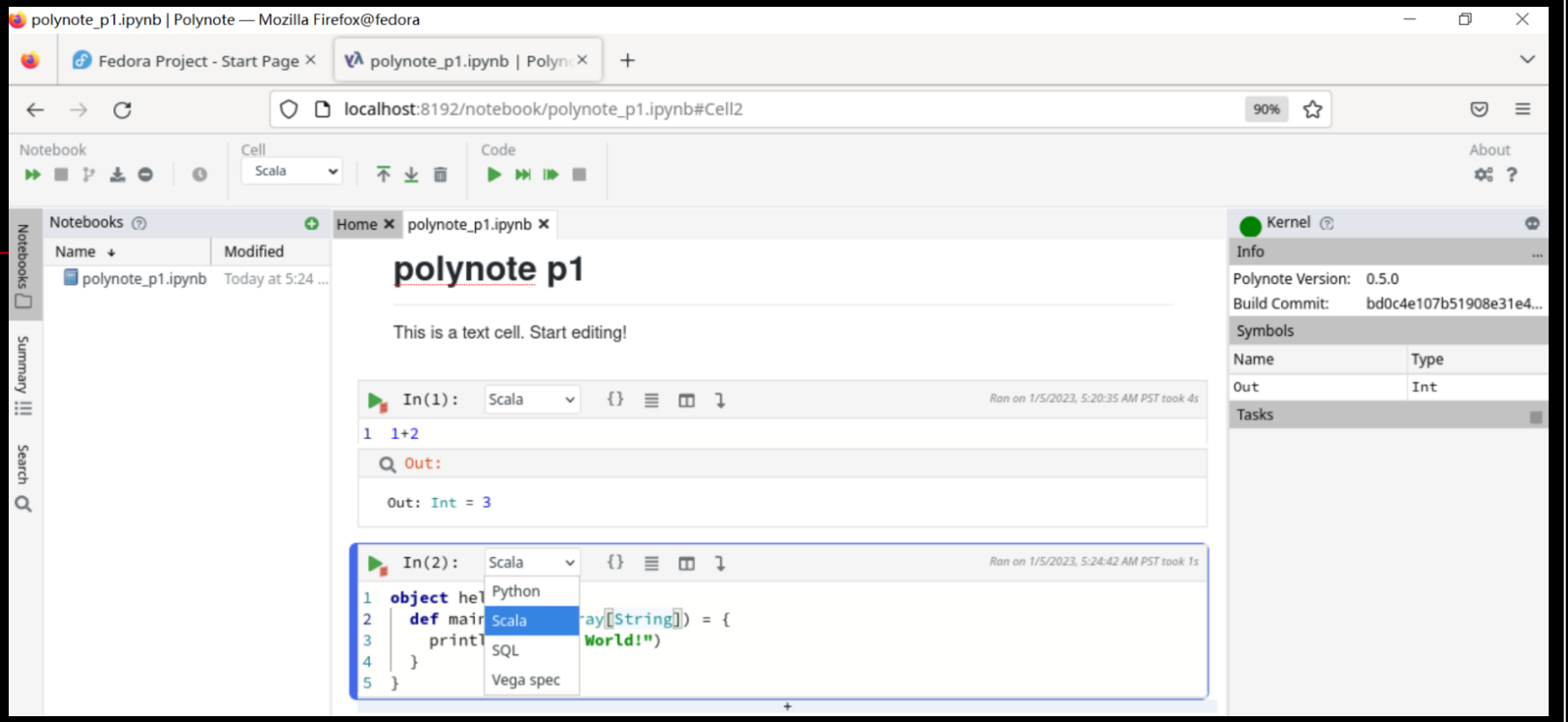
Our test (:

```
[mydev@fedora polynote-dist-0.5.0]$ tree -L 1 .
.
├── config-template.yml
├── deps
├── plugin
├── plugins
├── plugins.d
├── polynote.py
├── requirements.txt
└── static
```

```
[mydev@fedora polynote-dist-0.5.0]$ tree deps/2.13
deps/2.13
├── polynote-kernel-assembly-0.5.0.jar
├── polynote-runtime-assembly-0.5.0.jar
├── polynote-server-assembly-0.5.0.jar
├── polynote-spark-assembly-0.5.0.jar
├── polynote-spark-runtime-assembly-0.5.0.jar
├── scala-collection-compat_2.13-2.1.1.jar
├── scala-compiler-2.13.6.jar
├── scala-library-2.13.6.jar
├── scala-reflect-2.13.6.jar
└── scala-xml_2.13-1.2.0.jar
```

```
[mydev@fedora polynote-dist-0.5.0]$ ./polynote.py
['java', '-cp', '/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/s
ala-collection-compat_2.11-2.1.1.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-di
t-0.5.0/deps/2.11/scala-xml_2.11-1.2.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polyn
te-dist-0.5.0/deps/2.11/polynote-spark-runtime-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/
olynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-spark-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Not
book/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/scala-compiler-2.11.12.jar:/opt/MyWorkSpace/MyProjs
Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-kernel-assembly-0.5.0.jar:/opt/MyW
rkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-runtime-assembly-0
5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-s
rver-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps
2.11/scala-library-2.11.12.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5
0/deps/2.11/scala-reflect-2.11.12.jar:', '-Djava.library.path=/usr/lib64/python3.11/site-packages/jep', 'polynote
Main']
[INFO]   Loading configuration from config.yml
[INFO]   Loaded configuration: PolynoteConfig(Listen(8192,127.0.0.1),KernelConfig(None,None,None,None),Storage(tm
,notebooks,Map(),Wal(false)),List(),List(),Map(),None,Behavior(true,Always,List(),List()),Security(None,None),UI(
),Credentials(None),,Map(),StaticConfig(None,None))
...
[INFO]
   |      _____     _                    _
   |     |  __ \   | |                  | |
   |     | |__) |__ | |_   _ _ __   ___ | |_ ___
   |     |  ___/ _ \| | | | | '_ \ / _ \| __/ _ \
   |     | |  | (_) | | |_| | | | | (_) | ||  __/
   |     |_|   \___/|_|\__, |_| |_|\___/ \__\___|
   |                    __/ |
   |                   |___/
[INFO]   Polynote version 0.5.0
[INFO]   Server listening on http://localhost:8192/
```


Polynote — Mozilla Firefox@fedora
Fedora Project - Start Page
Polynote
localhost:8192
90%
Notebook
Cell
Text
About
Notebooks
Name
Modified
Home
Notebooks
Summary
Search
Polynote
Home
To get started, open a notebook by clicking on it in the Notebooks panel, or create a new notebook by clicking the Create Notebook ( ) button.
Recent Notebooks




```
[INFO]   Will write to polynote_p1.ipynb
[INFO]   Deploying with command:
   |     /opt/MyWorkSpace/DevSW/Java/JDK/GraalVM/CE/v23.0.0-dev-20221221/Java19/bin/java -cp /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/scala-collection-compat_2.11-2.1.1.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/scala-xml_2.11-1.2.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-spark-runtime-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-spark-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/scala-compiler-2.11.12.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-kernel-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-runtime-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/polynote-server-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/scala-library-2.11.12.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-dist-0.5.0/deps/2.11/scala-reflect-2.11.12.jar -Dlog4j.configuration=log4j.properties -Djava.library.path=/usr/lib64/python3.11/site-packages/jep polynote.kernel.remote.RemoteKernelClient --address 127.0.0.1 --port 34357 --kernelFactory polynote.kernel.LocalKernelFactory
```

Build from src

- On master branch, last commit: baae6540fd94971f2a146f6e42b4b92304a02c81

```
[mydev@fedora polynote-master]$ sbt compile
[info] welcome to sbt 1.7.2 (GraalVM Community Java 19.0.1)
[info] loading settings for project polynote-master-build from plugins.sbt ...
[info] loading project definition from /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/pro
ect
[info] Updating
https://repo1.maven.org/maven2/org/scoverage/sbt-scoverage_2.12_1.0/1.6.0/sbt-scoverage-1.6.0.pom
  100.0% [##########] 2.1 KiB (404B / s)
...
https://repo1.maven.org/maven2/org/apache/ivy/ivy/2.5.0/ivy-2.5.0.jar
  100.0% [##########] 1.3 MiB (7.3 KiB / s)
[info] Fetched artifacts of
[info] loading settings for project polynote from build.sbt ...
[info] set current project to polynote (in build file:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/poly
ote-master/)
[warn] there are 14 keys that are not used by any other settings/tasks:
[warn]
[warn] * polynote / circeVersion
[warn]   +- /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/build.sbt:111
...
[warn] note: a setting might still be used by a command; to exclude a key from this `lintUnused` check
[warn] either append it to `Global / excludeLintKeys` or call .withRank(KeyRanks.Invisible) on the key
[info] Executing in batch mode. For better performance use sbt's shell
[info] compiling 5 Scala sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynot
e-env/target/scala-2.11/classes ...
[info] compiling 1 Scala source to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote
-macros/target/scala-2.11/classes ...
[info] Non-compiled module 'compiler-bridge_2.11' for Scala 2.11.12. Compiling...
[info]   Compilation completed in 66.835s.
[info] compiling 13 Scala sources and 1 Java source to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/poly
note-master/polynote-runtime/target/scala-2.11/classes ...
[warn] there was one feature warning; re-run with -feature for details
[warn] one warning found
[info] compiling 1 Scala source to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote
-spark-runtime/target/scala-2.11/classes ...
[info] compiling 51 Scala sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polyno
te-kernel/target/scala-2.11/classes ...
```

maybe you will meet an error like below:

```
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/data/Rope.scala:117:22: type mismatch;
[error]  found   : a.type (with underlying type Array[Char])
[error]  required: ?{def isEmpty: ?}
[error] Note that implicit conversions are not applicable because they are ambiguous:
[error]  both method ArrayCharSequence in object Predef of type (__arrayOfChars: Array[Char])ArrayCharSequence
[error]  and method charArrayOps in object Predef of type (xs: Array[Char])scala.collection.mutable.ArrayOps[Char]
[error]  are possible conversion functions from a.type to ?{def isEmpty: ?}
[error]     if (a == null || a.isEmpty) {
[error]                      ^
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/data/Rope.scala:117:24: value isEmpty is not a member of Array[Char]
[error]     if (a == null || a.isEmpty) {
[error]                        ^
[error] two errors found
[error] (polynote-kernel / Compile / compileIncremental) Compilation failed
[error] Total time: 5335 s (01:28:55), completed Jan 14, 2023, 9:22:15 AM
[mydev@fedora polynote-master]$
[mydev@fedora polynote-master]$ vim /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynot
e-kernel/src/main/scala/polynote/data/Rope.scala
[mydev@fedora polynote-master]$
```

seems to be caused by our JDK: https://github.com/scala/bug/issues/12172, and the target scala version…just add a workaround like below:

```
[mydev@fedora polynote-master]$ git diff
diff --git a/polynote-kernel/src/main/scala/polynote/data/Rope.scala b/polynote-kernel/src/main/scala/polynote/data/Rope.sca
la
index 954063fb..9316c6d4 100644
--- a/polynote-kernel/src/main/scala/polynote/data/Rope.scala
+++ b/polynote-kernel/src/main/scala/polynote/data/Rope.scala
@@ -114,7 +114,7 @@ object Rope {

   /** Creates a rope from a character array. */
   def apply(a: Array[Char]): Rope =
-    if (a == null || a.isEmpty) {
+    if (a == null || a.length == 0) {
       RopeEmpty
     } else if (a.length > thresh) {
       val (a1, a2) = a.splitAt(a.length / 2)
[mydev@fedora polynote-master]$
```

rebuild and assmebly(omitted ‘sbt test’):

```
...
[warn] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-server/src/main/scala/polynote/server/repository/format/ipynb/ast.scala:183:25: type ObjectEncoder in package circe is deprecated: Use Encoder.AsObject
[warn]   implicit val encoder: ObjectEncoder[JupyterCell] = deriveEncoder[JupyterCell].contramapObject[JupyterCell] {
[warn]                         ^
[warn] two warnings found
[info] compiling 10 Scala sources to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-spark/target/scala-2.11/classes ...
[success] Total time: 574 s (09:34), completed Jan 15, 2023, 4:42:24 AM
[mydev@fedora polynote-master]$
...
[mydev@fedora polynote-master]$ sbt assembly
[info] welcome to sbt 1.7.2 (GraalVM Community Java 19.0.1)
[info] loading settings for project polynote-master-build from plugins.sbt ...
[info] loading project definition from /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/project
[info] loading settings for project polynote from build.sbt ...
[info] set current project to polynote (in build file:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/)
[warn] there are 14 keys that are not used by any other settings/tasks:
[warn]
[warn] * polynote / circeVersion
[warn]   +- /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/build.sbt:111
...
[warn] note: a setting might still be used by a command; to exclude a key from this `lintUnused` check
[warn] either append it to `Global / excludeLintKeys` or call .withRank(KeyRanks.Invisible) on the key
[warn] Ignored unknown package option FixedTimestamp(Some(1262304000000))
[info] compiling 1 Scala source to /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-runtime/target/scala-2.11/classes ...
[info] Strategy 'discard' was applied to 2 files (Run the task at debug level to see details)
[warn] Ignored unknown package option FixedTimestamp(Some(1262304000000))
[info] Strategy 'discard' was applied to a file (Run the task at debug level to see details)
[warn] Ignored unknown package option FixedTimestamp(Some(1262304000000))
[warn] Ignored unknown package option FixedTimestamp(Some(1262304000000))
[warn] Ignored unknown package option FixedTimestamp(Some(1262304000000))
[info] Strategy 'discard' was applied to 14 files (Run the task at debug level to see details)
[warn] Ignored unknown package option FixedTimestamp(Some(1262304000000))
[info] Strategy 'discard' was applied to 37 files (Run the task at debug level to see details)
[info] Strategy 'rename' was applied to 2 files (Run the task at debug level to see details)
[warn] Ignored unknown package option FixedTimestamp(Some(1262304000000))
[success] Total time: 93 s (01:33), completed Jan 15, 2023, 4:53:24 AM
[mydev@fedora polynote-master]$
```

the output lacks of Scala related jar files when compared with the official release:

```
[mydev@fedora polynote-master]$ find . -name "*.jar"
./polynote-env/target/scala-2.11/polynote-env-assembly-0.5.0.jar
./polynote-kernel/target/scala-2.11/polynote-kernel-assembly-0.5.0.jar
./polynote-runtime/target/scala-2.11/polynote-runtime-assembly-0.5.0.jar
./polynote-server/target/scala-2.11/polynote-server-assembly-0.5.0.jar
./polynote-spark-runtime/target/scala-2.11/polynote-spark-runtime-assembly-0.5.0.jar
./polynote-spark/target/scala-2.11/polynote-spark-assembly-0.5.0.jar
./target/scala-2.11/polynote-assembly-0.5.0.jar
[mydev@fedora polynote-master]$
```

prepare the jars to deps/2.11 directory manually, and run the polynote.py, it will failed as below:

```
[mydev@fedora polynote-master]$ scripts/polynote.py
['java', '-cp', '/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/scripts/deps/2.11/polynote
-env-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/scripts/deps/2.11/p
olynote-kernel-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/scripts/d
eps/2.11/polynote-runtime-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-maste
r/scripts/deps/2.11/polynote-server-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/poly
note-master/scripts/deps/2.11/polynote-spark-runtime-assembly-0.5.0.jar:/opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polyn
ote/Official/polynote-master/scripts/deps/2.11/polynote-spark-assembly-0.5.0.jar:', '-Djava.library.path=/usr/lib64/python3.
11/site-packages/jep', 'polynote.Main']
Error: Unable to initialize main class polynote.Main
Caused by: java.lang.NoClassDefFoundError: scala/collection/immutable/List
[mydev@fedora polynote-master]$
```

Checking the build system of Polynote to dig out the difference for the output between the official release and that of us, and try to provide a workaround…

in addition, take another try to build against host Scala version with the following change:

```
[mydev@fedora polynote-master]$ git diff
diff --git a/build.sbt b/build.sbt
index 2a0918e3..fa0b3acc 100644
--- a/build.sbt
+++ b/build.sbt
@@ -46,7 +46,7 @@ lazy val scalaBinaryVersions = scalaVersions.map {
 }.distinct

 val commonSettings = Seq(
-  scalaVersion := "2.11.12",
+  scalaVersion := "2.13.10",
   crossScalaVersions := scalaVersions,
   organization := "org.polynote",
   publishMavenStyle := true,
[mydev@fedora polynote-master]$
```

build with ‘sbt compile’, but more errors appeared this time:

...

```
[warn] -target is deprecated: Use -release instead to compile against the correct platform API.
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/kernel/dependency/CoursierFetcher.scala:249:33: zio.RIO[R, _] takes no type parameters, expected: 1
[error]   implicit def zioSync[R]: Sync[RIO[R, ?]] = new Sync[RIO[R, ?]] {
[error]                                 ^
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/kernel/dependency/CoursierFetcher.scala:39:50: type mismatch;
[error]  found   : coursier.util.Sync[coursier.util.Task]
[error]  required: coursier.util.Sync[polynote.kernel.dependency.CoursierFetcher.ArtifactTask]
[error] Error occurred in an application involving default arguments.
[error]   private val baseCache = FileCache[ArtifactTask]()
[error]                                    ^
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/kernel/dependency/CoursierFetcher.scala:128:41: could not find implicit value for parameter sync: coursier.util.Sync[[A
]polynote.kernel.dependency.CoursierFetcher.ArtifactTask[A]]
[error]     val repos = (repositories ++ Resolve(cache).repositories).map {
[error]                                         ^
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/kernel/dependency/CoursierFetcher.scala:165:12: could not find implicit value for parameter sync: coursier.util.Sync[[A
]polynote.kernel.dependency.CoursierFetcher.ArtifactTask[A]]
[error]     Resolve(cache)
[error]             ^
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/kernel/dependency/CoursierFetcher.scala:169:25: missing argument list for method countingFetcher
[error] Unapplied methods are only converted to functions when a function type is expected.
[error] You can make this conversion explicit by writing `countingFetcher _` or `countingFetcher(_)` instead of `countingFet
cher`.
[error]       .transformFetcher(countingFetcher)
[error]                         ^
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/kernel/dependency/CoursierFetcher.scala:171:17: missing argument list for method recover
[error] Unapplied methods are only converted to functions when a function type is expected.
[error] You can make this conversion explicit by writing `recover _` or `recover(_)` instead of `recover`.
[error]       .catchAll(recover)
[error]                 ^
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/kernel/dependency/CoursierFetcher.scala:181:16: could not find implicit value for parameter sync: coursier.util.Sync[[A
]polynote.kernel.dependency.CoursierFetcher.OuterTask[A]]
[error]       Artifacts(new TaskManagedCache(cache, blockingExecutor)).withResolution(resolution).withMainArtifacts(true).io
Result.map {
[error]                ^
[error] /opt/MyWorkSpace/MyProjs/Dev/Notebook/Polyglot/Polynote/Official/polynote-master/polynote-kernel/src/main/scala/poly
note/kernel/dependency/CoursierFetcher.scala:249:55: zio.RIO[R, _] takes no type parameters, expected: 1
[error]   implicit def zioSync[R]: Sync[RIO[R, ?]] = new Sync[RIO[R, ?]] {
[error]                                                       ^
[warn] one warning found
[error] 8 errors found
[error] (polynote-kernel / Compile / compileIncremental) Compilation failed
[error] Total time: 8314 s (02:18:34), completed Jan 17, 2023, 4:04:41 AM
[mydev@fedora polynote-master]$
```

Working on the fixes…

IV. New Exploration Technologies

1) Kernel-free Notebook

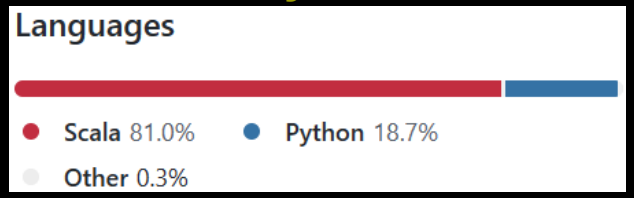
1.1 Vizier

1.1.1 Overview

- https://github.com/VizierDB/vizier-scala

  The world's first kernel-free notebook.

  Vizier is an interactive, reactive workbook: a workflow system with a notebook-style interface.



- No Kernel?

Unlike most notebooks, Vizier is not backed by a long-running kernel. Each cell runs in a fresh interpreter.

Cells communicate by creating "artifacts":

- datasets (e.g., Pandas or Spark dataframes)
- files
- parameters
- charts
- python code

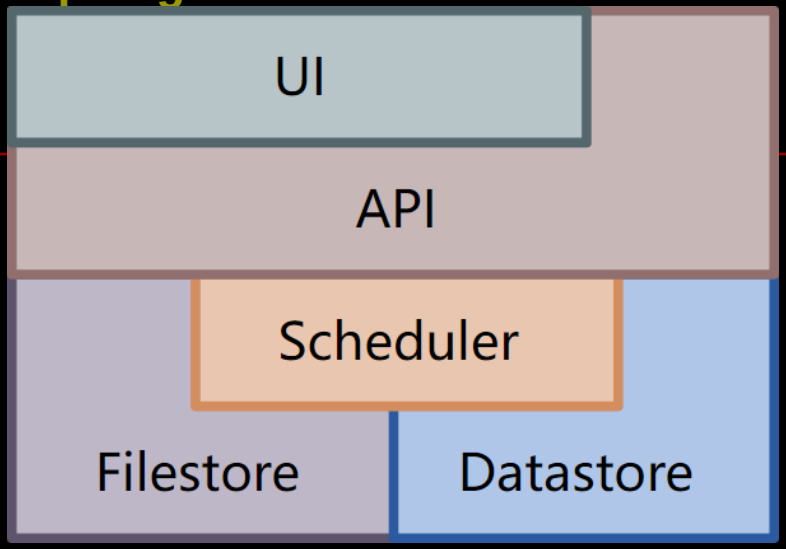
For example, you can define and export a function in a python cell, and use it as a User Defined Function in a SQL cell.
Vizier tracks which artifacts a cell uses, so that if you change something, it knows which cells need to be re-run. When an artifact is updated (e.g., when you modify the function), every cell that used it (e.g., the SQL cell) will be re-executed.

Features

- **No Kernels**: There's no long-running kernel with state to lose if you have to log out.
- **Reproducibility**: Vizier executes cells *in order*, and automatically re-executes cells when their inputs change so your notebook's outputs are always up-to-date.
- **Data Snapshots**: Vizier automatically snapshots data created by each cell, so you can re-run a cell without re-running all of its inputs.
- **Polyglot**: You can combine Python, SQL, and Scala, all seamlessly working with the same data.
- **Code-Optional**: Use a spreadsheet-style interface, or Vizier's "data lenses" to work with your data, code optional!
- **Workflow Snapshots**: Vizier automatically keeps a record of how you edit your workflow so you can always go back to an earlier version.
- **Scalable**: Vizier datasets are backed by Spark and Apache Arrow, allowing you to make big changes fast.

Architecture & Design

- https://github.com/VizierDB/vizier-scala/wiki/Vizier-Architecture



- *API*: Vizier uses an API layer to manages notebook state and mediates between the components. The API may be accessed directly (e.g., by scripts), or via Vizier's UI.
- *UI*: Vizier relies on a HTML/JS-based frontend for most user interactions.
- *Scheduler*: A scheduler is responsible for evaluating dependencies between notebook cells and re-executing cells that are out-of-date (whether because the cell was updated or one of its inputs changed in a new notebook version).
- *Datastore*: Structured data (dataframes) and simple unstructured data (blobs) are stored in the Datastore layer. In addition to keeping track of this state, the datastore layer is responsible for managing fine-grained provenance relationships between data elements, and profiling dataset state.
- *Filestore*: Vizier uses a file storage layer to manage large unstructured data.

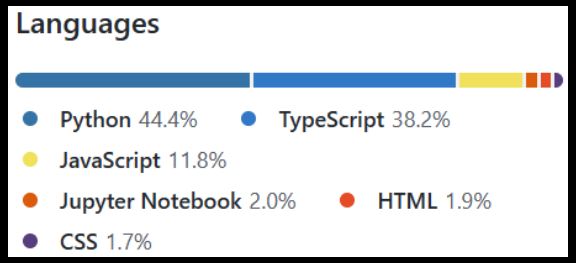
There are two versions of Vizier under development: Scala and Python. Their architectures are related but different:

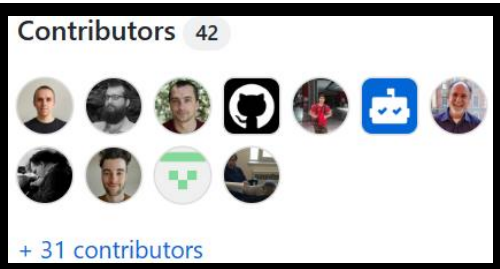
2) Wasm-based Notebook
2.1 JupyterLite
2.1.1 Overview

- https://github.com/jupyterlite/jupyterlite

Wasm powered Jupyter running in the browser.

JupyterLite is a JupyterLab distribution that **runs entirely in the browser** built from the ground-up using JupyterLab components and extensions.





- **Ease of Deployment**

- Served via well-cacheable, static HTTP(S), locally or on most static web hosts
- Embeddable within larger applications
- Requires no dedicated *application server* much less a container orchestrator
- Fine-grained configurability of page settings, including reuse of federated extensions

Browser-based Interactive Computing

JupyterLite is all about accessible browser-based interactive computing:

- Python kernel backed by Pyodide running in a Web Worker
  - Initial support for interactive visualization libraries such as `altair`, `bqplot`, `ipywidgets`, `matplotlib`, and `plotly`
- JavaScript and P5.js kernels running in an `IFrame`
- View hosted example Notebooks and other files, then edit, save, and download from the browser's `IndexDB` (or `localStorage`)
- Support for saving settings for JupyterLab/Lite core and federated extensions
- Basic session and kernel management to have multiple kernels running at the same time
- Support for Code Consoles

Related

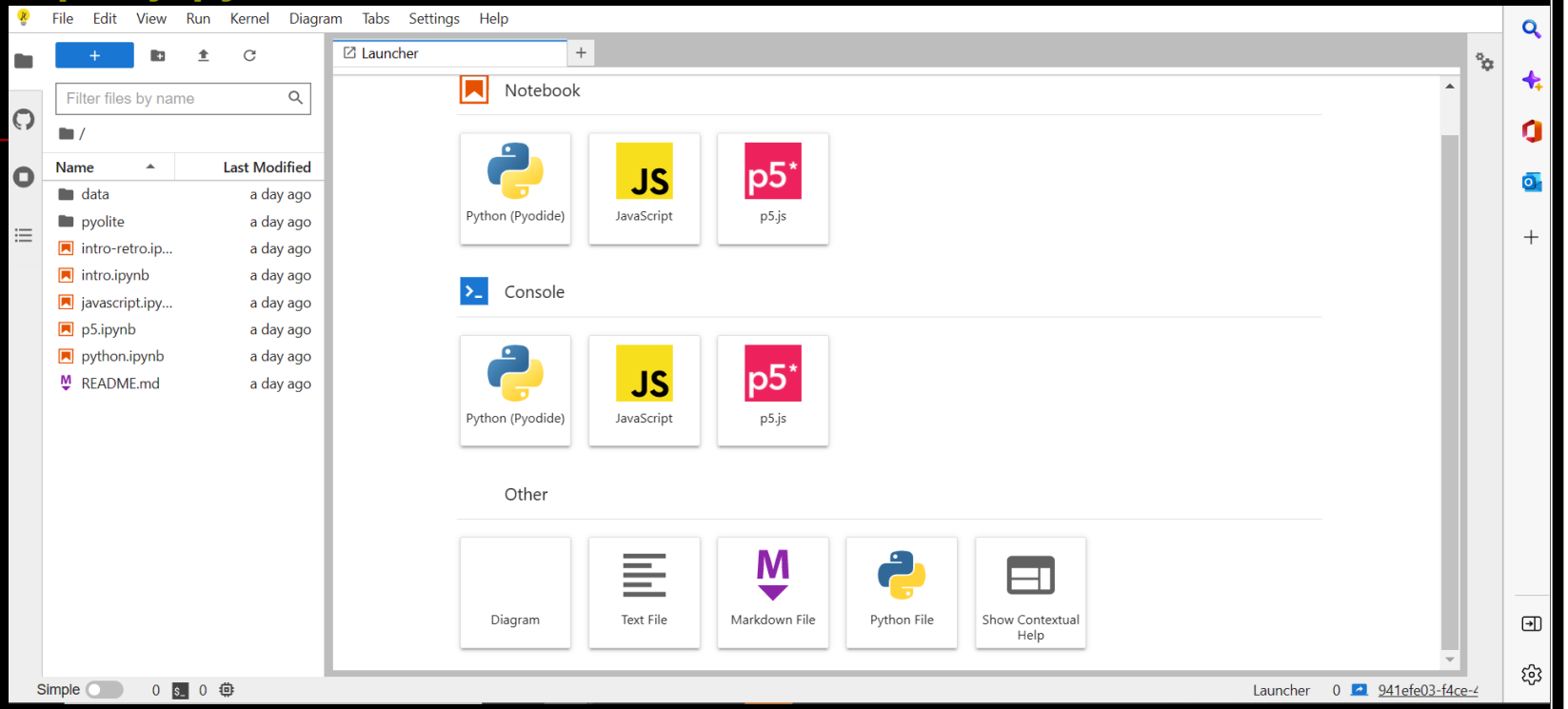
JupyterLite is a reboot of several attempts at making a full static Jupyter distribution that runs in the browser, without having to start the Python Jupyter Server on the host machine.

The goal is to provide a lightweight computing environment accessible in a matter of seconds with a single click, in a web browser and without having to install anything.

This project is a collection of packages that can be remixed together in variety of ways to create new applications and distributions. Most of the packages in this repo focus on providing server-like components that run in the browser (to manage kernels, files and settings), so existing JupyterLab extensions and plugins can be reused out of the box.

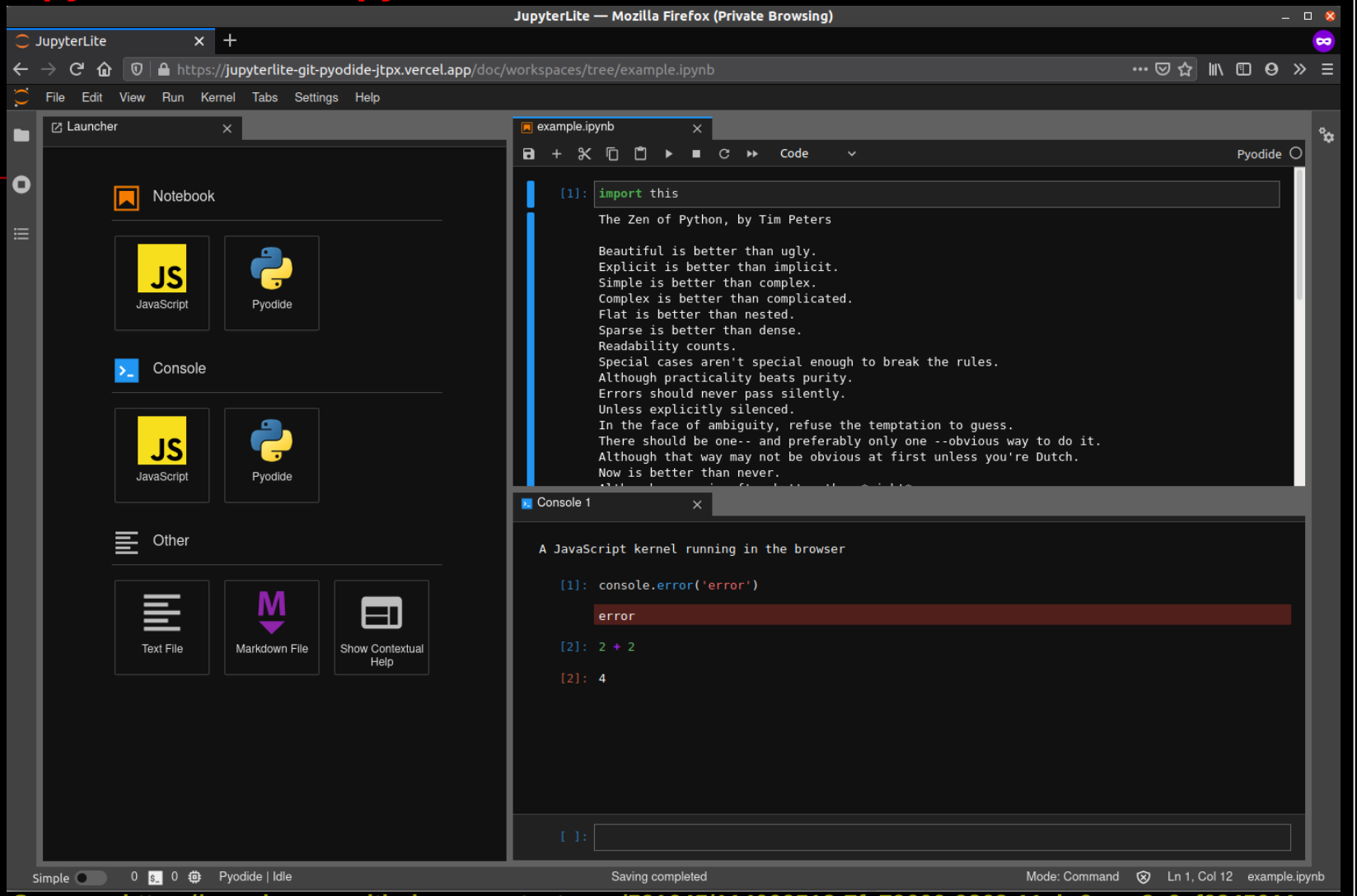
Example

- https://jupyterlite.readthedocs.io/en/latest/_static/lab/index.html



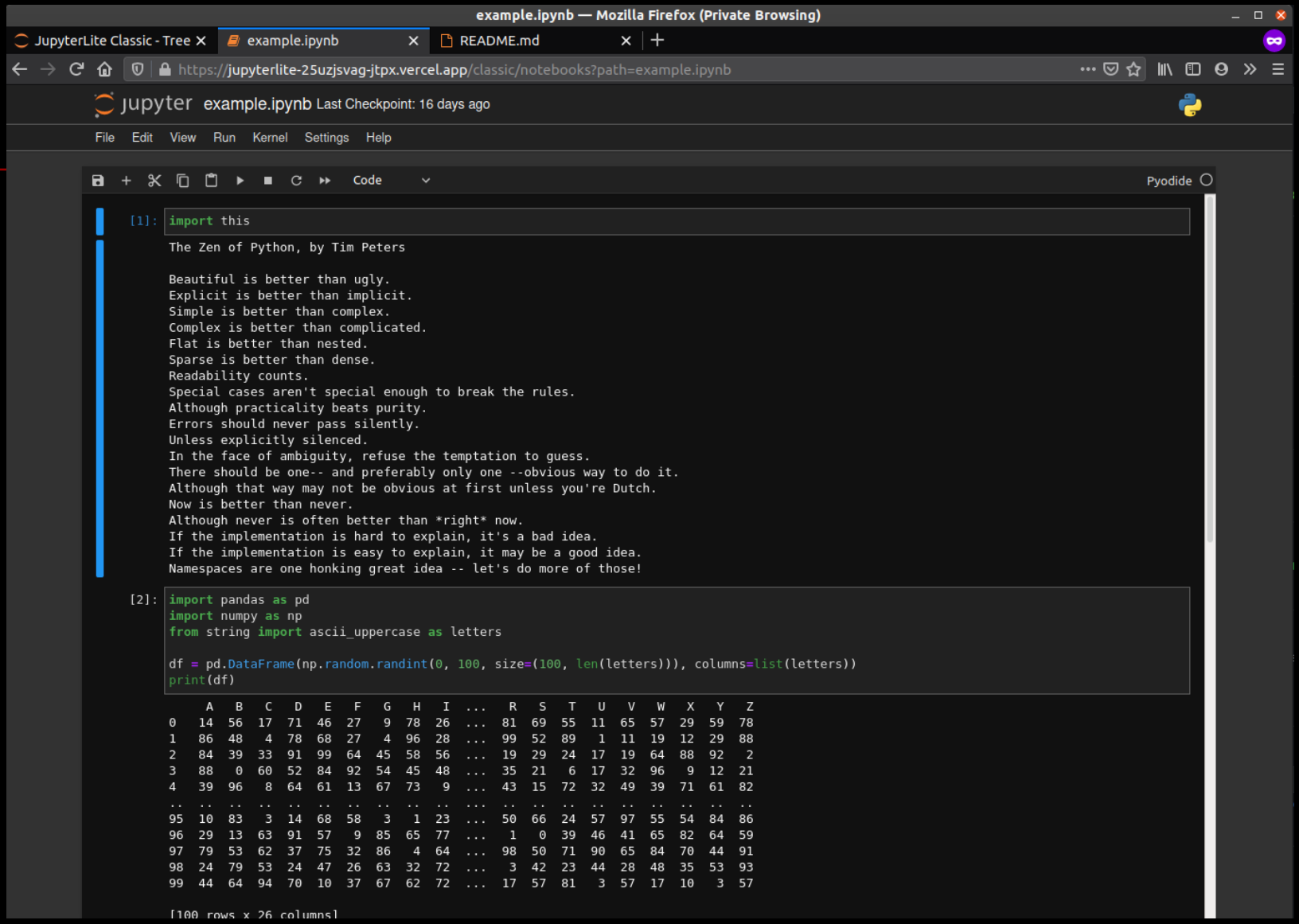
JupyterLite works with both JupyterLab and RetroLab.

JupyterLite with JupyterLab



Source: https://user-images.githubusercontent.com/591645/114009512-7fe79600-9863-11eb-9aac-3a9ef6345011.png

JupyterLite with RetroLab



Source: https://user-images.githubusercontent.com/591645/114454062-78fdb200-9bda-11eb-9cda-4ee327dd1c77.png

3) Data-oriented Notebook
3.1 Toree
3.1.1 Overview

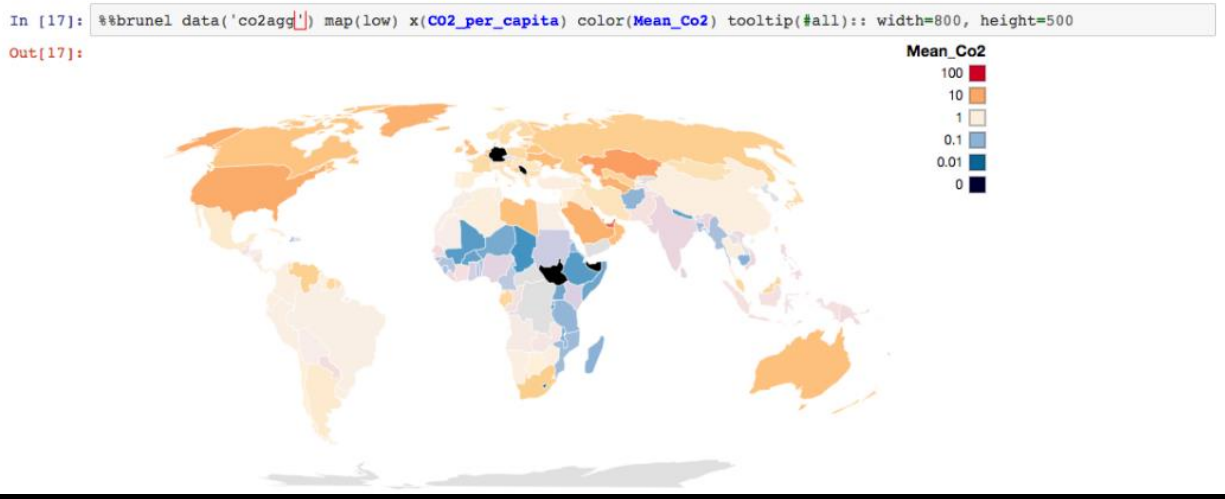
- https://toree.apache.org/

Apache Toree is a kernel for the Jupyter Notebook platform providing interactive access to Apache Spark. It has been developed using the IPython messaging protocol and 0MQ, and despite the protocol's name, Apache Toree currently exposes the Spark programming model in Scala, Python and R languages.

- Features

Visualizations

Apache Toree, via extensions like Brunel for Apache Toree, supports rich visualizations that integrates directly with Spark Data Frame APIs



Magics

Apache Toree provides a set of magics that enhances the user experience manipulating data coming from Spark tables or data

```
In [6]: %%sql
        select * from customers
```

```
Out[6]: +---+-------------+-------+---------+-------+
        |age|          job|marital|education|balance|
        +---+-------------+-------+---------+-------+
        | 30|   unemployed|married|  primary|   1787|
        | 33|     services|married|secondary|   4789|
        | 35|   management| single| tertiary|   1350|
        | 30|   management|married| tertiary|   1476|
        | 59|  blue-collar|married|secondary|      0|
        | 35|   management| single| tertiary|    747|
        | 36|self-employed|married| tertiary|    307|
        | 39|   technician|married|secondary|    147|
        | 41| entrepreneur|married| tertiary|    221|
        | 43|     services|married|  primary|    -88|
        +---+-------------+-------+---------+-------+
        only showing top 10 rows
```

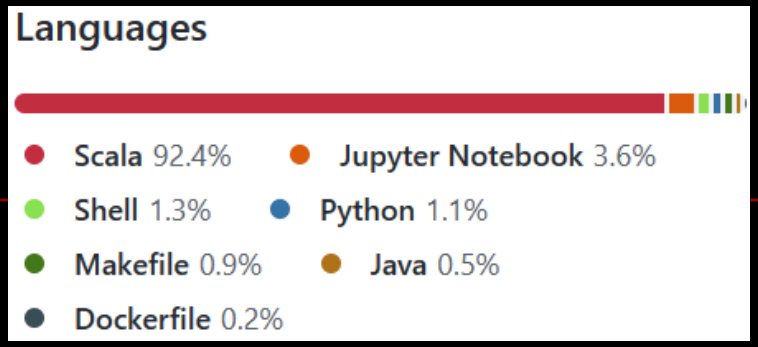
```
In [10]: %%dataframe
         customers
```

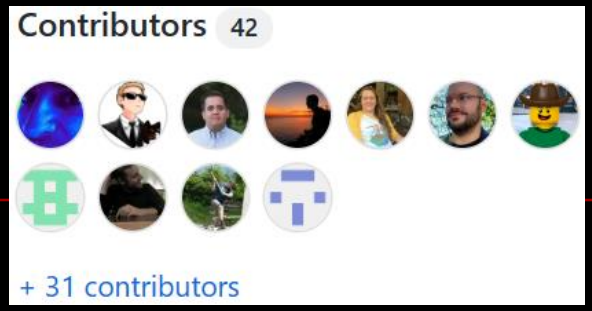
Out[10]:

age	job	marital	education	balance
30	unemployed	married	primary	1787
33	services	married	secondary	4789
35	management	single	tertiary	1350
30	management	married	tertiary	1476
59	blue-collar	married	secondary	0
35	management	single	tertiary	747
36	self-employed	married	tertiary	307
39	technician	married	secondary	147
41	entrepreneur	married	tertiary	221
43	services	married	primary	-88

https://github.com/apache/incubator-toree/blob/master/etc/examples/notebooks/magic-tutorial.ipynb

- **https://github.com/apache/incubator-toree**





Overview

Toree provides an interface that allows clients to interact with a Spark Cluster. Clients can send libraries and snippets of code that are interpreted and executed using a preconfigured Spark context. These snippets can do a variety of things:

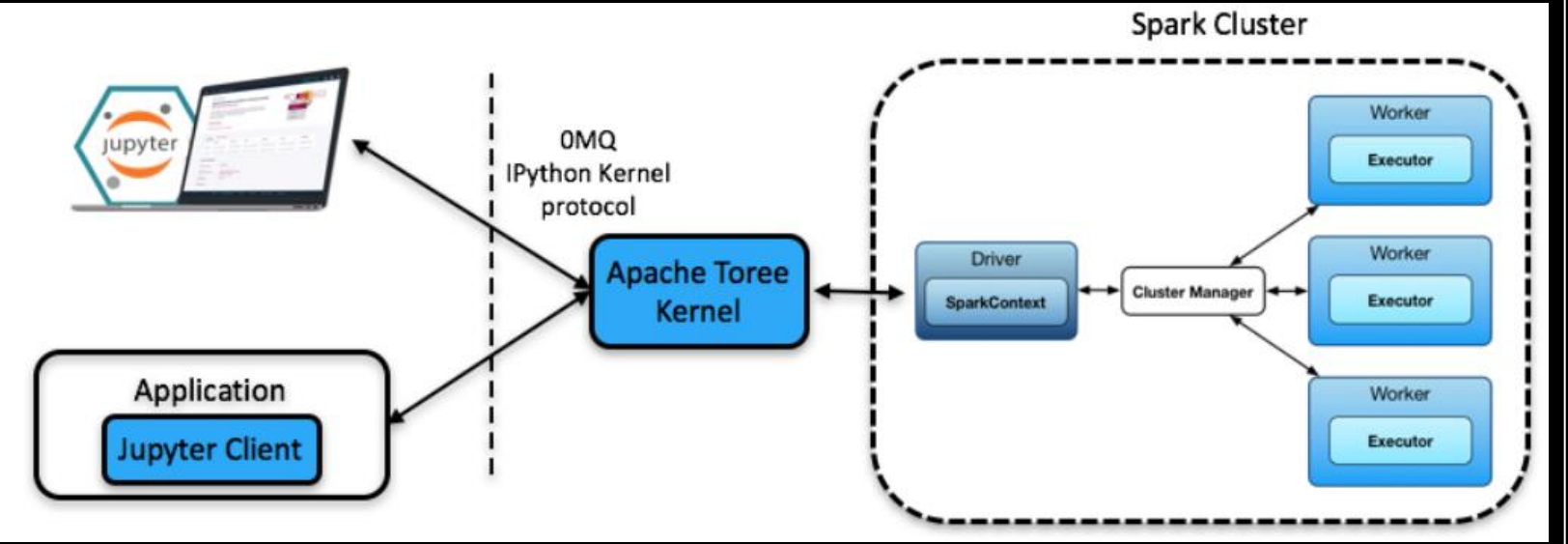
1. Define and run spark jobs of all kinds
2. Collect results from spark and push them to the client
3. Load necessary dependencies for the running code
4. Start and monitor a stream
5. ...

Apache Toree supports the `Scala` programming language. It implements the latest Jupyter message protocol (5.0), so it can easily plug into the latest releases of Jupyter/IPython (3.2.x+ and up) for quick, interactive data exploration.

Use Cases

- **https://toree.apache.org/**

Toree supports a number of interaction scenarios. In one common case, applications send snippets of code which are then executed by Spark, and the results are returned directly to the application. This style of interaction is what users of Notebooks experience when they evaluate code in cells. Instead of sending raw code, an application can send magics, which might be commands to add a JAR to the Spark execution context or a call to execute a shell command such as “ls”. Toree provides a well-defined mechanism to associate functionality with magics, and this is a useful point of extensibility of the system.



Applications wanting to work with Spark can be located remotely from a Spark cluster and use a Apache Toree Client or Jupyter Client to communicate with a Apache Toree Server running on the cluster, or they can communicate directly with the Apache Toree Server. Multiple clients/applications can communicate with a single Kernel which contains a Spark application context, and this provides a simple form of multi-tenancy.

- **…**

4) Ray.Graal

4.1 Our new scheme

- A GraalVM-oriented reimplementation of Ray by Python + Java/Scala to instead of Python + C++ in current implementation;
- If you’re interested, you may look forward to our new talk "First exploration of Ray.Graal".

5) Cross-platform integration for Polyglot Programming

5.1 Ongoing

- Continuously updating "Cross-platform integration for Polyglot Programming"

  (the latest slides is put at:

  https://github.com/XianBeiTuoBaFeng2015/MySlides/blob/master/LTS/Cross-platform%20integration%20for%20Polyglot%20Programming__20230114p.pdf)

V. Wrap-up

- Scala plays an important role in HW-SW collaboration!
- Notebook-based interactive computing is changing the way we think about development.
- More IDE and Cluster relate features and more will be integrated into Notebook.
- Cross-platform ecological integration for Polyglot programming is the future.
- How about a new design and implementation of a GraalVM-based Polyglot Notebook that brings together the best of Polynote, Almond, Vizier, Toree, Zeppelin, etc to better support both Python and Scala?

Q & A

Thanks!

Reference

Slides/materials from many and varied sources:

- http://en.wikipedia.org/wiki/
- http://www.slideshare.net/
- https://www.handsonscala.com/
- https://mybinder.org/
- https://queirozf.com/entries/jupyter-kernels-how-to-add-change-remove
- https://www.fullstackpython.com/jupyter-notebook.html
- https://github.com/jupyterlab/retrolab
- https://www.geeksforgeeks.org/how-to-install-scala-kernal-in-jupyter/
- https://thenewstack.io/how-apache-arrow-is-changing-the-big-data-ecosystem/
- https://www.projectpro.io/article/scala-vs-python-for-apache-spark/213
- https://github.com/Yogayu/awesome-jupyterlab-extension
- https://github.com/ml-tooling/best-of-jupyter
- https://www.graalvm.org/22.3/reference-manual/python/Interoperability/
- https://github.com/almond-sh/examples/blob/master/notebooks/interactive_computing_article.ipynb
- …