Akademy 2024

The Role of Language Bindings: Pythonizing Qt

Dr. Cristián Maureira-Fredes
@cmaureir

Akademy 2024

Why Akademy?

Cristián Maureira-Fredes | @cmaureir

blogs.kde.org/2010/10/27/tale-chile/

Cristián Maureira-Fredes | @cmaureir

Chilean Linux Meeting 2010

Cristián Maureira-Fredes | @cmaureir

btw I use arch since 2009

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Disclaimer

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Qt Language Bindings

Exposing Qt to other languages is nothing new.

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Qt Language Bindings

Binding experience, is also Qt experience.

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Name	Language	Latest version	Last commit	License	Company

(updated on Sep 2nd, 2024)

Other 26 projects (some inactive) in wiki.qt.io/Language_Bindings

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Name	Language	Latest version	Last commit	License	Company
PySide	Python	6.7.2	-	LGPLv3/Commercial	The Qt Company
PyQt	Python	6.7.2	-	GPLv3/Commercial	Riverbank Computing

(updated on Sep 2nd, 2024)

Other 26 projects (some inactive) in wiki.qt.io/Language_Bindings

Akademy 2024

Name	Language	Latest version	Last commit	License	Company
PySide	Python	6.7.2	-	LGPLv3/Commercial	The Qt Company
PyQt	Python	6.7.2	-	GPLv3/Commercial	Riverbank Computing
CXX-Qt	Rust	6.x*	-	MIT/Apache 2.0	KDAB
qmetaobject-rs	Rust	6.5.0	2024.08.07	MIT	Woboq

(updated on Sep 2nd, 2024)

Other 26 projects (some inactive) in wiki.qt.io/Language_Bindings

Akademy 2024

Name	Language	Latest version	Last commit	License	Company
PySide	Python	6.7.2	-	LGPLv3/Commercial	The Qt Company
PyQt	Python	6.7.2	-	GPLv3/Commercial	Riverbank Computing
CXX-Qt	Rust	6.x*	-	MIT/Apache 2.0	KDAB
qmetaobject-rs	Rust	6.5.0	2024.08.07	MIT	Woboq
QtJambi	Java/Kotlin	6.7.2	2024.06.25	LGPLv2	Omix Visualization

(updated on Sep 2nd, 2024)

Other 26 projects (some inactive) in wiki.qt.io/Language_Bindings

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Name	Language	Latest version	Last commit	License	Company
PySide	Python	6.7.2	-	LGPLv3/Commercial	The Qt Company
PyQt	Python	6.7.2	-	GPLv3/Commercial	Riverbank Computing
CXX-Qt	Rust	6.x*	-	MIT/Apache 2.0	KDAB
qmetaobject-rs	Rust	6.5.0	2024.08.07	MIT	Woboq
QtJambi	Java/Kotlin	6.7.2	2024.06.25	LGPLv2	Omix Visualization
QML-zig	Zig	5.15.2	2024.08.11	Apache 2.0	-

(updated on Sep 2nd, 2024)

Other 26 projects (some inactive) in wiki.qt.io/Language_Bindings

Akademy 2024

Name	Language	Latest version	Last commit	License	Company
PySide	Python	6.7.2	-	LGPLv3/Commercial	The Qt Company
PyQt	Python	6.7.2	-	GPLv3/Commercial	Riverbank Computing
CXX-Qt	Rust	6.x*	-	MIT/Apache 2.0	KDAB
qmetaobject-rs	Rust	6.5.0	2024.08.07	MIT	Woboq
QtJambi	Java/Kotlin	6.7.2	2024.06.25	LGPLv2	Omix Visualization
QML-zig	Zig	5.15.2	2024.08.11	Apache 2.0	-
QML.jl	Julia	6.5.2	2024.07.12	MIT	JuliaGraphs

(updated on Sep 2nd, 2024)

Other 26 projects (some inactive) in wiki.qt.io/Language_Bindings

Akademy 2024

Name	Language	Latest version	Last commit	License	Company
PySide	Python	6.7.2	-	LGPLv3/Commercial	The Qt Company
PyQt	Python	6.7.2	-	GPLv3/Commercial	Riverbank Computing
CXX-Qt	Rust	6.x*	-	MIT/Apache 2.0	KDAB
qmetaobject-rs	Rust	6.5.0	2024.08.07	MIT	Woboq
QtJambi	Java/Kotlin	6.7.2	2024.06.25	LGPLv2	Omix Visualization
QML-zig	Zig	5.15.2	2024.08.11	Apache 2.0	-
QML.jl	Julia	6.5.2	2024.07.12	MIT	JuliaGraphs
RingQt	Ring	5.15.15	2024.09.02	MIT	-
NodeGui	Node.js	6.6.0	2024.05.31	MIT	NodeGui
nimqt	Nim	6.4.3	2024.01.27	GPL2	-

(updated on Sep 2nd, 2024)

Other 26 projects (some inactive) in wiki.qt.io/Language_Bindings

Akademy 2024

Most bindings expose only 1:1 API

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

What about improving the Qt ecosystem?

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

...or at least offering a little bit more.

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

The case of Python

Qt Cristián Maureira-Fredes | @cmaureir

TIOBE Index for August 2024

August Headline: Python is chasing Java's TIOBE index records

This month, Python has a ranking of more than 18% for the first time in its history. The last time a language hit more than 18% was Java in November 2016. Java is also the language with the highest ranking ever: 26.49% in June 2001. Runner up C++ is now exactly 8% behind Python, and that difference between position #1 and position #2 is also almost a record. The highest difference ever between position #1 and position #2 was in November 2016 when Java was 9.55% ahead of C. In summary, Python's hegemony is now undeniable. It is likely that it is Python's next step to become the most popular programming language ever. Is there any new language expected to come close to Python soon? Possible contenders Rust and Kotlin are approaching the TIOBE index top 10 fast, but it will take a lot of time before they become a real threat to Python. *--Paul Jansen CEO TIOBE Software*

The TIOBE Programming Community index is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the number of skilled engineers world-wide, courses and third party vendors. Popular web sites Google, Amazon, Wikipedia, Bing and more than 20 others are used to calculate the ratings. It is important to note that the TIOBE index is not about the *best* programming language or the language in which *most lines of code* have been written.

The index can be used to check whether your programming skills are still up to date or to make a strategic decision about what programming language should be adopted when starting to build a new software system. The definition of the TIOBE index can be found here.

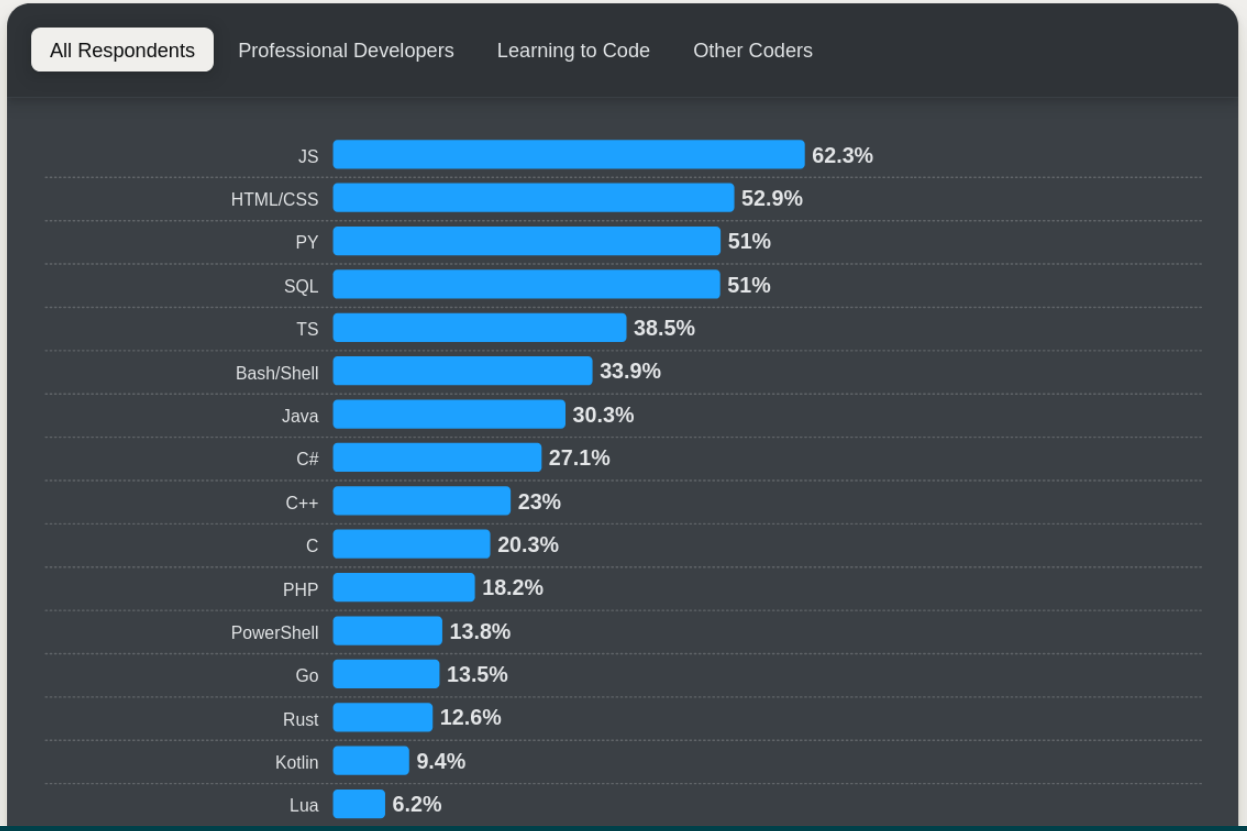
Aug 2024	Aug 2023	Change	Programming Language	Ratings	Change
1	1		Python	18.04%	+4.71%
2	3	↑	C++	10.04%	-0.59%
3	2	↓	C	9.17%	-2.24%
4	4		Java	9.16%	-1.16%
5	5		C#	6.39%	-0.65%
6	6		JavaScript	3.91%	+0.62%
7	8	↑	SQL	2.21%	+0.68%
8	7	↓	Visual Basic	2.18%	-0.45%
9	12	↑	Go	2.03%	+0.87%
10	14	⇈	Fortran	1.79%	+0.75%

Most popular technologies 2.1

Programming, scripting, and markup languages

JavaScript has been a mainstay in the developer survey and on Stack Overflow since our first survey. The most popular programming language has been JavaScript every year we have done the survey except for 2013 and 2014, when SQL was the most popular language.

Which **programming, scripting, and markup languages** have you done extensive development work in over the past year, and which do you want to work in over the next year? (If you both worked with the language and want to continue to do so, please check both boxes in that row.)



StackOverflow Survey 2024 Results

Akademy 2024

You can understand

why Python with a few things

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Python's syntax

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QPushButton>

class MainWindow : public QMainWindow
{
    Q_OBJECT
    public:
        MainWindow(QWidget *parent = nullptr);
    private slots:
        void handleButton();
    private:
        QPushButton *m_button;
};

#endif // MAINWINDOW_H
```

```
#include "mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
   : QMainWindow(parent)
{
    m_button = new QPushButton("My Button", this);
    connect(m_button, SIGNAL(clicked()), this,
      SLOT(handleButton()));
}

void MainWindow::handleButton()
{
    m_button->setText("Ready");
}
```

```
#include <QApplication>
#include "mainwindow.h"

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);
    MainWindow mainWindow;
    mainWindow.show();
    return app.exec(d);
}
```

```
// don't forget the CMakeLists.txt
```

Akademy 2024

Python's syntax

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QPushButton>

class MainWindow : public QMainWindow
{
    Q_OBJECT
    public:
        MainWindow(QWidget *parent = nullptr);
    private slots:
        void handleButton();
    private:
        QPushButton *m_button;
};

#endif // MAINWINDOW_H
```

```
#include "mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
   : QMainWindow(parent)
{
    m_button = new QPushButton("My Button", this);
    connect(m_button, SIGNAL(clicked()), this,
      SLOT(handleButton()));
}

void MainWindow::handleButton()
{
    m_button->setText("Ready");
}
```

```
#include <QApplication>
#include "mainwindow.h"

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);
    MainWindow mainWindow;
    mainWindow.show();
    return app.exec(d);
}
```

```
// don't forget the CMakeLists.txt
```

```
import sys
from PySide6.QtCore import Slot
from PySide6.QtWidgets import (
  QApplication, QMainWindow, QPushButton
)

class MainWindow(QMainWindow):
    def __init__(self, parent=None):
        QMainWindow.__init__(self, parent)
        self.b = QPushButton("My Button", self)
        self.b.clicked.connect(self.handle_button)

    @Slot()
    def handle_button(self):
        self.b.setText("Ready")

if __name__ == "__main__":
    app = QApplication(sys.argv)
    mainWindow = MainWindow()
    mainWindow.show()
    sys.exit(app.exec())
```

Akademy 2024

Python's heterogeneity

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Python's heterogeneity

Web Development

FastAPI

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Python's heterogeneity

Web Development

Data Science

PyTorch

FastAPI

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Python's heterogeneity

Web Development

Data Science

Embedded systems

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Python: package distribution

```
~ pip install PySide6
Collecting PySide6
  Using cached PySide6-6.6.0-cp38-abi3-manylinux_2_28_x86_64.whl.metadata (5.1 kB)
Collecting shiboken6==6.6.0 (from PySide6)
  Using cached shiboken6-6.6.0-cp38-abi3-manylinux_2_28_x86_64.whl.metadata (2.3 kB)
Collecting PySide6-Essentials==6.6.0 (from PySide6)
  Using cached PySide6_Essentials-6.6.0-cp38-abi3-manylinux_2_28_x86_64.whl.metadata (3.5
 kB)
Collecting PySide6-Addons==6.6.0 (from PySide6)
  Using cached PySide6_Addons-6.6.0-cp38-abi3-manylinux_2_28_x86_64.whl.metadata (3.7 kB)
Using cached PySide6-6.6.0-cp38-abi3-manylinux_2_28_x86_64.whl (6.7 kB)
Using cached PySide6_Addons-6.6.0-cp38-abi3-manylinux_2_28_x86_64.whl (125.2 MB)
Using cached PySide6_Essentials-6.6.0-cp38-abi3-manylinux_2_28_x86_64.whl (82.1 MB)
Using cached shiboken6-6.6.0-cp38-abi3-manylinux_2_28_x86_64.whl (169 kB)
Installing collected packages: shiboken6, PySide6-Essentials, PySide6-Addons, PySide6
Successfully installed PySide6-6.6.0 PySide6-Addons-6.6.0 PySide6-Essentials-6.6.0 shibok
en6-6.6.0
~ pip list | grep -i py
```

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

qtpip

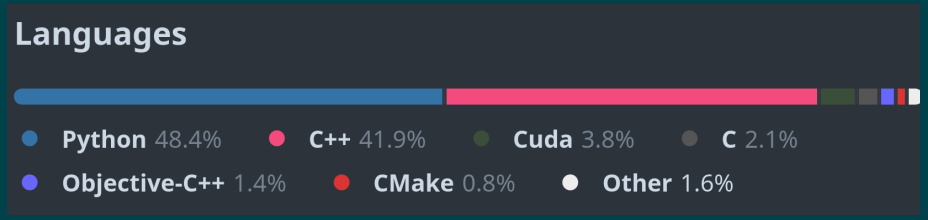
```
qt ~(env) % pip install qtpip
Collecting qtpip
  Downloading qtpip-0.1-py3-none-manylinux_2_28_x86_64.whl.metadata (975 bytes)
Downloading qtpip-0.1-py3-none-manylinux_2_28_x86_64.whl (4.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.9/4.9 MB 19.3 MB/s eta 0:00:00
Installing collected packages: qtpip
Successfully installed qtpip-0.1
qt ~(env) % qtpip install pyside6
[INFO] Commercial License found
[INFO] Starting downloading wheels
[INFO] Downloading wheels (1/5) : "6.6.0-202310171251PySide6-6.6.0.commercial-cp38
-abi3-manylinux_2_28_x86_64.whl.7z"
[INFO] Downloading wheels (2/5) : "6.6.0-202310171251shiboken6-6.6.0.commercial-cp
38-abi3-manylinux_2_28_x86_64.whl.7z"
[INFO] Downloading wheels (3/5) : "6.6.0-202310171251PySide6_M2M-6.6.0.commercial-
cp38-abi3-manylinux_2_28_x86_64.whl.7z"
Downloading ( 83.79%)
```

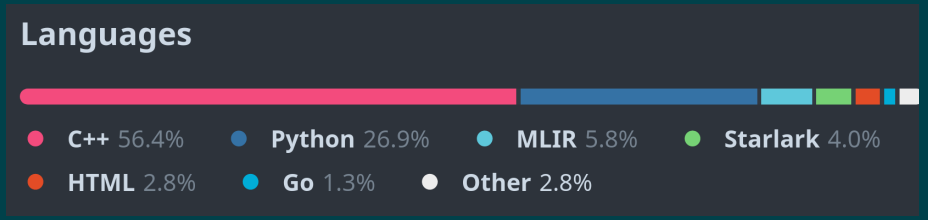
Cristián Maureira-Fredes | @cmaurer

qt.io/blog/qtpip-qtforpython-installer

Akademy 2024

Machine Learning popularity





PyTorch

TensorFlow

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

We got the perfect recipe 🏆

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

We got the perfect recipe 🏆

A Python user-facing experience with a C++ core.

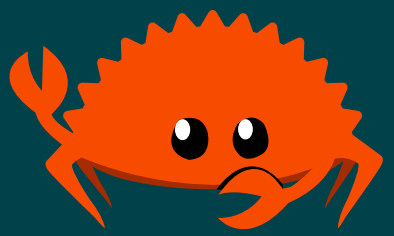
Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

We got the perfect recipe 🏆

A Python user-facing experience with a C++ core.

What about...



Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Let's go back to Python and Qt

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

PyQt & PySide

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

PySide

PyQt

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

PySide

- LGPLv3/Commercial

PyQt

- GPLv3/Commercial

Akademy 2024

PySide

- LGPLv3/Commercial
- Developed by The Qt Company

PyQt

- GPLv3/Commercial
- Developed by Riverbank Co.

Akademy 2024

PySide

- LGPLv3/Commercial
- Developed by The Qt Company
- Support Windows, macOS and Linux.

PyQt

- GPLv3/Commercial
- Developed by Riverbank Co.
- Support Windows, macOS and Linux.

Akademy 2024

PySide

- LGPLv3/Commercial
- Developed by The Qt Company
- Support Windows, macOS and Linux.
- Support for Android + Deployment

PyQt

- GPLv3/Commercial
- Developed by Riverbank Co.
- Support Windows, macOS and Linux.
- Support for Android + Deployment

Akademy 2024

PySide

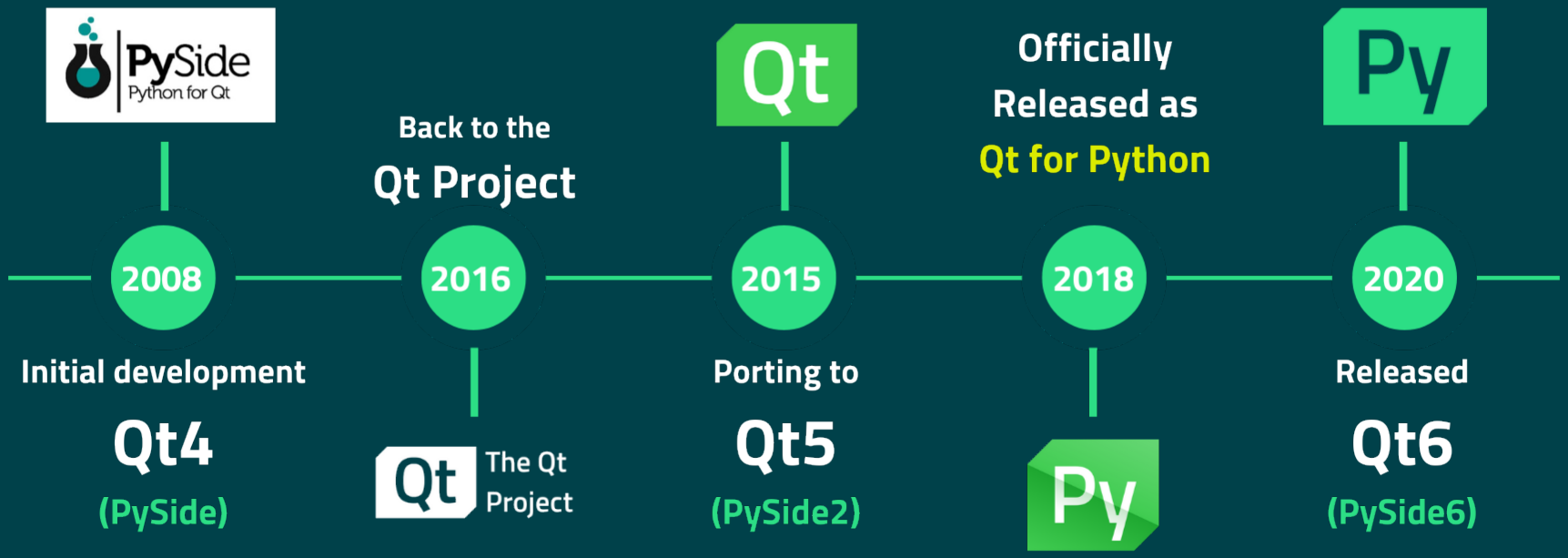
- LGPLv3/Commercial
- Developed by The Qt Company
- Support Windows, macOS and Linux.
- Support for Android + Deployment
- Exposes Qt API and more.

PyQt

- GPLv3/Commercial
- Developed by Riverbank Co.
- Support Windows, macOS and Linux.
- Support for Android + Deployment
- Exposes Qt API.

The story of PySide

Akademy 2024



Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

For PySide

exposing the Qt API (like PyQt)

is not enough

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Searching for new features

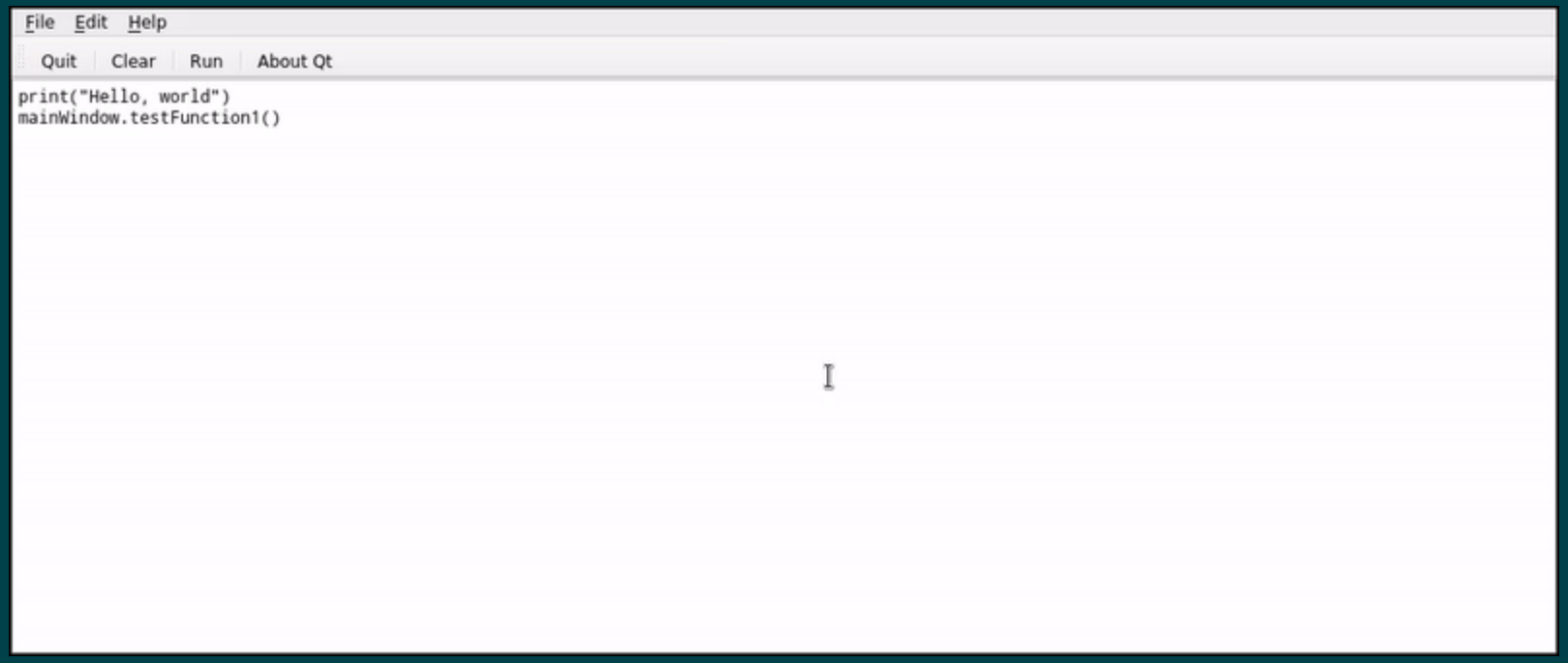
Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

1. Embedding the Python Interpreter

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024



scriptableapplication example

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

2. NumPy support

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

numpy.org The fundamental package for scientific computing with Python

```
>>> from PySide6.QtGui import QMatrix4x4
>>> import numpy as np
>>> m = np.eye(4)
>>> m.shape = 16
>>> m
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
>>> QMatrix4x4(m)
PySide6.QtGui.QMatrix4x4((1, 0, 0, 0, 0, 1, 0, 0,
```

```
// Also ad-hoc API
QPainter.drawPointsNp(PyArrayObject *x, PyArrayObject *y);

QXYSeries.appendNp(PyArrayObject *x, PyArrayObject *y);
QXYSeries.replaceNp(PyArrayObject *x, PyArrayObject *y);
```

Akademy 2024

3. Qt Creator support

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

doc.qt.io/qtcreator/creator-python-development.html

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

That was not enough 😒

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Adding Python-ish features

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

4. snake_case

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

```
# Common Qt structure
# - Using setter/getter


table = QTableWidget()
table.setColumnCount(2)

button = QPushButton("Add")
button.setEnabled(False)

layout = QVBoxLayout()
layout.addWidget(table)
layout.addWidget(button)
```

Akademy 2024

```
# Common Qt structure
# - Using setter/getter


table = QTableWidget()
table.setColumnCount(2)

button = QPushButton("Add")
button.setEnabled(False)

layout = QVBoxLayout()
layout.addWidget(table)
layout.addWidget(button)
```

```
from __feature__ import (
  snake_case
)

table = QTableWidget()
table.set_column_count(2)

button = QPushButton("Add")
button.set_enabled(False)

layout = QVBoxLayout()
layout.add_widget(table)
layout.add_widget(button)
```

Akademy 2024

```
# Common Qt structure
# - Using setter/getter


table = QTableWidget()
table.setColumnCount(2)

button = QPushButton("Add")
button.setEnabled(False)

layout = QVBoxLayout()
layout.addWidget(table)
layout.addWidget(button)
```

```
from __feature__ import (
  snake_case
)

table = QTableWidget()
table.set_column_count(2)

button = QPushButton("Add")
button.set_enabled(False)

layout = QVBoxLayout()
layout.add_widget(table)
layout.add_widget(button)
```

It might look strange to you, C++ developer.

Akademy 2024

5. True properties

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

```
# Common Qt structure
# - Using setter/getter
# - No writable properties

table = QTableWidget()
table.setColumnCount(2)

button = QPushButton("Add")
button.setEnabled(False)

layout = QVBoxLayout()
layout.addWidget(table)
layout.addWidget(button)
```

```
from __feature__ import (
  snake_case, true_property
)

table = QTableWidget()
table.column_count = 2

button = QPushButton("Add")
button.enabled = False

layout = QVBoxLayout()
layout.add_widget(table)
layout.add_widget(button)
```

```
# Common Qt structure
# - Using setter/getter
# - No writable properties

table = QTableWidget()
table.setColumnCount(2)

button = QPushButton("Add")
button.setEnabled(False)

layout = QVBoxLayout()
layout.addWidget(table)
layout.addWidget(button)
```

```
from __feature__ import (
  snake_case, true_property
)

table = QTableWidget()
table.column_count = 2

button = QPushButton("Add")
button.enabled = False

layout = QVBoxLayout()
layout.add_widget(table)
layout.add_widget(button)
```

This was the #1 requested feature

Akademy 2024

6. QmlElement

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

```python
from PySide6.QtQml import QmlElement
# ...

QML_IMPORT_NAME = "io.qt.some_name"
QML_IMPORT_MAJOR_VERSION = 1


@QmlElement
class Bridge(QObject):

    @Slot(str, result=str)
    def getColor(self, s):
        # ...
```

Akademy 2024

```python
from PySide6.QtQml import QmlElement
# ...

QML_IMPORT_NAME = "io.qt.some_name"
QML_IMPORT_MAJOR_VERSION = 1


@QmlElement
class Bridge(QObject):

    @Slot(str, result=str)
    def getColor(self, s):
        # ...
```

```qml
// ...
import io.qt.some_name
// ...
Bridge {
    id: bridge
}

Button {
  id: red
  text: "Red"
  highlighted: true
  Material.accent: Material.Red
  onClicked: {
    label.color = bridge.getColor(red.text)
  }
}
// ...
```

Akademy 2024

7. Tooling

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

```
# Wrappers
pyside6-assistant
pyside6-designer
pyside6-linguist
pyside6-lrelease
pyside6-lupdate
pyside6-qml
pyside6-qmlcachegen
pyside6-qmlimportscanner
pyside6-qmllint
pyside6-qmlsc
pyside6-qmltyperegistrar
pyside6-rcc
pyside6-uic
```

```
# Utilities
pyside6-genpyi
pyside6-metaobjectdump
```

```
# New tools
pyside6-project
pyside6-deploy
pyside6-android-deploy
```

doc.qt.io/qtforpython-6/tools/index.html

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Moaaaaar features!

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

8. Android support

Cristián Maureira-Fredes | @cmaureir

- Embrace a community project python-for-android
- Rely on the Qt/C++ scope (QtQuick)
- Automate the most from the process (one command line)

Akademy 2024

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

9. QtAsyncio

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

QtAsyncio

```python
# ...
import sys
import asyncio
import PySide6.QtAsyncio as QtAsyncio

class MainWindow(QMainWindow):
  def __init__(self):
    super().__init__()
    widget = QWidget()
    self.setCentralWidget(widget)
    layout = QVBoxLayout(widget)
    self.text = QLabel("The answer is 42.")
    layout.addWidget(self.text, alignment=Qt.AlignmentFlag.AlignCenter)

    async_trigger = QPushButton(text="What is the question?")
    async_trigger.clicked.connect(lambda: asyncio.ensure_future(self.set_text()))
    layout.addWidget(async_trigger, alignment=Qt.AlignmentFlag.AlignCenter)

  async def set_text(self):
    await asyncio.sleep(1)
    self.text.setText("What do you get if you multiply six by nine?")
```

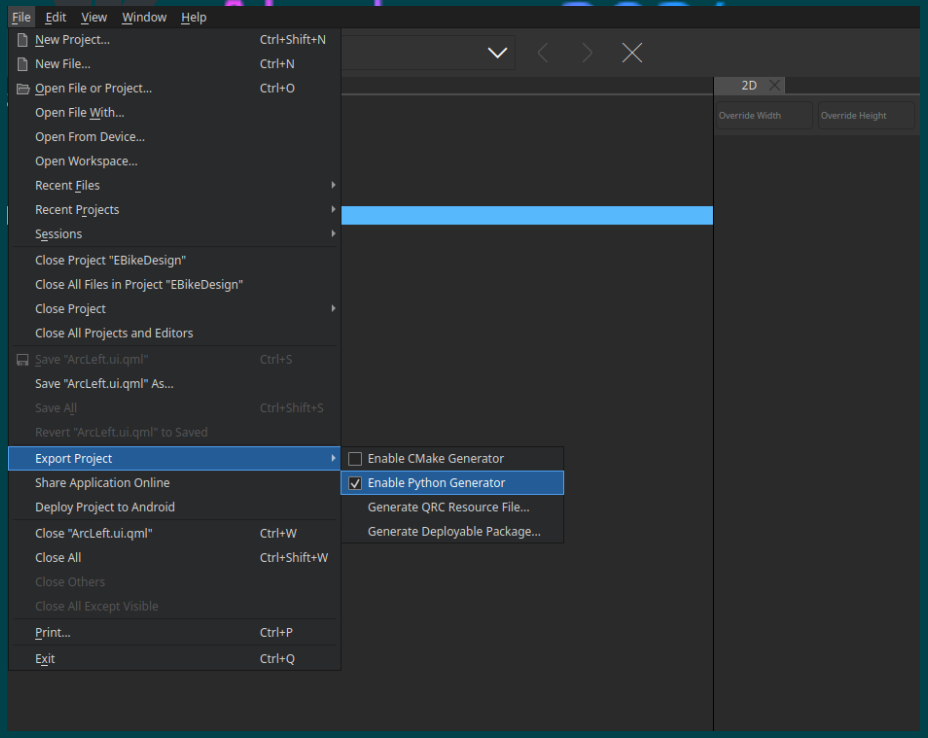
```python
if __name__ == "__main__":
    app = QApplication(sys.argv)
    main_window = MainWindow()
    main_window.show()

    QtAsyncio.run(handle_sigint=True)
```

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

10. Qt Design Studio

Cristián Maureira-Fredes | @cmaureir




```
% tree Python
Python
├── autogen
│   └── settings.py
└── main.py

2 directories, 2 files
```

```python
# Python/autogen/settings.py

url = "content/App.qml"
import_paths = [
        ".",
        "imports",
        "asset_imports",
]
```

```python
# Python/main.py
# ...
from autogen.settings import url, import_paths

if __name__ == '__main__':
    app = QGuiApplication(sys.argv)
    engine = QQmlApplicationEngine()

    app_dir = Path(__file__).parent.parent

    engine.addImportPath(os.fspath(app_dir))
    for path in import_paths:
        engine.addImportPath(os.fspath(app_dir / path))

    engine.load(os.fspath(app_dir/url))
    if not engine.rootObjects():
        sys.exit(-1)
    sys.exit(app.exec())
```

⭐ To be announced this month.

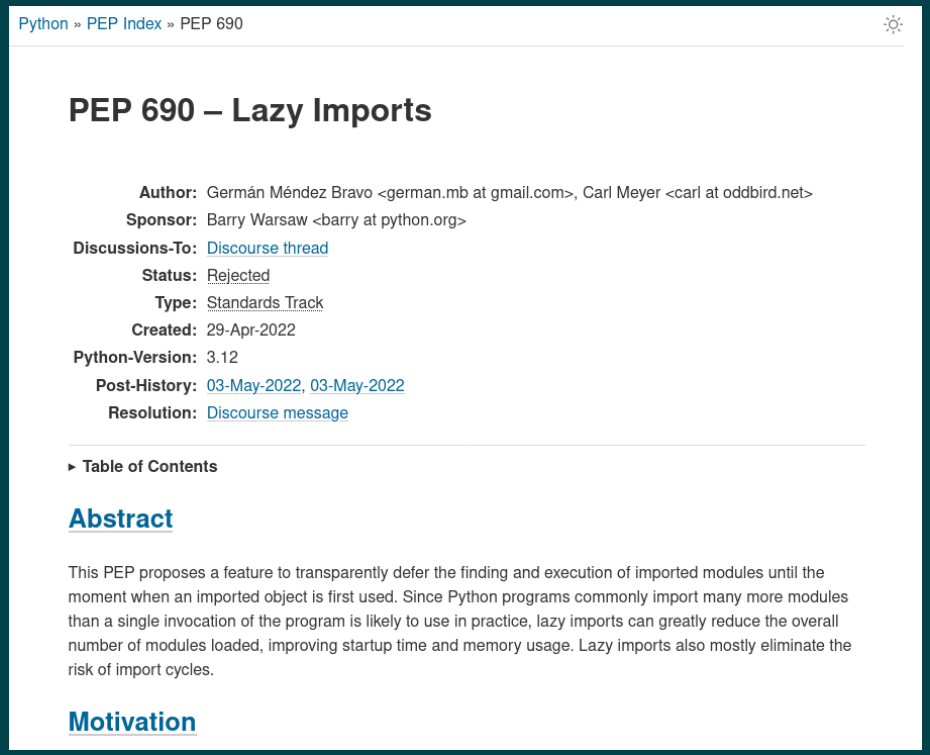
Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

11. Lazy loading

Cristián Maureira-Fredes | @cmaureir

Akademy 2024



Python » PEP Index » PEP 690

PEP 690 – Lazy Imports

Author: Germán Méndez Bravo <german.mb at gmail.com>, Carl Meyer <carl at oddbird.net>
Sponsor: Barry Warsaw <barry at python.org>
Discussions-To: Discourse thread
Status: Rejected
Type: Standards Track
Created: 29-Apr-2022
Python-Version: 3.12
Post-History: 03-May-2022, 03-May-2022
Resolution: Discourse message

▸ Table of Contents

Abstract

This PEP proposes a feature to transparently defer the finding and execution of imported modules until the moment when an imported object is first used. Since Python programs commonly import many more modules than a single invocation of the program is likely to use in practice, lazy imports can greatly reduce the overall number of modules loaded, improving startup time and memory usage. Lazy imports also mostly eliminate the risk of import cycles.

Motivation

Improve startup time of the PySide modules

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

12. QtScript...but Python

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

```
static Demo demos[] = {
    {"Main Window",
     R"(# main_win is bound from C++
print("name=", main_win.objectName)
# point is bound from C++
print("point.x,y=", point.x(), point.y())

# Play with properties
main_win.objectName = "TestName"
print("changed name=", main_win.objectName)

# Create a point in Python
python_point = QPoint()
print("python_point.x,y=", python_point.x(),
    python_point.y())
python_point.setX(500)
python_point.setY(500)
)"},
};
```

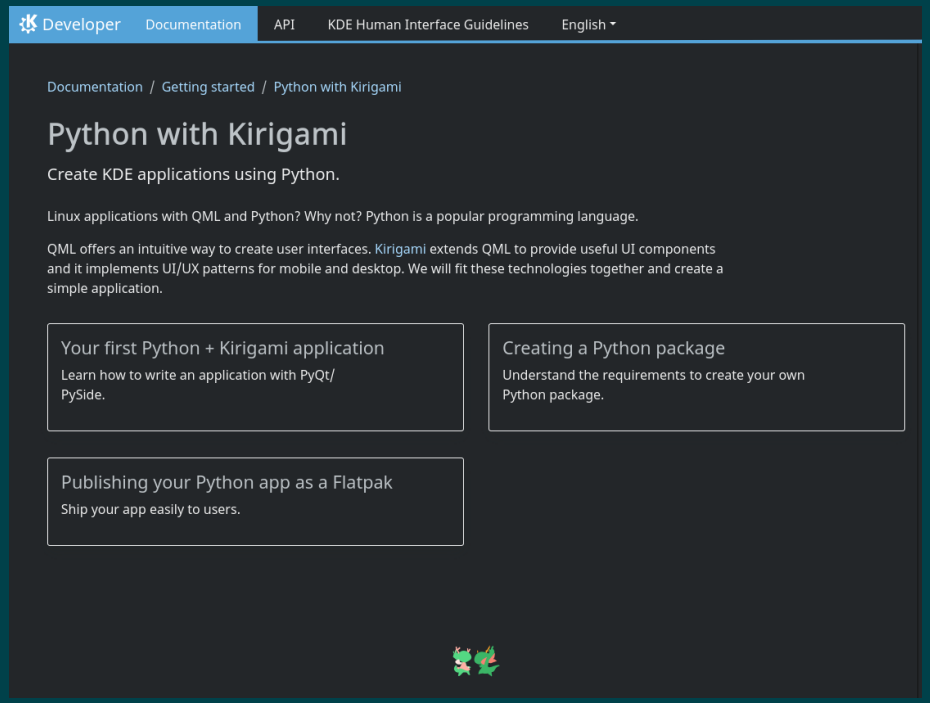
- Python within C++
- Embed the Python interpreter
- Dynamical bindings
- Bind the Qt/C++ application to the interpreter

Akademy 2024

13. Flatpak support

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024



- Enabling users to create Python apps for KDE
- PyQt and PySide are compatible
  - github.com/flathub/io.qt.PySide.BaseApp

develop.kde.org/docs/getting-started/python/

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

...since we are talking about KDE

Cristián Maureira-Fredes | @cmaureir

2024 Program | KDE Community

Contributor

Manuel Alcaraz

Python bindings for KDE Frameworks

Mentors	Organization	Technologies	Topics
Carl Schwan	KDE Community	python, c++, qt, cmake	desktop, ui

KDE Frameworks is a collection of C++ addon libraries to Qt that provide common functionality for desktop applications built with the Qt framework. While Qt itself provides support for developing applications using other programming languages, such as Python, KDE Frameworks does not. This project aims to create Python bindings (along with comprehensive developer documentation) for a subset of the KDE Frameworks using the same technologies (Shiboken) that Qt uses for their Python port.

Merge request: Add Python bindings

Akademy 2024

Is that enough 😰?

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

(

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

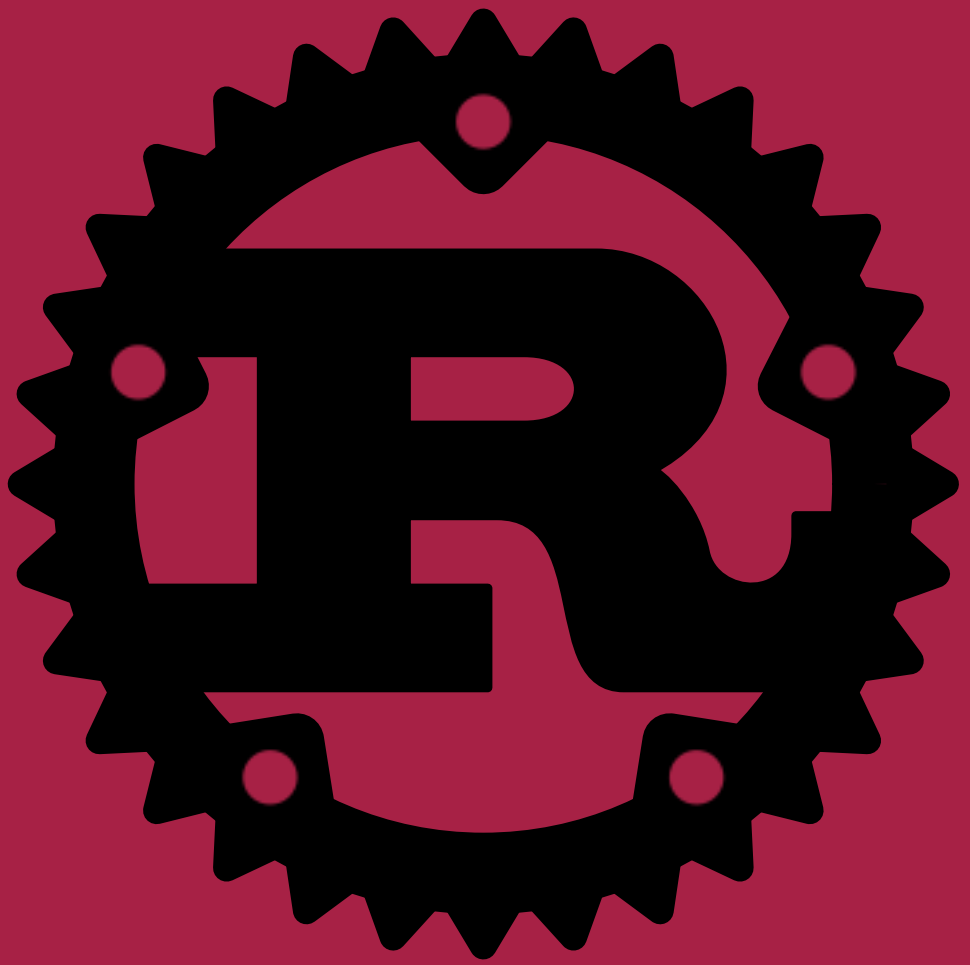
Re-writing Python code into a compiled language is faster

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Re-writing Python code into a compiled language is faster

Cristián Maureira-Fredes | @cmaureir



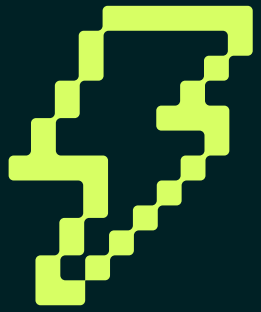
Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

ruff

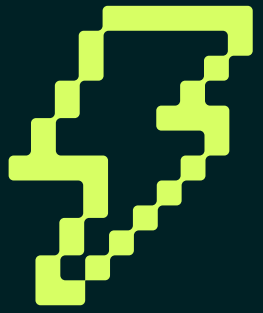


ruff.astral.sh

Qt Cristián Maureira-Fredes | @cmaureir

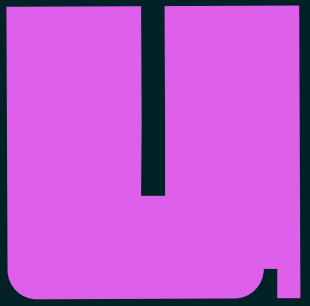
Akademy 2024

ruff



ruff.astral.sh

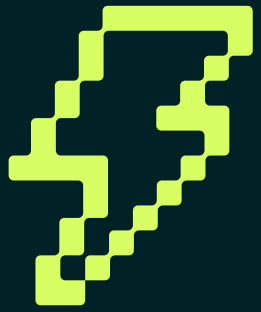
uv



uv.astral.sh

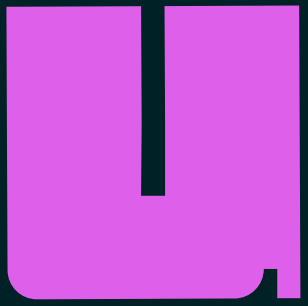
Qt Cristián Maureira-Fredes | @cmaureir

ruff



ruff.astral.sh

uv



uv.astral.sh

rye

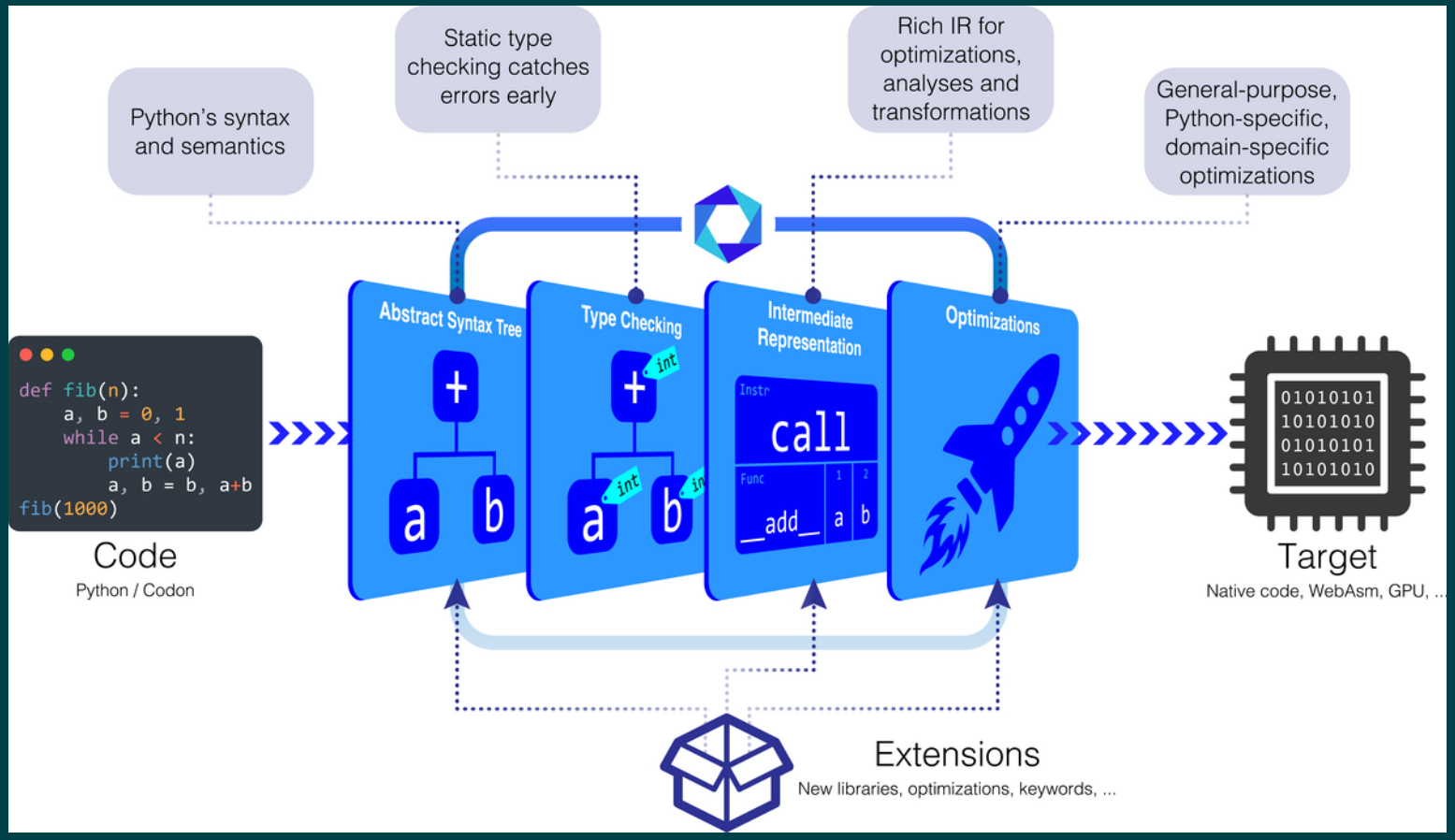
rye.astral.sh

Qt Cristián Maureira-Fredes | @cmaureir

A different-approach

Akademy 2024

Codon



docs.exaloop.io/codon

Akademy 2024

)

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

The future of the project

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

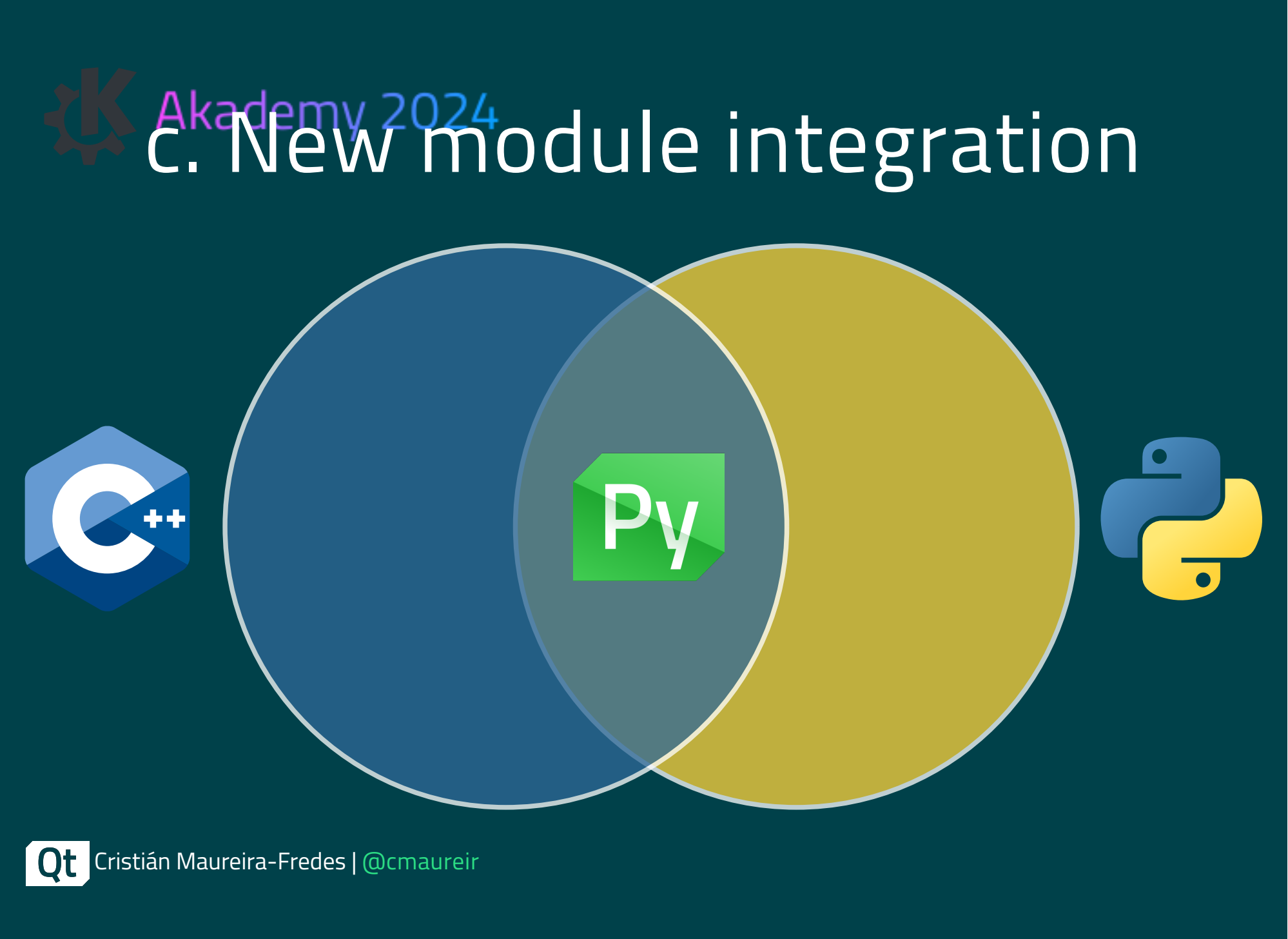
a. Simplifying User-facing experience

```python
import kimyou
from kimyou import Window, TextField, Button

def main(window: Window):
    window.title = "Testing title"
    window.layout_direction = "vertical"
    name = TextField(value="Enter your name")
    button = Button(text="Click me!",
        on_click=lambda: print(f"Hello {name.value}"))
    button2 = Button(text="Exit", on_click=window.exit)
    # Vertical
    #window.add(name)
    #window.add(button)
    #window.add(button2)
    # Horizontal
    window.add([name, button, button2])

kimyou.app(target=main)
```

Akademy 2024

b. Packaging and Tooling

Qt Cristián Maureira-Fredes | @cmaureir


Akademy 2024
c. New module integration
C++
Py
Qt
Cristián Maureira-Fredes | @cmaureir

Akademy 2024

d. New Python-only Qt modules

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

And other features requested by ~~users~~ you

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

What's next?

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Embrace the idea of a many-language framework

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Imagine everything we can learn from other languages

Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Be boring and

see what works for people

Qt Cristián Maureira-Fredes | @cmaureir

Akademy 2024

Q&A

The Role of Language Bindings: Pythonizing Qt

Dr. Cristián Maureira-Fredes
@cmaureir