Python in the browser

Mateusz Nowak

June 29 2022

🚫 We won't discuss these:

- cloud solutions (PythonAnywhere)
- web notebooks (Jupyter)
- Python-like programming languages (CoffeeScript)

🐍 What is Python?

- interpreted language, useful for rapid prototyping
- dynamic typing, but saner than vanilla JS
- multiple paradigms, context-aware keywords
- huge standard library, third-party repositories

🤔 Advantages

- unified FE/BE logic, such as forms validation
- faster and safer processing of user data
- reduced hosting costs (no application server)
- offline availability (WebView, PWA)

🚧 Problems

- Python interpreter not available out of the box
- native code of some libraries (C/C++/Rust)
- limited minification/bundling possibilities

```
__all__='Field','CharField','IntegerField','DateField','TimeField','DateTimeField','DurationField',
class M:
 widget=AG;hidden_widget=A9;default_validators=[];default_error_messages={K:D('This field is require
 def __init__(A,*,required=H,widget=A,label=A,initial=A,help_text=L,error_messages=A,show_hidden_ini
  B=widget;A.required,A.label,A.initial=required,label,initial;A.show_hidden_initial=show_hidden_ini
  if E(B,type):B=B()
  else:B=W.deepcopy(B)
  A.localize=localize
  if A.localize:B.is_localized=H
  B.is_required=A.required;D=A.widget_attrs(B)
  if D:B.attrs.update(D)
  A.widget=B;C={}
  for F in reversed(A.__class__.__mro__):C.update(getattr(F,'default_error_messages',{}))
  C.update(error_messages or{});A.error_messages=C;A.validators=[*A.default_validators,*validators];
 def prepare_value(A,value):return value
 def to_python(A,value):return value
```

Minified Django code

Use cases

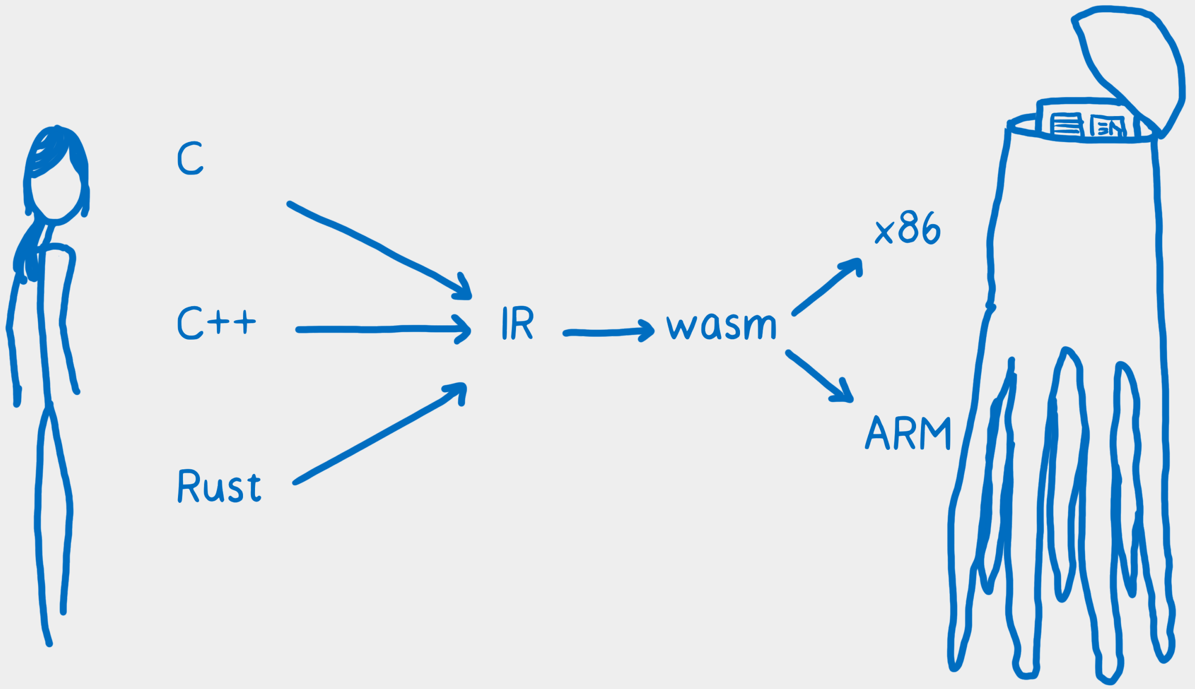
- generic web applications
- data analysis and processing
- interactive code documentation
- browser plugins (?)
- ...

Implementations (just a few)

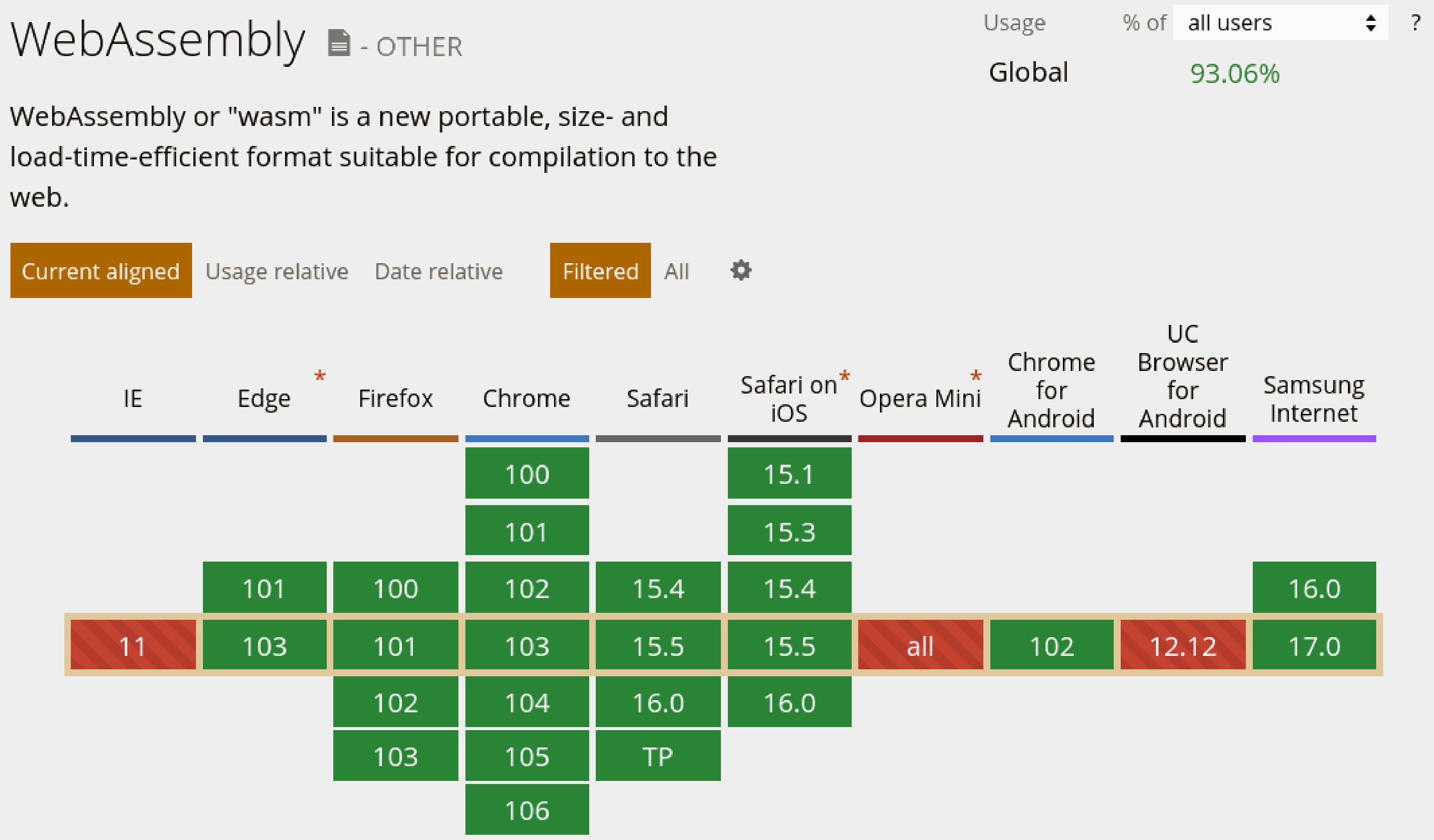
Batavia	***Brython***	Empythoned	Jiphy
MicroPython	Pyjamas	***Pyodide***	PyPyJS
RapydScript	RustPython	Skulpt	Transcrypt

PYODIDE

- based on CPython with (almost) complete stdlib
- supports native code (to some extent)
- precompiled distribution and Docker build container available



hacks.mozilla.org/2017/02/creating-and-working-with-webassembly-modules



WebAssembly - OTHER

Usage % of all users ?

Global 93.06%

WebAssembly or "wasm" is a new portable, size- and load-time-efficient format suitable for compilation to the web.

IE	Edge *	Firefox	Chrome	Safari	Safari on iOS *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			🟩 100		🟩 15.1				
			🟩 101		🟩 15.3				
	🟩 101	🟩 100	🟩 102	🟩 15.4	🟩 15.4				🟩 16.0
🟥 11	🟩 103	🟩 101	🟩 103	🟩 15.5	🟩 15.5	🟥 all	🟩 102	🟥 12.12	🟩 17.0
		🟩 102	🟩 104	🟩 16.0	🟩 16.0				
		🟩 103	🟩 105	🟩 TP					
			🟩 106						

caniuse.com/wasm

Pyodide example - external library

```
loadPyodide().then(async (py) => {
  await py.loadPackage("mypackage");
  const mypackage = py.pyimport("mypackage");

  const result = mypackage.myfunction("test value");
});
```

Pyodide example - React component

```python
import js
from pyodide import to_js

createElement = js.React.createElement
useEffect = js.React.useEffect
useState = js.React.useState

@to_js
def App(props, children):
    fact, set_fact = useState(None)

    @to_js
    async def fetch_cat_fact():
        response = await js.fetch("https://catfact.ninja/fact", method="GET")
        response_json = await response.json()
        set_fact(response_json.fact)

    useEffect(fetch_cat_fact, js.Array())

    return createElement('p', None, f"{fact}")
```

Pyodide example - package management

```
{
  "packages": {
    "libdemo": {
      "name": "libdemo",
      "version": "1.0.0",
      "file_name": "libdemo-0.1.0-py3-none-any.whl",
      "depends": ["pillow"]
    },
    "pillow": {
      "name": "Pillow",
      "version": "9.1.0",
      "file_name": "PIL-9.1.0-cp310-cp310-emscripten_wasm32.whl",
      "depends": []
    }
  }
}
```

Pyodide drawbacks

- limited debugging features
- potential memory issues

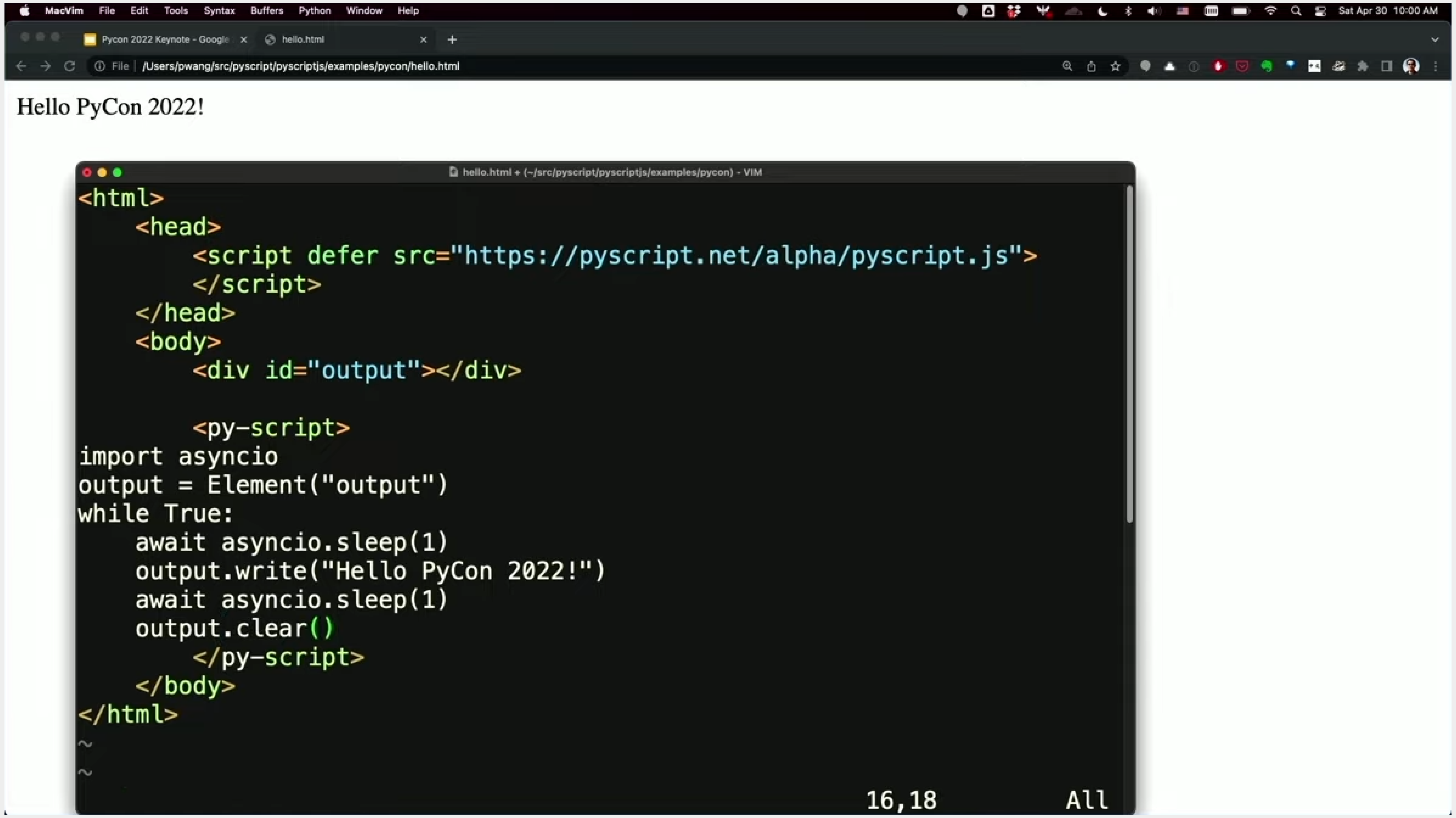
Is it worth it?

Upcoming changes to Pyodide

- `import { loadPyodide } from "pyodide"`
- new packages: `opencv`, `ffmpeg`, `cryptography`
- integrity checks
- upstream CPython and Emscripten fixes

CPython now has experimental support for cross compiling to WebAssembly platform `wasm32-emscripten`. The effort is inspired by previous work like Pyodide. (Contributed by Christian Heimes and Ethan Smith in bpo-40280.)

docs.python.org/3.11/whatsnew/3.11.html

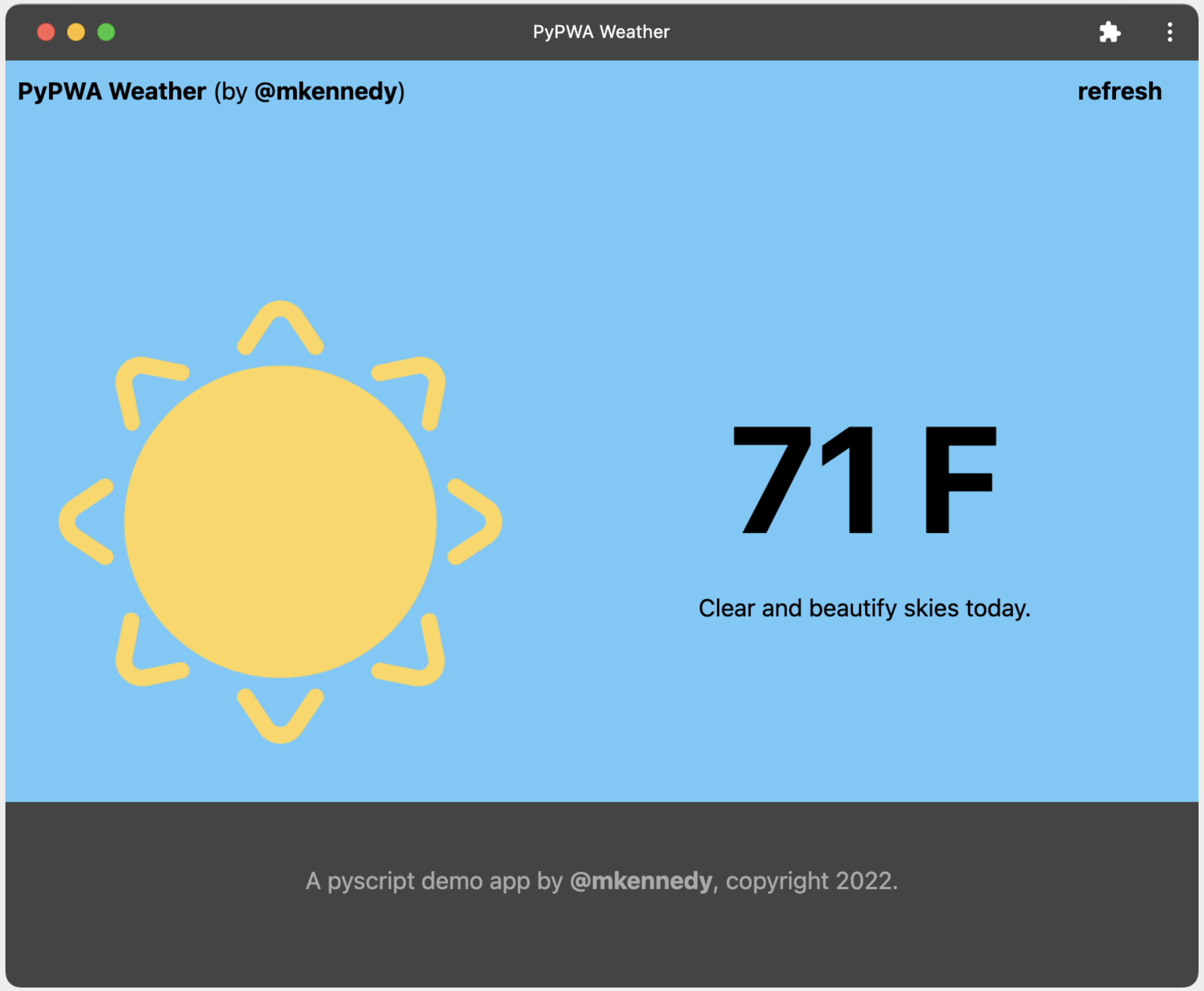

MacVim File Edit Tools Syntax Buffers Python Window Help
Sat Apr 30 10:00 AM
Pycon 2022 Keynote - Google
hello.html
File | /Users/pwang/src/pyscript/pyscriptjs/examples/pycon/hello.html
hello.html + (~/src/pyscript/pyscriptjs/examples/pycon) - VIM

Hello PyCon 2022!

```
<html>
    <head>
        <script defer src="https://pyscript.net/alpha/pyscript.js">
        </script>
    </head>
    <body>
        <div id="output"></div>

        <py-script>
import asyncio
output = Element("output")
while True:
    await asyncio.sleep(1)
    output.write("Hello PyCon 2022!")
    await asyncio.sleep(1)
    output.clear()
        </py-script>
    </body>
</html>
~
~
                                                        16,18         All
```

Programming for Everyone, Peter Wang, PyCon US 2022


PyPWA Weather

PyPWA Weather (by **@mkennedy**)

refresh

71 F

Clear and beautify skies today.

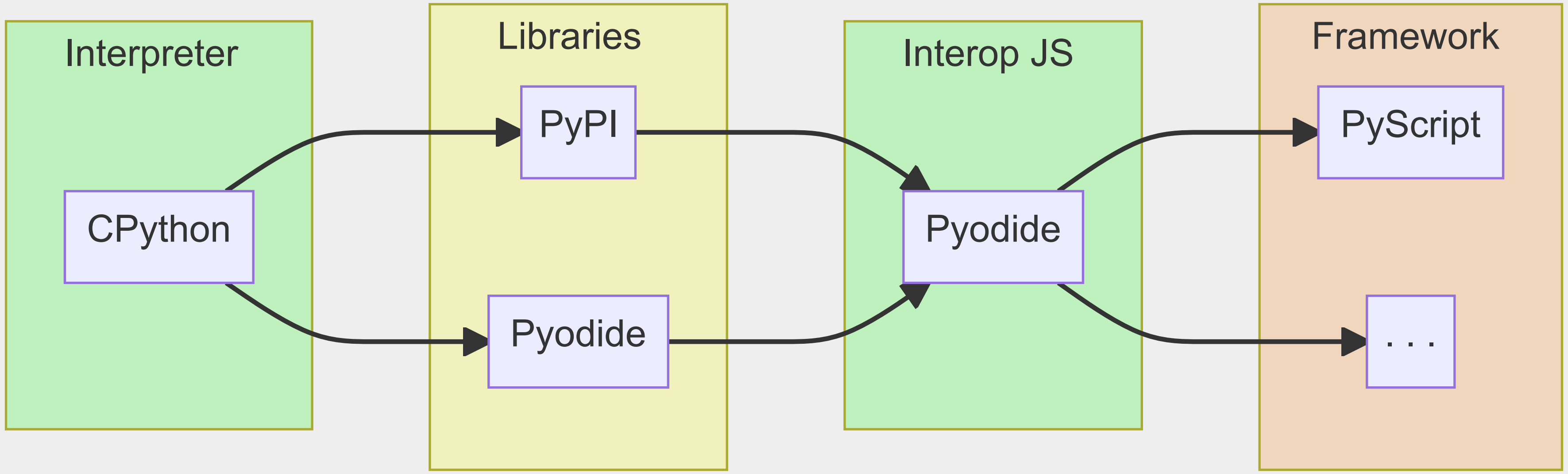
A pyscript demo app by **@mkennedy**, copyright 2022.

github.com/mikeckennedy/pyscript-pwa-example

PyScript is a new framework for running Python code inside a browser. (...)

PyCharm will recognize Python syntax for code inside <py-script> tags in html files and highlight it accordingly.

blog.jetbrains.com/pycharm/2022/06/2022-2-eap-2


Interpreter
CPython
Libraries
PyPI
Pyodide
Interop JS
Pyodide
Framework
PyScript
. . .

Online resources

Docs, examples, REPL:

- https://pyodide.org
- https://pyscript.net

Source code (dragons ahead):

- https://github.com/pyodide/pyodide/commits/main
- https://github.com/python/cpython/pulls?q=wasm

Demo

Thank you!