PyCon China 2020

PyConChina 2020
PYTHON 中国开发者大会 2020

Python For Good

Python作为嵌入脚本语言及RustPython简介

米明恒
mmh_web@163.com

演讲提要

- 嵌入脚本语言简介
- 个人对Python在嵌入脚本领域的看法
  - Python语言是否适合作为嵌入脚本语言
  - CPython解释器在嵌入脚本领域的短板
- RustPython 简介
  - RustPython 为什么适合作为嵌入脚本解释器
  - RustPython 项目，我们可以贡献什么

嵌入脚本语言简介

- 嵌入脚本语言通常是指:
  - 将一种动态类型、解释执行的脚本语言，嵌入到一种静态类型、编译执行的宿主语言中。
- 嵌入脚本语言的应用场景:
  - 将嵌入脚本语言作为主程序框架的扩展机制，处理频繁变更的业务逻辑
  - 将嵌入脚本语言作为领域特定语言（DSL），提高开发效率

嵌入脚本语言示例

Nginx + Lua -> OpenResty 、 Kong

Microsoft Office + VisualBasic -> 宏

浏览器 + Javascript -> 缤纷的互联网

Python适合作为嵌入脚本语言吗?

- 从脚本语言用户角度，从语言特性上来说
  - 非常适合 ✅
- 从宿主程序开发者角度，从解释器实现角度来说
  - CPython解释器： 不太适合 ❌
  - 其他Python解释器： 有些适合 ✅

为什么Python语言适合作为嵌入脚本语言

- Python 语法简洁，表达能力强
- 内置数据类型功能强大，易于使用
- 初级语法可以在短时间内掌握，适合各类用户学习
- 大势所趋，越来越多的人将会掌握基础的Python语法，采用Python作为嵌入脚本的行业软件会得到更多人的青睐，潜在用户量非常可观

为什么CPython解释器不适合嵌入脚本场景

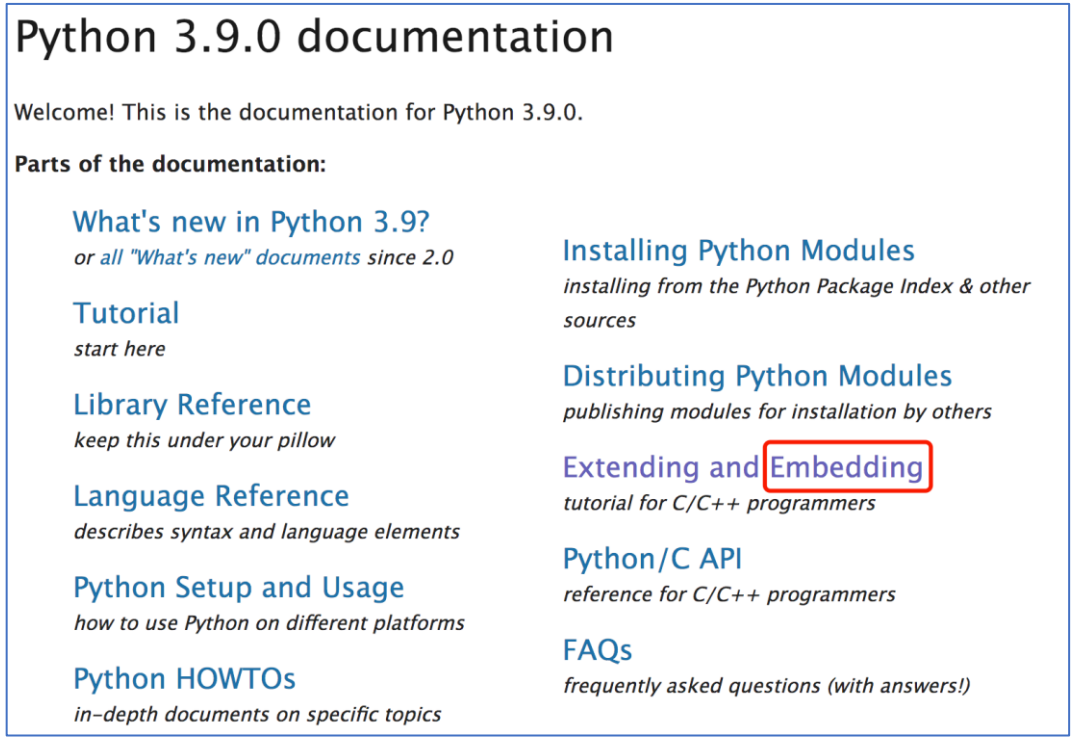
- 嵌入脚本语言通常需要较为安全的沙箱特性，通过宿主程序提供的有限API接口来与宿主程序交互。
- CPython致力于成为一种通用、易用的解释器，其标准库、解释器与操作系统有过多耦合。难以实现沙箱，在执行不可信用户的输入时有较大风险

- 嵌入脚本语言通常需要同时运行多个轻量的解释器实例，并行处理多个任务。
- CPython解释器的代码实现上，大量使用了全局变量，导致无法在一个宿主进程中开启多个互不影响的独立解释器实例。
- PEP 554提出的子解释器特性正在推动消除全局变量，但目前仍有大量遗留问题

为什么CPython解释器不适合嵌入脚本场景

如何理解CPython官方文档中的嵌入呢?

- 把Python当做胶水语言，程序初始化后直接交付给Python解释器，由CPython解释器调用C、C++开发的其他函数。
- 把嵌入的CPython解释器当做一个单例模式的解释器，宿主程序必要时调用CPython解释器，宿主程序与CPython解释器轮流运行。


Python 3.9.0 documentation

Welcome! This is the documentation for Python 3.9.0.

Parts of the documentation:

What's new in Python 3.9?
or all "What's new" documents since 2.0

Tutorial
start here

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs
in-depth documents on specific topics

Installing Python Modules
installing from the Python Package Index & other sources

Distributing Python Modules
publishing modules for installation by others

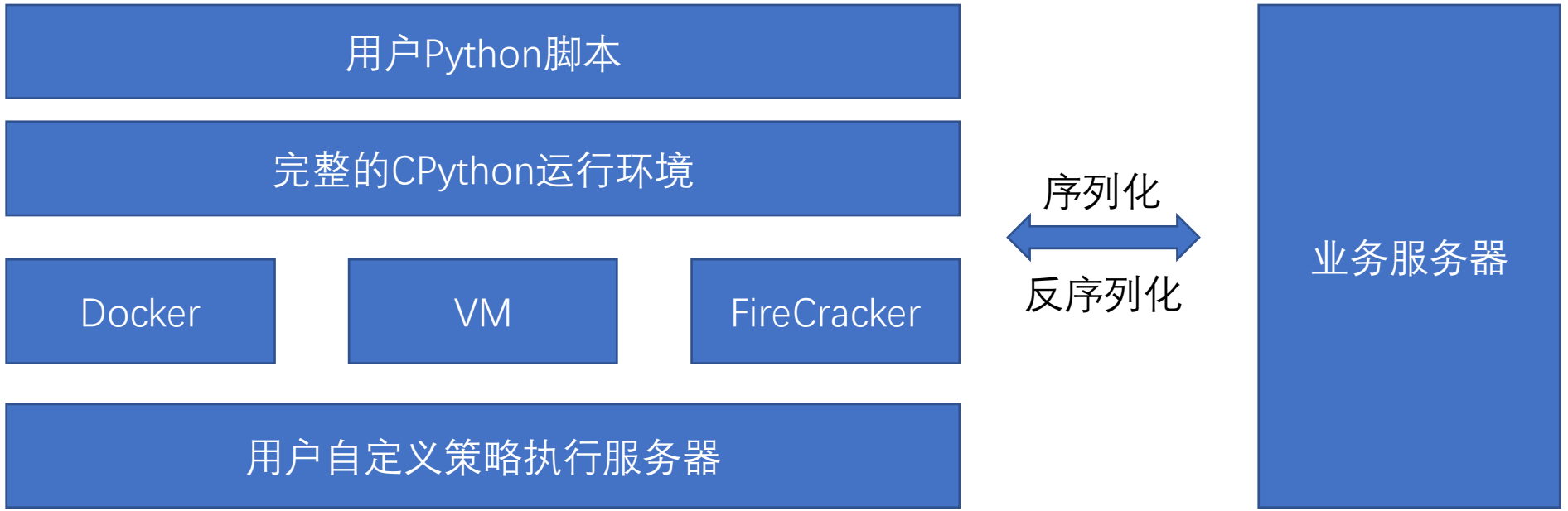
Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

FAQs
frequently asked questions (with answers!)

为什么CPython解释器不适合嵌入脚本场景

一些量化交易平台、AI开放平台等，虽然支持用户使用Python语言来自己定义逻辑，但为了服务集群的安全，不得不采用沉重的隔离技术，用户代码仍然是在完整的CPython解释器中独立运行，并不属于嵌入脚本语言。

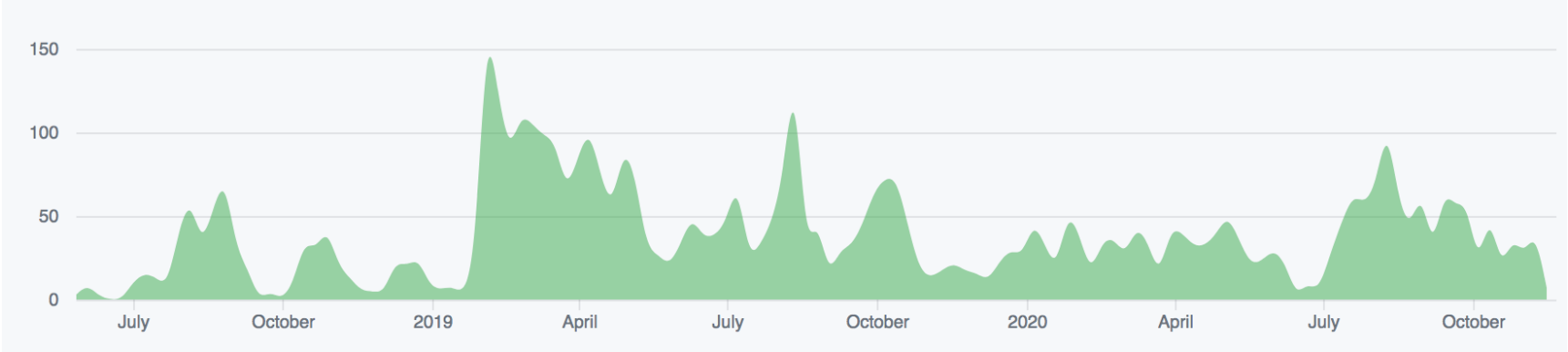


RustPython简介

既然CPython解释器难以支持Python作为嵌入脚本语言来使用，那么不妨看看另一个Python解释器的实现——RustPython

项目地址：https://github.com/RustPython/RustPython

6.2k Stars　持续活跃



RustPython简介

特性:

- 解释器的实现没有使用全局变量
  - 由于Rust语言天生对全局变量不友好，因此RustPython中与解释器状态有关的信息全部保存在一个VM结构体中，可以非常方便地创建多个解释器实例。

RustPython简介

特性:

- 模块化编译控制，与操作系统耦合低
  - 可以裁减掉大量与操作系统相关的模块，仅保留单纯的解释器及内置数据类型，便于实现沙箱环境。

RustPython简介

特性：

- 没有GIL
  - 虽然RustPython项目启动已经有几年时间了，但其对多线程的支持一直到今年4月份才开始进行
  - RustPython多线程的实现方式为：在新线程中运行一个新的VM实例，而这种实现方式，与嵌入语言通常会运行多个解释器实例的需求不谋而合
  - GC不再与线程有捋不清的关系，也体现了前面提到的RustPython模块划分清晰，与操作系统耦合小的特点
  - 多线程并发安全很大一部分由Rust在编译期保证

RustPython简介

特性：

- 使用纯Rust语言开发，而不是CPython bindings。
  - 可以保证解释器本身的内存安全性，非常匹配前面提到的嵌入脚本语言的应用场景。
  - 可以编译为WASM，RustPython解释器可以直接运行在浏览器中
    - Rust的编译期内存安全检查与WebAssembly的安全内存模型配合，进一步提升沙箱安全性？
    - 使用Python操作浏览器DOM？
    - 无限可能。。。

RustPython简介

特性：

- 由于没有CPython那样的历史包袱，可以尝试一些花哨的实验性功能
  - 例如3个月前，新增了极简的JIT编译功能

RustPython简介

RustPython目前的不足：

- 仍处于开发阶段，作为独立的Python解释器，还有一些标准库没有实现和CPython的兼容
- 目前局限于Rust生态，可以方便嵌入Rust程序中，但没有CAPI，更无法使用NumPy等科学计算库
- 解释器代码没有做深入优化，代码执行效率较CPython低很多

RustPython简介--我们能贡献什么

- 可以参与标准库的开发，完成尚未实现的标准库
- 可以优化RustPython的代码，提高执行效率
- 可以开发RustPython的C-FFI Wrapper，使得更多语言可以调用RustPython
  - API开发可以参考HPy项目，其目标是替代现有CPython提供的API，制定一套适用于不同Python实现的API标准，而不仅限于CPython，目前PyPy、Cython等都参与到了这个项目中。
  - https://github.com/hpyproject/hpy
- 当然，要先学会Rust

我在RustPython上做的一些尝试

- Fork自一个古老的分支，做了一些玩具级别的尝试：
  - 删减了操作系统、网络、文件IO相关的模块，使其尽量类似一个沙箱环境
  - 增加了解释器Cycle限额，超限后抛出异常（实现方法比较暴力）
  - 提供CAPI供其他语言调用，API采用HPy规范，显式传递虚拟机对象

  https://github.com/myrfy001/RustPython/tree/mini-rust-python

- 使用Golang的CGO调用上述分支编译的动态链接库：
  - https://github.com/myrfy001/py-as-dsl

RustPython简介

- 本次闪电演讲仅为抛砖引玉
- 我也是Rust语言的初学者，对RustPython的了解还不是非常深入
- 希望我的演讲能够鼓励大家将广义的Python语言应用到更多领域

PyCon China 2020

Python For Good

PyConChina 2020 | PYTHON 中国开发者大会 2020

THANK YOU

微信号，没有头像。很多小程序因此崩溃，可以当做测试用例

公众号，极客幼稚园。无广告，无软文，更新不频繁，但都是干货

GitHub：https://github.com/myrfy001

个人主页：http://blog.ideawand.com