如何快速的实现一门编程语言

主讲人：Turaiiao

黄菁（Jing），生于 01 年，

热爱编程，2015 年写下了第一行 C++ 代码，梦想是自己创作一门编程语言。

- 小目标 App 开发者，用户注册数万
- XYIIO 团队创始人

主讲内容

- 编译器与解释器
- 基本的开发过程和功能实现
- 使用字节码和栈实现四则运算
- 探索 Python 字节码

我会对比示例代码进行说明。

编译器与解释器

编译器（Compiler）和解释器（Interpreter）的运行效率差距很大，

编译器通常是直接编译成机器码或者字节码，且字节码需要通过自己的虚拟机（VM）解释执行。

这方便多平台和可移植性。

解释器不需要很多步骤，直接遍历语法树（AST）得到结果，这种方式效率低下。

基本的开发过程和功能实现

编译器和解释器第一步会把源代码解析成词法列表（tokens），这一步叫做 Lexer。

然后将词法列表进行语法分析得到抽象语法树，这一步通常叫做 Parser。

通常解释器到这一步就会访问（visitor）语法树求值，而编译器则会进行更多的步骤。

例如代码生成（CodeGen）**和**类型检查（TypeChecker），**最**简单的方式是编译成字节码，构建运行时（Runtime），**然后通**过我们的虚拟机解释执行。

使用字节码和栈实现四则运算

计算机不懂我们人类相知的四则运算，人类能看懂的叫做中缀表达式，计算机只懂后缀表达式。

1 + 2 * 3 - (4 + 5) -> Infix Expr **中**缀表达式

1 2 3 * + 4 5 + - -> Postfix Expr **后**缀表达式

分析表达式得到后缀表达式栈，然后转换成一个块（chunk）， 块内包含两个栈，字节码栈和数据栈，最后遍历字节码栈得到运算结果。

探索 Python 字节码

Python 是使用 C 语言开发的一门脚本语言，编译成字节码的方式进行执行，每个 Python 项目里都会包含隐式 .pyc 文件，这些文件就是字节码的缓存，再次执行而不用进行再次编译。

“Python is always slower than C.” -- PyCon 2018

Thanks！