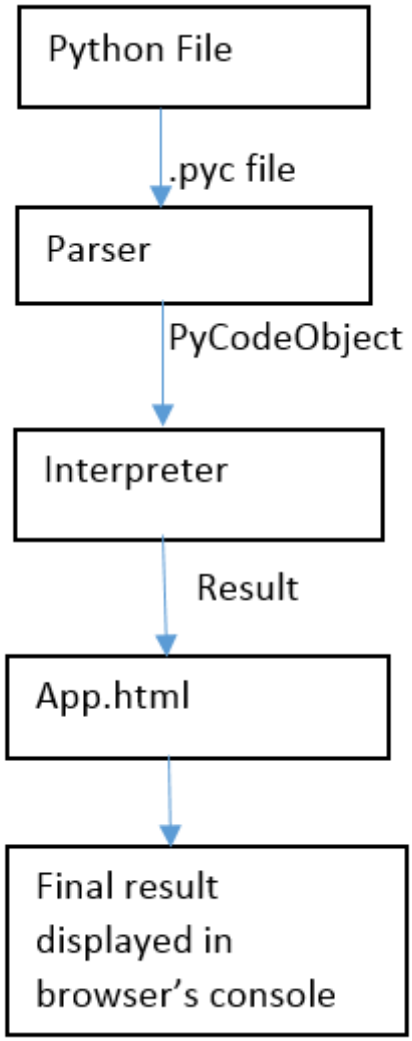
TyPy: A Python Bytecode Interpreter for the Browser

Puja Mishra, Kate Silverstein

27 October 2014

1 Understanding the Requirements

The goal of this project is to implement a Python bytecode interpreter which is engine dependent and also can be hosted on a variety of browsers. So the high level picture of how we have implemented is shown below.



2 Design Approach

The input to the system is the python program written by the user and compiled using "compileall" module. This *.pyc file is uploaded when the user opens application, this file is passed to the parser where parser parses the bytecodes and generates a PyCodeObject to the the interpreter (which here is a stack based interpreter) , which interprets it and gives the result on the browser's console.

2.1 Parser

2.1.1 Structure of a Python Bytecode File

The bytecode is an attribute of the code object and is found in cocode attribute of the code object and contains instructions for the interpreter. Bytecode is nothing but a series of bytes.The interpreter will loop through each byte, look up what it should do for each one, and then do that thing. Using the dis module we can disassemble the bytecode.

2.2 Interpreter Design

We borrowed the general structure of our interpreter from byterun and adopted it to our design. It is a stack based interpreter where the bytecodes are parsed and put on the stack and interpreted by manipulating the items on the stack.

2.2.1 Stack-Based Interpreters

2.2.2 Implementation for TyPy

2.3 Other Design Considerations

2.3.1 Client-Side vs. Server-Side Javascript

2.4 Test Suite

3 Implementation Results

3.1 Supported Functionality

The interpreter we have implemented can interpret simple "if" statements, for and while loops, basic list creation and indexing, basic dictionary creation and queries, basic math operations.

4 Future Work

As an extension of the project we can add more functionality which can help to interpret complex functions written in python.

5 Conclusion

Working on this project has been a very good and learning experience. The system implemented by us is a simple interpreter which can interpret basic functionality written in python.

6 References

References

[1] Ned Batchelder, *byterun* . https://github.com/nedbat/byterun