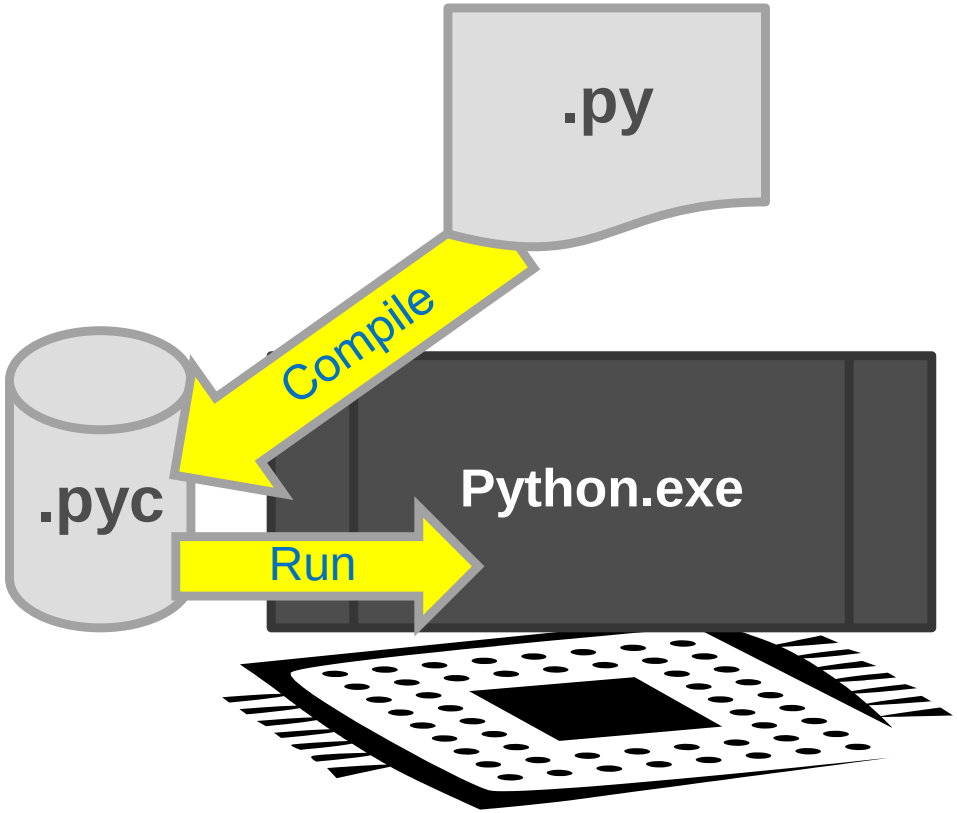
Python Alternative Execution

CPython



We know CPython...

- Source is compiled to byte-code by python exe
- Compiled code is cached in .pyc files
- Python is written in C, can be called from C code or extended by C code

CPython And C

- Python is written in C with nice Object model and memory management
- Implemented as a DLL
- Modules can be written in C to extend it
- Existing C DLLs can be glued from Python with CTypes

Extending Python

- Example – lets write some C code that executes an OS shell command – here in linux

```
>>> import spam
>>> status = spam.system("ls -l")
```

PASSCALL

Extending CPython

```
#include <Python.h>
```

```
static PyObject *
spam_system(PyObject *self, PyObject *args)
{
    const char *command;
    int sts;

    if (!PyArg_ParseTuple(args, "s", &command))
        return NULL;
    sts = system(command);
    return Py_BuildValue("i", sts);
}
```

http://docs.python.org/2/extending/

Extending CPython

```
static PyMethodDef SpamMethods[] = {
    ...
    {"system",  spam_system, METH_VARARGS,
     "Execute a shell command."},
    ...
    {NULL, NULL, 0, NULL}        /* Sentinel */
};
```

```
PyMODINIT_FUNC
initspam(void)
{
    (void) Py_InitModule("spam", SpamMethods);
}
```

```
>>> import spam
>>> status = spam.system("ls -l")
```

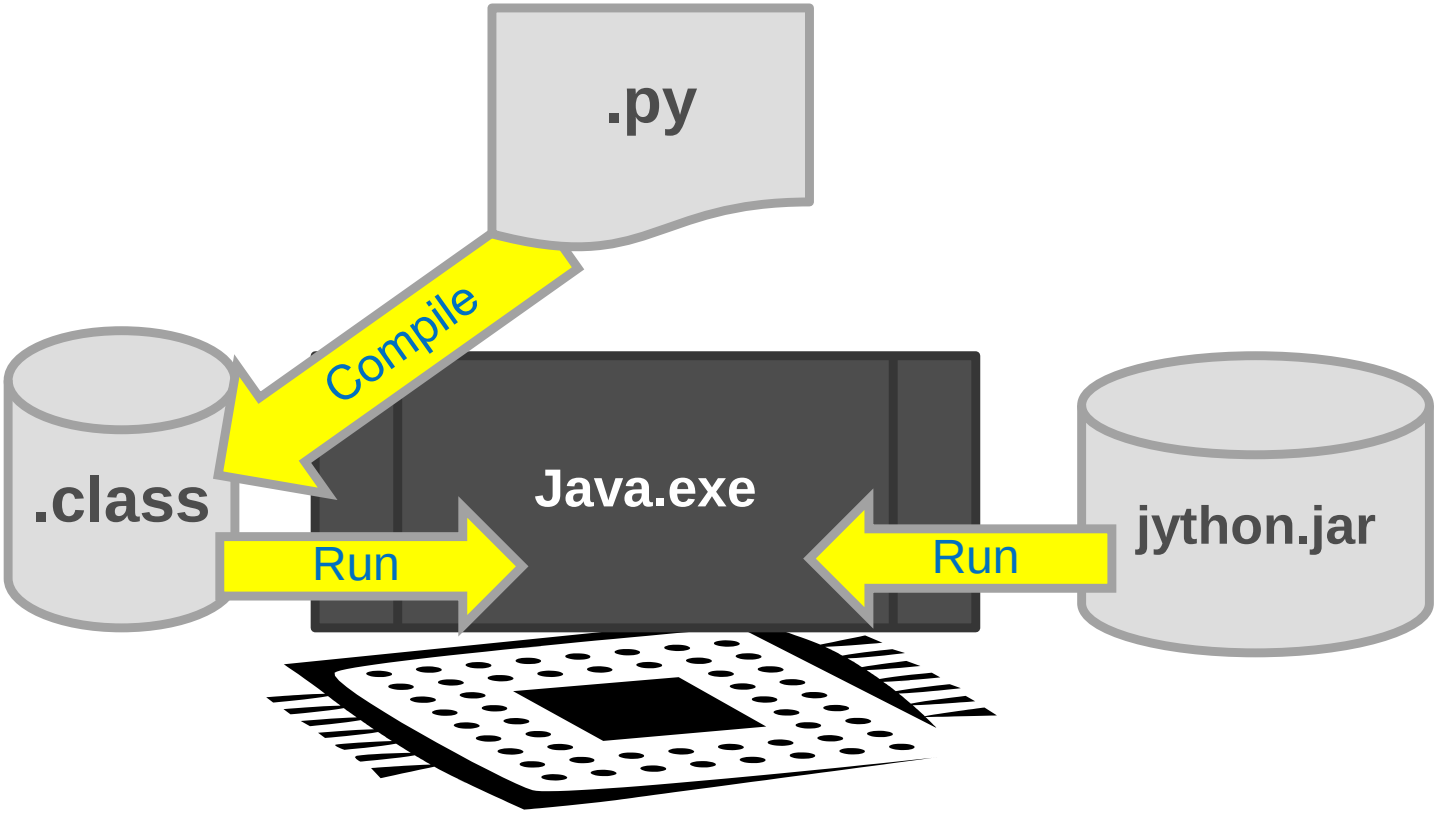
Calling CPython from C

- Very High Level Embedding

```
#include <Python.h>

int
main(int argc, char *argv[])
{
  Py_SetProgramName(argv[0]);  /* optional but recommended */
  Py_Initialize();
  PyRun_SimpleString("from time import time,ctime\n"
                     "print 'Today is',ctime(time())\n");
  Py_Finalize();
  return 0;
}
```

http://docs.python.org/2/extending/embedding.html#pure-embedding

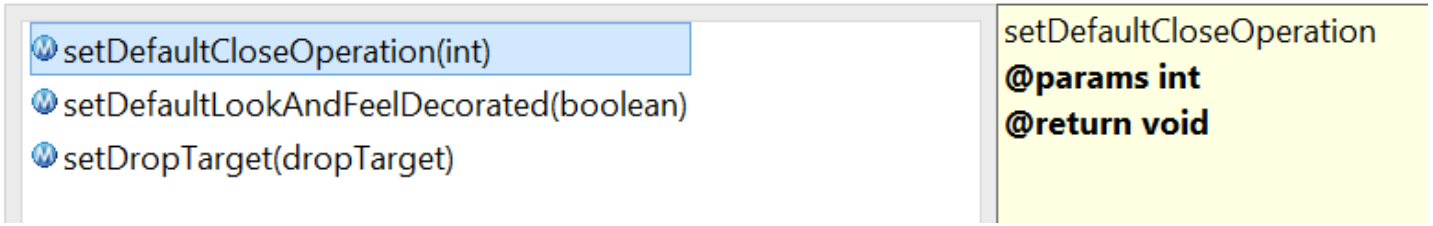
jython



jython

- Compiles Python to .class files
- Easy extension and integration with Java
- RTTI of compiled Java code!

```
>>> from javax.swing import JFrame, JLabel
>>> fr = JFrame("Hello Java")
>>> fr.setD
```



Jython – Example

```
from java.awt import Frame, Button, Font
fr = Frame("Hello")
btn = Button("Click Me!",
      font = Font("Arial", Font.BOLD, 50))
fr.add(btn)

fr.pack()
fr.show()
```

PASSCALL

Jython – Example (cont)

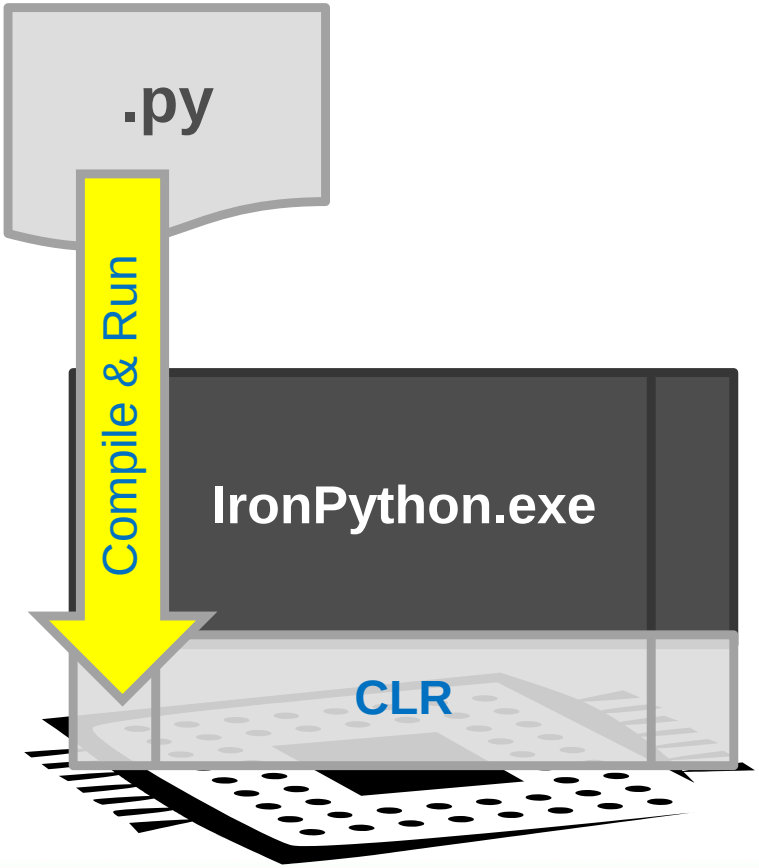
```
def pressed(evt):
    print "Pressed",

btn.actionPerformed = pressed
```

Jython – Summary

- Slower than CPython
- Very good for integration with Java (like pydev)
- Applets – currently not supported
  - Applets can be created with Jython 2.2.1
  - Java Web Start is still a web distribution option

Download: http://jython.org/

IronPython



IronPython

- Compiles to DLR (above CLR) and Mono

```
import clr
clr.AddReference("System.Windows.Forms")

from System.Windows.Forms import MessageBox
MessageBox.Show("Hello World")
```

IronPython (cont)

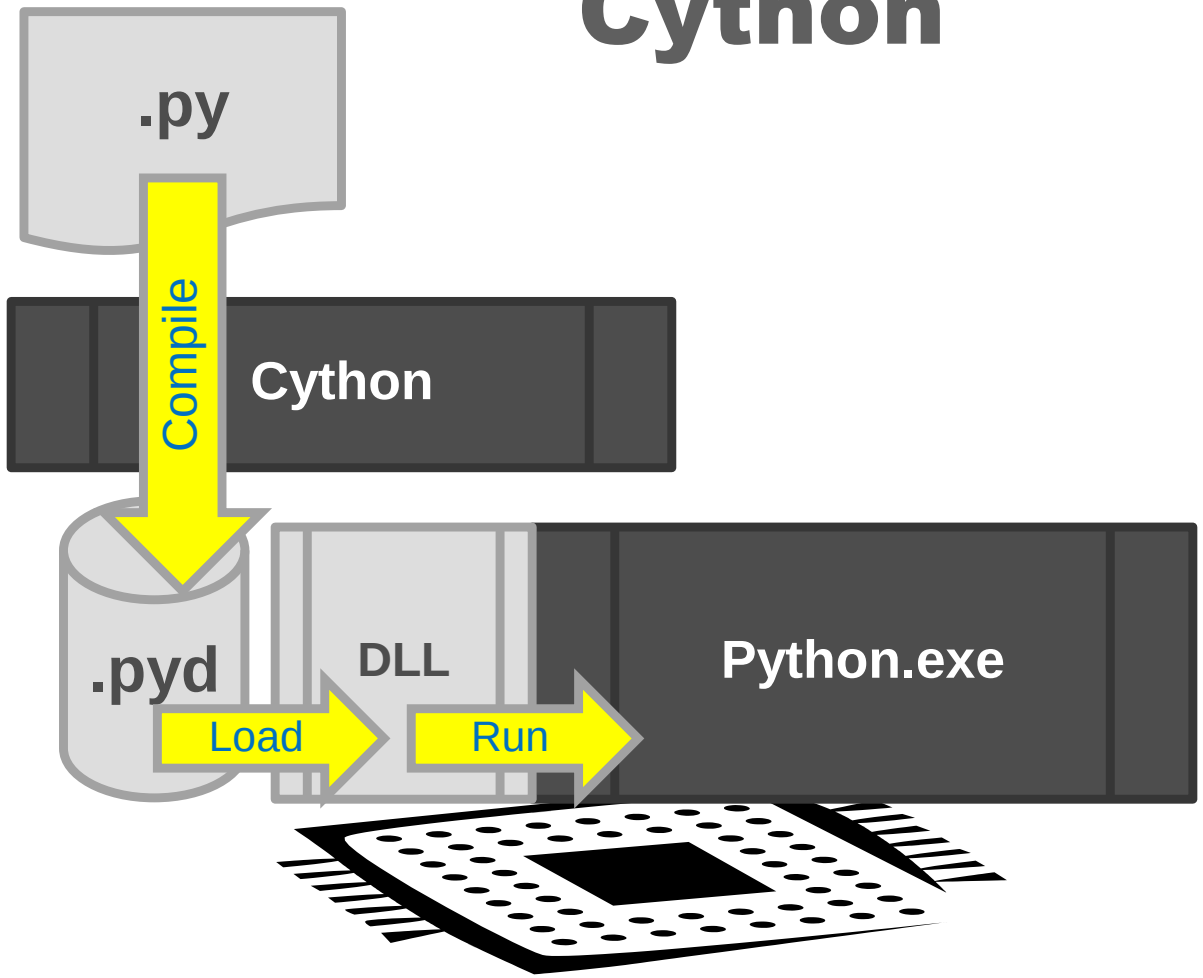
- Considered to be SLOWER than CPython in most benchmarks
- Starts slowly (like Jython)
- Has tools for generating EXE (needing lots of files), not part of Visual Studio
  - Currently even ‘Publish’ in VS does not seem to do anything

PASSCALL

IronPython

- Links:
  - VS Shell http://bit.ly/VncQCi
  - Python Tools http://pytools.codeplex.com/
  - IronPython http://ironpython.codeplex.com

Cython



Cython

- Compiles python to calls to Python DLL
- 99% Compatible with CPython
- Fully interoperable with Python
- X5-10 speed with minimal effort
- More performance increase by manually adding strong typing (.pyx files, breaks CPython syntax)

Cython - Setup

- ActivePython – pypm install cython
- Download Visual C++ 2008

http://download.microsoft.com/download/a/5/4/a54badb6-9c3f-478d-8657-93b3fc9fe62d/vcsetup.exe

```
from distutils.core import setup
from distutils.extension import Extension
from Cython.Distutils import build_ext

ext_modules = [Extension("primes", ["primes.pyx"])]

setup(
  name = 'prime app',
  cmdclass = {'build_ext': build_ext},
  ext_modules = ext_modules
)
```

Cython - compiling

- `python setup.py build_ext –inplace`
  - Builds a .c file and compiles to a .pyd file that can be imported
  - The .pyd file can be imported, here with:
    `import primes`

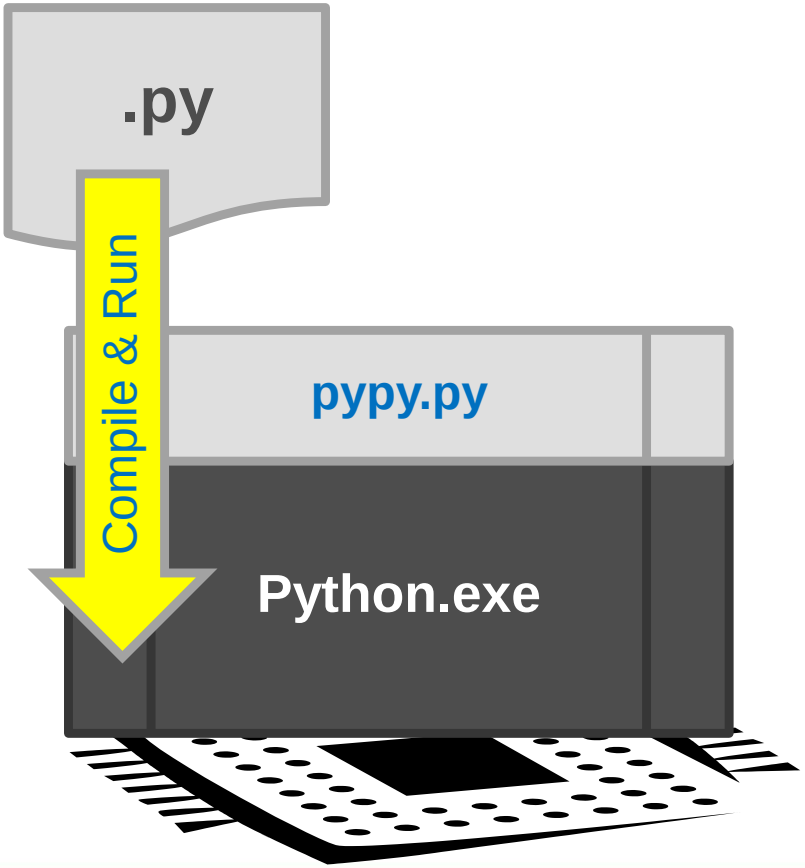
PASSCALL

Cython - optimizing

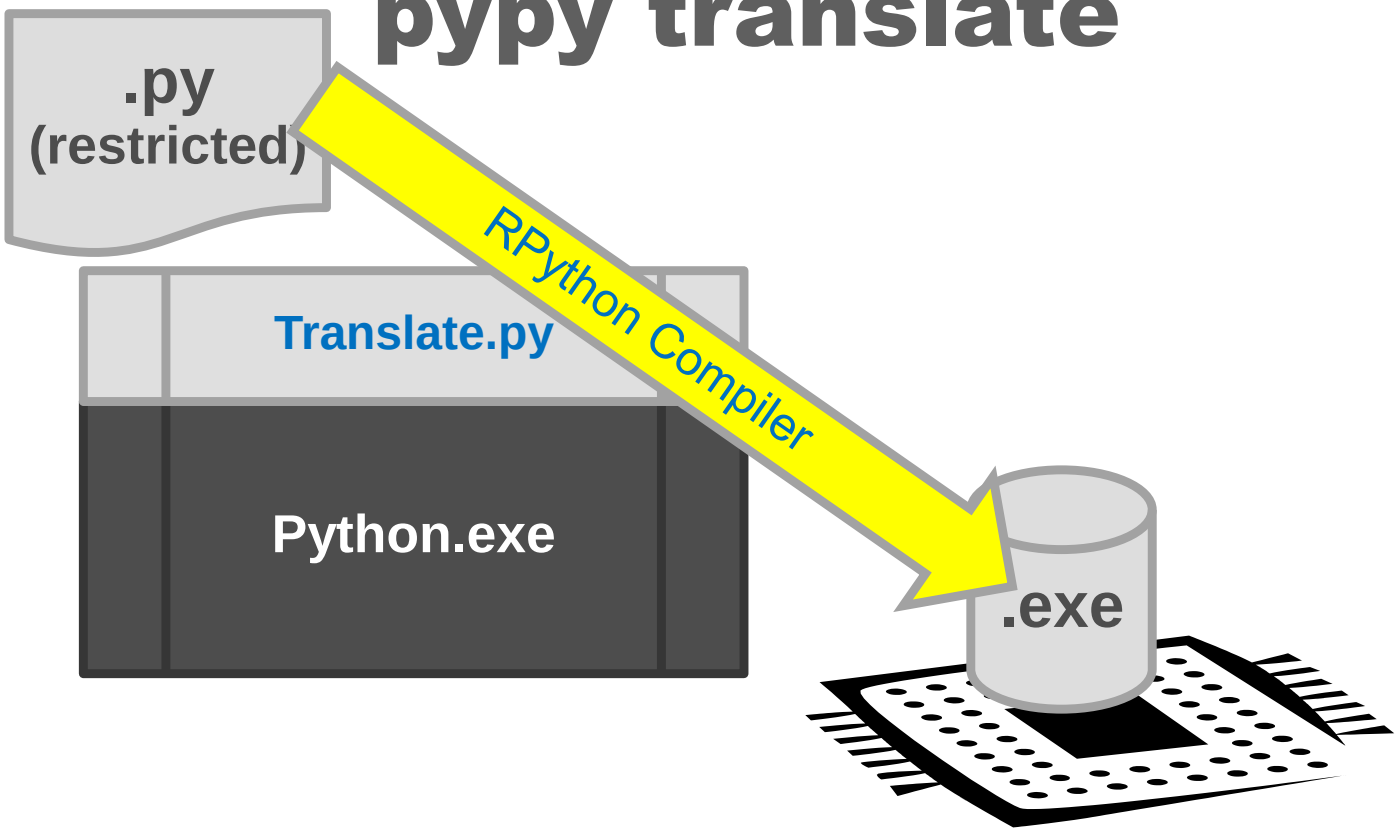
```
import time

def primes(int kmax):
    cdef int n, k, i
    cdef int p[100000]
    result = [] # A Python type.
    t = time.time()
    if kmax > 100000:
        kmax = 100000
    k = 0
    n = 2
    while k < kmax:
        i = 0
        while i<k and n%p[i] != 0:
            i = i + 1
        if i == k:
            result.append(n)
            p[k] = n
            k = k + 1
        n = n + 1
    return time.time() - t, result
```

```
import time

def primes(kmax):


    result = [] # A Python type.
    t = time.time()
    if kmax > 100000:
        kmax = 100000
    k = 0
    n = 2
    while k < kmax:
        i = 0
        while i<k and n%result[i] != 0:
            i = i + 1
        if i == k:
            result.append(n)


        n = n + 1
    return time.time() - t, result
```

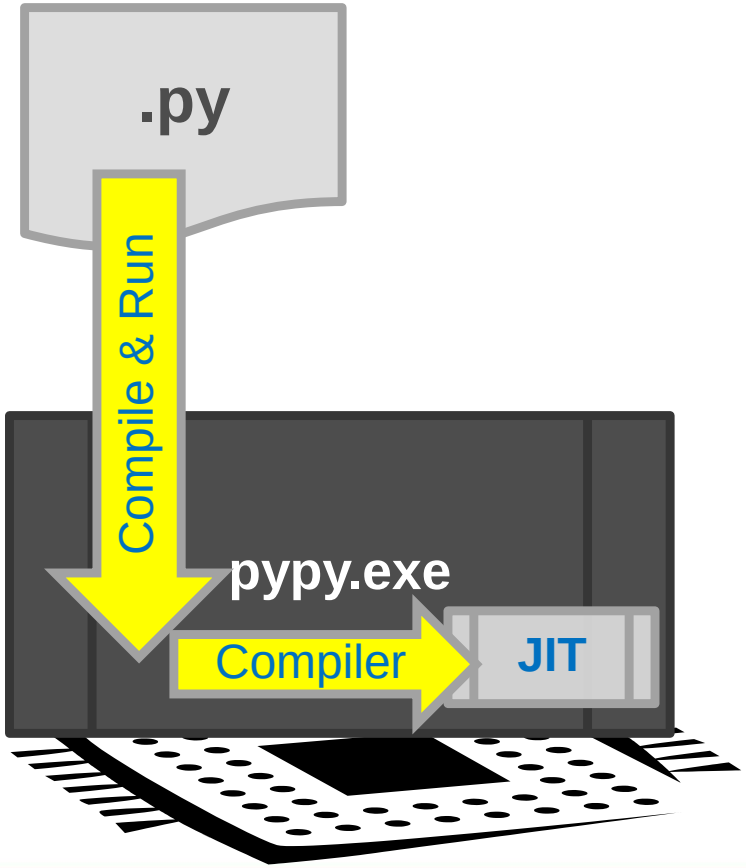
pypy



pypy translate



Pypy.exe



pypy.exe

- Compiles .py to .pyc, loads and runs with JIT → takes longer to load
- Most stuff is compatible
  - PyPy does not support refcounting semantics
  - You can't add a __del__ method to an existing class

Pypy.exe

- Download from:
  http://pypy.org/download.html

  Look for:
  Windows binary (32bit) (you may need the VS 2008 runtime library installer vcredist_x86.exe)

- See compatibility info at
  http://pypy.org/compat.html

Rpython

- The exact definition is **“RPython is everything that our translation toolchain can accept”** :)
- Can be described as a Python syntax for OO C programming
- Comes with a very small portion of the standard library

RPython Limitations

- Variables
  - Like in C and optimized Cython, each variable in every execution path, must be in a single type
  - Unlike C and Cython, no need to specify the type, it is automatically detected
  - Can only pass None instead of objects, lists etc.

RPython Limitations

- Global variables are considered CONST, except instances
- For loops can only iterate native types
- No run time definitions, eval, etc
- Tuples must be in same length and inner types in all paths

RPython Limitations

- Lists
  - Indices must be within bounds
  - Slice assignment can’t change the size
  - Indices are checked only in IndexError exc.
- Dictionaries must have the same key and value type

RPython Limitations

- Exceptions – catches only specified exceptions (otherwise may crash), not across functions
- Limited Implementation:
  - Range
  - Generators
  - Many others…

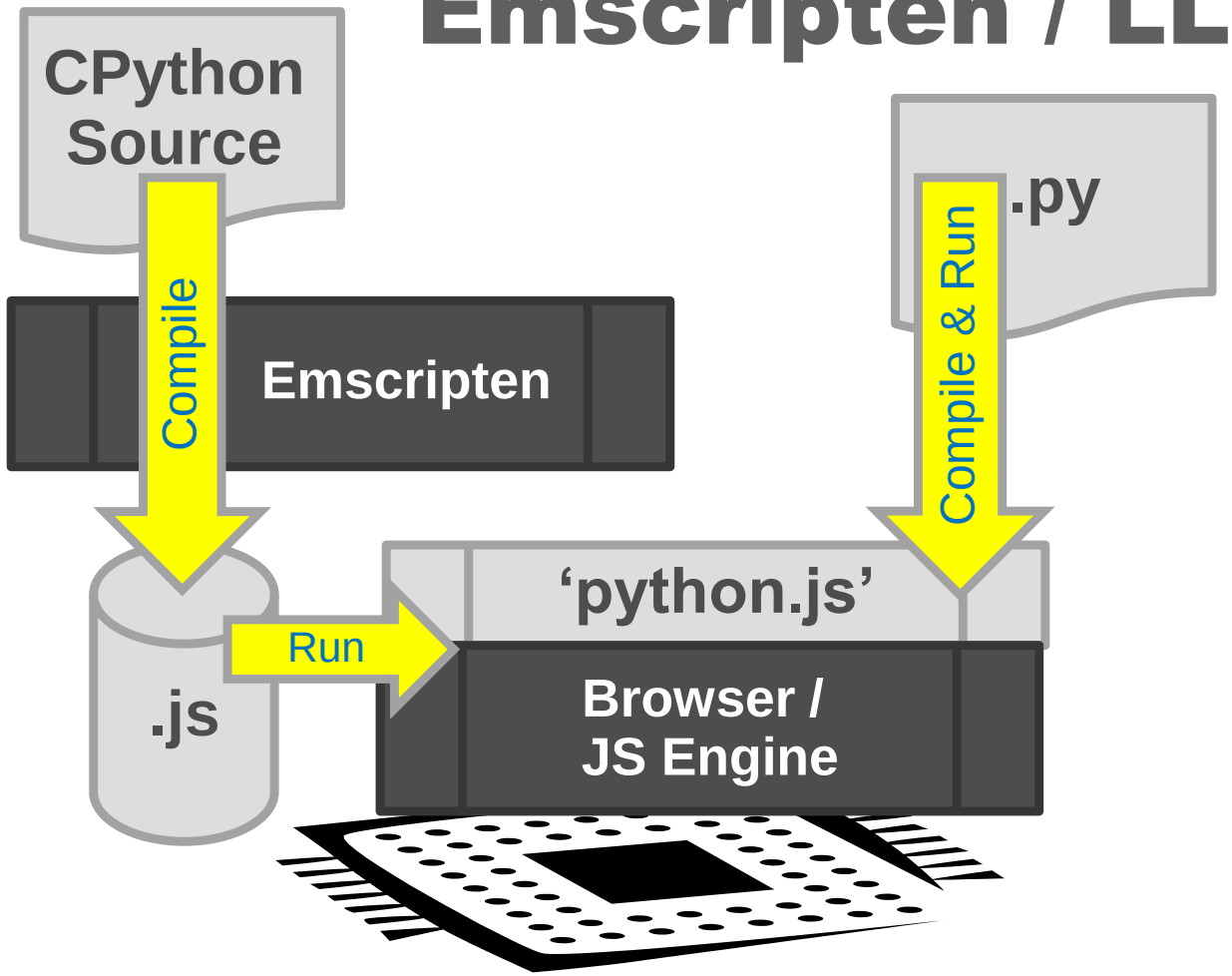
More RPython

- Download pypy sources with translate tool:
  http://pypy.org/download.html#translate
- Typical build command:

```
call vcvars32.bat

python C:\Python27\pypy\pypy\translator\goal\translate.py
--batch --output test.exe test.py
```

- Coding Guide:
  http://doc.pypy.org/en/latest/coding-guide.html
- Translate.py options:
  http://doc.pypy.org/en/latest/config/commandline.html#general-translation-options

Emscripten / LLVM

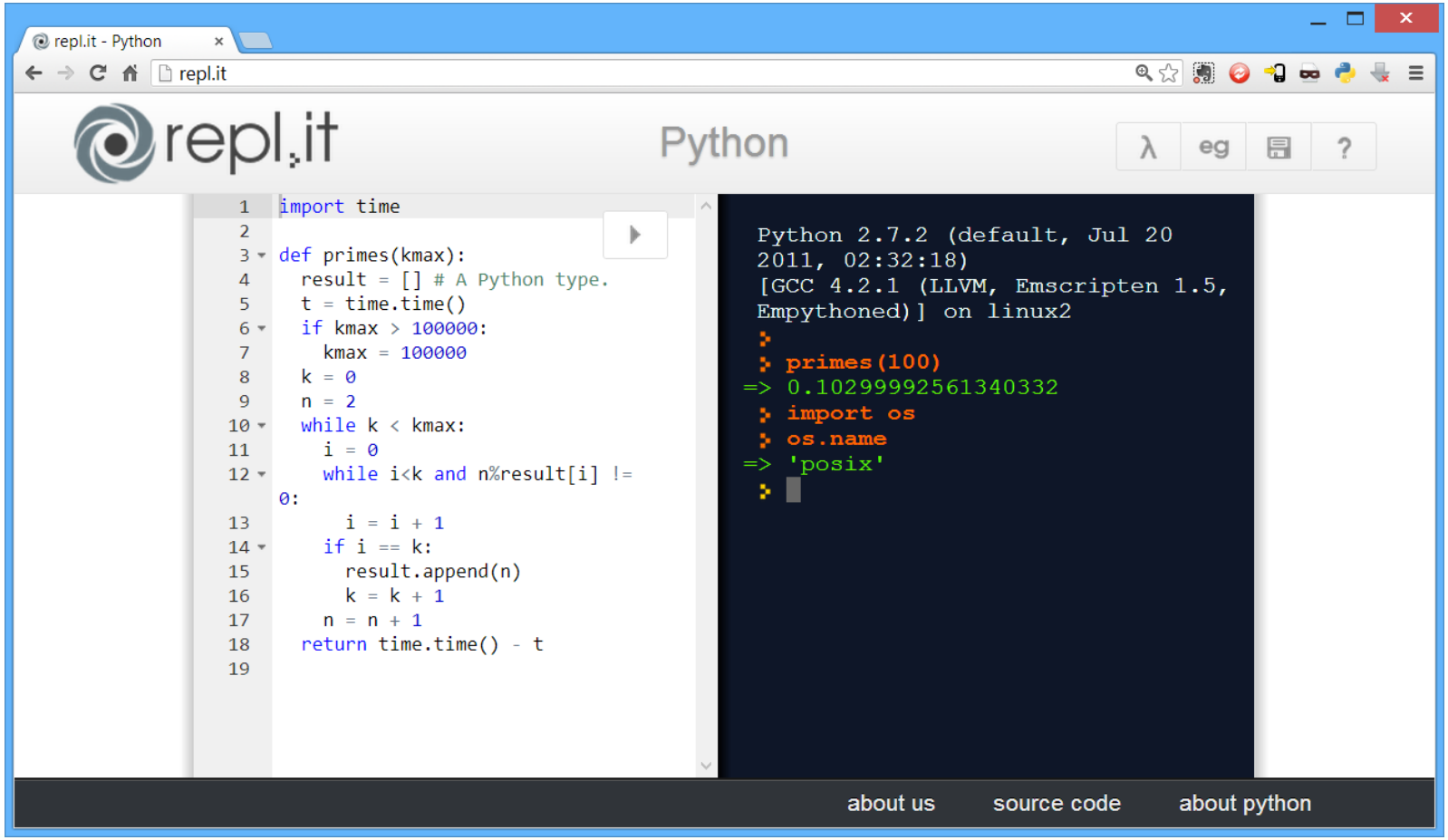


Emscripten / LLVM (JavaScript)

- Emscripten allows compiling C code to JS
- CPython was compiled to JS
- Can be used in a browser
- Example:
  - http://repl.it/languages/Python
- See Also:
  - http://emscripten.org, http://llvm.org

PASSCALL

Example – repl.it

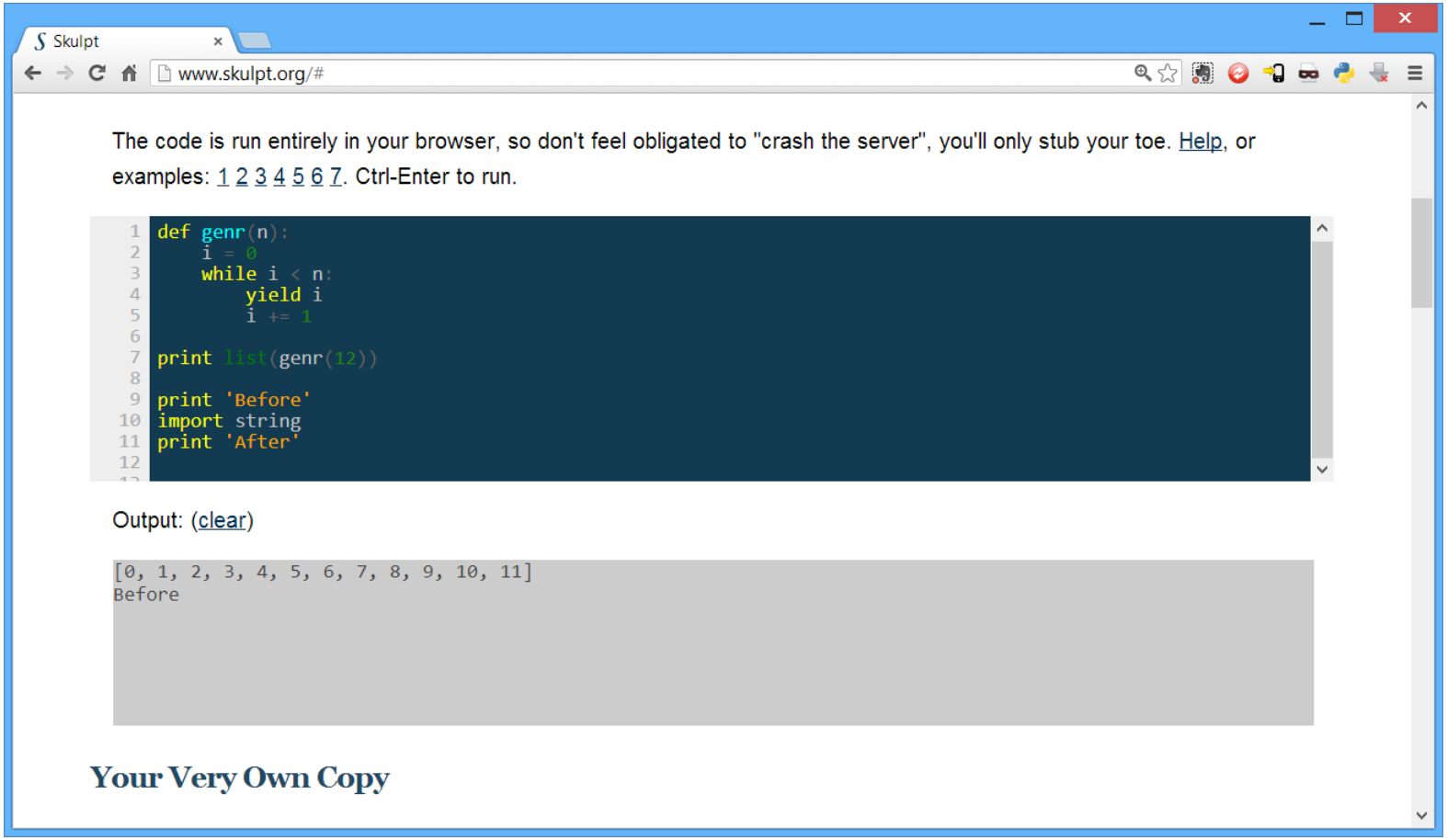


More Python in JS

- Pypy used to compile Python to JS, abandoned
- See also:
  - A survey of **Python to Javascript** converters
    http://chargen.blogspot.co.il/2011/08/survey-of-python-to-javascript.html
  - http://www.skulpt.org
  - http://www.brython.info/tests/console_en.html
- Performance – too low to mention…

PASSCALL

Example: skulpt.org



PASSCALL

Summary

Environment	CPython Compatibility	Source Security	Performance	Extras
jython	Medium - All pure python	Medium +	Low -	Java + RTTI, Eclipse, Android
CPython	Super	Medium	Low	Lots of modules
Cython as-is	High	High	Medium	-
IronPython	Medium - All pure python	Low	Low	.NET, Visual Studio, WinForms
Pypy.exe	High	Low (stupid)	Medium+	-
Cython Optimized	High	High	High	-
RPython	Low	Super	Super	Compiles to Java, JS

PASSCALL

Performance