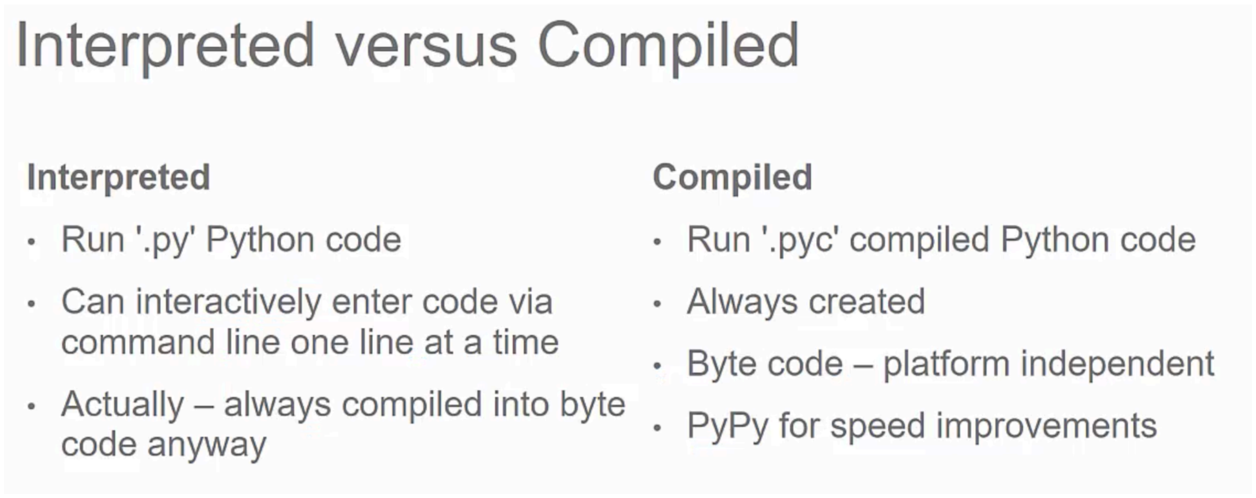


Interpreted versus Compiled

Interpreted
- Run '.py' Python code
- Can interactively enter code via command line one line at a time
- Actually – always compiled into byte code anyway

Compiled
- Run '.pyc' compiled Python code
- Always created
- Byte code – platform independent
- PyPy for speed improvements

Interpreted versus Compiled

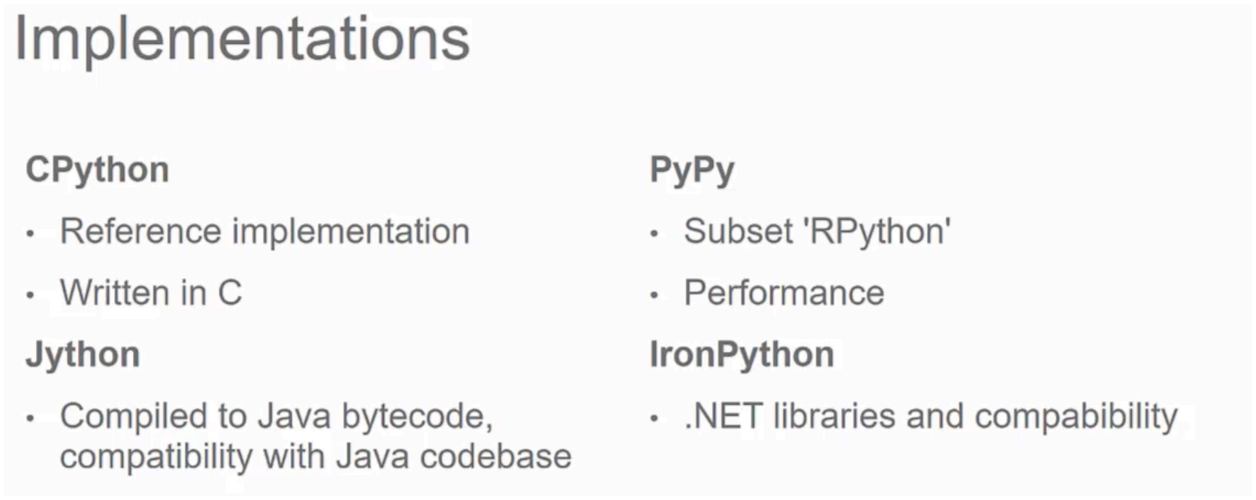
At a high level, Python applications can be thought of as being interpreted or compiled.

- **Interpreted**: These programs are executed a line at a time, similar to the way a script would work with Bash or another shell.
- **Compiled**: These programs go through a compilation step, which translates them into either machine code, specific for the type of platform on which it is run, or into byte code. Byte code is a format that allows for portability across platforms.

Python applications can be run as interpreted programs or they can be compiled. Interpreted Python programs will be named 'program-name.**py**'. Compiled Python programs will be named 'program-name.**pyc**'.

It can be beneficial to compile code that is unlikely to change, is shared by many applications, or has the need to run fast. Most 'normal' Python applications that you write will be of the '.py' variety, or interpreted.

Note that behind the scenes, Python itself will take your '.py' applications, and compile them into byte code before running them – but that's an implementation detail and not necessarily central to your learning of writing Python applications.



Implementations

CPython

- Reference implementation
- Written in C

Jython

- Compiled to Java bytecode, compatibility with Java codebase

PyPy

- Subset 'RPython'
- Performance

IronPython

- .NET libraries and compabibility

Implementations

It may be helpful to know that there are several different Python 'implementations' that can be used to interpret or compile, and run, your application. These Python implementations include:

- CPython is the original, reference implementation, which is written in C (hence the 'C' in the name). When you run Python normally, you are using this implementation.

- The Jython implementation will compile your Python application into Java byte code, which helps to make it possible to run in a Java-language environment.

- The PyPy implementation was built on a subset of Python called 'RPython', which includes most but not all the Python language. It is built for speed, making Python applications more comparable in terms of performance to applications written in other languages such as Java and C.

- The IronPython implementation is used for creating '.NET' compatible applications, suitable for executing in those environments.