Реализация Jython состоит из Java-классов, которые выполняют компиляцию программного кода на языке Python в байт-код Java и затем передают полученный байт-код виртуальной машине Java (JVM).

Документация по Jython (это реализация Python на Java-платформе) отмечает, что Jython обладает следующими неоспоримыми преимуществами над другими языками, использующими Java-байт-код:

• Jython-код динамически компилирует байт-коды Java, хотя возможна и статическая компиляция, что позволяет писать апплеты, сервлеты и т.п.;

• Поддерживает объектно-ориентированную модель Java, в том числе, возможность наследовать от абстрактных Java-классов;

• Jython является реализацией Python - языка с практичным синтаксисом, обладающего большой выразительностью, что позволяет сократить сроки разработки приложений в разы.

Правда, имеются и некоторые ограничения по сравнению с "обычным" Python. Например, Java не поддерживает множественного наследования, поэтому в некоторых версиях Jython нельзя наследовать классы от нескольких Java-классов (в то же время, множественное наследование поддерживается для Python-классов). Следующий пример (файл lines.py) показывает полную интеграцию Java-классов с интерпретатором Python:

# Импортируются модули из Java

from java.lang import System

from java.awt import \*

# А это модуль из Jython

import random

# Класс для рисования линий на рисунке

class Lines(Canvas):

# Реализация метода paint()

def paint(self, g):

X, Y = self.getSize().width, self.getSize().height

label.setText("%s x %s" % (X, Y))

for i in range(100):

x1, y1 = random.randint(1, X), random.randint(1, Y)

x2, y2 = random.randint(1, X), random.randint(1, Y)

g.drawLine(x1, y1, x2, y2)

# Метки, кнопки и т.п.

panel = Panel(layout=BorderLayout())

label = Label("Size", Label.RIGHT)

panel.add(label, "North")

button = Button("QUIT", actionPerformed=lambda e: System.exit(0))

panel.add(button, "South")

lines = Lines()

panel.add(lines, 'Center')

# Запуск панели в окне

import pawt

pawt.test(panel, size=(240, 240))

Программы на Jython можно компилировать в Java и собирать в jar-архивы. Для создания jar-архива на основе модуля (или пакета) можно применить команду jythonc, которая входит в комплект Jython. Из командной строки это можно сделать примерно так:

jythonс -d -c -j lns.jar lines.py

Для запуска приложения достаточно запустить lines из командной строки:

java -classpath "$CLASSPATH" lines

В переменной $CLASSPATH должны быть пути к архивам lns.jar и jython.jar.