JYTHON

tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About the Tutorial

Jython is the JVM implementation of the Python programming language. It is designed to run on the Java platform. Jython was created in 1997 by Jim Hugunin. It closely follows the standard Python implementation called CPython. Jython 2.7.0 was released in May 2015, which corresponds to CPython 2.7.

This is an introductory tutorial, which covers the basics of Jython and explains how to handle its various modules and sub-modules.

Audience

This tutorial will be helpful for Java programmers who want to utilize important features of Python i.e. Simple Syntax, Rich Data Types and Rapid Application Development in Java code. This will also be useful for Pythonistas to import feature Java class library into the Python Environment.

This tutorial is made to make the programmers comfortable in getting started with Jython and its various functions.

Prerequisites

Since Jython helps in integrating two very popular programming technologies namely Java and Python, a reasonable knowledge of both languages is required.

Copyright and Disclaimer

Table of Contents

1. JYTHON – OVERVIEW

Jython is the JVM implementation of the Python programming language. It is designed to run on the Java platform. A Jython program can import and use any Java class. Just as Java, Jython program compiles to **bytecode**. One of the main advantages is that a user interface designed in Python can use GUI elements of **AWT**, **Swing** or **SWT Package**.

Jython, which started as JPython and was later renamed, follows closely the standard Python implementation called **CPython** as created by **Guido Van Rossum**. Jython was created in 1997 by **Jim Hugunin**. Jython 2.0 was released in 1999. Since then, Jython 2.x releases correspond to equivalent CPython releases. Jython 2.7.0 released in May 2015, corresponds to CPython 2.7. Development of Jython 3.x is under progress.

Difference between Python and Java

Following are the differences between Python and Java:

- Python is a dynamically typed language. Hence, the type declaration of variable is not needed. Java on the other hand is a statically typed language, which means that the type declaration of variable is mandatory and cannot be changed.
- Python has only unchecked exceptions, whereas Java has both checked and unchecked exceptions.
- Python uses indents for scoping, while Java uses matching curly brackets.
- Since Python is an interpreter-based language, it has no separate compilation steps. A Java program however needs to be compiled to bytecode and is in turn executed by a JVM.
- Python supports multiple inheritance, but in Java, multiple inheritance is not possible. It however has implementation of an interface.
- Compared to Java, Python has a richer built-in data structures (lists, dicts, tuples, everything is an object).

Difference between Python and Jython

Following are the differences between Python and Jython:

- Reference implementation of Python, called CPython, is written in C language. Jython on the other hand is completely written in Java and is a JVM implementation.
- Standard Python is available on multiple platforms. Jython is available for any platform with a JVM installed on it.
- Standard Python code compiles to a **.pyc** file, while Jython program compiles to a **.class** file.
- Python extensions can be written in C language. Extensions for Jython are written in Java.

- Jython is truly multi-threaded in nature. Python however uses the **Global Interpreter Lock** (GIL) mechanism for the purpose.
- Both implementations have different garbage collection mechanisms.

In the next chapter, we will learn how to import the Java libraries in Jython.

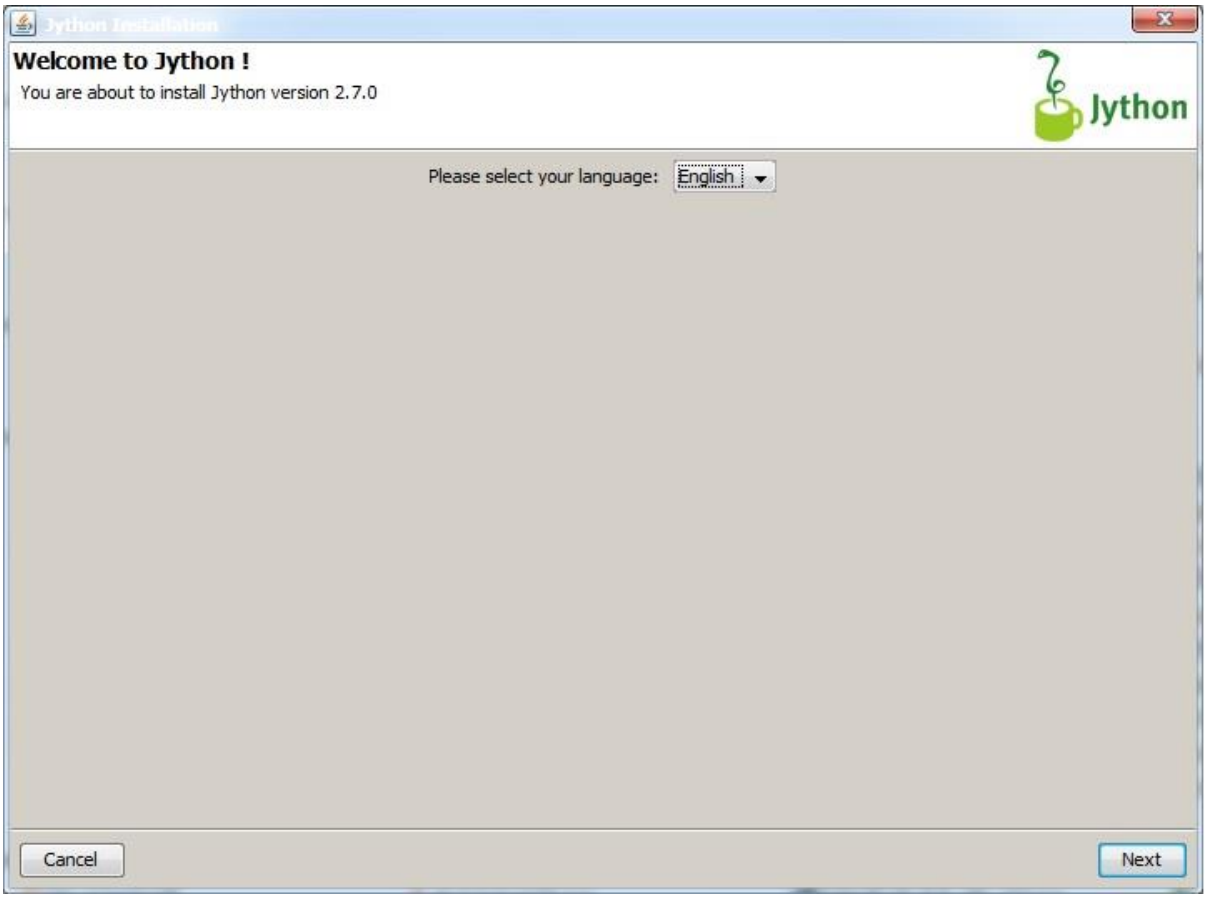
2. JYTHON – INSTALLATION

Before installation of Jython 2.7, ensure that the system has **JDK 7** or more installed. Jython is available in the form of an executable jar file. Download it from – http://www.jython.org/downloads.html and either double click on its icon or run the following command:

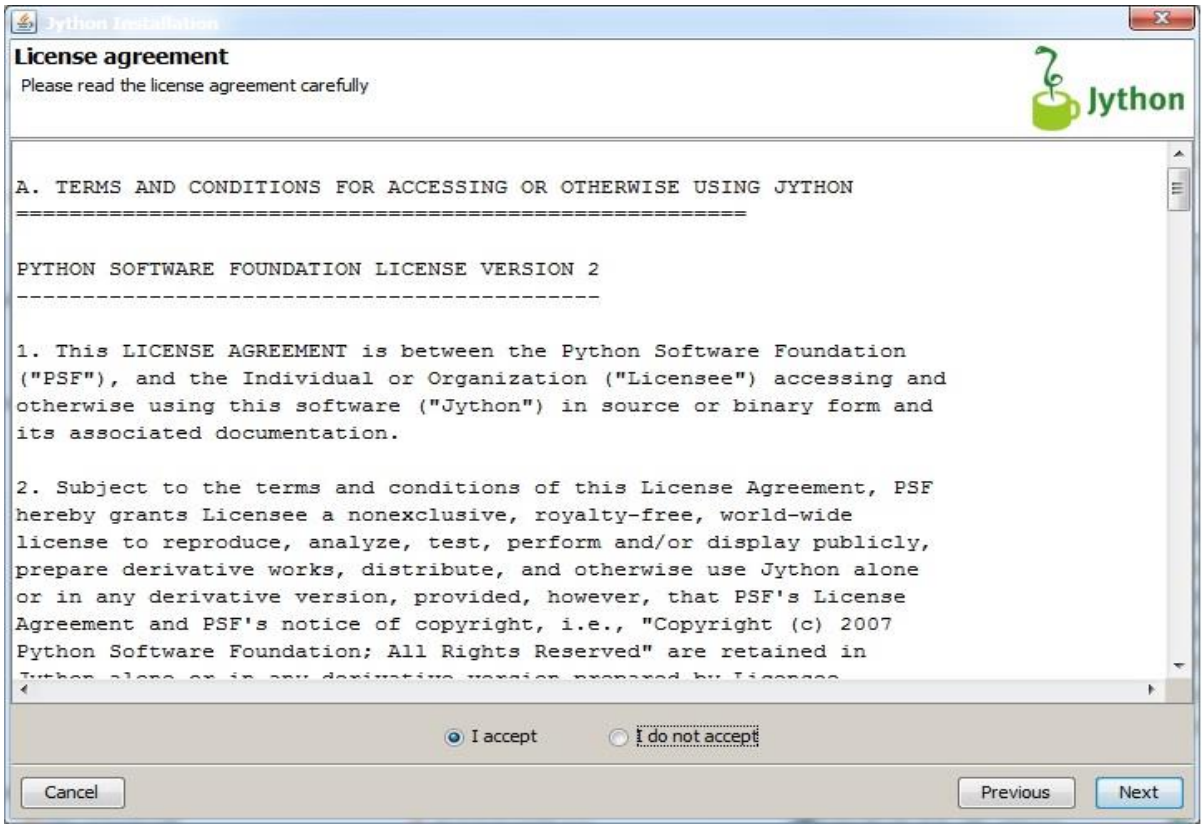
```
java -jar jython_installer-2.7.0.jar
```

An installation wizard will commence with which installation options have to be given. Here is the systematic installation procedure.

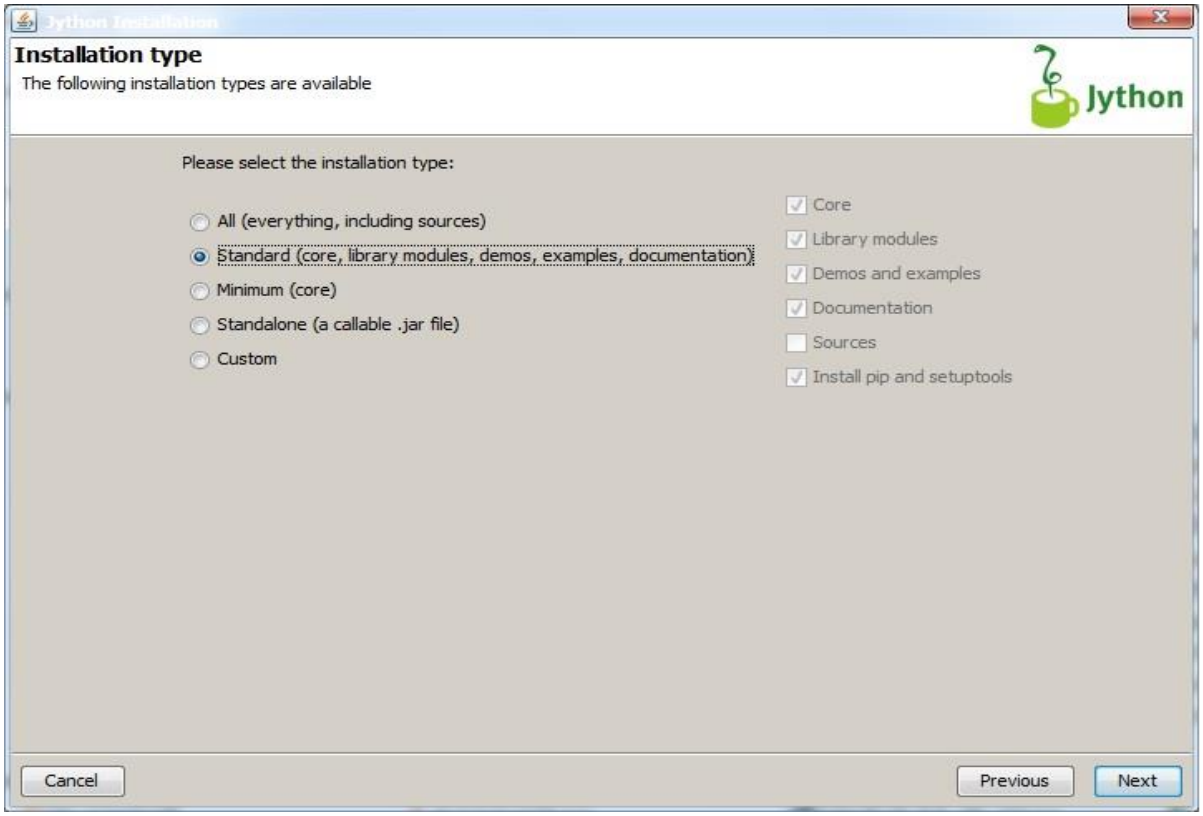
The first step in the wizard asks you to select the language.



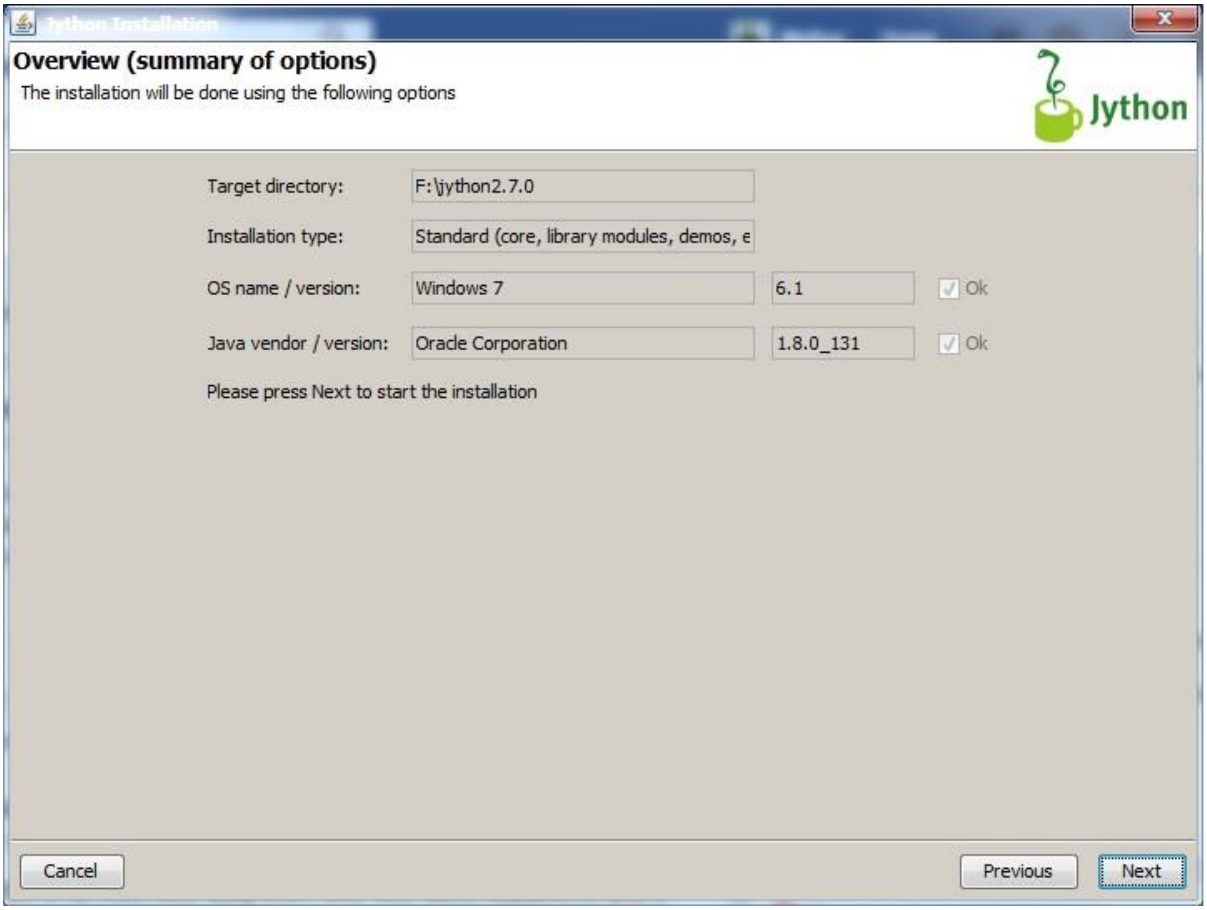
The second step prompts you to accept the licence agreement.



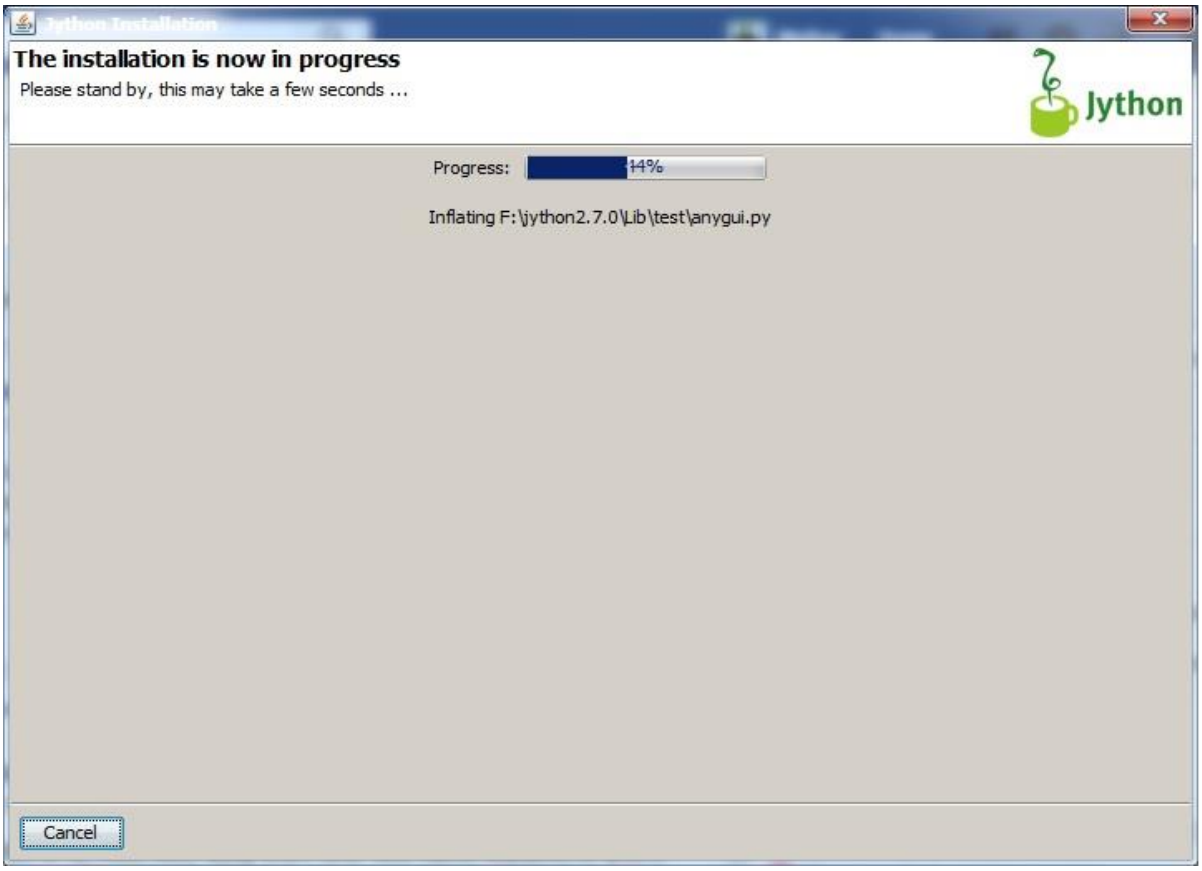
In the next step, choose the installation type. It is recommended to choose the Standard installation.



The next screen asks your confirmation about your options and proceeds to complete the installation.



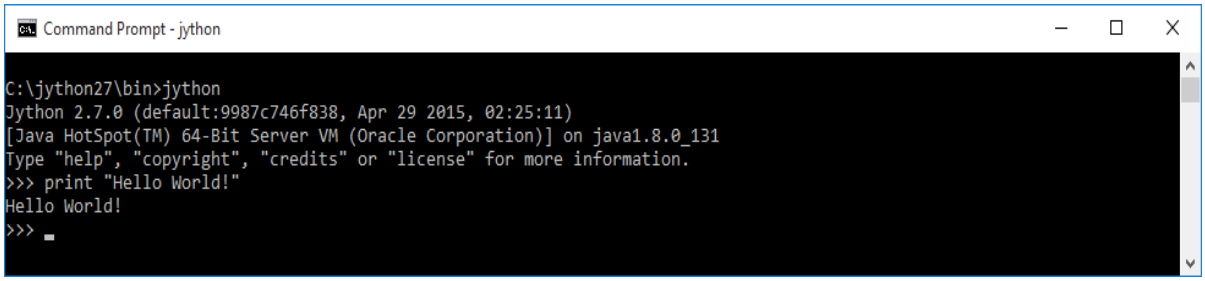
The installation procedure might take some time to complete.



After the installation is complete, invoke **jython.exe** from the bin directory inside the destination directory. Assuming that Jython is installed in **C:\jython27**, execute the following from the command line.

```
C:\jython27\bin\jython
```

A Python prompt (>>>) will appear, in front of which any Python statement or Python script can be executed.



3. JYTHON – IMPORTING JAVA LIBRARIES

One of the most important features of Jython is its ability to import Java classes in a Python program. We can import any java package or class in Jython, just as we do in a Java program. The following example shows how the **java.util** packages are imported in Python (Jython) script to declare an object of the Date class.

```
from java.util import Date
d = Date()
print d
```

Save and run the above code as **UtilDate.py** from the command line. Instance of the current date and time will be displayed.

```
C:\jython27\bin>jython UtilDate.py
Sun Jul 09 00:05:43 IST 2017
```

The following packages from the Java library are more often imported in a Jython program mainly because standard Python library either does not have their equivalents or are not as good.

- Servlets
- JMS
- J2EE
- Javadoc
- Swing is considered superior to other GUI toolkits

Any Java package for that matter can be imported in a Jython script. Here, the following java program is stored and compiled in a package called **foo**.

```
package foo;
public class HelloWorld {
     public void hello() {
        System.out.println("Hello World!");
     }

     public void hello(String name) {
        System.out.printf("Hello %s!", name);
     }
```

```
}
```

This **HelloWorld.class** is imported in the following Jython Script. Methods in this class can be called from the Jython script **importex.py**.

```
from foo import HelloWorld
h = HelloWorld()
h.hello()
h.hello("TutorialsPoint")
```

Save and execute the above script from the command line to get following output.

```
C:\jython27\bin>jython importex.py
Hello World!
Hello TutorialsPoint!
```

4. JYTHON – VARIABLES AND DATA TYPES

Variables are named locations in computer's memory. Each variable can hold one piece of data in it. Unlike Java, Python is a dynamically typed language. Hence while using Jython also; prior declaration of data type of variable is not done. Rather than the type of variable deciding which data can be stored in it, the data decides the type of variable.

In the following example, a variable is assigned an integer value. Using the type() built-in function, we can verify that the type of variable is an integer. But, if the same variable is assigned a string, the type() function will string as the type of same variable.

```
> x=10
>>> type(x)
<class 'int'>

>>> x="hello"
>>> type(x)
<class 'str'>
```

This explains why Python is called a dynamically typed language.

The following Python built-in data types can also be used in Jython:

- Number
- String
- List
- Tuple
- Dictionary

Python recognizes numeric data as a number, which may be an integer, a real number with floating point or a complex number. String, List and Tuple data types are called sequences.

Jython Numbers

In Python, any signed integer is said to be of type 'int'. To express a long integer, letter 'L' is attached to it. A number with a decimal point separating the integer part from a fractional component is called 'float'. The fractional part may contain an exponent expressed in the scientific notation using 'E' or 'e'.

A Complex number is also defined as numeric data type in Python. A complex number contains a real part (a floating-point number) and an imaginary part having 'j' attached to it.

In order to express a number in the Octal or the Hexadecimal representation, **0O** or **0X** is prefixed to it. The following code block gives examples of different representations of numbers in Python.

```
int     -> 10, 100, -786, 80
long    -> 51924361L, -0112L, 47329487234L
float   -> 15.2, -21.9, 32.3+e18, -3.25E+101
complex -> 3.14j, 45.j, 3e+26J, 9.322e-36j
```

Jython Strings

A string is any sequence of characters enclosed in single (e.g. 'hello'), double (e.g. "hello") or triple (e.g. '''hello''' o """hello""") quotation marks. Triple quotes are especially useful if content of the string spans over multiple lines.

The Escape sequence characters can be included verbatim in triple quoted string. The following examples show different ways to declare a string in Python.

```
str='hello how are you?'
str="Hello how are you?"
str="""this is a long string that is made up of
several lines and non-printable characters such as
TAB ( \t ) and they will show up that way when displayed.
NEWLINEs within the string, whether explicitly given like
this within the brackets [ \n ], or just a NEWLINE within
the variable assignment will also show up.
"""
```

The third string when printed, will give the following output.

```
this is a long string that is made up of
several lines and non-printable characters such as
TAB (        ) and they will show up that way when displayed.
NEWLINEs within the string, whether explicitly given like
this within the brackets [
 ], or just a NEWLINE within
the variable assignment will also show up.
```

Jython Lists

A List is a sequence data type. It is a collection of comma-separated items, not necessarily of the same type, stored in square brackets. Individual item from the List can be accessed using the zero based index.

The following code block summarizes the usage of a List in Python.

```
list1 = ['physics', 'chemistry', 1997, 2000];
list2 = [1, 2, 3, 4, 5, 6, 7 ];
print "list1[0]: ", list1[0]
print "list2[1:5]: ", list2[1:5]
```

The following table describes some of the most common Jython Expressions related to Jython Lists.

Jython Expression	Description
len(List)	Length
List[2]=10	Updation
Del List[1]	Deletion
List.append(20)	Append
List.insert(1,15)	Insertion
List.sort()	Sorting

Jython Tuples

A tuple is an immutable collection of comma-separated data items stored in parentheses. It is not possible to delete or modify an element in tuple, nor is it possible to add an element to the tuple collection. The following code block shows Tuple operations.

```
tup1=('physics','chemistry‘,1997,2000);
tup2=(1, 2, 3, 4, 5, 6, 7 );
print "tup1[0]: ", tup1[0]
print "tup2[1:5]: ", tup2[1:5]
```

Jython Dictionary

The Jython Dictionary is similar to Map class in Java Collection framework. It is a collection of key-value pairs. Pairs separated by comma are enclosed in curly brackets. A Dictionary object does not follow zero based index to retrieve element inside it as they are stored by hashing technique.

The same key cannot appear more than once in a dictionary object. However, more than one key can have same associated values. Different functions available with Dictionary object are explained below:

```
dict={'011':'New Delhi','022':'Mumbai','033':'Kolkata'}
print "dict[‘011’]: ",dict['011']
print "dict['Age']: ", dict['Age']
```

The following table describes some of the most common Jython Expressions related to Dictionary.

Jython Expression	Description
dict.get(‘011’)	Search
len(dict)	Length
dict[‘044’]=‘Chennai’	Append
del dict[‘022’]	Delete
dict.keys()	list of keys
dict.values()	List of values
dict.clear()	Removes all elements