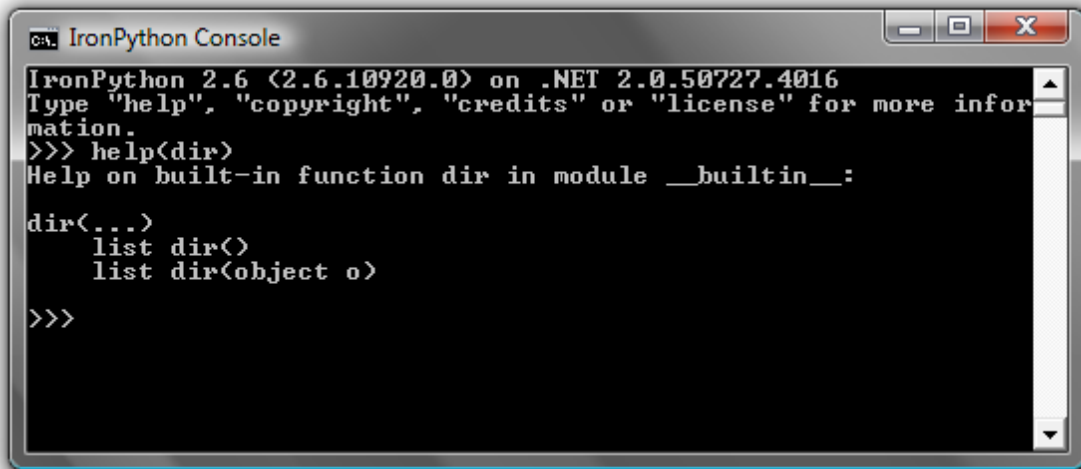


IronPython Console

In this article we take a deep look at IronPython Interactive Console. We will see some basics of Python and IronPython to become acquainted with the IronPython environment.



```
IronPython 2.6 (2.6.10920.0) on .NET 2.0.50727.4016
Type "help", "copyright", "credits" or "license" for more information.
>>> help(dir)
Help on built-in function dir in module __builtin__:

dir(...)
list dir()
list dir(object o)

>>>
```

IronPython interactive interpreter console is the easiest way to start using IronPython. We can easily explore whole IronPython world which includes ironpython modules, .Net libraries and Python Built-in's. For IronPython starters it's a great way to explore .Net Libraries. We can use IronPython Interactive Interpreter for start to developing WinForms, COM objects, WPF, XNA, Embedding C#, Silverlight, ... projects.

Let's check out some common commands which we will use for exploring Ipy on it.

help()

help() function that is particularly useful for exploring objects at the interactive console with dir() function.

help(dir)

With this command we can easily explore any object in IronPython!

dir()

This function returns all the members of an object as a list of strings.

>>> If we run dir() function we can see which modules included in ironpython current work space.

```
>>>dir(help)
```

returns list of members for help function.

```
>>>help.__doc__
```

With this command it returns how can we use help function. By the "__doc__" element we can easily explore .NET libraries which we don't know.

We can use IronPython Console as a calculator :)

```
>>>5+12
17
```



|İbrahim KIVANÇ
|ik@ibrahimkivanc.com
|www.ibrahimkivanc.com

Without variable pre-declaration of data types we can assign a value to a variable.

```
>>>ipy = "ironpython"
```

To get output we use print command, it is simple and easy as "print ipy". Programming is now very funny :)

```
>>>print ipy
ironpython
```

We can use every letter of a word as an element of array in IPy loop. As you see we didn't declare "i" in "for statement".

```
>>>for i in ipy:
...   print i
i
r
o
n
p
y
t
h
o
n
```

we assigned ipy variable as a string now we will assign it as an integer.

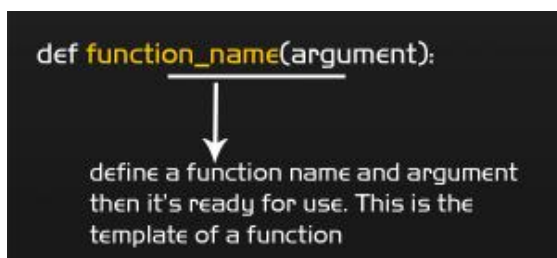
```
>>>ipy=10
>>>print ipy
10
```

Let's assign ipy variable value as an array.

```
>>>ipy = ['iron','python','ironpython']
>>>print ipy
['iron','python','ironpython']
```

As you see there is no error with assigning of data types. We can use same variable for several data types.

With functions we can develop application more programmatically.



we define functions with "def" like above template.



```
def function_name(argument):
    print argument
```

↓

Functions need to enter a space before it's content. We use it like C# curly braces.

Functions need to enter space before print. First line which include "def" and function name, arguments. We use whitespace in function code. There is same situation for loops and statements. Levels of spaces similar like C# curly braces; tell us where functions start and where functions end, also tell us nested structures.

```
>>>def addition(a,b):
...     return a+b

>>>addition(3,8)
11

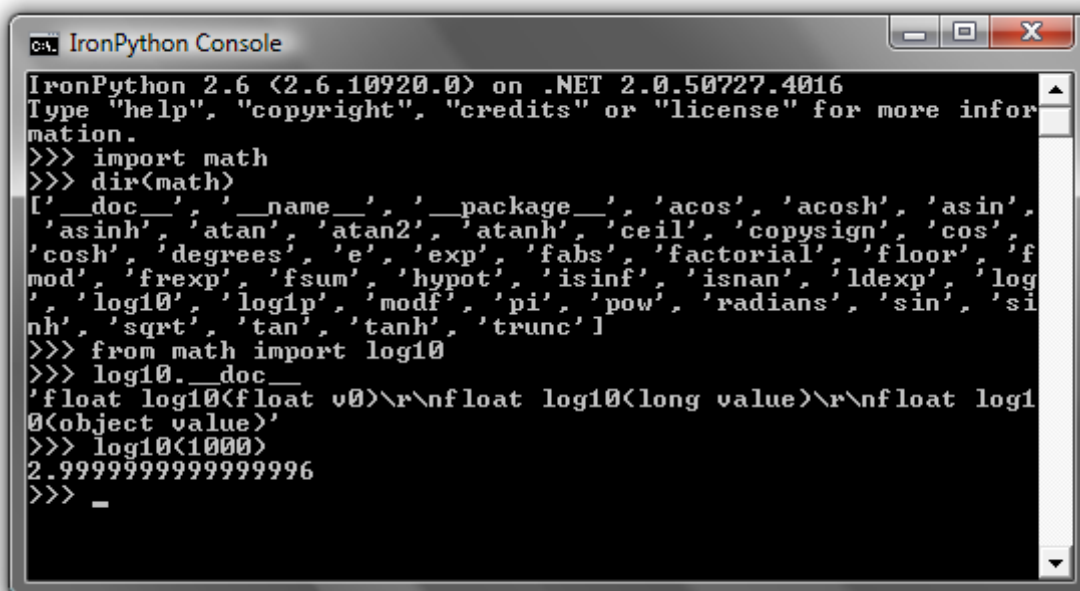
>>>addition("Iron","Python")
IronPython
```

The IronPython code above, we can add 2 string or 2 integer.

After importing libraries we can start developing.

```
>>>import math
>>>dir(math)
>>>from math import log10 #we imported log10 function from math library
>>>log10.__doc #we can see instruction of log10
>>>log10(1000) #we can use this function after lookin doc
```

After the enter codes above, the console contents will be like below.



```
IronPython 2.6 (2.6.10920.0) on .NET 2.0.50727.4016
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> dir(math)
['_doc_', '_name_', '_package_', 'acos', 'acosh', 'asin', 'asinh', 'atan', 'atan2', 'atanh', 'ceil', 'copysign', 'cos', 'cosh', 'degrees', 'e', 'exp', 'fabs', 'factorial', 'floor', 'frexp', 'fsum', 'hypot', 'isinf', 'isnan', 'ldexp', 'log', 'log10', 'log1p', 'modf', 'pi', 'pow', 'radians', 'sin', 'sinh', 'sqrt', 'tan', 'tanh', 'trunc']
>>> from math import log10
>>> log10.__doc_
'float log10(float v0)\r\nfloat log10(long value)\r\nfloat log10(object value)'
>>> log10(1000)
2.9999999999999996
>>> _
```

As you see finding a module and using it in IronPython is too easy!



For exit the IronPython Interactive console; Ctrl+Z or F6 followed by Enter. You can use ^Z or exit() command

If you have any questions or discover any errors / typos please let me know ik@ibrahimkivanc.com

All The Best!



| İbrahim KIVANÇ
| ik@ibrahimkivanc.com
| www.ibrahimkivanc.com