

# Variable names – Which are allowed?

How do we check if a variable name is a keyword or built-in function?

# How to check if a variable name is a keyword?

```
In [1]: 1 import keyword  
        2 'variable_name' in keyword.kwlist
```

```
Out[1]: False
```

```
In [2]: 1 'def' in keyword.kwlist
```

```
Out[2]: True
```

# How to list all keywords?

```
In [5]: ▶ 1 print(keyword.kwlist)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async',  
, 'await', 'break', 'class', 'continue', 'def', 'del',  
, 'elif', 'else', 'except', 'finally', 'for', 'from', 'gl  
obal', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal',  
, 'not', 'or', 'pass', 'raise', 'return', 'try', 'whil  
e', 'with', 'yield']
```

# How to check if a variable name is a built-in function?

```
In [9]: ▶ 1 import builtins  
        2 'variable_name' in dir(builtins)
```

```
Out[9]: False
```

```
In [10]: ▶ 1 'max' in dir(builtins)
```

```
Out[10]: True
```

# How to list all built-in functions?

```
In [6]: ▶ 1 print(dir(builtins))

['ArithmeticError', 'AssertionError', 'AttributeError', 'BaseException', 'BlockingIOError', 'BrokenPipeError', 'BufferError', 'BytesWarning', 'ChildProcessError', 'ConnectionAbortedError', 'ConnectionError', 'ConnectionRefusedError', 'ConnectionResetError', 'DeprecationWarning', 'EOFError', 'Ellipsis', 'EnvironmentError', 'Exception', 'False', 'FileExistsError', 'FileNotFoundError', 'FloatingPointError', 'FutureWarning', 'GeneratorExit', 'IOError', 'ImportError', 'ImportWarning', 'IndentationError', 'IndexError', 'InterruptedError', 'IsADirectoryError', 'KeyError', 'KeyboardInterrupt', 'LookupError', 'MemoryError', 'ModuleNotFoundError', 'NameError', 'None', 'NotADirectoryError', 'NotImplemented', 'NotImplementedError', 'OSError', 'OverflowError', 'PendingDeprecationWarning', 'PermissionError', 'ProcessLookupError', 'RecursionError', 'ReferenceError', 'ResourceWarning', 'RuntimeError', 'RuntimeWarning', 'StopAsyncIteration', 'StopIteration', 'SyntaxError', 'SyntaxWarning', 'SystemError', 'SystemExit', 'TabError', 'TimeoutError', 'True', 'TypeError', 'UnboundLocalError', 'UnicodeDecodeError', 'UnicodeEncodeError', 'UnicodeError', 'UnicodeTranslateError', 'UnicodeWarning', 'UserWarning', 'ValueError', 'Warning', 'WindowsError', 'ZeroDivisionError', '__IPYTHON__', '__build_class__', '__debug__', '__doc__', '__import__', '__loader__', '__name__', '__package__', '__spec__', 'abs', 'all', 'any', 'ascii', 'bin', 'bool', 'breakpoint', 'bytearray', 'bytes', 'callable', 'chr', 'classmethod', 'compile', 'complex', 'copyright', 'credits', 'delattr', 'dict', 'dir', 'display', 'divmod', 'enumerate', 'eval', 'exec', 'filter', 'float', 'format', 'frozenset', 'get_ipython', 'getattr', 'globals', 'hasattr', 'hash', 'help', 'hex', 'id', 'input', 'int', 'isinstance', 'issubclass', 'iter', 'len', 'license', 'list', 'locals', 'map', 'max', 'memoryview', 'min', 'next', 'object', 'oct', 'open', 'ord', 'pow', 'print', 'property', 'range', 'repr', 'reversed', 'round', 'set', 'setattr', 'slice', 'sorted', 'staticmethod', 'str', 'sum', 'super', 'tuple', 'type', 'vars', 'zip']
```

(Note: You do not have to memorize this list.)

# IPython Debugger Variables

These are a few more variables which have a special meaning by default while you are in the debug mode (not otherwise).

If you use the debug mode frequently, please go through this guide once and be aware of the IPython Debugger Variables.

An Example follows hereafter.

Summary: You can use any of the variable names of the IPython Debugger Variables as variable names for your own variables. You just have to be careful to not change their value by typing into the console while you are in the debug mode.

Let's work  
with this  
code:

The image shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows a table with columns: Name, Type, Size, and Value. It is currently empty.

The IPython console at the bottom shows the following output:

```
Python 3.7.4 (default, Aug 9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.8.0 -- An enhanced Interactive Python.

Restarting kernel...

In [1]:
```

The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 15, Column: 1, Memory: 47 %.

Enter the  
debug  
mode.

Go to Line 7.

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\Dom\.spyder-py3\temp.py

temp.py

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

Variable explorer

Name	Type	Size	Value
------	------	------	-------

Help Variable explorer File explorer

IPython console

Console 1/A temp.py/A

```
8 var_2 = 2

ipdb>
ipdb> > c:\users\dom\.spyder-
py3\temp.py(7)<module>()
5 names.
6 """
----> 7 var_1 = 1
8 var_2 = 2
9 a = [1,2]

ipdb>
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 7 Column: 1 Memory: 47 %



Execute  
Line 7.

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\Dom\.spyder-py3\temp.py

temp.py

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

Variable explorer

Name	Type	Size	Value
var_1	int	1	1

Help Variable explorer File explorer

IPython console

Console 1/A temp.py/A

```
8 var_2 = 2
9 a = [1,2]

> c:\users\dom\.spyder-
py3\temp.py(8)<module>()
6 """
7 var_1 = 1
----> 8 var_2 = 2
9 a = [1,2]
10 undisplay = 'maybe?'

ipdb>
ipdb>
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 8 Column: 10 Memory: 47 %

Execute  
Line 8.

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\Dom\.spyder-py3\temp.py

temp.py

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

Variable explorer

Name	Type	Size	Value
var_1	int	1	1
var_2	int	1	2

Help Variable explorer File explorer

IPython console

Console 1/A temp.py/A

```
10 undisplay = 'maybe?'

ipdb> > c:\users\dom\.spyder-
py3\temp.py(9)<module>()
      7 var_1 = 1
      8 var_2 = 2
----> 9 a = [1,2]
      10 undisplay = 'maybe?'
      11 print(var_1)

ipdb>
ipdb>
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 9 Column: 1 Memory: 47 %

Execute  
Line 9.

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\Dom\.spyder-py3\temp.py

temp.py

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

Variable explorer

Name	Type	Size	Value
a	list	2	[1, 2]
var_1	int	1	1
var_2	int	1	2

Help Variable explorer File explorer

IPython console

Console 1/A temp.py/A

```
11 print(var_1)

ipdb> > c:\users\dom\.spyder-
py3\temp.py(10)<module>()
      8 var_2 = 2
      9 a = [1,2]
--> 10 undisplay = 'maybe?'
    11 print(var_1)
    12 print(var_2)

ipdb>
ipdb>
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 10 Column: 1 Memory: 47 %

Execute  
Line 10.

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\Dom\.spyder-py3\temp.py

temp.py

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

Variable explorer

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1
var_2	int	1	2

Help Variable explorer File explorer

IPython console

Console 1/A temp.py/A

```
12 print(var_2)

ipdb> > c:\users\dom\.spyder-
py3\temp.py(11)<module>()
      9 a = [1,2]
     10 undisplay = 'maybe?'
--> 11 print(var_1)
     12 print(var_2)
     13 print(a)

ipdb>
ipdb>
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 11 Column: 1 Memory: 47 %

Execute  
Line 11.

(Here is the  
printed  
output)

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the current state of variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1
var_2	int	1	2

The IPython console at the bottom shows the execution of the script, with the output of line 11 highlighted by an orange box and an arrow pointing from the text "(Here is the printed output)":

```
ipdb> 1
> c:\users\dom\.spyder-
py3\temp.py(12)<module>()
10 undisplay = 'maybe?'
11 print(var_1)
---> 12 print(var_2)
13 print(a)
14 print(undisplay)

ipdb>
ipdb>
```

The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 12, Column: 1, Memory: 43 %.

Execute  
Line 12.

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\Dom\.spyder-py3\temp.py

temp.py

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

Variable explorer

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1
var_2	int	1	2

Help Variable explorer File explorer

IPython console

Console 1/A temp.py/A

```
ipdb> 2
> c:\users\dom\.spyder-
py3\temp.py(13)<module>()
    10 undisplay = 'maybe?'
    11 print(var_1)
    12 print(var_2)
----> 13 print(a)
    14 print(undisplay)

ipdb>
ipdb>
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 13 Column: 1 Memory: 43 %

Execute  
Line 13.

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\Dom\.spyder-py3\temp.py

temp.py

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

Variable explorer

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1
var_2	int	1	2

Help Variable explorer File explorer

IPython console

Console 1/A temp.py/A

```
ipdb> [1, 2]
> c:\users\dom\.spyder-
py3\temp.py(14)<module>()
    10 undisplay = 'maybe?'
    11 print(var_1)
    12 print(var_2)
    13 print(a)
---> 14 print(undisplay)

ipdb>
ipdb>
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 14 Column: 1 Memory: 43 %

Execute  
Line 14.

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Symbol finder

Editor - C:\Users\Dom\.spyder-py3\temp.py

temp.py

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

Variable explorer

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1
var_2	int	1	2

Help Variable explorer File explorer

IPython console

Console 1/A temp.py/A

```
ipdb> maybe?
--Return--
None
> c:\users\dom\.spyder-
py3\temp.py(14)<module>()
  10 undisplay = 'maybe?'
  11 print(var_1)
  12 print(var_2)
  13 print(a)
---> 14 print(undisplay)

ipdb>
ipdb>
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 14 Column: 1 Memory: 43 %



And, as usual, one more click on **next line** exits the debugger and brings us back to normal.

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The **Variable explorer** panel on the right shows the current state of variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1
var_2	int	1	2

The **IPython console** at the bottom shows the execution of the script, with the current line of execution highlighted as line 14:

```
> c:\users\dom\.spyder-py3\temp.py(14)<module>()
  10 undisplay = 'maybe?'
  11 print(var_1)
  12 print(var_2)
  13 print(a)
---> 14 print(undisplay)
```

The console also shows the `ipdb> --Return--` prompt, indicating that the debugger has exited.

A green arrow points from the text "next line" in the first block to the **next line** button in the Spyder toolbar.

So far, everything is normal.

Now, let's change some variable values in the debug mode by typing into the console while we execute the program.

Enter the debug mode and execute all lines until including line 10. You see the values of all variables in the Variable Explorer.

The screenshot displays the Spyder Python IDE interface. The main editor window shows a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable Explorer on the right shows the current state of the variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1
var_2	int	1	2

The IPython console at the bottom shows the execution of the script up to line 13:

```
12 print(var_2)

ipdb> > c:\users\dom\.spyder-
py3\temp.py(11)<module>()
      9 a = [1,2]
     10 undisplay = 'maybe?'
----> 11 print(var_1)
      12 print(var_2)
      13 print(a)

ipdb>
ipdb>
```

The status bar at the bottom indicates the current state: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 11, Column: 1, Memory: 43%.

Type the  
following into  
the Console:

`"var_1 = 1000"`.

You see, that  
the variable  
value changes  
to 1000 as  
expected. So  
far, so good.

The screenshot displays the Spyder Python IDE interface. The main editor window shows a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the current state of the variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1000
var_2	int	1	2

The IPython console at the bottom shows the execution of the script:

```
ipdb> > c:\users\dom\.spyder-py3\temp.py(11)<module>()
      9 a = [1,2]
     10 undisplay = 'maybe?'
----> 11 print(var_1)
      12 print(var_2)
      13 print(a)

ipdb>
ipdb> var_1 = 1000

ipdb>
```

Arrows indicate the flow of information: an orange arrow points from the `var_1` variable in the Variable explorer to the `var_1` variable in the code editor; a red arrow points from the `var_1` variable in the code editor to the `var_1` variable in the IPython console; a purple arrow points from the `var_1` variable in the IPython console to the `var_1` variable in the Variable explorer.

Let's try the  
same with  
"var\_2".

It works as  
well.

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the current state of variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1000
var_2	int	1	42

The IPython console at the bottom shows the execution of the script:

```
py3\temp.py(11)<module>()
  9 a = [1,2]
 10 undisplay = 'maybe?'
----> 11 print(var_1)
      12 print(var_2)
      13 print(a)

ipdb>
ipdb> var_1 = 1000
ipdb> var_2 = 42
ipdb>
```

The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 11, Column: 1, Memory: 43 %.

Let's try the same with "a".

It does not work.

This is NOT because of the data type of "a".

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the following variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1000
var_2	int	1	42

The IPython console at the bottom shows the execution of the script:

```
10 undisplay = 'maybe?'
---> 11 print(var_1)
12 print(var_2)
13 print(a)

ipdb>
ipdb> var_1 = 1000

ipdb> var_2 = 42

ipdb> a = 500

ipdb>
```

The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 11, Column: 1, Memory: 43 %.

Let's try the same with "undisplay".

It does not work either.

This is NOT because of the data type of "undisplay".

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the following variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1000
var_2	int	1	42

The IPython console at the bottom shows the execution of the script, with the following output:

```
13 print(a)

ipdb>
ipdb> var_1 = 1000

ipdb> var_2 = 42

ipdb> a = 500

ipdb> undisplay = 'hello'
*** not displaying = 'hello'

ipdb>
```

The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 11, Column: 1, Memory: 43 %.

Note how we  
also got more  
output than we  
thought? What  
does

```
*** not displaying  
= 'hello'
```

mean, and why  
is it there?

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the following variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1000
var_2	int	1	42

The IPython console at the bottom shows the execution of the script, with the following output:

```
13 print(a)

ipdb>
ipdb> var_1 = 1000

ipdb> var_2 = 42

ipdb> a = 500

ipdb> undisplay = 'hello'
*** not displaying = 'hello'

ipdb>
```

The status bar at the bottom indicates the current line is 11, column 1, and memory usage is 43%.



The debugger of Spyder, the IPython Debugger, has some reserved variable names of its own. “undisplay” has a special meaning, so does “a”.

The screenshot displays the Spyder Python IDE interface. The main editor window shows a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the current state of variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1000
var_2	int	1	42

The IPython console at the bottom shows the execution of the code, with the following output:

```
13 print(a)

ipdb>
ipdb> var_1 = 1000

ipdb> var_2 = 42

ipdb> a = 500

ipdb> undisplay = 'hello'
*** not displaying = 'hello'

ipdb>
```

The status bar at the bottom indicates the current line is 11, column 1, and memory usage is 43%.

These special variables allow us to do certain things. If you are interested what you can do with these variables, search the internet.

The screenshot shows the Spyder Python IDE interface. The main editor displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the current state of the variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1000
var_2	int	1	42

The IPython console at the bottom shows the execution of the script, with the output of `print(a)` being `[1, 2]`. The console also shows the assignment of `undisplay` to `'hello'` and a comment `*** not displaying = 'hello'`.

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 11 Column: 1 Memory: 43 %

By assigning new values to these variables, we can “break” the IPython Debugger accidentally. Click the **red square** to stop the current execution.

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following code:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the current state of variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1
var_2	int	1	2

The IPython console at the bottom shows the execution of the script, with the current line of execution highlighted as line 12:

```
ipdb> > c:\users\dom\.spyder-py3\temp.py(11)<module>()
9 a = [1,2]
10 undisplay = 'maybe?'
---> 11 print(var_1)
12 print(var_2)
13 print(a)

ipdb>
ipdb>
```

A red square icon in the IPython console window, which is highlighted by a red box and a red arrow from the text "Click the red square", indicates the stop button for the debugger.

Then, you are back to normal (not in the debug mode anymore).

The screenshot shows the Spyder Python IDE interface. The main editor displays a Python script named `temp.py` with the following content:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

The Variable explorer on the right shows the current state of variables:

Name	Type	Size	Value
a	list	2	[1, 2]
undisplay	str	1	maybe?
var_1	int	1	1000
var_2	int	1	42

The IPython console at the bottom shows the execution of the script, with the following output:

```
10 undisplay = 'maybe?'
---> 11 print(var_1)
12 print(var_2)
13 print(a)

ipdb>
ipdb> var_1 = 1000

ipdb> var_2 = 42

ipdb> a = 500

ipdb> undisplay = 'hello'
*** not displaying = 'hello'

ipdb>

In [22]:
```

The status bar at the bottom indicates the current state: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 11, Column: 1, Memory: 49 %.

How do we know  
which variables  
have a special  
meaning?

Enter the debug  
mode again.

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `temp.py` with the following content:

```
1#!/usr/bin/env python3
2"""
3IPython Debugger
4reserved variable
5names.
6"""
7var_1 = 1
8var_2 = 2
9a = [1,2]
10undisplay = 'maybe?'
11print(var_1)
12print(var_2)
13print(a)
14print(undisplay)
15
```

A blue arrow points from the text "Enter the debug mode again." to the `Debug` button in the top toolbar.

The `Variable explorer` panel on the right shows a table with columns `Name`, `Type`, `Size`, and `Value`. It is currently empty.

The `IPython console` at the bottom shows the following output:

```
In [15]: debugfile('C:/Users/Dom/.spyder-
py3/temp.py', wdir='C:/Users/Dom/.spyder-
py3')
> c:\users\dom\.spyder-
py3\temp.py(6)<module>()
      4 reserved variable
      5 names.
----> 6 """
      7 var_1 = 1
      8 var_2 = 2

ipdb>
ipdb>
```

The status bar at the bottom indicates: `Permissions: RW`, `End-of-lines: CRLF`, `Encoding: UTF-8`, `Line: 6`, `Column: 1`, and `Memory: 49 %`.

Then type “**help**”  
into the Console  
and the **list**  
gets displayed.

Note that you are good to  
use any of these variable  
names for your own  
variables in your code!

The only thing to be careful  
with is to not change their  
value by typing into the  
console WHILE you are in  
the debug mode. (It won’t  
change the value and you  
might “break” the  
debugger”).

