3. Built-in Constants

A small number of constants live in the built-in namespace. They are:

`False`

The false value of the `bool` type. Assignments to `False` are illegal and raise a `SyntaxError`.

`True`

The true value of the `bool` type. Assignments to `True` are illegal and raise a `SyntaxError`.

`None`

The sole value of the type `NoneType`. `None` is frequently used to represent the absence of a value, as when default arguments are not passed to a function. Assignments to `None` are illegal and raise a `SyntaxError`.

`NotImplemented`

Special value which should be returned by the binary special methods (e.g. `__eq__()`, `__lt__()`, `__add__()`, `__rsub__()`, etc.) to indicate that the operation is not implemented with respect to the other type; may be returned by the in-place binary special methods (e.g. `__imul__()`, `__iand__()`, etc.) for the same purpose. Its truth value is true.

> **Note:** When a binary (or in-place) method returns `NotImplemented` the interpreter will try the reflected operation on the other type (or some other fallback, depending on the operator). If all attempts return `NotImplemented`, the interpreter will raise an appropriate exception. Incorrectly returning `NotImplemented` will result in a misleading error message or the `NotImplemented` value being returned to Python code.
> See Implementing the arithmetic operations for examples.

> **Note:** `NotImplementedError` and `NotImplemented` are not interchangeable, even though they have similar names and purposes. See `NotImplementedError` for details on when to use it.

`Ellipsis`

The same as `...`. Special value used mostly in conjunction with extended slicing syntax for user-defined container data types.

`__debug__`

This constant is true if Python was not started with an `-O` option. See also the `assert` statement.

> **Note:** The names `None`, `False`, `True` and `__debug__` cannot be reassigned (assignments to them, even as an attribute name, raise `SyntaxError`), so they can be considered "true" constants.

3.1. Constants added by the `site` module

The `site` module (which is imported automatically during startup, except if the `-S` command-line option is given) adds several constants to the built-in namespace. They are useful for the interactive interpreter shell and should not be used in programs.

`quit`(*code=None*)
`exit`(*code=None*)

Objects that when printed, print a message like "Use quit() or Ctrl-D (i.e. EOF) to exit", and when called, raise `SystemExit` with the specified exit code.

`copyright`
`credits`

Objects that when printed or called, print the text of copyright or credits, respectively.

`license`

Object that when printed, prints the message "Type license() to see the full license text", and when called, displays the full license text in a pager-like fashion (one screen at a time).