Complex Number Objects

Python's complex number objects are implemented as two distinct types when viewed from the C API: one is the Python object exposed to Python programs, and the other is a C structure which represents the actual complex number value. The API provides functions for working with both.

Complex Numbers as C Structures

Note that the functions which accept these structures as parameters and return them as results do so *by value* rather than dereferencing them through pointers. This is consistent throughout the API.

`Py_complex`

The C structure which corresponds to the value portion of a Python complex number object. Most of the functions for dealing with complex number objects use structures of this type as input or output values, as appropriate. It is defined as:

```
typedef struct {
   double real;
   double imag;
} Py_complex;
```

Py_complex **`_Py_c_sum`**(Py_complex *left*, Py_complex *right*)

Return the sum of two complex numbers, using the C `Py_complex` representation.

Py_complex **`_Py_c_diff`**(Py_complex *left*, Py_complex *right*)

Return the difference between two complex numbers, using the C `Py_complex` representation.

Py_complex **`_Py_c_neg`**(Py_complex *complex*)

Return the negation of the complex number *complex*, using the C `Py_complex` representation.

Py_complex **`_Py_c_prod`**(Py_complex *left*, Py_complex *right*)

Return the product of two complex numbers, using the C `Py_complex` representation.

Py_complex **`_Py_c_quot`**(Py_complex *dividend*, Py_complex *divisor*)

Return the quotient of two complex numbers, using the C `Py_complex` representation.

If *divisor* is null, this method returns zero and sets `errno` to `EDOM`.

Py_complex **_Py_c_pow**(Py_complex *num*, Py_complex *exp*)

Return the exponentiation of *num* by *exp*, using the C `Py_complex` representation.

If *num* is null and *exp* is not a positive real number, this method returns zero and sets `errno` to `EDOM`.

Complex Numbers as Python Objects

PyComplexObject

This subtype of `PyObject` represents a Python complex number object.

PyTypeObject **PyComplex_Type**

This instance of `PyTypeObject` represents the Python complex number type. It is the same object as `complex` in the Python layer.

int **PyComplex_Check**(PyObject **p*)

Return true if its argument is a `PyComplexObject` or a subtype of `PyComplexObject`.

int **PyComplex_CheckExact**(PyObject **p*)

Return true if its argument is a `PyComplexObject`, but not a subtype of `PyComplexObject`.

PyObject* **PyComplex_FromCComplex**(Py_complex *v*)

Return value: New reference.

Create a new Python complex number object from a C `Py_complex` value.

PyObject* **PyComplex_FromDoubles**(double *real*, double *imag*)

Return value: New reference.

Return a new `PyComplexObject` object from *real* and *imag*.

double **PyComplex_RealAsDouble**(PyObject **op*)

Return the real part of *op* as a C `double`.

double **PyComplex_ImagAsDouble**(PyObject **op*)

Return the imaginary part of *op* as a C `double`.

Py_complex **PyComplex_AsCComplex**(PyObject **op*)

Return the `Py_complex` value of the complex number *op*.

If *op* is not a Python complex number object but has a `__complex__()` method, this method will first be called to convert *op* to a Python complex number object. Upon failure, this method returns `-1.0` as a real value.