File Objects

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 1)

Unknown directive type "highlight".

```
.. highlight:: c
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 8)

Unknown directive type "index".

```
.. index:: object: file
```

These APIs are a minimal emulation of the Python 2 C API for built-in file objects, which used to rely on the buffered I/O (:c:type:`FILE*`) support from the C standard library. In Python 3, files and streams use the new :mod:`io` module, which defines several layers over the low-level unbuffered I/O of the operating system. The functions described below are convenience C wrappers over these new APIs, and meant mostly for internal error reporting in the interpreter; third-party code is advised to access the :mod:`io` APIs instead.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 10); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 10); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 10); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 20)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyFile_FromFd(int fd, const char *name, const char *mode, int buffering, const char *encoding, const cha

   Create a Python file object from the file descriptor of an already
   opened file *fd*.  The arguments *name*, *encoding*, *errors* and *newline*
   can be ``NULL`` to use the defaults; *buffering* can be *-1* to use the
   default. *name* is ignored and kept for backward compatibility. Return
   ``NULL`` on failure. For a more comprehensive description of the arguments,
   please refer to the :func:`io.open` function documentation.

   .. warning::

     Since Python streams have their own buffering layer, mixing them with
     OS-level file descriptors can produce various issues (such as unexpected
     ordering of data).

   .. versionchanged:: 3.2
      Ignore *name* attribute.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 39)

Unknown directive type "c:function".

```
.. c:function:: int PyObject_AsFileDescriptor(PyObject *p)

   Return the file descriptor associated with *p* as an :c:type:`int`.  If the
   object is an integer, its value is returned.  If not, the
   object's :meth:`~io.IOBase.fileno` method is called if it exists; the
   method must return an integer, which is returned as the file descriptor
   value.  Sets an exception and returns ``-1`` on failure.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 48)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyFile_GetLine(PyObject *p, int n)

   .. index:: single: EOFError (built-in exception)

   Equivalent to ``p.readline([n])``, this function reads one line from the
   object *p*.  *p* may be a file object or any object with a
   :meth:`~io.IOBase.readline`
   method.  If *n* is ``0``, exactly one line is read, regardless of the length of
   the line.  If *n* is greater than ``0``, no more than *n* bytes will be read
   from the file; a partial line can be returned.  In both cases, an empty string
   is returned if the end of the file is reached immediately.  If *n* is less than
   ``0``, however, one line is read regardless of length, but :exc:`EOFError` is
   raised if the end of the file is reached immediately.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 63)

Unknown directive type "c:function".

```
.. c:function:: int PyFile_SetOpenCodeHook(Py_OpenCodeHookFunction handler)

   Overrides the normal behavior of :func:`io.open_code` to pass its parameter
   through the provided handler.

   The handler is a function of type :c:type:`PyObject *(\*)(PyObject *path,
   void *userData)`, where *path* is guaranteed to be :c:type:`PyUnicodeObject`.

   The *userData* pointer is passed into the hook function. Since hook
   functions may be called from different runtimes, this pointer should not
   refer directly to Python state.
```

```
   As this hook is intentionally used during import, avoid importing new modules
   during its execution unless they are known to be frozen or available in
   ``sys.modules``.

   Once a hook has been set, it cannot be removed or replaced, and later calls to
   :c:func:`PyFile_SetOpenCodeHook` will fail. On failure, the function returns
   -1 and sets an exception if the interpreter has been initialized.

   This function is safe to call before :c:func:`Py_Initialize`.

   .. audit-event:: setopencodehook "" c.PyFile_SetOpenCodeHook

   .. versionadded:: 3.8
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 91)

Unknown directive type "c:function".

```
.. c:function:: int PyFile_WriteObject(PyObject *obj, PyObject *p, int flags)

   .. index:: single: Py_PRINT_RAW

   Write object *obj* to file object *p*.  The only supported flag for *flags* is
   :const:`Py_PRINT_RAW`; if given, the :func:`str` of the object is written
   instead of the :func:`repr`.  Return ``0`` on success or ``-1`` on failure; the
   appropriate exception will be set.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)file.rst, line 101)

Unknown directive type "c:function".

```
.. c:function:: int PyFile_WriteString(const char *s, PyObject *p)

   Write string *s* to file object *p*.  Return ``0`` on success or ``-1`` on
   failure; the appropriate exception will be set.
```