Unicode Objects and Codecs

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

Unknown directive type "sectionauthor".

```
.. sectionauthor:: Marc-AndrÃ© Lemburg <mal@lemburg.com>
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 9)

Unknown directive type "sectionauthor".

```
.. sectionauthor:: Georg Brandl <georg@python.org>
```

Unicode Objects

Since the implementation of PEP 393 in Python 3.3, Unicode objects internally use a variety of representations, in order to allow handling the complete range of Unicode characters while staying memory efficient. There are special cases for strings where all code points are below 128, 256, or 65536; otherwise, code points must be below 1114112 (which is the full Unicode range).

:c:type:`Py_UNICODE*` and UTF-8 representations are created on demand and cached in the Unicode object. The :c:type:`Py_UNICODE*` representation is deprecated and inefficient.

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

Unknown interpreted text role "c:type".

Due to the transition between the old APIs and the new APIs, Unicode objects can internally be in two states depending on how they were created:

- "canonical" Unicode objects are all objects created by a non-deprecated Unicode API. They use the most efficient representation allowed by the implementation.
- "legacy" Unicode objects have been created through one of the deprecated APIs (typically :c:func:`PyUnicode_FromUnicode`) and only bear the :c:type:`Py_UNICODE*` representation; you will have to call :c:func:`PyUnicode_READY` on them before calling any other API.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 31); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 31); *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)unicode.rst, line 31); *backlink*

Unknown interpreted text role "c:func".

Note

The "legacy" Unicode object will be removed in Python 3.12 with deprecated APIs. All Unicode objects will be "canonical" since then. See PEP 623 for more information.

Unicode Type

These are the basic Unicode object types used for the Unicode implementation in Python:

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

Unknown directive type "c:type".

```
.. c:type:: Py_UCS4
            Py_UCS2
            Py_UCS1

   These types are typedefs for unsigned integer types wide enough to contain
   characters of 32 bits, 16 bits and 8 bits, respectively.  When dealing with
   single Unicode characters, use :c:type:`Py_UCS4`.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 59)

Unknown directive type "c:type".

```
.. c:type:: Py_UNICODE

   This is a typedef of :c:type:`wchar_t`, which is a 16-bit type or 32-bit type
   depending on the platform.

   .. versionchanged:: 3.3
      In previous versions, this was a 16-bit type or a 32-bit type depending on
      whether you selected a "narrow" or "wide" Unicode version of Python at
      build time.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 70)

Unknown directive type "c:type".

```
.. c:type:: PyASCIIObject
            PyCompactUnicodeObject
            PyUnicodeObject

   These subtypes of :c:type:`PyObject` represent a Python Unicode object.  In
   almost all cases, they shouldn't be used directly, since all API functions
   that deal with Unicode objects take and return :c:type:`PyObject` pointers.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 81)

Unknown directive type "c:var".

```
.. c:var:: PyTypeObject PyUnicode_Type

   This instance of :c:type:`PyTypeObject` represents the Python Unicode type.  It
   is exposed to Python code as ``str``.
```

The following APIs are really C macros and can be used to do fast checks and to access internal read-only data of Unicode objects:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 90)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_Check(PyObject *o)

   Return true if the object *o* is a Unicode object or an instance of a Unicode
   subtype.  This function always succeeds.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 96)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_CheckExact(PyObject *o)

   Return true if the object *o* is a Unicode object, but not an instance of a
   subtype.  This function always succeeds.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 102)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_READY(PyObject *o)

   Ensure the string object *o* is in the "canonical" representation.  This is
   required before using any of the access macros described below.

   .. XXX expand on when it is not required

   Returns ``0`` on success and ``-1`` with an exception set on failure, which in
   particular happens if memory allocation fails.

   .. versionadded:: 3.3

   .. deprecated-removed:: 3.10 3.12
      This API will be removed with :c:func:`PyUnicode_FromUnicode`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 118)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_GET_LENGTH(PyObject *o)

   Return the length of the Unicode string, in code points.  *o* has to be a
   Unicode object in the "canonical" representation (not checked).

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 126)

Unknown directive type "c:function".

```
.. c:function:: Py_UCS1* PyUnicode_1BYTE_DATA(PyObject *o)
                Py_UCS2* PyUnicode_2BYTE_DATA(PyObject *o)
                Py_UCS4* PyUnicode_4BYTE_DATA(PyObject *o)

   Return a pointer to the canonical representation cast to UCS1, UCS2 or UCS4
   integer types for direct character access.  No checks are performed if the
   canonical representation has the correct character size; use
   :c:func:`PyUnicode_KIND` to select the right macro.  Make sure
   :c:func:`PyUnicode_READY` has been called before accessing this.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 139)

Unknown directive type "c:macro".

```
.. c:macro:: PyUnicode_WCHAR_KIND
```

```
                PyUnicode_1BYTE_KIND
                PyUnicode_2BYTE_KIND
                PyUnicode_4BYTE_KIND

      Return values of the :c:func:`PyUnicode_KIND` macro.

      .. versionadded:: 3.3

      .. deprecated-removed:: 3.10 3.12
         ``PyUnicode_WCHAR_KIND`` is deprecated.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 152)

Unknown directive type "c:function".

```
   .. c:function:: unsigned int PyUnicode_KIND(PyObject *o)

      Return one of the PyUnicode kind constants (see above) that indicate how many
      bytes per character this Unicode object uses to store its data.  *o* has to
      be a Unicode object in the "canonical" representation (not checked).

      .. XXX document "0" return value?

      .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 163)

Unknown directive type "c:function".

```
   .. c:function:: void* PyUnicode_DATA(PyObject *o)

      Return a void pointer to the raw Unicode buffer.  *o* has to be a Unicode
      object in the "canonical" representation (not checked).

      .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 171)

Unknown directive type "c:function".

```
   .. c:function:: void PyUnicode_WRITE(int kind, void *data, Py_ssize_t index, \
                                        Py_UCS4 value)

      Write into a canonical representation *data* (as obtained with
      :c:func:`PyUnicode_DATA`).  This macro does not do any sanity checks and is
      intended for usage in loops.  The caller should cache the *kind* value and
      *data* pointer as obtained from other macro calls.  *index* is the index in
      the string (starts at 0) and *value* is the new code point value which should
      be written to that location.

      .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 184)

Unknown directive type "c:function".

```
   .. c:function:: Py_UCS4 PyUnicode_READ(int kind, void *data, Py_ssize_t index)

      Read a code point from a canonical representation *data* (as obtained with
      :c:func:`PyUnicode_DATA`).  No checks or ready calls are performed.

      .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 192)

Unknown directive type "c:function".

```
.. c:function:: Py_UCS4 PyUnicode_READ_CHAR(PyObject *o, Py_ssize_t index)

   Read a character from a Unicode object *o*, which must be in the "canonical"
   representation.  This is less efficient than :c:func:`PyUnicode_READ` if you
   do multiple consecutive reads.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 201)

Unknown directive type "c:macro".

```
.. c:macro:: PyUnicode_MAX_CHAR_VALUE(o)

   Return the maximum code point that is suitable for creating another string
   based on *o*, which must be in the "canonical" representation.  This is
   always an approximation but more efficient than iterating over the string.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 210)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_GET_SIZE(PyObject *o)

   Return the size of the deprecated :c:type:`Py_UNICODE` representation, in
   code units (this includes surrogate pairs as 2 units).  *o* has to be a
   Unicode object (not checked).

   .. deprecated-removed:: 3.3 3.12
      Part of the old-style Unicode API, please migrate to using
      :c:func:`PyUnicode_GET_LENGTH`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 221)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_GET_DATA_SIZE(PyObject *o)

   Return the size of the deprecated :c:type:`Py_UNICODE` representation in
   bytes.  *o* has to be a Unicode object (not checked).

   .. deprecated-removed:: 3.3 3.12
      Part of the old-style Unicode API, please migrate to using
      :c:func:`PyUnicode_GET_LENGTH`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 231)

Unknown directive type "c:function".

```
.. c:function:: Py_UNICODE* PyUnicode_AS_UNICODE(PyObject *o)
                const char* PyUnicode_AS_DATA(PyObject *o)

   Return a pointer to a :c:type:`Py_UNICODE` representation of the object.  The
   returned buffer is always terminated with an extra null code point.  It
   may also contain embedded null code points, which would cause the string
   to be truncated when used in most C functions.  The ``AS_DATA`` form
   casts the pointer to :c:type:`const char *`.  The *o* argument has to be
   a Unicode object (not checked).

   .. versionchanged:: 3.3
      This macro is now inefficient -- because in many cases the
      :c:type:`Py_UNICODE` representation does not exist and needs to be created
      -- and can fail (return ``NULL`` with an exception set).  Try to port the
      code to use the new :c:func:`PyUnicode_nBYTE_DATA` macros or use
      :c:func:`PyUnicode_WRITE` or :c:func:`PyUnicode_READ`.

   .. deprecated-removed:: 3.3 3.12
      Part of the old-style Unicode API, please migrate to using the
      :c:func:`PyUnicode_nBYTE_DATA` family of macros.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 253)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_IsIdentifier(PyObject *o)

   Return ``1`` if the string is a valid identifier according to the language
   definition, section :ref:`identifiers`. Return ``0`` otherwise.

   .. versionchanged:: 3.9
      The function does not call :c:func:`Py_FatalError` anymore if the string
      is not ready.
```

Unicode Character Properties

Unicode provides many different character properties. The most often needed ones are available through these macros which are mapped to C functions depending on the Python configuration.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 271)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISSPACE(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is a whitespace character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 276)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISLOWER(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is a lowercase character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 281)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISUPPER(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is an uppercase character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 286)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISTITLE(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is a titlecase character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 291)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISLINEBREAK(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is a linebreak character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 296)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISDECIMAL(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is a decimal character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 301)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISDIGIT(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is a digit character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 306)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISNUMERIC(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is a numeric character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 311)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISALPHA(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is an alphabetic character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 316)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISALNUM(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is an alphanumeric character.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 321)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_ISPRINTABLE(Py_UCS4 ch)

   Return ``1`` or ``0`` depending on whether *ch* is a printable character.
   Nonprintable characters are those characters defined in the Unicode character
   database as "Other" or "Separator", excepting the ASCII space (0x20) which is
   considered printable.  (Note that printable characters in this context are
   those which should not be escaped when :func:`repr` is invoked on a string.
   It has no bearing on the handling of strings written to :data:`sys.stdout` or
   :data:`sys.stderr`.)
```

These APIs can be used for fast direct character conversions:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 335)

Unknown directive type "c:function".

```
.. c:function:: Py_UCS4 Py_UNICODE_TOLOWER(Py_UCS4 ch)
```

```
   Return the character *ch* converted to lower case.

   .. deprecated:: 3.3
      This function uses simple case mappings.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 343)

Unknown directive type "c:function".

```
.. c:function:: Py_UCS4 Py_UNICODE_TOUPPER(Py_UCS4 ch)

   Return the character *ch* converted to upper case.

   .. deprecated:: 3.3
      This function uses simple case mappings.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 351)

Unknown directive type "c:function".

```
.. c:function:: Py_UCS4 Py_UNICODE_TOTITLE(Py_UCS4 ch)

   Return the character *ch* converted to title case.

   .. deprecated:: 3.3
      This function uses simple case mappings.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 359)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_TODECIMAL(Py_UCS4 ch)

   Return the character *ch* converted to a decimal positive integer.  Return
   ``-1`` if this is not possible.  This macro does not raise exceptions.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 365)

Unknown directive type "c:function".

```
.. c:function:: int Py_UNICODE_TODIGIT(Py_UCS4 ch)

   Return the character *ch* converted to a single digit integer. Return ``-1`` if
   this is not possible.  This macro does not raise exceptions.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 371)

Unknown directive type "c:function".

```
.. c:function:: double Py_UNICODE_TONUMERIC(Py_UCS4 ch)

   Return the character *ch* converted to a double. Return ``-1.0`` if this is not
   possible.  This macro does not raise exceptions.
```

These APIs can be used to work with surrogates:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 379)

Unknown directive type "c:macro".

```
.. c:macro:: Py_UNICODE_IS_SURROGATE(ch)
```

```
   Check if *ch* is a surrogate (``0xD800 <= ch <= 0xDFFF``).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 383)

Unknown directive type "c:macro".

```
.. c:macro:: Py_UNICODE_IS_HIGH_SURROGATE(ch)

   Check if *ch* is a high surrogate (``0xD800 <= ch <= 0xDBFF``).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 387)

Unknown directive type "c:macro".

```
.. c:macro:: Py_UNICODE_IS_LOW_SURROGATE(ch)

   Check if *ch* is a low surrogate (``0xDC00 <= ch <= 0xDFFF``).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 391)

Unknown directive type "c:macro".

```
.. c:macro:: Py_UNICODE_JOIN_SURROGATES(high, low)

   Join two surrogate characters and return a single Py_UCS4 value.
   *high* and *low* are respectively the leading and trailing surrogates in a
   surrogate pair.
```

Creating and accessing Unicode strings

To create Unicode objects and access their basic sequence properties, use these APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 404)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_New(Py_ssize_t size, Py_UCS4 maxchar)

   Create a new Unicode object.  *maxchar* should be the true maximum code point
   to be placed in the string.  As an approximation, it can be rounded up to the
   nearest value in the sequence 127, 255, 65535, 1114111.

   This is the recommended way to allocate a new Unicode object.  Objects
   created using this function are not resizable.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 416)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_FromKindAndData(int kind, const void *buffer, \
                                                    Py_ssize_t size)

   Create a new Unicode object with the given *kind* (possible values are
   :c:macro:`PyUnicode_1BYTE_KIND` etc., as returned by
   :c:func:`PyUnicode_KIND`).  The *buffer* must point to an array of *size*
   units of 1, 2 or 4 bytes per character, as given by the kind.

   If necessary, the input *buffer* is copied and transformed into the
   canonical representation.  For example, if the *buffer* is a UCS4 string
   (:c:macro:`PyUnicode_4BYTE_KIND`) and it consists only of codepoints in
   the UCS1 range, it will be transformed into UCS1
   (:c:macro:`PyUnicode_1BYTE_KIND`).

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 433)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_FromStringAndSize(const char *u, Py_ssize_t size)

   Create a Unicode object from the char buffer *u*.  The bytes will be
   interpreted as being UTF-8 encoded.  The buffer is copied into the new
   object. If the buffer is not ``NULL``, the return value might be a shared
   object, i.e. modification of the data is not allowed.

   If *u* is ``NULL``, this function behaves like :c:func:`PyUnicode_FromUnicode`
   with the buffer set to ``NULL``.  This usage is deprecated in favor of
   :c:func:`PyUnicode_New`, and will be removed in Python 3.12.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 445)

Unknown directive type "c:function".

```
.. c:function:: PyObject *PyUnicode_FromString(const char *u)

   Create a Unicode object from a UTF-8 encoded null-terminated char buffer
   *u*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 451)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_FromFormat(const char *format, ...)

   Take a C :c:func:`printf`\ -style *format* string and a variable number of
   arguments, calculate the size of the resulting Python Unicode string and return
   a string with the values formatted into it.  The variable arguments must be C
   types and must correspond exactly to the format characters in the *format*
   ASCII-encoded string. The following format characters are allowed:

   .. % This should be exactly the same as the table in PyErr_Format.
   .. % The descriptions for %zd and %zu are wrong, but the truth is complicated
   .. % because not all compilers support the %z width modifier -- we fake it
   .. % when necessary via interpolating PY_FORMAT_SIZE_T.
   .. % Similar comments apply to the %ll width modifier and

   .. tabularcolumns:: |l|l|L|

   +-------------------+---------------------+----------------------------------+
   | Format Characters | Type                | Comment                          |
   +===================+=====================+==================================+
   | :attr:`%%`        | *n/a*               | The literal % character.         |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%c`        | int                 | A single character,              |
   |                   |                     | represented as a C int.          |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%d`        | int                 | Equivalent to                    |
   |                   |                     | ``printf("%d")``. [1]_           |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%u`        | unsigned int        | Equivalent to                    |
   |                   |                     | ``printf("%u")``. [1]_           |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%ld`       | long                | Equivalent to                    |
   |                   |                     | ``printf("%ld")``. [1]_          |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%li`       | long                | Equivalent to                    |
   |                   |                     | ``printf("%li")``. [1]_          |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%lu`       | unsigned long       | Equivalent to                    |
   |                   |                     | ``printf("%lu")``. [1]_          |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%lld`      | long long           | Equivalent to                    |
   |                   |                     | ``printf("%lld")``. [1]_         |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%lli`      | long long           | Equivalent to                    |
   |                   |                     | ``printf("%lli")``. [1]_         |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%llu`      | unsigned long long  | Equivalent to                    |
   |                   |                     | ``printf("%llu")``. [1]_         |
   +-------------------+---------------------+----------------------------------+
   | :attr:`%zd`       | Py_ssize_t          | Equivalent to                    |
```

```
|                   |                     | ``printf("%zd")``. [1]_          |
+-------------------+---------------------+----------------------------------+
| :attr:`%zi`       | Py_ssize_t          | Equivalent to                    |
|                   |                     | ``printf("%zi")``. [1]_          |
+-------------------+---------------------+----------------------------------+
| :attr:`%zu`       | size_t              | Equivalent to                    |
|                   |                     | ``printf("%zu")``. [1]_          |
+-------------------+---------------------+----------------------------------+
| :attr:`%i`        | int                 | Equivalent to                    |
|                   |                     | ``printf("%i")``. [1]_           |
+-------------------+---------------------+----------------------------------+
| :attr:`%x`        | int                 | Equivalent to                    |
|                   |                     | ``printf("%x")``. [1]_           |
+-------------------+---------------------+----------------------------------+
| :attr:`%s`        | const char\*        | A null-terminated C character    |
|                   |                     | array.                           |
+-------------------+---------------------+----------------------------------+
| :attr:`%p`        | const void\*        | The hex representation of a C    |
|                   |                     | pointer. Mostly equivalent to    |
|                   |                     | ``printf("%p")`` except that     |
|                   |                     | it is guaranteed to start with   |
|                   |                     | the literal ``0x`` regardless    |
|                   |                     | of what the platform's           |
|                   |                     | ``printf`` yields.               |
+-------------------+---------------------+----------------------------------+
| :attr:`%A`        | PyObject\*          | The result of calling            |
|                   |                     | :func:`ascii`.                   |
+-------------------+---------------------+----------------------------------+
| :attr:`%U`        | PyObject\*          | A Unicode object.                |
+-------------------+---------------------+----------------------------------+
| :attr:`%V`        | PyObject\*,         | A Unicode object (which may be   |
|                   | const char\*        | ``NULL``) and a null-terminated  |
|                   |                     | C character array as a second    |
|                   |                     | parameter (which will be used,   |
|                   |                     | if the first parameter is        |
|                   |                     | ``NULL``).                       |
+-------------------+---------------------+----------------------------------+
| :attr:`%S`        | PyObject\*          | The result of calling            |
|                   |                     | :c:func:`PyObject_Str`.          |
+-------------------+---------------------+----------------------------------+
| :attr:`%R`        | PyObject\*          | The result of calling            |
|                   |                     | :c:func:`PyObject_Repr`.         |
+-------------------+---------------------+----------------------------------+

An unrecognized format character causes all the rest of the format string to be
copied as-is to the result string, and any extra arguments discarded.

.. note::
   The width formatter unit is number of characters rather than bytes.
   The precision formatter unit is number of bytes for ``"%s"`` and
   ``"%V"`` (if the ``PyObject*`` argument is ``NULL``), and a number of
   characters for ``"%A"``, ``"%U"``, ``"%S"``, ``"%R"`` and ``"%V"``
   (if the ``PyObject*`` argument is not ``NULL``).

.. [1] For integer specifiers (d, u, ld, li, lu, lld, lli, llu, zd, zi,
   zu, i, x): the 0-conversion flag has effect even when a precision is given.

.. versionchanged:: 3.2
   Support for ``"%lld"`` and ``"%llu"`` added.

.. versionchanged:: 3.3
   Support for ``"%li"``, ``"%lli"`` and ``"%zi"`` added.

.. versionchanged:: 3.4
   Support width and precision formatter for ``"%s"``, ``"%A"``, ``"%U"``,
   ``"%V"``, ``"%S"``, ``"%R"`` added.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 568)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_FromFormatV(const char *format, va_list vargs)

   Identical to :c:func:`PyUnicode_FromFormat` except that it takes exactly two
   arguments.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 574)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_FromEncodedObject(PyObject *obj, \
                                const char *encoding, const char *errors)

   Decode an encoded object *obj* to a Unicode object.

   :class:`bytes`, :class:`bytearray` and other
   :term:`bytes-like objects <bytes-like object>`
   are decoded according to the given *encoding* and using the error handling
   defined by *errors*. Both can be ``NULL`` to have the interface use the default
   values (see :ref:`builtincodecs` for details).

   All other objects, including Unicode objects, cause a :exc:`TypeError` to be
   set.

   The API returns ``NULL`` if there was an error.  The caller is responsible for
   decref'ing the returned objects.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 592)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_GetLength(PyObject *unicode)

   Return the length of the Unicode object, in code points.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 599)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_CopyCharacters(PyObject *to, \
                                                    Py_ssize_t to_start, \
                                                    PyObject *from, \
                                                    Py_ssize_t from_start, \
                                                    Py_ssize_t how_many)

   Copy characters from one Unicode object into another.  This function performs
   character conversion when necessary and falls back to :c:func:`memcpy` if
   possible.  Returns ``-1`` and sets an exception on error, otherwise returns
   the number of copied characters.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 613)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_Fill(PyObject *unicode, Py_ssize_t start, \
                        Py_ssize_t length, Py_UCS4 fill_char)

   Fill a string with a character: write *fill_char* into
   ``unicode[start:start+length]``.

   Fail if *fill_char* is bigger than the string maximum character, or if the
   string has more than 1 reference.

   Return the number of written character, or return ``-1`` and raise an
   exception on error.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 628)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_WriteChar(PyObject *unicode, Py_ssize_t index, \
                                        Py_UCS4 character)

   Write a character to a string.  The string must have been created through
```

```
:c:func:`PyUnicode_New`.  Since Unicode strings are supposed to be immutable,
the string must not be shared, or have been hashed yet.

This function checks that *unicode* is a Unicode object, that the index is
not out of bounds, and that the object can be modified safely (i.e. that it
its reference count is one).

.. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 642)

Unknown directive type "c:function".

```
.. c:function:: Py_UCS4 PyUnicode_ReadChar(PyObject *unicode, Py_ssize_t index)

   Read a character from a string.  This function checks that *unicode* is a
   Unicode object and the index is not out of bounds, in contrast to the macro
   version :c:func:`PyUnicode_READ_CHAR`.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 651)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Substring(PyObject *str, Py_ssize_t start, \
                                              Py_ssize_t end)

   Return a substring of *str*, from character index *start* (included) to
   character index *end* (excluded).  Negative indices are not supported.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 660)

Unknown directive type "c:function".

```
.. c:function:: Py_UCS4* PyUnicode_AsUCS4(PyObject *u, Py_UCS4 *buffer, \
                                          Py_ssize_t buflen, int copy_null)

   Copy the string *u* into a UCS4 buffer, including a null character, if
   *copy_null* is set.  Returns ``NULL`` and sets an exception on error (in
   particular, a :exc:`SystemError` if *buflen* is smaller than the length of
   *u*).  *buffer* is returned on success.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 671)

Unknown directive type "c:function".

```
.. c:function:: Py_UCS4* PyUnicode_AsUCS4Copy(PyObject *u)

   Copy the string *u* into a new UCS4 buffer that is allocated using
   :c:func:`PyMem_Malloc`.  If this fails, ``NULL`` is returned with a
   :exc:`MemoryError` set.  The returned buffer always has an extra
   null code point appended.

   .. versionadded:: 3.3
```

Deprecated Py_UNICODE APIs

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 684)

Unknown directive type "deprecated-removed".

```
.. deprecated-removed:: 3.3 3.12
```

These API functions are deprecated with the implementation of PEP 393. Extension modules can continue using them, as they will not be removed in Python 3.x, but need to be aware that their use can now cause performance and memory hits.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 691)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_FromUnicode(const Py_UNICODE *u, Py_ssize_t size)

   Create a Unicode object from the Py_UNICODE buffer *u* of the given size. *u*
   may be ``NULL`` which causes the contents to be undefined. It is the user's
   responsibility to fill in the needed data.  The buffer is copied into the new
   object.

   If the buffer is not ``NULL``, the return value might be a shared object.
   Therefore, modification of the resulting Unicode object is only allowed when
   *u* is ``NULL``.

   If the buffer is ``NULL``, :c:func:`PyUnicode_READY` must be called once the
   string content has been filled before using any of the access macros such as
   :c:func:`PyUnicode_KIND`.

   .. deprecated-removed:: 3.3 3.12
      Part of the old-style Unicode API, please migrate to using
      :c:func:`PyUnicode_FromKindAndData`, :c:func:`PyUnicode_FromWideChar`, or
      :c:func:`PyUnicode_New`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 712)

Unknown directive type "c:function".

```
.. c:function:: Py_UNICODE* PyUnicode_AsUnicode(PyObject *unicode)

   Return a read-only pointer to the Unicode object's internal
   :c:type:`Py_UNICODE` buffer, or ``NULL`` on error. This will create the
   :c:type:`Py_UNICODE*` representation of the object if it is not yet
   available. The buffer is always terminated with an extra null code point.
   Note that the resulting :c:type:`Py_UNICODE` string may also contain
   embedded null code points, which would cause the string to be truncated when
   used in most C functions.

   .. deprecated-removed:: 3.3 3.12
      Part of the old-style Unicode API, please migrate to using
      :c:func:`PyUnicode_AsUCS4`, :c:func:`PyUnicode_AsWideChar`,
      :c:func:`PyUnicode_ReadChar` or similar new APIs.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 728)

Unknown directive type "c:function".

```
.. c:function:: Py_UNICODE* PyUnicode_AsUnicodeAndSize(PyObject *unicode, Py_ssize_t *size)

   Like :c:func:`PyUnicode_AsUnicode`, but also saves the :c:func:`Py_UNICODE`
   array length (excluding the extra null terminator) in *size*.
   Note that the resulting :c:type:`Py_UNICODE*` string
   may contain embedded null code points, which would cause the string to be
   truncated when used in most C functions.

   .. versionadded:: 3.3

   .. deprecated-removed:: 3.3 3.12
      Part of the old-style Unicode API, please migrate to using
      :c:func:`PyUnicode_AsUCS4`, :c:func:`PyUnicode_AsWideChar`,
      :c:func:`PyUnicode_ReadChar` or similar new APIs.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 744)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_GetSize(PyObject *unicode)

   Return the size of the deprecated :c:type:`Py_UNICODE` representation, in
   code units (this includes surrogate pairs as 2 units).

   .. deprecated-removed:: 3.3 3.12
      Part of the old-style Unicode API, please migrate to using
      :c:func:`PyUnicode_GET_LENGTH`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 754)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_FromObject(PyObject *obj)

   Copy an instance of a Unicode subtype to a new true Unicode object if
   necessary. If *obj* is already a true Unicode object (not a subtype),
   return the reference with incremented refcount.

   Objects other than Unicode or its subtypes will cause a :exc:`TypeError`.
```

Locale Encoding

The current locale encoding can be used to decode text from the operating system.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 769)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeLocaleAndSize(const char *str, \
                                                        Py_ssize_t len, \
                                                        const char *errors)

   Decode a string from UTF-8 on Android and VxWorks, or from the current
   locale encoding on other platforms. The supported
   error handlers are ``"strict"`` and ``"surrogateescape"``
   (:pep:`383`). The decoder uses ``"strict"`` error handler if
   *errors* is ``NULL``.  *str* must end with a null character but
   cannot contain embedded null characters.

   Use :c:func:`PyUnicode_DecodeFSDefaultAndSize` to decode a string from
   :c:data:`Py_FileSystemDefaultEncoding` (the locale encoding read at
   Python startup).

   This function ignores the :ref:`Python UTF-8 Mode <utf8-mode>`.

   .. seealso::

      The :c:func:`Py_DecodeLocale` function.

   .. versionadded:: 3.3

   .. versionchanged:: 3.7
      The function now also uses the current locale encoding for the
      ``surrogateescape`` error handler, except on Android. Previously, :c:func:`Py_DecodeLocale`
      was used for the ``surrogateescape``, and the current locale encoding was
      used for ``strict``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 799)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeLocale(const char *str, const char *errors)

   Similar to :c:func:`PyUnicode_DecodeLocaleAndSize`, but compute the string
   length using :c:func:`strlen`.

   .. versionadded:: 3.3
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 807)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_EncodeLocale(PyObject *unicode, const char *errors)

   Encode a Unicode object to UTF-8 on Android and VxWorks, or to the current
   locale encoding on other platforms. The
   supported error handlers are ``"strict"`` and ``"surrogateescape"``
   (:pep:`383`). The encoder uses ``"strict"`` error handler if
   *errors* is ``NULL``. Return a :class:`bytes` object. *unicode* cannot
   contain embedded null characters.

   Use :c:func:`PyUnicode_EncodeFSDefault` to encode a string to
   :c:data:`Py_FileSystemDefaultEncoding` (the locale encoding read at
   Python startup).

   This function ignores the :ref:`Python UTF-8 Mode <utf8-mode>`.

   .. seealso::

      The :c:func:`Py_EncodeLocale` function.

   .. versionadded:: 3.3

   .. versionchanged:: 3.7
      The function now also uses the current locale encoding for the
      ``surrogateescape`` error handler, except on Android. Previously,
      :c:func:`Py_EncodeLocale`
      was used for the ``surrogateescape``, and the current locale encoding was
      used for ``strict``.
```

File System Encoding

To encode and decode file names and other environment strings, :c:data:`Py_FileSystemDefaultEncoding` should be used as the encoding, and :c:data:`Py_FileSystemDefaultEncodeErrors` should be used as the error handler (PEP 383 and PEP 529). To encode file names to :class:`bytes` during argument parsing, the `"O&"` converter should be used, passing :c:func:`PyUnicode_FSConverter` as the conversion function:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 839); *backlink*

Unknown interpreted text role "c:data".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 839); *backlink*

Unknown interpreted text role "c:data".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 839); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 839); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 846)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_FSConverter(PyObject* obj, void* result)

   ParseTuple converter: encode :class:`str` objects -- obtained directly or
   through the :class:`os.PathLike` interface -- to :class:`bytes` using
   :c:func:`PyUnicode_EncodeFSDefault`; :class:`bytes` objects are output as-is.
   *result* must be a :c:type:`PyBytesObject*` which must be released when it is
   no longer used.

   .. versionadded:: 3.1

   .. versionchanged:: 3.6
      Accepts a :term:`path-like object`.
```

To decode file names to :class:`str` during argument parsing, the "O&" converter should be used, passing :c:func:`PyUnicode_FSDecoder` as the conversion function:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 859); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 859); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 863)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_FSDecoder(PyObject* obj, void* result)

   ParseTuple converter: decode :class:`bytes` objects -- obtained either
   directly or indirectly through the :class:`os.PathLike` interface -- to
   :class:`str` using :c:func:`PyUnicode_DecodeFSDefaultAndSize`; :class:`str`
   objects are output as-is. *result* must be a :c:type:`PyUnicodeObject*` which
   must be released when it is no longer used.

   .. versionadded:: 3.2

   .. versionchanged:: 3.6
      Accepts a :term:`path-like object`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 877)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeFSDefaultAndSize(const char *s, Py_ssize_t size)

   Decode a string from the :term:`filesystem encoding and error handler`.

   If :c:data:`Py_FileSystemDefaultEncoding` is not set, fall back to the
   locale encoding.

   :c:data:`Py_FileSystemDefaultEncoding` is initialized at startup from the
   locale encoding and cannot be modified later. If you need to decode a string
   from the current locale encoding, use
   :c:func:`PyUnicode_DecodeLocaleAndSize`.

   .. seealso::

      The :c:func:`Py_DecodeLocale` function.

   .. versionchanged:: 3.6
      Use :c:data:`Py_FileSystemDefaultEncodeErrors` error handler.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 897)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeFSDefault(const char *s)

   Decode a null-terminated string from the :term:`filesystem encoding and
   error handler`.

   If :c:data:`Py_FileSystemDefaultEncoding` is not set, fall back to the
   locale encoding.

   Use :c:func:`PyUnicode_DecodeFSDefaultAndSize` if you know the string length.

   .. versionchanged:: 3.6
      Use :c:data:`Py_FileSystemDefaultEncodeErrors` error handler.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 911)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_EncodeFSDefault(PyObject *unicode)

   Encode a Unicode object to :c:data:`Py_FileSystemDefaultEncoding` with the
   :c:data:`Py_FileSystemDefaultEncodeErrors` error handler, and return
   :class:`bytes`. Note that the resulting :class:`bytes` object may contain
   null bytes.

   If :c:data:`Py_FileSystemDefaultEncoding` is not set, fall back to the
   locale encoding.

   :c:data:`Py_FileSystemDefaultEncoding` is initialized at startup from the
   locale encoding and cannot be modified later. If you need to encode a string
   to the current locale encoding, use :c:func:`PyUnicode_EncodeLocale`.

   .. seealso::

      The :c:func:`Py_EncodeLocale` function.

   .. versionadded:: 3.2

   .. versionchanged:: 3.6
      Use :c:data:`Py_FileSystemDefaultEncodeErrors` error handler.
```

wchar_t Support

:c:type:`wchar_t` support for platforms which support it:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 937); *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)unicode.rst, line 939)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_FromWideChar(const wchar_t *w, Py_ssize_t size)

   Create a Unicode object from the :c:type:`wchar_t` buffer *w* of the given *size*.
   Passing ``-1`` as the *size* indicates that the function must itself compute the length,
   using wcslen.
   Return ``NULL`` on failure.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 947)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_AsWideChar(PyObject *unicode, wchar_t *w, Py_ssize_t size)

   Copy the Unicode object contents into the :c:type:`wchar_t` buffer *w*.  At most
   *size* :c:type:`wchar_t` characters are copied (excluding a possibly trailing
   null termination character).  Return the number of :c:type:`wchar_t` characters
   copied or ``-1`` in case of an error.  Note that the resulting :c:type:`wchar_t*`
   string may or may not be null-terminated.  It is the responsibility of the caller
   to make sure that the :c:type:`wchar_t*` string is null-terminated in case this is
   required by the application. Also, note that the :c:type:`wchar_t*` string
   might contain null characters, which would cause the string to be truncated
   when used with most C functions.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 960)

Unknown directive type "c:function".

```
.. c:function:: wchar_t* PyUnicode_AsWideCharString(PyObject *unicode, Py_ssize_t *size)

   Convert the Unicode object to a wide character string. The output string
   always ends with a null character. If *size* is not ``NULL``, write the number
   of wide characters (excluding the trailing null termination character) into
   *\*size*. Note that the resulting :c:type:`wchar_t` string might contain
   null characters, which would cause the string to be truncated when used with
```

```
most C functions. If *size* is ``NULL`` and the :c:type:`wchar_t*` string
contains null characters a :exc:`ValueError` is raised.

Returns a buffer allocated by :c:func:`PyMem_Alloc` (use
:c:func:`PyMem_Free` to free it) on success. On error, returns ``NULL``
and *\*size* is undefined. Raises a :exc:`MemoryError` if memory allocation
is failed.

.. versionadded:: 3.2

.. versionchanged:: 3.7
   Raises a :exc:`ValueError` if *size* is ``NULL`` and the :c:type:`wchar_t*`
   string contains null characters.
```

Built-in Codecs

Python provides a set of built-in codecs which are written in C for speed. All of these codecs are directly usable via the following functions.

Many of the following APIs take two arguments encoding and errors, and they have the same semantics as the ones of the built-in :func:`str` string object constructor.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 990); *backlink*

Unknown interpreted text role "func".

Setting encoding to `NULL` causes the default encoding to be used which is UTF-8. The file system calls should use :c:func:`PyUnicode_FSConverter` for encoding file names. This uses the variable :c:data:`Py_FileSystemDefaultEncoding` internally. This variable should be treated as read-only: on some systems, it will be a pointer to a static string, on others, it will change at run-time (such as when the application invokes setlocale).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 994); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 994); *backlink*

Unknown interpreted text role "c:data".

Error handling is set by errors which may also be set to `NULL` meaning to use the default handling defined for the codec. Default error handling for all built-in codecs is "strict" (:exc:`ValueError` is raised).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1002); *backlink*

Unknown interpreted text role "exc".

The codecs all use a similar interface. Only deviations from the following generic ones are documented for simplicity.

Generic Codecs

These are the generic codec APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1016)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Decode(const char *s, Py_ssize_t size, \
                              const char *encoding, const char *errors)

   Create a Unicode object by decoding *size* bytes of the encoded string *s*.
   *encoding* and *errors* have the same meaning as the parameters of the same name
   in the :func:`str` built-in function.  The codec to be used is looked up
   using the Python codec registry.  Return ``NULL`` if an exception was raised by
   the codec.
```

**System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1026)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsEncodedString(PyObject *unicode, \
                              const char *encoding, const char *errors)

   Encode a Unicode object and return the result as Python bytes object.
   *encoding* and *errors* have the same meaning as the parameters of the same
   name in the Unicode :meth:`~str.encode` method. The codec to be used is looked up
   using the Python codec registry. Return ``NULL`` if an exception was raised by
   the codec.
```

UTF-8 Codecs

These are the UTF-8 codec APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1042)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUTF8(const char *s, Py_ssize_t size, const char *errors)

   Create a Unicode object by decoding *size* bytes of the UTF-8 encoded string
   *s*. Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1048)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUTF8Stateful(const char *s, Py_ssize_t size, \
                              const char *errors, Py_ssize_t *consumed)

   If *consumed* is ``NULL``, behave like :c:func:`PyUnicode_DecodeUTF8`. If
   *consumed* is not ``NULL``, trailing incomplete UTF-8 byte sequences will not be
   treated as an error. Those bytes will not be decoded and the number of bytes
   that have been decoded will be stored in *consumed*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1057)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsUTF8String(PyObject *unicode)

   Encode a Unicode object using UTF-8 and return the result as Python bytes
   object.  Error handling is "strict".  Return ``NULL`` if an exception was
   raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1064)

Unknown directive type "c:function".

```
.. c:function:: const char* PyUnicode_AsUTF8AndSize(PyObject *unicode, Py_ssize_t *size)

   Return a pointer to the UTF-8 encoding of the Unicode object, and
   store the size of the encoded representation (in bytes) in *size*.  The
   *size* argument can be ``NULL``; in this case no size will be stored.  The
   returned buffer always has an extra null byte appended (not included in
   *size*), regardless of whether there are any other null code points.

   In the case of an error, ``NULL`` is returned with an exception set and no
   *size* is stored.

   This caches the UTF-8 representation of the string in the Unicode object, and
   subsequent calls will return a pointer to the same buffer.  The caller is not
   responsible for deallocating the buffer.

   .. versionadded:: 3.3

   .. versionchanged:: 3.7
```

```
      The return type is now ``const char *`` rather of ``char *``.

   .. versionchanged:: 3.10
      This function is a part of the :ref:`limited API <stable>`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1088)

Unknown directive type "c:function".

```
.. c:function:: const char* PyUnicode_AsUTF8(PyObject *unicode)

   As :c:func:`PyUnicode_AsUTF8AndSize`, but does not store the size.

   .. versionadded:: 3.3

   .. versionchanged:: 3.7
      The return type is now ``const char *`` rather of ``char *``.
```

UTF-32 Codecs

These are the UTF-32 codec APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1104)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUTF32(const char *s, Py_ssize_t size, \
                            const char *errors, int *byteorder)

   Decode *size* bytes from a UTF-32 encoded buffer string and return the
   corresponding Unicode object.  *errors* (if non-``NULL``) defines the error
   handling. It defaults to "strict".

   If *byteorder* is non-``NULL``, the decoder starts decoding using the given byte
   order::

      *byteorder == -1: little endian
      *byteorder == 0:  native order
      *byteorder == 1:  big endian

   If ``*byteorder`` is zero, and the first four bytes of the input data are a
   byte order mark (BOM), the decoder switches to this byte order and the BOM is
   not copied into the resulting Unicode string.  If ``*byteorder`` is ``-1`` or
   ``1``, any byte order mark is copied to the output.

   After completion, *\*byteorder* is set to the current byte order at the end
   of input data.

   If *byteorder* is ``NULL``, the codec starts in native order mode.

   Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1131)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUTF32Stateful(const char *s, Py_ssize_t size, \
                            const char *errors, int *byteorder, Py_ssize_t *consumed)

   If *consumed* is ``NULL``, behave like :c:func:`PyUnicode_DecodeUTF32`. If
   *consumed* is not ``NULL``, :c:func:`PyUnicode_DecodeUTF32Stateful` will not treat
   trailing incomplete UTF-32 byte sequences (such as a number of bytes not divisible
   by four) as an error. Those bytes will not be decoded and the number of bytes
   that have been decoded will be stored in *consumed*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1141)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsUTF32String(PyObject *unicode)
```

```
   Return a Python byte string using the UTF-32 encoding in native byte
   order. The string always starts with a BOM mark.  Error handling is "strict".
   Return ``NULL`` if an exception was raised by the codec.
```

UTF-16 Codecs

These are the UTF-16 codec APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1154)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUTF16(const char *s, Py_ssize_t size, \
                           const char *errors, int *byteorder)

   Decode *size* bytes from a UTF-16 encoded buffer string and return the
   corresponding Unicode object.  *errors* (if non-``NULL``) defines the error
   handling. It defaults to "strict".

   If *byteorder* is non-``NULL``, the decoder starts decoding using the given byte
   order::

      *byteorder == -1: little endian
      *byteorder == 0:  native order
      *byteorder == 1:  big endian

   If ``*byteorder`` is zero, and the first two bytes of the input data are a
   byte order mark (BOM), the decoder switches to this byte order and the BOM is
   not copied into the resulting Unicode string.  If ``*byteorder`` is ``-1`` or
   ``1``, any byte order mark is copied to the output (where it will result in
   either a ``\ufeff`` or a ``\ufffe`` character).

   After completion, ``*byteorder`` is set to the current byte order at the end
   of input data.

   If *byteorder* is ``NULL``, the codec starts in native order mode.

   Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1182)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUTF16Stateful(const char *s, Py_ssize_t size, \
                           const char *errors, int *byteorder, Py_ssize_t *consumed)

   If *consumed* is ``NULL``, behave like :c:func:`PyUnicode_DecodeUTF16`. If
   *consumed* is not ``NULL``, :c:func:`PyUnicode_DecodeUTF16Stateful` will not treat
   trailing incomplete UTF-16 byte sequences (such as an odd number of bytes or a
   split surrogate pair) as an error. Those bytes will not be decoded and the
   number of bytes that have been decoded will be stored in *consumed*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1192)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsUTF16String(PyObject *unicode)

   Return a Python byte string using the UTF-16 encoding in native byte
   order. The string always starts with a BOM mark.  Error handling is "strict".
   Return ``NULL`` if an exception was raised by the codec.
```

UTF-7 Codecs

These are the UTF-7 codec APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1205)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUTF7(const char *s, Py_ssize_t size, const char *errors)

   Create a Unicode object by decoding *size* bytes of the UTF-7 encoded string
   *s*.  Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1211)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUTF7Stateful(const char *s, Py_ssize_t size, \
                              const char *errors, Py_ssize_t *consumed)

   If *consumed* is ``NULL``, behave like :c:func:`PyUnicode_DecodeUTF7`.  If
   *consumed* is not ``NULL``, trailing incomplete UTF-7 base-64 sections will not
   be treated as an error.  Those bytes will not be decoded and the number of
   bytes that have been decoded will be stored in *consumed*.
```

Unicode-Escape Codecs

These are the "Unicode Escape" codec APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1226)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeUnicodeEscape(const char *s, \
                              Py_ssize_t size, const char *errors)

   Create a Unicode object by decoding *size* bytes of the Unicode-Escape encoded
   string *s*.  Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1233)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsUnicodeEscapeString(PyObject *unicode)

   Encode a Unicode object using Unicode-Escape and return the result as a
   bytes object.  Error handling is "strict".  Return ``NULL`` if an exception was
   raised by the codec.
```

Raw-Unicode-Escape Codecs

These are the "Raw Unicode Escape" codec APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1246)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeRawUnicodeEscape(const char *s, \
                              Py_ssize_t size, const char *errors)

   Create a Unicode object by decoding *size* bytes of the Raw-Unicode-Escape
   encoded string *s*.  Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1253)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsRawUnicodeEscapeString(PyObject *unicode)

   Encode a Unicode object using Raw-Unicode-Escape and return the result as
   a bytes object.  Error handling is "strict".  Return ``NULL`` if an exception
   was raised by the codec.
```

Latin-1 Codecs

These are the Latin-1 codec APIs: Latin-1 corresponds to the first 256 Unicode ordinals and only these are accepted by the codecs during encoding.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1267)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeLatin1(const char *s, Py_ssize_t size, const char *errors)

   Create a Unicode object by decoding *size* bytes of the Latin-1 encoded string
   *s*.  Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1273)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsLatin1String(PyObject *unicode)

   Encode a Unicode object using Latin-1 and return the result as Python bytes
   object.  Error handling is "strict".  Return ``NULL`` if an exception was
   raised by the codec.
```

ASCII Codecs

These are the ASCII codec APIs. Only 7-bit ASCII data is accepted. All other codes generate errors.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1287)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeASCII(const char *s, Py_ssize_t size, const char *errors)

   Create a Unicode object by decoding *size* bytes of the ASCII encoded string
   *s*.  Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1293)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsASCIIString(PyObject *unicode)

   Encode a Unicode object using ASCII and return the result as Python bytes
   object.  Error handling is "strict".  Return ``NULL`` if an exception was
   raised by the codec.
```

Character Map Codecs

This codec is special in that it can be used to implement many different codecs (and this is in fact what was done to obtain most of the standard codecs included in the :mod:`encodings` package). The codec uses mappings to encode and decode characters. The mapping objects provided must support the :meth:`__getitem__` mapping interface; dictionaries and sequences work well.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1303); *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)unicode.rst, line 1303); *backlink*

Unknown interpreted text role "meth".

These are the mapping codec APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1311)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeCharmap(const char *data, Py_ssize_t size, \
                                PyObject *mapping, const char *errors)

   Create a Unicode object by decoding *size* bytes of the encoded string *s*
   using the given *mapping* object.  Return ``NULL`` if an exception was raised
   by the codec.

   If *mapping* is ``NULL``, Latin-1 decoding will be applied.  Else
   *mapping* must map bytes ordinals (integers in the range from 0 to 255)
   to Unicode strings, integers (which are then interpreted as Unicode
   ordinals) or ``None``.  Unmapped data bytes -- ones which cause a
   :exc:`LookupError`, as well as ones which get mapped to ``None``,
   ``0xFFFE`` or ``'\ufffe'``, are treated as undefined mappings and cause
   an error.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1327)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsCharmapString(PyObject *unicode, PyObject *mapping)

   Encode a Unicode object using the given *mapping* object and return the
   result as a bytes object.  Error handling is "strict".  Return ``NULL`` if an
   exception was raised by the codec.

   The *mapping* object must map Unicode ordinal integers to bytes objects,
   integers in the range from 0 to 255 or ``None``.  Unmapped character
   ordinals (ones which cause a :exc:`LookupError`) as well as mapped to
   ``None`` are treated as "undefined mapping" and cause an error.
```

The following codec API is special in that maps Unicode to Unicode.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1341)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Translate(PyObject *str, PyObject *table, const char *errors)

   Translate a string by applying a character mapping table to it and return the
   resulting Unicode object. Return ``NULL`` if an exception was raised by the
   codec.

   The mapping table must map Unicode ordinal integers to Unicode ordinal integers
   or ``None`` (causing deletion of the character).

   Mapping tables need only provide the :meth:`__getitem__` interface; dictionaries
   and sequences work well.  Unmapped character ordinals (ones which cause a
   :exc:`LookupError`) are left untouched and are copied as-is.

   *errors* has the usual meaning for codecs. It may be ``NULL`` which indicates to
   use the default error handling.
```

MBCS codecs for Windows

These are the MBCS codec APIs. They are currently only available on Windows and use the Win32 MBCS converters to implement the conversions. Note that MBCS (or DBCS) is a class of encodings, not just one. The target encoding is defined by the user settings on the machine running the codec.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1366)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeMBCS(const char *s, Py_ssize_t size, const char *errors)

   Create a Unicode object by decoding *size* bytes of the MBCS encoded string *s*.
   Return ``NULL`` if an exception was raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1372)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_DecodeMBCSStateful(const char *s, Py_ssize_t size, \
                                const char *errors, Py_ssize_t *consumed)

   If *consumed* is ``NULL``, behave like :c:func:`PyUnicode_DecodeMBCS`. If
   *consumed* is not ``NULL``, :c:func:`PyUnicode_DecodeMBCSStateful` will not decode
   trailing lead byte and the number of bytes that have been decoded will be stored
   in *consumed*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1381)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_AsMBCSString(PyObject *unicode)

   Encode a Unicode object using MBCS and return the result as Python bytes
   object.  Error handling is "strict".  Return ``NULL`` if an exception was
   raised by the codec.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1388)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_EncodeCodePage(int code_page, PyObject *unicode, const char *erro

   Encode the Unicode object using the specified code page and return a Python
   bytes object.  Return ``NULL`` if an exception was raised by the codec. Use
   :c:data:`CP_ACP` code page to get the MBCS encoder.

   .. versionadded:: 3.3
```

Methods & Slots

Methods and Slot Functions

The following APIs are capable of handling Unicode objects and strings on input (we refer to them as strings in the descriptions) and return Unicode objects or integers as appropriate.

They all return `NULL` or `-1` if an exception occurs.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1413)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Concat(PyObject *left, PyObject *right)

   Concat two strings giving a new Unicode string.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1418)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Split(PyObject *s, PyObject *sep, Py_ssize_t maxsplit)

   Split a string giving a list of Unicode strings.  If *sep* is ``NULL``, splitting
   will be done at all whitespace substrings.  Otherwise, splits occur at the given
   separator.  At most *maxsplit* splits will be done.  If negative, no limit is
   set.  Separators are not included in the resulting list.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1426)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Splitlines(PyObject *s, int keepend)

   Split a Unicode string at line breaks, returning a list of Unicode strings.
   CRLF is considered to be one line break.  If *keepend* is ``0``, the line break
   characters are not included in the resulting strings.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1433)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Join(PyObject *separator, PyObject *seq)

   Join a sequence of strings using the given *separator* and return the resulting
   Unicode string.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1439)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_Tailmatch(PyObject *str, PyObject *substr, \
                        Py_ssize_t start, Py_ssize_t end, int direction)

   Return ``1`` if *substr* matches ``str[start:end]`` at the given tail end
   (*direction* == ``-1`` means to do a prefix match, *direction* == ``1`` a suffix match),
   ``0`` otherwise. Return ``-1`` if an error occurred.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1447)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_Find(PyObject *str, PyObject *substr, \
                                Py_ssize_t start, Py_ssize_t end, int direction)

   Return the first position of *substr* in ``str[start:end]`` using the given
   *direction* (*direction* == ``1`` means to do a forward search, *direction* == ``-1`` a
   backward search).  The return value is the index of the first match; a value of
   ``-1`` indicates that no match was found, and ``-2`` indicates that an error
   occurred and an exception has been set.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1457)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_FindChar(PyObject *str, Py_UCS4 ch, \
                                Py_ssize_t start, Py_ssize_t end, int direction)

   Return the first position of the character *ch* in ``str[start:end]`` using
   the given *direction* (*direction* == ``1`` means to do a forward search,
   *direction* == ``-1`` a backward search).  The return value is the index of the
   first match; a value of ``-1`` indicates that no match was found, and ``-2``
   indicates that an error occurred and an exception has been set.

   .. versionadded:: 3.3

   .. versionchanged:: 3.7
      *start* and *end* are now adjusted to behave like ``str[start:end]``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1472)

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyUnicode_Count(PyObject *str, PyObject *substr, \
                                Py_ssize_t start, Py_ssize_t end)
```

```
   Return the number of non-overlapping occurrences of *substr* in
   ``str[start:end]``.  Return ``-1`` if an error occurred.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1479)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Replace(PyObject *str, PyObject *substr, \
                                PyObject *replstr, Py_ssize_t maxcount)

   Replace at most *maxcount* occurrences of *substr* in *str* with *replstr* and
   return the resulting Unicode object. *maxcount* == ``-1`` means replace all
   occurrences.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1487)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_Compare(PyObject *left, PyObject *right)

   Compare two strings and return ``-1``, ``0``, ``1`` for less than, equal, and greater than,
   respectively.

   This function returns ``-1`` upon failure, so one should call
   :c:func:`PyErr_Occurred` to check for errors.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1496)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_CompareWithASCIIString(PyObject *uni, const char *string)

   Compare a Unicode object, *uni*, with *string* and return ``-1``, ``0``, ``1`` for less
   than, equal, and greater than, respectively. It is best to pass only
   ASCII-encoded strings, but the function interprets the input string as
   ISO-8859-1 if it contains non-ASCII characters.

   This function does not raise exceptions.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1506)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_RichCompare(PyObject *left,  PyObject *right,  int op)

   Rich compare two Unicode strings and return one of the following:

   * ``NULL`` in case an exception was raised
   * :const:`Py_True` or :const:`Py_False` for successful comparisons
   * :const:`Py_NotImplemented` in case the type combination is unknown

   Possible values for *op* are :const:`Py_GT`, :const:`Py_GE`, :const:`Py_EQ`,
   :const:`Py_NE`, :const:`Py_LT`, and :const:`Py_LE`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1518)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_Format(PyObject *format, PyObject *args)

   Return a new string object from *format* and *args*; this is analogous to
   ``format % args``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1524)

Unknown directive type "c:function".

```
.. c:function:: int PyUnicode_Contains(PyObject *container, PyObject *element)

   Check whether *element* is contained in *container* and return true or false
   accordingly.

   *element* has to coerce to a one element Unicode string. ``-1`` is returned
   if there was an error.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1533)

Unknown directive type "c:function".

```
.. c:function:: void PyUnicode_InternInPlace(PyObject **string)

   Intern the argument *\*string* in place.  The argument must be the address of a
   pointer variable pointing to a Python Unicode string object.  If there is an
   existing interned string that is the same as *\*string*, it sets *\*string* to
   it (decrementing the reference count of the old string object and incrementing
   the reference count of the interned string object), otherwise it leaves
   *\*string* alone and interns it (incrementing its reference count).
   (Clarification: even though there is a lot of talk about reference counts, think
   of this function as reference-count-neutral; you own the object after the call
   if and only if you owned it before the call.)
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)unicode.rst, line 1546)

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyUnicode_InternFromString(const char *v)

   A combination of :c:func:`PyUnicode_FromString` and
   :c:func:`PyUnicode_InternInPlace`, returning either a new Unicode string
   object that has been interned, or a new ("owned") reference to an earlier
   interned string object with the same value.
```