String conversion and formatting

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

Unknown directive type "highlight".

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

Functions for number conversion and formatted string output.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]conversion.rst`, line 11)

Unknown directive type "c:function".

```
.. c:function:: int PyOS_snprintf(char *str, size_t size,  const char *format, ...)

   Output not more than *size* bytes to *str* according to the format string
   *format* and the extra arguments. See the Unix man page :manpage:`snprintf(3)`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]conversion.rst`, line 17)

Unknown directive type "c:function".

```
.. c:function:: int PyOS_vsnprintf(char *str, size_t size, const char *format, va_list va)

   Output not more than *size* bytes to *str* according to the format string
   *format* and the variable argument list *va*. Unix man page
   :manpage:`vsnprintf(3)`.
```

:c:func:`PyOS_snprintf` and :c:func:`PyOS_vsnprintf` wrap the Standard C library functions :c:func:`snprintf` and :c:func:`vsnprintf`. Their purpose is to guarantee consistent behavior in corner cases, which the Standard C functions do not.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]conversion.rst`, line 23); *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]conversion.rst`, line 23); *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]conversion.rst`, line 23); *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]conversion.rst`, line 23); *backlink*

Unknown interpreted text role "c:func".

The wrappers ensure that `str[size-1]` is always `'\0'` upon return. They never write more than *size* bytes (including the trailing `'\0'`) into str. Both functions require that `str != NULL`, `size > 0`, `format != NULL` and `size < INT_MAX`.

The return value (*rv*) for these functions should be interpreted as follows:

- When `0 <= rv < size`, the output conversion was successful and *rv* characters were written to *str* (excluding the trailing `'\0'` byte at `str[rv]`).
- When `rv >= size`, the output conversion was truncated and a buffer with `rv + 1` bytes would have been needed to succeed. `str[size-1]` is `'\0'` in this case.
- When `rv < 0`, "something bad happened." `str[size-1]` is `'\0'` in this case too, but the rest of *str* is undefined. The exact cause of the error depends on the underlying platform.

The following functions provide locale-independent string to number conversions.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]conversion.rst`, line 50)

Unknown directive type "c:function".

```
.. c:function:: double PyOS_string_to_double(const char *s, char **endptr, PyObject *overflow_exception)

   Convert a string ``s`` to a :c:type:`double`, raising a Python
   exception on failure.  The set of accepted strings corresponds to
   the set of strings accepted by Python's :func:`float` constructor,
   except that ``s`` must not have leading or trailing whitespace.
   The conversion is independent of the current locale.

   If ``endptr`` is ``NULL``, convert the whole string.  Raise
   :exc:`ValueError` and return ``-1.0`` if the string is not a valid
   representation of a floating-point number.

   If endptr is not ``NULL``, convert as much of the string as
   possible and set ``*endptr`` to point to the first unconverted
   character.  If no initial segment of the string is the valid
   representation of a floating-point number, set ``*endptr`` to point
   to the beginning of the string, raise ValueError, and return
   ``-1.0``.

   If ``s`` represents a value that is too large to store in a float
   (for example, ``"1e500"`` is such a string on many platforms) then
   if ``overflow_exception`` is ``NULL`` return ``Py_HUGE_VAL`` (with
   an appropriate sign) and don't set any exception.  Otherwise,
   ``overflow_exception`` must point to a Python exception object;
   raise that exception and return ``-1.0``.  In both cases, set
   ``*endptr`` to point to the first character after the converted value.

   If any other error occurs during the conversion (for example an
   out-of-memory error), set the appropriate Python exception and
   return ``-1.0``.

   .. versionadded:: 3.1
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\[cpython-main][Doc][c-api]conversion.rst`, line 84)

Unknown directive type "c:function".

```
.. c:function:: char* PyOS_double_to_string(double val, char format_code, int precision, int flags, int

   Convert a :c:type:`double` *val* to a string using supplied
   *format_code*, *precision*, and *flags*.

   *format_code* must be one of ``'e'``, ``'E'``, ``'f'``, ``'F'``,
   ``'g'``, ``'G'`` or ``'r'``.  For ``'r'``, the supplied *precision*
   must be 0 and is ignored.  The ``'r'`` format code specifies the
   standard :func:`repr` format.

   *flags* can be zero or more of the values ``Py_DTSF_SIGN``,
   ``Py_DTSF_ADD_DOT_0``, or ``Py_DTSF_ALT``, or-ed together:

   * ``Py_DTSF_SIGN`` means to always precede the returned string with a sign
     character, even if *val* is non-negative.

   * ``Py_DTSF_ADD_DOT_0`` means to ensure that the returned string will not look
     like an integer.

   * ``Py_DTSF_ALT`` means to apply "alternate" formatting rules.  See the
     documentation for the :c:func:`PyOS_snprintf` ``'#'`` specifier for
     details.

   If *ptype* is non-``NULL``, then the value it points to will be set to one of
   ``Py_DTST_FINITE``, ``Py_DTST_INFINITE``, or ``Py_DTST_NAN``, signifying that
   *val* is a finite number, an infinite number, or not a number, respectively.

   The return value is a pointer to *buffer* with the converted string or
   ``NULL`` if the conversion failed. The caller is responsible for freeing the
   returned string by calling :c:func:`PyMem_Free`.

   .. versionadded:: 3.1
```

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

Unknown directive type "c:function".

```
.. c:function:: int PyOS_stricmp(const char *s1, const char *s2)

   Case insensitive comparison of strings. The function works almost
   identically to :c:func:`strcmp` except that it ignores the case.
```

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

`main\Doc\c-api\[cpython-main][Doc][c-api]conversion.rst`, line 124)

Unknown directive type "c:function".

```
.. c:function:: int PyOS_strnicmp(const char *s1, const char *s2, Py_ssize_t  size)

   Case insensitive comparison of strings. The function works almost
   identically to :c:func:`strncmp` except that it ignores the case.
```