Python Initialization Configuration

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

Unknown directive type "versionadded".

```
.. versionadded:: 3.8
```

Python can be initialized with :c:func:`Py_InitializeFromConfig` and the :c:type:`PyConfig` structure. It can be preinitialized with :c:func:`Py_PreInitialize` and the :c:type:`PyPreConfig` structure.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 11**); ***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)init_config.rst`, **line 11**); ***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)init_config.rst`, **line 11**); ***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)init_config.rst`, **line 11**); ***backlink***

Unknown interpreted text role "c:type".

There are two kinds of configuration:

- The :ref:`Python Configuration <init-python-config>` can be used to build a customized Python which behaves as the regular Python. For example, environment variables and command line arguments are used to configure Python.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 17**); ***backlink***

  Unknown interpreted text role "ref".

- The :ref:`Isolated Configuration <init-isolated-conf>` can be used to embed Python into an application. It isolates Python from the system. For example, environment variables are ignored, the LC_CTYPE locale is left unchanged and no signal handler is registered.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 22**); ***backlink***

  Unknown interpreted text role "ref".

The :c:func:`Py_RunMain` function can be used to write a customized Python program.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 27**); ***backlink***

Unknown interpreted text role "c:func".

See also :ref:`Initialization, Finalization, and Threads <initialization>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 30**); ***backlink***

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 32**)

Unknown directive type "seealso".

```
.. seealso::
   :pep:`587` "Python Initialization Configuration".
```

Example

Example of customized Python always running in isolated mode:

```
int main(int argc, char **argv)
{
    PyStatus status;

    PyConfig config;
    PyConfig_InitPythonConfig(&config);
    config.isolated = 1;

    /* Decode command line arguments.
       Implicitly preinitialize Python (in isolated mode). */
    status = PyConfig_SetBytesArgv(&config, argc, argv);
    if (PyStatus_Exception(status)) {
        goto exception;
    }

    status = Py_InitializeFromConfig(&config);
    if (PyStatus_Exception(status)) {
        goto exception;
    }
    PyConfig_Clear(&config);

    return Py_RunMain();

exception:
    PyConfig_Clear(&config);
    if (PyStatus_IsExit(status)) {
        return status.exitcode;
    }
    /* Display the error message and exit the process with
       non-zero exit code */
    Py_ExitStatusException(status);
}
```

PyWideStringList

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 78)

Unknown directive type "c:type".

```
.. c:type:: PyWideStringList

   List of ``wchar_t*`` strings.

   If *length* is non-zero, *items* must be non-``NULL`` and all strings must be
   non-``NULL``.

   Methods:

   .. c:function:: PyStatus PyWideStringList_Append(PyWideStringList *list, const wchar_t *item)

      Append *item* to *list*.

      Python must be preinitialized to call this function.

   .. c:function:: PyStatus PyWideStringList_Insert(PyWideStringList *list, Py_ssize_t index, const wchar_t *item)

      Insert *item* into *list* at *index*.

      If *index* is greater than or equal to *list* length, append *item* to
      *list*.

      *index* must be greater than or equal to 0.

      Python must be preinitialized to call this function.

   Structure fields:

   .. c:member:: Py_ssize_t length

      List length.

   .. c:member:: wchar_t** items

      List items.
```

PyStatus

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 117)

Unknown directive type "c:type".

```
.. c:type:: PyStatus

   Structure to store an initialization function status: success, error
   or exit.

   For an error, it can store the C function name which created the error.

   Structure fields:

   .. c:member:: int exitcode

      Exit code. Argument passed to ``exit()``.

   .. c:member:: const char *err_msg

      Error message.
   .. c:member:: const char *func

      Name of the function which created an error, can be ``NULL``.

   Functions to create a status:

   .. c:function:: PyStatus PyStatus_Ok(void)

      Success.

   .. c:function:: PyStatus PyStatus_Error(const char *err_msg)

      Initialization error with a message.

      *err_msg* must not be ``NULL``.

   .. c:function:: PyStatus PyStatus_NoMemory(void)

      Memory allocation failure (out of memory).

   .. c:function:: PyStatus PyStatus_Exit(int exitcode)

      Exit Python with the specified exit code.

   Functions to handle a status:

   .. c:function:: int PyStatus_Exception(PyStatus status)

      Is the status an error or an exit? If true, the exception must be
      handled; by calling :c:func:`Py_ExitStatusException` for example.

   .. c:function:: int PyStatus_IsError(PyStatus status)

      Is the result an error?

   .. c:function:: int PyStatus_IsExit(PyStatus status)

      Is the result an exit?

   .. c:function:: void Py_ExitStatusException(PyStatus status)

      Call ``exit(exitcode)`` if *status* is an exit. Print the error
      message and exit with a non-zero exit code if *status* is an error.  Must
      only be called if ``PyStatus_Exception(status)`` is non-zero.
```

Note

Internally, Python uses macros which set `PyStatus.func`, whereas functions to create a status set `func` to `NULL`.

Example:

```
PyStatus alloc(void **ptr, size_t size)
{
    *ptr = PyMem_RawMalloc(size);
    if (*ptr == NULL) {
        return PyStatus_NoMemory();
    }
    return PyStatus_Ok();
}

int main(int argc, char **argv)
{
    void *ptr;
    PyStatus status = alloc(&ptr, 16);
    if (PyStatus_Exception(status)) {
        Py_ExitStatusException(status);
    }
    PyMem_Free(ptr);
    return 0;
}
```

PyPreConfig

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 209)

Unknown directive type "c:type".

```
.. c:type:: PyPreConfig

   Structure used to preinitialize Python.

   Function to initialize a preconfiguration:

   .. c:function:: void PyPreConfig_InitPythonConfig(PyPreConfig *preconfig)

      Initialize the preconfiguration with :ref:`Python Configuration
      <init-python-config>`.

   .. c:function:: void PyPreConfig_InitIsolatedConfig(PyPreConfig *preconfig)

      Initialize the preconfiguration with :ref:`Isolated Configuration
      <init-isolated-conf>`.

   Structure fields:

   .. c:member:: int allocator

      Name of the Python memory allocators:

      * ``PYMEM_ALLOCATOR_NOT_SET`` (``0``): don't change memory allocators
        (use defaults).
      * ``PYMEM_ALLOCATOR_DEFAULT`` (``1``): :ref:`default memory allocators
```

```
     <default-memory-allocators>`.
   * ``PYMEM_ALLOCATOR_DEBUG`` (``2``): :ref:`default memory allocators
     <default-memory-allocators>` with :ref:`debug hooks
     <pymem-debug-hooks>`.
   * ``PYMEM_ALLOCATOR_MALLOC`` (``3``): use ``malloc()`` of the C library.
   * ``PYMEM_ALLOCATOR_MALLOC_DEBUG`` (``4``): force usage of
     ``malloc()`` with :ref:`debug hooks <pymem-debug-hooks>`.
   * ``PYMEM_ALLOCATOR_PYMALLOC`` (``5``): :ref:`Python pymalloc memory
     allocator <pymalloc>`.
   * ``PYMEM_ALLOCATOR_PYMALLOC_DEBUG`` (``6``): :ref:`Python pymalloc
     memory allocator <pymalloc>` with :ref:`debug hooks
     <pymem-debug-hooks>`.

   ``PYMEM_ALLOCATOR_PYMALLOC`` and ``PYMEM_ALLOCATOR_PYMALLOC_DEBUG`` are
   not supported if Python is :option:`configured using --without-pymalloc
   <--without-pymalloc>`.

   See :ref:`Memory Management <memory>`.

   Default: ``PYMEM_ALLOCATOR_NOT_SET``.

.. c:member:: int configure_locale

   Set the LC_CTYPE locale to the user preferred locale?

   If equals to 0, set :c:member:`~PyPreConfig.coerce_c_locale` and
   :c:member:`~PyPreConfig.coerce_c_locale_warn` members to 0.

   See the :term:`locale encoding`.

   Default: ``1`` in Python config, ``0`` in isolated config.

.. c:member:: int coerce_c_locale

   If equals to 2, coerce the C locale.

   If equals to 1, read the LC_CTYPE locale to decide if it should be
   coerced.

   See the :term:`locale encoding`.

   Default: ``-1`` in Python config, ``0`` in isolated config.

.. c:member:: int coerce_c_locale_warn

   If non-zero, emit a warning if the C locale is coerced.

   Default: ``-1`` in Python config, ``0`` in isolated config.

.. c:member:: int dev_mode

   If non-zero, enables the :ref:`Python Development Mode <devmode>`:
   see :c:member:`PyConfig.dev_mode`.

   Default: ``-1`` in Python mode, ``0`` in isolated mode.

.. c:member:: int isolated

   Isolated mode: see :c:member:`PyConfig.isolated`.

   Default: ``0`` in Python mode, ``1`` in isolated mode.

.. c:member:: int legacy_windows_fs_encoding

   If non-zero:

   * Set :c:member:`PyPreConfig.utf8_mode` to ``0``,
   * Set :c:member:`PyConfig.filesystem_encoding` to ``"mbcs"``,
   * Set :c:member:`PyConfig.filesystem_errors` to ``"replace"``.

   Initialized the from :envvar:`PYTHONLEGACYWINDOWSFSENCODING` environment
   variable value.

   Only available on Windows. ``#ifdef MS_WINDOWS`` macro can be used for
   Windows specific code.

   Default: ``0``.

.. c:member:: int parse_argv

   If non-zero, :c:func:`Py_PreInitializeFromArgs` and
   :c:func:`Py_PreInitializeFromBytesArgs` parse their ``argv`` argument the
   same way the regular Python parses command line arguments: see
   :ref:`Command Line Arguments <using-on-cmdline>`.

   Default: ``1`` in Python config, ``0`` in isolated config.

.. c:member:: int use_environment

   Use :ref:`environment variables <using-on-envvars>`? See
   :c:member:`PyConfig.use_environment`.

   Default: ``1`` in Python config and ``0`` in isolated config.

.. c:member:: int utf8_mode

   If non-zero, enable the :ref:`Python UTF-8 Mode <utf8-mode>`.

   Set by the :option:`-X utf8 <-X>` command line option and the
   :envvar:`PYTHONUTF8` environment variable.

   Default: ``-1`` in Python config and ``0`` in isolated config.
```

Preinitialize Python with PyPreConfig

The preinitialization of Python:

- Set the Python memory allocators (:c:member:`PyPreConfig.allocator`)

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 345);** ***backlink***

  Unknown interpreted text role "c:member".

- Configure the LC_CTYPE locale (:term:`locale encoding`)

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 346);** ***backlink***

  Unknown interpreted text role "term".

- Set the :ref:`Python UTF-8 Mode <utf8-mode>` (:c:member:`PyPreConfig.utf8_mode`)

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 347);** ***backlink***

  Unknown interpreted text role "ref".

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 347);** ***backlink***

  Unknown interpreted text role "c:member".

The current preconfiguration (`PyPreConfig` type) is stored in `_PyRuntime.preconfig`.

Functions to preinitialize Python:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 355)**

Unknown directive type "c:function".

```
.. c:function:: PyStatus Py_PreInitialize(const PyPreConfig *preconfig)

   Preinitialize Python from *preconfig* preconfiguration.

   *preconfig* must not be ``NULL``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 361)**

Unknown directive type "c:function".

```
.. c:function:: PyStatus Py_PreInitializeFromBytesArgs(const PyPreConfig *preconfig, int argc, char * const *argv)

   Preinitialize Python from *preconfig* preconfiguration.

   Parse *argv* command line arguments (bytes strings) if
   :c:member:`~PyPreConfig.parse_argv` of *preconfig* is non-zero.

   *preconfig* must not be ``NULL``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, **line 370)**

Unknown directive type "c:function".

```
.. c:function:: PyStatus Py_PreInitializeFromArgs(const PyPreConfig *preconfig, int argc, wchar_t * const * argv)

   Preinitialize Python from *preconfig* preconfiguration.

   Parse *argv* command line arguments (wide strings) if
   :c:member:`~PyPreConfig.parse_argv` of *preconfig* is non-zero.

   *preconfig* must not be ``NULL``.
```

The caller is responsible to handle exceptions (error or exit) using :c:func:`PyStatus_Exception` and :c:func:`Py_ExitStatusException`.

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

Unknown interpreted text role "c:func".

For :ref:`Python Configuration <init-python-config>` (:c:func:`PyPreConfig_InitPythonConfig`), if Python is initialized with command line arguments, the command line arguments must also be passed to preinitialize Python, since they have an effect on the pre-configuration like encodings. For example, the :option:`-X utf8 <-X>` command line option enables the :ref:`Python UTF-8 Mode

<utf8-mode>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 382); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 382); *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)init_config.rst`, line 382); *backlink*

Unknown interpreted text role "option".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 382); *backlink*

Unknown interpreted text role "ref".

`PyMem_SetAllocator()` can be called after :c:func:`Py_PreInitialize` and before :c:func:`Py_InitializeFromConfig` to install a custom memory allocator. It can be called before :c:func:`Py_PreInitialize` if :c:member:`PyPreConfig.allocator` is set to `PYMEM_ALLOCATOR_NOT_SET`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 389); *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)init_config.rst`, line 389); *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)init_config.rst`, line 389); *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)init_config.rst`, line 389); *backlink*

Unknown interpreted text role "c:member".

Python memory allocation functions like :c:func:`PyMem_RawMalloc` must not be used before the Python preinitialization, whereas calling directly `malloc()` and `free()` is always safe. :c:func:`Py_DecodeLocale` must not be called before the Python preinitialization.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 394); *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)init_config.rst`, line 394); *backlink*

Unknown interpreted text role "c:func".

Example using the preinitialization to enable the :ref:`Python UTF-8 Mode <utf8-mode>`:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 399); *backlink*

Unknown interpreted text role "ref".

```
PyStatus status;
PyPreConfig preconfig;
PyPreConfig_InitPythonConfig(&preconfig);

preconfig.utf8_mode = 1;

status = Py_PreInitialize(&preconfig);
if (PyStatus_Exception(status)) {
    Py_ExitStatusException(status);
}

/* at this point, Python speaks UTF-8 */

Py_Initialize();
/* ... use Python API here ... */
Py_Finalize();
```

PyConfig

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 423)

Unknown directive type "c:type".

```
.. c:type:: PyConfig

   Structure containing most parameters to configure Python.

   When done, the :c:func:`PyConfig_Clear` function must be used to release the
   configuration memory.

   Structure methods:

   .. c:function:: void PyConfig_InitPythonConfig(PyConfig *config)

      Initialize configuration with the :ref:`Python Configuration
      <init-python-config>`.

   .. c:function:: void PyConfig_InitIsolatedConfig(PyConfig *config)

      Initialize configuration with the :ref:`Isolated Configuration
      <init-isolated-conf>`.

   .. c:function:: PyStatus PyConfig_SetString(PyConfig *config, wchar_t * const *config_str, const wchar_t *str)

      Copy the wide character string *str* into ``*config_str``.

      :ref:`Preinitialize Python <c-preinit>` if needed.

   .. c:function:: PyStatus PyConfig_SetBytesString(PyConfig *config, wchar_t * const *config_str, const char *str

      Decode *str* using :c:func:`Py_DecodeLocale` and set the result into
      ``*config_str``.

      :ref:`Preinitialize Python <c-preinit>` if needed.

   .. c:function:: PyStatus PyConfig_SetArgv(PyConfig *config, int argc, wchar_t * const *argv)

      Set command line arguments (:c:member:`~PyConfig.argv` member of
      *config*) from the *argv* list of wide character strings.

      :ref:`Preinitialize Python <c-preinit>` if needed.

   .. c:function:: PyStatus PyConfig_SetBytesArgv(PyConfig *config, int argc, char * const *argv)

      Set command line arguments (:c:member:`~PyConfig.argv` member of
      *config*) from the *argv* list of bytes strings. Decode bytes using
      :c:func:`Py_DecodeLocale`.

      :ref:`Preinitialize Python <c-preinit>` if needed.

   .. c:function:: PyStatus PyConfig_SetWideStringList(PyConfig *config, PyWideStringList *list, Py_ssize_t length,

      Set the list of wide strings *list* to *length* and *items*.

      :ref:`Preinitialize Python <c-preinit>` if needed.

   .. c:function:: PyStatus PyConfig_Read(PyConfig *config)

      Read all Python configuration.

      Fields which are already initialized are left unchanged.

      Fields for :ref:`path configuration <init-path-config>` are no longer
      calculated or modified when calling this function, as of Python 3.11.

      The :c:func:`PyConfig_Read` function only parses
      :c:member:`PyConfig.argv` arguments once: :c:member:`PyConfig.parse_argv`
      is set to ``2`` after arguments are parsed. Since Python arguments are
      strippped from :c:member:`PyConfig.argv`, parsing arguments twice would
      parse the application options as Python options.

      :ref:`Preinitialize Python <c-preinit>` if needed.

      .. versionchanged:: 3.10
         The :c:member:`PyConfig.argv` arguments are now only parsed once,
         :c:member:`PyConfig.parse_argv` is set to ``2`` after arguments are
         parsed, and arguments are only parsed if
         :c:member:`PyConfig.parse_argv` equals ``1``.

      .. versionchanged:: 3.11
         :c:func:`PyConfig_Read` no longer calculates all paths, and so fields
         listed under :ref:`Python Path Configuration <init-path-config>` may
         no longer be updated until :c:func:`Py_InitializeFromConfig` is
         called.

   .. c:function:: void PyConfig_Clear(PyConfig *config)

      Release configuration memory.

   Most ``PyConfig`` methods :ref:`preinitialize Python <c-preinit>` if needed.
   In that case, the Python preinitialization configuration
   (:c:type:`PyPreConfig`) in based on the :c:type:`PyConfig`. If configuration
   fields which are in common with :c:type:`PyPreConfig` are tuned, they must
   be set before calling a :c:type:`PyConfig` method:

   * :c:member:`PyConfig.dev_mode`
   * :c:member:`PyConfig.isolated`
   * :c:member:`PyConfig.parse_argv`
   * :c:member:`PyConfig.use_environment`

   Moreover, if :c:func:`PyConfig_SetArgv` or :c:func:`PyConfig_SetBytesArgv`
   is used, this method must be called before other methods, since the
   preinitialization configuration depends on command line arguments (if
   :c:member:`parse_argv` is non-zero).

   The caller of these methods is responsible to handle exceptions (error or
   exit) using ``PyStatus_Exception()`` and ``Py_ExitStatusException()``.
```

```
Structure fields:

.. c:member:: PyWideStringList argv

   Command line arguments: :data:`sys.argv`.

   Set :c:member:`~PyConfig.parse_argv` to ``1`` to parse
   :c:member:`~PyConfig.argv` the same way the regular Python parses Python
   command line arguments and then to strip Python arguments from
   :c:member:`~PyConfig.argv`.

   If :c:member:`~PyConfig.argv` is empty, an empty string is added to
   ensure that :data:`sys.argv` always exists and is never empty.

   Default: ``NULL``.

   See also the :c:member:`~PyConfig.orig_argv` member.

.. c:member:: wchar_t* base_exec_prefix

   :data:`sys.base_exec_prefix`.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` output.

.. c:member:: wchar_t* base_executable

   Python base executable: :data:`sys._base_executable`.

   Set by the :envvar:`__PYVENV_LAUNCHER__` environment variable.

   Set from :c:member:`PyConfig.executable` if ``NULL``.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` output.

.. c:member:: wchar_t* base_prefix

   :data:`sys.base_prefix`.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` output.

.. c:member:: int buffered_stdio

   If equals to 0 and :c:member:`~PyConfig.configure_c_stdio` is non-zero,
   disable buffering on the C streams stdout and stderr.

   Set to 0 by the :option:`-u` command line option and the
   :envvar:`PYTHONUNBUFFERED` environment variable.

   stdin is always opened in buffered mode.

   Default: ``1``.

.. c:member:: int bytes_warning

   If equals to 1, issue a warning when comparing :class:`bytes` or
   :class:`bytearray` with :class:`str`, or comparing :class:`bytes` with
   :class:`int`.

   If equal or greater to 2, raise a :exc:`BytesWarning` exception in these
   cases.

   Incremented by the :option:`-b` command line option.

   Default: ``0``.

.. c:member:: int warn_default_encoding

   If non-zero, emit a :exc:`EncodingWarning` warning when :class:`io.TextIOWrapper`
   uses its default encoding. See :ref:`io-encoding-warning` for details.

   Default: ``0``.

   .. versionadded:: 3.10

.. c:member:: int code_debug_ranges

   If equals to ``0``, disables the inclusion of the end line and column
   mappings in code objects. Also disables traceback printing carets to
   specific error locations.

   Set to ``0`` by the :envvar:`PYTHONNODEBUGRANGES` environment variable
   and by the :option:`-X no_debug_ranges <-X>` command line option.

   Default: ``1``.

   .. versionadded:: 3.11

.. c:member:: wchar_t* check_hash_pycs_mode

   Control the validation behavior of hash-based ``.pyc`` files:
   value of the :option:`--check-hash-based-pycs` command line option.

   Valid values:

   - ``L"always"``: Hash the source file for invalidation regardless of
     value of the 'check_source' flag.
   - ``L"never"``: Assume that hash-based pycs always are valid.
   - ``L"default"``: The 'check_source' flag in hash-based pycs
     determines invalidation.

   Default: ``L"default"``.
   See also :pep:`552` "Deterministic pycs".

.. c:member:: int configure_c_stdio

   If non-zero, configure C standard streams:

   * On Windows, set the binary mode (``O_BINARY``) on stdin, stdout and
     stderr.
   * If :c:member:`~PyConfig.buffered_stdio` equals zero, disable buffering
     of stdin, stdout and stderr streams.
   * If :c:member:`~PyConfig.interactive` is non-zero, enable stream
     buffering on stdin and stdout (only stdout on Windows).

   Default: ``1`` in Python config, ``0`` in isolated config.

.. c:member:: int dev_mode

   If non-zero, enable the :ref:`Python Development Mode <devmode>`.

   Default: ``-1`` in Python mode, ``0`` in isolated mode.

.. c:member:: int dump_refs

   Dump Python references?

   If non-zero, dump all objects which are still alive at exit.

   Set to ``1`` by the :envvar:`PYTHONDUMPREFS` environment variable.

   Need a special build of Python with the ``Py_TRACE_REFS`` macro defined:
   see the :option:`configure --with-trace-refs option <--with-trace-refs>`.

   Default: ``0``.

.. c:member:: wchar_t* exec_prefix

   The site-specific directory prefix where the platform-dependent Python
   files are installed: :data:`sys.exec_prefix`.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` output.

.. c:member:: wchar_t* executable

   The absolute path of the executable binary for the Python interpreter:
   :data:`sys.executable`.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` output.

.. c:member:: int faulthandler

   Enable faulthandler?

   If non-zero, call :func:`faulthandler.enable` at startup.

   Set to ``1`` by :option:`-X faulthandler <-X>` and the
   :envvar:`PYTHONFAULTHANDLER` environment variable.

   Default: ``-1`` in Python mode, ``0`` in isolated mode.

.. c:member:: wchar_t* filesystem_encoding

   :term:`Filesystem encoding <filesystem encoding and error handler>`:
   :func:`sys.getfilesystemencoding`.

   On macOS, Android and VxWorks: use ``"utf-8"`` by default.

   On Windows: use ``"utf-8"`` by default, or ``"mbcs"`` if
   :c:member:`~PyPreConfig.legacy_windows_fs_encoding` of
   :c:type:`PyPreConfig` is non-zero.

   Default encoding on other platforms:

   * ``"utf-8"`` if :c:member:`PyPreConfig.utf8_mode` is non-zero.
   * ``"ascii"`` if Python detects that ``nl_langinfo(CODESET)`` announces
     the ASCII encoding (or Roman8 encoding on HP-UX), whereas the
     ``mbstowcs()`` function decodes from a different encoding (usually
     Latin1).
   * ``"utf-8"`` if ``nl_langinfo(CODESET)`` returns an empty string.
   * Otherwise, use the :term:`locale encoding`:
     ``nl_langinfo(CODESET)`` result.

   At Python startup, the encoding name is normalized to the Python codec
   name. For example, ``"ANSI_X3.4-1968"`` is replaced with ``"ascii"``.

   See also the :c:member:`~PyConfig.filesystem_errors` member.

.. c:member:: wchar_t* filesystem_errors

   :term:`Filesystem error handler <filesystem encoding and error handler>`:
   :func:`sys.getfilesystemencodeerrors`.

   On Windows: use ``"surrogatepass"`` by default, or ``"replace"``  if
   :c:member:`~PyPreConfig.legacy_windows_fs_encoding` of
   :c:type:`PyPreConfig` is non-zero.

   On other platforms: use ``"surrogateescape"`` by default.

   Supported error handlers:

   * ``"strict"``
   * ``"surrogateescape"``
   * ``"surrogatepass"`` (only supported with the UTF-8 encoding)
```

```rst
   See also the :c:member:`~PyConfig.filesystem_encoding` member.

.. c:member:: unsigned long hash_seed
.. c:member:: int use_hash_seed

   Randomized hash function seed.

   If :c:member:`~PyConfig.use_hash_seed` is zero, a seed is chosen randomly
   at Python startup, and :c:member:`~PyConfig.hash_seed` is ignored.

   Set by the :envvar:`PYTHONHASHSEED` environment variable.

   Default *use_hash_seed* value: ``-1`` in Python mode, ``0`` in isolated
   mode.

.. c:member:: wchar_t* home

   Python home directory.

   If :c:func:`Py_SetPythonHome` has been called, use its argument if it is
   not ``NULL``.

   Set by the :envvar:`PYTHONHOME` environment variable.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` input.

.. c:member:: int import_time

   If non-zero, profile import time.

   Set the ``1`` by the :option:`-X importtime <-X>` option and the
   :envvar:`PYTHONPROFILEIMPORTTIME` environment variable.

   Default: ``0``.

.. c:member:: int inspect

   Enter interactive mode after executing a script or a command.

   If greater than 0, enable inspect: when a script is passed as first
   argument or the -c option is used, enter interactive mode after executing
   the script or the command, even when :data:`sys.stdin` does not appear to
   be a terminal.

   Incremented by the :option:`-i` command line option. Set to ``1`` if the
   :envvar:`PYTHONINSPECT` environment variable is non-empty.

   Default: ``0``.

.. c:member:: int install_signal_handlers

   Install Python signal handlers?

   Default: ``1`` in Python mode, ``0`` in isolated mode.

.. c:member:: int interactive

   If greater than 0, enable the interactive mode (REPL).

   Incremented by the :option:`-i` command line option.

   Default: ``0``.

.. c:member:: int isolated

   If greater than 0, enable isolated mode:

   * :data:`sys.path` contains neither the script's directory (computed from
     ``argv[0]`` or the current directory) nor the user's site-packages
     directory.
   * Python REPL doesn't import :mod:`readline` nor enable default readline
     configuration on interactive prompts.
   * Set :c:member:`~PyConfig.use_environment` and
     :c:member:`~PyConfig.user_site_directory` to 0.

   Default: ``0`` in Python mode, ``1`` in isolated mode.

   See also :c:member:`PyPreConfig.isolated`.

.. c:member:: int legacy_windows_stdio

   If non-zero, use :class:`io.FileIO` instead of
   :class:`io.WindowsConsoleIO` for :data:`sys.stdin`, :data:`sys.stdout`
   and :data:`sys.stderr`.

   Set to ``1`` if the :envvar:`PYTHONLEGACYWINDOWSSTDIO` environment
   variable is set to a non-empty string.

   Only available on Windows. ``#ifdef MS_WINDOWS`` macro can be used for
   Windows specific code.

   Default: ``0``.

   See also the :pep:`528` (Change Windows console encoding to UTF-8).

.. c:member:: int malloc_stats

   If non-zero, dump statistics on :ref:`Python pymalloc memory allocator
   <pymalloc>` at exit.

   Set to ``1`` by the :envvar:`PYTHONMALLOCSTATS` environment variable.

   The option is ignored if Python is :option:`configured using
   the --without-pymalloc option <--without-pymalloc>`.
   Default: ``0``.

.. c:member:: wchar_t* platlibdir

   Platform library directory name: :data:`sys.platlibdir`.

   Set by the :envvar:`PYTHONPLATLIBDIR` environment variable.

   Default: value of the ``PLATLIBDIR`` macro which is set by the
   :option:`configure --with-platlibdir option <--with-platlibdir>`
   (default: ``"lib"``, or ``"DLLs"`` on Windows).

   Part of the :ref:`Python Path Configuration <init-path-config>` input.

   .. versionadded:: 3.9

   .. versionchanged:: 3.11
      This macro is now used on Windows to locate the standard
      library extension modules, typically under ``DLLs``. However,
      for compatibility, note that this value is ignored for any
      non-standard layouts, including in-tree builds and virtual
      environments.

.. c:member:: wchar_t* pythonpath_env

   Module search paths (:data:`sys.path`) as a string separated by ``DELIM``
   (:data:`os.path.pathsep`).

   Set by the :envvar:`PYTHONPATH` environment variable.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` input.

.. c:member:: PyWideStringList module_search_paths
.. c:member:: int module_search_paths_set

   Module search paths: :data:`sys.path`.

   If :c:member:`~PyConfig.module_search_paths_set` is equal to 0,
   :c:func:`Py_InitializeFromConfig` will replace
   :c:member:`~PyConfig.module_search_paths` and sets
   :c:member:`~PyConfig.module_search_paths_set` to ``1``.

   Default: empty list (``module_search_paths``) and ``0``
   (``module_search_paths_set``).

   Part of the :ref:`Python Path Configuration <init-path-config>` output.

.. c:member:: int optimization_level

   Compilation optimization level:

   * ``0``: Peephole optimizer, set ``__debug__`` to ``True``.
   * ``1``: Level 0, remove assertions, set ``__debug__`` to ``False``.
   * ``2``: Level 1, strip docstrings.

   Incremented by the :option:`-O` command line option. Set to the
   :envvar:`PYTHONOPTIMIZE` environment variable value.

   Default: ``0``.

.. c:member:: PyWideStringList orig_argv

   The list of the original command line arguments passed to the Python
   executable: :data:`sys.orig_argv`.

   If :c:member:`~PyConfig.orig_argv` list is empty and
   :c:member:`~PyConfig.argv` is not a list only containing an empty
   string, :c:func:`PyConfig_Read` copies :c:member:`~PyConfig.argv` into
   :c:member:`~PyConfig.orig_argv` before modifying
   :c:member:`~PyConfig.argv` (if :c:member:`~PyConfig.parse_argv` is
   non-zero).

   See also the :c:member:`~PyConfig.argv` member and the
   :c:func:`Py_GetArgcArgv` function.

   Default: empty list.

   .. versionadded:: 3.10

.. c:member:: int parse_argv

   Parse command line arguments?

   If equals to ``1``, parse :c:member:`~PyConfig.argv` the same way the regular
   Python parses :ref:`command line arguments <using-on-cmdline>`, and strip
   Python arguments from :c:member:`~PyConfig.argv`.

   The :c:func:`PyConfig_Read` function only parses
   :c:member:`PyConfig.argv` arguments once: :c:member:`PyConfig.parse_argv`
   is set to ``2`` after arguments are parsed. Since Python arguments are
   strippped from :c:member:`PyConfig.argv`, parsing arguments twice would
   parse the application options as Python options.

   Default: ``1`` in Python mode, ``0`` in isolated mode.

   .. versionchanged:: 3.10
      The :c:member:`PyConfig.argv` arguments are now only parsed if
      :c:member:`PyConfig.parse_argv` equals to ``1``.

.. c:member:: int parser_debug

   Parser debug mode. If greater than 0, turn on parser debugging output (for expert only, depending
   on compilation options).
   Incremented by the :option:`-d` command line option. Set to the
   :envvar:`PYTHONDEBUG` environment variable value.

   Default: ``0``.

.. c:member:: int pathconfig_warnings

   If non-zero, calculation of path configuration is allowed to log
   warnings into ``stderr``. If equals to 0, suppress these warnings.

   Default: ``1`` in Python mode, ``0`` in isolated mode.

   Part of the :ref:`Python Path Configuration <init-path-config>` input.

   .. versionchanged:: 3.11
      Now also applies on Windows.

.. c:member:: wchar_t* prefix

   The site-specific directory prefix where the platform independent Python
   files are installed: :data:`sys.prefix`.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` output.

.. c:member:: wchar_t* program_name

   Program name used to initialize :c:member:`~PyConfig.executable` and in
   early error messages during Python initialization.

   * If :func:`Py_SetProgramName` has been called, use its argument.
   * On macOS, use :envvar:`PYTHONEXECUTABLE` environment variable if set.
   * If the ``WITH_NEXT_FRAMEWORK`` macro is defined, use
     :envvar:`__PYVENV_LAUNCHER__` environment variable if set.
   * Use ``argv[0]`` of :c:member:`~PyConfig.argv` if available and
     non-empty.
   * Otherwise, use ``L"python"`` on Windows, or ``L"python3"`` on other
     platforms.

   Default: ``NULL``.

   Part of the :ref:`Python Path Configuration <init-path-config>` input.

.. c:member:: wchar_t* pycache_prefix

   Directory where cached ``.pyc`` files are written:
   :data:`sys.pycache_prefix`.

   Set by the :option:`-X pycache_prefix=PATH <-X>` command line option and
   the :envvar:`PYTHONPYCACHEPREFIX` environment variable.

   If ``NULL``, :data:`sys.pycache_prefix` is set to ``None``.

   Default: ``NULL``.

.. c:member:: int quiet

   Quiet mode. If greater than 0, don't display the copyright and version at
   Python startup in interactive mode.

   Incremented by the :option:`-q` command line option.

   Default: ``0``.

.. c:member:: wchar_t* run_command

   Value of the :option:`-c` command line option.

   Used by :c:func:`Py_RunMain`.

   Default: ``NULL``.

.. c:member:: wchar_t* run_filename

   Filename passed on the command line: trailing command line argument
   without :option:`-c` or :option:`-m`.

   For example, it is set to ``script.py`` by the ``python3 script.py arg``
   command.

   Used by :c:func:`Py_RunMain`.

   Default: ``NULL``.

.. c:member:: wchar_t* run_module

   Value of the :option:`-m` command line option.

   Used by :c:func:`Py_RunMain`.

   Default: ``NULL``.

.. c:member:: int show_ref_count

   Show total reference count at exit?

   Set to 1 by :option:`-X showrefcount <-X>` command line option.

   Need a :ref:`debug build of Python <debug-build>` (the ``Py_REF_DEBUG``
   macro must be defined).

   Default: ``0``.

.. c:member:: int site_import

   Import the :mod:`site` module at startup?
   If equal to zero, disable the import of the module site and the
   site-dependent manipulations of :data:`sys.path` that it entails.

   Also disable these manipulations if the :mod:`site` module is explicitly
   imported later (call :func:`site.main` if you want them to be triggered).

   Set to ``0`` by the :option:`-S` command line option.

   :data:`sys.flags.no_site` is set to the inverted value of
   :c:member:`~PyConfig.site_import`.

   Default: ``1``.

.. c:member:: int skip_source_first_line

   If non-zero, skip the first line of the :c:member:`PyConfig.run_filename`
   source.

   It allows the usage of non-Unix forms of ``#!cmd``. This is intended for
   a DOS specific hack only.

   Set to ``1`` by the :option:`-x` command line option.

   Default: ``0``.

.. c:member:: wchar_t* stdio_encoding
.. c:member:: wchar_t* stdio_errors

   Encoding and encoding errors of :data:`sys.stdin`, :data:`sys.stdout` and
   :data:`sys.stderr` (but :data:`sys.stderr` always uses
   ``"backslashreplace"`` error handler).

   If :c:func:`Py_SetStandardStreamEncoding` has been called, use its
   *error* and *errors* arguments if they are not ``NULL``.

   Use the :envvar:`PYTHONIOENCODING` environment variable if it is
   non-empty.

   Default encoding:

   * ``"UTF-8"`` if :c:member:`PyPreConfig.utf8_mode` is non-zero.
   * Otherwise, use the :term:`locale encoding`.

   Default error handler:

   * On Windows: use ``"surrogateescape"``.
   * ``"surrogateescape"`` if :c:member:`PyPreConfig.utf8_mode` is non-zero,
     or if the LC_CTYPE locale is "C" or "POSIX".
   * ``"strict"`` otherwise.

.. c:member:: int tracemalloc

   Enable tracemalloc?

   If non-zero, call :func:`tracemalloc.start` at startup.

   Set by :option:`-X tracemalloc=N <-X>` command line option and by the
   :envvar:`PYTHONTRACEMALLOC` environment variable.

   Default: ``-1`` in Python mode, ``0`` in isolated mode.

.. c:member:: int use_environment

   Use :ref:`environment variables <using-on-envvars>`?

   If equals to zero, ignore the :ref:`environment variables
   <using-on-envvars>`.

   Default: ``1`` in Python config and ``0`` in isolated config.

.. c:member:: int user_site_directory

   If non-zero, add the user site directory to :data:`sys.path`.

   Set to ``0`` by the :option:`-s` and :option:`-I` command line options.

   Set to ``0`` by the :envvar:`PYTHONNOUSERSITE` environment variable.

   Default: ``1`` in Python mode, ``0`` in isolated mode.

.. c:member:: int verbose

   Verbose mode. If greater than 0, print a message each time a module is
   imported, showing the place (filename or built-in module) from which
   it is loaded.

   If greater or equal to 2, print a message for each file that is checked
   for when searching for a module. Also provides information on module
   cleanup at exit.

   Incremented by the :option:`-v` command line option.

   Set to the :envvar:`PYTHONVERBOSE` environment variable value.

   Default: ``0``.

.. c:member:: PyWideStringList warnoptions

   Options of the :mod:`warnings` module to build warnings filters, lowest
   to highest priority: :data:`sys.warnoptions`.

   The :mod:`warnings` module adds :data:`sys.warnoptions` in the reverse
   order: the last :c:member:`PyConfig.warnoptions` item becomes the first
   item of :data:`warnings.filters` which is checked first (highest
   priority).
```

```
          The :option:`-W` command line options adds its value to
          :c:member:`~PyConfig.warnoptions`, it can be used multiple times.

          The :envvar:`PYTHONWARNINGS` environment variable can also be used to add
          warning options. Multiple options can be specified, separated by commas
          (``,``).

          Default: empty list.

       .. c:member:: int write_bytecode

          If equal to 0, Python won't try to write ``.pyc`` files on the import of
          source modules.

          Set to ``0`` by the :option:`-B` command line option and the
          :envvar:`PYTHONDONTWRITEBYTECODE` environment variable.

          :data:`sys.dont_write_bytecode` is initialized to the inverted value of
          :c:member:`~PyConfig.write_bytecode`.

          Default: ``1``.

       .. c:member:: PyWideStringList xoptions

          Values of the :option:`-X` command line options: :data:`sys._xoptions`.

          Default: empty list.
```

If :c:member:`~PyConfig.parse_argv` is non-zero, :c:member:`~PyConfig.argv` arguments are parsed the same way the regular Python parses :ref:`command line arguments <using-on-cmdline>`, and Python arguments are stripped from :c:member:`~PyConfig.argv`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1198); *backlink*

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1198); *backlink*

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1198); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1198); *backlink*

Unknown interpreted text role "c:member".

The :c:member:`~PyConfig.xoptions` options are parsed to set other options: see the :option:`-X` command line option.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1203); *backlink*

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1203); *backlink*

Unknown interpreted text role "option".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1206)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.9

   The ``show_alloc_count`` field has been removed.
```

Initialization with PyConfig

Function to initialize Python:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1216)

Unknown directive type "c:function".

```
.. c:function:: PyStatus Py_InitializeFromConfig(const PyConfig *config)

   Initialize Python from *config* configuration.
```

The caller is responsible to handle exceptions (error or exit) using :c:func:`PyStatus_Exception` and :c:func:`Py_ExitStatusException`.

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

main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst, line 1220); *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)init_config.rst, line 1220); *backlink*

Unknown interpreted text role "c:func".

If :c:func:`PyImport_FrozenModules`, :c:func:`PyImport_AppendInittab` or :c:func:`PyImport_ExtendInittab` are used, they must be set or called after Python preinitialization and before the Python initialization. If Python is initialized multiple times, :c:func:`PyImport_AppendInittab` or :c:func:`PyImport_ExtendInittab` must be called before each Python initialization.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst, line 1223); *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)init_config.rst, line 1223); *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)init_config.rst, line 1223); *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)init_config.rst, line 1223); *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)init_config.rst, line 1223); *backlink*

Unknown interpreted text role "c:func".

The current configuration (`PyConfig` type) is stored in `PyInterpreterState.config`.

Example setting the program name:

```
void init_python(void)
{
    PyStatus status;

    PyConfig config;
    PyConfig_InitPythonConfig(&config);

    /* Set the program name. Implicitly preinitialize Python. */
    status = PyConfig_SetString(&config, &config.program_name,
                                L"/path/to/my_program");
    if (PyStatus_Exception(status)) {
        goto exception;
    }

    status = Py_InitializeFromConfig(&config);
    if (PyStatus_Exception(status)) {
        goto exception;
    }
    PyConfig_Clear(&config);
    return;

exception:
    PyConfig_Clear(&config);
    Py_ExitStatusException(status);
}
```

More complete example modifying the default configuration, read the configuration, and then override some parameters:

```
PyStatus init_python(const char *program_name)
{
    PyStatus status;

    PyConfig config;
    PyConfig_InitPythonConfig(&config);

    /* Set the program name before reading the configuration
       (decode byte string from the locale encoding).

       Implicitly preinitialize Python. */
    status = PyConfig_SetBytesString(&config, &config.program_name,
                                     program_name);
    if (PyStatus_Exception(status)) {
        goto done;
    }

    /* Read all configuration at once */
    status = PyConfig_Read(&config);
    if (PyStatus_Exception(status)) {
        goto done;
    }

    /* Append our custom search path to sys.path */
    status = PyWideStringList_Append(&config.module_search_paths,
                                     L"/path/to/more/modules");
    if (PyStatus_Exception(status)) {
        goto done;
    }

    /* Override executable computed by PyConfig_Read() */
    status = PyConfig_SetString(&config, &config.executable,
                                L"/path/to/my_executable");
    if (PyStatus_Exception(status)) {
        goto done;
    }

    status = Py_InitializeFromConfig(&config);

done:
    PyConfig_Clear(&config);
    return status;
}
```

Isolated Configuration

:c:func:`PyPreConfig_InitIsolatedConfig` and :c:func:`PyConfig_InitIsolatedConfig` functions create a configuration to isolate Python from the system. For example, to embed Python into an application.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1314); *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)init_config.rst`, line 1314); *backlink*

Unknown interpreted text role "c:func".

This configuration ignores global configuration variables, environment variables, command line arguments (:c:member:`PyConfig.argv` is not parsed) and user site directory. The C standard streams (ex: `stdout`) and the LC_CTYPE locale are left unchanged. Signal handlers are not installed.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1319); *backlink*

Unknown interpreted text role "c:member".

Configuration files are still used with this configuration to determine paths that are unspecified. Ensure :c:member:`PyConfig.home` is specified to avoid computing the default path configuration.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1324); *backlink*

Unknown interpreted text role "c:member".

Python Configuration

:c:func:`PyPreConfig_InitPythonConfig` and :c:func:`PyConfig_InitPythonConfig` functions create a configuration to build a customized Python which behaves as the regular Python.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1334); *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)init_config.rst`, line 1334); *backlink*

Unknown interpreted text role "c:func".

Environments variables and command line arguments are used to configure Python, whereas global configuration variables are ignored.

This function enables C locale coercion (PEP 538) and :ref:`Python UTF-8 Mode <utf8-mode>` (PEP 540) depending on the LC_CTYPE locale, :envvar:`PYTHONUTF8` and :envvar:`PYTHONCOERCECLOCALE` environment variables.

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

Unknown interpreted text role "ref".

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

Unknown interpreted text role "envvar".

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

Unknown interpreted text role "envvar".

Python Path Configuration

:c:type:`PyConfig` contains multiple fields for the path configuration:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1352); *backlink*

Unknown interpreted text role "c:type".

- Path configuration inputs:
  - :c:member:`PyConfig.home`

    System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1356); *backlink*

    Unknown interpreted text role "c:member".
  - :c:member:`PyConfig.platlibdir`

    System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1357); *backlink*

    Unknown interpreted text role "c:member".
  - :c:member:`PyConfig.pathconfig_warnings`

    System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1358); *backlink*

    Unknown interpreted text role "c:member".
  - :c:member:`PyConfig.program_name`

    System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1359); *backlink*

    Unknown interpreted text role "c:member".
  - :c:member:`PyConfig.pythonpath_env`

    System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1360); *backlink*

    Unknown interpreted text role "c:member".
  - current working directory: to get absolute paths
  - `PATH` environment variable to get the program full path (from :c:member:`PyConfig.program_name`)

    System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1362); *backlink*

    Unknown interpreted text role "c:member".
  - `__PYVENV_LAUNCHER__` environment variable
  - (Windows only) Application paths in the registry under "SoftwarePythonPythonCoreX.YPythonPath" of HKEY_CURRENT_USER and HKEY_LOCAL_MACHINE (where X.Y is the Python version).
- Path configuration output fields:
  - :c:member:`PyConfig.base_exec_prefix`

    System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1371); *backlink*

    Unknown interpreted text role "c:member".
  - :c:member:`PyConfig.base_executable`

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

    Unknown interpreted text role "c:member".
  - :c:member:`PyConfig.base_prefix`

    System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, line 1373); *backlink*

    Unknown interpreted text role "c:member".
  - :c:member:`PyConfig.exec_prefix`

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1374);** ***backlink***

Unknown interpreted text role "c:member".

- :c:member:`PyConfig.executable`

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1375);** ***backlink***

Unknown interpreted text role "c:member".

- :c:member:`PyConfig.module_search_paths_set`, :c:member:`PyConfig.module_search_paths`

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1376);** ***backlink***

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1376);** ***backlink***

Unknown interpreted text role "c:member".

- :c:member:`PyConfig.prefix`

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1378);** ***backlink***

Unknown interpreted text role "c:member".

If at least one "output field" is not set, Python calculates the path configuration to fill unset fields. If :c:member:`~PyConfig.module_search_paths_set` is equal to 0, :c:member:`~PyConfig.module_search_paths` is overridden and :c:member:`~PyConfig.module_search_paths_set` is set to 1.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1380);** ***backlink***

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1380);** ***backlink***

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1380);** ***backlink***

Unknown interpreted text role "c:member".

It is possible to completely ignore the function calculating the default path configuration by setting explicitly all path configuration output fields listed above. A string is considered as set even if it is non-empty. `module_search_paths` is considered as set if `module_search_paths_set` is set to 1. In this case, path configuration input fields are ignored as well.

Set :c:member:`~PyConfig.pathconfig_warnings` to 0 to suppress warnings when calculating the path configuration (Unix only, Windows does not log any warning).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1393);** ***backlink***

Unknown interpreted text role "c:member".

If :c:member:`~PyConfig.base_prefix` or :c:member:`~PyConfig.base_exec_prefix` fields are not set, they inherit their value from :c:member:`~PyConfig.prefix` and :c:member:`~PyConfig.exec_prefix` respectively.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1396);** ***backlink***

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1396);** ***backlink***

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst`, **line 1396);** ***backlink***

Unknown interpreted text role "c:member".

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

main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1396); *backlink*

Unknown interpreted text role "c:member".

:c:func:`Py_RunMain` and :c:func:`Py_Main` modify :data:`sys.path`:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1400); *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)init_config.rst, line 1400); *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)init_config.rst, line 1400); *backlink*

Unknown interpreted text role "data".

- If :c:member:`~PyConfig.run_filename` is set and is a directory which contains a `__main__.py` script, prepend :c:member:`~PyConfig.run_filename` to :data:`sys.path`.

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1402); *backlink*

  Unknown interpreted text role "c:member".

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1402); *backlink*

  Unknown interpreted text role "c:member".

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1402); *backlink*

  Unknown interpreted text role "data".

- If :c:member:`~PyConfig.isolated` is zero:

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1405); *backlink*

  Unknown interpreted text role "c:member".

  - If :c:member:`~PyConfig.run_module` is set, prepend the current directory to :data:`sys.path`. Do nothing if the current directory cannot be read.

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1407); *backlink*

    Unknown interpreted text role "c:member".

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1407); *backlink*

    Unknown interpreted text role "data".

  - If :c:member:`~PyConfig.run_filename` is set, prepend the directory of the filename to :data:`sys.path`.

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1409); *backlink*

    Unknown interpreted text role "c:member".

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1409); *backlink*

    Unknown interpreted text role "data".

  - Otherwise, prepend an empty string to :data:`sys.path`.

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main)(Doc)(c-api)init_config.rst, line 1411); *backlink*

    Unknown interpreted text role "data".

If :c:member:`~PyConfig.site_import` is non-zero, :data:`sys.path` can be modified by the :mod:`site` module. If :c:member:`~PyConfig.user_site_directory` is non-zero and the user's site-package directory exists, the :mod:`site` module appends the user's site-package directory to :data:`sys.path`.

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

Unknown interpreted text role "c:member".

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

Unknown interpreted text role "data".

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

Unknown interpreted text role "c:member".

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

Unknown interpreted text role "data".

The following configuration files are used by the path configuration:

- `pyvenv.cfg`
- `python._pth` (Windows only)
- `pybuilddir.txt` (Unix only)

The `__PYVENV_LAUNCHER__` environment variable is used to set :c:member:`PyConfig.base_executable`

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1425); *backlink*

Unknown interpreted text role "c:member".

Py_RunMain()

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1432)

Unknown directive type "c:function".

```
.. c:function:: int Py_RunMain(void)

   Execute the command (:c:member:`PyConfig.run_command`), the script
   (:c:member:`PyConfig.run_filename`) or the module
   (:c:member:`PyConfig.run_module`) specified on the command line or in the
   configuration.

   By default and when if :option:`-i` option is used, run the REPL.

   Finally, finalizes Python and returns an exit status that can be passed to
   the ``exit()`` function.
```

See :ref:`Python Configuration <init-python-config>` for an example of customized Python always running in isolated mode using :c:func:`Py_RunMain`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1444); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1444); *backlink*

Unknown interpreted text role "c:func".

Py_GetArgcArgv()

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1452)

Unknown directive type "c:function".

```
.. c:function:: void Py_GetArgcArgv(int *argc, wchar_t ***argv)
```

```
Get the original command line arguments, before Python modified them.

See also :c:member:`PyConfig.orig_argv` member.
```

Multi-Phase Initialization Private Provisional API

This section is a private provisional API introducing multi-phase initialization, the core feature of PEP 432:

- "Core" initialization phase, "bare minimum Python":
  - Builtin types;
  - Builtin exceptions;
  - Builtin and frozen modules;
  - The :mod:`sys` module is only partially initialized (ex: :data:`sys.path` doesn't exist yet).

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst, line 1470); *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) init_config.rst, line 1470); *backlink*

    Unknown interpreted text role "data".

- "Main" initialization phase, Python is fully initialized:
  - Install and configure :mod:`importlib`;

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst, line 1475); *backlink*

    Unknown interpreted text role "mod".

  - Apply the :ref:`Path Configuration <init-path-config>`;

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst, line 1476); *backlink*

    Unknown interpreted text role "ref".

  - Install signal handlers;
  - Finish :mod:`sys` module initialization (ex: create :data:`sys.stdout` and :data:`sys.path`);

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst, line 1478); *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) init_config.rst, line 1478); *backlink*

    Unknown interpreted text role "data".

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst, line 1478); *backlink*

    Unknown interpreted text role "data".

  - Enable optional features like :mod:`faulthandler` and :mod:`tracemalloc`;

    System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api) init_config.rst, line 1480); *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) init_config.rst, line 1480); *backlink*

    Unknown interpreted text role "mod".

  - Import the :mod:`site` module;

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

line 1481); *backlink*

Unknown interpreted text role "mod".

- etc.

Private provisional API:

- :c:member:`PyConfig._init_main`: if set to 0, :c:func:`Py_InitializeFromConfig` stops at the "Core" initialization phase.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1486); *backlink*

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1486); *backlink*

Unknown interpreted text role "c:func".

- :c:member:`PyConfig._isolated_interpreter`: if non-zero, disallow threads, subprocesses and fork.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1488); *backlink*

Unknown interpreted text role "c:member".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1491)

Unknown directive type "c:function".

```
.. c:function:: PyStatus _Py_InitializeMain(void)

   Move to the "Main" initialization phase, finish the Python initialization.
```

No module is imported during the "Core" phase and the `importlib` module is not configured: the :ref:`Path Configuration <init-path-config>` is only applied during the "Main" phase. It may allow to customize Python in Python to override or tune the :ref:`Path Configuration <init-path-config>`, maybe install a custom :data:`sys.meta_path` importer or an import hook, etc.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1495); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1495); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1495); *backlink*

Unknown interpreted text role "data".

It may become possible to calculatin the :ref:`Path Configuration <init-path-config>` in Python, after the Core phase and before the Main phase, which is one of the PEP 432 motivation.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\c-api\(cpython-main) (Doc) (c-api)init_config.rst`, line 1501); *backlink*

Unknown interpreted text role "ref".

The "Core" phase is not properly defined: what should be and what should not be available at this phase is not specified yet. The API is marked as private and provisional: the API can be modified or even be removed anytime until a proper public API is designed.

Example running Python code between "Core" and "Main" initialization phases:

```
void init_python(void)
{
    PyStatus status;

    PyConfig config;
    PyConfig_InitPythonConfig(&config);
    config._init_main = 0;

    /* ... customize 'config' configuration ... */

    status = Py_InitializeFromConfig(&config);
    PyConfig_Clear(&config);
    if (PyStatus_Exception(status)) {
        Py_ExitStatusException(status);
    }

    /* Use sys.stderr because sys.stdout is only created
       by _Py_InitializeMain() */
    int res = PyRun_SimpleString(
        "import sys; "
        "print('Run Python code before _Py_InitializeMain', "
               "file=sys.stderr)");
    if (res < 0) {
        exit(1);
    }

    /* ... put more configuration code here ... */

    status = _Py_InitializeMain();
    if (PyStatus_Exception(status)) {
        Py_ExitStatusException(status);
    }
}
```