

[Skip to content](#)

Python Errors

Precompiled Libraries

While not common practice, Python add-ons can be distributed with their own precompiled libraries. Unlike regular Python scripts, these are not portable between different platforms.

It is possible the library is incompatible with your Blender installation (attempting to load a library built for a different version of Python, or loading a 32-bit library on a 64-bit system).

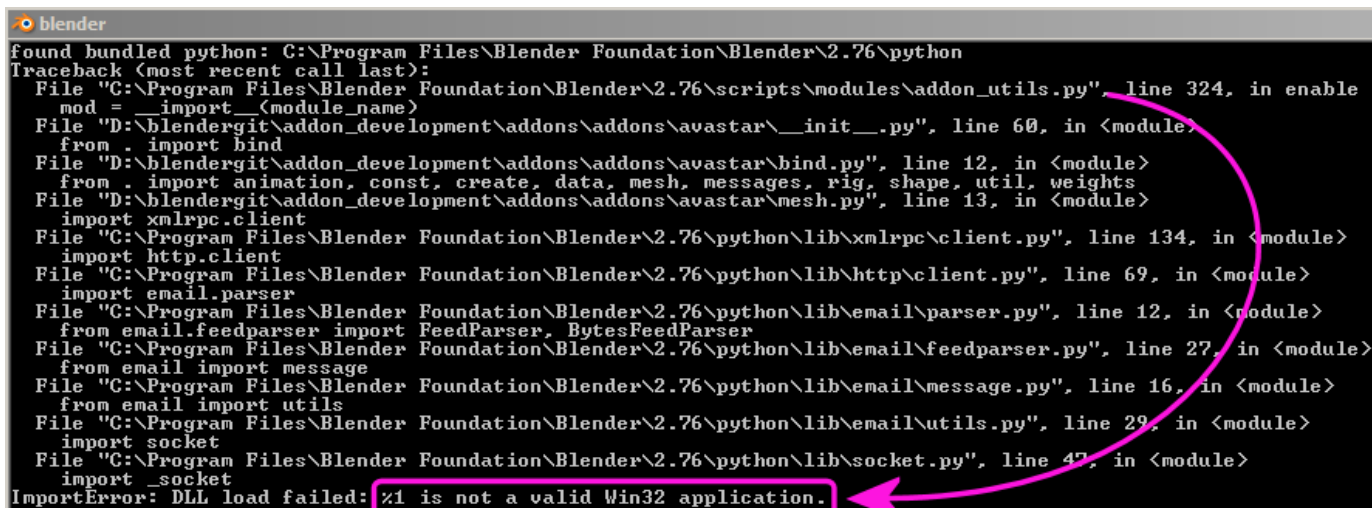
If the add-on contains `.pyd` or `.so` files, check that the distribution is compatible with your operating system.

Platform Specific

Windows

Mixed Python Libraries (DLLs)

If Python is raising errors or you have an add-on that just fails when enabled with an error – e.g. `... is not a valid Win32 application` – this may be caused by some inconsistency in the Python libraries. While Blender comes with its own bundled Python interpreter, duplicate, incompatible libraries can cause problems.



```
blender
found bundled python: C:\Program Files\Blender Foundation\Blender\2.76\python
Traceback (most recent call last):
  File "C:\Program Files\Blender Foundation\Blender\2.76\scripts\modules\addon_utils.py", line 324, in enable
    mod = __import__(module_name)
  File "D:\blendergit\addon_development\addons\addons\avastar\__init__.py", line 60, in <module>
    from . import bind
  File "D:\blendergit\addon_development\addons\addons\avastar\bind.py", line 12, in <module>
    from . import animation, const, create, data, mesh, messages, rig, shape, util, weights
  File "D:\blendergit\addon_development\addons\addons\avastar\mesh.py", line 13, in <module>
    import xmlrpc.client
  File "C:\Program Files\Blender Foundation\Blender\2.76\python\lib\xmlrpc\client.py", line 134, in <module>
    import http.client
  File "C:\Program Files\Blender Foundation\Blender\2.76\python\lib\http\client.py", line 69, in <module>
    import email.parser
  File "C:\Program Files\Blender Foundation\Blender\2.76\python\lib\email\parser.py", line 12, in <module>
    from email.feedparser import FeedParser, BytesFeedParser
  File "C:\Program Files\Blender Foundation\Blender\2.76\python\lib\email\feedparser.py", line 27, in <module>
    from email import message
  File "C:\Program Files\Blender Foundation\Blender\2.76\python\lib\email\message.py", line 16, in <module>
    from email import utils
  File "C:\Program Files\Blender Foundation\Blender\2.76\python\lib\email\utils.py", line 29, in <module>
    import socket
  File "C:\Program Files\Blender Foundation\Blender\2.76\python\lib\socket.py", line 47, in <module>
    import _socket
ImportError: DLL load failed: %1 is not a valid Win32 application.
```

A Python traceback.

To find out which Python Library caused the Problem check the error message.

This is normally reported somewhere around the bottom line of the traceback. With the error above you see the problem is caused while trying to import `_socket`. This corresponds to either a file named `_socket.py` or `_socket.pyd`.

To help troubleshoot this problem, the following script can be pasted into the Text editor and run to check for duplicate libraries in your search path. (The output will show in [Command Line Window](#).)

```
import os
import sys

# Change this based on the library you wish to test
test_lib = "_socket.pyd"

def GetSystemDirectory():
    from ctypes import windll, create_string_buffer, sizeof
    GetSystemDirectory = windll.kernel32.GetSystemDirectoryA
    buffer = create_string_buffer(260)
```

```

GetSystemDirectory(buffer, sizeof(buffer))
return os.fsdecode(buffer.value)

def library_search_paths():
    return (
        # Windows search paths
        os.path.dirname(sys.argv[0]),
        os.getcwd(),
        GetSystemDirectory(),
        os.environ["WINDIR"], # GetWindowsDirectory
        *os.environ["PATH"].split(";"),

        # regular Python search paths
        *sys.path,
    )

def check_library_duplicate(libname):
    paths = [p for p in library_search_paths()
              if os.path.exists(os.path.join(p, libname))]

    print("Library %r found in %d locations:" % (libname, len(paths)))
    for p in paths:
        print("- %r" % p)

check_library_duplicate(test_lib)

```

[Previous
Crashes](#)

Copyright ©: This page is licensed under a [CC-BY-SA 4.0 Int. License](#)

Made with [Furo](#)

Last updated on 2025-05-10

[View Source](#)
[View Translation](#)
[Report issue on this page](#)

[No
Recovering D](#)