``DEFAULT``

This feature corresponds to standard linking, essentially equivalent to

using no feature at all. It is typically only used with the

:prop\_tgt:`LINK\_LIBRARY\_OVERRIDE` and

:prop\_tgt:`LINK\_LIBRARY\_OVERRIDE\_<LIBRARY>` target properties.

``WHOLE\_ARCHIVE``

Force inclusion of all members of a static library. This feature is only

supported for the following platforms, with limitations as noted:

\* Linux.

\* All BSD variants.

\* SunOS.

\* All Apple variants. The library must be specified as a CMake target name,

a library file name (such as ``libfoo.a``), or a library file path (such as

``/path/to/libfoo.a``). Due to a limitation of the Apple linker, it

cannot be specified as a plain library name like ``foo``, where ``foo``

is not a CMake target.

\* Windows. When using a MSVC or MSVC-like toolchain, the MSVC version must

be greater than 1900.

\* Cygwin.

\* MSYS.

``FRAMEWORK``

This option tells the linker to search for the specified framework using

the ``-framework`` linker option. It can only be used on Apple platforms,

and only with a linker that understands the option used (i.e. the linker

provided with Xcode, or one compatible with it).

The framework can be specified as a CMake framework target, a bare framework

name, or a file path. If a target is given, that target must have the

:prop\_tgt:`FRAMEWORK` target property set to true. For a file path, if it

contains a directory part, that directory will be added as a framework

search path.

.. code-block:: cmake

add\_library(lib SHARED ...)

target\_link\_libraries(lib PRIVATE "$<LINK\_LIBRARY:FRAMEWORK,/path/to/my\_framework>")

# The constructed linker command line will contain:

# -F/path/to -framework my\_framework

File paths must conform to one of the following patterns (``\*`` is a

wildcard, and optional parts are shown as ``[...]``):

\* ``[/path/to/]FwName[.framework]``

\* ``[/path/to/]FwName.framework/FwName[suffix]``

\* ``[/path/to/]FwName.framework/Versions/\*/FwName[suffix]``

Note that CMake recognizes and automatically handles framework targets,

even without using the :genex:`$<LINK\_LIBRARY:FRAMEWORK,...>` expression.

The generator expression can still be used with a CMake target if the

project wants to be explicit about it, but it is not required to do so.

The linker command line may have some differences between using the

generator expression or not, but the final result should be the same.

On the other hand, if a file path is given, CMake will recognize some paths

automatically, but not all cases. The project may want to use

:genex:`$<LINK\_LIBRARY:FRAMEWORK,...>` for file paths so that the expected

behavior is clear.

.. versionadded:: 3.25

The :prop\_tgt:`FRAMEWORK\_MULTI\_CONFIG\_POSTFIX\_<CONFIG>` target property as

well as the ``suffix`` of the framework library name are now supported by

the ``FRAMEWORK`` features.

``NEEDED\_FRAMEWORK``

This is similar to the ``FRAMEWORK`` feature, except it forces the linker

to link with the framework even if no symbols are used from it. It uses

the ``-needed\_framework`` option and has the same linker constraints as

``FRAMEWORK``.

``REEXPORT\_FRAMEWORK``

This is similar to the ``FRAMEWORK`` feature, except it tells the linker

that the framework should be available to clients linking to the library

being created. It uses the ``-reexport\_framework`` option and has the

same linker constraints as ``FRAMEWORK``.

``WEAK\_FRAMEWORK``

This is similar to the ``FRAMEWORK`` feature, except it forces the linker

to mark the framework and all references to it as weak imports. It uses

the ``-weak\_framework`` option and has the same linker constraints as

``FRAMEWORK``.

``NEEDED\_LIBRARY``

This is similar to the ``NEEDED\_FRAMEWORK`` feature, except it is for use

with non-framework targets or libraries (Apple platforms only).

It uses the ``-needed\_library`` or ``-needed-l`` option as appropriate,

and has the same linker constraints as ``NEEDED\_FRAMEWORK``.

``REEXPORT\_LIBRARY``

This is similar to the ``REEXPORT\_FRAMEWORK`` feature, except it is for use

with non-framework targets or libraries (Apple platforms only).

It uses the ``-reexport\_library`` or ``-reexport-l`` option as appropriate,

and has the same linker constraints as ``REEXPORT\_FRAMEWORK``.

``WEAK\_LIBRARY``

This is similar to the ``WEAK\_FRAMEWORK`` feature, except it is for use

with non-framework targets or libraries (Apple platforms only).

It uses the ``-weak\_library`` or ``-weak-l`` option as appropriate,

and has the same linker constraints as ``WEAK\_FRAMEWORK``.