Frequently asked question

How do I suppress specific events only?

Passing the `suppress=True` flag to listeners will suppress all events system-wide. If this is not what you want, you will have to employ different solutions for different backends.

If your backend of choice is not listed below, it does not support suppression of specific events.

macOS

For *macOS*, pass the named argument `darwin_intercept` to the listener constructor. This argument should be a callable taking the arguments `(event_type, event)`, where `event_type` is any mouse related event type constant, and `event` is a CGEventRef. The `event` argument can be manipulated by the functions found on the Apple documentation.

If the interceptor function determines that the event should be suppressed, return `None`, otherwise return the `event`, which you may modify.

Here is a keyboard example:

```
    def darwin_intercept(event_type, event):
        import Quartz
        length, chars = Quartz.CGEventKeyboardGetUnicode
            event, 100, None, None)
        if length > 0 and chars == 'x':
            # Suppress x
            return None
        elif length > 0 and chars == 'a':
            # Transform a to b
            Quartz.CGEventKeyboardSetUnicodeString(event
        else:
            return event
```

Windows

For *Windows*, pass the argument named `win32_event_filter` to the listener constructor. This argument should be a callable taking the arguments `(msg, data)`, where `msg` is the current message, and `data`associated data as a MSLLHOOKSTRUCT or a *KBDLLHOOKSTRUCT*, depending on whether you are creating a mouse or keyboard listener.

If the filter function determines that the event should be suppressed, call `suppress_event` on the listener. If you return `False`, the event will be hidden from other listener callbacks.

Here is a keyboard example:

```
    # Values for MSLLHOOKSTRUCT.vkCode can be found here
    # https://docs.microsoft.com/en-us/windows/win32/inp
```

```
    def win32_event_filter(msg, data):
        if data.vkCode == 0x58:
            # Suppress x
            listener.suppress_event()
```

When using a packager I get an `ImportError` on startup

This happens when using a packager, such as *PyInstaller*, to package your application.

The reason for the error is that the packager attempts to build a dependency tree of the modules used by inspecting `import` statements, and *pynput* finds the platform dependent backend modules at runtime using `importlib`.

To solve this problem, please consult the documentation of your tool to find how to explicitly add modules.

Which modules to add depends on your distribution platform. The backend modules are those starting with an underscore (`'_'`) in the `pynput.keyboard` and `pynput.mouse` packages. Additionally, you will need modules with corresponding names from the `pynput._util` package.