:c:type:`uv_fs_event_t` --- FS Event handle

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master)(deps)(uv)(docs)(src)fs_event.rst`, line 4); ***backlink***

Unknown interpreted text role "c:type".

FS Event handles allow the user to monitor a given path for changes, for example, if the file was renamed or there was a generic change in it. This handle uses the best backend for the job on each platform.

Note

For AIX, the non default IBM bos.ahafs package has to be installed. The AIX Event Infrastructure file system (ahafs) has some limitations:

- ahafs tracks monitoring per process and is not thread safe. A separate process must be spawned for each monitor for the same event.
- Events for file modification (writing to a file) are not received if only the containing folder is watched.

See documentation for more details.

The z/OS file system events monitoring infrastructure does not notify of file creation/deletion within a directory that is being monitored. See the IBM Knowledge centre for more details.

Data types

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master)(deps)(uv)(docs)(src)fs_event.rst`, line 35)

Unknown directive type "c:type".

```
.. c:type:: uv_fs_event_t

    FS Event handle type.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master)(deps)(uv)(docs)(src)fs_event.rst`, line 39)

Unknown directive type "c:type".

```
.. c:type:: void (*uv_fs_event_cb)(uv_fs_event_t* handle, const char* filename, int events, int status)

    Callback passed to :c:func:`uv_fs_event_start` which will be called repeatedly
    after the handle is started. If the handle was started with a directory the
    `filename` parameter will be a relative path to a file contained in the directory.
    The `events` parameter is an ORed mask of :c:type:`uv_fs_event` elements.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master)(deps)(uv)(docs)(src)fs_event.rst`, line 46)

Unknown directive type "c:type".

```
.. c:type:: uv_fs_event

    Event types that :c:type:`uv_fs_event_t` handles monitor.

    ::

        enum uv_fs_event {
            UV_RENAME = 1,
            UV_CHANGE = 2
        };
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master)(deps)(uv)(docs)(src)fs_event.rst`, line 57)

Unknown directive type "c:type".

```
.. c:type:: uv_fs_event_flags

    Flags that can be passed to :c:func:`uv_fs_event_start` to control its
    behavior.

    ::

        enum uv_fs_event_flags {
            /*
            * By default, if the fs event watcher is given a directory name, we will
            * watch for all events in that directory. This flags overrides this behavior
            * and makes fs_event report only changes to the directory entry itself. This
            * flag does not affect individual files watched.
            * This flag is currently not implemented yet on any backend.
            */
            UV_FS_EVENT_WATCH_ENTRY = 1,
            /*
            * By default uv_fs_event will try to use a kernel interface such as inotify
            * or kqueue to detect events. This may not work on remote file systems such
            * as NFS mounts. This flag makes fs_event fall back to calling stat() on a
            * regular interval.
            * This flag is currently not implemented yet on any backend.
            */
            UV_FS_EVENT_STAT = 2,
            /*
            * By default, event watcher, when watching directory, is not registering
            * (is ignoring) changes in its subdirectories.
            * This flag will override this behaviour on platforms that support it.
            */
            UV_FS_EVENT_RECURSIVE = 4
        };
```

Public members

N/A

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master) (deps) (uv) (docs) (src) fs_event.rst, line 95)

Unknown directive type "seealso".

```
.. seealso:: The :c:type:`uv_handle_t` members also apply.
```

API

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master) (deps) (uv) (docs) (src) fs_event.rst, line 101)

Unknown directive type "c:function".

```
.. c:function:: int uv_fs_event_init(uv_loop_t* loop, uv_fs_event_t* handle)

    Initialize the handle.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master) (deps) (uv) (docs) (src) fs_event.rst, line 105)

Unknown directive type "c:function".

```
.. c:function:: int uv_fs_event_start(uv_fs_event_t* handle, uv_fs_event_cb cb, const char* path, unsigned

    Start the handle with the given callback, which will watch the specified
    `path` for changes. `flags` can be an ORed mask of :c:type:`uv_fs_event_flags`.

    .. note:: Currently the only supported flag is ``UV_FS_EVENT_RECURSIVE`` and
              only on OSX and Windows.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master) (deps) (uv) (docs) (src) fs_event.rst, line 113)

Unknown directive type "c:function".

```
.. c:function:: int uv_fs_event_stop(uv_fs_event_t* handle)

    Stop the handle, the callback will no longer be called.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master) (deps) (uv) (docs) (src) fs_event.rst, line 117)

Unknown directive type "c:function".

```
.. c:function:: int uv_fs_event_getpath(uv_fs_event_t* handle, char* buffer, size_t* size)

    Get the path being monitored by the handle. The buffer must be preallocated
    by the user. Returns 0 on success or an error code < 0 in case of failure.
    On success, `buffer` will contain the path and `size` its length. If the buffer
    is not big enough `UV_ENOBUFS` will be returned and `size` will be set to
    the required size, including the null terminator.

    .. versionchanged:: 1.3.0 the returned length no longer includes the terminating null byte,
                        and the buffer is not null terminated.

    .. versionchanged:: 1.9.0 the returned length includes the terminating null
                        byte on `UV_ENOBUFS`, and the buffer is null terminated
                        on success.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\node-master\deps\uv\docs\src\(node-master)(deps)(uv)(docs)(src)fs_event.rst`, line 132)

Unknown directive type "seealso".

```
.. seealso:: The :c:type:`uv_handle_t` API functions also apply.
```