shell

> *Manage files and URLs using their default applications.*

Process: Main, Renderer (non-sandboxed only)

The `shell` module provides functions related to desktop integration.

An example of opening a URL in the user's default browser:

```
const { shell } = require('electron')

shell.openExternal('https://github.com')
```

Note: While the `shell` module can be used in the renderer process, it will not function in a sandboxed renderer.

Methods

The `shell` module has the following methods:

`shell.showItemInFolder(fullPath)`

- `fullPath` string

Show the given file in a file manager. If possible, select the file.

`shell.openPath(path)`

- `path` string

Returns `Promise<string>` - Resolves with a string containing the error message corresponding to the failure if a failure occurred, otherwise "".

Open the given file in the desktop's default manner.

`shell.openExternal(url[, options])`

- `url` string - Max 2081 characters on windows.
- `options` Object (optional)
  - `activate` boolean (optional) *macOS* - `true` to bring the opened application to the foreground. The default is `true`.
  - `workingDirectory` string (optional) *Windows* - The working directory.

Returns `Promise<void>`

Open the given external protocol URL in the desktop's default manner. (For example, mailto: URLs in the user's default mail agent).

`shell.trashItem(path)`

- `path` string - path to the item to be moved to the trash.

Returns `Promise<void>` - Resolves when the operation has been completed. Rejects if there was an error while deleting the requested item.

This moves a path to the OS-specific trash location (Trash on macOS, Recycle Bin on Windows, and a desktop-environment-specific location on Linux).

`shell.beep()`

Play the beep sound.

`shell.writeShortcutLink(shortcutPath[, operation], options)` *Windows*

- `shortcutPath` string
- `operation` string (optional) - Default is `create`, can be one of following:
  - `create` - Creates a new shortcut, overwriting if necessary.
  - `update` - Updates specified properties only on an existing shortcut.
  - `replace` - Overwrites an existing shortcut, fails if the shortcut doesn't exist.
- `options` ShortcutDetails

Returns `boolean` - Whether the shortcut was created successfully.

Creates or updates a shortcut link at `shortcutPath`.

`shell.readShortcutLink(shortcutPath)` *Windows*

- `shortcutPath` string

Returns `ShortcutDetails`

Resolves the shortcut link at `shortcutPath`.

An exception will be thrown when any error happens.