Environment Variables

> Control application configuration and behavior without changing code.

Certain Electron behaviors are controlled by environment variables because they are initialized earlier than the command line flags and the app's code.

POSIX shell example:

```
$ export ELECTRON_ENABLE_LOGGING=true
$ electron
```

Windows console example:

```
> set ELECTRON_ENABLE_LOGGING=true
> electron
```

Production Variables

The following environment variables are intended primarily for use at runtime in packaged Electron applications.

`NODE_OPTIONS`

Electron includes support for a subset of Node's `NODE_OPTIONS`. The majority are supported with the exception of those which conflict with Chromium's use of BoringSSL.

Example:

```
export NODE_OPTIONS="--no-warnings --max-old-space-size=2048"
```

Unsupported options are:

```
--use-bundled-ca
--force-fips
--enable-fips
--openssl-config
--use-openssl-ca
```

`NODE_OPTIONS` are explicitly disallowed in packaged apps, except for the following:

```
--max-http-header-size
--http-parser
```

`GOOGLE_API_KEY`

Geolocation support in Electron requires the use of Google Cloud Platform's geolocation webservice. To enable this feature, acquire a Google API key and place the following code in your main process file, before opening any browser windows that will make geolocation requests:

```
process.env.GOOGLE_API_KEY = 'YOUR_KEY_HERE'
```

By default, a newly generated Google API key may not be allowed to make geolocation requests. To enable the geolocation webservice for your project, enable it through the API library.

N.B. You will need to add a Billing Account to the project associated to the API key for the geolocation webservice to work.

`ELECTRON_NO_ASAR`

Disables ASAR support. This variable is only supported in forked child processes and spawned child processes that set `ELECTRON_RUN_AS_NODE`.

`ELECTRON_RUN_AS_NODE`

Starts the process as a normal Node.js process.

In this mode, you will be able to pass cli options to Node.js as you would when running the normal Node.js executable, with the exception of the following flags:

- "–openssl-config"
- "–use-bundled-ca"
- "–use-openssl-ca",
- "–force-fips"
- "–enable-fips"

These flags are disabled owing to the fact that Electron uses BoringSSL instead of OpenSSL when building Node.js' `crypto` module, and so will not work as designed.

`ELECTRON_NO_ATTACH_CONSOLE` *Windows*

Don't attach to the current console session.

`ELECTRON_FORCE_WINDOW_MENU_BAR` *Linux*

Don't use the global menu bar on Linux.

`ELECTRON_TRASH` *Linux*

Set the trash implementation on Linux. Default is `gio`.

Options:

- `gvfs-trash`
- `trash-cli`
- `kioclient5`
- `kioclient`

Development Variables

The following environment variables are intended primarily for development and debugging purposes.

`ELECTRON_ENABLE_LOGGING`

Prints Chromium's internal logging to the console.

Setting this variable is the same as passing `--enable-logging` on the command line. For more info, see `--enable-logging` in command-line switches.

`ELECTRON_LOG_FILE`

Sets the file destination for Chromium's internal logging.

Setting this variable is the same as passing `--log-file` on the command line. For more info, see `--log-file` in command-line switches.

`ELECTRON_DEBUG_DRAG_REGIONS`

Adds coloration to draggable regions on `BrowserView`s on macOS - draggable regions will be colored green and non-draggable regions will be colored red to aid debugging.

`ELECTRON_DEBUG_NOTIFICATIONS`

Adds extra logs to `Notification` lifecycles on macOS to aid in debugging. Extra logging will be displayed when new Notifications are created or activated. They will also be displayed when common a tions are taken: a notification is shown, dismissed, its button is clicked, or it is replied to.

Sample output:

```
Notification created (com.github.Electron:notification:EAF7B87C-A113-43D7-8E76-F88EC9D73D44)
Notification displayed (com.github.Electron:notification:EAF7B87C-A113-43D7-8E76-F88EC9D73D4
Notification activated (com.github.Electron:notification:EAF7B87C-A113-43D7-8E76-F88EC9D73D4
Notification replied to (com.github.Electron:notification:EAF7B87C-A113-43D7-8E76-F88EC9D73
```

`ELECTRON_LOG_ASAR_READS`

When Electron reads from an ASAR file, log the read offset and file path to the system `tmpdir`. The resulting file can be provided to the ASAR module to optimize file ordering.

`ELECTRON_ENABLE_STACK_DUMPING`

Prints the stack trace to the console when Electron crashes.

This environment variable will not work if the `crashReporter` is started.

`ELECTRON_DEFAULT_ERROR_MODE` ***Windows***

Shows the Windows's crash dialog when Electron crashes.

This environment variable will not work if the `crashReporter` is started.

`ELECTRON_OVERRIDE_DIST_PATH`

When running from the `electron` package, this variable tells the `electron` command to use the specified build of Electron instead of the one downloaded by `npm install`. Usage:

```
export ELECTRON_OVERRIDE_DIST_PATH=/Users/username/projects/electron/out/Testing
```

Set By Electron

Electron sets some variables in your environment at runtime.

`ORIGINAL_XDG_CURRENT_DESKTOP`

This variable is set to the value of `XDG_CURRENT_DESKTOP` that your application originally launched with. Electron sometimes modifies the value of `XDG_CURRENT_DESKTOP` to affect other logic within Chromium so if you want access to the *original* value you should look up this environment variable instead.