crashReporter

> *Submit crash reports to a remote server.*

Process: Main, Renderer

The following is an example of setting up Electron to automatically submit crash reports to a remote server:

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

crashReporter.start({ submitURL: 'https://your-domain.com/url-to-submit' })
```

For setting up a server to accept and process crash reports, you can use following projects:

- socorro
- mini-breakpad-server

> ***Note:*** *Electron uses Crashpad, not Breakpad, to collect and upload crashes, but for the time being, the upload protocol is the same.*

Or use a 3rd party hosted solution:

- Backtrace
- Sentry
- BugSplat
- Bugsnag

Crash reports are stored temporarily before being uploaded in a directory underneath the app's user data directory, called 'Crashpad'. You can override this directory by calling `app.setPath('crashDumps', '/path/to/crashes')` before starting the crash reporter.

Electron uses crashpad to monitor and report crashes.

Methods

The `crashReporter` module has the following methods:

`crashReporter.start(options)`

- `options` Object
  - `submitURL` string (optional) - URL that crash reports will be sent to as POST. Required unless `uploadToServer` is `false`.
  - `productName` string (optional) - Defaults to `app.name`.
  - `companyName` string (optional) *Deprecated* - Deprecated alias for `{ globalExtra: { _companyName: ... } }`.
  - `uploadToServer` boolean (optional) - Whether crash reports should be sent to the server. If false, crash reports will be collected and stored in the crashes directory, but not uploaded. Default is `true`.
  - `ignoreSystemCrashHandler` boolean (optional) - If true, crashes generated in the main process will not be forwarded to the system crash handler. Default is `false`.
  - `rateLimit` boolean (optional) *macOS Windows* - If true, limit the number of crashes uploaded to 1/hour. Default is `false`.

- `compress` boolean (optional) - If true, crash reports will be compressed and uploaded with `Content-Encoding: gzip`. Default is `true`.
- `extra` Record<string, string> (optional) - Extra string key/value annotations that will be sent along with crash reports that are generated in the main process. Only string values are supported. Crashes generated in child processes will not contain these extra parameters to crash reports generated from child processes, call `addExtraParameter` from the child process.
- `globalExtra` Record<string, string> (optional) - Extra string key/value annotations that will be sent along with any crash reports generated in any process. These annotations cannot be changed once the crash reporter has been started. If a key is present in both the global extra parameters and the process-specific extra parameters, then the global one will take precedence. By default, `productName` and the app version are included, as well as the Electron version.

This method must be called before using any other `crashReporter` APIs. Once initialized this way, the crashpad handler collects crashes from all subsequently created processes. The crash reporter cannot be disabled once started.

This method should be called as early as possible in app startup, preferably before `app.on('ready')`. If the crash reporter is not initialized at the time a renderer process is created, then that renderer process will not be monitored by the crash reporter.

Note: You can test out the crash reporter by generating a crash using `process.crash()`.

Note: If you need to send additional/updated `extra` parameters after your first call `start` you can call `addExtraParameter`.

Note: Parameters passed in `extra`, `globalExtra` or set with `addExtraParameter` have limits on the length of the keys and values. Key names must be at most 39 bytes long, and values must be no longer than 127 bytes. Keys with names longer than the maximum will be silently ignored. Key values longer than the maximum length will be truncated.

Note: This method is only available in the main process.

`crashReporter.getLastCrashReport()`

Returns `CrashReport` - The date and ID of the last crash report. Only crash reports that have been uploaded will be returned; even if a crash report is present on disk it will not be returned until it is uploaded. In the case that there are no uploaded reports, `null` is returned.

Note: This method is only available in the main process.

`crashReporter.getUploadedReports()`

Returns `CrashReport[]`:

Returns all uploaded crash reports. Each report contains the date and uploaded ID.

Note: This method is only available in the main process.

`crashReporter.getUploadToServer()`

Returns `boolean` - Whether reports should be submitted to the server. Set through the `start` method or `setUploadToServer`.

Note: This method is only available in the main process.

`crashReporter.setUploadToServer(uploadToServer)`

- `uploadToServer` boolean - Whether reports should be submitted to the server.

This would normally be controlled by user preferences. This has no effect if called before `start` is called.

Note: This method is only available in the main process.

`crashReporter.addExtraParameter(key, value)`

- `key` string - Parameter key, must be no longer than 39 bytes.
- `value` string - Parameter value, must be no longer than 127 bytes.

Set an extra parameter to be sent with the crash report. The values specified here will be sent in addition to any values set via the `extra` option when `start` was called.

Parameters added in this fashion (or via the `extra` parameter to `crashReporter.start`) are specific to the calling process. Adding extra parameters in the main process will not cause those parameters to be sent along with crashes from renderer or other child processes. Similarly, adding extra parameters in a renderer process will not result in those parameters being sent with crashes that occur in other renderer processes or in the main process.

Note: Parameters have limits on the length of the keys and values. Key names must be no longer than 39 bytes, and values must be no longer than 20320 bytes. Keys with names longer than the maximum will be silently ignored. Key values longer than the maximum length will be truncated.

`crashReporter.removeExtraParameter(key)`

- `key` string - Parameter key, must be no longer than 39 bytes.

Remove an extra parameter from the current set of parameters. Future crashes will not include this parameter.

`crashReporter.getParameters()`

Returns `Record<string, string>` - The current 'extra' parameters of the crash reporter.

In Node child processes

Since `require('electron')` is not available in Node child processes, the following APIs are available on the `process` object in Node child processes.

`process.crashReporter.start(options)`

See `crashReporter.start()`.

Note that if the crash reporter is started in the main process, it will automatically monitor child processes, so it should not be started in the child process. Only use this method if the main process does not initialize the crash reporter.

`process.crashReporter.getParameters()`

See `crashReporter.getParameters()`.

`process.crashReporter.addExtraParameter(key, value)`

See `crashReporter.addExtraParameter(key, value)`.

`process.crashReporter.removeExtraParameter(key)`

See `crashReporter.removeExtraParameter(key)` .

Crash Report Payload

The crash reporter will send the following data to the `submitURL` as a `multipart/form-data` `POST` :

- `ver` string - The version of Electron.
- `platform` string - e.g. 'win32'.
- `process_type` string - e.g. 'renderer'.
- `guid` string - e.g. '5e1286fc-da97-479e-918b-6bfb0c3d1c72'.
- `_version` string - The version in `package.json` .
- `_productName` string - The product name in the `crashReporter` `options` object.
- `prod` string - Name of the underlying product. In this case Electron.
- `_companyName` string - The company name in the `crashReporter` `options` object.
- `upload_file_minidump` File - The crash report in the format of `minidump` .
- All level one properties of the `extra` object in the `crashReporter` `options` object.