Multithreading

With Web Workers, it is possible to run JavaScript in OS-level threads.

Multi-threaded Node.js

It is possible to use Node.js features in Electron's Web Workers, to do so the `nodeIntegrationInWorker` option should be set to `true` in `webPreferences`.

```
const win = new BrowserWindow({
  webPreferences: {
    nodeIntegrationInWorker: true
  }
})
```

The `nodeIntegrationInWorker` can be used independent of `nodeIntegration`, but `sandbox` must not be set to `true`.

Available APIs

All built-in modules of Node.js are supported in Web Workers, and `asar` archives can still be read with Node.js APIs. However none of Electron's built-in modules can be used in a multi-threaded environment.

Native Node.js modules

Any native Node.js module can be loaded directly in Web Workers, but it is strongly recommended not to do so. Most existing native modules have been written assuming single-threaded environment, using them in Web Workers will lead to crashes and memory corruptions.

Note that even if a native Node.js module is thread-safe it's still not safe to load it in a Web Worker because the `process.dlopen` function is not thread safe.

The only way to load a native module safely for now, is to make sure the app loads no native modules after the Web Workers get started.

```
process.dlopen = () => {
  throw new Error('Load native module is not safe')
}
const worker = new Worker('script.js')
```