Checking file existence asynchronously

> This article was originally published at https://gist.github.com/joepie91/bbf495e044da043de2ba.

Checking whether a file exists before doing something with it, can lead to race conditions in your application. Race conditions are extremely hard to debug and, depending on where they occur, they can lead to **data loss or security holes**. Using the synchronous versions will **not** fix this.

Generally, just do what you want to do, and handle the error if it doesn't work. This is much safer.

- **If you want to check whether a file exists, before reading it:** just try to open the file, and handle the `ENOENT` error when it doesn't exist.
- **If you want to make sure a file doesn't exist, before writing to it:** open the file using an exclusive mode, eg. `wx` or `ax`, and handle the error when the file already exists.
- **If you want to create a directory:** just try to create it, and handle the error if it already exists.
- **If you want to remove a file or directory:** just try to unlink the path, and handle the error if it doesn't exist.

If you're *really, really sure* that you need to use `fs.exists` or `fs.stat`, then you can use the example code below to do so asynchronously. If you just want to know how to promisify an asynchronous callback that doesn't follow the nodeback convention, then you can look at the example below as well.

> You should almost never actually use the code below. The same applies to `fs.stat` (when used for checking existence). Make sure you have read the text above first!

```
const fs = require("fs");
const Promise = require("bluebird");

function existsAsync(path) {
  return new Promise(function(resolve, reject){
    fs.exists(path, function(exists){
      resolve(exists);
```

```
    })
  })
}
```

Revision #1
Created 11 December 2024 12:29:03 by joepie91
Updated 11 December 2024 18:44:19 by joepie91