Wrapping callback functions

So far we have assumed that node-style callback-based functions have already been wrapped for us. How can we do the wrapping ourselves?

In bluebird, we can use this construct to do it manually:

```
var Promise = require('bluebird');

function readFilePromise() {
    return new Promise(function(fulfill, reject) {
        fs.readFile(path, function(error, content) {
            if (error) reject(error)
            else fulfill(content);
        })
    })
}
```

Or we could rely on bluebird's efficient `promisify` function

```
var readFilePromise = Promise.promisify(fs.readFile, fs);
```

If we want to promisify all functions in a given object we can use `Promise.promisifyAll()`

```
Promise.promisifyAll(fs);
```

This will generate functions with the Async suffix and add them to the object, e.g. `fs.readFileAsync()`