Callbacks

Why are Callbacks Useful?

- Allows for methods to be non-blocking on the main event loop
- Passes continuation of a calling method to the callee, allowing for flow control
- Many of the NodeJS internal libraries leverage callbacks for asynchronous operations

NodeJS Callback Convention

- Callback conventions are uniform in NodeJS
- Also known as "error-first", "errback", "Node-style callbacks"
- Callback functions accept an error argument as the first parameter, and optionally other parameters after that
- A guaranteed convention allows for code uniformity and ease of maintainability

NodeJS Callback Practices

- Callbacks in Node should only be called once
- The first `error` argument is typically (although not always) of type `Error`
- It is often helpful to define callbacks as anonymous inline functions

Example: callbacks + `setTimeout()`

```
function hello(name, callback) {
  //if the name is not provided,
  if (!name) {
    //call back with an Error
    return callback(new Error('no name provided!'));
  }
  //otherwise, saying goodbye takes a moment
  setTimeout(function(){
    //call back with a friendly goodbye
    callback(null, `farewell, ${name}!`)
  }, 500);
};

//define `goodbye` as a callback
function goodbye(err, message) {
  //if there was a problem,
  if (err) {
    //handle the error
    //in this case, print the problem
    console.log(`Problem: ${err}`);
    return;
  }
  //otherwise, log the message
  console.log(message);
};

//prints "farewell, Sally!" after ~500ms
hello('Sally', goodbye);
//prints "Problem: no name provided!"
hello(null, goodbye);
```

Exercise: Drink Refill

- Write a function that takes three parameters: `drinkType`, `hasIce`, and `callback`.
- `drinkType` is mandatory. If it is empty, null, or undefined, call back with an `Error` stating that it is
- If the drink is `"lemonade"` and does not have ice, call back with an `Error` stating that drink requires ice.
- Otherise, refill the drink, simulating the time for refill with a `setTimeout` of one second. Call back with a message specifying

Preview of future topic: `fs.readFile()`

- Read a file from disk without blocking:

```
var fs = require('fs');
//does the second parameter to `fs.readFile` look familiar?
fs.readFile('./my-file.txt', function(err, fileContents){
    console.log(fileContents);
})
```