Fibers

Traditional JavaScript and Node.js API will utilize the concept of asynchronous callbacks for virtually every input/output operation. It is a common pattern for Node.js API. Whether it's reading a disk file, accessing a database table, or calling a web service, the results are not directly returned to the caller; instead, they are passed into a separate function referred to as a callback.

Because of JavaScript's design, the default way to "freeze" a computation and have the "rest of it" execute later (asynchronously) is to put "the rest of it" inside of a callback function.

For certain types of applications, this can be quite useful and performant; however, it also creates the problem more commonly known as Callback Hell. A lot of code ends up looking like this:

```
fs.readdir(source, function (err, files) {
  files.forEach(function (filename, fileIndex) {
    gm(source + filename).size(function (err, values) {
      aspect = (values.width / values.height);
      widths.forEach(function (width, widthIndex) {
        height = Math.round(width / aspect);
        this.resize(width, height).write(dest + 'w' + width + '_' + filename, function(err) {
          if (err) console.log('Error writing file: ' + err)
        })
      })
    })
  })
})
```

Even when just a few simple API are being invoked, the pyramid shape and all the `})` at the end are needed because each API call returns its data into a separate anonymous callback function.

This is generally not conducive to writing top-down transactional business applications, where everything happens in a procedural manner, one step at a time.

Instead of:

```
fs.readdir(source, function (err, files) {
  // process files here
});
```

The business developer prefers this:

```
files = fs.readdir(source);
// process files here
```

Coding business applications in a top-down manner is more intuitive, especially if you come from an IBM i background, and have used languages like RPG and CL.

To combat this problem and help eliminate Callback Hell, Profound.js integrates a component called ***Fibers***. Through Profound.js Fibers, you can code input/output operations in a familiar top-down manner. All Profound.js input/output API utilize Fibers internally. You can also call any traditional asynchronous Node.js API in a top-down manner through our simple to use pjs.fiber.wrap() and pjs.fiber.run() API.

With Fibers, even though you will be coding your applications in a simple top-down manner, asynchronous calls will still happen behind the scene. This ensures that your Node.js applications continue to perform and scale well. The Node.js event loop is not blocked when Fibers is used.

Video Tutorial