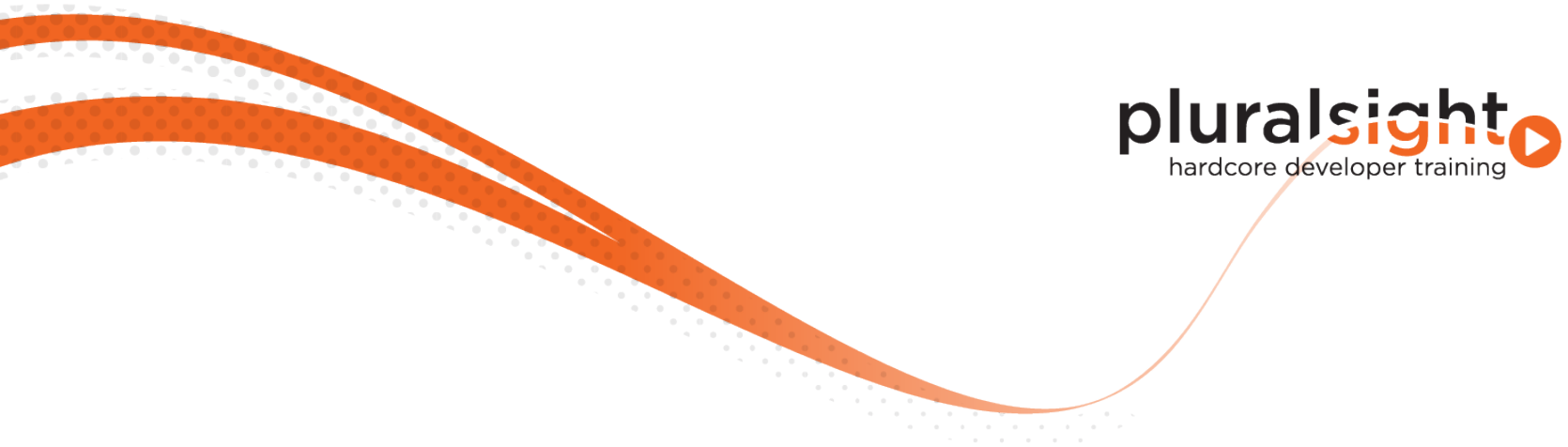
Events and Streams

Paul O'Fallon
@paulofallon



pluralsight
hardcore developer training

Outline

- Callbacks vs. Events
- Node's EventEmitter class
- Patterns for using EventEmitters
- Readable and Writable Streams
- Piping Between Streams

Non-blocking doesn't always mean callbacks

Callbacks:

```
getThem(param, function(err, items) {
  // check for error
  // operate on array of items
});
```

- Request / Reply
- No results until all results
- Either error or results

Events:

```
var results = getThem(param);

results.on('item', function(i) {
  // do something with this one item
});

results.on('done', function() {
  // No more items
});

results.on('error', function(err) {
  // React to error
});
```

- Publish / Subscribe
- Act on results as they arrive
- Partial results before error

Node's "EventEmitter" class

The publisher:

```
emitter.emit(event, [args]);
```

The subscriber:

```
emitter.on(event, listener);
```

- **The "event" can be any string**
- **An event can be emitted with zero or more arguments**
- **The set of events and their arguments constitute a "interface" exposed to the subscriber by the publisher (emitter).**

Two common patterns for EventEmitters:

1. **As a return value from a function call (see earlier example)**
2. **Objects that extend EventEmitter to emit events themselves**


pluralsight
hardcore developer training

Streams in Node.js

- **Streams are instances of (and extensions to) EventEmitter with an agreed upon "interface"**
- **A unified abstraction for managing data flow, including:**
  - Network traffic (http requests & responses, tcp sockets)
  - File I/O
  - stdin / stdout / stderr
  - … and more!
- **A stream is an instance of either**
  - ReadableStream
  - WritableStream
  - … or both!
- **A ReadableStream can be pipe()'d to a WritableStream**
  - Applies "backpressure"

ReadableStream & WritableStream

ReadableStream

- readable [boolean]
- event: ‘data’
- event: ‘end’
- event: ‘error’
- event: ‘close’
- pause()
- resume()
- destroy()
- pipe()

WritableStream

- writable [boolean]
- event: ‘drain’
- event: ‘error’
- event: ‘close’
- event: ‘pipe’
- write()
- end()
- destroy()
- destroySoon()

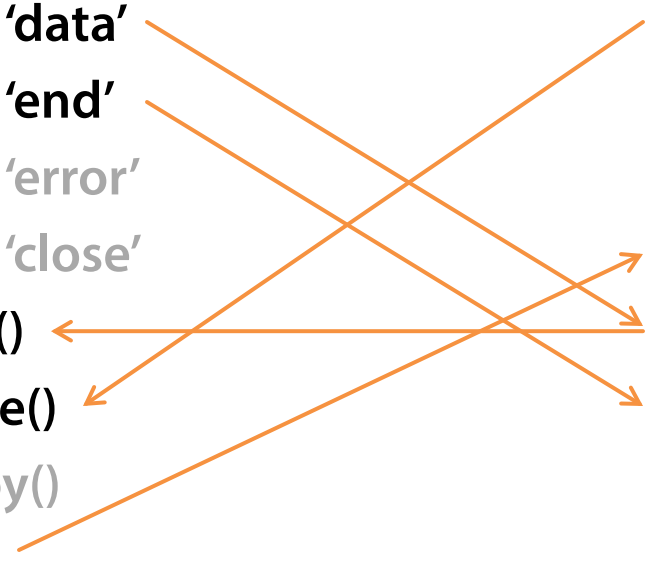
Piping Streams

ReadableStream

- readable [boolean]
- event: 'data'
- event: 'end'
- event: 'error'
- event: 'close'
- pause()
- resume()
- destroy()
- pipe()

WritableStream

- writable [boolean]
- event: 'drain'
- event: 'error'
- event: 'close'
- event: 'pipe'
- write()
- end()
- destroy()
- destroySoon()




pluralsight
hardcore developer training

Conclusion

- Callbacks vs. Events
- Node's EventEmitter class
- Patterns for implementing EventEmitters
- Readable and Writable Streams
- Piping Between Streams

References

- Streams, Pipes and Mega Pipes:
  http://felixge.s3.amazonaws.com/11/nodejs-streams.pdf
- Node.js Documentation
  http://nodejs.org/api/