JavaScript Promises and Async Programming

UNDERSTANDING PROMISES

Nathan Taylor
SOLUTION ARCHITECT

@taylonr www.taylonr.com

Building a Callback Pyramid

Why Is the Pyramid Bad?

Callback Pyramid of Doom

A common problem that arises when a program uses many levels of nested indentation to control access to a function.

https://en.wikipedia.org/wiki/Pyramid_of_doom_(programming)

Why “Pyramid”?

callbacks.js

```
a("test", (err, aResult) => {
    b(aResult, (err, bResult) => {
        c(bResult, (err, cResult) => {
            d(cResult);
        });
    });
});
```

Why Is It a Pyramid of Doom?

xhr.js

```
xhr.onload = () => {
    xhr2.onload = () => {
      const order = JSON.parse(xhr2.responseText);
      xhr3.onload = () => {
        const payments = JSON.parse(xhr2.responseText);
        xhr4.onload = () => {
          const paymentType = JSON.parse(xhr4.responseText);
        };
      };
    };
  };
```

Why Is It a Pyramid of Doom?

Dirty Code

Handling Errors

xhr.js

```
xhr.onload = () => {
    xhr2.onload = () => {
      const order = JSON.parse(xhr2.responseText);
      xhr3.onload = () => {
        const payments = JSON.parse(xhr2.responseText);
        xhr4.onload = () => {
          const paymentType = JSON.parse(xhr4.responseText);
        };
```

xhr3's request returns a 500

xhr.js

```
xhr.onload = () => {
    xhr2.onload = () => {
      xhr3.onload = () => {
        xhr4.onload = () => {};
        xhr4.onerror = () => {};
      };
      xhr3.onerror = () => {};
    };
    xhr2.onerror = () => {}
 };
xhr.onerror = () => {};
```

Solving the Callback Pyramid

Promise

Object that represents the eventual completion (or failure) of an asynchronous operation, and its resulting value.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

Promise

Object that represents the eventual completion (or failure) of an asynchronous operation, and its resulting value.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

Promise

Object that represents the eventual completion (or failure) of an asynchronous operation, and its resulting value.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

Readable asynchronous code

Promise States

Pending

Fulfilled

Rejected

Promise States

Settled

Resolved

Promises are not lazy

Setting up the Sample Project

Online sporting goods retailer

API Entities

Addresses	Items	Orders
Users	ItemCategories	OrderStatuses