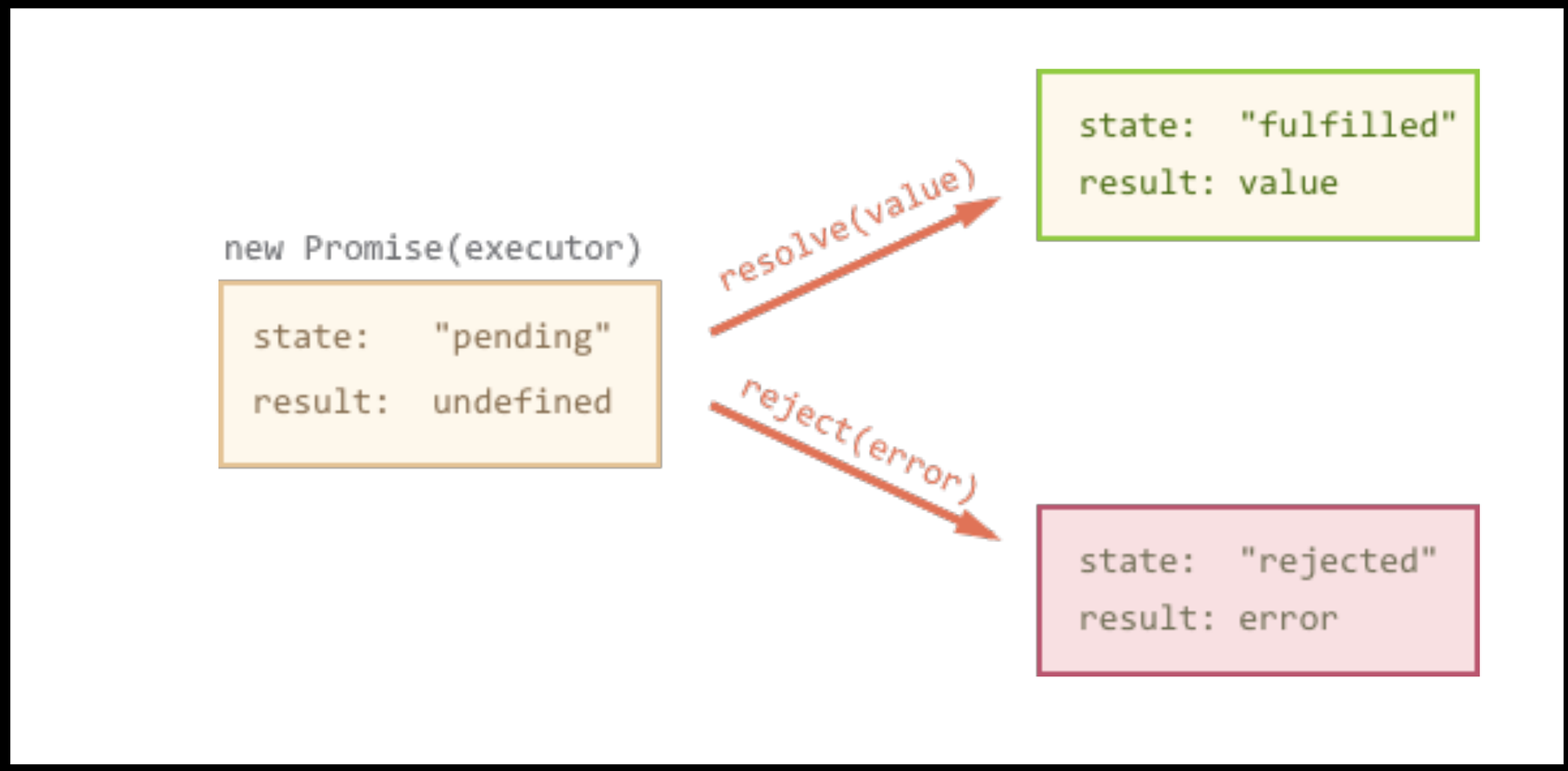
PROMISES

PROMISES SYNTAX



COMSUMERS

```
let promise = new Promise(function(resolve, reject) {
    setTimeout(() => reject(new Error("Oh NOOOOOO!")), 500);
    setTimeout(() => resolve("done!"), 1000);
  });


promise.then(
    (result)=> console.log(result) ,
    (error) => console.log(error)
)
```

Updated Timer

WHAT GETS PRINT OUT?

OH NOOOOOO!

CONSUMERS THEN & CATCH

```
let promise = new Promise(function(resolve, reject) {
    setTimeout(() => reject(new Error("Oh NOOOOOO!")), 1500);
    setTimeout(() => resolve("done!"), 1000);
  });
```

```
promise.then(
    (result)=> console.log(result)
    ).catch(
        (error) => console.log(error)
    )
```

CATCH SYNTAX

LET'S TRY OUR WEBREQUEST EXAMPLE

SETUP THE GOBAL VARIABLES AND OPTIONS AGAIN

```
const request = require('request')

var APP = {
    url: "https://www.cs.virginia.edu/~dgg6b/samples/JSON.txt",
    webResponse : ""
}

options = { json: true }
```

```
let webRequest = new Promise(function(resolve, reject) {
    handleRequest = (err,  response) => {
        if (err) { reject(err)}
        resolve(body)
        }
    request(APP.url, options,  handleRequest);
  });
```

```
webRequest.then(
    (result)=>console.log(result)
    ).catch(
        (error) => console.log(error)
    )
```

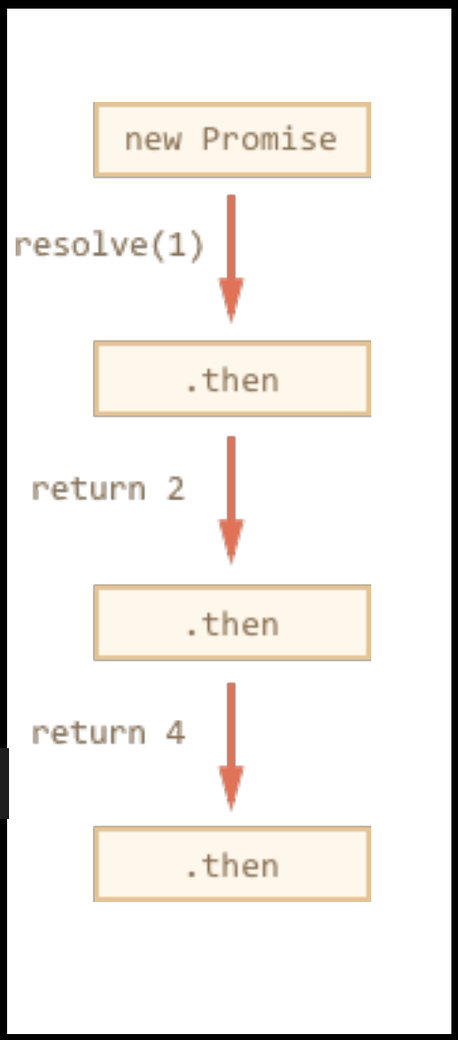
WHAT ABOUT THAT ISSUE OF SETTING THE GLOBAL VARIABLE

```
let webRequest = new Promise(function(resolve, reject) {
    handleRequest = (err, res, body) => {
        if (err) { reject(err)}
        APP.webResponse = body
        resolve(body)
        }
    request(APP.url, options,  handleRequest);
  });
```

```
webRequest.then(
    (result)=>{
        console.log(APP.webResponse)
        console.log(result)
    }
    ).catch(
        (error) => console.log(error)
    )
```

PROMISES AND CHAINING

```
chain = new Promise(function(resolve, reject) {
        setTimeout(() => resolve(1), 1000);
      }).then(function(result) {
        console.log("Level 1 result: "+ result)
        return result * 2;
      }).then(function(result) {
        console.log("Level 2 result: "+ result)
        return result * 2;
      }).then(function(result) {
        console.log("Level 3 result: "+ result)
      });
```



YOUR FRIST PIECE OF REACT NATIVE CODE

```
function getMoviesFromApiAsync() {
    return fetch('https://facebook.github.io/react-native/
movies.json')
      .then((response) => response.json())
      .then((responseJson) => {
        return responseJson.movies;
      })
      .catch((error) => {
        console.error(error);
      });
  }
```

https://facebook.github.io/react-native/docs/network

ASYNC/AWAIT

SPECIAL SYNTAX FOR PROMISES

ASYNC FUNCTIONS

ASYNC KEYWORD

```
async function simpleFunction() {
  return 1;
}
```

Async keyword means that function returns a promise

ASYNC FUNCTIONS

ASYNC KEYWORD

```
async function simpleFunction() {
    return 1;
}
```

But this does not return a promise

Async keyword means that function returns a promise

ASYNC FUNCTIONS

```
async function simpleFunction() {
    return 1;
  }
```

Equivalent

```
async function simpleFunction() {
    return Promise.resolve(1);
  }
```

THE RETURN VALUE GETS AUTOMATICALLY WRAPPED IN A RESOLVE

ASYNC FUNCTIONS

```
async function simpleFunction() {
    return Promise.resolve(1);
  }
```

```
simpleFunction().then((result)=>{
        console.log(result)
    }
  )
```

NOTICE THE BRACKETS

We are invoking the function which returns a promise that we access the then function from.

AWAIT KEYWORD

```
async function simpleFunction() {
    let promise = new Promise((resolve, reject) => {
        setTimeout(() => resolve("Promise Complete!"), 1000)
      })

      let result = await promise // wait till the promise resolves

      console.log(result) // “Promise Complete!”
    }

simpleFunction().then((result)=>{
        console.log("Then function Fired with result: "
                    + result +  " ")
    }
  )
```

THIS PRINTS UNDEFINED BECAUSE DEFAULT PROMISE IS RETURNED WITH AN EMPTY RESULT

AWAIT KEYWORD

```
async function simpleFunction() {
    let promise = new Promise((resolve, reject) => {
        setTimeout(() => resolve("Promise Complete!"), 1000)
      })

      let result = await promise // wait till the promise resolves (
        console.log(result) // "Promise Complete!
        return promise

    }

simpleFunction().then((result)=>{
        console.log("Then function Fired with result: "
        + result +  " ")
    }
  )
```

The object returned

State: fulfilled

result: Promise Complete

AWAIT KEYWORD

MISSING ASYNC KEYWORD

```
function badFunction() {
  let promise = Promise.resolve(1);
  let result = await promise; // Syntax error
}
```

```
SyntaxError: await is only valid in async function
    at new Script (vm.js:84:7)
    at createScript (vm.js:264:10)
```

CAT FACTS

```
const request = require('request')


var url = "https://catfact.ninja/fact"

async function getCatFact(){
  let webRequest = new Promise(function(resolve, reject) {
    handleRequest = (err, response) => {
        if (err) { reject(err)}
            resolve(JSON.parse(response.body))
        }
    request(url,  handleRequest);
  })
  catFact = await webRequest
  console.log(catFact)
}

getCatFact()
```

HANDLING ERRORS WITH AWAIT

```
    try{
        catfact = await webRequest
    }catch(err){
        console.log(err)
    }
```

MODULES (IMPORTS & EXPORTS)

EXPORT AND IMPORT

- The export keyword is used to export objects and primitives so that can be used by other programs.

- There are two types of exports
  - Named and default

EXPORTS AND IMPORTS

- Named exports
  - You must used the same name as the object or primitive that is export.
- Default export
  - You can import default exports with any name.

YOU CAN HAVE MULTIPLE NAME EXPORTS IN A MODULE BUT ONLY ONE DEFAULT EXPORT

EXPORT AND IMPORT

```
config = {
    version: '1.2.0',
    name: 'My Mobile Application'
}


export default config
```

export.js

DOES WORK DEFAULT NODE.JS INSTALLATION

For Named exports
The names in the
Import need to match

```
import configuration from './export'

console.log(configuration.version)
```

Import.js

NAMED EXPORTS

```
const equation = Math.PI + Math.SQRT2;
var shape = {
    options: {
        color:'white',
        thickness:'2px'
    },
}
export { equation, shape }
```

DOES WORK DEFAULT NODE.JS INSTALLATION

export.js

Import.js

```
import {equation, shape} from 'export'

console.log(equation + ' ' + shape.options.color)
```

REFERENCES

- Javascript the Good Parts - Douglas Crockford
- JavaScript Programmer's Reference- Christian MacAuley and Paul Jobson
- https://developer.mozilla.org/en-US/docs/web/javascript/reference
- https://javascript.info/function-expressions-arrows
- https://medium.com/poka-techblog/simplify-your-javascript-use-map-reduce-and-filter-bd02c593cc2d
- https://javascript.info/promise-basics
- https://javascript.info/async-await