Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

Clase Node.js

Tecnologías y Aplicaciones Web

Gabriel Vidal Salazar

Herramientas a utilizar

Hello World!

```
const http = require('http');

const server = http.createServer((req, res) => {
  res.end('Hello World!');
});

server.listen(3000, () => {
  console.log('Server running at http://localhost:3000/');
});
```

Hello World!

```
const http = require('http');

const server = http.createServer((req, res) => {
  res.end('Hello World!');
});

server.listen(3000, () => {
  console.log('Server running at http://localhost:3000/');
});
```

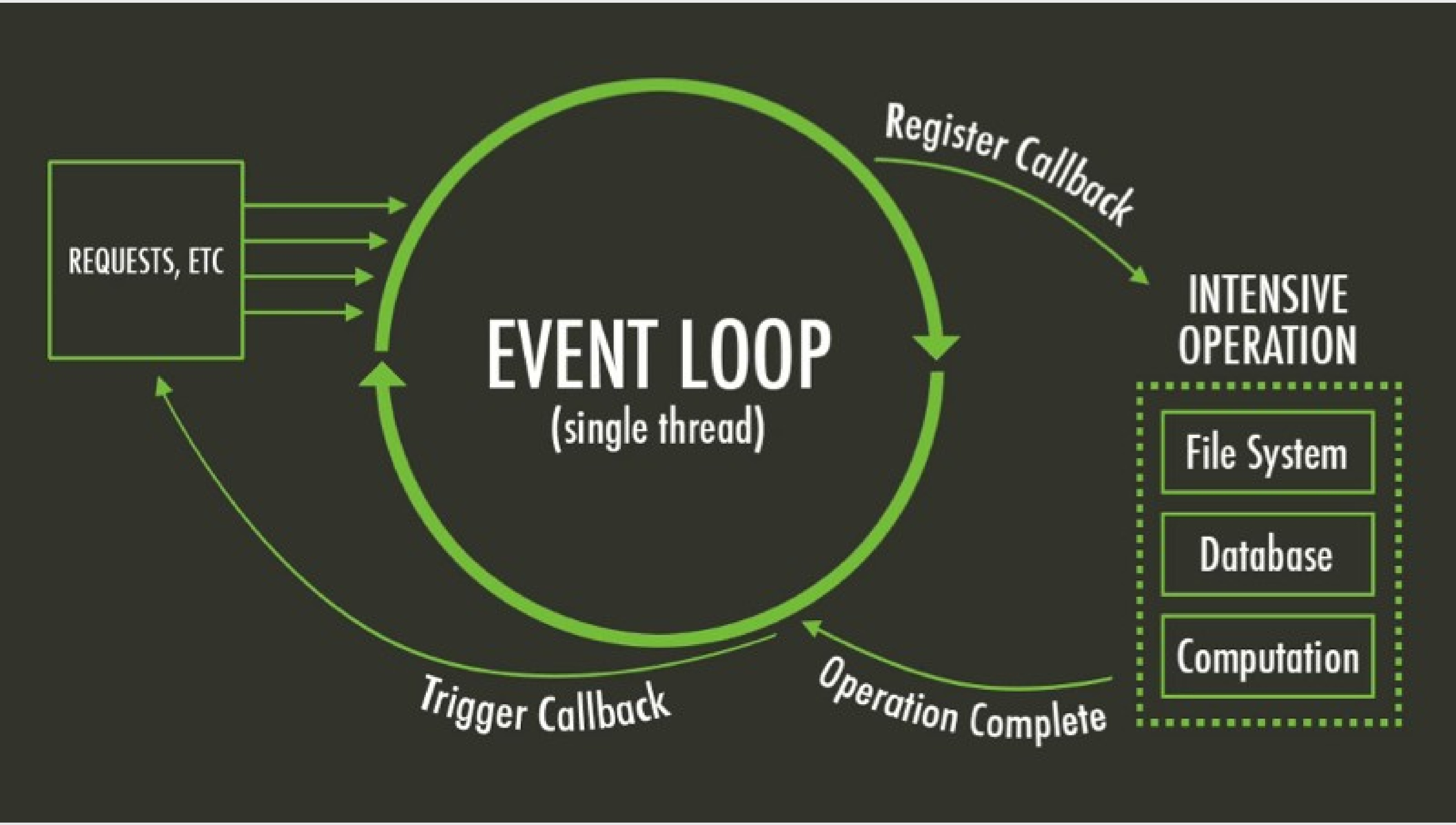
Arrow Functions (ES6)

```
const multiply = function(x, y) {
    return x * y;
}


const multiplyArrow = (x, y) => x * y;
```

Más información: MDN web docs

JavaScript is ***single threaded***



Source: Medium

Event Loop

```
const multiply =
  (x, y) => x * y;


const result = multiply(
  x,
  y,
);

console.log(result);
```

Call Stack

Event Loop

```
console.log('Starting Program');

/**
 * funcion que realiza una operación
 * asíncrona y demora 2 segundos. Al
 * terminar retorna 'IIC2513'
 */
asyncQuery(data => console.log(data));


console.log('End Program');
```

Event Loop: Más información

Video:

Philip Roberts: What the heck is the event loop anyway? | JSConf EU 2014

Artículos:

- MDN web docs
- Carbon Five

Interactivo:

- loupe

¿Qué ejecutamos al terminar?

Callbacks

```
function callback(data) {
   console.log(data);
}

doAsyncOp(callback);
```

```
function doAsyncOp(callback) {
   // obtain data asynchronously
   const data = 'Text';
   callback(data);
}
```

Más información: MDN web docs

Callbacks: Pero….

```
function doAsyncOp(function (data1) {
   function doAsyncOp2(data1, function (data2) {
      function doAsyncOp3(data2, function (data3) {
         console.log(data3);
      });
   });
});
```

Esto se conoce como ”Callback Hell” o ”Piramid of doom”

Promesas

```
doAsyncOp()
   .then(function (data1) {
      return doAsyncOp2(data1);
   })
   .then(function (data2) {
      return doAsyncOp3(data2);
   })
   .then(function (data3) {
      console.log(data3);
   });
```

Más información: MDN web docs

Promesas

```
const testPromise = doAsyncOp();

const newTestPromise = testPromise
   .then(function(data1) {
      return `Data: ${data1}`;
   });

newTestPromise.then(function(text) {
   console.log(text);
})
```

Async/Await

```
async function getData() {
   const data1 = await doAsyncOp();
   const data2 = await doAsyncOp2(data1);
   const data3 = await doAsyncOp3(data2);
   return data3;
}


getData().then(function(data3) {
   console.log(data3);
});
```

Más información: MDN web docs