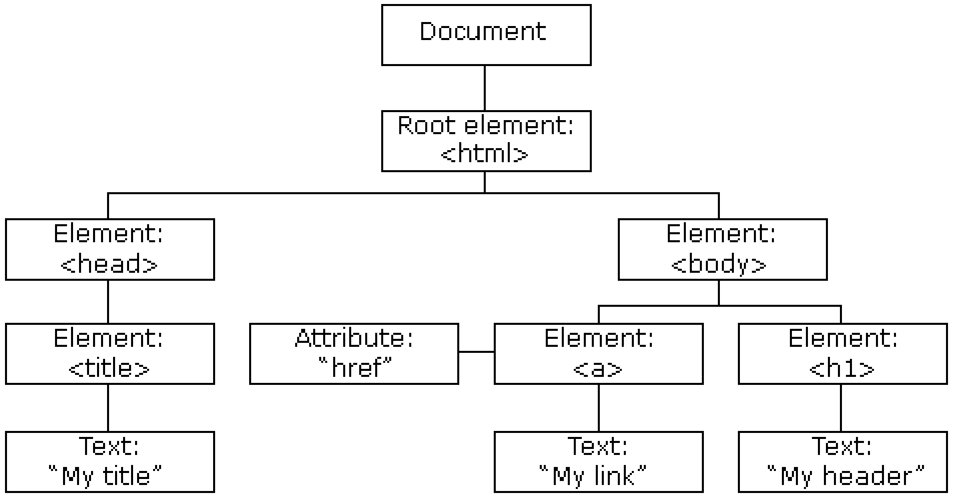
Html e DOM

The HTML DOM (Document Object Model)

When a web page is loaded, the browser creates a **D**ocument **O**bject **M**odel of the page.



With the HTML DOM, JavaScript can access and change all the elements of an HTML document.

https://www.w3schools.com/whatis/whatis_htmldom.asp

Javascript

JavaScript was initially created to *“make web pages alive”*.

Scripts are provided and executed as plain text. They don’t need special preparation or compilation to run.

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Statements</h2>

<p>A <b>JavaScript program</b> is a list of <b>statements</b> to be executed by a computer.</p>

<p id="demo"></p>

<script>
var x, y, z;  // Declare 3 variables
x = 5;    // Assign the value 5 to x
y = 6;    // Assign the value 6 to y
z = x + y;  // Assign the sum of x and y to z

document.getElementById("demo").innerHTML =
"The value of z is " + z + ".";
</script>

</body>
</html>
```

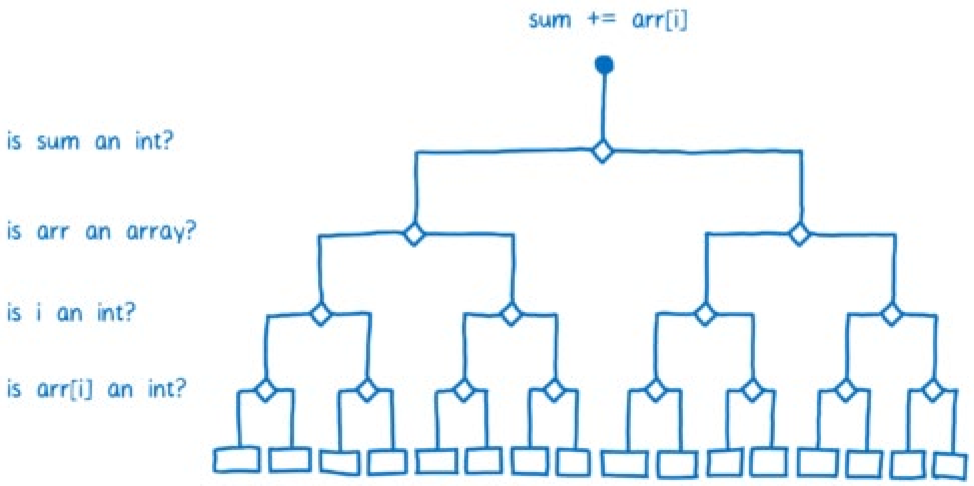
https://www.w3schools.com/js/js_examples.asp

Javascript

Javascript è un linguaggio debolmente tipizzato

https://hacks.mozilla.org/2017/02/a-crash-course-in-just-in-time-jit-compilers/

```
function arraySum(arr) {
  var sum = 0;
  for (var i = 0; i < arr.length; i++) {
    sum += arr[i];
  }
}
```



Javascript

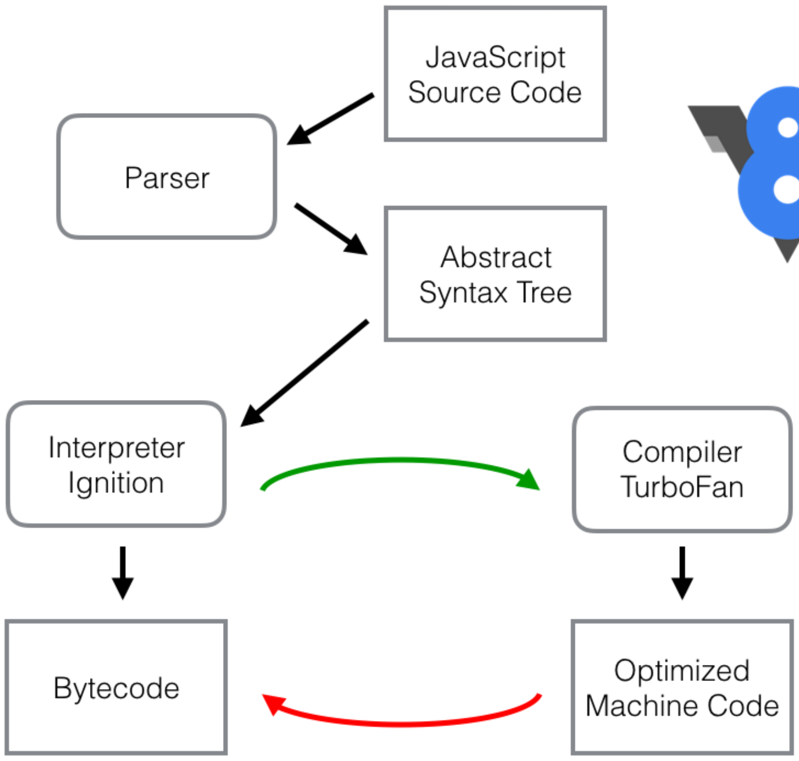
(SpiderMonkey)

(Nitro)

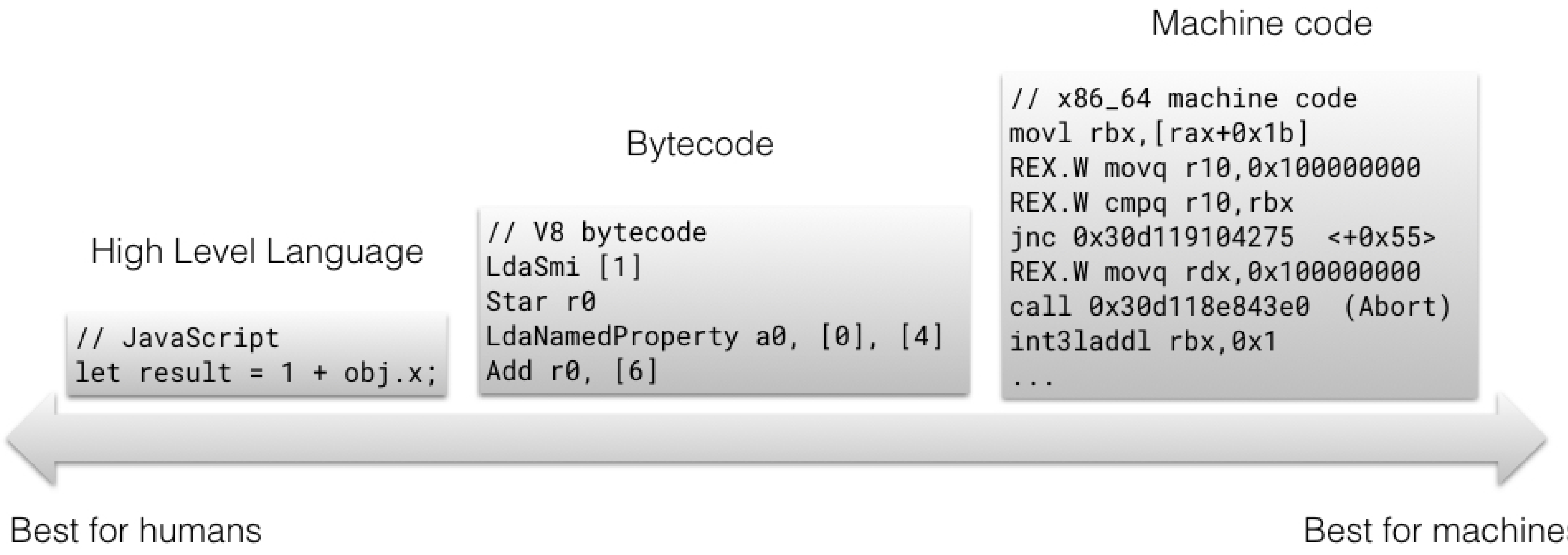
1. The engine (embedded if it’s a browser reads (“parses”) the script.

2. Then it converts (“compiles”) the script the machine language.

3. And then the machine code runs, pretty fast.



Javascript



```
// JavaScript
let result = 1 + obj.x;
```

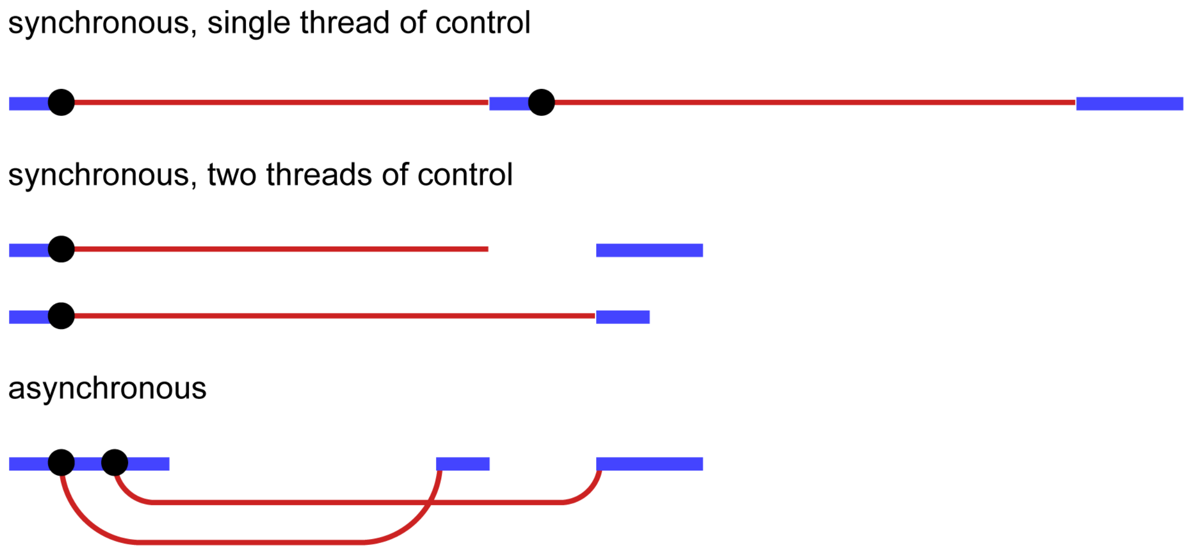
```
// V8 bytecode
LdaSmi [1]
Star r0
LdaNamedProperty a0, [0], [4]
Add r0, [6]
```

```
// x86_64 machine code
movl rbx,[rax+0x1b]
REX.W movq r10,0x100000000
REX.W cmpq r10,rbx
jnc 0x30d119104275  <+0x55>
REX.W movq rdx,0x100000000
call 0x30d118e843e0  (Abort)
int3laddl rbx,0x1
...
```

@fhinkel

Javascript

JavaScript is always synchronous and single-threaded. If you're executing a JavaScript block of code on a page then no other JavaScript on that page will currently be executed.


synchronous, single thread of control
synchronous, two threads of control
asynchronous

Javascript – Callback and Promise

One approach to asynchronous programming is to make functions that perform a slow action take an extra argument, a *callback function*. The action is started, and when it finishes, the callback function is called with the result.

```
setTimeout(() => console.log("Tick"), 500);
```

A *promise* is an asynchronous action that may complete at some point and produce a value. It is able to notify anyone who is interested when its value is available.

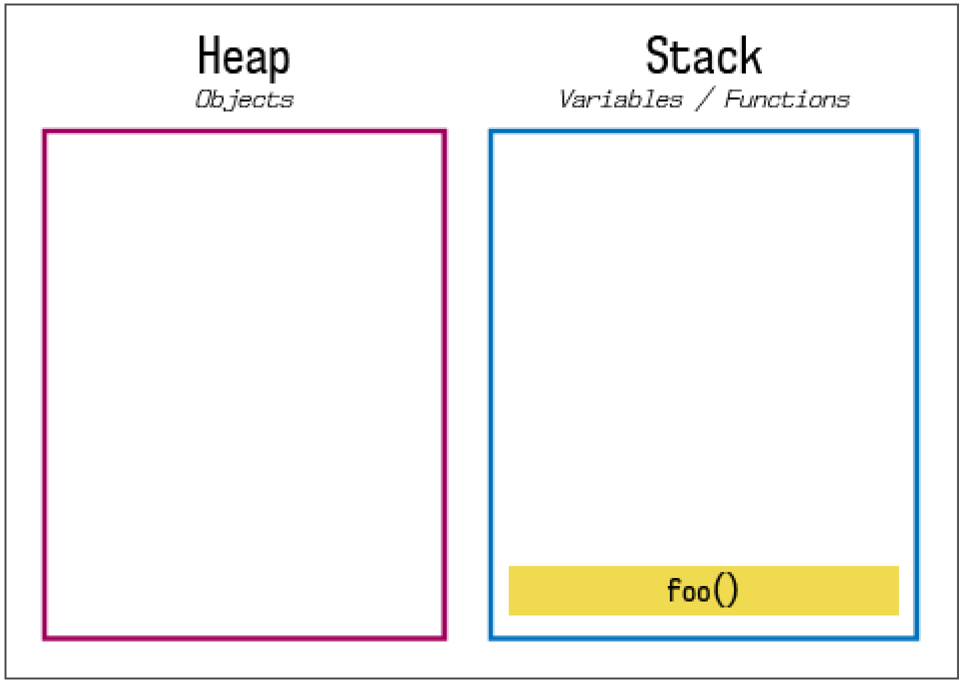
```
let fifteen = Promise.resolve(15);
fifteen.then(value => console.log(`Got ${value}`));
```

Javascript – Callback and Promise

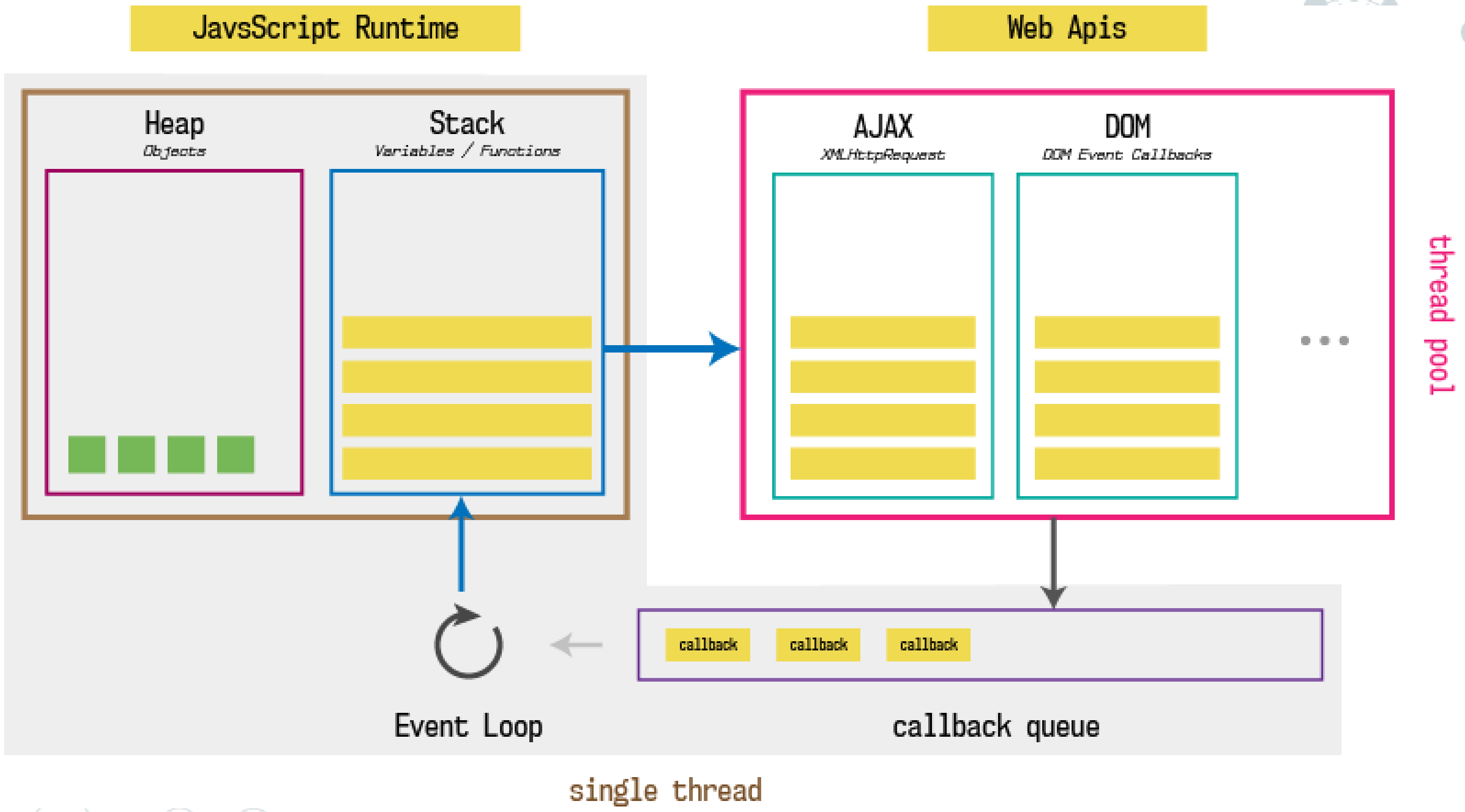
JavsScript Program

JavsScript Runtime

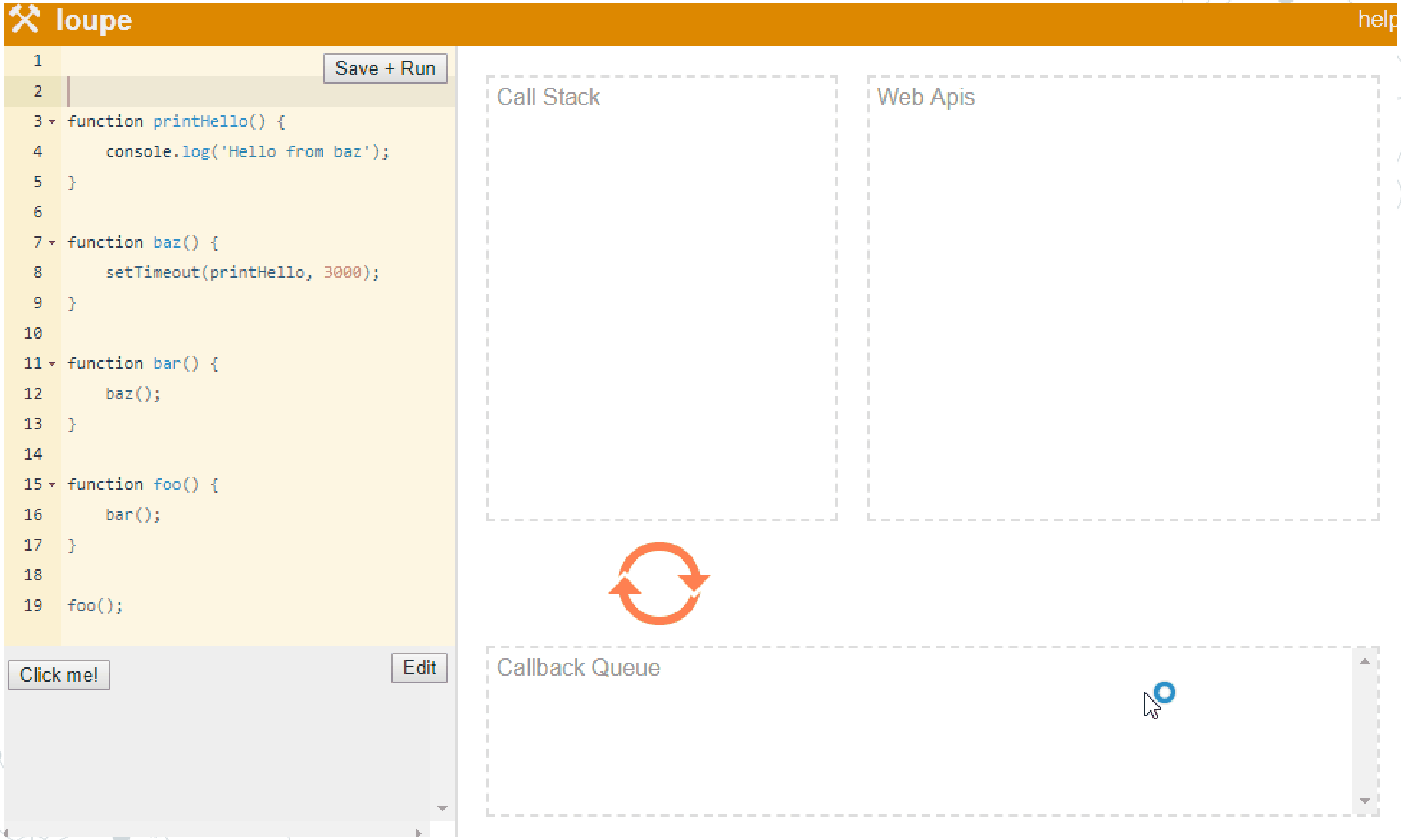
```
function baz(){
    console.log('Hello from baz');
}

function bar() {
    baz();
}

function foo() {
    bar();
}

foo();
```



Javascript – Callback and Promise



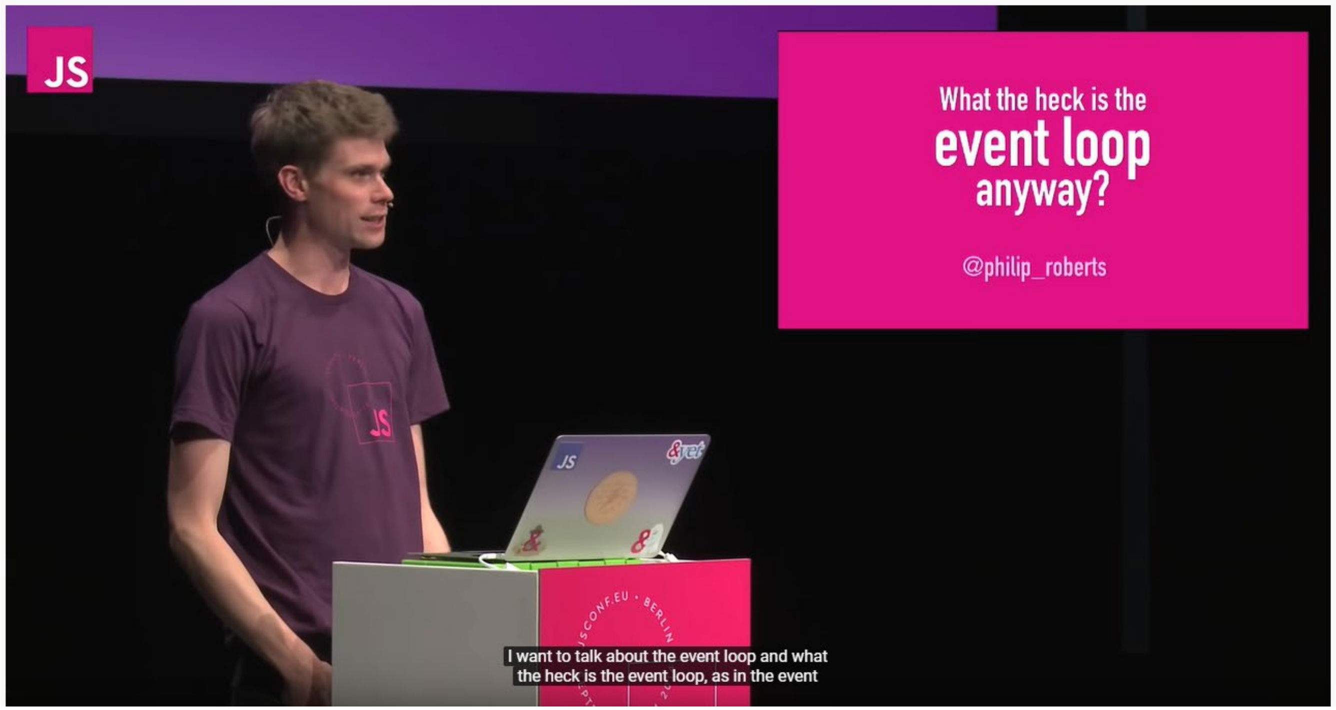
Javascript – Callback and Promise



```

function printHello() {
    console.log('Hello from baz');
}

function baz() {
    setTimeout(printHello, 3000);
}

function bar() {
    baz();
}

function foo() {
    bar();
}

foo();
```

Javascript – Callback and Promise

https://www.youtube.com/watch?v=8aGhZQkoFbQ


JS
What the heck is the
event loop
anyway?
@philip_roberts
I want to talk about the event loop and what
the heck is the event loop, as in the event

Backend

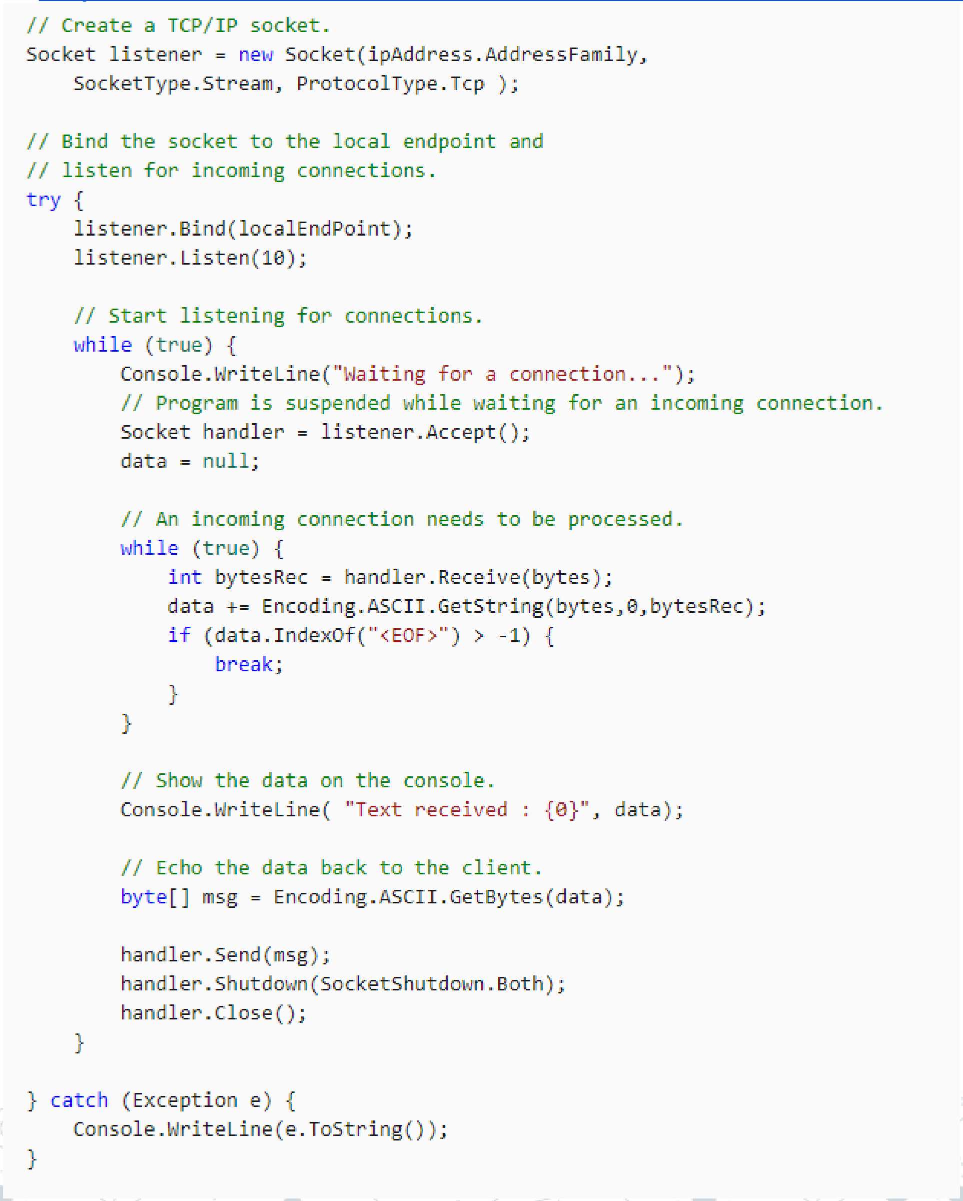
Di cosa si occupa il backend (o i backend)

- Rispondere a richieste da parte dei client su protocollo http/https/http2
- Interpretare le URL richieste/header/cookie
- Autenticare un utente
- Autorizzare un utente dopo la sua autenticazione
- Servire contenuti statici
- Generare pagine dinamiche
- Rispondere a chiamate REST da una SPA
- Gestire cache
- Servire contenuti in streaming
- ……

Come fa il backend a rispondere alle richieste?

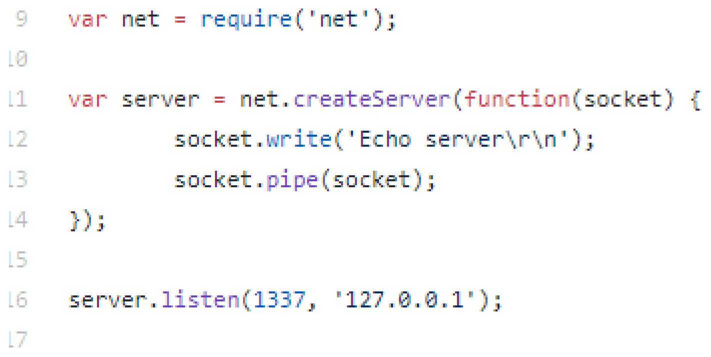
Semplicemente utilizzando i socket ed i metodi di listen

https://docs.microsoft.com/it-it/dotnet/framework/network-programming/synchronous-server-socket-example



```
// Create a TCP/IP socket.
Socket listener = new Socket(ipAddress.AddressFamily,
    SocketType.Stream, ProtocolType.Tcp );

// Bind the socket to the local endpoint and
// listen for incoming connections.
try {
    listener.Bind(localEndPoint);
    listener.Listen(10);

    // Start listening for connections.
    while (true) {
        Console.WriteLine("Waiting for a connection...");
        // Program is suspended while waiting for an incoming connection.
        Socket handler = listener.Accept();
        data = null;

        // An incoming connection needs to be processed.
        while (true) {
            int bytesRec = handler.Receive(bytes);
            data += Encoding.ASCII.GetString(bytes,0,bytesRec);
            if (data.IndexOf("<EOF>") > -1) {
                break;
            }
        }

        // Show the data on the console.
        Console.WriteLine( "Text received : {0}", data);

        // Echo the data back to the client.
        byte[] msg = Encoding.ASCII.GetBytes(data);

        handler.Send(msg);
        handler.Shutdown(SocketShutdown.Both);
        handler.Close();
    }

} catch (Exception e) {
    Console.WriteLine(e.ToString());
}
```

https://gist.github.com/tedmiston/5935757



```
var net = require('net');

var server = net.createServer(function(socket) {
        socket.write('Echo server\r\n');
        socket.pipe(socket);
});

server.listen(1337, '127.0.0.1');
```

Ma devo implementarmi il protocollo HTTP?