

Javascript Runtime Engine

```
// Follow the thread of execution
function outer() {
  // timer is set to 10 milliseconds, once resolved the console.log is stored in the callback queue, then call stack
  // logs last, since it arrived last in the call back queue
  setTimeout(() => {
    console.log("Hello world")
  }, 10);

  function inner() {
    // Although the timer is zero, this log occurs after the statement below due to the event loop
    // 3) logs third, since it needed to wait in the call back queue
    setTimeout(() => {
      console.log("Are you listening?")
    }, 0);
    // 1) logs first since it heads straight to the call stack
    console.log("Yes, I'm listening");
  }

  inner();
  // 2) goes to the call stack then resolves
  console.log("I like turtles")
}

outer();

// console
// Yes, I'm listening
// I like turtles
// Are you listening?
// Hello world
```

JavaScript

Heap

global

this

outer

Call Stack

WebAPI

Callback queue

JavaScript

Heap

global

this

outer

Call Stack

outer()

inner

WebAPI

Callback queue

JavaScript

Heap

global

this

outer

Call Stack

outer()

inner

WebAPI

```
setTimeout(() => {  
  console.log("Hello World")  
}, 10)
```

Callback queue

JavaScript

Heap

global

this

outer

Call Stack

inner()

outer()

inner

WebAPI

```
setTimeout(() => {  
  console.log("Hello World")  
}, 10)
```

Callback queue

JavaScript

Heap

global

this

outer

Call Stack

inner()

outer()

inner

WebAPI

```
setTimeout(() => {  
  console.log("Hello World")  
}, 10)
```

```
setTimeout(() => {  
  console.log("Are you listening")  
}, 0)
```

Callback queue

JavaScript

Heap

global

this

outer

Call Stack

inner()

outer()

inner

WebAPI

```
setTimeout(() => {  
  console.log("Hello World")  
}, 10)
```

Callback queue

```
() => {  
  console.log("Are you listening")  
};
```


JavaScript

Heap

global

this

outer

Call Stack

inner()

outer()

inner

WebAPI

```
setTimeout(() => {  
  console.log("Hello World")  
}, 10)
```

```
console.log("Yes, I'm  
listening");
```

Callback queue

```
() => {  
  console.log("Are you listening")  
};
```

JavaScript

Heap

global

this

outer

Call Stack

outer()

inner

WebAPI

```
setTimeout(() => {  
  console.log("Hello World")  
}, 10)
```

```
console.log("I like  
turtles.");
```

Callback queue

```
() => {  
  console.log("Are you listening")  
};
```

JavaScript

Heap

global

this

outer

Call Stack

outer()

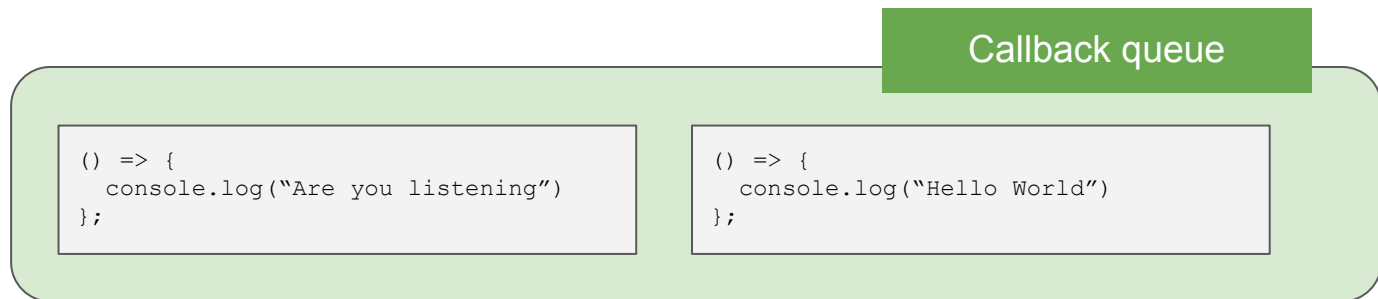
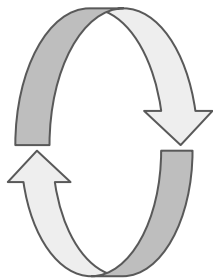
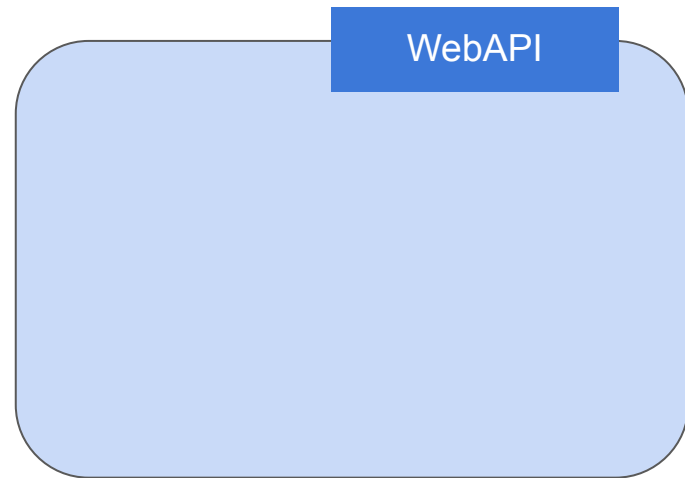
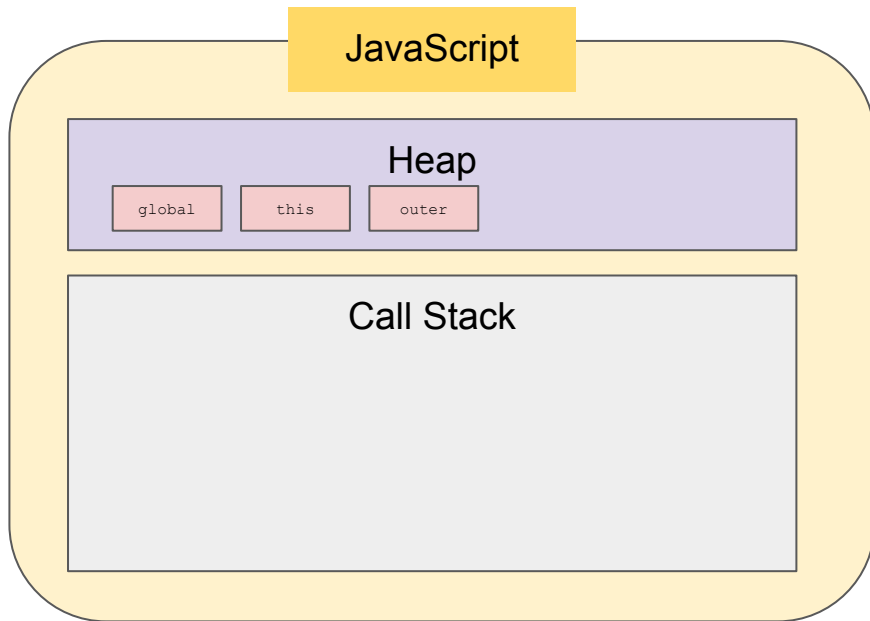
inner

WebAPI

Callback queue

```
() => {  
  console.log("Are you listening")  
};
```

```
() => {  
  console.log("Hello World")  
};
```



JavaScript

Heap

global

this

outer

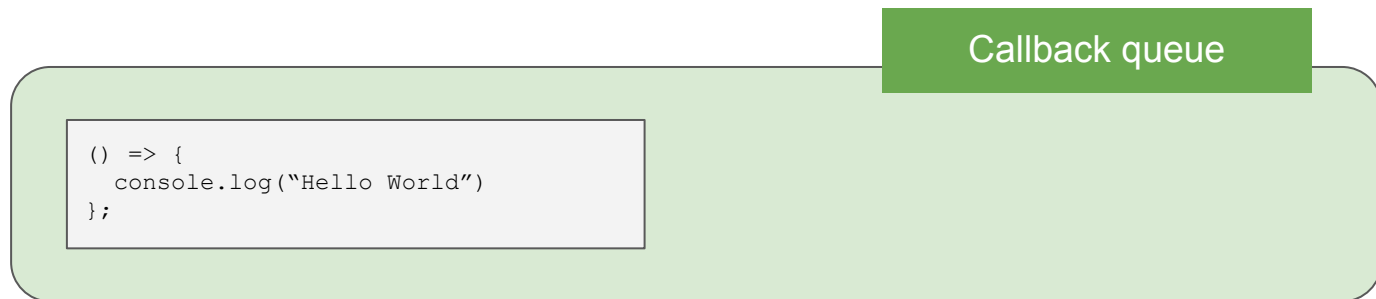
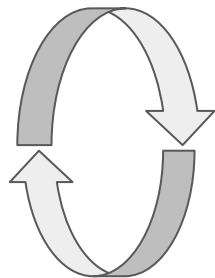
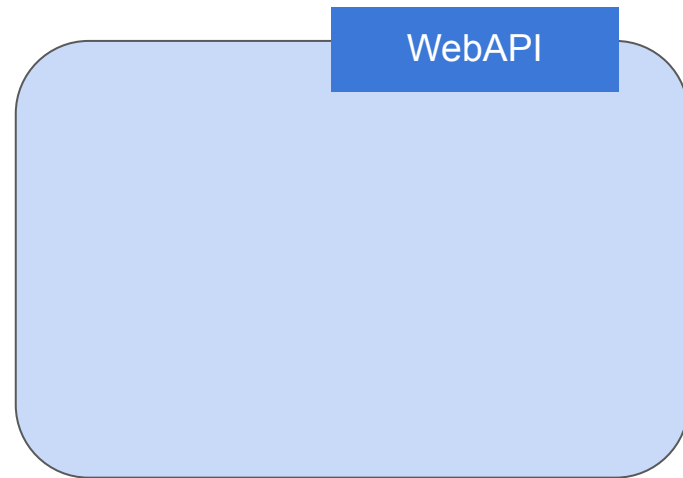
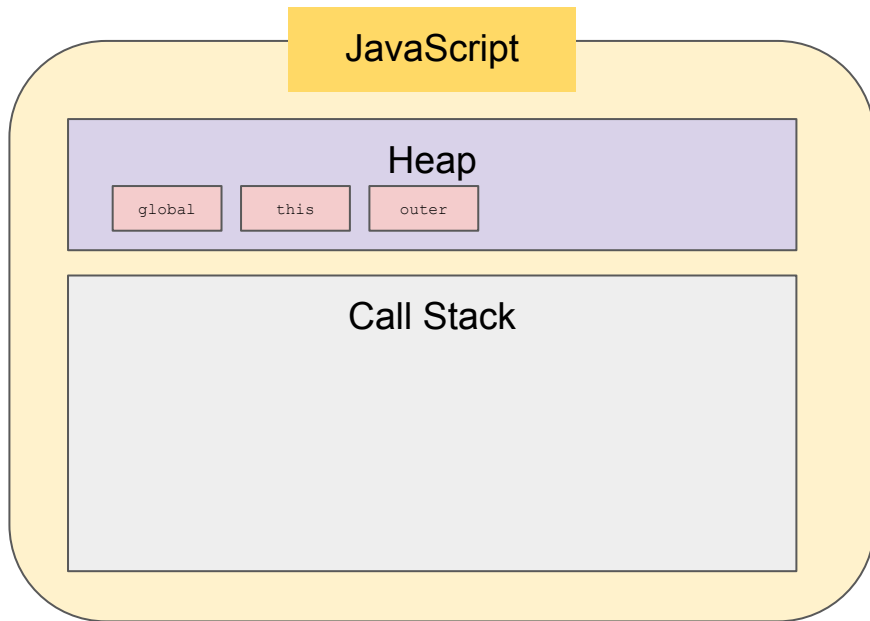
Call Stack

```
() => {  
  console.log("Are you listening")  
};
```

WebAPI

Callback queue

```
() => {  
  console.log("Hello World")  
};
```



JavaScript

Heap

global

this

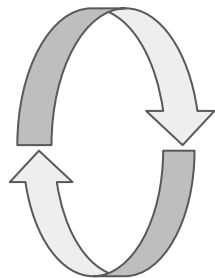
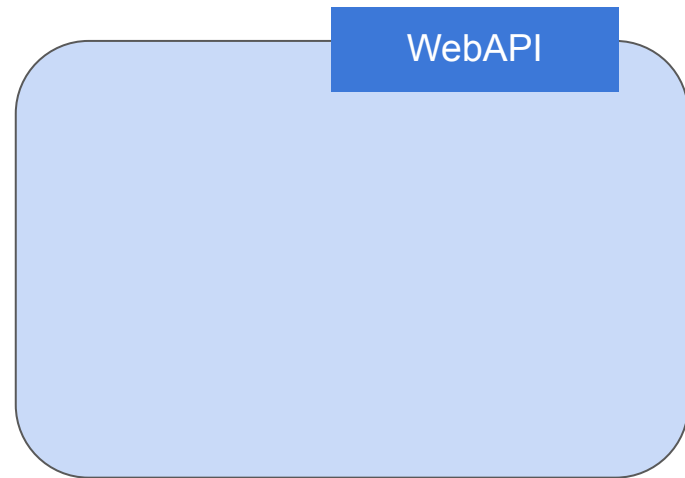
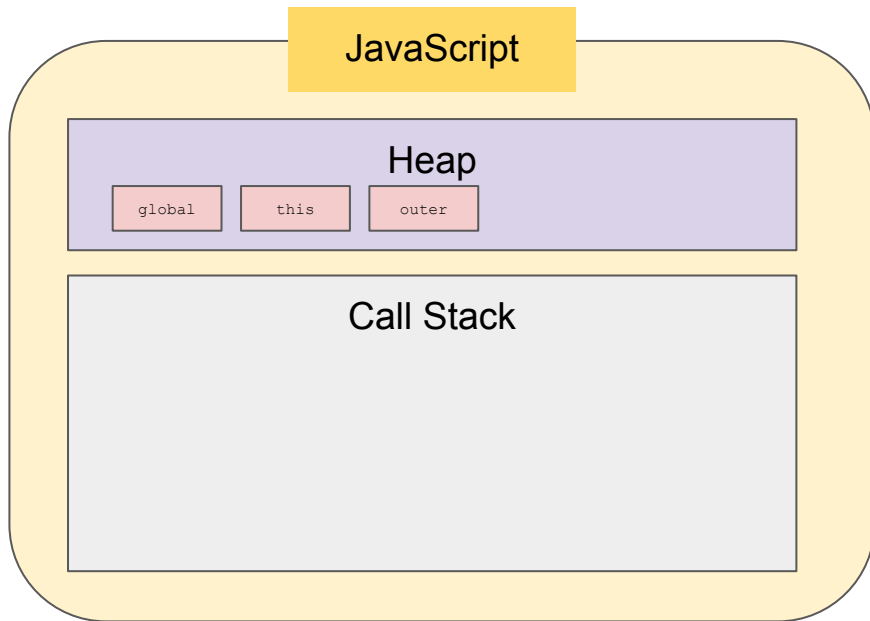
outer

Call Stack

```
() => {  
  console.log("Hello World")  
};
```

WebAPI

Callback queue



JavaScript



The diagram shows a yellow rounded rectangle representing the JavaScript runtime. Inside it are two stacked rectangles: a purple one labeled 'Heap' and a light gray one labeled 'Call Stack'.

Heap

Call Stack

WebAPI



The diagram shows a blue rounded rectangle representing the WebAPI component.

Callback queue



The diagram shows a green rounded rectangle representing the Callback queue component.