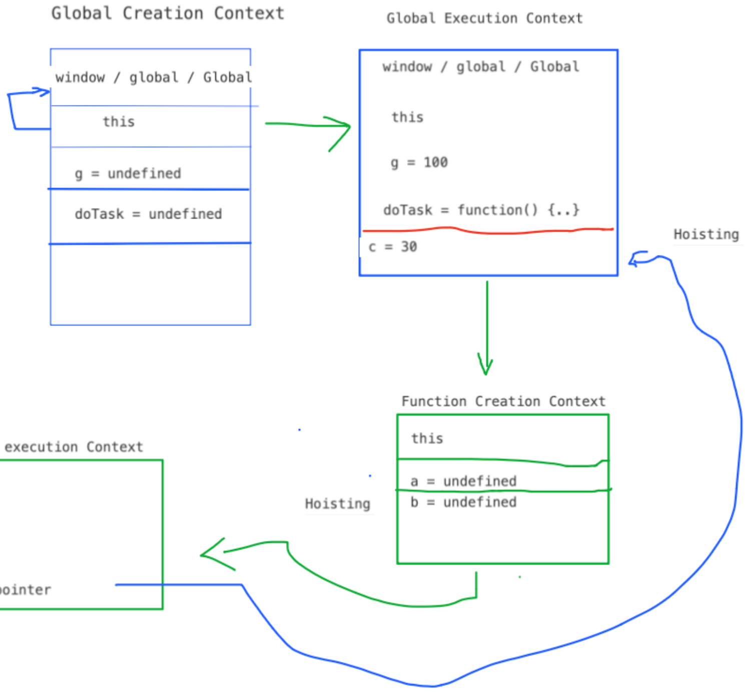
```
var g = 100;

function doTask() {
    var a = 10;
    if( g > a) {
        var b = 20;
        c = 30;
    }
    console.log(g, a, b, c);
}

doTask();
console.log(g, a, b, c);
```



Global Creation Context

window / global / Global

this

g = undefined

doTask = undefined

Global Execution Context

window / global / Global

this

g = 100

doTask = function() {..}

c = 30

Hoisting

Function Creation Context

this

a = undefined

b = undefined

Hoisting

Function execution Context

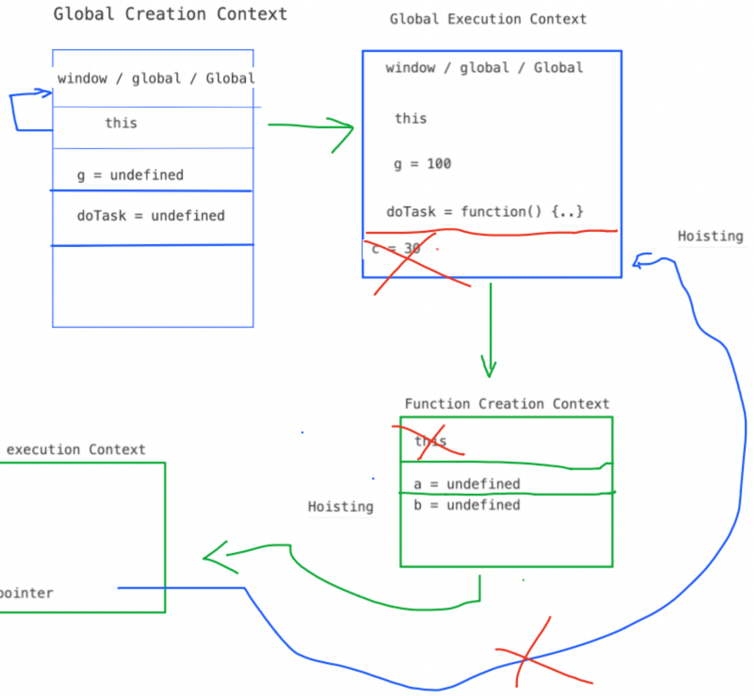
this

a = 10

b = 20

scope pointer

```
"use strict"

var g = 100;

function doTask() {
    var a = 10;
    if( g > a) {
        var b = 20;
        c = 30;
    }
    console.log(g, a, b, c);
}

doTask();
console.log(g, a, b, c);
```



Global Creation Context

window / global / Global

this

g = undefined

doTask = undefined

Global Execution Context

window / global / Global

this

g = 100

doTask = function() {..}

~~c = 30~~

Hoisting

Function Creation Context

~~this~~

a = undefined

b = undefined

Hoisting

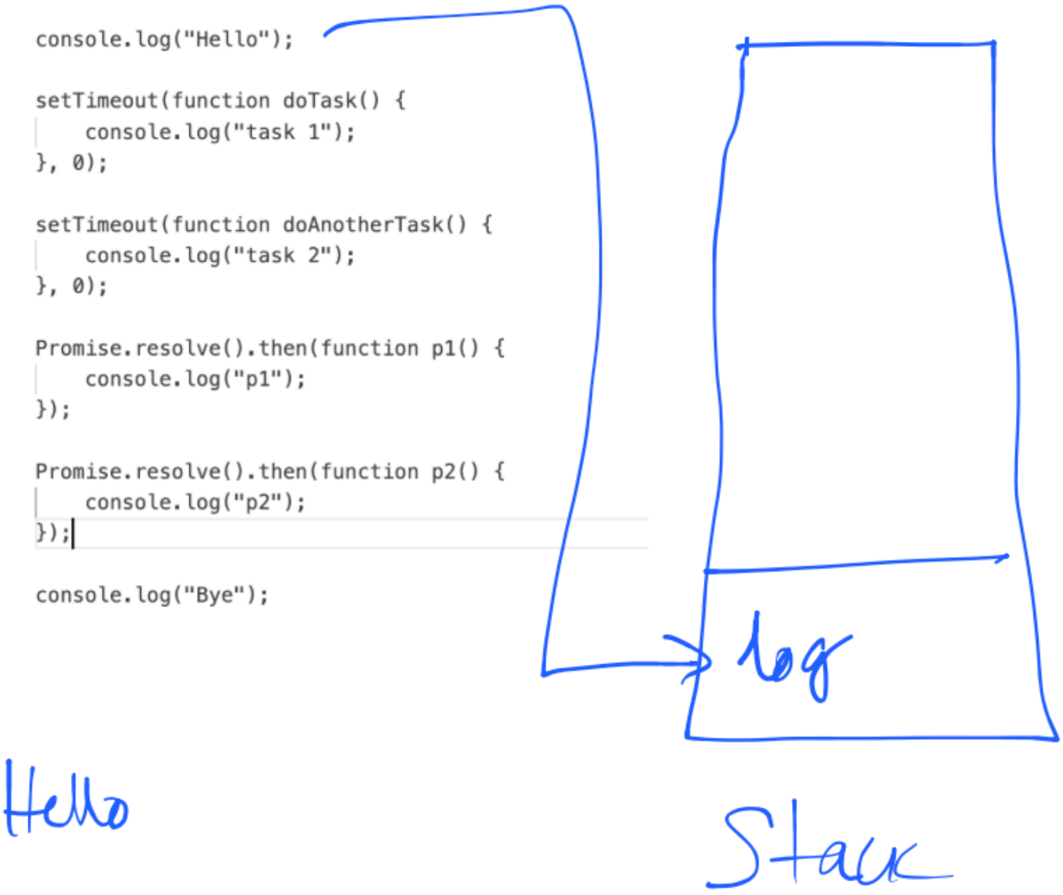
Function execution Context

this

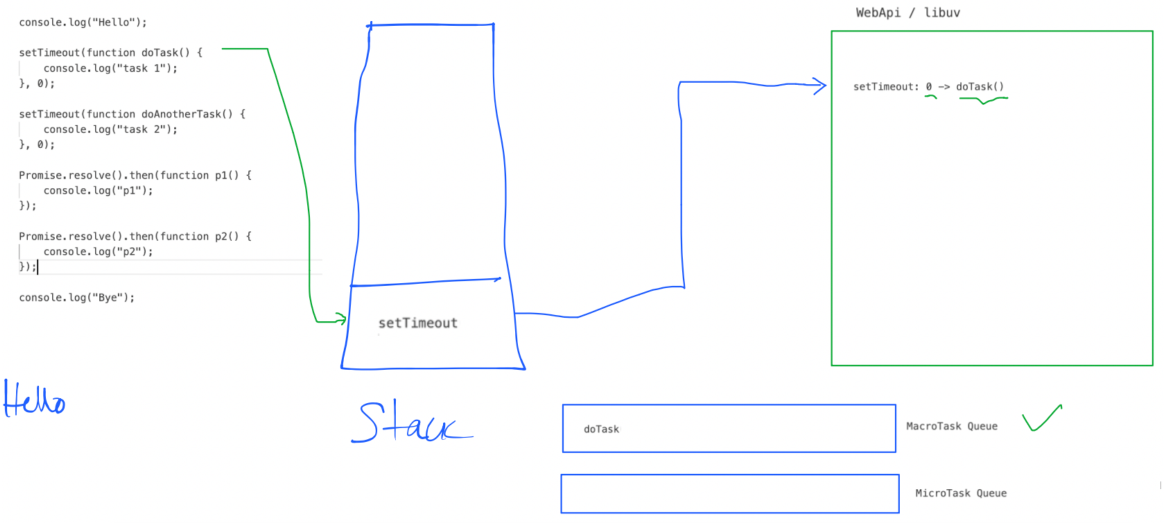
a = 10

b = 20

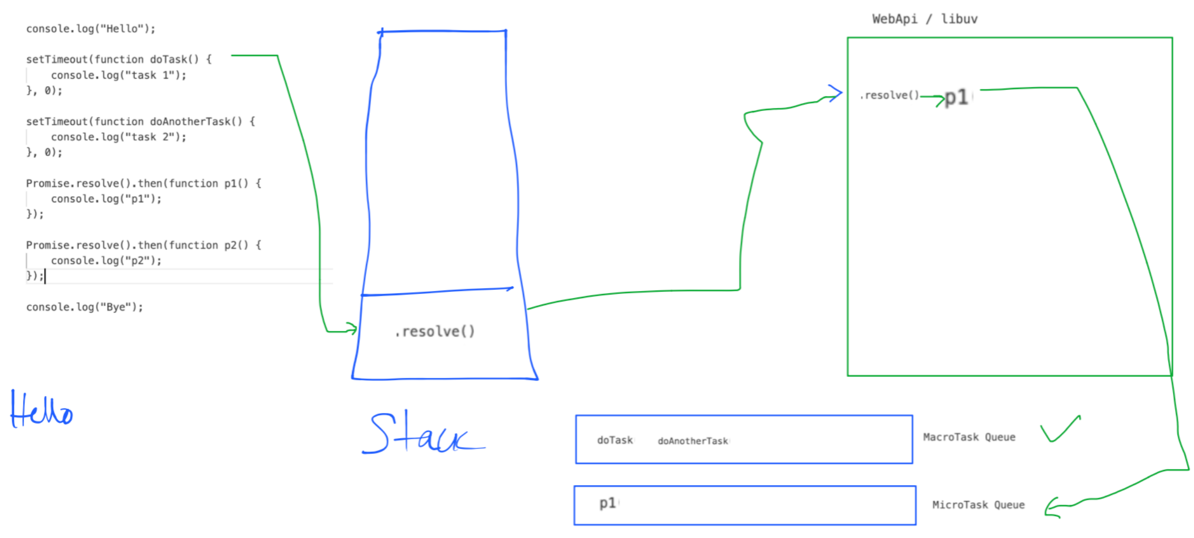
scope pointer



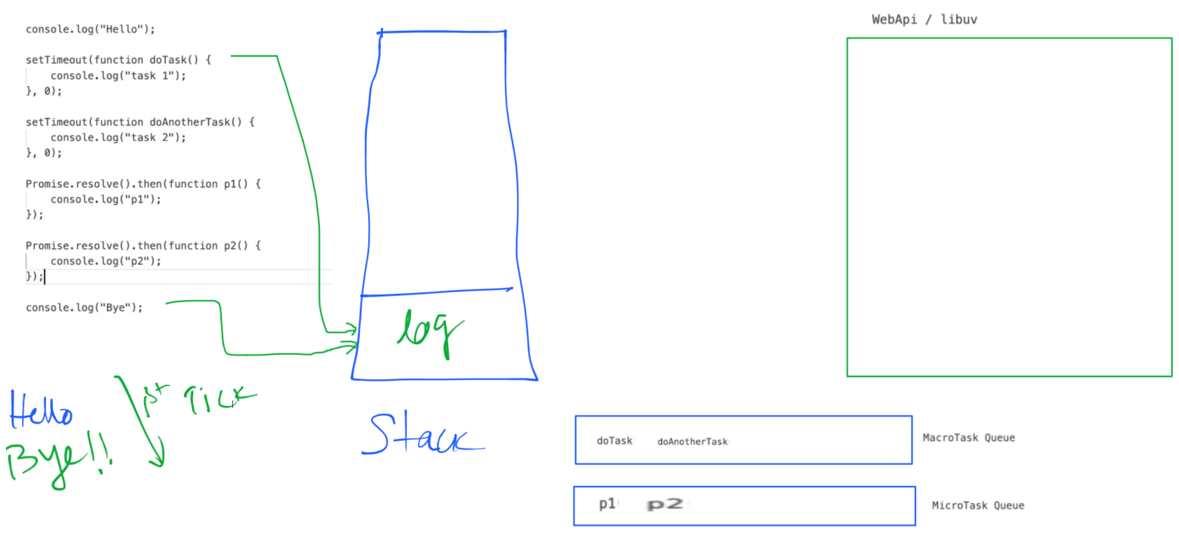
```
console.log("Hello");

setTimeout(function doTask() {
    console.log("task 1");
}, 0);

setTimeout(function doAnotherTask() {
    console.log("task 2");
}, 0);

Promise.resolve().then(function p1() {
    console.log("p1");
});

Promise.resolve().then(function p2() {
    console.log("p2");
});

console.log("Bye");
```



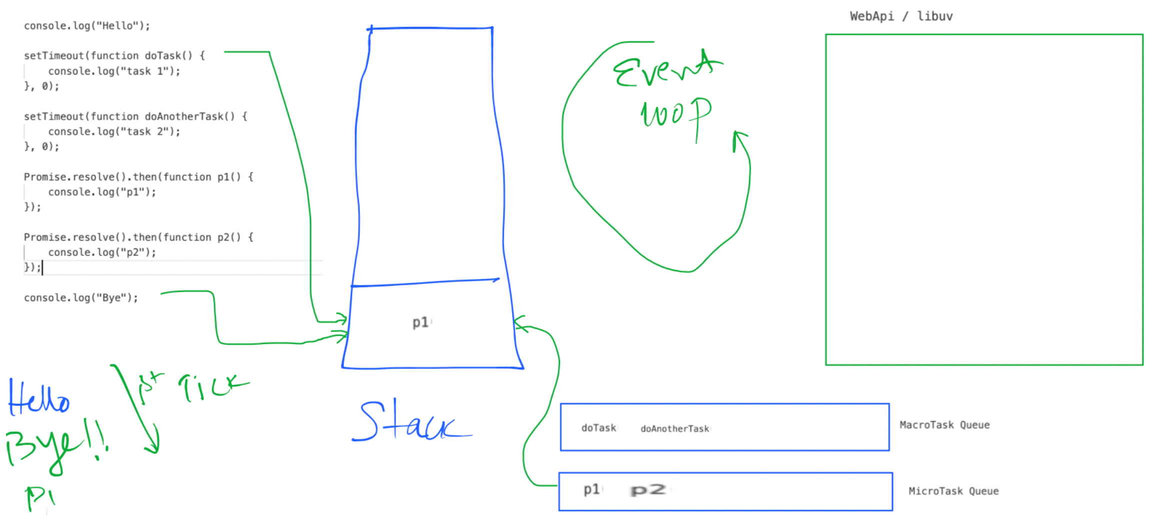
```
console.log("Hello");

setTimeout(function doTask() {
    console.log("task 1");
}, 0);

setTimeout(function doAnotherTask() {
    console.log("task 2");
}, 0);

Promise.resolve().then(function p1() {
    console.log("p1");
});

Promise.resolve().then(function p2() {
    console.log("p2");
});

console.log("Bye");
```



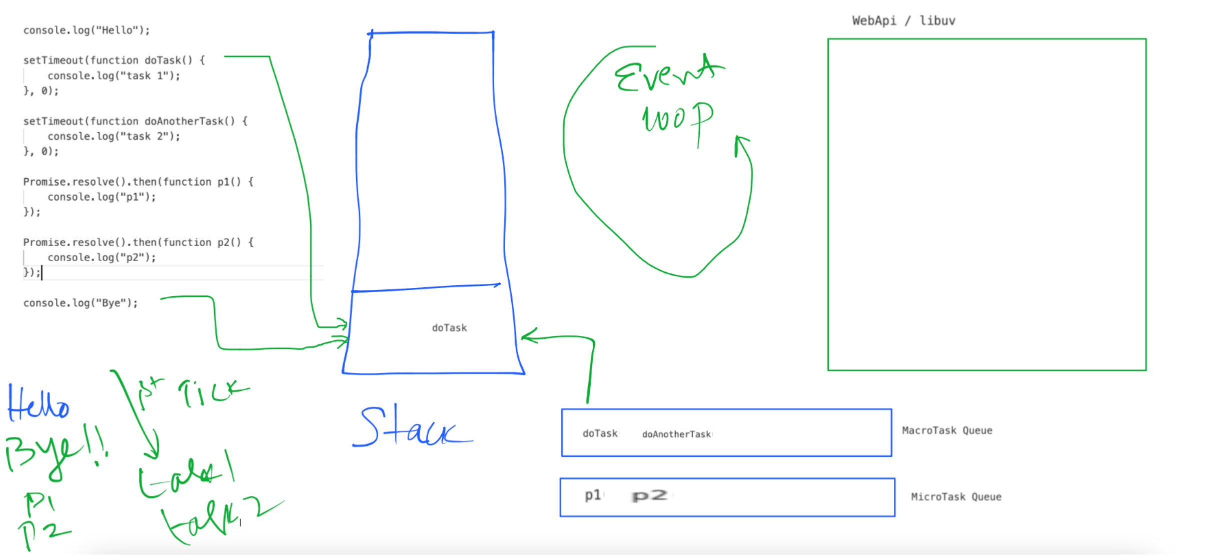
```
console.log("Hello");

setTimeout(function doTask() {
    console.log("task 1");
}, 0);

setTimeout(function doAnotherTask() {
    console.log("task 2");
}, 0);

Promise.resolve().then(function p1() {
    console.log("p1");
});

Promise.resolve().then(function p2() {
    console.log("p2");
});

console.log("Bye");
```



```
console.log("Hello");

setTimeout(function doTask() {
    console.log("task 1");
}, 0);

setTimeout(function doAnotherTask() {
    console.log("task 2");
}, 0);

Promise.resolve().then(function p1() {
    console.log("p1");
});

Promise.resolve().then(function p2() {
    console.log("p2");
});

console.log("Bye");
```

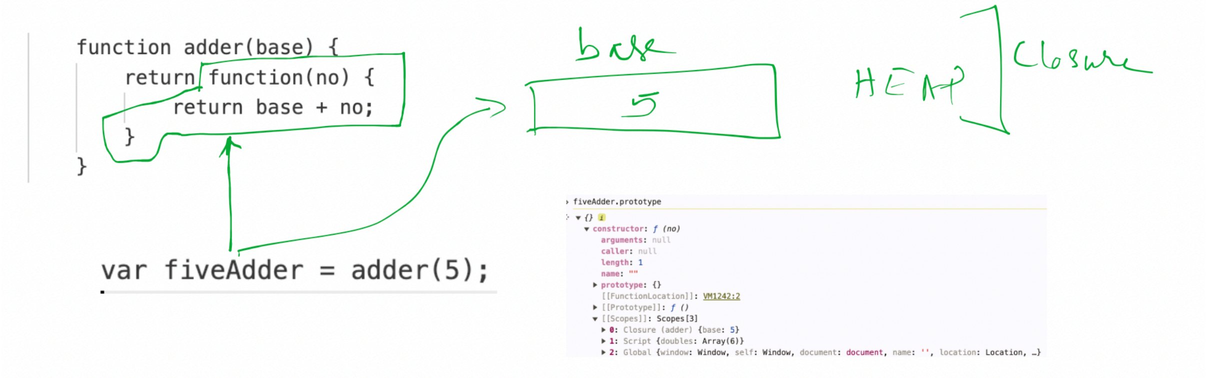


```
console.log("Hello");

setTimeout(function doTask() {
    console.log("task 1");
}, 0);

setTimeout(function doAnotherTask() {
    console.log("task 2");
}, 0);

Promise.resolve().then(function p1() {
    console.log("p1");
});

Promise.resolve().then(function p2() {
    console.log("p2");
});

console.log("Bye");
```



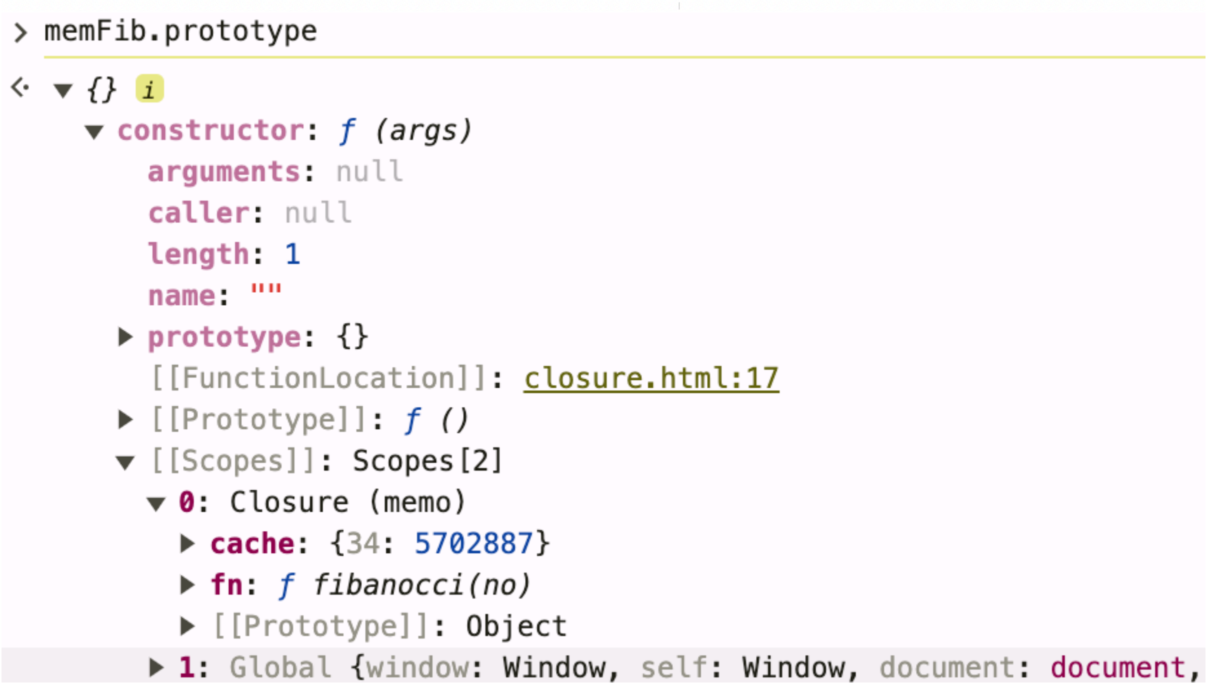
```
console.log("Hello");

setTimeout(function doTask() {
    console.log("task 1");
}, 0);

setTimeout(function doAnotherTask() {
    console.log("task 2");
}, 0);

Promise.resolve().then(function p1() {
    console.log("p1");
});

Promise.resolve().then(function p2() {
    console.log("p2");
});

console.log("Bye");
```

Closure:

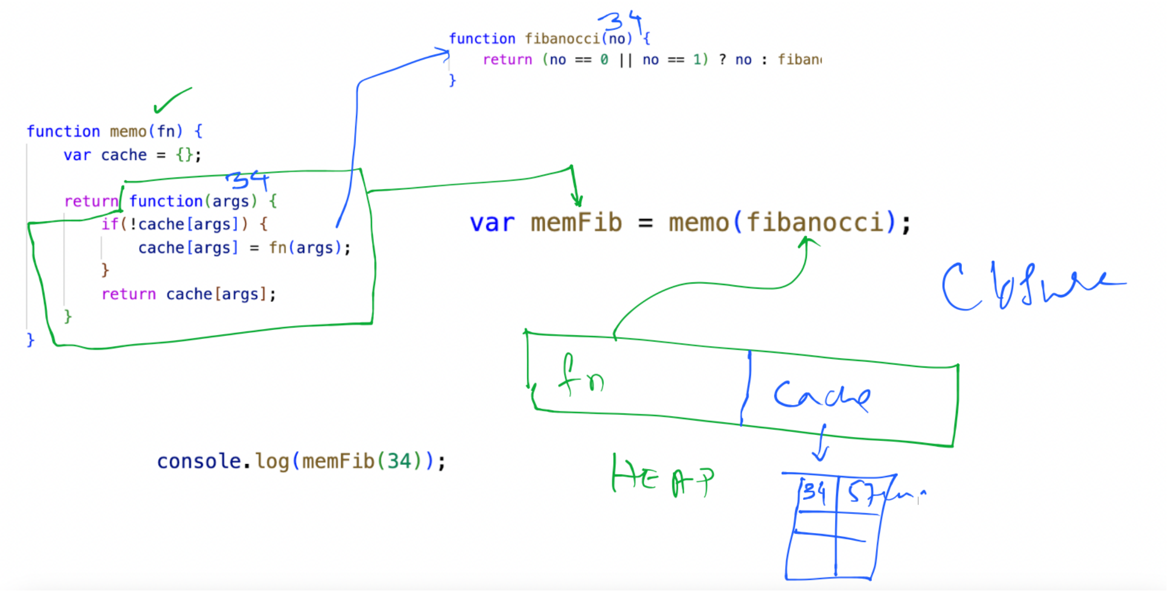


```
function adder(base) {
    return function(no) {
        return base + no;
    }
}

var fiveAdder = adder(5);
```

```
› fiveAdder.prototype
‹ ▼ {} i
    ▼ constructor: ƒ (no)
        arguments: null
        caller: null
        length: 1
        name: ""
      ▶ prototype: {}
        [[FunctionLocation]]: VM1242:2
      ▶ [[Prototype]]: ƒ ()
      ▼ [[Scopes]]: Scopes[3]
        ▶ 0: Closure (adder) {base: 5}
        ▶ 1: Script {doubles: Array(6)}
        ▶ 2: Global {window: Window, self: Window, document: document, name: '', location: Location, …}
```



```
> memFib.prototype
<· ▼ {} i
     ▼ constructor: ƒ (args)
         arguments: null
         caller: null
         length: 1
         name: ""
       ▶ prototype: {}
         [[FunctionLocation]]: closure.html:17
       ▶ [[Prototype]]: ƒ ()
       ▼ [[Scopes]]: Scopes[2]
         ▼ 0: Closure (memo)
           ▶ cache: {34: 5702887}
           ▶ fn: ƒ fibanocci(no)
           ▶ [[Prototype]]: Object
         ▶ 1: Global {window: Window, self: Window, document: document,
```



```
function fibanocci(no) {
    return (no == 0 || no == 1) ? no : fiban
}
```

```
function memo(fn) {
    var cache = {};

    return function(args) {
        if(!cache[args]) {
            cache[args] = fn(args);
        }
        return cache[args];
    }
}
```

```
var memFib = memo(fibanocci);
```

```
console.log(memFib(34));
```

34	57