

JSDN:

# Closures

# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

var counter  $\longrightarrow$  0

# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

a. Assignment

var counter  $\longrightarrow$  0

# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

- a. Assignment
  - a. Evaluate right side

var counter  $\longrightarrow$  0

# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

- a. Assignment
  - a. Evaluate right side
    - a. Create function

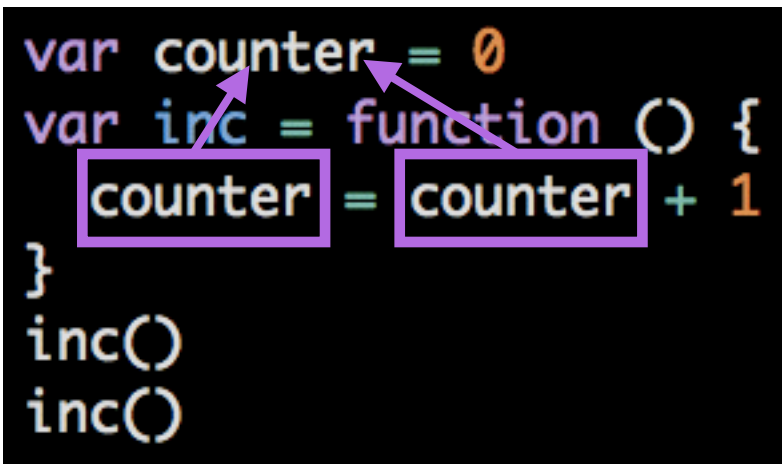
var counter  $\longrightarrow$  0

function()

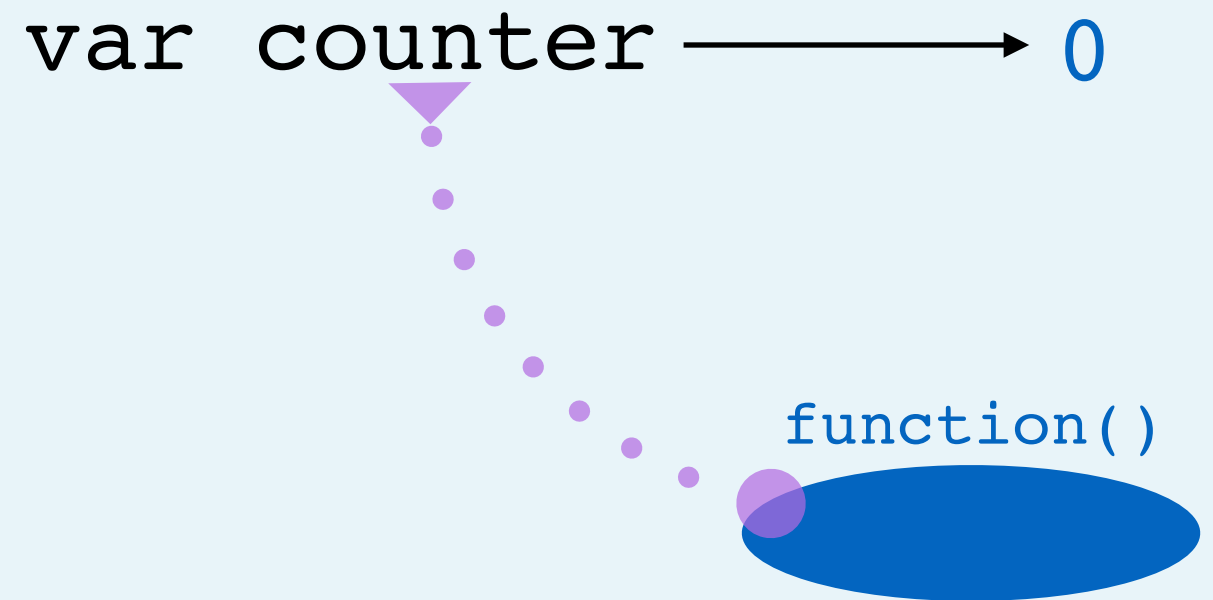


# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```



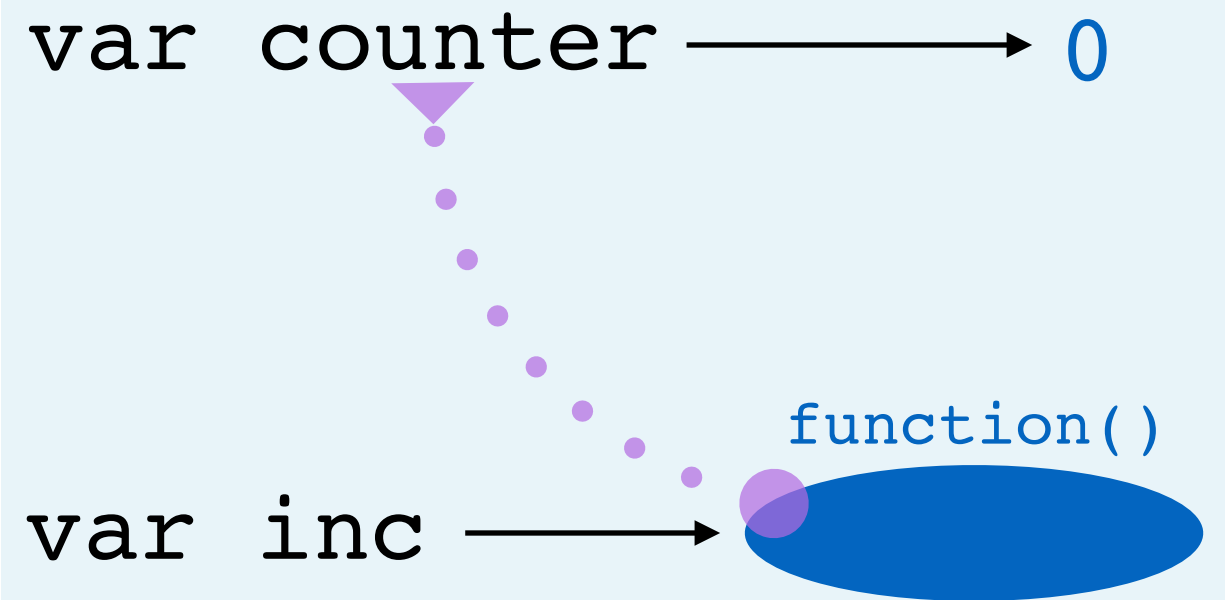
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (var counter)



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

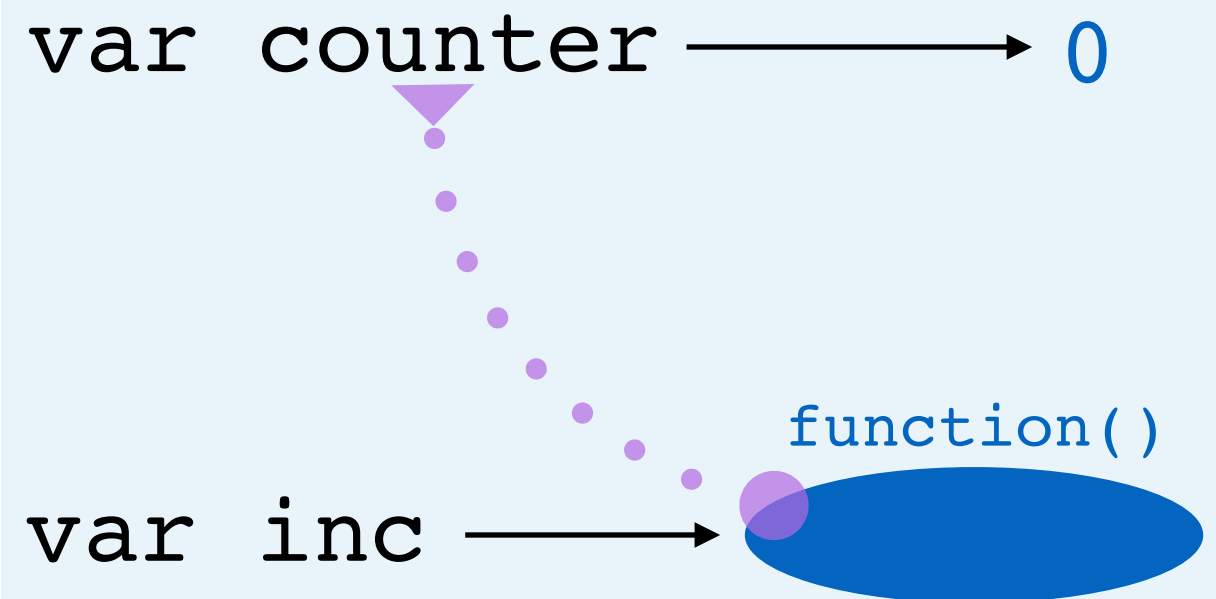
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call

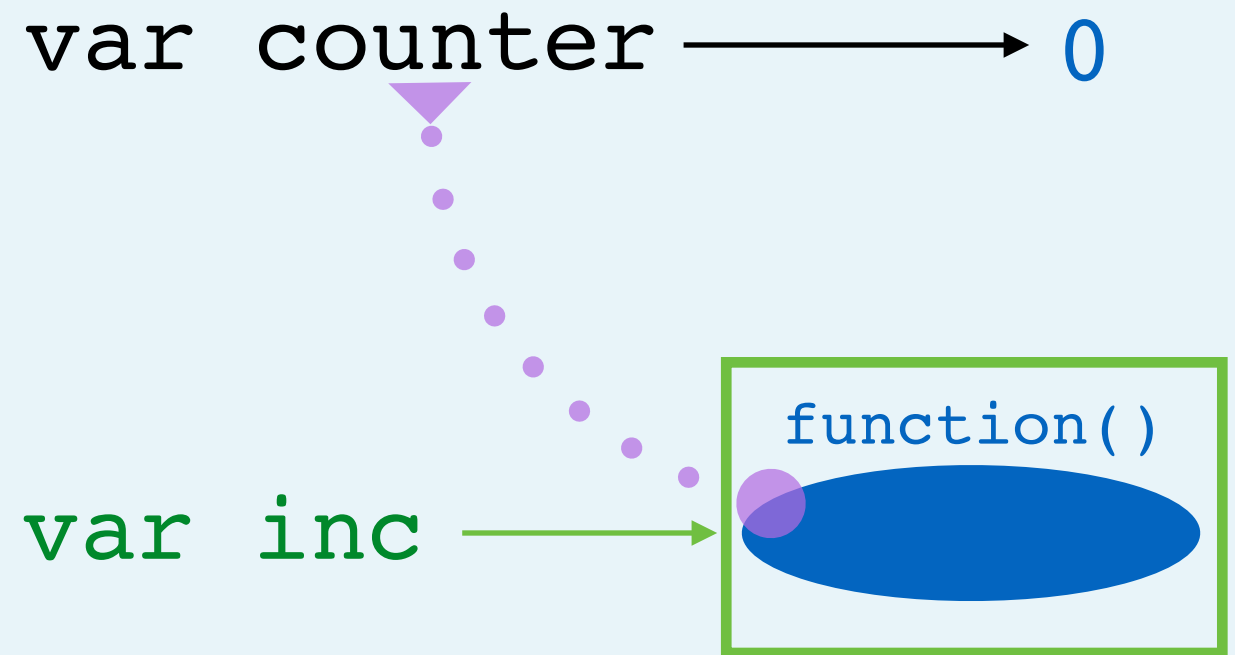




# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

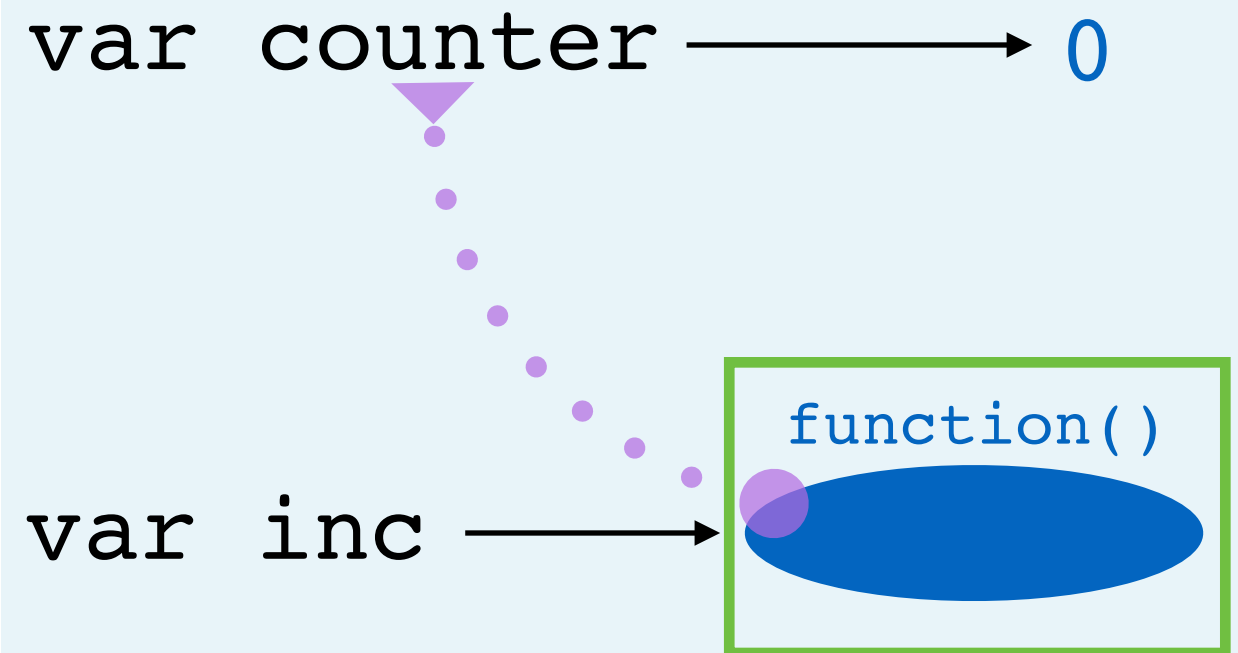
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

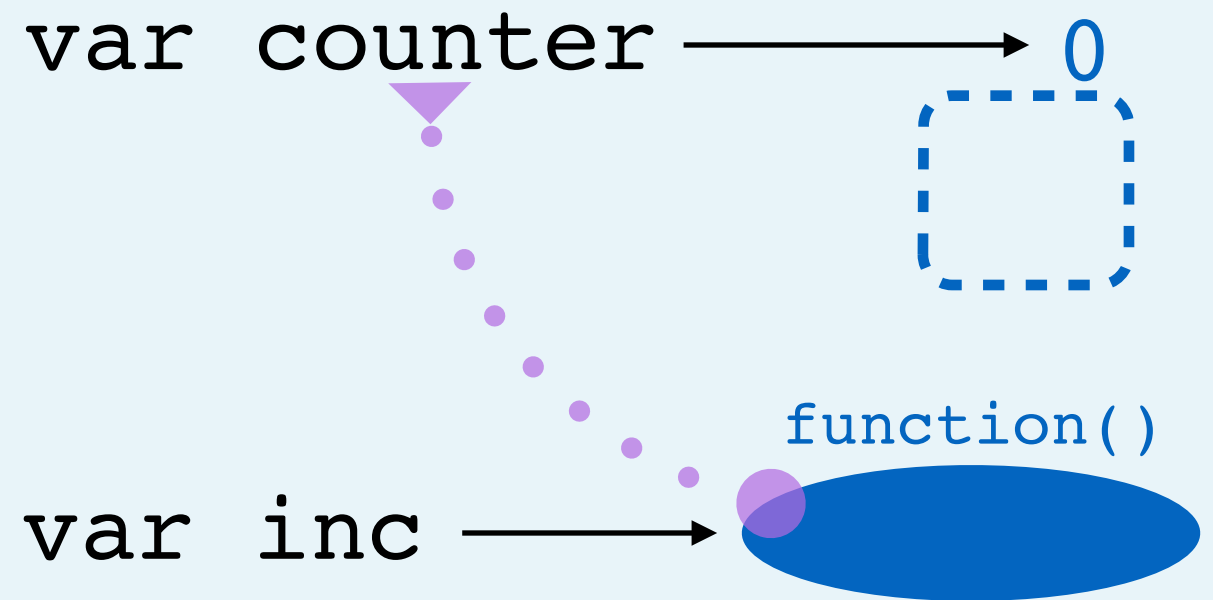
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)
    - b. Call function



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

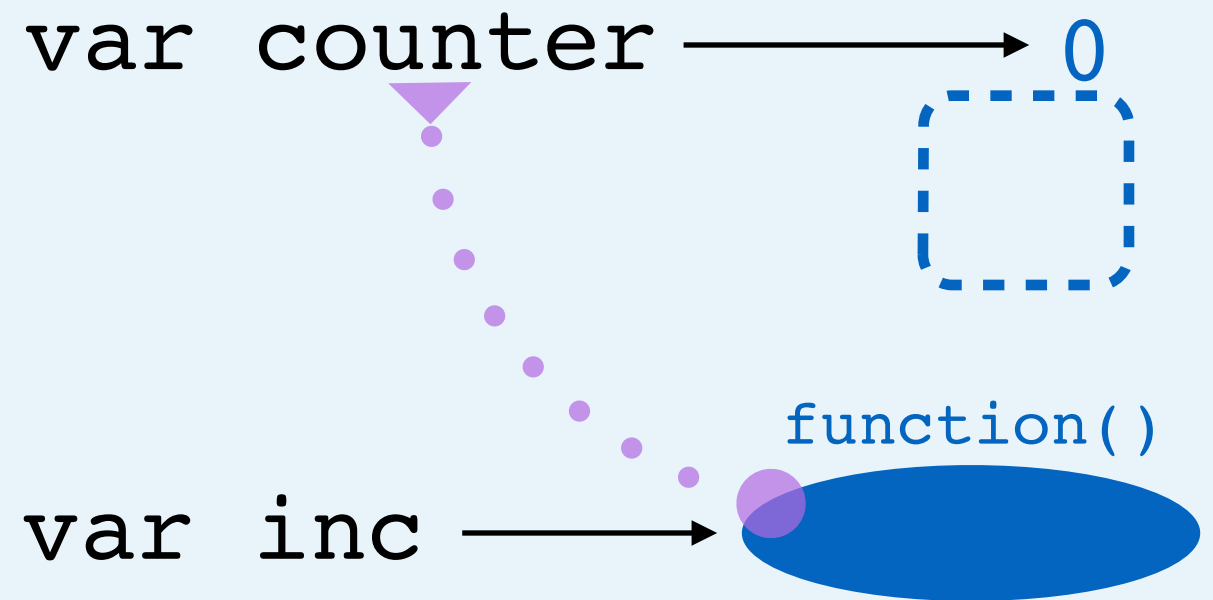
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)
    - b. Call function
      - a. Create scope



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

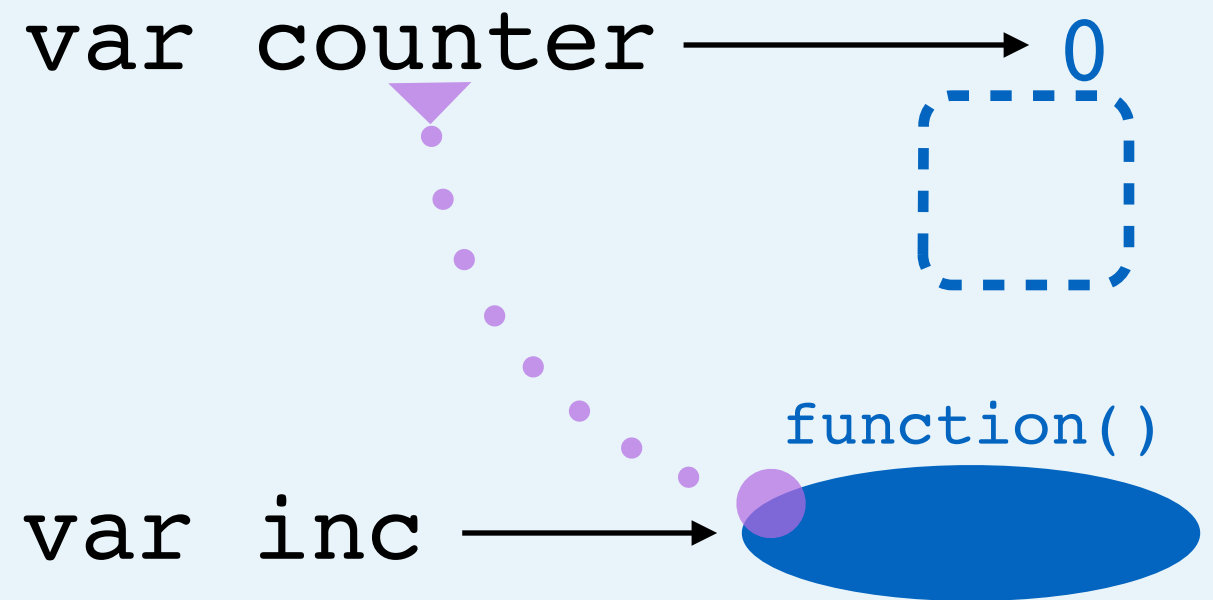
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)
    - b. Call function
      - a. Create scope
      - b. **Re**assignment



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

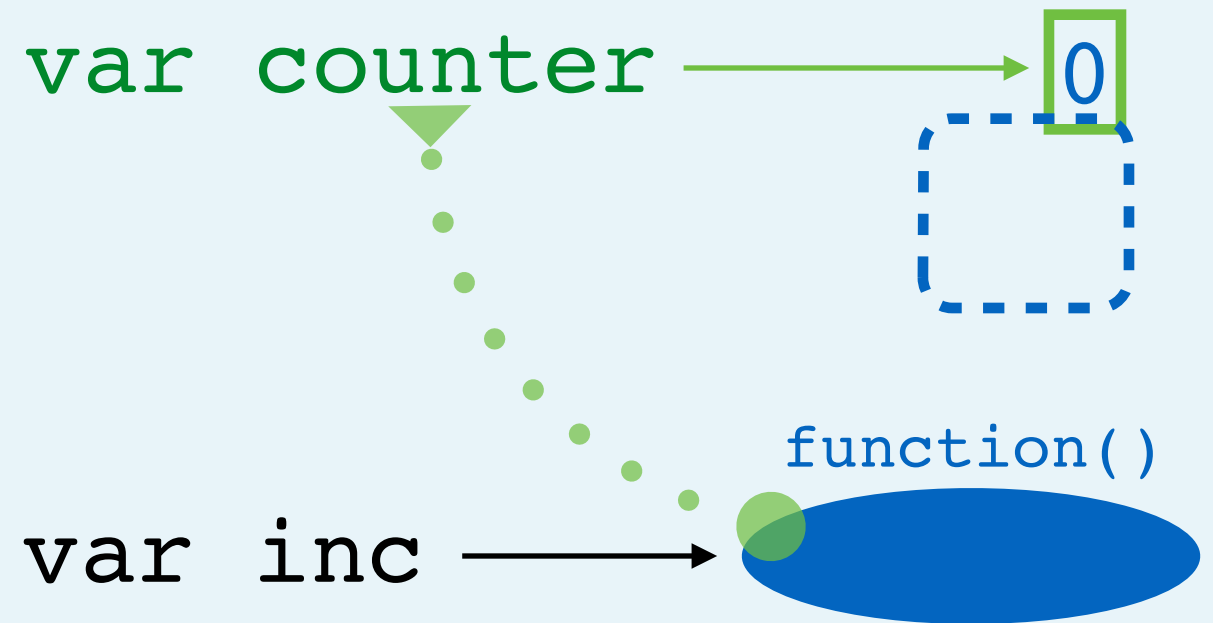
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)
    - b. Call function
      - a. Create scope
      - b. **Re**assignment
        - a. Binary Operation (addition)



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

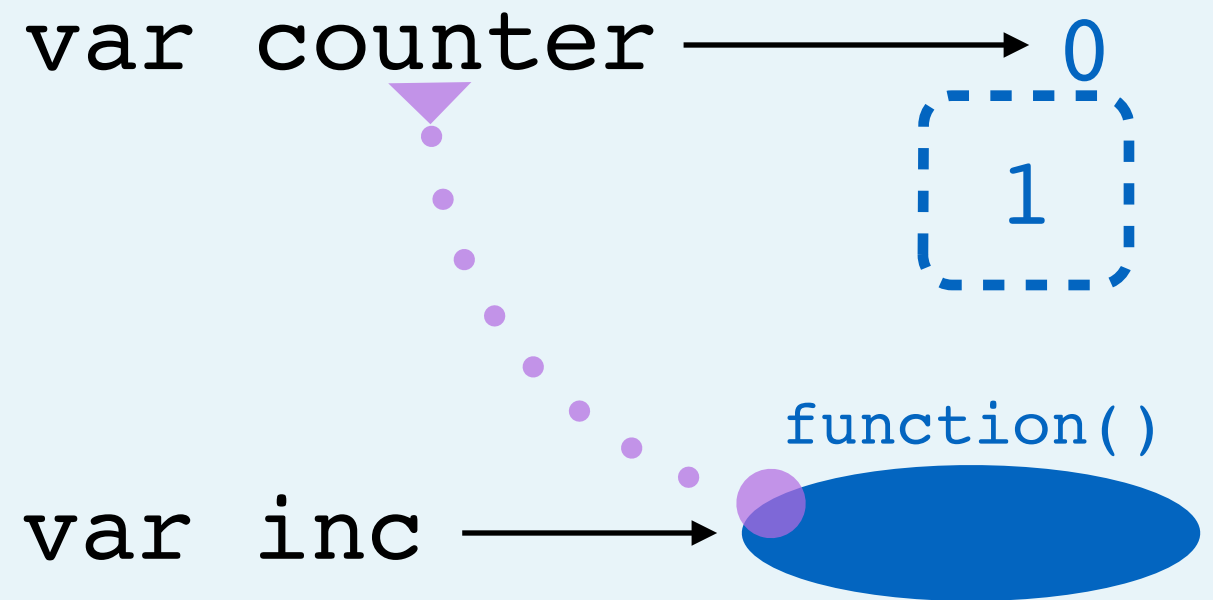
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)
    - b. Call function
      - a. Create scope
      - b. **Re**assignment
        - a. Binary Operation (addition)
          - a. Look up value of counter



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

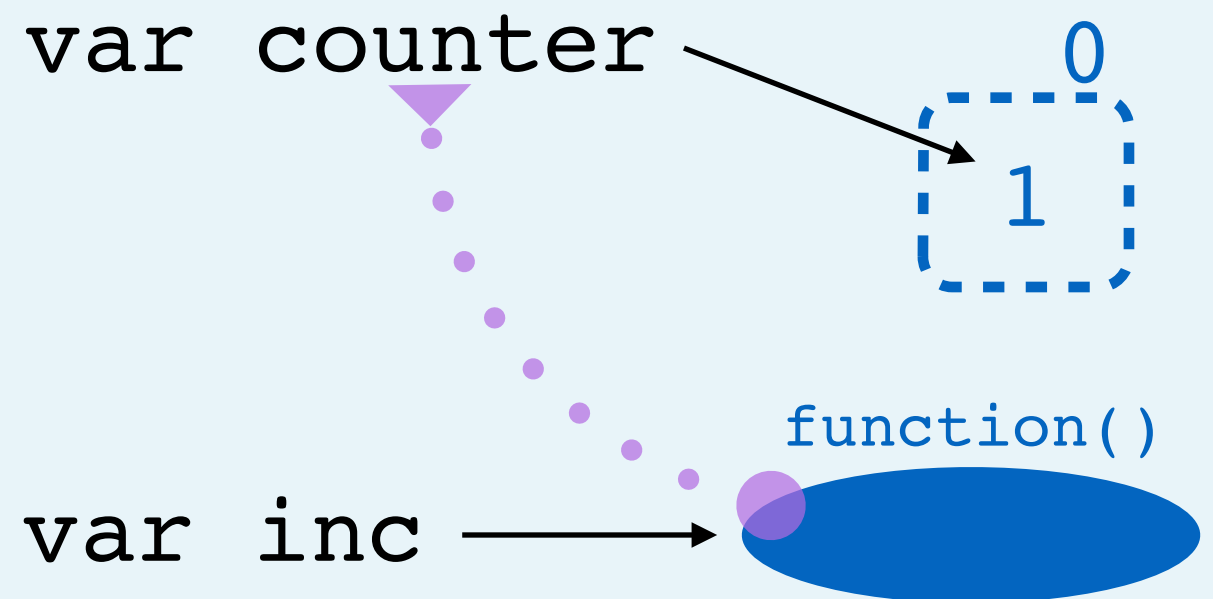
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)
    - b. Call function
      - a. Create scope
      - b. **Re**assignment
        - a. Binary Operation (addition)
          - a. Look up value of counter
          - b. Create value



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)
    - b. Call function
      - a. Create scope
      - b. **Re**assignment
        - a. Binary Operation (addition)
          - a. Look up value of counter
          - b. Create value
        - b. Set var to point to value

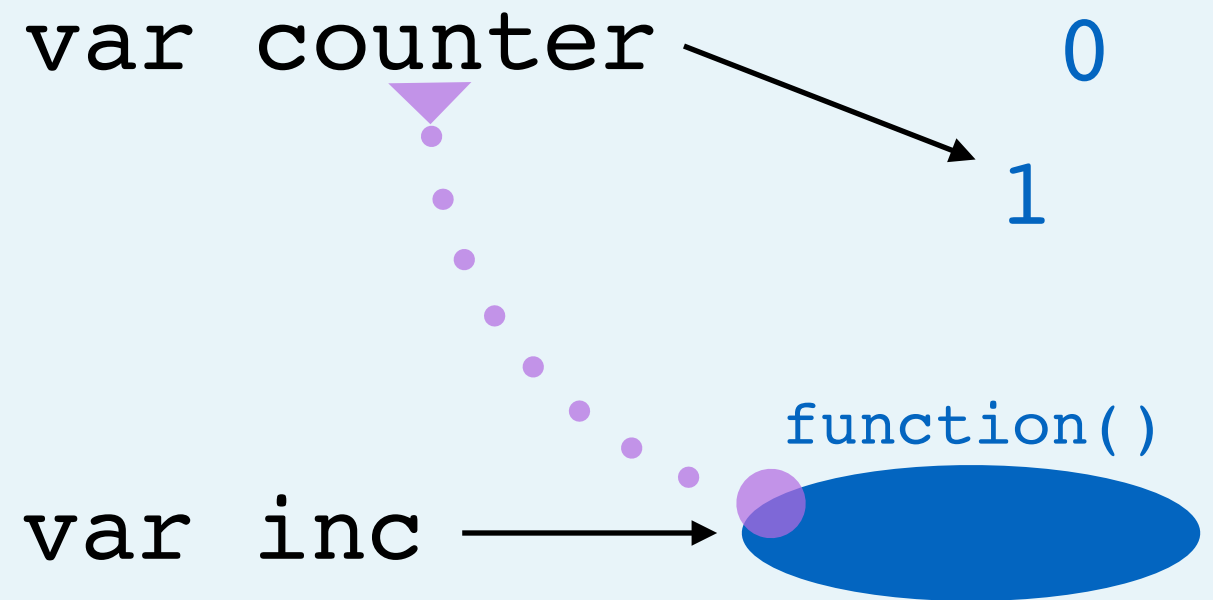




# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

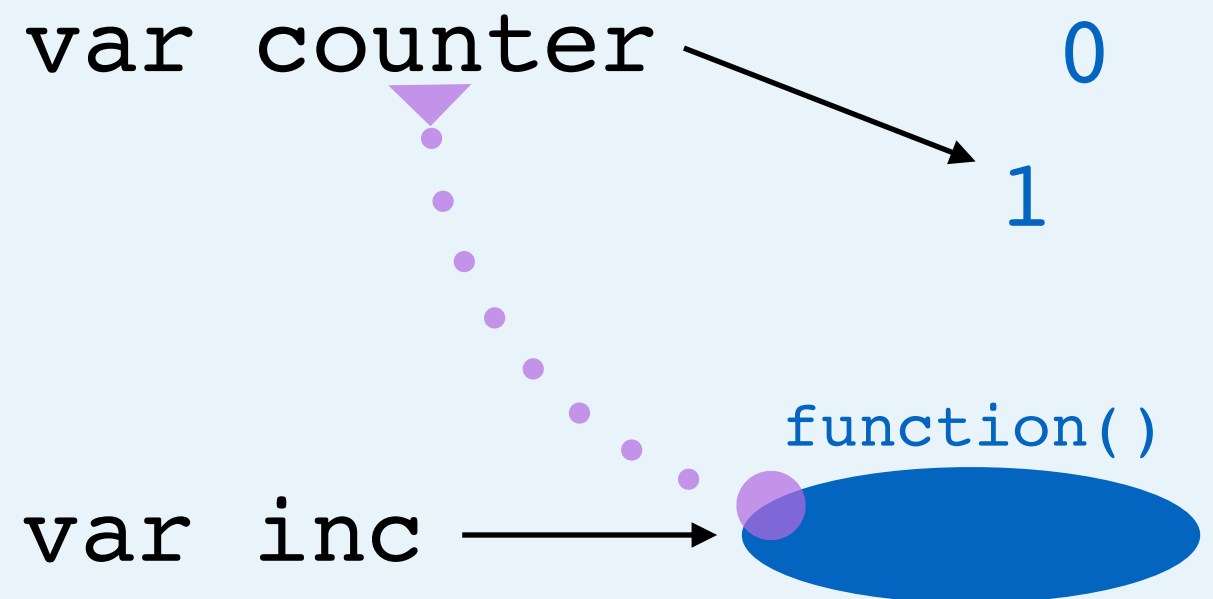
- a. Assignment
  - a. Evaluate right side
    - a. Create function
      - a. Create closure (counter)
    - b. Create var, point to value
  - b. Function call
    - a. Look up value of inc (it's a function)
    - b. Call function
      - a. Create scope
      - b. **Re**assignment
        - a. Binary Operation (addition)
          - a. Look up value of counter
          - b. Create value
        - b. Set var to point to value
      - c. Garbage collect scope



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

- a. Assignment
    - a. Evaluate right side
      - a. Create function
        - a. Create closure (counter)
      - b. Create var, point to value
    - b. Function call
      - a. Look up value of inc (it's a function)
      - b. Call function
        - a. Create scope
        - b. **Re**assignment
          - a. Binary Operation (addition)
            - a. Look up value of counter
            - b. Create value
          - b. Set var to point to value
        - c. Garbage collect scope
- c. Function call



# A Closure in Action

```
var counter = 0
var inc = function () {
  counter = counter + 1
}
inc()
inc()
```

- a. Assignment
    - a. Evaluate right side
      - a. Create function
        - a. Create closure (counter)
      - b. Create var, point to value
    - b. Function call
      - a. Look up value of inc (it's a function)
      - b. Call function
        - a. Create scope
        - b. **Re**assignment
          - a. Binary Operation (addition)
            - a. Look up value of counter
            - b. Create value
          - b. Set var to point to value
        - c. Garbage collect scope
- c. Function call
  - a. (Same as before; counter is incremented)

