

1. crearCuentaBancaria aún no tiene elementos y se genera con undefined.

JavaScript (ES6)

```
1 function crearCuentaBancaria(saldoinicial){
2   var saldo = saldoinicial;
3
4   function depositar (cantidad) {
5     if (cantidad > 0){
6       saldo += cantidad;
7     } else{
8       console.log ("La cantidad debe ser mayor a
9     }
10  }
11
12  function retirar (cantidad){
13    if (cantidad > 0 && cantidad <= saldo){
14      saldo -=cantidad;
15    } else {
16      console.log ("La cantidad a retirar debe se
17    }
18  }
19
20  return{
```

Print output (drag lower right corner to resize)

Global frame

- crearCuentaBancaria
- miCuenta
- undefined

Objects

```
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;

  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log ("La cantidad debe ser mayor a cero");
    }
  }

  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -=cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }

  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

Step 1 of 34

2.Saldo inicial se define como 1000.

JavaScript (ES6)

```
1 function crearCuentaBancaria(saldoinicial){
2   var saldo = saldoinicial;
3
4   function depositar (cantidad) {
5     if (cantidad > 0){
6       saldo += cantidad;
7     } else{
8       console.log ("La cantidad debe ser mayor a
9     }
10  }
11
12  function retirar (cantidad){
13    if (cantidad > 0 && cantidad <= saldo){
14      saldo -=cantidad;
15    } else {
16      console.log ("La cantidad a retirar debe se
17    }
18  }
19
20  return{
```

Print output (drag lower right corner to resize)

Global frame

- crearCuentaBancaria
- miCuenta
- undefined

crearCuentaBancaria

- saldoinicial: 1000
- saldo: undefined
- depositar
- retirar

Objects

```
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;

  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log ("La cantidad debe ser mayor a cero");
    }
  }

  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -=cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }

  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

```
function depositar(cantidad) {
  if (cantidad > 0){
    saldo += cantidad;
  } else{
    console.log ("La cantidad debe ser mayor a cero");
  }
}
```

Step 2 of 34

3.Se hace return y el valor de saldo cambia a 1000.

[illegible]

4. Retorna el valor.

JavaScript (ES6)
[known limitations](#)

```

6      saldo += cantidad;
7    } else{
8      console.log ("La cantidad debe ser mayor a
9    }
10 }
11
12 function retirar (cantidad){
13   if (cantidad > 0 && cantidad <= saldo){
14     saldo -=cantidad;
15   } else {
16     console.log ("La cantidad a retirar debe se
17   }
18 }
19
20 return{
21   consultarSaldo: function(){
22     return saldo;
23   },
24   realizarDeposito: function (cantidad){
25     depositar(cantidad);

```

[Edit this code](#)

⇒ line that just executed
→ next line to execute

<< First
 < Prev
 Next >
 Last >>

Step 4 of 34

Print output (drag lower right corner to resize)

Frames

Global frame

crearCuentaBancaria

miCuenta

undefined

Objects

crearCuentaBancaria
saldoinicial
saldo
depositar
retirar
Return value

```

function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;

  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log ("La cantidad debe ser mayor a cero");
    }
  }

  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -=cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }

  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarBatro: function (cantidad){
      retirar(cantidad);
    }
  };
}

function depositar(cantidad) {
  if (cantidad > 0){
    saldo += cantidad;
  } else{
    console.log ("La cantidad debe ser mayor a cero");

```

5. Se muestra en cosola el saldo inicial + la función: `miCuenta.consultarSaldo()`

```
22     return saldo;
23 },
24 realizarDeposito: function (cantidad){
25     depositar(cantidad);
26 },
27 realizarRetiro: function (cantidad){
28     retirar(cantidad);
29 }
30 };
31
32 }
33
34 var miCuenta = crearCuentaBancaria(1000);
35 console.log ("saldo inicial: " + miCuenta.consultar
36 miCuenta.realizarDeposito (500);
```

Global frame

- crearCuentaBancaria
- miCuenta

```
function crearCuentaBancaria(saldoinicial){
    var saldo = saldoinicial;

    function depositar (cantidad) {
        if (cantidad > 0){
            saldo += cantidad;
        } else{
            console.log ("La cantidad debe ser mayor a cero");
        }
    }

    function retirar (cantidad){
        if (cantidad > 0 && cantidad <= saldo){
            saldo -= cantidad;
        } else {
            console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
        }
    }

    return{
        consultarSaldo: function(){
            return saldo;
        },
        realizarDeposito: function (cantidad){
            depositar(cantidad);
        },
        realizarRetiro: function (cantidad){
            retirar(cantidad);
        }
    };
}
```

object	
consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad){ depositar(cantidad); }
realizarRetiro	function (cantidad){ retirar(cantidad); }

6.

Se retorna saldo:

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

```
12 function retirar (cantidad){
13     if (cantidad > 0 && cantidad <= saldo){
14         saldo -= cantidad;
15     } else {
16         console.log ("La cantidad a retirar debe ser
17     }
18 }
19
20 return{
21     consultarSaldo: function(){
22         return saldo;
23     },
24     realizarDeposito: function (cantidad){
25         depositar(cantidad);
26     },
27     realizarRetiro: function (cantidad){
28         retirar(cantidad);
29     }
30 };
31
```

JavaScript (ES6)

Print output (drag lower right corner to resize)

Frames

- Global frame
- crearCuentaBancaria
- miCuenta
- this
- parent:saldo 1000
- parent:depositar
- parent:retirar

Objects

```
function crearCuentaBancaria(saldoinicial){
    var saldo = saldoinicial;

    function depositar (cantidad) {
        if (cantidad > 0){
            saldo += cantidad;
        } else{
            console.log ("La cantidad debe ser mayor a cero");
        }
    }

    function retirar (cantidad){
        if (cantidad > 0 && cantidad <= saldo){
            saldo -= cantidad;
        } else {
            console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
        }
    }

    return{
        consultarSaldo: function(){
            return saldo;
        },
        realizarDeposito: function (cantidad){
            depositar(cantidad);
        },
        realizarRetiro: function (cantidad){
            retirar(cantidad);
        }
    };
}
```

object	
consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad){ depositar(cantidad); }
realizarRetiro	function (cantidad){ retirar(cantidad); }

7. Se retorna el valor que hemos agregado:

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

The screenshot shows the Python Tutor interface for JavaScript (ES6). The code defines a bank account object with methods for depositing, withdrawing, and checking the balance. The execution state shows the 'Global frame' with a 'miCuenta' object. The 'parent:saldo' property is highlighted with a value of 1000. The 'Return value' is also 1000. The 'Objects' panel shows the 'crearCuentaBancaria' function and the 'miCuenta' object.

```
JavaScript (ES6)
function retirar (cantidad){
  if (cantidad > 0 && cantidad <= saldo){
    saldo -= cantidad;
  } else {
    console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
  }
}

return {
  consultarSaldo: function(){
    return saldo;
  },
  realizarDeposito: function (cantidad){
    depositar(cantidad);
  },
  realizarRetiro: function (cantidad){
    retirar(cantidad);
  }
};
```

Print output (drag lower right corner to resize)

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1000

parent:depositar

parent:retirar

Return value 1000

Objects

```
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;

  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else {
      console.log ("La cantidad debe ser mayor a cero");
    }
  }

  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }

  return {
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

8. Se muestra en consola Mi saldo inicial + `miCuenta.consultarSaldo()`

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

The screenshot shows the Python Tutor interface for JavaScript (ES6). The code defines a bank account object and creates an instance 'miCuenta'. The execution state shows the 'Global frame' with a 'miCuenta' object. The 'parent:saldo' property is highlighted with a value of 1000. The 'Return value' is also 1000. The 'Objects' panel shows the 'crearCuentaBancaria' function and the 'miCuenta' object. The console output shows the initial balance: "saldo inicial: 1000".

```
JavaScript (ES6)
function retirar (cantidad){
  if (cantidad > 0 && cantidad <= saldo){
    saldo -= cantidad;
  } else {
    console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
  }
}

return {
  consultarSaldo: function(){
    return saldo;
  },
  realizarDeposito: function (cantidad){
    depositar(cantidad);
  },
  realizarRetiro: function (cantidad){
    retirar(cantidad);
  }
};
```

Print output (drag lower right corner to resize)

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1000

parent:depositar

parent:retirar

Return value 1000

Objects

```
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;

  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else {
      console.log ("La cantidad debe ser mayor a cero");
    }
  }

  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }

  return {
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

object

console.log ("saldo inicial: " + miCuenta.consultarSaldo());

9. Se llama a miCuenta miCuenta.realizarDeposito y se agregan (500)

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

```
JavaScript (ES6)
known limitations
24  realizarDeposito: function (cantidad){
25      depositar(cantidad);
26  },
27  realizarRetiro: function (cantidad){
28      retirar(cantidad);
29  }
30  };
31
32  }
33
34  var miCuenta = crearCuentaBancaria(1000);
35  console.log("saldo inicial: " + miCuenta.consultarSa
36  miCuenta.realizarDeposito(500);
37  console.log("Saldo despues del deposito:" + miCuenta.
38  miCuenta.realizarRetiro(200);
39  console.log("saldo despues del retiro:" + miCuenta.co
40
41
42  try{
43      miCuenta.depositar(100);
44  }
45  }

Print output (drag lower right corner to resize)
saldo inicial: 1000

Frames
Global frame
  crearCuentaBancaria
  miCuenta

Objects
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

10. Se comienza a llamar a la función para mostrar el contenido en consola

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

```
JavaScript (ES6)
known limitations
24  realizarDeposito: function (cantidad){
25      depositar(cantidad);
26  },
27  realizarRetiro: function (cantidad){
28      retirar(cantidad);
29  }
30  };
31
32  }
33
34  var miCuenta = crearCuentaBancaria(1000);
35  console.log("saldo inicial: " + miCuenta.consultarSa
36  miCuenta.realizarDeposito(500);
37  console.log("Saldo despues del deposito:" + miCuenta.
38  miCuenta.realizarRetiro(200);
39  console.log("saldo despues del retiro:" + miCuenta.co
40
41
42  try{
43      miCuenta.depositar(100);
44  }
45  }

Print output (drag lower right corner to resize)
saldo inicial: 1000

Frames
Global frame
  crearCuentaBancaria
  miCuenta
  this
  parent:saldo 1000
  parent:depositar
  parent:retirar
  cantidad 500

Objects
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

11. Se genera la condicional if diciendo que debe ser mayor a 0.

The screenshot shows a JavaScript IDE with the following code:

```
1 function crearCuentaBancaria(saldoinicial){
2   var saldo = saldoinicial;
3
4   function depositar (cantidad) {
5     if (cantidad > 0){
6       saldo += cantidad;
7     } else{
8       console.log ("La cantidad debe ser mayor a 0");
9     }
10  }
11
12  function retirar (cantidad){
13    if (cantidad > 0 && cantidad <= saldo){
14      saldo -= cantidad;
15    } else {
16      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
17    }
18  }
19
20  return {
21    depositar,
22    retirar,
23    consultarSaldo: function(){
24      return saldo;
25    },
26    realizarDeposito: function (cantidad){
27      depositar(cantidad);
28    },
29    realizarRetiro: function (cantidad){
30      retirar(cantidad);
31    }
32  };
33 }
34
```

The execution state shows the following:

- Print output:** saldo inicial: 1000
- Frames:**
 - Global frame: crearCuentaBancaria, miCuenta
 - miCuenta: this, parent:saldo (1000), parent:depositar, parent:retirar, cantidad (500)
 - depositar: parent:saldo (1000), parent:depositar, parent:retirar, cantidad (500)
- Objects:**
 - consultarSaldo: function () { return saldo; }

12. si la cantidad es mayor a 0 decimos que a saldo se le suma la cantidad,

The screenshot shows the same JavaScript IDE as above, but at a later execution step. The code is the same, but the execution state has changed:

- Print output:** saldo inicial: 1000
- Frames:**
 - Global frame: crearCuentaBancaria, miCuenta
 - miCuenta: this, parent:saldo (1000), parent:depositar, parent:retirar, cantidad (500)
 - depositar: parent:saldo (1000), parent:depositar, parent:retirar, cantidad (500)
- Objects:**
 - consultarSaldo: function () { return saldo; }

13. Se inicia a trabajar nuestro Else.

JavaScript (ES6)

```
14 // ...
15 } else {
16   console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
17 }
18 }
19
20 return{
21   consultarSaldo: function(){
22     return saldo;
23   },
24   realizarDeposito: function (cantidad){
25     depositar(cantidad);
26   },
27   realizarRetiro: function (cantidad){
28     retirar(cantidad);
29   }
30 };
31
32 }
33
34 var miCuenta = crearCuentaBancaria (1000);
```

Print output (drag lower right corner to resize)

saldo inicial: 1000

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1000

parent:depositar

parent:retirar

cantidad 500

Objects

function crearCuentaBancaria(saldoinicial){
 var saldo = saldoinicial;
 function depositar (cantidad) {
 if (cantidad > 0){
 saldo += cantidad;
 } else{
 console.log("La cantidad debe ser mayor a cero");
 }
 }
 function retirar (cantidad){
 if (cantidad > 0 && cantidad <= saldo){
 saldo -= cantidad;
 } else {
 console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
 }
 }
 return{
 consultarSaldo: function(){
 return saldo;
 },
 realizarDeposito: function (cantidad){
 depositar(cantidad);
 },
 realizarRetiro: function (cantidad){
 retirar(cantidad);
 }
 };
}

object

consultarSaldo

function (){
 return saldo;
}

14. Se crea la función realizar retiro

JavaScript (ES6)

```
14 // ...
15 } else {
16   console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
17 }
18 }
19
20 return{
21   consultarSaldo: function(){
22     return saldo;
23   },
24   realizarDeposito: function (cantidad){
25     depositar(cantidad);
26   },
27   realizarRetiro: function (cantidad){
28     retirar(cantidad);
29   }
30 };
31
32 }
33
34 var miCuenta = crearCuentaBancaria (1000);
35 console.log("saldo inicial: " + miCuenta.consultarSaldo());
```

Print output (drag lower right corner to resize)

saldo inicial: 1000

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1500

parent:depositar

parent:retirar

cantidad 500

Return value undefined

Objects

function crearCuentaBancaria(saldoinicial){
 var saldo = saldoinicial;
 function depositar (cantidad) {
 if (cantidad > 0){
 saldo += cantidad;
 } else{
 console.log("La cantidad debe ser mayor a cero");
 }
 }
 function retirar (cantidad){
 if (cantidad > 0 && cantidad <= saldo){
 saldo -= cantidad;
 } else {
 console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
 }
 }
 return{
 consultarSaldo: function(){
 return saldo;
 },
 realizarDeposito: function (cantidad){
 depositar(cantidad);
 },
 realizarRetiro: function (cantidad){
 retirar(cantidad);
 }
 };
}

object

function (){
 return saldo;
}

15. Se muestra en consola saldo después del depósito mas
`miCuenta.consultarSaldo()`;

```
JavaScript (ES6)
function crearCuentaBancaria(saldoInicial){
  var saldo = saldoInicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else {
      console.log ("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return {
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}

var miCuenta = crearCuentaBancaria (1000);
console.log ("saldo inicial: " + miCuenta.consultarSaldo());
miCuenta.realizarDeposito (500);
console.log("Saldo despues del deposito:" + miCuenta.consultarSaldo());
miCuenta.realizarRetiro(200);
console.log("saldo despues del retiro:" + miCuenta.consultarSaldo());

try{
  miCuenta.depositar(100);
} catch (e){
  console.log (e.message);
}
```

Print output (drag lower right corner to resize)

saldo inicial: 1000

Global frame

- crearCuentaBancaria
- miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial){
  var saldo = saldoInicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else {
      console.log ("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return {
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

object

consultarSaldo	function () { return saldo; }
----------------	-------------------------------------

Step 15 of 34

16. retornamos saldo

```
JavaScript (ES6)
function retirar (cantidad){
  if (cantidad > 0 && cantidad <= saldo){
    saldo -= cantidad;
  } else {
    console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
  }
}

function crearCuentaBancaria(saldoInicial){
  var saldo = saldoInicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else {
      console.log ("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return {
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}

var miCuenta = crearCuentaBancaria (1000);
console.log ("saldo inicial: " + miCuenta.consultarSaldo());
miCuenta.realizarDeposito (500);
console.log("Saldo despues del deposito:" + miCuenta.consultarSaldo());
miCuenta.realizarRetiro(200);
console.log("saldo despues del retiro:" + miCuenta.consultarSaldo());

try{
  miCuenta.depositar(100);
} catch (e){
  console.log (e.message);
}
```

Print output (drag lower right corner to resize)

saldo inicial: 1000

Global frame

- crearCuentaBancaria
- miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial){
  var saldo = saldoInicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else {
      console.log ("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return {
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

object

consultarSaldo	function () { return saldo; }
----------------	-------------------------------------

Step 16 of 34

17. Decimos que el valor que se retorna es 1500

The screenshot shows a JavaScript IDE with the following code:

```
JavaScript (ES6)
known limitations

12 function retirar (cantidad){
13   if (cantidad > 0 && cantidad <= saldo){
14     saldo -= cantidad;
15   } else {
16     console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
17   }
18 }
19
20 return{
21   consultarSaldo: function(){
22     return saldo;
23   },
24   realizarDeposito: function (cantidad){
25     depositar(cantidad);
26   },
27   realizarRetiro: function (cantidad){
28     retirar(cantidad);
29   }
30 };
31
```

The console output shows: `saldo inicial: 1000`.

The state of the `miCuenta` object is shown in the **Frames** and **Objects** panels:

- Global frame:** `crearCuentaBancaria` and `miCuenta`.
- miCuenta object:** `this` points to the object, `parent:saldo` is 1500, `parent:depositar` is a function, and `parent:retirar` is a function. The `Return value` is 1500.
- Objects panel:** `consultarSaldo` is a function that returns `saldo`.

18. se muestra en consola el saldo

The screenshot shows a JavaScript IDE with the following code:

```
JavaScript (ES6)
known limitations

27   realizarRetiro: function (cantidad){
28     retirar(cantidad);
29   }
30 };
31
32 }
33
34 var miCuenta = crearCuentaBancaria (1000);
35 console.log ("saldo inicial: " + miCuenta.consultarSaldo());
36 miCuenta.realizarDeposito (500);
37 console.log("Saldo despues del deposito:" + miCuenta.consultarSaldo());
38 miCuenta.realizarRetiro(200);
39 console.log("saldo despues del retiro:" + miCuenta.consultarSaldo());
40
41
42 try{
43   miCuenta.depositar(100);
44 } catch (e){
45   console.log (e.message);
46 }
47
```

The console output shows: `saldo inicial: 1000` and `Saldo despues del deposito:1500`.

The state of the `miCuenta` object is shown in the **Frames** and **Objects** panels:

- Global frame:** `crearCuentaBancaria` and `miCuenta`.
- miCuenta object:** `this` points to the object, `parent:saldo` is 1500, `parent:depositar` is a function, and `parent:retirar` is a function. The `Return value` is 1500.
- Objects panel:** `consultarSaldo` is a function that returns `saldo`.

19. Decimos que se hace un retiro de 200 `miCuenta.realizarRetiro(200);`

The screenshot shows a JavaScript debugger with the following components:

- Code Editor:** Displays JavaScript code. Line 38, `miCuenta.realizarRetiro(200);`, is highlighted in red, indicating it is the current line of execution.
- Print Output:** A box at the top right shows the output of `console.log` statements: `saldo inicial: 1000` and `Saldo despues del deposito:1500`.
- Frames:** A stack of frames on the right shows the call stack. The top frame is `Global frame` with `crearCuentaBancaria` and `miCuenta`. A blue arrow points from the `miCuenta` frame to the `Objects panel.`
- Objects:** A panel on the right showing the state of objects. It includes a `consultarSaldo` object and a `function ()` object.
- Debugger Controls:** At the bottom, there are buttons for `<< First`, `< Prev`, `Next >`, and `Last >>`, along with a step counter showing `Step 19 of 34`.

20. Decimos que se va a retirar la cantidad de 200

The screenshot shows the same JavaScript debugger as in the previous image, but at a different point in execution:

- Code Editor:** Line 38, `miCuenta.realizarRetiro(200);`, is still highlighted in red.
- Print Output:** The output box shows the same initial state: `saldo inicial: 1000` and `Saldo despues del deposito:1500`.
- Frames:** The call stack now includes a new frame for `miCuenta` with the following properties: `this`, `parent:saldo` (1500), `parent:depositar`, `parent:retirar`, and `cantidad` (200). A blue arrow points from the `miCuenta` frame to the `Objects` panel.
- Objects:** The `consultarSaldo` object and the `function ()` object are still present.
- Debugger Controls:** The step counter now shows `Step 20 of 34`.

21. Se crea el If de la función retirar

The screenshot shows a JavaScript IDE with the following code:

```
JavaScript (ES6)
var saldo = 1000;

function depositar (cantidad) {
  if (cantidad > 0){
    saldo += cantidad;
  } else{
    console.log ("La cantidad debe ser mayor a ce");
  }
}

function retirar (cantidad){
  if (cantidad > 0 && cantidad <= saldo){
    saldo -= cantidad;
  } else {
    console.log ("La cantidad a retirar debe ser ");
  }
}

return{
  consultarSaldo: function(){
    return saldo;
  }
}
```

The console output shows:

```
saldo inicial: 1000
Saldo despues del deposito:1500
```

The debugger shows the following frames and objects:

Frames:

- Global frame
- crearCuentaBancaria
- miCuenta
- parent:saldo 1500
- parent:depositar
- parent:retirar
- cantidad 200
- retirar
- parent:saldo 1500
- parent:depositar
- parent:retirar
- cantidad 200

Objects:

- function crearCuentaBancaria(saldoinicial){
 var saldo = saldoinicial;
 function depositar (cantidad) {
 if (cantidad > 0){
 saldo += cantidad;
 } else{
 console.log ("La cantidad debe ser mayor a cero");
 }
 }
 function retirar (cantidad){
 if (cantidad > 0 && cantidad <= saldo){
 saldo -= cantidad;
 } else {
 console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
 }
 }
 return{
 consultarSaldo: function(){
 return saldo;
 },
 realizarDeposito: function (cantidad){
 depositar(cantidad);
 },
 realizarRetiro: function (cantidad){
 retirar(cantidad);
 }
 };
}
- object
- consultarSaldo
- function (){
 return saldo;
}

22. Decimos que al saldo se le resta la cantidad:

The screenshot shows a JavaScript IDE with the following code:

```
JavaScript (ES6)
var saldo = 1000;

function depositar (cantidad) {
  if (cantidad > 0){
    saldo += cantidad;
  } else{
    console.log ("La cantidad debe ser mayor a ce");
  }
}

function retirar (cantidad){
  if (cantidad > 0 && cantidad <= saldo){
    saldo -= cantidad;
  } else {
    console.log ("La cantidad a retirar debe ser ");
  }
}

return{
  consultarSaldo: function(){
    return saldo;
  }
}
```

The console output shows:

```
saldo inicial: 1000
Saldo despues del deposito:1500
```

The debugger shows the following frames and objects:

Frames:

- Global frame
- crearCuentaBancaria
- miCuenta
- parent:saldo 1500
- parent:depositar
- parent:retirar
- cantidad 200
- retirar
- parent:saldo 1500
- parent:depositar
- parent:retirar
- cantidad 200

Objects:

- function crearCuentaBancaria(saldoinicial){
 var saldo = saldoinicial;
 function depositar (cantidad) {
 if (cantidad > 0){
 saldo += cantidad;
 } else{
 console.log ("La cantidad debe ser mayor a cero");
 }
 }
 function retirar (cantidad){
 if (cantidad > 0 && cantidad <= saldo){
 saldo -= cantidad;
 } else {
 console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
 }
 }
 return{
 consultarSaldo: function(){
 return saldo;
 },
 realizarDeposito: function (cantidad){
 depositar(cantidad);
 },
 realizarRetiro: function (cantidad){
 retirar(cantidad);
 }
 };
}
- object
- consultarSaldo
- function (){
 return saldo;
}

23.El valor que se retorna aun esta indefinido

JavaScript (ES6) known limitations

```
10 1
11
12 var saldo = 1000;
13
14 function depositar (cantidad) {
15   if (cantidad > 0){
16     saldo += cantidad;
17   } else{
18     console.log ("La cantidad debe ser mayor a cero");
19   }
20 }
21
22 function retirar (cantidad){
23   if (cantidad > 0 && cantidad <= saldo){
24     saldo -= cantidad;
25   } else {
26     console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
27   }
28 }
29
30 return{
31   consultarSaldo: function(){
32     return saldo;
33   }
34 }
```

saldo inicial: 1000
Saldo despues del deposito:1500

Frames

- Global frame
 - crearCuentaBancaria
 - miCuenta
 - this
 - parent:saldo 1300
 - parent:depositar
 - parent:retirar
 - cantidad 200
- retirar
 - parent:saldo 1300
 - parent:depositar
 - parent:retirar
 - cantidad 200
 - Return value undefined

Objects

- function crearCuentaBancaria(saldoinicial){
var saldo = saldoinicial;
function depositar (cantidad) {
if (cantidad > 0){
saldo += cantidad;
}
else{
console.log ("La cantidad debe ser mayor a cero");
}
}
function retirar (cantidad){
if (cantidad > 0 && cantidad <= saldo){
saldo -= cantidad;
}
else {
console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
}
}
return{
consultarSaldo: function(){
return saldo;
};
realizarDeposito: function (cantidad){
depositar(cantidad);
};
realizarRetiro: function (cantidad){
retirar(cantidad);
};
};
}
- object
 - consultarSaldo function (){
return saldo;
}

Step 23 of 34

24.Cerramos nuestro return:

JavaScript (ES6) known limitations

```
10 1
11
12 var saldo = 1000;
13
14 function depositar (cantidad) {
15   if (cantidad > 0){
16     saldo += cantidad;
17   } else{
18     console.log ("La cantidad debe ser mayor a cero");
19   }
20 }
21
22 function retirar (cantidad){
23   if (cantidad > 0 && cantidad <= saldo){
24     saldo -= cantidad;
25   } else {
26     console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
27   }
28 }
29
30 return{
31   consultarSaldo: function(){
32     return saldo;
33   }
34 };
35
36 var miCuenta = crearCuentaBancaria (1000);
37 console.log ("saldo inicial: " + miCuenta.consultarSaldo());
38 miCuenta.realizarDeposito (500);
39 console.log("Saldo despues del deposito:" + miCuenta.consultarSaldo());
40 miCuenta.realizarRetiro(200);
```

saldo inicial: 1000
Saldo despues del deposito:1500

Frames

- Global frame
 - crearCuentaBancaria
 - miCuenta
 - this
 - parent:saldo 1300
 - parent:depositar
 - parent:retirar
 - cantidad 200
- retirar
 - parent:saldo 1300
 - parent:depositar
 - parent:retirar
 - cantidad 200
 - Return value undefined

Objects

- function crearCuentaBancaria(saldoinicial){
var saldo = saldoinicial;
function depositar (cantidad) {
if (cantidad > 0){
saldo += cantidad;
}
else{
console.log ("La cantidad debe ser mayor a cero");
}
}
function retirar (cantidad){
if (cantidad > 0 && cantidad <= saldo){
saldo -= cantidad;
}
else {
console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
}
}
return{
consultarSaldo: function(){
return saldo;
};
realizarDeposito: function (cantidad){
depositar(cantidad);
};
realizarRetiro: function (cantidad){
retirar(cantidad);
};
};
}
- object
 - consultarSaldo function (){
return saldo;
}

Step 24 of 34

25. Se muestra en consola el saldo después del retiro

+ `miCuenta.consultarSaldo()`;

JavaScript (ES6)

```
29 }
30 };
31
32 }
33
34 var miCuenta = crearCuentaBancaria(1000);
35 console.log("saldo inicial: " + miCuenta.consultarSaldo());
36 miCuenta.realizarDeposito(500);
37 console.log("Saldo despues del deposito:" + miCuenta.consultarSaldo());
38 miCuenta.realizarRetiro(200);
39 console.log("saldo despues del retiro:" + miCuenta.consultarSaldo());
40
41 try{
42   miCuenta.depositar(100);
43 } catch (e){
44   console.log(e.message);
45 }
46
47
48 try {
```

Print output (drag lower right corner to resize)

saldo inicial: 1000
Saldo despues del deposito:1500

Frames

Global frame

crearCuentaBancaria

miCuenta

Objects

```
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

object

consultarSaldo

function (){
 return saldo;
}

26. Se retorna saldo:

JavaScript (ES6)

```
12 function retirar (cantidad){
13   if (cantidad > 0 && cantidad <= saldo){
14     saldo -= cantidad;
15   } else {
16     console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
17   }
18 }
19
20 return{
21   consultarSaldo: function(){
22     return saldo;
23   },
24   realizarDeposito: function (cantidad){
25     depositar(cantidad);
26   },
27   realizarRetiro: function (cantidad){
28     retirar(cantidad);
29   }
30 };
31
```

Print output (drag lower right corner to resize)

saldo inicial: 1000
Saldo despues del deposito:1500

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1300

parent:depositar

parent:retirar

Objects

```
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

object

consultarSaldo

function (){
 return saldo;
}

27. Retorna 1300

JavaScript (ES6)

```
12 function retirar (cantidad){
13   if (cantidad > 0 && cantidad <= saldo){
14     saldo -= cantidad;
15   } else {
16     console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
17   }
18 }
19
20 return{
21   consultarSaldo: function(){
22     return saldo;
23   },
24   realizarDeposito: function (cantidad){
25     depositar(cantidad);
26   },
27   realizarRetiro: function (cantidad){
28     retirar(cantidad);
29   }
30 };
31
```

Print output (drag lower right corner to resize)

saldo inicial: 1000
Saldo despues del deposito:1500

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1300

parent:depositar

parent:retirar

Return value 1300

Objects

```
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log ("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

object

consultarSaldo

function (){
 return saldo;
}

28 Se muestra en consola el saldo después del retiro

+ `miCuenta.consultarSaldo()`;

JavaScript (ES6)

```
29 }
30 };
31
32 }
33
34 var miCuenta = crearCuentaBancaria (1000);
35 console.log ("saldo inicial: " + miCuenta.consultarSaldo());
36 miCuenta.realizarDeposito (500);
37 console.log("Saldo despues del deposito:" + miCuenta.consultarSaldo());
38 miCuenta.realizarRetiro(200);
39 console.log("saldo despues del retiro:" + miCuenta.consultarSaldo());
40
41 try{
42   miCuenta.depositar(100);
43 } catch (e){
44   console.log (e.message);
45 }
46
47
48 try {
49   // ...
50 }
51
```

Print output (drag lower right corner to resize)

saldo inicial: 1000
Saldo despues del deposito:1500
saldo despues del retiro: 1300

Frames

Global frame

crearCuentaBancaria

miCuenta

Objects

```
function crearCuentaBancaria(saldoinicial){
  var saldo = saldoinicial;
  function depositar (cantidad) {
    if (cantidad > 0){
      saldo += cantidad;
    } else{
      console.log ("La cantidad debe ser mayor a cero");
    }
  }
  function retirar (cantidad){
    if (cantidad > 0 && cantidad <= saldo){
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
    }
  }
  return{
    consultarSaldo: function(){
      return saldo;
    },
    realizarDeposito: function (cantidad){
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad){
      retirar(cantidad);
    }
  };
}
```

object

consultarSaldo

function (){
 return saldo;
}

29. Iniciamos con try

The screenshot shows a JavaScript IDE with a code editor on the left and a console/output area on the right. The code editor displays a JavaScript script (ES6) with the following content:

```
29 }
30 };
31
32 }
33
34 var miCuenta = crearCuentaBancaria(1000);
35 console.log("saldo inicial: " + miCuenta.consultarSa
36 miCuenta.realizarDeposito(500);
37 console.log("Saldo despues del deposito:" + miCuenta.
38 miCuenta.realizarRetiro(200);
39 console.log("saldo despues del retiro:" + miCuenta.co
40
41
42 try{
43   miCuenta.depositar(100);
44 } catch (e){
45   console.log(e.message);
46 }
47
48 try {
```

The console output area shows the following text:

```
Print output (drag lower right corner to resize)
saldo inicial: 1000
Saldo despues del deposito:1500
saldo despues del retiro:1300
```

The IDE also shows a 'Frames' panel with 'Global frame' and 'crearCuentaBancaria' frames, and an 'Objects' panel showing the 'miCuenta' object and its methods.

30. Try intenta agregar 100 a la cuenta:

The screenshot shows a JavaScript IDE with a code editor on the left and a console/output area on the right. The code editor displays a JavaScript script (ES6) with the following content:

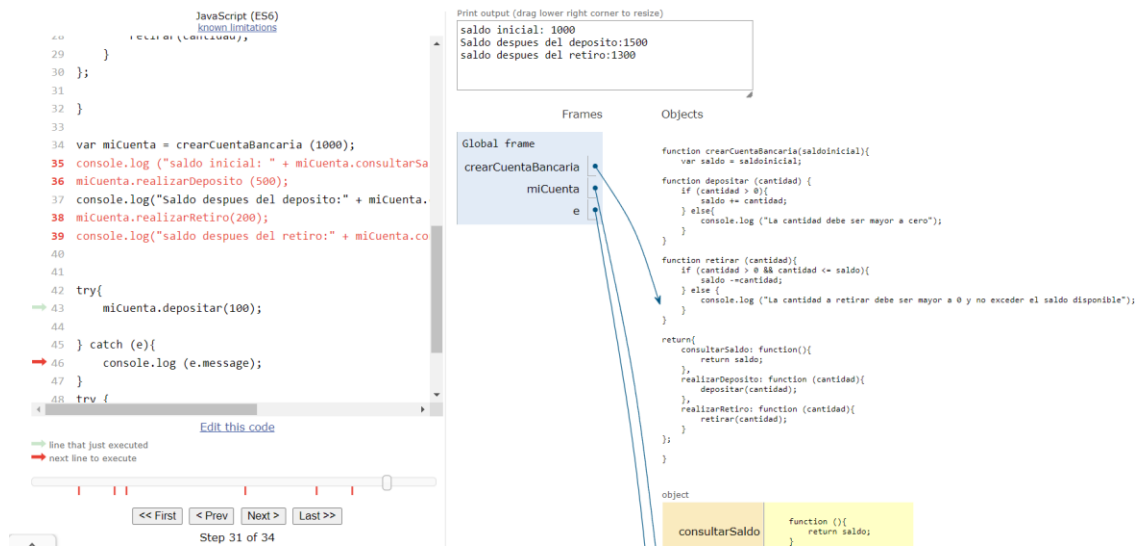
```
29 }
30 };
31
32 }
33
34 var miCuenta = crearCuentaBancaria(1000);
35 console.log("saldo inicial: " + miCuenta.consultarSa
36 miCuenta.realizarDeposito(500);
37 console.log("Saldo despues del deposito:" + miCuenta.
38 miCuenta.realizarRetiro(200);
39 console.log("saldo despues del retiro:" + miCuenta.co
40
41
42 try{
43   miCuenta.depositar(100);
44 } catch (e){
45   console.log(e.message);
46 }
47
48 try {
```

The console output area shows the following text:

```
Print output (drag lower right corner to resize)
saldo inicial: 1000
Saldo despues del deposito:1500
saldo despues del retiro:1300
```

The IDE also shows a 'Frames' panel with 'Global frame' and 'crearCuentaBancaria' frames, and an 'Objects' panel showing the 'miCuenta' object and its methods.

31. se intenta mostrar en consola e.message



JavaScript (ES6) known limitations

```
28 }  
29 }  
30 }  
31 }  
32 }  
33 }  
34 var miCuenta = crearCuentaBancaria(1000);  
35 console.log("saldo inicial: " + miCuenta.consultarSa  
36 miCuenta.realizarDeposito(500);  
37 console.log("Saldo despues del deposito:" + miCuenta.  
38 miCuenta.realizarRetiro(200);  
39 console.log("saldo despues del retiro:" + miCuenta.co  
40 }  
41 }  
42 try{  
43     miCuenta.depositar(100);  
44 } catch (e){  
45     console.log(e.message);  
46 }  
47 }  
48 try {
```

Print output (drag lower right corner to resize)

```
saldo inicial: 1000  
Saldo despues del deposito:1500  
saldo despues del retiro:1300
```

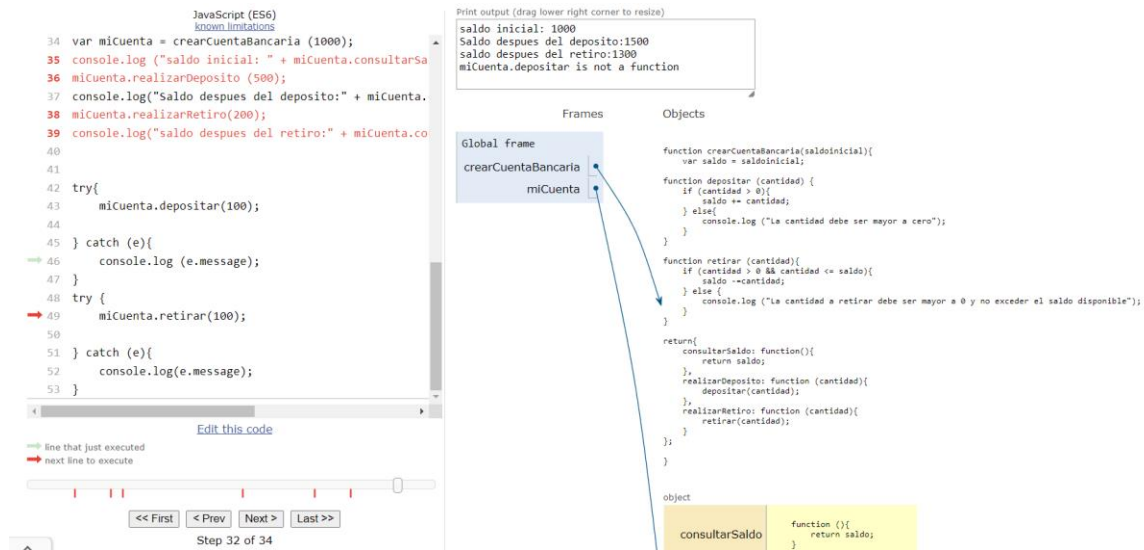
Frames

- Global frame
 - crearCuentaBancaria
 - miCuenta
 - e

Objects

- function crearCuentaBancaria(saldoinicial){
 var saldo = saldoinicial;
 function depositar (cantidad) {
 if (cantidad > 0){
 saldo += cantidad;
 } else{
 console.log ("La cantidad debe ser mayor a cero");
 }
 }
 function retirar (cantidad){
 if (cantidad > 0 && cantidad <= saldo){
 saldo -=cantidad;
 } else {
 console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
 }
 }
 return{
 consultarSaldo: function(){
 return saldo;
 },
 realizarDeposito: function (cantidad){
 depositar(cantidad);
 },
 realizarRetiro: function (cantidad){
 retirar(cantidad);
 }
 };
}
- object
 - consultarSaldo
 - function () {
 return saldo;
}

32. Se inicia un nuevo try



JavaScript (ES6) known limitations

```
34 var miCuenta = crearCuentaBancaria(1000);  
35 console.log("saldo inicial: " + miCuenta.consultarSa  
36 miCuenta.realizarDeposito(500);  
37 console.log("Saldo despues del deposito:" + miCuenta.  
38 miCuenta.realizarRetiro(200);  
39 console.log("saldo despues del retiro:" + miCuenta.co  
40 }  
41 }  
42 try{  
43     miCuenta.depositar(100);  
44 } catch (e){  
45     console.log(e.message);  
46 }  
47 }  
48 try {  
49     miCuenta.retirar(100);  
50 } catch (e){  
51     console.log(e.message);  
52 }  
53 }
```

Print output (drag lower right corner to resize)

```
saldo inicial: 1000  
Saldo despues del deposito:1500  
saldo despues del retiro:1300  
miCuenta.depositar is not a function
```

Frames

- Global frame
 - crearCuentaBancaria
 - miCuenta

Objects

- function crearCuentaBancaria(saldoinicial){
 var saldo = saldoinicial;
 function depositar (cantidad) {
 if (cantidad > 0){
 saldo += cantidad;
 } else{
 console.log ("La cantidad debe ser mayor a cero");
 }
 }
 function retirar (cantidad){
 if (cantidad > 0 && cantidad <= saldo){
 saldo -=cantidad;
 } else {
 console.log ("La cantidad a retirar debe ser mayor a 0 y no exceder el saldo disponible");
 }
 }
 return{
 consultarSaldo: function(){
 return saldo;
 },
 realizarDeposito: function (cantidad){
 depositar(cantidad);
 },
 realizarRetiro: function (cantidad){
 retirar(cantidad);
 }
 };
}
- object
 - consultarSaldo
 - function () {
 return saldo;
}

33. Se intentará retirar 100

The screenshot shows a JavaScript console and a debugger interface. The console output is as follows:

```
saldo inicial: 1000
Saldo despues del deposito:1500
saldo despues del retiro:1300
miCuenta.depositar is not a function
```

The debugger shows the following frames:

- Global frame: crearCuentaBancaria, miCuenta

The Objects panel shows the following objects:

- consultarSaldo: function () { return saldo; }

The code being executed is as follows:

```
34 var miCuenta = crearCuentaBancaria (1000);
35 console.log ("saldo inicial: " + miCuenta.consultarSa
36 miCuenta.realizarDeposito (500);
37 console.log("Saldo despues del deposito:" + miCuenta.
38 miCuenta.realizarRetiro(200);
39 console.log("saldo despues del retiro:" + miCuenta.co
40
41
42 try{
43     miCuenta.depositar(100);
44
45 } catch (e){
46     console.log (e.message);
47 }
48 try {
49     miCuenta.retirar(100);
50
51 } catch (e){
52     console.log(e.message);
53 }
```

The debugger shows the current step is 33 of 34.

34. Se muestra en consola e.message

The screenshot shows a JavaScript console and a debugger interface. The console output is as follows:

```
saldo inicial: 1000
Saldo despues del deposito:1500
saldo despues del retiro:1300
miCuenta.depositar is not a function
```

The debugger shows the following frames:

- Global frame: crearCuentaBancaria, miCuenta, e

The Objects panel shows the following objects:

- consultarSaldo: function () { return saldo; }

The code being executed is as follows:

```
34 var miCuenta = crearCuentaBancaria (1000);
35 console.log ("saldo inicial: " + miCuenta.consultarSa
36 miCuenta.realizarDeposito (500);
37 console.log("Saldo despues del deposito:" + miCuenta.
38 miCuenta.realizarRetiro(200);
39 console.log("saldo despues del retiro:" + miCuenta.co
40
41
42 try{
43     miCuenta.depositar(100);
44
45 } catch (e){
46     console.log (e.message);
47 }
48 try {
49     miCuenta.retirar(100);
50
51 } catch (e){
52     console.log(e.message);
53 }
```

The debugger shows the current step is 34 of 34.