MISSION READY

DARE TO **DEVELOP**

Debugging

Reuben Simpson

What is Debugging?

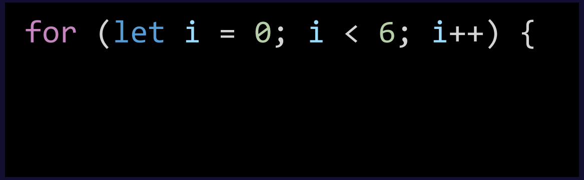
- When we talk about debugging we mean finding some code that is not performing in the way it is meant to (ie it has *bugs*), and changing it so that it performs as expected.
- Often it can be difficult to figure out exactly which parts of the code is causing the bugs, but there are some debugging techniques we can use to make that process easier.

Steps to take when debugging

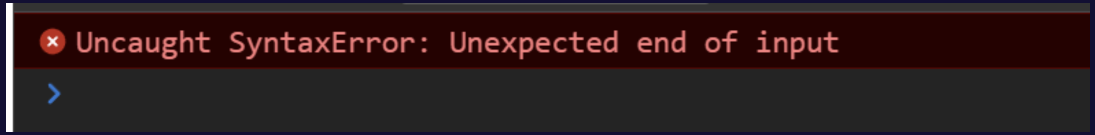
- Step 1: Check the console for an error message and read the message carefully.
- Step 2: use the browser debugger to step through the code and find out where it’s going wrong.
- Step 3: Delete everything and start over
- Step 4: Don’t actually do Step 3

Step 1

- Here we have a SyntaxError. Syntax refers to the “grammar” of our code, in this case we are missing a closing bracket } for our for loop.

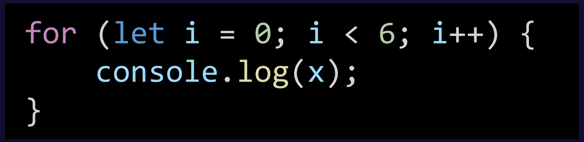


```
for (let i = 0; i < 6; i++) {
```

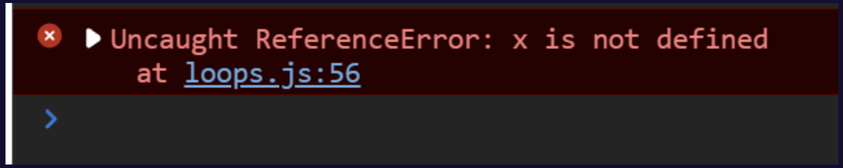


```
Uncaught SyntaxError: Unexpected end of input
>
```

- Here we have a ReferenceError. This is because we are trying to reference a variable x that we haven’t declared yet.



```
for (let i = 0; i < 6; i++) {
    console.log(x);
}
```



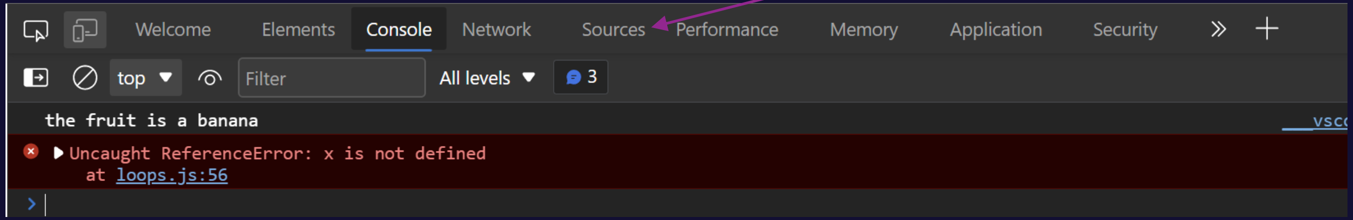
```
▶ Uncaught ReferenceError: x is not defined
      at loops.js:56
>
```

Step 2

- Your browser has a built-in “debugger” that we can use to help us debug our applications.
- Sometimes error messages may not be detailed enough to help us debug our application and other times there may not be an error message at all.
- In those cases, we can use the built-in debugger to help us fix our application.

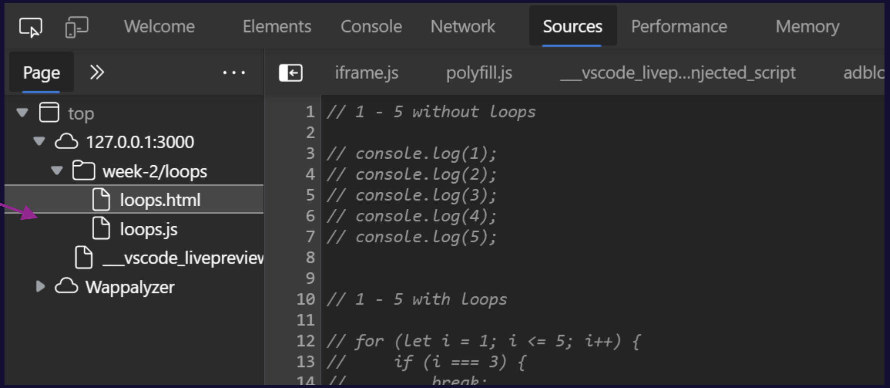
Step 2 continued…

- First make your way to the console in your browser.
- Next we will click on the sources tab



```
the fruit is a banana
Uncaught ReferenceError: x is not defined
    at loops.js:56
```

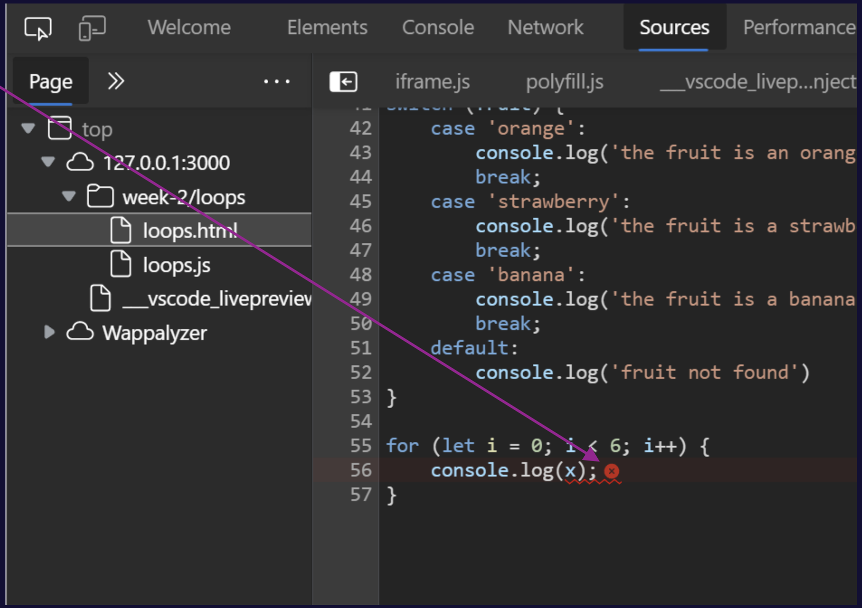
- This tab allows us to see the different files in our project, that are in use by the browser



```
// 1 - 5 without loops

// console.log(1);
// console.log(2);
// console.log(3);
// console.log(4);
// console.log(5);


// 1 - 5 with loops

// for (let i = 1; i <= 5; i++) {
//     if (i === 3) {
//         break;
```

Step 2 continued…

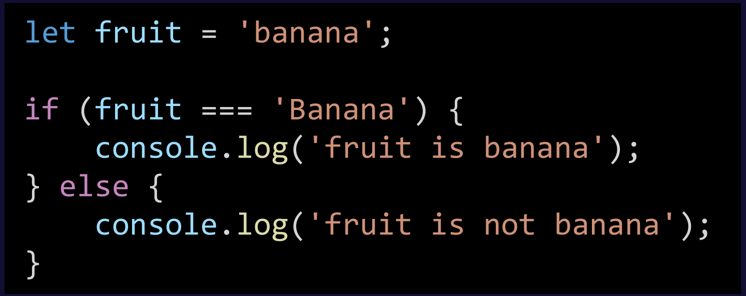
- If there is an error present, it will highlight exactly where it is in your file.



```
switch (fruit) {
    case 'orange':
        console.log('the fruit is an orang
        break;
    case 'strawberry':
        console.log('the fruit is a strawb
        break;
    case 'banana':
        console.log('the fruit is a banana
        break;
    default:
        console.log('fruit not found')
}

for (let i = 0; i < 6; i++) {
    console.log(x);
}
```

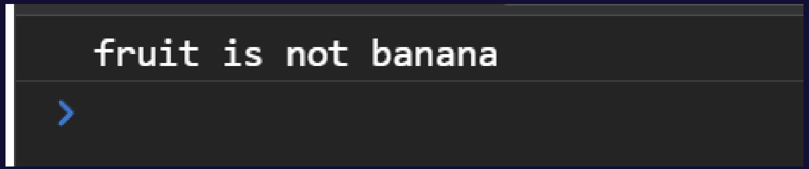
Step 2 continued…

- Let’s look at this example



```
let fruit = 'banana';

if (fruit === 'Banana') {
    console.log('fruit is banana');
} else {
    console.log('fruit is not banana');
}
```

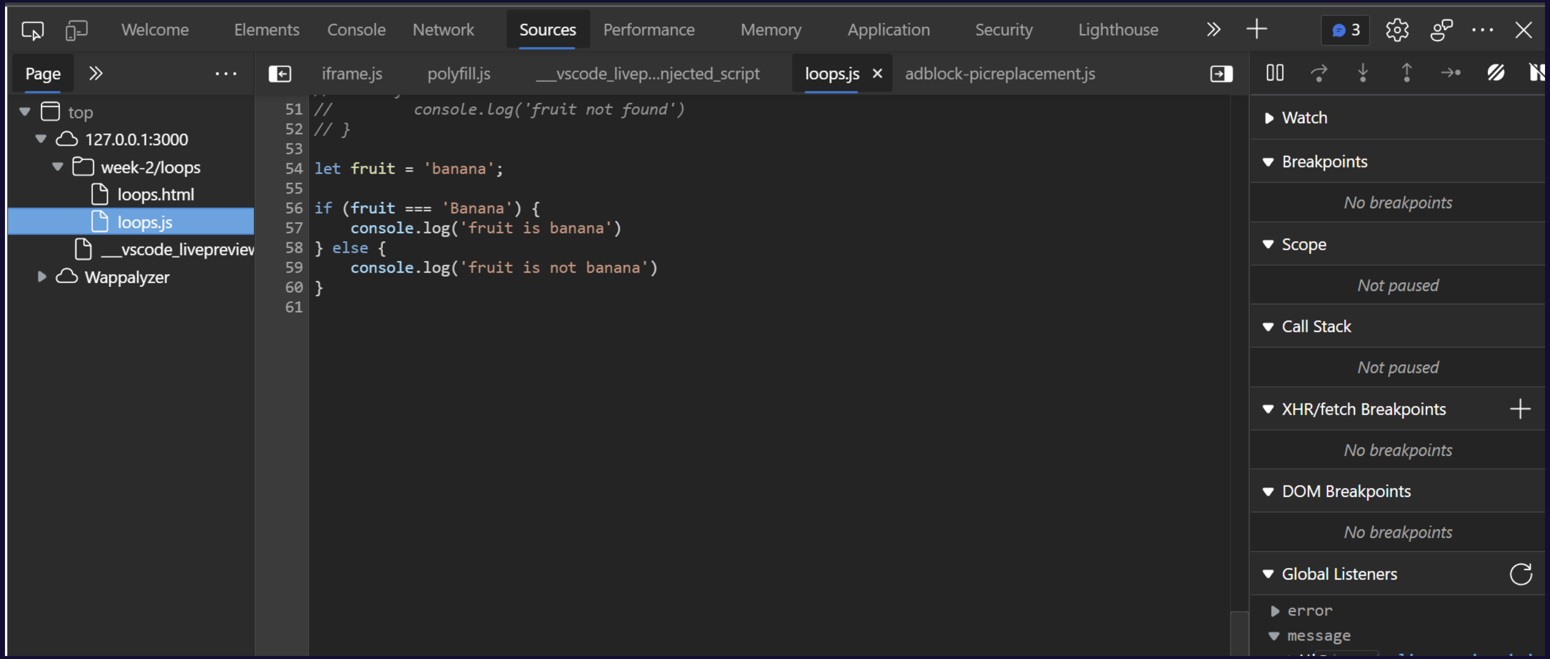
- We set the value of fruit to be ‘banana’. Then we check if fruit equals ‘Banana’, if it does then we want to log a message in the console, if it doesn’t then we want to log a different message.
- Let’s look at the output…



```
fruit is not banana
>
```

Step 2 continued…

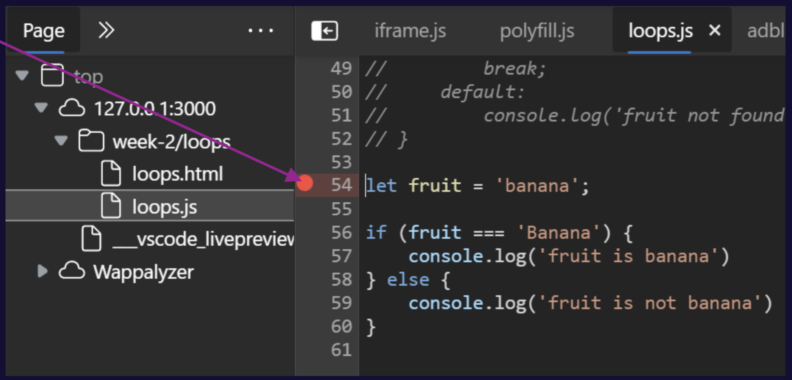
- That’s weird, we expected the output to be “fruit is banana” but we got something different.
- There’s no error in the console to help us out, so let’s try and use the debugger.



```
//          console.log('fruit not found')
// }

let fruit = 'banana';

if (fruit === 'Banana') {
    console.log('fruit is banana')
} else {
    console.log('fruit is not banana')
}
```

Step 2 continued...

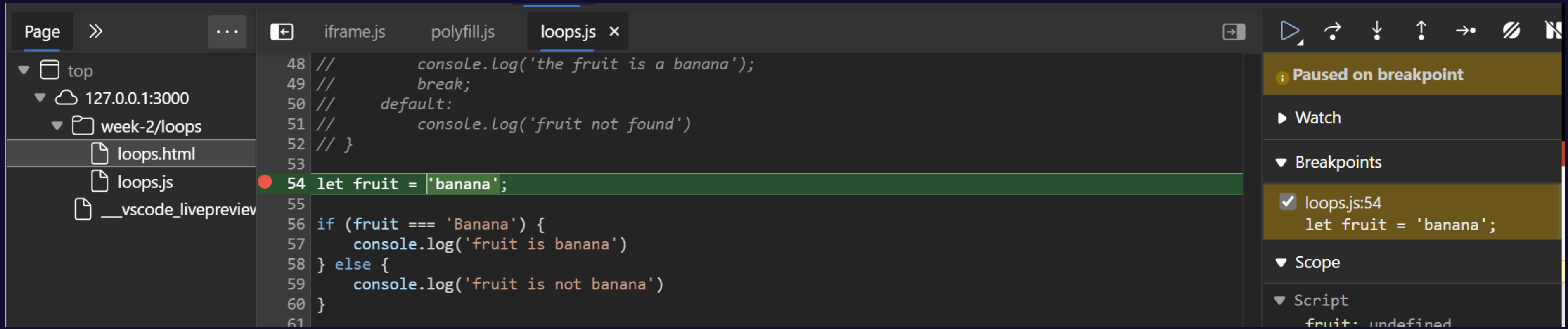
- Our code starts on line 54, so let's set a breakpoint on there.
- A breakpoint is a point in our code that pauses the program and allows us to move through it step by step and see some details about what's going on
- To set a breakpoint we click on the row we want to set the breakpoint on



```
49 //         break;
50 //     default:
51 //         console.log('fruit not found
52 // }
53
54 let fruit = 'banana';
55
56 if (fruit === 'Banana') {
57     console.log('fruit is banana')
58 } else {
59     console.log('fruit is not banana')
60 }
61
```

Step 2 continued…

- Now if we go back to our code in VS Code and hit save in our js file, it should reload the page and the program will stop at our breakpoint.
- If you see your line turn green and a message in the top right corner saying “Paused on breakpoint” then we know our breakpoint has worked.



```
//          console.log('the fruit is a banana');
//          break;
//      default:
//          console.log('fruit not found')
// }

let fruit = 'banana';

if (fruit === 'Banana') {
    console.log('fruit is banana')
} else {
    console.log('fruit is not banana')
}
```

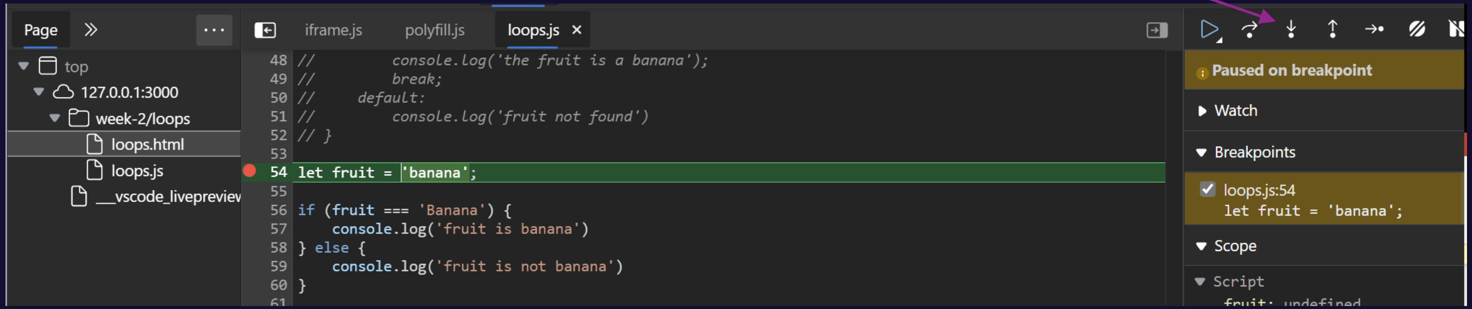
Paused on breakpoint

☑ loops.js:54
let fruit = 'banana';

fruit: undefined

Step 2 continued…

- We can now step through our code and see what's going wrong
- To step through the code we can use this button



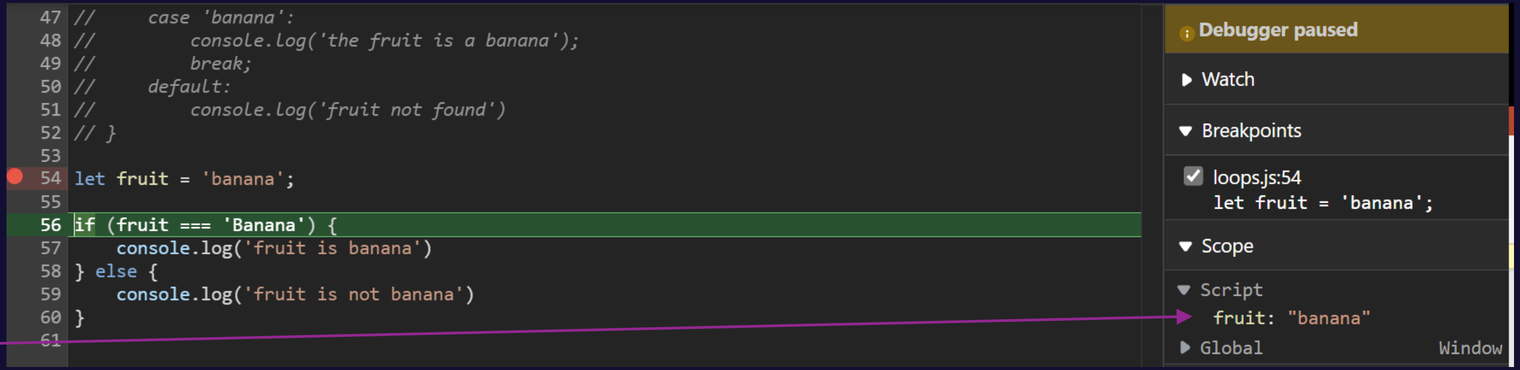
```
//         console.log('the fruit is a banana');
//         break;
//     default:
//         console.log('fruit not found')
// }

let fruit = 'banana';

if (fruit === 'Banana') {
    console.log('fruit is banana')
} else {
    console.log('fruit is not banana')
}
```

Paused on breakpoint

☑ loops.js:54
`let fruit = 'banana';`

`fruit: undefined`

- We can now see that our green line has moved down and on the left we can see that fruit now = banana



```
//     case 'banana':
//         console.log('the fruit is a banana');
//         break;
//     default:
//         console.log('fruit not found')
// }

let fruit = 'banana';

if (fruit === 'Banana') {
    console.log('fruit is banana')
} else {
    console.log('fruit is not banana')
}
```

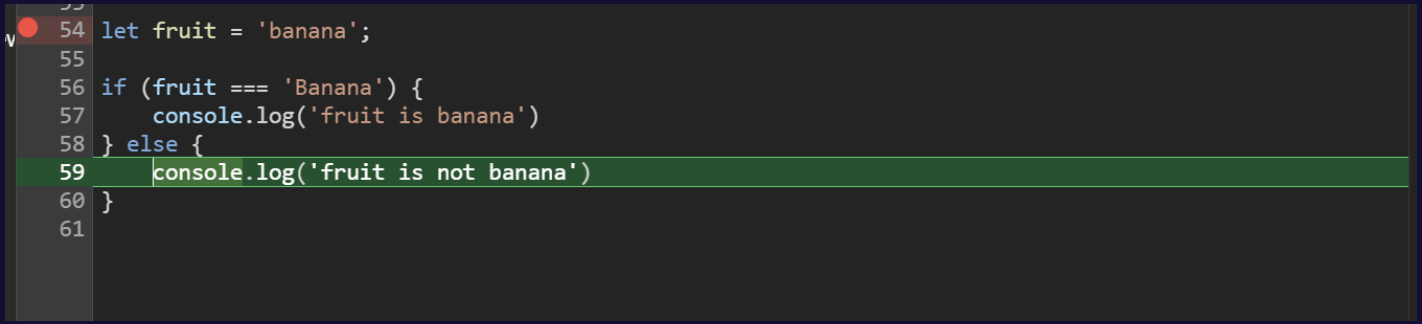
Debugger paused

☑ loops.js:54
`let fruit = 'banana';`

`fruit: "banana"`

Step 2 continued…

- Now we expect the next step to take us to the next line down.
- Let’s go to the next step and see what happens.



```
54 let fruit = 'banana';
55
56 if (fruit === 'Banana') {
57     console.log('fruit is banana')
58 } else {
59     console.log('fruit is not banana')
60 }
61
```

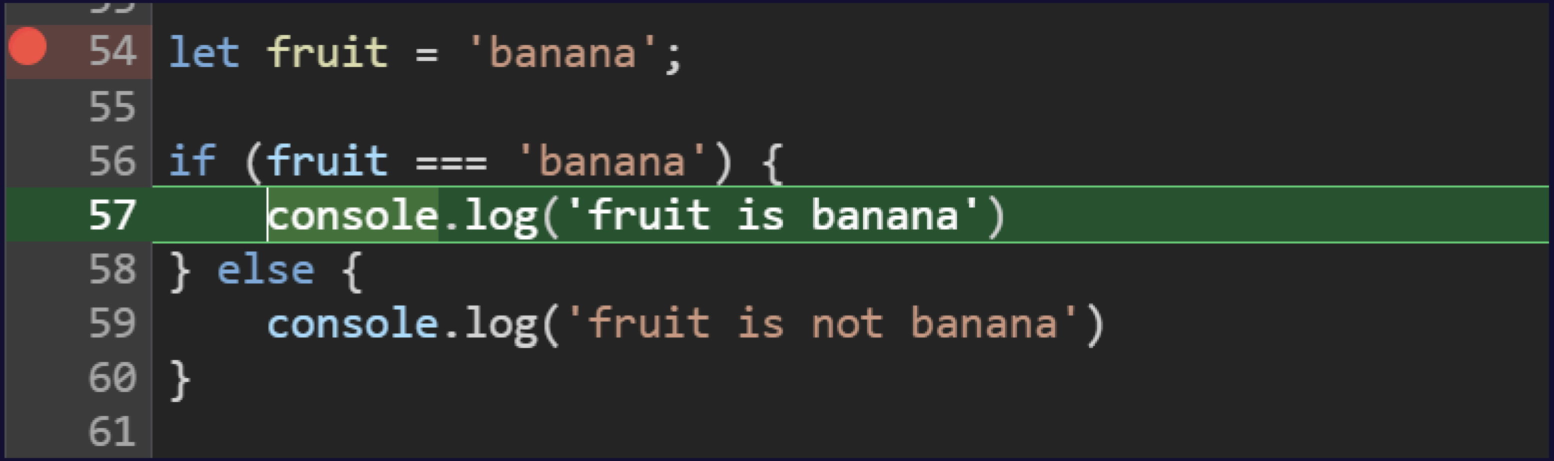
- We can see that the console.log(‘fruit is banana’) line was skipped, so we now know that the issue is on line 56.

Step 2 continued…

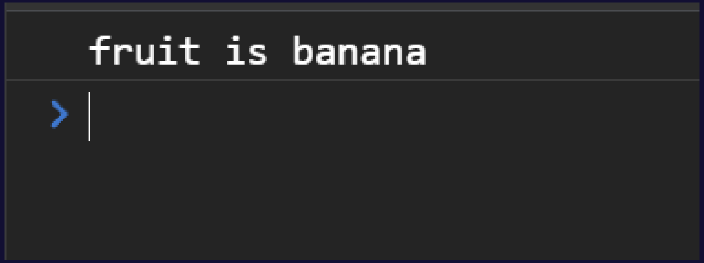
- Taking a closer look at line 56 we can see that ‘Banana’ is spelt with a capital B and that’s what is causing us issues.
- Let’s fix it and try again.

Step 2 continued…

- Now when we step through our code, we can see that console.log(‘fruit is banana’) can now be reached and logs the expected output to the console.



```
let fruit = 'banana';

if (fruit === 'banana') {
    console.log('fruit is banana')
} else {
    console.log('fruit is not banana')
}

```



```
fruit is banana
>
```

Actual Step 3

- If after debugging you still aren’t able to figure it out, try searching up your issue on google to see if someone else has had a similar issue and look at what they did to solve it.
- Good resources for this are GitHub and stack overflow

Step 4

- If all else fails try and ask a friend or a more senior developer for help.

Thank you | **Reuben Simpson**