@CODE.CLASH

JavaScript Exception - Handling

try_catch,finally,throw Statement

Next →

Hey Everyones 👋

In this post, you will learn about the try...catch...finally statements to handle exceptions in JavaScript

Do Like, save and Share This Post If You Found This Helpful.

Next →

JavaScript exceptions

- Errors that occur during runtime are called **exceptions**.

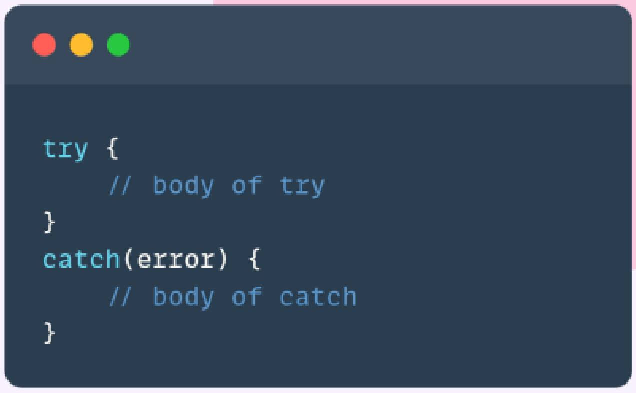
Types of Errors

- There can be two types of errors in the code:
- **Syntax Error:** Error in the syntax. For example, if you write consol.log('your result');
- **Runtime Error:** This type of error occurs during the execution of the program. For example,
- calling an invalid function or a variable.

NEXT →

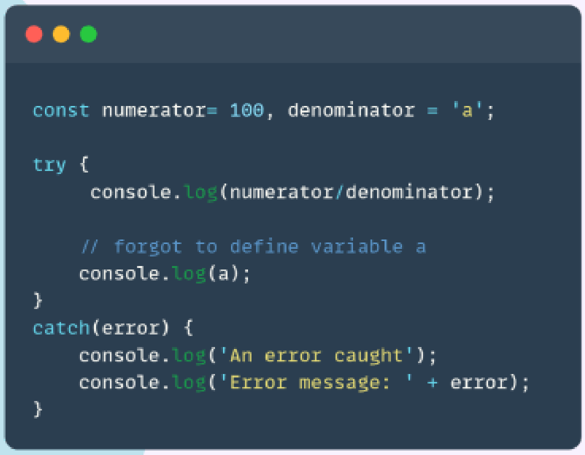
try...catch

- The try...catch statement is used to handle the exceptions.



```
try {
    // body of try
}
catch(error) {
    // body of catch
}
```

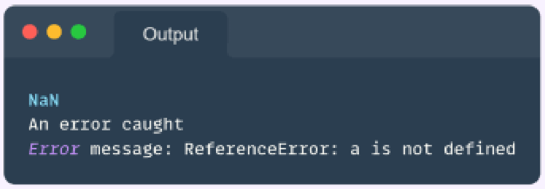
- The main code is inside the try block.
- While executing the try block, if any error occurs, it goes to the catch block.



```
const numerator= 100, denominator = 'a';

try {
     console.log(numerator/denominator);

    // forgot to define variable a
    console.log(a);
}
catch(error) {
    console.log('An error caught');
    console.log('Error message: ' + error);
}
```



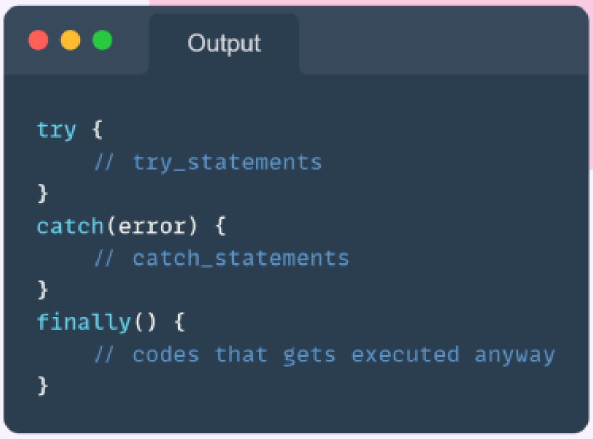
```
NaN
An error caught
Error message: ReferenceError: a is not defined
```

NEXT →

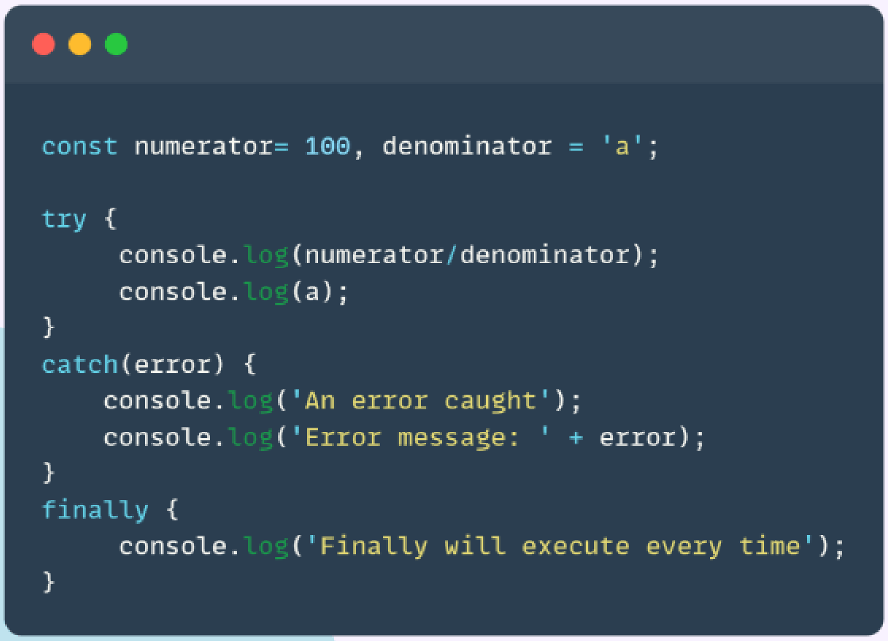
@CODE.CLASH

try...catch...finally

- You can also use the try...catch...finally statement to handle exceptions.
- The finally block executes both when the code runs successfully or if an error occurs.

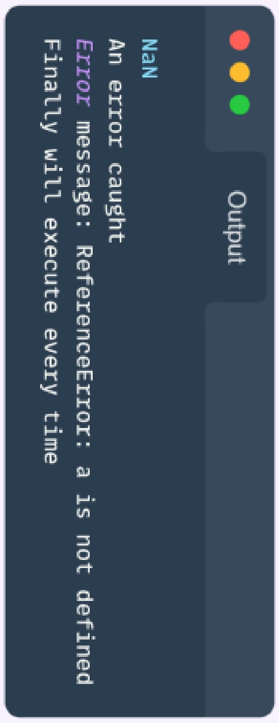

Output

```
try {
    // try_statements
}
catch(error) {
    // catch_statements
}
finally() {
    // codes that gets executed anyway
}
```



```
const numerator= 100, denominator = 'a';

try {
     console.log(numerator/denominator);
     console.log(a);
}
catch(error) {
    console.log('An error caught');
    console.log('Error message: ' + error);
}
finally {
     console.log('Finally will execute every time');
}
```


Output

```
NaN
An error caught
Error message: ReferenceError: a is not defined
Finally will execute every time
```

NEXT →

throw statement

- The syntax of throw statement is:

```
throw expression;
```

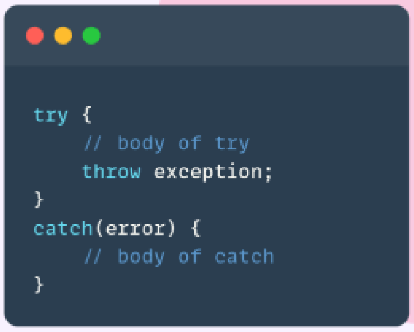
- Here, expression specifies the value of the exception.
- For example,

```
const number = 5;
throw number/0; // generate an exception when divided by 0
```

NEXT →

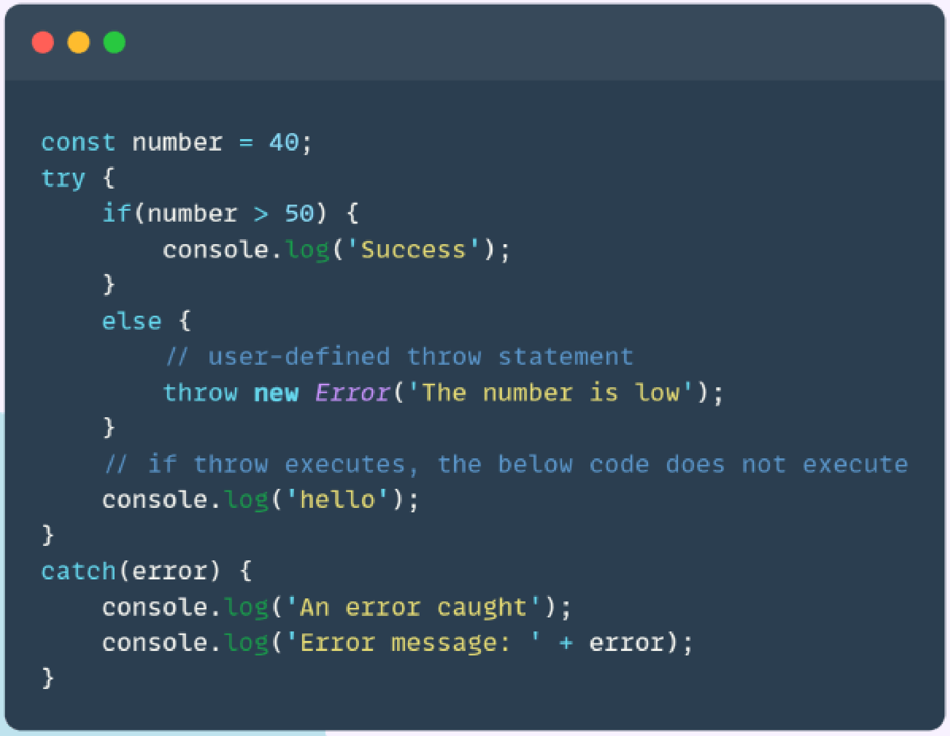
throw with try...catch

- The syntax of try...catch... throw is:

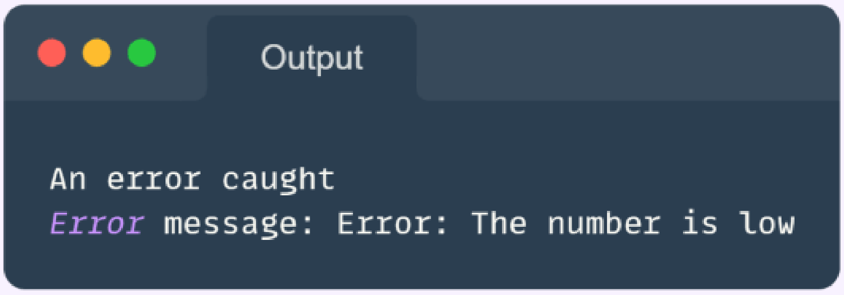


```
try {
    // body of try
    throw exception;
}
catch(error) {
    // body of catch
}
```

- Example :



```
const number = 40;
try {
    if(number > 50) {
        console.log('Success');
    }
    else {
        // user-defined throw statement
        throw new Error('The number is low');
    }
    // if throw executes, the below code does not execute
    console.log('hello');
}
catch(error) {
    console.log('An error caught');
    console.log('Error message: ' + error);
}
```

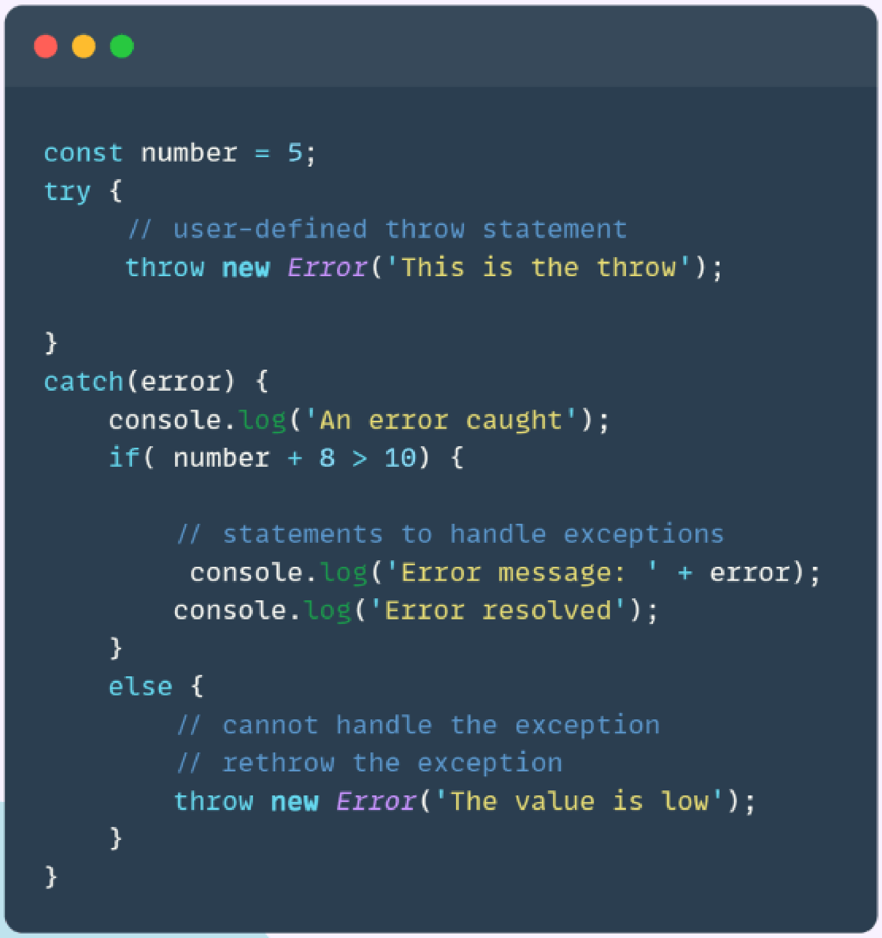


```
An error caught
Error message: Error: The number is low
```

Next →

Rethrow an Exception

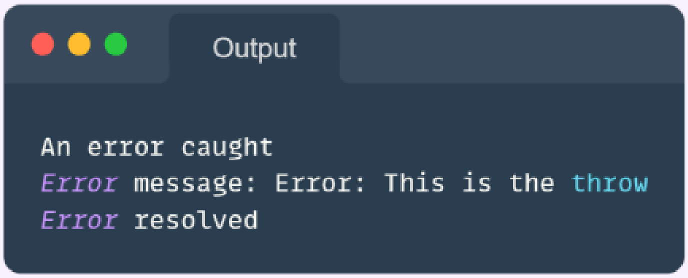
- You can also use throw statement inside the catch block to rethrow an exception.



```
const number = 5;
try {
     // user-defined throw statement
     throw new Error('This is the throw');

}
catch(error) {
    console.log('An error caught');
    if( number + 8 > 10) {

        // statements to handle exceptions
         console.log('Error message: ' + error);
        console.log('Error resolved');
    }
    else {
        // cannot handle the exception
        // rethrow the exception
        throw new Error('The value is low');
    }
}
```



Output

```
An error caught
Error message: Error: This is the throw
Error resolved
```

NEXT

Best Of Luck :)

NEXT →

@Code.Clash

THANKS FOR YOUR ATTENTION

Code.Clash

Imtiyaz Nandasaniya

LIKE AND SAVE IT

FOR LATER