Error Handling

Exception is an unwanted or unexpected event, which occurs during the execution of a program i.e at run time, that disrupts the normal flow of the program’s instructions.

For example referencing a variable or a method that is not defined can cause an exception.

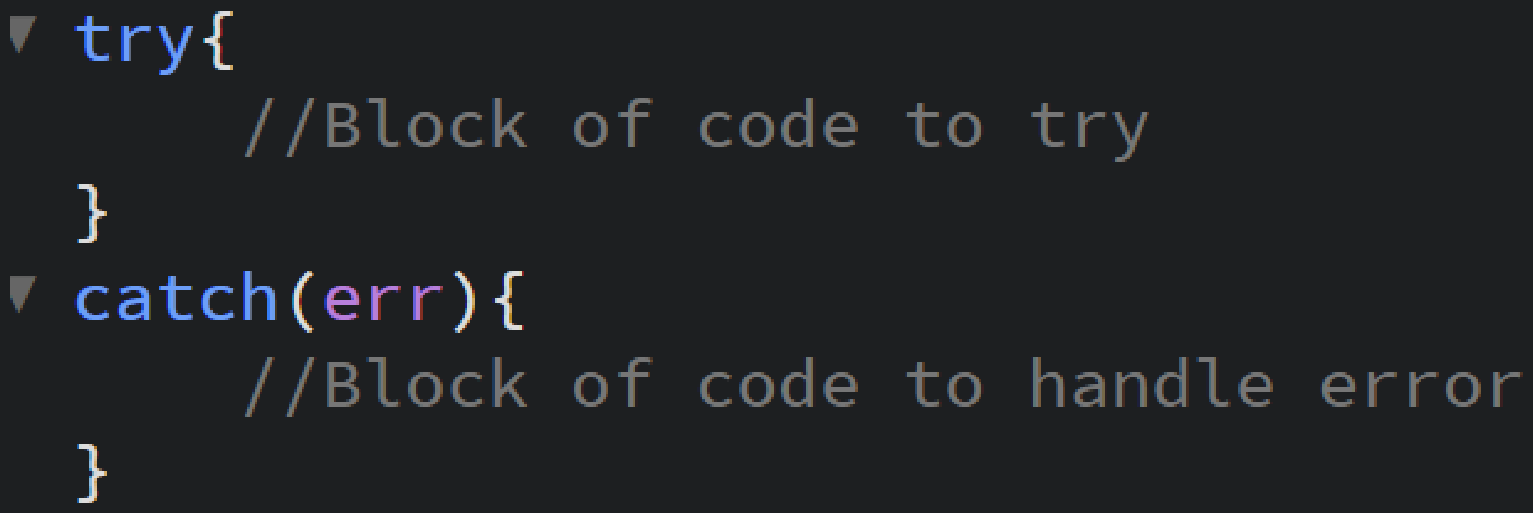
Error Handling

- try
- catch
- throw
- finally

try…catch block

- The JavaScript statements that can possibly cause exceptions should be wrapped inside a try block.
- When a specific line in the try block causes an exception, the control is immediately transferred to the catch block skipping the rest of the code in the try block.

try…catch block



```
try{
        //Block of code to try
}
catch(err){
        //Block of code to handle error
}
```

finally block

- Finally block is guaranteed to execute irrespective of whether there is an exception or not. It is generally used to clean and free resources that the script was holding onto during the program execution. For example, if your script in the try block has opened a file for processing, and if there is an exception, the finally block can be used to close the file.

finally block

```
try{
    //Block of code to try
}
catch(err){
    //Block of code to handle errors
}
finally{
    //Block of code to be executed regardless of the try/catch result
}
```

throw statement

- The throw statement allows you to create a custom error.
- The exception can be a JavaScript String, a Number, a Boolean, or an Object.

```
throw "Too big";  //Throw a text
throw 500;  //Throw a number
throw true;  //Throw a boolean
throw {toString: function(){return "I am on abject!"}};  //Throw a object
```