16. JavaScript Errors - Throw and Try to Catch

The **try** statement lets you test a block of code for errors.

The **catch** statement lets you handle the error.

The **throw** statement lets you create custom errors.

16.1. Errors Will Happen!

When the JavaScript engine is executing JavaScript code, different errors can occur:

It can be syntax errors, typically coding errors or typos made by the programmer.

It can be misspelled or missing features in the language (maybe due to browser differences).

It can be errors due to wrong input, from a user, or from an Internet server.

And, of course, it can be many other unforeseeable things.

16.2. JavaScript Throws Errors

When an error occurs, when something goes wrong, the JavaScript engine will normally stop, and generate an error message.

The technical term for this is: JavaScript will **throw** an error.

16.3. JavaScript try and catch

The **try** statement allows you to define a block of code to be tested for errors while it is being executed.

The **catch** statement allows you to define a block of code to be executed, if an error occurs in the try block.

The JavaScript statements **try** and **catch** come in pairs.

Syntax

```
try
  {
  //Run some code here
  }
catch(err)
  {
  //Handle errors here
  }
```

Examples

In the example below we have deliberately made a typo in the code in the try block.

The catch block catches the error in the try block, and executes code to handle it:

Example

```
<!DOCTYPE html>
<html>
<head>
<script>
var txt="";
function message()
{
try
  {
  adddlert("Welcome guest!");
  }
catch(err)
  {
  txt="There was an error on this page.\n\n";
  txt+="Error description: " + err.message +
"\n\n";
  txt+="Click OK to continue.\n\n";
  alert(txt);
  }
}
</script>
</head>

<body>
<input type="button" value="View message"
onclick="message()">
</body>

</html>
```

16.4. The Throw Statement

The throw statement allows you to create a custom error.

The correct technical term is to create or **throw an exception**.

If you use the throw statement together with try and catch, you can control program flow and generate custom error messages.

Syntax

```
throw exception
```

The exception can be a JavaScript String, a Number, a Boolean or an Object.

Example

This example examines the value of an input variable. If the value is wrong, an exception (error) is thrown. The error is caught by the catch statement and a custom error message is displayed:

```
<script>
function myFunction()
{
try
  {
  var x=document.getElementById("demo").value;
  if(x=="")     throw "empty";
  if(isNaN(x)) throw "not a number";
  if(x>10)      throw "too high";
  if(x<5)       throw "too low";
  }
catch(err)
  {
  var y=document.getElementById("mess");
  y.innerHTML="Error: " + err + ".";
  }
}
</script>

<h1>My First JavaScript</h1>
<p>Please input a number between 5 and 10:</p>
<input id="demo" type="text">
<button type="button" onclick="myFunction()">Test
Input</button>
<p id="mess"></p>
```

Note that the example above will also throw an error if the getElementById function fails.