JAVASCRIPT QUIRKS

JAVASCRIPT QUIRKS

ARE ANNOYING WHEN YOU FIRST ENCOUNTER THEM,
BUT OVER TIME THEY BECOME AMUSING, EVEN

STRANGE SPECIAL VALUES

STRINGS AND NUMBERS

STRANGE COMPARISON OPERATORS

TRUTHY AND FALSEY

JAVASCRIPT QUIRKS

ARE ANNOYING WHEN YOU FIRST ENCOUNTER THEM,
BUT OVER TIME THEY BECOME AMUSING, EVEN

STRANGE SPECIAL VALUES

STRINGS AND NUMBERS

STRANGE COMPARISON OPERATORS

TRUTHY AND FALSEY

STRANGE SPECIAL VALUES

`undefined`

INDICATE THAT A VALUE DOES NOT EXIST

`null`

A SPECIFIC VALUE THAT DOES EXIST, BUT IS NULL

`NaN`

"NOT A NUMBER" (BUT STILL A NUMBER,

`Infinity`

A SPECIAL VALUE, BIGGER THAN ANY NUMBER

EXAMPLE 31: UNDERSTANDING `undefined`

EXAMPLE 31: UNDERSTANDING `undefined`

`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

`undefined` IS NOT AN ERROR OR EXCEPTION, IT IS AN ACTUAL VALUE THAT YOU CAN TEST FOR

```
if(someVariable == undefined)
```

`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

CASE #1: A VARIABLE HAS BEEN DECLARED, BUT NOT INITIALISED, ITS VALUE IS `undefined`

```
var someVariable;  //undefined
```

`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

CASE #2: A FUNCTION HAS NO RETURN VALUE, BUT YOU ATTEMPT TO ACCESS IT - ITS VALUE IS `undefined`

```
function getPerimeter(rectangle) {
  var perimeter = 2*(rectangle.length + rectangle.bre
  // missing return statement
  // return perimeter;
}
var someVariable = getPerimeter(rectangle);
```

`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

CASE #2: A FUNCTION HAS NO RETURN VALUE, BUT YOU ATTEMPT TO ACCESS IT - ITS VALUE IS `undefined`

```
function getPerimeter(rectangle) {
  var perimeter = 2*(rectangle.length + rectangle.bre
  // missing return statement
  // return perimeter;
}
var someVariable = getPerimeter(rectangle);
```

`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

CASE #2: A FUNCTION HAS NO RETURN VALUE, BUT YOU ATTEMPT TO ACCESS IT - ITS VALUE IS `undefined`

```
function getPerimeter(rectangle) {
  var perimeter = 2*(rectangle.length + rectangle.bre
  // missing return statement
  // return perimeter;
}
var someVariable = getPerimeter(rectangle);
```

`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

CASE #3: A VARIABLE DOES NOT EXIST, AND YOU ATTEMPT TO USE ITS VALUE, IT WILL BE `undefined`

```
console.log(“Value of a non-existent variable = “ +
noSuchVar);
```

`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

CASE #4: A FUNCTION DOES NOT EXIST, AND YOU ATTEMPT TO CALL IT - AN ERROR RESULTS - NOT `undefined`

```
nonExistentFunction();
```

```
▶ Uncaught ReferenceError: nonExistentFunction is not defined
```

CASE #3: A VARIABLE DOES NOT EXIST, AND YOU ATTEMPT TO USE ITS VALUE, IT WILL BE `undefined`

```
console.log("Value of a non-existent variable = " +
noSuchVar);
```

CASE #4: A FUNCTION DOES NOT EXIST, AND YOU ATTEMPT TO CALL IT - AN ERROR RESULTS - NOT `undefined`

```
nonExistentFunction();
```

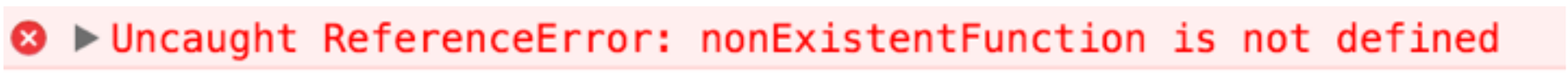
```
▶ Uncaught ReferenceError: nonExistentFunction is not defined
```

CASE #3: A VARIABLE DOES NOT EXIST, AND YOU ATTEMPT TO USE ITS VALUE, IT WILL BE `undefined`

```
console.log("Value of a non-existent variable = " +
noSuchVar);
```

CASE #4: A FUNCTION DOES NOT EXIST, AND YOU ATTEMPT TO CALL IT - AN ERROR RESULTS - NOT `undefined`

```
nonExistentFunction();
```



CASE #3: A VARIABLE DOES NOT EXIST, AND YOU ATTEMPT TO USE ITS VALUE, IT WILL BE `undefined`

```
console.log("Value of a non-existent variable = " +
noSuchVar);
```

CASE #4: A FUNCTION DOES NOT EXIST, AND YOU ATTEMPT TO CALL IT - AN ERROR RESULTS - NOT `undefined`

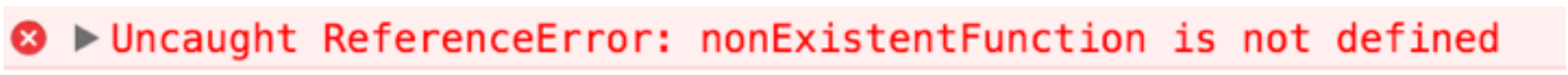
```
nonExistentFunction();
```

CASE #3: A VARIABLE DOES NOT EXIST, AND YOU ATTEMPT TO USE ITS VALUE, IT WILL BE `undefined`

```
console.log("Value of a non-existent variable = " +
noSuchVar);
```

CASE #4: A FUNCTION DOES NOT EXIST, AND YOU ATTEMPT TO CALL IT - AN ERROR RESULTS - NOT `undefined`

```
nonExistentFunction();
```



CASE #3: A VARIABLE DOES NOT EXIST, AND YOU ATTEMPT TO USE ITS VALUE, IT WILL BE `undefined`

```
console.log("Value of a non-existent variable = " +
noSuchVar);
```

CASE #4: A FUNCTION DOES NOT EXIST, AND YOU ATTEMPT TO CALL IT - AN ERROR RESULTS - NOT `undefined`

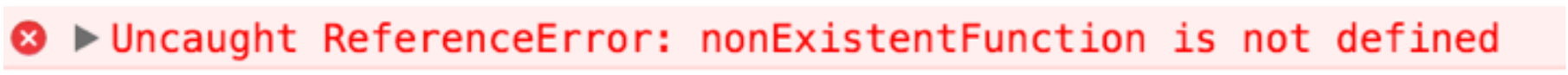
```
nonExistentFunction();
```

CASE #3: A VARIABLE DOES NOT EXIST, AND YOU ATTEMPT TO USE ITS VALUE, IT WILL BE `undefined`

```
console.log("Value of a non-existent variable = " +
noSuchVar);
```

CASE #4: A FUNCTION DOES NOT EXIST, AND YOU ATTEMPT TO CALL IT - AN ERROR RESULTS - NOT `undefined`

```
nonExistentFunction();
```



`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

CASE #5: YOU ATTEMPT TO ACCESS A NON-EXISTENT PROPERTY OF AN OBJECT - `undefined`

```
console.log(rectangle.radius);
```

BTW, THIS IS TRUE WHETHER THE PROPERTY NEVER EXISTED, OR EVEN IF IT DID EXIST, BUT WAS DELETED.

`undefined` IS A SPECIAL VALUE FOR VALUES THAT DO NOT EXIST.

CASE #6: YOU CALL THE `typeof` OPERATOR ON SOMETHING THAT IS `undefined` ALSO - `undefined`

```
 console.log("Type of undefined is ... wait for it.. " + (typeof unInitialisedVariable))
```

```
Type of undefined is ... wait for it.. undefined
```