Truth and Equality

Truth and Equality

- Truthy and falsy values
- Coercion
- Equality
- Short-circuiting expressions

Falsy values

- false
- null
- undefined
- NaN
- ""
- 0

https://developer.mozilla.org/en-US/docs/Glossary/Falsy

Falsy works as false

```
var name = prompt("Gimme your name");

if (!name) {
  name = prompt("Oh, c'mon!");
}
```

Falsy terminates the loop

```
var arr = [1, 2, 3, 4, 5];
var i = arr.length;

do {
    console.log(arr[arr.length - i]);
} while (--i);
```

If it ain't falsy it's truthy

https://developer.mozilla.org/en-US/docs/Glossary/Truthy

Truthy

```
var name = "Eirik";
var job = "Programmer";

if (name && job) {
  console.log("Thanks for filling out the application");
}
```

Whats the value of this expression?

```
var name = "Eirik";
var job = "Programmer";

assertEquals(name && job, job);
```

..and this?

```
var name = "Eirik";
var job = "Programmer";

assertEquals(!name, false);
```

We'll go more in depth on this

Coercion

- Implicit cast (happens "by itself")
- Not always obvious
- Applies to == and !=
- Applies to most operators
- Applies to some built-in functions

https://developer.mozilla.org/en-US/docs/Glossary/Type_coercion

Equality - Same Types

```
assert(undefined == undefined);
assert(null == null);
assert(3 == 3);
assert(+0 == -0);
assert(NaN != NaN);
assert("string" == "string");
assert(true == true);
assert(false == false);
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Equality_comparisons_and_sameness

Object Equality

== IS TRUE WHEN COMPARING OBJECT TO ITSELF

```
var obj = { name: "Eirik" };

assert(obj == obj);
```

Object Equality

== IS *NOT* TRUE WHEN COMPARING OBJECTS WITH 'SIMILAR/EQUAL CONTENT'

```
var obj = { name: "Eirik" };
var copy = { name: "Eirik" };

assert(obj != copy);
```

This also applies to arrays!

Coercion: Null and Undefined

```
assert(null == undefined);

assert(undefined == null);
```

Coercion: Null and Undefined

NULL AND UNDEFINED
ARE THE ONLY VALUES
== NULL

Did we receive a usable argument?

```
function add(a, b) {
  if (a == null || b == null) {
    return;
  }

  // ...
}

add(); // 'Stopped'
```

Number == String

Coerce string into number

```
assert(3 == "3");

// Interpreted as
// assert(3 == Number("3"));

assert("3" == 3);

// Interpreted as
// assert(Number("3") == 3);
```

Boolean: Coerced to numbers (!)

```
assert(true == 1);

// Interpreted as
// assert(Number(true) == 1);

Number(true) //=> 1
Number(false) //=> 0

assert(2 != true); // 2 != 1
```

Why?

```
assert(true != "Eirik");
assert(1 != "Eirik");
assert(1 != NaN);
```

Coercion in operators

Addition/Concatenation

- ToPrimitive(a), ToPrimitive(b)
- If either is a string: return string
- Otherwise, ToNumber(a) + ToNumber(b)

Addition/Concatenation

```
assert("Hey" + 3 == "Hey3");

assert(42 + "Hitchhiker" == "42Hitchhiker");

assert(40 + 2 == 42);
```

What's the result?

```
297 + true

// 298
```

Unary plus operator

Convert operand to number

```
var str = "42";

assert(typeof +str == "number");
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Unary_plus

Logical not operator

Convert operand to boolean and flip it

```
var str = "42";

assert(!str == false);
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical_NOT

Logical not operator x2

Convert operand to boolean

```
// Truthy
var name = "Eirik";
var isNameSet = !!name;
assert(isNameSet == true);
// Falsy
var person;
var isPersonSet = !!person;
assert(isPersonSet == false);
```

Strict Equality

!== and === operators

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Strict_equality

A === B

- false if typeof a != typeof b
- true for null === null
- true for undefined === undefined
- true for same number
- true for same string of characters
- true for same object

Strict Equality

```
assert(3 === 3);
assert("Hello" === "Hello");
assert(3 !== "3");

var person = { name: "Eirik" };
assert(person === person);
```

Short-Circuiting Expressions

&&

- Evaluates from left to right
- Stops evaluating on first *falsy* value
- Returns the last evaluated value

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical_AND

Defensively Accessing Object Properties

```
function addClassName(el, className) {
  el && el.className && (el.className += " " + className);
}
```

Better Readability

```
function addClassName(el, className) {
  if (el && el.className) {
    el.className += " " + className;
  }
}
```

||

- Evaluates from left to right
- Stops evaluating on first *truthy* value
- Returns the last evaluated value

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical_OR

||

```
function addClassName(el, className) {
  if (!el || !el.className) {
    return;
  }

  el.className += " " + className;
}
```

Combining && and ||

```
function prettyPrint(str, options) {
    var color = options && options.color || "#000000";

    // ...
}
prettyPrint("Porsche"); // Printed in black
prettyPrint("Lamborghini", { indent: 2 }); // ...black
prettyPrint("Ferrari", { color: "red" }); //
```

Summary

- Truthy and falsy
- Coercion in == and !=
- ToPrimitive in +, -, *, / and others
- valueOf and toString
- === and !==
- && and ||

Tasks

Continue with "exercises-language-fundamentals" from Canvas