

CMSC388B

Web Application Development with JavaScript



JavaScript IV

Department of Computer Science

University of MD, College Park

Slides material developed by Ilchul Yoon, Nelson Padua-Perez

typeof and instanceof operator

- **typeof** (returns string)
 - Returns “object” for all reference types
 - Name of primitive (e.g., boolean) for primitive types
- **instanceof** operator (returns boolean)
 - Returns **true** if a value is an instance of the specified type and false otherwise
 - **instanceof** can identify **inherited** types
- **Note: every object is an instance of Object**
- Checking if an object is an array or not
 - Although **instanceof** can identify arrays, use **Array.isArray()** instead, as **instanceof** will not work in all cases
- **Approach to follow**
 - Use **typeof** first, then **instanceof** if it is an object
- **Example:** TypeOfInstanceOf.html

let and const

```
x = 5; // Assign 5 to x

elem = document.getElementById("demo");
elem.innerHTML = x;

var x; // Declare x
```

- **var** declarations are moved to the top implicitly
 - So, a variable can be declared after it is used (e.g., assigning a value to it) -- called **Hoisting**
 - Only declaration will be hoisted, not initialized value
 - **Example:** Hoisting.html
- **let** replaces **var** for variable declarations and provides block scoping
 - **Does not allow hoisting!**
 - **Example:** BlockScope.html
- **const** allows you to declare a constant variable that has block scope
 - **Example:** Const.html
- **No block scope and no const before ES6**

Null and undefined

- **null**
 - A value indicating no value (nothing)
 - Has type “object”
- **undefined**
 - Value associated with **uninitialized** variables
 - Has type “undefined”
 - **Cases where undefined appears**
 - » `let x; /* In a function */`
 - » **undefined** is returned by a function when no explicit value is returned (**IMPORTANT case**) - You forgot a return
 - » Value associated with object properties that do not exist
- `==` considers **null** and **undefined** equal
- `===` considers **null** and **undefined** different
- **Example:** NullAndUndefined.html

Truthy vs. Falsy

- A **falsy** - **Definition**: value that is considered **false** in a boolean context
 - Falsy values are:
 - » false
 - » 0
 - » ""
 - » null
 - » undefined
 - » NaN
- A **truthy** value is:
 - A value that is considered **true** in a boolean context
 - All values are truthy unless they are defined as falsy
- **Example**: TruthyFalsy.html
 - Notice use of the library lodash.min.js included using <script></script>

Numeric Values

- **Infinity** is a global property
 - Default: `Number.POSITIVE_INFINITY`
- **isFinite()**
 - Returns **false** if the argument is NaN, positive/negative infinity;
 - Otherwise, it returns true.
- **isFinite() vs. Number.isFinite()**
 - `isFinite()` function converts the value to a Number, then tests it
 - `Number.isFinite()` does not convert the values to a Number and will return false for any value that is not of the type Number
- **Example:** `NumericValues.html`