CMSC335

Web Application Development with JavaScript

Objects, Operators

)epartment of Computer Science

University of MD, College Park

Slides material developed by Ilchul Yoon, Nelson Padua-Perez

Extending Built-in classes

- **Example:** ExtendedArray.html

Chaining (?) Operator

- **Chaining (?) operator** - placed after a property is accessed, which prevents access to the next level if the property does not exist
- **Example:** ChainingOp.html

Revising &&, || / Coalescing Operator

- **logical and (&&**) - if the first operand evaluates to false, the first operand is returned; otherwise, the second operand is returned
  - Whatever makes it false is the value returned
- **Logical or (||)** - if the first operand evaluates to true, the first operand is returned; otherwise, the second operand is returned
  - Whatever makes it true is the value returned
- **nullish coalescing operator (??)** - returns the value of the second operand if the left operator is **null** or **undefined**. **Example:** x ?? 18
- **Example:** AndOrCoalescing.html

Nonextensible and Sealed Obj

- In JavaScript, you can add properties and methods to an object at any time (extensible)
- You can restrict this behavior by using Object.preventExtensions()
- What if you don't want properties deleted as well?
  - **Seal the object**
  - By sealing an object, you create the same abstraction we have in Java class definitions, where once a class is defined, the class instance variables and methods are set
- **Example:** ExtensibleSealed.html

Freeze

- Strictest protection
- Not extensible, sealed, and data properties can not be modified
  - Constant object
- **Example:** Freeze.html

Function Context

- **Problem**
  - **Example:** FunctionContextIncorrect.html to see the state of **ButtonState**
- Approaches to address the problem?
  - Renaming this in function (closure)
  - Use arrow functions
    - » Arrow functions do not have their own this reference; they remember this at the point they are defined
  - Use bind 'this' to the desired object
- **Example:** FunctionContext*A*.html - using closure
- **Example:** FunctionContext*B*.html - using arrow function
- **Example:** FunctionContext*C*.html - using bind