OOP JavaScript- l3g3nd

* Object- Instance of a class
  + Property- Holds value within an object
  + Method- Defines logic within an object
    - Behavior- Multiple methods within an object
  + Factory/Constructor Functions
    - Factory Function creates an object based on contained properties
      * Returns object
        + Uses “Return” 🡪 function *createfactory* (property) {*return* {property, method}}
    - Constructor Function constructs a new object via demanding a new instance of the object
      * Uses “this”
        + Refers to global object by default
        + ALWAYS use “new” operator or the function will assign props/methods to the global “this”

Will throw error

* + - * Uses “new” operator 🡪 const *another* = new Circle(1);
      * Constructors are built into JavaScript
    - Can use either. Use what you prefer.
  + Every object has a constructor property. It references the function used to create that object.
    - new
      * String(); ‘’, “”, ``
      * Boolean(); True, False
      * Number(); 1,2,3
  + Primitives- Copied by their value
    - Key values
      * { x:1 };
  + Objects- Copied by their reference
  + Dot vs Bracket notation
    - circle.location = { x:1 };
* Commit to memory- be able to write on command
  + For Loop
  + Sort Function
    - Used to sort a list of objects based on a key; Name, age, etc.
      * Easily set up when using an integer as the sort by key
      * Could be prudent to add an integer { id: value} to beginning of object property list
    - oldAnimals.sort((a, b) => (a.Age > b.Age) ? 1 : -1);
    - variable.sort((a, b)) => (a.Age > b.Age) ? 1 : -1);
      * Sorts age lowest to highest. Could reverse for opposite effect
  + Push function
    - array.push({Name: animalFax[i].Name, Age: animalFax[i].Age})
      * oldAnimals.push({Name: animalFax[i].Name, Age: animalFax[i].Age})
      * Pushes objects and their properties to an array in object form
      * Enables them to be called by variable[object#}.property
      * Ensure correct spelling of property after the.
      * Can be used in a for loop via iterating over an array and send new objects to a new array based on criteria (certain key value, preferably integer)
        + for (let i = 0; i < animalFax.length; i++) {
        + if (animalFax[i].Age > targetAge) {
        + oldAnimals.push({Name: animalFax[i].Name, Age: animalFax[i].Age})
        + };
        + };
  + Object declaration
    - Declare constant. Open bracket, Declare key:, key value, comma, key:, key value…]:
      * const array = [
      * { Name: "Name", Age: 9 },
      * ];
      * const animalFax = [
      * { Name: "Giraffe", Age: 9 },
      * ];
    - Can declare a variety of key values
    - Allows the array to be accessed as objects and called directly via key.
    - new variable declaration : based on one object in the array
      * newVariableName = objectArray[i].Name;
        + youngestOld = oldAnimals[0].Name;

returns the first value in array based on integer key value

* Coding Notes
* Math
  + Inside of a function
    - let variable;
      * variable = (Mathmatical formula/algorithm)
      * area = (length \* width)
      * perimeter = 2 \* (length + width)
    - return variable
  + Syntax
    - Variable- a container for data values
    - Objects- an instance of a variable that can contain many values
      * Examples
        + Arrays
        + Lists

var car = {type:"Fiat", model:"500", color:"white"};

* + - * Keys and values
        + Written as name/key:value
        + Name/Key: an identifier for a value type

Name:

Age:

Type:

Color:

Model:

* + - * + Value: the value of the key

The information mapped to the key

* + - * + Property: the key:value pair

You can see these when you mouse over an instance of a variable in VS code

Shows the key and the type of value it contains

(property) Age: Number

* + - When using a .Identifier, ALWAYS PUT AFTER SQUARE BRACKETS
      * //console.log(JSON.stringify(animalFax[1].Name)) ALWAYS PUT THE .Identifier AFTER THE SQUARE BRACKETS
    - Object Syntax
      * Arrays
        + Arrays are treated as objects in JavaScript
        + When you want to add a new object to an array, make sure to separate keys and values with a comma

I.e, {Key: Value, Key: Value}

{Name: Ian, Age: 26}

* + - * + oldAnimals.push({Name: animalFax[i].Name, Age: animalFax[i].Age})
  + Arrays
    - Call a new array with var x = new Array();
      * Or var x = []
  + Important concepts
    - Equality (Syntax)
      * = : sets the value of something to a variable
        + youngestOld = oldAnimals[0].Name;
      * == : compares values with conversion
        + “32” == 32 : TRUE

Converts the first value (String) to Number and compares to second value (Number)

* + - * + “32” === 32 : FALSE

Tests absolute equality

Does not convert values for comparison