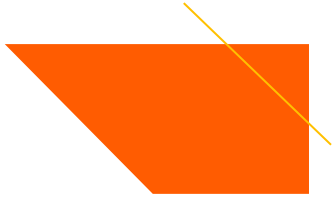
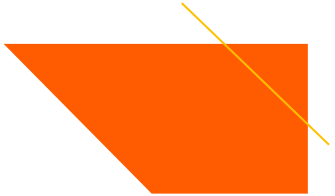
Web Application Development

COSI 152A

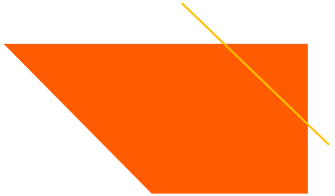


JavaScript Basics



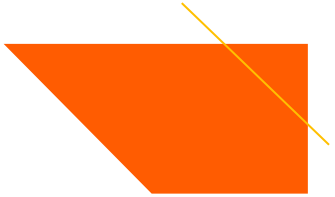
What is JavaScript?

- A lightweight programming language ("scripting language") used to make web pages interactive and dynamic.
- It is an object-oriented and event-driven language.
- NOT related to Java other than by name and some syntactic similarities



JavaScript syntax

- The syntax is like C and Java programming languages
  - But it also has some unique features, such as automatic semicolon insertion.



Running a JavaScript code

- JavaScript is a client-side scripting language that is executed on the web browser.
- Every browser has a JavaScript Engine which runs JavaScript code.
  - E.g., Chrome: V8
- Several ways are there we can write JavaScript code and then run it
  - On the browser console
  - In a HTML file
  - Outside the browser using Node.js

“Hello World!” JavaScript code

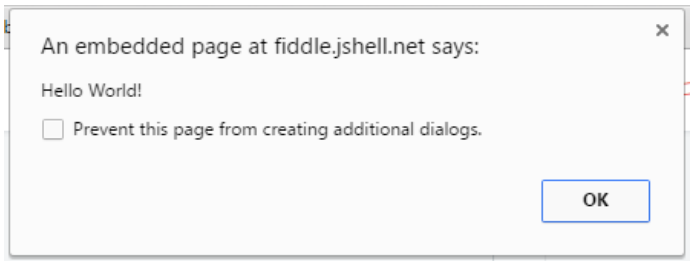
- Let’s write our first script displaying “Hello World!” on the console.

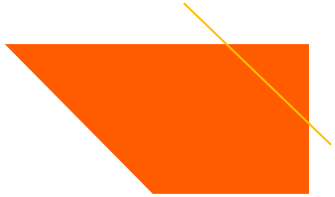
```
<script>
        console.log("Hello World");
</script>
```

A JavaScript statement: alert

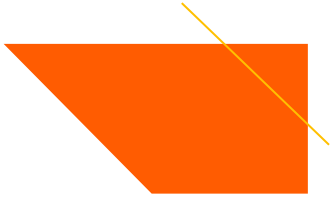
- A JS command that pops up a dialog box with a message

```
alert("Hello World!");
```





Variables



Variables

- Variables are named values and can store any type of JavaScript value.
- Variables are declared with the **var** keyword (case sensitive)
  - Replaced by **let** and **const** with ES6
- Types are not specified, but JS does have types ("loosely typed")
  - Types: Number, Boolean, String, Object (Array , Function) , Null, Undefined
  - Find out a variable's type by calling typeof

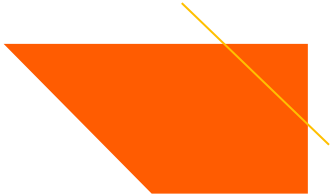
```
var name = expression;

var age = 32;
var weight = 127.4;
```

Using variables

- After you declare a variable, you can reference it by name elsewhere in your code.

```
let x = 100;
let y = x + 102;
console.log(y);
```

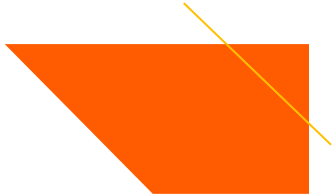
```
OUTPUT
202
```



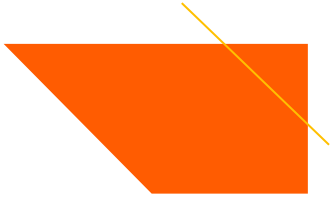
Constants (ES6)

- Also known as "immutable variables”
  - cannot be re-assigned new content
- Always use **const** or **let** when declaring variables
  - If any doubt, use const

```
const name = expression;
let name = expression;

const age = 32;
let weight = 127.4;
```

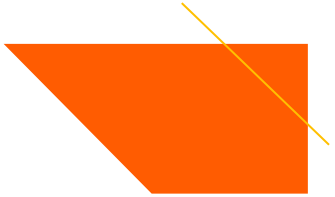


Data Types



Data Types

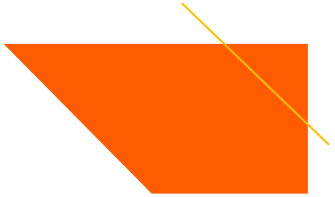
- JavaScript is a dynamically-typed language
  - variables are not bound to a specific data type
  - variable types can change during runtime
- Here are the primary data types in JavaScript:
  1. Number
  2. String
  3. Boolean
  4. Undefined
  5. Null
  6. Symbol
  7. Object (Array , Function)



Number Type

- Numbers are values that can be used in mathematical operations.
- Integers and real numbers are the same type (no int vs. double)
- Many operators auto-convert types: "2" * 3 is 6

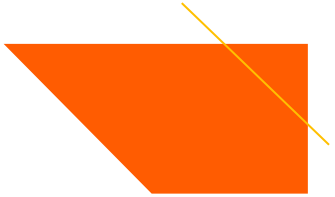
```
let enrollment = 99;
let medianGrade = 2.8;
let credits = 5 + 4 + (2 * 3);
```



String Type

- Strings are values made up of text and can contain letters, numbers, symbols, punctuation, and even emojis!
- Strings are contained within a pair of either single quotation marks ‘’ or double quotation marks “”.
  - Both quotes represent Strings but be sure to choose one and STICK WITH IT.

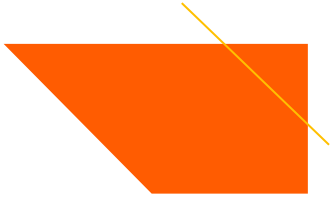
```
const s = "Connie Client";
const s2 = 'Melvin Merchant'; // can use "" or ' '
```



String Properties and Methods

- Strings have their own built-in properties and methods.
- Here are some of the most common ones:
- Methods: toLowerCase(), toUpperCase(), trim(), substring(), charAt(), indexOf() etc.
- Properties: length

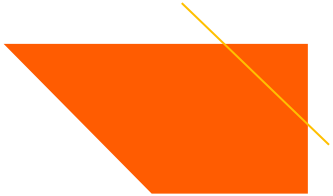
```
const s = "Connie Client";
let fName = s.substring(0, s.indexOf(" ")); // "Connie"
let len = s.length; // 13
```



Boolean Type

- A boolean value is one that can either be TRUE or FALSE.
- Anything that needs to be “on” or “off”, “yes” or “no”, “true” or “false”, is a good fit for booleans.

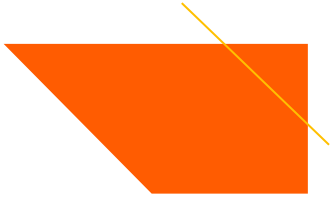
```
const iLikeWebApps = true;
const ieIsGood = false;
```



Special types: undefined and null

- **undefined:** has been declared, but no value assigned
- **null:** exists, and was specifically assigned a value of null
- Reference error when try to evaluate a variable that has not been declared

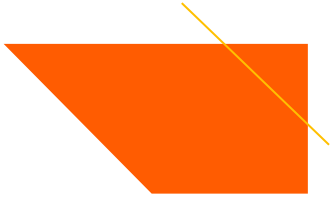
```
let ned = null; // ned is null
const benson = 9; // benson's 9
let caroline; // caroline is undefined
alert(fred);   // reference error, fred not declared
```



Arrays

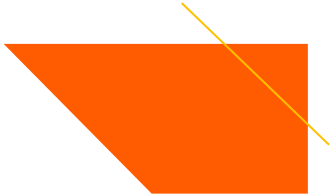
- Arrays are containers that can hold other values.
  - the values inside an array are called elements
- Elements of an array can be any kind of value — even other arrays.
- Size of an array is dynamic and can grow as needed.
- Two ways to declare and initialize an array:

```
let name = []; // empty array
name[index] = value; // store element

let name = [value, value, ..., value]; // pre-filled
```



Array methods and properties

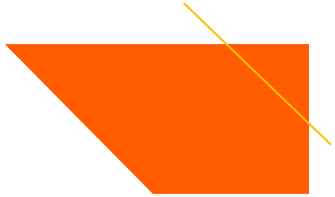
- Arrays have their own built-in properties and methods. Here are some of the most common ones.
- Methods:
  - push(): adds an element to the array and returns the array's length.
  - pop(): removes the last element in the array and returns that element's value.
  - concat(): returns a new array that combines the values of two arrays.
  - reverse(): returns a copy of the array in opposite order.
- Properties:
  - length: length property stores the number of elements inside the array.



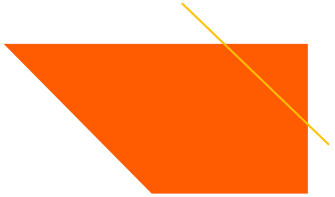
Function

- functions are reusable blocks of code that perform a specific task, taking some form of input and returning an output.
- Function declaration:

```
function name() {
  statement ;
  ...
  statement ;
}

function square(number) {
  return number* number;
}
```

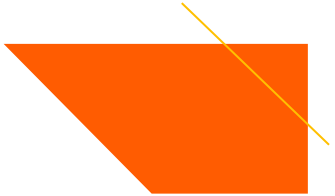


Operators



OPERATORS

- Operators are the symbols between values that allow different operations like addition, subtraction, multiplication, and more.
- Common JavaScript operators are:
  - Arithmetic (+, -, *, /, %, ++, --)
  - Logical (&&, ||, !)
  - Relational (>, <, >=, <=, ==, !=, ===, !==)
  - Assignment (=, +=, -=, *=, /=, %=)
  - Concatenation (+)
  - Grouping (())



OPERATORS

- Many arithmetic operators auto-convert types:

  "2" * 3 is 6

- Most relational operators auto-convert types:

  5 < "7" is true

  42 == 42.0 is true

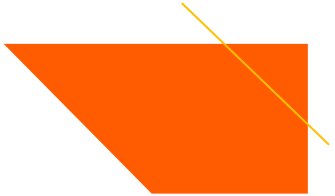
  "5.0" == 5 is true

- === and !== are strict equality tests (checks both type and value)

  "5.0" === 5 is false

- Concatenation with + :

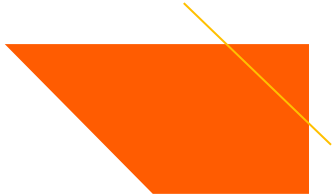
  1 + 1 is 2, but "1" + 1 is "11"



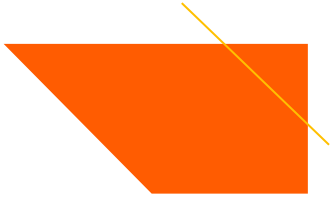
Comments

- Identical to Java's comment syntax
- Recall: 4 comment syntaxes
  - HTML: <!-- comment -->
  - CSS/JS/PHP/Java: /* comment */
  - Java/JS/PHP: // comment

```
// single-line comment
/* multi-line comment */
```



Control Flow Statements



Control flow statements

- Statements that control the flow of execution
  - Conditional statements
  - Repetition statements

Conditional statements

- Let us choose which statement will be executed next
  - “if” statements: if a condition is true, it’s block of code is executed.
  - “else” statements: if the condition is false, another block of code is executed.
  - “else if” statements: add new conditions to an if statement.

```
if (condition) {
  statements;
}
```

```
if (condition) {
  statements;
} else {
  statements;
}
```

```
if (condition1) {
  statements;
} else if (condition2) {
  statements;
} else {
  statements;
}
```

Repetition statements

- Repetition statements allow us to execute a statement multiple times.
  - for loop

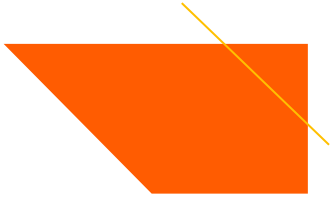
```
for (initialization; condition; update){
 statements;
}
```

  - while loop

```
while (condition) {
statements;
}
```

  - do-while loop

```
do {
statements;
} while (condition);
```



Thank You!