JavaScript Fundamentals Part 1

Notes

Please be sure to give Pallas Kennedy and Simon Kaminski permission to edit this document. Upload the link to this document in the comments of your file submission on Canvas to have it graded. Feel welcome to use color, erase the underline prompts, change the spacing and font, add additional notes and otherwise modify this document. However, please make it very easy and clear how to find what we are looking to grade in each submission.

## L1 Notes: Variables and Values

In JavaScript, lines of code end with \_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_ is the command that allows you to output to the console in JavaScript

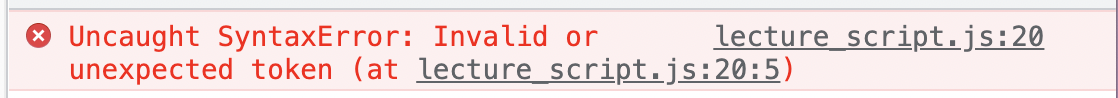
To link a JavaScript file, write

<script \_\_\_\_="filename.js"></\_\_\_\_\_\_>

between the </style>and </\_\_\_\_\_> tags in your html file.

* make sure that filename.js and the .html document are in the \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_.
* make sure that the filename.js is spelled the same in the script tag

This error:



Means:

In Javascript, by convention, variables are written using \_\_\_\_\_ case. For example, in Python we would write a variable name as my\_variable\_name. In JavaScript, the same variable would be written as: \_\_\_\_\_\_\_\_\_\_

Variables should be descriptive.

As an example, instead of using single letter variables for the the area formula A = l \* w, it would be written using descriptive JavaScript variables as:

Hard rules for variables:

| NO | Yes |
| --- | --- |
| Cannot start with a \_\_\_\_\_\_\_\_\_\_\_ | Can have numbers in other places |
| Cannot use special characters or other symbols and cannot contain spaces | However, you can use and start variables with \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ or $ |

The error message 'invalid or unexpected token' means or happens when \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## L2 Notes: Data Types

There are \_\_\_\_\_\_ data types:

* \_\_\_\_\_\_\_\_\_\_
  + Always floating point even if we don’t see the decimal point
  + Used for decimals and integers
* \_\_\_\_\_\_\_
  + Sequence of characters
  + Put in quotes (single or double quotes)
  + Used for text
* \_\_\_\_\_\_\_\_
  + true or false <- **lowercase in JS**
  + Used for taking decisions
* Undefined
  + Value taken by a variable that is \_\_\_\_\_\_\_\_\_
  + Empty value
* Null
  + Also means \_\_\_\_\_\_ value
  + Used in different circumstances than undefined
* Symbol
  + Defines a value that is unique and cannot be \_\_\_\_\_\_\_
* big int
  + Larger \_\_\_\_\_\_\_\_ than the Number type can hold

\_\_\_\_\_\_\_\_\_ is an operator that lets you check the data type.

Dynamic Typing means when you create a variable you don’t have to \_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_is the value taken by a variable that has been declared but not yet given a value.

## L3 Notes: let const var

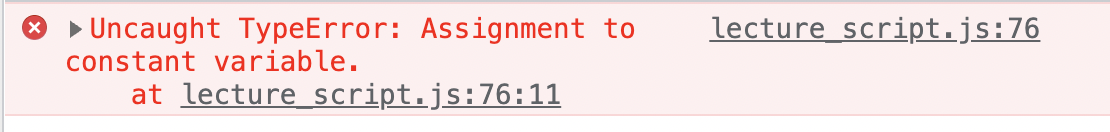
let : used to declare variables that \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_.

const: used to declare variables that \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_. This is used by default.

var: this is the \_\_\_\_\_\_\_\_\_\_\_\_ to declare variables and should not be used.

Always properly declare variables by using \_\_\_\_\_\_ or \_\_\_\_\_ to make sure JavaScipt is handling them correctly.

This error message:



means:

This error message:



means:

## L4 Notes: Basic Operators

| Operation or what it does | Operator |
| --- | --- |
| Addition |  |
|  | - |
| Multiplication |  |
|  | / |
|  | \*\* |
| Is assigned to |  |
|  | += |
| Subtract and assign back to the variable |  |
| Add 1 and assign back to the variable |  |
|  | - - |
|  | \*= |
| Divide by the value and assign back to the variable |  |
| Greater than ( A > B: A is larger than B) |  |
|  | < |
|  | >= |
| Less than or equal to (A <= B : A is smaller than B or the same as B) | <= |

## L5 Notes: Operator Precedence

To find the precedence table, google \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ and select the link with this url: development.mozilla.org

Rearrange these in order of which comes first (top) to last (bottom)

multiplication/ division

Parentheses (grouping)

addition/subtraction

Assignment

Unlike most operations, the assignment operator works from the \_\_\_\_\_\_to the \_\_\_\_\_, performing the assignment as the last operation.

Use \_\_\_\_\_\_\_\_\_ to override the default precedence in a calculation.

## L6 Notes: Strings and Template Literals

To create a template literal, you must use \_\_\_\_\_\_\_\_\_\_ instead of quote marks (found above the \_\_\_\_\_key). Then you must enclose the variable or expression with \_\_\_\_\_\_\_\_\_\_.

An example of a template literal using a variable is:

When logging to the console, using the enter key inside of a template literals creates \_\_\_\_\_\_\_\_\_\_\_\_ output.

Backticks can be used for all strings instead of \_\_\_\_\_\_\_\_\_\_\_\_\_ marks.

## L7 Notes: if else Statements

The form of the if-else block in JavaScript is:

if ( \_\_\_\_\_\_\_\_\_\_ ) {

\_\_\_\_\_\_\_\_\_\_\_\_

}

Variables defined in the control block \_\_\_\_\_\_\_ \_\_ \_\_\_\_\_\_\_\_ outside of it.

This error message:



means:

## L8 Notes: Type Conversion and Coercion

In the console, \_\_\_\_\_\_\_ are white and numbers are \_\_\_\_\_\_\_\_\_\_\_\_.

The function for converting to a number is :

The function for converting to a string is :

NaN means:

The data type of NaN is best thought of as:

Fill in the coercion column with string or number and fill in the result of the example.

| Operator | Coerces to | Example | Result |
| --- | --- | --- | --- |
| + |  | ‘2’ + 3 |  |
| + |  | 2 + ‘3’ |  |
| + |  | ‘2’ + ‘3’ |  |
| - |  | 10 - ‘4’ |  |
| - |  | ‘10’ - 4 |  |
| - |  | ‘10’ - ‘4’ |  |
| \* |  | ‘12’ \* ‘2’ |  |
| \* |  | ‘12’ \* 2 |  |
| / |  | ‘12’ / ‘2’ |  |
| / |  | ‘12’ / 2 |  |

## L9 Notes: Truthy and Falsy Values

Falsy values are values that are \_\_\_ \_\_\_\_\_\_\_\_ , but will become false when we convert them into a \_\_\_\_\_\_\_\_\_.

The 5 falsy value are:

* \_\_\_\_\_
* \_\_\_\_\_
* \_\_\_\_\_
* \_\_\_\_\_
* \_\_\_\_\_

False is already false so it is not classified as a falsy value.

All other values become \_\_\_\_\_ when we convert them into a boolean.

To typecast to boolean, use the following function:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Warning about type coercion: A variable assigned to \_\_\_\_ will coerce to false even though it is a perfectly valid number. This can cause weird results.

Javascript uses type coercion to Boolean in two circumstances: When using \_\_\_\_\_\_\_\_\_ operators and in a logical context such as the \_\_\_\_\_\_\_\_\_\_\_\_ of an if-else control structure.

## 10 Notes: Equality Operators

The single line if control structure has this form: if (\_\_\_\_\_\_\_) \_\_\_\_\_\_\_\_\_\_\_

The strict equality operator is \_\_\_\_\_\_\_\_\_. This should be used by default. It means \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The loose equality operator is \_\_\_\_\_\_\_\_\_. This one allows type \_\_\_\_\_\_\_\_\_ which can introduce weird behavior and hard to find bugs. This one should be avoided when possible.

The assignment operator is \_\_\_\_\_\_\_\_\_. This is used for \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

To get user input use the \_\_\_\_\_\_\_\_\_ function. As with Python, user input is data type \_\_\_\_\_\_\_\_\_. User input needs to be stored in a \_\_\_\_\_\_\_\_\_\_\_\_. When we expect the user to input a number, we can wrap the prompt() function in the \_\_\_\_\_\_\_\_\_ function.

The form is:

\_\_\_\_\_\_ variableName = Number( \_\_\_\_\_\_\_ (`User input question here'))

To add more conditions to an if control structure:

if (condition) {

block of code to execute if the condition true

} \_\_\_\_ \_\_\_ (new condition) {

block of code to execute if the new condition is true

}

You can have and may else-if as you want and also have an else.

The strict version of the different operator is \_\_\_\_\_\_\_\_\_. This should always be used.

The loose version of the different operator is \_\_\_\_\_\_\_\_\_\_. This should be avoided when possible.

## L11a Notes: Boolean Logic

Boolean AND Table

| AND | A:True | A:False |
| --- | --- | --- |
| B:True |  |  |
| B: False |  |  |

AND: The AND operator returns true only all values are \_\_\_\_\_\_. If at least one value is \_\_\_\_ the AND operator returns false.

Boolean OR Table

| OR | A:True | A:False |
| --- | --- | --- |
| B:True |  |  |
| B: False |  |  |

OR: If at least one value is \_\_\_\_\_\_ then the OR operator returns true. Only if all values are false does the OR operator return \_\_\_\_\_\_.

NOT: inverts the \_\_\_\_\_/\_\_\_\_\_ value

## 11b Notes: Boolean Logic

JavaScript Logical Operators

| Operator | JavaScript symbol |
| --- | --- |
| And |  |
|  | || (above the return/enter key) |
| Not |  |

In JavaScript, true and false are not capitalizied.

## L12 Notes: The Switch Statement

The form of the \_\_\_\_\_\_ statement:

switch (variable) {

case `Strict equality value`:

code to execute if variable is the same as strict equality value

break;

case…

.

.

.

default:

code to execute if no cases match

}

In the form above, the word variable represents \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

In the form above, the phrase `Strict equality value` means or represents \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A \_\_\_\_\_\_\_\_\_\_\_\_\_\_ must be used at the end of the line of code starting with case.

At the end of the executable block of code, a \_\_\_\_\_\_\_\_ must be included.

Multiple \_\_\_\_\_\_\_\_\_ can execute the same block of code.

The block of code for a case executed can have \_\_\_\_\_\_\_\_\_\_\_\_\_ lines.

There can be as many cases as you'd like.

Default is like the \_\_\_\_\_\_\_\_\_ clause of the \_\_\_\_\_\_\_\_\_\_\_ control structure.

The keyword break means or does \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The keyword switch means or does \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The keyword case means or does \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_