JavaScript Review

What is JavaScript

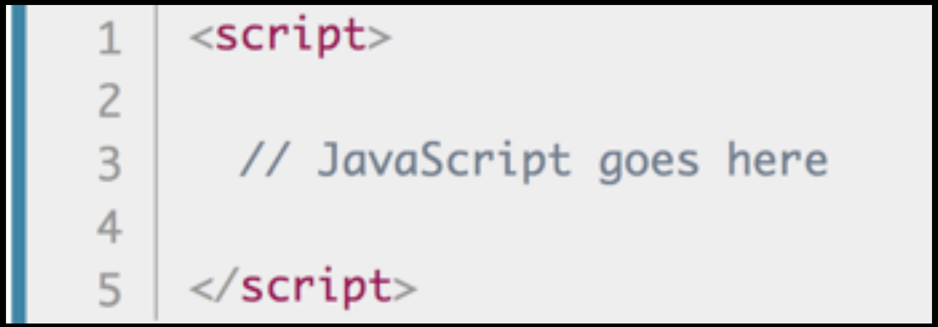
JavaScript is a cross-platform, object-oriented scripting language.

Client-side JavaScript extends the core language by supplying objects to control a browser and its Document Object Model (DOM).

Server-side JavaScript extends the core language by supplying objects relevant to running JavaScript on a server.

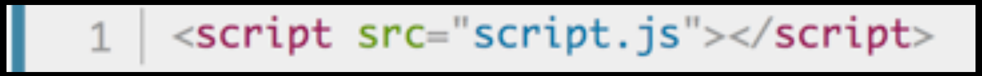
How to add

Internal JavaScript



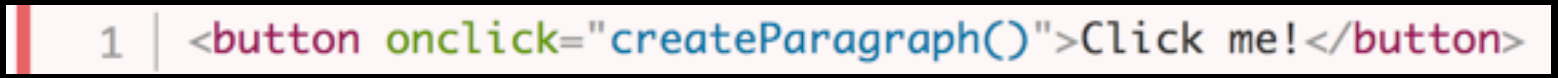
```
<script>

  // JavaScript goes here

</script>
```

External JavaScript



```
<script src="script.js"></script>
```

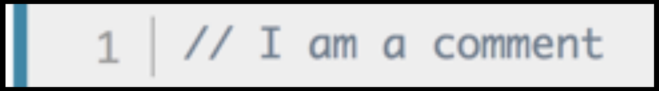
Inline JavaScript handlers



```
<button onclick="createParagraph()">Click me!</button>
```

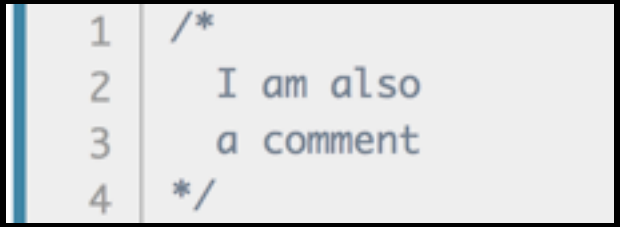
Comments

A Single line comments



```
// I am a comment
```

A Multi-line comments



```
/*
  I am also
  a comment
*/
```

Troubleshooting

Type of error

- Syntax errors: These are spelling errors in your code that actually cause the program not to run at all, or stop working part way through
- Logic errors: These are errors where the syntax is actually correct but the code is not what you intended it to be

How to fix errors

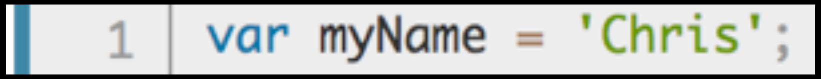
- Use 'Developer Tools'
- Use 'console.log()' function

Variable

A variable is a container for a value

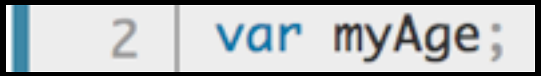
Declaring a variable

★With initial value



```
var myName = 'Chris';
```

★ With no value: set undefined



```
var myAge;
```

Variable types

★**Numbers**

★**Strings**

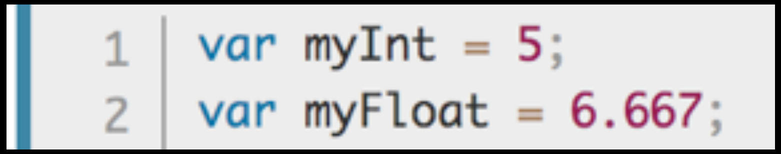
★**Booleans**

★**Arrays**

★**Objects**

Numbers

Creating a number



```
var myInt = 5;
var myFloat = 6.667;
```

Types of numbers

- ★ Integers
- ★ aFloats
- ★ Doubles
- ★ Binary, Octal, Hexadecimal

Arithmetic operators

+	addition
-	Subtraction
*	Multiplication
/	Division
%	Reminder(modulo)
++	Increment
--	Decrement

Assignment operators

= assignment

+= Addition assignment.

e.g. x +=3 same as x = x + 3

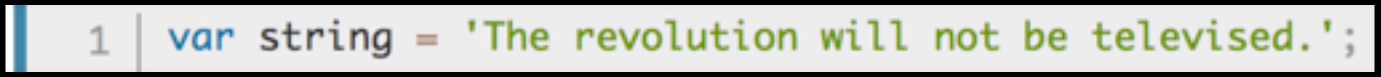
-=, *=, /=

Comparison operators

===	Strict equality
!==	Strict non equality
==, !=	
<	Less than
>	Greater than
<=, >=	

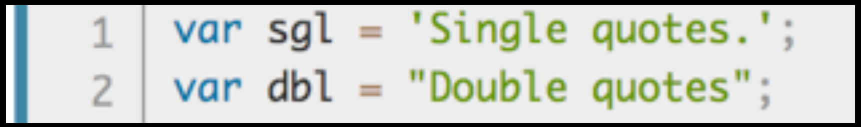
Strings

Create a string



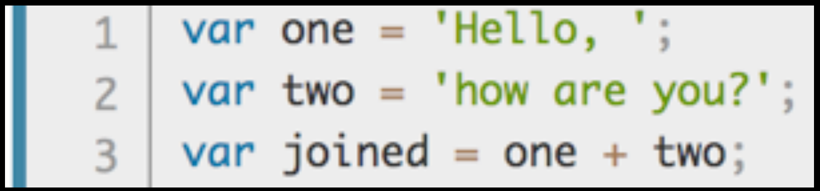
```
var string = 'The revolution will not be televised.';
```

Single quotes vs. double quotes



```
var sgl = 'Single quotes.';
var dbl = "Double quotes";
```

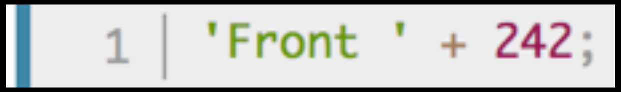
Concatenating strings



```
var one = 'Hello, ';
var two = 'how are you?';
var joined = one + two;
```

Numbers vs. strings

Add a string and a number: convert to string

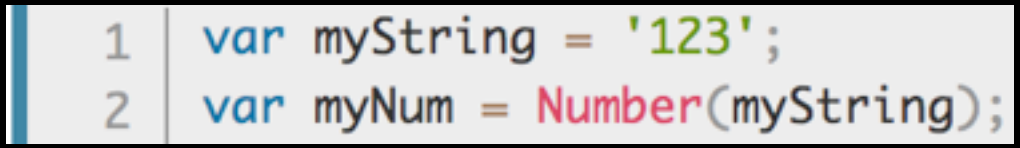


```
'Front ' + 242;
```

Strings

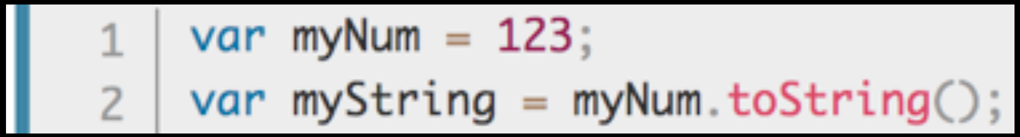
Convert

String to number: Number() function



```
var myString = '123';
var myNum = Number(myString);
```

Number to string: String.toString() method



```
var myNum = 123;
var myString = myNum.toString();
```

Member variable(property) and function(method)

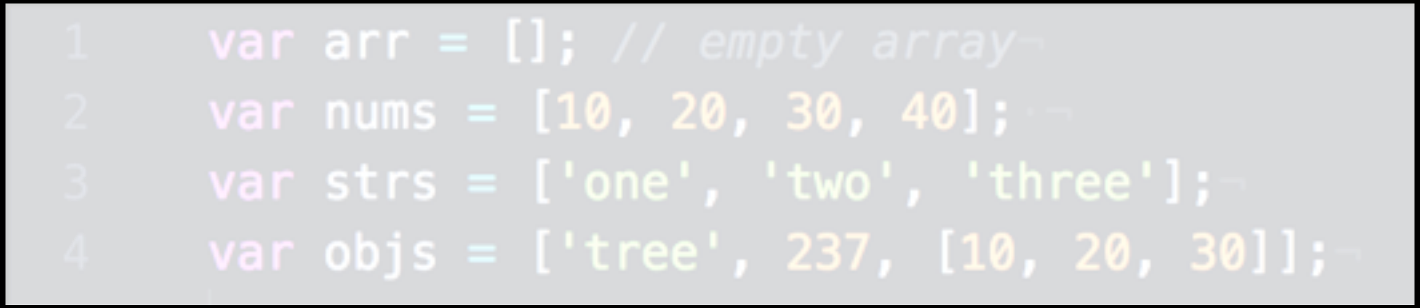
code	Name	Description
String.length	Length property	Return length of string
String.indexOf(‘str’)	Finding index method	Return finding index or -1
String.toLowerCase()	Lower case method	Return Lower case string
String.toUpperCase()	Upper case method	Return Upper case string
String.replace(‘find’,’repl’)	Replace method	Return replaced string

Arrays

Arrays are generally described as "list-like objects"; they are basically single objects that contain multiple values stored in a list.

Creating an array

Arrays are constructed of square brackets, which contain a list of items separated by commas.

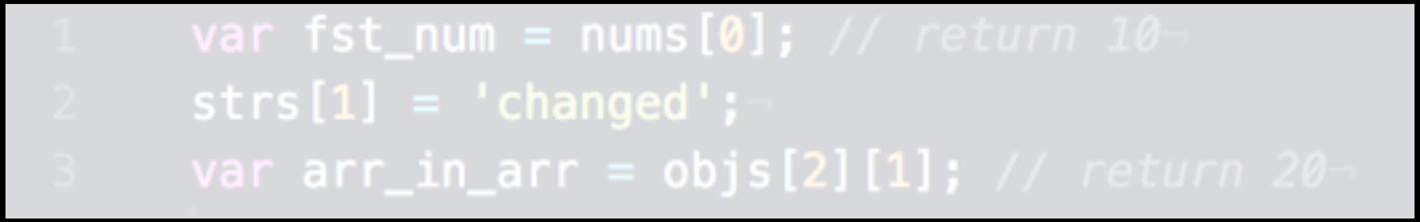


```
var arr = []; // empty array
var nums = [10, 20, 30, 40];
var strs = ['one', 'two', 'three'];
var objs = ['tree', 237, [10, 20, 30]];
```

Accessing and modifying array items

You can then access individual items in the array using bracket notation,

in the same way that you accessed the letters in a string.

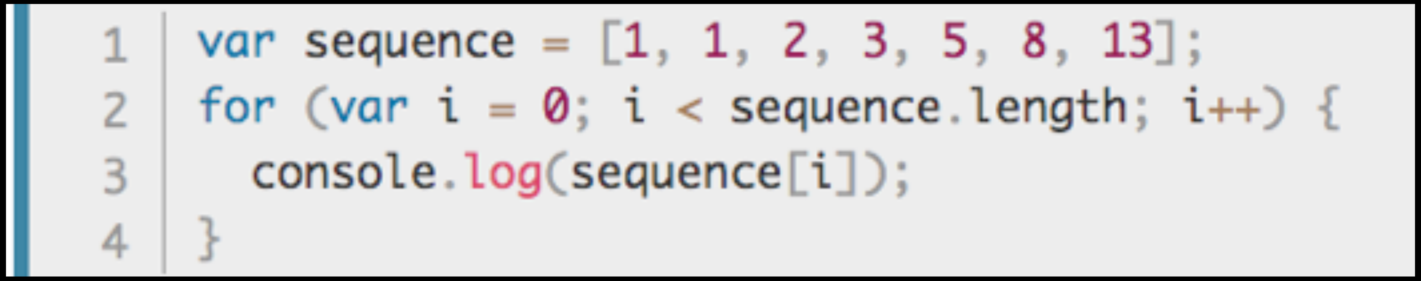


```
var fst_num = nums[0]; // return 10
strs[1] = 'changed';
var arr_in_arr = objs[2][1]; // return 20
```

Arrays

Finding the length of an array

You can find out the length of an array by using the **length** property.



```
var sequence = [1, 1, 2, 3, 5, 8, 13];
for (var i = 0; i < sequence.length; i++) {
  console.log(sequence[i]);
}
```

Member variable(property) and function(method)

Code	Name	Description
Array.length	Length property	Return length of array
String.split(str)	String split method	Return array of splitted by str
Array.join(str)	Array join method	Return string of joined by str
Array.toString()	toString method	Return string of joined by ‘,’
Array.push(obj)	Array push method	Put object to array
Array.pop()	Array pop method	Removing the last object and return object

if statements

Basic if ... else syntax

```
if (condition) {
   code to run if condition is true
} else {
   run some other code instead
}
```

- ⋆if: keyword
- ⋆condition: A condition to test
    - ⋆comparison operators: ===, !==, ==, !=, <, >, <=, >=
    - ⋆logical operators: &&(AND), ||(OR), !(NOT)
  - ⋆true block: true, any object except empty string(‘ ’), zero(0)
- ⋆false block: false, undefined, null, 0, NaN, empty string(‘’)

if block without the curly braces

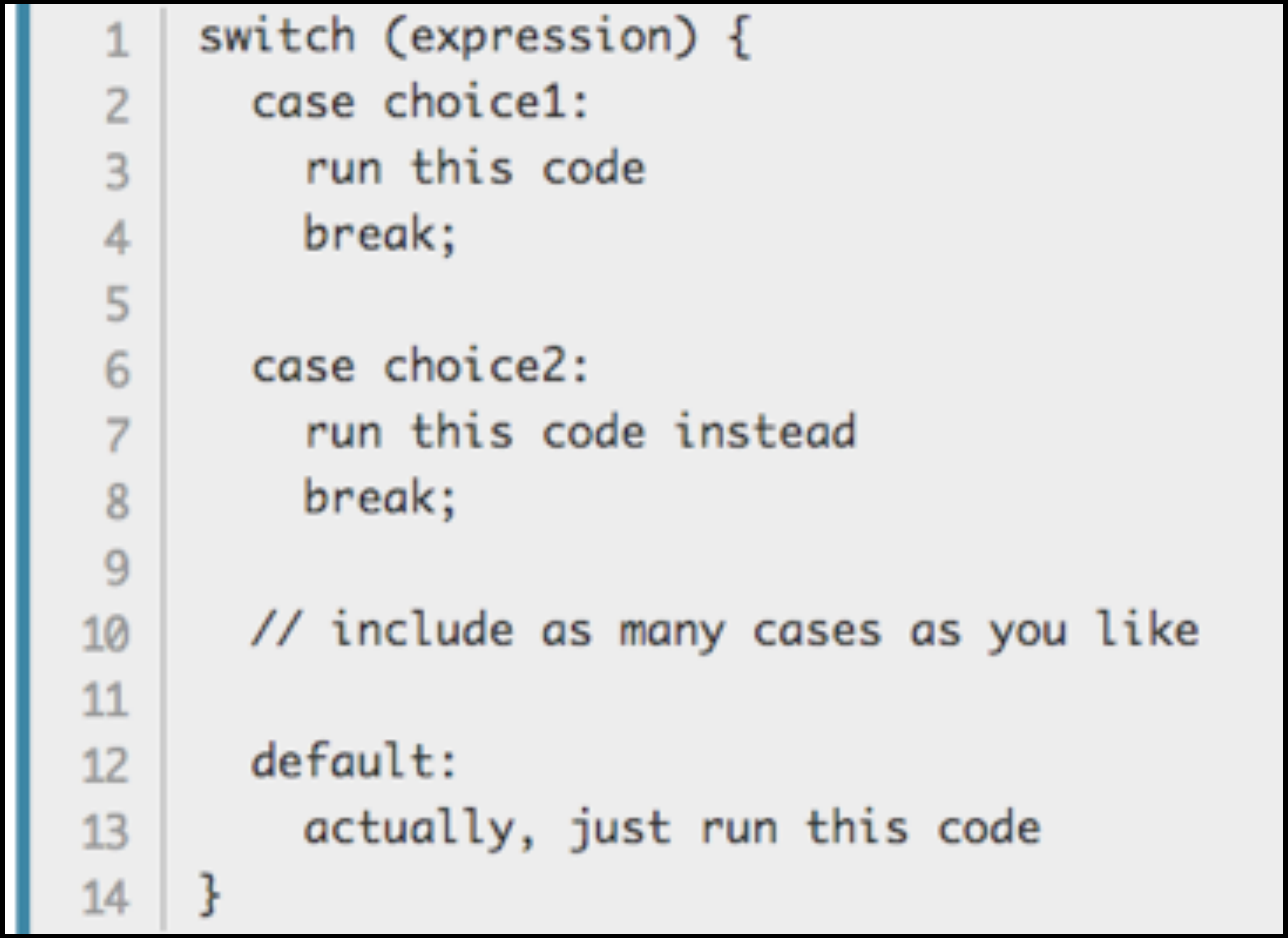
```
if (condition) code to run if condition is true
else run some other code instead
```

if() only

```
if (condition) {
   code to run if condition is true
}
```

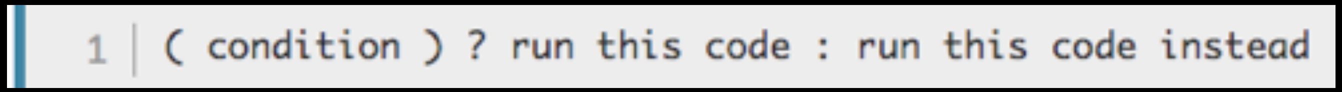
if() ... else if() ... else if() ... else block

switch statement



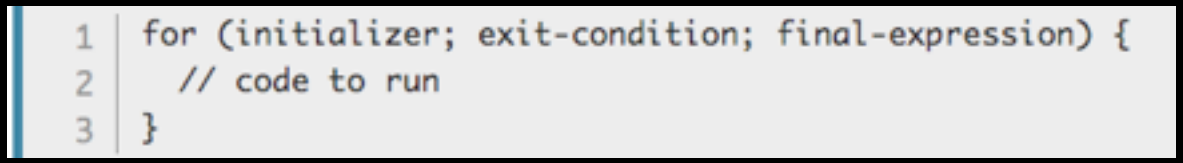
```
switch (expression) {
  case choice1:
    run this code
    break;

  case choice2:
    run this code instead
    break;

  // include as many cases as you like

  default:
    actually, just run this code
}
```

Ternary operator



```
( condition ) ? run this code : run this code instead
```

for loop

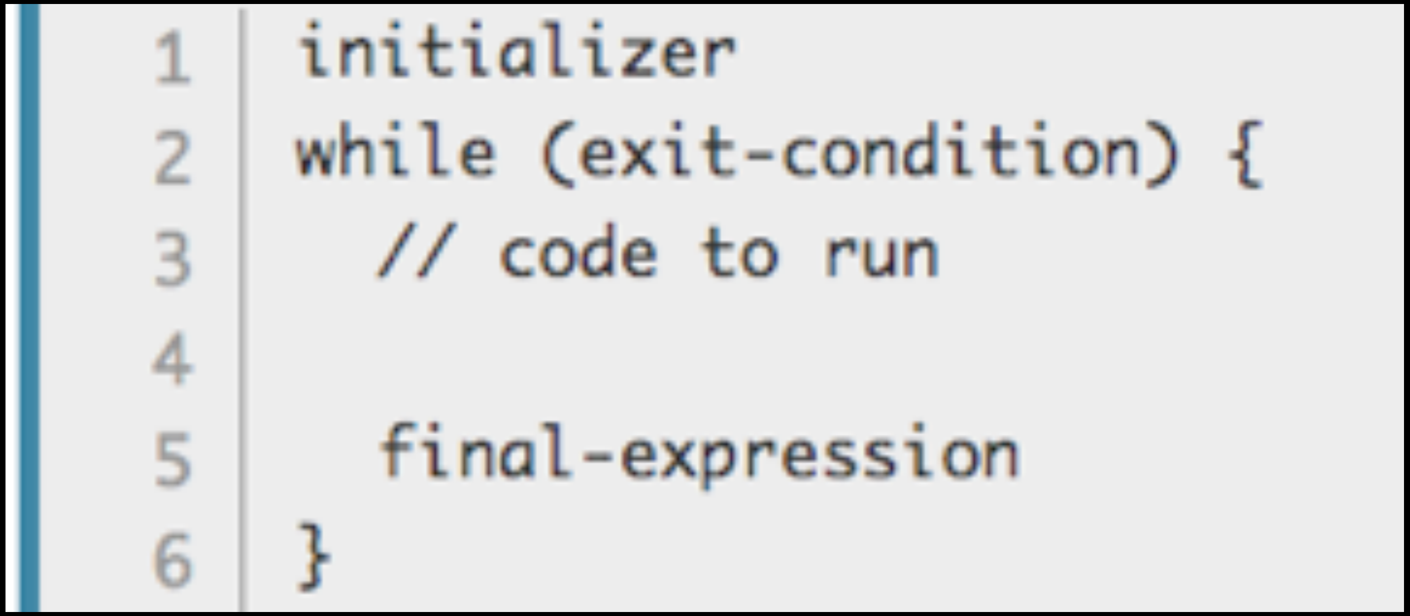


```
for (initializer; exit-condition; final-expression) {
  // code to run
}
```

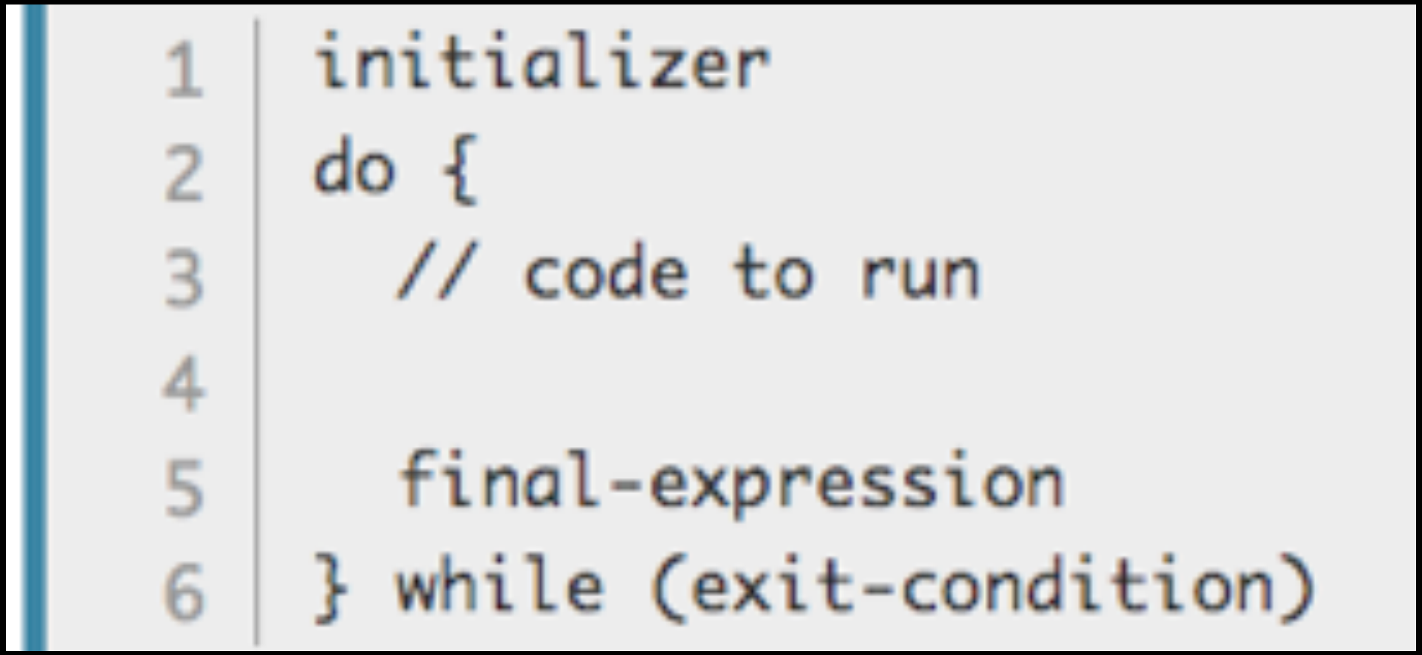
Exiting loops with “break”

Skipping iterations with “continue”

While and do ... while loop



```
initializer
while (exit-condition) {
  // code to run

  final-expression
}
```



```
initializer
do {
  // code to run

  final-expression
} while (exit-condition)
```