DECT-KS© TOOLS

Web Engineering Front-end Pt. 3

8. JavaScript: Loops II

erb 僱員再培訓局 Employees Retraining Board
SCOPE School of Continuing and Professional Education 專業進修學院 香港城市大學 City University of Hong Kong
KRYSTAL INSTITUTE DIGITAL ECONOMY CORE TECHNOLOGY

DECT-KS©
TOOLS

Revision

What are loops?

loops allow us to execute many iterations of similar code more efficiently.

while loop

```
while (condition) {
  // code to be run if condition is true
}
// program then returns to the beginning of the loop
```

do-while loop

```
do {
  // code to be run if it is the first iteration or if condition is true
} while (condition);
// program then returns to the beginning of the loop
```

break and continue statements

- break statements can be used to escape from a loop
- continue statements can be used to skip an iteration of a loop

Contents

8.1 for loops

for loops

for loop is another type of loop in JavaScript. It differs from while loop in that it can take multiple parameters or statements.

for loop syntax

```
for (statement1; statement2; statement3) {
  // code to be executed
}
```

statement1

statement1 is used to initialize the variable to be incremented throughout the loop.

statement2

statement2 is the condition that determines whether the loop runs or not.

statement3

statement3 is an expression that will execute at the end of the loop. It is usually used to increment the value initialized in statement1.

for loop syntax

All the statements in for loop are actually optional, JavaScript won’t spit an error if you forget any. But you need to be careful not to create an infinite loop.

for loop example

```
for (i = 1; i <= 10; i++) {
   console.log(i);
}
```

Practice: for loop

Try re-writing the loops you created from the previous lesson into for loops. Which do you prefer?

8.2 for-in loops

for-in loops

for-in loops can be used to loop through the properties of an object.

for-in loop syntax

```
for (property in object) {
  // code to be executed
}
```

for-in loop example

```
var dog = {
   name: “Marley”,
   gender: “Male”,
   age: 2,
   breed: “Labrador retriever”
};

for (x in dog) {
   console.log(x+”: “+dog[x]);
}

// name: Marley, gender: Male, age: 2, breed: Labrador retriever
```

for-in loop example

In the previous example, x represents the key of the property, and we can use dog[x] to get the value.

for-in loops and arrays

Using for-in loops with arrays is not recommended because the values may not be accessed in numerical order in some cases. The for-in loop could also loop through the properties of the array as well.

Practice: for-in loops

Try looping through the properties of the objects you created in lesson 6 using for-in loops.

8.3 for-of loops

for-of loop

for-in loop can be used to loop through the values of an iterable object (strings, arrays etc.)

for-of loop syntax

```
for (variable of iterable object) {
   // code to be executed
}
```

for-of loop example (array)

```
var animals = [“Cat”, “Dog”, “Cow”, “Horse”, “Chicken”]

for (x of array) {
  console.log(x);
}

// Cat, Dog, Cow, Horse, Chicken
```

for-of loop example (string)

```
for (x of “JavaScript”) {
  console.log(x);
}

// J
// a
// v
// a
// S
// c
// r
// i
// p
// t
```

for-in vs for-of loop

```
var animals = [“Cat”, “Dog”, “Cow”, “Horse”, “Chicken”]

for (x in animals) {
  console.log(x);
}
// 0, 1, 2, 3, 4

for (x of animals) {
  console.log(x);
}
// Cat, Dog, Cow, Horse, Chicken
```

Practice: for-of loops

Try looping through the arrays you created in lesson 6 using for-of loops.

8.4 Nested Statements

What are nested statements?

We can stack statements together to perform more complex functions. This is called nesting.

Nested statements syntax

```
statement1 {
   statement2 {
   }
}
```

Nested statements syntax

When nesting statements, pay close attention to the {curly braces}, as they represent which layer you are currently working in.

Nested if example

```
if (userinput == username) {
  if (passinput == password) {
    console.log(“Log-in successful!”);
  } else {
    console.log(“Incorrect password”);
  }
} else {
  console.log(“Incorrect username”);
```

Nested if example

The example is used to authenticate a user’s log-in details. Using nested ifs, it can display different messages for different combinations of valid and invalid inputs.

Nested statements

The nested statements can also be of different types. For example, in the previous lesson we placed an conditional break statement inside a while loop.

if inside while loop

```
var i = 0

while (i <= 10) {
  if (i == 4) {
    i++;
    break;
  }
  console.log(i);
  i++;
}

// 0, 1, 2, 3
```

break with nested statements

break and continue will only escape from the innermost loop that surrounds the statement.

break with nested statements

```
for (...) {
  for (...) {
    if (...) {
      break;
    }
  }
  // break target
}
```

Labelled break

To break entirely from a nested loop, you can use a labelled break.

Labelled break syntax

```
label:
for (...) {
  for (...) {
    if (...) {
      break label;
    }
  }
}
// break target
```

Nested statements

Nested statements can be very useful, but be careful not to overcomplicate your things by trying to fit too much into one block of code.

Practice: Nested statements

Spend some time to practice writing nested statements. Try to use all the statements we have learned so far.

DECT-KS©
TOOLS

The End

Reference 1: W3 Schools JavaScript Tutorial https://www.w3schools.com/js/default.asp