

JavaScript: Strutture di Controllo

Sommario

- Introduzione
- Algoritmi e Pseudocodice
- Strutture di Controllo
- if, if ...else
- while
- Assegnamento e Incremento / Decremento
- Tipizzazione
- Ciclo for
- Selezione multipla (switch)
- Ciclo do...while
- Istruzioni break e continue
- Operatori Logici
- Web Resources

Obiettivi

- Capire le tecniche di base di problem-solving
- Essere in grado di sviluppare algoritmi
- Essere in grado di usare i costrutti di base per la selezione e iterazione
- Essere in grado di usare gli operatori di incremento/decremento

Introduzione

- Per scrivere uno script è necessario
 - Capire **precisamente** il problema
 - Pianificare **dettagliatamente** l'approccio
 - Capire gli elementi di base disponibili
 - Applicare i principi di buona programmazione

Algoritmi

- Specificano le azioni che devono essere eseguite per giungere alla soluzione

Pseudocodice

- Artificiale
- Informale
- Aiuta il programmatore a sviluppare algoritmi

Strutture di Controllo (1)

- Esecuzione sequenziale
 - Le istruzioni sono eseguite nell'ordine con cui sono scritte, una dopo l'altra
- Trasferimento del controllo
 - Talvolta l'istruzione che deve essere eseguita potrebbe non essere quella immediatamente successiva

Strutture di Controllo (2)

- Tre strutture di controllo
 - Sequenza
 - Selezione
 - i f
 - i f...e l se
 - swi tch
 - Ripetizione
 - whi l e
 - do...whi l e
 - for
 - for...i n

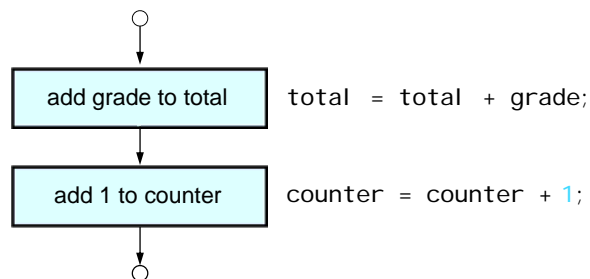
Strutture di Controllo (3)

- Flowchart
 - Rappresentazione grafica di (una parte di) un algoritmo
 - Linee di flusso (Flowlines)
 - Indicano l'ordine con cui sono eseguite le azioni specificate dall'algoritmo

Strutture di Controllo (4)

- Rettangolo
 - Indica un generico tipo di azione
- Ovale
 - Un algoritmo completo
- Cerchio
 - Una parte di un algoritmo
- Diamante
 - Indica un punto di decisione relativamente al valore di verità di una condizione

Strutture di Controllo (5)



Parole Riservate di JavaScript

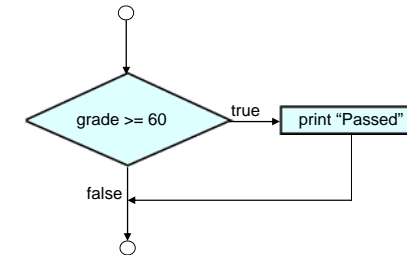
JavaScript Keywords				
break	case	catch	continue	default
delete	do	else	finally	for
function	if	in	instanceof	new
return	switch	this	throw	try
typeof	var	void	while	with
<i>Keywords that are reserved but not currently used by JavaScript</i>				
abstract	boolean	byte	char	class
const	debugger	double	enum	export
extends	final	float	goto	implements
import	int	interface	long	native
package	private	protected	public	short
static	super	synchronized	throws	transient
volatile				

Fig. 8.2 JavaScript keywords.

Selezione i f (1)

- Indica l'azione che deve essere eseguita solo quando la condizione è vera

Selezione i f (2)



Selezione i f...el se (1)

- Indica le diverse possibili azioni che devono essere eseguite quando la condizione è vera o quando è falsa
- Operatore Condizionale (?:)
 - È l'unico operatore ternario di JavaScript
 - Rappresenta un'espressione condizionale
 - Tre operandi
 - L'espressione booleana che deve essere valutata;
 - Il valore assunto dall'espressione condizionale nel caso in cui l'espressione sia vera;
 - Il valore assunto dall'espressione condizionale nel caso in cui l'espressione sia falsa

Selezione i f...el se (2)

- Dangling-else problem
 - L'interprete JavaScript associa ogni else all'if precedente più vicino
- Ad esempio:

```
if (x>5)
  if (y>5)
    document.writeln("sia x che y sono > 5")
  else
    document.writeln("x è <=5")
```

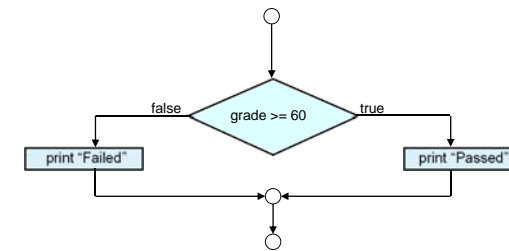
Se x = 6 e y = 3, l'output è

x è <=5

Selezione i f...el se (3)

- Per evitare il problema si usano i delimitatori di blocco { e }
 - L'esempio corretto è
- ```
if (x>5)
{
 if (y>5)
 document.writeln("sia x che y sono > 5)
}
else
 document.writeln ("x è <=5")
```

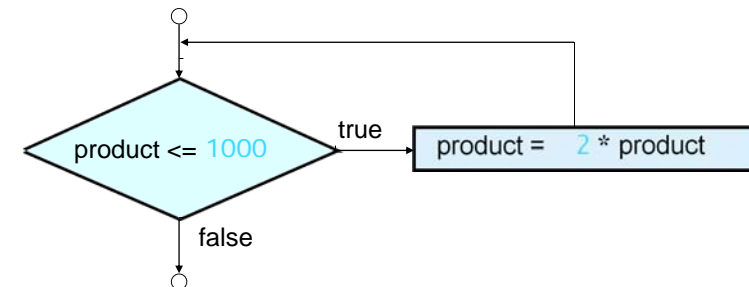
## Selezione i f...el se (4)



## Ripetizione while (1)

- Struttura di ripetizione (loop)
  - Ripete l'azione fin tanto che la condizione è vera

## Ripetizione while (2)



```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 8.7: average.html -->
6 <!-- Class Average Program -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Class Average Program</title>
11
12 <script type = "text/javascript">
13 <!--
14 var total, // sum of grades
15 gradeCounter, // number of grades entered
16 gradeValue, // grade value
17 average, // average of all grades
18 grade; // grade typed by user
19
20 // Initialization Phase
21 total = 0; // clear total
22 gradeCounter = 1; // prepare to loop
23

```

JavaScript: Strutture di Controllo

21

```

24 // Processing Phase
25 while (gradeCounter <= 10) { // loop 10 times
26
27 // prompt for input and read grade from user
28 grade = window.prompt("Enter Integer grade:", "0");
29
30 // convert grade from a string to an integer
31 gradeValue = parseInt(grade);
32
33 // add gradeValue to total
34 total = total + gradeValue;
35
36 // add 1 to gradeCounter
37 gradeCounter = gradeCounter + 1;
38 }
39
40 // Termination Phase
41 average = total / 10; // calculate the average
42
43 // display average of exam grades
44 document.writeln(
45 "<h1>Class average is " + average + "</h1>");
46 // -->
47 </script>

```

JavaScript: Strutture di Controllo

22

```

48
49 </head>
50 <body>
51 <p>Click Refresh (or Reload) to run the script again</p>
52 </body>
53 </html>

```



JavaScript: Strutture di Controllo

23

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 8.9: average2.html -->
6 <!-- Sentinel-controlled Repetition -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Class Average Program:
11 Sentinel-controlled Repetition</title>
12
13 <script type = "text/javascript">
14 <!--
15 var gradeCounter, // number of grades entered
16 gradeValue, // grade value
17 total, // sum of grades
18 average, // average of all grades
19 grade; // grade typed by user
20
21 // Initialization phase
22 total = 0; // clear total
23 gradeCounter = 0; // prepare to loop
24

```

JavaScript: Strutture di Controllo

24

```

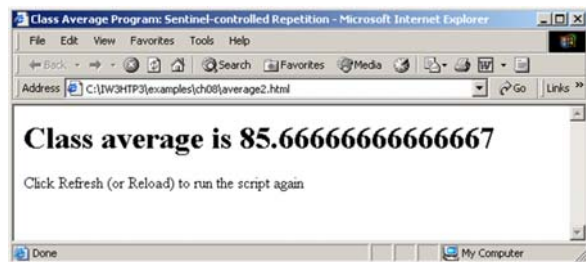
25 // Processing phase
26 // prompt for input and read grade from user
27 grade = window.prompt(
28 "Enter Integer Grade, -1 to Quit:", "0");
29
30 // convert grade from a string to an Integer
31 gradeValue = parseInt(grade);
32
33 while (gradeValue != -1) {
34 // add gradeValue to total
35 total = total + gradeValue;
36
37 // add 1 to gradeCounter
38 gradeCounter = gradeCounter + 1;
39
40 // prompt for input and read grade from user
41 grade = window.prompt(
42 "Enter Integer Grade, -1 to Quit:", "0");
43
44 // convert grade from a string to an Integer
45 gradeValue = parseInt(grade);
46 }
47

```

```

48 // Termination phase
49 if (gradeCounter != 0) {
50 average = total / gradeCounter;
51
52 // display average of exam grades
53 document.writeln(
54 "<h1>Class average is " + average + "</h1>");
55 }
56 else
57 document.writeln("<p>No grades were entered</p>");
58 // -->
59 </script>
60 </head>
61
62 <body>
63 <p>Click Refresh (or Reload) to run the script again</p>
64 </body>
65 </html >

```



```

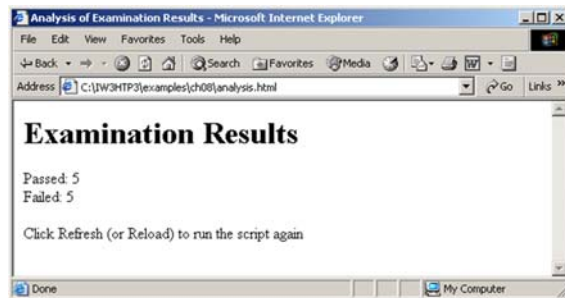
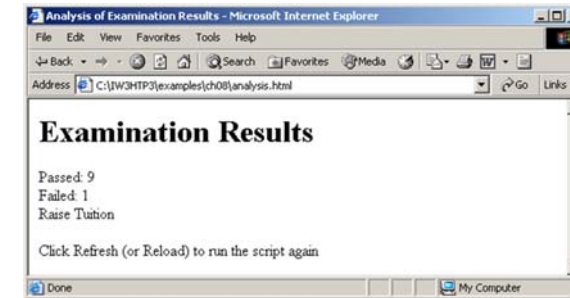
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 8.11: analysis.html -->
6 <!-- Analyzing Exam Results -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Analysis of Examination Results</title>
11
12 <script type = "text/javascript">
13 <!--
14 // Initializing variables in declarations
15 var passes = 0, // number of passes
16 failures = 0, // number of failures
17 student = 1, // student counter
18 result; // one exam result
19
20 // process 10 students; counter-controlled loop
21 while (student <= 10) {
22 result = window.prompt(
23 "Enter result (1=pass,2=fail)", "0");
24

```

```

25 if (result == "1")
26 passes = passes + 1;
27 else
28 failures = failures + 1;
29
30 student = student + 1;
31 }
32
33 // termination phase
34 document.writeln("<h1>Examination Results</h1>");
35 document.writeln(
36 "Passed: " + passes + "
Failed: " + failures);
37
38 if (passes > 8)
39 document.writeln("
Raise Tuition");
40 // -->
41 </script>
42
43 </head>
44 <body>
45 <p>Click Refresh (or Reload) to run the script again</p>
46 </body>
47 </html >

```



## Operatori di Assegnamento

| Assignment operator | Initial value of variable | Sample expression | Explanation | Assigns |
|---------------------|---------------------------|-------------------|-------------|---------|
| +=                  | c = 3                     | c += 7            | c = c + 7   | 10 to c |
| --=                 | d = 5                     | d -= 4            | d = d - 4   | 1 to d  |
| *=                  | e = 4                     | e *= 5            | e = e * 5   | 20 to e |
| /=                  | f = 6                     | f /= 3            | f = f / 3   | 2 to f  |
| %=                  | g = 12                    | g %= 9            | g = g % 9   | 3 to g  |

Fig. 8.12 Arithmetic assignment operators.



# Operatori di Incremento e Decremento (1)

- Sono utilizzati per incrementare / decrementare il valore di una variabile che deve essere usato in un'espressione
  - Operatore di Preincremento / Predecremento:
    - l'operatore è posto prima della variabile
    - la variabile viene incrementata/decrementata e nell'espressione viene utilizzato il **nuovo valore**
  - Operatore di Postincremento / Postdecremento:
    - l'operatore è posto dopo la variabile
    - nell'espressione viene utilizzato il **vecchio valore** della variabile, e alla fine del calcolo dell'espressione, la variabile viene incrementata/decrementata

# Operatori di Incremento e Decremento (2)

| Operator | Called        | Sample expression | Explanation                                                                             |
|----------|---------------|-------------------|-----------------------------------------------------------------------------------------|
| ++       | preincrement  | ++a               | Increment a by 1, then use the new value of a in the expression in which a resides.     |
| ++       | postincrement | a++               | Use the current value of a in the expression in which a resides, then increment a by 1. |
| --       | predecrement  | --b               | Decrement b by 1, then use the new value of b in the expression in which b resides.     |
| --       | postdecrement | b--               | Use the current value of b in the expression in which b resides, then decrement b by 1. |

Fig. 8.13 increment and decrement operators.

```

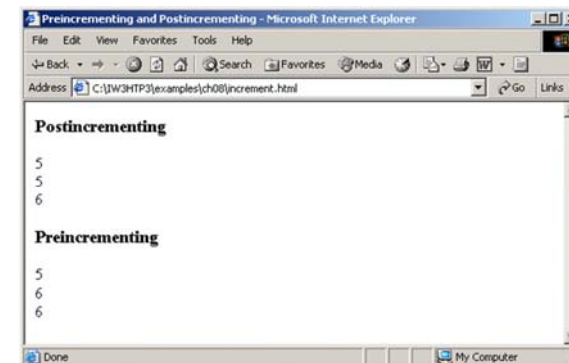
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 8.14: Increment.html -->
6 <!-- Preincrementing and Postincrementing -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Preincrementing and Postincrementing</title>
11
12 <script type = "text/javascript">
13 <!--
14 var c;
15
16 c = 5;
17 document.writeln("<h3>Postincrementing</h3>");
18 document.writeln(c); // print 5
19 // print 5 then Increment
20 document.writeln("
" + c++);
21 document.writeln("
" + c); // print 6
22
23 c = 5;
24 document.writeln("<h3>Preincrementing</h3>");
25 document.writeln(c); // print 5
 </script>
 </head>
</html>

```

```

26 // Increment then print 6
27 document.writeln("
" + ++c);
28 document.writeln("
" + c); // print 6
29 // ---
30 </script>
31
32 </head><body></body>
33 </html >

```



## Operatori di Incremento e Decremento: Precedenza e Associatività

| Operator         | Associativity | Type           |
|------------------|---------------|----------------|
| ++ --            | right to left | unary          |
| * / %            | left to right | multiplicative |
| + -              | left to right | additive       |
| < <= > >=        | left to right | relational     |
| == !=            | left to right | equality       |
| ?:               | right to left | conditional    |
| = += -= *= /= %= | right to left | assignment     |

Fig. 8.15 Precedence and associativity of the operators discussed so far.

## Tipizzazione dei Dati

- JavaScript è debolmente tipizzato
  - Fornisce una conversione automatica di valori di tipi diversi

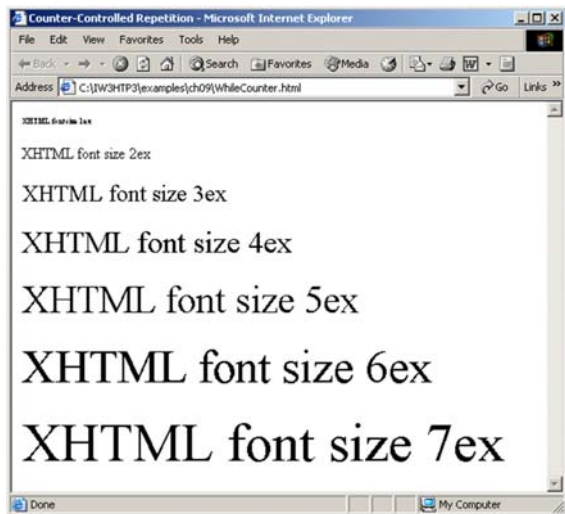
## Ripetizione controllata da un contatore

- Nome del contatore di controllo
- Valore iniziale
- Incremento / decremento
- Valore finale

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.1: WhileCounter.html -->
6 <!-- Counter-Controlled Repetition -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Counter-Controlled Repetition</title>
11
12 <script type = "text/javascript">
13 <!--
14 var counter = 1; // Initialization
15
16 while (counter <= 7) { // repetition condition
17 document.writeln("<p style = \"font-size: \" +
18 counter + \"ex\\>XHTML font size \" + counter +
19 \"ex</p>\");
20 ++counter; // Increment
21 }
22 // -->
23 </script>
24

```



## Ripetizione for (1)

- Gestisce i dettagli della ripetizione controllata da contatore

## Ripetizione for (2)

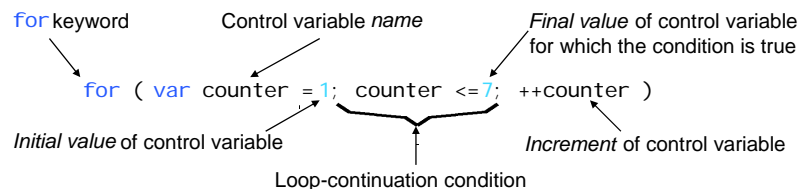


Fig. 9.3 For statement header components.

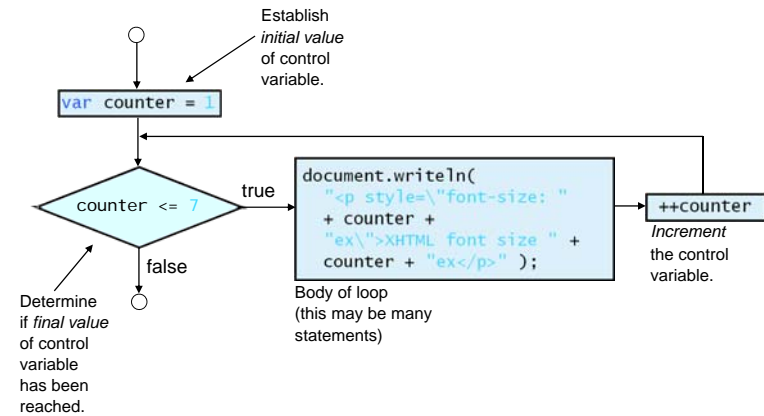
```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.2: ForCounter.html -->
6 <!-- Counter-Controlled Repetition with for statement -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Counter-Controlled Repetition</title>
11
12 <script type = "text/javascript">
13 <!--
14 // Initialization, repetition condition and
15 // incrementing are all included in the for
16 // statement header.
17 for (var counter = 1; counter <= 7; ++counter)
18 document.writeln("<p style = \"font-size: \" +
19 counter + \"ex\">XHTML font size \" + counter +
20 \"ex</p>\");
21 // -->
22 </script>
23
24 </head><body></body>
25 </html >

```



## Ripetizione for: Flow Chart



## Ripetizione for: Esempio

- Esempio: somma
  - Oggetto Math
    - Metodo pow
    - Metodo round

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.5: Sum.html -->
6 <!-- Using the for repetition statement -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Sum the Even Integers from 2 to 100</title>
11
12 <script type = "text/javascript">
13 <!--
14 var sum = 0;
15
16 for (var number = 2; number <= 100; number += 2)
17 sum += number;
18
19 document.writeln("The sum of the even integers " +
20 "from 2 to 100 is " + sum);
21 // -->
22 </script>
23
24 </head><body></body>
25 </html >

```



```

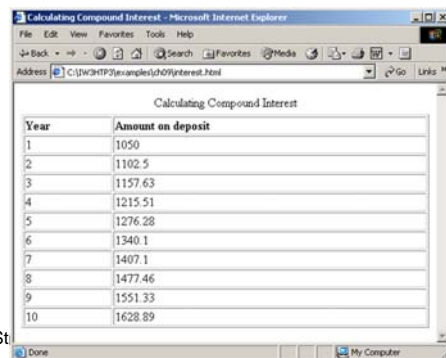
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.6: Interest.html -->
6 <!-- Using the for repetition statement -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Calculating Compound Interest</title>
11
12 <script type = "text/javascript">
13 <!--
14 var amount, principal = 1000.0, rate = .05;
15
16 document.writeln(
17 "<table border = '1' width = '100%'>");
18 document.writeln(
19 "<caption>Calculating Compound Interest</caption>");
20 document.writeln(
21 "<thead><tr><th align = 'left'>Year</th>");
22 document.writeln(
23 "<th align = 'left'>Amount on deposit</th>");
24 document.writeln("</tr></thead>");
25

```

```

26 for (var year = 1; year <= 10; ++year) {
27 amount = principal * Math.pow(1.0 + rate, year);
28 document.writeln("<tbody><tr><td> " + year +
29 "</td><td> " + Math.round(amount * 100) / 100 +
30 "</td></tr>");
31 }
32
33 document.writeln("</tbody></table>");
34 // -->
35 </script>
36
37 </head><body></body>
38 </html >

```



## Selezione Multipla switch

- Espressione di controllo
- Etichette dei possibili casi
- Caso di default

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.7: SwitchTest.html -->
6 <!-- Using the switch statement -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Switching between XHTML List Formats</title>
11
12 <script type = "text/javascript">
13 <!--
14 var choice, // user's choice
15 startTag, // starting list item tag
16 endTag, // ending list item tag
17 validInput = true, // indicates if input is valid
18 listType; // list type as a string
19
20 choice = window.prompt("Select a list style:\n" +
21 "1 (bullet), 2 (numbered), 3 (lettered)", "1");
22

```

```

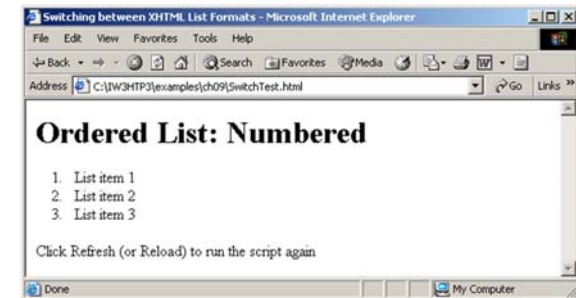
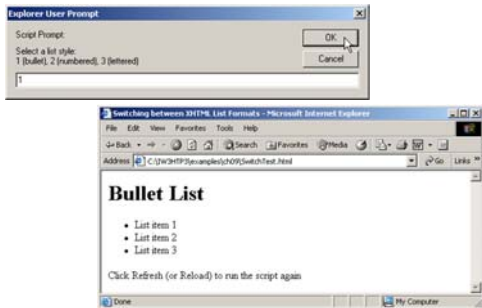
23 switch (choice) {
24 case "1":
25 startTag = "";
26 endTag = "";
27 listType = "<h1>Bullet List</h1>";
28 break;
29 case "2":
30 startTag = "";
31 endTag = "";
32 listType = "<h1>Ordered List: Numbered</h1>";
33 break;
34 case "3":
35 startTag = "<ol type = 'A'>";
36 endTag = "";
37 listType = "<h1>Ordered List: Lettered</h1>";
38 break;
39 default:
40 validInput = false;
41 }
42
43 if (validInput == true) {
44 document.writeln(listType + startTag);
45
46 for (var i = 1; i <= 3; ++i)
47 document.writeln("List item " + i + "");

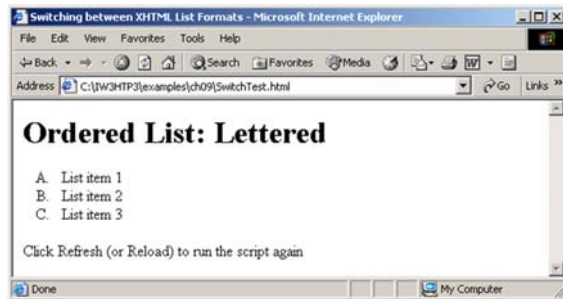
```

```

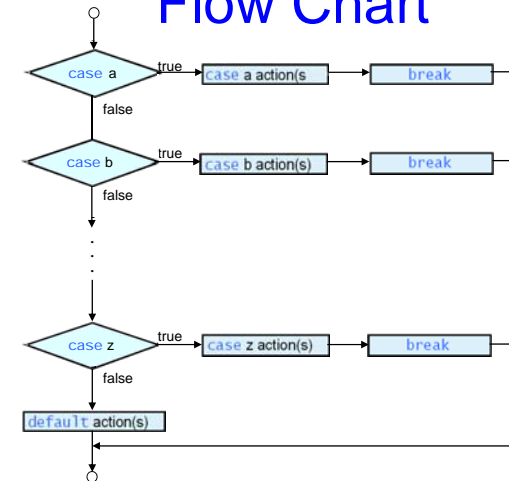
48 document.writeln(endTag);
49 }
50 }
51 else
52 document.writeln("Invalid choice: " + choice);
53 // -->
54 </script>
55
56 </head>
57 <body>
58 <p>Click Refresh (or Reload) to run the script again</p>
59 </body>
60 </html >

```





## Selezione Multipla switch : Flow Chart



## Ripetizione do...while

- Analoga alla while
- La condizione viene verificata dopo l'esecuzione del corpo del ciclo
- Il corpo del ciclo è sempre eseguito almeno una volta

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.9: DoWhileTest.html -->
6 <!-- Using the do...while statement -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Using the do...while Repetition Statement</title>
11
12 <script type = "text/javascript">
13 <!--
14 var counter = 1;
15
16 do {
17 document.writeln("<h" + counter + ">This is " +
18 "an h" + counter + " level head" + "</h" +
19 counter + ">");
20
21 ++counter;
22 } while (counter <= 6);
23 // -->
24 </script>

```

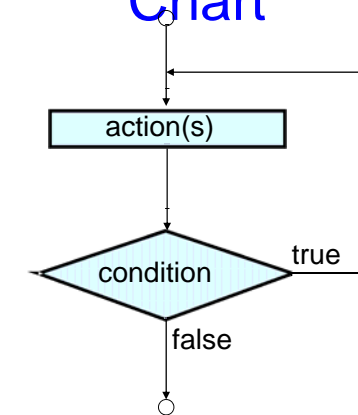
```

25
26 </head><body></body>
27 </html >

```



## Ripetizione do...whi l e : Flow Chart



## break e conti nue

- break
  - Forza l'uscita immediata da una struttura
  - Salta ciò che rimane dell'istruzione swi tch
- conti nue
  - Salta ciò che rimane dell'istruzione swi tch
  - Continua con la successiva iterazione di un ciclo

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.11: BreakTest.html -->
6 <!-- Using the break statement -->
7
8 <html xml ns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>
11 Using the break Statement In a for Structure
12 </title>
13
14 <scri pt type = "text/javascript">
15 <!--
16 for (var count = 1; count <= 10; ++count) {
17 if (count == 5)
18 break; // break loop only if count == 5
19
20 document.wri tel n("Count is: " + count + "
");
21 }
22

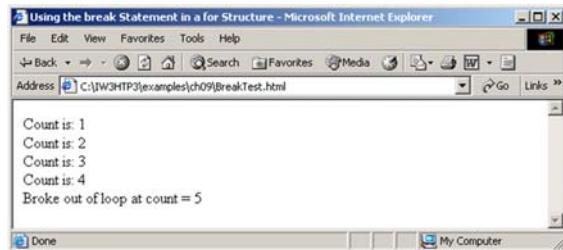
```



```

23 document.writeln(
24 "Broke out of loop at count = " + count);
25 // -->
26 </script>
27
28 </head><body></body>
29 </html >

```



JavaScript: Strutture di Controllo

65

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.12: ContinueTest.html -->
6 <!-- Using the break statement -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>
11 Using the continue Statement In a for Structure
12 </title>
13
14 <script type = "text/javascript">
15 <!--
16 for (var count = 1; count <= 10; ++count) {
17 if (count == 5)
18 continue; // skip remaining code in loop
19 // only if count == 5
20
21 document.writeln("Count is: " + count + "
");
22 }
23

```

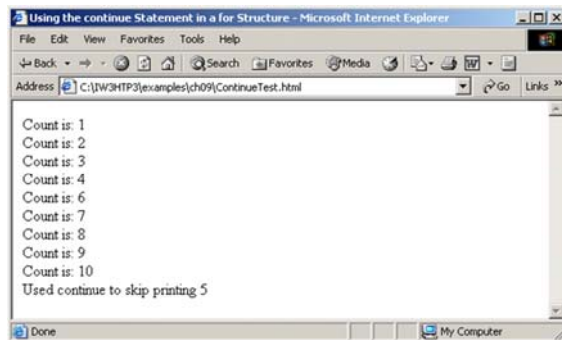
JavaScript: Strutture di Controllo

66

```

24 document.writeln("Used continue to skip printing 5");
25 // -->
26 </script>
27
28 </head><body></body>
29 </html >

```



JavaScript: Strutture di Controllo

67

```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.13: BreakLabelTest.html -->
6 <!-- Using the break statement with a Label -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Using the break Statement with a Label</title>
11
12 <script type = "text/javascript">
13 <!--
14 stop: { // labeled block
15 for (var row = 1; row <= 10; ++row) {
16 for (var column = 1; column <= 5; ++column) {
17
18 if (row == 5)
19 break stop; // jump to end of stop block
20
21 document.write(" ");
22 }
23
24 document.writeln("
");
25 }

```

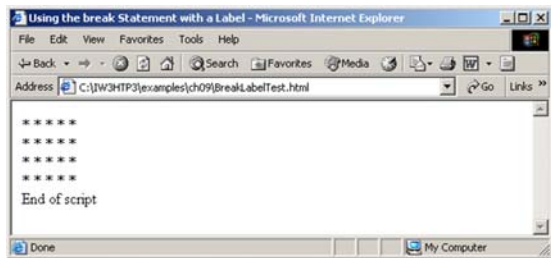
JavaScript: Strutture di Controllo

68

```

26
27 // the following line is skipped
28 document.writeln("This line should not print");
29 }
30
31 document.writeln("End of script");
32 // -->
33 </script>
34
35 </head><body></body>
36 </html >

```



```

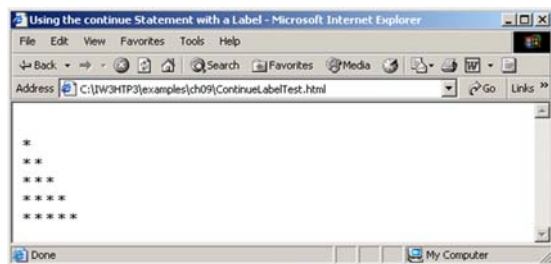
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.14: ContinueLabel Test.html -->
6 <!-- Using the continue statement -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Using the continue Statement with a Label</title>
11
12 <script type = "text/javascript">
13 <!--
14 nextRow: // target label of continue statement
15 for (var row = 1; row <= 5; ++row) {
16 document.writeln("
");
17
18 for (var column = 1; column <= 10; ++column) {
19
20 if (column > row)
21 continue nextRow; // next iteration of
22 // labeled loop
23
24 document.write(" ");
25 }

```

```

26 }
27 // -->
28 </script>
29
30 </head><body></body>
31 </html >

```



## Operatori Logici (1)

- AND Logico ( && )
- OR Logico ( || )
- NOT Logico ( ! )

## Operatori Logici (2)

| expression1 | expression2 | expression1 && expression2 |
|-------------|-------------|----------------------------|
| false       | false       | false                      |
| false       | true        | false                      |
| true        | false       | false                      |
| true        | true        | true                       |

Fig. 9.15 Truth table for the && (logical AND) operator.

## Operatori Logici (3)

| expression1 | expression2 | expression1    expression2 |
|-------------|-------------|----------------------------|
| false       | false       | false                      |
| false       | true        | true                       |
| true        | false       | true                       |
| true        | true        | true                       |

Fig. 9.16 Truth table for the || (logical OR) operator.

| expression | ! expression |
|------------|--------------|
| false      | true         |
| true       | false        |

Fig. 9.17 Truth table for operator ! (logical negation).

```

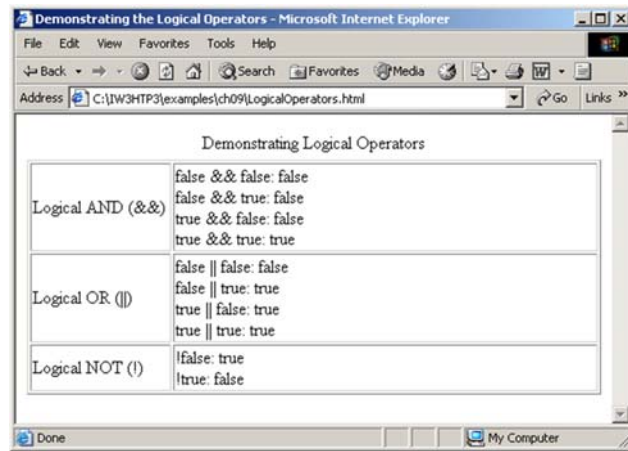
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 9.18: Logical Operators.html -->
6 <!-- Demonstrating Logical Operators -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Demonstrating the Logical Operators</title>
11
12 <script type = "text/javascript">
13 <!--
14 document.writeln(
15 "<table border = \"1\" width = \"100%\">");
16
17 document.writeln(
18 "<caption>Demonstrating Logical " +
19 "Operators</caption>");
20
21 document.writeln(
22 "<tr><td width = \"25%\">Logical AND (&&)</td> +
23 "<td>false && false: " + (false && false) +
24 "
false && true: " + (false && true) +
25 "
true && false: " + (true && false) +

```

```

26 "
true && true: " + (true && true) +
27 "</td>");
28
29 document.writeln(
30 "<tr><td width = \"25%\">Logical OR (||)</td> +
31 "<td>false || false: " + (false || false) +
32 "
false || true: " + (false || true) +
33 "
true || false: " + (true || false) +
34 "
true || true: " + (true || true) +
35 "</td>");
36
37 document.writeln(
38 "<tr><td width = \"25%\">Logical NOT (!)</td> +
39 "<td>! false: " + (! false) +
40 "
! true: " + (! true) + "</td>");
41
42 document.writeln("</table>");
43 // -->
44 </script>
45
46 </head><body></body>
47 </html >

```



## Operatori Logici: Precedenze e Associatività

| Operator         | Associativity | Type           |
|------------------|---------------|----------------|
| ++ -- !          | right to left | unary          |
| * / %            | left to right | multiplicative |
| + -              | left to right | additive       |
| < <= > >=        | left to right | relational     |
| == !=            | left to right | equality       |
| &&               | left to right | logical AND    |
|                  | left to right | logical OR     |
| ?:               | right to left | conditional    |
| = += -= *= /= %= | right to left | assignment     |

Fig. 9.19 Precedence and associativity of the operators discussed so far.

## Web Resources

- [www.javascriptal.com](http://www.javascriptal.com)
- [developer.netscape.com/tech/javascript](http://developer.netscape.com/tech/javascript)
- [www.mozilla.org/js/language](http://www.mozilla.org/js/language)