Chapter 3

The essential JavaScript statements

Objectives

- How to code conditional expressions
- How to code the basic control statements
- Three illustrative applications
- How to work with arrays
- The Test Scores application with an array

How to code conditional expressions

Conditional Expression

- ***Conditional expression*** are expressions that evaluate to either true or false.
- It uses in if statement and loop statement.
- **Relational operators** is used to compare the result of two expression and return Boolean value.
- **Logical operators** is used to join two or more conditional expression to be a compound conditional expression.

The relational operators

Operator	Name	Description
==	Equality	Returns a true value if both operands are equal.
!=	Inequality	Returns a true value if the left and right operands are not equal.
>	Greater Than	Returns a true value if the left operand is greater than the right operand.
<	Less Than	Returns a true value if the left operand is less than the right operand.
>=	Greater Than Or Equal	Returns a true value if the left operand is greater than or equal to the right operand.
<=	Less Than Or Equal	Returns a true value if the left operand is less than or equal to the right operand.

Example:

```
lastName == "Harris"
testScore == 10

firstName != "Ray"
months != 0

age < 18

investment <= 0

testScore > 100

rate / 100 >= 0.1
```

The logical operators

Operator	Description	Example
`!`	NOT	`!isNaN(age)`
`&&`	AND	`age > 17 && score < 70`
`||`	OR	`isNaN(rate) || rate < 0`

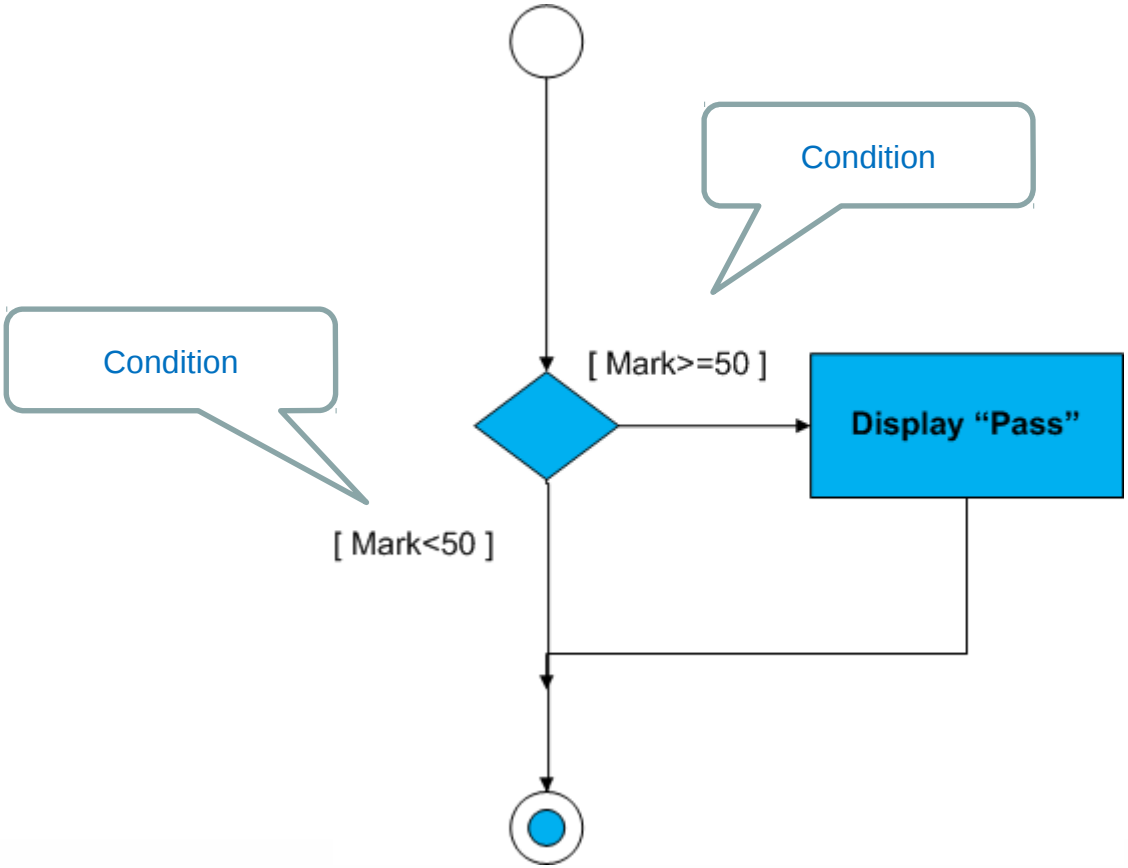
How to code
the basic control statements

IF statement

- Syntax:

```
if (booleanExpression) {
        statements;
}
If (booleanExpression) {
        statements1;
}else{
            statements2;
}
```

IF statement (cont.)



IF statement (cont.)

An if statement with an else clause

```
if ( age >= 18 ) {
    alert ("You may vote.");
} else {
    alert ("You are not old enough to vote.");
}
```

An if statement with else if and else clauses

```
if ( isNaN(rate) ) {
    alert ("You did not provide a number for the rate.");
} else if ( rate < 0 ) {
    alert ("The rate may not be less than zero.");
} else if ( rate > 12 ) {
    alert ("The rate may not be greater than 12.");
} else {
    alert ("The rate is: " + rate + ".");
}
```

IF statement (cont.)

An if statement with a compound conditional expression

```
if ( isNaN(userEntry) || userEntry <= 0 ) {
    alert ("Please enter a valid number greater than zero.");
}
```

Two ways to test whether a Boolean variable is true

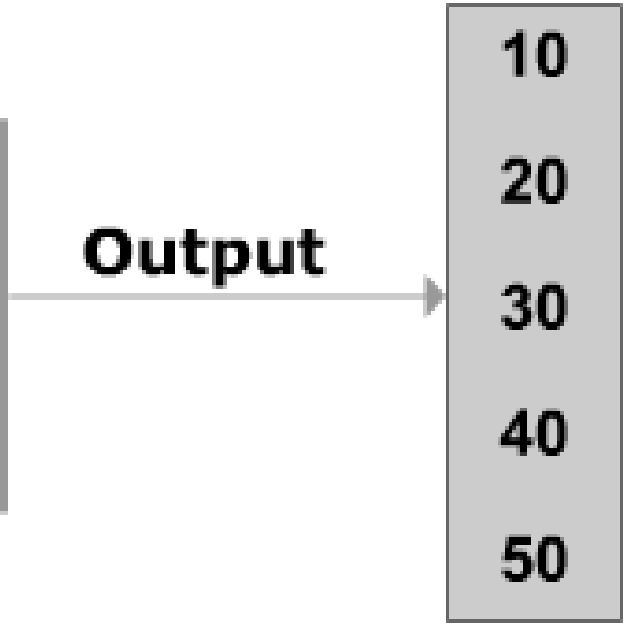
```
if ( isValid == true ) { }
if ( isValid ) { }                    // same as isValid == true
```

Three ways to test whether a Boolean variable is false

```
if ( isValid == false ) { }
if ( !isValid == true ) { }
if ( !isValid ) { }                   // same as !isValid == true
```

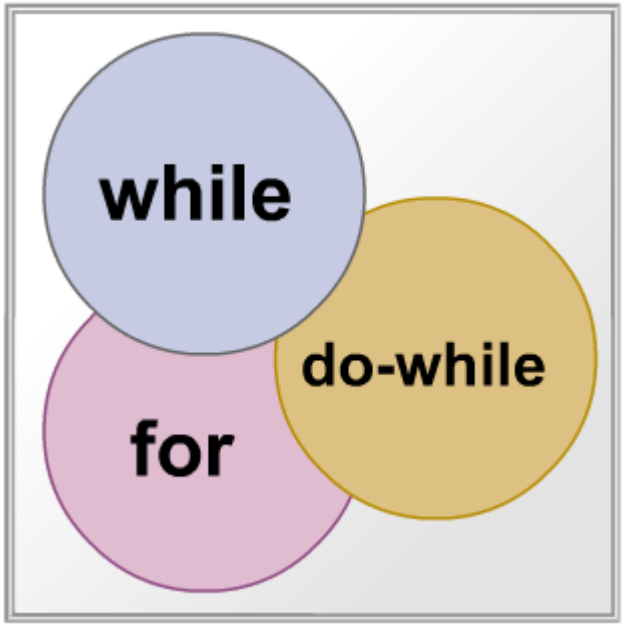
Loops

```
BEGIN
    COMPUTE result1 AS 1 * 10
    COMPUTE result2 AS 2 * 10
    COMPUTE result3 AS 3 * 10
    COMPUTE result4 AS 4 * 10
    COMPUTE result5 AS 5 * 10
END
```



Types of Loops

- A loop comprises a statement or block of statement s that are execute repeatedly until a particular condition evaluate to true or false



while loop

- **while** loop statement is used to execute a statement or block of statement while a particular condition is true

The syntax of a while loop

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

A while loop that adds the numbers from 1 through 5

```
var sumOfNumbers = 0;
var numberOfLoops = 5;
var counter = 1;
while (counter <= numberOfLoops) {
    sumOfNumbers += counter;     // adds counter to sumOfNumbers
    counter++;                   // adds 1 to counter
}
alert(sumOfNumbers);             // displays 15
```

do-while Statement

- The do-while statement check the condition at the end of the loop rather than at the beginning to assure that the loop is executed at least once.

The syntax of a do-while loop

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

A while loop that adds the numbers from 1 through 5

```
var sumOfNumbers = 0;
var numberOfLoops = 5;
var counter = 1;
do {
    sumOfNumbers += counter;     // adds counter to sumOfNumbers
    counter++;                   // adds 1 to counter
}
while (counter <= numberOfLoops);
alert(sumOfNumbers);             // displays 15
```

for Statement

The syntax of a for statement

```
for ( counterInitialization; condition; incrementExpression ) {
    statements
}
```

A for loop that adds the numbers from 1 through 5

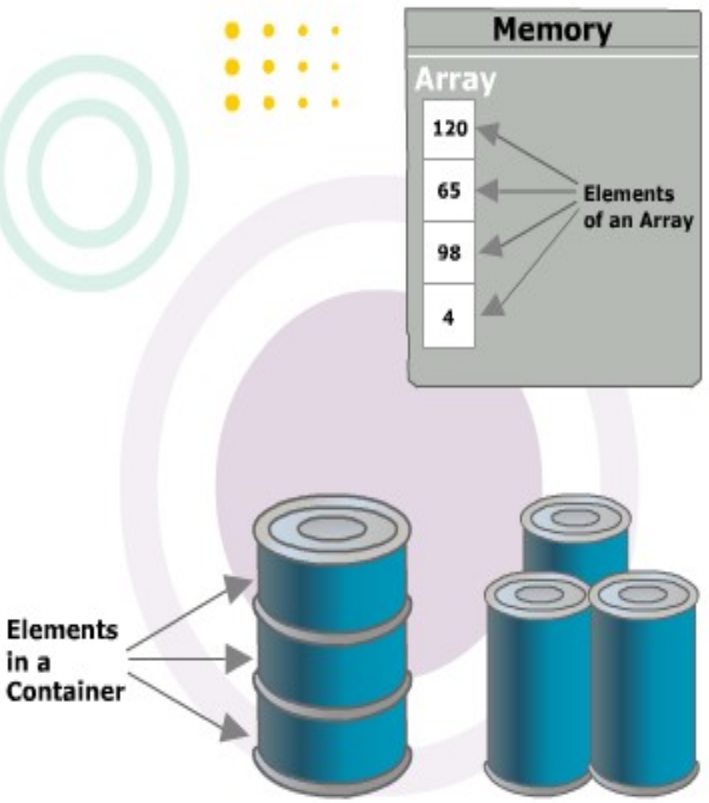
```
var sumOfNumbers = 0;
var numberOfLoops = 5;
for ( counter=1; counter <= numberOfLoops; counter++ ) {
    sumOfNumbers += counter;      // adds counter to sumOfNumbers
}
alert(sumOfNumbers);              // displays 15
```

Three illustrative applications (Page 96-101)

How to work with arrays

Introduction to Array

- An array can store one or more elements
- The length of an array is the number of elements in the array
- Each of the element in the array is accessible through an integer value called a subscript
- An array subscript begins from zero and is also called array index



How to create and use arrays

The syntax for creating an array

Using the new keyword with the Array object name

```
var arrayName = new Array(length);
```

Using the brackets literal

```
var arrayName = [];
```

How to create an array and assign values in one statement

```
var rates = new Array(14.95, 12.95, 11.95, 9.95);
var names = ["Ted Lewis", "Sue Jones", "Ray Thomas"];
```

The syntax for referring to an element of an array

```
arrayName[index]
```

Code that refers to the elements in an array

```
rates[2]          // Refers to the third element in the rates array
names[1]          // Refers to the second element in the names array
```

How to use for loops to work with arrays

Code that puts the numbers 1 through 10 into an array

```
var numbers = [];
for (var i = 0; i < 10; i++) {
    numbers[i] = i + 1;
}
```

Code that displays the numbers array created above

```
var numbersString = "";
for (var i = 0; i < numbers.length; i++) {
    numbersString += numbers[i] + " ";
}
alert (numbersString);
```

The message that's displayed

How to use for loops to work with arrays (cont.)

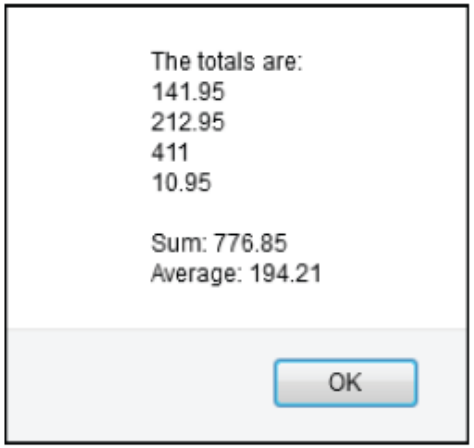
Code that computes the sum and average of an array of totals

```
var totals = [141.95, 212.95, 411, 10.95];
var sum = 0;
for (var i = 0; i < totals.length; i++) {
    sum += totals[i];
}
var average = sum / totals.length;
```

Code that displays the totals array, the sum, and the average

```
var totalsString = "";
for (var i = 0; i < totals.length; i++) {
    totalsString += totals[i] + "\n";
}
alert ("The totals are:\n" + totalsString + "\n" +
       "Sum: " + sum.toFixed(2) + "\n" + "Average: " + average.toFixed(2) );
```

The message that's displayed



The Test Score application with an array (Page 106-107)

Summary

- ***Conditional expression*** are expressions that evaluate to either true or false. It uses in if statement and loop statement.
- **A loop** comprises a statement or block of statement s that are execute repeatedly until a particular condition evaluate to true or false.
- An **array** can store one or more elements. The length of an array is the number of elements in the array
- Each of the element in the array is accessible through an integer value called a subscript.