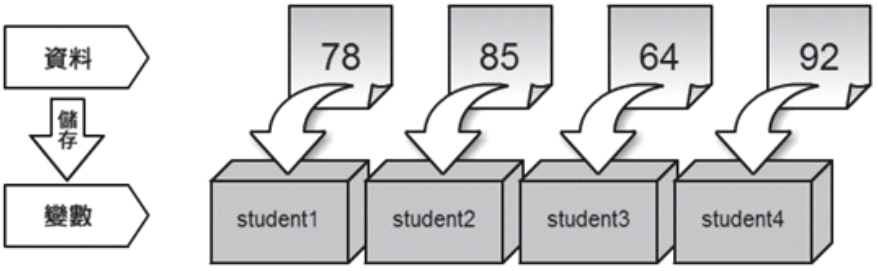
20.JavaScript - 陣列與物件

一、陣列的使用

（一）認識陣列

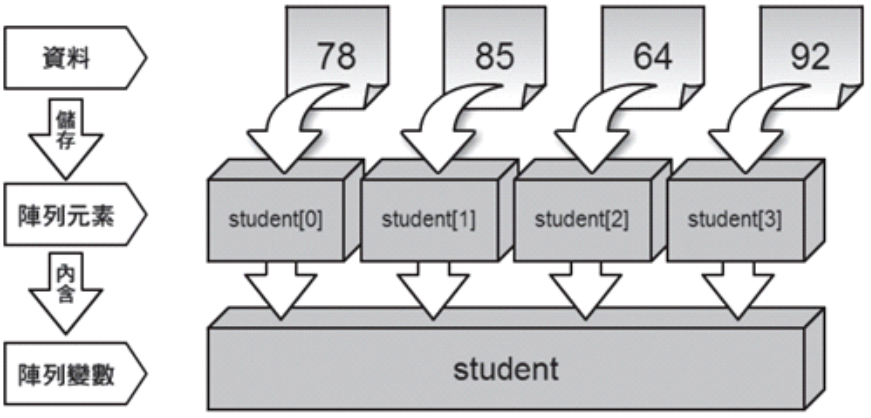
- 為什麼要使用陣列？

  程式中的資料通常是以變數來儲存，如果有大量的同類型資料需要儲存時，必須宣告大量的變數，不但耗費程式碼，執行效率也不佳。



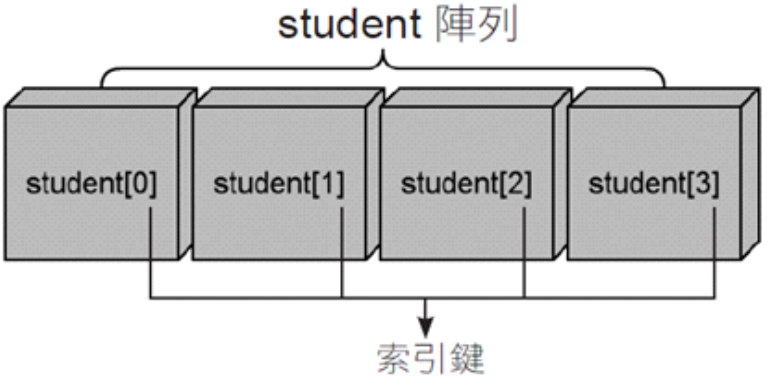
- **陣列儲存資料的方式**

  陣列可說是一群性質相同變數的集合，相同陣列中擁有一個變數名稱，做為識別該陣列的標誌；陣列中的每一份資料稱為：陣列元素，每一個陣列元素相當於一個變數，如此就可輕易建立大量的資料儲存空間。



- **陣列資料的識別方式**

  要如何區分放置在陣列中的資料呢？在預設的狀態下是使用 索引鍵 值。索引鍵允許使用整數或是字串，如果未指定索引鍵值，程式會自動由 0 開始計算，也就是在陣列中放置的第一個元素的索引鍵為 0，第二個元素的索引鍵為 1，以此類推，第 n 個元素的索引鍵即為 n-1。



（二）建立一維陣列

- **一維陣列的建立方式**

  在JavaScript 中是利用 Array() 建立物件，例如要建立一個名為 student，共有3 個元素的一維陣列，接著再一一指定其中的值，方法如下：

```
var student = new Array(3);
student[0] = "David";
student[1] = "Lily";
student[2] = "Perry";
```

  也可以直接在建立物件時就指派元素值，方法如下：

```
var student = new Array("David","Lily","Perry");
```

  也可以善用 [ ] 符號來建立物件並指派元素值，方法如下：

```
var student = ["David","Lily","Perry"];
```

- **取用陣列中的資料**

  建立陣列物件並設定元素值後，可以利用陣列的元素索引值取得其中的值，要特別注意的是，陣列元素的索引值是由0 算起。

  例如若要在頁面上顯示 student 陣列中第一筆資料，方法如下：

```
document.write(student[0]);
```

  如果想知道目前這個陣列中到底有多少元素，可以使用 .length 屬性來取得，例如想要知道 student 這個陣列的元素數量，方法如下：

```
student.length
```

  知道如何取得陣列元素的數量之後，就能將陣列中的所有元素值顯示出來，例如我們想將 student 陣列中所有的元素顯示出來，這裡可以利用for 迴圈來執行，方法如下：

```
for(var i=0; i < student.length; i++) {
 document.write(student[i] + "<br/>");
}
```

  這裡還可以使用 for/in 迴圈，這是特別為顯示物件內容所設計的迴圈，方式如下：

```
for ( i in student) {
 document.write(student[i] + "<br/>");
}
```

範例：

21-01.html

```
<script>
    let student = ["David","Lily","Perry"];
    document.write("<table border='1'><tr><td>編號</td><td>姓名</td></tr>");
    for (let i = 0; i < student.length; i++) {
        document.write("<tr><td>" + (i+1) + "</td><td>" + student[i] + "</td></tr>");
    }
    document.write("</table>");
</script>
```

（三）建立多維陣列

其實無論建立幾維的陣列，採取的方式都與建立二維陣列相似，都可以由一維組成二維，再架構到多維的陣列。例如班級中每個同學都有三科成績，如下可以整理成一個 3 列 X 3 行的表格

班級資料

	第 1 行	第 2 行	第 3 行
	姓名	國文	英文
第 1 列 →	David	95	80
第 2 列 →	Lily	82	85
第 3 列 →	Perry	90	92

要特別注意的是陣列中的索引值是由0 開始計算，下表是二維陣列中二個索引值的表示方式

Student 陣列

	第 1 行	第 2 行	第 3 行
	姓名	國文	英文
第 1 列 →	[0][0]	[0][1]	[0][2]
第 2 列 →	[1][0]	[1][1]	[1][2]
第 3 列 →	[2][0]	[2][1]	[2][2]

接著在程式中建立這二維陣列並設定值，方法如下：

```
var student=new Array();
student[0]=new Array(), student[1]=new Array(), student[2]=new Array();
student[0][0]="David", student[0][1]="95", student[0][2]="80";
student[1][0]="Lily", student[1][1]="82", student[1][2]="85";
student[2][0]="Perry", student[2][0]="90", student[2][0]="92";
```

也可以使用另外一種建立的方式，更加的簡潔：

```
var student = new Array();
student[0]=["David", "95", "80"];
student[1]=["Lily", "82", "85"];
student[2]=["Perry", "90", "92"];
```

範例：

21-02.html

```
<script>
let student = new Array();
student[0]=["David", "95", "80"];
student[1]=["Lily", "82", "85"];
student[2]=["Perry", "90", "92"];
document.write("<table border='1'><tr><td>姓名</td><td>國文</td><td>英文</td></tr>");
for (let i=0; i<student.length; i++) {
    document.write("<tr>");
    for (let j=0; j<student[i].length; j++) {
        document.write("<td>" + student[i][j] + "</td>");
    }
    document.write("</tr>");
}
document.write("</table>");
</script>
```

（三）陣列相關函式

- length - 計算陣列長度。
- index - 指定陣列索引鍵字元。
- push() - 在陣列最後面增加一個值。
- pop() - 刪除陣列最後面的值。
- shift() - 刪除陣列最前面的值。
- unshift() - 在陣列最前面增加一個值。

二、物件

（一）認識物件

- 物件的屬性與方法

  物件可以說是屬性與方法的組合。以實際的範例來說，在生活週遭其實就充滿了物件，桌上的杯子、書本是物件，路上的車子、房子也都是物件。每個物件都有它的特徵與功能，化為程式術語就是屬性與方法。就以一個人來說：人有姓名、有身高、有體重，這些特徵都是人的屬性；人能說話、唱歌、走路，這些能力都是人的方法。

- **屬性與方法的使用**

  JavaScript 的中也充滿了各種物件：瀏覽器視窗、網頁、字串、數字、日期等，每個物

件也擁有屬於它的屬性及行為。例如，在本章中談到陣列時為了要得到student陣列中的元素數量，可以使用：

```
student.length;    //length 屬性的使用
```

若要將指定的訊息顯示在頁面上或是顯示彈出式視窗，可以使用：

```
document.write('hello');  //write() 方法的使用
window.alert('hello');    //alert() 方法的使用
```

在物件中就是使用「.」點符號來與屬性、方法連繫在一起，屬性與方法之間其實很好區分，因為方法的後面會加上() 括號。

（二）自訂物件的建立與使用

- **建立自訂物件**

在JavaScript 中也能建立自訂的物件，例如想要新增一個名為Person 的物件並設定基本的屬性，首先是基本物件建立方式：

```
var Person = new Object();
Person.name="David";
Person.age=25;
Person.weight=75;
Person.height=180;
```

也可以用以下的替代方式：

```
var Person = {name:"David", age:25, weight:75, height:180};
```

如果建立物件的動作很頻繁，可以將建立物件的動作化為函式，方式如下：

```
function Person(name, age, weight, height){
 this.name = name;
 this.age = age;
 this.weight = weight;
 this.height = height;
}
person1 = new Person("David", 25, 75, 180);  // 新增一個 Person 物件
```

- **取得物件屬性**

如果要取得自訂物件中的屬性，可以使用以下二種方式：

```
person.name;
person["name"];
```

- **在自訂物件中建立方法**

  在自訂物件中除了屬性之外，也能自訂方法，例如我們在Person 中新增一個名為sayHello() 的方法，方式如下：

```
var Person = {
 name:"David", age:25, weight:75, height:180,
 sayHello:function(){return "Hello, my name is " + name;}
};
document.write(person1.sayHello());  // 執行自訂物件中的方法
```

  也可以在建立物件函式裡加入自訂的方法，方式如下：

```
function Person(name, age, weight, height){
 this.name = name; this.age = age;
 this.weight = weight; this.height = height;
 this.sayHello = function(){return "Hello, my name is " + name;}
};
person1 = new Person("David", 25, 75, 180);  // 新增自訂物件
document.write(person1.sayHello());  // 執行自訂物件中的方法
```

範例：

21-03.html

```
<script>
    function Person(name, age, weight, height){
        this.name = name;
        this.age = age;
        this.weight = weight;
        this.height = height;
        this.sayHello = function(){
            return "您好，我是"+name +"，今年"+age+ "歲，身高"+height+"公分，體重"+weight+"公斤。";
        }
    };
    person1 = new Person("David", 25, 75, 180);
    person2 = new Person("Ken", 20, 65, 175);
    document.write(person1.sayHello() + "<br/>"+ person2.sayHello());
</script>
```