JavaScript 程式設計新手村

單元12 JavaScript 流程控制

@kdchang

Outline

1. 流程控制基礎
2. if...else
3. switch
4. for loop
5. while
6. do...while

流程控制基礎

一般而言，JavaScript 程式是由上往下依序執行下去，在功能上若需要依條件不同而改變執行順序，這時候流程控制就扮演關鍵角色

1. 循序結構（Sequential）
   預設的程式執行方式，亦即一個程式一個程式執行下來
2. 選擇結構（Selection）
   是一種條件控制，亦即選擇題，可分為單選或二選一或多選
3. 重複結構（Iteration）
   亦即迴圈控制，可以重複執行程式區塊，直到符合結束條件

流程控制基礎

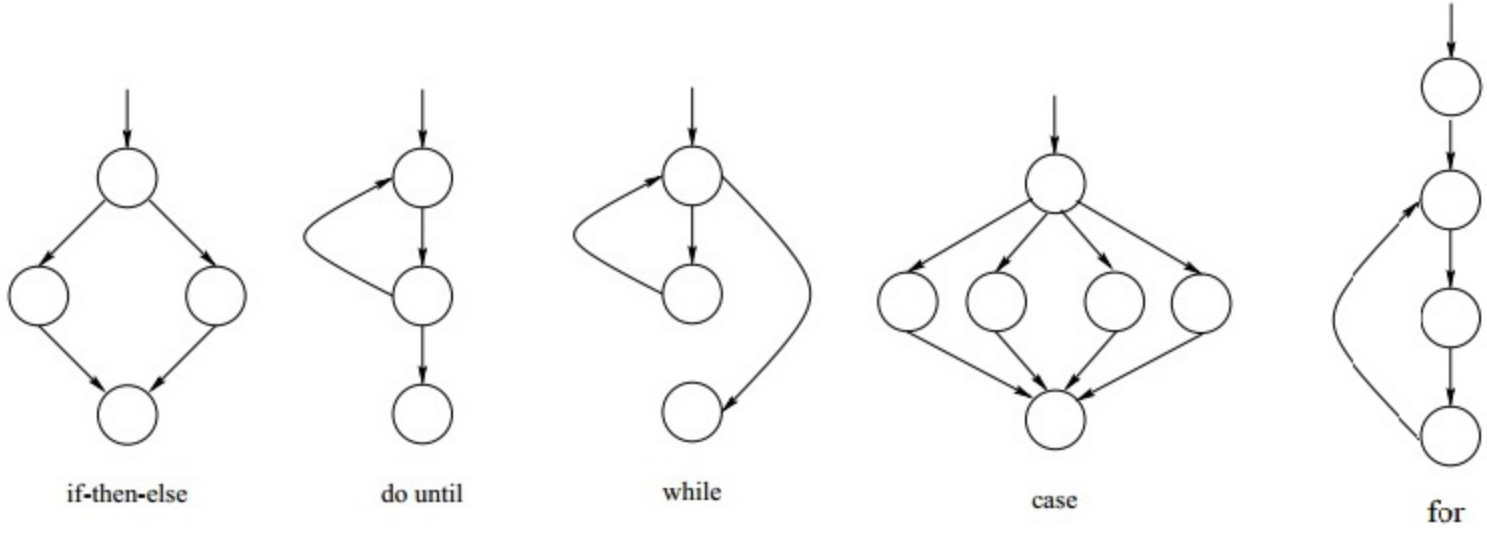


Figure 1: Flow graph representation.

imave via geekdetected

邏輯運算子

1. `!` 表非(取相反)
2. `&&` 表邏輯且
3. `||` 表邏輯或

* 如果使用 && 連結兩運算子來給定變數時，當第一個運算元為 true，則回傳第二個運算元。反之，則回傳第一個運算元

```
console.log(z = 1 === 1 && 1 === 2);
```

* 如果使用 || 連結兩運算子，第一個運算元為 true，則回傳第一個運算元。反之，則回傳第二個運算元

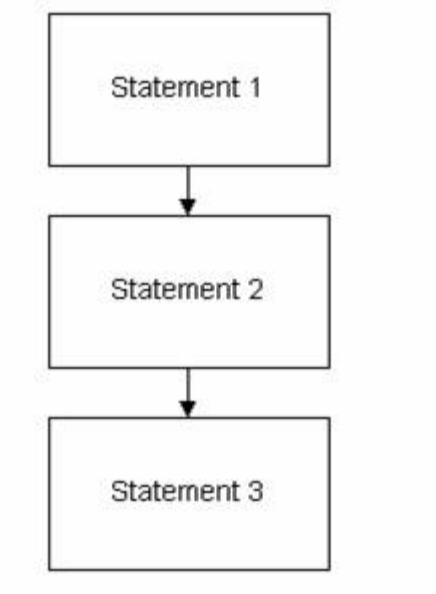
```
console.log(z = 1 === 1 || 1 === 2);
```

比較運算子

關係運算子	說明	範例	範例結果
>	是否大於	5 > 2	true
>=	是否大於等於	5 >= 2	true
<	是否小於	5 < 2	false
<=	是否小於等於	5 <= 2	false
==	是否等於	5 == 2	false
!=	是否不等於	5 != 2	true
===	是否等於（型別必須相同）	5 === "5"	false
!==	是否不等於	5 !== "5"	true

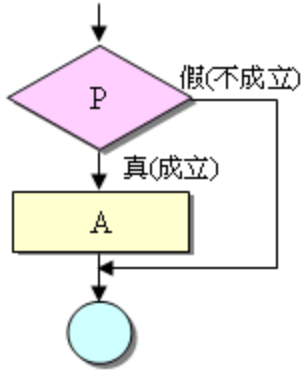
image via pcstar

循序結構

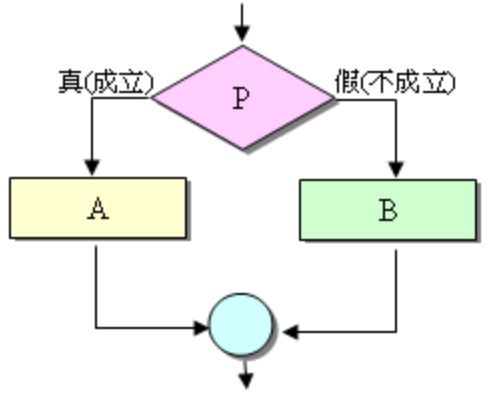


一般情況，JavaScript 程式是由上往下依序執行

選擇結構



(a)假如 P 成立則執行 A 程序



(b)假如 P 成立則執行 A 程序，否則執行 B 程序

if 是非題

if 條件敘述是一種是否執行的是非題，如果符合條件運算結果為true則執行程式區塊（括號內程式碼）

```
const grades = 77;
if(grades > 60){
        console.log('真假，你竟然及格了！？');
} //若區塊內只有一行敘述可以不用大括號，但建議使用 {} 方便閱讀
```

if / else 二選一

若有兩個區塊，則可以變成二選一的選擇題。符合條件執行第一個區塊，不符合則執行第二個區塊

```
const age = 20;

if(age >= 20){
  console.log('你可以投票了！');
}
else {
  console.log('小朋友，你還小喔，別急');
}
```

if / else if / else 多選一

若需要有多選一的情況，則可以運用 if/else if/else 建立多選一的條件選擇

```
const grade = 60;

if (grade >= 90 && grade <= 100) {
        console.log('A+');
} else if (grade >= 80 && grade < 90) {
        console.log('A');
} else {
        console.log('A-');
}
```

練習一

參考前面範例做一個收入判斷程式。當成績大於 90 - 100 為甲上，80 - 90 為甲，70 - 80 以上為乙

Switch 多選一

建立多選一的另一種方式為使用switch條件敘述，若是條件非邏輯判斷且選擇很多，建議使用switch，避免過多的if else判斷導致程式閱讀不易

- 請注意break敘述為必須，才會在符合條件執行完該區塊後跳出迴圈，否則會依序一直執行下去，至於最後的default值為選擇性，主要是提供都不符合條件執行的區塊

Switch 多選一

```
const planet = '地球';

switch(planet) {
  case '地球':
    console.log('地球好！');
    break;
  case '火星':
    console.log('火星很危險地');
    break;
  case '水星':
    console.log('水星沒有人');
    break;
  default:
    console.log('地球人');
}
```

break / continue

除了依照條件控制程式的流程外，我們也可以使用 `break` 和 `continue` 來控制程式的流程。注意的是兩者只能在程式區塊中使用

1. break
   當某些條件成立時，強迫迴圈終止，跳出迴圈(程式區塊)，如同我們之前看到switch在條件符合時整個跳出迴圈
2. continue
   和break不同的是continue不會完全跳出程式區塊，只會結束這次迴圈，直接開始下次的迴圈

break / continue

```
for(let i = 0; i < 10; i++) {
        if(i === 5) {
                break; // continue
        }
    console.log(i);
}
```

練習二

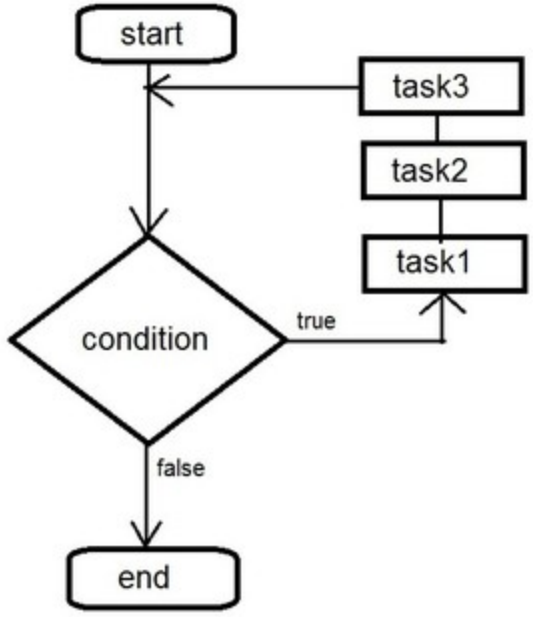
使用 switch 判斷輸入的中文對照以下英文單字輸出在 console 中，
運動：sport，寫程式：coding，閱讀：reading

三元運算子 ? :

三元運算為簡化的條件判斷，其中 `?` 代表 `if`，`:` 代表 `else`

```
const age = 30;
const vote = (age >= 20) ? "成年" : "未成年";
console.log(vote);
```

重複結構（前測、後測）



前測試重複結構

for 迴圈

for迴圈可以透過計數器的增加減少執行固定次數的程式區塊，由於是在迴圈開始前檢查條件，故稱前測迴圈（若不明確知道要做幾次就使用 while）

```
for (起始值; 條件式; 更新值) {
  執行指令;
}
```

範例：

```
for(let i = 0; i < 10; i++) {
        console.log(i);
}
```

while 迴圈

while迴圈同樣也是前測迴圈，但和 for 迴圈不同的地方是，while 需要自行在程式區塊內處理計數器的增減，若符合條件(true)才進入迴圈內，否則(false)則離開迴圈

＊請注意不要讓while變成無窮迴圈 (1.忘記加計數器 2.讓條件永遠成立 EX. while(1) or while(true))避免系統耗盡當機

```
let sum = 0;
let i = 1;
while(i <= 10){
  sum += i;
  i++;
}

console.log(sum);
```

練習四

請使用 for 和 while 各做一個計算從 1 加到 10 的程式

後測式程式結構

do while 迴圈

do/while 迴圈和 while 類似，只是do/while迴圈是在迴圈結尾才檢查條件是否符合，亦即迴圈區塊`至少會執行一次`

```
let i = 11;
let sum = 0;

do {
  sum += i;
  i++;
} while(i <= 10);

console.log(sum);
```

巢狀迴圈

巢狀迴圈亦即在迴圈中擁有其他迴圈，EX. while 中有 for，for 中有 while迴圈

```
for(var i = 1; i <=9; i++){
  console.log(i + ":");
  var j = 1;
  while(j <= 9){

    console.log(i * j);
    j++;
  }
}
```

練習五

請使用巢狀迴圈製作一個九九乘法表

總結

在這個單元我們學會了流程控制基礎，讓我們的程式可以依條件不同而改變執行順序

1. 流程控制基礎
2. if...else
3. switch
4. for loop
5. while
6. do...while