

scratch <https://scratch.mit.edu/>

mblock <http://www.mblock.cc/zh-home/software/>

學校舊網站: <http://163.23.76.129/>

學校新網站: <https://www.caes.chc.edu.tw>

## 一、中小學生為何要學程式語言

賴健二(2002)指出，中小學學生應該學習程式語言，因為學習程式語言可以增進其邏輯思考能力、組織能力和問題解決能力；豐佳燕與陳明溥(2008)在國小學生學習電腦程式的研究上指出九成以上的學生認為程式不是那樣令人畏懼的；簡易而功能強大的兒童程式 Logo 的發展者 Papert(1980)認為兒童學習程式設計可以使兒童思考更有條理,並提升邏輯判斷能力。

## 二、有哪些產品可供學習

許銘津與劉明洲(1993)的研究指出要發展一套電腦教學軟體,在教學法、教材的安排和人機介面的設計等過程都必須考量兒童本身的認知結構及心理成熟因素。

所以，從事國小程式設計教學的教師，大多使用簡易、有趣且圖像化的程式語言,像是 Logo 語言(林裕雲,2002;徐龍

政,1995;崔夢萍,1999;黃文聖,2001)、電腦樂高 (林智皓,2007;施能木,2007;蔡錦豐,2008)、Stagecast Creator(豐佳燕、陳明溥, 2008) 均有此特徵,尤其近年來剛推出的 Scratch(楊書銘,2008)軟體語言工具由於具有中文的相容性,更是可為考量應用。

### 三、為何是Scratch

Scratch 軟體是美國麻省理工學院所發展的一套免費的程式語言,可以用來創造互動式故事、動畫、遊戲、音樂和藝術,並且可以把你的創意發表在網路上與人共享。

它是設計來用於幫助八歲以上的兒童學習數學以及計算能力,增加創意思考、有系統的推理和合作學習(麻省理工Scratch 團隊,2009)。Scratch 是一個視覺化的直觀式積木組合式程式語言,它使用拖曳的方式將指令從程式指令區移進程式腳本區。這大大地減少了打字錯誤的機會,也不需要初學之時苦於記誦指令的拼法。

積木組合的編寫方式可幫助初學者在程式編寫時更容易注意到指令程序的問題,哪一項指令該在哪一項指令的前面,都可以依積木組合的方式而不弄反,如此就不會在學習階段時,因為不熟悉語法順序或格式問題而出錯,因而造成太多的失敗經驗而降低學習動機。

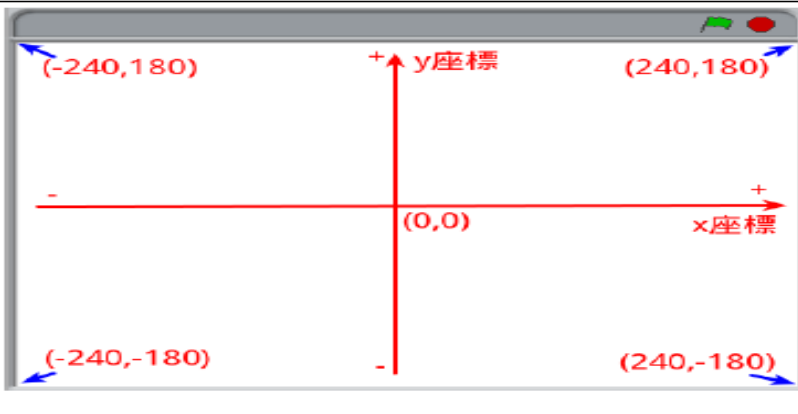
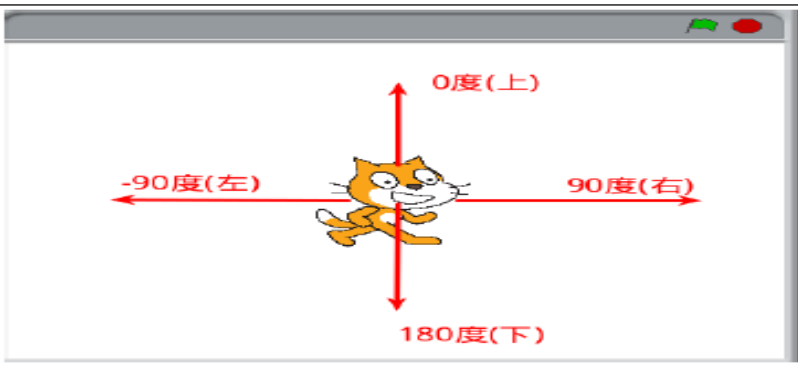
Scratch 2軟體的儲存副檔名為.sb2,儲存後的檔案即可被裝有可執行java的瀏覽器展示出來,官方網站亦有提供展示

平台讓全球學習者分享創意作品，提供評論。更重要的是這些作品都可以下載回去讓學習者研究，不需反組譯作品檔即可觀看原始碼，充份做到知識共享的精神。

## ★設計概念

Scratch的設計概念，就是想像你是一個「導演」，在一個「舞台」上命令各個「角色」去表演一場戲，而每一個角色可以有不同的「造型」呈現，舞台也可以有不同的「背景」呈現。

## ★Scratch的座標與方向系統

 <p>A diagram of the Scratch stage coordinate system. It shows a 2D Cartesian coordinate system with a central origin labeled (0,0). The horizontal axis is labeled 'x座標' and the vertical axis is labeled 'y座標'. The four corners of the stage are marked with coordinates: top-left is (-240,180), top-right is (240,180), bottom-left is (-240,-180), and bottom-right is (240,-180). The origin is also labeled with a '+' sign above and a '-' sign below.</p>	<p>一、座標</p> <ul style="list-style-type: none"><li>(1)舞台是利用「二維座標」來判別位置的</li><li>(2)以正中央為中心點 (0,0)</li><li>(3)左右為 X 座標，右邊為正，左邊為負</li><li>(4)上下為 Y 座標，上面為正，下面為負</li><li>(5)整個舞台可以呈現的範圍為 上下共 360(180+180) 左右共 480(240+240)</li></ul>
 <p>A diagram showing the Scratch character, a yellow cat, in the center of the stage. Four red arrows point outwards from the character, representing directions: up is labeled '0度(上)', down is '180度(下)', left is '-90度(左)', and right is '90度(右)'.</p>	<p>二、方向</p> <p>每個角色有面向的方向，要注意的是，角色的方向不是以圖片的臉朝哪邊為依據喔！</p> <ul style="list-style-type: none"><li>朝上為 0 度</li><li>朝下為 180 度</li><li>朝左為 -90 度</li><li>朝右為 90 度</li></ul>

Scratch Desktop

Scratch11專案

程式 造型 音效

動作 外觀 音效 事件 控制 偵測 運算 變數 函式庫

當被點擊

隱藏

定位到 x: -186 y: 130

重複 3 次

重複 5 次

建立 自己 的分身

等待 0.2 秒

x 改變 100

定位到 x: -186 y: y座標 - 60

當分身產生

顯示

迴轉方式設為 左-右

重複無限次

移動 10 點

等待 0.3 秒

碰到邊緣就反彈

如果 碰到 Ball ? 那麼

變數 分數 改變 10

分身刪除

當分身產生

建立 Beachball 的分身

本尊隱藏  
產生三排分身

分身運動方式...  
得分方式

分數 110

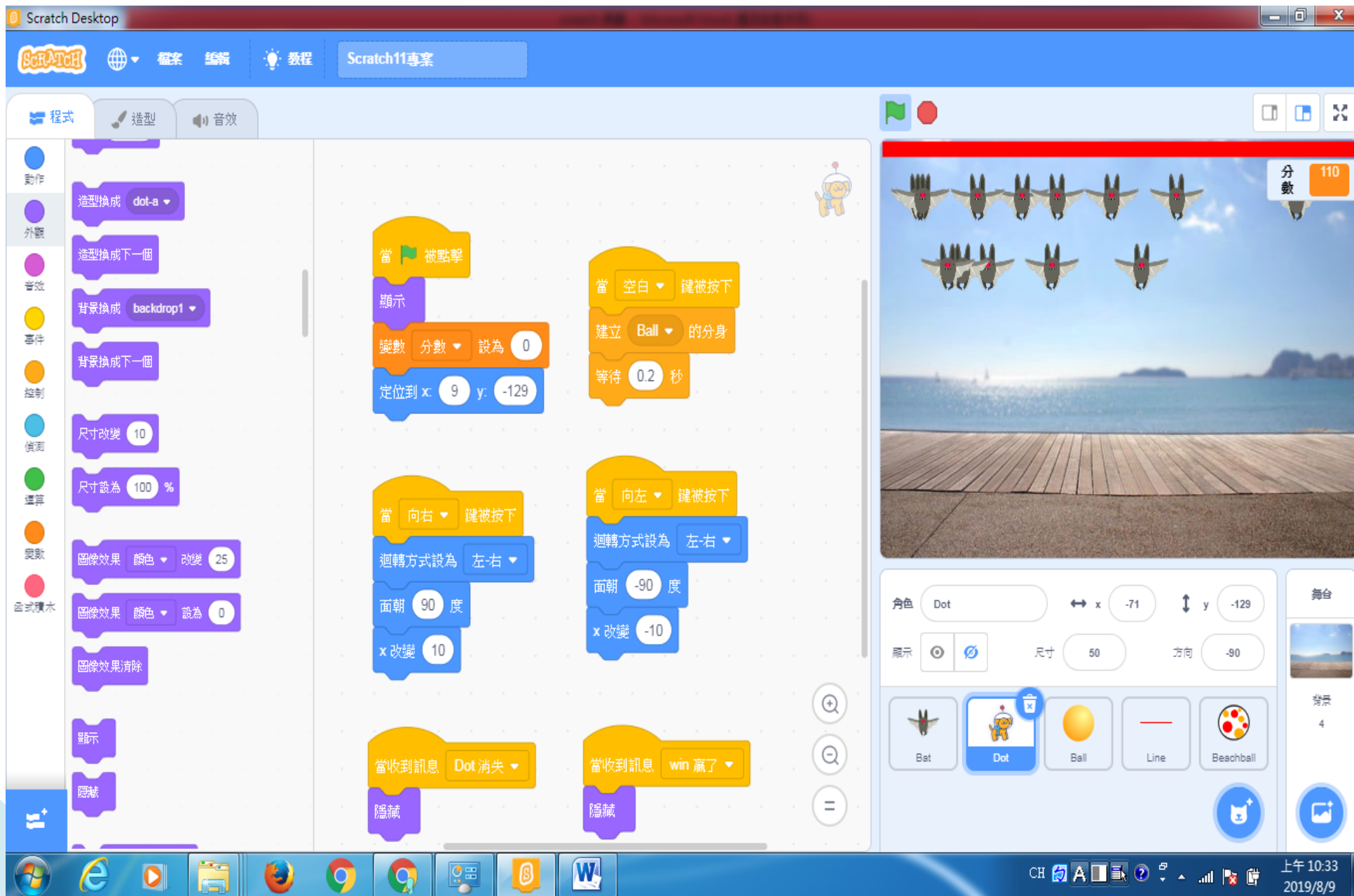
角色 Bat

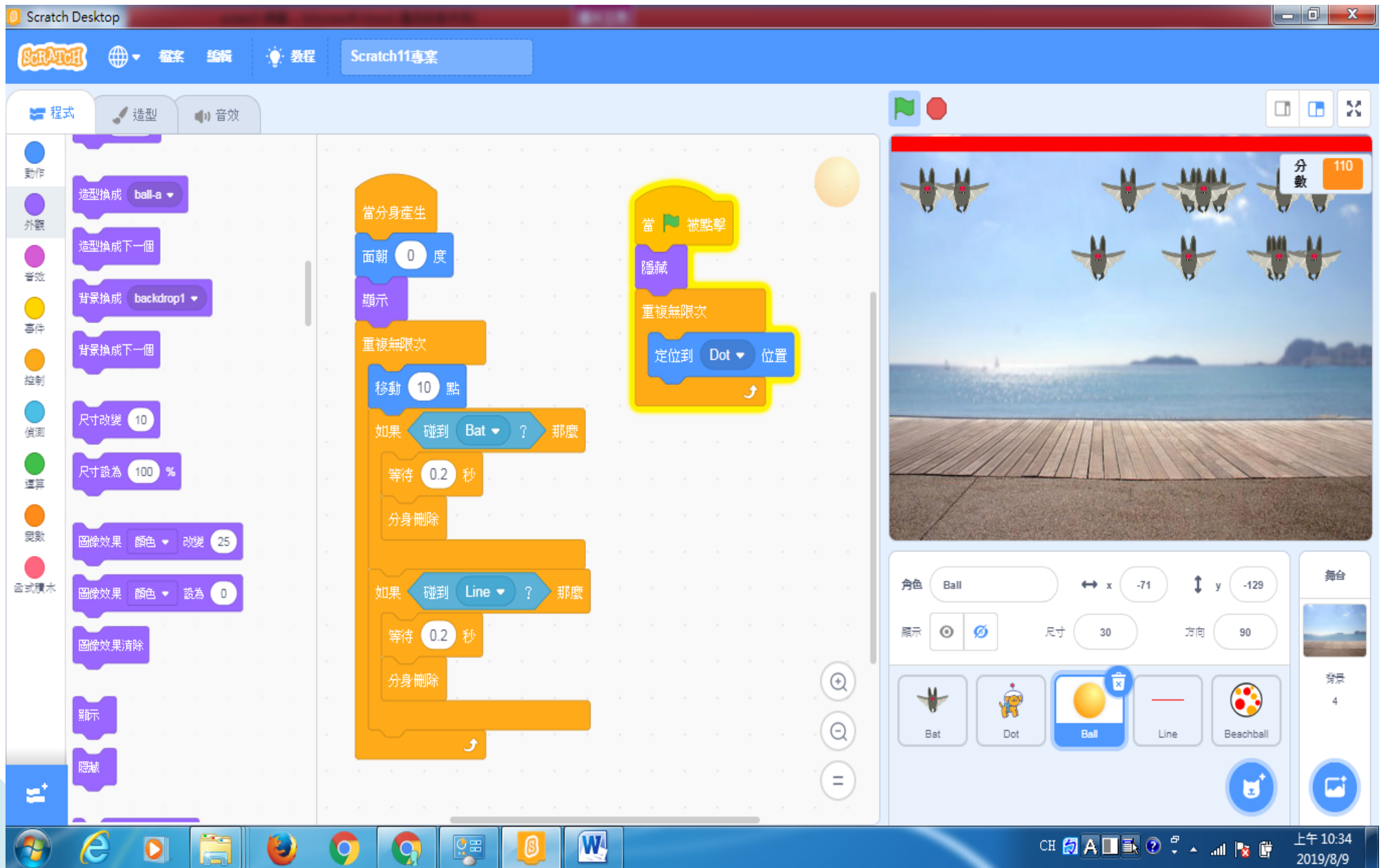
顯示 尺寸 35 方向 90

舞台

背景 4

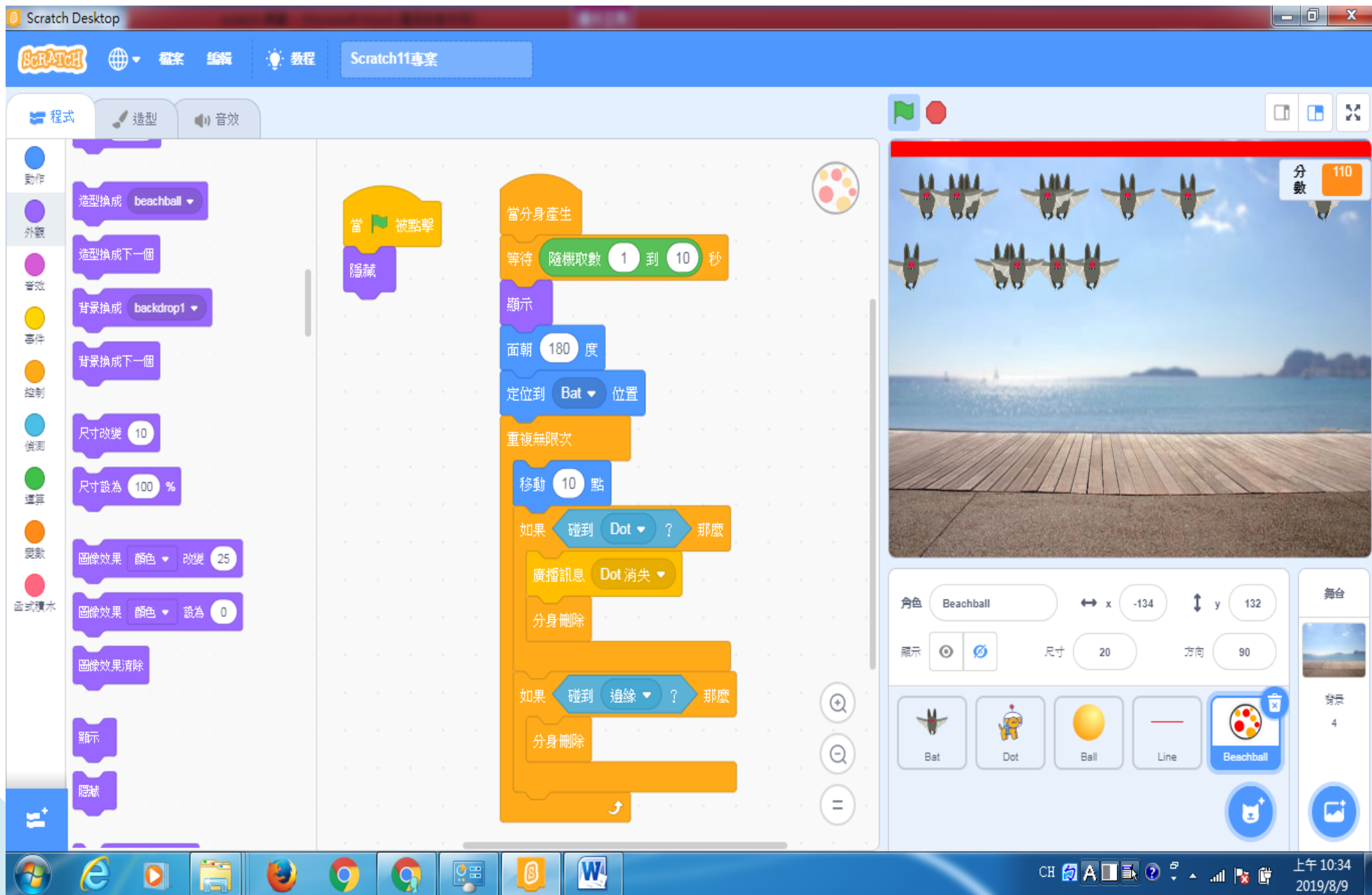
上午 10:31  
2019/8/9



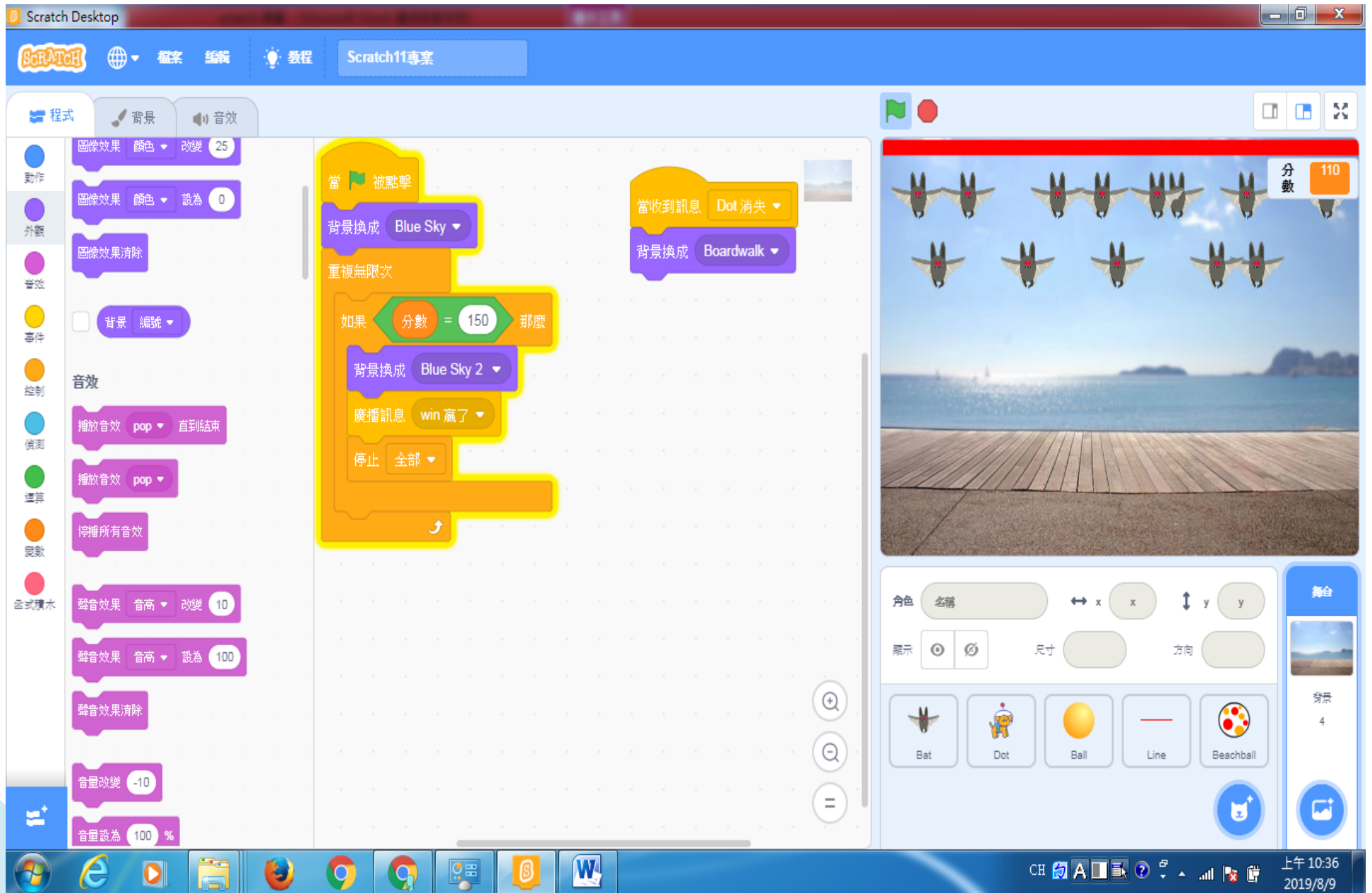


7









9

# 人人都該學程式設計的反思

瞭解工具，與實際地解決身邊問題

文/林信良 | 2013-06-14 發表

在程式設計領域，近來不斷見到「人人都該學寫程式」相關話題，多到不禁令人思考這是否為單一價值觀。更有不少人宣揚從小進程式設計教育的重要性，這彷彿看到了從小就學英文、從小就學理財、從小學圍棋……「XX就要從小開始」的論調。

如雨後春筍般地呼籲在一堆呼籲當中，最具代表性的，就是Code.org網站在2月發佈的影片《What most schools don't teach》。

影片以Apple創辦人Steve Jobs引言開頭，並邀請了Microsoft、Twitter、Facebook、Dropbox等各公司的創辦人Bill Gates、Jack Dorsey、Mark Zuckerberg、Drew Houston等各科技巨頭，談到他們第一次程式設計經驗、程式設計對他們來說是什麼？為什麼每個人都可學程式設計？至今這段影片仍在Code.org首頁播放中。

另一具代表性的呼籲，是MIT Media Lab計劃執行長Mitch Resnick的演講《Reading, Writing, and Programming》，演講重點之一在於透過程式設計，年輕人將能夠更加熟練地掌握新科技，熟練不單指與新科技互動，而是指可以運用科技表達自己、表達想法、輕鬆創作。

他並強調，程式設計不一定是撰寫程式碼，透過MIT Media Lab研發的Scratch，撰寫程式就像是組合積木，每個人都可以創造互動程式並在

網路上分享，即使是他那80歲的母親也不例外。他並談到愛沙尼亞通過法令，將程式設計納入小學一年級課程中，連英國也在討論是否該教導小學生撰寫程式。

在《How to Summon Your Child's Inner Coder》中，Raspberry Pi設計者Eben Upton談到他發明卡片電腦的動機，在於讓小孩不單是使用直覺到令人感到無聊的電腦平臺，而是可以重新把電腦當作嗜好，讓他們在玩樂之餘，同時學習如何分解問題並解決，任何涉及解決問題的工作都可以從中獲益。

為了引發小孩駕馭電腦的樂趣，除了價格人人可負擔的Raspberry Pi之外，還搭配了Raspbian作業系統，在Debian中預先安裝Scratch與Python，低年級孩童可透過Scratch的積木語法，獲得控制電腦的成就感，並習得基本邏輯，更大的孩子可透過Python，以文字來描述問題與解法。

而在《What most schools don't teach》中，真正吸引我的有兩段話，其中之一，是MarkZuckerberg談到：「我開始寫程式的理由很簡單，我想這麼做，僅此而已，我想為自己和妹妹做些有趣的事.....當我需要學習一些新東西的時候，我會嘗試在書裏或網路上尋找答案.....」另一段是位女工程師：「軟體有其人文的一面，它其實是藉由電腦科技來幫助人們...」這段影片與《Reading, Writing, and Programming》中都談到的部份是，電腦深入每個領域，每個人隨時隨地都會使用到與電腦相關的產品，它們是我們隨手可用的工具，我們應當運用它解決手邊的問題，而不是被它們駕馭。

Mitch Resnick在演講中談到「建立程式碼的同時也在學習如何設計，怎樣將某個靈光乍現，變成一個成熟的、可運作的專案...怎樣將複雜的概念分割成更小的單位.....」，《How to Summon Your Child's Inner Coder》中，作者也談過類似概念。

這些人倡導學習程式設計，並非要大家將來都成為程式設計師，而是以此來學會如何增加學習、表達、創作與對問題的分解能力。那麼，反過來思考，身為程式設計師的你，是否就真的擁有這些能力呢？你的想法是否被程式語言、技術給綑綁？你的生活是被網路、電腦、智慧型手機等工具給束縛，還是駕馭過它們來解決你身邊實際的問題？事實上我們知道，沒有學習、表達、創作與分解問題能力的程

式設計師很多！如果身為程式設計師都不具備這些能力，那不是程式設計師的你呢？看了這些影片、訪談或文章，「人人都該學程式設計」、

「應該讓我的小孩也學一下程式設計」就會是你的結論嗎？

如果是個有獨立思考能力的人，無論工具是什麼，在想要解決問題或創造些什麼時，自然就會去學習與駕馭這些工具，即使工具不是程式設計，也能得到Mitch Resnick談到的「學習到的事物將更為多樣」。如果你不是這樣的人，我完全不建議你學習程式設計，否則在擁有一知半解知識的情況下，只會危害你身邊真正的程式設計師。

### 從小學程式設計？

現今世界有不少的話題都與程式設計有關，擁有優秀程式設計能力就可以進Google、Facebook這類響亮公司，擁有吃不完的零食、健身設施與娛樂器材等福利，或者可能寫出憤怒鳥這類一夕致富的App。人人學程式設計的下一步，就讓人開始想到從小學程式設計，將來才能擁

有這份能力在世界上擁有競爭力。嗯？上一句話不覺得很眼熟嗎？把「程式設計」換成「英文」等學習題材，就是經常看到的補習班廣告術語，幸運地是，程式設計在臺灣並不算許多人心中的好職業，許多程式設計師也不想讓小孩將來走上相同的路，目前從小學程式設計，並沒有成為補習

班極力推捧的對象。最近網路上熱傳的《The Lost Interview》，是Steve Jobs於18年前談到的，「這個國家每個人都該學程式設計……因為程式設計教你如何思考……」，這也是《What most schools don't teach》開頭的引言，Steve Jobs並不是基於功利角度來談程式設計的學習，傳達的是程式設計會教你另一種思考方式，雖然他確實在當時就看好程式和Web，不過，有多少人因此讓小孩開始學程式設計呢？誰能預料十年、二十年後，引領世界話題的還是程式設計？

撇開功利不談，不少訴求從小學程式設計好處的文章，主要著眼於程式設計可培養邏輯，以及前面談到的分解問題、學習、表達、創作等能力，就算不投入程式設計產業，這些能力在其他領域也用得上。嗯？這句話照樣好眼熟，把「程式設計」換成「圍棋」、「繪畫」、「音樂」等，好像也行得通。我曾思考過，如果當初填上的志願不是電機而是資工，還會不會喜歡程式設計？我的答案是未必！

大學的時候，玩電腦是我逃避課業壓力時的嗜好；Eben Upton希望的也是小孩可以把電腦當作嗜好。嗜好這種東西的樂趣，通常不在最後成果，而是來自探索時，不斷主動獲取知識、技能，不斷思考如何解決遇到之問題的過程，能力不過是獲得這些樂趣的過程中自然形成的產物，並不是學了程式設計、圍棋、繪畫、音樂就會擁有這些能力。這樣想的話，程式設計不過是另一門才藝課程。

## 瞭解工具與實際地解決身邊問題

回到人人該學程式設計的呼籲，既然程式設計有這麼多好處，如果你是程式設計師，試問自己，除了為公司、老闆或客戶的問題寫程式之外，是否為自己的問題寫過程式？你的程式設計技能是否真正為家人解決過問題？你是否曾將呼籲當中的那些能力展現在其他領域中？

就如呼籲中宣稱的，那些能力在任何領域中都受用？如果你不是程式設計師，記得程式設計就只是個工具罷了，把前面「程式設計」換成你其他工具技能，問自己相同的問題，別急著馬上學程式設計。

至於從小學程式設計，我不是教育專家，只知道小孩會模仿父母，如果你從沒在小孩面前解決過任何問題，沒創作過任何東西，沒有展現過任何探索過程的樂趣，那麼還是多留點時間，讓小孩可以擁有自己的嗜好。幸運的話，也許小孩的嗜好剛好是程式設計，或許你面對那些從小學程式設計的論調，就不會有那麼多煩惱了。

## 專欄作家簡介

林信良

因在網路上經營「良葛格學習筆記」( [openhome.cc](http://openhome.cc) ) 而聞名，曾任昇陽教育訓練中心技術顧問、甲骨文教育訓練中心授權講師，目前為自由工作者，專長為技術寫作、翻譯與教育訓練。喜好研究程式語言、框架、社群，從中學習設計、典範及文化。閒暇之餘記錄所學，技術文件涵蓋C/C++、Java、Ruby/Rails、Python、JavaScript、Haskell等多個領域。



文章取自：[http://www.ithome.com.tw/node/80951\\_\\_](http://www.ithome.com.tw/node/80951__)

