# **Lesson 1: Innovations in Computing**

第1課：創新電腦

**Research | Unplugged**

研究｜不插電

## **Overview**

課程概述

To kick off the final unit of this course, students will do some research into interesting innovations in computing. This lesson will expose students to wider variety of computing form factors (what a computer looks like) and fields that are impacted by computing. Later in this unit students will look back on the devices they encountered in this lesson as they develop their own physical computing devices.

在這個課程的最後一個單元, 學生會研究一些有趣的創新電腦。這個課程會讓學生探索不同電腦的型態(電腦長什麼樣子), 電腦運用在各個領域的應用。在這個單元的最後, 學生會用這堂課所學到的裝置設計一台自己的電腦。

**Purpose**

課程目的

This lesson will lay the groundwork for students' understanding of how their Circuit Board could be used to model an innovative computing device. The goal is to get them thinking about how computers can be embedded into just about anything, and to start considering the potential impacts of such applications.

這堂課將會是學生了解電路板在創新電腦設備中的用途, 目的是讓他們了解電腦可以鑲嵌運用在幾乎所有的設備裡, 開始思考各種應用的可能性。

**Agenda**

* [Warm Up (10 min)](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing0)
* [Get Inspired](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing1)
* [Activity (45 min)](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing2)
* [Innovation Research](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing3)
* [Wrap Up (2 min)](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing4)
* [Journaling](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing5)
* [Extensions](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing6)
* [Innovation Posters](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing7)
* [Innovation Websites](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing8)

## **課程流程**

* 暖身 [(10 min)](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing0)
* 發想
* 活動 [(45 分鐘)](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing2)
* 創新研究
* 總結 [(2 分鐘)](https://curriculum.code.org/csd-18/unit6/1/#innovations-in-computing4)
* 紀錄日誌
* 延伸討論
* 創意網站

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/1/puzzle/1/)**

逛逛程式碼工作室

## **Objectives**

## **課程目標**

### **Students will be able to:**

* Identify computing innovations within a given field
* For a given device, articulate the likely inputs and outputs
* Suggest improvements to help a device better solve a specific problem

### **學生將具備以下能力**

* 辨認在特定領域的電腦應用
* 說出指定設備可能的輸入輸出
* 對一個設備要解決的問題提出更好的解決方案

## **Preparation**

* Review the resource pages linked in Code Studio
* Cue up [The Internet of Things - Video](https://youtu.be/xQGsubJNbQw) or [Computer Science is Changing Everything - Video](https://youtu.be/1x54GqfL3UY)
* Print out a copy of [Computing Innovations - Activity Guide](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/edit) for each student

## **課程準備**

* 閱讀程式碼工作室裡的參考網頁連結
* 準備以下影片的播放連結 [物聯網影片](https://youtu.be/xQGsubJNbQw) 、 [電腦科學改變一切影片](https://youtu.be/1x54GqfL3UY)
* 列印每位學生一份 [Computing Innovations - Activity Guide](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/edit)

## **Links**

**Heads Up!** Please make a copy of any documents you plan to share with students.

## **連結**

**提醒!** 請列印所有要分享給學生的文件

**For the Students**

* [Computing Innovations](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/edit) - Activity Guide Make a Copy
  + [PDF](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/copy)
* [The Internet of Things](https://youtu.be/xQGsubJNbQw) - Video
* [Computer Science is Changing Everything](https://youtu.be/1x54GqfL3UY) - Video ([download](https://videos.code.org/2015/csp/cs_is_changing_everything.mp4))

**給學生**

* 創新電腦 - 請列印活動指南
  + [PDF](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/copy)
* [物聯網影片](https://youtu.be/xQGsubJNbQw)
* [電腦科學改變一切影片](https://youtu.be/1x54GqfL3UY) (下載)

## **Vocabulary**

* **Innovation** - A new or improved idea, device, product, etc, or the development thereo

## **詞彙**

**創新**- 一個新的想法, 新的設備, 新的產品, 或是改善, 發展出想法, 設備, 產品

## **Support**

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

### [**Report a Bug**](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%201%20curriculum.code.org/csd-18/unit6/1/)

## **課外資源**

### **[討論區](https://forum.code.org/c/csd6/)**

### [**問題回報**](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%202%20curriculum.code.org/csd-18/unit6/2/)

# **Teaching Guide**

# **教學指南**

## **Warm Up (10 min)**

Teaching Tip

This unit requires the Adafruit Circuit Playground. You can read more about this microcontroller board at<https://code.org/circuitplayground>

## [**暖身活動(5分鐘)**](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code0)

教學技巧

這個單元需要用到開源電路板玩具Adafruit Circuit Playground, 你可以從以下連結學到更多官 於這個微控制器電路板<https://code.org/circuitplayground>

### **Get Inspired**

**Video:** Watch either [The Internet of Things - Video](https://youtu.be/xQGsubJNbQw) or [Computer Science is Changing Everything - Video](https://youtu.be/1x54GqfL3UY) as a class.

**發想**

**影片:** 觀看 [物聯網影片](https://youtu.be/xQGsubJNbQw) 、 [電腦科學改變一切影片](https://youtu.be/1x54GqfL3UY)

## **Activity (45 min)**

**活動 (45 分鐘)**

### **Innovation Research**

**創新研究**

**Group:** Place students in groups of 3-4

**Distribute:** Hand out copies of [Computing Innovations - Activity Guide](https://docs.google.com/document/d/1-fvLAoZcGopplA-0bwCFrCRoPFYPCPgSeb2ablNteCg/edit)

Discussion Goal

This should be a quick discussion. The primary goal is for the class to come to a common understanding of what it means for something to be innovative. That it's not enough for a product to look sleeker than the last version, but that innovation means to really do something better than it's been done before, or to do things that have *never* been done before.

**Discuss:** What is an innovation? What does it mean for something to be innovative?

**分組:** 3-4個學生一組

**講義:** 分發創新電腦 活動指南

討論目標

這應該是一個很快速的討論, 主要目標是要讓班上同學理解什麼是創新, 創新不只是讓產品看起來比舊版本更有時尚感, 創新是比過去做得更好, 或是之前*從未*做過的事情

**討論:** 什麼是創新? 當我們說創新的時候是什麼意思?

#### **Computing Innovations**

During this activity student groups will research the recent technological innovations related to a chosen topic. Once they have identified a few interesting innovations, they will choose one to analyze in greater depths and report back to the class about.

#### **電腦創新**

在這個活動裡, 學生小組會選擇一個討論話題, 研究一個新的科技產品, 等他們選定一個有趣的科技產品, 要深入研究這項產品並在班上報告分享。

#### **Innovation Research**

創新研究

* **Introduce the topics**: Make sure that students understand the scope of each of the potential topics.
  + Wearable Technology (eg. clothing, jewelry, or accessories with built-in computers)
  + Health and Safety (eg. devices that treat disease, track your health, or protect users from danger)
  + Agriculture (eg. technology to improve the effectiveness, sustainability, or efficiency of farming)
  + Manufacturing (eg. advancements in rapid prototyping, industrial robotics, and the production of goods)
  + Art and Design (eg. interactive art or public installations)
  + Smart Home (eg. devices that allow you to interact with your thermostat, locks, or lights using computers)
* **介紹主題**: 確保學生了解相關主題的範疇.
  + 穿戴式裝置 (例. 有電子產品的衣服, 珠寶, 裝飾品等)
  + 健康與安全產品 (例. 對抗疾病的產品, 健康追蹤, 保護使用者遠離危險)
  + 農業產品 (例. 提升產能的產品, 永續經營, 提升效率的產品)
  + 生產製造 (例. 快速成型製造的技術, 工業用機器人, 生產產品)
  + 藝術與設計 (例. 互動式藝術 或是 公共裝置)
  + 智慧家庭 (例. 可以與家裡的電子裝置互動的設備, 比如說控制冷暖氣, 門鎖, 燈光控制等)
* **Explain the research task**: The goal of this research is twofold:
  + First, develop a deeper understanding of your chosen topic. How is computer technology changing this field, what are some of the problems that people are trying to solve with technology?
  + Second, identify a handful of innovative devices within this topic. Students should focus on finding *hardware* devices that demonstrate unique or novel *form factors*. That is to say, computers that don't *look* like computers.
* **解釋研究工作**: 這個研究的目標有兩項:
  + 第一, 深入的研究所選的目標, 電腦怎麼樣改變這個領域的行為, 人們試著用電腦解決什麼問題?
  + 第二, 選定一個研究目標內可取得的創新設備. 學生應該探索這個設備的硬體設計, 展示分享不同型態的電腦, 電腦不一定是長得像電腦的形狀.
* **Send to Code Studio for resource links**: On Code Studio we have compiled more detailed descriptions of the topics as well as couple of recommended sites to learn more about each topic. Use this as a jumping off point for student research.
* **到程式碼工作室得到參考連結:** 在程式碼工作室, 我們有編輯相關主題的詳細描述, 以及一些推薦網站, 學生可以使用這些參考連結做進一步的研究.

### **Code Studio levels**

程式碼工作室方面

* Lesson Overview
* [Student Overview](https://curriculum.code.org/csd-18/unit6/1/#level-expando-1-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/1/puzzle/1)

* Innovation Research
* [Teacher Overview](https://curriculum.code.org/csd-18/unit6/1/#level-expando-1-2)
* [Student Overview](https://curriculum.code.org/csd-18/unit6/1/#level-expando-1-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/1/puzzle/2)

* 課程總覽
* 學生總覽

在程式碼工作室觀看

* 創新研究
* 教師總覽
* 學生總覽

在程式碼工作室觀看

以下有一些推薦網站, 可以讓學生開始做研究, 雖然這些網站的內容通常是適合學生閱讀, 但網站資料更新快速, **強烈建議教師在提供給學生參考之前, 檢查所有的推薦網站是否有不適合的**內容.

### **Wearable Technology**

* [Warable.com](https://www.wareable.com/)

### **穿戴裝置**

* [Warable.com](https://www.wareable.com/)

### **Health and Safety**

* [Modern Healthcare](http://www.modernhealthcare.com/section/medical-devices)
* [Medstartr](http://www.medstartr.com/)

### **健康與安全**

* 現代健康照護
* [Medstartr](http://www.medstartr.com/)

### **Agriculture**

* [National Institute of Food and Agriculture](https://nifa.usda.gov/topic/agriculture-technology)
* [Farm Industry News](http://www.farmindustrynews.com/technology)

### **農業**

* 國家食品與農業機構
* 農業產業新聞

### **Manufacturing**

* [Industry Week](http://www.industryweek.com/technology)
* [3D Printing](https://3dprinting.com/)

### **生產製造**

* 產業周報

### [3D Printing](https://3dprinting.com/)**Art and Design**

* [ArtFab](http://artfab.art.cmu.edu/)
* [Instructables](http://www.instructables.com/tag/type-id/category-technology/)

### [3D](https://3dprinting.com/)列印**藝術與設計**

* 藝術實驗室
* [Instructables](http://www.instructables.com/tag/type-id/category-technology/)

### **Smart Home**

* [CNet](https://www.cnet.com/smart-home/)
* [IoT Evolution](http://www.iotevolutionworld.com/smart-home/)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/1/puzzle/2)

### **智慧家庭**

* 科技行者
* 物聯網革命

在程式碼工作室觀看

# **Innovation Research**

Choose one of the following topics to look into. You'll want to find information about some of the most recent innovative computing devices related to your category.

# **創新研究**

## 從以下幾個主題選定一個進行討論. 你在你的主題範疇內針對最新的創新電腦設備找一些補充資料

## **Wearable Technology**

Wearable technology includes any form of computing that is designed to be worn by the user, whether it's embedded into the clothing or worn as an accessory. Many wearables, such as smart pedometers, overlap somewhat with health and safety as the device is both wearable *and* intended to help improve your personal health.

## **穿戴裝置**

## 穿戴裝置包含各種不同形式的電腦, 是設計用來穿戴在使用者身上, 不管是鑲嵌在衣服內, 或是配戴的裝飾品, 很多可穿戴裝置比如說計步器, 是用來做健康與安全裝置, 改善個人健康

## **Health and Safety**

The broad range topic includes both consumer technology (things that you might use to improve your health or safety) and professional technology (gear that a doctor or other professional might use).

## **健康與安全**

這部分廣泛的泛指所有與健康、安全相關的消費性產品, 以及醫生或是醫療產業使用的專業產品.

## **Agriculture**

Agriculture might not be an obvious topic, but technological innovations have had a huge impact on how we approach farming. This includes incorporating computers into existing farm equipment, using sensors and robotics to monitor crops, and analyzing data to improve efficiency and sustainability.

## **農業**

農業不是一個很熱門的話題, 但科技的應用已經大幅的影響我們處理農作物的方式, 包含將電腦設備架設在農場內, 使用各種感應監測設備, 機器人, 分析各種數據來提升效率與永續性.

## **Manufacturing**

Advancements in rapid prototyping, industrial robotics, and automation have had a major impact on how goods are produced. Whether it's technology to enable affordable small-scale manufacturing (like 3D printers and laser cutters) or the integration of computers into factory work, there are many interesting areas to research here.

## **生產**

## 快速成型製造的技術, 工業用機器人, 自動化生產對於產品生產製造有大幅的影響, 科技應用在小規模的生產(比如3D列印跟雷射切割), 也被整合應用到工廠大規模生產, 有很多有趣的研究在這個領域

## **Art and Design**

The intersection of artistic expression and computing can yield impressive and unexpected results. While you are less likely to find commercially available products related to art and design, you will find a wide variety of homemade creations that use all manners of computing devices in creative and expressive ways.

## **藝術與設計**

藝術與電腦的互動結果, 常常產生讓人感動並令人出乎意料的結果, 我們現在很少看到大規模的商業產品, 但個人或家庭的藝術與科技整合產品, 常常是非常具創造力, 並讓人引象深刻

## **Smart Home**

The dream of a house that understands and even anticipates your needs is closer than ever. Whether controlled explicitly through voice commands, or by using sensors to monitor and respond to changing conditions, smart home devices take care of any number of mundane or repetitive tasks around the house.

## **智慧家庭**

智慧家庭的夢想已經接近實現, 可以用語音控住, 有感應器去監測並回覆環境的改變, 智慧家庭設備可以完成家裡各種平凡又重複的工作

# **Research Considerations**

As you research devices on your topic, keep the following considerations in mind:

* Focus on hardware based solutions over software programs
* Focus on recent innovations over old news
* Follow what interests you

In addition to the recommended sites listed for each topic, consider looking towards the various crowd funding websites (such as Kickstarter and Indiegogo) to see the products that haven't quite made it off the ground yet.

# **研究的注意事項**

當你在研究主題的時候, 請注意以下事項:

* 最好是挑選硬體相關的產品而不是軟體解決方案
* 最好是挑選最新的科技而不是舊的
* 跟著你感興趣的主題

除了一些推薦的網站之外, 可以考慮參考一些群眾募資的平台(比如Kickstarter 何Indiegogo), 可以看到一些很新穎未量產的產品.

#### **An Innovative Solution**

Once groups have selected a specific innovative device, they can complete the second page of this activity guide, which asks them to do the following:

* **What Problem Does It Solve?:** For some topics this may be more clear (for example healthcare devices typically have a very specific goal), but for others students may need to think more broadly about what they consider problems. More frivolous wearables or digital art installations may be more concerned with personal expression than solving a specific problem.
* **What Is Innovative About It?:** Encourage students to refer back to the earlier discussion about innovation for this. It can also be useful to compare against other devices found during the research phase.
* **How Do You Interact With It?:** Ask students to think back to the *Input, Output, Storage, Processing* model that was introduced way back in Unit 1. The goal here is to identify specifically the Inputs and Outputs provided by a given device - the more specific these are, the more easily students will be able to connect elements of the Circuit Playground to these devices.
* **How Could You Improve It?:** Feel free to make this as realistic or aspirational as you like. The goal here is just to get students thinking about how they might develop an innovation of their own by using an existing product as a jumping off point.

#### **一個創新的解決方案**

小組決定想要研究的設備之後, 他們要完成第二頁的活動指南, 要求他們完成以下:

* **想要解決什麼問題?:** 在某些主題, 問題會比較明確(比如說健康裝置通常有很明確的目標), 但也有些設備需要更廣泛的思考解決的問題, 也有些裝飾性的穿戴裝置或數位藝術品是了彰顯個人特色而不是解決特定問題
* **這個創新是做什麼?:** 鼓勵學生參照回去前面關於創新的討論, 關於創新的討論可以應用在接下來的創新設備研究裡, 比較不同設備的創新.
* **你要如何與這個設備互動?:** 請學生從 輸入, 輸出, 儲存, 運算等模組, 參照第一單元的學習, 目標是讓學生清楚的辨別特定設備的輸入輸出, 學生越清楚理解這個觀念, 之後越容易將這樣的概念運用在設計Circuit Playground的模組
* **你想要如何改進這個裝置?:** 盡可能的發揮靈感, 付諸實現, 目標是讓學生從既有產品當作起始點, 思考如何發展自己的創新產品

**Share:** Give groups a minute each to share their findings. See the extension activities for alternate ways to share.

**分享:** 給小組幾分鐘的時間分享報告他們的研究結果, 可以參照延伸活動的分享方式.

## **Wrap Up (2 min)**

### **Journaling**

**Journal:** What was the most surprising, cool, or impressive thing that you found in your research? If you could develop an innovation of your own, what would it be?

## **總結 (2 分鐘)**

### **紀錄日誌**

## **日誌:** 在你的研究裡最酷, 最令人訝異, 最印象深刻的發現是什麼, 如果你想要設計自己的創意裝置, 你想做什麼呢?!

## **Extensions**

### **延伸活動**

### **Innovation Posters**

Ask groups to create a poster that shares the innovate device that they researched. Posters should include:

* Who invented the device
* What problem they were trying to solve
* Why it is unique or innovative
* How users interact with it (specifically, what Inputs does it take and how does it provide Output)

Keep these posters up throughout the unit for students to refer back to as they start to develop physical computing devices of their own.

### **創新海報**

要求各組做出一張海報分享報告他們的研究結果, 海報應該包含以下內容:

* 誰發明這個設備
* 這個設備試著解決什麼問題
* 為什麼這個創新是獨一無二的
* 使用者怎麼跟這個設備互動? (怎麼輸入, 如何提供輸出)

### 將這個海報保留在整個課程, 學生可以在後面開發自己的硬體設備的時候可以隨時回來參照.

### **Innovation Websites**

Instead of (or in addition to) you posters, have students develop websites in Web Lab that include the same required content. Make sure that students include links to their source websites.

### **創新網頁**

學生在海報之外(或兩者都做)可以創建一個網頁在Web Lab, 包含一些報告內容, 請確保學生列出參考資料出處連結.

## **Standards Alignment**

#### **[View full course alignment](https://curriculum.code.org/csd-18/standards/)**

#### **CSTA K-12 Computer Science Standards**

**IC** - Impacts of Computing

* **2-IC-20** - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.

## **標準協定**

#### **閱讀完整課程的協定**

#### **CSTA K-12 電腦科學標準**

**IC** - I電腦的影響

* **2-IC-20** - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.

# **Lesson 2: Designing Screens with Code**

# **使用程式碼設計畫面**

#### App Lab

## **Overview 課程概述**

In Unit 4 students learned a very simple approach to app development in App Lab that required a separate screen for most interactions. To expand the kinds of apps that students can make, and to encourage them to think in new ways about how users interact with apps, we introduce the [setProperty()](https://docs.code.org/applab/setProperty/) block. This command can be used to set the content and properties of various UI elements, allowing students to write programs that update information on a single screen, instead of manually creating duplicate screens. In this lesson students build up simple apps that only require a single screen, the content of which is changed using [setProperty()](https://docs.code.org/applab/setProperty/).

在第4單元中，學生們在App Lab中學習了一種非常簡單且大部分的互動在單獨畫面的應用程序開發方法。為了拓展學生們可以製做的應用程式類別，並鼓勵他們用新的想法思考使用者如何與應用程式互動，我們介紹區塊-[setProperty()](https://docs.code.org/applab/setProperty/) 。這項指令可以用來設定有多樣元素使用者介面的內容和屬性，並允許學生在單一畫面撰寫程式來更新資訊，而不是手動地創建重複的畫面。在本課程中，學生可以構建簡單且只需一個畫面的應用程式，其內容可以使用 [setProperty()](https://docs.code.org/applab/setProperty/)更改。

## **Purpose 課程目的**

This lesson allows students some time to get back into programming with App Lab before introducing the Circuit Playground, but also introduces the useful concept of a setter. In Unit 4, students primarily used [setScreen()](https://docs.code.org/applab/setScreen/) to make their apps respond to user interaction. While this is a simple and useful technique, it can lead to a lot of duplication of content across multiple screens. By using setters, and later getters, students can write apps that actually change the content on a single screen, by showing and hiding or changing the content or look of various elements.

Once students have learned about using getters and setters with UI elements, encourage them to think critically about when to use a separate screen in the design phase of an app. If screens are more alike than different, it might be more effective to just change the elements on screen in reaction to input instead of duplicating content across multiple screens. While students are only introduced to [setProperty()](https://docs.code.org/applab/setProperty/) right now, they will later learn the partner command [getProperty()](https://docs.code.org/applab/getProperty/).

本課程允許學生在介紹Circuit Playground之前有時間重新使用App Lab寫程式，同時還介紹了setters的有用概念。在第4單元中，學生主要使用 [setScreen()](https://docs.code.org/applab/setScreen/) 使他們的應用程式回應使用者的互動。雖然這是一種簡單而有用的技術，但它可能導致多個畫面上有大量重複的內容。透過使用setters和後來的getters，學生們可以通過顯示和隱藏或更改各種元素的內容或外觀來編寫內容僅在單一畫面上更改的應用程式。

一旦學生們學會使用帶有使用者介面元素的getters和setters，鼓勵他們思考該在應用程式設計的哪個階段使用單獨畫面。如果畫面相同處較多，那麼只更改畫面上的元素以回應輸入而不是跨多個畫面更改重複的內容可能更有效率。雖然現在只介紹 [setProperty()](https://docs.code.org/applab/setProperty/)給學生們，但他們稍後會學習夥伴指令- [getProperty()](https://docs.code.org/applab/getProperty/)。

## **Agenda 課程流程**

* **[Warm Up (5 min) 暖身活動(5分鐘)](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code0)**
* [UI Element Properties Refresher](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code1)
* [使用者介面元素屬性複習](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code1)
* **[Activity (45 to 60 minutes) 活動內容( 40-60分鐘)](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code2)**
* [Designing with setProperty()](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code3)
* 使用 [setProperty() 設計](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code3)
* **[Wrap Up (5 min) 總結(5分鐘)](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code4)**
* [Reflecting on Unit 4](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code5)
* 單元4反思

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/2/puzzle/1/)**

## **Objectives 課程目標**

### **Students will be able to:**

* Set the properties of UI elements using code
* Respond to user input using an event handler
* Write programs that change multiple elements on a single screen instead of changing screen

### **學生將具備以下能力:**

* 使用程式碼設定使用者介面元素的屬性
* 使用事件處理程序來回應使用者的輸入
* 編寫可以在單一畫面更改多樣元素而不是更改多個畫面的程式

## **Introduced Code介紹程式碼**

* [setProperty(id, property, value)](https://docs.code.org/applab/setProperty/)

## **Support 課外資源**

### **[Lesson Forum 討論區](https://forum.code.org/c/csd6/)**

### **[Report a Bug 問題回報](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%202%20curriculum.code.org/csd-18/unit6/2/)**

# **Teaching Guide 教學指南**

## **Warm Up (5 min)** [**暖身活動(5分鐘)**](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code0)

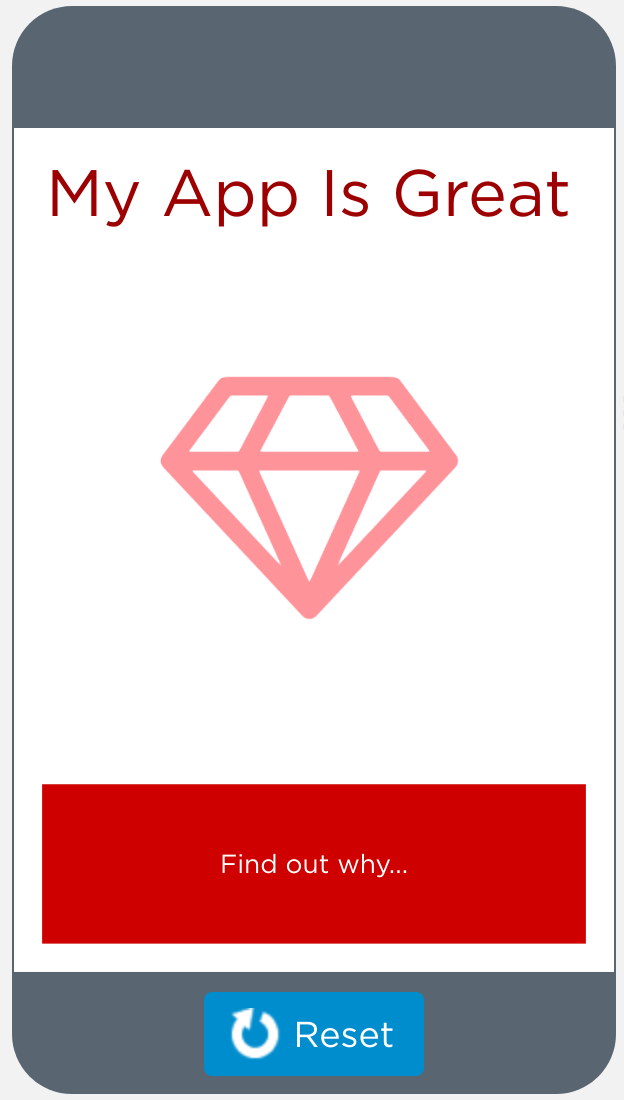
### UI Element Properties Refresher [使用者介面元素屬性複習](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code1)

Discussion Goal 討論目標

The goal of this disucssion is to prime students to think about the properties of various design elements, which they will learn to change with code later in this lesson. You might ask students to think back to a similar discussion we had in Unit 2 when trying to describe what a web page looks like, or to the properties of a sprite from Unit 3

這種討論的目的是讓學生思考各種設計元素的屬性，他們將在接下來的課程中學習如何用程式碼更改屬性。您可以要求學生回想在第2單元中嘗試描述網頁的外觀或第3單元中精靈的屬性的類似討論。

Display: Show students this example image



Discuss: How would you describe to somebody else how to recreate this screen? What specific details would they need to know about each design element?

討論： 請和其他人敘述你會如何重新建構這個畫面? 他們需要了解每個設計元素的哪些具體細節？

## **Activity (45 to 60 minutes)** [**活動內容( 40-60分鐘)**](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code2)

### Designing with setProperty() 使用 [setProperty() 設計](https://curriculum.code.org/csd-18/unit6/2/#designing-screens-with-code3)

Transition: Head to Code Studio

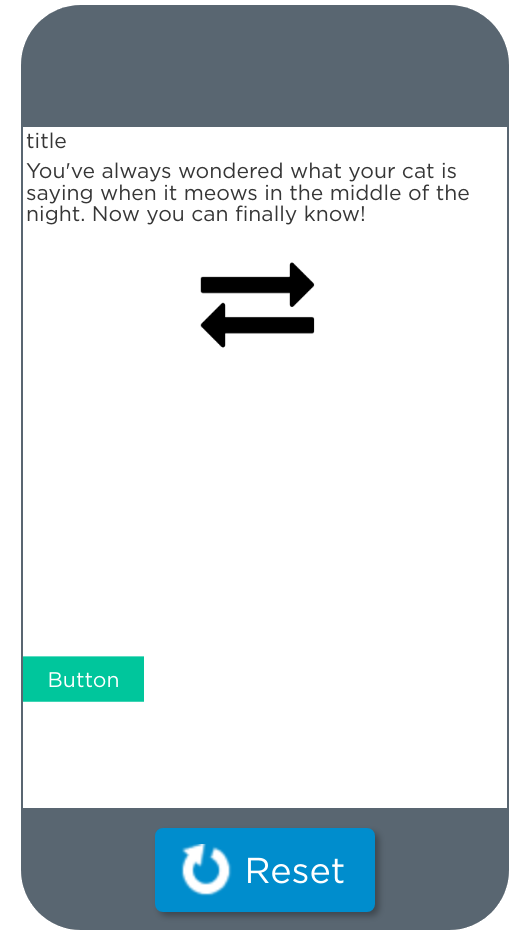
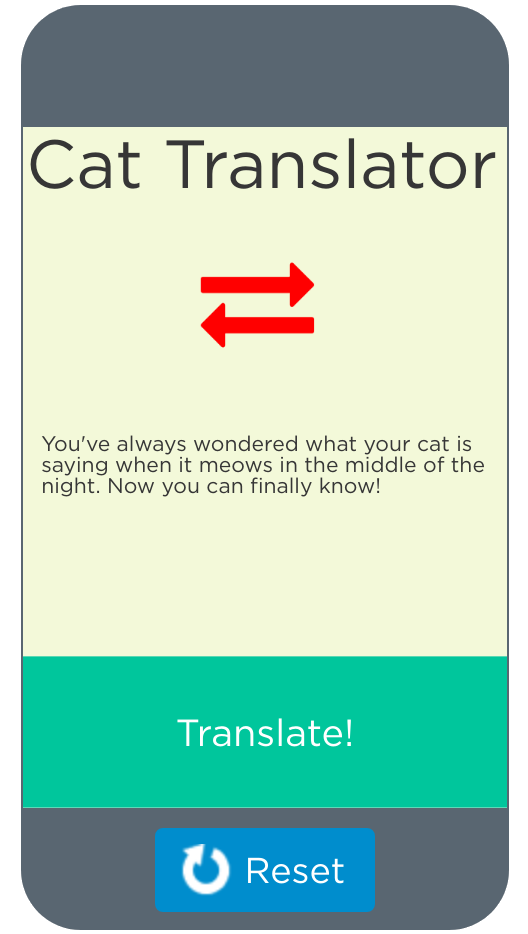
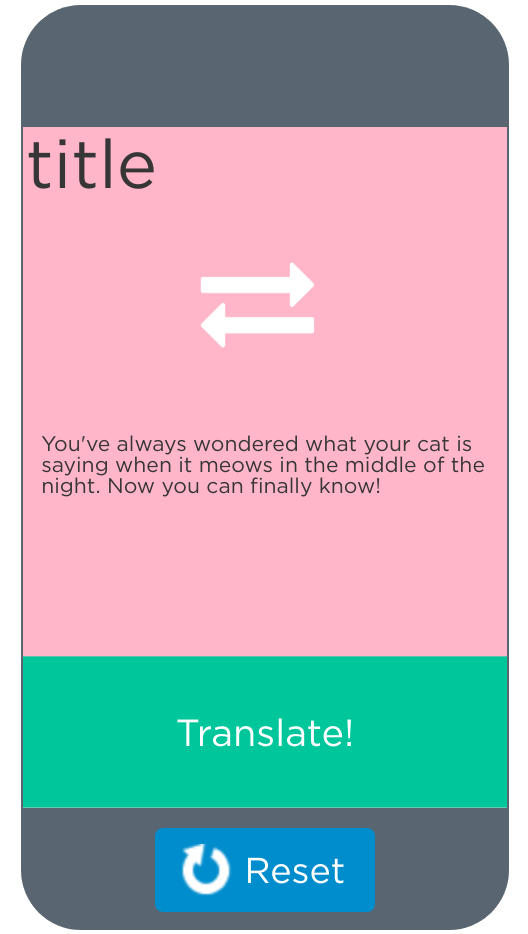
### Code Studio levels 關卡

* **Lesson Overview 課程概述**
* Student Overview 給學生的概述
* **Designing Screens with Code 使用程式碼設計畫面**
* Student Overview 給學生的概述
* **Setting Properties 屬性設定**

### Student Instructions學生操作指示

Read the code and predict what this app will look like when run.

閱讀程式碼並預測執行時應用程式的模樣



### Student Instructions 學生操作指示

# **Changing the Text 更改文字**

The [setProperty](https://docs.code.org/applab/setProperty/) block can be used to change lots of things about your design elements, including the text they contain. [setProperty](https://docs.code.org/applab/setProperty/)區塊可以用於更改各種你的設計元素，包括文字

# **Do This 動手做**

* Drag out a [setProperty](https://docs.code.org/applab/setProperty/) block 拖出[setProperty](https://docs.code.org/applab/setProperty/)區塊
* For the first parameter, select "name" (the element's id) 對於第一個參數，選擇“name”（元素的身分證）
* For the second parameter, select "text" (the property you want to set) 對於第二個參數，選擇“text”（要設置的屬性）
* For the third parameter, write your name in quotes (the value to set) 對於第三個參數，請在引號中寫下您的名字（要設置的值）

Note: We've removed the Design Mode tab for now so you can practice working with [setProperty](https://docs.code.org/applab/setProperty/), but it'll come back later once you're building apps of your own.

注意：我們現在已經刪除了“設計模式”鍵，因此您可以用[setProperty](https://docs.code.org/applab/setProperty/)練習，但是一旦您構建了自己的應用程序，它就會再次出現。

### Student Instructions學生操作指示

# **Changing Position 改變位置**

Every design element (except for the screen) has an "x" and a "y" property that controls its position.

每個設計元素（畫面除外）都有一個“x”和一個“y”屬性來控制它的位置。

# **Do This 動手做**

Using the [setProperty](https://docs.code.org/applab/setProperty/) block, move "red\_button" so that it matches the image to the right. (Click on the image to see the full size.)

Note: We've removed the Design Mode tab for now so you can practice working with setProperty, but it'll come back later once you're building apps of your own.

使用 [setProperty](https://docs.code.org/applab/setProperty/) 區塊，移動“red\_button”，使其與右側的圖像匹配。 （點擊圖像查看完整尺寸。）  
注意：我們現在已經刪除了“設計模式”鍵，因此您可以用setProperty練習，但是一旦您構建了自己的應用程序，它就會再次出現。

* **Events and Properties 事件和屬性**

### Student Instructions學生操作指示

# **Changing With a Click 使用點擊改變**

It's not really useful to only set a property when your program is first run. After all, you could just do all of that in Design Mode. When you use an [onEvent](https://docs.code.org/applab/onEvent/) block with [setProperty](https://docs.code.org/applab/setProperty/), you're able to do things you couldn't do with Design Mode alone.

在程序首次運行時僅設置屬性並不是很有用。畢竟，你可以在設計模式中完成所有工作。當您使用帶有[setProperty](https://docs.code.org/applab/setProperty/)的[onEvent](https://docs.code.org/applab/onEvent/)區塊時，您可以執行在“設計模式”無法執行的操作。

# **Do This 動手做**

We've provided an empty [onEvent](https://docs.code.org/applab/onEvent/) block that's watching the "smiley" image. Inside the [onEvent](https://docs.code.org/applab/onEvent/) block, use two [setProperty](https://docs.code.org/applab/setProperty/) blocks to move the smiley somewhere else (it doesn't really matter where for now). Test your program by running it and clicking on the smiley.

我們已經提供了一個空的onEvent的區塊用來觀看”smiley”這張圖片，使用兩個setProperty區塊來移動圖片(圖片目前在哪邊並不重要)。藉由點擊圖片來測試你的程式

### Student Instructions 學生操作指示

# **Randomization** 隨機性

Moving on the first click is fine, but this smiley would be even cooler if it moved somewhere new every time we clicked. Fortunately, we can use the [randomNumber](https://docs.code.org/applab/randomNumber/) block to do just that.

藉由一次點擊來移動圖片是很好的，但若能讓每次點擊都能移動圖片會更酷。幸運的，我們可以使用randomNumber區塊來做。

# **Do This**動手做

Instead of setting the "x" and "y" properties to a specific number, use the [randomNumber](https://docs.code.org/applab/randomNumber/) block to pick a new random "x" and "y" each time the smiley is clicked.

Tip: The App Lab screen is 320 pixels wide by 450 pixels tall. What range of random numbers will ensure that the smiley is always visible on screen?

在每次點擊圖片後會使用randomNumber區塊來選取新的隨機的x與y，而非設定x與y為特定的數字。

提示:App Lab的大小設定為寬320、長450像素。該設定什麼樣的隨機數字來確保我們可以在螢幕上看見圖片?

### Student Instructions 學生操作指示

# **Hidden 隱藏**

Design elements have a property called "hidden" that controls whether the element should show on screen or not. We are going to use this property to create an alarm app. Notice that when the program starts one button is hidden, and one is not.

We have already set up an event handler for the "triggerAlarmButton" which will do two things when clicked:

設計元素中有個property叫hidden，用來控制元素使否顯示在螢幕上。我們將用這個property來製作一個警報app。要注意在這個程式中一個按鈕式hidden，另一個按鈕則否。

* Change the "text" property of "statusLabel" to "on"
* Set the "hidden" property of "stopAlarmButton" to false (meaning it should not be hidden)

改變text中的statusLabel至on

設定hidden中的stopAlarmButton至false

# **Do This 動手做**

* Run the program and click "Trigger Alarm" to see how it works
* Add code to the provided event handler that hides the "triggerAlarmButton"

啟動程式並點擊Trigger Alarm來看它是如何運作的

增加程式碼到提供的event handler(trggerAlarmButton隱藏在裡面)

Note: the button "stopAlarmButton" is already on the screen, but because its "hidden" property is marked true, it isn't shown when the program starts.

stopAlarmButton按鈕已經在螢幕上，但因為它的hidden屬性設定為true，故當程式開始執行時不會顯示

### Student Instructions

# **Showing/Hiding 顯示/隱藏**

This alarm app isn't super useful if you can't turn the alarm off once it's been triggered. Let's fix that.

這個警報app若啟動後無法停止警報，則會難以使用，我們來改善它。

# **Do This 動手做**

Add a second [onEvent](https://docs.code.org/applab/onEvent/) block to watch for the "stopAlarmButton" to be clicked. Inside that block:

* Change the status text to "OFF"
* Hide the "stopAlarmButton"

Hint: "hidden" is a boolean property. If we used false to make the button show up, what should you use to make it hidden again?

增加第二個onEvent區塊來等候stopAlarmButton被點擊，在區塊內：

更改text的狀態為off

隱藏stopAlarmButton

提示:hidden是一個布林(true or false)屬性，若我們用false來使按鈕出現，我們應該用什麼來使其再次隱藏?

* **Emotion Machine Example**
* Student Overview

# **The Emotion Machine 表情機器**

Check out this Emotion Machine app that you'll be building next. Try clicking on the buttons to see how it works.

確認這個我們將要製作的Emotion Machine app，試著點擊按鈕來看它是如何運作的

# **Do This 動手做**

Try out the app, and then with a neighbor discuss:

* What events handlers does this app use?
* Which elements are being changed with code?
* Which properties are being changed when you click a button?

試試這個app，然後我們討論以下幾點:

這支app使用什麼樣event handlers?

程式碼中什麼元素被改變了?

當按下按鈕時，什麼特性被改變了?

* **Building an App 製作一個應用程式**

### Student Instructions 學生操作指示

# **Changing Images 改變圖片**

The first thing you'll need your emotion machine to do is change to the appropriate image when a button is clicked. If you look at the starting code, you'll see that we already added [setProperty("face", "image", "icon://fa-smile-o")](https://docs.code.org/applab/setProperty/) to change to a smiley face when you click happy.

要做的第一件事就是當按下按鈕時，你的emotion machine必須改變成適當的圖片。若你看到程式碼的前面，你會看到我們已經增加了"setProperty(“face”, “image”, “icon://fa-smile-o”)”到其中，當按下happy時，圖片會變成笑臉。

# **Do This 動手做**

Create an event handler for "sad\_button" and make it change to a sad emoji. To do this you'll need to set the "image" property of the image with id "face".

Tip: when you click the last dropdown in the [setProperty](https://docs.code.org/applab/setProperty/) block click "Choose..." to go to your media library. From there you can select an icon from our icon library or upload an image of your own.

為”sad\_button”新增一個event handler並讓它變成哭臉。我們必須設定id"face"中的image特性來達成

Tip:當點擊setProperty區塊的最後一個下拉選單，點擊Choose來進入媒體庫。從這邊你可以從圖示庫中選擇圖示或是自行上傳圖示。

### Student Instructions 學生操作指示

# **Changing Color 改變顏色**

If you look back at the example for this app, you should notice that not only do the emojis change when you click a button, but their color changes as well.

當你再次看這個app的範例，你應該會注意到當你按下按鈕後，不只是表情改變，顏色也一同改變了

# **Do This 動手做**

In each of the event handlers, set the "icon-color" property of "face". Pick a warm color for happy and a cool color for sad.

在每個event handlers中，設定face中的”icon-color”特性。為高興選擇一個暖色，為悲傷選擇一個冷色。

### Student Instructions 學生操作指示

# **Setting Text 設定文字**

Your happy and sad buttons should be almost completely functional now. The only thing they still need to do is change the text at the bottom of the screen.

現在你的Happy及Sad按鈕應該可以使用了，最後他們還需要改變螢幕上按鈕中的文字。

# **Do This 動手做**

Make each of the event handlers change the text of "feeling" to the appropriate emotion (either "happy" or "sad")

使event handlers來改變”feeling”這個文字使其變成適當的emotion(happy或是sad)

### Student Instructions 學生操作指示

# **Add Another Emotion 增加另一個表情**

In the example app you saw earlier, you could set the emotion to happy, sad, or meh. Your app only has happy and sad, can you add a third emotion?

在稍早看見的範例app，你可以設定emotion為happy、sad或是meh。目前你的app只有happy或是sad，是否能再增加第三個emotion?

# **Do This 動手做**

Using all of the same techniques you used for the happy and sad buttons, create a third emotion button. You could stick with meh, or you could pick an emotion of your own.

When you're all done, feel free to add more emotions or features before submitting your final version.

使用所有跟設定sad或是happy同樣的技巧來新增第三個emotion按鈕。可以新增meh或是選擇任何你想要的emotion

Share: The final level in this lesson allows students to customize and submit an "Emotion Machine" app. If time, allow students to share their programs.

Share: 在這個課程的最後我們允許學生客製化並提交一個”Emotion Machine” app。若時間上允許，可以讓學生分享他們的程式

## **Wrap Up (5 min)**

### Reflecting on Unit 4

Discussion Goal

Goal: This discussion is intended to clarify for students why we are changing UI elements with code, and how it might actually allow them to solve problems that came up in Unit 4. From this point on students will need to make reasoned choices about when to use separate screens in a program and when to update elements with code.

目標: 這次的討論是要為學生釐清為何要更改程式碼來改變UI的元素，並且使學生們能解決在unit 4中出現的問題。學生會需要為一支程式中何時該使用分離螢幕以及在程式碼中何時該更新元素做出合理的選擇。

Prompt: Think back to the app you prototyped in Unit 4. Knowing what you know about using [setProperty()](https://docs.code.org/applab/setProperty/) to change UI elements, how might you change your app prototype?

Prompt: 回想在unit4中的製作的app原型，在知道使用setProperty()來改變UI元素後，有什麼樣的可能性來改變你的App原型

Discuss: Have the class share the kinds of things they came up with. See if the class can come up with some broad types of features that weren't possible, or were cumbersome, to do with screens alone.

Discuss:讓班級分享他們題中的各種事物，看班級上是否能提出一些不可能的特性或是使用單獨的螢幕

# **Lesson 3: The Circuit Playground**

#### **App Lab | Maker Toolkit App Lab | 創客工具組**

## **Overview 課程概述**

In this lesson students get their first opportunity to write programs that use the Circuit Playground. After first inspecting the board visually and hypothesizing possibly functionalities, students move online where they will learn to write applications that control an LED. By combining App Lab screens with the Circuit Playgrounds, students can gradually start to integrate elements of the board as an ouput device while relying on App Lab for user input.  
在本課程中，學生們第一次有機會使用Circuit Playground的編寫程序。 首先檢查實驗板的外觀和假設可能的功能之後，學生們會在線學習，以便學習編寫控制LED的應用程序。 藉由將App Lab屏幕與Circuit Playgrounds相結合，學生可以逐漸開始將實驗板的元素整合為輸出設備，同時依靠App Lab進行使用者輸入。

## **Purpose 課程目的**

As the first introduction to using the Circuit Playground, this lesson leaves time for students to get comfortable with getting the hardware plugged in. By leveraging students' existing knowledge of event handling in App Lab, we can quickly get an app up and running that shows the potential of physical computing with little more than a single red LED.  
作為使用Circuit Playground的第一次介紹，本課程讓學生有時間熟悉硬體外掛程式。通過利用學生現有的App Lab中的事件處理知識，我們可以快速獲得一個應用程序並運行顯示物理計算的潛力超越一個紅色LED。

## **Agenda 課程流程**

* [Warm Up (5 min)](https://curriculum.code.org/csd-1718/unit6/3/#the-circuit-playground0)
* [Board Inspection](https://curriculum.code.org/csd-1718/unit6/3/#the-circuit-playground1)
* [Activity (45)](https://curriculum.code.org/csd-1718/unit6/3/#the-circuit-playground2)
* [Connecting the Board](https://curriculum.code.org/csd-1718/unit6/3/#the-circuit-playground3)
* [Programming on Hardware](https://curriculum.code.org/csd-1718/unit6/3/#the-circuit-playground4)
* [Wrap Up (5 min)](https://curriculum.code.org/csd-1718/unit6/3/#the-circuit-playground5)
* [What's in a Board](https://curriculum.code.org/csd-1718/unit6/3/#the-circuit-playground6)
* 暖身活動 (5 min)
* 實驗板檢查
* 活動內容 (45)
* 連接實驗板
* 對硬體做程式編輯
* 總結 (5 min)
* 實驗板內有甚麼

## **Objectives 課程目標**

### **Students will be able to: 學生將具備以下能力:**

* Connect and troubleshoot external devices  
  連接和排除外部設備故障
* Turn on and off an LED with code  
  使用代碼打開和關閉LED
* Use code to control a physical device  
  使用代碼控制物理設備

## **Preparation 課前準備**

* Make sure that student computers have the drivers and software necessary to connect to the Circuit Playground ([details here](https://studio.code.org/maker/setup))  
  確保學生的電腦具有連接到Circuit Playground所需的驅動程序和軟體
* Prepare a board and USB cable for each pair of students  
  為每對學生準備一塊實驗板和USB線

## **Introduced Code 介紹程式碼**

* [le](https://docs.code.org/applab/led.on/)[d.on()](https://docs.code.org/applab/led.on/)
* [led.off()](https://docs.code.org/applab/led.off/)
* led.blink(interval);
* [led.pulse(interval);](https://docs.code.org/applab/led.pulse/)
* [led.toggle()](https://docs.code.org/applab/led.toggle/)

## **Support 課外資源**

### **[Lesson Forum 討論區](https://forum.code.org/c/csd6/)**

### **[Report a Bug 問題回報](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202017%20unit%206%20lesson%203%20curriculum.code.org/csd-1718/unit6/3/)**

# **Teaching Guide 教學指南**

## **Warm Up (5 min) 暖身活動**

### **Board Inspection 實驗板檢查**

**Distribute:** Pass out a board and USB cable to each pair of students. Let students know that they should not yet plug the boards in.  
分發：為每對學生分發一塊實驗板和USB線。 讓學生知道他們還不能插入電路板。

**Prompt:** Ask pairs to spend one minute looking over the board, focusing on the details. What do you think this board does (or could do) and why?  
提問：要求每組人花一分鐘時間查看實驗版，了解細節。 你認為這個實驗版做了什麼（或可以做什麼）以及為什麼？

**Share:** Have groups share back their thoughts to the whole group, keeping track of ideas on the board. Push students to support their ideas with evidence from reviewing the board, but don't worry about ensuring correctness at this point. As students go through this unit, they can refer back and refine this list.  
分享：讓小組將他們的點子分享給整個小組，持續觀察實驗版的構想。 通過實驗板顯示的構想，推動學生根據前一個步驟對實驗板子檢查的心得去思考 (說明) 如何支持他們的設計構想

，但在這一點上不用擔心正確性與否。 當學生通過這個單元時，他們可以參考並改進這個列表。

## **Activity (45) 活動內容**

### **Connecting the Board 連接實驗板**

**Transition:** Ask students to plug their boards in and head to [the Maker Toolkit setup page](https://studio.code.org/maker/setup) to confirm that the software has been correctly configured.  
接下來：要求學生插入實驗板並前往Maker Toolkit設置頁面來確認軟體已經被正確的設定。

### **Programming on Hardware 在硬體上寫程式**

**Transition:** Send students to Code Studio  
接下來：讓學生使用Code Studio

### **Code Studio levels Code Studio 關卡**

* Lesson Overview  
   課程概述
* [Student Overview  
   給學生的概述](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-1-sfmd)

[View on Code Studio   
觀看Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/1)

* Hardware and Software  
  硬體 and 軟體
* [Student Overview  
  給學生的概述](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/2)[觀看Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/1)

[Download Video  
下載Video](http://videos.code.org/cs-discoveries/concept/06_HardwareSoftware_sm.mp4)

* Circuit Playground: LEDs
* [Student Overview](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-3-sfmd)[給學生的概述](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/3)[觀看Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/1)

* Using the LED
* [4](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-4)
* [5](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-5)
* [6](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-6)
* [7](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-7)
* [8](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-8)
* [9](https://curriculum.code.org/csd-1718/unit6/3/#level-expando-3-9)
* (click tabs to see student view) ( 點擊看學生所看到的內容 )

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/4)

### **Student Instructions** 學生操作指示

# **Test Your Board 測試你的實驗板**

This level is just a quick test to make sure your board is working. To run this test, do the following:  
這個級別只是一個快速測試，以確保你的實驗板正常工作。 要運行此測試，請執行以下操作：

1. Plug your board into your computer with a USB cable  
   使用USB線將實驗板插入電腦
2. Click "Run"  
   點擊 “運行”
3. Wait a few seconds for either a Success or Error screen to pop up  
   等待幾秒鐘，彈出“成功”或“錯誤”螢幕
4. If your test was successful, move on to the next level. If not, try some of the troubleshooting steps  
   如果您的測試成功，請轉到下一級別。 如果沒有，請嘗試一些故障排除步驟

## **Troubleshooting** 故障排除

There are a number of things that could prevent your test from passing.  
有許多事情可能會阻止您的測試通過。

1. Try refreshing the page and running the test again (you should refresh the page for each of the following ideas as well)
2. Try connecting your board to a different USB port
3. Try a different USB cable. Some USB cables can only be used for charging, so make sure your cable supports data transfer!
4. Try a different board
5. Go to [studio.code.org/maker/setup](https://studio.code.org/maker/setup) to run a more detailed test

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/5)

### 嘗試更新頁面並再次運行測試（您應該更新以下每個做法的頁面）

### 嘗試將實驗板連接到其他USB端口

### 嘗試使用其他USB線。 有些USB線只能用於充電，因此請確保您的USB線能支持數據傳輸！

### 嘗試不同的實驗板

### 轉到studio.code.org/maker/setup運行更詳細的測試

### 在Code Studio上查看

### **Student Instructions** 學生操作指示

# **Control the LED**

Now that you know your board is working, you can try out the different LED blocks.

現在你知道你的板子可以動作, 你可以試試不同的LED區塊.

# **Do This**

* Click "Run" and discuss with your partner how the code works.
* Try the different LED blocks to see what they do.
* Hint: Don't forget to hit "Reset" and run your code again after you change it
* 按下 "Run” 並且和你的夥伴討論這些程式是如何運作的
* 試試不同的LED區塊並且看看它們是如何動作的
* 注意: 在你改變它後,不要忘記去押 "Reset” 並且再一次執行你的程式

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/6)

### **Student Instructions**

學生操作指示

Embedded Prediction

崁入式預測

# **Using Events**

使用事件

This program uses events. Look at the code with a partner and discuss what you think it will do. Once you have answered the questions click "Run" to check your answer.

* Will the light turn on when the program is "Run"? Why or why not?

該程序使用事件。與合作夥伴一起查看代碼並討論你的想法。一旦你回答了問題，請點擊“Run”以檢查你的答案。

* 當程序“運行”時，指示燈會亮起嗎？為什麼或者為什麼不？

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/7)

### **Student Instructions**

# **Using Events**

This program has two buttons. The "on" button works, but the "off" button still needs to be programmed.

這個程序有兩個按鈕。 “on” 按鈕有效，但“off”按鈕仍然需要被編寫程式才會動作。

# **Do This**

* Add code so that the second button turns the LED off.
* 增加一段程式碼，以便第二個按鈕能把LED關閉。

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/8)

### **Student Instructions**

# **Light Switch**

There are three more LED blocks that you haven't used yet. One of them can make the button work like a light switch, turning the LED on when it's off, but off when it's on.

那裏還有三個LED區塊你還尚未使用。其中一個可以使按鈕像一個燈開關一樣工作，當它關閉時打開LED，但是當它打開時關閉LED。

# **Do This**

* Try the [blink](https://docs.code.org/applab/blink/), [toggle](https://docs.code.org/applab/toggle/) and pulse blocks to see what they do.
* Use one of the blocks to make the button work like a light switch.
* If the LED is off, the button should turn it on.
* If the LED is on, the button should turn it off.
* 嘗試blink, toggle and pulse 那些區塊以查看它們的作用
* 使用其中一個塊使按鈕像燈開關一樣工作。
* 如果LED熄滅，按鈕應打開。
* 如果LED亮起，按鈕應將其關閉。

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/9)

### **Student Instructions**

# **Multi-function Light**

多功能燈

This app should let you choose between three types of lights: on, blink, and pulse. It should also have a button that turns the light completely off. Right now only the "on" button works, and the "off" button is completely missing.

這個應用程式應該能夠讓你在三種類型的燈模式做選擇：on, blink, and pulse. 它應該還有一個可以完全關閉燈的按鈕。現在只有 “on” 按鈕工作，“off” 按鈕完全失去作用。

# **Do This**

* Add code to make the "blink" and "pulse" buttons work.
* In design mode, add a new "off" button.
* Add code to make the "off" button work.
* LED Apps
* 添加程式代碼以使“blink”和“pulse”按鈕起作用。
* 在設計模式下，添加新的“關閉”按鈕。
* 添加程式碼以使“關閉”按鈕起作用。
* LED應用程序

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/10)

### **Student Instructions**

# **LED Apps**

In the last level, you made an app to control the LED with different buttons on the screen. In the next two levels, you'll see some other examples of apps, then have a chance to make one of your own.

As you look at the examples, think of what code might be useful for the app that you and your partner want to create.

在最後一級，你製作了一個應用程式通過營幕上的不同按鈕控制LED。在接下來的兩個級別中，您將看到其他一些應用程序範例，然後有機會創建自己的應用程式。 當你在查看範例時，請思考看看哪些程式碼可能對你和你的合作夥伴想要創建的應用程式有用。

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/11)

### **Student Instructions**

# **Put on a Light Show**

燈光秀展示

This app lets you put on a light show with the LED.

這個應用程式讓你用LED進行燈光秀。

Look at the code with your partner and answer the following questions:

與你的合作夥伴一起查看程式碼並回答以下問題：

* What does this line of code do: setProperty("screen","background-color",rgb(randomNumber(0,255),randomNumber(0,255),randomNumber(0,255))) });? ([Show me where.](https://curriculum.code.org/csd-1718/unit6/3/#triggercallout=code_triggered))
* 這行程式代碼做了什麼：setProperty（“screen”，“background-color”，rgb（randomNumber（0,255），randomNumber（0,255），randomNumber（0,255）））}）;？

Once you have answered the question, run the code and click the buttons to see what happens.

當你回答了問題，請運行程式碼並按下按鈕以查看發生的情況。

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/12)

### **Student Instructions**

Embedded Prediction

崁入式預測

# **Catch the Mouse**

抓住滑鼠

This is a clicker game, similar to the one that you built in the last lesson.

Look at the code with your partner and answer the following question:

這是一個點擊遊戲，類似於你在上一課中構建的遊戲。

與你的合作夥伴一起查看程式碼並回答以下問題：

* How many times do you need to click the mouse before the LED starts to blink?

Write your answer in the box below, then run the code and see what happens.

* 在LED開始閃爍之前，你需要多少次點擊滑鼠？

在下面的框中寫下你的答案，然後運行程式碼，看看會發生什麼。

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/13)

### **Student Instructions**

# **Make Your App - The Screen**

製作你的應用程序 – 螢幕

Now, you and your partner can make your own app using the LED. Discuss with your partner what kind of app you want to make.

現在，你和你的合作夥伴可以使用LED製作你們自己的應用。 與你的合作夥伴討論你們想要製作的應用程序類型。

# **Do This**

* Discuss with your partner what screen elements you will need in your app. Do you want...
* Buttons?
* Images?
* Text?
* In Design Mode, add the elements, making sure you give each a meaningful ID.
* 與你的合作夥伴討論你們的應用程式中將會需要的營幕元素。 你想要...
* 鈕扣？
* 圖片？
* 文本？
* 在設計模式下，添加元素，確保為每個元素提供有意義的ID。

[View on Code Studio](https://studio.code.org/s/csd6/stage/3/puzzle/14)

### **Student Instructions**

# **Make Your App - The Code**

製作你的應用程序 - 程式碼

Next, you'll need to add the code that makes your app work.

接下來，你需要添加使你的應用有效的程式碼。

# **Do This**

* Discuss with your partner what code you will need in your app. Do you need...
* Events?
* Variables?
* Setters?
* Random Numbers?
* In Code Mode, add in the code that will make your app work.
* 與你的合作夥伴討論你們在應用中將會需要的程式碼。 你需要...
* 活動？
* 變量？
* Setters？
* 隨機數？
* 在程式碼模式下，添加將使你的應用程式正常工作的程式碼。

## **Wrap Up (5 min)**

暖身活動

### **What's in a Board**

甚麼在版子裡面

**Journal:** Ask students to reflect on their introduction to the Circuit Playground. What did they think it was at first inspection? How did those expectations change after having programmed on the board?

日誌：讓學生反思他們對Circuit Playground的介紹。 他們在初次檢查時認為是什麼？ 在板子上編輯程式後，觀察這些預期中的改變是如何變化的？

## **Standards Alignment**

標準的通用準則

#### **[View full course alignment](https://curriculum.code.org/csd-1718/standards/)**

#### **CSTA K-12 Computer Science Standards**

CSTA K-12計算機科學標準

**AP** - Algorithms & Programming

* **2-AP-11** - Create clearly named variables that represent different data types and perform operations on their values.
* **2-AP-13** - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
* **2-AP-16** - Incorporate existing code, media, and libraries into original programs, and give attribution.
* **2-AP-19** - Document programs in order to make them easier to follow, test, and debug.
* AP - 算法和編程
* 2-AP-11 - 創建明確命名的變量，表示不同的數據類型並對其值執行操作。
* 2-AP-13 - 將問題和子問題分解為多個部分，以便於程序的設計，實施和審查。
* 2-AP-16 - 將現有程式碼，媒體和資料庫合併到原始程序中，並提供歸屬。
* 2-AP-19 - 文建程序，以便於追蹤，測試和障礙排除。

**CS** - Computing Systems

CS - 計算系統

* **2-CS-02** - Design projects that combine hardware and software components to collect and exchange data.
* 2-CS-02 - 設計項目，結合硬體和軟體組件來收集和交換數據。
* **2-CS-03** - Systematically identify and fix problems with computing devices and their components.
* 2-CS-03 - 系統地識別和修復計算設備及其組件的問題。

**Lesson 4: Input Unplugged**

不插電輸入

Unplugged

不插電

**Overview**

In preparation for delving deeper into programming with App Lab, students will explore how a handful of different programs written in both Game Lab and App Lab handle taking input from the user. After comparing and contrasting the approaches they saw in the example apps, students group up to act out the two different models for input (conditionals in an infinite loop and asynchronous events) to gain a better understanding of how they work.

**概覽**

為準備深入研究 App Lab的編程，學生將探索在Game Lab 和 App Lab 環境中不同程序如何處理來自用戶的輸入。 在比較和對比他們在示例應用程序中看到的各種方法後，學生們將分組完成兩個不同的輸入模型（無限循環中的條件和異步事件）以理解它們的工作方式。

**Purpose**

This lesson is intended to help students transition from the draw loop and conditional model of input used in Game Lab to the event-driven model used in App Lab. While students have experienced and learned a bit about event-driven programming in Unit 4, a deeper understanding of how the events work will help when it comes to responding to events on the Circuit Playground later in this unit.

**課程宗旨**

本課程旨在協助學生從 Game Lab 中使用的畫圖迴圈和輸入條件模型過渡到 App Lab中使用的事件驅動模型。在第 4 單元中，學生已經體驗學習了關於事件驅動編程的部分知識，但，更深入理解事件的運作原理將有助於在本單元後期回應 Circuit Playground上的事件。

**Agenda**

· **[Warm Up (10 min)](https://curriculum.code.org/csd-18/unit6/4/#input-unplugged0)**

· [Comparing Input Methods](https://curriculum.code.org/csd-18/unit6/4/#input-unplugged1)

· **[Activity (40 min)](https://curriculum.code.org/csd-18/unit6/4/#input-unplugged2)**

· [Input Unplugged](https://curriculum.code.org/csd-18/unit6/4/#input-unplugged3)

· [Activity Debrief](https://curriculum.code.org/csd-18/unit6/4/#input-unplugged4)

· **[Wrap Up (2 min)](https://curriculum.code.org/csd-18/unit6/4/#input-unplugged5)**

· [Reflection](https://curriculum.code.org/csd-18/unit6/4/#input-unplugged6)

[**View on Code Studio**](https://studio.code.org/s/csd6-2018/stage/4/puzzle/1/)

**課程大綱**

**課程暖身（10分鐘）**

·比較輸入法

**活動（40分鐘）**

·不插電輸入法

·活動總結

**課程總結（2分鐘）**

·檢討反思

在Code Studio上查看

**Objectives**

**Students will be able to:**

· Compare and contrast multiple ways to take input

· Describe the elements of an event handler

· Model different methods of taking user input

**課程目標**

學生將能夠：

·比較和對比多種輸入方式

·描述事件處理程序的元素

·模擬不同的用戶輸入方法

**Preparation**

· Prepare to display example programs for the whole class.

· A half deck of cards for each group of three students or [deck-of-cards.js.org](https://deck-of-cards.js.org/).

· Print one copy of [Input and Events - Activity Guide](https://docs.google.com/document/d/1Pqq09UzF8xNP688FwK7x9zrsjxxM4WajAVtEwM1wYPE/edit) for each group of three students.

**課程準備**

準備要展示給全班同學的示範程式

為各組（三名學生為一組）準備半疊卡片或預備 deck-of-cards.js.org

為各組印製一份《輸入和事件 - 活動指南》。

**Links**

Heads Up! Please make a copy of any documents you plan to share with students.

**推薦連結**

**注意：請將要分享給學生的檔案備份**

**For the Students**

· [Input and Events](https://docs.google.com/document/d/1Pqq09UzF8xNP688FwK7x9zrsjxxM4WajAVtEwM1wYPE/edit) - Activity Guide Make a Copy

**給學生**

《輸入和事件 - 活動指南》

**Support**

**[Lesson Forum](https://forum.code.org/c/csd6/)**

[**Report a Bug**](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%204%20curriculum.code.org/csd-18/unit6/4/)

**協助**

**課程論壇**

**問題回報**

**Teaching Guide**

**Warm Up (10 min)**

Comparing Input Methods

Discussion Goal

Goal: This is intended to be a very broad question, and as such could go in a number of different directions. If your students are struggling, encourage them to think about specific programs that they wrote in units 3 and 4.

Discuss: Yesterday we wrote some apps that took input from a user, but how did the program know when to take input?

Display: Open up the Code Studio stage for this level. For each of the example apps, ask the class to identify:

1. Where is the input coming from? (e.g. keyboard, mouse, etc)

2. What input value is the program looking for?

3. How will the program respond to input?

Code Studio levels

· **Lesson Overview**

· [Student Overview](https://curriculum.code.org/csd-18/unit6/4/#level-expando-4-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/4/puzzle/1)

· **Input Examples**

·  [2](https://curriculum.code.org/csd-18/unit6/4/#level-expando-4-2)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/4/puzzle/2)

**教學指南**

**課程暖身 (十分鐘)**

比較輸入法

討論目標

目標: 這是一個範圍很廣的議題，因此，討論可能會往許多不同方向發展。如果你發現學生在討論過程中卡住、苦苦掙扎，請鼓勵他們回想在第 3 和第 4 單元時他們所寫的程式。

討論：昨天我們寫了一些從用戶那裡獲取資訊輸入的應用程式，但程式本身該如何判斷應該接受輸入的時間點呢？

展示：打開此級別的 Code Studio 階段。 對於每個示例應用程序，要求班級確定以下幾項：

1.輸入來自哪裡？ （例如鍵盤，鼠標等）

2.程序要查找什麼輸入值？

3.程序如何響應輸入？

Display: Open up the Code Studio stage for this level. For each of the example apps, ask the class to identify:

1. Where is the input coming from? (e.g. keyboard, mouse, etc)

2. What input value is the program looking for?

3. How will the program respond to input?

Code Studio levels

· **Lesson Overview**

· [Student Overview](https://curriculum.code.org/csd-18/unit6/4/#level-expando-4-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/4/puzzle/1)

· **Input Examples**

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/4/puzzle/2)

Student Instructions

**Input in Game Lab**

You don't need to make any changes to the code here

Take a minute to read through the code and run the program. Discuss with a partner:

1. What is the source of input?
2. Where and how does the program check for input?
3. What does the program do when it gets the correct input?
4. What is the program doing when there is no input?

Student Instructions

**Input in App Lab**

You don't need to make any changes to the code here

Take a minute to read through the code and run the program. Discuss with a partner:

1. What is the source of input?
2. Where and how does the program check for input?
3. What does the program do when it gets the right input?
4. What is the program doing when there is no input?

Prompt: Ask the class the following questions, keeping track of answers on the board. The goal at this point isn't to answer questions that come up, but to record them so we can see how the class feels again after the lesson.

* How did the different apps we looked at approach taking input similarly or differently?
* Which way made the most sense to you?
* Which way made the least sense?
* What questions to you have about how these programs take input?

**Activity (40 min)**

Input Unplugged

***Remarks***

In the previous activity we explored several different kinds of input, but how do our programs actually process that input? As a class, and in small groups, we're going to model how input is taken both in Game Lab and App Lab.

Group: Place students in groups of 4

Distribute: Hand out a copy of [Input and Events - Activity Guide](https://docs.google.com/document/d/1Pqq09UzF8xNP688FwK7x9zrsjxxM4WajAVtEwM1wYPE/edit) and half of a deck of cards to each group (the deck doesn't need to be perfectly split, but should include both red and black cards in each half). The activity guide includes four different role pages, printed front and back.

Model: Choose one group to model the activity for the class.

***Input and Events***

The goal of this activity is to help students better understand how events work, and how using events differs from the input model that they used in Game Lab. In groups of four, students will model the two different approaches to taking input. By the end of the activity students should understand that the [onEvent](https://docs.code.org/applab/onEvent/) block sets up a input to watch for events in the background, allowing input handling without explicitly asking each input for a value constantly. The roles for this activity are:

Program: The Program reads the code aloud and performs any actions dictated by the inputs. Inputs 1-3: The Inputs draw cards from the deck, which represent their changing input values.

Version A: Asking for Input

Teaching Tip

The programs for both versions of this activity are purposely very simple to allow students to focus on the processes involved in each input method. If your students pick up on the ideas quickly, consider providing some more detailed example programs to push on their understanding. In particular, placing the inputs or events in a different order and nesting conditionals can be useful in revealing misconceptions.

This first version models the Game Lab approach to taking input. In this scenario the Program explicitly asks each input for a value every time the draw loop is run.

Rules for Version A:

* The Program reads each line of code aloud.
* At the beginning of each draw loop, each Input draws a new card from the deck. This represents the current state of that input.
* Whenever the Program reaches an input checking command (inputOne(), inputTwo(), or inputThree()), the Program asks to see that inputs card.
* If the value of the card shown matches the associated conditional, the Program performs the action(s) inside the conditional.

After groups have run the program a few times and seem to understand how it works, have them switch up roles so that each student gets a chance to be the Program.

Version B: Input Events

This time around, instead of constantly asking for input in a loop, the Program will assign each Input an Event to watch for. In the context of this activity, an Event is either a red or black card, but in a program that Event could be a click on a button, movement of the mouse, press of a key, or more. This is a simplified model for the way input is handled in App Lab.

Rules for Version B:

* Each Input draws a card from the deck at a rate of roughly one per second.
* The Program reads each line of code aloud.
* When the Program reaches an [onEvent](https://docs.code.org/applab/onEvent/) command, they tell the specified Input which Event to watch for, and what the Response should be.
* When assigned an Event, the Input writes down the details in their Events to Watch table.
* Every time an Input draws a card, they check to see if it matches one of the Events in their table. If it does, they tell the Program to perform the Response.

There are a few important simplifications that are made in this model to minimize the number of roles and rules:

* This model implies that Inputs are actually watching for Events on their own. In reality, an Event Handler running in the background watching for the specified event.
* In this model the Input tells the Program what actions to take. In reality, when an Event Handler is triggered, the Program goes to that portion of their code and runs everything in the function.

Activity Debrief

Discuss: Looking back at the comments gathered from the warm up activity, what things are more clear after having run the activity? Do you have any new questions?

**Wrap Up (2 min)**

Reflection

Journal: What's the difference between the way that Game Lab and App Lab handle inputs?

**Standards Alignment**

[View full course alignment](https://curriculum.code.org/csd-18/standards/)

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming

If you are interested in licensing Code.org materials for commercial purposes, [contact us.](https://code.org/contact)

**Lesson 5: Board Events**

#### **App Lab | Maker Toolkit**

## Overview

This lesson transitions students from consider the Circuit Playground as strictly an output device towards using it as a tool for both input and output. Starting with the hardware buttons and switch,using the hardware buttons and switch, students learn to use [onBoardEvent()](https://docs.code.org/applab/onBoardEvent/), analogously to [onEvent()](https://docs.code.org/applab/onEvent/), in order to take input from their Circuit Playgrounds.

這堂課幫助學生了解 Circuit Playground 電路板不只是個輸出裝置，也能是個輸出兼輸入的裝置。學生將從使用硬體按鈕及開關開始，學習如何應用onBoardEvent() (類似OnEvent() )來接收Circuit Playground給的輸入訊號。

## Purpose

This lesson marks the transition from using the board solely as an output device to using it for both input and output. The [onBoardEvent()](https://docs.code.org/applab/onBoardEvent/) block works much like [onEvent()](https://docs.code.org/applab/onEvent/), with the most significant different being that the first parameter is a board object (a variable) while [onEvent()](https://docs.code.org/applab/onEvent/) takes a UI element ID (a string).

此課程幫助學生認識到這電路板不只是個輸出裝置，亦能是個輸出兼輸入裝置。onBoardEvent()的功能很像是OnEvent() ，最大的差別是前者的第一個參數是電路板的觸發事件（為一變數），而後者的第一個參數則是介面上的一個元件識別碼（為一字串）。

## Agenda

* [Warm Up (5 min)](https://curriculum.code.org/csd-18/unit6/5/#board-events0)
* [Board Inspection: Inputs](https://curriculum.code.org/csd-18/unit6/5/#board-events1)
* [Activity (45 min)](https://curriculum.code.org/csd-18/unit6/5/#board-events2)
* [Taking Input from the Board](https://curriculum.code.org/csd-18/unit6/5/#board-events3)
* [Wrap Up (2 min)](https://curriculum.code.org/csd-18/unit6/5/#board-events4)

### [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/1/)

## Objectives

課程目標

### Students will be able to:

* Attach an event handler to a hardware input
* Choose the appropriate event for a given scenario

學生將學會：

* 聯結事件處理器與硬體輸入
* 為特定情況選擇合適的事件種類

## Introduced Code

* [onBoardEvent(component, event, function(event) {...});](https://docs.code.org/applab/onBoardEvent/)
* [toggleSwitch](https://docs.code.org/applab/toggleSwitch/)
* [toggleSwitch.isOpen](https://docs.code.org/applab/toggleSwitch.isOpen/)
* [button(L/R)](https://docs.code.org/applab/buttons/)
* [buttonL.isPressed](https://docs.code.org/applab/isPressed/)

## Support

### [Lesson Forum](https://forum.code.org/c/csd6/)

### [Report a Bug](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%205%20curriculum.code.org/csd-18/unit6/5/)

# **Teaching Guide**

## Warm Up (5 min)

暖身活動

### Board Inspection: Inputs

認識電路板：輸入

**Distribute:** Pass out a board and USB cable to each pair of students. Let students know that they should not yet plug the boards in.

**Prompt:** Ask pairs to spend one minute looking over the board, focusing on potential input devices. Based on what you already know about this board, how do you think you might use it to get input?

**Share:** Have groups share back their thoughts to the whole group, keeping track of ideas on the board. Push students to support their ideas with evidence from reviewing the board, but don't worry about ensuring correctness at this point.

**分配：**將學生分為兩兩一組，每組學生發放一個電路板以及一條USB傳輸線，並說明先不要將傳輸線插到電路板上。

**提問：**請每組學生花一分鐘端詳電路板，找出可能是輸入裝置的元件。詢問學生依其目前對於這電路板的認識，推斷他們會如何利用這電路板得到輸入？

**分享：**請各組跟大家分享他們的看法，並把大家的想法都記下來。鼓勵學生以在板子上實際看到的東西為依據，來支持他們的看法，不用擔心是否正確。

## Activity (45 min)

### Taking Input from the Board

從電路板得到輸入

**Transition:** Send students to Code Studio.

接下來：讓學生進入 Code Studio

### Code Studio levels

Lesson Overview

[[Student Overview]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/1)

Responding to Board Events

[[Student Overview]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/2)

# **Board Events**

電路板事件

The [onEvent()](https://docs.code.org/applab/onEvent/) block is used to set up *event handlers* that can watch for certain *events* to occur in your app, such as clicking on a button, selecting an item from a dropdown, or moving the mouse. The Circuit Playground has a similar block called [onBoardEvent()](https://docs.code.org/applab/onBoardEvent/) to watch for events that occur on the board. Similar to [onEvent()](https://docs.code.org/applab/onEvent/), you need to specify what the handler should be watching, what event to watch for, and how to respond to that event.

onEvent() 的區塊是用來建立事件處理器，以偵測應用程式中某些事件的發生，例如按按鈕、選取下拉式選單的物件、或是移動滑鼠。Circuit

Playground有個類似的區塊叫做 onBoardEvent()，它可以找出電路板上所發生的事件。就像 onEvent() 一樣，你需要指示事件處理器要觀測什麼、要尋找哪些事件，以及就特定事件要做什麼反應。

### onBoardEvent

It's important to understand in Event-Driven programming that:

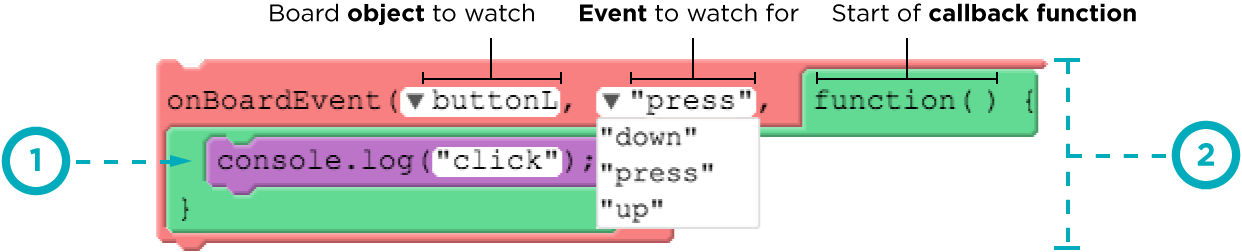
* **Users trigger events** - Events occur when users interact with inputs on the board, such the buttons, the switch, and others.
* **Events trigger code** - When an event occurs or "fires" it can be used to trigger a particular function.

對於事件觸發型的程式，你必須知道：

* 使用者觸發事件：事件會在使用者與電路板的輸入裝置有互動時被觸發，像是按按鈕、切換開關等。
* 事件觸發程式：某個事件的發生可以用來觸發某個特定函數的執行。

The Maker Toolkit [onBoardEvent()](https://docs.code.org/applab/onBoardEvent/) block is a type of event-handling function called an Event Listener. It wraps all of the setup up you need to do event handling into one command that has several parts. Here is an example with everything labeled:

Maker Toolkit onBoardEvent() 區塊是一種事件處理函數，叫作事件監聽器(Event Listener)。它把事件處理所需的全部設定含括在一個可分為數個組成部分的指令裡。以下為一個實際範例（每個部分都有標示）：



* Board Events

[3]

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/3)

### Student Instructions

See contained level for markdown

# **Board Events**

電路板事件

The [onBoardEvent()](https://docs.code.org/applab/onBoardEvent/) block allows you to start taking input from your Circuit Playground. Look at the program with a partner and discuss what you think it'll do. Once you've come up with an answer, click "Run" to check your answer.

onBoardEvent() 區塊讓你可以開始從Circuit Playground接收輸入。與同組夥伴一起研究此程式並討論你們認為它是如何進行的。想出答案後，按下”Run”來確認你們的答案是否正確。

* What will happen when you press the left button down?
* 按下左邊按鈕時會發生什麼事？

[4]

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/4)

### Student Instructions

# **Using Board Events**

使用電路板事件

Right now the LED turns on when the left button (buttonL) is pressed down. It should turn off when the right button (buttonR) is pressed down.

至目前為止，當按下左邊按鈕(buttonL)時，LED燈就會被打開。當按下右邊按鈕(buttonR)時，LED燈應要被關掉。



# **Do this**

請做：

* Add another [onBoardEvent()](https://docs.code.org/applab/onBoardEvent/) block to turn the LED off when the right button is pressed down.
* 加入另一個 onBoardEvent()區塊，使當右邊按鈕被按下時，LED燈就會被關掉。

[5]

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/5)

### Student Instructions

# **Other Board Events with the Buttons**

其他電路板事件的按鈕

Instead of using the right button, you can make the LED turn off when the left button comes back up.

除了使用右邊按鈕之外，也可以設定當左邊按鈕被按起時，把LED燈關掉。

# **Do this**

請做：

* Create an 'up' event for the left button.
* 新增一個左邊按鈕的按起事件。
* Add code to turn the LED off when the button pops back up.
* 加入程式，讓按鈕被按起來時，會關掉LED燈。

Buttons and Switch

[[Student Overview]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-6-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/6)

# **Physical Input**

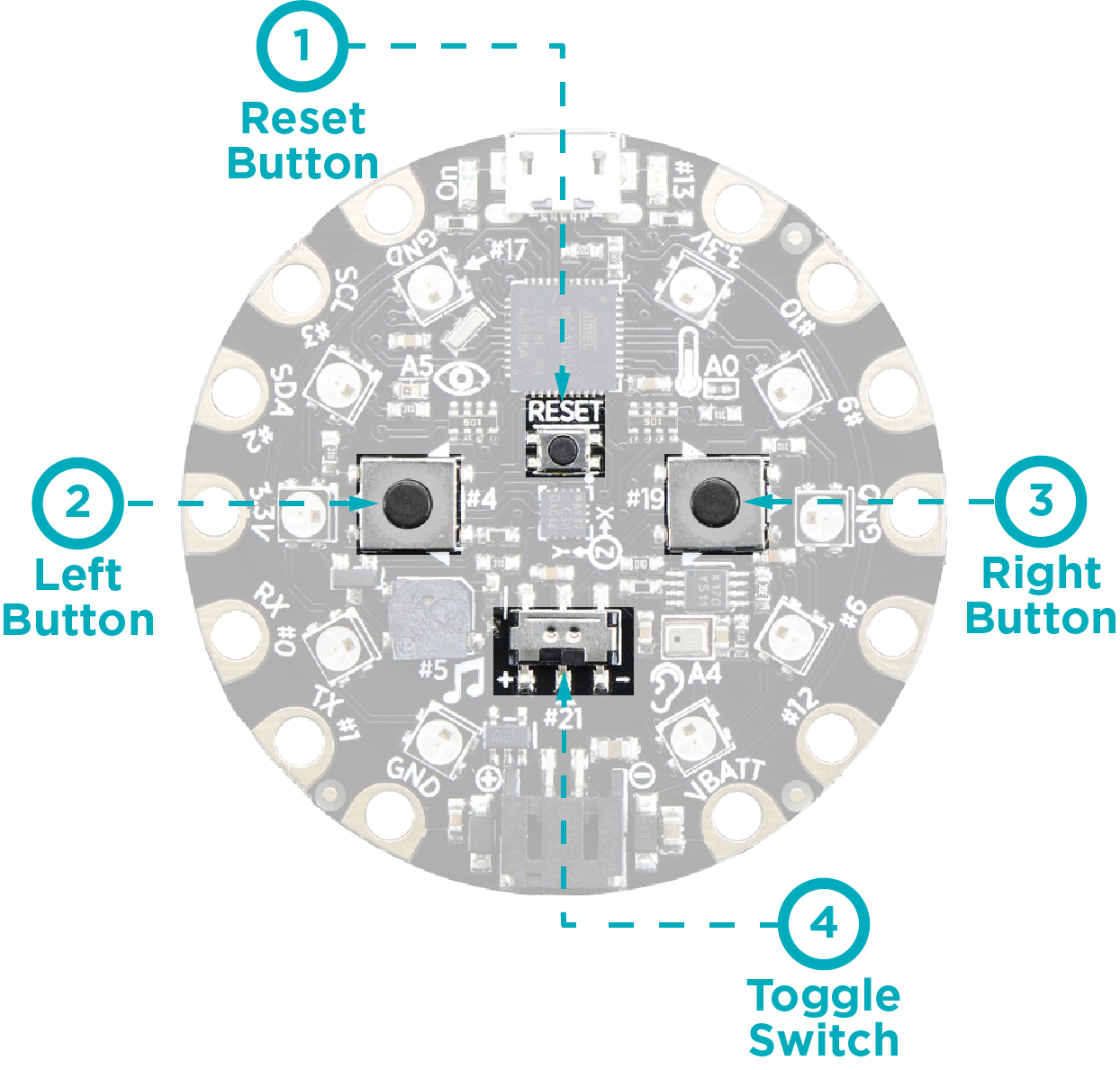
實際輸入

The Circuit Playground can be used for more than just *outputting* information (like you've done with LED), it can also be used as an *input* device. The simplest forms of input available on the board are the **buttons** and the **switch**.

Circuit Playground不但可以單純的用來輸出資訊(就像你已經做過的LED燈實驗)，也可以作為輸入裝置。電路板上最簡單的輸入形式就是按鈕及開關。

## Identifying the Inputs

辨別輸入裝置

****

Using the Switch

* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/7)

### Student Instructions

# **Using the Toggle**

使用切換開關

The toggle switch can flip in one of two positions: open and close. This program uses it to make a light switch. Right now it can only turn the light on.

切換開關有兩種狀態：開或關。本程式用它來作燈的開關。它現在的功能只有開燈與關燈。

# **Do this**

請做：

* Create a new toggle switch event that is triggered when the switch is flipped to close.
* 新增一個切換開關事件，使它在開關被切換至關時觸發。
* Add code so that when the toggle switch is flipped to close the light turns off.
* 加入一段程式使開關被切到關時熄滅燈光。

[[8]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-8)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/8)

### Student Instructions

See contained level for markdown

# **More on the Toggle Switch**

關於切換開關的更多事

You can also check whether the toggle switch is in the open or close position anywhere in your program.

What different things does the button do when the toggle switch is open or close?

在程式的各處其實都可以確認切換開關當時的狀態是開還是關。

當切換開關是開或關時，按鈕可以做的事有什麼不一樣呢？

[[9]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-9)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/9)

### Student Instructions

# **State and the Toggle Switch**

切換開關與其狀態

This program tells you whether the toggle switch is opened or closed, but it needs the [toggleSwitch.isOpen](https://docs.code.org/applab/toggleSwitch.isOpen/) block to work. [toggleSwitch.isOpen](https://docs.code.org/applab/toggleSwitch.isOpen/) is true if the switch is opened and false if the switch is closed.

此程式會告訴你切換開關的狀態是開啟還是關閉，但須使用[toggleSwitch.isOpen](https://docs.code.org/applab/toggleSwitch.isOpen/)區塊。如果切換開關是開啟的，[toggleSwitch.isOpen](https://docs.code.org/applab/toggleSwitch.isOpen/)則為真，反之則為假。

# **Do this**

請做：

* Complete the if-statement with the [toggleSwitch.isOpen](https://docs.code.org/applab/toggleSwitch.isOpen/) property.
* 完成於if條件式中使用[toggleSwitch.isOpen](https://docs.code.org/applab/toggleSwitch.isOpen/)。
* Run the program and press the left button, then flip the toggle switch and press the button again.
* 執行程式並按下左按鈕，然後切換開關後再按一次左按鈕。

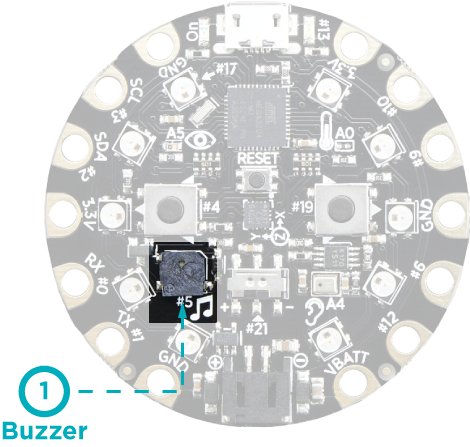
Circuit Playground: Buzzer

[[Student Overview]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-10-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/10)

# **Producing Output**

產生輸出

****

The LED is probably the simplest form of output available, and though there are many useful things a simple light can communicate, sometimes we need a different form of output. We've used light as an output, but what about sound? Your board has a [buzzer](https://docs.code.org/applab/buzzer/) object that can play simple sounds.

大概是最簡單的輸出形式了。雖然有很多事情可以用光來傳達，有時我們還是需要不同形式的輸出。我們已可使用光作為輸出，聲音呢？你的電路板上有個可以發出簡單聲音的蜂鳴器物件。

### The Buzzer

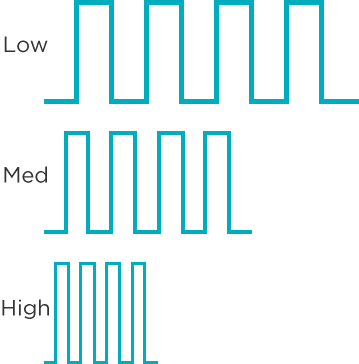
蜂鳴器

Don't expect the [buzzer](https://docs.code.org/applab/buzzer/) to make beautiful music - it's a simple tool for simple output!

請不要期待蜂鳴器會發出美妙的音樂，它只是個簡單的輸出工具！

1. This cube contains a simple buzzer (also known as a *piezo* or *piezoelectric* buzzer). When electricity is sent to the buzzer, it vibrates and makes noise.
2. 這個小方塊有一個簡單的蜂鳴器（也被稱作壓電，或是壓電蜂鳴器）。當電被傳送到蜂鳴器時，它就會震動並且發出蜂鳴。

### Making Different Sounds

****

When using the LED our choice of output was pretty simple; it's either on or off. The buzzer can actually play a range of different sounds. We're able to do that by sending pulses of electricity to the buzzer at different speeds. This is like turning on and off a light repeatedly - the faster we flip the switch, the higher the sound it will produce.

當使用LED燈當作輸出裝置時，輸出的選擇只有簡單的兩種方式：開或關。蜂鳴器則可以有許多不同的聲音變化。這是藉由使用不同的速度傳送電脈波給蜂鳴器所致。這就像用不同的速度重複開關燈，開關的速度越快，所發出的聲音越高。

The Buzzer

[[11]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-11)

* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/11)

### Student Instructions

# **Using the Buzzer**

使用蜂鳴器

You can use the buzzer in a lot of different ways, but right now we're just going to use the [buzzer.frequency()](https://docs.code.org/applab/buzzer.frequency/).

你可以用許多不同的方式來使用蜂鳴器，但現在我們先使用[buzzer.frequency()](https://docs.code.org/applab/buzzer.frequency/)（蜂鳴器的頻率）。

# **Do this**

請做：

* Pull out a [buzzer.frequency()](https://docs.code.org/applab/buzzer.frequency/) block from the toolbox and play around with it. You'll notice it has two parameters, what happens when you change either, or both of those?
* 從工具組取用[buzzer.frequency()](https://docs.code.org/applab/buzzer.frequency/)區塊並試著操作看看。你會發現它有兩個參數，改變它們其中一個或是兩個都改，會造成什麼變化呢？
* Run the program and enjoy the music of your Circuit Playground.
* 執行程式並聆聽你的Circuit Playground所發出的音樂。

[[12]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-12)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/12)

### Student Instructions

# **Buzzer Duration**

蜂鳴器持續時間

As you've noticed, [buzzer.frequency()](https://docs.code.org/applab/buzzer.frequency/) has two parameters: frequency and duration. Both of these take a number as input.

如你所見，[buzzer.frequency()](https://docs.code.org/applab/buzzer.frequency/)有兩個參數，頻率持續時間。這兩個參數都是輸入數字

* Frequency changes how the buzzer sounds. The bigger the number you give it, the higher the buzzer's sound will be.
* 頻率改變的是蜂鳴器的音高效果，數字越大，蜂鳴器會發出越高的聲音。
* Duration determines how long the buzzer is going to play a sound for in milliseconds.
* 持續時間則決定了蜂鳴器發出聲音的長短，單位是毫秒。

For right now we're going to focus on duration, the second parameter.

現在我們將把重點放在第二個參數，持續時間。

# **Do this**

請做：

Try out a few things with duration:

試試關於持續時間的不同實驗：

* Change the duration of the buzzer to something longer than 100.
* 將持續時間設定為大於100的數字。
* Try setting the duration to 0, what happens?
* 試試看把持續時間改為零。會變成怎麼樣呢？

Run the program multiple times to see what happens before you move on.

多執行此程式幾次來看它的變化。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/puzzle/13)

### Student Instructions

# **Buzzer and the Buttons**

蜂鳴器以及按鈕

This program should use the buttons to play long and short buzzes.

此程式應使用按鈕來播放長及短的蜂鳴。

# **Do this**

請做：

* Change the duration parameter of [buzzer.frequency](https://docs.code.org/applab/buzzer.frequency/) to make the left button play a long buzz, and the right button play a short buzz.
* 改變[buzzer.frequency](https://docs.code.org/applab/buzzer.frequency/)的持續時間參數，使左邊按鈕播放長蜂鳴、右邊按鈕播放短蜂鳴。

Levels

級別

[[Extra]](https://curriculum.code.org/csd-18/unit6/5/#level-expando-5-14)

* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/5/extras)

### Student Instructions

給學生的指示

# **Challenge: Sound Board**

挑戰：聲音電路板

Check with your teacher before starting the challenge.

Now that you know how to use board events, the buttons, switch, and the buzzer, try making a program that turns your board into a sound board when you press the buttons or flip the switch. Play around with the duration and frequencies of the sounds, and listen to how it turns out.

請在開始此項挑戰之前與你的老師確認。

既然你已熟悉如何使用電路板事件、按鈕、開關、及蜂鳴器，現在請試著寫一個程式，當按下按鈕或是切換開關時，可將板子變成聲音電路板。嘗試變換不同的聲音持續時間及頻率參數，並聽聽看它所發出的聲音。

## Wrap Up (2 min)

**Journal:** Given the types of board events that you saw today, how might you take input from other elements on the board? What other events do you think might exist?

## Standards Alignment

#### **[View full course alignment](https://curriculum.code.org/csd-18/standards/)**

#### **CSTA K-12 Computer Science Standards**

**AP** - Algorithms & Programming

* **2-AP-11** - Create clearly named variables that represent different data types and perform operations on their values.
* **2-AP-12** - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
* **2-AP-13** - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
* **2-AP-16** - Incorporate existing code, media, and libraries into original programs, and give attribution.
* **2-AP-17** - Systematically test and refine programs using a range of test cases.
* **2-AP-19** - Document programs in order to make them easier to follow, test, and debug.

**CS** - Computing Systems

* **2-CS-01** - Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices.
* **2-CS-02** - Design projects that combine hardware and software components to collect and exchange data.
* **2-CS-03**

# **Lesson 6: Getting Properties**

#### **App Lab | Maker Toolkit**

## **Overview**

課程概述

This lesson introduces students to the [getProperty](https://docs.code.org/applab/getProperty/) block, which allows them to access the properties of different elements with code. Students first practice using the block to determine what the user has input in various user interface elements. Students later use [getProperty](https://docs.code.org/applab/getProperty/) and [setProperty](https://docs.code.org/applab/setProperty/) together with the counter pattern to make elements move across the screen. A new screen element, the slider, and a new event trigger, onChange, are also introduced.

這堂課向學生介紹 getProperty 區塊，讓他們透過程式碼存取不同元件的屬性。學生先練習使用 getProperty 區塊，取得使用者在畫面上各種欄位中輸入的資料，然後再使用 getProperty 與 setProperty 區塊，配合計數器的使用，讓元件在螢幕上移動。這堂課也會介紹一個新的顯示元件 slider 和一個新的事件 onChange。

## **Purpose**

課程目的

So far, students have only used the button press event to gather information from the App Lab screen. Using [getProperty](https://docs.code.org/applab/getProperty/) opens the students to gathering a wide range of information from the user. Getting properties also allows students to create programs that don't rely on knowing an element's properties when the code is written, so their programs can be more dynamic and interactive. These features will be important as students move toward making an interactive game in the next few lessons.

到目前為止, 學生只使用按鍵事件來取得 App Lab螢幕上的資訊. 使用[getProperty](https://docs.code.org/applab/getProperty/) 能讓學生從使用者蒐集更廣泛的資訊. 動態取得屬性允許學生在撰寫程式碼時,不需要知道元件有那些屬性, 因此寫出來的程式會更具有動態效果和互動性.在接下來的幾堂課中,學生會學習製作互動遊戲,這些功能將會變得非常重要.

## **Agenda**

課程流程

* [Warm Up (5 min)](https://curriculum.code.org/csd-18/unit6/6/#getting-properties0)
* [Activity (40 min)](https://curriculum.code.org/csd-18/unit6/6/#getting-properties1)
* [Wrap Up (10 min)](https://curriculum.code.org/csd-18/unit6/6/#getting-properties2)
* 暖身活動 (5 min)
* 活動內容 (40 min)
* 總結 (10 min)

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/1/)**

## **Introduced Code**

介紹程式碼

* [getProperty(id, property)](https://docs.code.org/applab/getProperty/)

## **Support**

課外資源

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

討論區

### [**Report a Bug**](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%206%20curriculum.code.org/csd-18/unit6/6/)

問題回報

# **Teaching Guide**

教學指南

## **Warm Up (5 min)**

暖身活動

**Review:** Ask students to think of the different types of information that they were able to get from the board in the last lesson (e.g. button presses, toggle state, etc.)

複習: 要求學生回想在上一堂課，他們從實驗板得到那些不同類型的資訊。例如，按下按鈕和切換狀態等。

Remarks

So far we've seen lots of ways that we can get information from our Circuit Playground. Today we're going to look at more ways that we can get information from the screen of our app.

備註

目前為止，我們已經看過許多從 Circuit Playground 上取得資訊的方法。今天我們要看看更多從 app 畫面上取得資訊的方法。

## **Activity (40 min)**

活動內容

Send students to Code Studio.

### **Code Studio levels**

* Lesson Overview
* [Student Overview](https://curriculum.code.org/csd-18/unit6/6/#level-expando-6-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/1)

* getProperty
* [Student Overview](https://curriculum.code.org/csd-18/unit6/6/#level-expando-6-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/2)

* Getting Properties
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/3)

### **Student Instructions**

學生操作指示

# **Updating Properties**

Here's an example of an app that uses both [getProperty](https://docs.code.org/applab/getProperty/) and [setProperty](https://docs.code.org/applab/setProperty/) to update elements on the screen. Read through the code, and then test it out by filling out the input fields and clicking "Save"

這裡有一個 app的範例，它同時使用 getProperty 和 setProperty 來更新畫面上的欄位。仔細閱讀程式碼，然後在畫面上的欄位輸入資料並按下 Save 按鈕執行測試。

*Note: you don't need to change any code here, but read through the provided code to see how it works.*

*說明: 你不需要更改程式碼*，*但是請仔細閱讀所提供的程式碼*，*看看它如何運作*。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/4)

### **Student Instructions**

學生操作指示

# **getProperty and Variables**

This program uses the name variable to display the name typed into "name\_input" when the Submit button is clicked. It should do the same for *friend\_input* and *friend\_output*, but it's missing a block.

當使用者按下 Submit 按鈕時，這支程式使用 “name” 變數來顯示使用者輸入到 “”name\_input” 欄位的名字。你可以用相同的方式處理 friend\_input 和 friend\_output 欄位，但是請注意，那裡少了一個區塊。

# **Do This**

動手做

* Read the code and comments, then run the program to see how it works. (Make sure you type in your name and your friend's name before hitting "Submit".)
* Use a [getProperty](https://docs.code.org/applab/getProperty/) block to store the text property of *friend\_input* into your the variable "friend\_name". ([Show me where.](https://curriculum.code.org/csd-18/unit6/6/#triggercallout=code_triggered))
* 閱讀程式碼和註解，然後執行程式以便瞭解它如何運作。(在按下 "Submit" 之前，你必須輸入你的名字和你朋友的名字)
* 使用[getProperty](https://docs.code.org/applab/getProperty/)區塊將*friend\_input*的文字屬性存放到 "friend\_name"變數中。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/5)

### **Student Instructions**

學生操作指示

# **Dropdowns**

Text Inputs are nice when you want users to be able to enter *anything* they want, but often you want to restrict them to just a few choices. For example, if you want your user to pick a color, you probably want to make sure they can only choose colors that will actually work.

The provided dropdown with the ID "color\_input" allows a user to choose from a handful of background colors. When the "Set Color" button is clicked, the screen will turn that color.

當你希望使用者能夠輸入任何他們想要的資訊, 文字輸入是很好的方式. 但是通常你會限制他們僅使用幾種選擇. 舉個例子, 如果你希望使用者選一種顏色, 你或許希望他們只能選擇少數適當的顏色. 在"color\_input"欄位提供下拉式選單,就能讓使用者選擇少數幾種背景顏色. 當"Set Color"按鍵被點擊, 螢幕就會換成新的顏色.

# **Do This**

動手做

* Inside the event block, create a variable called [color](https://docs.code.org/applab/color/) and assign it a value using the [getProperty](https://docs.code.org/applab/getProperty/) block.
* Hint: You'll need to use the "value" property to get the user's choice.
* Use the [setProperty](https://docs.code.org/applab/setProperty/) block to set the "background-color" property of "screen1" to your variable [color](https://docs.code.org/applab/color/).
* Hint: Make sure you type [color](https://docs.code.org/applab/color/) without quotation marks.
* Test your program by picking a color and clicking "Set Color".
* 在事件區塊中, 建立一個新變數叫做[color](https://docs.code.org/applab/color/), 然後使用[getProperty](https://docs.code.org/applab/getProperty/)區塊指定一個值給它.
* 提示: 你將會需要使用"value"屬性來取得使用者的選擇.
* 使用[setProperty](https://docs.code.org/applab/getProperty/)區塊來設定"screen1"的"background-color"屬性到你的變數[color](https://docs.code.org/applab/color/)中.
* 提示: 確認你輸入[color](https://docs.code.org/applab/color/)時, 不含有引號.
* 挑一個顏色並按下”Set Color”來測試你的程式.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/6)

### **Student Instructions**

學生操作指示

Predict Level (See contained level for markdown)

# **Using the Circuit Playground**

You can also use getters and setters with the Circuit Playground. For this level, you'll need to plug in your board.

Read through the code and predict what the program will do, then hit "Run" to test your prediction.

你也可以將getters和setters用在Circuit Playground上. 現在,你將會需要插入你的板子.

仔細閱讀程式碼並且預測程式如何運作, 然後點擊 "Run"來驗證你的預測.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/7)

### **Student Instructions**

學生操作指示

# **Using the Circuit Playground**

Here's the same code you saw in the last level, but this time there's another dropdown menu to control the buzzer. You'll have to add the code to make it work.

這裡的程式碼跟你在上個單元看到的一樣, 但是這次用另一個下拉式選單來控制蜂鳴器. 你將會需要增加程式碼來讓它運作.

# **Do This**

動手做

* Using the LED code as a model, add new code inside the buttonL event that will play a long or short buzz according to the chosen value property of the *buzzer\_input* dropdown menu.
* Add new code that will turn the buzzer off when the right button is pressed.

(Hint: the event is already in the code.)

* 用LED程式碼當作範本, 新增程式碼到buttonL事件中, 將會讓蜂鳴器依據*buzzer\_input*下拉式選單的value屬性, 發出長或短的嗶聲.
* 新增程式碼讓蜂鳴器在右按鈕被按下時關閉.

(提示: 此事件已經在程式碼中了.)

* Sliders

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/8)

### **Student Instructions**

學生操作指示

# **Sliders**

Last time, your users were only able to choose between a short and long buzz. To give the user control over the exact duration of the buzzer, you'll want to use a slider.

This slider lets users choose a value between 100 and 1000 for the duration of the buzz, but there's something not quite right about how it's working.

上一次使用者只能讓蜂鳴器發出長或短的嗶聲. 如果要讓使用者能精確地控制發聲時間長短, 你將需要使用到slider元件.

Slider讓使用者將蜂鳴器的發聲時間長短設定為100到1000之間, 但有些東西不太正確因此這隻程式無法正確運作.

# **Do This**

動手做

* Look over the code and find the bug that's keeping the duration from being set correctly.
  + (Hint: Hover over the yellow triangles for clues as to what might be wrong with the code.)
* 仔細閱讀程式碼,找出讓發聲時間控制不正確的原因.

(提示: 將滑鼠游標停留在黃色三角形, 那裏會有程式碼可能出錯的線索)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/9)

### **Student Instructions**

# **Frequency**

This buzzer controller has been expanded to control frequency as well, but the code isn't finished yet.

蜂鳴控制器也已經被用來控制頻率, 但程式碼還沒完成.

# **Do This**

動手做

* Using the duration variable as a model, create a frequency variable that gets the value from the *frequency* slider.
* Modify the line of code that makes the buzzer sound so that it uses the frequency variable rather than the default frequency of 500.
* Run your code, then change the frequency and the duration to hear the difference in the buzzer. (Don't forget to hit the button after you make a change!)
* 用duration變數當作範本,建立一個frequency變數,從*frequency slider*取得設定值.
* 修改那行讓蜂鳴器發聲的程式碼, 讓它用frequency變數的值而非預設的頻率值500.
* 執行你的程式碼, 然後變更發聲頻率和時間長度, 聽聽看蜂鳴器發出的聲音有什麼不同. (不要忘記變更後要點擊"Run"按鈕)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/10)

### **Student Instructions**

學生操作指示

Predict Level

# **Interval**

時間間隔

Now we're using the slider to affect the behavior of the LED. Just as we could change the values for the frequency and duration of the buzzer, we can change the value for the blink or pulse interval of the LED.

現在我們使用slider來影響LED的行為模式. 就像是我們能變更蜂鳴器的發聲頻率和時間長度一樣, 我們可以控制LED的閃爍頻率和燈亮的時間.

# **Do This**

動手做

* Read the code and predict what the program will do, then hit "Run" to test your prediction.
* 仔細閱讀程式碼, 並且預測程式會如何運作, 然後點擊"Run"來驗證你的預測.
* Event Types
* [Student Overview](https://curriculum.code.org/csd-18/unit6/6/#level-expando-6-11-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/11)

* Change

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/12)

### **Student Instructions**

學生操作指示

# **Change**

改變

In the last level, the program only got the interval value from the slider when you pressed the left button. Sometimes, though, you'll want to change the interval of the LED as soon as the user chooses a new value. For that, you'll need the "change" event.

在上一堂課，當你按下左按鈕時程式才會從 slider 取得時間間隔值。不過，有時候你可能想在使用者一調整 slider 的時候就改變 LED 的時間間隔值，這時候你就得靠 change 事件才能辦到。

# **Do This**

動手做

* Look at the first event block in the program and find where it says "change".
* Inside the event block, get the interval value from the slider and use it to make the LED blink at that new rate.
* Run your program, moving the slider to see whether the LED speed changes automatically.
* 看一下程式中的第一個事件區塊，找到顯示 change 的地方。
* 在事件區塊內，取得 slider 的時間間隔值，並且讓 LED 以新的時間間隔值持續閃爍。
* 執行程式後試著移動 Slider，看看 LED 閃爍的速度是否跟著改變。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/13)

### **Student Instructions**

學生操作指示

# **Change**

改變

This program has events that detect three different changes: the toggle switch, the interval slider, and the duration slider. Because there are so many different things that cause the LED and buzzer to reset, it uses functions to organize the code.

這隻程式用事件偵測三種不同的改變，包括切換開關以及用Slider調整時間間隔和時間長度。因為有許多不同的情況會導致LED和蜂鳴器被重置，所以這隻程式用函式來組織相關的程式碼。

# **Do This**

動手做

* Read the updateLED function to see what it does.
* Add code to updateBuzzer to make the buzzer only turn on when the toggle switch is open.
* 閱讀 updateLED 函式以便了解它的功能。
* 在 updateBuzzer 函式加入程式碼，使得蜂鳴器只有在切換開關是開啟狀態時，才會被打開。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/14)

### **Student Instructions**

學生操作指示

# **LED and Buzzer Controller**

LED與蜂鳴器控制器

This program operates the LED and the buzzer, but it has some problems.

這隻程式讓 LED 和蜂鳴器運作，不過它有點問題。

# **Do This**

動手做

* Read the program to understand how it is supposed to work.
* Find and correct the bugs so that the program works properly.
* Motorcycle
* 閱讀程式以便了解它應當如何運作。
* 找出程式中的錯誤並加以更正，讓程式能夠正確地執行。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/15)

### **Student Instructions**

學生操作指示

# **Movement with Properties**

利用屬性來移動

Earlier we used [setProperty](https://docs.code.org/applab/setProperty/) to change the x and y position of elements on the screen, but we could only move them to either a single specific location, or a random location. By using [getProperty](https://docs.code.org/applab/getProperty/) to find out an element's current position, we can use the **counter pattern** from Unit 3 to update that position.

之前我們用 setProperty 改變了元件在螢幕上的 X 和 Y 位置，不過我們只能把它們移到單一特定的位置或是隨機的位置。藉由使用 getProperty 找出元件目前的位置，我們可以用第三單元介紹的計數器來更新位置。

# **Do This**

動手做

* Read the program and discuss with your partner how the left button makes the motorcycle go up.
* Using the working left button as an example, program the right button to make the motorcycle go down.
* 閱讀程式並與你的夥伴討論，怎麼利用左邊按鈕讓摩托車向上移動。
* 以上述的左邊按鈕控制程式當作範例，用程式控制右邊按鈕，讓摩托車向下移動。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/puzzle/16)

### **Student Instructions**

學生操作指示

# **Add Your Own Elements**

加入你自己的元件

Now it's time to add some elements of your own. The motorcycle should be able to move up and down right now, but how can you control its speed?

現在是時候加入一些你自己的元件了。目前摩托車能夠向上與向下移動，不過你要怎麼控制它的速度呢?

# **Do This**

動手做

* Add a slider in design mode.
* Make sure you set the min and max values of the slider in design mode.
* Create an event that will trigger when the slider value is changed.
* Add code to get the slider value and store it into the [speed](https://docs.code.org/applab/speed/) variable when the event is triggered.
* Levels
* [Extra](https://curriculum.code.org/csd-18/unit6/6/#level-expando-6-17)
* (click tabs to see student view)
* 在設計模式下加入一個 slider。
* 確認你在設計模式中設定了 slider 的最小值與最大值。
* 建立一個事件，當 slider 的設定值改變時，這個事件會被觸發。
* 加入程式碼以便取得 slider 的設定值，當事件被觸發時，將設定值存放在 speed 變數中。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/6/extras)

### **Student Instructions**

學生操作指示

# **Challenge**

挑戰

Use this program to experiment some more with getting and setting properties. Some ideas you might try include:

* Use the toggle switch to change the motorcycle's movement to go up and down.
* Use a dropdown with color names to change the motorcycle's color
* Make the buzzer or LED turn on when the motorcycle reaches the top or bottom of the screen.

你可以用這隻程式進行更多讀寫屬性的實驗，有一些點子或許你可以試試看，包括:

* 使用切換開關來改變摩托車的移動方向，使它向上走或向下走。
* 使用下拉式清單來改變摩托車的顏色。
* 當摩托車到達螢幕上方或下方時，開啟蜂鳴器或 LED。

## **Wrap Up (10 min)**

總結 (10 min)

Teaching Tip

教學建議

For this prompt, you can "jigsaw" the answers by assigning different students or different groups each a type of input and answering questions on each type. The different answers can then be combined for a class reference on input types.

關於以下的提示，你可以讓學生以個人或分組的方式，回答不同輸入類型的問題。完成後將這些不同輸入類型的答案合併在一起，成為課堂的參考資料。

**Journal Prompt:** So far, you've seen several different types of input, some from the screen, and some from the circuit playground. Choose one type of input and answer the following questions about it.

**期刊提示:** 目前為止，你看過好幾種不同的輸入類型，有些從螢幕，有些從 circuit playground。選擇一種輸入類型並回答下列有關的問題:

1. What code do you need to get information from this input?
2. What's one example of when you would want to use this input?
3. What's an example of when you **wouldn't** want to use this in/put?
4. 你需要怎樣的程式碼，才能從這種輸入類型取得資訊?
5. 什麼時候你會使用這種輸入類型，可以舉一個例子嗎?
6. 什麼時候你不會使用這種輸入類型，可以舉一個例子嗎?

# **Lesson 7: Analog Input**

第7課：類比訊號輸入

#### **App Lab | Maker Toolkit**

App Lab| 創客工具組

## **Overview**

課程概述

In this lesson, students explore how the three analog sensors (sound, light, and temperature) can be used to write programs that respond to changes in the environment. The use of these sensors marks a transition in terms of how users interact with a program. By using sensors as an input, the user of an app doesn't have to directly interact with it at all, or may interact without actually realizing they are doing so.

在本課程中，學生將探索如何使用三個類比訊號感測器（聲音感測，光感測和溫度感測）來編寫回應環境變化的程式。 藉著使用這些感測器偵測環境的變化了解使用者如何和程式互動。使用這些感測器偵測值來做為應用程式的輸入，應用程式的使用者完全不用直接手動輸入偵測值，也不需瞭接這是怎麼辦到的 。

**Purpose**

This lesson builds on the previous by introducing analog sensors as a new form of input. These sensors all perform analog-to-digital conversion, allowing programs to sense things as represented by a 10 bit number (0-1023).

課程目的

本課程以前面的課程為基礎，來介紹感測器作為一種新的輸入形式。 這些感測器用來執行類比訊號與數位訊號轉換，讓程式自動感測環境變化並轉換成 10 位元的數字 ( 0 - 1023 ) 。

## **Agenda**

* [Warm Up (10 min)](https://curriculum.code.org/csd-18/unit6/7/#analog-input0)
* [Analog and Digital](https://curriculum.code.org/csd-18/unit6/7/#analog-input1)
* [Activity (40 min)](https://curriculum.code.org/csd-18/unit6/7/#analog-input2)
* [Analog Inputs](https://curriculum.code.org/csd-18/unit6/7/#analog-input3)
* [Wrap Up (5 min)](https://curriculum.code.org/csd-18/unit6/7/#analog-input4)
* [Sensors All Around](https://curriculum.code.org/csd-18/unit6/7/#analog-input5)

# **課程流程**

* 暖身活動 (10分鐘)
  + 類比訊號與數位訊號
* 課程活動 ( 40 分鐘)
  + 類比訊號輸入
* 總結 ( 5 分鐘 )
  + 各種不同的感測器

請想一想在日常生活中有那些感測器會運用這些類比訊號轉換成轉換成數位訊提供智慧型裝置運用? 試舉出例子。

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/1/)**

## **Objectives**

### **Students will be able to:**

* Develop programs that respond to analog input
* Scale a range of numbers to meet a specific need
* Represent a sensor value in a variety of ways

**課程目標**

**學生將具備以下能力：**

* 建立回應類比訊號的程式
* 縮放數值區間以符合需求
* 以不同方式呈現感測器的回覆值

**Preparation**

* Cue up the [Difference between Analog and Digital Signals - Video](https://www.youtube.com/watch?v=RQc_frX50BA)
* If your room isn't very bright, it's useful to have some flashlights or other light sources on hand for testing the light sensor

**課前準備**

* 影片參考: [Difference between Analog and Digital Signals - Video](https://www.youtube.com/watch?v=RQc_frX50BA)
* 如果您的房間不是很明亮，可以準備手電筒或其他光源來測試光感測器

## **Links**

**Heads Up!** Please make a copy of any documents you plan to share with students.

相關連結

注意！請將所有需要給學生的文件先印出來！

### **For the Students**

* [Difference between Analog and Digital Signals](https://www.youtube.com/watch?v=RQc_frX50BA) - Video

給學生的資料

* 影片參考 : [Difference between Analog and Digital Signals](https://www.youtube.com/watch?v=RQc_frX50BA) - Video

## **Vocabulary**

* **Analog** - Any continuously changing signal that is not restricted to finite set of values. For example, the wave forms of spoken words are an analog signal.
* **Digital** - Data or signals represented by a finite number of values. Analog signals (which can have infinite values) must be converted to digital in order to be computed with.

**相關字彙**

* 類比訊號 - 任何連續變化的訊號, 沒有固定限定數值

例如語音訊號就是屬於類比訊號

* 數位訊號 - 通常有一組固定數值或資料的變化，類比訊號必須經過轉換成數位訊號才能被電腦所計算或處理。

## **Introduced Code**

* [soundSensor](https://docs.code.org/applab/soundSensor/)
* [soundSensor.value](https://docs.code.org/applab/value/)
* [soundSensor.setScale(min, max)](https://docs.code.org/applab/setScale/)
* [lightSensor](https://docs.code.org/applab/lightSensor/)
* [tempSensor](https://docs.code.org/applab/tempSensor/)

## **Support**

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

### **[Report a Bug](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%207%20curriculum.code.org/csd-18/unit6/7/)**

# **Teaching Guide**

## **Warm Up (10 min)**

### **Analog and Digital**

Discussion Goal

The goal here is to get students thinking about how an natural (or analog) value can be converted into a computable format. Encourage students to consider what the upper and lower bounds of each value might be (for example, is there a minimum or maximum amount of sound?).

**Think, Pair, Share:** How would you quantify each of your five senses (touch, taste, sight, smell, and hearing). Think back to the previous unit and the various ways you came up with to encode data for a computer.

**Video:** Today we're going to write programs that have "senses" of their own! Show [Difference between Analog and Digital Signals - Video](https://www.youtube.com/watch?v=RQc_frX50BA)

## **暖身活動 ( 10 分鐘)**

類比訊號與數位訊號

討論目標

本課程的目地在讓學生思考自然界上的類比訊號如何轉換成可被電腦計算及處理的數位訊號。鼓勵學生思考數值的最大值與最小值 ( 例如，聲音訊號是否有最大值與最小值 ? )

分組討論分享 ； 討論並分享如何量化我們五種知覺 ( 觸覺, 味覺, 視覺，嗅覺，聽覺 )。想一想並應用前面所學的單元將這五種知覺轉換成電腦可處理的數位數值。

影片分享: 今天我們將會寫有自我知覺的程式

[Difference between Analog and Digital Signals - Video](https://www.youtube.com/watch?v=RQc_frX50BA)

## **Activity (40 min)**

課程活動 ( 40 分鐘 )

### **Analog Inputs**

**Distribute:** Pass out Circuit Playgrounds.

**Transition:** Send students to Code Studio. Let them know that today they will be experimenting with some sensors that detect analog signals and convert them to digital values that can be used by the computer.

類比訊號輸入

分發: 分發 Circuit Playgrounds

接下來 : 讓學生進入 Code Studio，讓學生瞭解本課程將會實驗類比感測器如何轉換成數值以運用在電腦計算。

### **Code Studio levels**

* Lesson Overview
* [Student Overview](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/1)

* Sensor Experiment
* [Student Overview](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/2)

# **Sensor Experiment**

Run the program to the right and experiment with your board. Try interacting with your board in many different ways to figure out what each sensor might be. As you experiment with each sensor, discuss with a neighbor:

* What does this sensor measure?
* What is its maximum value?
* What is its minimum value?
* Analog Sensors
* [Student Overview](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-3-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/3)

* Reading Sensors

# **Sensor Experiment**

感測器實驗

在右邊執行你的程式，並用你的實驗板來實驗。試著改變環境來了解你的實驗板上的感測器蕙如反應，與你的左右同學討論下面事項:

* 這個感測器是偵測甚麼 ?
* 感測器最大值是甚麼 ?
* 感測器的最小值是甚麼 ?
* 類比感測器
* What does this sensor measure?
* What is its maximum value?
* What is its minimum value?
* Analog Sensors
* [Student Overview](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-3-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/3)

* Reading Sensors

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/4)

### **Student Instructions**

# **Reading the Sound Sensor**

從聲音感測器來讀資料

The [soundSensor.value](https://docs.code.org/applab/value/) block allows you to get the current sound sensor reading

[oundSensor.value](https://docs.code.org/applab/value/) 可以讓你從聲音感測器讀出資料

# **Do This**

實作

In this app we've already added a text label with the id "sound\_value". Using the [soundSensor.value](https://docs.code.org/applab/value/) block, display the value of the sound sensor.

在這個 app 你加了一個標示為 “Sound\_value” 的字串, 現在利用 [soundSensor.value](https://docs.code.org/applab/value/) 區塊來顯示從聲音感測器讀出來的數值。

*Tip: You can drag the* [*soundSensor.value*](https://docs.code.org/applab/value/) *block directly into any other block where you could type a value instead.*

*秘訣 : 你也可以直接用拖拉的方式將* [*soundSensor.value*](https://docs.code.org/applab/value/) *區塊拉到任何你需要這個數值的地方。*

**

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/5)

### **Student Instructions**

# **Light Sensor Updates**

This app is very similar to the last, but we've added a button with the ID "update\_button". Write a program that displays the current value of the light sensor every time the update button is clicked.

光感測器的使用

和目前的程式類似，但在這裡我們加一個 ID 是 "update\_button” 的按鍵，試著寫一個程式當使用者按 “update\_button” 時，程式會顯示現在光感測器的值。

# **Do This**

實作

Use an event handler to update the text of "light\_value" using the lightSensor.value block.

用一個事件處理器 ( Event Handler ) 並使用 lightSensor.value 區塊來更新 “light\_value” 的文字。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/6)

### **Student Instructions**

# **Make a Thermometer App**

寫一個溫度計 (Thermometer) App

不像光感測器和聲音感測器只回報原始數值， 溫度感測器的 tempSensor.value 中 value 的屬性可以指定並轉換原始數值成攝氏(tempSensor.C)或華氏(tempSensor.F) 的溫度值

While the light and sound sensors have just a raw value with the [value](https://docs.code.org/applab/value/) property, the temperature sensor is a little bit smarter. Instead of tempSensor.value, there are two properties tempSensor.F and tempSensor.C which convert the raw input value to either Fahrenheit or Celsius.

# **Do This**

實作

Make a simple thermometer app that reads in values from the temperature sensor. Your app can display the temperature in either Fahrenheit or Celsius (or both!)

* Using Design Mode, create a button for updating and a label to display the temperature
* Add an event handler to respond to your update button being clicked
* Add code to your event handler to read the temperature sensor and display the value on screen

寫一個溫度計 (Thermometer)App從溫度感測器讀出值。你的程式可以顯示攝氏或華氏溫度 ( 或是顯示兩種溫度 )

* 先使用設計模式 ( Design Mode) ，先設計一個按鍵來更新溫度和一個字串來顯示溫度。
* 試著加入一個事件處理器 ( Event Handler) 來處理當使用者按了按建的事件

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/7)

### **Student Instructions**

# **Make a Prediction**

預測結果

Read the code in this program and make a prediction below. What do you think this program will do when you run it? How is the user intended to interact with the board? You may need to blow onto the board to see the full range of this app.

讀程式碼來預測下面結果。你覺得當你執行程式的結果為何? 使用者如何和實驗版互動 ? 你可能需要在電路板上觀察執行結果 。

* Polling Events
* [Student Overview](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-8-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/8)

* Sensor Events

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/9)

### **Student Instructions**

# **Data Event**

資料事件( Data Event )

Each of these analog sensors emits a special event called "data" which occurs every time new data comes in from the sensor. Using this event, we can write programs that continually update.

每當類比感測器上偵測有新的資料 ( Data ) 時就會出發一個事件( Event ) ，我們可以利用這個觸發事件來寫一個可以有連續資料更新輸入的程式。

# **Do This**

實作

This is the first program you worked on in this lesson, let's see if we can make it continually update the sound reading using the "data" event.

* Create an event handler to respond to sound sensor's "data" event
* Move the existing code into your event handler

這是這堂課的第一個程式，讓我們來試試是否我們可以利用資料事件 ( Data Event) 來觸發聲音資料的連續輸入。

* 建立一個事件處理器 ( Event Handler) 來處理感測器的資料事件 ( Data Event)
* 把現在我們寫好的程式碼搬進事件處理器中

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/10)

### **Student Instructions**

# **Change Event**

事件觸發 (Change Event)

The data event is fine if you don't mind your code running constantly, but sometimes that's not the most efficient solution to your problem. The sensor "change" event only fires if the sensor value has changed since the last reading, which can make sure your program isn't running code when it's not necessary.

假如你不介意你的程式是一直不斷地處理理資料，利用資料事件來當作資料的持續輸入還算是可以的方式，但是有時候這不算是一個有效率的方式。感測器的事件觸發是從你上次讀取資料以來的任何改變都會發生，有時候這在程式處理上並不是有效率的方式 。

# **Do This**

實作

Using the provided event handler, write a program that buzzes the buzzer and updates "temp\_value" whenever the temperature sensor reading changes. Place your thumb over the temperature sensor to get it to change and see how your program responds.

現在利用你的事件處理器，寫一段程式來讀取溫度感測器，在每次有溫度變化時讀出 “temp\_value並讓蜂鳴器發出聲音。試試看把你的手指放在感測器上讓溫度改變，看看你的程式如何運作。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/11)

### **Student Instructions**

# **Change Threshold**

調整條件值

By default a sensor's "change" event fires every time the sensor value changes at all, even by one. The light and sound sensors are constantly fluctuating, so you probably want the "change" event to be a little less responsive. The threshold property allows you to set how much a sensor should change before considering it a "change" event - so setting soundSensor.threshold = 100 would cause the sound sensor's "change" event only to trigger if its value increased or decreased by at least 100.

調整條件值

在原始狀態下感測器任何資料的改變都會觸發資料改變的事件，甚至是任何細微的改變。以光和聲音感測器來說，偵測到的值是不斷地改變的，在這種情況下，或許你需要這個值的改變可以不要那麼敏感。這時候你可以考慮設定一個條件值來調整資料改變事件發生的頻率。例如設 soundSendor.thread =100 只有聲音感測器感測到聲音降到 ( 或升到) 100 時才會使得聲音感測器觸發資料改變的事件。

# **Do This**

實作ˋ

This app changes the background color of the screen whenever the sound sensor value changes, but we want to only respond to big changes in volume (like a loud noise). Use the soundSensor.threshold block to fix it.

* Add a soundSensor.threshold block before the event handler
* Test out which how different threshold values impact the "change" event
* Choose a threshold value that makes the screen background change only when a loud noise is sensed.

*Tip: Your threshold depends on how much ambient noise there is in the room, and how much it changes. A threshold that works in a quiet room may not work in a louder classroom.*

* Changing Sensor Scale
* [Student Overview](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-12-sfmd)

寫一段程式來反應當偵測到大音量的改變時，改變背景顏色。試著使用 soundSensor.threshold 區塊來完成。

* 在事件處理器前面加上 soundSensor.threshold 區塊。
* 用不同的條件值來放入 soundSensor.threshold 試試看這個值怎麼影響資料改變的事件。
* 選定一個值可以反應大音量改變, 用這個值來偵測大音量並改變背景顏色。

秘訣﹔你選用的條件值和你所處的環境噪音大小有關， 在安靜的環境下的條件值並不適用在一個吵雜的環境。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/12)

* Sensors to Colors
* [13](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-13)
* [14](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-14)
* [15](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-15)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/13)

### **Student Instructions**

# **Displaying Sensor Readings in Color**

In the earlier prediction level you saw how the [setScale()](https://docs.code.org/applab/setScale/) method can convert the full range of sensor readings (0-1023) into a range that can be used for RGB values (0-255). Let's build on that idea to make an app that can communicate the current value of all three analog sensors with a single color.

用顏色來表示從感測器讀出的數值

從前面的預測值我們看到 [setScale()](https://docs.code.org/applab/setScale/) 這個函式( method) 如何將感測器回報的 10 位元數值 (0-1023) 轉換成8位元的 RGB 數值 ( 0-255)，讓我們來寫一個程式將三個感測器所回報的值都用顏色來表示。

# **Do This**

To start, we just need to set the sensor scales correctly - don't worry about actually setting the color of the background yet.

* Using the sensor.setScale() block, set each of light and sound sensors to the right scale for an RGB color channel.
* Create a variable for each of the three colors and assign each one to the value of a different sensor.
* Use [console.log()](https://docs.code.org/gamelab/console.log/) to test that your variables are in the correct range.

*Hint: Order matters. You need to set the scale of a sensor before checking its value*

*Hint: Remember that the temperature sensor can be read in both Celsius and Fahrenheit - either way it should already give you a value that's within the necessary range for RGB*

實作

一開始， 我們只需要正確設定感測器的數字區間，還先不用煩惱怎麼設定背景顏色。

* 利用 sensor.setScale()區塊，給光感測器及聲音感測器設定正確數值區間反應 RGB 背景顏色改變。
* 設定一個變數設定 RGB 三色值給每一個感測器。
* 使用 [console.log()](https://docs.code.org/gamelab/console.log/) 來測試你的變數值是否在正確的數值區間 。

秘訣 :

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/14)

### **Student Instructions**

給學生的指導

# **Putting Color on the Screen**

Now that your sensor data is in a scale that works for RGB, you can go about setting the screen color.

在螢幕上顯示顏色

現在你已經可以讓你的感測器的值對應在 RGB三色值，現在可以在螢幕上顯示你設定的顏色了。

**Do This**

* Add a [setProperty()](https://docs.code.org/applab/setProperty/) block and set it to change the screen's background color.
* Add an [rgb()](https://docs.code.org/applab/rgb/) block to the last parameter of [setProperty()](https://docs.code.org/applab/setProperty/).
* Use the three color variables you created as inputs to the [rgb()](https://docs.code.org/applab/rgb/) block.

實作

* 加上 [setProperty()](https://docs.code.org/applab/setProperty/) 區塊並設定參數來改變背景顏色。
* 使用 [rgb()](https://docs.code.org/applab/rgb/) 區塊在 [setProperty()](https://docs.code.org/applab/setProperty/)區塊內的最後一個參數來當背景色的設定。
* 使用你調出來的顏色來當作 [rgb()](https://docs.code.org/applab/rgb/) 區塊的輸入參數。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/puzzle/15)

### **Student Instructions**

給學生的指導

# **Continuous Updates**

This sensor-to-color app isn't super useful if it only sets the color once when you start the app. You can use the sensors' "data" event to make your app continually update the background color every time the value changes.

Move your [setProperty()](https://docs.code.org/applab/setProperty/) block into a an event handler for one of the sensors to make the app update continually.

如果你的感測器變換顏色程式只是在一開始讀感測器並設定一個顏色, 這樣並不算好用。你可以使用感測器的資料事件 ( Dat Event) 來驅動你的程式可以持續接收輸入並根據資料變化改變背景顏色。

搬移你的[setProperty()](https://docs.code.org/applab/setProperty/)區塊到所有的事件處理器 ( Event Handler ) 中來處理並執行每次的感測器資料變化的事件 ( Data event) 。

* Levels
* [Extra](https://curriculum.code.org/csd-18/unit6/7/#level-expando-7-16)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/7/extras)

### **Student Instructions**

給學生的指導

# **Challenge**

挑戰

*Check with your teacher before pursuing this challenge*

Now that you've practiced taking input from the various analog sensors and changing their scale to meet your needs, can you come up with an interesting use for sensor data? It's easy to display the raw numbers coming out of a sensor, but far more interesting to present that data in different ways.

當你要開始進行挑戰時先與講師確定一下挑戰的內容 。

現在你已經練習了從不同的感測器讀出值，並且可以縮放這些讀出值到你需要的數值區間，現在你是不是已經有一些有趣的想法來應用這些數值 ? 你當然可以直接顯示這些從感測器讀出值， 但更有趣的挑戰是你如何用不同的方式來呈現這些值。

Consider the following ways you might communicate sensor data:

* Use the board outputs
* Change the width, height, or position of a UI element
* Use conditionals to display different images based on sensor values

想一想如何用下面的方式來呈現從感測器讀出來的值

* 利用板子上的輸出
* 利用 UI 原件來改變寬，高或位置
* 根據感測器讀出值設定不同的條件來顯示不同的圖示

## **Wrap Up (5 min)**

### **Sensors All Around**

各種不同的感測器

**Journal:** Considering all of the computing devices that you interact with on a regular basis, identify as many potential analog sensors as you can. Where do your computing devices take a continuously changing signal and convert it into digital data?

筆記 : 請想一想在日常生活中有那些感測器會運用這些類比訊號轉換成轉換成數位訊提供智慧型裝置運用? 試舉出例子。

## **Standards Alignment**

#### **[View full course alignment](https://curriculum.code.org/csd-18/standards/)**

#### **CSTA K-12 Computer Science Standards**

**AP** - Algorithms & Programming

* **2-AP-11** - Create clearly named variables that represent different data types and perform operations on their values.
* **2-AP-12** - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
* **2-AP-13** - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
* **2-AP-16** - Incorporate existing code, media, and libraries into original programs, and give attribution.
* **2-AP-17** - Systematically test and refine programs using a range of test cases.
* **2-AP-19** - Document programs in order to make them easier to follow, test, and debug.

**CS** - Computing Systems

* **2-CS-01** - Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices.
* **2-CS-02** - Design projects that combine hardware and software components to collect and exchange data.
* **2-CS-03** - Systematically identify and fix problems with computing devices and their components.

# **Lesson 8: The Program Design Process**

第8課：程式設計流程

#### **App Lab | Maker Toolkit**

App Lab| 創客工具組

## **Overview**

課程概述

This lesson introduces students to the process they will use to design programs of their own throughout this unit. This process is centered around a project guide which asks students to sketch out their screens, identify elements of the Circuit Playground to be used, define variables, and describe events before they begin programming. This process is similar to the Game Design Process that we used in Unit 3. In this lesson students begin by playing a tug o' war style game where the code is hidden. They discuss what they think the board components, events, and variables would need to be to make the program. They are then given a completed project guide which shows one way to implement the project. Students are then walked through this process through a series of levels. At the end of the lesson students have an opportunity to make improvements to the program to make it their own.

本課程將向學生介紹如何在這整個單元中設計出自己程式的流程。這個流程是以專案指南為中心，在開始寫程式之前，要求學生勾勒出他們的構思、找出要使用的Circuit Playground元素、定義出變數、並描述出事件。這個流程與我們在第3單元中學到的遊戲設計流程有點像。在本課程中，學生們首先玩一個需要寫程式的類似拉鋸比賽(笑臉賽跑)的遊戲。他們將討論出他們認為需要用到程式的實驗板組件、事件和函數。然後他們會拿到有一份完整描述實施專案方法的專案指南。隨後學生便可以據此來執行一系列關卡來走完這個流程。在這課程結束時，學生有機會對程式進行改進，使其成為自己的遊戲。

## **Purpose**

課程目的

This lesson gives students to practice developing a larger scale program in order to prepare them to do so independently. While previous lessons have focused on building skills around using specific elements of the Circuit Playground and related programming concepts, this lesson focuses on combining everything learned so far into a more complex program. The lesson heavily scaffolds the software development process by providing students a completed project guide, providing starter code, and walking students through its implementation. In the subsequent lessons students will need to complete a greater portion of this guide independently, and for the final project they will follow this process largely independently.

本課程的主要目的是讓學生藉由練習開發更大規模程式的機會，來學習獨立開發程式的能力。雖然之前的課程側重於使用Circuit Playground的特定元素和相關的程式概念，但本課程的重點則是將之前學到的所有內容應用到一個更複雜的程式裡。本課程為了支援軟體開發流程，特別為學生提供完整的專案指南及啟始代碼，並引導學生完成實作。在隨後的課程中，學生將需要獨立完成本指南剩下的一大部分，對於最終專案，學生們將在會更獨立地遵循此流程。

## **Agenda**

* [Warm Up (15 min)](https://curriculum.code.org/csd-18/unit6/8/#the-program-design-process0)
  + [Play Emoji Race](https://curriculum.code.org/csd-18/unit6/8/#the-program-design-process1)
  + [Stop: Review Project Guide](https://curriculum.code.org/csd-18/unit6/8/#the-program-design-process2)
* [Activity (45-60 min)](https://curriculum.code.org/csd-18/unit6/8/#the-program-design-process3)
  + [Implement Project Guide](https://curriculum.code.org/csd-18/unit6/8/#the-program-design-process4)
* [Wrap Up (20 min)](https://curriculum.code.org/csd-18/unit6/8/#the-program-design-process5)
  + [Make It Your Own](https://curriculum.code.org/csd-18/unit6/8/#the-program-design-process6)

# 課程流程

* 暖身活動 (15分鐘)
* 玩笑臉賽跑遊戲
* 停下來：檢視專案指南
* 活動內容 (40-60分鐘)
* 專案指南的實作
* 總結 (20 分鐘)
* 自己的程式

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/8/puzzle/1/)**

## **Objectives**

### **Students will be able to:**

* Implement different features of a program by following a structured project guide
* Develop a program that responds to events from a hardware input
* Create a function that uses parameters to generalize behavior

課程目標

學生將具備以下能力：

- 遵循結構化的專案指南來寫出有多功能的程式

- 開發一個可以回應硬體輸入事件的程式  
- 寫出一個可以根據不同參數來運作的函數

## **Preparation**

* Provide students with copies of the [Emoji Race - Project Guide](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/)

課前準備

- 為學生們列印''笑臉賽跑遊戲的專案指南''

## **Links**

**Heads Up!** Please make a copy of any documents you plan to share with students.

相關連結

注意！請將所有需要給學生的文件先印出來！

### **For the Students**

* [Emoji Race](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/) - Project Guide Make a Copy
  + [PDF](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/copy)

給學生的資料

- 笑臉賽跑遊戲 專案指南 - 列印

- PDF

- Microsoft Word

- Google Docs

## **Support**

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

### **[Report a Bug](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%208%20curriculum.code.org/csd-18/unit6/8/)**

課外支援

討論區

問題回報

# **Teaching Guide**

## **Warm Up (15 min)**

教學指南

暖身活動（15分鐘）

### **Play Emoji Race**

玩笑臉賽跑遊戲

**Group:** Place students in pairs

**Demo:** Show students the Emoji Race demo, available at the beginning of this lesson.

**Transition:** Send students online to try out the game on their own. Each pair should play the game and follow the instructions which ask them to list the board components, events, and variables they think are necessary to create this game.

分組：學生每兩人一組

展示：在本課程一開頭，先向學生講解一下笑臉賽跑遊戲。

接下來：讓學生在線上自己嘗試遊戲。每一對學生都應該玩遊戲並按照說明列出他們認為編寫這遊戲所需的實驗板組件、事件及函數。

Teaching Tip

**Keeping Focus:** Students can easily get distracted by the fun of playing the game. Let them play for a while but eventually encourage them to follow the on-screen instructions and make a list of the board components, events, and variables that would be necessary to create the game.

教學建議

注意：學生很容易因玩遊戲的樂趣而分心。可以先讓他們玩一會兒，但最終還是要鼓勵他們按照畫面上的說明進行操作，並列出編寫這遊戲所需的電路板組件、事件及函數。

### **Stop: Review Project Guide**

停下來：檢視專案指南

**Discuss:** Students should have individually created a list of board components, events, and functions they would need to make the game they played. Ask students to share their lists with a neighbor before discussing as a class.

討論：學生應該已經可以個別列出他們製作遊戲所需的電路板組件，事件及函數。要求學生在課堂討論之前與鄰座同學分享他們的想法。

Discussion Goal

**Running the Conversation:** You can write "Board", "Events", and "Functions" on the board and record their ideas below each. Ask students to justify their decisions but don't feel the need to settle on one right answer.

討論目標

對話活動：您可以在白板上寫下“電路板組件”，“事件”和“函數”，並在每項下面記錄學生們的想法。然後要求學生為自己的想法加以說明，但不一定要找到一個正確的答案。

Teaching Tip

**Sharing the Project Guide:** Students are not actually writing on the project guide in this lesson. You can give each student their own copy for reference but you might also choose to print one copy per pair, share digital copies, or just display the guide on the projector. So long as it is available for reference any approach will work fine.

教學建議

分享專案指南：因為學生實際上並沒有在本課程中編寫專案指南。所以您可以為每個學生提供各自的專案指南以供參考，也可以選擇每組給一份，或是大家都用電子檔，或是只在投影機上顯示專案指南。只要它可供大家參考，任何方法都沒問題。

**Distribute:** Give each student or pair of students a copy of the [Emoji Race - Project Guide](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/)

**分發**：給每個學生或每組學生一份笑臉賽跑遊戲 - 專案指南

**Prompt:** Compare the list of things you thought would be needed to the ones on this project guide. Do you notice any differences?

**Discuss:** As a class compare the list you had on the board to the list of board components, events, and functions on the project guide. Note the similarities. Where there are differences try to understand why. Don't approach one set as "right" vs. "wrong" but just confirm both would be able to make the game students played.

提問：將您認為需要的內容(“電路板組件”，“事件”和“函數”) 與本專案指南中的內容進行比較。你注意到有什麼不同嗎？  
討論：在課堂中，將您在白板上的列表與專案指南中的電路板組件，事件和函數進行比較。標出注意相似之處。另外也找出哪裡有差異，並試著理解為什麼。不要嘗試去區分哪個是“正確”的，哪個是“錯誤”的，而只要確認兩者都能夠做出可讓學生玩的遊戲。

Remarks

There's usually lots of ways you can structure a program to get it to work the way you want. The important thing when writing complex or large programs is that you start with a plan. Today we're going to look at how we could implement this plan to build our own emoji race game. By the end of the lesson you'll not only have built your game, but you'll know how to change it and make it your own. Let's get going!

老師點評  
通常有很多方法可以編寫出一個程式，讓它按照你想要的方式來工作。然而編寫複雜或大型程式時，最重要的是從計劃開始。今天我們將看看如何實施這個計劃來構建我們自己的笑臉賽跑遊戲。在課程結束時，你不僅可以寫出遊戲，還可以知道如何修改遊戲並使其成為你自己的遊戲。讓我們開始吧！

## **Activity (45-60 min)**

### **Implement Project Guide**

Students are given starter code to establish the functions and event handlers needed for this project. These levels guide students through how to use the provided variables and what ought to occur within the event handlers. As students move through the levels point out how the project guide is being used. Though this project is highly scaffolded, the next lesson will ask students to develop a hardware-based program of their own design, so make sure to reinforce the connection between the planning that was done in the project guide and the programming levels.

活動內容 (45-60 分鐘)

專案指南的實作  
學生將獲得啟始代碼，用以建立此專案所需的函數和事件處理程序。這些關卡指導學生如何使用變數以及事件處理程序中應該包含什麼。學生可透過關卡來學習專案指南的使用方式。雖然此專案是有高度支援的，但下一課仍將要求學生開發自己設計與硬體有關的程式，因此請學生務必加強專案指南中的規劃與程式關卡之間關聯性的瞭解。

### **Code Studio levels**

* Lesson Overview
* [Student Overview](https://curriculum.code.org/csd-18/unit6/8/#level-expando-8-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/8/puzzle/1)

Code Studio 關卡

課程概述

給學生的概述

在Code Studio上查看

## **Overview**

This lesson introduces students to the process they will use to design programs of their own throughout this unit. This process is centered around a project guide which asks students to sketch out their screens, identify elements of the Circuit Playground to be used, define variables, and describe events before they begin programming. This process is similar to the Game Design Process that we used in Unit 3. In this lesson students begin by playing a tug o' war style game where the code is hidden. They discuss what they think the board components, events, and variables would need to be to make the program. They are then given a completed project guide which shows one way to implement the project. Students are then walked through this process through a series of levels. At the end of the lesson students have an opportunity to make improvements to the program to make it their own.

## **Resources**

* [Emoji Race](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/export?format=pdf) - Project Guide (copy as [MS Word](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/export?format=doc), [Google Doc](https://docs.google.com/document/d/1dn4FIRKYaVb1o7TNiyeKqoT1fBL9FPoqOmJV-nvJYUM/copy))

課程概述  
本課程將向學生介紹如何在這整個單元中設計出自己程式的流程。這個流程是以專案指南為中心，在開始寫程式之前，要求學生勾勒出他們的構思、找出要使用的Circuit Playground元素、定義出變數、並描述出事件。這個流程與我們在第3單元中學到的遊戲設計流程有點像。在本課程中，學生們首先玩一個需要寫程式的類似賽跑的遊戲。他們將討論出他們認為需要用到程式的實驗板組件、事件和變數。然後他們會拿到有一份完整描述實施專案方法的專案指南。隨後學生便可以據此來執行一系列關卡來走完這個流程。在這課程結束時，學生有機會對程式進行改進，使其成為自己的遊戲。

課程資源

* 笑臉賽跑遊戲 - 專案指南（MS Word，Google Doc）

Sample Program

[Student Overview](https://curriculum.code.org/csd-18/unit6/8/#level-expando-8-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/8/puzzle/2)

範例程式

給學生的概述

在Code Studio上查看

# **Emoji Race!**

This is an example of a race game that you'll build by the end of this lesson. To play, pair up with another student. The student on the left will the click the left button on the Circuit Playground as fast as they can, while the student on the right clicks the right button. Whoever can get their emoji to the bottom of the screen first wins.

笑臉賽跑遊戲！  
這是您將在本課程結束時將產生的賽跑遊戲範例。在開始玩時，必須和另一個學生一組。左側的學生將盡快地點擊Circuit Playground上的左鍵，而右側的學生則點擊右鍵。看誰能先將他們的笑臉符號跑到畫面的最底部就能獲勝。

# **Do This**

Turn to a classmate and make a list of the following information.

* What components of the board does this program use?
* What events is this program responding to?
* What functions might you create to make this program work?

動手做

轉向同學並列出以下信息。

* 該程式使用了哪些電路板組件？
* 這個程式回應響應了哪些事件？
* 這個程式應該包含哪些函數才能正常運作？

Plan Your Project

* [Student Overview](https://curriculum.code.org/csd-18/unit6/8/#level-expando-8-3-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/8/puzzle/3)

計劃您的專案  
 給學生的概述  
 在Code Studio上查看

# **Stop**

Before you move on you'll need to look at the Project Guide for this project. Wait for instructions from your teacher as well.

停下來

在繼續之前，您需要查看該專案的專案指南。並等待老師的指示。

* Screen Design

### **Student Instructions**

學生操作指南

# **Getting Started: Screen Design**

You should have already reviewed the planning guide for this project. Some of the work to turn this project guide into a working program has already been started. Based on the project guide you're going to do the rest of this work.

入門：畫面設計  
您應該已經查看了該專案的規劃指南。將該專案指南轉變為可正常運作的程式的一些工作應該已經開始。根據專案指南，您將完成剩下的工作。

# **Do This**

We've already created a "start" screen and included a title and "play" button. Using your planning guide, finish creating this screen. Specifically you'll want to:

* Update the title text to say "Emoji Race"
* Add instructions for playing using the ID "instructions"

Feel free to add any other design tweaks that you'd like.

動手做

我們已經產生了一個“start”畫面，其中包括一個標題和“play”按鈕。使用您的計劃指南，完成創建此畫面。具體來說，你可能會想：

* 更新標題文字為“Emoji Race”
* 使用ID“instructions”來添加遊戲說明

隨意添加想要的任何其他設計。

### **Student Instructions**

學生操作指南

# **Game Screen Design**

We've also added a "game" screen where the game will be played. It just needs a couple of tweaks.

# **Do This**

Switch to the "game" screen and, using your planning guide, fix this screen by changing the color of the two emoji images.

Feel free to add any other design tweaks that you'd like.

遊戲畫面設計  
在遊戲進行中，我們還添加了一個“game”畫面。它需要一些設定來達到。

動手做  
切換到“遊戲”畫面，並根據計劃指南，更改兩個笑臉圖像的顏色。  
隨意添加您想要的任何其他設計。

### **Student Instructions**

學生操作指南

# **Designing the Win Screen**

The final screen in this program shows who won. It's totally blank right now.

# **Do This**

Switch to "win" and design it based on your project guide. Make sure to pay close attention to the ID of each element. Once you've added the required elements, feel free to add any other design tweaks that you'd like.

設計Win 畫面  
這是這個程式的最後一個畫面，主要是顯示誰獲勝了。目前它完全是空白的。

動手做  
切換到“Win”畫面，並根據專案指南進行設計。請務必密切注意每個元素的ID。添加完程式所需元素後，你可以添加想要的任何其他設計。

* Finishing Functions

### **Student Instructions**

學生操作指南

# **The startGame() Function**

If you look at the events we planned for in the planning guide, you may notice that two events do essentially the same thing. The events that respond to both the "play" and "replay" buttons could be written with identical code, but duplicate code like this is a perfect place to use a **function**.

# **Do This**

We've already added blocks that call a new function named startGame() to the appropriate event handlers, and we've created an empty function for you to build out.

* Find the startGame() function definition ( [Show me where](https://curriculum.code.org/csd-18/unit6/8/#triggercallout=block_trigger) )
* Inside the function:
  + Move both of the emoji images to the top of the screen by setting each one's "y" property to 0
  + Change the screen to "game"

Once you've fleshed out the startGame() function, try clicking the "play" button to check that it's working.

startGame（）函數  
如果你看過規劃指南中計劃的事件，可能會注意到其中有兩個事件的說明是相同的。回應“play”和“replay”按鈕的事件可以用相同的代碼編寫，像這樣在函數中使用重複的代碼是沒有問題的。

動手做  
我們已經在相對的事件處理程序中添加了一個名為startGame（）的新函數，其內容是空的，以供接下來的使用。

* 找到startGame（）函數定義的位置（顯示在哪裡）
* 在這函數內：
  + 把這兩個笑臉符號的“y”屬性設置為0，可將這它們的圖像移動到畫面的最頂端
  + 將畫面更改為“game”

一旦你完成了startGame（）函數，請試著點擊“play”按鈕來檢查它是否正常運作。

**Student Instructions**

學生操作指南

# **Functions with Parameters**

Most of the existing functions you use in App Lab need inputs, or **parameters**, to pass the function necessary information. For example, when changing a screen, you pass the ID of that screen as a parameter, which looks like [setScreen("win")](https://docs.code.org/applab/setScreen/).

These parameters show up as a variable that can be used inside the function, and you can create functions that use parameters too!

包含參數的函數  
  
在App Lab中，大多數函數都需要輸入或參數來傳遞函數所需的訊息。例如，在更改畫面時，您將該畫面的ID作為參​​數來傳遞，就像是 setScreen（“win”）。  
這些參數就像是可在函數內部使用的變數，您也可以自己編寫使用參數的函數！

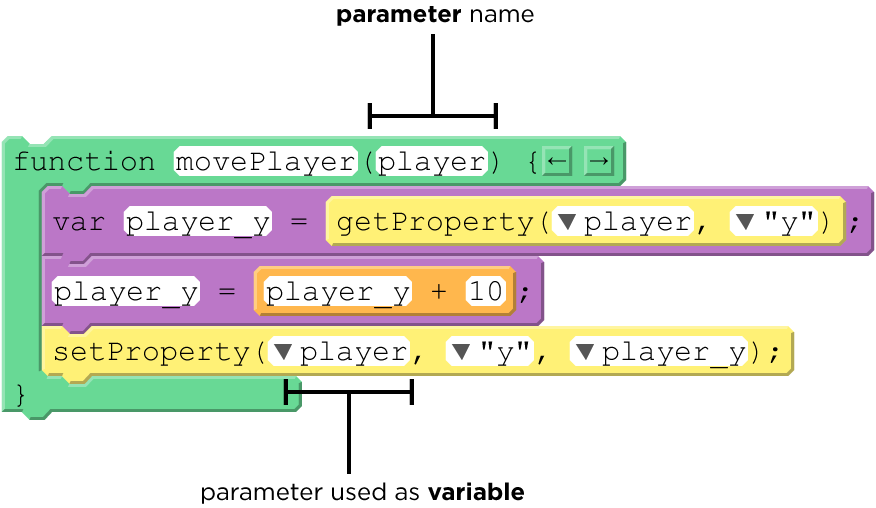
Teaching Tip

### **Exposure to Functions with Parameters**

Parameters are a powerful way to make functions more useful by giving them specific input. We are introducing this here primarily as a way to expose students to this concept, but functions with parameters won't be taught thoroughly until chapter 2. For students who are interested in experimenting more with this technique now, direct them to [the documentation](https://docs.code.org/applab/functionParams_n/).

教學建議  
介紹包含參數的函數  
參數是經由對函數提供特定的輸入，它可使函數的功能變得更強大。我們在這裡介紹這個主要是為了讓學生了解這個概念，但是在第2章之前不會詳細教授包含參數的函數。對於有興趣想再多地嘗試這種技術的學生，可以請他們參考這些文件。

# **Do This**

****

Click to expand (點擊展開)

Instead of a separate function to move the red and the blue players, we've created one function called movePlayer() that takes the ID of the player image as a parameter called player. Inside this function, you can use player any place where you need to specify the ID of the player you want to control.

* Find where movePlayer() is defined ( [Show me where](https://curriculum.code.org/csd-18/unit6/8/#triggercallout=block_trigger) )
* Inside the movePlayer() definition:
* Get the current "y" property of the player and save it to a variable called player\_y
* Increase the value of player\_y by 10
* Set the "y" property of player to your new value player\_y

Once you've added your code, test it. You should see that *both buttons* work even though we only created one function - cool!

動手做

我們創建了一個名為movePlayer（）的函數，它不是一個單獨移動紅色或藍色笑臉的函數，而是將玩家圖像的ID作為一個名為player的參數。在此功能中，您可以在任何需要控制指定 玩家 ID的地方使用player。

* 找到movePlayer（）定義的位置（顯示在哪裡）
* 在movePlayer（）定義內：
* 獲取player的“y”屬性並將其保存到名為player\_y的變數中
* 將player\_y的值增加10
* 將player的“y”屬性設置為新值player\_y

添加完程式碼後，請對其進行測試。你應該可以看到兩個按鈕都可正常運作，即便我們只編寫了一個函數 - 很酷吧！

* [9](https://curriculum.code.org/csd-18/unit6/8/#level-expando-8-9)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/8/puzzle/8)

### **Student Instructions**

學生操作指南

# **The checkWin() Function**

The last function that you need to figure out is the checkWin() function, which after each player is moved, and is used to both check if that player has made it to the bottom of the screen and to announce the winner if necessary. For now, we're just going to find out where the player is and log it to the console.

# **Do This**

Just like the movePlayer() function, checkWin() takes a single parameter called player. If you call checkWin("red"), then the variable player will have the value "red".

* Find where the function is defined ( [Show me where](https://curriculum.code.org/csd-18/unit6/8/#triggercallout=block_trigger) )
* Create a variable called player\_y and assign it the "y" property of the player
* Use either a [console.log()](https://docs.code.org/gamelab/console.log/) command or a watcher to report the value of player\_y
* Play the game with only one button, keeping an eye on the value of player\_y
* Decide what value of player\_y should be considered the bottom of the screen

checkWin（）函數  
你要弄清楚的最後一個函數是checkWin（）函數，在移動每個玩家圖像之後，它用於檢查該玩家圖像是否已將其移至畫面的最底部並在那時宣布獲勝者。現在，我們只是想知道玩家圖像的位置並將其記錄到控制台。

動手做  
就像movePlayer（）函數一樣，checkWin（）接受一個名為player的參數。如果你呼叫checkWin（“red”），那麼player 變數的值將是“red”。

* 找到定義函數的位置（顯示在哪裡）
* 新建一個名為player\_y的變數，並為其值設定為player的“y”屬性
* 使用console.log（）命令或使用watcher 來顯示player\_y的值
* 只用一個按鈕來玩遊戲，並留意player\_y的值
* 當玩家圖像到達畫面的最底部時，決定player\_y的值應該應該為何

Checking for a Winner

檢查誰是獲勝者

### **Student Instructions**

學生操作指南

# **Check for a Winner**

Your game should be pretty playable by now. Two players can click their respective buttons, and their emojies will race down the screen. We still need a way for a player to win though.

# **Do This**

Using a conditional inside the checkWin() function, you want to check if a player has won each time they click. For now, we can just use [console.log](https://docs.code.org/applab/console.log/) to report the winner.

* Add an if statement to the bottom of checkWin()
* For the condition of your if statement, check if player\_y is greater than 350 (or use the bottom of screen value you decided on in the last level)
* In your conditional, add a [console.log](https://docs.code.org/applab/console.log/) that reports which player won
* Test your program to make sure that it reports a winner when one of the emojis makes it to the bottom of the screen

檢查誰是獲勝者  
你的遊戲現在應該可以玩了。兩個玩家可以點擊他們各自的按鈕，他們的笑臉符號將在畫面上往下賽跑。我們仍然需要一個方法來找出獲勝的玩家。

動手做  
要想檢查玩哪個家在比賽時是否獲勝，就必須在checkWin（）函數中使用條件式。目前我們可以使用console.log來記錄獲勝者。

* 在checkWin（）的下面添加if語句
* 在if語句的條件式中，檢查player\_y是否大於350（或使用您在最後一級決定的畫面底部值）
* 在條件式中，添加一個console.log來記錄哪個玩家贏了
* 測試您的程式，以確保當其中一個笑臉符號顯示在畫面最底部時，它會記錄獲勝者

**Student Instructions**

學生操作指南

# **Change to the Win Screen**

Now that we have conditionals to check which player won, let's switch to the "win" screen when a player wins.

# **Do This**

Inside the conditional that you've created to check if a player has reached the bottom, add a [setScreen](https://docs.code.org/applab/setScreen/) block to change to the "win" screen. Make sure you test that your new code works before moving on!

切換到Win畫面  
目前有條件式可用來檢查哪個玩家獲勝了，如果有任何一位玩家獲勝時，便切換到“win”畫面。

動手做  
在你編寫的條件式中，檢查玩家圖像是否已跑到最底部，若是跑到了，便添加一段setScreen程式區塊以切換到“win”畫面。在繼續下一步驟之前，測試新加的程式碼確保其有效！

### **Student Instructions**

學生操作指南

# **Display the Winner**

The winning screen doesn't actually know *who* won the game. Let's fix that now.

# **Do This**

Before each call to [setScreen](https://docs.code.org/applab/setScreen/), add a [setProperty](https://docs.code.org/applab/setProperty/) block to change the "text" property of "winner" to display the actual winner.

顯示獲勝者  
獲勝的畫面本身並不知道誰贏了這場比賽。讓我們現在來解決這個問題。

動手做  
在每次呼叫setScreen之前，添加一個setProperty程式區塊來改變 winner 的“text”屬性以顯示實際的獲勝者。

* [13](https://curriculum.code.org/csd-18/unit6/8/#level-expando-8-13)

### **Student Instructions**

學生操作指南

# **Winning Buzzer**

That last thing that that is missing from your project guide's description of this project is the buzzer. It should buzz a high tone if player 1 wins and a low tone if player 2 wins.

# **Do This**

So far we've been able to just use the player parameter directly to report which player won, but that won't work for buzzing different sounds. Inside the conditional that checks if a player won, you'll need to add *another* conditional that checks which player it was. If player == "red" the buzzer should play a high note, otherwise (else) it should play a low note.

獲勝者鈴聲  
專案指南對此專案的描述中遺漏的最後一件事情就是蜂鳴器。如果玩家1獲勝，它應該是高音鈴聲，如果玩家2獲勝則是低音鈴聲。

動手做  
到目前為止，我們已經能夠直接使用player參數來記錄是哪個玩家贏了，但還沒做到這用蜂鳴器發出不同音調的鈴聲。在條件式中檢查玩家是否獲勝，你需要添加另一個條件來檢查它是哪個玩家。如果player==“red”，則蜂鳴器應該播放高音鈴聲，否則它應該播放低音鈴聲。

* Make it Your Own
* [Student Overview](https://curriculum.code.org/csd-18/unit6/8/#level-expando-8-14-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/8/puzzle/14)

# **Make It Your Own**

You just walked through someone else's plan for creating a program, so now it's time to make it your own. What additional features or challenges do you want to create?

# **Do This**

Select one of the challenges below to add to the game or come up with a challenge of your own.

* Change the look and layout of the screens
* Use the buzzer or the led to show when one of the buttons has been clicked
* Set the "icon-color" property of "winner\_image" to the winning color
* Make the emojis start out sad, change to meh halfway, and end up happy
* Use a variable and conditionals to make sure that players can only move their emojis when the "game" screen is showing

自己的程式  
你剛剛執行完其他人所編寫的程式計劃，所以現在是時候讓它成為你自己的計劃了。您想要增加哪些其他功能或挑戰呢？

動手做

選擇以下挑戰之一添加到遊戲中或提出自己的挑戰。

* 更改畫面的外觀和佈局
* 點擊其中一個按鈕時，使用蜂鳴器出聲或指示燈顯示
* 將“winner\_image”的“icon-color”屬性設置為獲勝顏色
* 讓表情符號開始由悲傷，變到平和，最後變成快樂
* 使用變數和條件式確保玩家只能在“game”畫面顯示時移動他們的笑臉符號

## **Wrap Up (20 min)**

### **Make It Your Own**

This last level encourages students to make the game their own. If students have made their way to this point they have all the skills they need to progress through the curriculum, so there is no pressure to complete any of the modifications suggested in this level. If you have time, however, getting practice planning and implementing new features will be a useful skill. Even just modifying the look of the screens is an easy way students can make the game their own.

**Share:** Once students have completed the project they can share their work with their classmates. Encourage students to showcase the additional code they wrote and explain how it has changed the way the game works.

總結（20分鐘）  
自己的程式  
這是最後一個關卡來鼓勵學生製作自己的遊戲。如果學生已經完成到這一步驟，也就擁有完成課程所需的所有技能，因此他們可以自行己決定是否要完成此關卡中建議的任何修改。然而，如果有時間的話，練習計劃和施作新功能將是一項有用的技能。即使只是修改畫面的外觀也是學生可以自己製作遊戲的簡單方法。  
分享：一旦學生完成了專案，他們就可以與同學分享他們的成果。鼓勵學生展示他們編寫的額外程式碼，並解釋它如何改變遊戲的運作方式。

## **Standards Alignment**

#### **[View full course alignment](https://curriculum.code.org/csd-18/standards/)**

#### **CSTA K-12 Computer Science Standards**

**AP** - Algorithms & Programming

**CS** - Computing Systems

標準對照  
查看完整課程對照  
CSTA K-12計算機科學標準  
AP - 演算法和程式設計  
CS - 計算機系統

# **Lesson 9: Project: Make a Game**

第9課：專案：製作遊戲

#### **App Lab | Maker Toolkit | Project**

App Lab| 創客工具組 | 專案

## **Overview**

Students take what they've learned through chapter one, and develop an app of their own design that uses the board to output information.

課程概述

學生們可以透過第一章所學到的東西，開發一個自己設計的應用程式，使用實驗板來輸出資訊。

## **Purpose**

This end of chapter assessment is a good place for students to bring together all the pieces they have learned (event handlers, loops, using the board as output) in one place. This project is purposefully left very open to allow students thing broadly about how physical output might be useful in an app - this is a great opportunity to encourage students to revisit programs they've written earlier in this unit or in Unit 4 that would benefit from using the board to output information.

## 課程目的

## 本章最後的評估部分, 是一個能讓學生們將學習過的所有東西（事件處理器，迴圈，使用實驗板作為輸出）整合在一起的好地方。這個專案是非常開放且沒有任何限制的，它能讓學生廣泛地了解實體的輸出在應用程式中的好用之處 - 而且, 這是一個很好的機會來鼓勵學生們重新回顧他們稍早之前在本單元或單元四中寫出來的程式, 並且利用實驗版將程式輸出成資訊。

## **Agenda**

* [Warm Up (10 min)](https://curriculum.code.org/csd-18/unit6/9/#project-make-a-game0)
* [Demo Project Exemplars](https://curriculum.code.org/csd-18/unit6/9/#project-make-a-game1)
* [Activity (90 - 120 min)](https://curriculum.code.org/csd-18/unit6/9/#project-make-a-game2)
* [Unplugged: Program Planning](https://curriculum.code.org/csd-18/unit6/9/#project-make-a-game3)
* [Prototyping the Program](https://curriculum.code.org/csd-18/unit6/9/#project-make-a-game4)
* [Wrap Up (10 min)](https://curriculum.code.org/csd-18/unit6/9/#project-make-a-game5)
* [Sharing Projects](https://curriculum.code.org/csd-18/unit6/9/#project-make-a-game6)
* [Self Assessment](https://curriculum.code.org/csd-18/unit6/9/#project-make-a-game7)

課程流程

- 熱身（10分鐘）

- 展示專案範例

- 活動內容（90 - 120分鐘）

- 不插電練習：程式規劃

- 程式原型設計

- 總結（10分鐘）

- 分享專案

- 自我評估

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/1/)**

### 查看關卡

**Objectives**

### **Students will be able to:**

* Use event handlers to respond to user interaction
* Design a piece of software that uses hardware for non-traditional input and output
* Prototype a program that integrates software and hardware

課程目標

學生將具備以下能力：

- 使用事件處理器回應使用者的動作

- 設計一個使用硬體進行非傳統輸入和輸出的軟體

- 設計一個整合軟體和硬體的程式

## Preparation

* Print a copy of [Make a Game - Project Guide](https://docs.google.com/document/d/1vPeG-bJlGJTVU91eNzxIzp1gTCK-s5AGAX_T-SrfBZE/edit) for each group of students
* Print a copy of [Make a Game - Rubric](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/edit) for each student

課前準備

- 為每組學生列印''製作遊戲的專案指南''

- 為每個學生列印''製作遊戲的評量表''

## Links

**Heads Up!** Please make a copy of any documents you plan to share with students.

相關連結

注意！請將所有需要給學生的文件先印出來

### **For the Students**

* [Make a Game](https://docs.google.com/document/d/1vPeG-bJlGJTVU91eNzxIzp1gTCK-s5AGAX_T-SrfBZE/edit) - Project Guide Make a Copy
  + [PDF](https://docs.google.com/document/d/1vPeG-bJlGJTVU91eNzxIzp1gTCK-s5AGAX_T-SrfBZE/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1vPeG-bJlGJTVU91eNzxIzp1gTCK-s5AGAX_T-SrfBZE/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1vPeG-bJlGJTVU91eNzxIzp1gTCK-s5AGAX_T-SrfBZE/copy)

給學生的資料

- 製作遊戲 - 列印 專案指南

- PDF

- Microsoft Word

- Google Docs

* [Make a Game](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/edit) - Rubric Make a Copy
  + [PDF](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/copy)

[- 製作遊戲 - 評量表 列印](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/copy)

[- PDF](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/copy)

[- Microsoft Word](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/copy)

[- Google Docs](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/copy)

## **Support**

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

### [**Report a Bug**](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%209%20curriculum.code.org/csd-18/unit6/9/)

課外支援

討論區

問題回報

# **Teaching Guide**

## **Warm Up (10 min)**

教學指南

暖身活動（10分鐘）

### **Demo Project Exemplars**

**Goal:** Students see an example of a final project and discuss the different elements that went into making it.

**Display:** On the projector, show the example projects at the beginning of this lesson.

**Prompt:** Analyze the exemplar for the following elements:

* Where is input being taken?
  + What events do you think are being used for each input?
* What information is being output through the board?
* Where might this program be using a loop?

展示專案範例

目標：給學生們參考一個已經完成的專案範例，並討論可以用哪些元件製作該範例。

展示：在課程開始的時候使用投影機展示範例。

提示：分析該範例的以下元素：

- 輸入在哪裡？

- 你認為每個輸入使用了哪些事件？

- 透過實驗板, 輸出什麼樣的資訊?

- 這個程式可能在哪裡使用迴圈？

**Discuss:** Have students share their observations and analyses of the exemplar.

**Distribute:** Provide each student with a copy of the [Make a Game - Rubric](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/edit). Review the different components of the rubric with them to make sure they understand the components of the project.

討論：讓學生分享他們對範例的觀察和分析。

分發評量：發給每個學生製作遊戲的評量表，和學生們一起檢視評量中不同的元件以確保他們了解專案使用的組成元件。

## **Activity (90 - 120 min)**

### **Unplugged: Program Planning**

**Group:** Place students in groups of 2-3.

**Distribute:** Hand out the [Make a Game - Project Guide](https://docs.google.com/document/d/1vPeG-bJlGJTVU91eNzxIzp1gTCK-s5AGAX_T-SrfBZE/edit) to students. This is the tool students will use to scope out their projects before getting onto the computers. Give students some time to brainstorm the type of program they want to make.

活動內容（90 - 120分鐘）

不插電練習：程式規劃

分組：將學生分成2-3組。

分發指南：發給學生 製作遊戲 - 專案指南。這個表格是讓學生在使用電腦寫程式之前用來定義他們的專案範疇的工具。給學生一些時間來腦力激盪出他們想要製作的程式類型。

#### **Make a Game Project Guide**

#### **Brainstorming**

Give groups some time to brainstorm ideas for their project. The goal is to come up with a simple game that can be controlled by using elements of the Circuit Playground. Encourage students to think about all of the programs and projects that they've made so far as potential starting points (including the prototypes from Unit 4).

製作遊戲專案指南

腦力激盪

給小組們一些時間來集體腦力激盪想出他們專案的點子。目標是想出一個可以使用Circuit Playground的元件來控制的簡單遊戲。鼓勵學生考慮他們到目前為止所做過的所有程式和專案當作開始（包括第4單元的程式原型）。

Teaching Tip

Students may find it particularly difficult to predict *all* of the events that they'll want to respond to or the functions that they'll need to create. Make sure that they know this is an *iterative* process and planning is only the first step. We want to start this project with as clear a plan as possible, but there will likely be things that weren't considered from the start and the plan will need to change accordingly.

教學建議

學生可能會發現要預測他們所有想要做的事件或需要的函數是非常困難的。確保他們了解這是一個重複的過程，而規劃只是第一步。我們希望以一個盡可能明確的計劃來開始這個專案，但可能會有一些事情一開始沒有考慮，那麼就需要相應的改變規劃。

1. Once groups have settled on an idea, they can record it on the activity guide under Project Brainstorming.

一旦小組們確定了想法，他們就可以將其記錄在專案腦力激盪下的活動指南中。

1. Next students think through the Events they'll need to respond to.

接下來,學生思考有哪些需要回應的事件。

1. Finally, students consider the functions that they may need to create.

最後,學生考慮他們可能需要創建的功能有哪些。

### **Prototyping the Program 程序原型設計**

### **Transition:** Once teams have completed their planning sheet, they can head to Code Studio to work on prototyping their projects

### 接下來:一旦小組完成了規劃表，他們就可以前往 Code Studio 進行設計。

### 

### **Code Studio levels**

* Lesson Overview
* [Student Overview](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/1)

* Demo - Bug Grab
* [Student Overview](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/2)

Code Studio 關卡

- 課程概述

- 給學生的概述

在 關卡上查看

- 展示 - 抓蟲蟲

- 給學生的概述

查看關卡

# **Grab that Bug!**

This is an example of a tug o' war game that could be built in this lesson. To play, pair up with another student. The student on the left will click the left button on the Circuit Playground as fast as they can, while the student on the right clicks the right button. Whoever can get the bug over to their side first wins.

抓蟲蟲！

這是可以在本課程中實作的拉鋸比賽遊戲的例子。和另一個學生組隊來玩。左側的學生將盡可能地快速按Circuit Playground上的左鍵，而右側的學生則按右鍵。能夠將蟲蟲拉到他們那一邊的人就能獲勝。

* Plan Your Project

計劃你的專案

* [Student Overview](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-3-sfmd)

給學生的概述

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/3)

查看關卡

# **Stop**

Before you move on you'll need to look at the Project Guide for this project. Wait for instructions from your teacher as well.

停下來

在繼續之前，你需要查看該專案的專案指南和等待老師的指示。

* Screen Design
* [4](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-4)
* (click tabs to see student view)

- 畫面設計

- 4（點擊看學生所看到的內容）

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/4)

查看關卡

### **Student Instructions**

學生操作指南

# **Create** **Your Screens**

製作你的畫面

The first thing you'll need to do is create all of the screens that you've sketched in your planning guide.

# 你需要做的第一件事是製作你在規劃指南中的所有畫面。

# **Do This**

動手做

Use Design Mode to create all of the screens that your program will need. Don't forget to **pick sensible IDs** for all of your design elements.

* Events
* [5](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-5)
* [6](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-6)
* (click tabs to see student view)

使用 設計模式 創建程式所需的所有畫面。不要忘記為所有設計元素選擇合理的ID。

- 事件

- 5 - 6 (點擊看學生所看到的內容)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/5)

查看關卡

### **Student Instructions**

學生操作指南

# **UI Element Events**

使用者介面元素事件

Now that all of your screens are designed, you can add any event handlers that will respond to *screen interaction*. These should be listed in the "Events" section of your activity guide. If you run into new events that you hadn't thought of in the planning stage, make sure you add them to your project guide.

現在你的所有畫面都已設計好，你可以添加任何可以回應畫面互動的事件處理器。這些都應該被列在活動指南的「事件」中。如果你遇到計劃階段中沒有想到的新事件，請務必將它們加到專案指南中。

Don't worry about making these completely functional yet. If your events rely on board elements or your own functions, we'll take care of that in later levels.

不要擔心這些功能是否能完全正常動作。如果你的事件仰賴實驗板上的元件或你自己的函數才能執行的話，我們會在後面更進階關卡中處理。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/6)

查看關卡

### **Student Instructions**

# **Board Events**

Go back to the Events section of your activity guide and find any events that respond to your board (ones that would use [onBoardEvent()](https://docs.code.org/applab/onBoardEvent/)). Create event handlers for each of those now. If these events rely on functions you haven't written yet, just leave yourself a comment in the code.

學生操作指南

實驗板的事件

回到活動指南的事件表格並且找到任何一個能反應在你的實驗板上的事件（那些使用 onBoardEvent() 的事件），為他們創建事件處理器。如果這些事件需要仰賴你尚未編寫的函數才能執行的話，只需在程式中留下註解當作提醒。

* Functions
* [7](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-7)
* [8](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-8)
* (click tabs to see student view)

- 功能

- 函數

- 7 - 8 - （點擊看學生所看到的內容）

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/7)

查看關卡

### 

### StudentInstructions

# **Define Your Functions**

Define all of your functions at the bottom of the program, under the comment Create your functions here.

Right now we're only worrying about writing the function *definitions*, the part which looks like:

function myFunction() {  
 // Things my function does  
}

學生操作指南

定義你的函數

在程序底部定義所有函數，在註釋下新增函數。

現在,我們只需要擔心如何編寫函數定義，而這個部分看起來就像這樣：

function myFunction（）{

//我的函數所做的事情

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/8)

查看關卡

### **Student Instructions**

# **Calling your Functions**

With your functions defined, you can call them wherever you need. Go through your event handlers, or anywhere else in your program that your function should be used, and add calls to the function you created.

* Finishing Touches
* [9](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-9)
* (click tabs to see student view)

學生操作指南

呼叫你的函數

定義了你的函數後，你可以在需要時候隨時地呼叫它們。瀏覽你的事件處理器，或程式中應該使用你的函數的任何其他地方，並添加能夠呼叫你創建的函數的指令。

- 完成

- 9 - （點擊看學生所看到的內容）

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/9)

查看關卡

### Student **Instructions**

# **Finishing Touches and Testing**

At this point you should have most of your program in place. Now you need to add whatever finishing touches are needed and do a little testing. When your app is ready, have your classmates try it out. See if you can find any bugs, confusing design choices, or missing features you'd like to work on.

學生操作指南

完成和測試

此時，你應該已經寫好大部分的程式了，現在, 你必須做最後的修改並進行一些測試。當你的應用程式準備就緒時，讓你的同學也嘗試一下。看看你是否可以找到任何錯誤, 會令人混淆的設計或缺少你想要處理的功能。

Once you're all done with this version of your program, click Submit to turn it in.

* Reflection
* [10](https://curriculum.code.org/csd-18/unit6/9/#level-expando-9-10)
* (click tabs to see student view)

一旦完成程式後，按提交按鈕繳交作業。

- 反思

- 10 - （點擊看學生所看到的內容）

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/9/puzzle/10)

查看關卡

### **Student Instructions**

## **Wrap Up (10 min)**

### **Sharing Projects**

**Share:** Find a way for groups to share their projects with each other. It will likely be helpful to use the share link for the project so that students can share the project with other students.

### 學生操作指南

### 總結（10分鐘）

### 分享專案

### 分享：為小組們找到一種彼此分享專案的方式。使用專案的共享鏈接可能會有所幫助，以便學生可以與其他學生共享專案。

### **Self Assessment**

Using the [Make a Game - Rubric](https://docs.google.com/document/d/1wHZM3BVbm2gOchvAXUTIXaC820pqlRi5pzg5CLIF6Pw/edit), students should assess their own project before submitting it.

Send students to Code Studio to complete their reflection on their attitudes toward computer science. Although their answers are anonymous, the aggregated data will be available to you once at least five students have completed the survey.

自我評估

使用製作遊戲評量表，學生應在提交之前評估自己的專案。

讓學生到 Code Studio 中完成他們對計算機科學想法的回饋。雖然他們的回答是匿名的，但是只要最少有五名學生完成調查，你將可以獲得匯總的數據。

# **Lesson 10: Arrays and Color LEDs**

#### **第10課：陣列和彩色LED指示燈**

#### **App Lab | Maker Toolkit**

## **Overview** 課程概述

## An array is an ordered collection of items, usually of the same type. In this lesson, students learn ways to access either a specific or random value from a list using its index. They then learn how to access the colorLEDs array that controls the behavior of the color LEDs on the Circuit Playground. Students will control the color and intensity of each LED, then use what they have learned to program light patterns to create a light show on their Circuit Playground.

陣列是項目的有序集合，通常是相同類型的。 在本課程中，學生將學習如何使用索引從列表中找到特定值或隨機值。 然後，他們學習如何透過colorLEDs陣列來控制Circuit Playground上的彩色LED的行為。 學生將控制每個LED的顏色和強度，然後利用他們學到的東西來編寫燈光模式，在他們的Circuit Playground上創建一個燈光秀。

## **Purpose 課程目的**

# Arrays are a common data structure in computer science, used to make storing and accessing information easier. The lesson prepares students to access items within an array, but doesn't yet cover creating and modifying arrays, which will be addressed in the following lesson. Students are also introduced to the ring of color LEDs, which are exposed as an array called [colorLeds](https://docs.code.org/applab/colorLeds/). Students learn how to access and control each LED in the array individually, preparing them to access multiple LEDs through iteration later in the chapter.

陣列是計算機科學中常見的數據結構，用於使信息的存儲和讀取更容易。 該課培養學生的陣列中讀取的項目，但尚未涵蓋創建和修改陣列，這將在下一課中介紹。 學生們還會了解到彩色LED環，它們被稱為colorLeds陣列。 學生將學習如何分別讀取和控制陣列中的每個LED，透過它們讀取本章後面的多個LED。

## **Assessment Opportunities 評估機會**

1. Access an element in an array using its index 使用索引讀取陣列中的元素  
   Code Studio: See the rubric on bubble 7 and the quick check on bubble 9

Code Studio：查看氣泡7上的指示和快速檢查氣泡9

1. Use the color LED array to individually control each color LED 使用彩色LED陣列分別控制每個顏色的LED  
   Code Studio: See the rubric on bubble 17 Code Studio:查看氣泡17上的指示

## **Agenda 課程大綱**

# [Warm Up (5 min) 暖身（5分鐘）](https://curriculum.code.org/csd-18/unit6/10/#arrays-and-color-leds0)

# [Discussion: What is a List? 討論：什麼是清單？](https://curriculum.code.org/csd-18/unit6/10/#arrays-and-color-leds1)

# [Activity (45 min) 活動（45分鐘）](https://curriculum.code.org/csd-18/unit6/10/#arrays-and-color-leds2)

# [Arrays and Color LEDs 陣列和彩色LED](https://curriculum.code.org/csd-18/unit6/10/#arrays-and-color-leds3)

# [Wrap Up (5 min) 總結（5分鐘）](https://curriculum.code.org/csd-18/unit6/10/#arrays-and-color-leds4)

# [Discussion: Objects Vs Arrays 討論：對象與陣列](https://curriculum.code.org/csd-18/unit6/10/#arrays-and-color-leds5)

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/1/)**

## **Objectives** 課程目標

### **Students will be able to: 學生將能具備以下能力：**

# Access an element in an array using its index 使用索引讀取陣列中的元素

# Use the color LED array to individually control each color LED 使用彩色LED陣列分別控制每個顏色的LED

## **Vocabulary 詞彙**

# **Array** - A data structure in JavaScript used to represent a list.

* Array - JavaScript中用來表示列表的資料結構。

## **Introduced Code 介紹程式碼**

# [colorLeds](https://docs.code.org/applab/colorLeds/)

# [colorLeds[index].on()](https://docs.code.org/applab/on/)

# [colorLeds[index].blink(interval)](https://docs.code.org/applab/blink/)

# [colorLeds[index].intensity(brightness)](https://docs.code.org/applab/intensity/)

# [colorLeds[index].color(color)](https://docs.code.org/applab/color/)

# [list[index];](https://docs.code.org/applab/accessListItem/)

## **Support 支持**

### **[Lesson Forum](https://forum.code.org/tags/c/csd/csd-unit-6-lesson-10)** [**討論區**](https://forum.code.org/tags/c/csd/csd-unit-6-lesson-10)

### [**Report a Bug 問題回報**](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%2010%20curriculum.code.org/csd-18/unit6/10/)

# **Teaching Guide 教學指南**

## **Warm Up (5 min) 暖身（5分鐘）**

### **Discussion: What is a List? 討論：什麼是清單？**

# **Set Up:** Have students take out their journals. In their journals have students respond to the following prompt.

設置：讓學生拿出他們的學習日誌。在他們的學習日誌中，讓學生回應以下提問。

# Discussion Goal 討論目標

# While the types of lists students come up with vary from class to class, the two main characteristics of lists that students should understand is that they usually are comprised of items of the same type (tasks to complete, names of people, song titles, grocery items, etc.) and that they have an order. Some common types of lists include a to-do list, a class roster, a playlist, or a grocery list.

雖然學生提出的列表類型因班級而異，但學生應該理解列表的兩個主要特性是它們通常由相同類型的項目組成（要完成的任務，人名，歌名，雜貨物品等）而且他們有一定的順序。一些常見類型的列表包括待辦事項列表，班級名單，播放列表或購物清單。

# **Prompt:** What is a list? What do you use lists for? 提示：什麼是清單？你用什麼列表？

# **Discuss:** Once students have had time to write down their own thoughts, have them share out to the whole class. Work as a class to try to get an understanding of what a list is.

討論：一旦學生有時間寫下自己的想法，讓他們分享給全班同學。全班一起來嘗試了解什麼是列表

# Remarks 備註

# When we program, we sometimes want to store lists of items. In Javascript, we do this using an array. An array is a collection of items that are stored in a particular order. We're going to look at different ways to use arrays, and then see how they will help us to control more LEDs on the Circuit Playground.

當我們寫程式時，我們有時想要存儲項目列表。 在Javascript中，我們使用陣列執行此操作。 陣列是按特定順序存儲的項目的集合。 我們將看看使用陣列的不同方法，然後看看它們將如何幫助我們控制Circuit Playground上的更多LED。

## **Activity (45 min) 活動（45分鐘）**

### **Arrays and Color LEDs 陣列和彩色LED**

# **Transition:** Head to Code Studio 轉換：前往Code Studio

### **Code Studio levels Code Studio級別**

# Lesson Overview 課程概述[Student Overview](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-1-sfmd) 給學生的概述

## **Overview 課程概述**

Learn how lists can be used to store multiple values in a single variable name.

了解列表如何在一個變數名稱中儲存多個變數。

## **Vocab 詞彙**

* Array - A data structure in JavaScript used to represent a list.

Array - JavaScript中用於表示列表的資料結構。

## **New Code**

* [colorLeds](https://docs.code.org/applab/colorLeds/)
* [colorLeds[index].on()](https://docs.code.org/applab/on/)
* [colorLeds[index].blink(interval)](https://docs.code.org/applab/blink/)
* [colorLeds[index].intensity(brightness)](https://docs.code.org/applab/intensity/)
* [colorLeds[index].color(color)](https://docs.code.org/applab/color/)
* [list[index];](https://docs.code.org/applab/accessListItem/)

# [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/1)

# Introduction to Arrays 介紹陣列

# [Student Overview 給學生的概述](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-2-sfmd)

# [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/2)

# Introduction to Arrays 陣列介紹

# [3](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-3)

# [4](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-4)

# [5](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-5)

# [6](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-6)

# [7](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-7)

# (click tabs to see student view)

# [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/3)

### **Student Instructions 學生操作指南**

# **Predict 預測**

# The code below will display four things on the screen. What do you think they will be?

下面的代碼將在屏幕上顯示四個內容。 你覺得他們會是什麼樣的？

# [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/4)

### **Student Instructions 學生操作指示**

# **Index Practice 索引練習**

# 

# **Starter code** has been provided that creates a list called colors and uses it to set the colors of the rectangles on the screen. Unfortunately, the colors are out of order.

啟動代碼已經提供了創建一個稱為顏色的列表，並使用它來設置在屏幕上矩形的顏色。不幸的是，顏色的順序亂了。

# **Do This 做這個**

# Change the index numbers inside each selection block to make the rainbow appear as in the picture on the right. 更改每個選擇塊內的索引號，使彩虹顯示在右側圖片中。

# [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/5)

### **Student Instructions 學生操作指南**

# **Accessing Items in an Array 讀取陣列中的項目**

# This program uses the Circuit Playground. It creates an array of the days of the week. When the left button is pressed, it displays the day as "Tuesday".

該程式使用Circuit Playground。 它將一周中的每天創建了一個陣列。當按下左按鈕時，它將日期顯示為“星期二”。

# **Do This 做這個**

# Add code that will display the day as "Thursday" when the right button is pressed.

添加代碼，當按下右鍵時，該日期將顯示為“星期四”。

# [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/6)

### **Student Instructions 學生操作指南**

# **Random Selection 隨機選擇**

# You can also select a random element in the list by using the [randomNumber](https://docs.code.org/applab/randomNumber/) block instead of an index. This program selects a random icon from a list and displays it on the screen when the left button is pressed. Right now, the icons are always blue.

您還可以使用randomNumber塊而不是索引來選擇列表中的隨機元素。 該程式從列表中選擇一個隨機圖標，並在按下左鍵時在屏幕上顯示。 現在，圖標總是藍色的。

# **Do This 做這個**

# Add code that will select a random color from the list and use that as the icon color.

添加代碼，將在列表中選擇一個隨機的顏色，並使用它作為圖標的顏色。

# [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/7)

### **Student Instructions 學生操作指南**

# **Using Variables 使用變數**

# This program is like the last one, but the icons should always match the colors (red heart, yellow star, green leaf, and blue water). You can do this using the variable "myIndex", which will store the value of a random number so that it can be used to get the icon and the color from the same place on each list.

這個程式就像最後一個，但圖標應該總是與顏色相對應（紅心，黃星，綠葉和藍水）。 您可以使用變數“myIndex”來執行此操作，該變數將存儲隨機的數值，以便可以使用它來獲取每個列表上相同位置的圖標和顏色。

# **Do This 做這個**

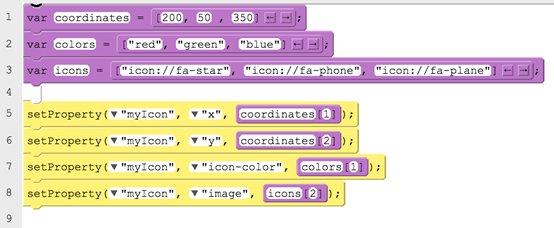
# Change the code so that both the icon and its color are chosen using the "myIndex" variable.

* 更改程式碼，以便使用“myIndex”變數選擇圖標及其顏色。

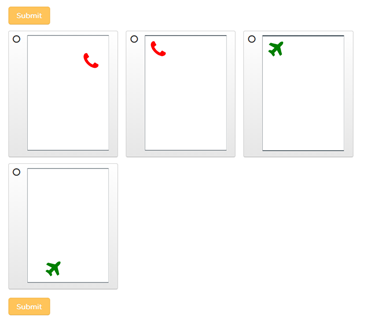
# Using Arrays 使用陣列

# [Student Overview](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-8-sfmd)

# [View on Code Studio](https://studio.code.org/s/csd6-2018/stage/10/puzzle/8)



* Circuit Playground: Color LEDs Circuit Playground:彩色LED
* [Student Overview](https://curriculum.code.org/csd-18/unit6/10/#level-expando-10-2-sfmd)



[View on Code Studio](https://studio.code.org/s/csd6/stage/10/puzzle/9)

* Color LEDs 彩色的LED
* [10](https://curriculum.code.org/csd-1718/unit6/10/#level-expando-10-10)
* [11](https://curriculum.code.org/csd-1718/unit6/10/#level-expando-10-11)
* [12](https://curriculum.code.org/csd-1718/unit6/10/#level-expando-10-12)
* [13](https://curriculum.code.org/csd-1718/unit6/10/#level-expando-10-13)
* [14](https://curriculum.code.org/csd-1718/unit6/10/#level-expando-10-14)
* (click tabs to see student view)
* [View on Code Studio](https://studio.code.org/s/csd6/stage/10/puzzle/10)

Student Instructions **學生操作指示**

****

**colorLEDs** **彩色LED**

This program uses the colorLEDs array, which lets you control the color LEDs in your Circuit Playground.

What do you think this code will do? Write your prediction in the box below, then run the code to see what happens.

這個程式是使用彩色LED陣列來進行編寫，而彩色LED能夠讓你在**Circuit Playground**中控制LED的顏色。

你覺得這段程式碼會怎樣執行呢? 在下面的框中寫下你的預測，然後執行程式碼，看看會發生什麼。

[View on Code Studio](https://studio.code.org/s/csd6/stage/10/puzzle/11)

**Student Instructions 學生操作指示**

**colorLeds 彩色的LED**

The color LEDs can use the same basic commands that you used with the red LED, such as [on()](https://docs.code.org/applab/on/) and [off()](https://docs.code.org/applab/off/). This program should turn on and off the bottom LEDs as shown in the picture, but only one LED is working.

彩色LED可以與紅色LED使用相同的基本指令，例如on（）和off（）。 該程式應該要打開和關閉圖片中最下方的LED，但只有一個LED能被執行。

**Do This 動手做**

* Read and run the code to see how the program turns the bottom left LED on and off.

讀取並執行程式碼，以查看程式如何打開和關閉左下方的LED。

* Add more code to make both bottom LEDs turn on and off.

添加更多程式碼以打開和關閉底部的LED。

[View on Code Studio](https://studio.code.org/s/csd6/stage/10/puzzle/12)

**Student Instructions 學生操作指示**

**Adding Some Color 添加一些顏色**

You probably noticed when you used colorLeds[0].on() in the last level, the LEDs turned on as white. If you don't specify a color, [on()](https://docs.code.org/applab/on/) will turn on all three parts of the LED (red, green, and blue) to produce white. You'll see a new command in the toolbox called colorLeds[0].color() which lets you specify a color, either as a string like "blue" or an RGB value using color [color()](https://docs.code.org/applab/color/) command.

您可能已經注意到，當您在最後一關使用彩colorLeds[0] .on（）時，LED會亮起為白色。 如果未指定顏色，on() 將會打開LED的所有三個顏色（紅色，綠色和藍色）來產生白色。 您將在工具箱中看到一個名為colorLeds [0].color（）的新指令，這個新指令允許您指定一個顏色，這個新指令可以為“blue”字串或使用顏色 [color()](https://docs.code.org/applab/color/) 指令的RGB值。

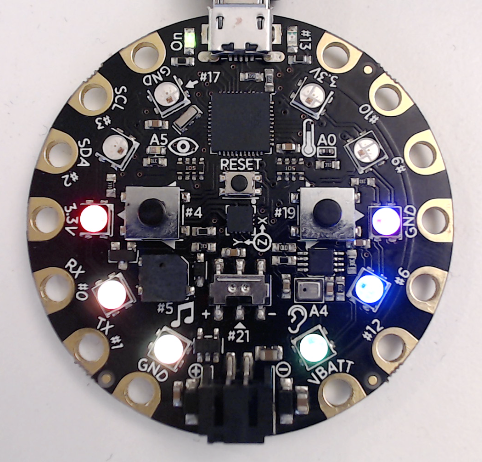
**Do This動手做**

Use the .color() command to set each of the color LEDs to a different color.

使用.color（）指令將每個彩色LED設置為不同的顏色。

[View on Code Studio](https://studio.code.org/s/csd6/stage/10/puzzle/13)

**Student Instructions 學生操作指示**

****

**Debug: colorLeds 除錯:彩色LED**

Can you debug the index values so that the rainbow shows up on the board like in the picture?

你可以嘗試調整索引值，如下圖中一樣，讓彩虹顯示在電路板上嗎?

[View on Code Studio](https://studio.code.org/s/csd6/stage/10/puzzle/14)

**Student Instructions 學生操作指示**

**Changing Brightness 改變明亮度**

The [intensity()](https://docs.code.org/applab/intensity/) command lets you set an LED's brightness (or *intensity*) from 0 - 100. This program lets you choose three levels of brightness for your rainbow lights, but the medium level doesn't work yet.

使用intensity() 指令可以將LED的亮度（或強度）設置為0到100之間.此程式可讓您選擇為彩虹燈選擇三個亮度級別，但中等級別還沒辦法執行。

**Do This 動手做**

* Read the program and see how the "Bright" and "Dim" buttons work.

閱讀程式並查看“亮”和“暗”按鈕如何運作。

* Add code to make the "Medium" button make the rainbow a medium level of brightness (between "Bright" and "Dim"). ([Show me where](https://curriculum.code.org/csd-1718/unit6/10/#triggercallout=code_triggered))

添加程式碼以使“中”按鈕讓彩虹成為中等亮度（在“亮”和“暗”之間）。 （告訴我在哪裡）

* Light Show 燈光秀
* [15](https://curriculum.code.org/csd-1718/unit6/10/#level-expando-10-15)
* [16](https://curriculum.code.org/csd-1718/unit6/10/#level-expando-10-16)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6/stage/10/puzzle/15)

**Student Instructions 學生操作指示**

**Light Patterns 燈光模式**

This app already has one light pattern program, which can be activated by clicking a button. Notice that we put this pattern into a function to make the code cleaner and clearer.

這個應用程式已經有一個可以透過單擊按鈕來執行的燈光模式程式。

請注意，我們將此模式放入函數中以使程式碼越來越清晰。

**Do This 動手做**

Create a light pattern function of your own - maybe the colors of your favorite holiday, college, sports team, or try to create an image with the lights.

After you've created your new function, add a button and event handler so that you can switch between the two light patterns.

創建你自己的燈光模式功能 - 也許是您最喜歡的假期，大學，運動隊的顏色，或嘗試創立一個有燈光的圖像。

創立你自己的新功能後，添加一個按鈕和事件處理程序，以便可以在兩種指示燈模式之間切換。

[View on Code Studio](https://studio.code.org/s/csd6/stage/10/puzzle/16)

**Student Instructions 學生操作指示**

**Turning It Off 關掉它**

When you start this app your lights are off, but as soon as choose a pattern, there's no way to turn the lights back off!

當你啟動這個應用程式時，你的指示燈熄滅，但是一旦選擇了一個模式，就無法關閉燈光！

**Do This 動手做**

Create a function that turns all of the LEDs off, and then add a "Off" button to your app and make it turn off all of the color LEDs.

創立一個能關閉所有LED的功能，然後為您的應用程式添加“關閉”按鈕，並關閉所有彩色LED。

**Wrap Up (5 min) 總結**

**Discussion: Objects Vs Arrays 討論: 物件vs 陣列**

Content Corner 補充說明

Arrays are frequently used to collect items of the same type (a bunch of numbers, a bunch of strings, a bunch of button names, etc), but in JavaScript all list items need not be of the same type. You can create lists the include numbers, strings, objects, and even other lists.

陣列經常用於收集相同類型的項目（一堆數字，一串字串，一堆按鈕名稱等），但在JavaScript中，所有列表項不必是相同類型。 您可以創建一個同時包含數字，字符串，物件甚至其他項目的列表。

An Object is a data structure that combines many values (such as numbers, strings, and even functions) that represent the attributes of that object (height, width, rotation, name) which are easier to keep track of if done in one place. The attributes can be of many different types but they all describe one object.

物件則是一種數據結構，它結合了許多值（例如數字，字串甚至函數），這些值代表了該物件的屬性（高度，寬度，旋轉，名稱），有了這些屬性，我們能更容易來追蹤這些物件。 屬性可以有許多不同的類型，但它們都是用來描述物件。

**Prompt:** You have now seen both objects, such as the Sprite object, and arrays. How is an array different from the attributes of an object? When would you use an array? When would you use an object?

提示：您現在已經看到了兩種物件，例如Sprite物件和陣列。 陣列與物件的屬性有何不同？ 你什麼時候使用陣列？ 你什麼時候使用物件？

You can either use this as an exit ticket or have a class discussion if you have time.

如果您有時間，可以將其作為驗收門檻或在課堂進行討論。

**Standards Alignment 標準化**

**[View full course alignment](https://curriculum.code.org/csd-1718/standards/)**

**CSTA K-12 Computer Science Standards 運算科學的標準**

**AP** - Algorithms & Programming 演算法與編成

* **2-AP-11** - Create clearly named variables that represent different data types and perform operations on their values.

2-AP-11 - 創立能夠表示不同數據類型的明確命名變量，並對其值執行操作。

* **2-AP-13** - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-13 - 將問題和子問題分解成多個部分，以便於程序的設計，執行和檢查

* **2-AP-16** - Incorporate existing code, media, and libraries into original programs, and give attribution.

2-AP-16 - 將現有代碼，媒體和庫合併到原始程式中，並給予屬性。

* **2-AP-19** - Document programs in order to make them easier to follow, test, and debug.

2-AP-19 - 文檔程序，以便於追蹤，測試和除錯。

**CS** - Computing Systems計算系統

* **2-CS-02** - Design projects that combine hardware and software components to collect and exchange data.
* 2-CS-02 - 設計項目，結合硬件和軟件組件來收集和交換數據。

**Lesson Feedback**

&lt;i class='fa fa-cog fa-spin' style="font-size: 100px; text-align: center"&gt;&lt;/i&gt;

Close

If you are interested in licensing Code.org materials for commercial purposes, [contact us.](https://code.org/contact)



# If you are interested in licensing Code.org materials for commercial purposes, [contact us.](https://code.org/contact)

# **Lesson 11: Making Music 製作音樂**

#### **App Lab | Maker Toolkit**

## **Overview** 課程概述

In this lesson students will use the buzzer to its full extent by producing sounds, notes, and songs with the buzzer. Students start with a short review of the buzzer's frequency and duration parameters, then move on to the concept of notes. Notes allow students to constrain themselves to frequencies that are used in Western music and provide a layer of abstraction that helps them to understand which frequencies might sound good together. Onces students are able to play notes on the buzzer, they use arrays to hold and play sequences of notes, forming simple songs.

在本課程中，學生將通過蜂鳴器產生聲音，音符和歌曲。 學生首先簡要回顧一下蜂鳴器的頻率和持續時間參數，然後繼續學習音符的概念。 允許學生將音符約束在西方音樂中使用的頻率，並提供一層抽象，幫助他們理解哪些頻率可能聽起來很好。 一旦學生能夠在蜂鳴器上彈奏音符，他們就會使用數組來保持和播放音符序列，形成簡單的歌曲。

## **Purpose** 課程目的

This lesson allows students to get more creative with the buzzer by introducing the concept of a "note" and by using arrays to hold sequential collections of notes. Using notes rather than frequencies provides a layer of abstraction that makes it easier for students to identify sounds that can be used to make music. Arrays provide a way to group together sounds in sequence, so that they can be played as music. Using the playNotes block to iterate over an array of notes provides a conceptual foundation for working with for loops in the next lesson.

本課程通過引入音符的概念和使用數組來保存連續的音符集，使學生能夠通過蜂鳴器獲得更多創意。 使用音符而不是頻率提供了一層抽象，使學生更容易識別可用於製作音樂的聲音。 數組提供了一種按順序將聲音組合在一起的方法，以便它們可以作為音樂播放。 使用playNotes塊迭代一系列註釋，為下一課中使用for循環提供了概念基礎。

## **Agenda** 課程流程

* [Warm-Up (5 min)](https://curriculum.code.org/csd-18/unit6/11/#making-music0) 暖身活動 (5 min)
* [Color LEDs vs the Buzzer 彩色LED與蜂鳴器](https://curriculum.code.org/csd-18/unit6/11/#making-music1)
* [Activity 活動](https://curriculum.code.org/csd-18/unit6/11/#making-music2)
* [Making Music 製作音樂](https://curriculum.code.org/csd-18/unit6/11/#making-music3)
* [Wrap-Up (5 min) 總結（5分鐘）](https://curriculum.code.org/csd-18/unit6/11/#making-music4)
* [Journal 日誌](https://curriculum.code.org/csd-18/unit6/11/#making-music5)

### **[View on Code Studio 在Code Studio上查看](https://studio.code.org/s/csd6-2018/stage/11/puzzle/1/)**

## **Objectives 目標**

### **Students will be able to: 學生將能夠**

* Create and modify an array 創建和修改數組
* Use an array to produce sound on the buzzer 使用陣列在蜂鳴器上產生聲音
* Recognize an array as a list of elements that can be operated on sequentially. 將數組識別為可以按順序操作的元素列表。

## **Introduced Code 介紹代碼**

* [buzzer.note(note, duration)](https://docs.code.org/applab/buzzer.note/)
* [buzzer.playNotes(array, tempo)](https://docs.code.org/applab/buzzerplaynotes/)
* [list.length](https://docs.code.org/applab/listLength/)
* [var list = ["a","b","d"];](https://docs.code.org/applab/declareAssign_list_abd/)

## **Support 支持**

### **[Lesson Forum](https://forum.code.org/tags/c/csd/csd-unit-6-lesson-11)**

### **[Report a Bug](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%2011%20curriculum.code.org/csd-18/unit6/11/)**

# **Teaching Guide 教學指南**

## **Warm-Up (5 min)** 暖身活動 (5 min)

### **Color LEDs vs the Buzzer** 彩色LED與蜂鳴器

Discussion Goal 討論目標

Students may come up with many different ideas, but there are two aspects of this problem that should be highlighted before students move on to the activity. First, specifying frequencies is not the best way to indicate how high or low a buzz should be, since music only uses a limited number of frequencies, and other frequencies are considered "out of tune". Music is usually written down as specific notes on a scale, which makes it much easier to see which sounds will go well together. (Students with a musical background will likely have a much better grasp of this issue.) Second, students will need a way to string sounds, or notes, together to make a song. This second point should be more universally understood.

學生可能會提出許多不同的想法，但在學生進入活動之前，應該強調這個問題的兩個層面。首先，指定頻率不是指示嗡嗡聲應該有多高或多低的最佳方式，因為音樂僅使用有限數量的頻率，而其他頻率被認為是“失調”。音樂通常被寫為音階上的特定音符，這使得更容易看到哪些聲音將很好地融合在一起。 （具有音樂背景的學生可能會更好地掌握這個問題。）其次，學生需要一種將聲音或音符串起來製作歌曲的方法。應該更普遍地理解第二點。

**Prompt:** We've been using the buzzer to make sounds, but those buzzes didn't always sound too great. What do you think you need to make real music on the buzzer?

提示：我們一直在使用蜂鳴器發出聲音，但那些嗡嗡聲並不總是聽起來太棒了。您認為在蜂鳴器上製作真正的音樂需要什麼？

## **Activity 活動**

### **Making Music 製作音樂**

**Transition:** Send students to Code Studio.

轉換：將學生送到寫程式的工作室

### **Code Studio levels**寫程式的工作室的層級

* Lesson Overview　課程概論
* [Student Overview](https://curriculum.code.org/csd-18/unit6/11/#level-expando-11-1-sfmd)

## **Overview 概觀**

In this lesson students will use the buzzer to its full extent by producing sounds, notes, and songs with the buzzer. Students start with a short review of the buzzer's frequency and duration parameters, then move on to the concept of notes. Notes allow students to constrain themselves to frequencies that are used in Western music and provide a layer of abstraction that helps them to understand which frequencies might sound good together. Once students are able to play notes on the buzzer, they use arrays to hold and play sequences of notes, forming simple songs.

在本課程中，學生將通過生產的聲音，筆記，和歌曲與蜂鳴器用蜂鳴器以充分發揮其程度。 學生首先簡要回顧一下蜂鳴器的頻率和持續時間參數，然後繼續學習音符的概念。 Notes允許學生將自己限制在西方音樂中使用的頻率，並提供一層抽象，幫助他們理解哪些頻率可能聽起來很好。 一旦學生能夠在蜂鳴器上彈奏音符，他們就會使用數組來保持和播放音符序列，形成簡單的歌曲。

## **New Code**

* [buzzer.note(note, duration)](https://docs.code.org/applab/buzzer.note/)
* [buzzer.playNotes(array, tempo)](https://docs.code.org/applab/buzzerplaynotes/)
* [list.length](https://docs.code.org/applab/listLength/)
* [var list = ["a","b","d"];](https://docs.code.org/applab/declareAssign_list_abd/)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/1)

* Making Sounds　製作聲音
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/2)

### **Student Instructions　學生指引**

# **Circuit Piano　鋼琴線路**

Look at the code below. You're not going to be changing it, but try to see what it does.

看看下面的程式. 不用去更改它, 但是請試著去看看它是如何運作的

# **Do this　做這個**

* Run the program and play with the piano on the screen.

執行這個程式並且在螢幕上彈琴

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/3)

### **Student Instructions　學生指引**

# **Making the Buzzer Buzz　製作蜂鳴器嗡嗡聲**

The frequency of the buzzer determines how high or how low the buzzer will sound when it plays. The higher the number you give frequency, the higher the buzzer will sound.

蜂鳴器的頻率決定了蜂鳴器播放時的聲音有多高或多低。 您給出的頻率越高，蜂鳴器的聲音就越高。

# **Do this做這個**

* Right now the buzzer plays a high note when the left button is pressed. Make the buzzer play a low note instead.

現在，當按下左按鈕時，蜂鳴器會發出高音。 讓蜂鳴器發揮低音。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/4)

### **Student Instructions學生指引**

# **Making the Buzzer Buzz製作蜂鳴器嗡嗡聲**

Now that you can use the buzzer's frequency, try to make the buzzer play a high sound when the same button pops back up.

現在您可以使用蜂鳴器的頻率，當相同的按鈕彈回時，嘗試讓蜂鳴器播放高音。

# **Do this做這個**

* Add another onBoardEvent block so the buzzer plays a high sound when the left button comes back **up**

添加另一個在板子上的事件區塊以便當左按鈕重新啟動時蜂鳴器發出高音

* Making Music**製作音樂**
* [Student Overview](https://curriculum.code.org/csd-18/unit6/11/#level-expando-11-5-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/5)

Today you’re going to learn how to use and create **functions**.

Creating a function lets you make your own blocks! There are two parts to a function:

Making a newly named block that has all of the code you want to run inside of it. -xml of function definition-

The single green block itself that you use, or call, to run the function you created. -xml of function call-

Prepare to get familiar with functions, learn how to edit them, and create your own!

今天，您將學習如何使用和創建功能。

創建一個函數可以讓你自己創建塊！ 功能分為兩部分：

創建一個新命名的塊，其中包含您要在其中運行的所有代碼。 -xml的函數定義 -

您使用或調用的單個綠色塊本身，用於運行您創建的功能。 -xml函數調用 -

準備熟悉功能，學習如何編輯它們，並創建自己的功能！

* Playing with Notes　照著筆記撥放音樂
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/6)

### **Student Instructions學生指引**

# **Piano Notes鋼琴筆記**

This is the same piano that you saw at the beginning of the lesson, but there are some bugs!

這是你在課程開始時看到的同一架鋼琴，但也有一些錯誤！

# **Do this做這個**

* Edit the code so that all keys are using a [buzzer.note()](https://docs.code.org/applab/buzzer.note/) block to play the corresponding note.

•編輯代碼，以便所有鍵都使用buzzer.note（）塊來播放相應的音符。

* Make sure that all the values passed in to [buzzer.note()](https://docs.code.org/applab/buzzer.note/) are notes, and not frequencies.

確保傳入buzzer.note（）的所有值都是註釋，而不是頻率。

*Hint: Remember to put quotes around the notes! e.g* [*buzzer.note("A4", 100);*](https://docs.code.org/applab/buzzer.note/)

提示：記得在筆記周圍加上引號！ 例如buzzer.note（“A4”，100）;

* Modifying Arrays修改陣列
* [Student Overview](https://curriculum.code.org/csd-18/unit6/11/#level-expando-11-7-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/7)

Today you’re going to learn how to use and create **functions**.

Creating a function lets you make your own blocks! There are two parts to a function:

Making a newly named block that has all of the code you want to run inside of it. -xml of function definition-

The single green block itself that you use, or call, to run the function you created. -xml of function call-

Prepare to get familiar with functions, learn how to edit them, and create your own!

今天，您將學習如何使用和創建功能。

創建一個函數可以讓你自己創建塊！ 功能分為兩部分：

創建一個新命名的塊，其中包含您要在其中運行的所有代碼。 -xml的函數定義 -

您使用或調用的單個綠色塊本身，用於運行您創建的功能。 -xml函數調用 -準備熟悉功能，學習如何編輯它們，並創建自己的功能！

* Musical Arrays音樂陣列
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/8)

### **Student Instructions**

# **Piano Panic鋼琴練習**

So far you’ve just been using the elements of an array. Now, you're going to change what's inside the array.

Our good friend the piano is back, except now it plays notes from two arrays: notes, which holds natural notes (C, D, E, F, G, A, B), and sharpNotes, which holds sharp notes (C#, D#, F#, G#, A#).

到目前為止，您剛剛使用了數組的元素。 現在，您將要更改數組中的內容。

我們的好朋友鋼琴回來了，除了它現在播放兩個陣列的音符：音符，其中包含自然音符（C，D，E，F，G，A，B）和sharpNotes，它們具有清晰音符（C＃，D＃） ，F＃，G＃，A＃）。

# **Do this做這個**

The sharpNotes array only has one item inside of it right now.

* Fix the array so that the sharp keys on the piano can actually play the right note.

*Hint: Modifying arrays is much easier in text mode. You can always switch between block and text mode by clicking the button at the top right of your code workspace.*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/9)

sharpNotes數組現在只有一個項目。

•修復陣列，使鋼琴上的尖銳鍵實際上可以播放正確的音符。

提示：在文本模式下修改數組要容易得多。 您始終可以通過單擊代碼工作區右上角的按鈕在塊模式和文本模式之間切換。

在Code Studio上查看

### **Student Instructions學生指引**

# **Random Song Generator隨機歌曲生成器**

You're going to make a random song generator. When you press a button, a random note should play. Press it enough times, and you have yourself a beautiful arrangement of bleeps and bloops.

You used [randomNumber()](https://docs.code.org/gamelab/randomNumber/) to select an element from an array in the previous lesson, so this should be familiar.

你要製作一個隨機的歌曲生成器。 按下按鈕時，應播放隨機音符。 按下它足夠的次數，你自己有一個美麗的嗶嗶聲的安排。

您使用randomNumber（）從上一課中的數組中選擇一個元素，因此這應該是熟悉的。

# **Do This做這個**

* Add code to randomly select a note from the notes array and play it.
* Run the program and press the left button to hear your buzzer sing its beautiful tune.

添加程式代碼以從notes數組中隨機選擇一個音符並播放它。

運行程式並按左按鈕，聽到蜂鳴器唱出美妙的曲調。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/10)

### **Student Instructions**

# **Making New Arrays製造新的陣列**

You know how to change arrays, now it's time for you to make your own.

你知道如何更改陣列，現在是你自己創建數組的時候了。

# **Do this做這個**

* Make a new array with whatever notes you want. They can be in different octaves, arranged in a certain key, etc. Anything you want!
* Have the buzzer play a random note from your array when the **right** button is pressed.
* Making Songs

•使用您想要的任何註釋創建一個新陣列。 它們可以是不同的八度音程，按某個鍵排列等等。你想要的任何東西！

•按下右鍵時，蜂鳴器會從陣列中隨機播放音符。

•製作歌曲

(click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/11)

### **Student Instructions學生指引**

Markdown is in contained level

# **Make a Prediction做一個預測**

Look at the code below. What do you think it's going to do?

Give it a guess and run the code to find out!

看下面的代碼。 你覺得它會怎麼樣？

給它一個猜測並運行代碼來找出答案！

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/12)

### **Student Instructions學生指引**

# **Making Songs製作歌曲**

You can make and play full songs with [buzzer.playNotes()](https://docs.code.org/applab/buzzerplaynotes/). It takes two parameters: an array of notes to play, and a tempo at which to play the notes. In previous bubbles you created arrays of notes and played through them randomly, but [buzzer.playNotes()](https://docs.code.org/applab/buzzerplaynotes/) plays each note consecutively instead.

您可以使用buzzer.playNotes（）製作和播放完整的歌曲。 它需要兩個參數：要播放的音符數組和播放音符的速度。 在之前的氣泡中，您創建了音符陣列並隨機播放，但是buzzer.playNotes（）會連續播放每個音符。

# **Do this做這個**

* Pull out a [buzzer.playNotes](https://docs.code.org/applab/buzzerplaynotes/) block and change the starting array inside.
* Make sure the new array you give it has at least two pairs of notes that are next to each other in the list (Example: ["A5", "A5", "G4", "G4"])

*Hint: Don't forget that you can always switch back and forth between block and text mode.*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/13)

•拉出一個buzzer.playNotes區塊並更改內部的起始陣列。

•確保您給出的新陣列在列表中至少有兩對彼此相鄰的音符（例如：[“A5”，“A5”，“G4”，“G4”]）

提示：不要忘記您始終可以在區塊和文本模式之間來回切換。

在Code Studio上查看

### **Student Instructions學生指引**

# **Null Notes空的註解**

You may have noticed that if you put the same note side-by-side in the array, the buzzer just plays that note as one long sound instead of as individual notes. In music we often want a break (or *rest*) between notes. You can get your buzzer to play a rest by sending it the value null.

null is a special value that represents emptiness in computer science. playNotes plays each note for the same duration, so a null element in the array is just telling playNotes to play nothing for the same duration as everything else.

您可能已經註意到，如果您在陣列中並排放置相同的音符，則蜂鳴器只會將該音符作為一個長音而不是單個音符播放。 在音樂中，我們經常希望在音符之間中斷（或休息）。 您可以通過向其發送值null來讓您的蜂鳴器休息。

null是一個特殊值，表示計算機科學中的空虛。 playNotes播放每個音符的持續時間相同，因此數組中的null元素只是告訴playNotes在與其他所有內容相同的持續時間內不播放任何內容。

# **Do this做這個**

* Run the code to hear what it sounds like first.
* Examine the array inside [buzzer.playNotes()](https://docs.code.org/applab/buzzerplaynotes/).
* Put a pause between notes that are the same and side-by-side, and any other place you deem appropriate.

*Tip: null references a value, just like a variable name. Make sure you don't put it in quotation marks!*

*•運行代碼以首先聽到它的聲音。*

*•檢查buzzer.playNotes（）中的數組。*

*•在相同且並排的音符之間以及您認為合適的任何其他位置之間暫停。*

*提示：null引用一個值，就像變量名一樣。 確保你沒有把它放在引號中！*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/puzzle/14)

### **Student Instructions學生指引**

# **Sound Board 2.0聲音板子2.0**

You can now play sounds, notes, and songs! With this new knowledge, you can even turn your board into a sound board.

In the toolbox you have all of the buzzer and led-related blocks at your disposal. See what cool stuff you can come up with!

您現在可以播放聲音，音符和歌曲！ 有了這些新知識，您甚至可以將您的電路板變成音板。

在工具箱中，您可以使用所有與蜂鳴器和LED相關的塊。 看看你能想出什麼很酷的東西！

# **Do This做這個**

* Use all of the button related board events (up, down), to play a unique sound or song at each event.
* Are there any other events you could use to play even more sounds/songs?
* Levels
* [Extra](https://curriculum.code.org/csd-18/unit6/11/#level-expando-11-15)
* [Extra](https://curriculum.code.org/csd-18/unit6/11/#level-expando-11-16)
* (click tabs to see student view)

•使用所有與按鈕相關的電路板事件（向上，向下），在每個事件中播放獨特的聲音或歌曲。

•是否還有其他可用於播放更多聲音/歌曲的活動？

•（單擊選項卡查看學生視圖）

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/extras)

### **Student Instructions學生指引**

# **Challenge: 2D Arrays 2D陣列**

Arrays can hold all sorts of data, like numbers, strings, and even other arrays. When we put an array inside another, this becomes a 2D array. The [buzzer.playSong()](https://docs.code.org/applab/buzzerplaysong/) block can be used to play songs that sound more precise with 2D arrays.

The structure that you've been using to play notes looks like this:

[buzzer.playNotes( [array of notes to play] , tempo of the whole song);](https://docs.code.org/applab/buzzerplaynotes/)

The main difference with [buzzer.playSong()](https://docs.code.org/applab/buzzerplaysong/) is each element inside the [array of notes to play], is an array with two elements: the note you want to play, and the duration of the note. Sometime you'll want a quarter note in your song, other times you'll want a full note, or maybe somewhere in between. A full song might look something like this:

數組可以包含各種數據，如數字，字符串甚至其他數組。 當我們將一個數組放入另一個數組時，它就變成了一個2D數組。 buzzer.playSong（）塊可用於播放使用2D陣列聽起來更精確的歌曲。

您用來播放筆記的結構如下所示：

buzzer.playNotes（[要播放的音符陣列]，整首歌的節奏）;

與buzzer.playSong（）的主要區別在於[要播放的音符數組]中的每個元素，是一個包含兩個元素的數組：要播放的音符和音符的持續時間。 有時你會在你的歌曲中想要一個四分音符，有時你會想要一個完整音符，或者介於兩者之間。 一首完整的歌曲可能看起來像這樣：

# **Do this做這個**

* Use the [buzzer.playSong](https://docs.code.org/applab/buzzerplaysong/) block to make a different song.

使用buzzer.playSong塊製作不同的歌曲。

* When you get comfortable with the structure, add more notes to the song.

•當您對結構感到滿意時，請為歌曲添加更多音符。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/11/extras)

### **Student Instructions學生指引**

# **Stopping the Music停止音樂**

Sometimes you need to be able to stop the music when you want, like in the code below. Duration wasn't specified so now the note plays continuously. Luckily, the [buzzer.stop()](https://docs.code.org/applab/buzzer.stop/) method is here to save your ears.

有時你需要能夠在你想要的時候停止音樂，就像下面的代碼一樣。 沒有指定持續時間，所以現在筆記連續播放。 幸運的是，buzzer.stop（）方法可以節省你的耳朵聽力。

# **Do this做這個**

* Find a way to get the note to stop playing with [buzzer.stop()](https://docs.code.org/applab/buzzer.stop/).
* Celebrate your win over the endless note!

•找到一種讓筆記停止播放buzzer.stop（）的方法。

•慶祝您贏得無盡的音符！

## **Wrap-Up (5 min) 熱身 (五分鐘 )**

### **Journal**

**Goal:** Help students make the connection between the playNotes block and the for loops that they will use in the next lesson.

**Prompt:** Today, instead of just choosing one element from an array, we used the playNotes block to do something to every element. Think back to the other arrays you have seen. How might doing something to every element in an array be useful there?

**Share** After giving students time to reflect in their own journals, have them share with a partner or in small groups.

日誌

目標：幫助學生在playNotes塊和他們將在下一課中使用的循環之間建立連接。

提示：今天，我們使用playNotes塊對每個元素執行某些操作，而不是僅從陣列中選擇一個元素。 回想一下你見過的其他陣列。 怎麼對陣列中的每個元素去做一些有用的事情？

分享: 給予學生時間反思他們自己的作品後，讓他們與合作夥伴或小組分享。

# **Lesson 12: Arrays and For Loops**

#### App Lab | Maker Toolkit

# **Overview 課程概要**

Using a for loop to iterate over all of the elements in an array is a really useful construct in most programming languages. In this lesson, students learn the basics of how a for loop can be used to repeat code, and then combine it with what they've already learned about arrays to write programs that process all elements in an array. Students use for loops to go through each element in a list one at a time without having to write code for each element. Towards the end of the lesson students will apply this with the colorLed list on the board to create an app that changes all of the LEDs each time a button

在大多數程式語言中，用 for 迴圈來訪問陣列中的所有元素是個很好用的方式。在本課程中，學生會學到如何使用 for 迴圈來重複執行代碼。接著再搭配他們已經學過的陣列來編寫程式處理陣列中的所有元素。學生可以使用 for 迴圈一次一個地走過列表中的每個元素，而不用為每個元素單獨編寫程式碼。 在本課程的後半，學生將運用這個方法撰寫一個程式，在每次有按鈕被按下時，能夠改變板子上所有LED的顏色。

## **Purpose 課程目的**

As students start using arrays more frequently, a common pattern emerges wherein you want to run some code on each element of an array. While for loops are a generally useful structure for repeating code, they are a particularly useful for iterating over an array. In this lesson we build on the understanding of arrays that students have developed in the last two lessons by introducing the for loop, which combines a variable, the counter pattern, and a conditional all in a single construct.  
  
在學生們開始經常使用陣列之後，他們就會發現一種用來對陣列中各項元素執行某些程式碼的共通方式。雖然一般來說， for 迴圈在重複執行程式碼上是一種很有用的方式，但它在訪問整個陣列時特別好用。 在本堂課中，我們會在前兩堂課學過的陣列基礎上，進一步介紹 for 迴圈的結構，它是一種包含了一個變數，計算模式，以及結束條件的程式結構。

## **Agenda 課程大綱**

* **[Warm Up (5 min) 暖身 (５分鐘）](https://curriculum.code.org/csd-18/unit6/12/#arrays-and-for-loops0)**
* [Run Code on All Elements of an Array 運行一個陣列中的所有元素](https://curriculum.code.org/csd-18/unit6/12/#arrays-and-for-loops1)
* **[Activity (40 min) 活動（40 分鐘）](https://curriculum.code.org/csd-18/unit6/12/#arrays-and-for-loops2)**
* [Arrays and For Loops 陣列與 for 迴圈](https://curriculum.code.org/csd-18/unit6/12/#arrays-and-for-loops3)
* **[Wrap Up 小結](https://curriculum.code.org/csd-18/unit6/12/#arrays-and-for-loops4)**
* [Journal 日誌](https://curriculum.code.org/csd-18/unit6/12/#arrays-and-for-loops5)

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/12/puzzle/1/)**

## **Objectives 課程目標**

### **Students will be able to:**

* Modify the exit condition of a for loop to control how many times it repeats
* Use a for loop to iterate over an array

學生將能夠：  
1. 修改一個 for 迴圈的退出條件，進而控制迴圈重複的次數  
2. 使用一個 for 迴圈來重複一個陣列

# **Vocabulary 詞彙**

* Array - A data structure in JavaScript used to represent a list. 陣列 – 一個 JavaScript 用來表達清單的資料結構
* For Loop - Loops that have a predetermined beginning, end, and increment (step interval). For 迴圈 - 事先定義了起始、結尾與增量（步驟區間）的迴圈

# **Introduced Code 介紹程式碼**

* [for(var i=0; i<4; i++){ //code }](https://docs.code.org/applab/forLoop_i_0_4/)
* [function myFunction(n){ //code }](https://docs.code.org/applab/functionParams_n/)
* [Call a function with parameters](https://docs.code.org/applab/callMyFunction_n/)

## **Support**

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

### **[Report a Bug](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%2012%20curriculum.code.org/csd-18/unit6/12/)**

# **Teaching Guide 教學指南**

# **Warm Up (5 min) 暖身**

### Run Code on All Elements of an Array 運行一個陣列中的所有元素

Discussion Goal 討論目標

You shouldn't expect students to know how exactly a computer would do this, but to start thinking about what is needed to keep track of where you are in an array (the index), how you would know you've reached the end of the array (the array length), and how you might incrementally increase where you are in the list (the counter pattern).  
  
您不應該期待學生知道電腦如何實際運作此程式碼，但應該期待他們能夠開始思考需要什麼來追蹤自己位於陣列的位置、要如何知道自己已經達到陣列的結尾 （陣列的長度）跟清單中的數值將如何逐漸增加（計數模式）

Discuss: If you had a list full of all of your ToDos and you wanted the computer to print out each one, how might you do it? Don't worry about the specific code, focus on the what information your program would need to know and keep track of. Would your approach still work if you added or removed a ToDo from your list?  
  
討論：如果你有一個記滿你的代辦事項的清單，你想要透過電腦將待辦事項各自印出，您會怎麼做呢？不需要擔心特定的程式碼，專注思考你的程式需要知道並追蹤什麼樣的資訊。如果你將你的代辦事項移出一條項目，你的方法能成功運作嗎？

***Remarks 備註***

Doing something to every element of an array is a really common problem in Computer Science, and figuring out how to solve this problem will let us write more efficient programs. Today we're going to focus on three things to help us with this problem  
  
在電腦科學中，如何對於陣列中的每一項元素採取一些行動是一個很普遍的問題，思考如何解決這個問題能夠幫助我們寫出一個有效率的程式。今天，我們將要著重三件事情來幫助我們解決這個問題：

1. Using the index to do something to elements in an array   
   使用指數來對陣列中的元素採取行動
2. Keeping track of a counter so we can move through the elements of an array by index  
   追蹤一個計數器，讓我們能夠透過指數移動一個陣列元素
3. Using the length of an array to know when we've reached the end  
   使用陣列長度來理解我們什麼時候抵達陣列的結尾

# **Activity (40 min) 活動**

### Arrays and For Loops 陣列與 For 迴圈

Transition: Send students to Code Studio 轉換：將學生送到程式碼教室

### Code Studio levels 程式碼教室層級

* **Lesson Overview 課程概要**
* [Student Overview](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-1-sfmd) 學生概要

# **Overview 概要**

Combining \_lists\_ and \_for loops\_ allows you to write code that impacts every element of a list, regardless of how long it is. The class uses this structure to write programs that process all of the elements in lists, include the list of color LEDs.  
Combining \_lists\_ and \_for loops\_ 使你能夠寫出影響每個清單元素的程式碼，不論清單多長。這堂課使用這個架構來寫出能處裡清單中所有元素的程式，包含彩色 LEDs 清單。

# **Vocab 詞彙**

* Array - A data structure in JavaScript used to represent a list. 陣列 – 一個 JavaScript 用來表達清單的資料結構
* For Loop - Loops that have a predetermined beginning, end, and increment (step interval). For 迴圈 - 事先定義了起始、結尾與增量（步驟區間）的迴圈

# **New Code 新程式碼**

* [for(var i=0; i<4; i++){ //code }](https://docs.code.org/applab/forLoop_i_0_4/)
* [function myFunction(n){ //code }](https://docs.code.org/applab/functionParams_n/)
* [Call a function with parameters](https://docs.code.org/applab/callMyFunction_n/)
* **Programs that Repeat 重複的程式**
* [2](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-2)

### Student Instructions 學生指示

# **Make a Prediction 做出預測**

* + Read through the code for this program and predict what will happen each time the button is clicked? 讀完這個程式所有的程式碼，然後預測每次按下按鍵會發生什麼事情？
* [3](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-3)

### Student Instructions 學生指示

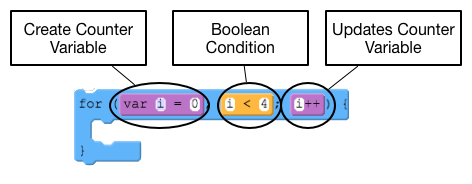
# **Knowing When to Stop 知道什麼時候該停下來**

* + If you clicked the button too many times in the last level, you got an error. Whenever you're writing code that repeats, you should think about when to stop repeating.  
    當你按太多次最後一個階層的按鍵，你會得到一個錯誤提示。每當你在寫會重復的程式碼時，請進一步思考什麼時候停止重複。

# **Do This 請這麼做**

* + This program is similar to the previous one, but there is a conditional inside the event handler. You'll need to complete the conditional so that we don't try to toggle an LED that doesn't exist. 這個程式跟前一個程式非常相似，但每個處理器裡面都有一個條件式。為了不試圖切換不存在的 LED，你需要完成每一個條件式。
  + Would your code work without changes for a board with more or fewer LEDs? If not, could you modify it so it would?   
    當板子上的 LED 燈變多或變少時，你的程式碼是否能持續不變地發揮作用？如果不行的話，你能修改程式碼，讓它能正常運作嗎？
* **(click tabs to see student view)**
* **For Loops For 迴圈**
* [Student Overview](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-4-sfmd)

# **For LoopsFor 迴圈**

* + It's common to want to repeat a set of commands a particular number of times. The for loop was created to wrap all of those components related to counting loops into a single line of code.  
    重複一組命令數次是件很普遍的事情。For 迴圈本身是用來將所有計數迴圈相關的元件包裝成一行程式碼。
  + 
  + Programmers would typically read a loop for (var i = 0; i < 10; i++) out like this:
  + "for variable i starting at 0, while i is less than 10, increment i by 1"  
    工程師一般來說會用以下讀法來讀出迴圈 for (var i = 0; i < 10; i++)：  
    “從 0 開始的變數 i，當 i 小於 10 時，每次將 i 增加 1”

# **Three Parts of For Loop For 迴圈的三部分**

### Initialize Variable 初始化變數

* + Whenever you are counting, you need a variable to keep track of the count. The first part of the for loop sets up the variable, often called i, that will be used to count the number of times the loop will run. This also sets up the starting value at which to start counting. The variable is set up before the loop is  
    當你開始計數時，你需要一個變數去追蹤次數。迴圈的第一部分要定義變數（通常稱為 i），變數會用來計算迴圈運行的次數。這個變數也設定了迴圈開始計算的起始數值。這個變數需要在迴圈前被設定。(???)
* **CSD: For Loop\_2018**
* [Student Overview](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-5-sfmd)
* **For Loops**
* [6](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-6)

### Student Instructions 學生指示

# **Predict 預測**

* + What will print as a result of this loop? 迴圈的結果出來會什麼呢？
* [7](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-7)

### Student Instructions

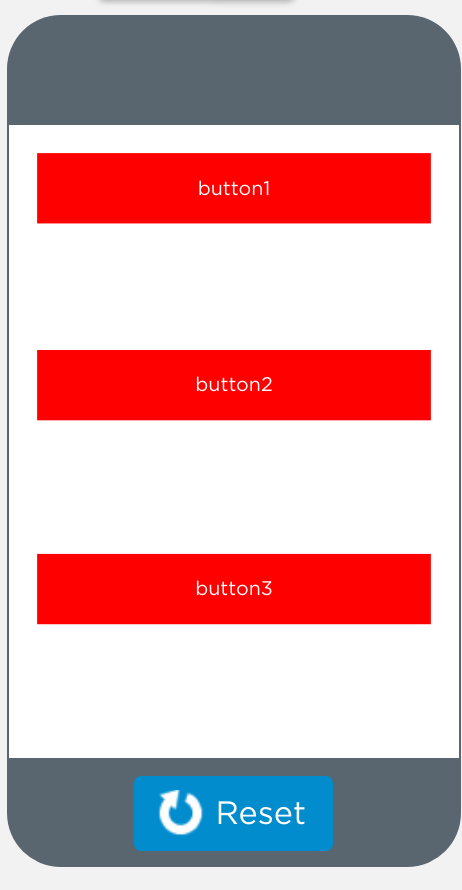
# **Looping Over Arrays 陣列循環**

One of the most powerful ways to use a for loop is to loop over an array, running code on each item in the array. We can do this by using the for loop counter variable (usually i) as the index of your array.  
  
將迴圈程式碼用在陣列中的每個項目是一個最效運用迴圈的方式。我們可以透過使用 for 迴圈計數變數（通常為 i ）來做為你的陣列的指數。

# **Do This 請這麼做**

This program should loop over the array [buttons](https://docs.code.org/applab/buttons/) and do two things to each button - change the background color to red, and change the height to 50 px. 程式應該在陣列尾端開始循環並且對應每個按鍵進行以下兩件事 - 改變背景顏色為紅色，改變文字高度為 50 px。

* Add a second [setProperty](https://docs.code.org/applab/setProperty/) block inside the loop 在迴圈中增加第二個 [setProperty](https://docs.code.org/applab/setProperty/) 區塊
* Change the target to buttons[i] 改變目標至 buttons[i]
* Set the "height" property to 50 改變高度到 50



* [8](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-8)

### Student Instructions

# **Array Length** 陣列的長度

* + In the last program we told the loop to run three times with the code for (var i = 0; i < 3; i++). This works, but we could write smarter programs by using the length of the array to decide how many times to loop.
  + 在前一個程式裡我們用for (var i = 0; i < 3; i++)的程式來讓迴圈執行三次。雖然這種作法可行，但我們有更聰明的辦法可以讓程式根據陣列的長度來決定該執行幾次迴圈。

# **Do This 請這麼做**

* + Change the exit condition of this for loop so that it runs while i < buttons.length.
  + 修改迴圈的結束條件，讓它在 i < buttons.length 時繼續執行
* [9](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-9)

### Student Instructions

# **Constructing a for loop from scratch從零開始寫迴圈**

Now that you've had some practice modifying for loops to process arrays, let's see if you can do it yourself.

到目前為止，你已經做過不少修改現有迴圈來處理陣列的練習了。現在來試試看你能不能自己寫一個。

# **Do This請這樣做**

We've provided the design elements and an array to start with, but the rest is on you.

我們提供了設計要素和陣列讓你開始動手，剩下的就全靠你嘍。

* Add an event handler to respond to the "thumbsup\_button" being clicked
* Place a for loop inside the event handler
* Modify the exit condition of the for loop so that it will run until it gets to the end of the array images
* Inside your for loop change the current image to "icon://fa-thumbs-o-up"
* 加入一段事件處理器來回應 “thumbsup\_button” 被按下時的行為
* 在事件處理器中加入一段迴圈
* 修改迴圈的結束條件，讓它處理到images陣列的最後才能結束。
* 在迴圈中將目前的圖片設定為 "icon://fa-thumbs-o-up"

Challenge: Can you add a second button that turns all of the images back to thumbs down?

挑戰: 加入第二個按鍵，按下它之後會把所有圖片改回拇指向下。你辦得到嗎？

* **(click tabs to see student view)**
* **For Loops and Color LEDs / For 迴圈與彩色LED**
* [10](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-10)

### Student Instructions

# **Turning all the Color LEDs On 點亮所有的彩色LED**

Now that you know how to use a for loop to process all of the elements in an array, you can turn on all of the Color LEDs much more easily than before.

你已經學到了如何使用迴圈來處理陣列中的所有元素。你可以用更簡單的方法來點亮所有的彩色LED。

# **Do This 請這樣做**

Over the next few levels, you'll create an app that will control all of the Color LEDs on your board. The first step is to wire up the button that turns all of the LEDs on. (Don't worry about the rest of the comments in the workspace. You'll work on those in later levels.)

在接下來的幾關裡，你要建立一個小程式來控制板子上的所有彩色LED。首先，要先為按鍵拉好線路，才能用它來點亮所有LED。(先別管工作區裡的其它說明。接下來的關卡中會處理它們的。)

* Add an event handler for "button\_on"
* Place a for loop in your event handler that repeats until it reaches in the end of the array [colorLeds](https://docs.code.org/applab/colorLeds/)
* Inside the for loop, call [colorLeds[i].on()](https://docs.code.org/applab/on/) to turn on the current color LED
* 為 “button\_on” 加上一個事件處理器
* 在事件處理器中寫一個迴圈，執行到colorLeds的結束為止。
* 在迴圈中呼叫 colorLeds[i].on() 來點亮目前的彩色LED。
* [11](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-11)

### Student Instructions

# **Turning all the Color LEDs Off關閉所有的彩色LED**

* + Now that you've got one button to turn the color LEDs on, you can make another turn them off.
  + 你已經做好了一個按鈕可以點亮所有的彩色LED，現在來做一個關上它們的吧。

# **Do This請這樣做**

* + Add an event handler to "button\_off" with a for loop that turns each color LED off.
  + 為 “button\_off” 加上一個事件處理器，並在裡面寫一個迴圈來關閉每一個彩色LED。
* **Switching Colors with a Function使用函數來切換顏色**

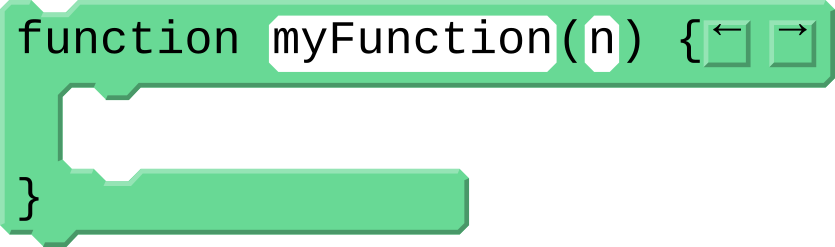
We could write a for loop for each different color that we want to set LEDs to, but that would be a lot of duplicate code that's almost the same. This is the perfect place to use a function with a parameter! You've seen and used functions with parameters before, but now you're going to make one from scratch.

我們可以為每種顏色寫一個迴圈，但這樣會寫出太多重複又只差一點點的程式。這種時候用帶參數的函數就對了！在這之前你已經有見過也使用過帶參數的函數，但這一次我們要從零開始。

# **Do This請這樣做**

Create a function called setLedsColor that takes a parameter [color](https://docs.code.org/applab/color/) and uses it to change the color of all of the LEDs

建立一個函數並取名為 setLedsColor ，它接受一個參數 color 並且根據參數來變更所有LED的顏色。

* At the bottom of your program, drag out a function with parameter block 
* Change the name from myFunction to setLedsColor
* Change the parameter name from n to [color](https://docs.code.org/applab/color/)
* Inside your function, add a for loop that repeats until it reaches the end of [colorLeds](https://docs.code.org/applab/colorLeds/)
* Inside your for loop, set the color of the current led to the parameter [color](https://docs.code.org/applab/color/) (eg. colorLeds[i].color(color)
* 在你程式的最後，拖出一個帶參數的函數區塊如圖
* 把函數的名字從myFunction 改為 setLedsColor
* 把參數的名字從 n 改成 color
* 在函數內加上一個迴圈，執行到 colorLeds 的結束為止。
* 在迴圈內部，將目前的LED顏色設定為參數 color。(範例: colorLeds[i].color(color))

We'll add event handlers that use this function in the next level, but for now you can test your function by your program and typing this into the debug console: setLedsColor("blue")

下一關中，我們會加入事件處理器來呼叫這個函數，不過目前你可以用另一個方式來測試你的函數。在除錯控制臺中輸入 setLedsColor(“blue”)

# **Calling Your Function呼叫你的函數**

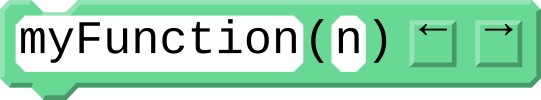
Now that you've created a function that can behave differently based on the parameter it is passed, we can use it in multiple different event handlers to change the lights to different colors.

你已經建立好了一個可以根據參數而有不同的行為的函數。我們可以在各個事件處理器裡面呼叫它來為燈光設定不同的顏色。

# **Do This請這麼做**

We'll start simple by just creating an event handler for "button\_red"

我們先從簡單的開始，為”button\_red”建一個事件處理器。

* Create a new event handler to respond to "button\_red"
* Inside your event handler, add a "call function with parameter" block 
* Change the name from myFunction to setLedsColor
* Change the parameter from n to "red" (note the quotation marks!)
* Test your program, the "Red" button should turn all of your LEDs red
* 建新一個新的事件處理器來回應 “button\_red”
* 在事件處理器中，加入一個”呼叫帶參數的函數”區塊，如圖
* 將函數名稱從 myFunction 改為 setLedsColor
* 將參數從 n 改為 "red" (注意要加上雙引號!)

# **Finish the App完成程式**

If your red button worked, the only thing left is to call your new function when all of the remaining buttons is pressed.

如果你的紅色按鈕可以正常動作，接下來就可以讓每種顏色的按鈕都去呼叫剛才寫好的新函數了。

# **Do This請這樣做**

For each of the remaining buttons (green, blue, and white):

對剩下的三個按鈕(綠，藍，白)：

* Create a new event handler
* Call your setLedsColor function with the appropriate color ("green", "blue", or "white")
* Test your program to make sure all of the buttons work
* 建立新的事件處理器
* 呼叫你的setLedsColor 函數，並傳入適當的參數(“green”, “blue”，或是”white”)

When you've got all of your buttons working properly, click "Submit" to turn your program in.

當你的每個按鈕都能正常工作之後，按下”Submit”鍵提交你的程式。

# **Challenge: Add More Buttons 挑戰：加入更多按鈕**

* + Extend the program by adding more buttons that change the board to different colors.
  + 改寫你的程式，加入更多可以用來把板子切換到不同顏色的按鈕

# **Do This 請這樣做**

* + Add your new buttons in Design Mode and create event handlers to respond to your buttons and change the LED colors.
  + 在 Design Mode 下加入新的按鈕，並且建立事件處理器來回應按鈕事件，在其中改變LED的顏色。
* [Extra](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-17)

### Student Instructions

# **Challenge: Color Patterns挑戰：彩色圖樣**

* + Extend the program by adding more buttons that create different patterns of light.
  + 修改你的程式，加入更多按鈕，可以將燈光設定成不同的圖樣。

# **Do This請這樣做**

* + Add your new buttons in Design Mode and create event handlers to respond to your buttons and change the LED colors.
  + Hint: As you design your patterns, consider whether a function with parameters could help simplify your program.
  + 在 Design Mode 下加入新的按鈕，並且建立事件處理器來回應按鈕事件，在其中改變LED的顏色。
  + 小提示：在設計圖樣的時候可以考慮看看。使用帶有參數的函數能不能幫忙簡化你的程式？
* [Extra](https://curriculum.code.org/csd-18/unit6/12/#level-expando-12-18)

### Student Instructions

# **Challenge: Music挑戰：音樂**

* + Use what you learned in the Making Music lesson to make the board play music when the lights change.
  + 運用你在製作音樂那門課中學到的知識，讓你的板子在燈光變換時同時播放音樂。

# **Do This請這樣做**

* + Create some arrays of notes for different songs and add code to your event handlers to play your music.
  + 建立一些音符組成的陣列代表不同的歌曲，並在你的事件處理器中加入播放音樂用的程式碼。
* **(click tabs to see student view)**

# **Wrap Up總結**

### Journal

Prompt: Have students reflect on their development of the [five practices of CS Discoveries](https://docs.google.com/document/d/1FhHPqlC6dU_z9retuBYb-duUwyKpnjwuEgjF4zfdhvI/edit#heading=h.5u3d4jfozbgc) (Problem Solving, Persistence, Creativity, Collaboration, Communication). Choose one of the following prompts as you deem appropriate.

提醒：讓學生們重新回顧他們在[five practices of CS Discoveries](https://docs.google.com/document/d/1FhHPqlC6dU_z9retuBYb-duUwyKpnjwuEgjF4zfdhvI/edit#heading=h.5u3d4jfozbgc) (Problem Solving, Persistence, Creativity, Collaboration, Communication)上的成長。可以從以下的臺詞中挑一個你覺得適當的來使用：

* Choose one of the five practices in which you believe you demonstrated growth in this lesson. Write something you did that exemplified this practice.
* Choose one practice you think you can continue to grow in. What’s one thing you’d like to do better?
* Choose one practice you thought was especially important for the activity we completed today. What made it so important?
* 從這五項做法當中挑出你覺得在這門課中成長最多的一項。舉出一些你實踐那項做法的例子。
* 選出一項你覺得可以繼續加強的做法。你希望自己那一方面變得更好？
* 從這五項做法當中選出你覺得在這門課最重要的一項。為什麼？

# **Standards Alignment**

#### [View full course alignment](https://curriculum.code.org/csd-18/standards/)

#### CSTA K-12 Computer Science Standards (2017)

AP - Algorithms & ProgrammingCS - Computing Systems



If you are interested in licensing Code.org materials for commercial purposes, [contact us.](https://code.org/contact)

**Lesson 13: Accelerometer**

**第13課: 加速計**

App Lab | Maker Toolkit

**Overview**

In this lesson, students will explore the accelerometer and its capabilities. They’ll become familiar with its events and properties, as well as create multiple programs utilizing the accelerometer similar to those they’ve likely come across in real world applications.

**課程概述**

在本課中，學生將探索加速計及其功能。 學生將熟悉加速計事件和其屬性，並使用加速計創建多個類似他們在現實生活中可能已經看過的程式。

**Purpose**

This lesson gives students an opportunity to work with the accelerometer sensor and explore its orientation properties and accelerometer-specific board events. Students will see the purpose and uses of an accelerometer in real world devices and programs and create their own versions of some of these applications. To do this, students will need to refer back to their past knowledge of the counter pattern to create functional accelerometer-based apps.

**課程目的**

本課程讓學生有機會使用加速計感應器，並探索其方向屬性和加速計特定的電路板事件。 學生將看到加速計在現實世界的設備和程式中的目的和用途，並就其中一些應用程式創建自己的版本。為此，學生需回顧他們過去對計數器模型的了解，以加速計為基礎做出有各種功能的~~有用~~應用程式。

**Agenda**

* **[Warm Up (5 Min)](https://curriculum.code.org/csd-18/unit6/13/#accelerometer0)**
* [What Makes a Sensor?](https://curriculum.code.org/csd-18/unit6/13/#accelerometer1)
* **[Activity](https://curriculum.code.org/csd-18/unit6/13/#accelerometer2)**
* [The Accelerometer](https://curriculum.code.org/csd-18/unit6/13/#accelerometer3)
* **[Wrap Up (5 Min)](https://curriculum.code.org/csd-18/unit6/13/#accelerometer4)**
* [Journal](https://curriculum.code.org/csd-18/unit6/13/#accelerometer5)

**課程流程**

* 暖身活動 (5分鐘)
* 什麼是感應器?
* 活動內容
* 總結 (5分鐘)
* 學習日誌

**[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/13/puzzle/1/)**

在Code Studio上查看

**Objectives**

**課程目標**

**Students will be able to:**

* Recognize the use and need for accelerometer orientation (pitch and roll).
* Identify and explain the difference between the shake, data and change events.
* Refer back to and use their past knowledge of the counter pattern.

學生將能夠：

* 認識加速度計定向（間距和卷）的使用和需求。
* 辨認並解釋震動、數據和變化事件之間的差異。
* 參考並運用他們過去對計數器模型的了解。

**Introduced Code**

* [accelerometer.getOrientation()](https://docs.code.org/applab/accelerometer.getOrientation/)

**介紹的程式碼**

* [accelerometer.getOrientation()](https://docs.code.org/applab/accelerometer.getOrientation/)

**Support**

**[Lesson Forum](https://forum.code.org/c/csd6/)**

**[Report a Bug](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%2013%20curriculum.code.org/csd-18/unit6/13/)**

**課外資源**

討論區

問題回報

**Teaching Guide**

**教學指南**

**Warm Up (5 Min)**

**暖身活動 (5分鐘)**

What Makes a Sensor?

Discussion Goal

Students have been exposed to sensors in previous lessons but have only seen a few aspects of what a sensor can measure. This discussion gives students a chance to think about the characteristics of a sensor, while also thinking about the other possible characteristics of a sensor they haven’t been exposed to.

Prompt: Refer back to the analog sensors, what makes a sensor a sensor? Could an accelerometer be a sensor?

Share: Have students share their thoughts and ideas in small groups.

**什麼構成感應器?**

討論目標

學生們在之前的課程已接觸過感應器，但只看到感應器可以測量的幾個面向。 這個討論讓學生有機會思考感應器的特性，同時猜想其他他們尚未接觸到的感應器的可能特性。

提問：請回想類比感應器，感應器的特性是什麼？ 加速計可以是感應器嗎？

分享：讓學生分成小組分享他們的意見和想法。

**Activity**

**活動內容**

The Accelerometer

加速計

Transition: Send students to Code Studio.

移動：讓學生進入Code Studio

Code Studio levels

* **Lesson Overview**
* [Student Overview](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-1-sfmd)

Code Studio關卡

* 課程總覽
* 給學生的概述

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/13/puzzle/1)

在Code Studio上查看

**Overview**

**課程概述**

In this lesson, students will explore the accelerometer and its capabilities. They’ll become familiar with its events and properties, as well as create multiple programs utilizing the accelerometer similar to those they’ve likely come across in real world applications.

在本課中，學生將探索加速計及其功能。 他們將熟悉加速計事件及其屬性，並使用加速計創建一些類似於他們在現實生活中可能已經遇過的應用程式。

**New Code**

**新的程式碼**

* [accelerometer.getOrientation()](https://docs.code.org/applab/accelerometer.getOrientation/)

**The Accelerometer\_2018**

**加速計\_2018**

[Student Overview](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-2-sfmd)

給學生的概述

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/13/puzzle/2)

在Code Studio上查看

Today you’re going to learn how to use and create functions.

Creating a function lets you make your own blocks! There are two parts to a function:

Making a newly named block that has all of the code you want to run inside of it. -xml of function definition-

The single green block itself that you use, or call, to run the function you created. -xml of function call-

Prepare to get familiar with functions, learn how to edit them, and create your own!

今天，你將學習如何使用和創建函數。

創建函數可以讓你創造自己的程式積木！函數可分成兩個部分：

重新命名已經包含所有你想要的程式碼的程式積木 -xml的函數定義 -

用來跑你所創建的函數的綠色區塊本身 -xml的函數呼叫 -

準備熟悉函數，學習如何編輯並創建自己的函數吧！

# The Accelerometer

加速計

Most of the inputs provided by the Circuit Playground are relatively simple devices, which give a single value. The accelerometer is a more complex kind of sensor, which measure a number of different things all at once, and therefore provices more complex information to the user. An accelerometer is commonly used to detect movement and position in devices like laptops, phones, virtual reality controllers, and and health trackers. For example, pedometers use the movement from accelerometers to track how many steps you take.

大多由Circuit Playground所提供的輸入都是相對簡單的設備，只能提供單一數值。 加速計是一種更複雜的感應器，它可以同時測量許多不同的東西，因此可提供給使用者更複雜的訊息。 加速計通常用於偵測筆記型電腦、電話、虛擬現實控制器和健康追蹤器等設備中的移動和位置。 例如，計步器利用加速計的移動次數來計算你的步數。

## Measuring Orientation

衡量定向

The accelerometer can determine the orientation of the board to figure out how the board is positioned.

加速計能決定電路板的定向，以確定電路板的位置。

|  |  |  |
| --- | --- | --- |
| **Orientation** | **Type of Motion Being Tracked** | **Range of Values** |
| Pitch | Forward and back motion | -90 to 90 |
| Roll | Side to side motion | -90 to 90 |
| Inclination | Rotational motion | -180 to 180 |

|  |  |  |
| --- | --- | --- |
| **定向** | **追蹤的動作種類** | **數值範圍** |
| 間距 | 前後運動 | -90 to 90 |
| 卷 | 左右運動 | -90 to 90 |
| 傾斜 | 旋轉運動 | -180 to 180 |

Using the [accelerometer.getOrientation()](https://docs.code.org/applab/accelerometer.getOrientation/) block, we write programs that respond to changes in pitch, roll, and inclination. This block returns a number for each of these orientations which indicates which way the board is tilting. For example, pitch is positive when it's being tilted back, and negative when being tilted forward.

我們使用accelerometer.getOrientation（）區塊編寫描述間距、卷和傾斜變化的程式。 該區塊能就每個定向回傳一個數值，對應電路板傾斜的方向。 例如，當電路板向後傾斜時，間距為正，向前傾斜時間距為負。

* **Orientation**
* [3](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-3)
* [4](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-4)
* [5](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-5)
* [6](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-6)
* ***(click tabs to see student view)***

**定向**

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/13/puzzle/6)

在Code Studio上查看

* [3](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-3)

**Accelerometer**

加速計

Look at the code below. What do you think it's going to do? Give it a guess, and run the code to find out!

看看下面的程式碼。 你覺得它是做什麼的？ 猜猜看，然後跑程式碼看它是什麼！

* [4](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-4)

Student Instructions

學生操作指示

**Pitch and Roll**

This program displays the board's pitch and roll, and it will also change colors depending on their values.

間距和卷

該程式顯示電路板的間距和卷，並根據其值改變顏色。

**Do This**

Make the program change the following colors by tilting the board to the correct pitch and roll.

**動手做**

改變電路板的水平位置使其間距和卷軸數值如下，以使程式改變為所列顏色。

Pitch: 0 | Roll: -20

Pitch: 30 | Roll: 0

Pitch: -15 | Roll: 10

Pitch: 5 | Roll: -25

Pitch: 10 | Roll: 30

Which way should you tilt to get a positive pitch?

Which way should you tilt to get a negative roll?

你應該將電路板往哪個方向傾以獲得正數的間距？

你應該將電路板往哪個方向傾以獲得負數的間距？

* [5](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-5)

Student Instructions

學生操作指示

**Pitch**

This program lights the top LEDs (0 and 9) when you tilt the board forward. If you tilt the board back, the LEDs at the bottom (4 and 5) should light.

**間距**

當你向前傾斜電路板時，該程式會點亮頂部的LED燈（0和9）。 如果向後傾斜電路板，則底部的LED燈（4和5）應亮起。

**Do This**

* Use the board's pitch to make LEDs 4 and 5 light up when you tilt the board back.
* Run your code and test it a few times.

**動手做**

* 向後傾斜電路板時，用電路板的間距使LED燈 4和5亮起。
* 運作你的程式碼並反覆測試幾次。
* [6](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-6)

Student Instructions

學生操作指示

**Roll**

This program should display "Left" or "Right" on the screen, depending on which way the board is tilted.

**卷**

該程式應依照電路板的傾斜方向，在螢幕上顯示“左”或“右”。

**Do this**

* Use the "roll" property of the accelerometer to determine whether the board is tilted to the left or right.
* Use a conditional to change the text for the "direction" element to be either "Left" or "Right", depending on the board's orientation.

**動手做**

* 使用加速計的“卷”屬性來確定電路板是向左還是向右傾斜。
* 就電路板的方向，使用條件把“方向”的文字改為“左”或“右”。

**Using Accelerometer Events\_2018**

**使用加速計事件\_2018**

[Student Overview](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-7-sfmd)

給學生的概述

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/13/puzzle/7)

在Code Studio上查看

Today you’re going to learn how to use and create functions.

Creating a function lets you make your own blocks! There are two parts to a function:

Making a newly named block that has all of the code you want to run inside of it. -xml of function definition-

The single green block itself that you use, or call, to run the function you created. -xml of function call-

Prepare to get familiar with functions, learn how to edit them, and create your own!

今天，你將學習如何使用和創建函數。

創建函數可以讓你創造自己的程式積木！函數可分成兩個部分：

重新命名已經包含所有你想要的程式碼的程式積木 -xml的函數定義 -

用來跑你所創建的函數的綠色區塊本身 -xml的函數呼叫 -

準備熟悉函數，學習如何編輯並創建自己的函數吧！

**Accelerometer Events**

**加速計事件**

Just like the other sensors, the accelerometer has the data and change events. These are particularly useful because they allow you to constantly track the updated values of the accelerometer, like the orientation.

就像其他感應器一樣，加速計事件有數據和更改功能，讓你能持續追踪加速計的最新數值，如方向等。

|  |  |
| --- | --- |
| **Event** | **Description** |
| data | Fires every 100ms to check for data from the accelerometer. |
| change | Fires whenever the accelerometer detects a change in motion. This checks to see if the current data from sensor is the same as it was 100ms ago. If this is the case, the event won't trigger. Otherwise, the event triggers. |
| shake | Fires whenever the board is shaken. |

|  |  |
| --- | --- |
| **事件** | **說明** |
| 數據 | 每100毫秒觸發一次，檢查加速度計的數據。 |
| 更改 | 只要加速計檢測到運動變化就會觸發。這檢查感應器的數據是否與100毫秒前相同。 如果相同，就不會觸發事件。 否則，事件觸發。 |
| 搖動 | 只要電路板被搖動，事件就會觸發。 |

Sometimes we don't care what the orientation of the board is, we just cared that it moved at all. This is where events like change and shake come in.

有時我們並不在意電路板的方向是什麼，而只在意它動了與否。此時就需要改變和震動等事件。

* **Accelerometer Events**
* [8](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-8)
* [9](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-9)
* ***(click tabs to see student view)***

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/13/puzzle/8)

在Code Studio上查看

* [8](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-8)

Student Instructions

學生操作指示

**Change with the Accelerometer**

This program uses the change event to increase a movement meter every time the board moves. When the movement meter reaches 100, the alarm will sound.

**Do This**

**動手做**

This program has two bugs that you need to fix:

* The movement meter is being triggered by a button press, instead of the accelerometer's change event.
* The programmer forgot how to use the counter pattern to increase the movement variable.

此程式碼有兩個錯誤需要修正

* 動作的測量沒被加速計的更改事件觸發，反被按鈕觸發了。
* 沒把計數器遞增的操作運用於增長動作變數。

Once you have found the bugs, run the program and try to pass the board to your partner without setting off the alerm.

找到這些問題後，執行這程式並傳給組員，別關掉警報喔！

* [9](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-9)

Student Instructions

學生操作指示

**Jogger**

**慢跑者**

This program should use the shake event, which is triggered when the board is shaken. It should add a step to the jogger's total every time the user shakes the board.

此程式應在每次震動時都應觸發震動事件，並把慢跑者的總步數加上一步。

**Do This**

* Change the event so it triggers every time the board is shaken.
* 把事件修改成每次震動都會被觸發。
* Add code inside the event so that steps increases every time the board is shaken.
* 在事件裡加入程式碼讓每次震動時步數都會增加（一步）。

· **Revisiting the Counter Pattern\_2018**

[Student Overview](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-10-sfmd)

給學生的概述

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/13/puzzle/10)

在Code Studio上查看

Today you’re going to learn how to use and create functions.

Creating a function lets you make your own blocks! There are two parts to a function:

Making a newly named block that has all of the code you want to run inside of it. -xml of function definition-

The single green block itself that you use, or call, to run the function you created. -xml of function call-

Prepare to get familiar with functions, learn how to edit them, and create your own!

今天，你將學習如何使用和創建函數。

創建函數可以讓你創造自己的程式積木！函數可分成兩個部分：

重新命名已經包含所有你想要的程式碼的程式積木 -xml的函數定義 -

用來跑你所創建的函數的綠色區塊本身 -xml的函數呼叫 -

準備熟悉函數，學習如何編輯並創建自己的函數吧！

**Revisiting The Counter Pattern**

**複習計數器的操作**

In Game Lab we relied on the counter pattern to continually update sprite properties and create animations. While the elements in App Lab don't have the same kinds of properties that sprites do, we can still apply the counter pattern to create some simple animations.

在遊戲練習中我們藉由操作計數器來更新精靈的屬性以及創建動畫。雖然在應用程式練習中沒有那些精靈的屬性，我們仍可以把計數器的操作運用於一些簡單的動畫上。

Like the draw loop in Game Lab, "data" events run a single block of code over and over again. Because these events run code repeatedly, you can use the counter pattern inside a "data" event to bring some simple . You've been using the data event to track the accelerometer itself, but in order to make more complex programs you're going to have to update additional values outside of the accelerometer.

像是在遊戲練習中繪製的迴圈，「數據」事件也是反覆的執行單一程式區塊。由於這些事件一直在重複執行程式，你可以在「數據」事件的裡面使用計數器的操作來完成簡化程式。你已經用過數據事件來追蹤加速計本身，但為了能寫出更複雜的程式，你還需要追蹤加速計以外的數值。

**Moving Design Elements with the Counter Pattern**

**操作計數器來移動設計出的元件**

The above code uses the counter pattern in a "data" event to move a design element down the screen. In order to make our code more readable, we've saved the current "y" value position of "anchor" to a variable before using it in the counter pattern, but you could also write the pattern as:

以上的程式藉由操作在「數據」事件裡的計數器，將設計出的元件往下移動。為了提高程式的可讀性，”anchor”的”y”已被儲存在變數中，然後再拿來計數，但你也可以把它改寫成以下程式：

[setProperty("anchor", "y", getProperty("anchor", "y") + 1);](https://docs.code.org/applab/setProperty/)

· **Accelerometer Events**

·  [11](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-11)

·  [12](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-12)

·  [13](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-13)

· ***(click tabs to see student view)***

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/13/puzzle/11)

在Code Studio上查看

[11](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-11)

Student Instructions

學生操作指示

**Updating Values with Data**

In this program the goalie is trying to block the soccer ball, but right now the ball isn't moving.

**Do This**

**動手做**

* Use the counter pattern to update the ball's [x](https://docs.code.org/gamelab/x/) value so that it moves across the screen.
* 操作計數器來將球的x值更新，進而將球在螢幕上移動。

·  [12](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-12)

Student Instructions

學生操作指示

**Steering Wheel**

**方向盤**

This game is similar to the goalie game that you just played, except a little different. Use your board as a steering wheel for the car (moving side to side). You have to get to the checkered flag at while avoiding the red flags along the way!

這遊戲與你剛玩的守門員遊戲只有一點點的不同。請把電路板視為車子的方向盤（可以移來移去）。你必須到達格子旗但避開紅旗。

**Do This**

**動手做**

* Make the car move down the screen while the program runs.
* 程式要讓車子在螢幕上往下移動。
* Give the car the ability to steer side to side.
* 車子要可以移來移去。

Think back to the programs you've made up until this point, how can you update these values?

參考你目前為止所寫的程式，有沒有什麼方法可以更新這些數值呢？

·  [13](https://curriculum.code.org/csd-18/unit6/13/#level-expando-13-13)

Student Instructions

學生操作指示

**Limiting Turn**

**限制轉動的程度**

You might have noticed that it was really hard to steer the car unless your hands were moving slowly and carefully. This is because the number that roll returns can be a really big number depending on how much the accelerometer is being turned. Remember, roll has a range from -90 to 90.

你可能已經注意到車子很難控制，除非你的手穩穩的慢速移動。這是因為卷的數值可能會因為加速計被轉很多而變很大。請記得卷的數值是-90到90。

**Do This**

**動手做**

Find a way to make roll a smaller number while still using it to steer the car. You have all of the math operators available to you for this.

找個方法讓卷的數值變小的同時還是可以控制車子。你現在已經學會所有你需要的數學運算子。

Challenge: Can you keep the car from going off the side of the screen?

挑戰看看：你可以讓車子不要衝出螢幕範圍嗎？

**Wrap Up (5 Min)**

**總結 (5分鐘)**

Journal

學習日誌

Goal: Students should be able to recognize the use for an accelerometer.

Prompt: Now that you’ve created your own programs with the accelerometer, can you think of some ways you interact with accelerometers every day through technology?

目標：學生應該要學會加速計的操作

提問：既然你已經自己寫出有關加速計的程式了，你可以想出一些日常生活中與加速計互動的科技嗎？

# **Lesson 14: Functions with Parameters**

#### **App Lab | Maker Toolkit**

## **Overview**

課程概述

The lesson starts with a quick review of parameters, in the context of App Lab blocks that they students have seen recently. Students then look at examples of parameters within user-created functions in App Lab and create and call functions with parameters for themselves, using them to control multiple elements on a screen. Afterwards, students use for loops to iterate over an array, passing each element into a function. Last, students use what they have learned to create a star catching game.

先前學生已經在App Lab的環境中看過區塊的參數,這堂課一開始先快速重溫參數.接著讓學生在App Lab中建立使用者自訂函式,看一些自訂函式的參數範例,並且用他們自己的參數呼叫那些函式來控制螢幕上的多個元件.然後,學生使用for迴圈循環訪問(迭代)一個陣列,將每個陣列元素傳入函式中.最後,學生用他們學到的東西寫一個抓星星的遊戲.

## **Purpose**

課程目的

In previous lessons, students have used functions to define blocks of code that can be used in multiple places in a program. In this lesson, students learn how to use parameters to generalize the purpose of a function. Parameters allow a program to specify the details of how a function works when it is called, rather than when the program is defined. Although students have seen functions with parameters earlier in the unit, this is the first time that they are expected to define and call their own. Students also learn how to use a for loop to iteratively pass in the elements of an array as parameters to a function, allowing them to use the same function on multiple elements on the screen.

在前一堂課,學生已經使用函式來定義程式區塊,一個函式可以被用在程式中的多個地方. 這堂課, 學生將學習如何使用參數進行函式功能的一般化. 參數讓程式得以在函式被呼叫時才決定函式的實際運作方式,而不是在程式被定義時就決定. 雖然學生稍早在本單元已經看過使用參數的函式,不過 這卻是他們第一次自己定義和呼叫. 學生也將學習使用for迴圈來反覆地將陣列的元素傳入函式作為參數, 允許他們用相同的函式處理螢幕上的多個元件.

## **Agenda**

課程流程

* [Warm Up](https://curriculum.code.org/csd-18/unit6/14/#functions-with-parameters0)
* [Activity](https://curriculum.code.org/csd-18/unit6/14/#functions-with-parameters1)
* [Wrap Up](https://curriculum.code.org/csd-18/unit6/14/#functions-with-parameters2)

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/1/)**

## **Objectives**

課程目標

### **Students will be able to:**

* Use parameters to generalize the purpose of a function.

學生將俱備以下能力:

使用參數對函式功能進行一般化.

## **Vocabulary**

字彙解釋

* **Parameter** - An extra piece of information passed to a function to customize it for a specific need

參數 -- 傳遞至函式中的額外資訊, 用來客製化函式以滿足特定需求

## **Introduced Code**

介紹程式碼

* [function myFunction(n){ //code }](https://docs.code.org/applab/functionParams_n/)
* [Call a function with parameters](https://docs.code.org/applab/callMyFunction_n/)

## **Support**

課外資源

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

### **[Report a Bug](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%2014%20curriculum.code.org/csd-18/unit6/14/)**

# **Teaching Guide**

教學指南

## **Warm Up**

暖身活動

Discussion Goal

討論目標

**Goal:** Students should eventually see that parameters give a program flexibility. Sometimes, multiple commands are similar enough that it makes sense to combine them into one, but with a parameter to distinguish between their differences. Sometimes, such as the case of [buzzer.frequency](https://docs.code.org/applab/buzzer.frequency/), there are too many options to make a separate block for each one. Parameters allow a programmer to use a single solution to solve multiple, related problems.

目標: 學生最終將了解使用參數讓程式變得有彈性. 有時候, 將多個相似的指令合而為一,並利用參數來區別它們是非常合理的做法. 有時候, 比方說在[buzzer.frequency](https://docs.code.org/applab/buzzer.frequency/)的例子, 有太多個選項以至於無法為每個選項產生個別的區塊. 參數讓編程者使用單一解決方案來解決多種相關聯的問題.

**Prompt:** When you needed to access the pitch and roll of the accelerometer in the last lesson, you used the block [accelerometer.getOrientation](https://docs.code.org/applab/accelerometer.getOrientation/), and you chose whether to get the pitch or the roll. Why do you think the creators make the program work that way, rather than having two blocks, one for the pitch and one for the roll?

What other blocks that you have seen take parameters? Why are parameters so useful?

提問: 當你需要取得上堂課中加速度計的pitch和roll資訊, 你使用[accelerometer.getOrientation](https://docs.code.org/applab/accelerometer.getOrientation/)區塊,而且你能選擇取得pitch或是roll資訊. 為什麼你認為創造者讓程式那樣運作,而不是使用兩個區塊,一個用在pitch,另一個用在roll?

你看過其他用到參數的區塊嗎? 為什麼參數這麼有用?

**Remarks**

So far, the functions that you have used always did the exact same thing. Today, we're going to look at a way to make functions even more useful by giving them parameters, just like some of the blocks that you just talked about.

教學備註:

目前為止, 你所用過的函式總是做一樣的事情. 今天, 我們想要看看怎麼利用參數讓函式變得更有用, 正如同某些你剛討論過的區塊一樣.

## **Activity**

活動內容

Send students to Code Studio

### **Code Studio levels**

Code Studio關卡

* Lesson Overview

課程概述

* [Student Overview](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/1)

* Functions with Parameters

具有參數的函式

* [2](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-2)
* [3](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-3)
* [4](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-4)
* [5](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-5)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/2)

### **Student Instructions**

學生操作指示

# **Bug Crawl**

抓蟲

Look at the program below, paying special attention to the function crawl(bug). What happens if you press the left or right button?

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/3)

看一下底下的程式,特別注意crawl(bug)函式. 當你按下滑鼠左鍵或右鍵時, 會發生什麼事情?

### **Student Instructions**

學生操作指示

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/4)

### **Student Instructions**

學生操作指示

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/5)

### **Student Instructions**

學生操作指示

# **Color Picker**

選顏色

In this program, pressing the buttons at the bottom of the screen changes the color of the images and the color LEDs.

在這支程式, 按下螢幕下方的按鈕可以改變圖片和彩色LED的顏色.

# **Do This**

動手做

* Create a function chooseColor(color) that changes each of the images and the color LEDs to the given color

建立一個chooseColor(color)函式,用來將每張圖片和彩色LED改成指定的顏色.

* Call your function in the event blocks so that all the buttons work.

在事件區塊中呼叫你的函式, 讓所有按鈕正常運作.

* Iteration and Parameters

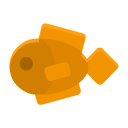
重複和參數

* [6](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-6)
* [7](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-7)
* [8](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-8)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/6)

### **Student Instructions**

學生操作指示

****

# **Fish Bubbles**

魚的泡泡

This program uses the sound sensor in the same way as the clouds program. How many bubbles will move when you blow on the board?

這程式使用聲音感測器, 使用方式就像clouds程式. 有多少泡泡可以移動, 當你吹你的實驗板?

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/7)

### **Student Instructions**

學生操作指示

# **Bug Shaker**

抖抖蟲子

This program shakes bugs around when the board is shaken, but it's only working for one bug.

這支程式在實驗板被搖晃時也搖晃蟲子, 但它只能對一隻蟲產生作用.

# **Do This**

動手做

* Change the moveBugs() function so that it calls moveBug(color) on every bug color.
* (Hint: Look at the detectHits() function for clues how to do this.)

修改moveBugs()函式, 使它以蟲的顏色作為參數呼叫moveBug(color).

(提示: 仔細看detectHits()函式, 尋找完成的線索.)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/8)

### **Student Instructions**

學生操作指示

# **Music Player**

音樂玩家

This program lets you play music by waving a wand over the different notes, but it's missing a function definition.

這支程式讓你藉由揮動指揮棒,在不同的音符上演奏音樂, 但它漏了函式的定義.

# **Do This**

動手做

* Create a checkNotes() function that will call checkNote(note) on everything in the "notes" array.

建立一個checkNotes()函式,以任何在"notes" 陣列中的東西作為參數,呼叫checkNote(note).

* Star Chaser

追星者

* [9](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-9)
* [10](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-10)
* [11](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-11)
* [12](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-12)
* [13](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-13)
* [14](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-14)
* [Extra](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-15)
* [Extra](https://curriculum.code.org/csd-18/unit6/14/#level-expando-14-16)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/9)

### **Student Instructions**

學生操作指示

# **Star Chaser**

追星者

In this game, the user tries to catch the stars according to the color of the LEDs.

在這個遊戲,使用者根據LED的顏色試著抓星星

# **Do This**

動手做

* Play the game, and discuss the following with a partner:

玩一玩這個遊戲,並且和同伴討論以下問題

* What functions might this program need?

這支程式可能需要哪些函式?

* What parameters should these functions have?

這些函式應該要有那些參數?

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/10)

### **Student Instructions**

學生操作指示

# **Make your loopStar(color) function**

撰寫你自己的 loopStar(color) 函式

Right now only the red star works, so you'll need to change this code.

目前只有紅星正常運作,因此你必須修改這段程式碼.

# **Do This**

動手做

* Use the code from the loopRedStar() function to create a loopStar(color) function that will work with any star.

利用 loopRedStar() 函式中的程式碼, 建立一個能用在任何星星的 loopStar(color) 函式.

* Test your function by calling loopStar("red") and loopStar("blue") inside the loopStars() function.

在 loopStars() 函式內部呼叫 loopStar("red") 與 loopStar("blue"), 藉此測試你的函式.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/11)

### **Student Instructions**

學生操作指示

# **Star Chaser**

追星者

Now you'll need to make all the stars loop around the screen.

現在你必須讓所有的星星繞行螢幕.

# **Do This**

動手做

* Use a for loop inside your loopStars() function to make every star in "starArray" loop around the screen.

在你的 loopStars() 函式中使用 for 迴圈, 讓 "starArray" 中的每顆星星繞行螢幕.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/12)

### **Student Instructions**

學生操作指示

# **Star Chaser**

追星者

All the stars loop, but you can still only catch the red star.

所有星星都在繞行,但是你卻只能捕抓紅星.

# **Do This**

動手做

* Use code from the moveRedStar() function to make a moveStar(color) function that can move any star.

利用 moveRedStar() 函式的程式碼撰寫一個 moveStar(color) 函式, 該函式能移動任何星星.

* Use code from the checkRedStar() function to make a checkStar(color) function that works for any star.

利用 checkRedStar() 函式的程式碼撰寫一個 checkStar(color) 函式, 該函式能用於任何一種星星.

* Test your code by calling checkStar("red") and checkStar("blue") inside your event block.

在你的事件區塊中呼叫 checkStar("red") 與 checkStar("blue"), 藉此測試你的程式碼.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/13)

### **Student Instructions**

學生操作指示

# **Star Chaser**

追星者

Now make it work for all the stars

現在讓它能用於所有星星.

# **Do This**

動手做

* Create a checkStars() functions that uses a for loop to check every star in the star array.

建立一個 checkStars() 函式, 讓它用 for 迴圈來檢查 star 陣列中的每顆星星.

* Test your code by calling the checkStars() function inside the event block.

在事件區塊中呼叫 checkStars(), 藉此測試你的程式碼.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/puzzle/14)

### **Student Instructions**

學生操作指示

# **Star Chaser**

追星者

Now you can add extra stars, and your functions will work with them, too.

現在你可以加入額外的星星, 而且你的函式也能處理它們.

# **Do This**

動手做

* In design mode, copy one of the stars, and change its name and color to "yellow".

在設計模式下複製其中一顆星星, 重新命名並將它的顏色改成 “yellow”.

* Add a new element, "yellow", to your star array.

加入一個新的元件 “yellow” 到你的 star 陣列.

* Test the game with your new star.

用新加入的星星測試遊戲.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/extras)

### **Student Instructions**

學生操作指示

# **Star Chaser**

追星者

Now that you have a working game, you can challenge yourself with some different features.

既然你已經有一個可以正常執行的遊戲, 你可以試著挑戰自己,加入一些不同的功能.

# **Do This**

動手做

* Start the user with 5 "lives", and take one away every time the user touches the **wrong** star.

一開始給使用者五條命, 每次當使用者觸碰到”錯誤”的星星時, 就取走一條命.

* Make an end screen, and change to that screen when the user runs out of lives.

製作一個結束畫面, 當使用者用完生命時就切換到該畫面.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/14/extras)

### **Student Instructions**

學生操作指示

# **Star Chaser**

追星者

Now that you have a working game, you can challenge yourself with some different features.

既然你已經有一個可以正常執行的遊戲, 你可以試著挑戰自己,加入一些不同的功能.

# **Do This**

動手做

* Make the player go back to the middle of the screen when the left button is pressed.

當左按鈕被按下時, 讓玩家回到螢幕中央.

## **Wrap Up**

暖身活動

**Prompt:** Think back to some of the programs that you have made before. What are two times that you could have used functions with parameters? What would the parameter be? How should the function's behavior change when the parameter changes?

提問: 回想你從前寫過的一些程式, 有哪兩次你原本能將參數用在函式卻沒那麼做? 那會是什麼參數? 當參數改變時, 函式的行為應當如何改變?

# **Lesson 15: Circuits and Physical Prototypes**

#### **App Lab | Maker Toolkit**

# **第15課：電路和實體原型**

#### **App Lab | Maker Toolkit**

## **Overview**

In preparation for this chapter's final project, students will learn how to develop a prototype of a physical object that includes a Circuit Playground. Using a modelled project planning guide, students will learn how to wire a couple of simple circuits and to build prototypes that can communicate the intended design of a product, using cheap and easily found materials such as cardboard and duct tape.

## **概觀**

## 在準備本章的最後一個專案，學生將學習如何利用 Circuit Playground 的元件來開發一個實體原型。使用建模的項目規劃指南，學生將學習如何連接幾個簡單的電路，並使用便宜且易於找到的材料（如紙板和膠帶）構建可以傳達產品預期設計的原型。

## **Purpose**

The goal of this lesson is both to model for students how thinking about the physical design of a product impacts the prototyping process, and to introduce a handful of practical skills that will make creating their final projects easier.

## **目的**

本課程的目標是為學生建模如何思考產品的實體設計如何影響原型製作過程，並介紹一些實用技能，以便更輕鬆地創建最終產品。

## **Agenda**

* [Warm Up (5 min)](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes0)
* [Designing a Physical Device](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes1)
* [Activity 1 (30 min)](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes2)
* [Introducing the Smart Bike](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes3)
* [Activity 2 (30-45 min)](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes4)
* [Building the Prototype](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes5)
* [Activity 3 (30-45 min)](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes6)
* [Adding Inputs](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes7)
* [Wrap Up (15 min)](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes8)
* [Sharing Designs](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes9)

### [**View on Code Studio**](https://studio.code.org/s/csd6-2018/stage/15/puzzle/1/)

## **議程**

* **[熱身（5分鐘）](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes0)**
* [設計物理設備](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes1)
* **[活動1（30分鐘）](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes2)**
* [介紹智能自行車](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes3)
* **[活動2（30-45分鐘）](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes4)**
* [構建原型](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes5)
* **[活動3（30-45分鐘）](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes6)**
* [添加輸入](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes7)
* **[總結（15分鐘）](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes8)**
* [分享設計](https://curriculum.code.org/csd-18/unit6/15/#circuits-and-physical-prototypes9)

### [**View on Code Studio**](https://studio.code.org/s/csd6-2018/stage/15/puzzle/1/)

## **Objectives**

### **Students will be able to:**

* Create and debug simple circuits
* Develop an interactive physical protype that combines software and hardware
* Consider the needs of diverse users when designing a product

## **目標**

### **學生將能夠：**

## 創造和偵錯簡單電路

## 開發一種結合了能互相溝通的軟件和硬件的實體原型

## 在設計產品時考慮不同用戶的需求

## **Preparation**

* Gather prototyping materials, such as:
  + Structural material (cardboard, construction paper, etc)
  + Connective material (tape, glue, hot glue, etc)
  + Construction tools (scissors, staplers, etc)
  + Other materials (cups, binder clips, paper plates, etc)
* Prepare circuit wiring materials, such as:
  + Alligator clip wires (included in Circuit Playground classroom kit)
  + LEDs (included in Circuit Playground classroom kit)
  + Other conductive material (wire, paper clips, foil, etc)
  + (optional) Buttons or switches
* Print a copy of [Physical Prototyping - Project Guide](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/edit) for each group of 2-3 students
* Prepare a model button to show the class

## **準備**

* 收集原型所需材料，例如：
  + 結構材料（紙板，建築紙等）
  + 連接材料（膠帶，膠水，熱膠等）
  + 施工工具（剪刀，訂書機等）
  + 其他材料（杯子，活頁夾，紙盤等）
* 準備電路佈線材料，例如：
  + 鱷魚夾線（包含在Circuit Playground教室套件中）
  + LED（包含在Circuit Playground教室套件中）
  + 其他導電材料（電線，紙夾，鋁箔等）
  + （其它）按鈕或開關
* 為每組2-3名學生列印一份[物理原型 - 項目指南](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/edit)
* 準備一個模型按鈕來顯示課程

## **Links**

**Heads Up!** Please make a copy of any documents you plan to share with students.

## **鏈接**

**提醒！**請複製您計劃與學生分享的任何文件。

### **For the Students**

* [How Computers Work: Circuits & Logic](https://youtu.be/ZoqMiFKspAA) - Video ([download](http://videos.code.org/cs-discoveries/concept/04_CircuitsLogic_sm.mp4))
* [Physical Prototyping](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/edit) - Project Guide Make a Copy
  + [PDF](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/copy)

**給學生**

* [計算機如何工作：電路與邏輯](https://youtu.be/ZoqMiFKspAA) [- 視頻（](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/copy)[下載](http://videos.code.org/cs-discoveries/concept/04_CircuitsLogic_sm.mp4)[）](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/copy)
* [實體原型](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/edit) [- 項目指南](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/copy) 
  + [PDF](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/copy)

## **Vocabulary**

* **Circuit** - A device that provides a path for an electric current to flow, often modifying that current. In computers, circuits allow for simple logical and mathematical operations using electricity.
* **Prototype** - A first or early model of a product that allows you to test assumptions before developing a final version.

## **詞彙**

* **電路 -** 提供電流流動路徑的設備，通常會修改電流。在計算機中，電路允許使用電進行簡單的邏輯和數學運算。
* **原型 -** 產品的第一個或早期模型，允許您在開發最終版本之前測試假設。

## **Introduced Code**

## **介紹程式代碼**

* [pinMode(pin, mode);](https://docs.code.org/applab/pinMode/)
* [digitalWrite()](https://docs.code.org/applab/digitalWrite/)
* [var myLed = createLed(pin);](https://docs.code.org/applab/createled/)
* [var myButton = createButton(pin);](https://docs.code.org/applab/createbutton/)

## **Support**

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

### [**Report a Bug**](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%2015%20curriculum.code.org/csd-18/unit6/15/)

## **支持**

### **[課程論壇](https://forum.code.org/c/csd6/)**

### **[報告錯誤](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%2015%20curriculum.code.org/csd-18/unit6/15/)**

# **Teaching Guide**

## **Warm Up (5 min)**

### **Designing a Physical Device**

Discussion Goal

The point of this short discussion is to get students thinking about how we might use a board like the Circuit Playground as part of a larger computing device. It's not important that we're able to completely replicate student's ideas with the board, but that we can start thinking about where the board could be used and where we might need to add additional functionality.

**Prompt:** If you could create any kind of computational device, what would it be? What would it do? How would people interact with it?

**Share:** After a few minutes of thinking time, have students share their ideas.

**Discuss:** How might a device like the Circuit Playground help us design and prototype some of these ideas? Consider picking a couple of ideas to put up on the board and list with the class which things features of a given device could be replicated with an element of the board, and which might require additional hardware.

# **教學指南**

## **熱身（5分鐘）**

### **設計實體裝置**

討論目標

這個簡短討論的重點是讓學生思考我們如何使用像Circuit Playground這樣的電路板作為大型計算設備的一部分。是否能夠利用電路板完全複製學生的想法並不重要，重要的是我們可以考慮怎麼使用電路板以及可能需要添加其他功能的地方。

**提示：**如果你可以創建任何類型的計算設備，它會是什麼？它會做什麼？人們將如何與之互動？

**分享：**經過幾分鐘的思考，讓學生分享他們的想法。

**討論：** Circuit Playground這樣的設備如何幫助我們設計和製作這些想法？考慮選擇一些想法，以便在白板上列出並列出給定設備的某些功能可以與板上的元件一起複製，並且可能需要額外的硬件來實現。

## **Activity 1 (30 min)**

### **Introducing the Smart Bike**

**Group:** Place students into groups of 3-4.

**Distribute:** Give each group a copy of [Physical Prototyping - Project Guide](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/edit), and introduce the project. Give students a moment to look over the activity guide. This guide is similar to the one they will be using for their own final projects, but it's already been mostly completed, which allows for them to focus on how to implement the idea, instead of what the idea should be.

Discussion Goal

Students should be able to identify that almost all of the functionality (such as blinking lights, responding to button presses, and buzzing) can be done with the Circuit Playground as is. The major barrier we want to identify is how the physical requirements of the plan (such as placing the blinker lights on the ends of the handlebars) would require adding new hardware to the board. This is intended to tee up the next activity, in which students wire additional circuits onto their boards.

**Discuss:** Focusing on the description and sketch on the first page, how might we use the Circuit Playground to develop this prototype. Which elements are we currently *unable* to replicate with the board?

**Display:** Watch the [How Computers Work: Circuits & Logic - Video](https://youtu.be/ZoqMiFKspAA) (level two in the progression) as a class.

Remarks

The elements of our Circuit Playground are all made up of circuits so small that we can't even see most of them, but you can create a simple circuit on your own by attaching wires to the copper pads on the edge of the board.

**Distribute:** Wires, LEDs, and any other hardware you want available for this portion. Let the class know that before digging too far into building our prototypes, we'll need to learn how to add additional hardware to the board.

**Transition:** Head to Code Studio to work on the Simple Circuits section.

## **活動1（30分鐘）**

### **介紹智慧自行車**

**分組**：將學生分成3-4組。

**分發**：為每個組提供[物理原型 - 項目指南](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/edit)的副本，並介紹該專案。給學生一些時間查看活動指南。本指南類似於他們將用於他們自己的最終專案的指南，但它已經基本完成，這使他們能夠專注於如何實現這個想法，而不是去思考想法應該是什麼。

### 討論目標

### 學生應該能夠確認幾乎所有的功能（例如閃爍的燈光，響應按鈕按下和嗡嗡聲）都可以在Circuit Playground中完成。我們想要確定的主要障礙是, 計劃的實際要求（例如將閃光燈放在把手的末端）如何將需要在電路板上添加新的硬件。這是為了開展下一個活動，學生將額外的電路連接到他們的電路板上。

**討論：**關注第一頁的描述和草圖，我們如何使用Circuit Playground開發這個原型。我們目前無法在電路板上複製哪些項目？

**顯示：**觀看[計算機如何工作：電路和邏輯 - 視頻](https://youtu.be/ZoqMiFKspAA)（進展中的第二級）。十

***備註***

我們的Circuit Playground的元件都是由很小的電路組成，我們甚至看不到它們中的大部分，但你可以通過將電線連接到電路板邊緣的銅焊盤來自行創建一個簡單的電路。

**分發**：;您希望為此部分提供好電線，LED和任何其他硬件。讓班級知道，在深入研究構建原型之前，我們需要學習如何在電路板上添加額外的硬件。

**轉換 :** 前往Code Studio處理Simple Circuits部分。

### **Code Studio levels**

* Levels
* [3](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-3)
* [4](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-4)
* [5](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-5)
* [6](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-6)
* [7](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-7)
* [8](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-8)
* [9](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-9)
* [10](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-10)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/3)

### **Student Instructions**

# **Make a Prediction**

All of the devices that you've used so far are actually circuits connected to numbered pins! Look for **#13** on your board to see which circuit is connected to pin 13, then read through this code and predict what will happen when the program is run.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/4)

### 學生指示

# **做一個預測**

到目前為止，您使用的所有設備實際上都是連接到編號引腳的電路！查看電路板上的＃13，看看哪個電路連接到引腳13，然後讀取此代碼並預測程序運行時會發生什麼。

Teaching Tip

#### **Learning More About LEDs**

We include LED sequins in the Circuit Playground classroom pack, but if you're careful you can use just about any LEDs you like. To learn more about pairing LEDs with the proper resistor, check out [Adafruit's LED primer](https://learn.adafruit.com/all-about-leds/overview).

### 教學提示

#### 更多了解LED

我們在Circuit Playground教室包中加入LED亮片，但如果你小心使用，你可以使用你喜歡的任何LED。要了解有關將LED與適當電阻配對的更多信息，請查看[Adafruit的LED手冊](https://learn.adafruit.com/all-about-leds/overview)。

### **Student Instructions**

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/5)

Teaching Tip

### **Why Are We Talking Pins?**

You may notice that in the early levels of this lesson students are programming in a much different way than they have before, directly manipulating pins. The [digitalWrite()](https://docs.code.org/applab/digitalWrite/) and [pinMode()](https://docs.code.org/applab/pinMode/) commands are useful to get our wiring set up quickly, but are not the core concerns of this lesson. As soon as students are comfortable wiring LED circuits we will introduce techniques to control those LEDs with the higher level commands that have been used up to this point (like [led.on()](https://docs.code.org/applab/led.on/) and [led.off()](https://docs.code.org/applab/led.off/)).

教學提示

### 我們為什麼要談引腳 (pin)？

您可能會注意到，在本課程的早期階段，學生的編程方式與以前完全不同，直接操作引腳。用[digitalWrite()](https://docs.code.org/applab/digitalWrite/) 和 [pinMode()](https://docs.code.org/applab/pinMode/) 命令，讓我們的接線馬上有作用，但這不是這節課的核心問題。一旦學生們能輕易地接線LED電路，我們將介紹技術來控制這些LED，使用到目前為止使用的更高級別的命令（如[led.on()](https://docs.code.org/applab/led.on/) 和 [led.off()](https://docs.code.org/applab/led.off/))）。

### **Student Instructions**

# **Wire a Circuit**

You can use any of the numbered pads to add additional circuits to your board. Let's use pin 2 to add another LED.

# **Do This**

* Using a wire, connect pin 2 to the positive (+) side of an LED
* Using another wire, connect the negative (-) side of the LED to a ground (GND) pin (it doesn't matter which one)
* Run this program to test your circuit. If it worked, your LED should turn on.

*Debugging Tip: LEDs only work if the electricity is flowing from positive to negative. If your LED doesn't light up, make sure that it's oriented the right way.*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/6)

### **學生指示**

# **連接電路**

您可以使用任何編號的焊盤為電路板添加額外的電路。讓我們使用引腳2添加另一個LED。

# **做這個**

* 使用導線將引腳2連接到LED的正極（+）側
* 使用另一根導線，將LED的負極（ - ）側連接到接地（GND）引腳（無論哪一個都無關緊要）
* 運行此程序以測試您的電路。如果有效，您的LED應該會點亮。

偵錯提示：只有當電流從正電流流向負電壓時，LED才有效。如果您的LED沒有亮起，請確保它的方向正確。

### **Student Instructions**

# **Creating Board Objects**

You might have noticed that the new blocks we're using are in a different toolbox drawer. The **Circuit** drawer contains all of the board objects that are built into the Circuit Playground, but when you start wiring your own circuits the Maker Toolkit no longer knows where everything is.

The new **Maker** drawer contains general purpose commands instead of ones that are customized for the Circuit Playground. In addition to the [pinMode()](https://docs.code.org/applab/pinMode/) and [digitalWrite()](https://docs.code.org/applab/digitalWrite/) commands you've seen, it includes commands to create new objects on the board that can be programmed in the same way as the blocks in the **Circuit** drawer. The var myLed = createLed() command, for example, creates a new LED object that behaves just like the [led](https://docs.code.org/applab/led/) blocks you've been using.

# **Do This**

Now that you're creating new board objects that we don't have blocks for, you'll need to work in text mode. You can still drag out blocks that you're familiar with from the **Circuit** drawer, you'll just need to change the name of the object.

* Keep your LED wired just as it was before (connected to pin 2)
* Make sure you're in text mode, not block mode.
* Drag out an [led.blink()](https://docs.code.org/applab/led.blink/) block below the comment Blink myLed.
* Replace the text [led](https://docs.code.org/applab/led/) with myLed.
* Test your code.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/7)

### 學生指示

# **創建電路板物件**

您可能已經注意到我們正在使用的新板位於不同的工具箱抽屜中。該電路抽屜 ( Circuit Drawer) 包含所有內置到Circuit Playground 板的物件，但是當你開始接線自己的電路時，Maker Toolkit不再知道這裡的一切。

新的Maker抽屜包含通用命令，而不是為Circuit Playground定制的命令。除了您看到的[pinMode()](https://docs.code.org/applab/pinMode/)和[digitalWrite()](https://docs.code.org/applab/digitalWrite/)命令之外，它還包括在電路板上創建新物件的命令，可以使用與“ 電路”抽屜中的塊相同的方式進行編程。例如 var myLed = createLed()，該命令創建一個新的LED物件，其行為就像 您一直使用的[led](https://docs.code.org/applab/led/)區塊一樣。

# **做這個**

現在您正在創建沒有區塊的一個新的板物件，您需要在文件模式下工作。您仍然可以從Circuit抽屜中拖出您熟悉的區塊，您只需要更改物件的名稱。

* 讓LED保持原來的連接（連接到引腳2）
* 確保您處於文件模式，而不是區塊模式。
* 從下面拖出[led.blink()](https://docs.code.org/applab/led.blink/) 上面註解著 Blink myLed。
* 替換文本[led](https://docs.code.org/applab/led/)與myLed。
* 測試你的代碼。

### **Student Instructions**

# **Wiring Multiple LEDs**

Using the [createLed()](https://docs.code.org/applab/createled/) block you can connect and control as many LEDs as your board has room for. Each LED needs to be connected to a separate numbered pin, but they can all share the same ground pin.

# **Do This**

Leave the current LED connected to pin 2, but add another one to a numbered pin of your choice. For your new LED:

* Add a var myLed = createLed() block.
* Replace the variable label myLed with a unique label.
* Make sure you're in text mode, not block mode.
* Add an [led.blink()](https://docs.code.org/applab/led.blink/) command.
* Replace the text [led](https://docs.code.org/applab/led/) with your new LED variable.
* Test your code.

*Challenge: Try adding a third LED and make all three LED blink at different intervals.*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/8)

### 學生指示

# **連接多個LED**

使用該[createLed()](https://docs.code.org/applab/createled/)板，您可以連接和控制許多LED ，只要您的電路板放得下。每個LED需要連接到一個單獨的編號引腳，但它們都可以共用同一個接地引腳。

# **做這個**

將當前LED連接到引腳2，但將另一個LED添加到您選擇的編號引腳。對於您的新LED：

* 添加一個var myLed = createLed()塊。
* 改變 myLed 標籤變數成另一個唯一的標籤變數。
* 確保您處於文本模式，而不是區塊模式。
* 添加[led.blink()](https://docs.code.org/applab/led.blink/)命令。
* 用你新的 LED 變數取代[led](https://docs.code.org/applab/led/)。
* 測試你的代碼。

挑戰：嘗試添加第三個LED並使所有三個LED以不同的間隔閃爍。

### **Student Instructions**

# **Smart Bike - Blinkers**

Using your planning guide, wire up the two LEDs that will serve as the blinkers. In order to make sure that the blinkers can be mounted at the end of the handlebars, make sure you wire them so that they can stretch out in opposite directions.

# **Do This**

Using alligator clips, wire, or other conductive material, connect two LEDs to your board, one for each turn signal blinker.

* Hook up each LED to a different numbered pin on the board.
* Use the [createLed()](https://docs.code.org/applab/createled/) block to create an LED object for each blinker.

*Tip: Make your wiring easier by considering how your blinkers will be mounted when selecting a pin to use.*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/9)

### 學生指示

# **智能自行車 - 閃光燈**

使用您的規劃指南，連接兩個將用作閃光燈的LED。為了確保將閃光燈安裝在車把的末端，請確保將它們連接起來，以便它們能夠以相反的方向伸展。

# **做這個**

使用鱷魚夾，電線或其他導電材料，將兩個LED連接到電路板，每個轉向一個信號燈閃爍。

* 將每個LED連接到電路板上不同編號的引腳。
* 使用該[createLed()](https://docs.code.org/applab/createled/)塊為每個閃光燈創建一個LED物件。

提示：在選擇要使用的引腳時，考慮如何安裝閃光燈，使佈線更容易。

### **Student Instructions**

# **Smart Bike - Blinker Controls**

With your turn signal LEDs hooked up, you just need to program some buttons to control them. You may want to place buttons elsewhere on the bike to make controlling your turn signals easier, but for now we'll just use the built in left and right buttons.

# **Do This**

* Add event handlers to blink the left turn signal when the left button is pressed.
* Add event handlers to blink the right turn signal when the right button is pressed.
* Test your code!

*Hint: You'll need to be in text mode to make the blinkers work, since there are no built-in blocks for the elements that you add on to the board.*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/10)

### 學生指示

# **智能自行車 - 閃光燈控制**

當您的轉向信號LED連接起來時，您只需要編程一些按鈕來控制它們。您可能希望將按鈕放在自行車的其他位置，以便更輕鬆地控制轉向燈，但現在我們只需使用內置的左右按鈕。

# **做這個**

* 按下左按鈕時，添加事件處理程序以使左轉向燈閃爍。
* 按下右鍵時，添加事件處理程序以使右轉向燈閃爍。
* 測試你的代碼！

提示：您需要處於文本模式以使閃爍燈工作，因為沒有內置板塊用於添加到電路板上的元素。

### **Student Instructions**

# **Build Your Blinkers**

Using the circuit you just built, return to your smart bike prototype and add the blinkers.

### 學生指示

# **建立你的閃光燈**

## 使用您剛製造的電路，返回您的智慧自行車原型並添加閃光燈。

## 

## **Activity 2 (30-45 min)**

Teaching Tip

**Sharing Boards?** If you are sharing boards among multiple classes, you'll need to take that into consideration before groups start to build their projects. You may want to add a design constraint that the board must be easily removable from the design, or to create group sizes large enough to ensure that each group in every class has access to a board at all times.

### **Building the Prototype**

**Transition:** At this point students should have put together the two additional LED circuits necessary for their turn signals. Now we're going to transition off of the computers for a bit to work on building that circuit into a physical prototype.

**Distribute:** Make available any remaining building materials that you've gathered, such as cardboard and tape.

Teaching Tip

To provide more student ownership and creativity, consider allowing students to modify the form (but not necessary functionality) of the smart bike controller. For example, they may want to turn it into something that the rider could wear, or something that works on skateboards or other modes of transportation. The import element here is that we work in a form that requires adding additional circuits to the board, instead of continuing to work solely within the contrainsts of the board's form.

**Build:** Allow groups some time to build their prototypes, using the board and LED circuits. While there's a lot of potential functionality that we could build at this point based on the guide, make sure that students are focusing on making the turn signals work.

**Test:** Once the basic prototype is built, have students test the code they wrote. It's not uncommon at this point to see new bugs that weren't present when testing the software alone. Encourage students to test all elements of their new circuits, ensuring that all of the connection points are solid, that the circuits are connected to the correct pins, and that the LEDs are oriented correctly.

## **活動2（30-45分鐘）**

## 教學提示

## **共享電路板？**如果您在多個課堂中分享電路板，在分組開始構建專案之前，您需要考慮這一點。您可能希望添加一個設計限制，該設計限制必須能夠輕鬆地從設計電路中移除該，或者創建足夠大的分組大小，以確保班級中的每個組都可以由時間輪流使用電路板。

### **構建原型**

## **轉換：**此時學生應該將轉向信號所需的兩個額外LED電路組合在一起。現在我們將從電腦上的電路成實體原型。

## **分發：**提供您收集的任何剩剩建構材料，例如紙板和膠帶。

## 教學提示

## 為了更能讓學生掌握自主設計及提高創造力，請考慮允許學生修改智慧自行車控制器的表格（但不是必要的功能）。例如，他們可能想把它變成騎車人可以穿的東西，或者是滑板或其他運輸方式的東西。這裡的導入元素是我們需要向電路板添加額外電路的方式，而不是繼續僅在電路板上增加線路。

## **構建：**使用電路板和LED電路，給分組一些時間來構建原型。雖然根據設計指南建立有很多潛在的功能可以在板子上做，但要確保學生專注於轉向信號正常工作。

## **測試：**一旦構建了基本原型，讓學生測試他們編寫的代碼。單獨測試軟件時不存在的新錯誤並不罕見。鼓勵學生測試新電路的所有元件，確保所有連接點都牢固，電路連接到正確的引腳，並確保LED正確定向。

## **Activity 3 (30-45 min)**

### **Adding Inputs**

**Display:** Put up the sketch on the first page of [Physical Prototyping - Project Guide](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/edit) where the whole class can see it. The first prototype that we've constructed takes care of the turn signals, but now we're going to focus on the "horn" feature. Clearly we can use the buzzer to make the horn sound, but how can we control the horn?

**Transition:** Send students back to the online progression to learn about creating button circuits, which they will then add to their smart bike prototypes. This portion of the progression will take them all the way through finalizing their designs and submitting the code.

## **活動3（30-45分鐘）**

### 添加輸入

### **顯示：**將草圖放在[Physical Prototyping - Project Guide](https://docs.google.com/document/d/1MYLYmwNW4DH7n_oCTZFGnYz52e99htwzKTqOcmL5txQ/edit)的第一頁上，全班都可以看到它。我們構建的第一個原型機負責轉向信號，但現在我們將專注於“喇叭”功能。顯然我們可以使用蜂鳴器發出喇叭聲，但我們怎樣才能控制喇叭？

### **轉換：**將學生送回線上課程，了解如何創建按鈕電路，然後將其添加到智能自行車原型中。這部分進展將完成他們的設計並提交代碼。

### 

### **Code Studio levels**

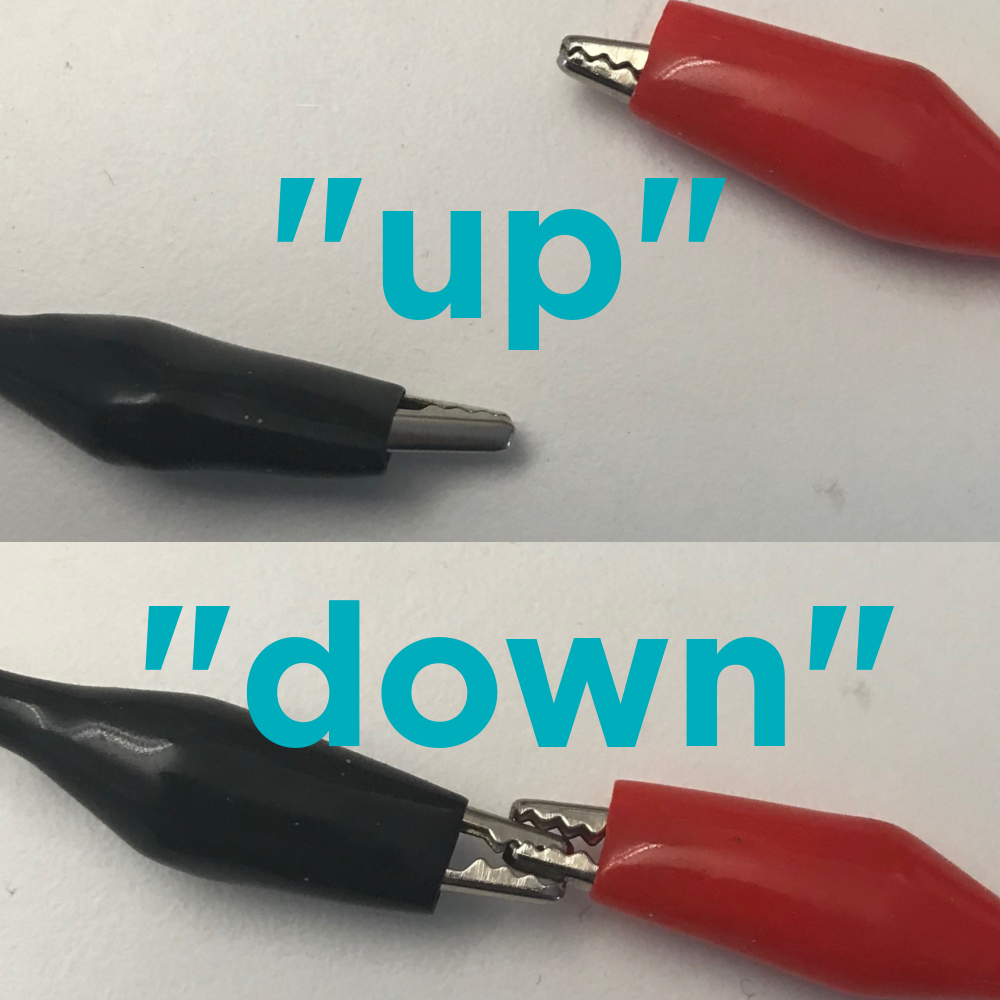
* Levels
* [11](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-11)
* [12](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-12)
* [13](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-13)
* [14](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-14)
* [15](https://curriculum.code.org/csd-18/unit6/15/#level-expando-15-15)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/11)

### **Student Instructions**

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/12)

### **Student Instructions**

****

# **Make Your Own Buttons**

Similar to LEDs, buttons are a really simple circuit that you can add to your board pretty easily. Like LEDs, buttons should be wired from a numbered pin to a ground pin, but unlike LEDs, a button circuit should be disconnected in the middle. When you connect the circuit, it will produce a button press event.

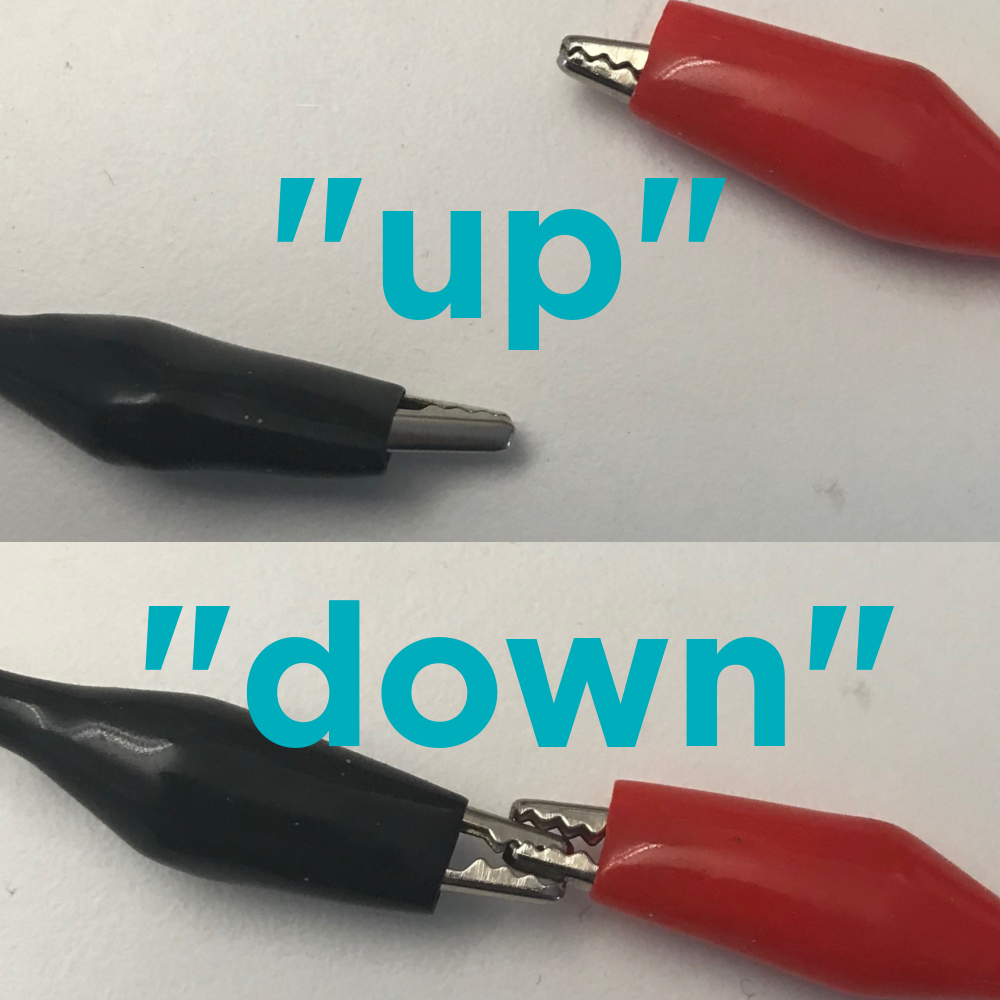
# **Do This**

* Grab two wires.
* Connect one wire to an open numbered pin.
* Connect the second wire to a ground pin.
* Update line 2 so that it's referencing the pin you chose.
* Run the provided code.
* With the program running, touch the unconnected ends of both wires together to "press" the button.

*Tip: A button circuit can be made with many different kinds of materials, as long as they are electrically conductive. Try making buttons with foil, silverware, or paper clips.*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/13)

### 學生指示



# **製作你自己的按鈕**

與LED類似，按鈕是一個非常簡單的電路，您可以很容易地添加到電路板上。與LED一樣，按鈕應從編號引腳連接到接地引腳，但與LED不同，按鈕電路應在中間斷開。當您連接電路時，它將產生按鈕按下事件。

# **做這個**

* 抓兩根電線。
* 將一根電線連接到一個開口編號的引腳。
* 將第二根導線連接到接地針。
* 更新第2行，使其引用您選擇的引腳。
* 運行提供的代碼。
* 程序運行時，將兩根電線的未連接端接觸到“按下”按鈕。

提示：按鈕電路可以用許多不同種類的材料製成，只要它們是導電的即可。嘗試用鋁箔，銀器或紙夾製作按鈕。

### **Student Instructions**

# **Smart Bike - Buzzer**

Now that we know how to add more buttons, you can add a button to control the smart bike's horn.

# **Do This**

Using the button that you've already wired to the board, find a good spot to place your horn button. Then:

* Create a button object for your horn button
* Add an event handler to buzz when button is pressed

*Hint: The button object that you create won't be in the* [*onBoardEvent()*](https://docs.code.org/applab/onBoardEvent/) *dropdown, so you'll need to type the name you've chosen in. Make sure not to use quotation marks!*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/14)

### 學生指示

# **智能自行車 - 蜂鳴器**

### 現在我們知道如何添加更多按鈕，您可以添加一個按鈕來控制智能自行車的喇叭。

# **做這個**

### 使用已經連接到電路板的按鈕，找到放置喇叭按鈕的好位置。然後：

### 為喇叭按鈕創建一個按鈕物件

### 按下按鈕時，向buzz添加事件處理程序

### 提示：您創建的按鈕對像不在[onBoardEvent()](https://docs.code.org/applab/onBoardEvent/)下拉列表中，因此您需要鍵入您選擇的名稱。確保不要使用引號！

### 

### **Student Instructions**

# **Smart Bike - Headlight**

The last part of the smart bike plan that we need to figure out is the automatic headlight.

# **Do This**

First you'll need to decide what to use for your headlight. You could add another LED circuit, or perhaps you can find a way to mount the board that allows for using the color LEDs as a headlight. Once you've figured out the physical layout of your lights, add code to your program that turns on and off the headlight based on how light or dark it is.

*Hint: If you're using the built-in color LEDs, you might need to protect the light sensor to make sure that it's responding to the ambient light level and not the light from the LEDs.*

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/15/puzzle/15)

### 學生指示

# **智能自行車 - 車燈**

我們需要弄清楚智能自行車計劃的最後一部分是自動的大燈。

# **做這個**

首先，您需要決定用於大燈的用途。您可以添加另一個LED電路，或者您可以找到一種安裝電路板的方法，允許使用彩色LED作為前燈。一旦你弄清楚燈光的實際佈局，就可以在你的程序中添加代碼，根據環境的亮度或暗度來打開和關閉大燈。

提示：如果您使用的是內置彩色LED，則可能需要保護光線感應器以確保其反應環境光線而不是LED光線。

### **Student Instructions**

# **Smart Bike - Final Touches**

At this point your smart bike should have all of its basic functionality in place. Now is your chance to add any finishing touches.

# **Do This**

You may want to divide and conquer at this point, allowing some members of your group to focus on the physical aspects of the prototype while others work on improving the code. As this is a prototype, don't worry about making everything perfect, but do try to ensure that the prototype communicates your design well enough to test and get feedback.

### 學生指示

# **智慧自行車 - 最後的檢查**

## **此時，您的智能自行車應具備所有基本功能。現在，您有機會添加任何收尾。**

# **做這個**

## **此時您可能想要分而治之，允許您的組中的某些成員專注於原型的實際設計，而其他成員則致力於改進代碼。由於這是一個原型，不要擔心一切都很完美，但要盡量確保原型能夠很好地傳達您的設計，以便測試並獲得反饋。**

## 

## **Wrap Up (15 min)**

### **Sharing Designs**

**Share:** Give each group an opportunity to share their designs. Take some time to notice and celebrate the differences in design - even in a situation where we are all working from the same plan, each individual will bring their own experiences and perspectives to the design of a product.

Discussion Goal

This discussion is intended to tie the physical design work students are doing now to the work they did in unit 4. Though we haven't explicitly asked students to spend the same amount of energy thinking about users and their needs in this project, it's still an essential element of good design. Push your students to consider how the design of *physical* computing devices brings new challenges and opportunities when it comes to designing for a diverse set of users.

**Discuss:** Using some of the specific design choices as an example, discuss with the class how their design choices impact the usability of their products. What assumptions did we make about our users, and how might our design choices have excluded or disadvantaged different user groups.

## **總結（15分鐘）**

### **分享設計**

**分享：**為每個小組機會分享設計。花一些時間注意並慶祝設計上的差異 - 即使在我們都在同一計劃中工作的情況下，每個人都會將自己的經驗和觀點帶到產品的設計中。

討論目標

本次討論旨在將學生現在正在進行的實際設計工作與他們在單元4中所做的工作聯繫起來。雖然我們沒有明確要求學生在這個項目中花費相同的精力來思考用戶及其需求，但它仍然是良好設計的基本要素。鼓勵您的學生考慮實際計算裝置的設計如何為多樣化的用戶設計帶來新的挑戰和機會。

**討論：**以一些特定的設計選擇為例，與課堂討論他們的設計選擇如何影響其產品的可用性。我們對用戶做出了哪些假設，以及我們的設計選擇如何排除或弱化不同的用戶群。

## **Standards Alignment**

#### **[View full course alignment](https://curriculum.code.org/csd-18/standards/)**

#### **CSTA K-12 Computer Science Standards (2017)**

**AP** - Algorithms & Programming

* **2-AP-10** - Use flowcharts and/or pseudocode to address complex problems as algorithms.
* **2-AP-11** - Create clearly named variables that represent different data types and perform operations on their values.
* **2-AP-14** - Create procedures with parameters to organize code and make it easier to reuse.
* **2-AP-19** - Document programs in order to make them easier to follow, test, and debug.

**CS** - Computing Systems

* **2-CS-02** - Design projects that combine hardware and software components to collect and exchange data.
* **2-CS-03** - Systematically identify and fix problems with computing devices and their components.

**IC** - Impacts of Computing

* **2-IC-21** - Discuss issues of bias and accessibility in the design of existing technologies.

×

#### CSTA K-12計算機科學標準（2017）

#### **AP -演 算法和程式設計**

#### **2-AP-10 - 使用流程圖和/或虛擬代碼將復雜問題作為演算法來解決。**

#### **2-AP-11 - 創建明確命名的變數，表示不同的數據類型並對其值執行操作。**

#### **2-AP-14 - 使用參數創建程序以組織代碼並使其更易於重用。**

#### **2-AP-19 - 文件紀錄程序，以便於跟踪，測試和偵錯。**

#### **CS - 計算系統**

#### **2-CS-02 - 設計專案，結合硬件和軟件組件來收集和交換數據。**

#### **2-CS-03 - 系統地識別和解決計算設備及其組件的問題。**

#### **IC - 計算的影響**

#### **2-IC-21 - 討論現有技術設計中的偏見和可取得性問題。**

#### 

#### **如果您有興趣將Code.org資料用於商業用途，請[與我們聯繫。](https://code.org/contact)**

#### **Lesson Feedback**

&lt;i class='fa fa-cog fa-spin' style="font-size: 100px; text-align: center"&gt;&lt;/i&gt;

Close



If you are interested in licensing Code.org materials for commercial purposes, [contact us.](https://code.org/contact)

# **Lesson 16: Project: Prototype an Innovation**

# **課程16: 專案: 建立創新原型**

#### **App Lab | Maker Toolkit | Project**

## **Overview**

In this final project for the course, students team to develop and test a prototype for an innovative computing device based on the Circuit Playground. Using the inputs and outputs available on the board, groups will create programs that allow for interesting and unique user interactions.

## **概論**

在課程的最終中，學生團隊開發並測試使用Circuit Playground的創新計算設備的原型。 使用實驗板上可用的輸入和輸出，團隊將創建有趣和獨特使用者介面的程式。

## **Purpose**

This lesson is the culmination of Unit 6 and provides students an opportunity to build a Maker Toolkit project of their own from the ground up. This project is an opportunity to showcase technical skills, but they will also need to demonstrate collaboration, constructive peer feedback, and iterative problem solving as they encounter obstacles along the way. This project should be student-directed whenever possible, and provide an empowering and memorable conclusion to the final unit of CS Discoveries.

## **目的**

本課程是第6單元的最後課程，為學生提供了從頭開始構建自己的創客工具組專案的機會。 該專案是展示技術的機會，但他們還需要展示協作，建設性的夥伴回饋，和遇到障礙時返復的解決問題。該專案應盡可能以學生為導向，並為CS Discoveries的最終單元提供一個賦予權力和令人難忘的結論。

## **Agenda**

* [Warm Up (10 min)](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation0)
* [Review Project Guide](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation1)
* [Activity (80-200 min)](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation2)
* [Define - Scope Innovation](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation3)
* [Prepare - Complete Project Guide](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation4)
* [Try - Develop Prototypes](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation5)
* [Reflect - Peer Review](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation6)
* [Iterate - Revise Prototypes](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation7)
* [Wrap Up (30-60 min)](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation8)
* [Share](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation9)
* [Extensions](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation10)
* [Pitch Video](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation11)
* [Marketing Website](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation12)
* [Crowdfunding Campaign](https://curriculum.code.org/csd-18/unit6/16/#project-prototype-an-innovation13)

## **議題**

* 暖身活動（10分鐘）
* 查看專案指南
* 活動內容（80-200分鐘）
* 定義 - 範圍創新
* 準備 - 完整的專案指南
* 嘗試 - 開發原型
* 回饋 - 夥伴回饋
* 重複 - 修改原型
* 總結（30-60分鐘）
* 分享
* 推廣
* 視頻推廣
* 營銷網站
* 集資活動

### **[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/1/)**

## **Objectives**

### **Students will be able to:**

* Independently scope the features of a piece of software
* Prototype a physical computing device
* Implement a plan for developing a piece of software that integrates hardware inputs and outputs

## **課程目標**

### **學生將能夠:**

* 獨立定位一個軟體的功能
* 做出這計算設備的實體原型
* 實作一個計劃，這計劃是用於開發整合硬體輸入和輸出的軟體

## **Preparation**

* Collect materials for physical prototyping, eg.
  + Cardboard
  + Scissors
  + Tape
  + Glue
  + Foil
* Print a copy of [Innovation Prototype - Project Guide](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/edit) for each pair of students
* Print a copy of [Prototype An Innovation - Peer Review](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/edit) for each student
* Print a copy of [Prototype An Innovation - Rubric](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/edit) for each student

## **課前準備**

* 收集實體原型的製作材料，例如
  + 紙板
  + 剪刀
  + 膠帶
  + 膠水
  + 金屬薄片
* 為每組學生印出一份 建立創新原型 - 專案指南 ([Innovation Prototype - Project Guide](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/edit))
* 為每位學生印出一份 建立創新原型 - 夥伴回饋 (Prototype An Innovation - Peer Review) 的副本
* 為每個學生印出一份 建立創新原型 - (Prototype An Innovation - Rubric)

## **Links**

**Heads Up!** Please make a copy of any documents you plan to share with students.

### **For the Students**

* [Innovation Prototype](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/edit) - Project Guide Make a Copy
  + [PDF](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/copy)
* [Prototype An Innovation](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/edit) - Peer Review Make a Copy
  + [PDF](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/copy)
* [Prototype An Innovation](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/edit) - Rubric Make a Copy
  + [PDF](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/copy)

## **相關連結**

注意！請將所有需要給學生的文件先印出來！

給學生的資料

* 建立創新原型 - 專案指南 製作副本
  + [PDF](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/copy)
* 建立創新原型 - 夥伴回饋 製作副本
  + [PDF](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/copy)
* 建立創新原型 - 專欄製作副本
  + [PDF](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/export?format=pdf)
  + [Microsoft Word](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/export?format=doc)
  + [Google Docs](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/copy)

## **Support**

### **[Lesson Forum](https://forum.code.org/c/csd6/)**

### [**Report a Bug**](https://support.code.org/hc/en-us/requests/new?description=Bug%20in%20CS%20Discoveries%202018%20unit%206%20lesson%2016%20curriculum.code.org/csd-18/unit6/16/)

課外支援

討論區

問題回報

# **Teaching Guide**

## **Warm Up (10 min)**

### **Review Project Guide**

**Group:** Place students into groups of 2-4 for this project. At your discretion you may choose to have students form larger groups or work independently.

**Distribute:** Provide a copy of [Innovation Prototype - Project Guide](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/edit) to each group. As a class, review the different steps of the project and where they appear in the project guide. Direct students towards the [Prototype An Innovation - Rubric](https://docs.google.com/document/d/1uJoGRcxOOSDWYr2ZXsFFTFAwx5clFRFl9siw0Xa-vAU/edit) so that they know from the beginning what components of the project you will be looking for.

# **教學指南**

## **暖身活動 (10 分鐘)**

### **查看專案指南**

**分組：將學生分成2-4個小組進行該專案。 您可以自行決定讓學生組成更大的小組或獨立工作。**

**分發：為每個組提供建立創新原型 - 專案指南。 作為一個班級，請查看專案的不同步驟以及它們在專案指南中的顯示位置。 引導學生走向 Prototype An Innovation - Rubric，以便他們從一開始就知道您將要尋找的專案組成部分。**

## **Activity (80-200 min)**

### **Define - Scope Innovation**

**Brainstorm:** Students should spend the first 15-20 minutes brainstorming ideas for innovative devices built around the features of the Circuit Playground. Encourage students to review their prior work in this unit as well as the real world innovations they researched earlier for inspiration.

### **Prepare - Complete Project Guide**

**Distribute:** Make available any construction materials that students may need for building their prototypes. While the focus of this step isn't to build the actual prototype, having these materials available can help with the brainstorming process.

**Circulate:** Once students have discussed their ideas for the project they should complete the [Innovation Prototype - Project Guide](https://docs.google.com/document/d/14fdmAmOJpOJw6KXielJAws1wf2AWEnSqQnZueUC6GvM/edit). While this should be a fairly familiar process, encourage students to make each component as clear and detailed as they can at this point. Planning ahead can help them identify issues in their plan before they'll need to make more significant changes to their code or physical device. Encourage students to use additional paper to sketch out screens for their app or additional views of their physical device.

### **Try - Develop Prototypes**

**Distribute:** Make available the physical prototyping materials (such as cardboard, tape, scissors, etc). Let students know that, just as we used paper prototypes to quickly test software ideas, hardware developers often used cheap materials such as cardboard and tape to quickly iterate on the design of physical devices. While not all student ideas may require a physically prototyped component, you should encourage students to consider how the shape and design (or form factor) or their innovation could impact its usability.

**Transition:** Depending on the nature of their innovations, students may need to spend some time building the physical components of their projects before moving online. When students are ready to program, they can transition to Code Studio. These levels provide some guidance on how students may go about implementing their projects, but are left quite open to allow for a broad range of ideas. If they wish, students can work in a different order than the one suggested in these levels.

### **Code Studio levels**

* Lesson Overview
* [Student Overview](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-1-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/1)

* Design Your Prototype
* [Student Overview](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-2-sfmd)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/2)

## **活動內容（80-200分鐘）**

### **定義 - 創新的範圍**

**腦力激盪：學生應該花前面15-20分鐘的腦力激盪，圍繞在創新設備Circuit Playground的功能上。鼓勵學生回顧他們之前在這個單元中的工作以及他們之前研究的現實世界創新的靈感。**

**準備 - 完成專案指南**

**分發：提供學生構建原型所需的任何材料。雖然這一步的重點不是構建實際的原型，但是提供這些材料可以幫助進行腦力激盪過程。**

**發布：一旦學生討論了他們對專案的想法，他們就應該完成創新原型 - 專案指南。雖然這應該是一個相當熟悉的過程，但鼓勵學生在這一點上使每個組件盡可能清晰和詳細。提前規劃可以幫助他們在需要對程式碼或實體設備進行更重大的更改之前確定計劃中的問題。鼓勵學生使用其他紙張為其應用程式或其實體設備的其他觀點繪製草圖。**

**嘗試 - 開發原型**

**分發：提供實體原型材料（如紙板，膠帶，剪刀等）。讓學生知道，正如我們使用紙質原型來快速測試軟件創意一樣，硬體開發人員經常使用紙板和磁帶等廉價材料來快速迭代實體設備的設計。雖然並非所有學生的想法都需要實體原型組件，但您應該鼓勵學生考慮形狀和設計（或形狀因素）或其創新如何影響其可用性。**

**過渡：根據創新的性質，學生在專案上線之前可能需要花一些時間來構建專案的實體組件。當學生準備好編程時，他們可以轉換到Code Studio。這些**關卡**為學生如何實施他們的專案提供了一些指導，但是保持開放以允許廣泛的想法。如果他們願意，學生可以不按照這些**關卡**建議的順序來工作。**

### **Code Studio 級別**

* 課程概述
* 學生概述

在Code Studio上查看

* 設計原型
* 學生概述

在Code Studio上查看

# **Prototype your own innovation**

Now that you have all the skills you need, it's time to make your own game!

With a partner or team, brainstorm some different physical devices you could prototype. You can build on the programs that you've already made in previous lessons, innovative devices that you've seen in the real world, or any wild ideas you can come up with.

Once you have settled on an idea with your team, fill out the Project Guide with the sketches of the physical components of your device, inputs and outputs that you'll need, and plans for what information you'll need to process.

# **原型化您自己的創新**

現在您已擁有所需的所有技能，現在是時候製作自己的遊戲了！

與合作夥伴或團隊一起，腦力激盪出一些您可以製作原型的不同實體設備。 您可以建立在之前課程中已經完成的程式，您在現實世界中看到的創新設備，或者您可以提出的任何瘋狂想法。

一旦您與團隊確定了想法，請填寫專案指南，其中包含設備實體組件的草圖，您需要的輸入和輸出，以及您需要處理哪些信息的計劃。

## **Innovation Ideas**

To help kick off your brainstorming, check out these example projects:

## **創新理念**

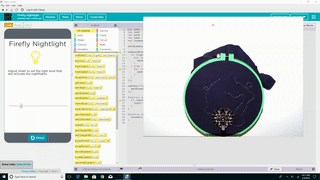
為了幫助您開始腦力激盪，請查看以下範例專案:

### **Automatic Night Light**

Using conductive thread, LEDs, and the light sensor, create a decorative piece of wall art the becomes a night light when the lights go out. This one is designed to look like flickering fireflies.

### **自動夜燈**

使用導電線，LED和光感器，創造一個裝飾性的牆藝術品，當燈光熄滅時變成夜燈。 這個看起來像閃爍的螢火蟲。

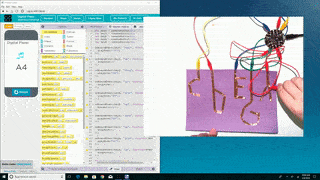


### **Musical Instrument**

Design a musical instrument that can be played any way you like, using sensors or buttons for input. In this case we created five new buttons using conductive copper tape, though you could also use aluminum foil or any other conductive material.

### **樂器**

使用偵測器或按鈕進行輸入，設計可以任何方式播放的樂器。 在這種情況下，我們使用導電銅帶創建了五個新按鈕，但您也可以使用鋁箔或任何其他導電材料。

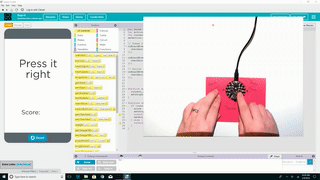


### **Bop-it**

Make a game! This one uses an array to store the different commands, which are picked at random each time you correctly follow the instructions on screen.

BOP-它

製作遊戲！ 這個使用陣列來存儲不同的命令，看看你是否可以每次正確地按照螢幕上隨機選出的說明做出操作



* User Interface

# **使用者介面**

* [3](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-3)
* [4](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-4)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/3)

### **Student Instructions**

學生操作指南

# **Designing the User Interface**

Before worrying about the code for your program, start by laying out the user interface. As you add elements in Design Mode, make sure to give everything a meaningful ID. Even if you don't plan on accessing an element with code, making sure that all of your IDs are descriptive will help make your program more readable and easier to program.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/4)

設計使用者介面  
在思考程式碼之前，先從規劃使用者介面開始。在設計模式中添加元素時，請確實為所有元素提供有意義的ID。即使您不打算使用程式碼來存取這元素，也要確保所有ID都是描述清楚的，這將有助於使您的程式更易讀，更容易編寫。

### **Student Instructions**

學生操作指南

# **Wiring Up the UI**

With your user interface in place, you can now add event handlers for your interface elements. At this point you may want to just include [console.log()](https://docs.code.org/gamelab/console.log/) commands to make sure that your events are working as expected - you can add the functional code later one.

# **連接使用者介面**

使用者介面設計好之後，現在可以為使用者介面元素添加事件處理程式。此時，你可能只想加一個 console.log（）指令以確定事件會如預期地工作 - 你可以稍後再添加相關的功能程式碼。

* Board Input and Output

# **實驗板的輸入與輸出**

* [5](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-5)
* [6](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-6)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/5)

### **Student Instructions**

學生操作指南

# **Board Input Events**

Now you can add the events which will take input from the board. Again, feel free to just use [console.log()](https://docs.code.org/gamelab/console.log/) statements at this point to ensure that your events are working.

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/6)

實驗板的輸入事件  
現在你可以添加一個可從電路板輸入的事件。同樣的，此時只需使用console.log（）指令即可確定你的事件是否正常運行。

### **Student Instructions**

學生操作指南

# **Board Output**

If you haven't already, use some of the output elements on the board to communicate useful information to the user.

實驗板的輸出

如果您尚未加上輸出元素，那就加上實驗板上的一些輸出元素，藉以傳達想表達的信息。

* Finishing Touches

# **最後潤飾**

* [7](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-7)
* [8](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-8)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/7)

### **Student Instructions**

學生操作指南

# **Functions**

As you continue to work through your program, look for patterns that can be abstracted into functions. Can you find code that is repeated multiple places, or chunks of code that could be better understood of given a name? Don't forget that you can use parameters to write functions for repeated code that is *mostly* the same, but varies a bit (such as setting color on multiple LEDs).

## **函數**

當您繼續完成程式碼的同時，也請尋找可以抽象化為函數的程式區塊。可否找到重複多個地方的程式區塊，或是有適合命名來幫助理解的程式區塊？不要忘記，可以使用參數為重複的程式碼編寫函數。這些重複的程式碼是指大致相同，而僅有一點點不同的程式碼（例如在多個LED上設置顏色）。

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/8)

### **Student Instructions**

學生操作指南

# **Finishing Touches**

Add any finishing touches still needed to make your innovation awesome. If you have time, test out your innovation with some potential users to gather feedback and find opportunities for improvement. When you're totally finished, click the "Submit" button to turn it in!

最後潤飾  
添加任何仍然需要的潤飾，以使你的創作更棒。如果你還有時間，請與一些潛在使用者一起測試你的創新作品，以收集回饋並找到改進的機會。當你完成後，點擊“提交”按鈕將其打開！

* Reflection
* [9](https://curriculum.code.org/csd-18/unit6/16/#level-expando-16-9)
* (click tabs to see student view)

[View on Code Studio](https://studio.code.org/s/csd6-2018/stage/16/puzzle/9)

### **Student Instructions**

學生操作指南

### **Reflect - Peer Review**

**Group:** Pair up groups to review each other's projects.

**Distribute:** Give each student a copy of [Prototype An Innovation - Peer Review](https://docs.google.com/document/d/1pPTgdSLwAHs7AutiL5fBcHh2s_g5PrxEv0do5PHnvxU/edit). Students should spend 15 minutes reviewing the other group's project and filling out the peer review guide.

回饋 - 夥伴回饋  
  
分組：兩兩配對分組，互相審查彼此的專案。  
分發：給每個學生一份 Prototype An Innovation - Peer Review。學生應花15分鐘時間審查另一組的專案並填寫夥伴回饋指南。

Teaching Tip

If you have the time for it, this can be a good opportunity to pull in the user testing process that students learned in Unit 4.

教學建議

如果你有時間，這可以是一個很好的機會，可以導入在第4單元所學到的使用者測試流程。

### **Iterate - Revise Prototypes**

**Circulate:** Students should complete the peer review guide's back side where they decide how to respond to the feedback they were given. They should then use that feedback to improve their game.

重複 - 修改原型  
循環：學生應該完成夥伴回饋指南的背面，他們決定如何回應所收到的回饋。然後使用這些回饋來改進他們的遊戲。

## **Wrap Up (30-60 min)**

### **Share**

**Share:** Give students a chance to share their innovations, either within the class or to a broader audience. If you choose to let students do a more formal presentation of their projects the project guide provides students a set of components to include in their presentations including:

* The original innovation they set out to build
* A description of the programming process including at least one challenge they faced and one new feature they decided to add
* A description of the most interesting or complex piece of code they wrote
* A live demonstration of the actual innovation

Send students to Code Studio to complete their reflection on their attitudes toward computer science. Although their answers are anonymous, the aggregated data will be available to you once at least five students have completed the survey.

Teaching Tip

**Celebrate:** As this is the culminating project for the entire course, consider going big with this share out. Bring in parents and administrators, or even host a after school event to provide students with a real audience to share their accomplishments with.

## 總結（30-60分鐘）

### 分享

分享：讓學生有機會在課堂上或對更多的聽眾分享他們的創作。如果你想讓學生做更正式地展示他們的專案，專案指南可以幫學生提供展示文稿中的大綱，其中包括：

* 他們打算建立的原始創作想法
* 編寫程式過程的描述，包括他們面臨的挑戰至少一個以及他們決定添加新功能至少一個
* 描述他們編寫的最有趣或最複雜的程式碼
* 現場展示他們實際的創作

將學生送到Code Studio，完成他們對計算機科學態度的回饋。雖然他們的回答是匿名的，只要至少有五名學生完成問卷調查，你將可以獲得匯總的數據。

### 教學建議

慶祝：由於這是整個課程的最後專案，可以考慮做更大規模的分享。甚至邀請家長和主管舉辦課後活動，為學生提供真實的聽眾分享他們的成就。

## **Extensions**

### **Pitch Video**

Record and edit a short video to pitch your innovation.

### **Marketing Website**

Using Web Lab, design a website to market your innovation.

### **Crowdfunding Campaign**

Have students design a crowdfunding campaign (along the lines of Kickstarter or Indiegogo) for their innovations. This could include:

* Design mockups for the final product
* A rough cost analysis for production
* A short pitch video

## 推廣

### 視頻推廣

錄製和編輯視頻以推廣你的創作

### 營銷網站

使用Web Lab，設計一個網站來推廣您的創作

### 集資活動

讓學生為他們的創作設計一個集資活動（沿著Kickstarter或Indiegogo的路線）。這可能包括：

* 為最終產品設計模型
* 生產的粗略成本分析
* 準備簡短推廣視頻

## 

## **Standards Alignment**

#### **[View full course alignment](https://curriculum.code.org/csd-18/standards/)**

#### **CSTA K-12 Computer Science Standards (2017)**

**AP** - Algorithms & Programming

* **2-AP-10** - Use flowcharts and/or pseudocode to address complex problems as algorithms.
* **2-AP-11** - Create clearly named variables that represent different data types and perform operations on their values.
* **2-AP-12** - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
* **2-AP-13** - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
* **2-AP-15** - Seek and incorporate feedback from team members and users to refine a solution that meets user needs.
* **2-AP-16** - Incorporate existing code, media, and libraries into original programs, and give attribution.
* **2-AP-17** - Systematically test and refine programs using a range of test cases.
* **2-AP-18** - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.
* **2-AP-19** - Document programs in order to make them easier to follow, test, and debug.

**CS** - Computing Systems

* **2-CS-01** - Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices.
* **2-CS-02** - Design projects that combine hardware and software components to collect and exchange data.
* **2-CS-03** - Systematically identify and fix problems with computing devices and their components.

## 標準對照

#### 查看完整課程對照