



#CodeTheirDreams



# Code Their Dreams ปั้นโปรแกรมเมอร์รุ่นเยาว์

คู่มือเรียนรู้การเขียนโปรแกรมด้วย



CODE  
THEIR  
DREAMS  
CDG

# สารบัญ

หัวข้อ	หน้า
● เกี่ยวกับ Scratch	1
● ส่วนประกอบของ Scratch	1
● การกำหนดคำสั่งให้กับเหตุการณ์	26
● การกำหนดตัวแปรและการคำนวณ	38
● การสร้างเงื่อนไขและการวนซ้ำ	54
● การออกแบบและสร้างโปรแกรม	72

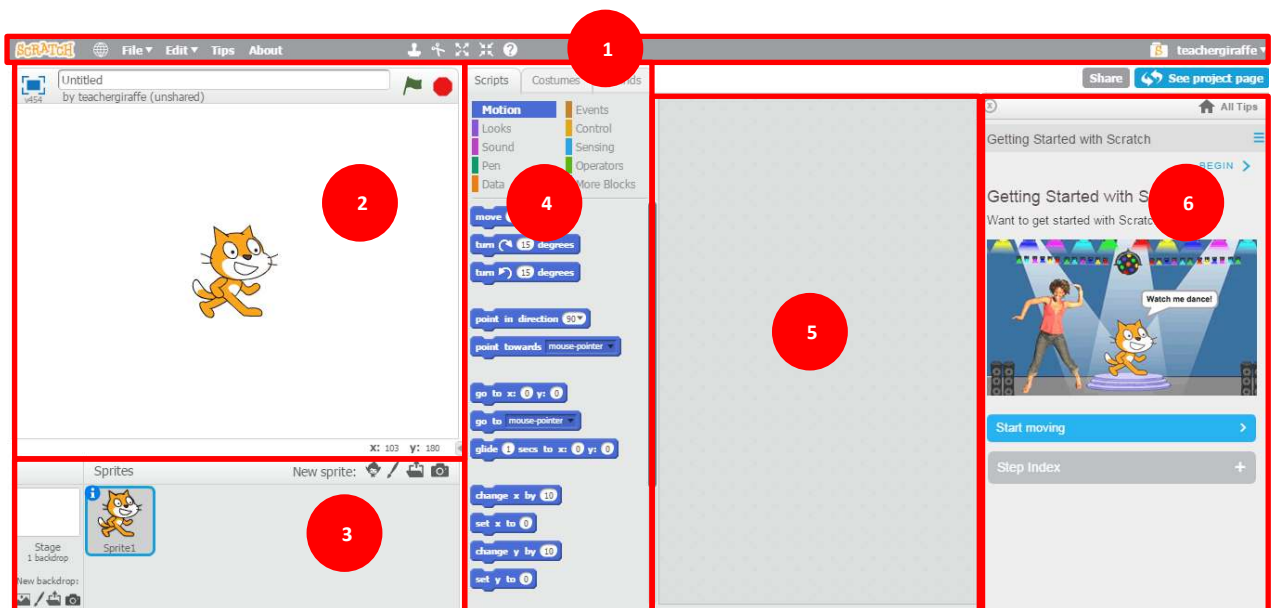
## เกี่ยวกับ Scratch

Scratch ทำให้เราสามารถเขียนโปรแกรมที่สามารถตอบโต้ได้ในหลายรูปแบบทั้งการสร้างเรื่องราว เกม การทำภาพเคลื่อนไหว และอื่น ๆ อีกมากมาย ผ่านลิงก์นี้ <http://scratch.mit.edu> และสามารถแชร์สิ่งที่เราสร้างขึ้นมากับเพื่อน ๆ ในสังคมออนไลน์ของเราได้

Scratch ทำให้เราได้คิดอย่างสร้างสรรค์ มีเหตุผล เป็นระบบ และสามารถทำงานร่วมกันได้ดีขึ้น ซึ่งเป็นทักษะที่ควรจะมีในศตวรรษที่ 21

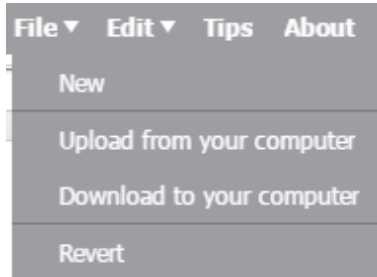
Scratch เป็นโครงการของ Lifelong Kindergarten Group ที่ MIT Media Lab ซึ่งเปิดให้คนทั่วไปใช้ได้โดยไม่เสียค่าใช้จ่าย

## ส่วนประกอบของ Scratch



### 1. Menu Bar แถบเมนูด้านบนสุดของหน้าต่าง Scratch ประกอบด้วย

-  สำหรับเปลี่ยนภาษา โดยโปรแกรม Scratch มีภาษาไทย ให้เลือกได้ด้วย



- เมนู File ประกอบไปด้วยเครื่องมือย่อย
  - “New” ใช้สำหรับสร้างไฟล์ Scratch ใหม่
  - “Upload from your computer” สำหรับอัปโหลด Scratch จากคอมพิวเตอร์ของเรา
  - “Download to your computer” สำหรับดาวน์โหลด Scratch ลงเครื่องของตัวเอง
  - “Revert” สำหรับย้อนไฟล์ปัจจุบันไปที่จุดเริ่มต้น





- เมนู Edit ประกอบด้วยเครื่องมือย่อย
  - “Undo” ใช้สำหรับการนำสิ่งที่เพิ่งลบไปกลับมา คล้าย ๆ คำสั่ง Undo
  - “Small stage layout” ใช้สำหรับการย่อหน้าต่าง Stage ให้เล็กลง เพื่อทำให้หน้าต่าง Script Area ใหญ่ขึ้น เห็น Block ที่เราสร้างได้กว้างขึ้น ซึ่งเราสามารถเลือกที่เครื่องมือนี้อีกครั้งเพื่อทำให้หน้าต่าง Stage กลับมามีขนาดเท่าเดิม

- “Turbo mode” ใช้สำหรับการแสดงผลโปรแกรมเร็วขึ้น จากการลดระยะเวลาหน่วงระหว่าง Block ลง เราสามารถทำให้กลับสู่โหมดปกติ โดยกดที่ปุ่มนี้เหมือนเดิม

- **Tips** ใช้สำหรับเรียกดูหน้าจอกทางด้านขวา สำหรับเรียนรู้วิธีการใช้ Scratch ด้วยตนเอง step-by-step

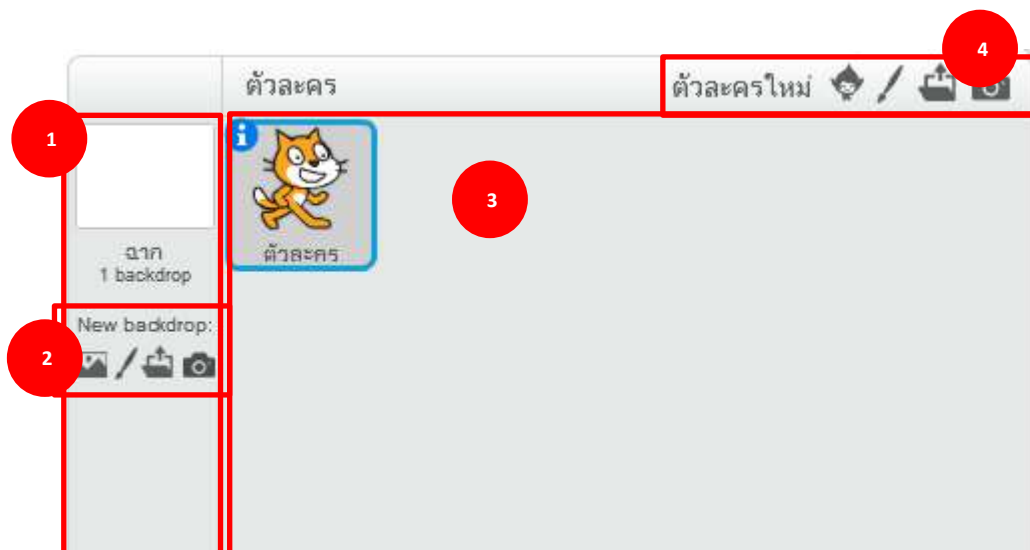
- **About** ใช้สำหรับไปที่หน้าจอ about ของ Scratch

(<https://scratch.mit.edu/about/>)

2. **Stage** เวทีสำหรับการจัดวางตัวละคร และแสดงพื้นหลัง ขณะที่โปรแกรมทำงานจะแสดงผลตรงหน้าต่างนี้ โดย ปุ่มธงเขียว  ใช้สำหรับการเริ่มโปรแกรม และ ปุ่ม 8 เหลี่ยมสีแดง  ใช้สำหรับหยุดโปรแกรม

### 3. Sprite List

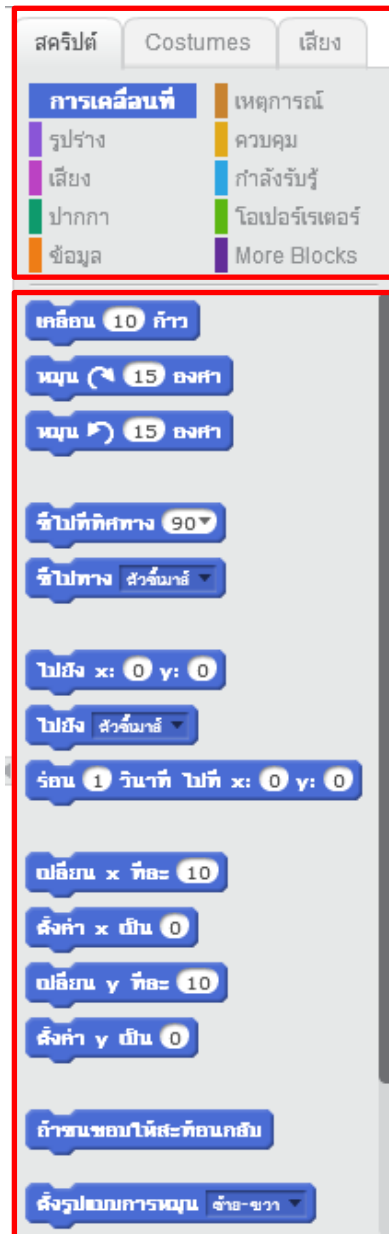
เป็นส่วนสำหรับการเก็บตัวละคร ชั้นส่วนต่าง ๆ ในโปรแกรมของเรา โดยเราสามารถเลือกเพิ่มตัวละครและชั้นส่วนต่าง ๆ รวมถึงฉากหลังได้จากส่วนนี้



1. ส่วนสำหรับการแสดงฉากหลังปัจจุบัน การเพิ่มฉากหลัง และการ set ฉากหลังใหม่
2. ส่วนนี้เราสามารถเพิ่มฉากหลังใหม่ได้ 4 วิธี ได้แก่ การเลือกจากรูปที่มีมาให้จากโปรแกรม การวาดเองโดยโปรแกรมการวาดสำเร็จรูปที่มีให้ การเลือกจากรูปในเครื่องของเรา และการถ่ายรูปผ่านกล้องทันที
3. ส่วนตัวละครและชิ้นส่วนต่าง ๆ ของโปรแกรมที่ถูกเพิ่มเข้ามาจะมาอยู่ตรงส่วนนี้
4. ส่วนเพิ่มตัวละครใหม่สามารถทำได้ 4 วิธี ได้แก่ การเลือกตัวละครจากที่มีอยู่ในโปรแกรม การวาดเองโดยใช้โปรแกรมสำเร็จรูปที่มีให้ การเลือกตัวละครจากภาพในเครื่องของเรา และการถ่ายภาพตัวละครผ่านกล้องเราทันที

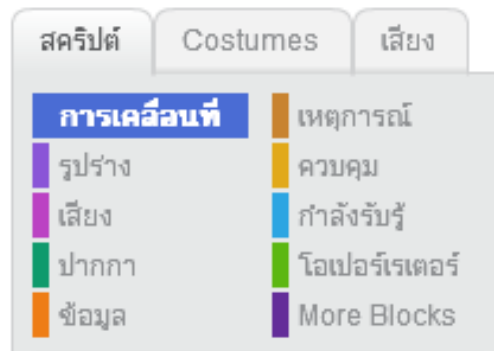
#### 4. Blocks Palette

เป็นส่วนของคำสั่งที่เราจะนำไปประกอบกันเป็นโปรแกรมแบ่งออกเป็นหมวดหมู่ต่าง ๆ ตามประเภทการใช้งาน ประกอบด้วย 2 ส่วนใหญ่ ๆ คือ ส่วนหมวดหมู่ด้านบน และ ส่วนตัวคำสั่งด้านล่าง



ส่วนด้านบน แบ่งออกเป็น 3 เมนู ได้แก่

1. **สคริปต์** ใช้เลือกหาคำสั่งเพื่อนำมาต่อกันเป็นโปรแกรม แบ่งออกเป็น 10 หมวดด้วยกัน แต่ละหมวดก็จะมี Block คำสั่ง การกระทำที่แตกต่างกัน ดังนี้



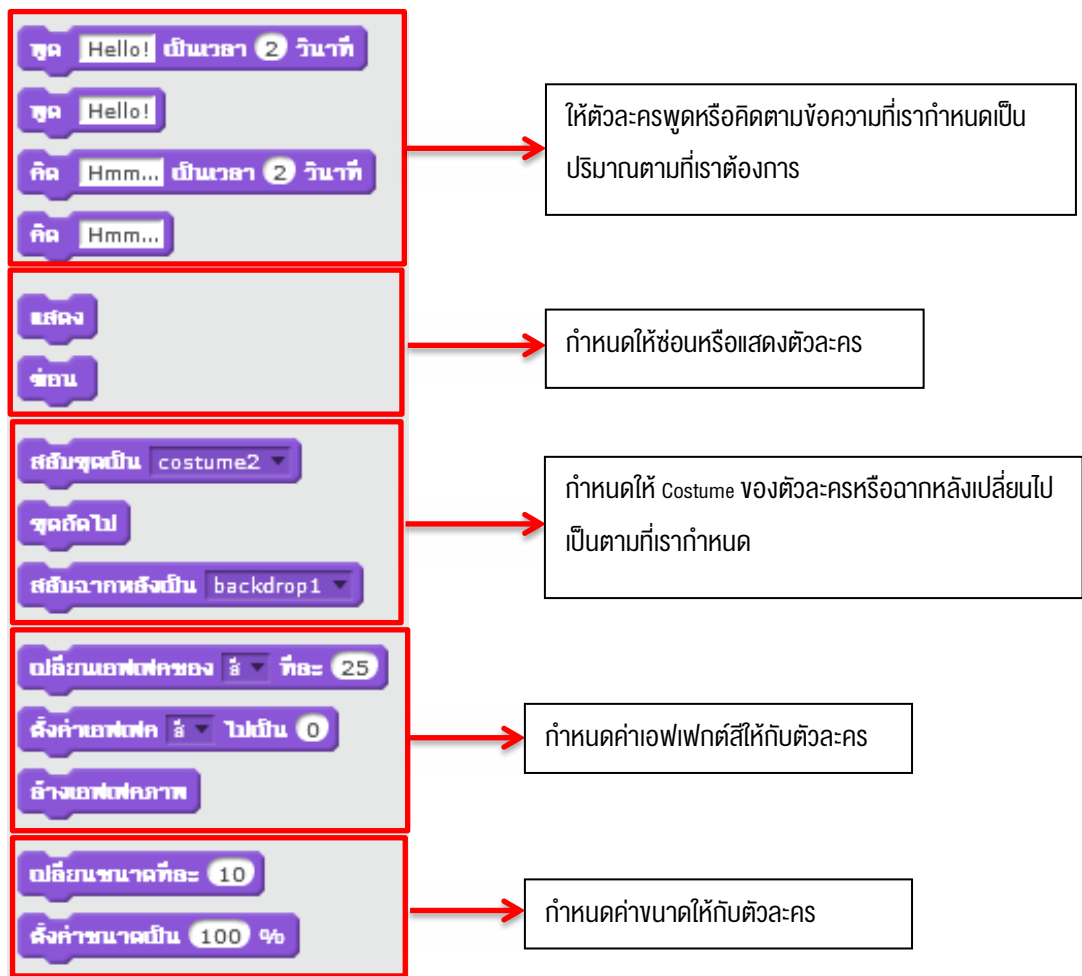
1. หมวดการเคลื่อนที่ เราสามารถทำให้ตัวละครเคลื่อนไหวได้ตามที่เราต้องการประกอบไปด้วย

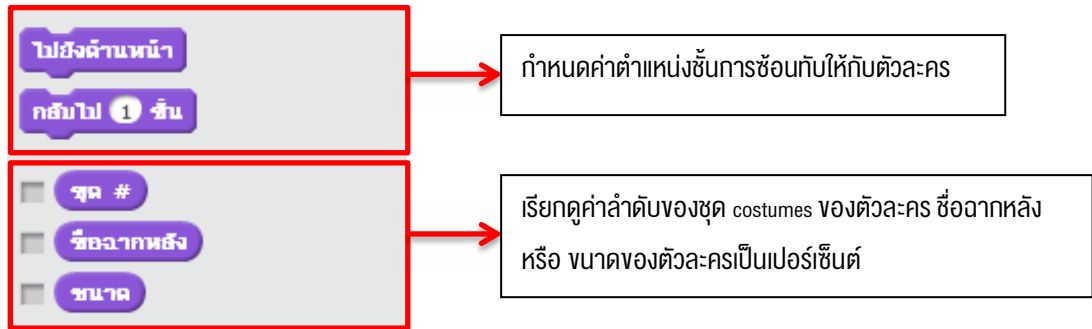




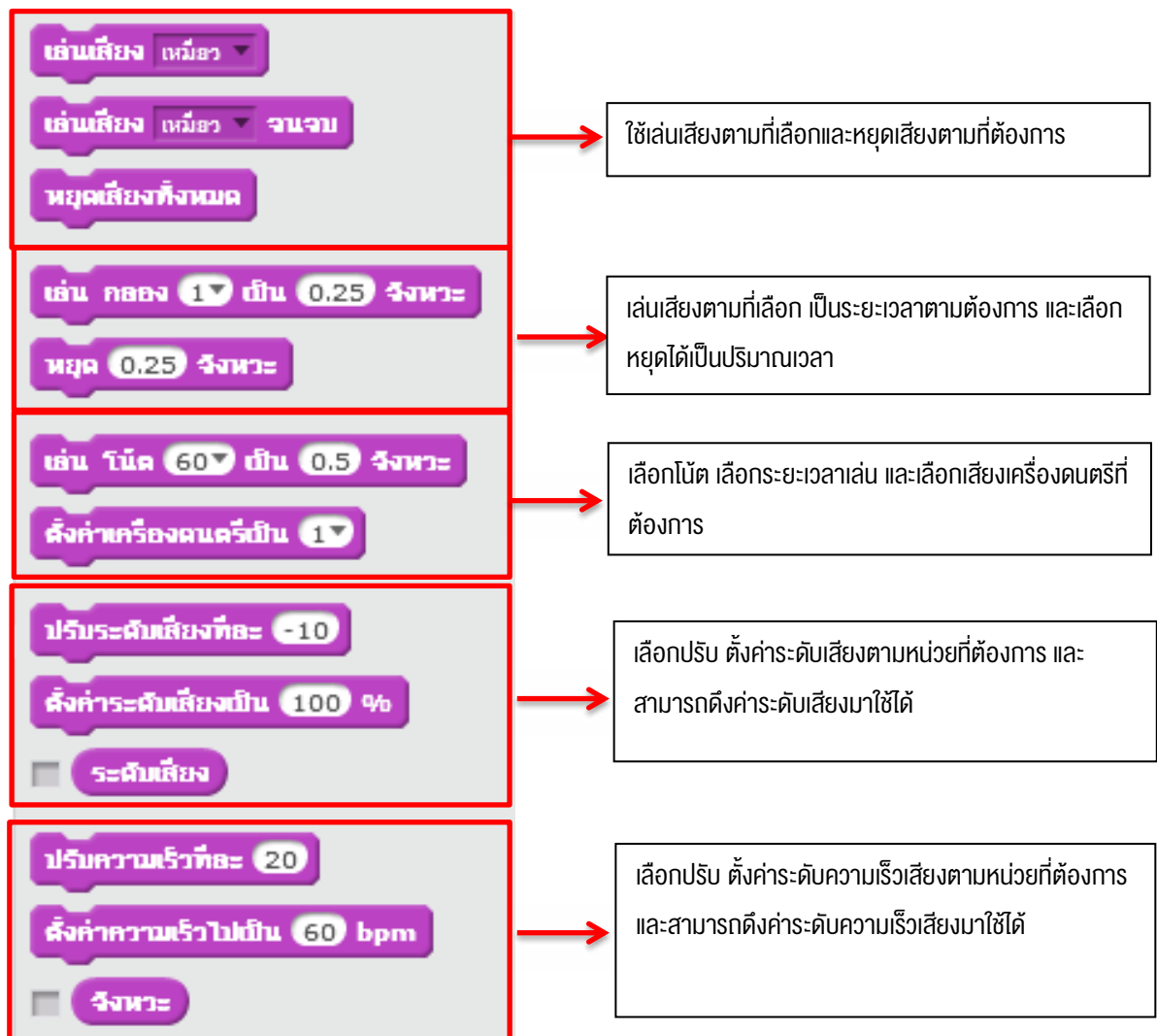


2. หมวดการรูปร่าง เราสามารถเปลี่ยนแปลงตัวละครและฉากให้เหมาะกับสถานการณ์ได้ ประกอบด้วย

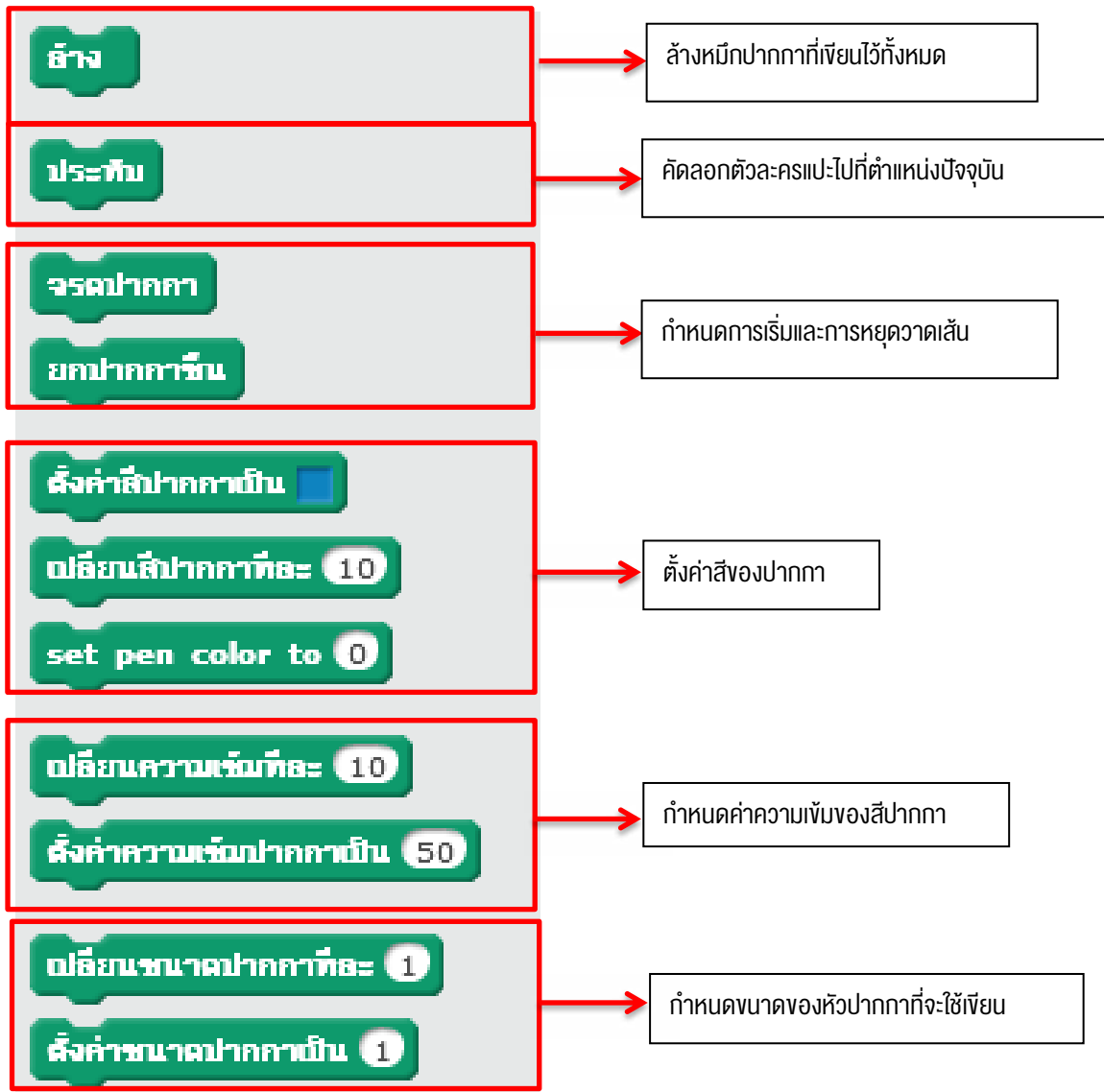




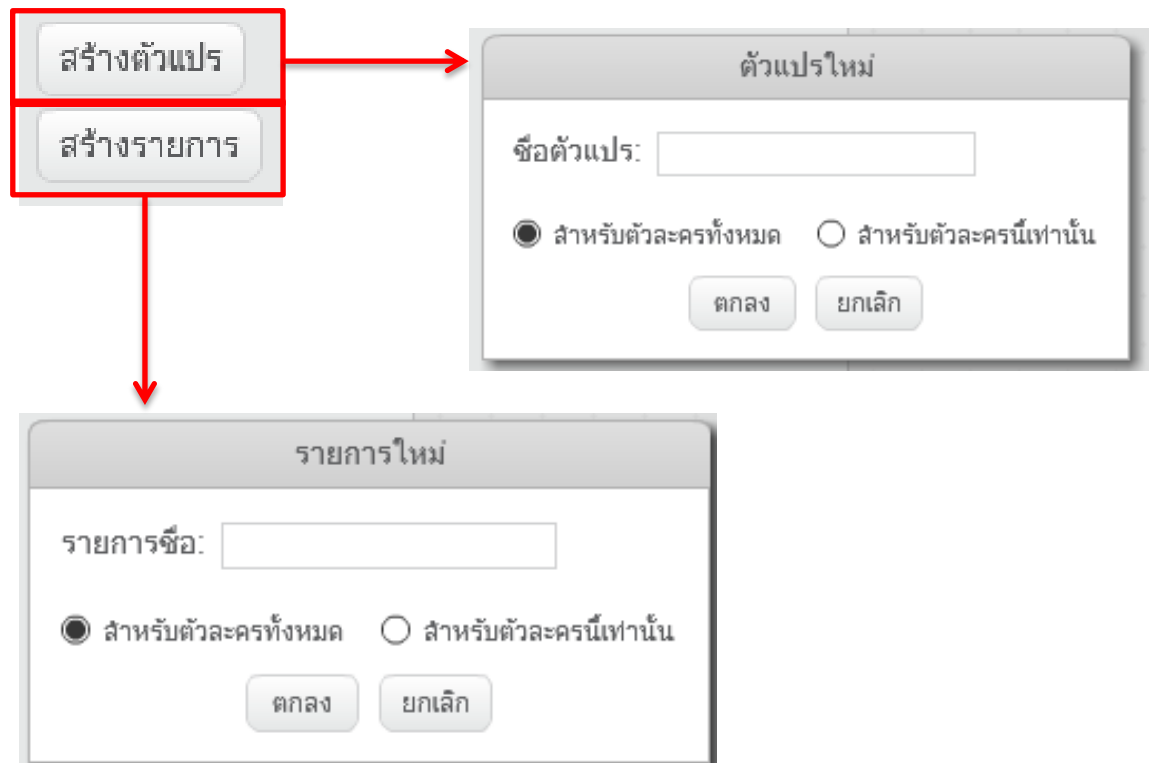
### 3. หมวดเสียง ทำให้เราเล่นเสียงได้ตามประเภทและจังหวะที่เราต้องการ ประกอบด้วย



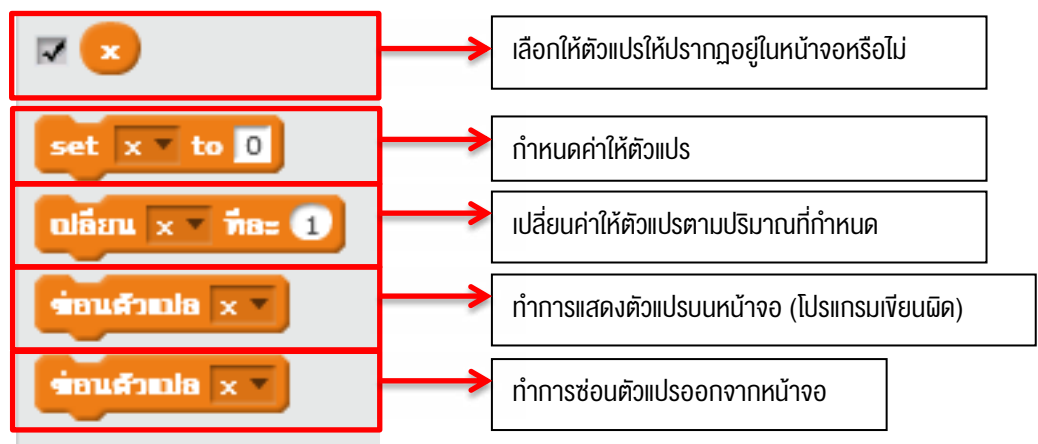
4. หมวดปากกา สามารถวาดเส้นตามการเคลื่อนที่ของตัวละคร ประกอบด้วย



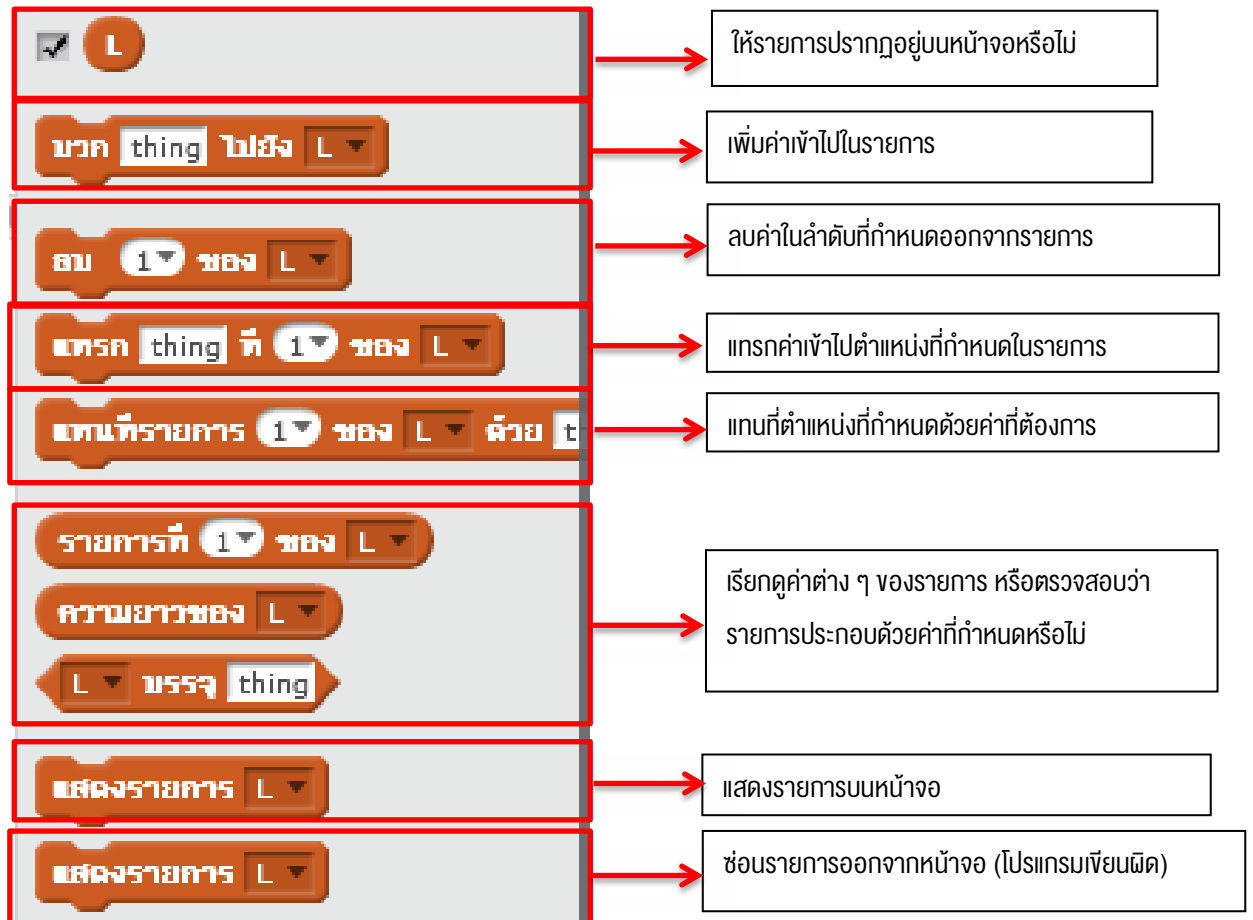
5. หมวดข้อมูล มีการเก็บข้อมูลและการเรียกใช้ข้อมูล มี 2 รูปแบบ ประกอบด้วย



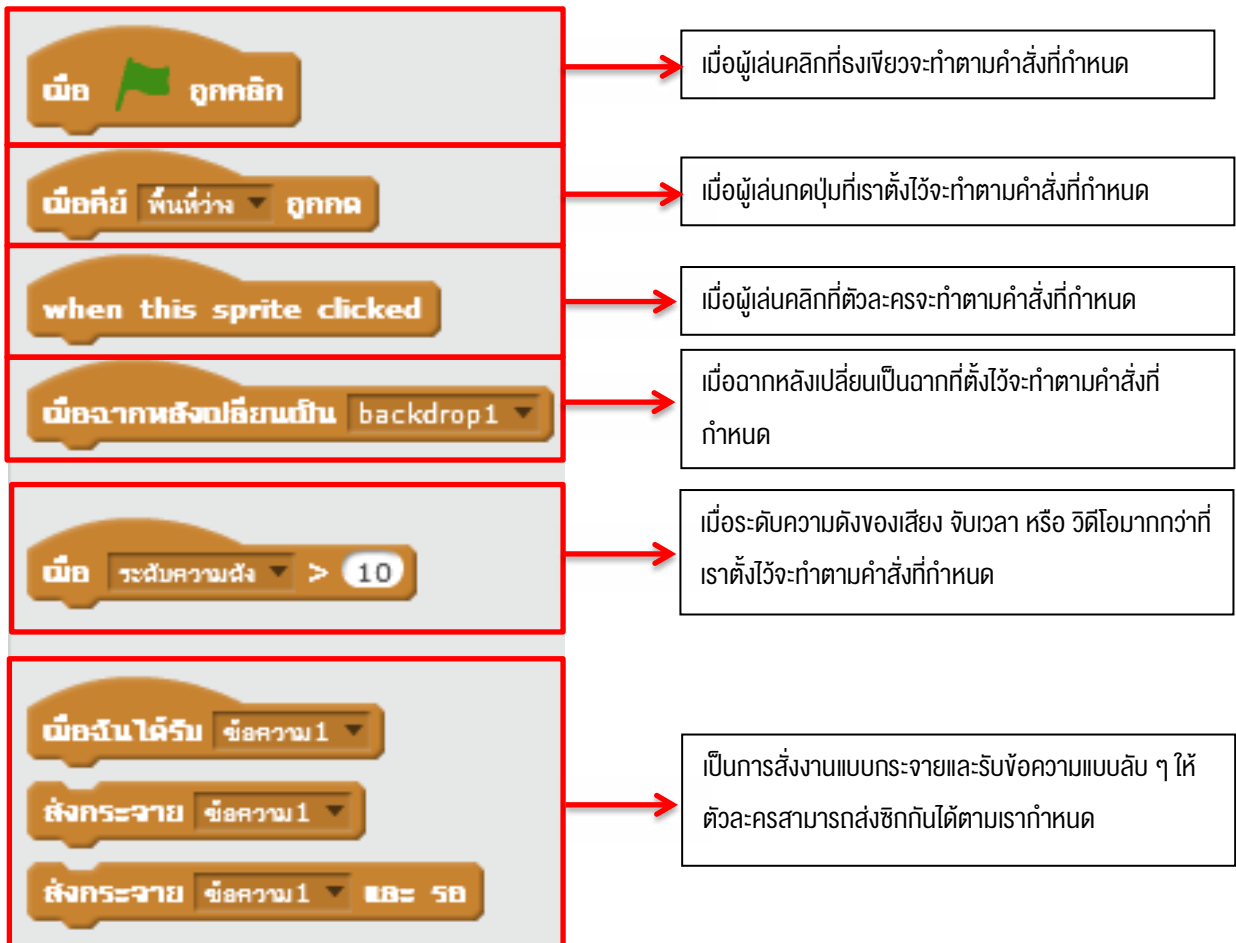
1. การสร้างตัวแปร คือตัวเก็บค่าที่เราสามารถเรียกมาใช้ได้ตลอด สมมุติเราสร้างตัวแปรชื่อ x



2. การสร้างรายการ ใช้สำหรับการต้องการเก็บค่าหลาย ๆ ค่าเป็นลำดับ  
 สมมุติให้ชื่อรายการ L



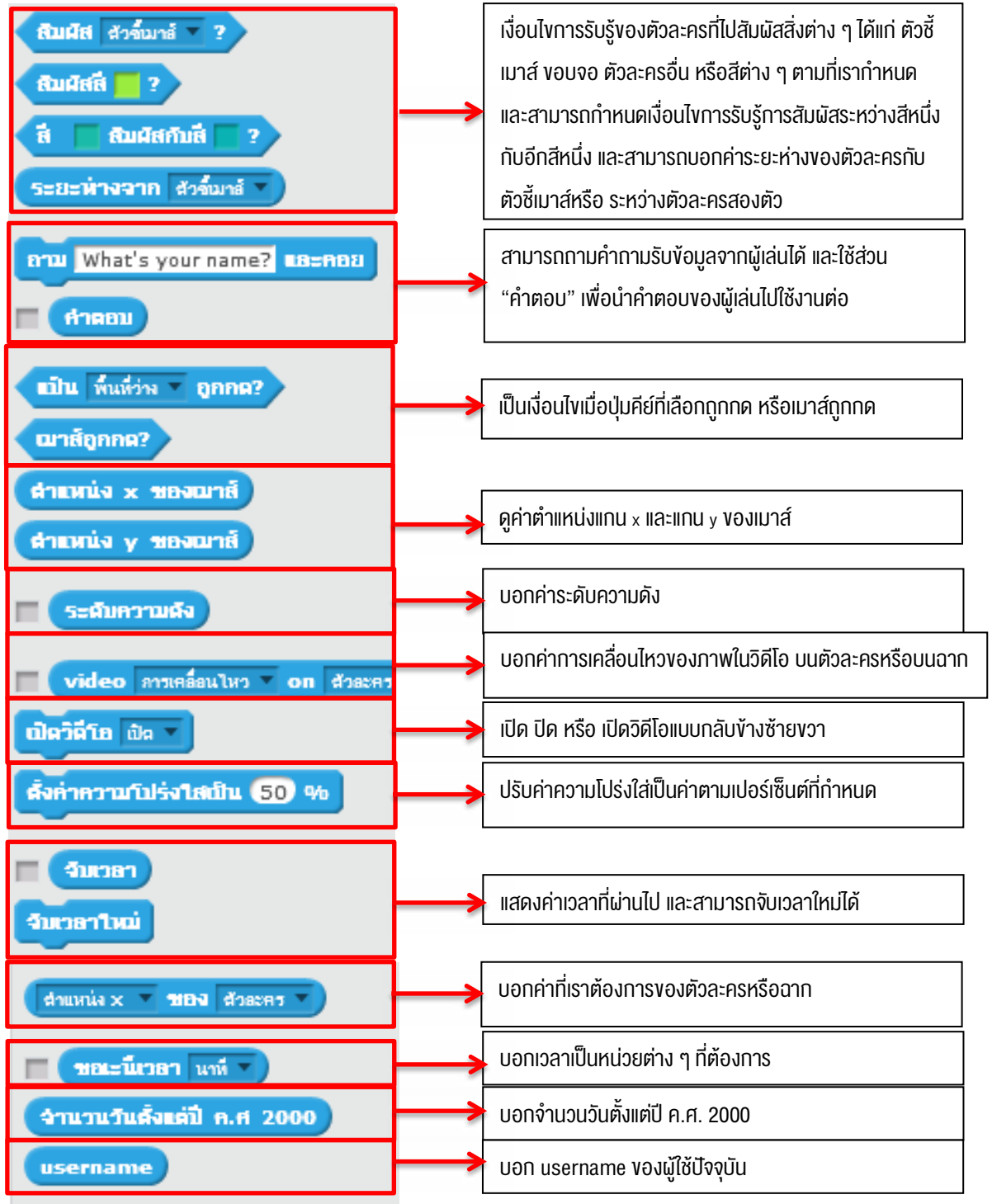
6. หมวดเหตุการณ์ เป็นหมวดที่เกี่ยวข้องกับการกระทำเมื่อมีเหตุการณ์ตามที่กำหนด ประกอบด้วย



7. หมวดควบคุม ช่วยกำหนดการไหลของคำสั่งให้ทำคำสั่งใดตอนไหน ทำซ้ำ หรือ ไม่ทำเมื่อไหร่ ประกอบด้วย

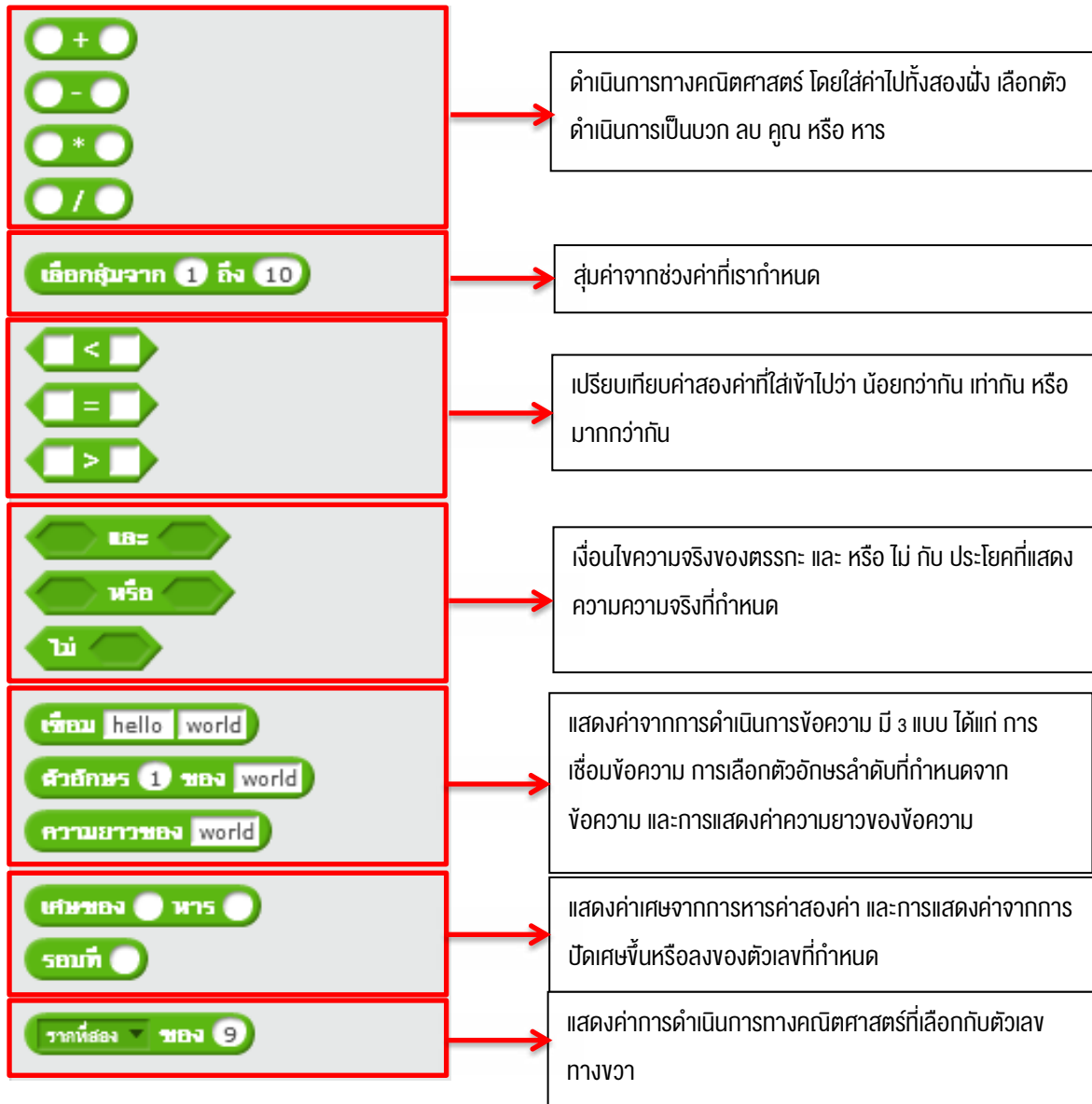


8. หมวดคำสั่งรับรู้ คำสั่งในหมวดนี้จะเหมือนไวยากรณ์และค่าแสดงการรับรู้ต่าง ๆ ของตัวละคร ประกอบด้วย





9. หมวดโอเปอร์เรเตอร์ ใช้สำหรับการดำเนินการทางคณิตศาสตร์ ตรรกศาสตร์ และดำเนินการกับข้อความ ประกอบด้วย

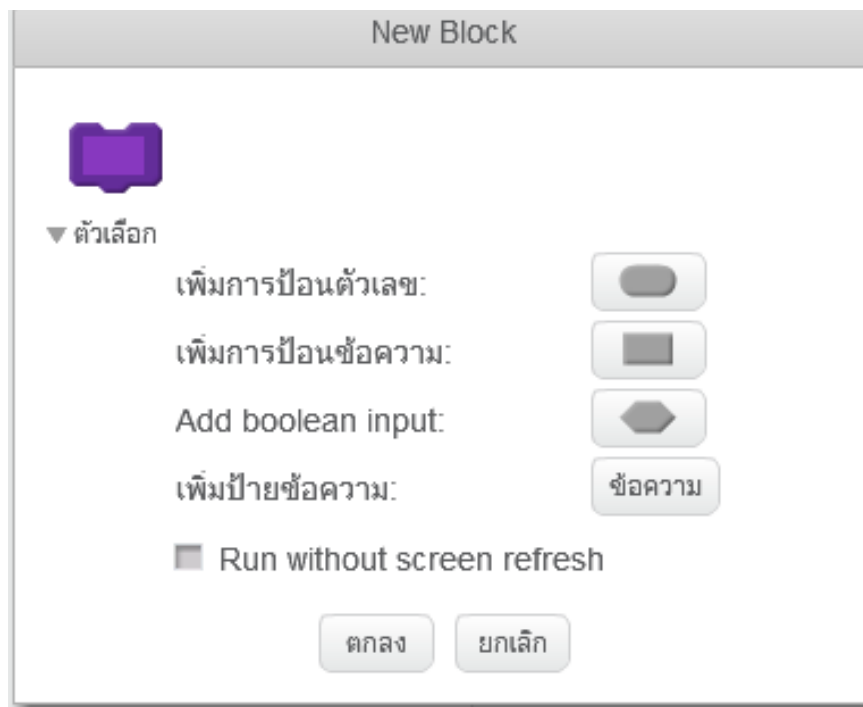


10. หมวด More Blocks ใช้สร้าง Block คำสั่งเพิ่มเติม และเพิ่มส่วนขยาย



1. ส่วน Make a Block คือการสร้าง Block คำสั่งใหม่ที่เราต้องการ โดยการสร้างชุด Block คำสั่ง ช่วยให้ไฟล์โปรแกรมมีขนาดเล็กลง และเมื่อต้องการใช้ Block คำสั่งใหม่ จากตรรกะของโปรแกรม เราสามารถเรียกได้เลยโดยไม่ต้องลาก Block คำสั่งย่อย ๆ มาสร้าง หน้าตาการสร้าง Block คำสั่งใหม่ ประกอบด้วย

1. ส่วนการสร้างหน้าตา Block ใหม่

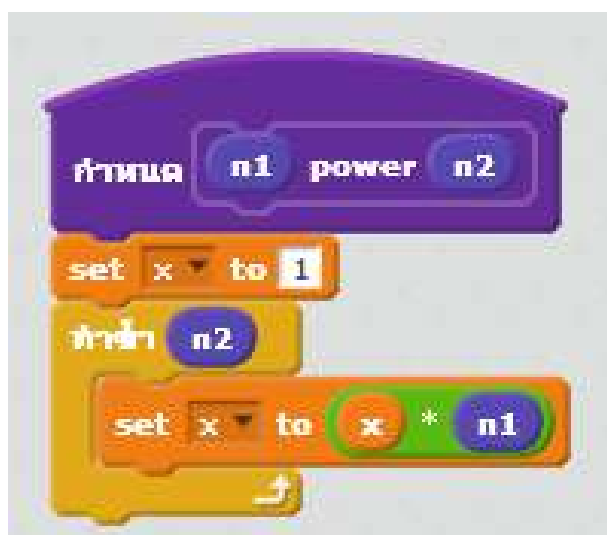


ตัวอย่าง การสร้าง Block สำหรับการหาค่าเลขยกกำลัง (power)

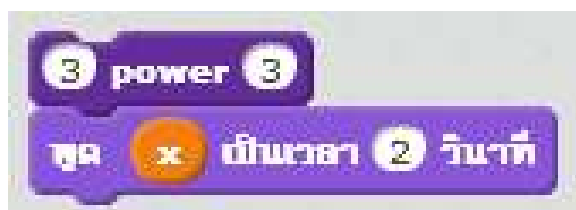


2. ส่วนของการประกอบคำสั่งเพื่อใช้งานตามที่เราต้องการ

ตัวอย่าง ชุดคำสั่งสำหรับการยกกำลัง  $n1$  ด้วยเลขชี้กำลัง  $n2$  แล้วเก็บผลลัพธ์ไว้ที่ตัวแปร  $x$  เริ่มชุดคำสั่งโดย กำหนดค่าเริ่มต้นของตัวแปร  $x$  เป็น 1 แล้วทำซ้ำคูณค่า  $n1$  จำนวน  $n2$  รอบ ทุกครั้งที่คูณ จะเก็บผลไว้ที่ตัวแปร  $x$



วิธีการเรียกใช้ก็สามารถเรียกโดยใส่ค่าตัวเลข 2 ตัว เป็นเลขฐานและเลขชี้กำลัง



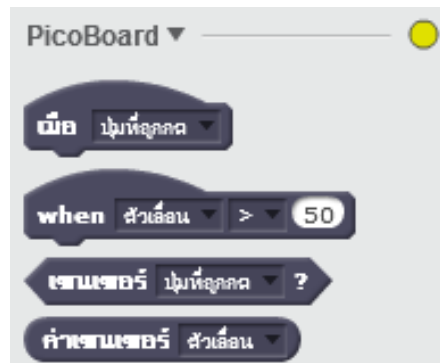
ตัวอย่างเป็น 3 ยกกำลัง 3 ส่วนคำตอบจะอยู่ในตัวแปร x ดังนั้น  
หลังจากที่กำหนดคำสั่งให้ตัวละครพูดค่า x ออกมาเป็นเวลา 2 วินาที จะ  
ได้



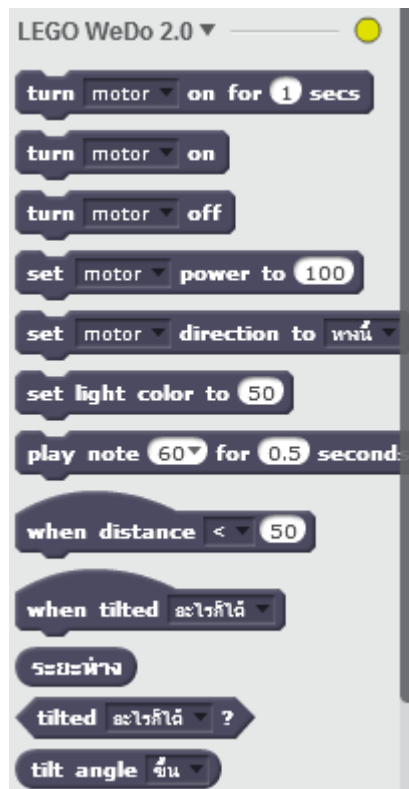
2. เพิ่มส่วนขยาย เป็นการเลือกเครื่องมือติดต่อเสริมเข้าไปให้กับ Scratch โดยแต่ละเครื่องมือจะมีชุดคำสั่งที่แตกต่างกัน



ตัวอย่างชุดคำสั่งของ PicoBoard มีเซนเซอร์และปุ่มกดอยู่ที่อุปกรณ์ Block คำสั่งของอุปกรณ์นี้ก็จะล๊อคกับตัวอุปกรณ์ มีคำสั่งเมื่อเกิดเหตุการณ์ปุ่มถูกกด ได้แก่ A B C และ D เชื่อมต่อ จะทำตามคำสั่งที่กำหนด หรือทำการตรวจสอบ เซนเซอร์แสงหรือเสียงว่ามากกว่าหรือน้อยกว่าค่าที่กำหนด ให้ดำเนินการต่ออย่างไร

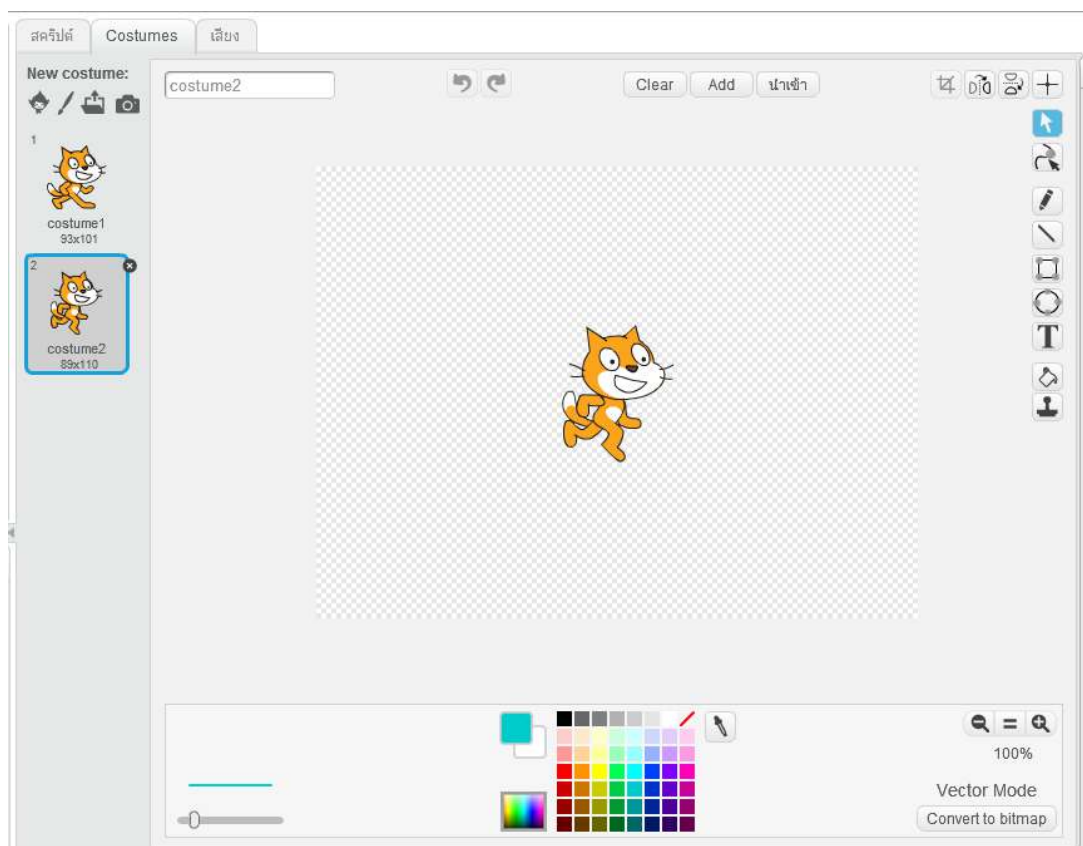


ตัวอย่าง LEGO WeDo 2.0 จะมีมอเตอร์เข้ามาเกี่ยวข้องกับระบบ สามารถสร้างโปรแกรมเพื่อควบคุมการทำงานของมอเตอร์ และตรวจสอบสถานะภาพของมอเตอร์ได้



## 2. Costumes

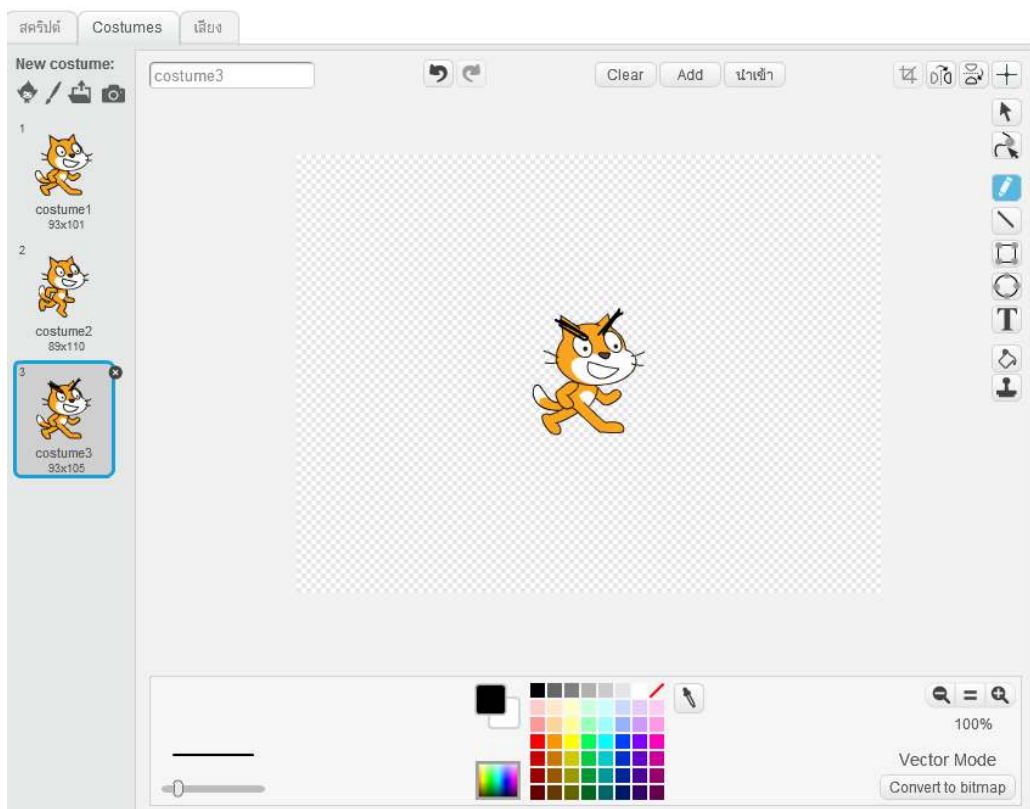
ตัวละครหนึ่งตัวจะมีลักษณะที่แตกต่างกันตามบริบท ตัวอย่างแมวส้มตัวนี้มี 2 Costumes สังเกตได้ว่าแมวสองตัวนี้เมื่อเราเรียกใช้สลับกันจะเป็นเหมือนแมวกำลังเดินอยู่ Costumes ในตัวละครมีประโยชน์ในการทำให้ตัวละครสามารถแสดงหน้าตาท่าทางหรือคำพูดที่ต่างกันตามฉากหรือบริบทที่ต่างกันตามที่เรากำหนด



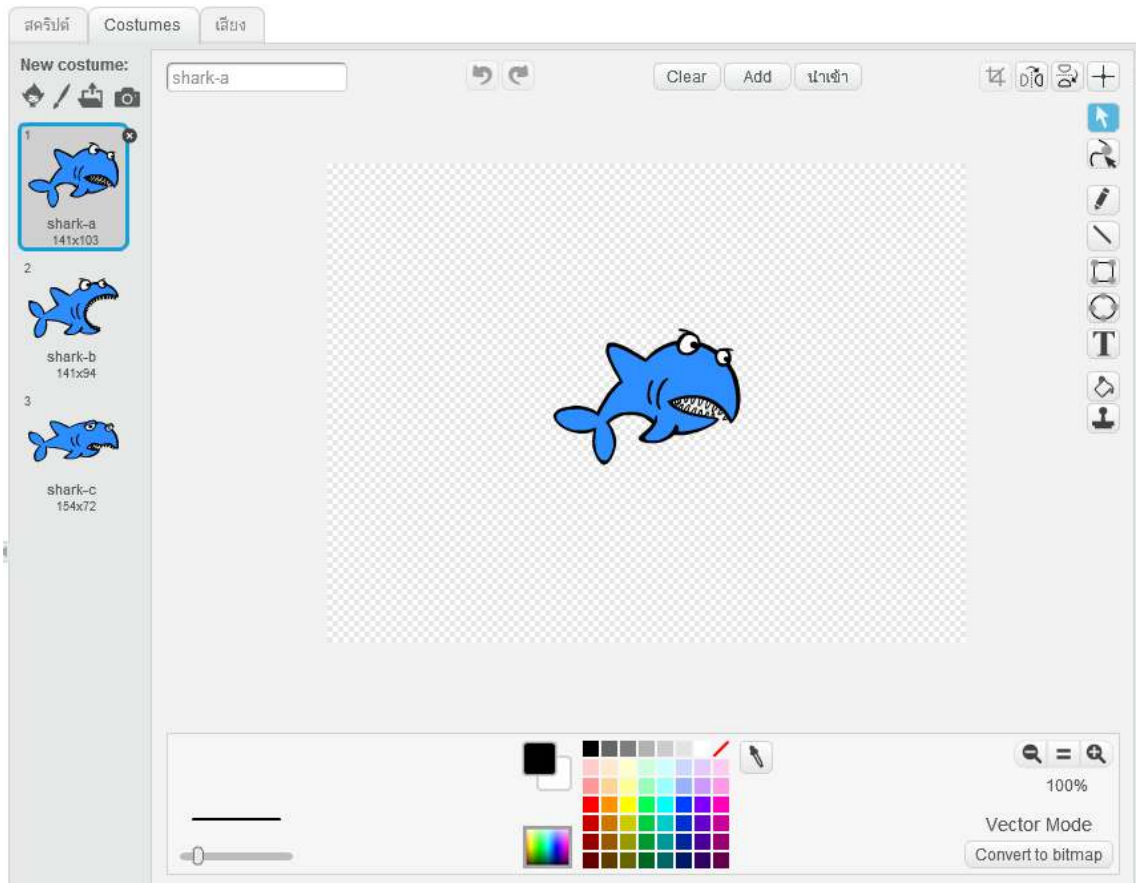
เราสามารถเพิ่มเติมลักษณะอื่นเข้าไปโดยเพิ่มเติมลักษณะจากตัวละครเดิม ให้คลิกขวาที่ Costume ตัวละครที่เราต้องการ แล้วเลือก “ทำซ้ำ”



เราสามารถวาดเส้นเพิ่มเติมตามที่เราต้องการด้วยเครื่องมือสำเร็จรูปของ Scratch ในการทำให้ตัวละครมี Costume ที่เราต้องการ ตัวอย่างการสร้าง Costume ของแมวส้มให้มีลักษณะมุ่งมั่น หรือโกรธมากขึ้น อาจนำไปใช้เมื่อต้องต่อสู้



กรณีเป็นตัวละครตัวอื่นก็จะมีจำนวน Costume ที่โปรแกรมมีให้ไม่เท่ากัน และมีลักษณะอารมณ์ท่าทางที่นำไปใช้ได้แตกต่างกัน ตัวอย่างเจ้าฉลามน้อยตัวนี้ มีถึง 3 Costumes ด้วยท่าทางและอารมณ์ที่ต่างกัน



### 3. เสียง

ในเมนูเสียงของแต่ละตัวละครจะมีไม่เหมือนกันขึ้นอยู่กับลักษณะเฉพาะของตัวละครนั้น



## ตัวอย่าง เสียงของกระดิ่ง Bells



มีไฟล์เสียงที่ชื่อว่า xylo1 ความยาวเกือบ 11 วินาที

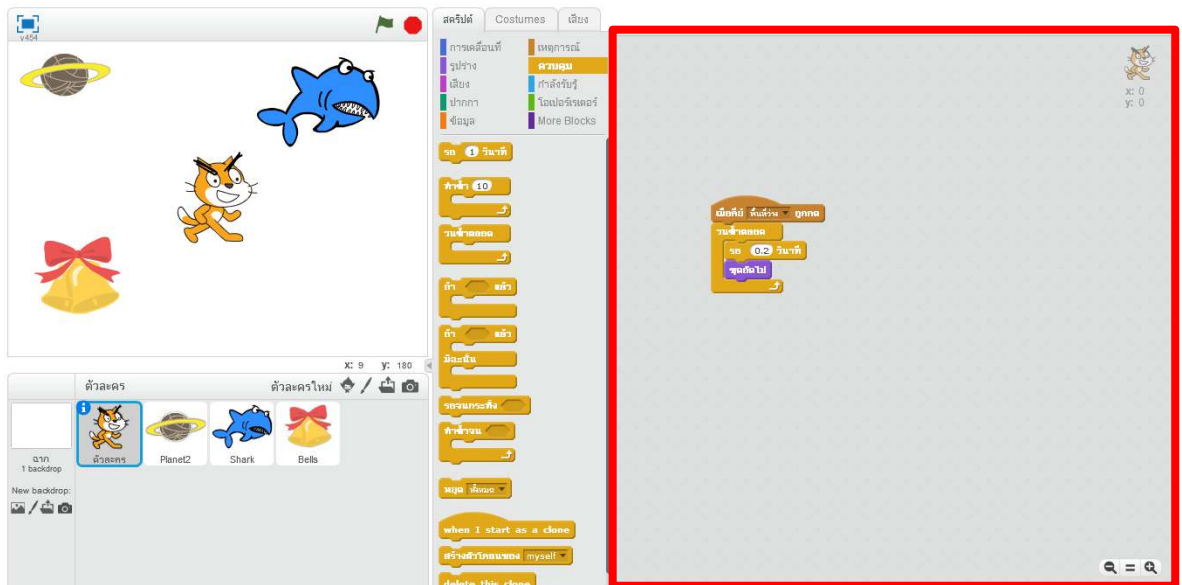


ซึ่งเราจะใช้คู่กับสคริปต์หมวดที่ 3 หมวดเสียง



เราสามารถเพิ่มเสียงที่ต้องการเข้ามาใหม่ได้ 3 รูปแบบ ได้แก่ เพิ่มเสียงจากไฟล์ที่มากับโปรแกรม บันทึกเสียงใหม่เอง และนำมาจากไฟล์เสียงที่อยู่ในเครื่องเรา

## 5. Scripts Area

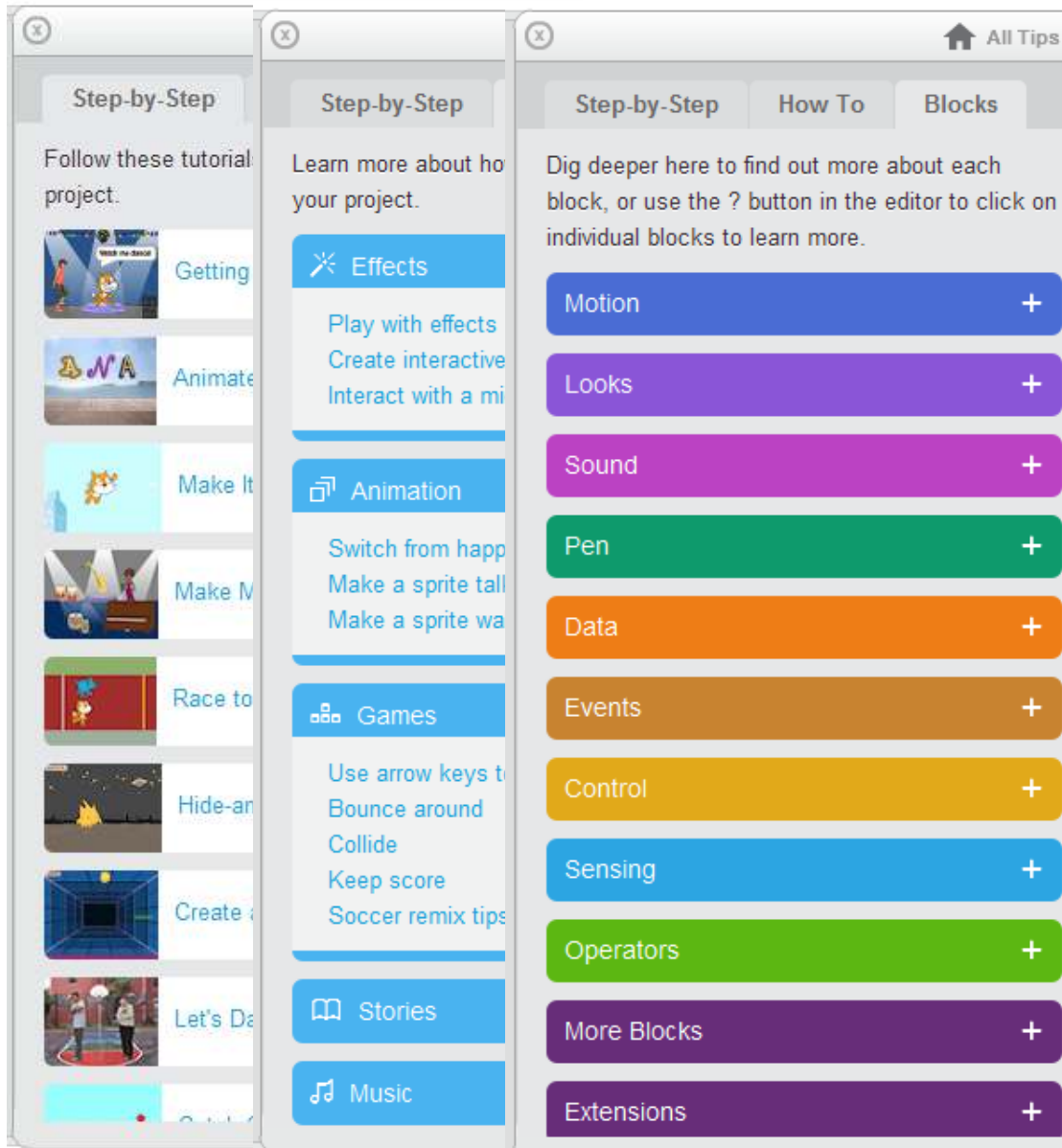


ส่วนสำหรับใส่สคริปต์ เราสามารถลาก Block จากส่วน Block Palette มาประกอบกันใส่ในส่วน Scripts Area นี้ เพื่อให้ตัวละครและฉากทำงานตามที่เราต้องการ

## 6. Tip box

เป็นส่วนช่วยให้ผู้ใช้ใหม่สามารถเริ่มเรียนรู้ความสามารถของโปรแกรม Scratch ได้อย่างง่ายเป็นขั้นตอน แบ่งตามหมวดหมู่ไว้เรียกใช้งานได้ง่าย และมีตัวอย่างเกม เรื่องราว การทำภาพเคลื่อนไหว และวิดีโอเพลง ไว้ให้ผู้ใช้สามารถทำตามเพื่อเกิดความคิดสร้างสรรค์ต่อยอดได้ ซึ่งถึงแม้ว่าจะเป็นผู้ใช้งานไประยะหนึ่งแล้วแต่หลงลืมบางการใช้งานโปรแกรมก็สามารถเลือกดูส่วนที่ต้องการได้สะดวกจากตัวเลือกที่มีให้เหล่านี้

## ภาพหน้าตา Tip Box ทั้ง 3 ส่วน



## การกำหนดคำสั่งให้กับเหตุการณ์

ในชีวิตประจำวัน เรามีการตั้งโปรแกรมตัวเองในการตอบโต้กับสิ่งรอบข้างที่แตกต่างกัน โดยไม่รู้ตัว เช่น การที่เราโดนแดด เรานำมือหรือสิ่งของขึ้นมาบังทิศทางที่แดดส่อง เราเจอคนพูดเสียงดังใส่ เรารู้สึกตกใจ และพยายามเดินออกจากบริเวณนั้น หรือนักกีฬาวิ่งแข่ง 400 เมตร ที่อยู่ในท่าเตรียมพร้อม พอได้ยินเสียงสัญญาณเริ่มวิ่ง นักกีฬาจึงเริ่มออกตัววิ่ง เป็นต้น

ในการสร้างโปรแกรม เราจะต้องเจอกับเหตุการณ์หลายอย่างที่แตกต่างกันจากผู้ใช้ โดยเหตุการณ์เหล่านั้นผู้สร้างโปรแกรมจะเป็นคนกำหนดว่า เมื่อเกิดเหตุการณ์แบบไหนให้ทำอะไร

ตัวอย่าง การสร้างเหตุการณ์เมื่อต้องการควบคุมตัวละครให้ไปตามทิศทาง ซ้าย ขวา บน ล่าง โดยการกดปุ่มลูกศรทิศทาง



โดยในโปรแกรม Scratch มีเหตุการณ์ให้เราเลือกใช้ได้หลายแบบ ได้แก่

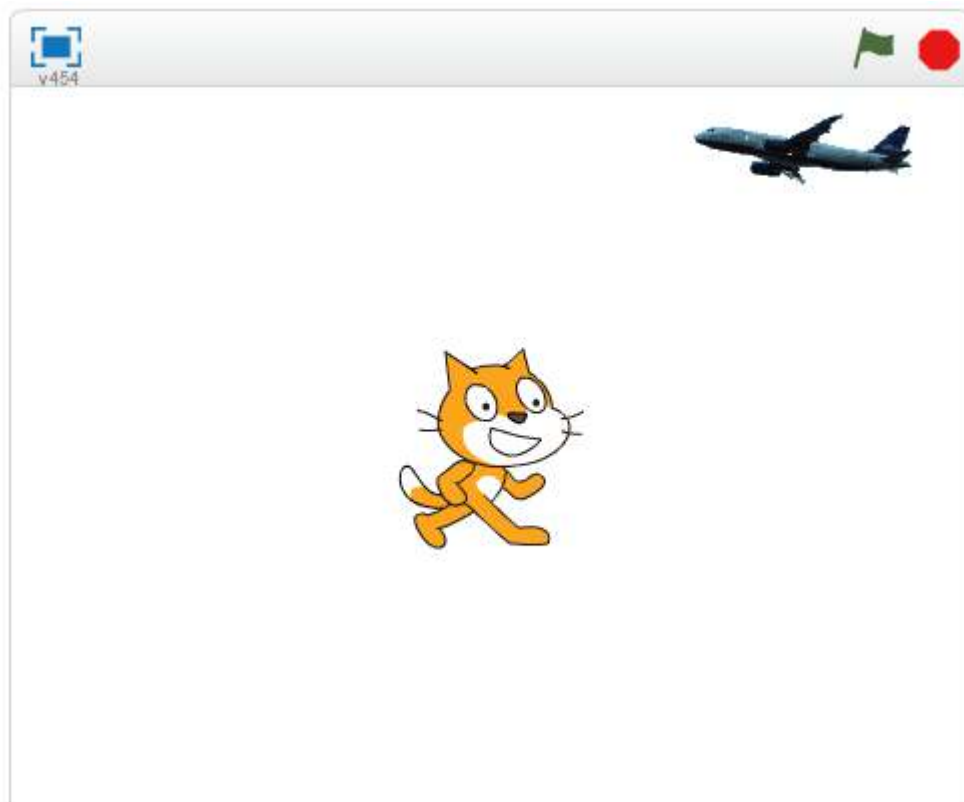
1. เมื่อธงเขียวถูกคลิก
2. เมื่อปุ่มคีย์ที่เรากำหนดถูกกด
3. เมื่อตัวละครที่เราใส่สคริปต์ไว้ถูกกด
4. เมื่อฉากหลังเราถูกเปลี่ยนเป็นฉากหลังที่กำหนด
5. เมื่อความดัง ระยะเวลา หรือการเคลื่อนไหวของภาพในวิดีโอที่กำลังจับอยู่มีค่ามากกว่าที่กำหนด
6. เมื่อตัวละครที่เราเลือกได้รับข้อความที่กำหนด

### 1. เมื่อธงเขียวถูกคลิก

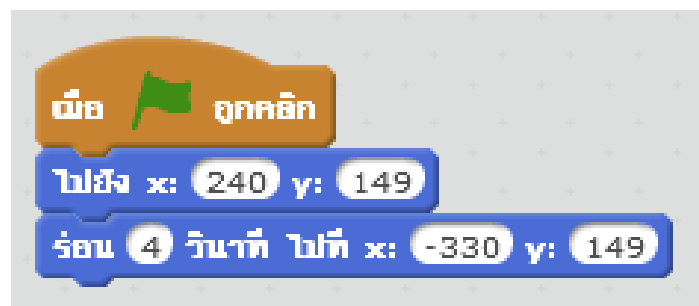


โปรแกรมจะทำงานเมื่อธงเขียวบริเวณด้านบนขวาของฉากถูกคลิก

ตัวอย่างการใช้งาน เมื่อคลิกที่ธงเขียว ให้เครื่องบินบินผ่าน



เราจะใส่สคริปต์นี้ไว้ที่เครื่องบิน



## 2. เมื่อบุคคีย์ที่เรากำหนดถูกกด



โดยเราสามารถเลือกบุคคีย์ได้หลายบุคคีย์ ได้แก่ ลูกศรทั้งสี่ทิศทาง พื้นที่ว่าง (spacebar) ตัวอักษรภาษาอังกฤษทั้ง 26 ตัว (A - Z) ตัวเลข 0-9 และการคลิกที่บุคคีย์ใดก็ได้

ตัวอย่าง เมื่อกด spacebar (พื้นที่ว่าง) ให้เล่นเสียงเครื่องดนตรี และแสดงข้อความว่า “โตนเคาะ”



โดยการใส่สคริปต์ตามนี้



อีกหนึ่งตัวอย่าง เป็นตัวอักษร A



เมื่อเรากดที่ปุ่ม 'a' บน keyboard ตัวอักษรจะตัวใหญ่ขึ้น



ด้วยสคริปต์นี้





### 3. เมื่อตัวละครที่เราใส่สคริปต์ไว้ถูกกด

when this sprite clicked

การใช้เมาส์เพื่อคลิกไปที่ส่วนต่าง ๆ ของฉาก โดยเฉพาะเกมที่ต้องใช้เมาส์เป็นสิ่งหลักในการเล่น จะต้องใช้ Block เหตุการณ์นี้อย่างแน่นอน

ตัวอย่าง เมื่อเจ้าแมวส้มตัวนี้ถูกคลิก จะแสดงผลออกมาว่าถูกคลิก



ด้วยสคริปต์หน้าต่างแบบนี้



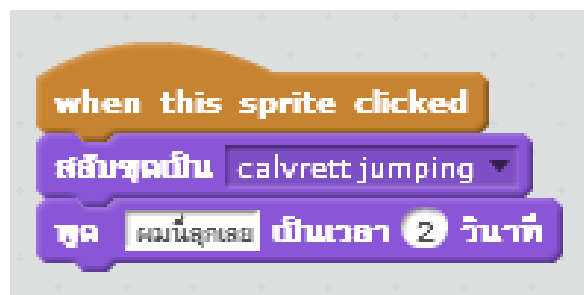
อีกหนึ่งตัวอย่าง เป็นผู้ชายชื่อ Calvrett กำลังนั่งอยู่



เมื่อถูกคลิก ทำการเปลี่ยนท่าและแสดงคำพูด



ด้วยสคริปต์ดังนี้

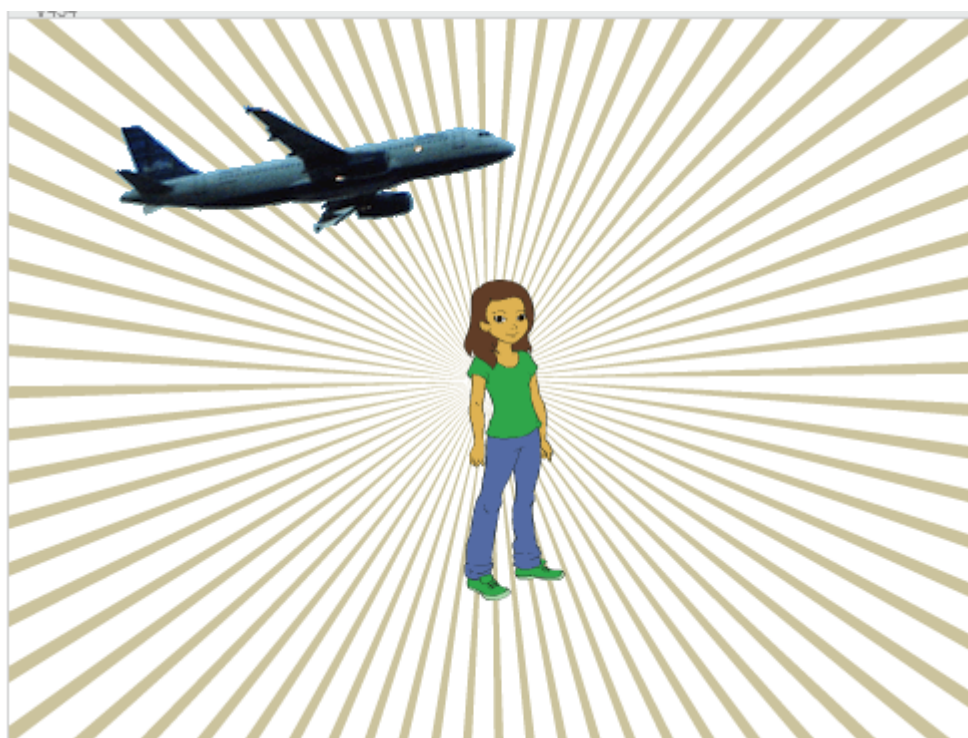


#### 4. เมื่อฉากหลังเราถูกเปลี่ยนเป็นฉากหลังที่กำหนด



บางครั้งในการเล่าเรื่องราว หรือ สร้างเกม การเปลี่ยนฉากเพื่อให้ผู้ใช้รู้ว่าเรากำลังอยู่ในฉากที่จะมีวิธีการใช้หรือเรื่องราวที่แตกต่างกัน เช่น เมื่อฉากเปลี่ยนเป็นทะเล ให้เริ่มเล่นเสียงคลื่น หรือเมื่อฉากอยู่ในสนามรบ ให้มีทหารค่อย ๆ เดินเข้ามาในฉาก เป็นต้น

ตัวอย่าง มีเครื่องบิน กับผู้หญิงชื่อ Abby



เมื่อคลิกที่เครื่องบิน จะทำให้ฉากหลังเปลี่ยนเป็นทะเลทราย (gravel desert)  
แล้วเมื่อฉากหลังเปลี่ยน Abby จะเปลี่ยนชุดเป็น ชุด b



เรามาดูสคริปต์ของเครื่องบิน และสคริปต์ของ Abby กัน



สคริปต์ของเครื่องบิน



สคริปต์ของ Abby

5. เมื่อความดัง ระยะเวลา หรือการเคลื่อนไหวของภาพในวิดีโอที่กำลังจับอยู่มีค่ามากกว่าที่กำหนด



เราสามารถใช้ในการตรวจสอบสามสิ่งนี้เพื่อ เพื่อโปรแกรมจะเปลี่ยนสถานะหรือทำงานบางอย่างให้สอดคล้องกับสิ่งที่เปลี่ยนแปลง

ตัวอย่าง เราจะจับเวลา 10 วินาที เมื่อถึง 10 วินาทีให้แมวสั้มกลับหัว



โดยมีตัวช่วยนับเวลาให้เราเห็นโดยการขีดเครื่องหมายถูกที่ส่วนจับเวลาในหมวดการรับรู้



และสคริปต์ที่ใส่ไปในตัวแมว



จะได้แมวตีสลังกา หลังจากผ่านไป 10 วินาที



#### 6. เมื่อตัวละครที่เราเลือกได้รับข้อความที่กำหนด

การสื่อสารระหว่างตัวละครหรือสิ่งของบางอย่างจะทำให้เมื่อตัวละครตัวหนึ่งทำอะไรบางอย่างแล้วต้องการให้ตัวละครอีกตัวหรือสิ่งของบางอย่างทำงานต่อเนื่องจากการกระทำของตัวละครตัวนั้น เราสามารถใช้การกระจายข้อความช่วยให้ตัวละครตัวอื่นรับรู้ถึงคำสั่งจากตัวละครนี้

ตัวอย่าง มีรถทำความสะอาดถนนคันใหญ่ กับลูกศรชี้ไปทางขวา



เมื่อคลิกที่ลูกศร รถจะเคลื่อนไปทางขวา



โดยเราใส่สคริปต์ที่มีการคุยกันได้ของสองตัวละครนี้

ที่ลูกศร เราจะใส่สคริปต์นี้



และที่รถเราจะใส่สคริปต์นี้

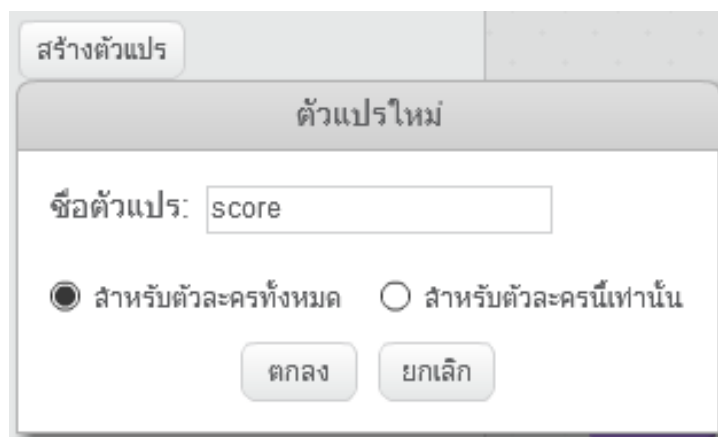


## การกำหนดตัวแปรและการคำนวณ

เราสามารถเก็บข้อมูลตัวเลขและตัวอักษรบรรจุใส่สิ่งที่เรียกว่าตัวแปร และสามารถใช้ในการคำนวณทางคณิตศาสตร์เพื่อใช้หาผลลัพธ์ในบางส่วนของโปรแกรม

### การสร้างตัวแปร

คลิกที่สร้างตัวแปร และใส่ชื่อตัวอย่าง ชื่อ score และเลือกสำหรับตัวละครทั้งหมด



กดเช็คที่กล่องสี่เหลี่ยมหน้า Number เพื่อให้ ตัวเลขที่สุ่มมาอยู่บนหน้าจอสำหรับดูผลที่เกิดขึ้น



มี Block ตัวเลือกในการดำเนินการเกี่ยวกับตัวแปรตามนี้



ใช้ในการใส่ค่าให้กับตัวแปรได้ โดยค่าเริ่มต้นกรณีเรายังไม่ได้ใส่จะเป็น 0





ใช้เพื่อเปลี่ยนค่าของตัวแปรเพิ่มตามจำนวนที่กำหนดด้านหลัง เช่น เราสามารถใช้ในการเพิ่มแต้มในเกมได้ เป็นต้น

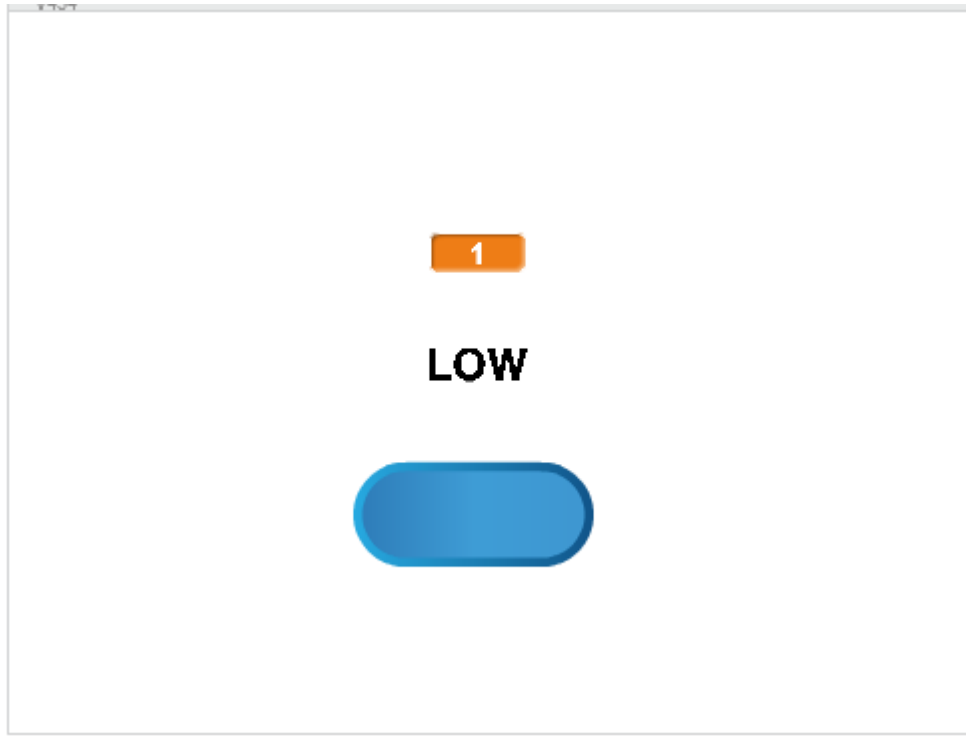


Block นี้ควรจะเป็นคำว่า “แสดงตัวแปร” แต่เข้าใจว่าคนแปลจากภาษาอังกฤษมาผิด ใช้สำหรับแสดงตัวแปรนี้ทางหน้าจอ



ส่วน Block นี้คือการ “ซ่อนตัวแปร” จริง ๆ ใช้ซ่อนตัวแปรออกจากหน้าจอ

ตัวอย่างการใช้ตัวแปรเป็นส่วนหนึ่งในการสร้างเกม เกมที่เราจะสร้างเป็นเกมบอกว่า ตัวเลขที่สุ่มมาได้มีค่าน้อยหรือมาก โดย แสดงคำว่า High หาก เลขอยู่ในช่วง 4 ถึง 6 และแสดง คำ Low หากเลขที่ได้อยู่ในช่วง 1 ถึง 3 เราจะเริ่มโปรแกรมสุ่มโดยคลิกที่ปุ่มสีฟ้า



การนำตัวแปรไปใส่ไว้ในสคริปต์จุดต่าง ๆ ของโปรแกรม เริ่มตั้งแต่ส่วนปุ่มสีฟ้า จะเป็น การกำหนดค่าจากการสุ่มใส่ไว้ในตัวแปร Number



ส่วนป้ายบอก High และ Low จะนำค่าจากตัวแปร Number มาตรวจสอบ



## การคำนวณ

ในการสร้างโปรแกรมหรือเกม เราจะมีการคำนวณในบางส่วน เราใช้คอมพิวเตอร์ในการคำนวณให้เราได้เลย การคำนวณทางคณิตศาสตร์และตรรกศาสตร์ในโปรแกรม Scratch มีให้เลือกหลายตัว ได้แก่

### 1. การคำนวณพื้นฐาน

การคำนวณพื้นฐานอย่างบวก ลบ คูณ หรือ หาส



โดยในช่องวงกลมสีเทาเราสามารถพิมพ์ตัวเลขใส่เข้าไป หรือ Block ที่เป็นตัวเลข ตัวแปร หรือการคำนวณใส่เข้าไปได้เช่นกัน

ตัวอย่าง การนำ Block ตัวแปร การคำนวณ และการพิมพ์ตัวเลข ใส่เข้าไปใน Block การคำนวณการบวก



## 2. การสุ่มตัวเลข

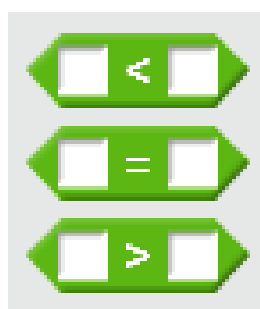
เราจะใช้การสุ่มตัวเลขส่วนมากในการสร้างเกม



จาก Block ตัวอย่างด้านบน เราสามารถเลือกสุ่มตัวเลขจาก 1 ถึง 10 จริง ๆ เราสามารถแก้ไขตัวเลขตรงนี้ได้ เช่น การสุ่มเลข 1 ถึง 100 เป็นต้น

## 3. การเปรียบเทียบทางคณิตศาสตร์

เราใช้การเปรียบเทียบทางคณิตศาสตร์กับส่วนที่เป็นเงื่อนไข เพื่อตัดสินใจว่า โปรแกรมจะทำงานส่วนใดต่อ



ตัวอย่างการนำไปใช้กับส่วนการตัดสินใจของโปรแกรม หรือส่วนที่มีช่องให้ใส่  
กล่อง 6 เหลี่ยม



#### 4. การดำเนินการทางตรรกศาสตร์



ส่วนของการตัดสินใจบางครั้งเราจะใช้การเปรียบเทียบหลายชุดร่วมกันในการตัดสินใจทำหรือไม่ทำชุดคำสั่ง การดำเนินการทางตรรกศาสตร์จะช่วยให้การตัดสินใจนั้นครบถ้วนมากขึ้น ตัวอย่างการตัดสินใจในชีวิตประจำวัน เช่น

ถ้าฝนตก และ แดดออก จะเกิดรุ้งกินน้ำ กรณีนี้เงื่อนไขของเหตุการณ์คือ “ฝนตก และ แดดออก” ซึ่งขึ้นด้วย “และ” ถ้าฝนไม่ตก หรือ แดดไม่ออก อย่างไม่อย่างหนึ่ง ก็จะไม่เกิดรุ้งกินน้ำ

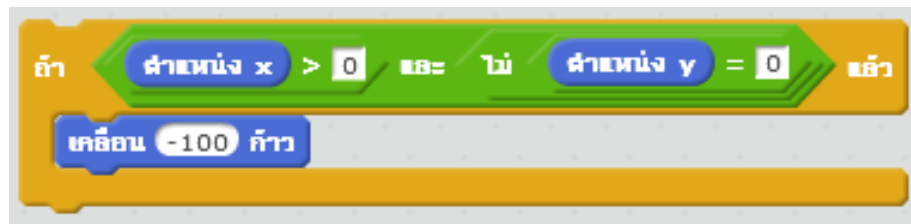
ถ้าฝนตกจนน้ำขัง หรือ มีอุบัติเหตุรถชน รถจะติด กรณีนี้เงื่อนไขของเหตุการณ์คือ “ฝนตกจนน้ำขัง หรือ มีอุบัติเหตุรถชน” แต่ขึ้นด้วย “หรือ” ดังนั้น ไม่ว่าฝนตกจนน้ำขัง หรือมีอุบัติเหตุรถชน อย่างไม่อย่างหนึ่ง ก็จะทำให้รถติด

ส่วน “ไม่” เป็นการทำให้ค่าความจริงเป็นเท็จ และค่าความจริงเป็นจริง ตรงข้าม  
กับ

ตัวอย่างโปรแกรม การใช้ตรรกศาสตร์เข้ามาเกี่ยวข้อง



ตามสคริปต์นี้ มีเงื่อนไขเกี่ยวกับตำแหน่งของตัวละคร ต้องเป็นจริงทั้งตำแหน่ง x และ ตำแหน่ง y ถ้าอยู่ในเงื่อนไขแมงจะเคลื่อนที่ไปด้านหลัง 100 ก้าว



จะได้ผลลัพธ์ดังนี้



## 5. การดำเนินการกับข้อความ

การสร้างเรื่องราว เกม หรือการทำภาพเคลื่อนไหว ที่มีการพูดคุย หรือสื่อสาร ให้ผู้ใช้งานเข้าใจ จำเป็นต้องอาศัยข้อความที่หลากหลายและยืดหยุ่น Scratch จึงมี Block คำสั่งพื้นฐานให้กับผู้สร้างโปรแกรมเลือกใช้งานได้อย่างง่ายตามความต้องการ ได้แก่

1. การเชื่อมข้อความ
2. การเลือกตัวอักษรของคำโดยการใช้ตำแหน่งในการเลือก
3. การบอกค่าความยาวของข้อความ



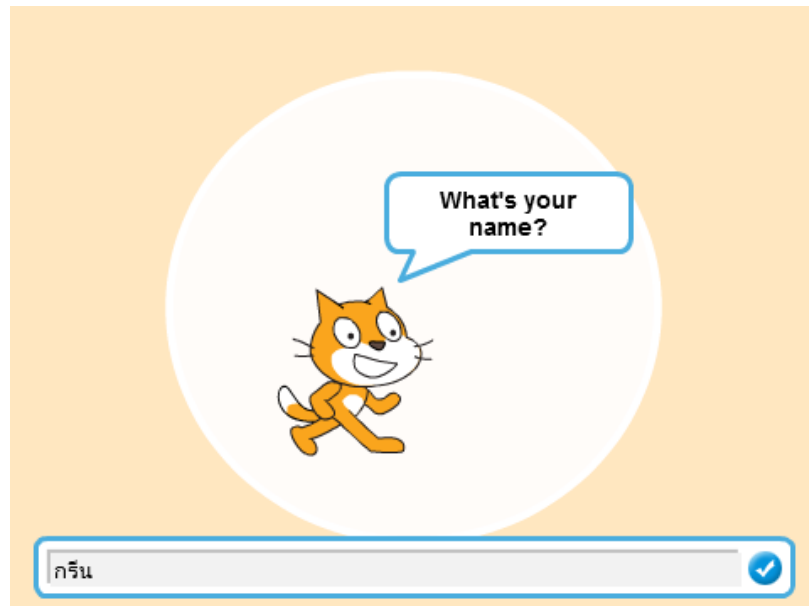
### 1. การเชื่อมข้อความ

เราสามารถเชื่อมข้อความสองข้อความโดยอาศัย Block คำสั่งนี้





ตัวอย่างการใช้ควมคู่กับการรับชื่อจากผู้ใช้มาแสดงคำทักทาย



สคริปต์ที่ใช้



2. การเลือกคูตัวอักษรของข้อความโดยการใช้ตำแหน่งในการเลือก เราสามารถใช้งานโดยใส่สคริปต์ และตัวเลขแสดงตำแหน่ง



จากตัวอย่าง ตัวอักษรตัวที่ 2 ของคำว่า “English” ก็จะได้ตัว ‘น’

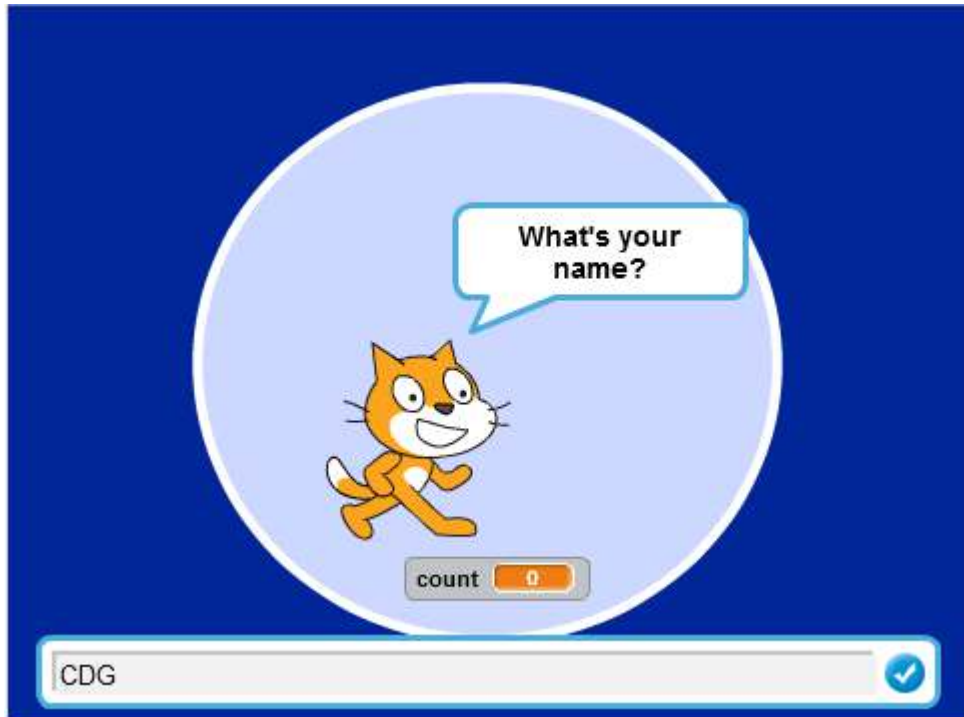
3. การบอกค่าความยาวของข้อความ

ตัวอย่างเมื่อใช้สคริปต์นี้เพื่อบอกค่าความยาวของคำว่า “Thailand” ก็จะได้ผลออกมาเป็น 8 ซึ่งก็คือจำนวนตัวอักษรของข้อความ

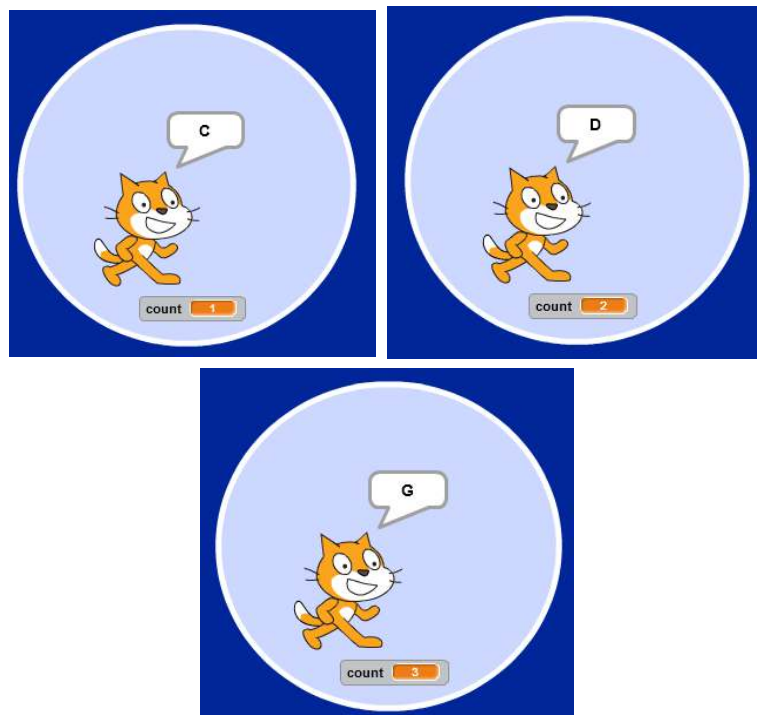


หลายครั้งเราอาจจะใช้การใส่คูตัวอักษรในคำคู่กับความยาว

ตัวอย่าง รับชื่อเข้าไปแล้วให้แสดงตัวอักษรทีละตัวที่อยู่ในชื่อ



เมื่อใส่ชื่อ CDG เข้าไป แมวจะพูดออกมาเป็นตัวอักษรทีละตัวโดย count ก็  
จะเพิ่มขึ้นเรื่อย ๆ ตั้งแต่ 1 ตามจำนวนตัวอักษรของคำ

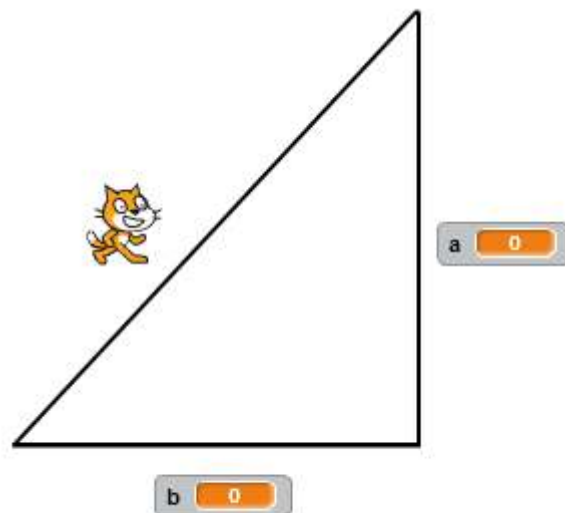


ส่วนหน้าตาของสคริปต์จะยังมีส่วนที่ไม่คุ้นเคยอยู่บ้าง เราจะได้เรียนรู้กัน  
ต่อไปในหัวข้อต่อไป

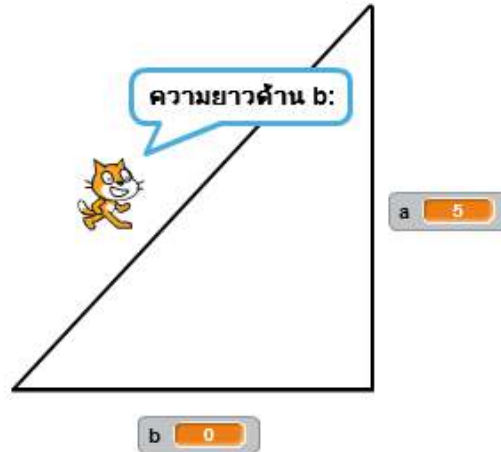
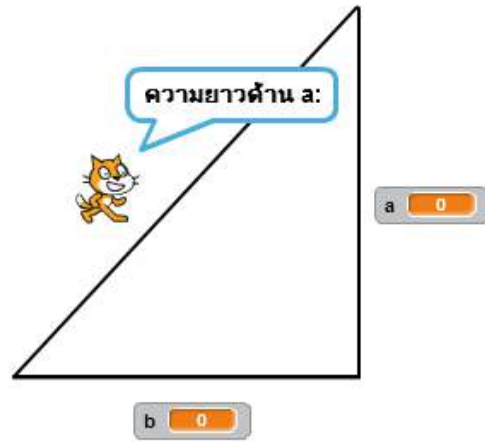


มาดูตัวอย่างการประยุกต์ใช้การดำเนินการทางคณิตศาสตร์มาคำนวณการหาด้าน  
ตรงข้ามมุมฉากของพีทาโกรัส และนำการดำเนินการหาค่าความมาประกอบการใช้งาน

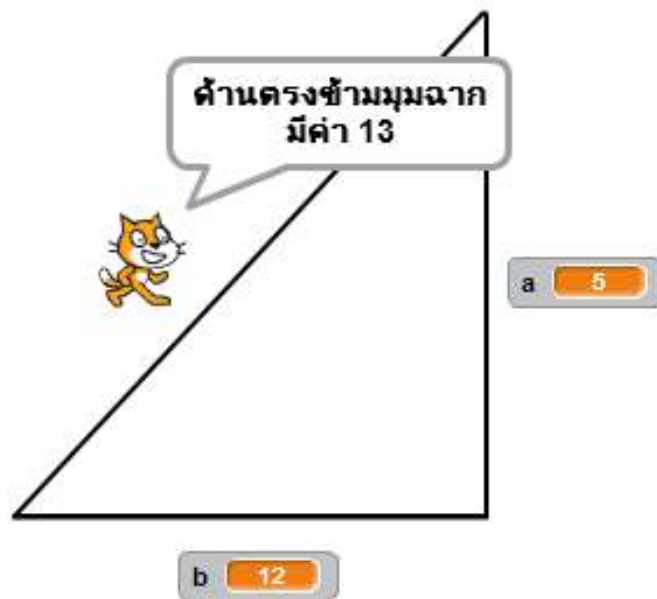
เริ่มจากการวาดสามเหลี่ยมมุมฉาก นำตัวแปรไปวางรอเก็บค่าจากผู้ใช้ และนำเจ้าแมว  
ส้มไปคอยบอกค่าด้านตรงข้ามมุมฉากจากการคำนวณ



ใส่สคริปต์ให้กับตัวแมวเมื่อคลิกธงเขียวให้เริ่มทำงานโดยรับค่าความยาวด้าน a และ ความยาวด้าน b เพื่อนำมาคำนวณ



เสร็จแล้วเจ้าแมวก็จะบอกค่าด้านตรงข้ามมุมฉากจากการคำนวณมาให้



นี่คือสคริปต์ที่ใช้ในโปรแกรม เน้นประยุกต์ใช้ความรู้ที่ผ่านมา

```
เมื่อ ธงเขียว ถูกคลิก
ถาม ความยาวด้าน a: และคอย
set a to คำตอบ
ถาม ความยาวด้าน b: และคอย
set b to คำตอบ
พูด เขียน ด้านตรงข้ามมุมฉากมีค่า  $a^2 + b^2$  เป็นเวลา 5 วินาที
```

## การสร้างเงื่อนไขและการวนซ้ำ

### การสร้างเงื่อนไข

ในชีวิตประจำวันเราสร้างเงื่อนไขให้กับการกระทำบางอย่างแบบไม่รู้ตัว เช่น เราตั้งเงื่อนไขของวันหยุด กับวันธรรมดา เป็นเหตุการณ์ที่ส่งผลต่อการกระทำที่ต่างกัน โดยรูปแบบเงื่อนไขของการสร้างโปรแกรมมี 2 รูปแบบ ได้แก่ “ถ้า.. แล้ว.. มิฉะนั้น..” และ “ถ้า.. แล้ว..” ซึ่งแบบแรกจะเป็นการบอกว่าถ้าเงื่อนไขดังกล่าวเป็นจริง จะมีการกระทำบางอย่าง แต่ถ้าไม่ตรงตามเงื่อนไขนั้น จะกระทำอีกแบบหนึ่ง แต่แบบที่สองจะกระทำบางอย่างเมื่อเงื่อนไขเป็นจริงเท่านั้น ถ้าไม่เป็นจริงจะไม่มีการกระทำอะไรที่เกิดขึ้นต่อ

*ตัวอย่างรูปแบบเงื่อนไข “ถ้า.. แล้ว.. มิฉะนั้น..”*

- ถ้าวันนี้เป็นวันหยุด แล้วฉันจะไปเที่ยว มิฉะนั้นฉันจะไปโรงเรียน
- ถ้าวันนี้เป็นวันพระ แล้วฉันจะออกไปใส่บาตร มิฉะนั้นฉันจะนอนต่อ

*ตัวอย่างรูปแบบเงื่อนไข “ถ้า.. แล้ว..”*

- ถ้าช่วงนี้เป็นฤดูฝน แล้วฉันจะพกร่มออกจากบ้าน
- ถ้าฝนไม่ตกเย็นวันอังคาร แล้วฉันจะซักผ้า

สำหรับโปรแกรมของเราจะมีคำสั่งที่เป็นรูปแบบคล้ายกับเรื่องราวที่ยกให้ด้านบน ดังนี้

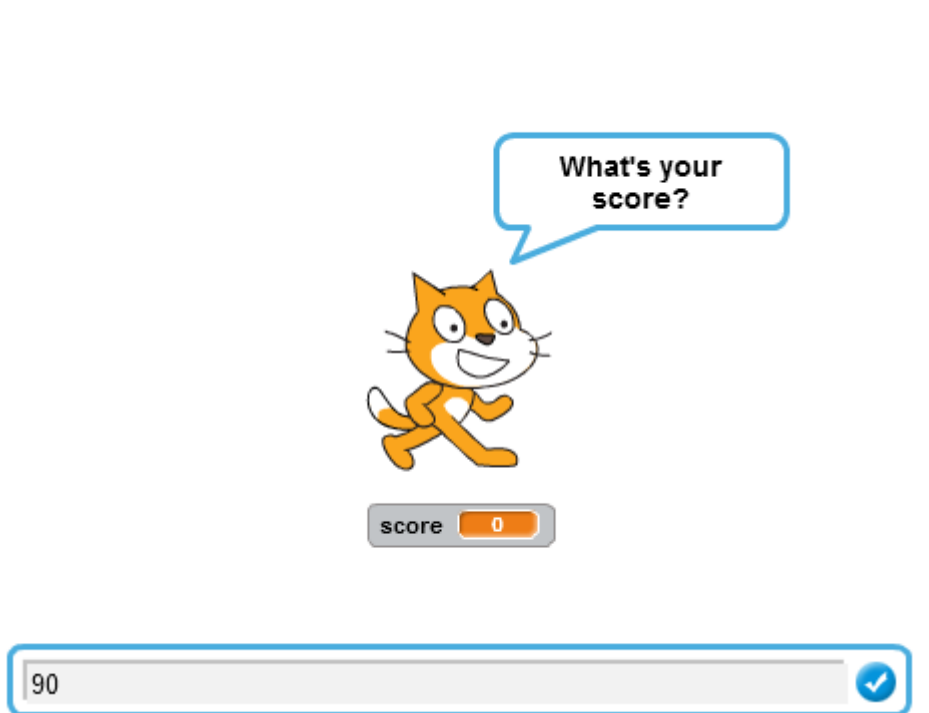
1. ถ้า.. แล้ว..



## ตัวอย่างการใช้งาน



เมื่อรันสคริปต์นี้ โปรแกรมจะให้เราใส่ค่าคะแนนลงไป





จากนั้นเจ้าแมวจะตอบกลับมาว่าคุณได้ A ถ้าคะแนนคุณมากกว่า 79



แต่ถ้าคะแนนที่คุณใส่ไปไม่มากกว่า 79 ละ สิ่งที่แมวจะทำคือ



อยู่นิ่ง ๆ ไม่พูดอะไร เพราะเงื่อนไขไม่เป็นจริง



## 2. ถ้า.. แล้ว.. มิฉะนั้น..



เรามาประยุกต์เพิ่มเติมจากตัวอย่างก่อนหน้านี้ เมื่อแมวรับค่าคะแนนไปแล้วแต่คะแนนกลับไม่มากกว่า 79 เราอยากให้แมวทำอย่างอื่น เช่น บอกว่าเราไม่ได้ A นะ เป็นต้น เราจะใช้เงื่อนไขแบบ “ถ้า.. แล้ว.. มิฉะนั้น”

ลองดูตัวอย่างสคริปต์นี้ เราได้เปลี่ยนจากเงื่อนไขแบบ “ถ้า.. แล้ว..” เป็น “ถ้า.. แล้ว.. มิฉะนั้น..”



จะทำให้เมื่อเราใส่คะแนนที่ไม่มากกว่า 79 เข้าไป จะได้ผลลัพธ์บอก  
กลับมว่าเราไม่ได้ A แทนที่จะเ็ยบ ๆ ไป

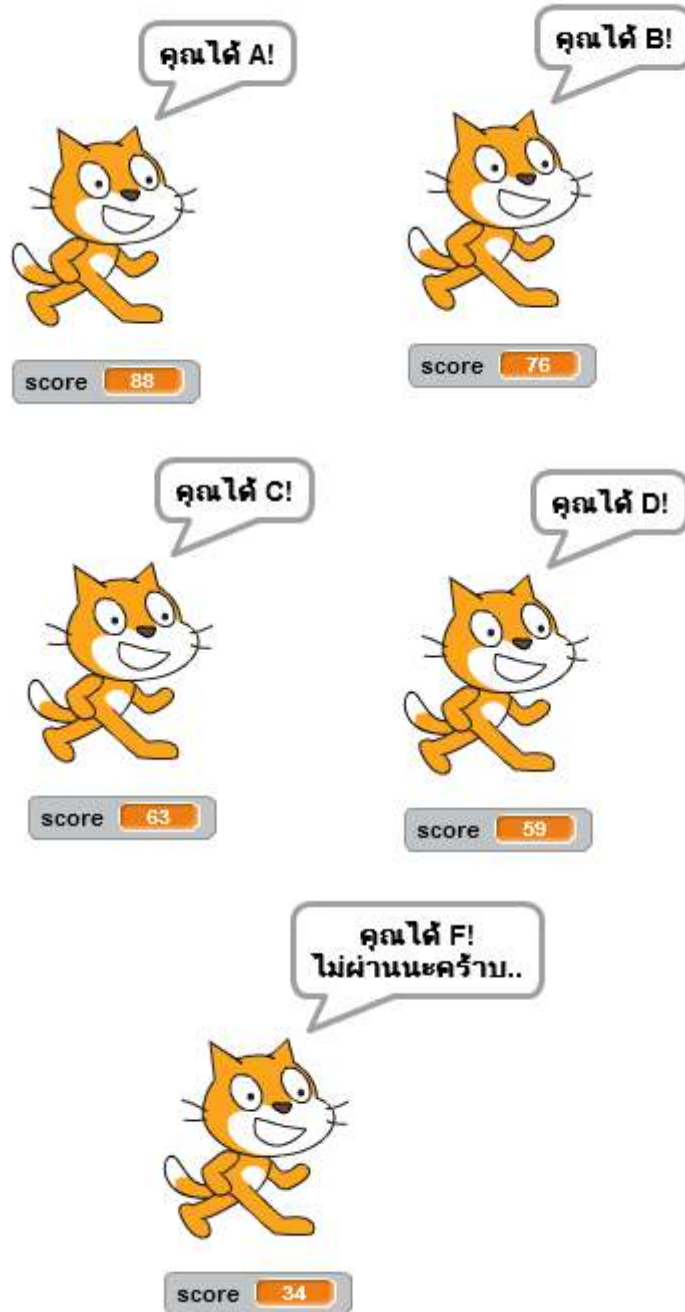


มากไปกว่านั้น เราสามารถเพิ่มกรณีที่ยากขึ้นกว่าเดิมได้โดยใช้เงื่อนไข  
“ถ้า.. แล้ว.. มิฉะนั้น..” ซ้อนกันเข้าไป

ตัวอย่างการสร้างโปรแกรมตัดเกรด ให้สามารถบอกได้ทุกเกรดของการเรียน



ทำให้เมื่อเราใส่คะแนนเท่าไรก็ตามเราก็จะสามารถรู้ได้ว่าเราได้เกรด  
เท่าไร



## การวนซ้ำ

การวนซ้ำเป็นเรื่องที่คอมพิวเตอร์สามารถทำได้ดีกว่ามนุษย์ ช่วยทุ่นแรงได้เยอะ เพราะคอมพิวเตอร์ถนัดในการทำงานซ้ำ ๆ ที่เราตั้งคำสั่งเอาไว้โดยไม่เหนื่อย รวดเร็ว เป็นหลักมิลลิวินาที และที่สำคัญคือมีความแม่นยำมากเพราะคอมพิวเตอร์มีปัจจัยที่มารบกวนน้อย เราเลยเลือกใช้คอมพิวเตอร์ในการทำงานแทนเราในบางครั้งที่เราต้องทำอะไรซ้ำ ๆ เป็นจำนวนมาก ๆ ๆ ๆ เช่น การคำนวณเลขจำนวนมากซ้ำ ๆ การหาค่ารากที่สองของจำนวนจริงที่ไม่ลงตัว การคูณเลขจำนวนหลายหลักหลาย ๆ ครั้ง หรือ การแสดงผลซ้ำ ๆ กันหลายพันครั้ง เป็นต้น

Block สำหรับการวนซ้ำในโปรแกรม Scratch มี 3 รูปแบบ ได้แก่

1. การวนซ้ำตามจำนวนรอบที่กำหนด



2. การวนซ้ำไปเรื่อย ๆ ไม่รู้จบ



3. การวนซ้ำจนกระทั่งเป็นไปตามเงื่อนไขที่กำหนด



## 1. การวนซ้ำตามจำนวนรอบที่กำหนด

ใช้กับโปรแกรมที่สามารถกำหนดจำนวนรอบได้เป็นตัวเลข

ตัวอย่าง โปรแกรมการแสดงดาวตามจำนวนที่กำหนด

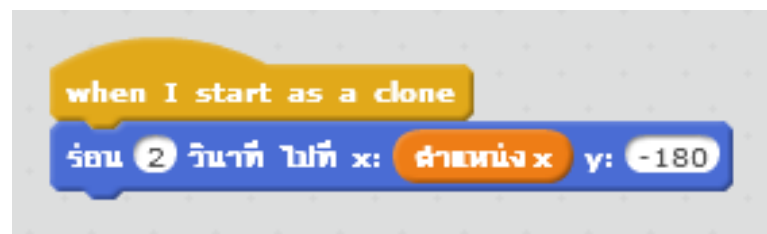
เริ่มจากการนำดาวมาใส่ไว้ด้านบนตรงกลาง



ตามด้วยตัวแปรเก็บจำนวนดาวที่จะกระจายออกมา และตัวแปรเก็บตำแหน่ง x ที่เปลี่ยนไป



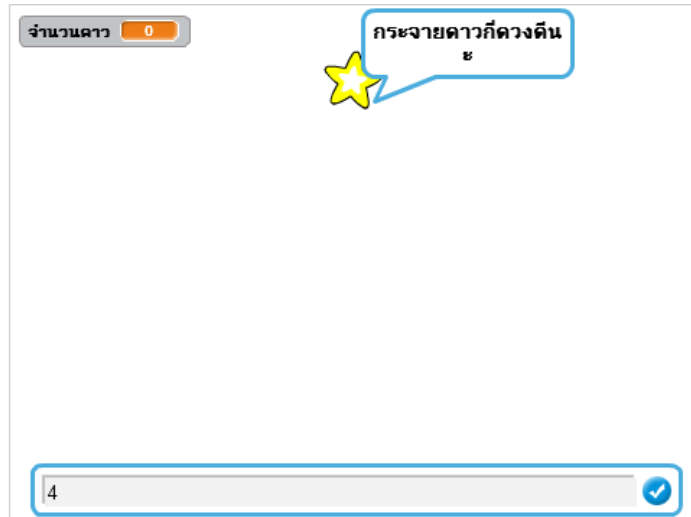
เราจะนำสองสคริปต์นี้ใส่ไว้ที่ดาว เพื่อให้ดาวกระจายออกมาตาม  
จำนวนที่ใส่เข้าไป



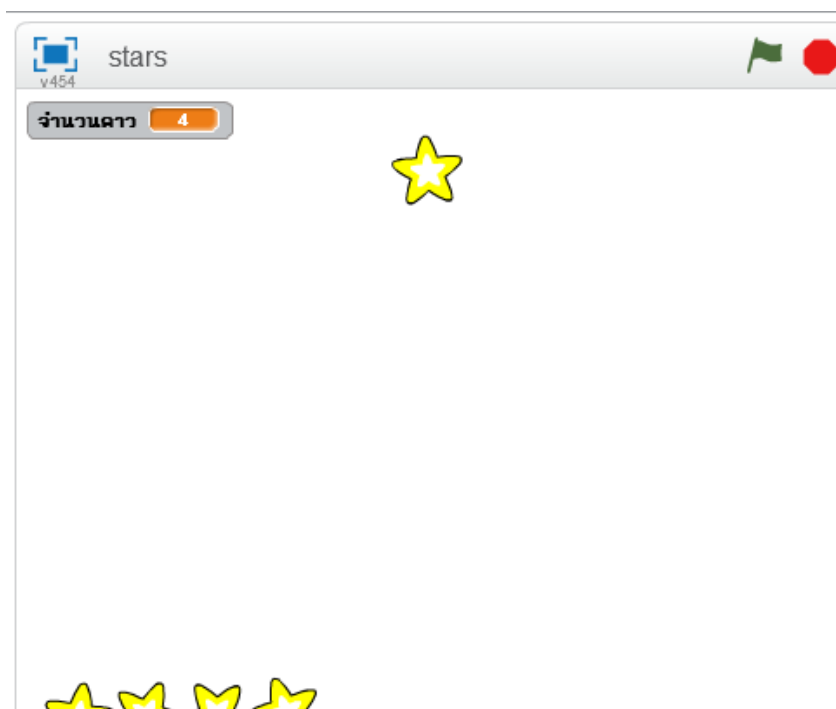


เมื่อเริ่มโปรแกรมด้วยการคลิกที่ธงเขียว โปรแกรมจะให้ใส่จำนวนดาวเข้า

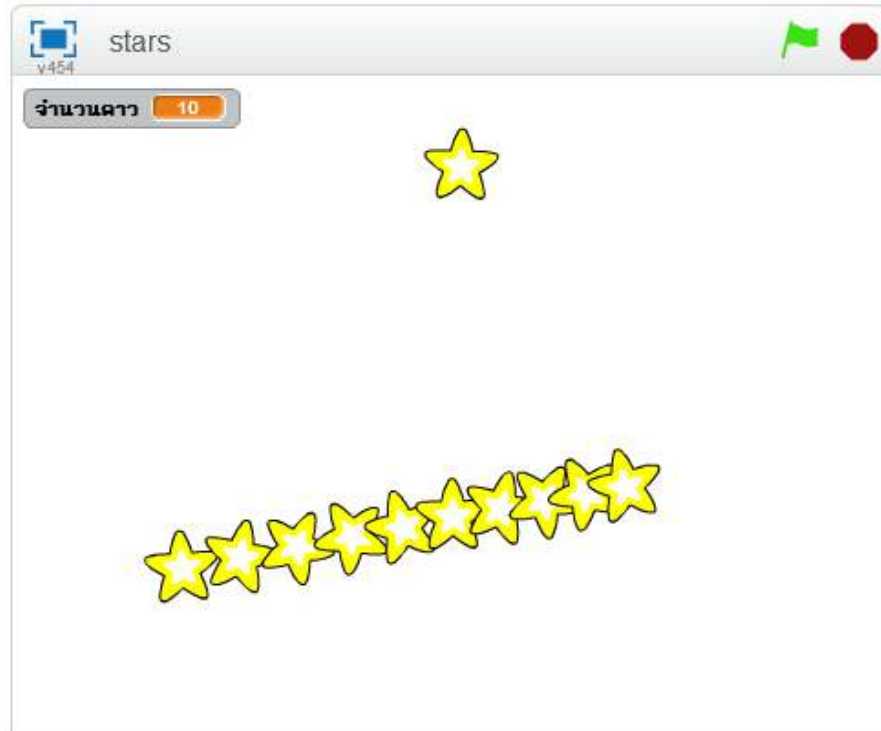
ไป



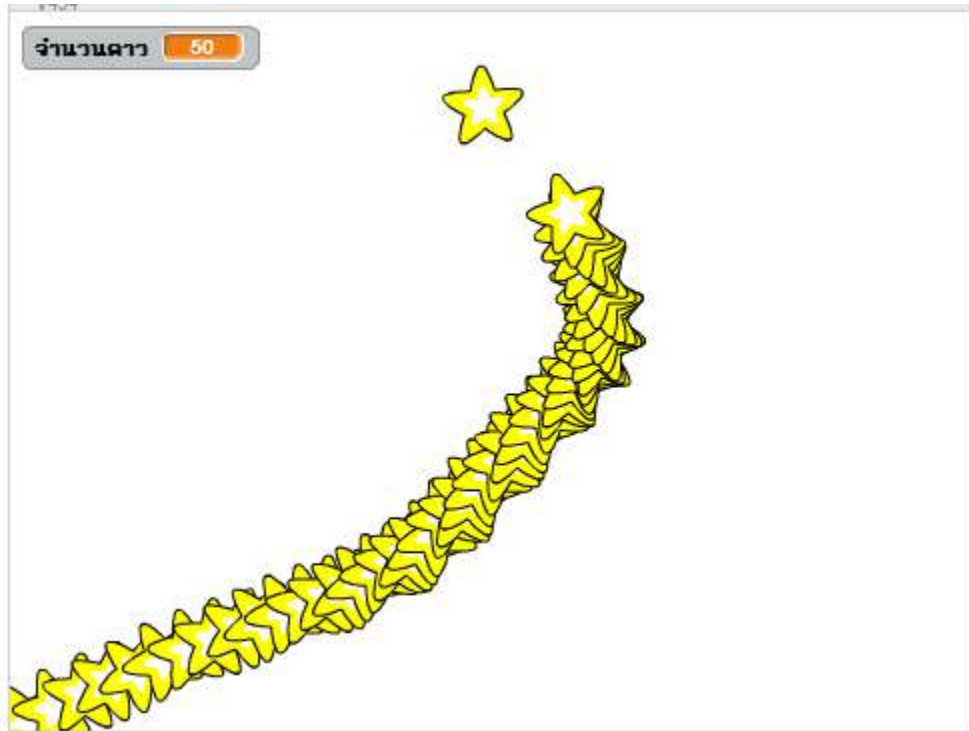
เราจะได้ดาวกระจายตัวออกมาจำนวน 4 ดวง



กรณีที่เราใส่จำนวนดาวไป 10 ดวง ดาวก็จะกระจายออกมา 10 ดวง



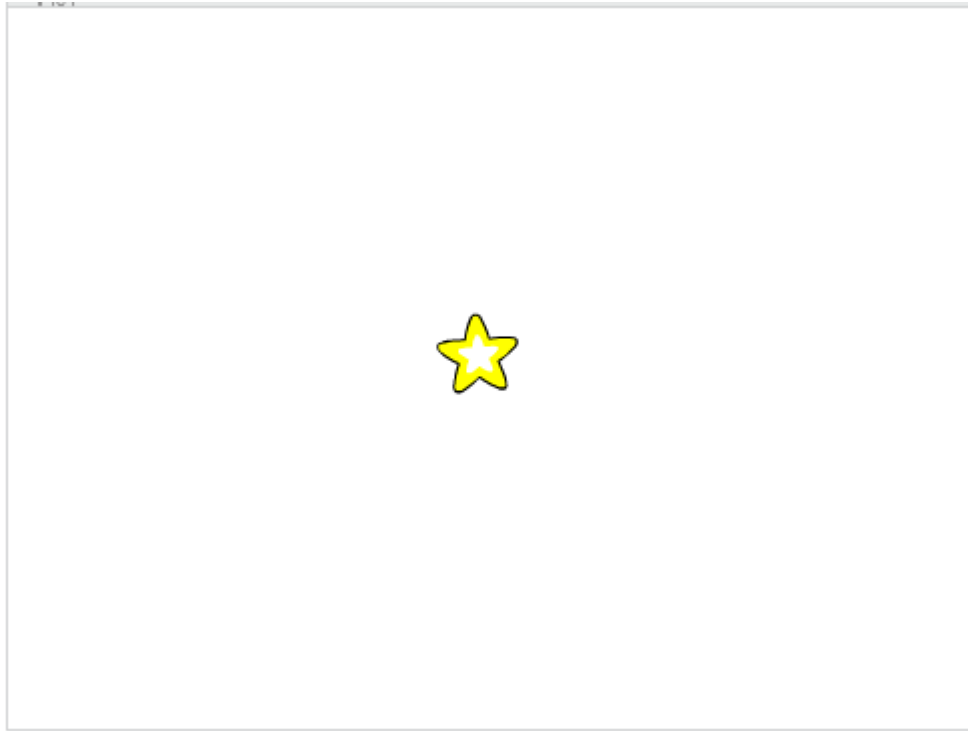
เมื่อเปลี่ยนจำนวนดาวเป็น 50 ดวง และเปลี่ยนตำแหน่ง  $x$  ให้เพิ่มขึ้นทีละ 10 จะได้ดาวกระจายเรียงกันออกมามากมายต่อเนื่องกัน ซึ่งถ้าเราค่อย ๆ กำหนดดาวทีละดวงให้กระจายออกมาเป็นชุดลักษณะนี้ทีละดวงคงเสียเวลามาก ลองจินตนาการว่าเราจะทำให้ดาวกระจายออกมา 1 ล้านดวงดูสิ.. เราคงต้องใช้เวลาเป็นปีในการสร้างมันดูก็ไหม แต่การเขียนโปรแกรมคอมพิวเตอร์ช่วยเราได้มากในการทำอะไรซ้ำ ๆ ๆ ๆ กันในลักษณะนี้



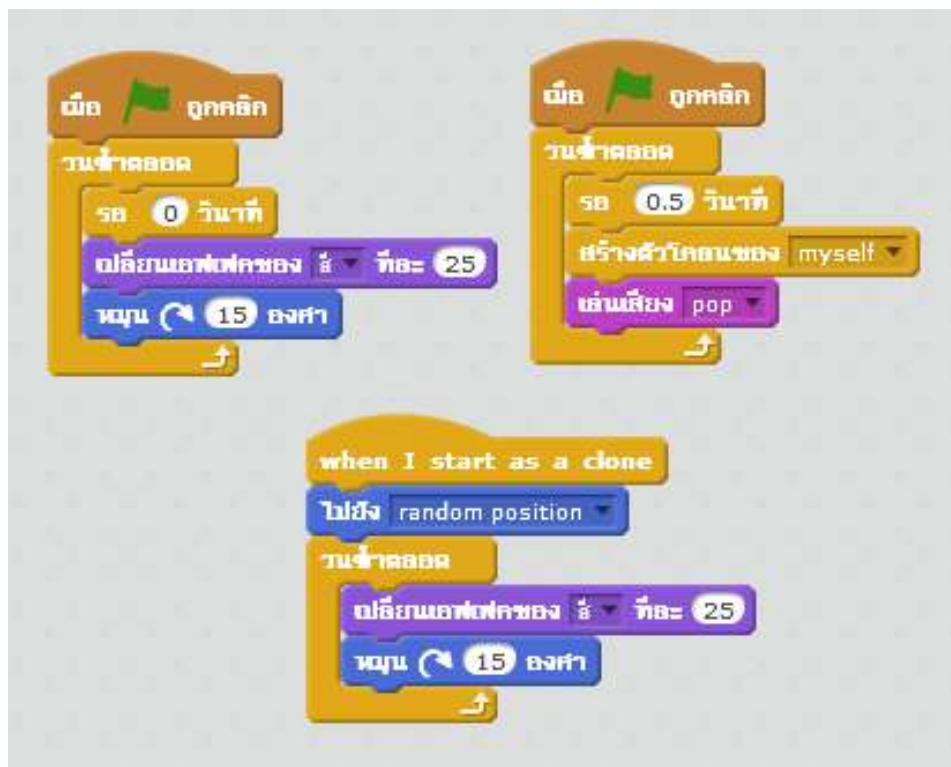
## 2. การวนซ้ำไปเรื่อย ๆ ไม่รู้จบ

เราลองมาทำดาวหมุน ๆ ไปเรื่อย ๆ ไม่รู้จบกัน โดยดาวจะเกิดเพิ่มขึ้นเรื่อย ๆ ไม่รู้จบเช่นกัน

เริ่มจากนำดาวหนึ่งดวงมาไว้ตรงกลางหน้าจอ



เราจะใส่สคริปต์ 3 ก้อนนี้เข้าไปในดาว สังเกตได้ว่าทั้ง 3 ชุดคำสั่งจะมีส่วนของการวนซ้ำตลอด อยู่ในชุดสคริปต์ เพื่อให้บางส่วนของสคริปต์ได้ทำงานตลอดเวลา



ส่วนแรกเป็นส่วนทำให้ดาวตรงกลางเริ่มหมุนโดยเปลี่ยนสีไปเรื่อย ๆ



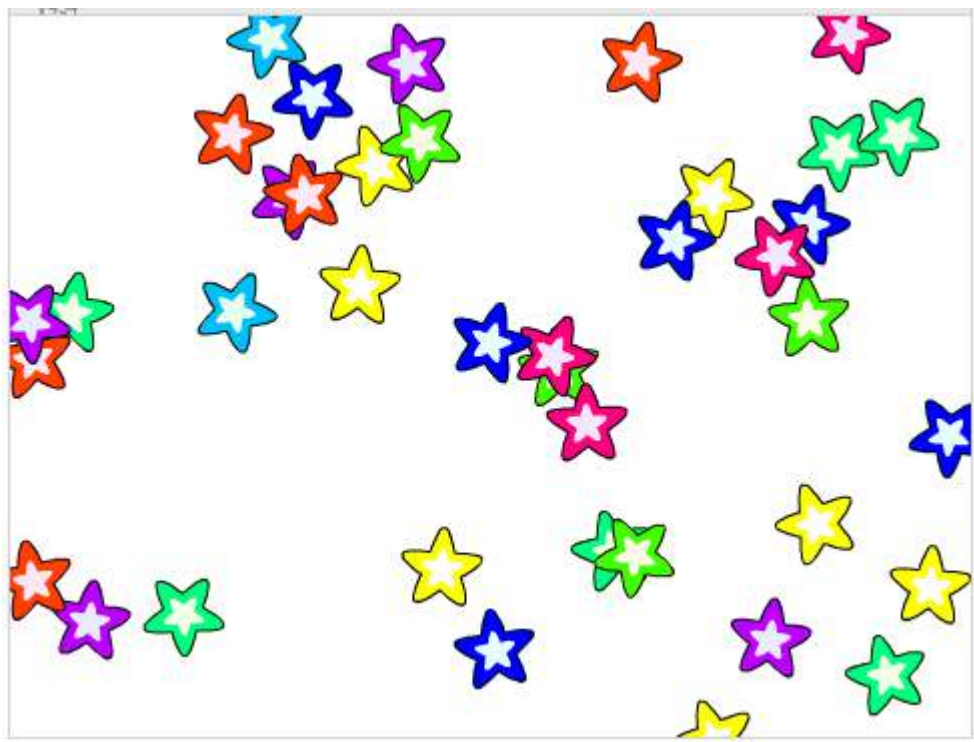
ส่วนที่สองเป็นส่วนที่ใช้สร้างดาวออกมาเรื่อย ๆ ทุก ๆ 0.5 วินาที โดยทุกครั้งที่มีการสร้างดาว ให้เล่นเสียง pop ออกมาด้วย แสดงว่าทุกครั้งที่เราได้ยินเสียง pop แสดงว่ามีดาวเกิดขึ้น



ส่วนที่สาม เป็นส่วนที่บอกให้ดาวที่ถูกสร้างขึ้นมาใหม่เกิดในตำแหน่งที่ถูกสุ่มขึ้นมาและทำการหมุนและเปลี่ยนสีไปเรื่อย ๆ เหมือนดาวดวงแรก



เมื่อเวลาผ่านไปช่วงหนึ่ง เราจะเห็นดวงดาวระยิบระยับกระจายอยู่บนหน้าจอของเรามากมาย



### 3. การวนซ้ำจนกระทั่งเป็นไปตามเงื่อนไขที่กำหนด

ก่อนหน้านี้เราสามารถวนซ้ำได้จนไม่มีที่สิ้นสุด ซึ่งบางครั้งเราอาจจะอยากให้การวนซ้ำนั้นวนไปได้เรื่อย ๆ แต่ก็อยากให้หยุดด้วยในบางเงื่อนไข เราจะสามารถทำได้โดยใช้ Block “ทำซ้ำจน..”

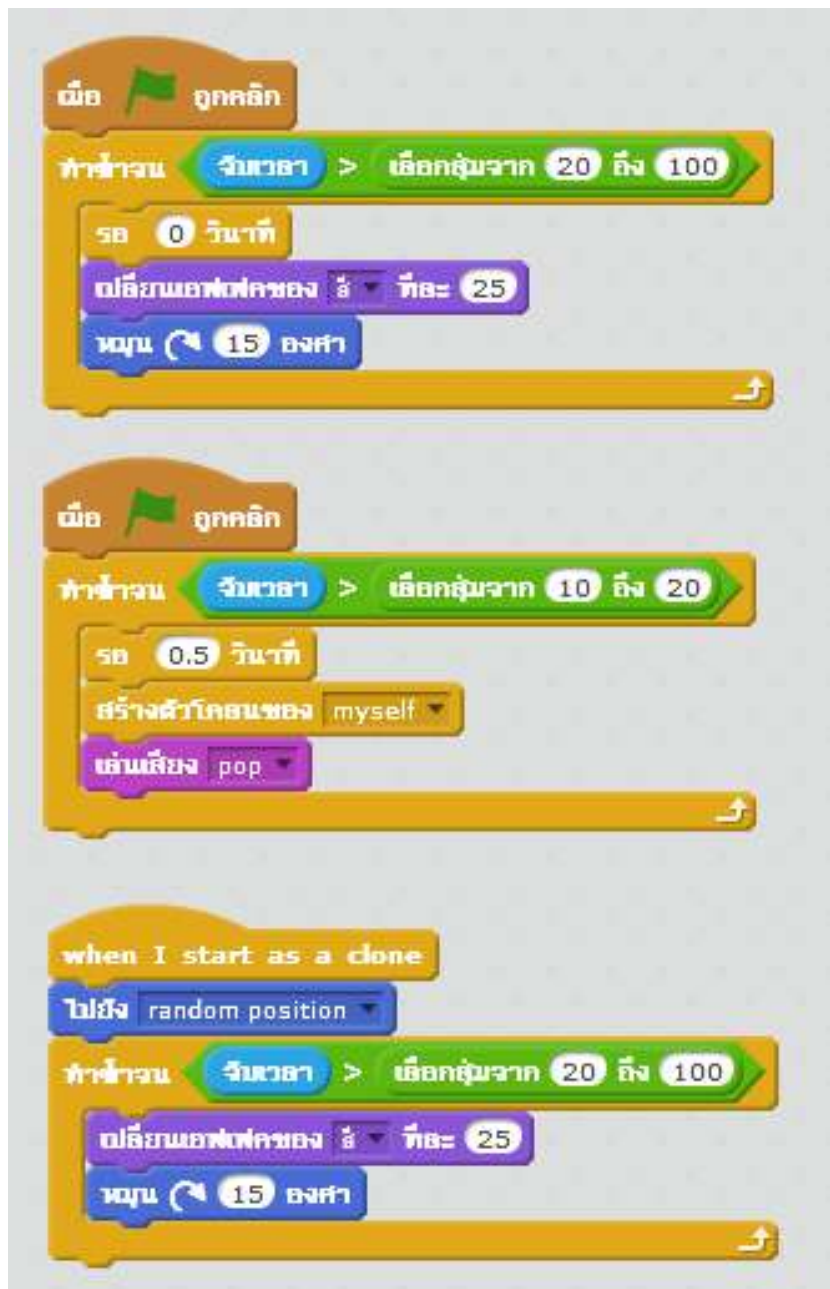


จากการสร้างดาวหมุนในตัวอย่างก่อนหน้านี้โดยการใช้ Block “วนซ้ำตลอด” คราวนี้ลองมาใส่เงื่อนไขการหยุดให้กับการสร้างดาว การหมุน และการเปลี่ยนสีของดาวกันดูดีกว่า

เราจะเปลี่ยน Block จาก “วนซ้ำตลอด” เป็น Block “ทำซ้ำจน..” ก่อน แล้วใส่เงื่อนไขเกี่ยวกับเวลาเข้าไป จะให้หยุดเมื่อเวลาผ่านไปเท่าไร

โดยเงื่อนไขที่เลือกใส่ไปในตัวอย่างนี้จะเป็นเรื่องของเวลา คือ เมื่อเวลาผ่านไปจำนวนหนึ่งตามการสุ่มจะทำให้การหมุนกับการเปลี่ยนสีหยุดลง ซึ่งด้วยการสุ่มเวลา ทำให้เอฟเฟกต์ของดาวหยุดไม่พร้อมกัน ส่วนการสร้างดาวใหม่จะหยุดสร้างก่อนที่ดาวจะหยุดหมุนและเปลี่ยนสี

มาดูสคริปต์กัน..

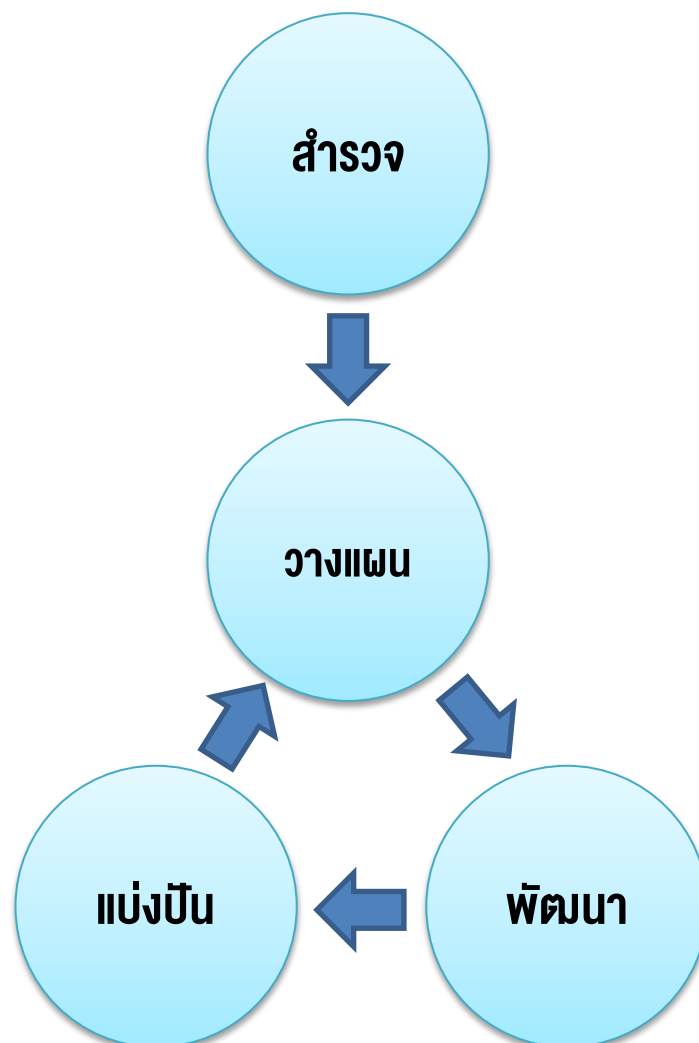




## การออกแบบและสร้างโปรแกรม

เราได้เรียนรู้ทฤษฎีการเขียนโปรแกรมกันมาเต็มที่แล้ว ถึงเวลาสร้างสรรค์โปรเจกของตัวเองกันเต็มที! แต่.. ก่อนที่เราจะเริ่มทำ มาลองศึกษาขั้นตอนกระบวนการสร้างโปรเจค ได้แก่

1. สำรวจ (Explore)
2. วางแผน (Plan)
3. พัฒนา (Develop)
4. แบ่งปัน (Share)



เราลองมาดูรายละเอียดของแต่ละขั้นตอนกันดีกว่าว่าในวงกลมแต่ละอันน่าสนใจ  
อย่างไร

### 1. สำรวจ (Explore)

ก่อนเริ่มขั้นตอนของการพัฒนาโปรแกรม จากแผนภาพข้างต้น จะเห็นได้ว่า  
ขั้นตอนของการสำรวจ จะมาแยกจากขั้นตอนอื่น ๆ ที่วงเป็นวงรอบ เพราะเราควร  
ทราบก่อนว่าเราอยากที่จะทำอะไรจริง ๆ เรามีแรงบันดาลใจในการแก้ไขปัญหาอะไร  
สิ่งเหล่านี้จะทำให้เราพัฒนาโปรแกรมได้อย่างเต็มความสามารถ โดยเรามีขั้นตอน  
การสำรวจความสนใจด้วยคำถามเหล่านี้

- a. โครงการที่เรียนรู้ผ่านมาโครงการไหนที่ชอบที่สุด
- b. มีความสนใจในเรื่องใด (เรื่องอะไรก็ได้ในชีวิต)
- c. สิ่งใดบ้างที่เราอยากจะใช้โปรแกรมที่เราจะพัฒนาขึ้นในการแก้ปัญหา
- d. การพัฒนาโปรแกรมแบบใดที่ตอบโจทย์การแก้ปัญหของเรา เป็นเกม  
เรื่องราว การทำภาพเคลื่อนไหว หรืออื่น ๆ
- e. อะไรเป็นความถนัด หรือความสามารถพิเศษของเรา

คำถามด้านบนใช้สำหรับสำรวจตัวตนของเรา เมื่อเราสำรวจความต้องการและ  
ความสามารถของเราได้เรียบร้อยแล้ว คำแนะนำถัดไปคือการหาทีม แน่นอนว่าเรา  
อาจจะถนัดในบางอย่าง และไม่ถนัดในบางอย่างแต่สิ่งที่เราสนใจ อาจจะได้มีแต่เรา  
เพียงคนเดียว เราสามารถหาเพื่อนร่วมทีมที่มีความถนัดต่างจากเรา เพื่อมาช่วยกัน  
แก้ไขปัญหาที่สนใจร่วมกันได้

หลังจากเราตอบคำถามด้านบน และหาเพื่อนร่วมทีมได้แล้ว ถึงเวลาคิดถึง  
โปรแกรมที่เราจะร่วมกันพัฒนาขึ้นมา

## 2. วางแผน (Plan)

ขั้นตอนการวางแผน เราจะได้คำตอบสำหรับ 4 คำถามนี้

- a. เราจะสร้างโปรแกรมอะไร
- b. มีขั้นตอนอย่างไรบ้างในการสร้างโปรแกรมนี
- c. สิ่งที่เรามีอยู่แล้ว สำหรับโปรแกรมที่เราจะสร้าง เช่น ทีม ตัวอย่างโปรแกรม ไฟล์ภาพ เป็นต้น
- d. สิ่งที่เราต้องการเพิ่มเติม เพื่อใช้ในการสร้างโปรแกรม

### คำถามแรก *เราจะสร้างโปรแกรมอะไร*

วิธีที่เราแนะนำในการหาคำตอบของคำถามนี้ เรียกว่า “การระดมความคิด (Brainstorm)” การระดมความคิดมีขั้นตอน ดังนี้

1. ให้ทุกคนในทีมเสนอไอเดียที่อยากทำให้มากที่สุดเท่าที่เป็นไปได้ในเวลาที่จำกัด เช่น 3 นาที เป็นต้น โดยอาจเขียนใส่กระดาษโพสต์-อิท แล้วพูดออกมาว่าไอเดียในการแก้คืออะไรแล้วแปะลงไปบนกระดาษหรือกระดานที่เตรียมไว้
2. ทีมช่วยกันจัดกลุ่มไอเดียที่มีความใกล้เคียงกันเป็นหมวดหมู่
3. พูดคุยกันถึงความน่าสนใจ และความเป็นไปได้ของแต่ละไอเดีย
4. แต่ละคนเลือกไอเดียที่ชอบ 3 อันดับในใจ และเขียนอันดับของตนเองลงใส่โพสต์-อิทไอเดียที่เลือกเป็นคะแนน 1-3 โดย 3 คะแนน คือไอเดียที่ชอบที่สุด
5. รวมคะแนนแต่ละไอเดียและจัดอันดับไอเดียที่มีคะแนนมากที่สุดไปหาน้อยที่สุด

6. ช่วยกันเลือกไอเดียที่ทุกคนชอบมากที่สุดอาจจะ 3-5 อันดับแรก โดยพูดคุยถึงเหตุผลในการเลือกของแต่ละคน ซึ่งผลสุดท้ายอาจจะได้ข้อสรุป คือ การเลือกอันดับที่ 3 ก็เป็นไปได้



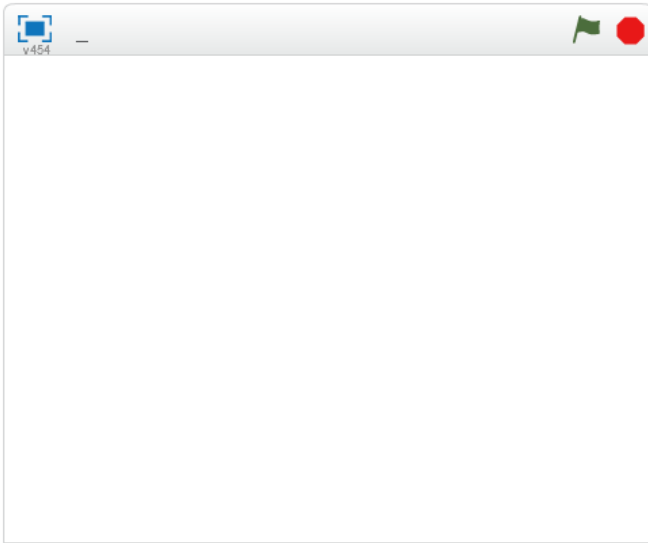
## คำถามที่เหลือ

- มีขั้นตอนอย่างไรบ้างในการสร้างโปรแกรมนี สิ่งที่มีอยู่แล้ว
- สำหรับโปรแกรมนี่เราจะสร้าง เช่น ทีม ตัวอย่างโปรแกรม ไฟล์ภาพ เป็นต้น
- สิ่งที่เราต้องการเพิ่มเติม เพื่อใช้ในการสร้างโปรแกรม

หลังจากได้ไอเดียออกมาแล้วว่าจะทำอะไร ขั้นตอนนี้ เราจะมาช่วยกันในทีมลองเขียนขั้นตอนการสร้างโปรแกรมดูว่า เราจะทำอะไรบ้างเป็นลำดับ เพื่อดูว่าแต่ละขั้นตอนเราต้องการอะไรบ้าง สิ่งใดมีอยู่แล้ว สิ่งใดต้องทำเพิ่ม เรามีแบบฟอร์มในการเขียนให้ตามเอกสารหน้าถัดไปนี้

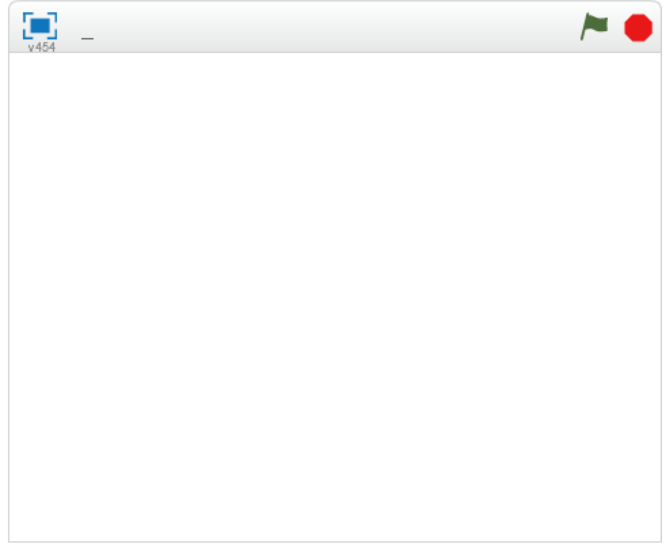
ชื่อโปรแกรม .....

ลองเขียนภาพคร่าว ๆ ของการดำเนินของโปรแกรมเราด้านล่างนี้ และเขียนด้านล่างของหน้าจอบว่าในแต่ละขั้นตอนอาศัยสิ่งใดบ้างที่ใช้ในหน้านั้น



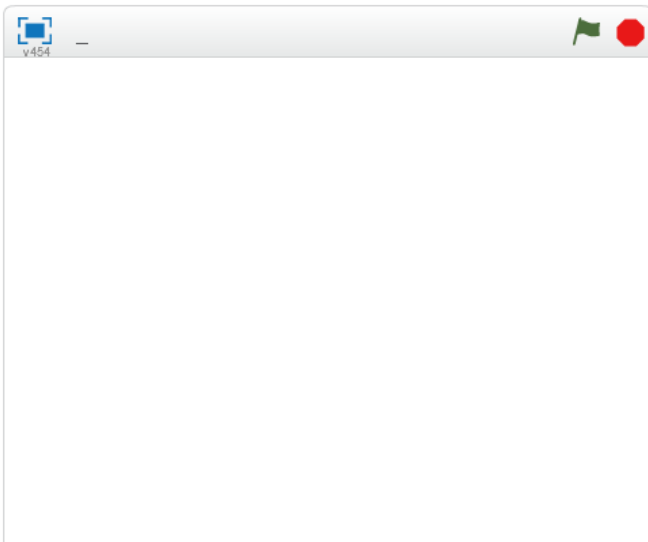
อธิบายสิ่งที่เกิดขึ้นในหน้านี้

สิ่งที่ต้องใช้ในหน้านี้ เช่น ตัวละคร เสียง ตัวอย่างโปรเจก เป็นต้น



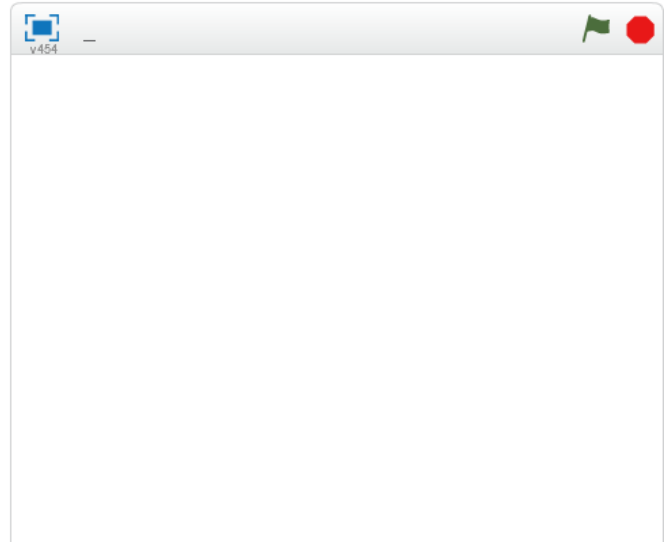
อธิบายสิ่งที่เกิดขึ้นในหน้านี้

สิ่งที่ต้องใช้ในหน้านี้ เช่น ตัวละคร เสียง ตัวอย่างโปรเจก เป็นต้น



อธิบายสิ่งที่เกิดขึ้นในหน้านี้

สิ่งที่ต้องใช้ในหน้านี้ เช่น ตัวละคร เสียง ตัวอย่างโปรเจก เป็นต้น



อธิบายสิ่งที่เกิดขึ้นในหน้านี้

สิ่งที่ต้องใช้ในหน้านี้ เช่น ตัวละคร เสียง ตัวอย่างโปรเจก เป็นต้น

### 3. พัฒนา (Develop)

ขั้นตอนนี้คือขั้นตอนของการพัฒนาโปรแกรม ทุกคนในทีมจะมีหน้าที่ที่ต่างกันในการพัฒนา สิ่งที่น่าำในขั้นตอนนี้คือ ในทีมควรกำหนดเวลาการมาอพบตงานกัน ขึ้นอยู่กับเวลาว่ามีมากหรือน้อย กรณีเวลามีทั้งหมด 6 ชั่วโมงในการทำโปรแกรม ทีมอาจจะมาอพบตข้อมูลการทำงานของสมาชิกในทีม กันทุก ๆ 1 ชั่วโมง เพื่อนำปัญหาที่แต่ละคนเจอมาหาทางแก้ไรร่วมกัน หรือจัดแบ่งหน้าที่กันใหม่ให้สำเร็จตามเวลา

คำถามแนะนำสำหรับการอพบตการทำงาน

1. อะไรเป็นส่วนที่ชอบที่สุดในการทำงานที่ผ่านมา
2. ส่วนใดที่ยังต้องทำต่อไป
3. ส่วนไหนที่คนอื่นในทีมจะต้องทำต่อจากเรา
4. ส่วนใดที่ต้องการความช่วยเหลือจากคนอื่นเพื่อให้งานเราไปต่อได้

### 4. แบ่งปัน (Share)

หลังจากที่เราวางแผน ออกแบบ พัฒนาได้ถึงระดับหนึ่งแล้ว เป็นโอกาสดีที่เราจะแบ่งปันเรื่องราวการทำงานของโปรแกรมเราให้กับคนอื่น ๆ ู้ เพื่อรับความแนะนำ ดีชม (Feedback) และรับฟังไอเดียการทำงานของโปรแกรมของคนอื่นเพื่อช่วยออกความเห็นเพื่อการพัฒนาของเขาและได้รับมุมมองที่แตกต่างสำหรับการพัฒนาโปรแกรมของเราเช่นกัน

คำถามสำหรับการให้ความเห็นกับโปรแกรมของคนอื่น

1. สิ่งทำงานได้ดีหรือสิ่งที่เราชอบมากในโปรแกรม
2. สิ่งที่ยังสับสนอยู่ว่าทำงานอย่างไร แนะนำว่าควรปรับเป็นแบบใด
3. สิ่งที่ทำให้ดีขึ้นได้หรือสิ่งที่ไม่ควรอยู่ในโปรแกรม

หลังจากได้รับคำแนะนำที่ชมมาเรียบร้อย เราจะกลับไปที่ขั้นตอนของการวางแผน (Plan) พัฒนาโปรแกรม (Develop) และกลับมาที่แบ่งปัน (Share) ใหม่ วนไปจนกระทั่งถึงเวลาที่จะแสดงให้คนอื่นเห็น (Showcase)

มาถึงจุดนี้ เราอาจจะรู้สึกกังวลหรือมีความเครียดกับการทำโปรแกรมไม่เสร็จดีนัก แต่จริง ๆ แล้วเราได้รับประสบการณ์มากมายที่เป็นจุดเริ่มต้นในการเป็นผู้พัฒนาโปรแกรม และการที่เราทำโปรแกรมมาได้ถึงขนาดนี้ก็ถือว่าสุดยอดมาก ๆ แล้ว

สุดท้ายสิ่งที่สร้างจะมีคุณค่ามากขึ้นเมื่อเราได้หันกลับมาถามตนเองว่าสิ่งที่เราได้สร้างสรรค์ ได้คิดอย่างเป็นระบบ ฯลฯ ส่งผลอย่างไรต่อตัวเราบ้าง ซึ่งเรามีชุดคำถามให้ตามนี้

1. โปรแกรมของเราคืออะไร
2. ไอเดียโปรแกรมของเราได้ช่วยแก้ปัญหาอย่างไร เราได้ไอเดียนี้มาอย่างไร
3. ขั้นตอนการพัฒนาโปรแกรมของกลุ่มเราเป็นอย่างไร
4. สิ่งใดในการพัฒนาน่าสนใจ ท้าทาย และน่าประหลาดใจ และทำไมถึงเกิดความรู้สึกแบบนั้นขึ้นมา
5. เราได้เรียนรู้อะไรบ้าง
6. สิ่งที่ชอบมากเกี่ยวกับโปรเจกต์นี้
7. อะไรที่เราอยากจะทำเปลี่ยน ถ้าเรากลับไปแก้ไขได้
8. ความรู้สึกต่อการเขียนโปรแกรมหรือการสร้างโปรแกรมเปลี่ยนไปจากตอนแรกอย่างไร





**CODE  
THE/R  
DREAMS**  
CDG

AFFILIATES OF CDG

CONTROL DATA (THAILAND) LTD. / CDG SYSTEMS LTD. / COMPUTER PERIPHERAL & SUPPLIES LTD. / GIS CO., LTD. / GLOBETECH CO., LTD. / ESRI (THAILAND) CO., LTD.