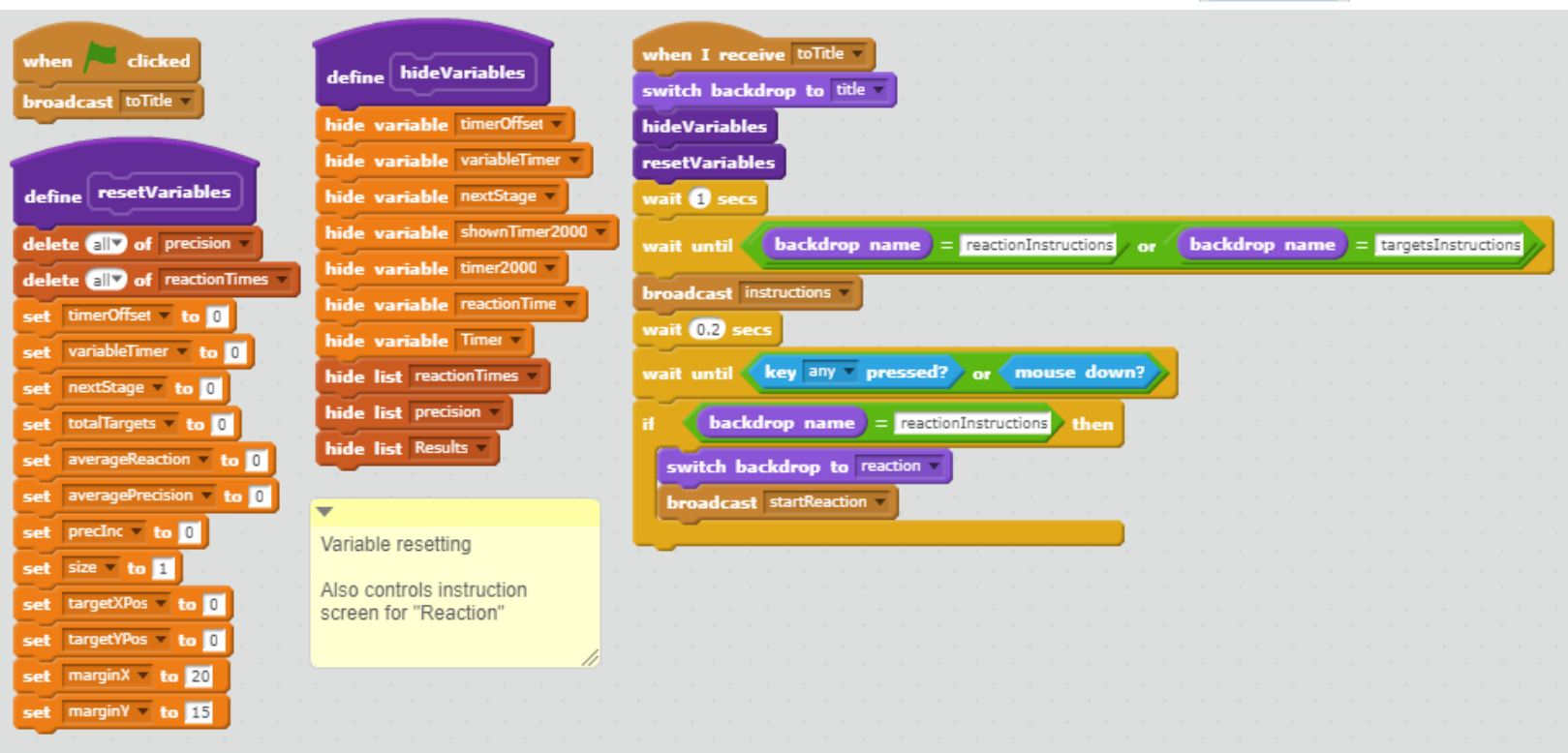


# Full Workspace

[illegible]

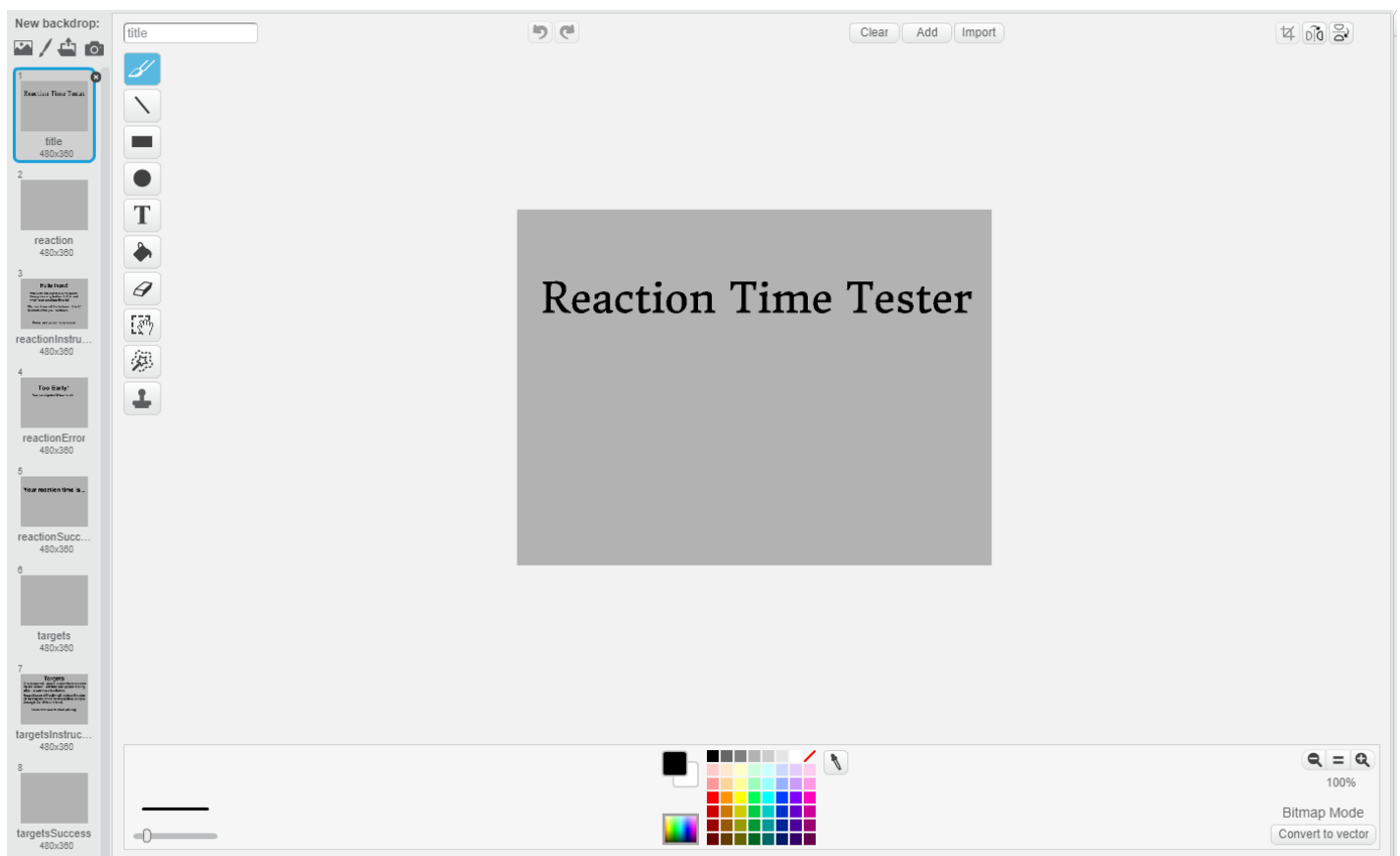
# Stage Code

Stage  
8 backdrops



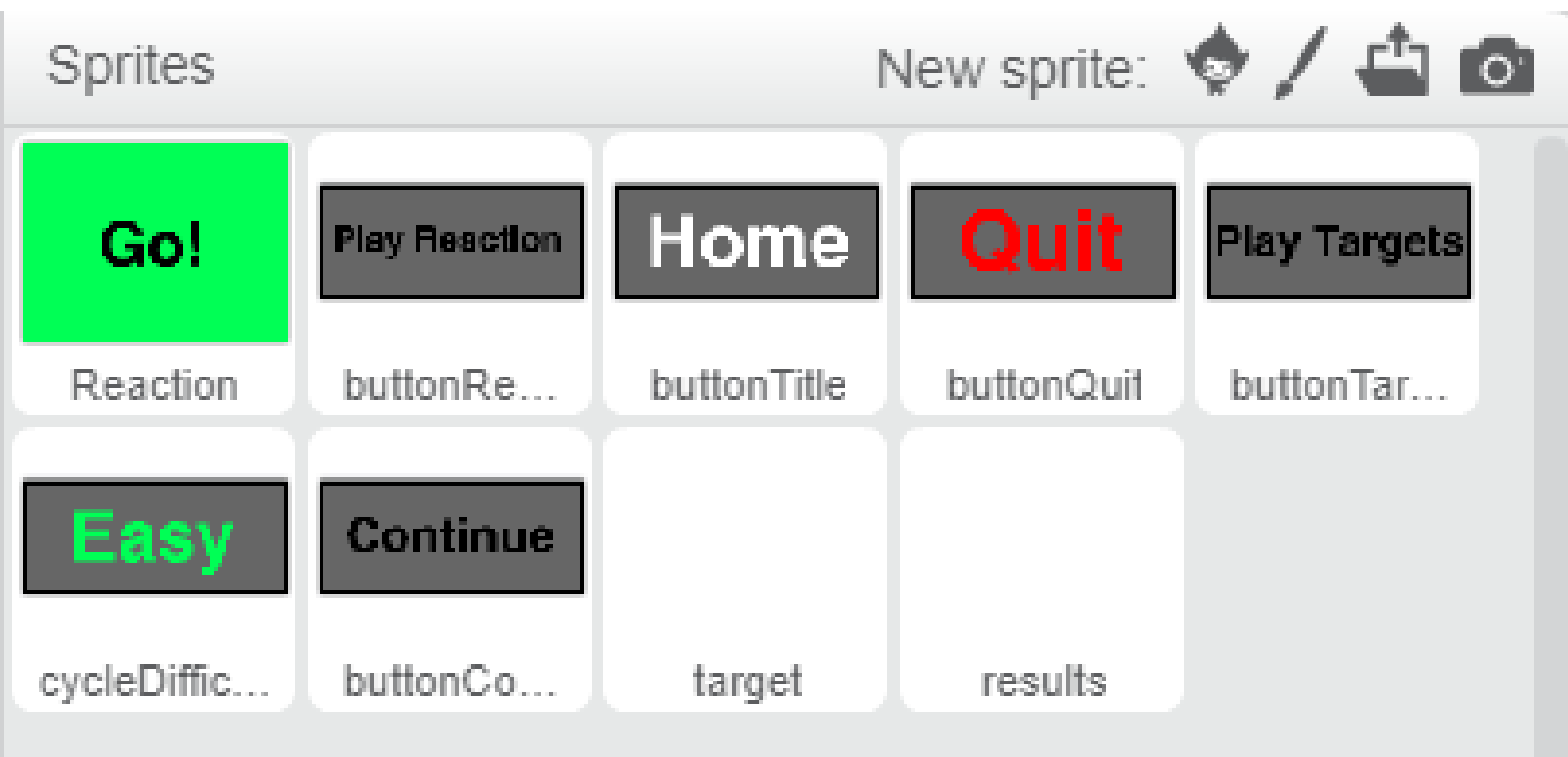
The image displays a collection of Scratch code blocks organized into three main sections. The first section, on the left, contains a 'when green flag clicked' event block followed by a 'broadcast to title' block. Below these is a 'define resetVariables' function block that includes multiple 'delete all of' blocks for 'precision' and 'reactionTimes', followed by a series of 'set' blocks for 'timerOffset', 'variableTimer', 'nextStage', 'totalTargets', 'averageReaction', 'averagePrecision', 'precInc', 'size', 'targetXPos', 'targetYPos', 'marginX', and 'marginY'. The second section, in the middle, is a 'define hideVariables' function block that lists 'hide variable' and 'hide list' blocks for 'timerOffset', 'variableTimer', 'nextStage', 'shownTimer2000', 'timer2000', 'reactionTime', 'Timer', 'reactionTimes', 'precision', and 'Results'. A yellow note box below this section states: 'Variable resetting' and 'Also controls instruction screen for "Reaction"'. The third section, on the right, is a 'when I receive toTitle' event block followed by a 'switch backdrop to title' block, 'hideVariables', 'resetVariables', 'wait 1 secs', and a 'wait until' block with conditions 'backdrop name = reactionInstructions' or 'backdrop name = targetsInstructions'. This is followed by a 'broadcast instructions' block, 'wait 0.2 secs', another 'wait until' block with conditions 'key any pressed?' or 'mouse down?', and an 'if' block with condition 'backdrop name = reactionInstructions' containing a 'switch backdrop to reaction' block and a 'broadcast startReaction' block.

# Backdrops



The image shows the Scratch Backdrops interface. On the left is a 'New backdrop:' panel with a list of 8 backdrops: 1. 'Reaction Time Tester' (480x360), 2. 'reaction' (480x360), 3. 'Pictorial' (480x360), 4. 'reactionInstru...' (480x360), 5. 'Too Early' (480x360), 6. 'reactionError' (480x360), 7. 'reactionSucc...' (480x360), and 8. 'targets' (480x360). The main workspace shows a grey backdrop with the text 'Reaction Time Tester' centered. The bottom of the interface features a toolbar with drawing tools, a color palette, and a zoom level of 100%.

# Sprites



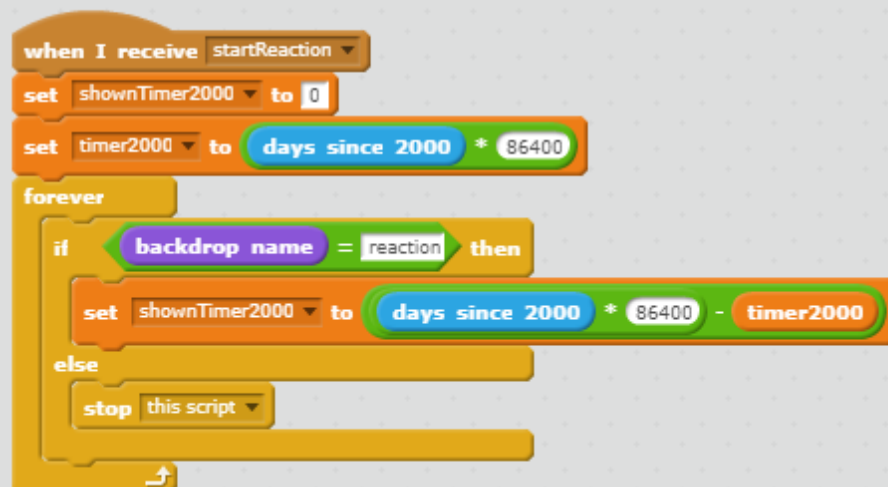
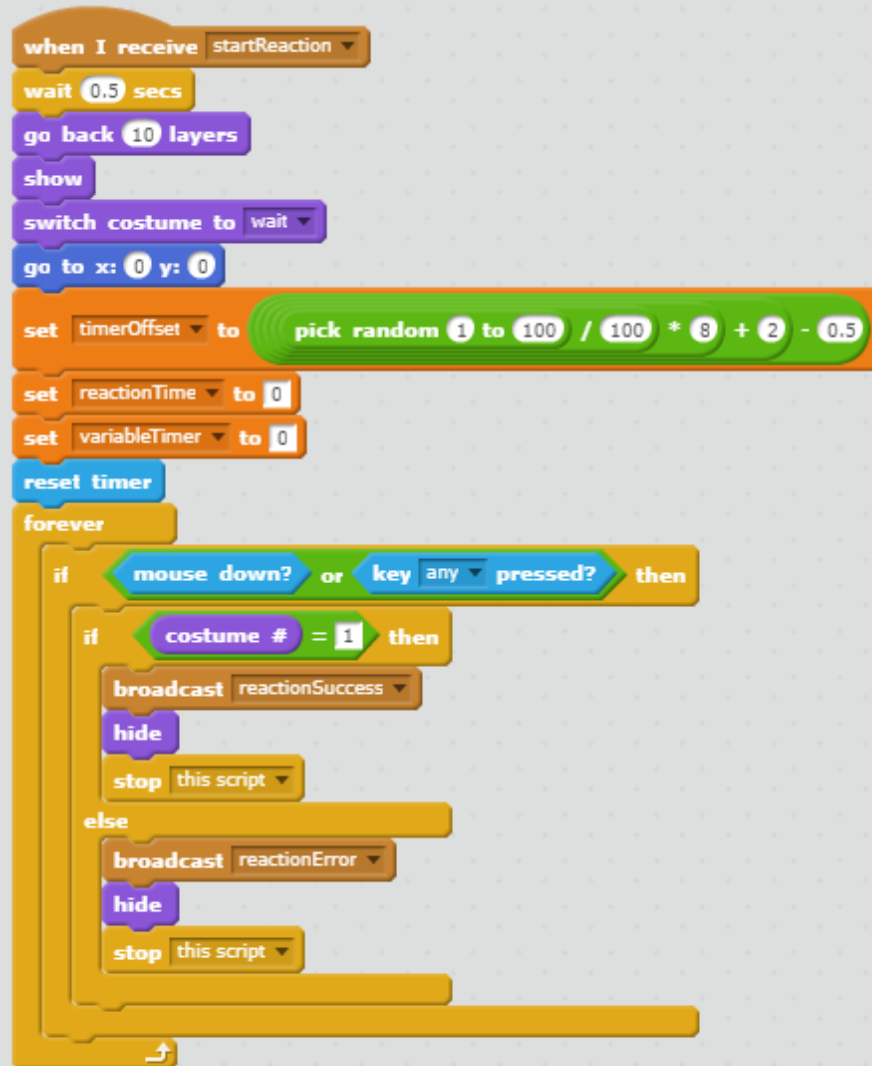
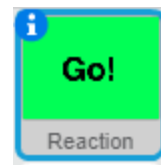
## Variables



## Lists



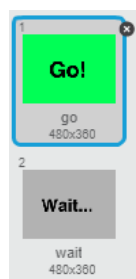
# Reaction



startReaction is called when the player continues on from the instructions when playing "Reaction".

There is a timer built into Scratch, but this only displays time to tenths of a second, which is not fast enough for a reaction game

## Costumes



# buttonReaction

A Scratch script for a button reaction. The script is organized into two columns. The left column contains: a 'when I receive' block for 'reactionError' followed by a 'hide' block; a 'when I receive' block for 'reactionSuccess' followed by a 'hide' block; a 'when I receive' block for 'instructions' followed by a 'hide' block; a 'define' block for 'hide' containing 'go to x: 0 y: 0' and a 'hide' block; and a 'when this sprite clicked' block followed by 'hide', 'switch backdrop to reactionInstructions', 'set nextStage to 1', and 'stop this script'. The right column contains: a 'when I receive' block for 'toTitle' followed by 'set nextStage to 0', 'set x to -100', 'set y to -100', and 'show'; and a 'when I receive' block for 'instructions' followed by 'stop other scripts in sprite'.

```
when I receive reactionError
hide

when I receive reactionSuccess
hide

when I receive instructions
hide

define hide
go to x: 0 y: 0
hide

when this sprite clicked
hide
switch backdrop to reactionInstructions
set nextStage to 1
stop this script

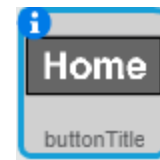
when I receive toTitle
set nextStage to 0
set x to -100
set y to -100
show

when I receive instructions
stop other scripts in sprite
```

Costumes



# buttonTitle

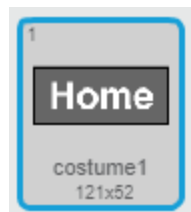


The script for the buttonTitle sprite consists of three event listeners and a custom function definition:

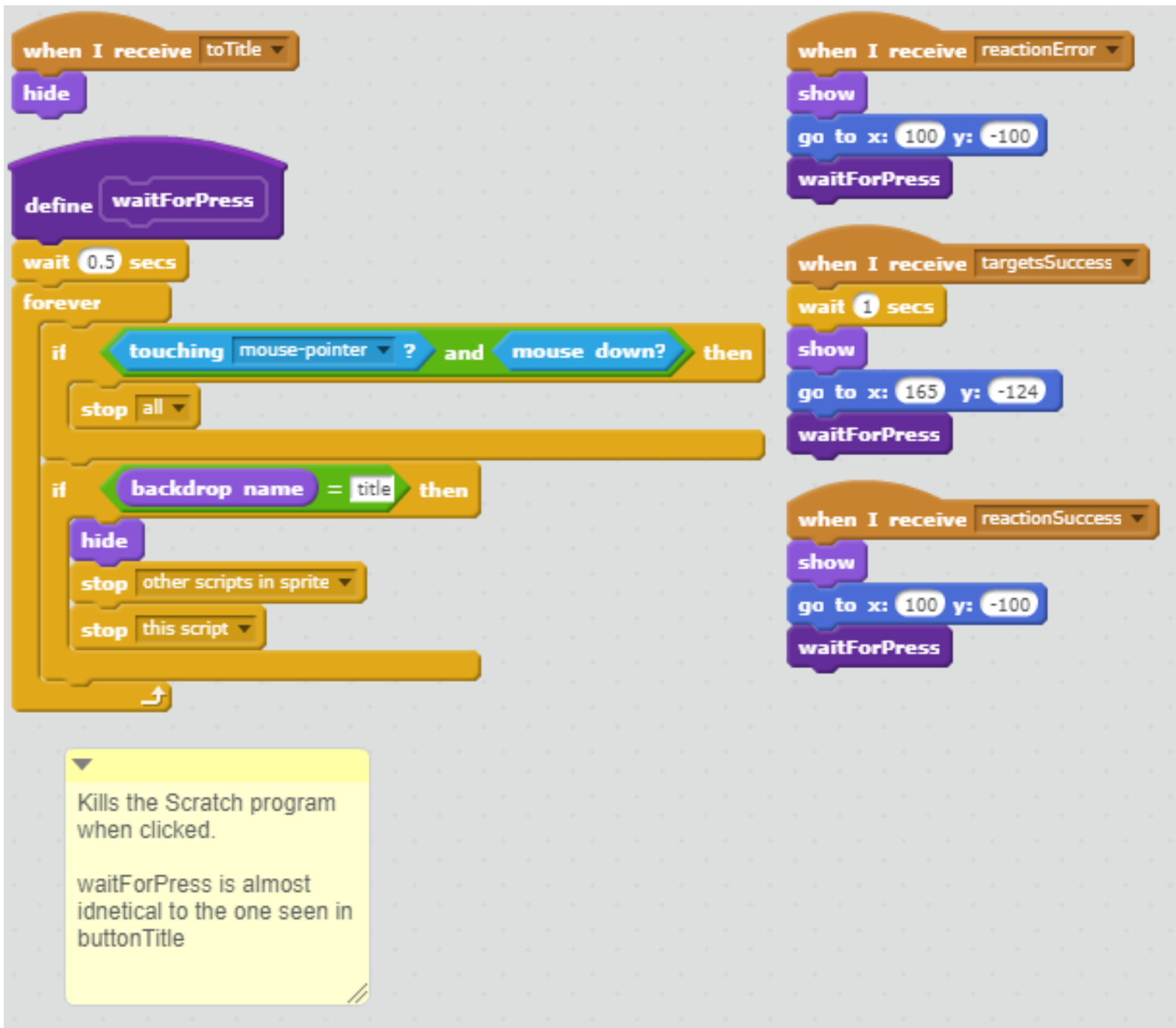
- when I receive toTitle**: go to x: 0 y: 0, hide.
- when I receive targetsSuccess**: wait 1 secs, show, go to x: 0 y: -125, waitForPress.
- when I receive reactionError**: show, go to x: -100 y: -100, waitForPress.
- define waitForPress**: wait 0.5 secs, forever loop containing:
  - if touching mouse-pointer ? and mouse down? then:
    - switch backdrop to title
    - hide
    - broadcast toTitle
    - stop this script

A yellow note box at the bottom left explains: "waitForPress waits until the user clicks the sprite or holds down the mouse button and travels over the button. It then executes the code inside the if statement".

## Costumes



# buttonQuit



```
when I receive toTitle
  hide

  define waitForPress
    wait 0.5 secs
    forever
      if touching mouse-pointer ? and mouse down? then
        stop all
      if backdrop name = title then
        hide
        stop other scripts in sprite
        stop this script

when I receive reactionError
  show
  go to x: 100 y: -100
  waitForPress

when I receive targetsSuccess
  wait 1 secs
  show
  go to x: 165 y: -124
  waitForPress

when I receive reactionSuccess
  show
  go to x: 100 y: -100
  waitForPress
```

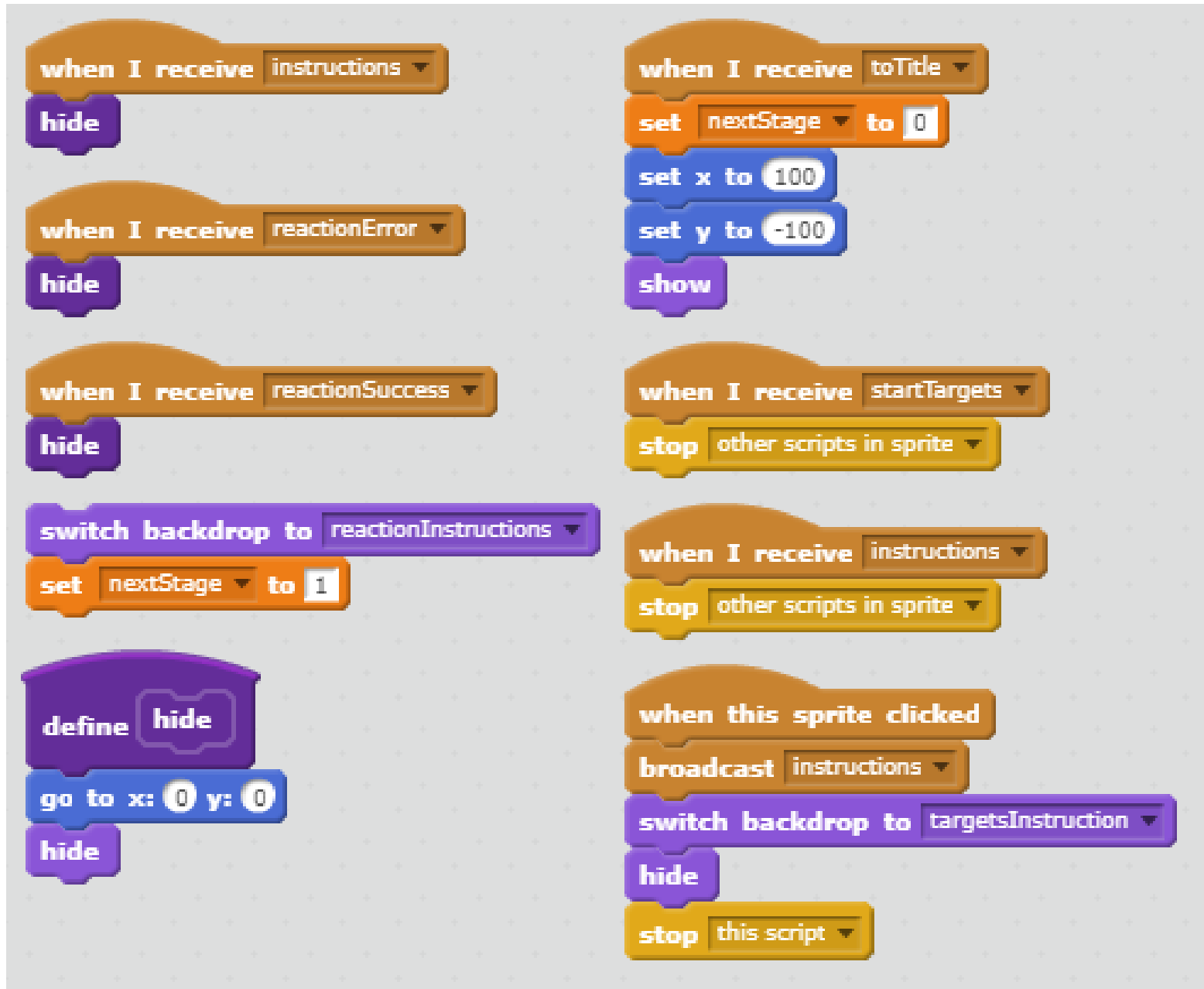
Kills the Scratch program when clicked.

waitForPress is almost identical to the one seen in buttonTitle

## Costumes



# buttonTargets

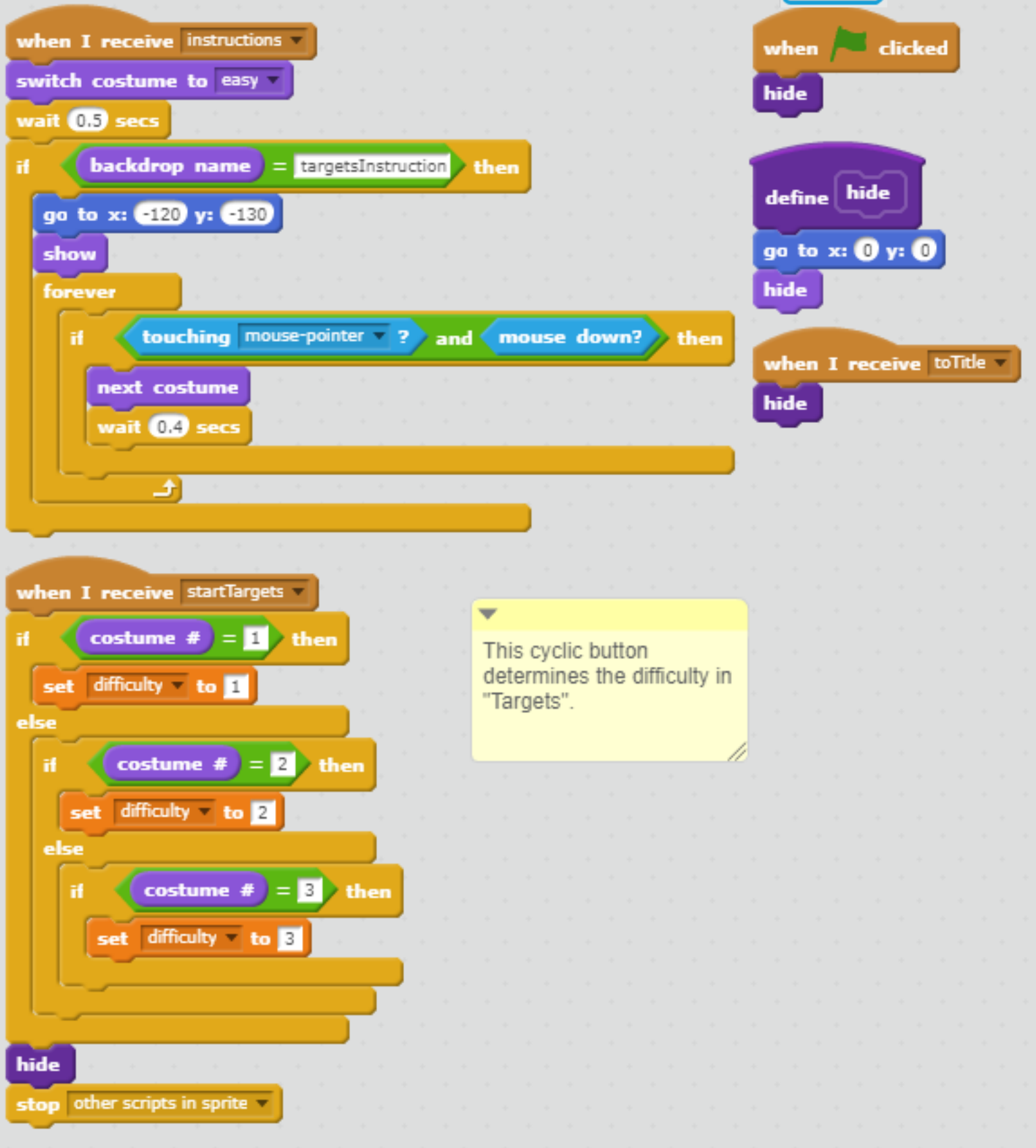


Costumes





# cycleDifficulty



The image displays two Scratch scripts for a 'cycleDifficulty' button. The first script, triggered by 'when I receive instructions', switches the costume to 'easy', waits 0.5 seconds, and then enters a 'forever' loop. Inside the loop, it checks if the backdrop name is 'targetsInstruction'. If true, it moves the button to x: -120, y: -130, shows it, and enters another 'forever' loop. This second loop checks if the button is 'touching mouse-pointer?' and 'mouse down?'. If both are true, it cycles to the next costume and waits 0.4 seconds. The second script, triggered by 'when I receive startTargets', checks the current costume number (1, 2, or 3) and sets the 'difficulty' variable accordingly. After setting the difficulty, it hides the button and stops other scripts in the sprite.

**Script 1: when I receive instructions**

- switch costume to easy
- wait 0.5 secs
- if backdrop name = targetsInstruction then
  - go to x: -120 y: -130
  - show
  - forever loop:
    - if touching mouse-pointer? and mouse down? then
      - next costume
      - wait 0.4 secs

**Script 2: when I receive startTargets**

- if costume # = 1 then
  - set difficulty to 1
- else
  - if costume # = 2 then
    - set difficulty to 2
  - else
    - if costume # = 3 then
      - set difficulty to 3
- hide
- stop other scripts in sprite

**Function: define hide**

- go to x: 0 y: 0
- hide

**Script: when I receive toTitle**

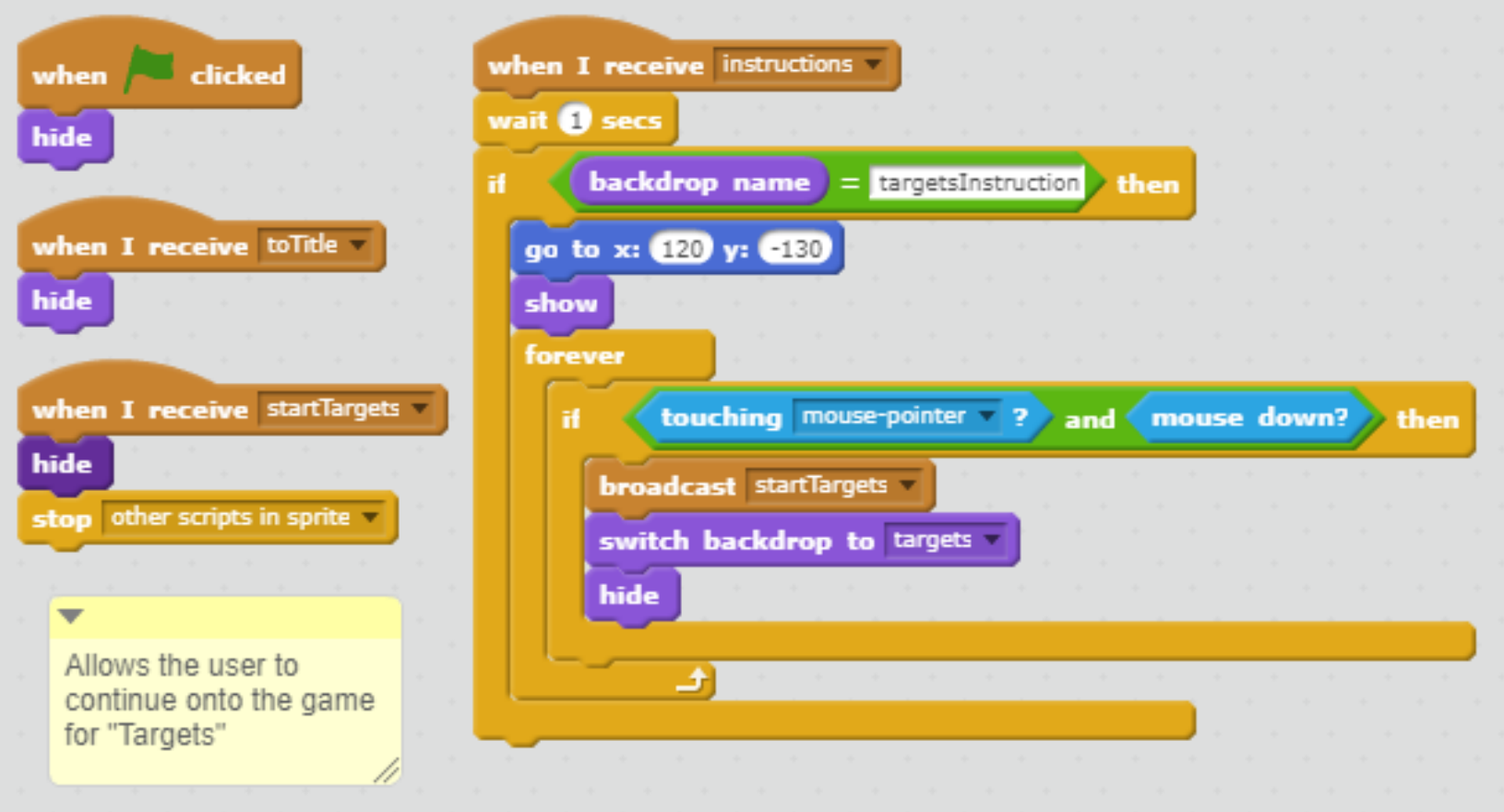
- hide

**Annotation:** This cyclic button determines the difficulty in "Targets".

Costumes



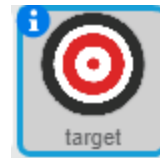
# buttonContinue



Costumes



# target



when I receive toTitle

hideTarget

when clicked

hideTarget

when I receive startTargets

wait 0.7 secs

forever

set Timer to round 30 - timer

wait 0.05 secs

when I receive startTargets

set Timer to 60

show variable Timer

reset timer

wait pick random 1 to 100 / 100 \* 2 + 2 secs

create clone of target

wait 24 secs

forever

if timer > 30 then

hide variable Timer

stop other scripts in sprite

repeat totalTargets + 1

hide

delete this clone

broadcast targetsSuccess

stop this script

define addData

add shownTimer2000 to reactionTimes

add sort of mouse x · x position · mouse x · x position + mouse y · y position · mouse y · y position to precision

change totalTargets by 1

when I start as a clone

setupTarget

set timer2000 to days since 2000 · 86400

forever

set shownTimer2000 to days since 2000 · 86400 · timer2000

if targetClicked = 1 then

hideTarget

addData

wait pick random 1 to 100 / 100 \* 2 + 0.5 secs

create clone of target

delete this clone

define hideTarget

switch costume to hidden

hide

set size to 10 %

when I receive targetsSuccess

delete this clone

define setupTarget

set targetXPos to pick random -240 + marginX to 240 · marginX

set targetYPos to pick random -180 + marginY to 180 · marginY

go to x: targetXPos y: targetYPos

switch costume to target

if difficulty = 3 then

set size to 10

else

if difficulty = 2 then

set size to 11

else

if difficulty = 1 then

set size to 20

set size to size %

show

set targetClicked to 0

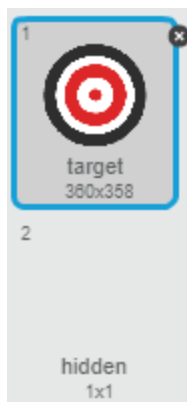
The target in "Targets"

Initially, it creates a clone of itself to start "Targets". Everytime it is clicked, it hides itself, adds data to precision and reaction time lists, then creates a new target, and finally deletes the current clone. Everytime a new clone is created, the cycle repeats.

Margins are used in "setupTarget" to prevent the target from spawning extremely close to the game border.

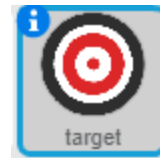
Target Image Source:  
<http://pluspng.com/img-png/target-symbol-image-4534-1323.png>

Costumes



Better resolution snips on following pages

# target

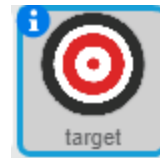


```
when I start as a clone
  setupTarget
  set timer2000 to days since 2000 * 86400
  forever
    set shownTimer2000 to days since 2000 * 86400 - timer2000
    if targetClicked = 1 then
      hideTarget
      addData
      wait pick random 1 to 100 / 100 * 2 + 0.5 secs
      create clone of target
      delete this clone
```

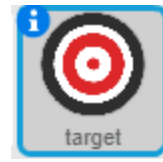
```
define setupTarget
  set targetXPos to pick random -240 + marginX to 240 - marginX
  set targetYPos to pick random -180 + marginY to 180 - marginY
  go to x: targetXPos y: targetYPos
  switch costume to target
  if difficulty = 3 then
    set size to 10
  else
    if difficulty = 2 then
      set size to 18
    else
      if difficulty = 1 then
        set size to 25
  set size to size %
  show
  set targetClicked to 0
```

Abstraction

# target



# target



when this sprite clicked

set targetClicked to 1

define hideTarget

switch costume to hidden

hide

set size to 10 %

when I receive targetsSuccess

delete this clone

The target in "Targets"

Initially, it creates a clone of itself to start "Targets". Everytime it is clicked, it hides itself, adds data to precision and reaction time lists, then creates a new target, and finally deletes the current clone. Everytime a new clone is created, the cycle repeats.

Margins are used in "setupTarget" to prevent the target from spawning extremely close to the game border.

Target Image Source:  
<http://pluspng.com/img-png/target-symbol-image-4534-1323.png>

Target Image Source:

<http://pluspng.com/img-png/target-symbol-image-4534-1323.png>

define addData

add shownTimer2000 to reactionTimes

add sqrt of  $\sqrt{(\text{mouse } x - x \text{ position})^2 + (\text{mouse } y - y \text{ position})^2}$  to precision

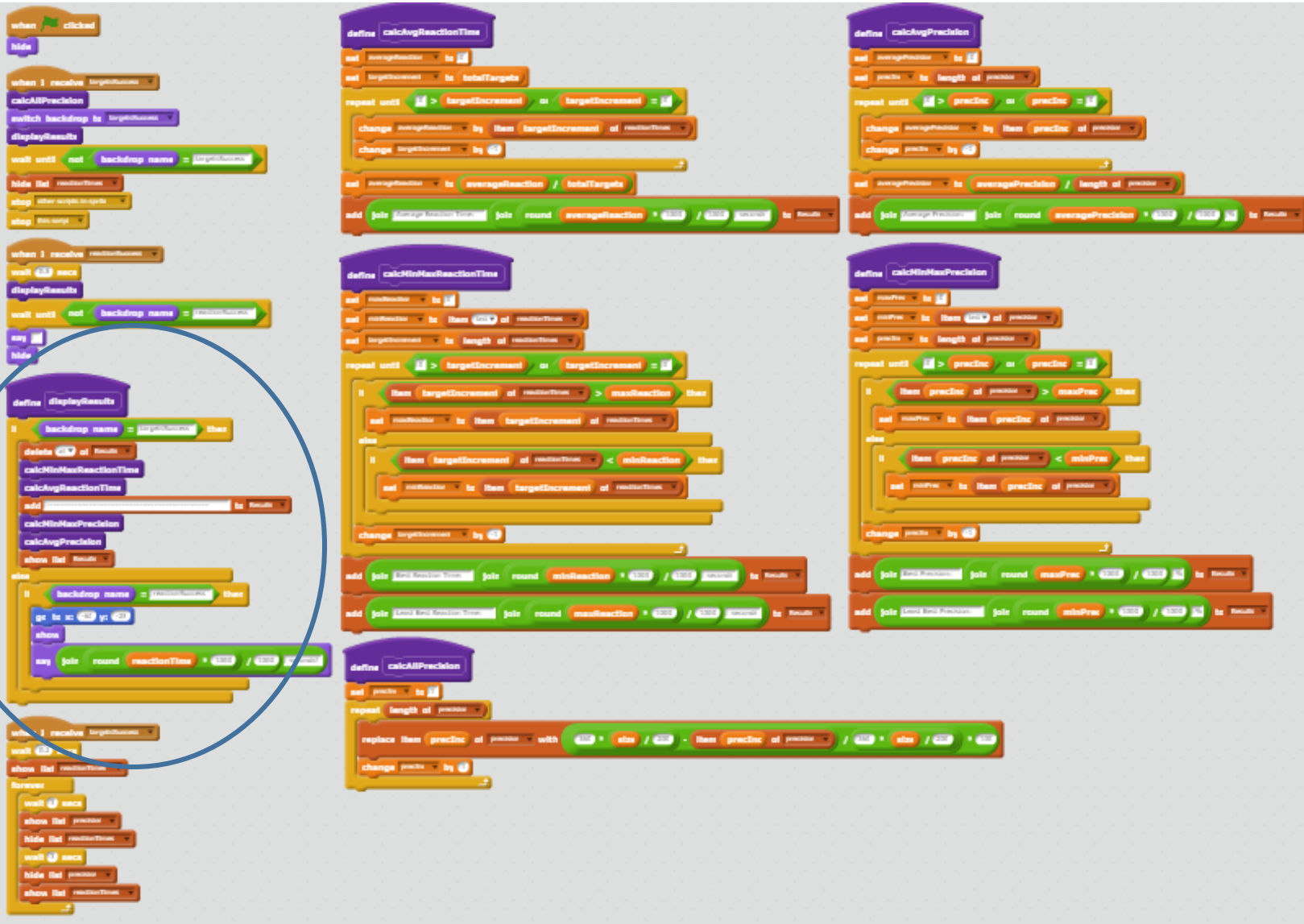
change totalTargets by 1

Set Size to Size %

show

set targetClicked to 0

# results

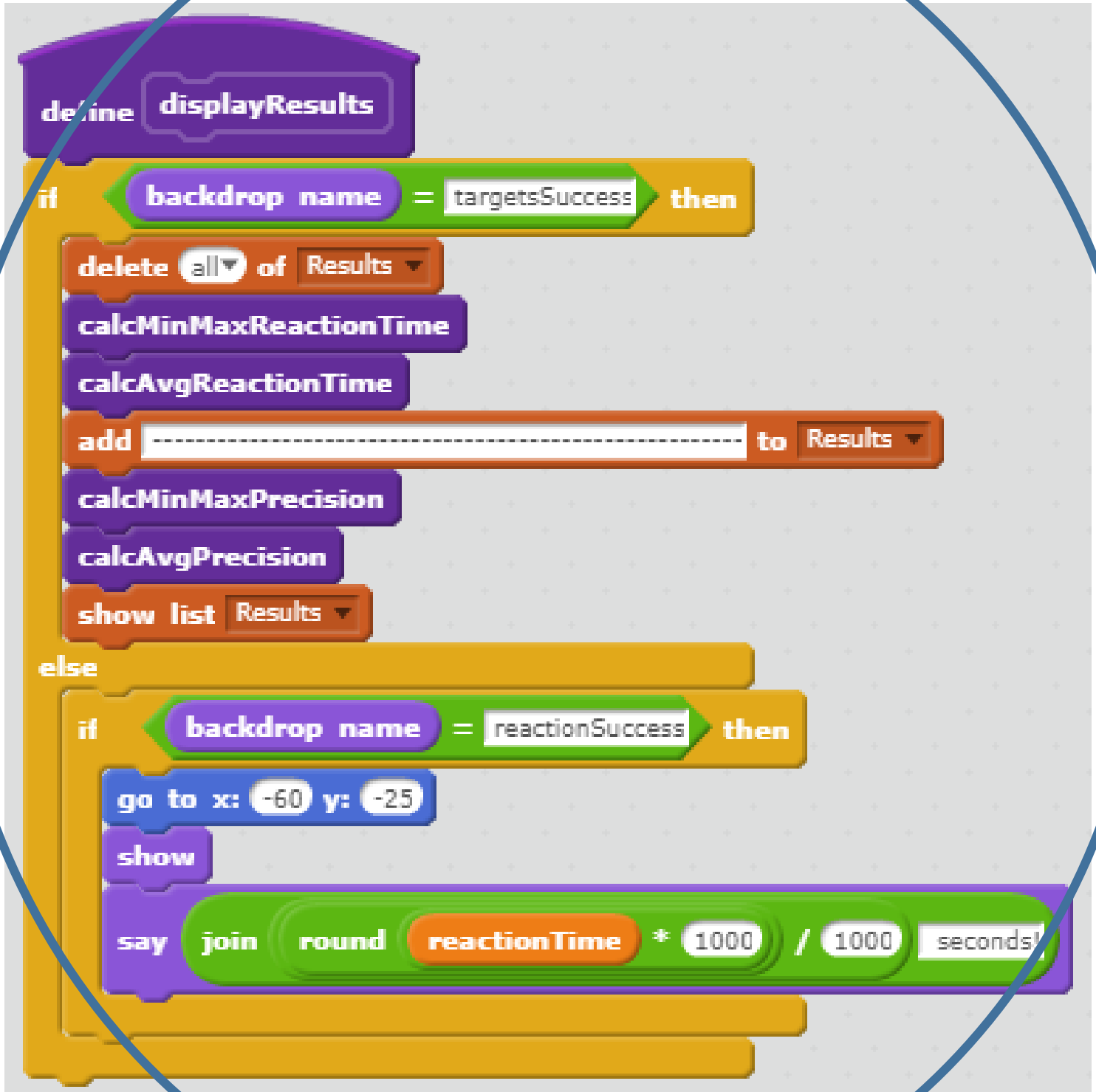


Costumes



Better resolution snips on following pages

# results

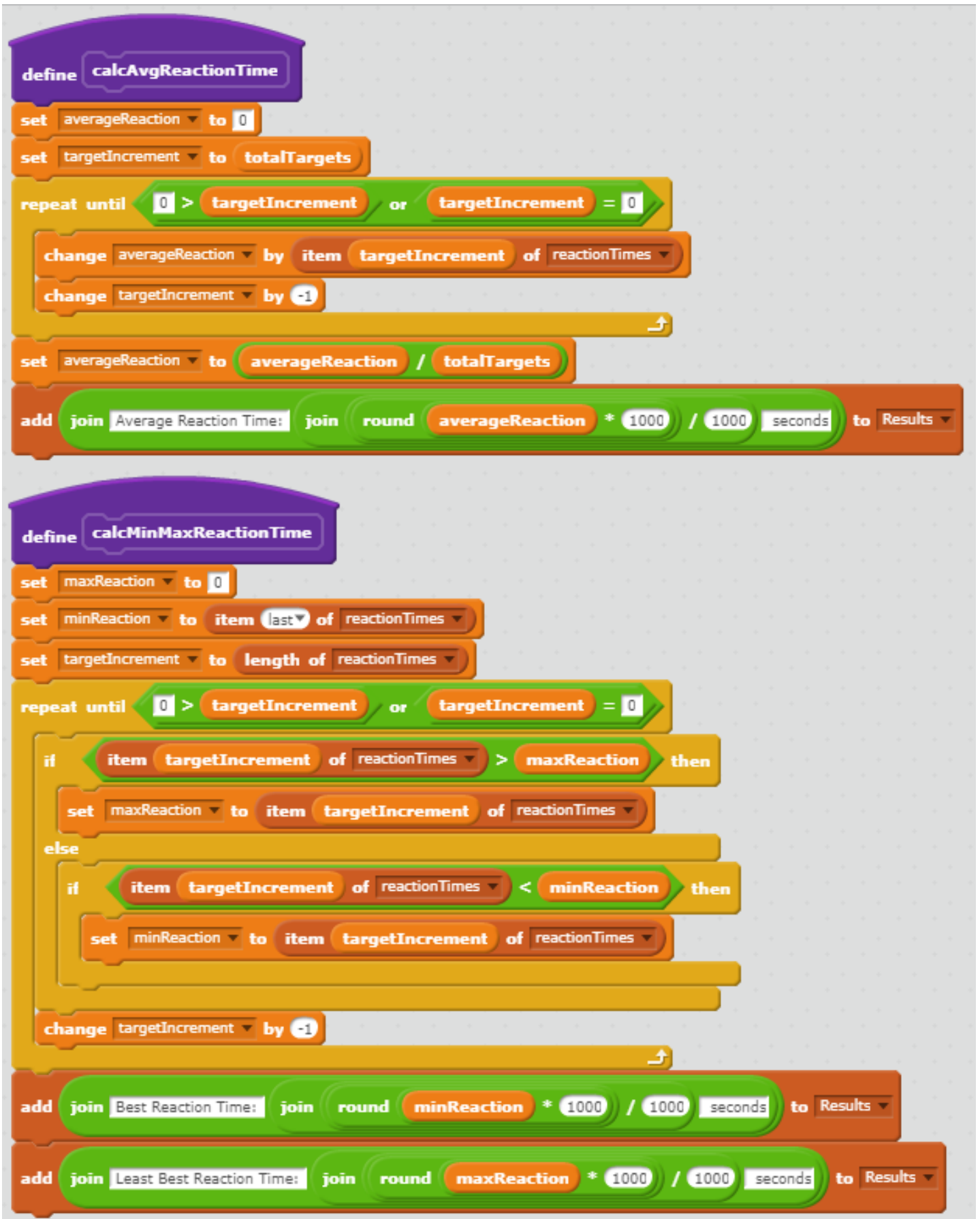


## Algorithm

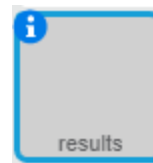
Sub-algorithms on  
following pages



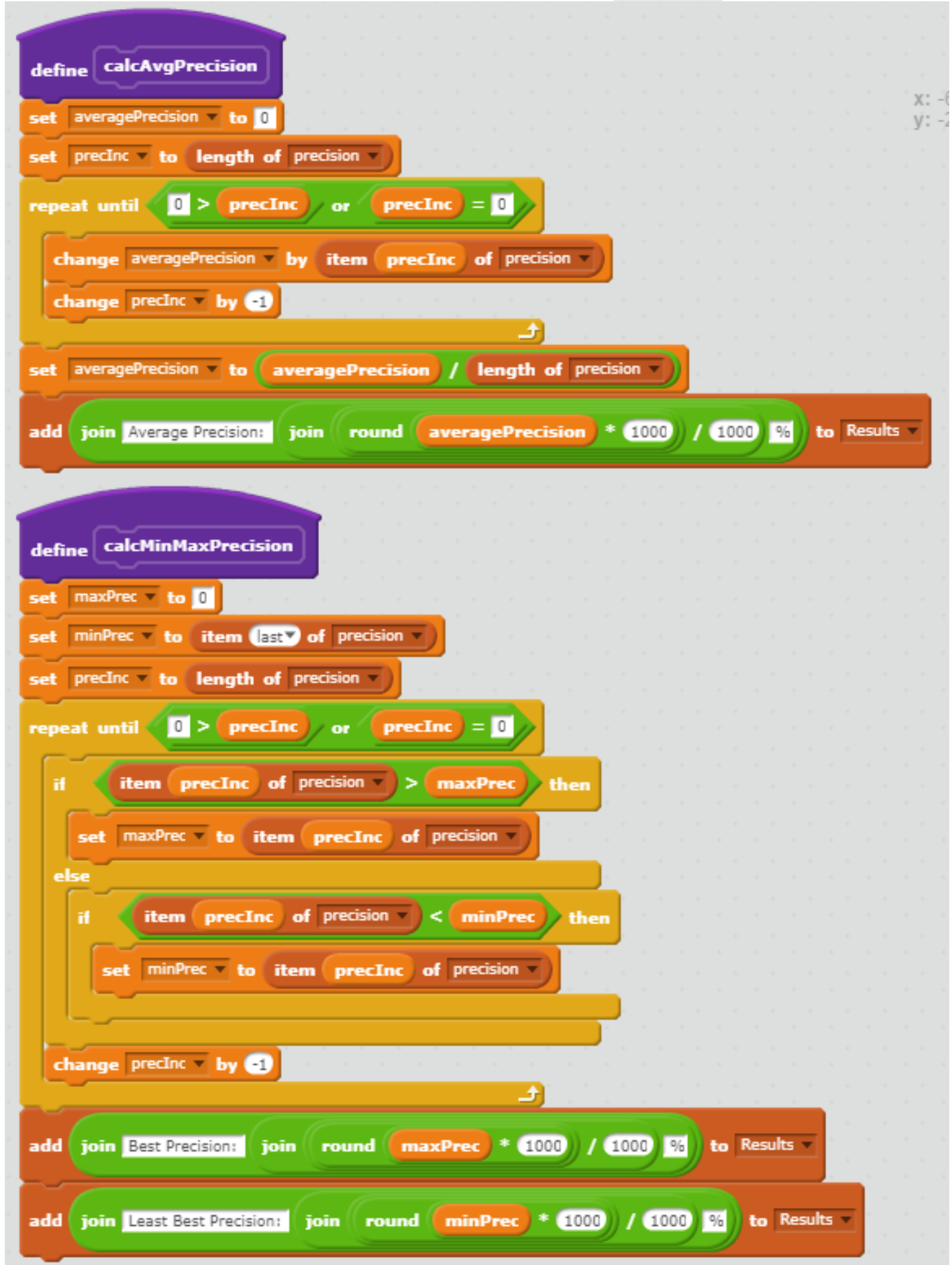
# results



# results



x: -6  
y: -2



# results

A Scratch script on a light gray grid background. The script consists of two main event-driven blocks. The first block is triggered by 'when clicked' and contains a 'hide' block. The second block is triggered by 'when I receive targetsSuccess' and contains a sequence of actions: 'calcAllPrecision', 'switch backdrop to targetsSuccess', 'displayResults', a 'wait until' loop with the condition 'not backdrop name = targetsSuccess', 'hide list reactionTimes', 'stop other scripts in sprite', and 'stop this script'. The third block is triggered by 'when I receive reactionSuccess' and contains: 'wait 0.5 secs', 'displayResults', a 'wait until' loop with the condition 'not backdrop name = reactionSuccess', 'say' (with an empty text box), and 'hide'.

```
when clicked
hide

when I receive targetsSuccess
calcAllPrecision
switch backdrop to targetsSuccess
displayResults
wait until not backdrop name = targetsSuccess
hide list reactionTimes
stop other scripts in sprite
stop this script

when I receive reactionSuccess
wait 0.5 secs
displayResults
wait until not backdrop name = reactionSuccess
say 
hide
```

# results

