

# Snap! Manual:

Version 3.1, 4.0

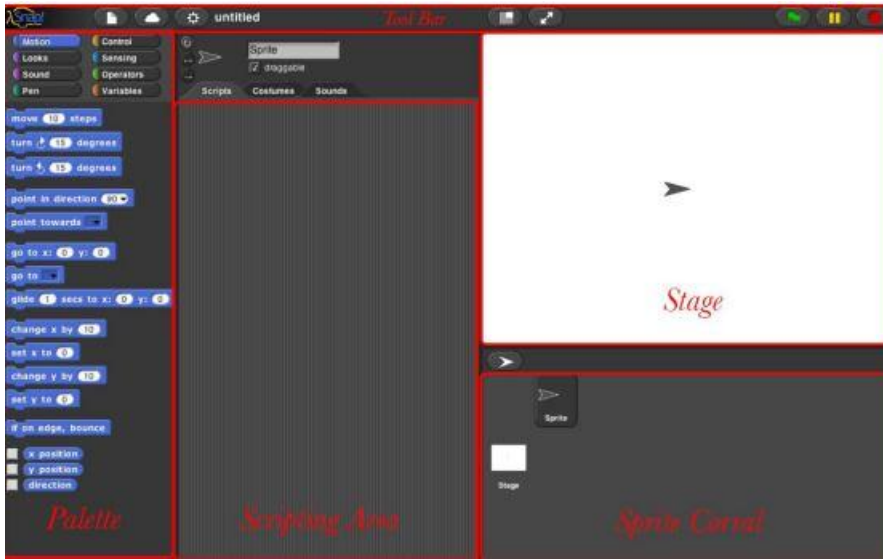
---

## Build Your Own Blocks



Build Your Own Blocks

## Blocks, Scripts, and Sprites : 1장



Snap!은 나만의 블록을 만들수 있는 스크래치(<http://scratch.mit.edu>)의 확장판

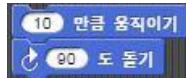
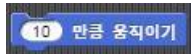
상단의 툴바, 하단의 팔레트, 스크립트, 무대, 스프라이트 구역으로 화면구성

Typical script:



블럭형 프로그래밍 언어  
Hat Blocks, Command Blocks

## A. Sprites and Parallelism



hat block, command blocks을 이용하여  
간단한 코딩하기



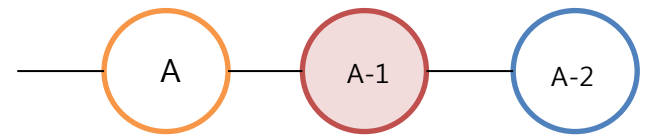
- 1) 녹색깃발을 눌러 실행하기
- 2) 스페이스키를 눌러 실행하기



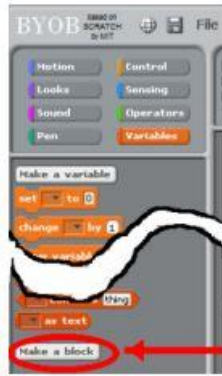
- 3) 스프라이트 모양 바꾸기
- 4) 스프라이트 음향 효과



Build Your Own Blocks




## A-1. Building a Block



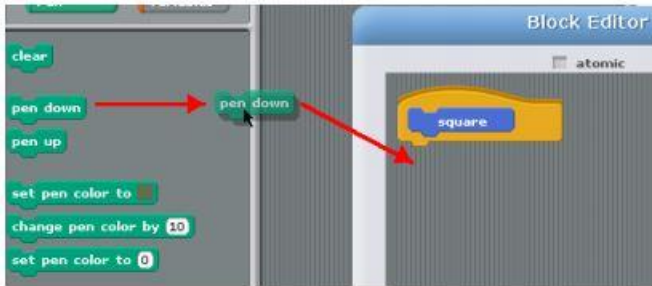
변수팔레트 아래 "Make a block." 실행



- 1) 모션팔레트 선택
- 2) 커맨드 블록 선택
- 3) "square"  삽입

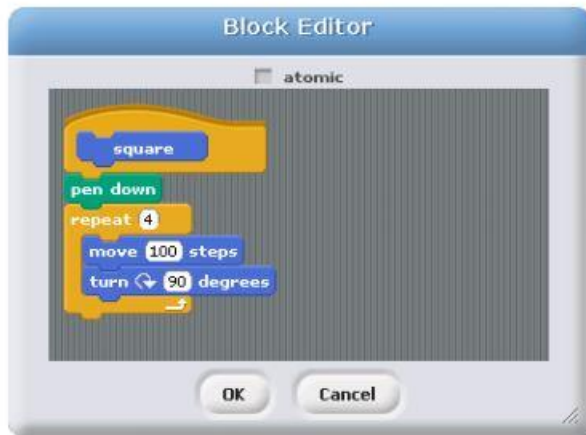
## A-1. Building a Block

---

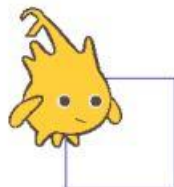
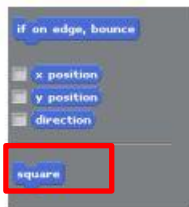


The jigsaw-puzzle

- 1) 블럭에디터에서 코딩하기
- 2) 새로 만든 블록 실행하기

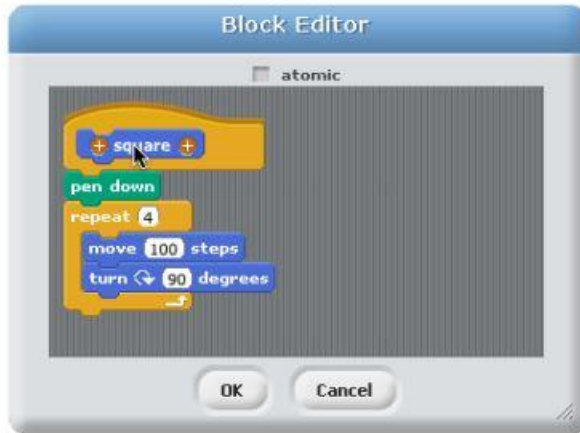


"prototype" is unrelated to the prototyping object oriented programming discussed later.



Build Your Own Blocks

## A-1. Building a Block



블록에디터로 변수 블록만들기

- 1) square블록 마우스 가져가기
- 2) 십자모양 클릭
- 3) size 입력하기

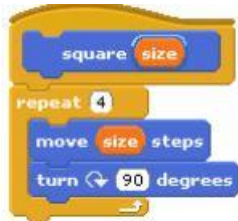


## A-1. Building a Block

---

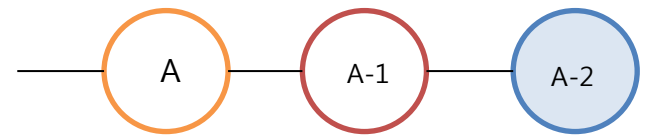


size를 달리하여 사각형 만들기

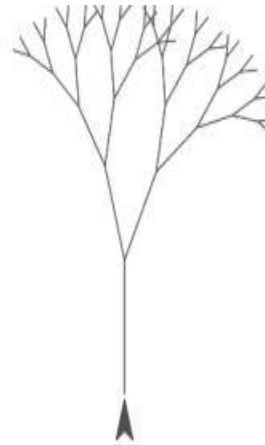
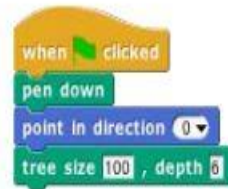
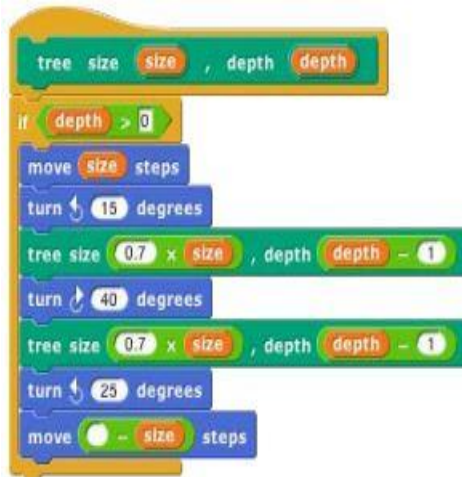


1) 원형 size블럭을 "move ( ) steps" 블록으로 옮기기

2) 모션팔레트에 새로 생긴 블록  실행하기



## A-2. Recursion



프렉탈트리 만들기

예제파일<http://>



Build Your Own Blocks



## A-2. Recursion

The image shows a Scratch code editor window titled "stamp". The script is designed to draw a complex, fractal-like shape using a recursive process. The code is as follows:

```
clear
pen up
go to x: 0 y: 0
pen down

set pen color to red

repeat until key space pressed?
  move pick random 4 to 8 steps
  turn pick random -180 to 180 degrees
  set pen color to pick random 1 to 100
  stamp
  if on edge, bounce

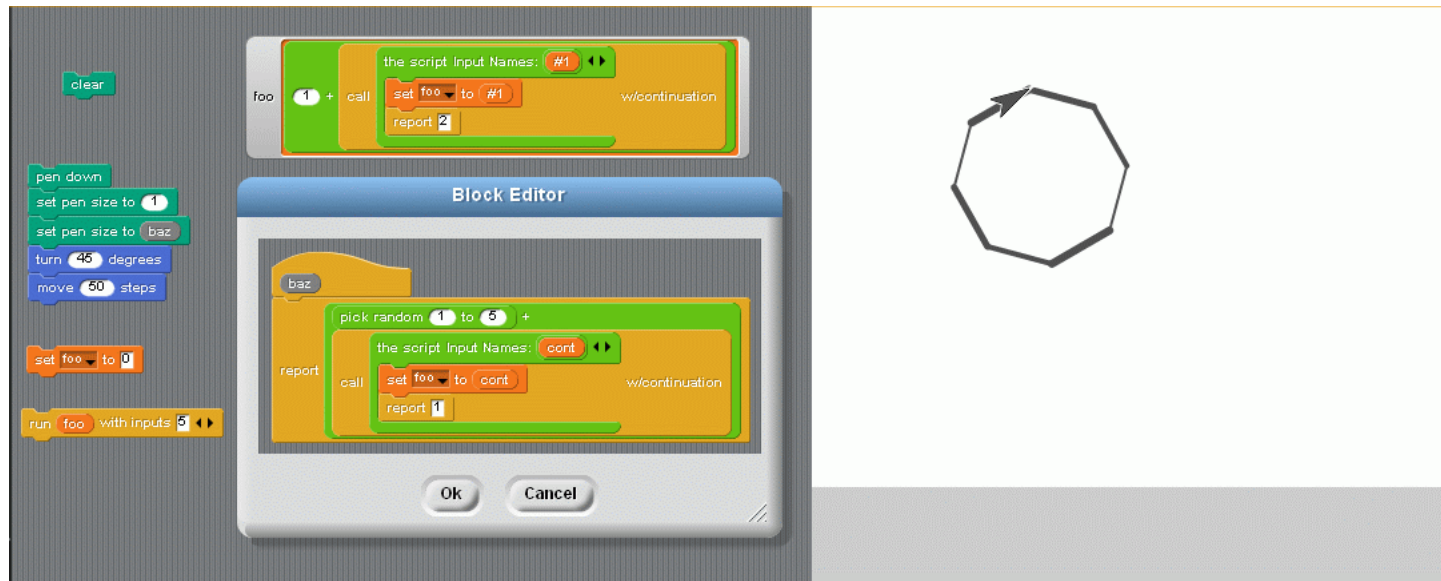
set pen size to pick random 3 to 10

repeat 5000
  move pick random 4 to 8 steps
  turn pick random -180 to 180 degrees
  set pen size to pick random 3 to 10
  set pen color to pick random 1 to 100
  stamp
  if on edge, bounce
```

The right side of the editor shows the stage with the resulting drawing. The drawing is a complex, fractal-like shape composed of many small, overlapping, colorful triangles (red, green, blue, yellow, purple, etc.) arranged in a pattern that resembles a stylized letter 'A' or a complex geometric design. The stage also shows a "Turtle" icon and a "Stage" label.

## A-2. Recursion

---



## A-2. Recursion

The image shows the Scratch IDE with four scripts for a turtle named 'Turtle' in the 'spiral01' workspace. Each script uses a 'report' block to select a different drawing pattern.

- MOTIF:** Starts at (0,0), points in direction 90, clears, and enters a repeat loop of 50. Inside the loop, it moves  $a \times 3$  steps, turns 50 degrees, and changes  $a$  by 2.
- ROSACE:** Starts at (0,0), points in direction 90, clears, and enters a repeat loop of 3. Inside, it repeats 18 times: move 30 steps, turn 100 degrees, and turn 60 degrees.
- SOLEIL:** Starts at (-150, 50), points in direction 90, clears, and enters a repeat loop of 30. Inside, it moves 200 steps and turns 150 degrees.
- SPIRALE QUADRATIQUE:** Starts at (0,0), points in direction 90, clears, and enters a repeat loop of 40. Inside, it repeats 4 times: move  $a$  steps, turn 90 degrees, change  $a$  by 3, and turn 15 degrees.

The 'report' block in the top right is set to 'ROSACE', and the stage displays the resulting fractal pattern.



## A-2. Recursion

The screenshot displays the Snap! Block Editor interface. On the left is the 'Block Palette' with categories: Motion, Looks, Sound, Pen, Control, Sensing, Operators, and Variables. The main workspace is titled 'Block Editor' and contains a script for a recursive drawing function. The script is as follows:

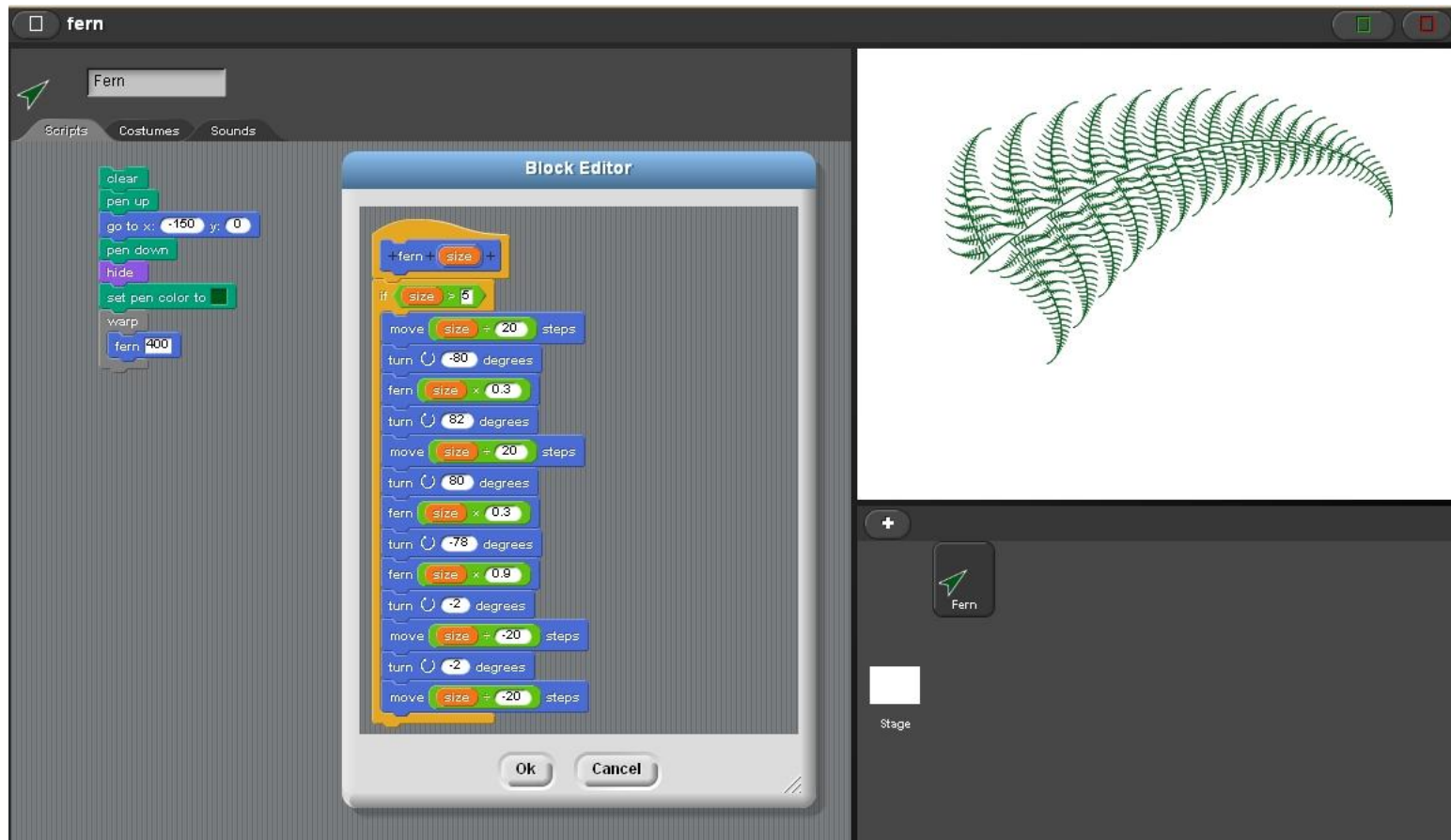
```
script variables ta
set ta to vary
repeat 11
  the script
  script variables sa
  set sa to 1
  repeat 12
    pen up
    point in direction 90
    move sa steps
    pen down
    the script
    script variables fa fb
    set fa to 40
    set fb to 80
    repeat 5
      move fa steps
      turn fb degrees
  change ta by 40
  pen up
  go to x: varx y: ta
```

On the right side of the workspace, there is a separate block of code for setting up the stage:

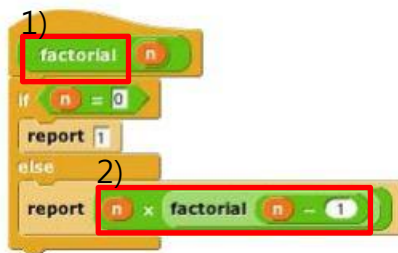
```
set varx to 240
set vary to 220
clear
set pen color to black
set pen size to 1
pen up
go to x: varx y: vary
pen down
hide
Grid
```

At the bottom right, there is a 'Stage' area with a 'Turtle' icon and a 'Grid' button.

## A-2. Recursion



## A-2. Recursion



- 1) 팩토리얼 변수 n
- 2) 1부터 변수n까지의 정수의 곱
- 3) 팩토리얼 5의 값



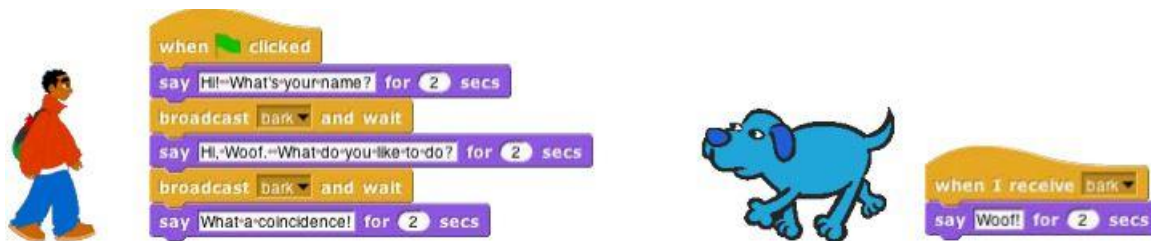
factorial 함수

```
For(i=0; i<n; i++)
{
  sum = sum*(i+1);
}
```



Build Your Own Blocks

## A. Sprites and Parallelism



### 방송하기

- 1) 형태팔레트에서 말하기 선택
- 2) 제어팔레트에서 방송하기 선택

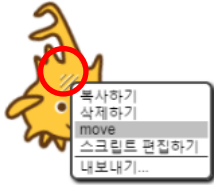
\*방송하기는 서로 다른 모양의 스프라이트를 활용하여 대화하듯이 말을 이어 나간다.



### 스프라이트 모양 바꾸기

## B. Nesting Sprites: Anchors and Parts

---



1)



2)

스프라이트 제어하기

1) 모양이 회전하며 변한다.

2) 모양이 변하지 않고 그대로이다.



1)

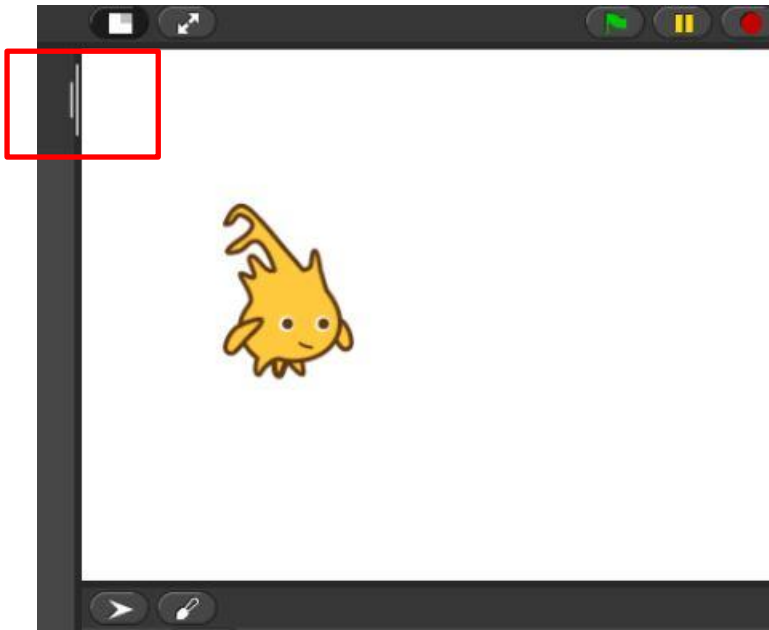


2)



## B. Nesting Sprites: Anchors and Parts

---

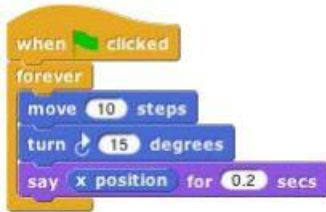


무대 크기 조절

- 1) 마우스 드레그
- 2) 무대크기를 조절할 수 있음

## C. Reporter Blocks & Expressions

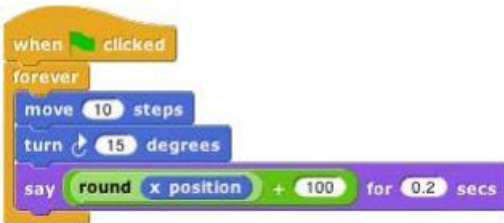
---



35.39058011260056

X, Y좌표 읽기

- 1) 마우스 블록 클릭 - 좌표 표시
- 2) 소수점은 연산블럭을 이용하여 자연수로 표시 가능



135



Build Your Own Blocks

## D. Predicates & Conditional Evaluation



두개의 블록으로 만드는 '놀라운' 마우스 따라가기

true & false



조건에 따른 반복 동작

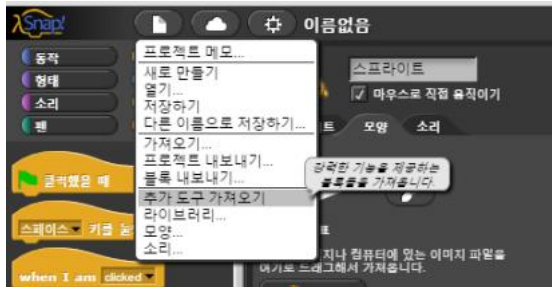


까먹지 않기 위한 조치



X좌표( $x > 0$ )에 따라 왼쪽과 오른쪽 구분

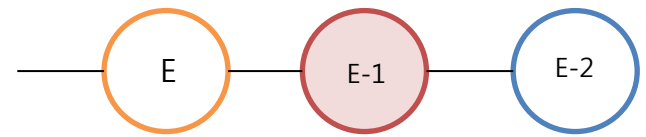
## E. Variables



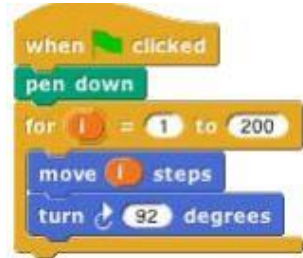
추가도구 가져오기  
\*For문



- 1) "Make a variable"
- 2) name입력 후 팔레트에 변수 확인 체크



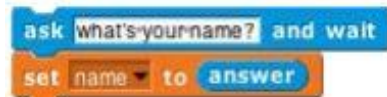
## E-1. For문



- 1) i변수 1~200 반복
- 2) i변수를 마우스로 드레그 한 후 move블럭에 넣는다.



name

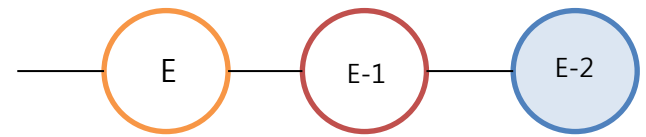


<변수활용 예시> 이름 알아내기

- 1) 이름 묻고 기다리기
- 2) 입력창에 이름 입력하기



Build Your Own Blocks

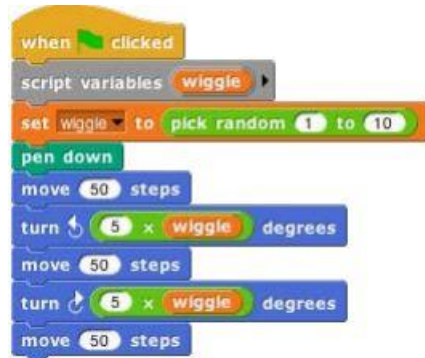


## E-2. Script Variables



스크립트 변수

- 1) 스크립트 변수 블록
- 2) 'score' 예제파일
- 3) 낙서



Build Your Own Blocks

# Snap! Manual:

Version 3.1, 4.0

---

## Build Your Own Blocks

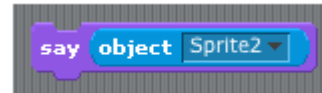


Build Your Own Blocks

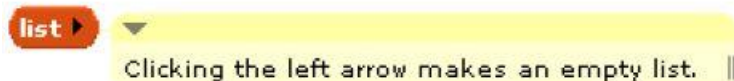
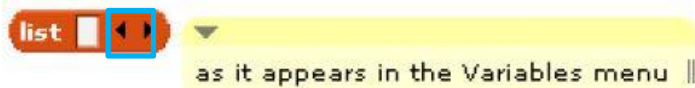
## First Class Lists : 2장



all data should be 'first class'



\*First Class Sprites



리스트 종류

- 1) 리스트 왼쪽, 오른쪽 화살표 클릭
- 2) 리스트 추가 및 삭제
- 3) lists of lists



Build Your Own Blocks



## A. The list block

### 1) 변수 말하기

say list she loves you <>

1 she  
2 loves  
3 you  
+ length: 3



### 2) 변수의 길이

length of list she loves you <>

3

### 3) 블록 만들기

vowel? letter

report list a e i o u contains letter

### 4) 리스트 만들기

item 3 of list Good day sunshine <>

item 3 of list Good day sunshine <>

```
for( i = 0; i<mylist.length; i++ ) {  
    if( mylist[i].checked ) {  
        total += parseInt(mylist[i].value);  
    }  
}
```



Build Your Own Blocks

## B. Lists of lists

list  
list John Lennon ▶ list Paul McCartney ◀ list George Harrison ▶ list Ringo Starr ◀ ▶

ad hoc structures

list  
list small medium large ◀ list chocolate rum raisin pumpkin lychee ◀ list cone cup ▶ ▶

binary tree, datum datum , left child left , right child right  
report list \*binary-tree\* datum left right ◀ ▶

bt-datum tree  
if is tree a list ?  
if item 1 of tree = \*binary-tree\*  
report item 2 of tree  
else  
say join words tree as text isn't a binary tree. ◀ ▶  
stop script  
else  
say join words tree isn't a binary tree. ◀ ▶  
stop script

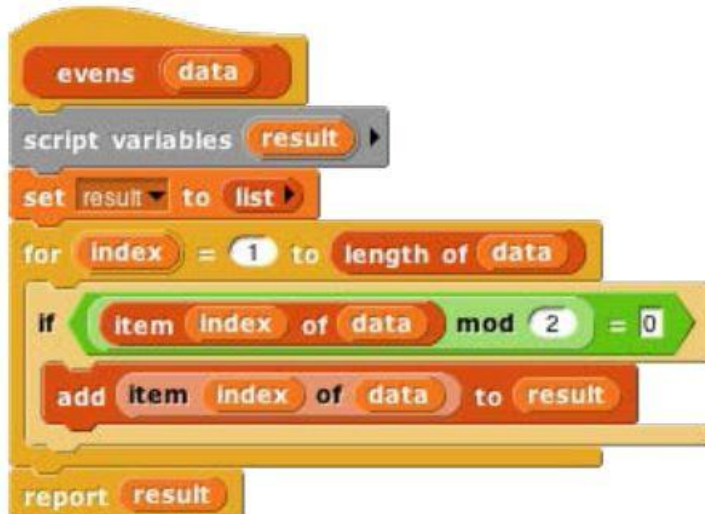
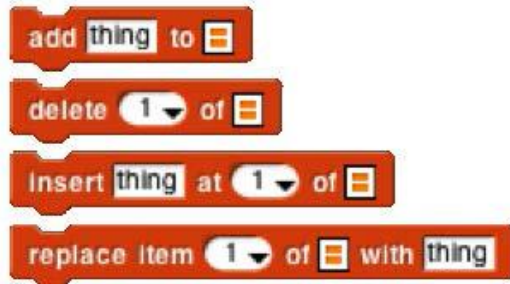
computer science data structure  
예제파일<http://>



Build Your Own Blocks

## C. Functional and Imperative List Programming

---



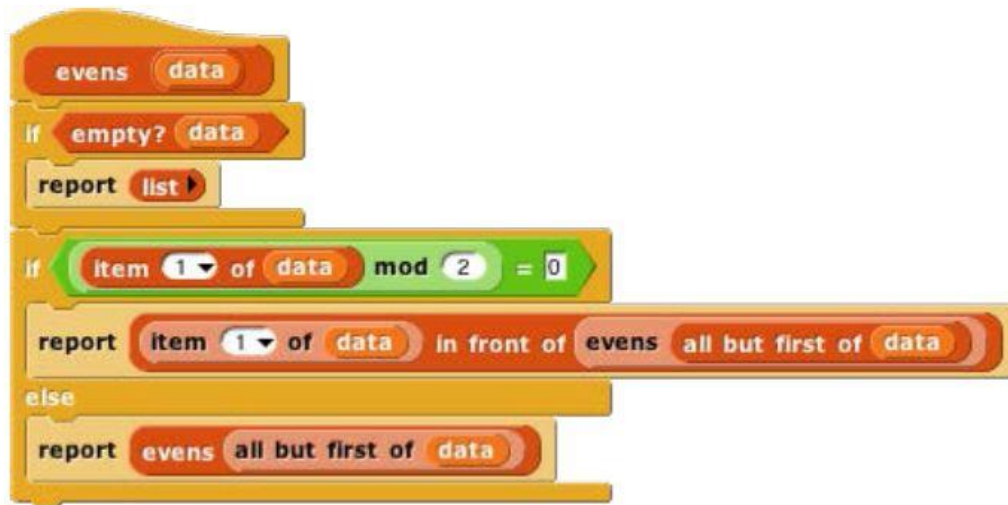
item ( ) of [ ] block  
예제파일 <http://>



Build Your Own Blocks

## C. Functional and Imperative List Programming

---



item ( ) of [ ] block  
예제파일 <http://>



Build Your Own Blocks

## D. Higher Order List Operations and Rings

map over

keep items such that from

combine with items of

Map, Keep, Combine  
예제파일 <http://>

# map the letter 1 of block over  
list magical mystery tour

1 magical  
2 mystery  
3 tour  
+ length: 3



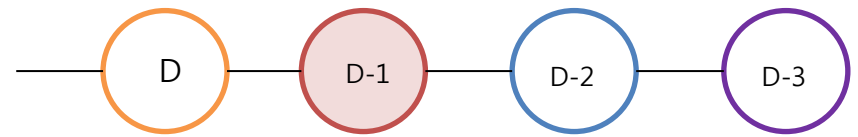
1 m  
2 m  
3 t  
+ length: 3

1 love cry  
2 me baby  
3 do cry  
+ length: 3

# map join words over  
list love me do list cry baby cry



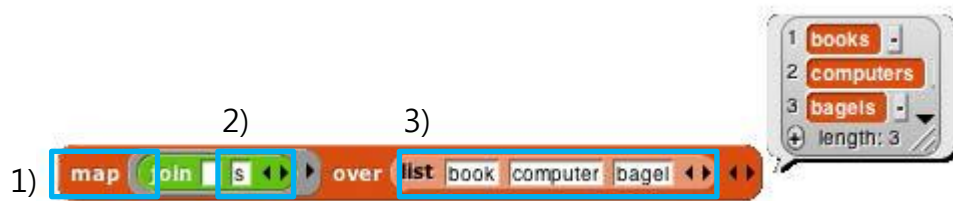
Build Your Own Blocks



## D-1. Map



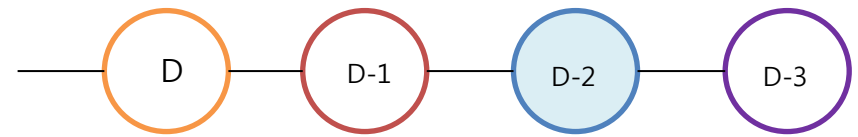
"Import tools..."



- 1) 영어의 규칙동사와 불규칙동사
- 2) + ed
- 3) "대표동사" 삽입



Build Your Own Blocks



## D-2. Keep

2로 나눈 나머지가 0인 수 구하기



길이가 3보다 큰 문자 찾기

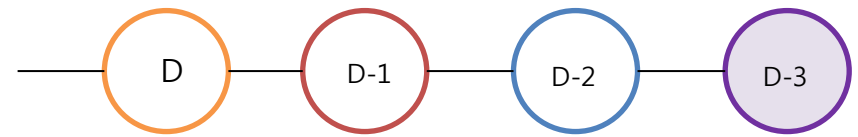


1) keep items such that from

2)  $2 = 3$  false

3)  $2 = 3$

- 1) Keep 블록 (링 from 리스트)
- 2) T/F
- 3) 링(ring)

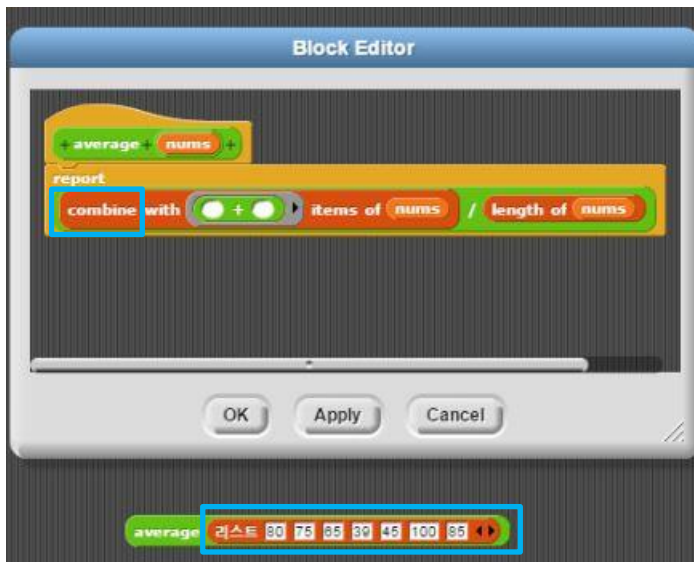


## D-3. combine



$(2+3)+4 = 2+(3+4)$ , but  $(2-3)-4 \neq 2-(3-4)$

1)



2)

평균값 구하기

- 1) combine 블록
- 2) 덧셈블록
- 3) lists of lists 응용하기



## D. Higher Order List Operations and Rings

---

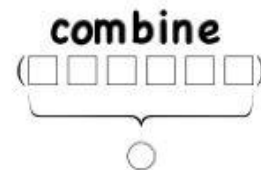
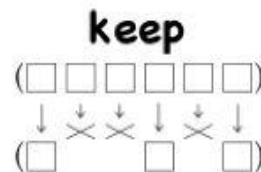
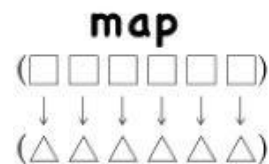
map  over 

keep items such that  from 

combine with  items of 

computer science data structure

예제파일 <http://>



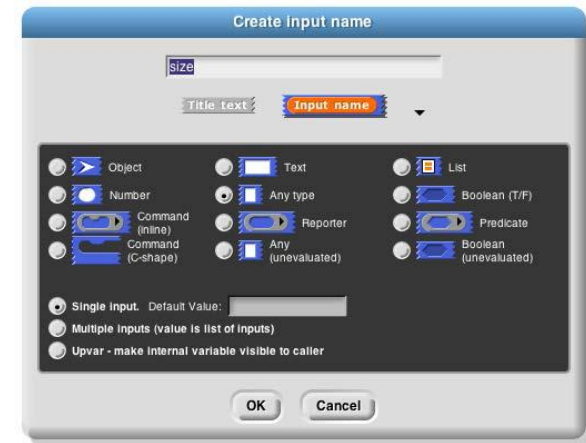
Build Your Own Blocks

## Typed Inputs : 3장

### A. Input Type Dialog



\*화살표 클릭



**Snap! non-procedure types**  
**Types that exist in Scratch**  
**Snap! procedure types**  
**Snap! unevaluated procedure types**



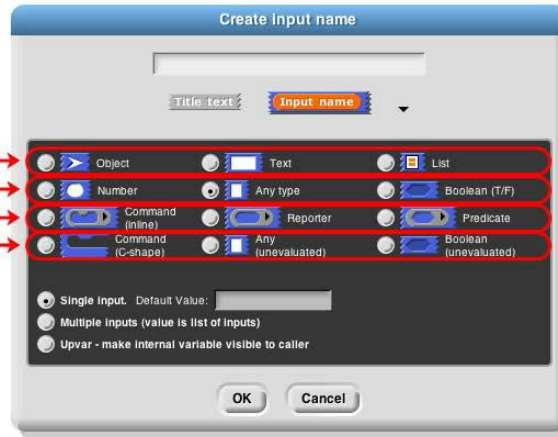
- 1) 년(non)프로시저타입 (루틴, 함수 X)
- 2) 스크래치 타입
- 3) 프로시저 타입
- 4) 결과값이 없는 프로시저 타입



Build Your Own Blocks

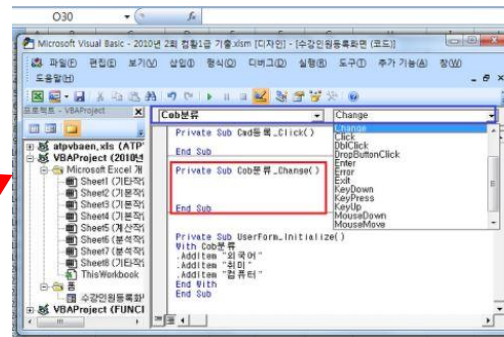
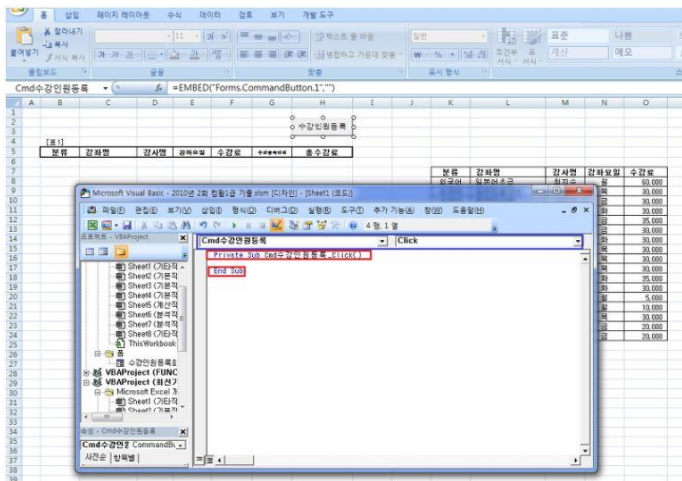
## A. Input Type Dialog

*Snap! non-procedure types*  
*Types that exist in Scratch*  
*Snap! procedure types*  
*Snap! unevaluated procedure types*



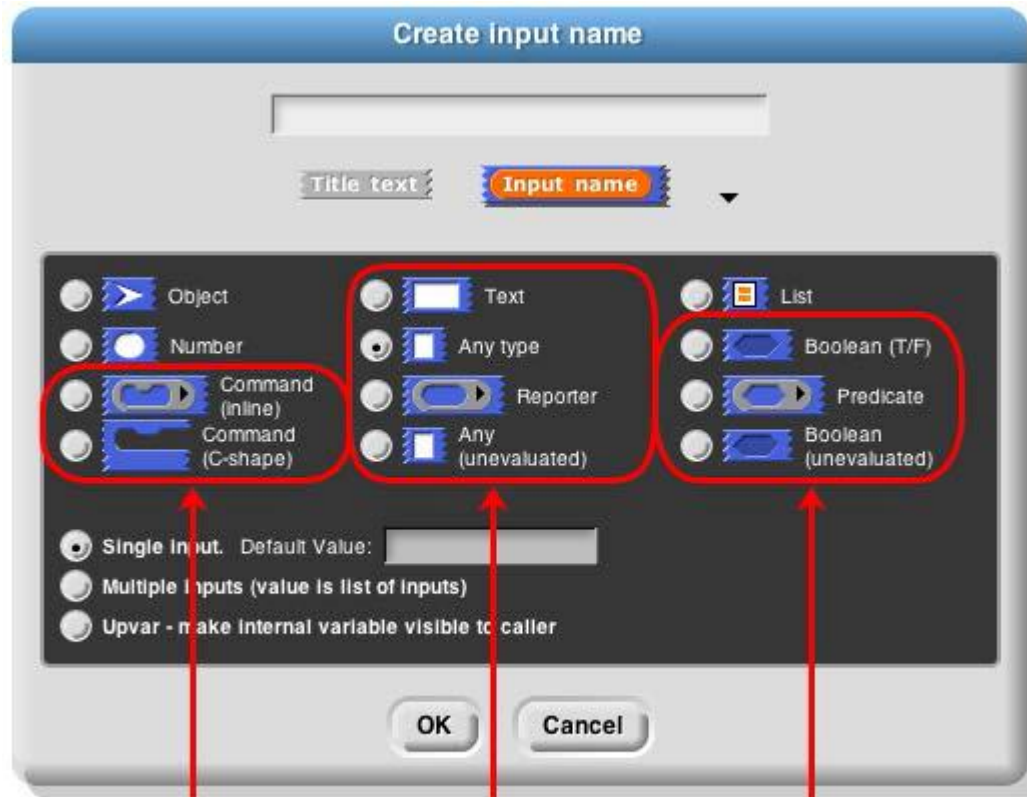
- 1) 프로시저 = 언어프로그램
- 2) 스크래치 타입
- 3) command – inline, C-shape, \*G-shape

\*12가지 Input Type: 오브젝트, 텍스트, 리스트 + 스크래치타입 + 프로시저 타입



\*엑셀 프로시저

## B. Procedure Types



**Commands**

1) 명령

**Reporters**

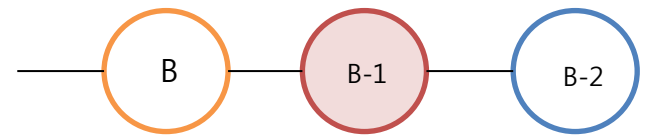
2) 출력

**Predicates**

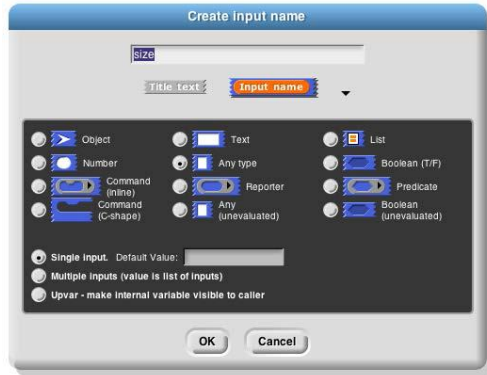
3) T/F



Build Your Own Blocks



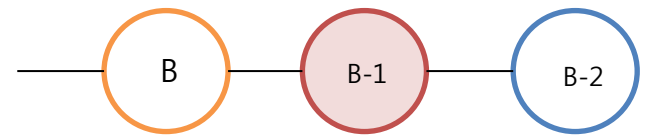
## B-1. Prototype Hints



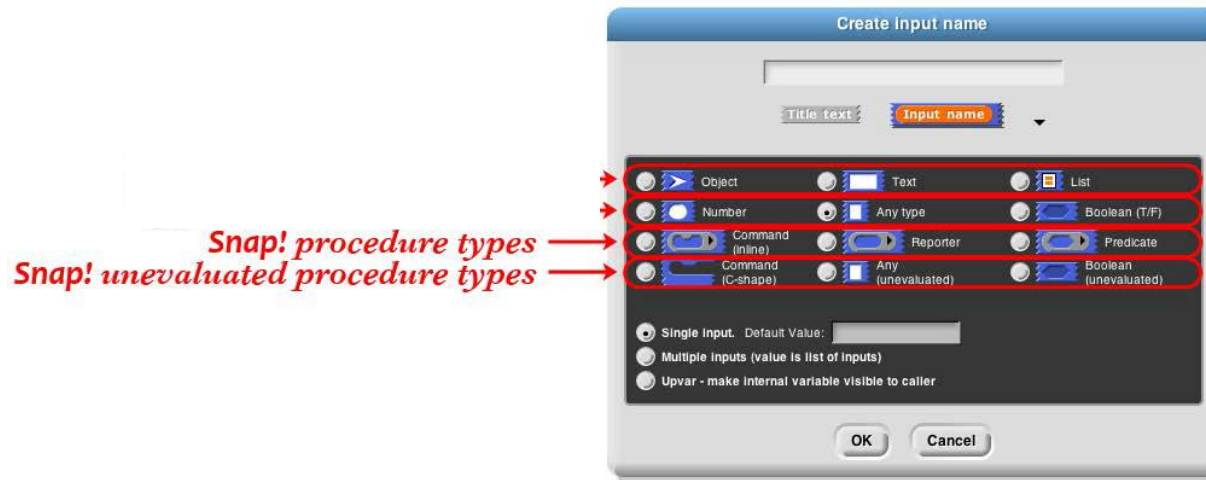
= default value  
 $\lambda$  procedure types  
 ... multiple inputs  
 : list  
 $\uparrow$  upvar  
 ? Boolean



\*Single input = default value  
 Multiple input = list  
 Upvar = variable



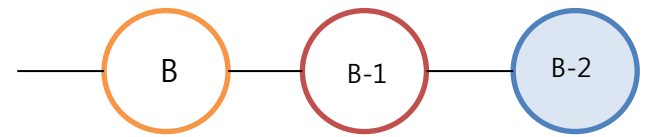
## B-1. Prototype Hints



= default value    ... multiple inputs    ↑ upvar  
 $\lambda$  procedure types    ⋮ list    ? Boolean



Build Your Own Blocks



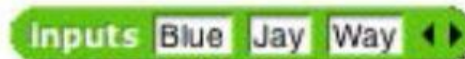
## B-2. Input variants

---

1) Single input



2) Multiple input



3) Upvar



Build Your Own Blocks

# Snap! Manual:

Version 3.1, 4.0

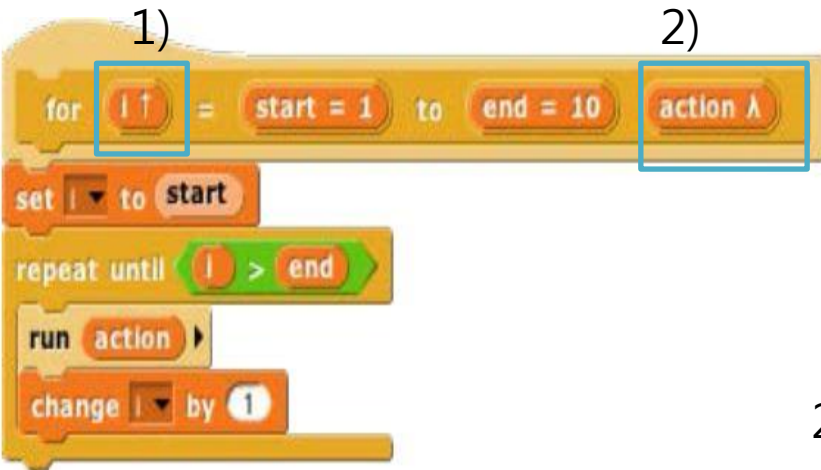
---

## Build Your Own Blocks



Build Your Own Blocks

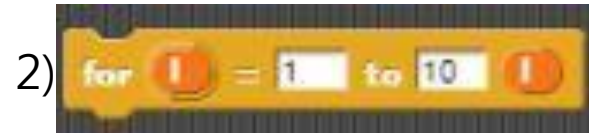
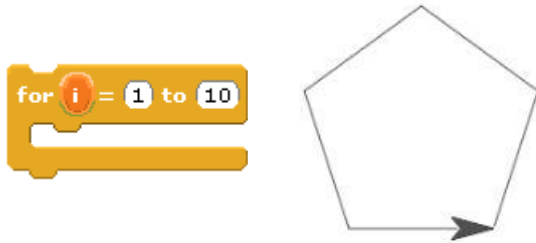




## "Command (C-shaped)"



## A. Call and Run



- 1) 변수 i를 C-shape에 넣기
- 2) 변수 i를 ring으로 바꾸기
- \*Command C-shape변수 설정
- 3) ring속에 동작넣기
- 4) 오각형 만들기

## A. Call and Run

Call a reporter or predicate block.

The call block has a ring for its first input slot because it expects a reporter or predicate as its input. Almost always, though, you'll use a variable to provide the input value, as in the example above. When you put a variable in a ringed input slot, the ring disappears because it's the *value* of the variable that gives the function to call. If you drag anything but a variable into the ring, the ring remains around the input:

The input variable function has the reporter to call.

The input variable function has the reporter to call.

Reporter-type input  
예제파일: <http://>

```
map function over list
script variables result i
set result to list
set i to 0
repeat length of list
  change i by 1
  add call function with inputs item i of list to result
report result
```

map  $\times 10$  over list 7 8 1

1	70
2	80
3	10
+ length: 3	

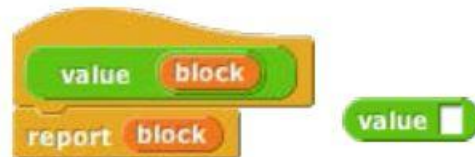
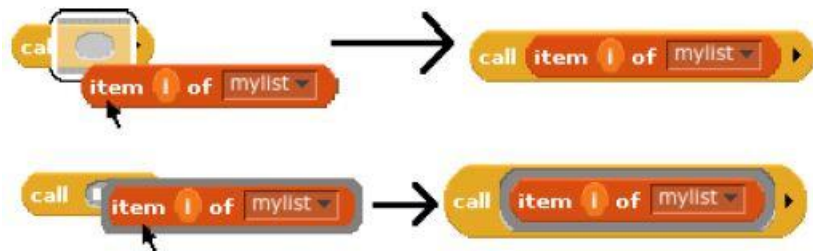
## A. Call and Run

참고: 한글 블록



## A. Call and Run

Call.Run with inputs



Build Your Own Blocks



## A. Call and Run

Call.Run with inputs



Call.Run with inputs  
Input names(매개변수)



Build Your Own Blocks

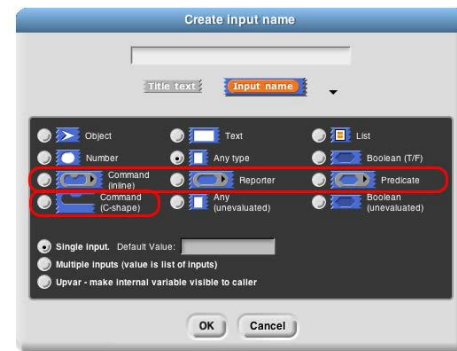
## B. Wirting Higher Order Procedure

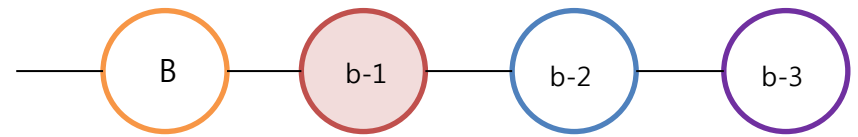
1)



2)

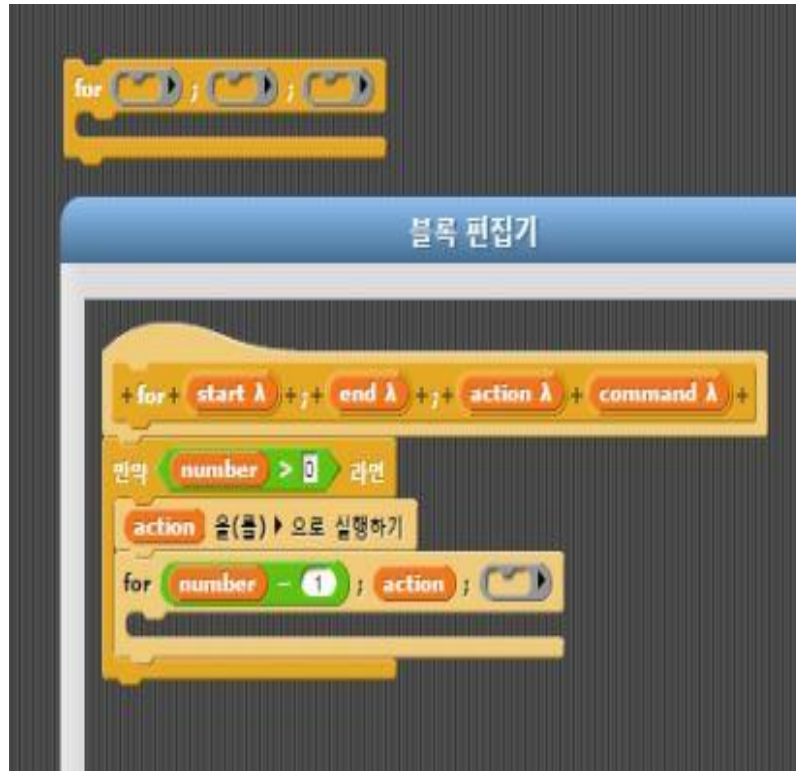
"Command (C-shaped)"



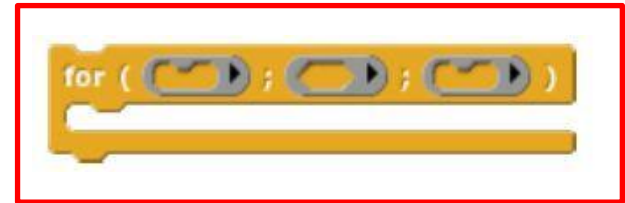


## B-1. C/C++/Java **for loop**

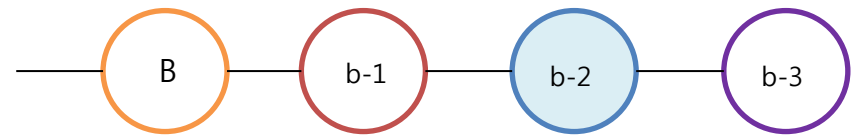
1)



2)







## B-2. Empty Input Slot

---

1) 변수 2개 차례대로

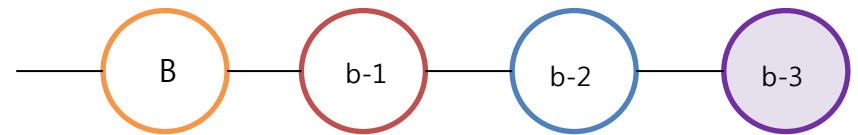


2) 변수 1개 모두



3) 이외에 빈곳은 입력되지 않음

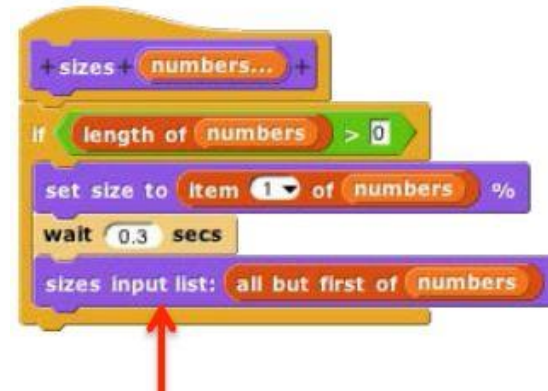
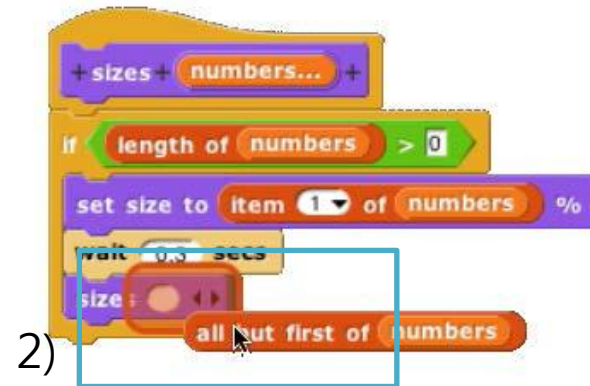
## B-3. Recursive Calls to Multiple-Input Blocks



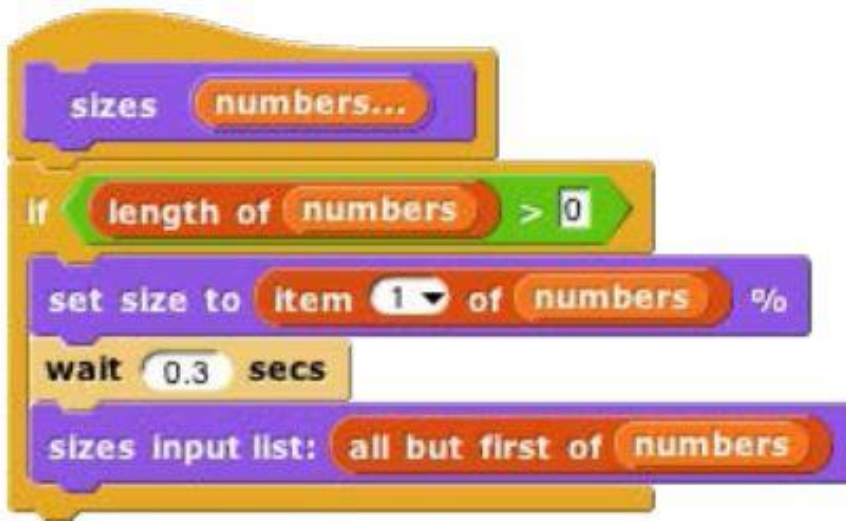
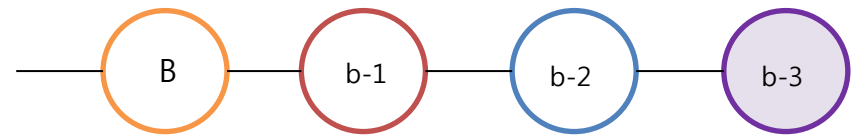
1)



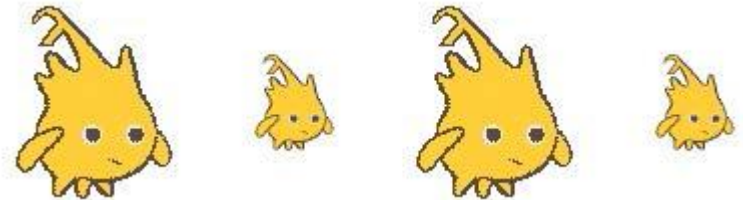
"input list:"



### B-3. Recursive Calls to Multiple-Input Blocks



"input list:"



Build Your Own Blocks

## C. Formal Parameters(매개 변수)

"#1, #2" 디폴트값

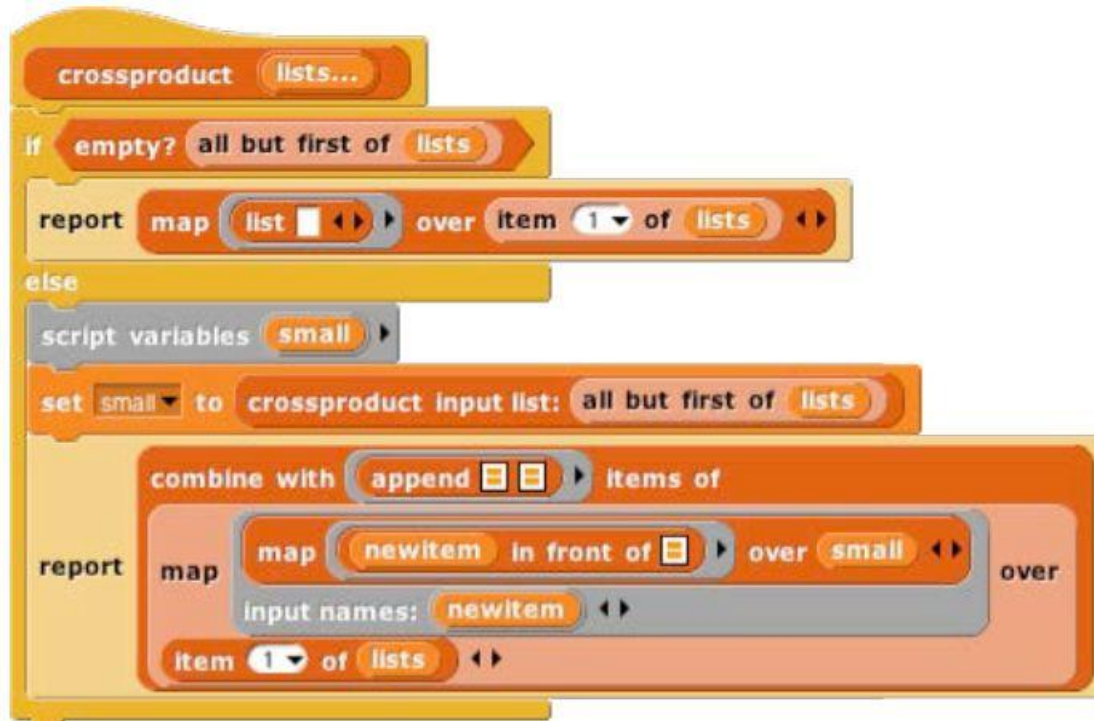


#1, #2에 x, y 입력

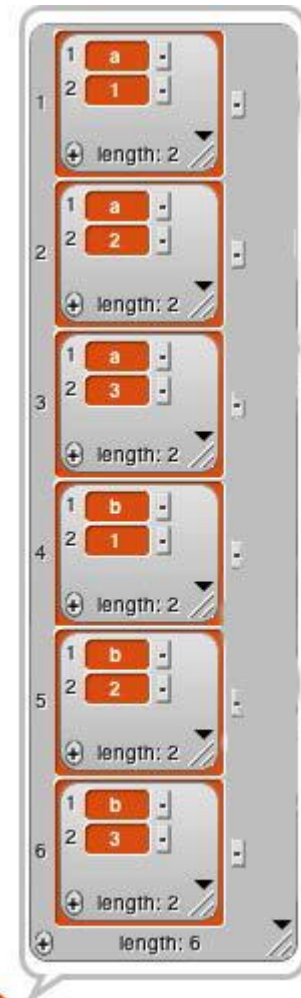


Build Your Own Blocks

## C. Formal Parameters(매개 변수)



crossproduct list a b list 1 2 3



Build Your Own Blocks

## C. Formal Parameters(매개 변수)

crossproduct list a b list 1 2 3



Build Your Own Blocks



## D. Procedures as Data

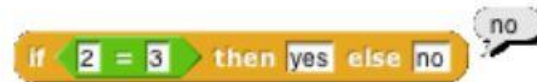
---

list block: 연산과 프로시저

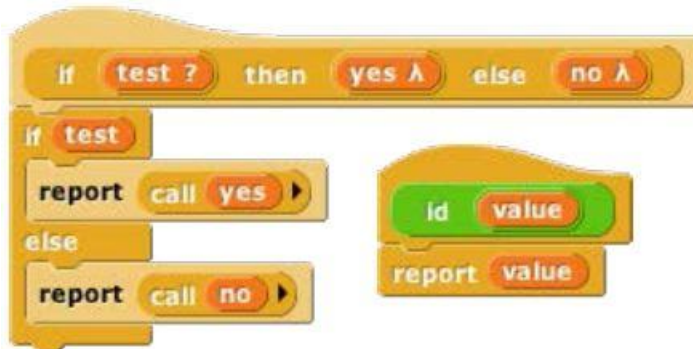


Build Your Own Blocks

## E. Special Forms

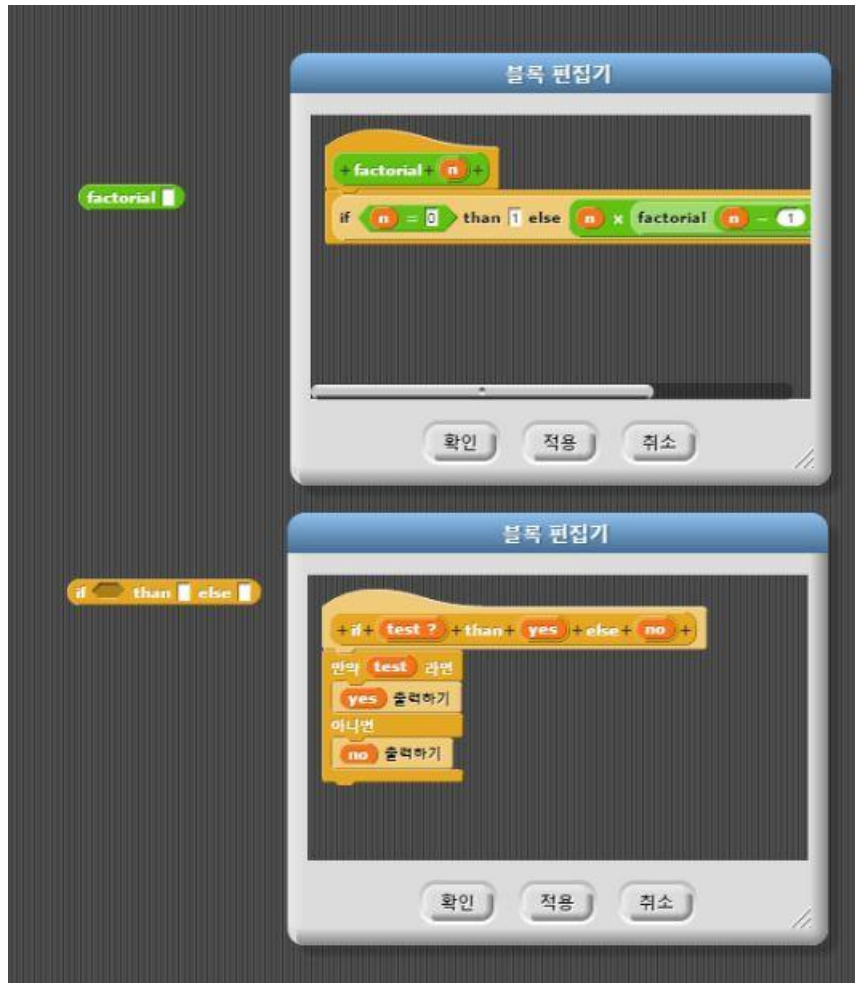


Simple; more and more





## E. Special Forms



**report** Report a value from a custom block to its caller.

absolute value of num

absolute value of -7.5

7.5

If this **report** block runs because the number is negative, then that ends the custom block, and so this **report** block (or whatever else comes later in the block's script) doesn't happen.

**report** works for predicate custom blocks too; just drag a hexagonal true/false reporter into its input slot.

empty? data

report data = list



Build Your Own Blocks

## E. Special Forms

---

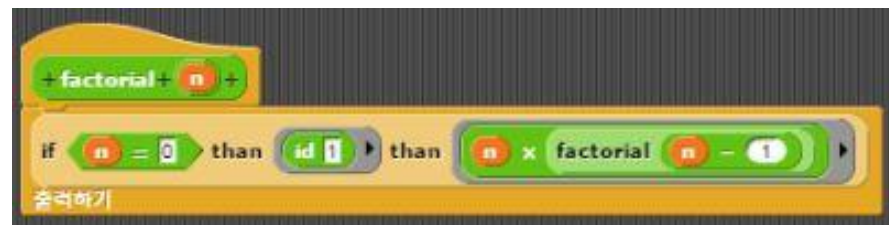
1)



2)



3)



Build Your Own Blocks

# Snap! Manual:

Version 3.1, 4.0

---

## Build Your Own Blocks



Build Your Own Blocks



## A. Local State with Script Variables

### Pulldown input 만들기

예제파일<http://>



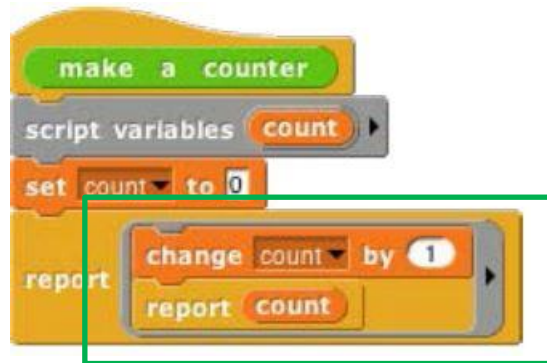
Build Your Own Blocks



## A. Local State with Script Variables

make a counter block 만들기

예제파일<http://>



Build Your Own Blocks

## B. Messages and Dispatch Procedures

---

*dispatch procedure*

예제파일 <http://>

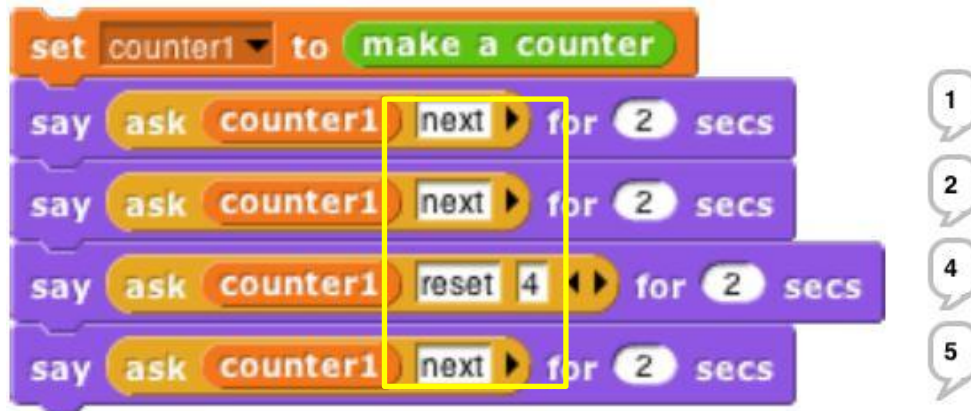
- 1) next
- 2) reset
- 3) 매개변수 message



## B. Messages and Dispatch Procedures

*dispatch procedure (ring형태가 아님)*

예제파일 <http://>



- 1) next 1씩 누적
- 2) reset 4부터 시작
- 3) 5, 6

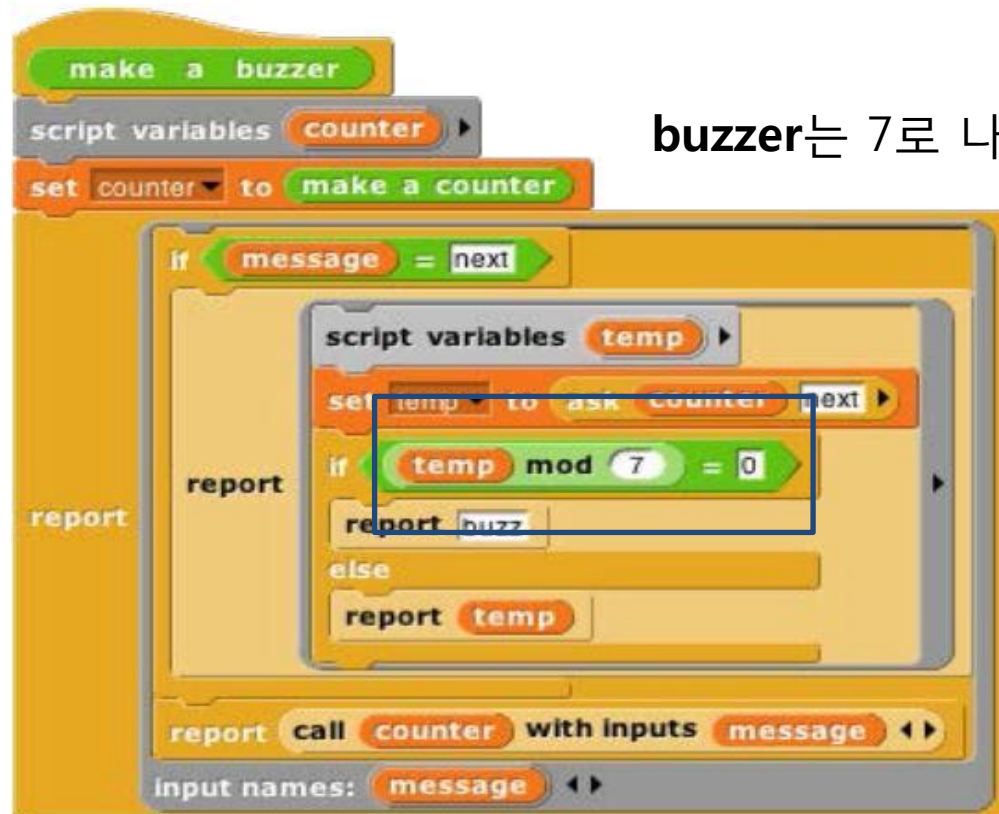


Build Your Own Blocks



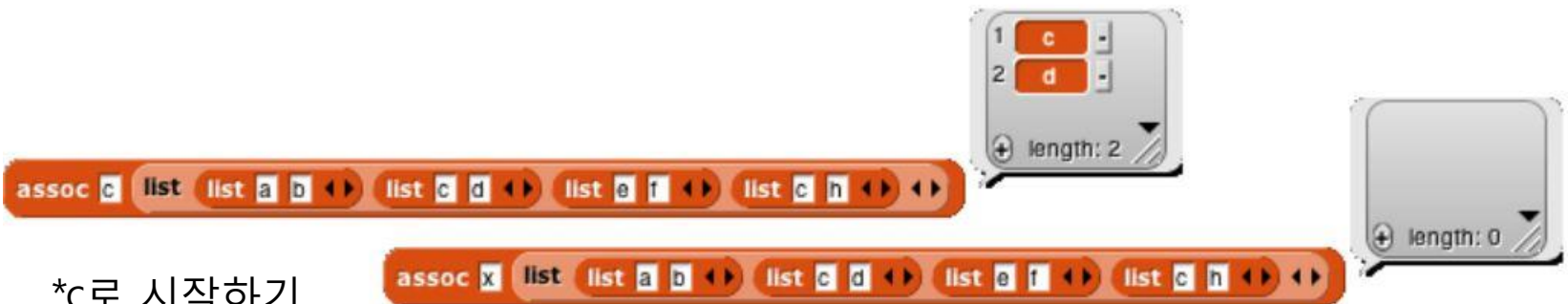
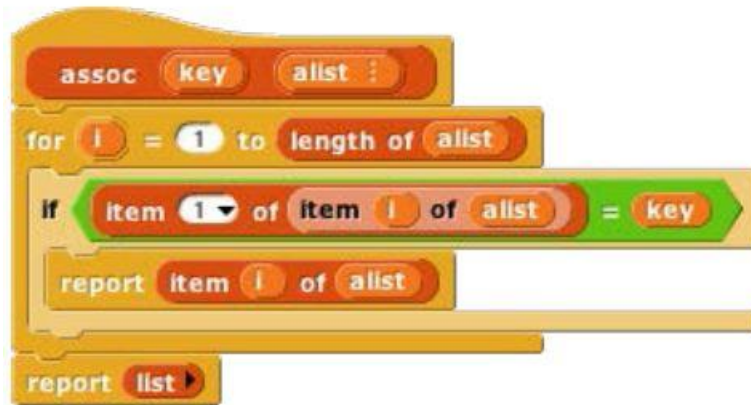
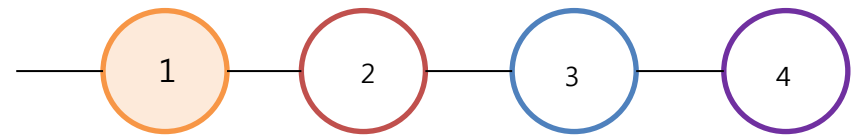
## C. Inheritance via Delegation

---



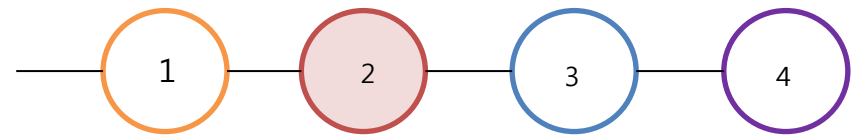
buzzer는 7로 나누어 떨어질때 **reset**

## D. An Implementation of Prototyping OOP

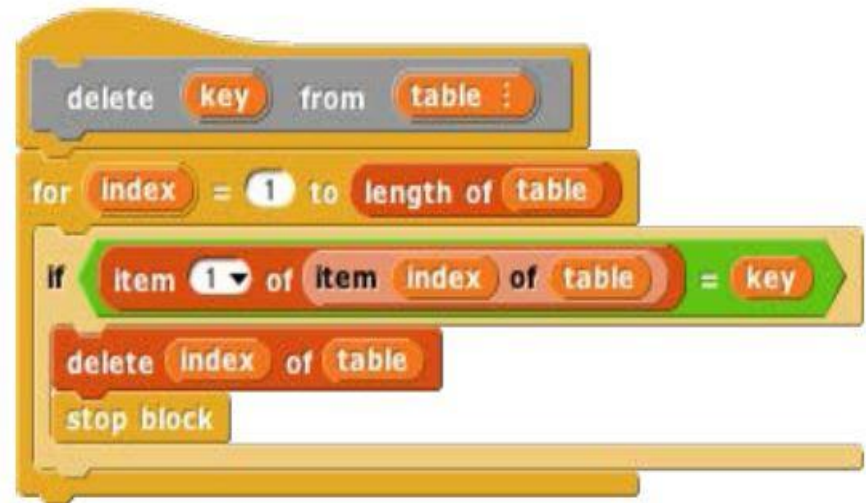
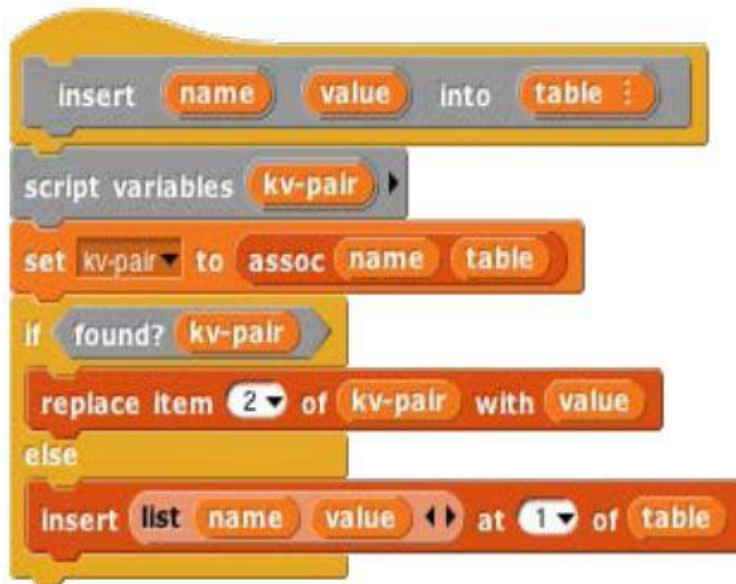


\*c로 시작하기  
x로 시작하기

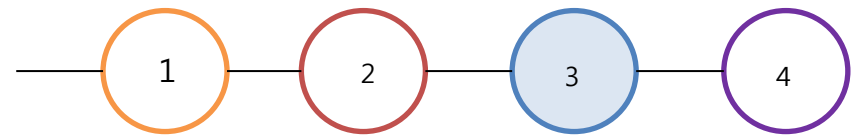
## D. An Implementation of Prototyping OOP



\*테이블 만들기(삽입, 삭제)



## D. An Implementation of Prototyping OOP

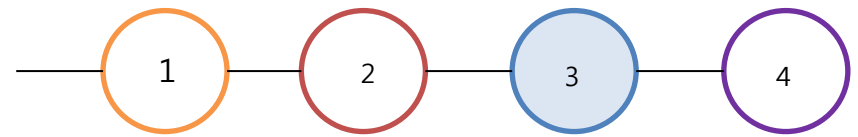


\*로봇 프로토타입 OOP  
전진, 후진  
센서(거리, IR)

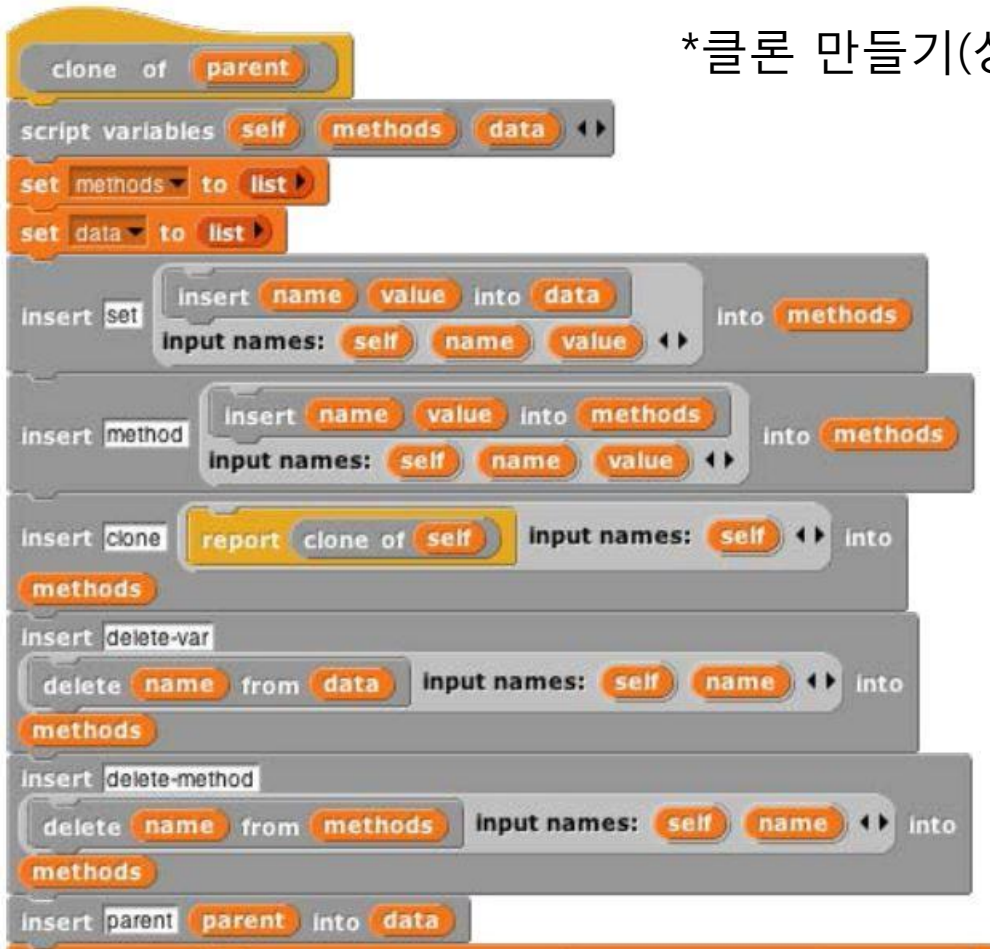
파이썬 예제파일  
<http://>  
<http://>



Build Your Own Blocks



## D. An Implementation of Prototyping OOP

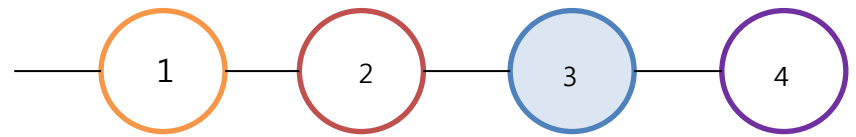


\*클론 만들기(상속개념)



Build Your Own Blocks





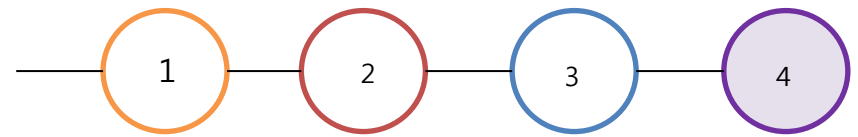
## D. An Implementation of Prototyping OOP

---

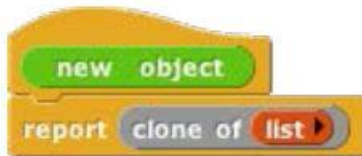


## D. An Implementation of Prototyping OOP

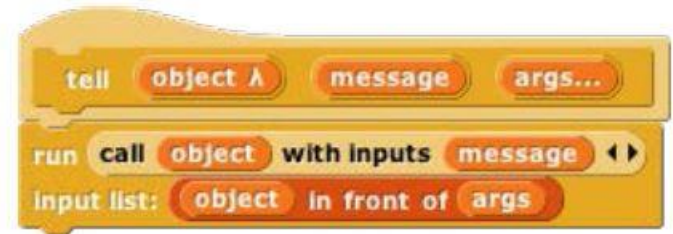
---



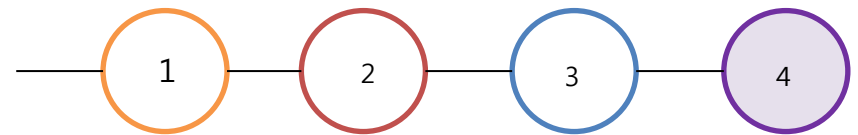
\*새로운 객체(클론 리스트)



dispatch procedure  
Ask & Tell

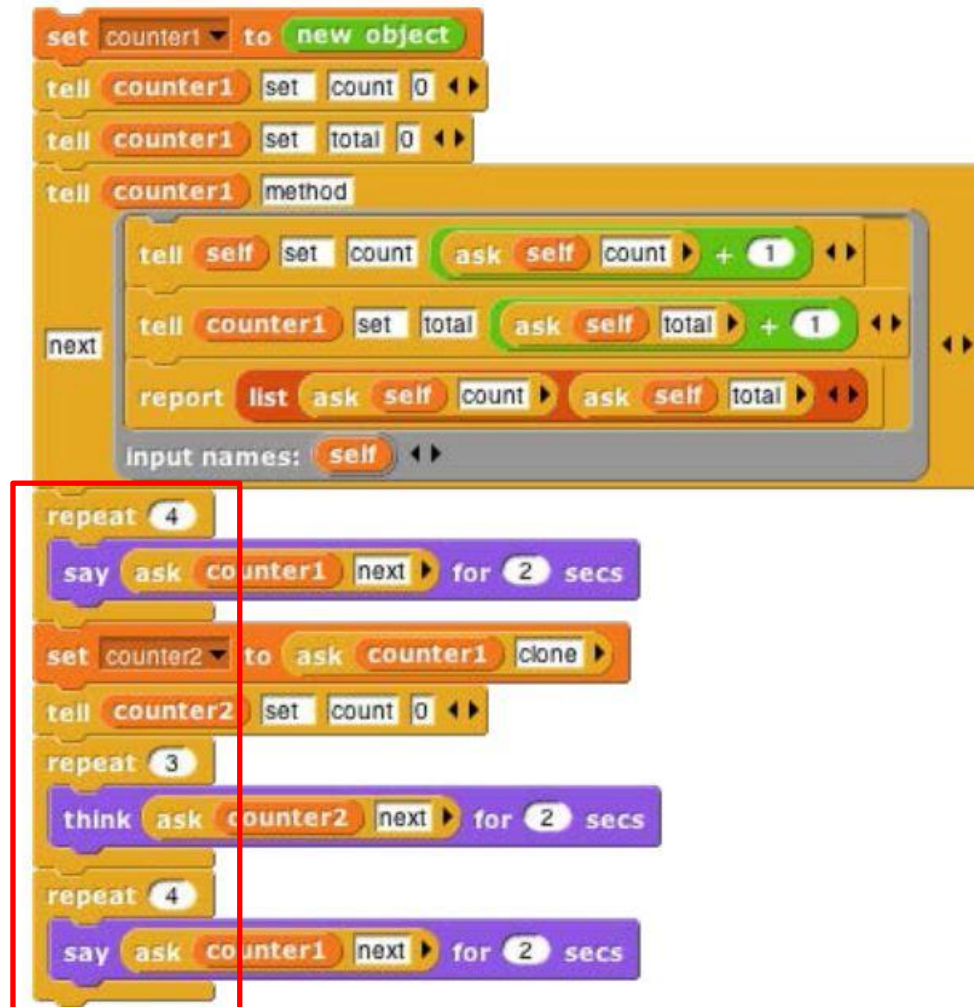


## D. An Implementation of Prototyping OOP



[1 1] [2 2] [3 3] [4 4] (1 5) (2 6) (3 7) [5 8] [6 9] [7 10] [8 11]

\*결과값



Build Your Own Blocks