

02 가상현실에 코딩 접목하기

학습 목표

- 로직 명령어를 활용하여 동적으로 3D 객체를 생성할 수 있다.
- 반복문을 활용하여 여러 개의 객체를 동시에 생성할 수 있다.

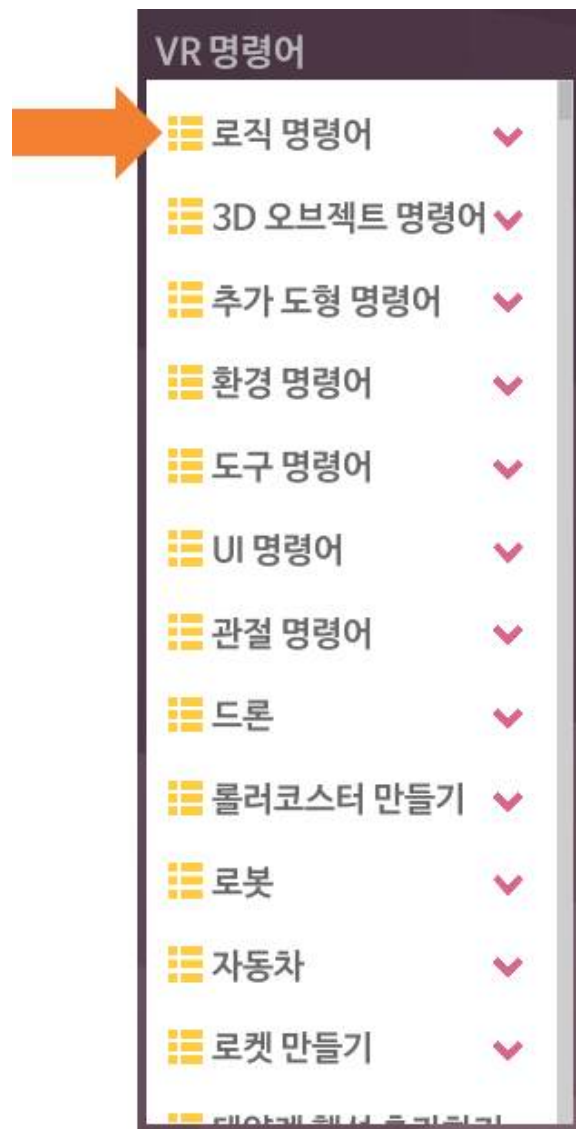
실습 개요

- 반복문과 문자출력 명령어를 이용하여 구구단을 출력해 본다.
- 랜덤 함수를 활용하여 임의의 값을 생성해 본다.
- 함수를 직접 생성해 본다.
- 조이스틱 기능을 활용하여 콘텐츠를 제어해 본다.
- 도미노 시나리오를 이용하여 나만의 재미있는 콘텐츠를 창작해 본다.

2.1 코딩 로직 명령어의 종류

로직 명령어

- VR 코딩 편집기에서 왼쪽 위쪽에는 로직 구현에 필요한 명령어들이 있다. 아래는 로직 명령어들의 종류를 보여준다.



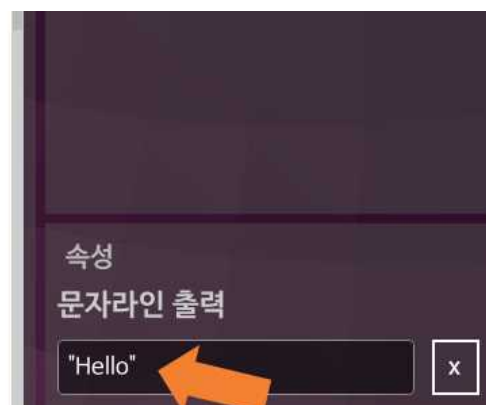
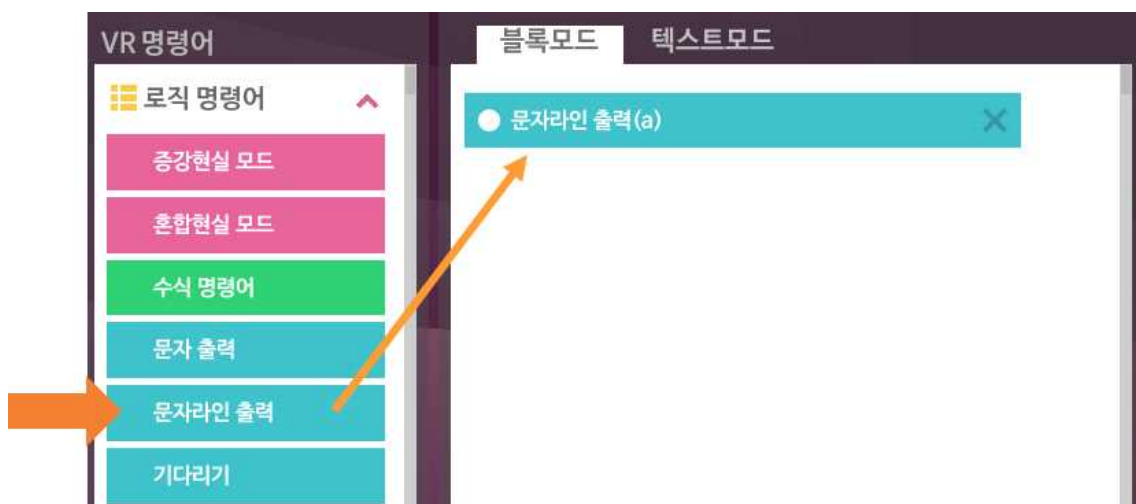
증강현실 모드	for
혼합현실 모드	while
수식 명령어	함수
문자 출력	Setup 함수
문자라인 출력	Loop 함수
기다리기	함수 시작하기
if	함수 반복하기
else	스크립트 불러오기
else if	스크립트 파일 실행하기

- : 증강현실모드 - 카메라 영상을 배경영상으로 처리한다.
- : 혼합현실모드 - 카메라 영상을 입체 배경영상으로 처리한다.
- : 수식명령어 - 변수 선언 및 일반적인 프로그램 수식을 입력한다.
- : 문자라인출력 - 문자열을 한 줄씩 출력한다. 출력한 후 줄을 바꾼다.
- : if - 조건을 비교한다.
- : for - 반복문을 실행한다.
- : while - 반복문을 실행한다.
- : 함수 - 함수를 추가한다.

2.2 문자라인출력과 문자출력 명령어

문자라인 출력

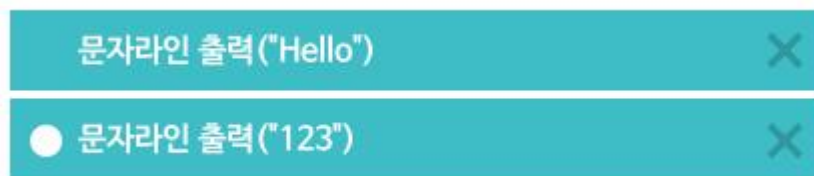
- 문자라인 출력 명령어는 한줄씩 값을 화면에 출력한다.
- 문자라인 출력 명령어를 추가한 후, 속성값에서 “Hello”를 입력해 본다.



- 코드를 실행하면 화면에 Hello 라는 단어가 표시되는 것을 볼 수 있다.

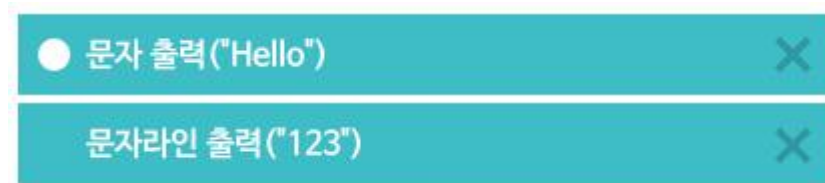


- 다음과 같이 문자라인 출력 명령어를 추가하여 실행하면 줄을 바꾸어서 값이 출력되는 것을 볼 수 있다.



문자 출력

- 문자 출력 명령어는 문자라인 출력 명령어와 달리 문자열을 출력한 후, 줄을 바꾸지 않는다. 즉 옆으로 이어서 계속 값이 출력된다.



실습

- ▶ 화면에 “나의 이름은 홍길동 입니다”를 출력해 본다.

2.3 변수와 자료형

변수와 수식명령어

- 프로그램에서 변수는 값을 저장하는 역할을 수행한다.
- 변수는 수식 명령어를 이용하여 선언하며, 값을 읽어 오거나 변경할 수 있다.



- 아래의 예는 수식명령어를 이용하여 변수를 선언한 후에 값을 출력해 본 예를 보여준다.



다양한 자료형의 변수

- 변수에 저장되는 값은 정수, 실수, 문자열, 논리값 등 다양한 형식의 값이 저장될 수 있다.



숫자 변수의 연산

- 정수나 실수 등 숫자 값을 저장하고 있는 변수들은 서로 더하거나 빼는 등 4칙 연산 등을 적용하여 값을 계산할 수 있다.

```
a = 10
b = 35
c = a * b
● 문자라인 출력(c)
```

350

문자열 변수의 연산

- 문자열 변수는 서로 더하는 연산만 가능하며, 각 문자열이 더해져서 하나의 문자열로 저장된다.

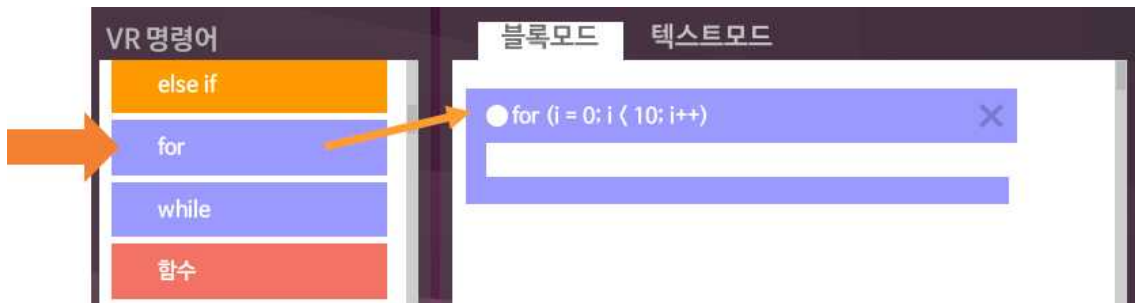
```
a = "12345"
b = ""
c = "Hello"
d = a + b + c
● 문자라인 출력(d)
```

12345 Hello

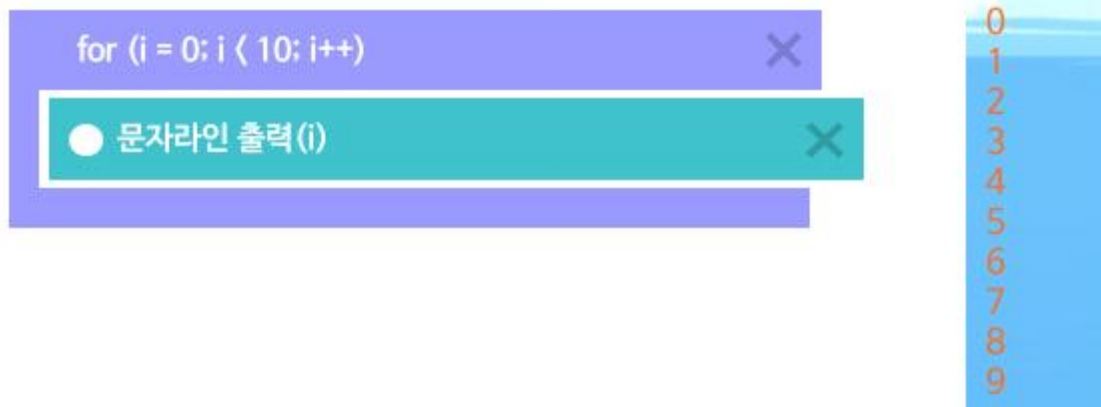
2.4 for 반복문

for 반복문

- for 반복문은 내부적으로 값이 자동으로 증가하는 변수를 포함하고 있다.
- 아래의 경우는 반복문 안에서 변수 i가 사용되었으며, 변수 i가 0부터 시작하여 9까지 1씩 증가하는 수식을 보여준다.



- 반복문 수식 안에 있는 변수의 값을 출력해 보면 다음과 같다.



실습

▶ 1부터 100까지의 합을 구하여 출력하는 프로그램을 작성해 본다.

```
s = 0
for (i = 1; i <= 100; i++)
    s = s + i
문자라인 출력(s)
```

5050

구구단 출력하기

■ for 반복문을 이용하여 구구단을 출력해 보다. 아래의 예는 구구단 7단을 출력하는 과정을 보여준다.

```
for (i = 1; i < 10; i++)
    문자 출력("7 x ")
    문자 출력(i)
    문자 출력(" = ")
    문자라인 출력(7 * i)
```

7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63

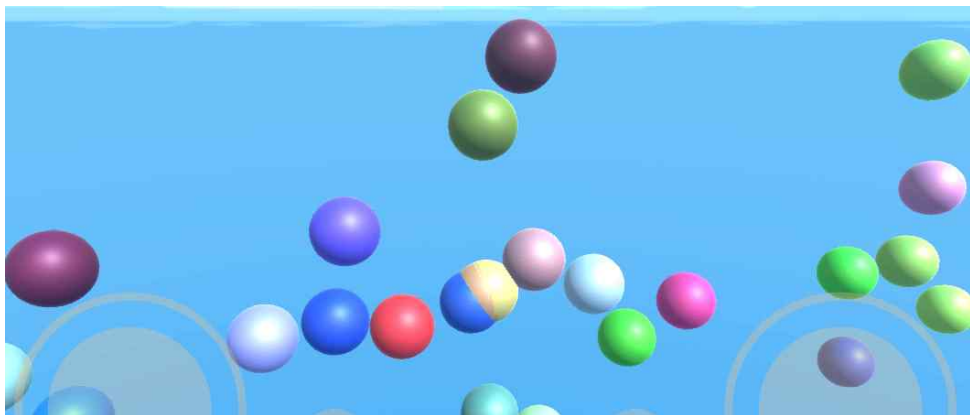
2.5 반복문으로 공 생성하기

while 반복문으로 공 생성하기

- while 반복문은 조건이 만족될 때 까지 횟수에 관계없이 계속 반복되는 것이 특징이다.
- 아래의 예는 무한히 반복하여 공을 생성하는 것을 보여준다.
- 공이 이름이 자동으로 겹치지 않고 생성되도록 공의 이름을 지운 것에 유의하기 바란다.

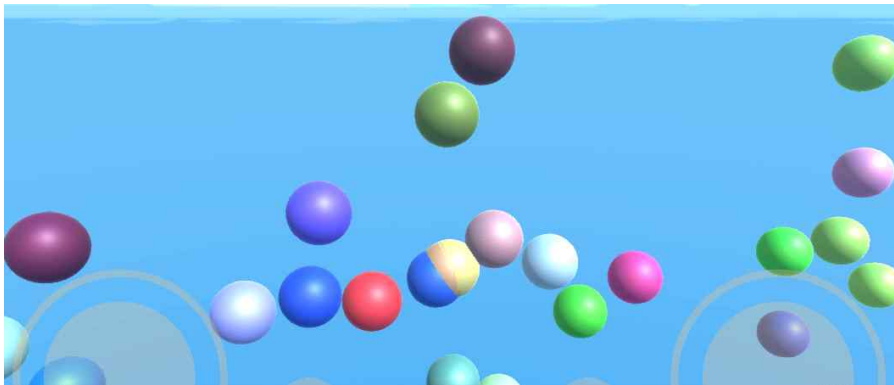
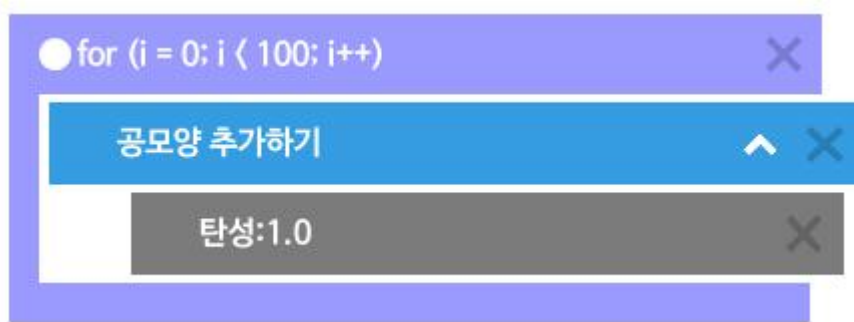


공의 이름을 지워 주어야 공의 이름이 자동으로 생성된다. 공의 이름이 겹치면 공이 생성되지 않는다.



for 반복문으로 공 생성하기

- for 반복문은 반복하는 횟수를 지정할 수 있다. 아래의 예는 공 100개를 반복하여 생성하는 과정을 보여준다.
- 공이 이름이 자동으로 겹치지 않고 생성되도록 공의 이름을 지운 것에 유의하기 바란다.



loop 함수로 반복하기

- 위의 for나 while 명령어는 해당 반복문이 끝날 때 까지 그 다음 명령어로 진행이 되지 않는 문제가 있다. 아래의 예는 loop 함수를 이용하는 방법으로서 무한 반복적인 작업이 필요할 경우에는 while 명령어 보다는 loop 함수를 이용하는 것이 훨씬 더 효율적이다.



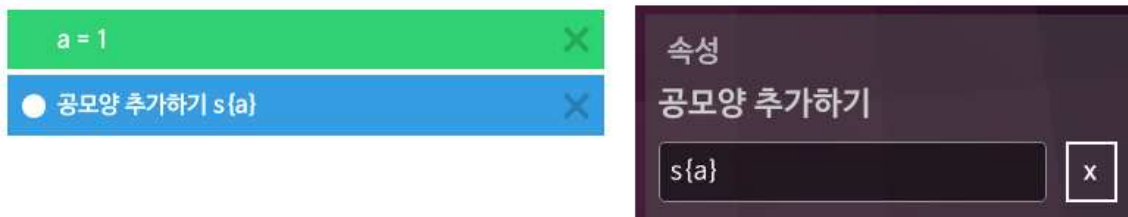
3D 오브젝트 명령어에서 변수 사용하기

- 3D 오브젝트 명령어는 일반적인 로직 명령어 방식이 아닌 명령어와 옵션 구조로 되어 있다.

```
공모양 추가하기 sphere1
  위치: 0, 5, 0
  크기: 1, 1, 1
```

3D 오브젝트 명령어

- 3D 오브젝트는 반드시 이름을 가져야 하며 서로 겹쳐서는 안된다. 단, 이름을 따로 지정하지 않는 경우에는 자동으로 겹치지 않도록 이름이 추가된다.
- 3D 오브젝트 이름과 옵션에는 문자열 값만 올 수 있다. 만약, 오브젝트 이름과 옵션에 변수를 사용하여 값을 자동으로 지정해 주려면 수식이나 변수를 { } 로 감싸서 표기해 주어야 한다.

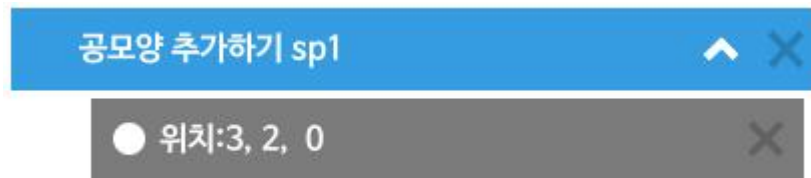


- 위의 코드를 실행하면 {a} 부분은 1로 대체가 되며, 따라서 공의 이름은 s1 이 되게 된다.

- 옵션에도 수식 또는 변수를 사용할 수 있으며, 다음의 코드를 실행하면 공의 위치나 크기 값이 변수에서 선언된 값으로 대체가 되는 것을 볼 수 있을 것이다.



- 위의 프로그램은 아래와 동일한 결과를 보여준다.



for 반복문으로 위치 지정하기

- 아래의 코드는 for 반복문에서 1씩 증가하는 변수를 생성되는 공의 위치에 적용한 사례를 보여준다. 위치 값 중 X축의 값이 {} 로 표기되어 있으며, 실제 실행시에는 각각 0, 1, 2 등의 값으로 대체가 된다.

for (i = 0; i < 100; i++)

공모양 추가하기

탄성:1.0

위치: {}, 0, 0

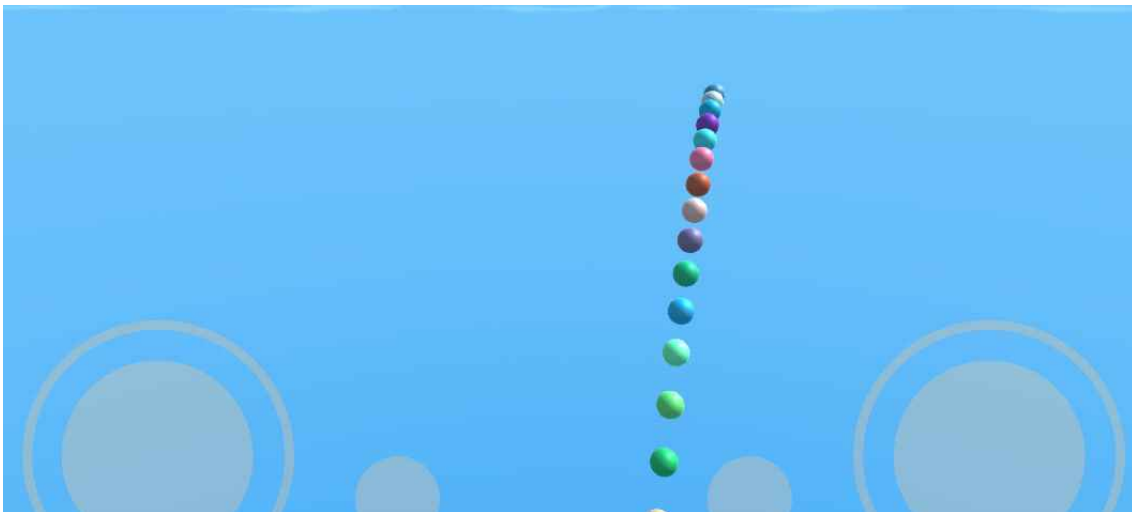
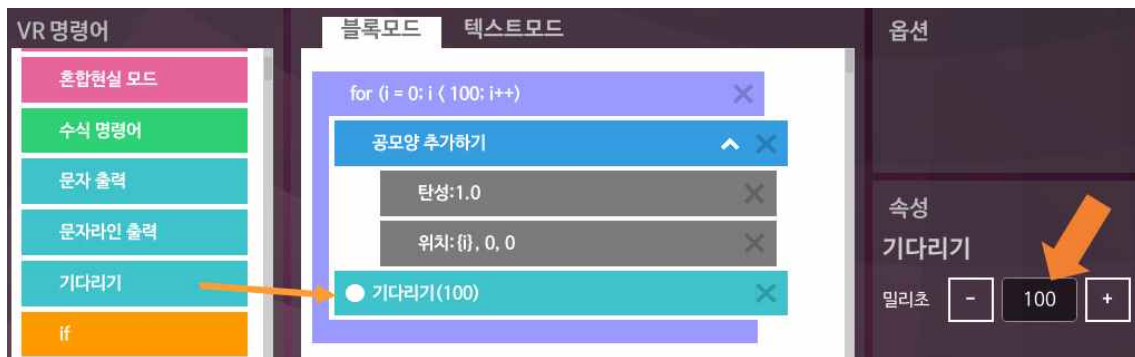
위치

X	-	{}	+
Y	-	0	+
Z	-	0	+



기다리기 명령어

- 로직 명령어에서 기다리기 명령어는 밀리초 단위로 실행을 지연시키는 기능을 수행하며, 해당 시간 만큼 기다렸다가 그 다음 명령어가 실행되도록 기능을 수행한다.
- 아래의 코드는 100개의 공을 한 번에 생성하지 않고 0.1초(100 밀리초)에 한 개씩 생성하는 과정을 보여준다. 기다리기 명령어를 추가한 후, 속성값에서 값을 1000에서 100으로 수정한다.



2.6 랜덤함수 활용하기

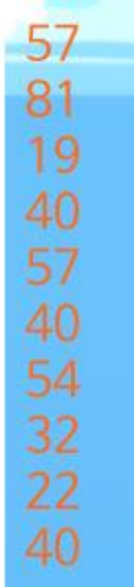
랜덤함수

- 랜덤함수는 임의의 값을 발생시키는 함수이다. 함수의 괄호 안에 숫자값을 입력하면 0 ~ 입력한 숫자값 사이의 값이 임의로 생성된다.
- 아래의 수식을 실행할 경우, 변수 a는 0에서 99사이의 임의의 숫자가 지정되며, 100은 발생하지 않는다.

```
a = random(100)
```

- 반복문을 활용하여 0 ~ 99사이의 임의의 숫자를 10개 생성시키는 예제는 다음과 같다.

```
for (i = 0; i < 10; i++)  
  ● a = random(100)  
  문자라인 출력(a)
```



57
81
19
40
57
40
54
32
22
40

랜덤함수로 위치 지정하기

- 랜덤함수를 활용하면 임의의 위치 값을 생성하여 해당 위치에 공을 생성시킬 수 있다.
- 랜덤함수의 괄호 안에는 인수로 시작값과 끝 값을 지정할 수 있는데, 만약 시작값을 지정하게 되면 임의의 값은 (시작값) ~ (끝값 보다 1 작은 값) 사이의 숫자가 임의로 생성된다. 시작값이 생략되면 시작값은 자동으로 0값이 된다.

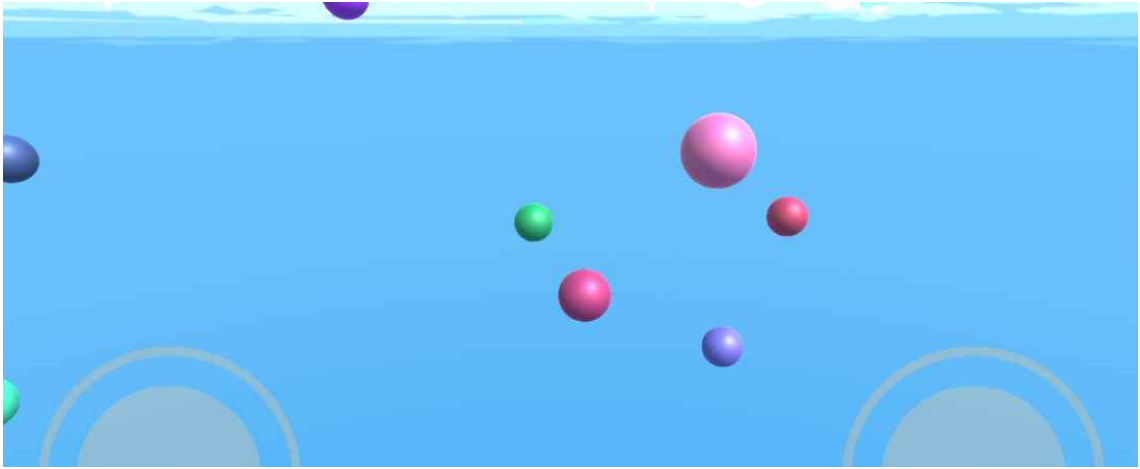
```
a = random(100)
b = random(0, 100)
c = random(50, 100)
d = random(-10, 10)
```

a에는 0 ~ 99 사이의 값이 저장된다.
b에는 0 ~ 99 사이의 값이 저장된다.
c에는 50 ~ 99 사이의 값이 저장된다.
d에는 -10 ~ 9 사이의 값이 저장된다.

- 아래의 예제는 X축, Y축, Z축에 대해 임의의 값을 생성한 후 공의 위치 옵션에 이 변수 값을 적용한 예를 보여준다.

```
● for (i = 0; i < 100; i++)  
  x = random(-10, 10)  
  y = random(0, 20)  
  z = random(-10, 10)  
  공모양 추가하기  
    위치: {x}, {y}, {z}  
    탄성: 1.0  
  기다리기(100)
```

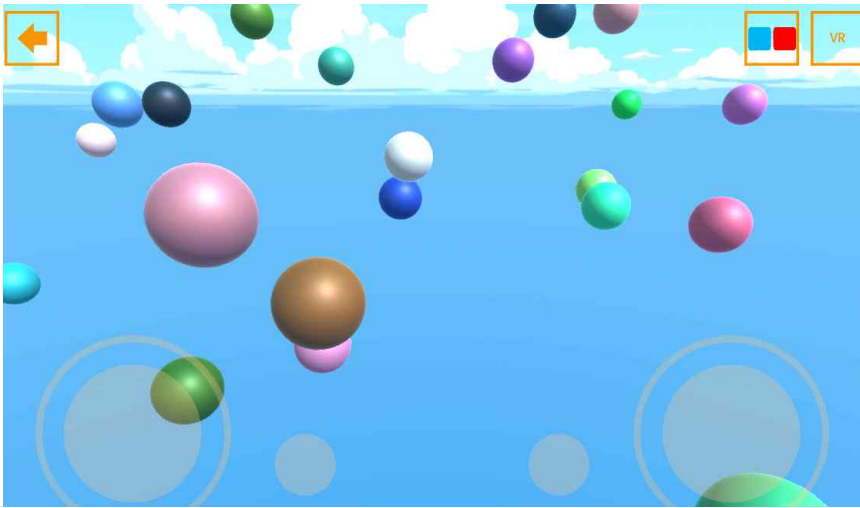
- 위의 코드에서 변수 x에는 -10 ~ 9 사이의 값이 임의로 저장되며, y에는 0 ~ 19, z에는 -10 ~ 9 사이의 임의의 값이 저장된다.
- 위의 코드를 실행하면 마치 눈이 떨어지듯이 공이 임의의 공간에 배치되어 떨어지는 것을 볼 수 있다.



실습

- ▶ 앞의 활동에서 작성한 예제에서 공의 탄성 옵션을 제거하고 실행하면 어떠한 결과가 나올지 예측해 보고 결과를 비교해 본다.

```
● for (i = 0; i < 100; i++)  
  x = random(-10, 10)  
  y = random(0, 20)  
  z = random(-10, 10)  
  공모양 추가하기  
    위치: {x}, {y}, {z}  
  기다리기(100)
```



2.7 함수 만들기

나만의 함수

- VR 코딩 언어는 SPL (Simple Programming Language)라는 간단한 언어와 C언어 문법을 지원한다. SPL은 기본적으로 C언어의 기본 규칙을 따른다.
- 함수는 여러 명령어들의 집합으로서, 이 함수 이름을 호출해 주면 해당 함수 안에 있는 명령어들이 순서대로 실행되게 되어, 코드의 길이를 줄이고 재사용성을 높일 수 있다.
- 기본적으로 제공되는 대표적인 함수는 setup 함수와 loop 함수가 있다. setup 함수는 처음에 한번 만 실행되며, loop 함수는 무한히 반복적으로 실행된다, 이 두 함수는 프로세싱이나 아두이노의 코딩 방식과 동일하다.



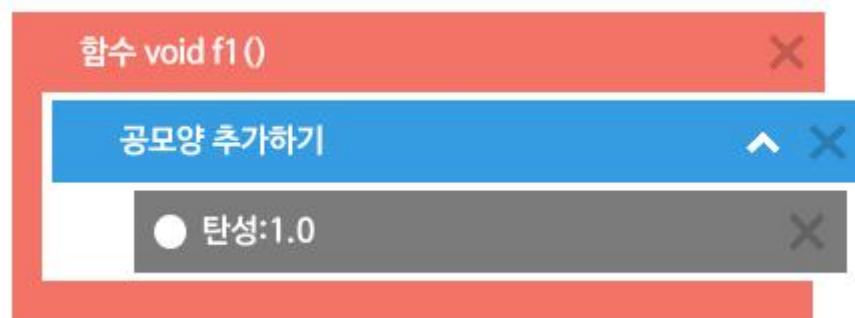
- 기본 제공되는 함수 외에 사용자가 직접 함수를 추가하여 작성할 수 있다.
- 로직 명령어에서 함수 명령어를 드래그하여 추가하면 기본적인 함수의 틀이 추가된다. 함수 블록 사이에 필요한 명령어들을 추가하면 되며, 함수의 이름은 속성값에서 변경할 수 있다.



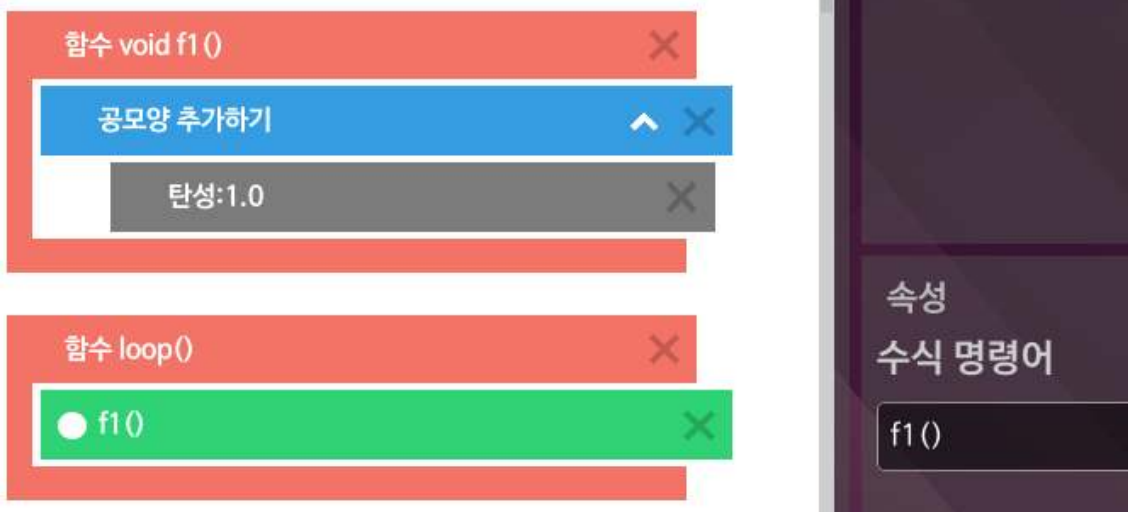
- 위의 예에서 함수의 이름은 f1() 이며, 이름 앞에 void라는 표시가 있는 것은 이 함수가 명령어 실행이 끝난 후, 따로 돌려주는 값이 없다는 의미이다.

함수에서 공 생성하기

- 함수가 호출될 때 마다 공이 생성되도록 함수 안에 코드를 추가해 보자. 아래의 예는 함수 안에 공 모양을 생성시키는 명령어를 추가한 예를 보여준다.



- 위의 프로그램은 이 상태로 실행하면 아무런 결과가 표시되지 않는다. 그 이유는 함수를 호출해 주는 부분이 없기 때문이다. 이제 loop 함수를 추가하여 위에서 생성한 함수를 반복적으로 호출해 보자.



- 함수를 호출하기 위해서는 수식 명령어를 사용하며, 수식명령어 내용에 다음과 같이 호출하고자 하는 함수를 입력해 주면 된다.

f1()

2.8 조이스틱 버튼 활용하기

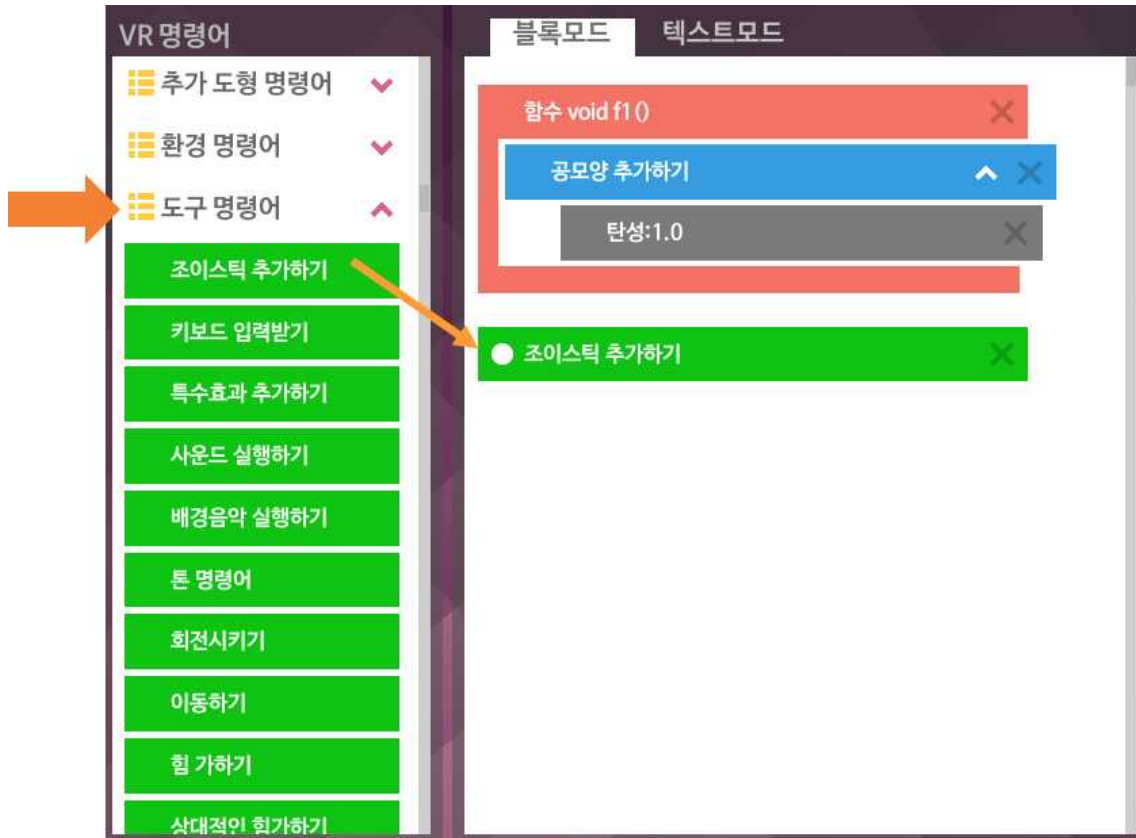
조이스틱 버튼

- VR 코딩 편집기는 실행시에 자동으로 조이스틱 버튼이 화면에 표시된다. 이 때 좌우 2개의 버튼은 카메라를 회전시키거나 이동시키는 경우에도 사용되지만 해당 버튼을 클릭할 때에도 이벤트를 발생시킨다.

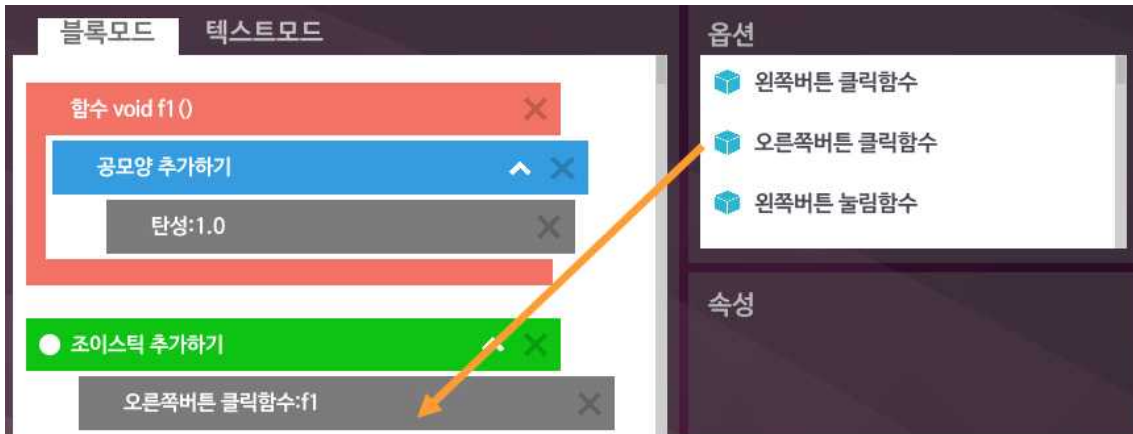


- 조이스틱 버튼을 활용하기 위해서는 코드에 조이스틱 추가하기 명령어를 추가해 주어야 한다. 이전 활동의 코드에서 loop 함수 코드를 제거하고 다음과 같이 조이스틱 추가하기 명령어를 추가해 준다.

- 조이스틱 추가하기 명령어는 도구 명령어 그룹에 포함되어 있다.



- 조이스틱 추가하기 명령어를 선택한 후, 오른쪽 옵션에서 두 번째에 옵션값인 오른쪽 버튼 클릭함수 옵션을 추가해 준다.



- 위의 코드의 의미는 조이스틱에서 오른쪽 버튼이 클릭되면, f1() 함수를 실행시키라는 의미이다.

- 코드를 실행한 후 아래 그림과 같이 오른쪽 조이스틱 버튼을 클릭해 보자.



실습

- ▶ 오른쪽 조이스틱 버튼을 클릭했을 때 공이 임의의 위치에 생성되도록 기능을 수정해 본다.

2.9 도미노 생성하기

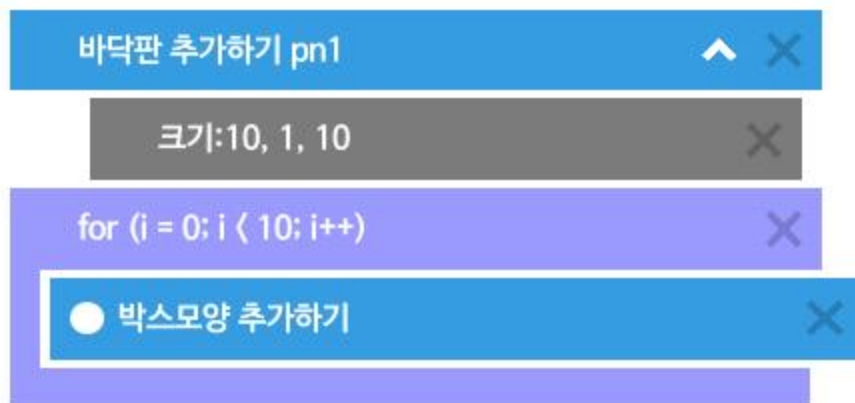
도미노 바닥판 만들기

- 바닥판 추가하기 명령어를 이용하여 도미노가 만들어질 바닥판을 생성해 본다.
- 바닥판의 크기는 10, 1, 10으로 지정해 준다.



반복문으로 도미노 블록 만들기

- 이제 반복문을 이용하여 넘어지는 도미노 블록을 만들어 본다.
- for 반복문을 추가하고 반복문 영역 안에 박스모양 추가하기 명령어를 추가한다. 박스 모양의 이름이 자동 생성되도록 속성 값에서 이름을 지워준다.



- 박스모양 추가하기 명령어에 크기 옵션을 다음과 같이 추가해 준다.



- 박스모양 추가하기 명령어에 위치 옵션을 다음과 같이 추가해 준다.

The image shows a code editor interface with the following elements:

- A blue header for "바닥판 추가하기 pn1" with expand/collapse and close icons.
- A grey input field for "크기:10, 1, 10" with a close icon.
- A purple header for "for (i = 0; i < 10; i++)" with a close icon.
- A blue header for "박스모양 추가하기" with expand/collapse and close icons.
- Two grey input fields under "박스모양 추가하기": "크기:0.5, 3, 1" and "위치: {i}, 1.5, 0", both with close icons.
- A 3x3 grid of configuration options to the right:

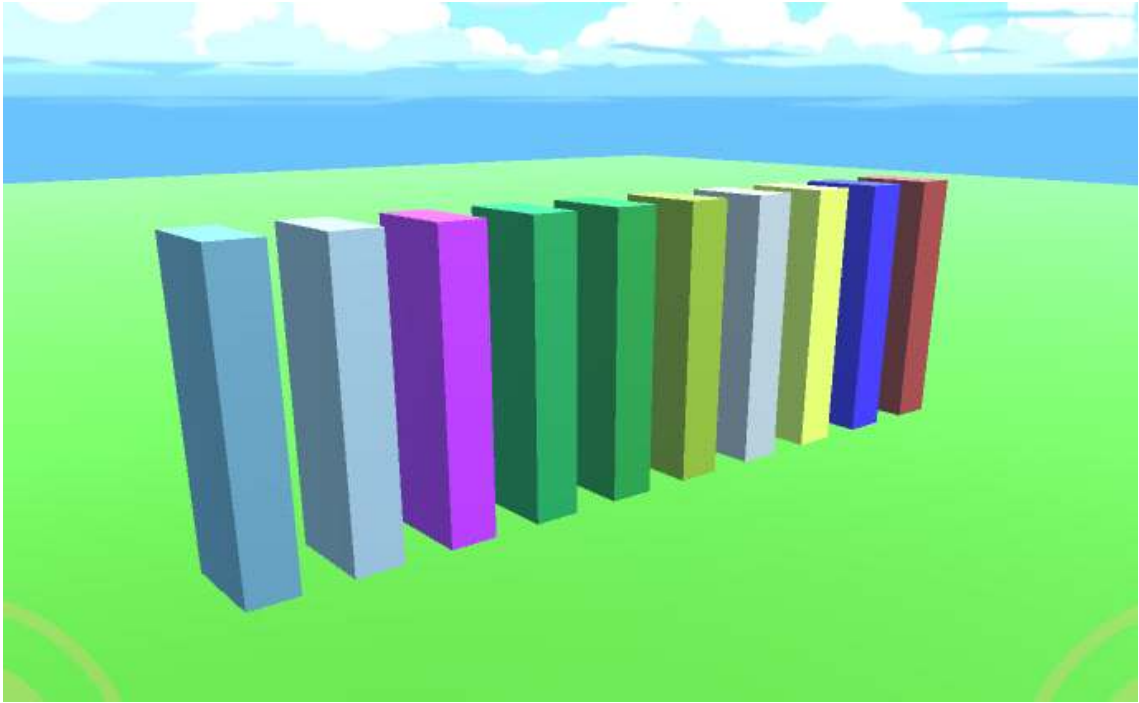
X	-	{i}	+
Y	-	1.5	+
Z	-	0	+

- 상자가 넘어질 수 있도록 탄성 옵션을 추가해 준다.

The image shows the same code editor interface as above, but with an additional option in the "박스모양 추가하기" panel:

- A grey input field for "탄성:1.0" with a close icon, added below the "위치" field.

- 코드를 실행하여 상자가 잘 만들어 지는 지 확인해 본다.



떨어지는 공 추가하기

- 상자를 넘어뜨리기 위해 5m 높이에서 공이 떨어지도록 공 모양을 추가해 보자.
- 다음의 코드를 기존 프로그램 맨 아래에 추가해 본 후, 코드를 다시 실행해 본다.





실습

- ▶ 도미노 블록의 개수를 더 증가시켜 본다.
- ▶ 도미노 블록의 방향을 변형시키기 위해서는 어떠한 옵션이 필요할지 생각해 보고 기능을 수정해 본다.