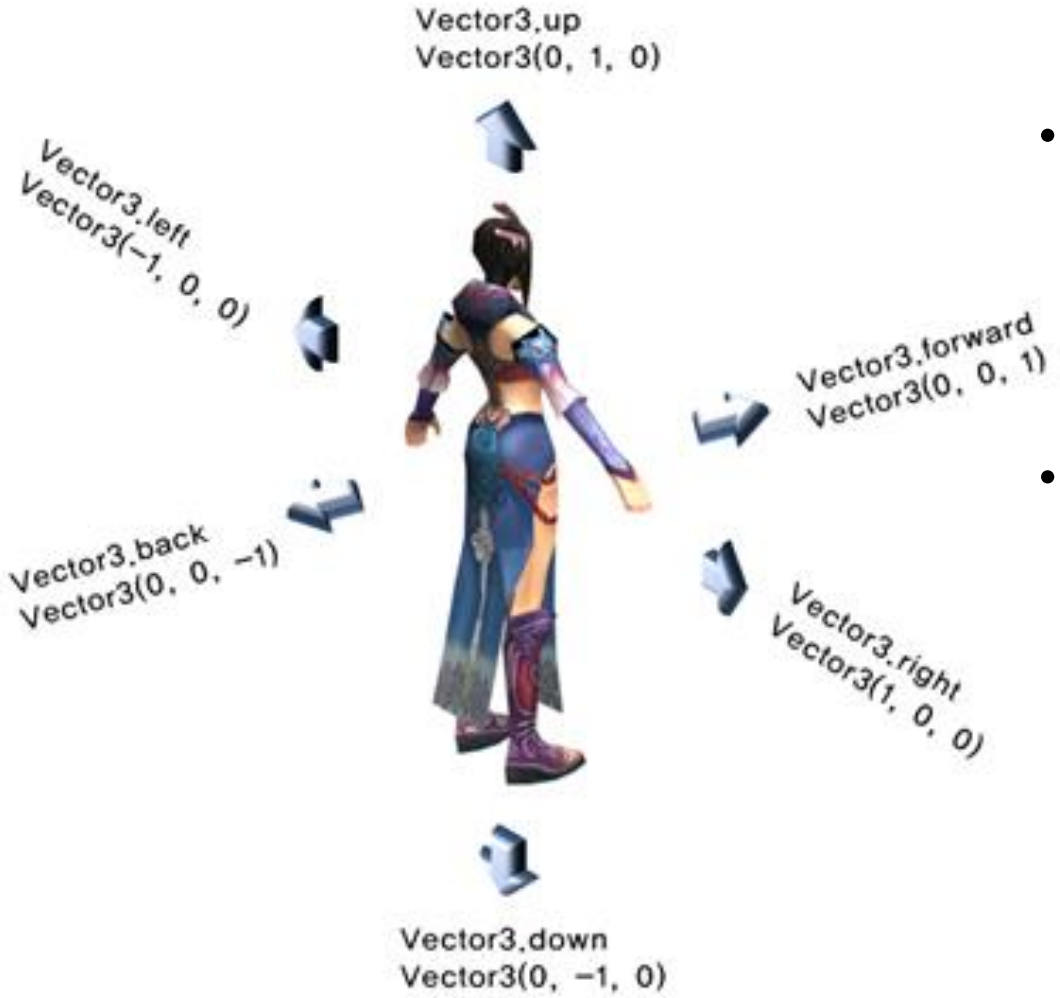
게임기획실무

스크립트 활용하기

스크립트 작성하기



- 방향을 지시하는 용도로 사용할 때는 (x, y, z) 값을 0과 1로 표시한다.
- 거리를 지시하는 용도로 사용할 대는 (x, y, z) 값을 해당 크기만큼 표시한다.

Time.deltaTime

- Update 함수 호출 주기는 컴퓨터의 성능, 게임의 최적화 정도에 따라서 달라진다.

```
void Update(){
    a+=1;
}
```

- Update가 **1초에 60번씩 호출**된다고 하면 10초후 a의 값은 **10*60*1=600**
- Update가 **1초에 30번씩 호출**된다고 하면 10초후 a의 값은 **10*30*1=300**

Time.deltaTime

```
void Update(){
a+=60*Time.deltaTime;
}
```

- A는 1프레임을 처리하는데 1/60초가 걸린다. Time.deltaTime = 1/60
- 10초후의 a값 = 10*60*60*(1/60)=600
- B는 1프레임을 처리하는데 1/30초가 걸린다. Time.deltaTime = 1/30
- 10초후의 a값 = 10*30*60*(1/30)=600
- deltaTime을 사용함으로써 사양이 다른 두 상황에서도 같은 값

Input.axis

Negative Button : 음의 값을 반환되는 버튼

Positive Button : 양의 값을 반환되는 버튼

Alt Negative Button : 음의 값을 반환되는 다른 버튼

Alt Positive Button : 양의 값을 반환되는 다른 버튼

Gravity : 키를 누르면 1이 되고 그 1이 다시 0으로 되돌아오는 속도

Dead : 이 범위에 속하는 모든 아날로그 장치 값은 중립되게 만든다.(조이스틱 유용)

Sensitivity : 값이 커질수록 반응 시간이 빨라지며, 값이 작을수록 입력이 더 부드러워진다.

Snap : Type이 key/mouse button인 경우만 해당되며 반대의 키값을 눌렀을 경우 값이 0으로 초기화 된다.

Rotate

- 1번째 씬을 저장한다. [Rotate_1]
- 스크립트를 추가한다. [Rotate1]

```
public class Rotate1 : MonoBehaviour {
    void Start () {
        // 트랜스폼의 rotation 값 초기화 : 회전 #1
        transform.eulerAngles = new Vector3 (0.0f, 50.0f, 0.0f);
```

Rotate

```
        // 트랜스폼의 rotation 값 초기화 : 회전 #2
        // rotation은 Quaternion 값으로 대입해 주어야 한다.
        Quaternion target = Quaternion.Euler (0.0f, 100.0f, 0.0f);
        transform.rotation = target;
        // 회전
        transform.Rotate (Vector3.up * 90.0f);
    }
}
```

Scene 전환

```
using UnityEngine.SceneManagement;


public void Click () {
        SceneManager.LoadScene("Scene name");
    }
```

스크립트 활용하기 – Rotate_2

- 2번째 씬을 저장한다. [Rotate_2]
- Cube가 전후좌우로 회전하게 만든다.
- 스크립트 생성하기 – Rotate2

스크립트 작성하기

```
public class Rotate2 : MonoBehaviour {
        float speed = 50.0f;

    void Update () {
            // 키보드 입력 스크립트
            float h = Input.GetAxis("Horizontal");
            float v = Input.GetAxis("Vertical");

            // 이동 거리 보정 스크립트
            h = h * speed * Time.deltaTime;
            v = v * speed * Time.deltaTime;
```

스크립트 작성하기

```
            // 회전 #1
            transform.Rotate(Vector3.right * h);
            transform.Rotate(Vector3.forward * v);
        }
}
```

스크립트 작성하기

```
        // 회전 #2
    transform.rotation *= Quaternion.AngleAxis
                                (h, Vector3.forward);
    transform.rotation *= Quaternion.AngleAxis
                                (v, Vector3.right);
    }
}
```