포트폴리오 설명문서

작성자 : 박민섭

작성일 : 2019.03.13

목차

1. 게임 설명

↑ 게임 시작시 타이틀화면

게임 제목 : Caken

게임 장르 : 아케이드 게임

사용 엔진 : Unity3D

사용 언어 : C#

목표 플랫폼 : 안드로이드 (모바일)

화면 해상도(Pixel) : 2560 X 1440(16 : 9, 가로고정)

프로젝트 소스 링크 : https://github.com/BakMinSeop/Caken.git

APK 파일 링크 : https://drive.google.com/open?id=1Ooha-cC-nFv3jgPXagwofYZgP-tPqxFE

설명 :

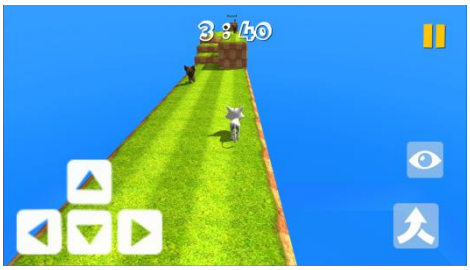
Caken은 영화 Taken과 고양이(Cat)을 더한 합성어이며, 평화롭던 길냥이 들에게 한 사람이 나타나 새끼 고양이를 데리고 가버리면서 어미 고양이가 찾아 나선다는 설정으로 만들었습니다.

안드로이드 플랫폼으로 제작되었으며 간단하게 이동버튼과 점프버튼으로 구성되어 있고, 여기에 카메라 시점변경 버튼을 더하여 상황에 따라 대처 할 수 있도록 하는 재미요소를 추가 하였습니다.

2. 소스 코드

카메라의 뷰를 Orthographic 또는 Perspective로 변경 하도록 하여 각 설정마다 카메라의 위치를 변경하고 플레이어와 일정 거리에 위치를 고정하여 따라다니도록 코딩 하였습니다.

↑ 카메라 뷰 설정이 Orthographic일 경우



↑ 카메라 뷰 설정이 Perspective일 경우

```
void FollowCamera()
  {
    if (ChangeView.orthographic)
    {
      Vector3 targetpos = PlayerObj.transform.position - (Vector3.forward * 5) + (Vector3.up * 2.5f);
      m_UseCamera.transform.position =
                                  Vector3.Lerp(m_UseCamera.transform.position, targetpos, LerpVel);
      CameraView.rotation = Quaternion.Euler(0, 0, 0);
    }
    else
    {
      Quaternion rot = Quaternion.Euler(0, 90, 0);
      Vector3 targetpos = Target.position - (rot * Vector3.forward * Floating) + (Vector3.up * Height);
      CameraView.position = Vector3.Lerp(CameraView.position, targetpos, LerpVel);
      CameraView.LookAt(Target);
    }
  }
```

2. 소스 코드

캐릭터의 움직임을 표현한 코드입니다.
카메라의 뷰 방식에 따라 캐릭터가 바라보는 각도 및 앞뒤로 움직일 것인지 4방향으로 움직일 것인지 조건에 따라 바뀌도록 코딩 하였습니다.

```
void moveObject()
    {
        float keyHorizontal = m_HorizontalVal;
        float keyVertical = m_VerticalVal;
        bool ISAKey = false;
        bool ISDKey = false;
        bool ISWKey = false;
        bool ISSKey = false;

        if (m_ChangeCamera.ChangeView.orthographic)
        {
            if (keyHorizontal <= -1)
            {
                ISAKey = true;
            }
            else if (keyHorizontal >= 1)
            {
                ISDKey = true;
            }
        }
...
```

2. 소스 코드

```
…
        else
        {
            if (keyHorizontal <= -1)
            {
                ISAKey = true;
            }
            else if (keyHorizontal >= 1)
            {
                ISDKey = true;
            }

            if (keyVertical <= -1)
            {
                ISSKey = true;
            }
            else if (keyVertical >= 1)
            {
                ISWKey = true;
            }
        }

        if (m_ChangeCamera.ChangeView.orthographic)
        {
            m_OffsetMove.x = speed * Time.deltaTime * keyHorizontal;
…
```

2. 소스 코드

```
…
        Vector3 OrthographicPos = transform.position;
        OrthographicPos.z = -2f;
        transform.position = OrthographicPos;

        if (ISAKey)
        {
            transform.rotation = Quaternion.Euler(0, -90, 0);
        }
        if (ISDKey)
        {
            transform.rotation = Quaternion.Euler(0, 90, 0);
        }
        if ((keyHorizontal != 0 || keyVertical != 0) && !isjump)
        {
            ismove = true;
        }

        GetComponent<Animator>().SetBool("Move", ismove);
        m_LinkRigidBody.constraints =
                    RigidbodyConstraints.FreezePositionZ | RigidbodyConstraints.FreezeRotation;
    }
        else
        {
            m_OffsetMove.x = speed * Time.deltaTime * keyVertical;
            m_OffsetMove.z = -speed * Time.deltaTime * keyHorizontal;
…
```

2. 소스 코드

```
…
        if (ISSKey)
        {
            transform.rotation = Quaternion.Euler(0, -90, 0);
        }
        if (ISWKey)
        {
            transform.rotation = Quaternion.Euler(0, 90, 0);
        }
        if (ISAKey)
        {
            transform.rotation = Quaternion.Euler(0, 0, 0);
        }
        if (ISDKey)
        {
            transform.rotation = Quaternion.Euler(0, -180, 0);
        }
        if (ISAKey && ISSKey)
        {
            transform.rotation = Quaternion.Euler(0, -45, 0);
        }
…
```

2. 소스 코드

```
...
        if (ISAKey && ISWKey)
        {
            transform.rotation = Quaternion.Euler(0, 45, 0);
        }
        if (ISDKey && ISSKey)
        {
            transform.rotation = Quaternion.Euler(0, -135, 0);
        }
        if (ISDKey && ISWKey)
        {
            transform.rotation = Quaternion.Euler(0, 135, 0);
        }
        if ((keyHorizontal != 0 || keyVertical != 0) && !isjump)
        {
            ismove = true;
        }

        GetComponent<Animator>().SetBool("Move", ismove);
        m_LinkRigidBody.constraints = RigidbodyConstraints.FreezeRotation;
    }
}
```

3. 특징

Playerprefs를 사용하여 Stage의 클리어 여부를 저장하고 불러오도록 했습니다.

↑ 스테이지 클리어 전

↑ 스테이지 클리어 후

게임내 설정 또한 **Playerprefs**를 사용하여 어느 씬에서 설정을 해도 모든 씬에 적용 되도록 하였습니다.

4. 개선 방향

이후 개선하려고 생각하는 바는

1. 프롤로그 에필로그 삽입
   - 텍스쳐와 자막을 이용
2. 추가 맵 제작
   - 다양한 배경과 환경의 맵을 제작
   - 지루하지 않도록 적당한 선에서 엔딩
3. 아이템 및 NPC 추가
   - 맵 곳곳에 숨겨진 루트를 만들어 스토리 진행에 도움을 줄 NPC와의 이벤트 추가
   - 획득시 스킬이 추가되는 아이템 생성
4. 캐릭터 & 스킨 추가
   - 고양이 모델 추가

감사합니다.