BeatRoads 아이디어

전체 구조 :

1. 유니티 커스텀 Tool을 통해 해당 노래에 대한 노트를 만들고 노트의 시작시간, 끝 시간이 기록된 JSON 파일 생성

2. 1에서 생성된 JSON파일을 [SerializeField]를 통해 외부에서 담아 시간에 맞춰 노트를 생성해주는 NoteMaker. NoteMaker은 AudioSource와 JSON을 데이터로 받는다

3. PlayerController에서 플레이어 캐릭터 그리드 이동

4. JudgeManager가 노트 판정 맡음. 떨어지면 캐릭터 깜빡이며 재생성

1. 오디오 스펙트럼 분석해서 비트 만들기

a. 적절한 비트 만들기 문제

i. window를 통해 앞뒤 50개의 spectrum의 평균을 구하고 적절한 beat 판별 ratio를 곱함

>> 현재 스펙트럼의 값이 i의 결과보다 크다면 beat로 간주

>> 단 이전 비트 생성 시간보다 일정 시간 이후부터 생성 가능

>> 추가할 사항으로 노래의 bpm에 따른 판별식도 고려

ii. 기존의 (https://medium.com/giant-scam/algorithmic-beat-mapping-in-unity-real-time-audio-analysis-using-the-unity-api-6e9595823ce4) 코드 활용시 해당 샘플의 비트는 잘 뽑아내지만 비트가 너무 많이 생성된다

- 기존의 방식 :

>> 직전 프레임과 현재 프레임의 스펙트럼 음역대별 비교로 스펙트럼 증가분(flux) 합 계산

>> 현재 스펙트럼 기준 앞뒤로 window/2개 만큼에 대해

>> window 전체의 flux 평균에 비트 계산 상수를 곱하여 현 스펙트럼의 비트 기준값 결정

>> 해당 비트 기준값을 기준으로 현 프레임의 스펙트럼이 얼마나 큰지 계산 - prune

>> 이때 크지 않다면 0으로 저장

>> 이후 index - 1의 prune값을 기준으로 Index -2, Index를 비교

>> index -1의 prune 값이 가장 크다면 피크값으로 판정

- 내 보정

>> 곡의 실제 bpm과 비교해보면 거의 10배 가까이 더 생긴다(10배 이상)

>> sol1. 비트 상수의 크기를 기운다

>> 테스트 결과 비트 상수의 크기가 2일 때 실제 비트와 유사한 결과를 보여준다

>> sol2. 비트가 과추출되는 것을 방지하기 위해 && 현실적인 플레이를 위해 비트 갱신 최소시간을 둔다

>> sol3. 비트 갱신 최소 시간을 넘기는 비트를 기준으로 주변의 비트들 중 일정 시간 내의 비트들 중 가장 스펙트럼의 값이 큰 것을 비트로 결정한다

b. 적절한 길이의 노트 만들기

>> i. 이후 비트의 시작시간 = 이전 비트의 종료시간

>>ii. note에 디버깅 용으로 str와 ed 시간 및 길이, 좌표를 public으로 에디터상에 표기하는 스크립트 만들어서 붙이기

2. 비트의 시작시간, 노트 만들 위치를 엑셀로 뽑기

a. 커스텀 에디터를 만들기는 했는데 해당 Editor 폴더 내에서는 폴더 외부의 클래스를 접근할 수 없었음

>> 에디터 파일은 다른 모든 것에 대한 참조와 함께 마지막으로 컴파일된다

(https://discussions.unity.com/t/acces-a-script-in-the-editor-folder/67107)

>> 노트를 생성할 스크립트를 에디터 폴더 내로 가져와도 클래스 참조는 하지만 preNoteMaker에 대한 인스턴스 생성이 안됨

>> 현재 문제 :

a. 엉뚱한 경로에 JSON 파일이 생성됨

>> 경로 오류였음

>> 경로에 권한 문제로 파일생성이 안 됨

>> Path.Combine(Application.datapath, path)등으로 해봤지만 계속 권한 문제

>> 걍 터미널서 권한 변경해주니 됐음

b. 원하는대로 json 파일이 생성되지 않음

>> totalNote 리스트가 계속 누적되고 있는 문제

>> System.GC.Collect()를 해주니 첨에 괜찮다가 또 리스트 누적됨

>> 각 리스트에 대해 인스턴스 할당을 전역이 아니라 해당하는 지역에서 선언했더니 괜찮아짐

c, FileInfo가 처음엔 되더니 안됨

>> 아마도 +".json"때문으로 추정

>> 위의 이유가 맞았음

3. 엑셀을 통해 수정해서 유니티에 JSON으로 불러오기

4. 3에서 불러온 JSON 데이터를 커스텀 에디터를 통해 노트 위치 / 시작 / 끝을 수치로 보기

5. 4의 커스텀 에디터에서 위치, 시작, 끝을 수정할 수 있도록 하기

6. 노트메이커

a. 노트의 생성 타이밍 및 속도 문제

>> 노트의 영역이 중복되는 (이전 노트 영익이 끝나기 전에 이후 노트의 영역이 중복되는 문제)발생

>> 이전 노트와 이전 노트의 도착 시간 차이가 각 노트의 시작 시간 차이와 다름

노트 영역이 중복되는 문제 해결

a. 속력 = 거리 \* 시간

b. 노트의 scale을 시간에 비례해서 만들었는데, 노트의 좌표 기준은 오브젝트의 '중심'이고 노트의 충돌판정은 노트의 '끝'이기 때문에 단순히 노트를 noteMaker 좌표에 생성하면 안 된다

>> 노트 길이 - scale - 에 따른 보정 필요

>> 모든 노트가 NoteMaker을 기준으로 동일선상에서 출발하고 같은 거리를 이동하도록 노트 생성시 좌표의 z값에 scale / 2만큼 더해 보정한다

>> 동일 선상에서 출발하여 Abs( PrevStartTime, CurStartTime)이 Abs(PreConflicTie, CurConflicTime)과 같아짐

b. 플레이어가 노트 타이밍마다 grid를 이동할 수 있도록 각 노트는 종료시점에 새로운 노트가 겹쳐져야 함. 하지만 각 노트의 startTime마다 노트를 생성하면 노트와 노트 사이의 공백이 발생

>> 단순히 노트의 길이를 길게 하는 것으로는 노트마다 노트간 겹쳐지는 영역의 차이가 발생

(단순히 endTime - startTime \* ratio기 때문)

해결방안

a. 거리 = 속력 \* 시간 & 속력 = 거리 / 시간으로 뭔가 될 것 같은대..

>> 노트의 생성지점부터 판정지점까지의 이동시간 + 노트오브젝트가 판정지점을 전부 통과하는 시간을 이용

>> 노트의 구간 이동 시간 + 노트 통과시간 = 다음 노트의 도착 시간