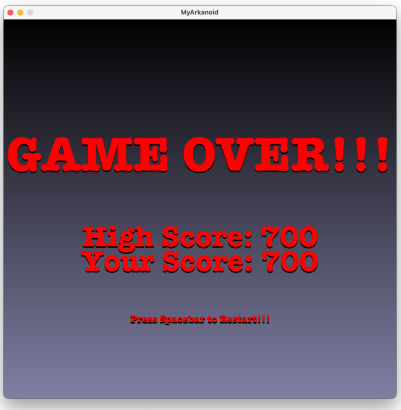
HW5: Block Breaker에 관한 보고

21011794 황재동(2024.12.23)

개요:

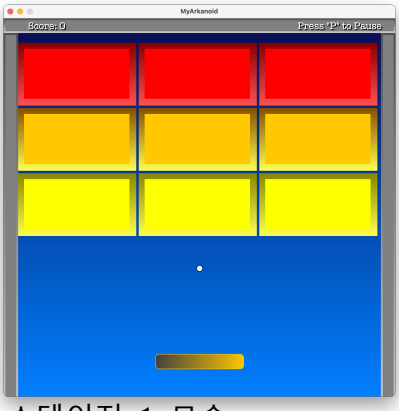
Java Swing을 활용하여 Block Breaker(Arkanoid)를 제작하는 과제다. Game Title Scene, In Game Scene, Game Over Scene 총 3개의 씬으로 나뉜다. JPanel를 상속받은 3개의 클래스들을 씬처럼 사용할 것이고 스테이지는 1단계부터 시작해 모든 공이 화면 밖을 나갈 때까지 무한으로 진행된다. 일반 블록과 아이템 블록이 존재하고 아이템 블록을 파괴하면 공이 3개로 나뉜다. 점수 시스템이 존재해 자신의 현재 기록 및 최고기록을 기록할 수 있다. Thread및 간단한 물리를 사용하여 애니메이션을 구현하고 사운드 시스템을 도입하여 몰입감 있는 경험을 제공한다. 하단은 개발종료 후 게임의 모습을 캡처한 것이다.

게임 시작시 모습

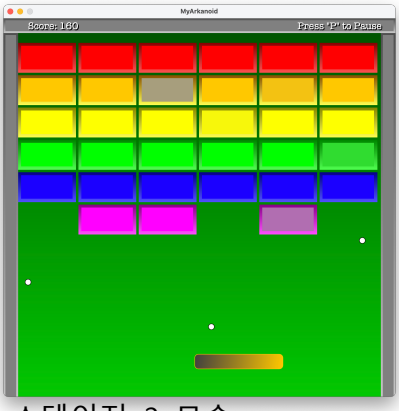


게임 종료시 모습

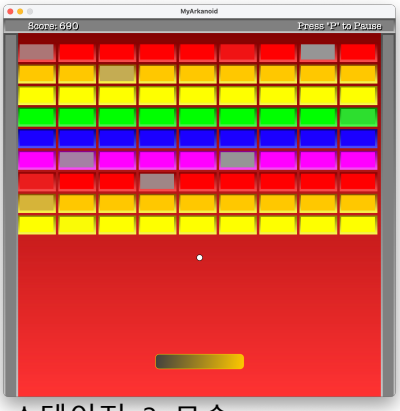
스테이지 완료시 모습



스테이지 1 모습



스테이지 2 모습



스테이지 3 모습

방법 구상:

클래스는 크게 기본 틀을 담당하는 MyArkanoidFrame, 타이틀 화면을 보여주는 MyStartScene, 게임을 플레이할 MyArkanoidInGame, 모든 공이 화면 밖으로 나갔을 때 보여줄 MyFailScene 4개로 나뉘고 인게임에서 보여지는 모든 물체를 담은 MyGameObject 클래스, 음원들을 관리하는 SoundManager 클래스가 있다. MyGameObject, SoundManager 클래스를 제외한 모든 클래스는 JPanel형 mainPanel이라는 변수를 갖고 있고 스페이스바 입력 또는 게임 종료등 씬 전환이 필요할 때 이를 다른 클래스로 넘겨준다. mainPanel에 여러 패널들을 띄워 씬을 전환하는 방식이다.

가장 중요한 씬인 MyArkanoidInGame은 스테이지에 따라서 블록 개수가 증가한다. 이는 현재 스테이지에 따라서 가로 세로를 나눠 블록을 그리면 될 것이다. 공과 라켓,벽,블록의 충돌은 교수님께서 주신 코드를 응용해 완성 시키면 된다.

구조 및 기능 설명:

1. MyArkanoidFrame class:

게임의 기본틀이 되는 클래스다. 사이즈는 정사각형으로 가로, 세로 800으로 설정하였고 씬을 전환하는 효과를 주기 위해서 JPanel형 mainPanel을 초기화하여 시작씬은 MyStartScene을 넣어줬다. 이때 mainPanel을 계속 가져가기 위해서 MyStartScene에 있는 mainPanel에 현재의 mainPanel도 같이 넘겨준다.

2. MyStartScene class:

MyStartScene은 게임 진입 전 게임의 제목을 보여주고 스페이스 바를 누르면 게임을 시작할 수 있다는 것을 보여주는 간단한 씬이다. "Press Space to Play!"라는 문구가 깜빡거리는 효과를 주기위해서 쓰레드를 발생시킨다. 현재 시간의 경과에 따라서 paintComponent 메서드에서 이 문구를 보여줬다, 안 보여줬다를 반복한다. 가장 핵심기능은 스페이스 바를 눌렀을 때 게임 화면으로 전환되는 것이다. 이때 중요한 점이 스페이스 바 입력을 위해서 생성자에서 setFocusable을 했었는데 이것을 현재 패널에서 중단 후 다음 패널에 키보드 권한을 넘기는 것이다. 그렇지 않으면 게임 화면에서 키보드 입력이 동작하지 않는다.

또한 run메서드에 있는 while문도 true가 아니라 불리언 변수를 하나 만들어 다른 패널로 교체하기 전에 false로 바꿔준다. 앞으로 나오는 모든 run 메서드는 이와 동일한 매커니즘으로 동작한다. 한가지 더 주의할 점이 있었는데 다음 스테이지를 담고 있는 패널을 생성자에서 바로 초기화하면 안되고 키입력을 받는 순간 초기화를 시켜야 했다. 미리 초기화하니 백그라운드에서 패널이 작동해 스페이스바를 누르고 넘어가니 이미 게임이 종료된 직후의 모습을 보여줬다.

3. MyGameObject class:

MyArkanoidInGame 클래스에서 사용될 모든 게임 오브젝트가 정리 되어 있는 중요한 클래스 중 하나다. 테두리가 되는 벽, 일반 블록, 공, 라켓의 정보를 담고 있는 클래스들이 모두 이 추상클래스를 상속받는다. 아이템 블록은 일반 블록과 비슷한 속성을 갖고 있기 때문에 유일하게 일반 블록을 상속 받

는 클래스다.

게임에 있는 모든 오브젝트는 그려질 필요가 있기 때문에 draw메서드를 추상 메서드로 선언했다. 공에 대한 것을 갖고 있는 MyHWBall 클래스는 공과 벽 또는 라켓과의 충돌을 처리해준다. 기본적으로 부딪힐 경우 기존의 vx,vy를 상황에 맞게 -1을 곱한다. 라켓은 예외가 있는데 막대의 중앙을 0이라하고 이를 기준으로 양끝을 -1/1이라고 했을 때 공이 맞는 위치에 따라서 반사각을 다르게 하려고 노력했다. 쉽게 말해 공이 끝에 맞을 수록 더욱 그쪽으로 반사각이 늘어난다. 이는 공이 수직으로 들어오면 항상 수직으로 반사되던 필자 게임의 단점을 보완해주어 게임에 변수를 만들어주어 재미요소로 추가했다.

일반 블록과 아이템 블록은 공이 벽과 충돌했을 때와 유사하지만 블록이 공과 충돌했을 때 파괴됐다는 것을 표시하기 위한 isDestroyed변수가 존재한다.

4. MyArkanoidInGame class:

가장 중요한 클래스다 MyGameObject의 오브젝트가 모두 사용되기도 하고 게임의 대부분이 여기서 진행되기 때문에 리스트의 초기화 문제나 쓰레드의 문제도 여기서 가장 많이 발생했다. 먼저 생성자에서는 스테이지를 초기화하는 메서드를 호출하고 쓰레드를 발생시킨다.

스테이지를 초기화하는 역할을 하는 initializeStage메서드는 먼저 상,좌,우에 벽 오브젝트를 objs 리스트에 추가하고 라켓과 공을 추가한다. 이때 공의 초기 방향을 위쪽만 향하게 하면 사용자가 최초에 조작하지 않고 가만히 있어도 가장 끝 블록까지 깰 수 있는 경우가 발생했다. 이 때문에 전체적인 방향이 위쪽을 향하되 랜덤하게 좌우로 30도 범위 안에서 꺾여 발사되게 설정하였다. 추가적으로 공의 속도도 스테이지에 따라서 달라지게 설정했다. 일반 블록과 아이템 블록의 개수는 가로, 세로가 스테이지 번호에 비례해서 증가할 수 있게 했다. 아이템 블록은 20%의 확률로 생성되고 일반 블록은 80%의 확률로 생성된다.

paintComponent메서드는 조건에 따라서 다른 역할을 수행한다. 가장 먼저 현재 스테이지에 따라 배경을 다르게 색칠해주고 스테이지가 클리어됐다는 판단을 내리면 화면에 스테이지 클리어 문구를 띄운다. 일반적으로는 반복문을 돌면서 오브젝트 배열에 있는 오브젝트들을 게임화면 상에 그려주며 아

이템 블록이 파괴됐다는 판단을 내리면 배열에 새로운 공들을 추가한다. 공이 계속 생성되는 것을 방지하기 위해 아이템 블록에는 isDestroyed외에 itemUsed라는 변수도 있다. 아이템이 사용되면 true로 바꿔준다. 그래도 혹시 모르니까 마지막에는 파괴된 블록들은 배열에서 제거한다. 추가적으로 좌상단에는 점수를 표기해주고 우상단에는 "p"를 누르면 일시정지할 수 있다는 것을 알린다.

방향키 조작시 라켓의 이동 구현을 위해 keyListener에 관한 메서드도 오버라이드한다. run메서드는 보기보다 간단한데 실시간으로 움직이는 오브젝트들을 update메서드를 통해 업데이트 해주고 collisionResolution메서드를 통해 충돌을 감지한다. 또한 모든 공이 화면 밖으로 나갔는지 판단하고 모두 나갔으면 게임오버씬을 부른다.

5. MyFailScene class:

MyStartScene과 굉장히 유사한 구조로 MyStartScene에서는 게임의 이름을 그렸던 것과 달리 이 클래스에서는 사용자의 최고점수와 가장 최근에 했던 게임의 점수를 알려준다. 이 클래스도 스페이스 바를 누르면 다시 메인 화면으로 돌아간다는 것을 알려주기 위해서 해당 정보를 깜빡거리게 하고 스페이스바를 누르면 MyStartScene이 나온다.

6. SoundManager class:

이름에서도 알 수 있듯이 게임에서 재생할 노래들을 저장해 놓은 클래스다. 싱글톤으로 구현되지는 않았지만 전역으로 선언해 어디서든 사용할 수 있다. 실제 필자가 "Unity"엔진을 통해 개발한 게임에 있는 Audio Manager 스크립트에서 아이디어를 얻어 구현한 클래스다. 음원을 재생, 정지할 수 있는 기능을 제공하며 원하면 루프를 할 수도 있다.

비쥬얼 및 사운드를 통한 디테일:

1. 스테이지 별 다른 배경색

스테이지 1에서는 무언가 쉬운 느낌을 주는 파랑 배경으로 시작해 스테이지 2는 초록배경으로 하였고 스테이지 3이상은 빨강배경으로 하였다. 모두 그라디언트 페인트를 사용하였는데 패널의 상단은 찐한 색으로하고 하단 부분은 밝은 색으로 하여 자연스러운 느낌을 줬다.

2. 무지개색 블록들

일반 블록 및 아이템 블록을 위에서부터 차례대로 무지개색으로 표현하여 실제 Arkanoid게임과 유사한 느낌을 냈다. Color배열에 무지개색들을 저장하고 현재 행의 인덱스에 따라서 색들이 골라지게 했다. 이때 아이템 블록에 차별성을 주기 위해서 현재시간에 따른 사인함수로 알파 값을 얻어 원래의 색과 은색을 조절하여 깜빡거리는 효과를 줬다. 여기서 더 나아가 모든 아이템 블록이 동시에 깜빡거리는 것이 아니라 랜덤하게 깜빡 거릴 수 있도록 현재 시간에 랜덤한 수를 더해서 모두 다른 타이밍에 깜빡 거리게 했다.

3. 스테이지 클리어시 깜빡거리는 안내문구

스테이지를 클리어하면 "Stage (스테이지 넘버) Cleared!!!"라는 문구가 뜨게 되는데 MyStartScene에 있는 "Press Space bar to Start" 문구와 동일한 원리로 깜빡이게 하여 사용자에게 재미를 제공한다.

4. 중앙 정렬 메서드

기능들을 구현하다보니 문구를 중앙에 띄울 일이 많았는데 이를 위해 대부분의 클래스에 midAlign이라는 메서드를 만들어서 사용했다. 이 메서드는 현재 사용자가 작성할 String의 가로폭을 측정해줘서 정확히 화면의 중앙에 놓일 수 있도록 해준다. 문자를 정확히 화면 중앙에 놓지 않으면 어딘가 이상한 느낌을 줄 수 있는데 이를 통해 해결했다.

5. Sound Manager 구현

앞서 클래스를 설명할 때도 설명했지만, 조금 더 자세히 설명하겠다. 영화에서도 배경음과 효과음 모두 중요하듯이 게임에서도 소리는 상당히 중요한 역할을 한다. 필자는 게임을 시작할 때 신나는 배경음을 재생하여 사용자가 게임을 하기 전 흥분할 수 있게 만들었고 스페이스 바를 누를 때 "딸깍" 소리가 나는 효과음을 넣어 사소한 디테일을 주려고 노력했다. 게임을 시작하면 긴박한 노래가 흘러 나오는데 이는 사용자의 긴장감을 높인다. 공이 아이템 블록과 일반 블록, 라켓과 충돌했을 때 모두 다른 효과음을 재생함으로써 타격감을 줄 수 있다고 생각했다. 게임이 종료가 되면 절망적인 효과음을 재생해서 사용자의 승부욕을 끌어올려 한 판 더 하고 싶게 만들도록 유도했다.

시행착오

1. 라켓 움직임 끊김

라켓을 그저 단순하게 방향키 왼쪽을 누르면 왼쪽으로 이동하고 오른쪽을 누르면 오른쪽으로 이동하게 하면 동시 키 입력이 안되고 급격하게 방향전환을 할 때 끊기는 느낌이 들어서 맘에 들지 않았다. 부드러운 움직임을 만들기 위해 공을 많이 들였는데 키가 눌리면 HashSet에 저장하여 동시에 양쪽키가 다 눌리면 라켓을 움직이지 않고 그렇지 않으면 눌린 방향으로 움직이게 했다. 또한 키 입력이 중단되면 HashSet에서 제거하는 방식을 사용했다. 이렇게 하니 양쪽 키를 동시에 누르게 되면 어느쪽으로도 움직이지 않고 정지해 있으며, 급격한 방향전환을 줄 때 끊기지 않고 부드럽게 이동했다.

2. 아이템 블록 파괴 시 쓰레드 멈춤 현상 및 새로운 공의 방향

아이템 블록이 파괴되면 새로운 공이 생성되어 게임 진행이 돼야 하는데 이상하게 아이템 블록만 파괴되면 쓰레드가 정지되는듯한 이상한 현상이 발생됐다. 필자가 판단한 바로는 공이 생성되는 순간 아이템블록과 충돌판단이 되어 공이 무한 생성되어 프로그램이 정지되는 것이 아닌가 생각했다. 이를 해결하기 위해서 충돌 당시에는 새로운 리스트에 공들을 추가하고(공 2개) run메서드에서 충돌검사를 완료한 후에 objs리스트에 리스트에 담긴 공을 추가하는 방식을 채택했다. 이렇게 하니 아이템 블록과 공이 충돌해도 이상이 생기지 않고 공이 성공적으로 3개로 나뉘었다.

이제 문제가 되는 것은 새로운 공들의 진행되는 방향이다. 무작정 방향을 랜덤으로 설정하기에는 물리법칙을 너무 무시하는 행동이기에 지양하려고 했다. 공과 아이템 블록이 충돌하면 아이템 블록이 충돌하는 순간 아이템 블록에 자신을 파괴시킨 공의 좌표과 vx,vy를 저장해서 새로운 공에 적용할 수 있도록 했다. 단 분산되는 효과를 줘야되기 때문에 원래 공의 vx,vy값을 atan2메서드에 넣어 각도를 역으로 뽑아냈고 이 각도에 첫번째 공은 10도 두번째 공은 20도를 추가하여 분산하는 효과를 줬다. 덕분에 공이 움직임이 조금은 자연스러워질 수 있었다.

3. 스테이지 클리어 시 공 생성

위와 연관되는 문제로 공이 아이템블록을 깼을 때 새로운 리스트에 추가하면서 생긴 문제다. 이상하게 아주 가끔 스테이지가 시작되자마자 블록들 사이

에서 공이 스폰돼 블록들이 다 깨져버리는 오류가 있었다. 조금 더 분석해보니 가장 마지막 블록이 아이템 블록이었을 때 이 문제가 발생했다. 다음 스테이지로 넘어가기 전에 objs 리스트는 초기화를 해줬지만 새로운 공들을 담고 있는 newBalls리스트는 초기화해주지 않아 스테이지가 시작되자마자 생성돼야 하는 위치에 생성되고 있던 것이었다. 스테이지가 종료될 때 newBalls.clear();를 해주면서 이 문제를 해결할 수 있었다.

4. 음원 재생 오류

해당 문제는 사소한 문제였는데 필자의 사용환경이 Mac OS인데 Window OS에서 실행할 때만 jar의 음원이 재생이 안되는 오류가 있었다. 인터넷을 찾아보니 파일의 음질에 관한 오류일 수 있다고 하여 기존에 24bit로 추출했던 음원을 16bit로 추출하니 이상없이 재생할 수 있었다.

결론 및 느낀 점

이번 과제를 통해 Java Swing 을 이용하여 하나의 완성도 높은 게임을 설계하고 구현하는 전 과정을 경험할 수 있었다. 처음에는 단순히 "공을 튕겨서 블록을 깨는" 로직만 구현하면 될 것 같았지만, 실제로는 씬(Scene) 전환 구조를 잡고, 키 입력과 충돌 처리, 그리고 스테이지 관리와 점수 체계, 사운드까지 고려해야 할 요소가 많았다. 특히 아이템 블록이 새 공을 생성하는 부분에서는 기존의 공들과 충돌 판정을 나누어 관리해야 해 쓰레드가 멈추는 등 예기치 못한 오류가 발생했으며, 이를 해결해나가는 과정에서 객체지향적인 설계와 게임 루프 구조에 대한 이해가 한층 높아졌다.

또한 그래픽 측면에서 그라디언트 페인트와 깜빡임 효과를 적용해 블록과 배경을 다채롭게 표현했고, 사운드 매니저를 통해 배경음과 효과음을 적절히 배치함으로써 게임의 몰입감을 높일 수 있었다. 무엇보다 프로젝트를 진행하며 화면 그리기와 충돌 판정, 속도와 각도 제어 등의 물리 로직을 종합적으로 다루게 되어, 게임 제작 과정 전반에 대한 감각을 익힐 수 있었다.

추후에는 마우스 입력이나 네트워크, 혹은 좀 더 정교한 충돌 판정 알고리즘(예: 픽셀 단위 충돌, 탄성 반사각 조절)을 추가해볼 수도 있을 것이다. 게임 내 아이템 종류를 늘리거나, 블록 파괴 시 다양한 이펙트를 더해 완성도를 높이는 작업도 가능하다. 이번 과제에서의 경험을 토대로, 앞으로 더 복잡한 프로젝트에서도 U, 이벤트, 쓰레드 관리를 유연하게 적용할 수 있을 것이라 기대한다.

→윈도우에서 실행 시 폰트가 바뀌는 문제가 생기는 것 같습니다. 폰트를 보여드리기 위해서 첫 장에 제가 의도한 폰트 보여드립니다. 감사합니다.