웹 프로그래밍 팀 프로젝트 과제 #1

<음악을 만들어 보자!>



202111307 서현하

202011266 김수연

201910162 문무현

201911191 안정근

제출일 : 2022-06-03

1. 게임 시나리오 및 게임 동작 방법

<*음악을 만들어 보자!>*

*시나리오 :*

유망한 음악가가 있었습니다.

재능이 넘치던 그는 언제나 좋은 평가를 받으며 자신감에 가득 차 있었습니다.

하지만 너무 빠르게 달렸던 탓일까요, 아니면 실패를 경험한 적이 없었기 때문일까요? 어느 순간부터 그는 연이은 혹평을 받으며 슬럼프에 빠집니다.

추락하던 그는 다시 날아오르기 위해 날개짓을 시작했습니다. 세간의 평가를 반전시키기 위한 역작을 만들고자 고민을 거듭하던 어느 날, 새로운 악상을 생각하던 중 잠에 들었습니다.

공교롭게도 꿈 속에는 구상 중이던 신곡의 악보와 그가 존경하던 전설적인 음악가들이 등장합니다.

꿈 속의 악보대로 곡을 완성할 수 있다면 명곡이 탄생할 것만 같습니다.

당신은 그의 꿈 속에 등장한 전설적인 음악가들이 되어 그를 돕게 됩니다.

이 꿈은 혹시 슬럼프를 극복할 기회가 아닐까요?

그가 다시 날아오를 수 있도록 함께 악보를 연주해주세요!

*게임 동작 방법 :*

음악을 만들어보자! 는 벽돌 깨기 게임입니다. 세 개의 난이도로 구성되어 있으며, 마우스를 따라서 움직이는 패들로 공을 받아내지 못할 경우 라이프가 하나 줄어듭니다. 세 개의 라이프를 모두 잃으면 패배하며, 패배한 경우 게임을 다시 시작하거나 메인 메뉴로 돌아갈 수 있습니다. 3단계까지 승리한 경우 승리 화면이 출력됩니다.

메인 메뉴

텍스트, 표지판이(가) 표시된 사진

자동 생성된 설명

메인 메뉴 화면은 제목과 다섯 개의 버튼으로 구성되어 있습니다.

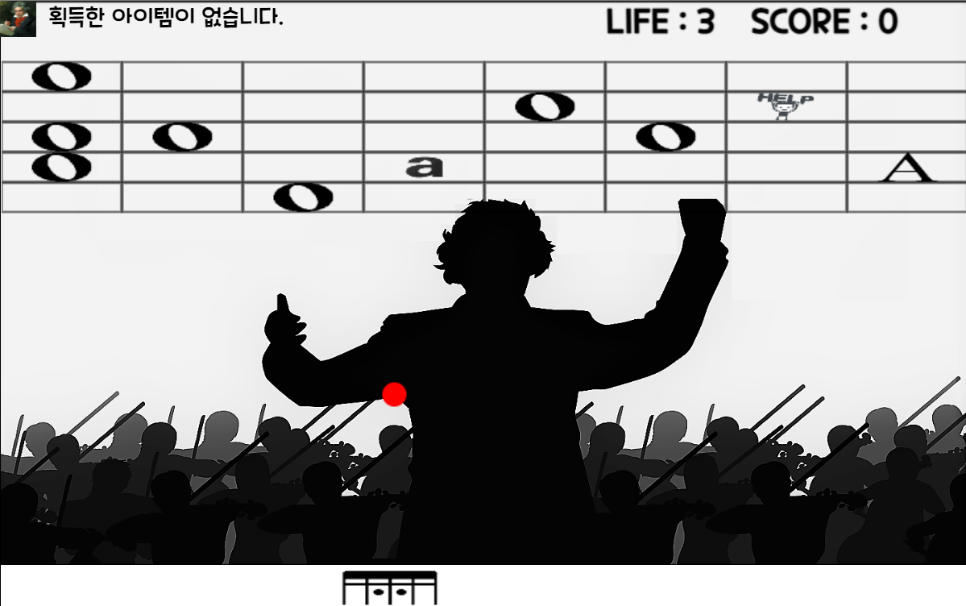
난이도 선택

텍스트, 사람, 남자, 정장이(가) 표시된 사진

자동 생성된 설명

게임 시작을 누르면 난이도 선택 화면으로 넘어가고, 난이도를 선택하면 게임을 시작합니다.

게임 화면



게임 정보

텍스트이(가) 표시된 사진

자동 생성된 설명

게임 정보는 게임 방법과 난이도 정보, 아이템 정보를 출력합니다.

점수 보기



점수 보기는 게임에서 패배하거나 승리한 경우에 기록된 점수를 출력합니다.

환경 설정



환경 설정에서는 공의 색과 벽돌의 모양, 음량을 조절할 수 있습니다.

*아이템 정보 :*

|  |  |
| --- | --- |
| 아이템 | 설명 |
| adagio | 공의 속도가 느려집니다. |
| largamente | 패들의 길이가 길어집니다. |
| 텍스트이(가) 표시된 사진  자동 생성된 설명double | 자동으로 움직이며 벽돌을 깨는 두번째 공과 패들이 생성됩니다. |
| help | 일정 시간 동안 자동으로 패들이 움직입니다. |
| 검은색, 옅은, 어두운이(가) 표시된 사진  자동 생성된 설명allegro | 공의 속도가 빨라집니다. |
| 실내, 블라인드이(가) 표시된 사진  자동 생성된 설명staccato | 공이 일정 주기로 점멸하여 잠시 동안 보이지 않습니다. |

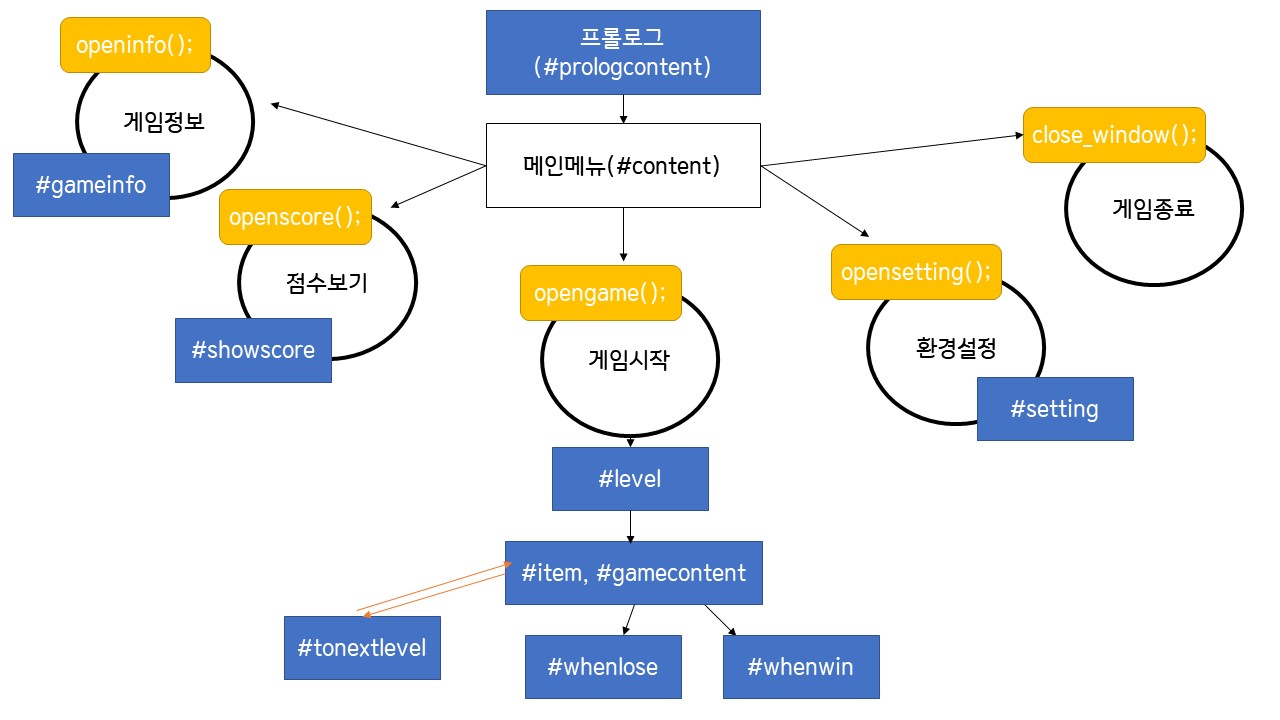
*점수 정보 :*

가장 왼쪽 줄에 있는 벽돌을 맞출 경우 10점, 그 외의 벽돌을 맞추면 1점을 획득합니다. 1, 2단계를 클리어하여 다음 단계로 넘어가는 경우에는 이전 난이도에서 획득한 점수가 누적됩니다. 게임에서 패배하거나 3단계까지 클리어한 경우 점수가 기록되며, 이전에 기록된 점수와 비교하여 9위보다 높을 경우 순위표가 갱신됩니다.

*난이도 정보 :*

드뷔시, 베토벤, 모차르트 난이도가 있습니다. 드뷔시 난이도가 가장 쉽고, 모차르트 난이도가 가장 어렵습니다. 난이도가 높아질수록 패들의 길이가 빨라지며, 벽돌이 더 많이 생성될 확률이 높습니다. 드뷔시 난이도는 공의 속도가 더 느리며, 모차르트 난이도는 벽돌이 좌우로 움직입니다.

1. 주요기능 및 기능 구현을 위한 기술적 상세 명세



게임 html 구조도는 위의 사진과 같습니다.

각 label에 원 클릭 이벤트가 연결되어 있고, 이후 각 div를 id를 통해 show(), hide()를 하며 게임을 진행시킵니다.

코드 설명 – 표 배경 색깔 회색 : html, 초록 : css, 흰색 : js

프롤로그

|  |
| --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| Html을 실행시키면 제일 먼저 $(document).ready를 통해 함수 pronext를 실행시킵니다. 프롤로그는 이미지와 텍스트 두 부분으로 이루어져 있습니다. 각 장면마다 다른 이미지와 텍스트가 존재하고 텍스트는 배열에 저장되어 있습니다. onclick이벤트가 등록된 다음 버튼을 통해 다음 장면으로 넘어갈 수 있습니다. |

댓글 효과

|  |
| --- |
| showcomment()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 2번째 장면에서 악플로 시달리는 음악가의 모습을 표현하기 위해 각각의 댓글에 수행시간이 다른 fade-in 효과를 넣었습니다. |

타이핑 효과

|  |
| --- |
| pronext(), typing()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 텍스트가 한번에 출력되는 것이 아닌 동적인 재미를 주기 위해 타이핑 효과를 추가하였습니다. 각 장면의 텍스트를 한글자씩 배열로 변수 text에 저장합니다. 변수 i는 text의 몇번째 index인지를 가리킵니다. 함수typing()에 interval을 적용시키면 출력되는 영역에 글자를 하나씩 추가하게 됩니다. text의 모든 글자를 추가했다면 interval을 종료시킵니다. |

스킵 버튼

|  |
| --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 보다 역동적인 버튼을 만들기 위해 위와 같은 css를 적용시켰습니다.  버튼이 hover되었을 때 버튼의 크기가 1.2배 커지고 그림자가 생깁니다.  버튼이 hover되면 새로운 내용이 버튼 태그 사이에 추가됩니다. 그 내용은 고리 모양이 퍼져나가 사라지는 애니메이션으로 translate(-50%,-50%)을 통해 버튼 정가운데에서 시작되도록 설정하였고 @keyframes로 애니메이션을 구체적으로 정의해주었습니다. |
| proskip() |
| 스킵 버튼을 누를 시 바로 메인 메뉴가 나오도록 설정했습니다. |

메인메뉴

|  |
| --- |
| 텍스트, 명판이(가) 표시된 사진  자동 생성된 설명 |
| 메인 화면에서 원하는 창으로 이동할 수 있게 하기 위해 <input>태그에 각기 다른 id를 준 button을 생성해주었습니다. 그리고 label의 for로 각각의 id를 연결해서 각 버튼마다 다른 이름을 가진 버튼이 나오도록 하였습니다. |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| Label에 float 속성을 부여해서 좌우로 나란히 배치되도록 하였고, hover 값을 설정하여 마우스가 hover되면 글씨색상과 배경색상이 변경되도록 하였습니다. |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 그 후 각 id에 onclick 이벤트를 달아주어서 버튼을 클릭하면 실행되길 원하는 함수를 연결해주었습니다. |
| opengame(), openinfo(), openscore(), opensetting()  텍스트이(가) 표시된 사진  자동 생성된 설명  텍스트이(가) 표시된 사진  자동 생성된 설명  텍스트이(가) 표시된 사진  자동 생성된 설명  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 그렇게 호출된 각각의 함수는 전에 display가 block 상태였던 <div>창을 display=none으로 변경시켜 보이지 않게 해주고, display가 none이었던 현재 보이게 하고 싶은 <div>화면의 display를 block 시켜줌으로써 화면에 보이도록 하였습니다.  Opensetting()의 경우에는 #setting을 pop up으로 띄우므로 전에 떠있던 #content의 display를 none으로 변경하는 게 아닌 투명도를 0.5로 변경해주었습니다. |

환경설정

|  |
| --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 환경설정창을 팝업으로 띄우기 위해서 <div id=”setting”>인 div에 popup이라는 class를 부여하였습니다.  그리고 box-shadow 속성을 부여하여 팝업의 뒷배경에 파란 그림자가 지도록 하였습니다. |

환경설정 : 공 색

|  |
| --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| setting()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 공 색을 선택하기 위해 수업시간에 배운 colorTable을 이용하였습니다.  Id=setup인 버튼을 클릭하면 실행되는 setting함수입니다. 그 안에 createElement와 setAttribute를 이용하여 ndiv에 class=ctbox의 속성을 부여하였습니다. 그리고 appendChild를 통하여 colorNames배열의 개수만큼 div를 붙여서 colortable을 만들어주었습니다. 그 후 this를 통해 어떤 div가 click되었는지 알아낼 수 있도록 하였습니다.  setting함수가 실행될 때마다 colortable이 늘어나는 것을 방지하기 위해 while(child[0])인 동안 colordiv.removeChild(child[0])을 이용하여 실행될 때마다 초반에 colortable을 지워주었습니다. |

환경설정 : 음표 벽돌 이미지 변경

|  |
| --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명  텍스트이(가) 표시된 사진  자동 생성된 설명  텍스트, 칠판이(가) 표시된 사진  자동 생성된 설명 |
| Img.src를 전역변수로 설정한 후 각 그림이 click되면 img.src를 변경하도록 하여 벽돌의 이미지를 변경하였습니다. |

환경설정 : 소리 조절

|  |
| --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 배경음과 효과음의 소리 크기를 바 모양으로 조절하기 위해 input의 type을 range로 주었습니다. |
| music\_volume(), effect\_volume()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 그리고 각각의 input에 변화가 있으면 music\_volume()과 effect\_volume() 함수에서 변화한 value값을 가져와 각각의 volume값을 재설정해주었습니다. |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 텍스트, 장치, 조종판이(가) 표시된 사진  자동 생성된 설명 |
| Setting 내에 X라는 버튼을 만들어서 클릭되면 #setting 창을 닫고 메인 메뉴로 돌아가도록 설정했습니다. |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 위치는 absolute로 오른쪽 위에 위치하도록 하였고, hover가 되면 글씨 크기를 크게하여 마우스가 위에 있다는 표시를 했습니다. |

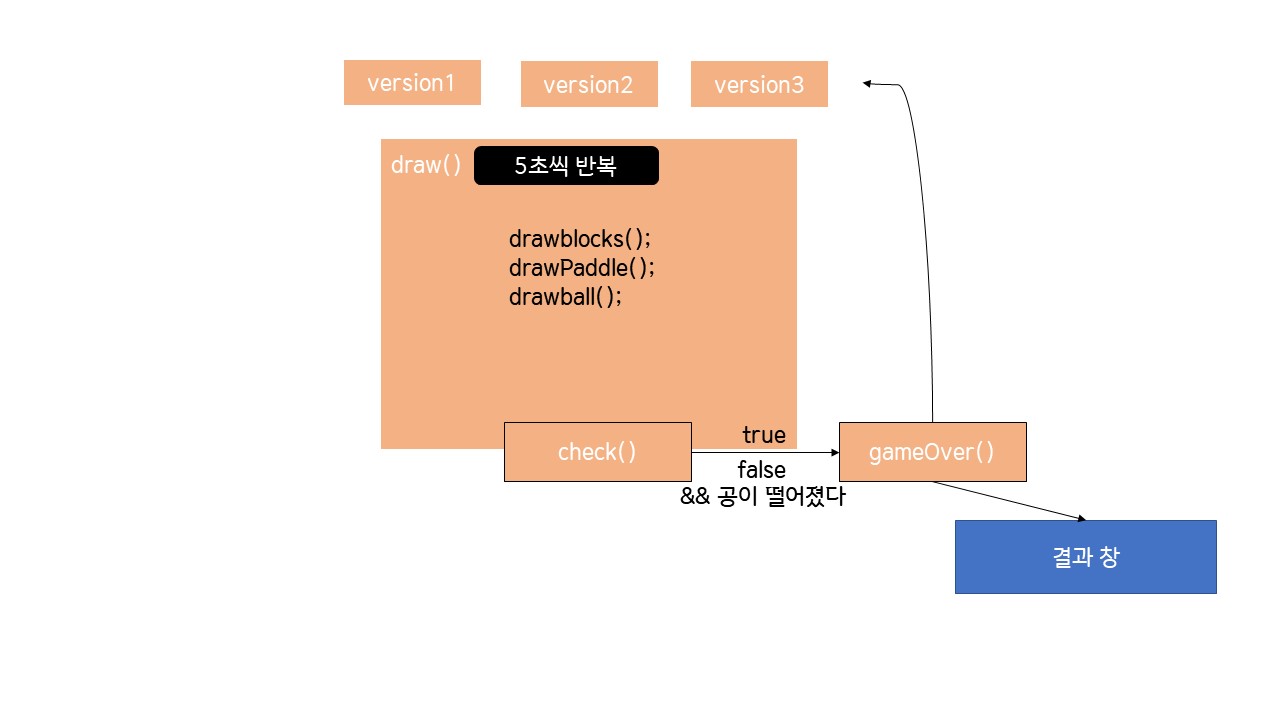
게임 종료

|  |
| --- |
| close\_window()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 게임종료 버튼을 클릭하면 close\_window함수가 실행되는데 그 안에서는 close()를 통해 화면을 꺼지도록 설정했습니다. |

난이도 화면

|  |
| --- |
|  |
| 난이도 별 선택 창  Selectlevel로 각 레벨 선택 창을 만들어주고 이미지와 텍스트를 배치했습니다. |
|  |
| 난이도 별 선택 창을 왼쪽 수평 정렬을 하고, 이미지 위에 텍스트를 위치시키기 위해 포지션 설정을 합니다.  Selectlevel 에 상대적으로 level text를 하단에 위치시켜야 하므로 selectlevel은 relative, leveltext는 absolute 포지션을 설정해줍니다.  이미지 width는 260px이고 body width는 800이므로 800-260\*3 = 20  따라서 중간 레벨 양쪽에 패딩을 10씩 줘서 body width에 딱 맞게 값을 줬습니다.  반투명한 텍스트 배경 색상을 주기 위해 rgba를 이용해 설정해줬습니다. (Rgba의 4번째 값은 투명도) |
|  |
| Selectlevel 선택 시 게임 화면과 상단 item 화면 show()  각각 선택한 난이도에 따라 선택한 버전의 게임 함수를 실행시킵니다. |
| version1(), version2(), version3() |
|  |
| 난이도에 따라 패들의 너비, 마디 수 (COLS), 공 속도를 지정해줍니다.  마디수가 난이도에 따라 달라지므로 난이도 선택 후 블록들 개수를 지정하고 기본 세팅을 하는 함수 blockset()을 호출합니다.  난이도별로 다른 배경이미지를 출력하기 위해 jqery로 css를 접근하여 지정해줍니다.  음악과 음악가 이미지도 난이도별로 지정해준 후 draw함수를 실행시켜 실제 게임을 진행시킵니다.  제일 높은 난이도(version 3)를 선택한 경우 movingspeed = 1을 통해 악보가 움직일 수 있도록 설정합니다. |

벽돌깨기 게임 기능



기본 게임 구현 구조도 입니다. 5초마다 게임의 기본 사항인 벽돌, 패들, 공을 그리는 함수를 호출합니다. 공의 위치에 따라 벽돌이 다 깨졌는지 확인하는 check() 함수를 통해 값에 따라 gameOver() 함수를 호출한 뒤 현재 난이도에 따라 다음 난이도로 넘어가거나 결과 창을 띄웁니다.

게임 페이지 로드

|  |
| --- |
| bodyposition() |
| 페이지가 로드되면 body와 팝업창의 위치를 정중앙에 설정합니다.  Jquery를 이용해 현재 윈도우의 너비와 높이를 가져와 정중앙에 둡니다. |

벽돌 그리기

|  |
| --- |
| blockset() |
| JS에서는 이차원 배열을 C처럼 한 번에 만들 수 없습니다.  <https://gent.tistory.com/296>를 참고하여, 이차원 배열을 생성하는 함수입니다. Math.random()을 활용해 0~4까지의 난수를 발생시켜 이차원배열에 값을 저장합니다.  0이면 벽돌,  아니면 빈 마디. |
| drawblocks() |
|  |
| drawblocks() 함수는 draw() 함수에서 호출되며 벽돌과 아이템을 그리는 기능을 합니다. blocks 이차원 배열에서 벽돌은 0으로, 여섯 개의 아이템은 각각 10에서 15로 정의되어 있습니다. 해당하는 숫자에 맞는 이미지를 그리며, 3단계인 경우 movingBlocks와 movingSpeed 변수에 값이 할당되어 함수가 호출될 때마다 이미지의 위치가 달라지게 됩니다. |

패들

|  |
| --- |
|  |
| 이벤트 리스너를 통해 등록된 mouseMoveHandler 함수는 사용자의 마우스가 움직일 때마다 호출됩니다. paddleX 값은 패들을 그리는 시작 위치입니다. 사용자의 마우스의 수평 좌표에서 캔버스 왼쪽 끝의 값을 빼서 캔버스 상의 상대적인 위치를 구한 후, 그 값을 paddleX에 대입하여 패들이 사용자의 마우스를 따라 움직이도록 합니다. 또한 캔버스 범위 밖으로는 패들이 마우스를 따라 나가지 않도록 범위를 제한합니다. |
| drawPaddle() |
| 도돌이표 이미지가 패들로 사용되도록 drawImage를 이용해 캔버스에 출력한다.  이미지를 캔버스에 그리기 위해서는 Image 생성자를 통해 새로운 image를 생성해야 한다. 이미지 링크와 x, y, 너비, 높이 값을 drawImage함수에 할당해준다. |

공 그리기

|  |
| --- |
| drawball() |
| 캔버스에 원을 그리는 함수입니다. 반복 호출해야 하므로 함수로 구현했습니다.  환경설정에서 지정한 공 색깔에 따라 fillStyle을 변경할 수 있도록 합니다. |

캔버스에 게임 그리는 draw() 함수

|  |
| --- |
|  |
| 공의 흔적이 남지 않도록 clearRect를 해줍니다.  공의 위치와 방향은 dx와 dy를 통해 조절합니다.  가장 높은 난이도를 설정하면(version == 3) 악보가 움직일 수 있도록 movingBlocks의 값을 통해 왼쪽, 오른쪽 이동할 수 있게 movingSpeed 값을 변경시켜줍니다. |

충돌판정

|  |
| --- |
| (draw()) |
| 상, 좌, 우, 그리고 패들에 부딪혔을 때 / 벗어났을 때로 나뉘어서 충돌을 판정합니다.  먼저 공의 radius(BALLRADIUS) 값 때문에 x,y 값만으로 판정하면 범위에 들어가지 않는 경우가 있으므로 모서리에 닿았을 때도 추가하여 판정해야 합니다.  위쪽 모서리와 패들 모서리에 닿았을 때는 dx와 dy모두 바꿔주었습니다.  상, 좌, 우에 닿았을 때는 반대 방향으로 갈 수 있도록 dx나 dy의 부호를 바꿉니다.  패들의 왼쪽에 닿았을 때는 왼쪽으로 튕겨나가게, 패들의 오른쪽에 닿았을 땐 오른쪽으로 튕겨나가게 dx와 dy의 부호를 조정합니다. |
|  |
| 공이 하단에 있는데 패들의 범위에 있지 않을 땐 떨어진 것입니다. 이 때도 모서리까지 체크해줍니다. 패들의 범위에 있지 않았을 때도 공은 똑같이 튕기게 되며, 깨지는 소리가 납니다.  다만, 패들 밖에 떨어지면 생명 변수인 LIFE가 감소합니다. LIFE가 0이 되면 게임에 지게됩니다. 게임에 지면 게임 결과 창을 show()하고, 게임창은 hide() 합니다. |
|  |
| 공이 하나 더 생기는 아이템을 먹었을 때, 두번째 공의 충돌을 판정합니다.  다른 사항은 모두 위와 같지만 패들의 오른쪽과 왼쪽을 구분하지 않습니다. 왜냐하면 두번째 공은 언제나 떨어지지 않으며 패들의 중간부분에 맞게 되기 때문입니다. |

게임 판정과 점수 계산

|  |
| --- |
| check() |
| 이차원배열의 값을 확인해 벽돌이 없으면 true, 하나라도 있으면 false를 리턴합니다.  이 때, 행 순서대로 체크하지 않고 열 순서대로 체크합니다. 왼쪽 마디부터 음표 벽돌을 깼는지 확인하기 위해서입니다.  flag에 음표 벽돌이 있는 col을 저장해 추후 사용합니다. |
| (draw()) |
| check값이 true면 벽돌을 다 깼다는 의미입니다.  다음 난이도로 자동으로 넘어가도록 version을 1씩 증가시킨 후 다음 단계로 넘어간다는 팝업창을 show()합니다.  만약 3단계를 깼다면 현재 게임창과 아이템 창을 hide하고 이겼다는 내용이 담긴 #whenwin div를 show() 합니다. 이 때, 랜덤으로 설명 텍스트를 출력합니다.  팝업창의 확인 버튼을 누르면 팝업창을 hide()하고 게임창을 show()한 뒤 다음 단계의 함수를 실행시킵니다. |
|  |
| 캔버스를 음표벽돌의 크기만큼 나눈 다음, 공이 현재 어디에 위치해 있는지 계산합니다. 캔버스 전체가 거대한 이차원 배열이 됐다 생각합시다. (=캔버스 배열)    y-MARGIN을 해서 상단 생명과 스코어, 아이템 공간은 포함하지 않습니다.  y-MARGIN을 벽돌 높이와 패딩으로 나눈 후 Math.floor()로 현재 공의 위치가 캔버스 배열의 몇 행인지 계산합니다.  X를 벽돌 너비+패딩으로 나눈 후 Math.floor()로 현재 공의 위치가 캔버스 배열의 몇 열인지 계산합니다. |
| (draw()) |
| 계산했을 때 공의 위치가 벽돌이 그려지는 악보에 포함되는지 if(row < ROWS)로 확인합니다.  현재 공의 위치가 벽돌이 그려진 (blocks[][] == 0) 위치에 해당하면 배열 값을 1로 바꿔주고 공의 방향을 바꿔줍니다.  모서리에 있다면 x와 y 방향 모두 바꿔주고, 아니면 y방향만 바꿔줍니다.  check() 함수에서 계산한 flag와 col이 같다면 존재하는 음표들 중 왼쪽 마디부터 음표 벽돌이 깨진 것이므로 SCORE를 10씩 증가시킵니다.  flag와 계산한 col이 다르면 왼쪽 마디부터 음표 벽돌이 깨진 것이 아니므로 SCORE를 1씩 증가시킵니다. |

점수

|  |
| --- |
|  |
| 게임에서 승리하거나 패배할 경우 gameOver() 함수가 호출됩니다. 3단계를 클리어했거나, 게임에서 패배한 경우에 leaderBoard 배열에 점수를 추가하고, 내림차순으로 정렬합니다.  openscore() 함수는 메인 메뉴에서 점수 보기를 선택했을 경우 호출됩니다. leaderBoard 배열이 비어있을 경우에는 “아직 순위가 없습니다!” 를 출력하고, 그렇지 않은 경우에는 최대 9개의 점수를 출력합니다. |

게임 결과 창

|  |
| --- |
|  |
| 게임 결과 div이다. 3행 1열 테이블로 구현해 텍스트 출력, 이미지 출력, 버튼 출력으로 구역을 나누었습니다.  졌을 땐 게임 다시하기, 메인 메뉴 버튼을 띄웁니다.  이겼을 땐 등수 보러가기, 메뉴로 돌아가기(메인 메뉴) 버튼을 띄웁니다. |
|  |
| 게임 결과에 따라 출력되어야 하므로 일단 display: none; 으로 설정해줍니다.  예쁘게 출력되기 위해 패딩 값을 상 우 하 좌 값을 설정해주었습니다. |

메뉴로 돌아가기

|  |
| --- |
| backtomain() |
| show속성이 바뀌어지는 div를 다 hide()처리한 후 메뉴만 show() 합니다. |

게임 다시하기

|  |
| --- |
| replaygame() |
| 게임창과 아이템창을 show()하고, 현재 창이었던 게임 결과창을 hide()한다.  버전 변수를 체크하여 버전에 맞는 게임을 실행시킨다. |

아이템

|  |
| --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 음표블록이 2차원배열을 통해 구현되었으므로 2차원배열의 데이터의 10~15를 아이템공간으로 활용하였습니다.  아이템이 중복으로 생성되는 것을 방지하기 위해 배열에 각 아이템 data를 저장하였고 양수, 음수에 따라 오름차순, 내림차순으로 정렬되는 것을 이용한 shuffle알고리즘을 사용하였습니다.  이렇게 랜덤하게 섞인 아이템배열을 음표, 아이템블록이 없는 곳에 앞에서부터 차례로 넣어주어 아이템이 랜덤으로 생성되도록 구현하였습니다. |
| 텍스트, 명판이(가) 표시된 사진  자동 생성된 설명 |
| 공이 각각의 아이템들을 만나게 되면 배열에 저장된 아이템 데이터 값에 따라 아이템함수를 실행합니다. |
|  |

아이템 : 속도 adagio, allegro

|  |
| --- |
| adagio(), original()  텍스트이(가) 표시된 사진  자동 생성된 설명텍스트이(가) 표시된 사진  자동 생성된 설명 |
| adagio 아이템을 획득하면 잠시 속도가 느려지고 5초뒤 원래의 속도로 돌아오게 됩니다. 현재 난이도를 체크하여 난이도 별로 기본 속도보다 하나씩 느리게 만듭니다. |
| allegro()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| allegro 아이템을 획득할 시 속도가 잠깐 빨라집니다. 일정시간 이후 원래 속도로 돌아오게 됩니다. |

아이템 : double

|  |
| --- |
| double()텍스트이(가) 표시된 사진  자동 생성된 설명 |
| 유저가 조종할 필요가 없는 두번째 공을 생성합니다. 전역변수인 secondball의 값을 true로 변경시켜 drawball2() 함수와 drawPaddle2() 함수가 작동되도록 합니다.  drawPaddle2() 함수에서는 임의의 패들의 위치를 두번째 공의 위치를 기준으로 설정하여 패들이 공을 따라가는 듯한 효과를 주었습니다. |

아이템 : largament

|  |
| --- |
| largamente()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| largament아이템을 획득할 시 패들의 크기가 2배로 커지고 5초 뒤 원래 크기로 돌아옵니다. |

아이템 : help

|  |
| --- |
| help()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| Help 아이템을 획득할 시 유저가 패들을 조작할 수 있게 해주는 mousemove이벤트를 제거하고 패들의 위치를 공의 위치를 기준으로 설정하여 패들이 공을 따라갈 수 있도록 설정하였습니다. |

아이템 : staccato

|  |
| --- |
| staccato()  텍스트이(가) 표시된 사진  자동 생성된 설명 |
| staccato 아이템을 획득할 시 두 개의 interval을 시간차를 두어 실행합니다. 전역변수 ballvisible은 true일 때 캔버스에 공을 그리고 false일 때 drawball()이 실행되지 않아 캔버스에 공이 그려지지 않게 됩니다.  캔버스에 공이 그려지지 않아도 공의 위치를 전역변수를 통해 저장되어있으므로 공은 없어지지 않습니다. |

1. 토의사항
2. 팀프로젝트 진행에 있어서

먼저 git처럼 서로가 수정한 부분을 쉽게 파악할 수 있고, 버전을 관리해주는 프로그램을 활용하지 않은 것에 대한 아쉬움을 느꼈습니다. 서로가 맡은 부분을 코딩하여 압축파일을 카톡으로 보내는 방식으로 협업을 하였습니다. 파일을 받으면 어느 부분이 변경되었는지, 오류가 발생하였을 경우에는 어디서 잘못된 것인지 혼자서 찾고 파악해야 했습니다. 또한, 같은 버전을 놓고 동시에 작업한 결과가 올라왔을 때, 서로 다른 파일을 합쳐야 했습니다. 어렵지는 않았지만, 시간을 많이 소모하는 일이었습니다. 이번 프로젝트를 진행하면서 이러한 문제를 해결하기 위한 프로그램의 필요성을 느끼고, git의 사용법을 공부하게 할 동기가 되었습니다. 프로젝트를 시작할 무렵에는 그 존재는 알고 있었지만 그 유용성은 깨닫지 못했고, 활용법을 배우고자 하는 의지가 없었습니다. 이제는 다음에 다시 누군가와 협업을 할 일에 대비하여 공부하고 싶습니다.

둘 째, 각자의 작업 영역이 겹치지 않도록 기능을 분담하는 것의 중요성을 느꼈습니다. 코드를 작성하기 전에 서로의 작업 영역이 겹치지 않는지, 또는 결과를 쉽게 합칠 수 있는지를 고려하여 역할을 나누는 것의 중요성을 느꼈습니다. 그러기 위해서는 코드를 작성하는 것에 앞서 구현해야 하는 기능들을 명확히 하고, 적절하게 분담해야 할 것입니다. 이번 프로젝트를 진행하면서 처음에는 각자 역할을 나누고 시작했습니다. 하지만 작업이 진행되는 동안 각자 필요하다고 생각하는 부분을 추가하거나 수정하면서, 같은 부분이 동시에 수정되는 등의 일이 있었습니다. 맡은 기능과 역할을 명확히 하고, 지속적으로 소통하는 것이 중요하다고 느꼈습니다.

마지막으로, 코드를 정리하고 주석을 작성하는 것의 필요성을 느꼈습니다. 서로의 파일을 계속 합치다 보니 점점 코드의 순서나 변수의 선언 위치 등이 뒤죽박죽 섞이게 되었고, 가독성이 떨어지며 코드의 의도를 파악하는 것도 어려워졌습니다. 다행히 중간부터 팀원들이 파일을 합치는 과정에서 코드를 정리하고, 주석을 작성하며 이러한 어려움을 조금 해결할 수 있었습니다. 다른 사람들과 협업을 진행할 때는 항상 누가 보았을 때 쉽게 이해할 수 있도록 최대한 배려하며 코드를 작성해야 하겠다고 생각했습니다.

1. 기능 구현에 있어서

벽돌 깨기 구현을 할 때 공의 진행 방향을 마음대로 조절할 수 있도록 패들의 왼쪽, 오른쪽에 공이 닿았을 때의 진행 방향을 구분하였습니다. 더 나아가 패들에 맞는 각도를 조절하고, 가운데에 맞으면 수직으로 올라가는 등의 기능은 구현하지 못하였습니다. 공 방향을 세밀하게 변경시키는 시도를 해보았으나 좋은 방법을 떠올리지 못했었는데, 여러 개의 조건문을 추가하여 패들을 10개의 구역으로 나눈다거나, 패들이 공에 맞았을 때의 위치를 통해 수학적으로 진행 방향을 하는 등의 방법들이 있을 것 같습니다.

또한, 게임 도중에 게임을 멈추고 환경설정 창이 뜨게 되면 공의 속도가 빨라지는 문제점이 있다는 것을 발견하였습니다. Interval을 환경설정 창이 뜰 때마다 clearInterval() 해줬다가 창을 끄면 다시 setInterval()하는 방법을 생각해서 코드를 짜보았지만, 공의 속도는 여전히 계속 빨라 졌습니다. 아마 다른 부분에서 Interval의 중첩이 더 일어났던 것 같습니다.

그 외에도 공이 빨라지는 경우들이 있었지만, 그러한 오류들을 해결해나가며 게임을 완성해가는 성취감이 있었습니다. 지금까지 배운것으로 이러한 게임을 만들 수 있다는 것이 재미있었고, 다음에 혼자서 다른 프로젝트를 해보아야겠다고 생각했습니다.

1. 기여도 상세 기술

서현하 25%

* 게임 화면 – 공, 공 방향, 랜덤 벽돌, 패들 이미지, 생명∙점수 판정 및 출력, 충돌 판정, 승리 확인
* 난이도 – 난이도 선택 창, 난이도 별 bgm 출력 및 배경 이미지 변경
* 게임 결과 – 게임 결과 창 구현, 랜덤 텍스트 출력
* css 정리 – 중앙 정렬, 폰트, 이미지 출력 사이즈 등

김수연 25%

* 게임 초반 디자인
* 메인메뉴-게임창 위치 설정, 메인메뉴 버튼 기능
* 환경설정-공색깔, 벽돌모양 변경, 배경음/효과음 소리크기 조절
* 게임성공/실패시-다음단계 팝업 창, 다시하기/등수보기 버튼, 게임점수 업데이트 조건 설정
* 게임종료 버튼 추가

문무현 25%

* 게임 기획
* 패들, 벽돌(3단계) 움직임
* 게임 정보, 점수 보기
* 게임 시작, 종료 시 함수(init(), gameOver())
* 효과음, 배경 음악, 배경 이미지 소스 찾기 및 연결
* css 정리 – 메뉴로 돌아가기 버튼, 메인화면 디자인 등
* 코드 정리

안정근 25%

* 게임 기획
* 게임 – 모서리 충돌 판정
* 프롤로그-댓글 효과, 타이핑 효과, 동적인 스킵버튼
* 아이템-속도변화, 독립적인 공 생성, 패들 길이 조절, 패들 자동 조종, 공의 점멸 효과
* 코드 정리