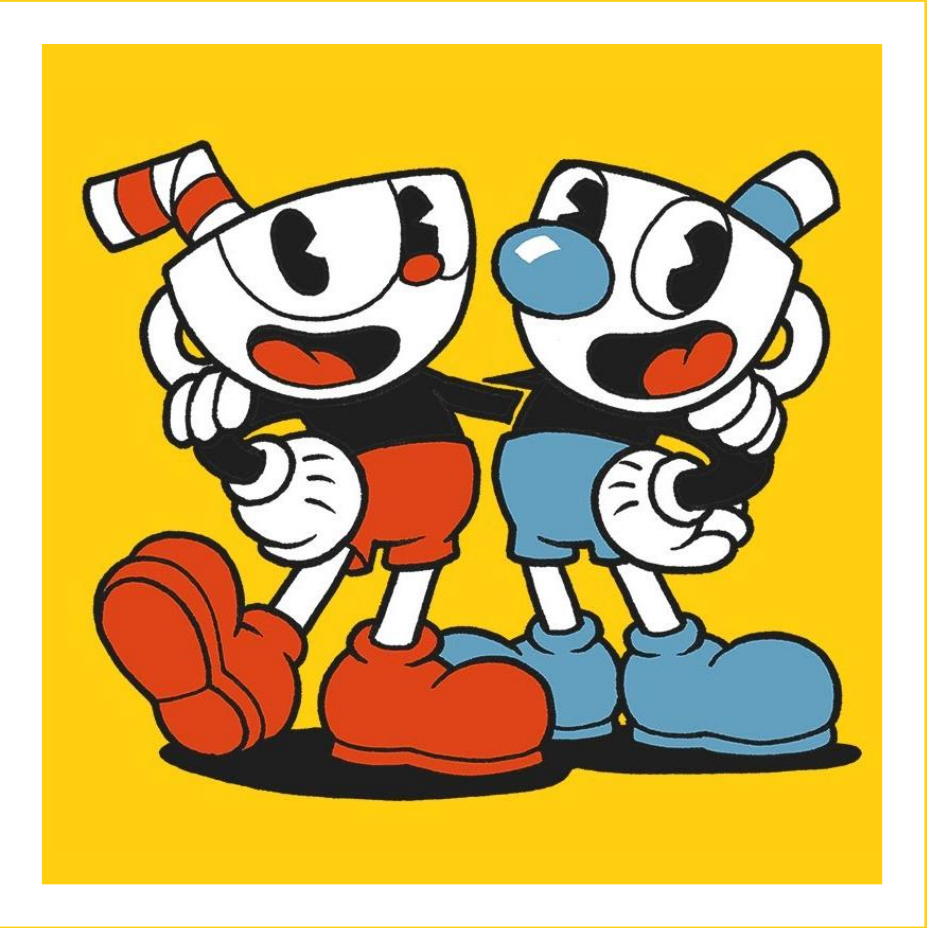


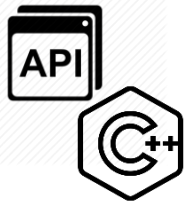
CUP HEAD

MADE BY. 송우현

INDEX

기획의도

사용된 기술



C/C++, Window API

C++ 언어
Window API

01
사용언어

Visual Studio

솔루션관리
디버깅, 빌드

02
디버깅

03
사진편집

Photo Shop

각각의 이미지 편집

Ps

04
SPRITE 편집

Illustrator

애니메이션을 위한 SPRITE 편집

Ai

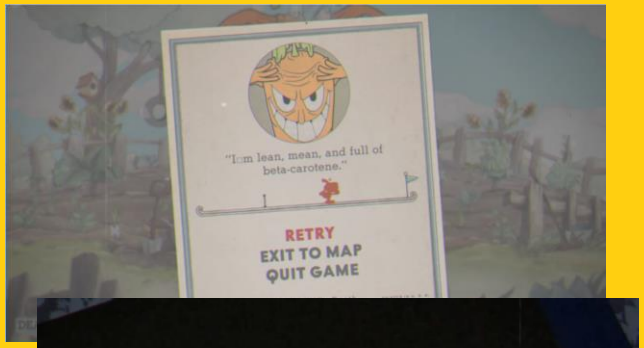
STAGE

CupHead의 게임진행 방식처럼 적의 공격을 피하며 총알을 발사해 적을 맞추는 게임

STAGE

```
void Game::DrawAll(HDC hdc)
{

    switch (m_StageNum)
    {
    case INTRO:
        DrawIntro();
        DrawFX();
        break;
    case STAGE1:
        DrawBG();
        DrawEnemy1(hdc);
        DrawBG2();
        DrawHeroBullet(hdc);
        DrawHero(hdc);
        DrawHeroBulletExplode();
        DrawEnemy1Bullet();
        DrawEnemy1BulletExplode();
        DrawBG3();
        DrawScore(hdc);
        DrawFade();
        DrawFX();
        break;
    case STAGE2:
        DrawBG();
        DrawEnemy2(hdc);
        DrawBG2();
        DrawHeroBullet(hdc);
        DrawHero(hdc);
        DrawHeroBulletExplode();
        DrawEnemy1Bullet();
        DrawEnemy1BulletExplode();
        DrawBG3();
        DrawScore(hdc);
        DrawFade();
        DrawFX();
        break;
    case CLEAR:
        DrawClear();
        DrawFX();
```



프로그래밍으로 정해준 순서에 따른 진행방식

UI / UX

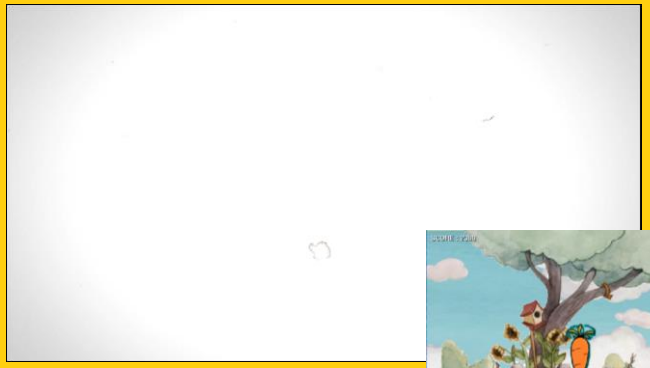
캐릭터의 상태에 따른 각각의 애니메이션 호출

UI / UX

```
//Alpha
m_Alpha[0].hbmp = (HBITMAP)LoadImage(NULL, L"image//base.bmp", IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE);
GetObject(m_Alpha[0].hbmp, sizeof(BITMAP), &m_Base.bit);

m_Alpha[0].dc = CreateCompatibleDC(hdc);

m_Alpha[0].holdbmp = (HBITMAP)SelectObject(m_Alpha[0].dc, m_Alpha[0].hbmp);
m_Alpha[0].bf.AlphaFormat = 0;
m_Alpha[0].bf.BlendFlags = 0;
m_Alpha[0].bf.BlendOp = 0;
m_Alpha[0].bf.SourceConstantAlpha = 127;    //0~255  127은 반투명임
```



Alpha 블랜딩을 이용한 시네마틱효과 연출

UI / UX

```
void Game::InitEnemy1Bullet(int i)
{
    //srand((unsigned int)time(NULL));          // 총알2개가 동시에(같은시간)에 초기화되면 무한루프

    m_uEnemy1Bullet[i].rt.left = ((XRES + 1) / 61)*(rand() % 61 + 1);       //61배수의 랜덤수
    m_uEnemy1Bullet[i].rt.right = m_uEnemy1Bullet[i].rt.left + 61;
    m_uEnemy1Bullet[i].rt.top = 0 - (YRES / 105) * (rand() % 105 +1);       //105배수의 랜덤수
    m_uEnemy1Bullet[i].rt.bottom = m_uEnemy1Bullet[i].rt.top + 105;

    if (InitOverLap(i))     // x,y 좌표가 겹친다면 다시좌표초기화
    {
        return;
    }
    else
    {
        InitEnemy1Bullet(i);
    }
}


bool Game::InitOverLap(int i)       //x좌표가 서로같은게 있다면 false, 아니면 true
{
    for (int j = 0; j < i ; j++)
    {
        if (m_uEnemy1Bullet[i].rt.left == m_uEnemy1Bullet[j].rt.left)
            return false;
        else
            true;
    }
    for (int j = i + 1; j < ENEMYBULLETNUM; j++)
    {
        if (m_uEnemy1Bullet[i].rt.left == m_uEnemy1Bullet[j].rt.left)
            return false;
        else
            true;
    }
}
```

Rand함수로 무작위로 겹쳐지지 않게 떨어지는 적 총알 구현

UI / UX

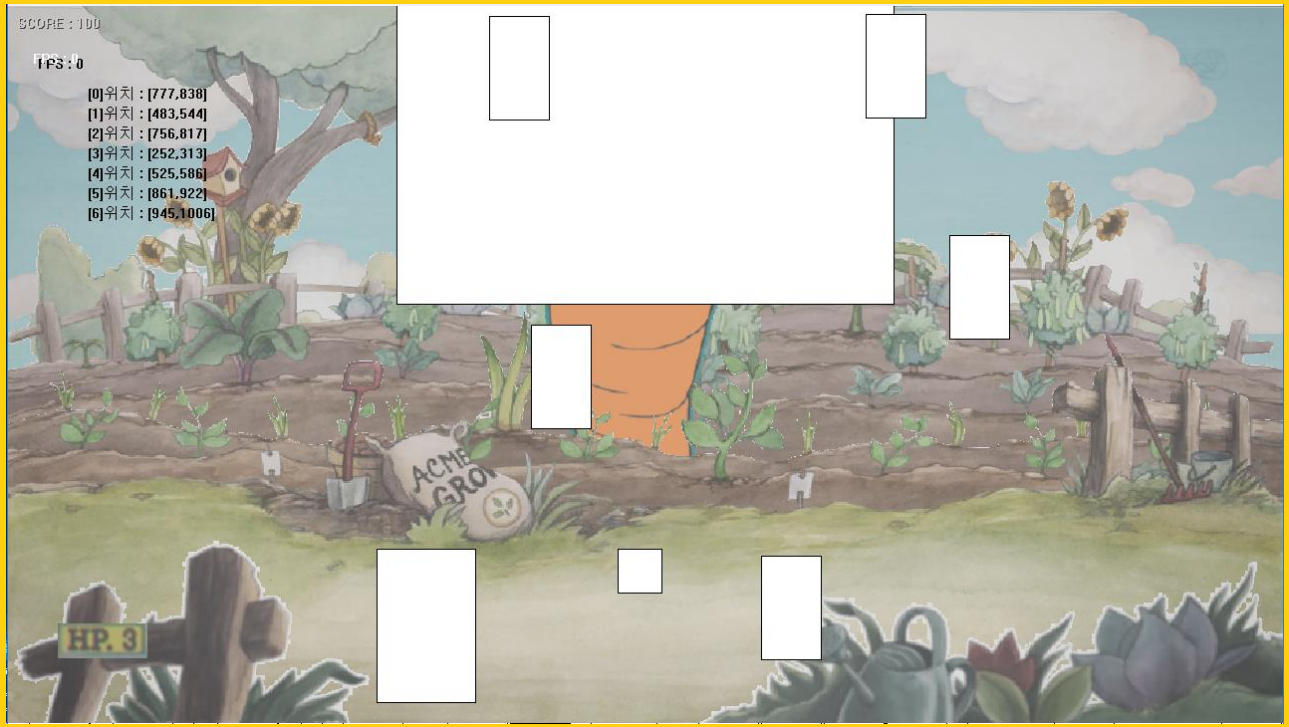
```
void Game::AttariCheck()
{
    POINT pt;
    for (int i = 0; i < HEROBULLETNUM; i++)          //히어로총알 vs 적
    {
        pt.x = m_uHeroBullet[i].rt.left;
        pt.y = m_uHeroBullet[i].rt.top+200;
        //피격됨
        if (PtInRect(&m_uEnemy1[0].rt, pt) != FALSE)
        {
            InitHer
            m_uHero
            m_nScor
            m_uHero
            if (m_n
            {
                m_S
                m_E
                m_u
                m_u
                //2
            }
            if (m_n
            {
                m_S
            }

        }
    }
```

```
    for (int i = 0; i < ENEMYBULLETNUM; i++)        //적총알 vs 히어로
    {

        RECT rcTemp;
        //피격됨
        //if (PtInRect(&m_uHero[0].rt, pt) != FALSE)
        if (IntersectRect(&rcTemp, &m_uHero[0].rt, &m_uEnemy1Bullet[i].rt) != FALSE)
        {
            m_uEnemy1BulletExplode[0].bTrigger = TRUE;              //폭발 이펙트 활성화
            m_uEnemy1BulletExplode[0].rt.left = m_uEnemy1Bullet[i].rt.left;
            m_uEnemy1BulletExplode[0].rt.top = m_uEnemy1Bullet[i].rt.top;
            m_nHeroHit += 1;
            m_nHeroHealth -= 1;
            m_uEnemy1BulletExplode[0].test++;
            //DrawEnemy1BulletExplode(m_uEnemy1Bullet[i].rt.left, m_uEnemy1Bullet[i].rt.top);
            //DrawEnemy1BulletExplode(m_uHero[0].rt.left, m_uHero[0].rt.top);
            DrawEnemy1BulletExplode();
            m_uHeroBullet[i].rt.top += 10;
            m_uHeroBullet[i].rt.bottom += 10;
            if (UpdateCheck(m_uHeroBullet[i], 1000) == true)
            {
                m_uHeroBullet[i].rt.top -= 10;
                m_uHeroBullet[i].rt.bottom -= 10;
            }
            if (m_nHeroHealth < 1)
            {
                m_StageNum = GAME_OVER;
                //스테이지끝나면 초기화 해야될것 은 updateclear에서
            }
            InitEnemy1Bullet(i);//총알초기화

        }
```

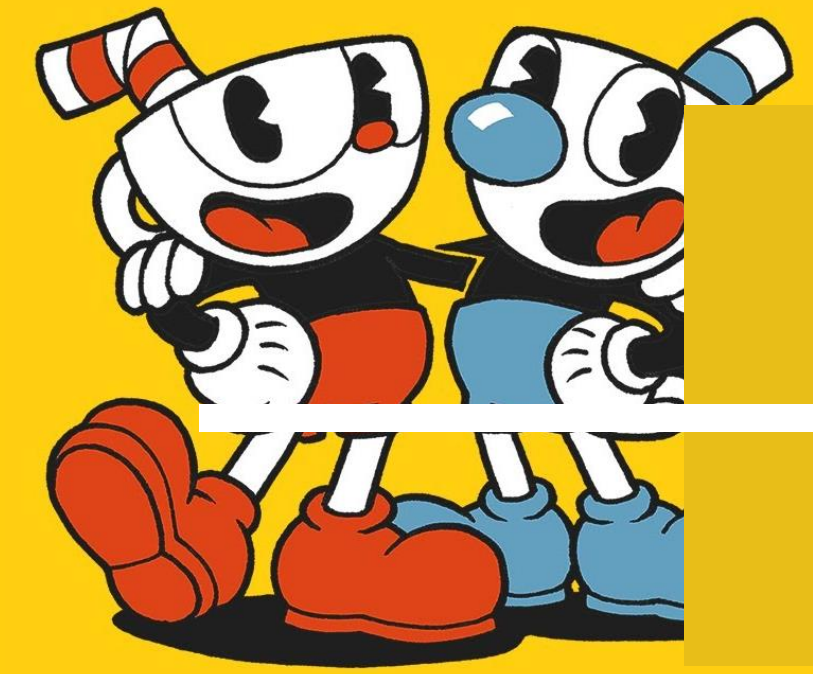


F3을 눌러서 디버깅모드로 충돌범위 검토

AttariCheck함수로 총알 / 적 / 히어로 사이의 사각형모양 충돌범위를 계산

플레이 영상

플레이 영상



THANK YOU!

MADE BY. 송우현