메탈카드봇
듀얼아레나
(국가지원사업프로젝트)

메탈카드봇 듀얼아레나

- 플랫폼 : PC(아케이드)
- 제작툴 : Unity,Visual Studio
- 언어 : C#
- 프로젝트인원 : 6명
- 담당분야 : 메인프로그래머
- 프로젝트 설명 : 지류카드를 이용한 아케이드 게임

능력치 룰렛


```
/// <summary>
/// 룰렛타이머 코루틴
/// </summary>
/// <returns></returns>
참조 6개
IEnumerator CardSetCor(bool _isReuIst = false)...
/// <summary>
/// 룰렛 코루틴
/// </summary>
/// <param name="_t"></param>
/// <returns></returns>
참조 1개
IEnumerator MoveCor(Transform _t, Vector2 _last)...
/// <summary>
/// 룰렛을통해 움직이는 오브젝트
/// </summary>
/// <param name="_rect"></param>
/// <returns></returns>
참조 1개
IEnumerator MoveObject(RectTransform _rect)...
/// <summary>
/// 멈출 위치
/// </summary>
/// <param name="_rect"></param>
/// <param name="_last"></param>
/// <param name="c"></param>
/// <returns></returns>
참조 1개
IEnumerator StopObject(RectTransform _rect, Vector3 _last, Coroutine c)...
/// <summary>
/// 진행 눌렀을때
/// </summary>
/// <returns></returns>
```

스파인 애니메이션


```
/// <summary>
/// 스파인 사운드이벤트 추가
/// </summary>
/// <param name="trackEntry"></param>
/// <param name="e"></param>
참조 2개
private void SoundPlay(TrackEntry trackEntry, Spine.Event e)...
/// <summary>
/// 스파인 진행완료 체크
/// </summary>
/// <param name="trackEntry"></param>
참조 5개
public void IsPlaying(TrackEntry trackEntry)...
참조 1개
IEnumerator NameAnimation()...
/// <summary>
/// 스파인 애니메이션 시작
/// </summary>
/// <returns></returns>
참조 1개
IEnumerator SpineStarAni()...
/// <summary>
/// 스파인 능력치패널 보여주기
/// </summary>
/// <returns></returns>
참조 1개
IEnumerator AbiltyPanelShow()...
```

QR코드 스캔


```
if (CanQrScan)
{
    if (Input.inputString.Length > 0)
    {
        //뛰어쓰기가 있을경우
        if (Input.inputString.IndexOf(wordEnd) >= 0 || Input.inputString.Contains("₩n") || Input.inputString.IndexOf('₩r') >= 0)
        {
            code += Input.inputString;
            code = code.Replace("₩n", "");
            code = code.Replace("₩r", "");
            code_last = code;
            code = "";
        }
        //뛰어쓰기가아닐경우
        else
        {
            code += Input.inputString;
            startTime = Time.time;
        }
        if (!string.IsNullOrEmpty(code_last))
        {
            QRScan(code_last);
            code_last = "";
            code = "";
            startTime = 0;
        }
    }
    else
    {
        if (startTime > 0.1f)
        {
            if (Time.time > startTime + 0.1f)
            {
                code = "";
                startTime = 0;
            }
        }
    }
}
```

랭킹창 제작


```
/// <summary>
/// 랭킹 로드하기
/// </summary>
참조 1개
private void LoadRankingData()...
/// <summary>
/// 어느 랭킹타입을 보여줄지 세팅
/// </summary>
/// <param name="_type"></param>
참조 1개
public void SetRanking(ranktype _type = ranktype.none)...
/// <summary>
/// 현재 점수로 랭킹업데이트
/// </summary>
/// <param name="playerName"></param>
/// <param name="playerScore"></param>
참조 2개
public void UpdatePlayerRanking(string playerName, int playerScore)...
/// <summary>
/// 랭킹이없을때 기본랭킹 제작
/// </summary>
/// <param name="folderPath"></param>
참조 1개
private void CreateDefaultRankingData(string folderPath)...
/// <summary>
/// 랭킹창에 어떻게 표시할지 세팅
/// </summary>
/// <param name="rankingData"></param>
참조 3개
private void ApplyRankingData(RankingData rankingData)...
/// <summary>
/// 랭킹업데이트 json에 저장
/// </summary>
/// <param name="rankingData"></param>
/// <param name="playerName"></param>
/// <param name="playerScore"></param>
참조 1개
private void UpdateRankingData(RankingData rankingData, string playerName, int playerScore)...
```

시트에서 데이터 가져오기&스크립트 생성

stat_def_range	원거리 방어력	Ranged defense
stat_atk_speed	공격 속도	Attack speed

번역 | StageDataTable | MonsterDataTable | SkillCardDataTable | SkillCardWeightDataTable | ResultDataTable | CommonDataTable | ProductionDataTable | SoundDataTable | QRTable


```
public class CommonDataTable
{
    public float attack_damage_minimum;
    public float attack_damage_maximum;
    public float battle_time;
    public float battle_deathblow_cell;
    public float battle_combo_time;
    public float battle_combo1;
    public float battle_combo2;
    public float battle_combo3;
    public float battle_combo4;
    public float first_strike_time;
    public float battle_manual_time;
    public float battle_manual_meleehit;
    public float battle_manual_rangehit;
    public float battle_deathblow_time;
    public float battle_deathblow_atk;
    public float battle_finalattack_time;
    public float battle_manual_finalhit;
    public float battle_manual_defensehit;
    public float manual_time_delay;
    public float battle_deathblow_height;
    public float first_strike_enemy;
    public float first_strike_player;
    public float battle_groggy_turn;
    public float stage_difficulty_value;
    public float defense_skill_max;
    public float game_update_season;
}
```

```
참조 2개
private static void Import()...

[MenuItem("Sheet/CreateCSV")]
참조 0개
public static void CreateCSV()...

참조 2개
public static void InitSercive()...


[MenuItem("Sheet/TableSheets %g")]
참조 0개
static void Open()...



Unity 메시지 | 참조 0개
public void OnGUI()...

/// <summary>
/// 10nWz6UvLxdZ0HVgxWKqCxAlGaPJipFLC-32rP-MSSBg
/// </summary>

참조 1개
public void InputWWW()...

참조 1개
public void FindTable()...
참조 1개
public void CreateButton()...

참조 1개
public void AllCreateScripts()...

참조 1개
public void CreateScript(int i)...
```

버튼 입력

```
public abstract class KeyInput : MonoBehaviour, KeyInterFace
{
    public delegate void InputQ();
    public delegate void InputW();
    public delegate void InputE();
    public delegate void InputLeft();
    public delegate void InputDown();
    public delegate void InputRight();
    public InputQ inputQ;
    public InputW inputW;
    public InputE inputE;
    public InputLeft inputLeft;
    public InputDown inputDown;
    public InputRight inputRight;
    참조 13개
    public abstract void InputLeftQ();
    참조 13개
    public abstract void InputLeftW();
    참조 13개
    public abstract void InputLeftE();
    참조 9개
    public abstract void InputRightQ();
    참조 9개
    public abstract void InputRightW();
    참조 9개
    public abstract void InputRightE();
    Unity 메시지 | 참조 0개
    protected void Start()...
    Unity 메시지 | 참조 0개
    protected void Update()...
    /// <summary>
    /// 해당 콜백에 넣을 함수지정
    /// </summary>
    /// <param name="q"></param>
    /// <param name="w"></param>
    /// <param name="e"></param>
    /// <param name="left"></param>
    /// <param name="down"></param>
    /// <param name="right"></param>
    참조 37개
    public void Setting(InputQ q = null, InputW w = null, InputE e = null, InputLeft left = null, InputDown down = null, InputRight right = null)...
```

```
public bool GetButtonDown(SerialButton btn)
{
    switch (btn)
    {
        case SerialButton.Player1Note1: return _ButtonBuffer._Player1Buttons[0] && !_ButtonBuffer._PrevPlayer1Buttons[0];
        case SerialButton.Player1Note2: return _ButtonBuffer._Player1Buttons[1] && !_ButtonBuffer._PrevPlayer1Buttons[1];
        case SerialButton.Player1Note3: return _ButtonBuffer._Player1Buttons[2] && !_ButtonBuffer._PrevPlayer1Buttons[2];
        case SerialButton.Player1Skill: return _ButtonBuffer._Player1Buttons[3];

        case SerialButton.Player2Note1: return _ButtonBuffer._Player2Buttons[0] && !_ButtonBuffer._PrevPlayer2Buttons[0];
        case SerialButton.Player2Note2: return _ButtonBuffer._Player2Buttons[1] && !_ButtonBuffer._PrevPlayer2Buttons[1];
        case SerialButton.Player2Note3: return _ButtonBuffer._Player2Buttons[2] && !_ButtonBuffer._PrevPlayer2Buttons[2];
        case SerialButton.Player2Skill: return _ButtonBuffer._Player2Buttons[3];

        case SerialButton.ManagerMode: return _ButtonBuffer._ManagerButton;
        case SerialButton.ServiceCredit: return _ButtonBuffer._CreditButton;
    }

    return false;
}
참조 6개
public bool GetButtonUp(SerialButton btn)
{
    switch (btn)
    {
        case SerialButton.Player1Note1: return !_ButtonBuffer._Player1Buttons[0] && _ButtonBuffer._PrevPlayer1Buttons[0];
        case SerialButton.Player1Note2: return !_ButtonBuffer._Player1Buttons[1] && _ButtonBuffer._PrevPlayer1Buttons[1];
        case SerialButton.Player1Note3: return !_ButtonBuffer._Player1Buttons[2] && _ButtonBuffer._PrevPlayer1Buttons[2];
        case SerialButton.Player1Skill: return _ButtonBuffer._Player1Buttons[3];

        case SerialButton.Player2Note1: return !_ButtonBuffer._Player2Buttons[0] && _ButtonBuffer._PrevPlayer2Buttons[0];
        case SerialButton.Player2Note2: return !_ButtonBuffer._Player2Buttons[1] && _ButtonBuffer._PrevPlayer2Buttons[1];
        case SerialButton.Player2Note3: return !_ButtonBuffer._Player2Buttons[2] && _ButtonBuffer._PrevPlayer2Buttons[2];
        case SerialButton.Player2Skill: return _ButtonBuffer._Player2Buttons[3];

        case SerialButton.ManagerMode: return _ButtonBuffer._ManagerButton;
        case SerialButton.ServiceCredit: return _ButtonBuffer._CreditButton;
    }

    return false;
}
```

RF카드(데이터저장)


```
void DeviceInventoryCheck()
{
    int iret = 0;
    UIntPtr dnInvenParamList = RFIDLIB.rfidlib_reader.RDR_CreateInvenParamSpecList();
    RFIDLIB.rfidlib_aip_iso15693.ISO15693_CreateInvenParam(dnInvenParamList, (byte)0, (byte)0, (byte)0, (byte)0);
    iret = RFIDLIB.rfidlib_reader.RDR_TagInventory(hreader, RFIDLIB.rfidlib_def.AI_TYPE_NEW, 0, null, dnInvenParamList);
    if (iret != 0)
    {
        return;
    }

    UIntPtr TagDataReport = UIntPtr.Zero;
    TagDataReport = RFIDLIB.rfidlib_reader.RDR_GetTagDataReport(hreader, RFIDLIB.rfidlib_def.RFID_SEEK_FIRST); //first
    while (TagDataReport != UIntPtr.Zero)
    {
        TagInfo tag = new TagInfo();
        UInt32 aip_id = 0;
        UInt32 tag_id = 0;
        UInt32 ant_id = 0;
        Byte dsfid = 0;
        Byte[] uid = new Byte[16];
        string strUid = "";

        iret = RFIDLIB.rfidlib_aip_iso15693.ISO15693_ParseTagDataReport(TagDataReport, ref aip_id, ref tag_id, ref ant_id, ref dsfid, uid);
        if (iret == 0) //ISO15693
        {
            strUid = BitConverter.ToString(uid, 0, 8).Replace("-", string.Empty);
            tag.uid = strUid;
            tag.aip_id = aip_id;
            tag.tag_id = tag_id;
            MyData = tag;
            if (cor != null)
            {
                StopCoroutine(cor);
                cor = null;
            }
            Debug.Log($"{strUid} 카드 인식");
            ConnectCard(strUid);
        }

        TagDataReport = RFIDLIB.rfidlib_reader.RDR_GetTagDataReport(hreader, RFIDLIB.rfidlib_def.RFID_SEEK_NEXT); //next
    }
}
```

```
StringBuilder ascii = new StringBuilder(hex.Length / 2);

for (int i = 0; i < hex.Length; i += 2)
{
    string hexByte = hex.Substring(i, 2);
    byte asciiByte = byte.Parse(hexByte, System.Globalization.NumberStyles.HexNumber);
    ascii.Append((char)asciiByte);
}

return ascii.ToString();

3개
lic static string AsciiToHex(string ascii)

if (string.IsNullOrEmpty(ascii))
    return "00000000";

StringBuilder hex = new StringBuilder(ascii.Length * 2);

foreach (char c in ascii)
{
    string hexValue = ((int)c).ToString("X2"); // 문자를 16진수로 변환 (두 자리)
    hex.Append(hexValue);
}
while (hex.Length < 8)
{
    hex.Append("00");
}

return hex.ToString();

2개
lic static byte[] StringToByteArrayFastest(string hex)

if (hex.Length % 2 == 1)
    throw new Exception("The binary key cannot have an odd number of digits");

int len = hex.Length >> 1;
byte[] arr = new byte[len];

for (int i = 0; i < len; ++i)
{
    arr[i] = (byte)((GetHexVal(hex[i << 1]) << 4) + (GetHexVal(hex[(i << 1) + 1])));
}

return arr;
```

신용카드결제

```
second *= 10;
int count = second / 250;
isCardTrading = true;
while (count >= 0 && !success)
{
    ushort senderIndex = 0x0E01;
    ushort receiverIndex = 0x0A01;
    byte commandCode = 0xCA;
    ushort dataLength = 70; // 요청 데이터 길이
    byte[] requestData = new byte[dataLength];
    byte byteTime = (byte)Mathf.Clamp(second, 0, 250); // int time을 byte로 변환
    requestData[0] = byteTime; // 타임 아웃

    //시간
    string hexTimeString = "0x" + byteTime.ToString("X2");
    requestData[0] = Convert.ToByte(hexTimeString, 16);

    //돈
    byte[] amountBytes = BitConverter.GetBytes(purchaseAmount);
    Array.Reverse(amountBytes);
    Array.Copy(amountBytes, 0, requestData, 1, 4); // 구매 금액
    requestData[5] = column; // 컬럼

    Array.Copy(System.Text.Encoding.ASCII.GetBytes(FillLine(line1)), 0, requestData, 6, 16);
    Array.Copy(System.Text.Encoding.ASCII.GetBytes(FillLine(line2)), 0, requestData, 22, 16);
    Array.Copy(System.Text.Encoding.ASCII.GetBytes(FillLine(line3)), 0, requestData, 38, 16);
    Array.Copy(System.Text.Encoding.ASCII.GetBytes(FillLine(line4)), 0, requestData, 54, 16);

    // CRC 계산
    ushort crcValue = CRC_Update(new byte[] {
    seqNo,
    (byte)(senderIndex >> 8),
    (byte)senderIndex,
    (byte)(receiverIndex >> 8),
    (byte)receiverIndex,
    commandCode,
    (byte)(dataLength >> 8),
    (byte)dataLength
}, requestData);

    // 메시지 생성
    byte[] message = new byte[dataLength + 12]; // STX + ETX
    message[0] = 0x02; // STX
    message[1] = seqNo; // Seq. No
    message[2] = (byte)(senderIndex >> 8);
    message[3] = (byte)senderIndex;
```

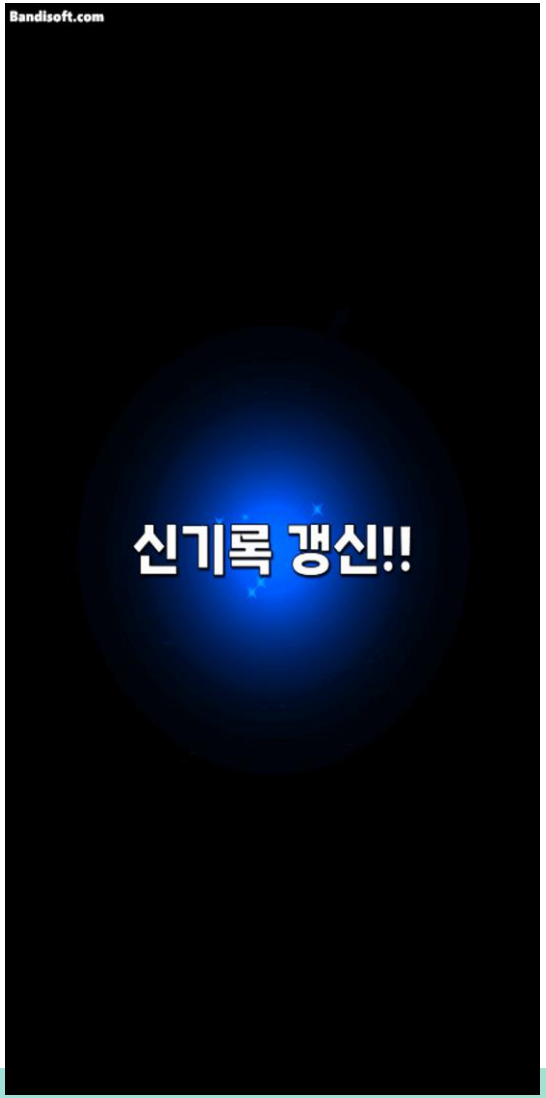
```
int actualDataLength = serialPort.BytesToRead;
byte[] responseData = new byte[actualDataLength];
serialPort.Read(responseData, 0, actualDataLength);

string logMessage = "STX: " + responseData[0].ToString("X2") +
    " Seq. No: " + responseData[1].ToString("X2") +
    " Sender Index: " + BitConverter.ToUInt16(responseData, 2).ToString("X4") +
    " Receiver Index: " + BitConverter.ToUInt16(responseData, 4).ToString("X4") +
    " Command Code: " + responseData[6].ToString("X2") +
    " Data Length: " + BitConverter.ToUInt16(responseData, 7).ToString("X4") +
    " Response Code: " + responseData[9].ToString("X2") +
    " Reply Data: " + BitConverter.ToString(responseData, 10, responseData.Length - 13).Replace("-", " ") +
    " CRC: " + BitConverter.ToUInt16(responseData, responseData.Length - 3).ToString("X4") +
    " ETX: " + responseData[responseData.Length - 1].ToString("X2");
ProcessResponseData(responseData[9].ToString("X2"));
Debug.Log(logMessage);
switch (_code)
{
    case CommandCode.StopTrading:
    case CommandCode.TradingCancel:
        isCardTrading = false;
        break;
    case CommandCode.TimeReading:
        {
            if (_code == CommandCode.TimeReading)
            {
                ProcessResponse(responseData);
            }
        }
        break;
    case CommandCode.bufferReset:

        string resetMessage = ConvertBytesToHex(responseData);
        Debug.Log("응답 메시지 : " + ConvertHexToAsciiReset(resetMessage));
        break;
    case CommandCode.Trading:
        string replyMessage = ConvertHexToAsciiTrading(ConvertBytesToHex(responseData));
        Debug.Log("응답 메시지 : " + replyMessage);
        if (string.IsNullOrEmpty(replyMessage))
        {
            Debug.Log("결제 안됨");
```

필요 이펙트 제작&수정


Bandisoft.com
체력 : 11,300
2790
1650
1100
약점 체크 중....
11
카드가 없다면
버튼을 눌러주세요!


Bandisoft.com
신기록 갱신!!


Bandisoft.com
혼자놀기 랭킹
1
ChoiWooseok
1,000,000
2
ParkSanghee
900,000
3
LeeMinwoo
800,000
30
랭킹을 확인했으면
버튼을 눌러주세요.


Bandisoft.com
혼자놀기 랭킹
1
ChoiWooseok
1,000,000
2
ParkSanghee
900,000
3
LeeMinwoo
800,000
30
랭킹을 확인했으면
버튼을 눌러주세요.

- 게임영상과 사용하지않는 신용카드결제,RF카드기능의 코드가 들어있는 드라이브링크입니다.

https://drive.google.com/drive/folders/1TLnTK4ulL6UceQMtY69cZcjtKPVeOF7L

더 자세한 코드는 회사자산이라 보여드릴수 없는점 양해부탁드립니다.

봐주셔서 감사합니다.