


포트폴리오 및 경력기술서 진행 프로젝트 이력

윤준혁 Yun Joonhyuk

 010-9258-46312

 alex6464@naver.com

주요 경험 Experiences

2022

넵툰 근무 (2020.4 – 현재)

Line Puzzle TanTan, 프렌즈 사천성 등의 Unity 2D 캐주얼 퍼즐 게임 개발 및 서비스 중

2021

2020

인하대학교 게임개발 교육과정 수료 (2019.7 – 2020.4)

Unreal Engine 4, MFC 등을 이용한 게임개발 교육 수료

2019

2018

졸업프로젝트 <Cube Animals> (2018.3 – 2018.11)

Unity3D를 이용한 캐주얼 게임 제작

2017

TRIPEAKS 인턴 (2017.12 – 2018.2)

대학연계 단기 인턴활동으로 WIDE CROSS라는 Unity 3D MOBA게임 QA, 개발보조로 참여

2013

가천대학교 컴퓨터 공학과 입학

취업 전 제작 경험

가천대학교 컴퓨터공학과, 인하대학교 인재개발교육원 (2013. 3 – 2020. 4)

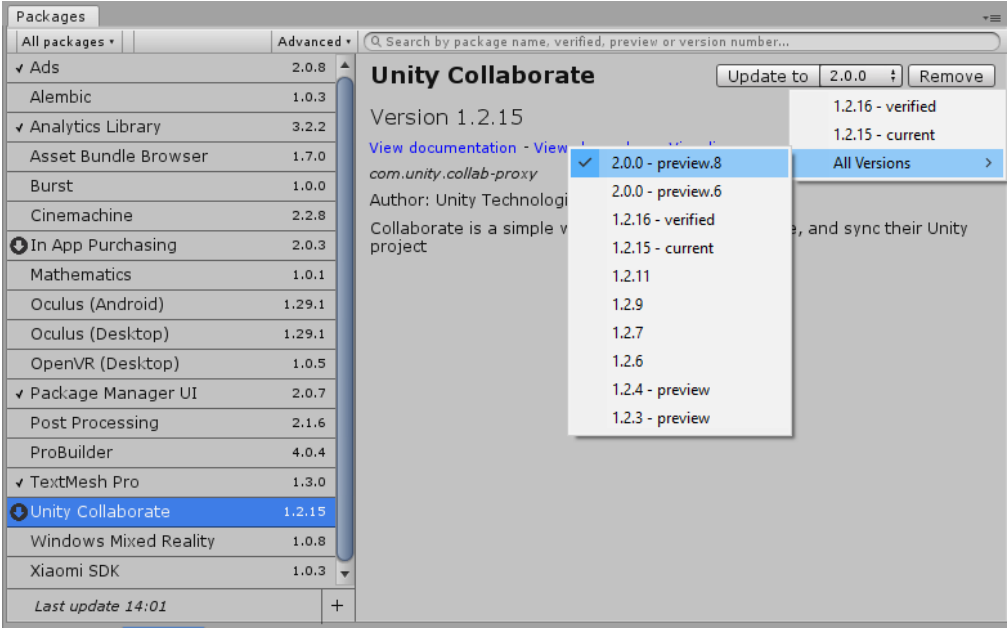
Cube Animals (2018)



게임명	Cube Animals
게임 개요	시작 지점에서 골인 지점까지 타일을 잇는 퍼즐 게임
장르	캐주얼 퍼즐
플랫폼	Android
엔진	Unity 2017
완성도	구글 플레이스토어 출시
개발 인원	프로그래머 3명
개발 기간	2018. 3 - 2018. 11 (약 9개월)
담당 파트	특수 타일 기능 개발, CSV 매니저 구현, 로딩 씬 구현, UI UX 개발, 도전과제, 튜토리얼 등 콘텐츠 개발

- UGUI를 이용한 3D 캐주얼 퍼즐 게임.
- 대학교 졸업 프로젝트로 해본 첫 장기 프로젝트.
- 기간 안에 목표했던 플레이스토어 출시까지 완료. 우수 프로젝트로 선정되어 교내 프로젝트 전시.

Cube Animals (2018)



no	mission2	mission3		
0	100	일자 다리만 사용해서 클리어		
1	200	과일을 획득하고 클리어		
2	200	회전을 10회 하고 클리어		
3	200	일자 다리를 사용하지 않고 클리어		
4	200	과일을 획득하고 클리어		
5	200	과일을 획득하지 않고 클리어		
6	300	과일을 획득하고 클리어		
7	600	회전을 9번 이상 하고 클리어		

```
List<Dictionary<string, object>> data = CSVReader.Read("mission");
selectedStagenum.text = (chapternum + 1) + " - " + (stagenum + 1);
mission2.text = "미션 2 : 과일을 " + data[realstagenum]["mission2"] + "개 이상 남기고 클리어";
mission3.text = "미션 3 : " + data[realstagenum]["mission3"];
```

구현 내용 소개

형상관리 툴

- 당시엔 git에 대한 지식이 얇았던 상태
- 유니티에서 제공하는 Collaborate 기능을 이용해 팀원끼리 프로젝트를 동기화 및 버전을 관리

데이터 관리

- CSV를 이용해 테이블 데이터를 읽어와 UI에 표시
- 데이터 저장은 PlayerPrefs을 이용해 대부분 처리

Cube Animals (2018)

```
0 references
void Start()
{
    StartCoroutine(StartLoad(loading_next_scenename));
}

0 references
void Update()
{
    fTime += Time.deltaTime;
    slider.value = fTime;

    if (fTime >= 1)
    {
        if(async_operation == null)
        {
            SceneManager.LoadScene("Lobby");
        }
        else
        {
            async_operation.allowSceneActivation = true;
        }
    }
}
```

```
public IEnumerator StartLoad(string strSceneName)
{
    async_operation = SceneManager.LoadSceneAsync(strSceneName);
    async_operation.allowSceneActivation = false;

    if (IsDone == false)
    {
        IsDone = true;

        while (async_operation.progress < 0.9f)
        {
            slider.value = async_operation.progress;

            yield return true;
        }
    }
}
```

구현 내용 소개

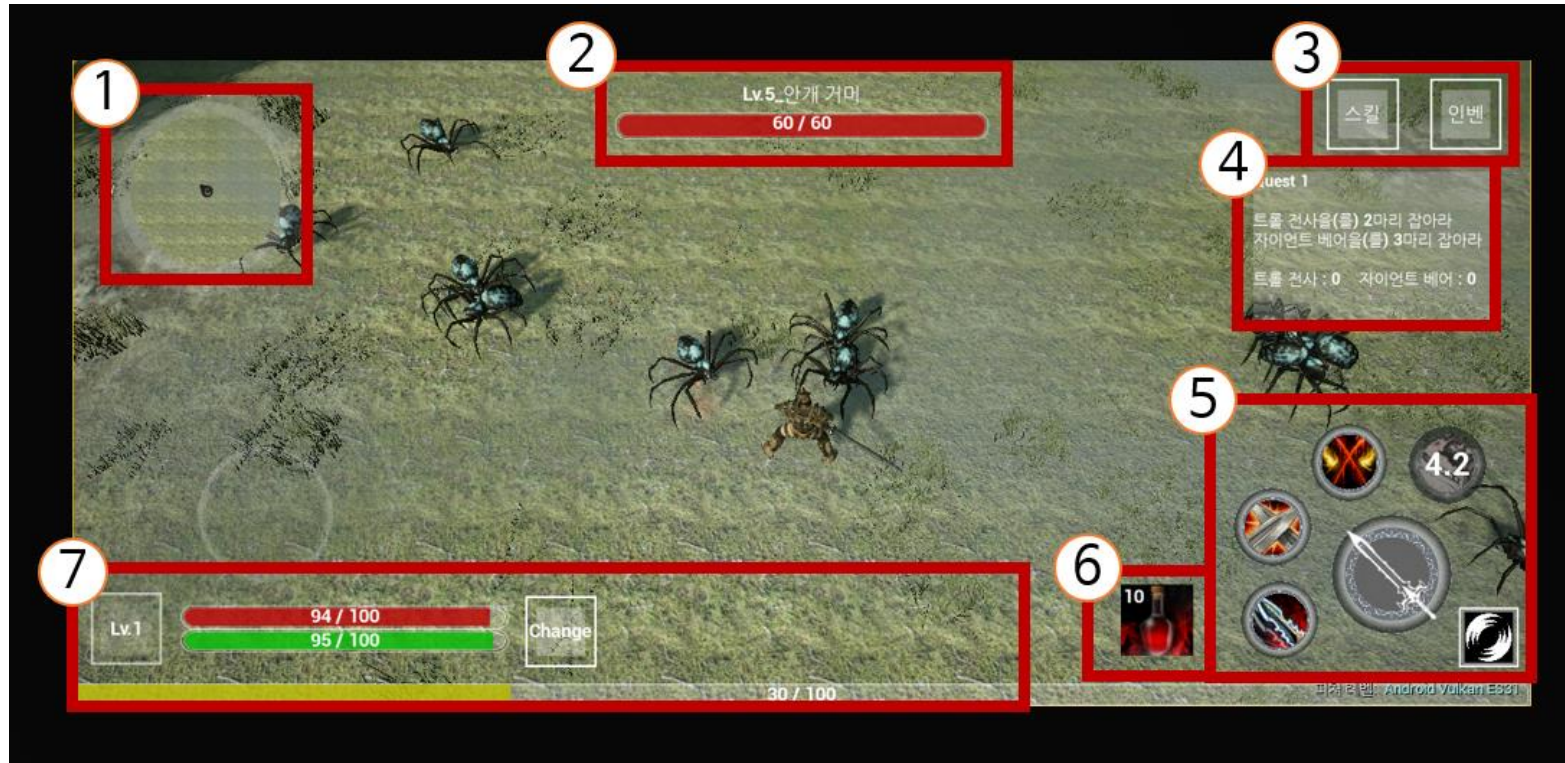
Loading Scene

- Scene을 이동하는 과정에서 로딩 즉시 이동할 수 있도록 비동기적 연산 코루틴 작업을 처리



게임명	모글레이 전기
게임 개요	모바일로 가볍게 즐길 수 있는 모바일 선형 RPG 게임
장르	RPG
플랫폼	Android
엔진	Unreal Engine 4.22
완성도	안드로이드 빌드
개발 인원	기획 1명, 프로그래머 4명
개발 기간	2019. 10 - 2019. 11 (약 2개월)
담당 파트	스킬 UI 및 로직, 애니메이션, 아이템 인벤토리 UI 및 로직

- UE4를 이용한 3D 싱글 RPG게임
- 인하대학교 게임교육기관에서 가장 마지막으로 개발한 팀 프로젝트.
- 목표했던 모든 기능이 구현된 건 아니지만, 필수로 들어가야 되는 기능은 전부 구현 완료.



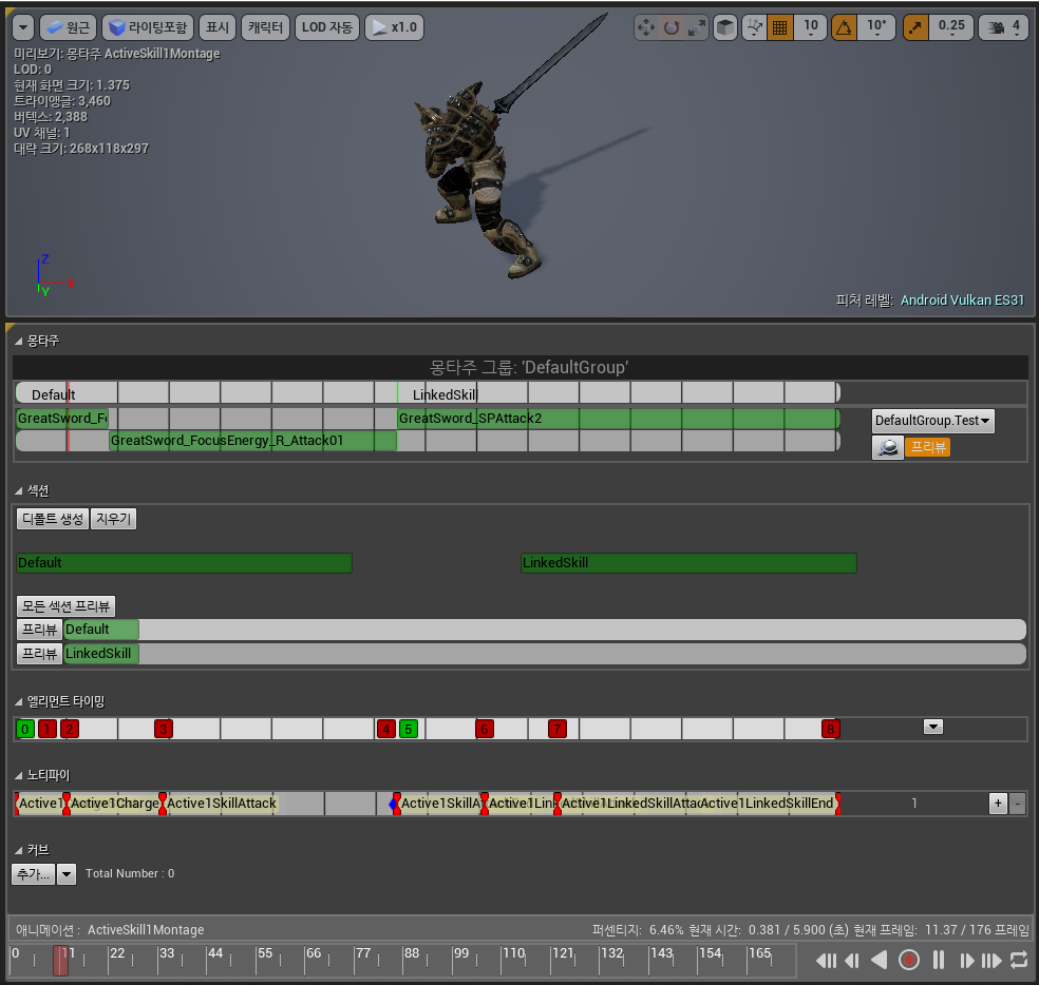
구현 내용 소개

형상관리 툴

- Git으로 관리하려 하였으나 프로젝트 용량이 너무 커 업로드가 제대로 되지 않는 문제가 있었음
- 대신 svn을 사용해서 버전관리

HUD

- 3번 스킬, 인벤토리와 5번 스킬 UI 개발 담당



구현 내용 소개

스킬 모션 몽타주

- 한 몽타주를 두개의 섹션으로 분리해 각 섹션을 한 개의 스킬에 할당해 사용.
- 총 8개의 스킬을 제작
- 각 섹션 공격 타이밍에 AnimNotify를 적용.
- Anim_BP에서 각 AnimNotify에 실행할 스킬 어택 로직을 제작해 사용

스킬 버튼 로직

- 일반 스킬, 연계스킬 상태에 따라 표시할 이미지 변경
- 스킬 종료 후 SkillTable에 할당된 쿨타임 만큼 사용 불가





구현 내용 소개

스킬 UI

- 패시브, 액티브 스킬의 레벨을 올리는 UI 구현

인벤토리 UI

- 물약과 아이템 위젯, 클래스 구현
- 인벤토리 매니저에 저장되는 순서대로 아이템 위젯 생성 후 표기하는 기능 제작
- 아이템 등급과 장착중인 아이템에 추가적인 UI를 표시하는 기능 제작

장비 장착 UI

- 현재 장착중인 장비와 선택한 장비 데이터를 표시하는 기능 제작



경력 사항

넵툰 클라이언트 프로그래머 (2020. 4 - 현재)



재직 기간 | 2020. 4. 27 – 현재

주요 업무 | <Line TanTan>, <프렌즈 사천성> 등 라이브 게임 콘텐츠 개발 및 유지보수
<Onet Adventure> 등 신작 프로젝트 개발 참여

주요 경험 | Git, svn을 이용한 버전관리,
Kakao, Line, google 등에서 제공하는 sdk 적용 및 업그레이드,
UI/UX 개선 작업,
신규 콘텐츠 개발,
인게임 로직 리팩토링 등.



Line Puzzle TanTan, 탄탄 for Kakao



게임명 | Line Puzzle TanTan

장르 | 캐주얼 퍼즐

플랫폼 | Android, IOS

엔진 | Unity3D

완성도 | 라이브 서비스 중

담당 파트 | UI UX개발, 신규 콘텐츠 개발, 인게임 로직 개선

- Unity3D를 이용한 2D 캐주얼 퍼즐게임.
- NGUI, Spine 등을 이용하여 UI를 구현.
- 빌드 스크립트를 이용해 Android, IOS 앱 빌드와 애셋 빌드를 젠킨스와 빌드머신을 통해 빌드.
- Line과 Kakao를 통해 퍼블리싱 하여 라이브 서비스 중인 게임.

Line Puzzle TanTan, 탄탄 for Kakao

```
bannerTable.GetComponent<UIWrapContentExtend>().IndexMinusAction = new System.Action(() => BannerIndexCalc(false));  
bannerTable.GetComponent<UIWrapContentExtend>().IndexPlusAction = new System.Action(() => BannerIndexCalc(true));  
  
fullBannerTable.GetComponent<UIWrapContentExtend>().IndexMinusAction = new System.Action(() => FullBannerIndexCalc(false));  
fullBannerTable.GetComponent<UIWrapContentExtend>().IndexPlusAction = new System.Action(() => FullBannerIndexCalc(true));
```



```
if (distance < -extents)  
{  
    Vector3 pos = t.localPosition;  
  
    while (distance < -extents)  
    {  
        pos.x += ext2;  
        distance = pos.x - center.x;  
    }  
    int realIndex = Mathf.RoundToInt(pos.x / itemSize);  
    if (IndexPlusAction != null) IndexPlusAction();  
  
    // NGUI modified by neptune  
    if (ignoreUpdateItem != true && (minIndex == maxIndex  
    // if (minIndex == maxIndex || (minIndex <= realIndex  
    {  
        t.localPosition = pos;  
        UpdateItem(t, i);  
    }  
    else allWithinRange = false;  
}  
else if (distance > extents)  
{  
    Vector3 pos = t.localPosition;  
    // pos.x -= ext2;  
    // distance = pos.x - center.x;  
  
    while (distance > extents)  
    {  
        pos.x -= ext2;  
        distance = pos.x - center.x;  
    }  
    int realIndex = Mathf.RoundToInt(pos.x / itemSize);  
    if (IndexMinusAction != null) IndexMinusAction();  
  
    // NGUI modified by neptune  
    if (ignoreUpdateItem != true && (minIndex == maxIndex  
    // if (minIndex == maxIndex || (minIndex <= realIndex  
    {  
        t.localPosition = pos;  
        UpdateItem(t, i);  
    }  
    else allWithinRange = false;  
}  
else if (mFirstTime) UpdateItem(t, i);
```

구현 내용 소개

이벤트 배너

- 각각 분리되어 있던 풀 배너와 일반 배너를 합친 통합 배너를 새로 제작
- 기존엔 하나씩 노출되던 배너들을 하나로 모아 좌우 슬라이드 하여 표시하도록 구현
- NGUI의 UIWrapContent 클래스를 상속받아 WrapContent 메소드를 오버라이딩하여 다음 배너로 이동되는 시점을 알아오도록 구현
- 해당 기능을 이용해 배너 좌 우로 이동시 하단 점을 키고 끌 수 있도록 사용

Line Puzzle TanTan, 탄탄 for Kakao

구현 내용 소개

```
2 references
IEnumerator StartBannerAutoScroll()
{
    while (true)
    {
        yield return new WaitForSeconds(5f);

        //배너를 누르고 있지 않은 상태라면 다음 배너로 자동으로 이동
        if (!isPressBanner)
        {
            SpringPanel.Begin(bannerList, new Vector3(GetNearestspxPivot(bannerList.transform.localPosition.x, spxDefault, dwidth),
                bannerList.transform.localPosition.y), 10.0f);
        }
    }
}

6 references
float GetNearestspxPivot(float positionX, float spxDefault, float width)
{
    float defaultX = spxDefault;

    int nextBannerNumber = (int)((positionX - defaultX) / width);
    return (nextBannerNumber - 1) * width + defaultX;
}
```

이벤트 배너

- 일정 시간이 지나면 자동으로 하나씩 넘어갈 수 있도록 SpringPanel을 작동하도록 구현
- 무한 스크롤 뷰를 적용해 좌 우로 계속 넘겨도 반복해서 배너가 나올 수 있도록 구현

Line Puzzle TanTan, 탄탄 for Kakao

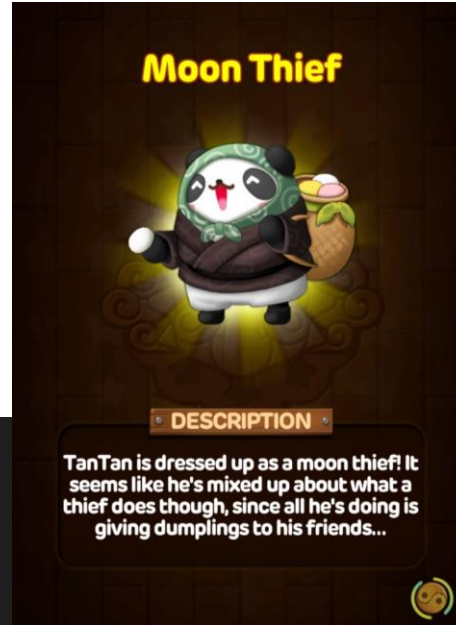
```
void Start()
{
    transform.Find("Collabo")?.gameObject.SetActive(false);

    if (!ShowPetCostumeInformation())
    {
        //설정된 펫 코스튬이 아무것도 없다면 기존방식 진행
        ShowTip();
        waitUntilTime += Time.time + 1f;
    }
}
```

```
bool SetInformation()
{
    List<string> costumes = SharedData.instance.loadingResource.costumeIds;
    List<int> pets = SharedData.instance.loadingResource.petIds;

    int petNum = -1;
    string costumeId = "";

    if ((costumes == null || costumes.Count == 0) && (pets == null || pets.Count == 0))
    {
        //로딩 씬에 보여줄 펫과 코스튬이 없음
        return false;
    }
    else if (costumes == null || costumes.Count == 0)
    {
        petNum = pets[UnityEngine.Random.Range(0, pets.Count)];
        if (PetInspector.instance.IsPetExistInMeta(petNum))
        {
            if (SetPetInformation(petNum) == false)
                return false;
        }
        else
            return false;
    }
    else if (pets == null || pets.Count == 0)
    {
        costumeId = costumes[UnityEngine.Random.Range(0, costumes.Count)];
        if (CostumeInspector.instance.IsCostumeExistInMeta(costumeId) && CostumeInspector.instance.IsExistLoadingCostumePrefab(costumeId))
            SetCostumeInformation(costumeId);
        else
            return false;
    }
}
```



구현 내용 소개

로딩 씬 펫, 코스튬 노출 기능

- 운영 툴에서 설정한 펫과 코스튬을 로딩 씬에서 노출시켜 현재 가차에서 나오는 픽업 아이템을 홍보하는 기능을 제작
- 서버에서 받아온 데이터로 노출할 펫과 코스튬 리스트를 확인 후, 어떤 데이터를 노출시킬지, 해당 데이터가 있는지 확인하는 작업을 거침
- 만약 데이터가 없다면 기존에 나오던 일반 팁 중 한가지를 랜덤하게 표시하도록 구현

Line Puzzle TanTan, 탄탄 for Kakao

Google Mobile Ads Unity Plugin v7.1.0 Latest




Plugin:

- Added AppStateEventNotifier as a better option to OnApplicationPause for app open ads.

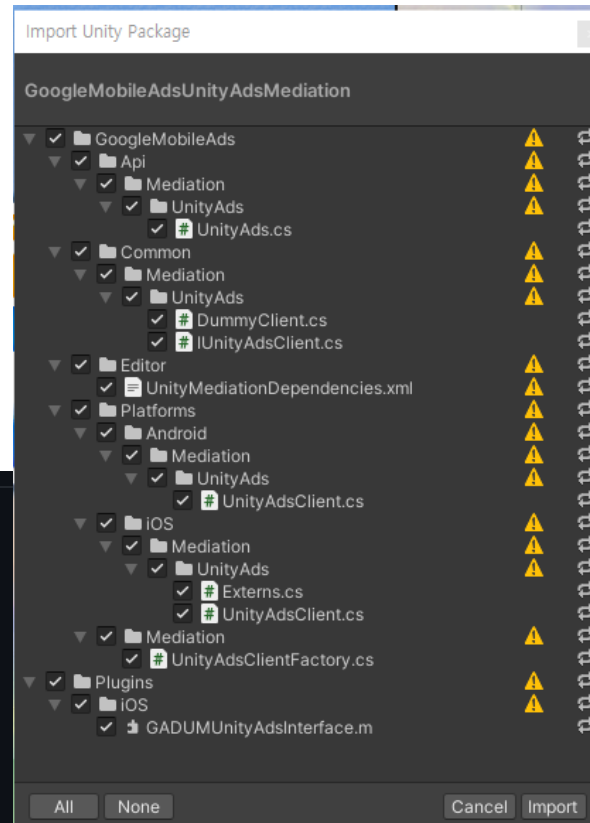
Built and tested with:

- Google Mobile Ads Android SDK 21.0.0.
- Google Mobile Ads iOS SDK 9.0.0
- External Dependency Manager for Unity 1.2.171.

▼ Assets 3

 GoogleMobileAds-v7.1.0.unitypackage	2.06 MB	16 Jun 2022
 Source code (zip)		16 Jun 2022
 Source code (tar.gz)		16 Jun 2022

 13  4  2  5  1 19 people reacted



구현 내용 소개

구글 애드몹 업데이트

- 구글 애드몹 플러그인과 미디어이션 어댑터 버전 업그레이드 작업

Line Puzzle TanTan, 탄탄 for Kakao

Mobile Notifications

Version 2.0.0 - February 08, 2022

Name

com.unity.mobile.notifications

Links

[View documentation](#)
[View changelog](#)
[View licenses](#)

Author

Unity Technologies

Registry

Unity

Published Date

February 08, 2022

Mobile Notifications package adds support for scheduling local repeatable or one-time notifications on iOS and Android.

Requires iOS 10 and Android 4.4 or above.

Dependencies

Android JNI 1.0.0 (enabled ✓)

Samples

Notification Samples 3.61 MB

Import into Project

```
#if UNITY_IPHONE
    var notification = new iOSNotification();
    notification.Identifier = pushId.ToString();
    notification.Title = ScriptLocalization.Get("Text.LocalPushMessageTitle");
    notification.Body = pushInfo.body;
    notification.Trigger = new iOSNotificationTimeIntervalTrigger()
    {
        TimeInterval = (pushInfo.sendAt - DateTime.Now),
    };

    iOSNotificationCenter.ScheduleNotification(notification);
#elif UNITY_ANDROID
    if (Application.platform == RuntimePlatform.Android)
    {
        var notification = new AndroidNotification();
        notification.Title = ScriptLocalization.Get("Text.LocalPushMessageTitle");
        notification.Text = pushInfo.body;
        notification.FireTime = pushInfo.sendAt;
        AndroidNotificationCenter.SendNotificationWithExplicitID(notification, ChannelId, pushId);
    }
#endif
```

구현 내용 소개

로컬 푸시 구현

- Mobile Notification 패키지를 이용해 앱 내 로컬푸시 기능 구현

Line Puzzle TanTan, 탄탄 for Kakao

AppendSpiritWindow

스파인 데이터와 이미지가 들어있는 폴더를 지정해주세요.

Spirit ID :

20

9_정령캐릭터_배경

D:/workspace/line_tantan_image/01_업데이트/09_정령캐릭터_배경/06

Select

10_정령육성

Select

정령 스파인 폴더

Open Directory

Clear Path

imageCount : 8

spiritCount : 0

Refresh

MakeImage

MakeSpine

MakePrefab

<작업 상태>

makeImage : False

makeSpine : False

makePrefab : False

```
void CopySpiritImage()
{
    if (string.IsNullOrEmpty(spiritImagePath) || imageCount < 8) return;

    string destFileName = string.Empty;
    string destFilePath = string.Empty;
    string destFolderPath = string.Format("{0}Spirit_{1}", SpiritPrefabImageDir, spiritNumber);
    if (!Directory.Exists(destFolderPath))
        Directory.CreateDirectory(destFolderPath);

    TextureImporter importer = null;

    DirectoryInfo dicInfo = new DirectoryInfo(this.spiritImagePath);
    List<FileInfo> files = new List<FileInfo>(dicInfo.GetFiles());

    foreach (FileInfo info in files)
    {
        if (info.Name.Contains("spirit") == false) continue;

        var names = info.Name.Split('_');
        List<string> imagePathList = new List<string>();

        if (names.Length >= 4)
        {
            //spirit_08_1_4 -> texture_spirit_sp_8_4
            //spirit_08_1_4_g -> texture_spirit_sp_8_4_sh
            destFileName = string.Format("texture_spirit_sp_{0}_{1}{2}", spiritNumber, names[3], names[4]);
            destFilePath = string.Format("{0}/{1}", destFolderPath, destFileName);
            File.Copy(info.FullName, destFilePath, true);

            imagePathList.Add(destFilePath);
        }

        for (int i = 0; i < imagePathList.Count; ++i)
        {
            importer = ChangeFileFormat(imagePathList[i], textureSettingInfo);
            if (importer == null)
            {
                Debug.LogError("fail to find texture: " + imagePathList[i]);
                continue;
            }
            importer.npotScale = TextureImporterNPOTScale.None;
            importer.SaveAndReimport();
        }
    }

    makeImage = true;
    Baram.Logger.LogWarning("makeImage Complete");
    AssetDatabase.Refresh();
}
```

구현 내용 소개

에셋 업데이트 툴 제작

- 자주 추가되는 펫, 코스튬 등의 에셋 데이터 추가를 용이하게 하기 위한 툴 제작.

프렌즈 사천성



게임명 | 프렌즈 사천성

장르 | 캐주얼 퍼즐

플랫폼 | Android, IOS

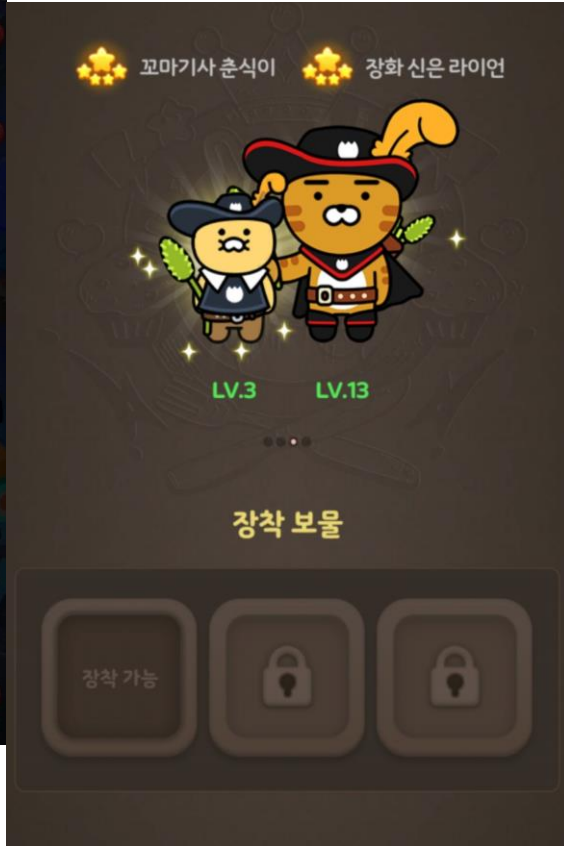
엔진 | Unity3D

완성도 | 라이브 서비스 중

담당 파트 | UI UX개발, 신규 콘텐츠 개발, 인게임 로직 개선

- Unity3D를 이용한 2D 캐주얼 퍼즐게임.
- NGUI, Spine 등을 이용하여 UI를 구현.
- 빌드 스크립트를 이용해 Android, IOS 앱 빌드와 애셋 빌드를 젠킨스와 빌드머신을 통해 빌드.
- Kakao를 통해 퍼블리싱 하여 라이브 서비스 중인 게임.

프렌즈 사천성



구현 내용 소개

춘식이 시스템

- 펫 기능을 하는 춘식이 시스템의 기능 구현
- 인게임, 메인 월드, 로딩 화면, 결과 화면, 랭킹 화면 등 UI에서 Spine 오브젝트 또는 Sprite 이미지가 노출되도록 구현

구현 내용 소개

```
static public void CreateFriendsAndChoonsik(string friendCode, string choonsikCode, GameObject parent, out Friend friend, out Friend choonsik, string mid = null)
{
    friend = Create(friendCode, parent, mid);

    bool createChoonsik = true;
    if (choonsikCode == null || choonsikCode == string.Empty)
        createChoonsik = false;

    if (createChoonsik)
        choonsik = Create(choonsikCode, parent, mid);
    else
        choonsik = null;
}
```

```
public static void SetFriendAndChoonsikLocalposition(ref Friend friend, ref Friend choonsik, float interval)
{
    if (friend == null || choonsik == null) return;

    friend.transform.SetLocalpositionX(friend.transform.localPosition.x + interval);
    choonsik.transform.SetLocalpositionX(choonsik.transform.localPosition.x - interval);
}
```

춘식이 시스템

- Spine 오브젝트가 나와야 되는 경우와 일반 Sprite 이미지가 나와야 하는 경우를 나눠, 같이 나와야 하는 프렌즈 오브젝트와 동시 생성 후 간격을 조정할 수 있도록 구현

```
public class ChoonsikInspector
{
    4 references
    private static ChoonsikInspector _instance;
    2 references
    private bool _initialized;

    147 references
    public static ChoonsikInspector instance
    {
        get
        {
            if (null == _instance)
            {
                _instance = new ChoonsikInspector();
            }
            _instance._init();
            return _instance;
        }
    }

    1 reference
    private void _init()
    {
        if (_initialized)
            return;

        _initialized = true;
    }
}
```

구현 내용 소개

춘식이 시스템

- 매니저 역할을 하는 ChoonsikInspector 클래스를 만들어 싱글톤 패턴을 이용해 하나의 클래스 변수만 사용할 수 있도록 구현

프렌즈 사천성

```
public class IngameSkillController : SingletonGameObject<IngameSkillController>
{
    1 reference
    bool initialized = false;

    4 references
    E_CAS_SkillState CAS_State = E_CAS_SkillState.NON_ACTIVE;
    5 references
    ChoonsikSkillType CAS_Type = ChoonsikSkillType.NONE;

    33 references
    public enum E_CAS_SkillState
    {
        2 references
        NON_ACTIVE, //스킬 사용 불가(사용조건 불충족 또는 액티브 스킬이 없는 춘식)
        3 references
        STAY, //스킬 사용 체크중(15초마다 사용가능여부 판단)
        2 references
        READY, //스킬 사용 준비(버튼 활성화)
        9 references
        ACTION, //스킬 사용중(무한히트 등에 사용)
        14 references
        COMPLETE, //스킬 사용 완료
    }

    1 reference
    public void Initialize()
    {
        initialized = true;
    }
}
```

```
public void SetCAS_State(E_CAS_SkillState state)
{
    Baram.Logger.LogWarning("@===== state set <color=yellow>" + state.ToString() + "</color>");
    CAS_State = state;

    if (state == E_CAS_SkillState.COMPLETE)
    {
        blockManager.RecalculateMatchable();
    }
}

8 references
public E_CAS_SkillState GetCAS_State()
{
    return CAS_State;
}

1 reference
public void SetCAS_Type(ChoonsikSkillType type)
{
    CAS_Type = type;
}

7 references
public ChoonsikSkillType GetCAS_Type()
{
    return CAS_Type;
}
```

구현 내용 소개

춘식이 시스템

- 특수 춘식이가 가지는 액티브 스킬을 제작.
- 인게임 블록 파괴, 일부 방해블록을 일반 블록으로 변경, 아이템 무료로 사용 등의 기능을 제작
- 해당 스킬을 관리할
IngameSkillController 클래스를 싱글톤 패턴으로 생성하여 관리
- 인게임 내 스킬 상태를 스테이트 패턴을 사용하여 스킬 발동 여부 확인에 용이하도록 관리

프렌즈 사천성

```
public void ChoonsikSkillCount(float time)
{
    if (CAS_State != E_CAS_SkillState.STAY)
        return;
    skillTimerCount--;

    if (skillTimerCount <= 0)
    {
        float randomValue = UnityEngine.Random.Range(0f, 100f);
        float skillValue = (float)ChoonsikInspector.instance.GetEquipChoonsikSkillPoint();
        bool skillActivated = randomValue <= skillValue;

        if (skillActivated)
        {
            IngameHeader.instance.SetChoonsikSkillIcon(true);
            SetCAS_State(E_CAS_SkillState.READY);
        }
        else
        {
            skillTimerCount = DAFAULT_CHOONSİK_SKILL_COUNT;
        }
    }
}
```

```
#region CAS(Choonsik Active Skill) area. 춘식 액티브 스킬

1 reference
public void CAS_ChangeQuestion(int count) //y
{
    Baram.Logger.LogWarning("CAS_ChangeQuestion!");
    TransformQuestionBlock(false, count, true);

    IngameSkillController.instance.SetCAS_State(IngameSkillController.E_CAS_SkillState.COMPLETE);
}

1 reference
public void CAS_ChangeBlink(int count) //u
{
    Baram.Logger.LogWarning("CAS_ChangeBlink!");
    TransformBlinkBlock(false, count, true);

    IngameSkillController.instance.SetCAS_State(IngameSkillController.E_CAS_SkillState.COMPLETE);
}

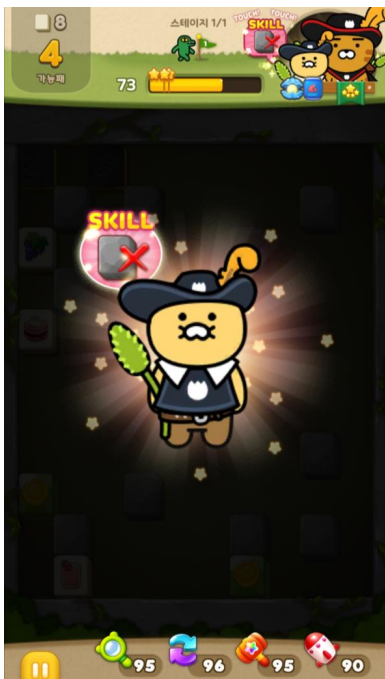
1 reference
public void CAS_DestroyAngelDevil(int count) //i
{
    Baram.Logger.LogWarning("CAS_DestroyAngelDevil!");

    List<IBlock[]> blocks = new List<IBlock[]>();
    List<IBlock> tmp = new List<IBlock>();

    for (int i = 0; i < count; ++i)
    {
        IBlock[] t = null;
        t = GetRandomBlockSet((b) => b is Block_Destiny && !b.GetType().IsSubclassOf(typeof(Block_Destiny)) && !tmp.Contains(b));
        if (t == null || t.Length != 2)
        {
            RecalculateMatchable();
            break;
        }

        if (t != null && t.Length == 2)
        {
            blocks.Add(t);
            tmp.Add(t[0]);
            tmp.Add(t[1]);
        }
    }
    RemoveBlockList(blocks);

    IngameSkillController.instance.SetCAS_State(IngameSkillController.E_CAS_SkillState.COMPLETE);
}
```



구현 내용 소개

춘식이 시스템

- 스킬 컨트롤러에서 춘식이 스킬 사용 가능 상태로 변경할지 결정하는 로직 구현
- 각 스킬마다 수행할 로직을 블록 로직 매니저에서 수행하도록 구현
- QA 및 알파체크에 용이하도록 키보드 및 디바이스를 이용한 치트 기능 구현

Onet Adventure



게임명 | Onet Adventure

장르 | 캐주얼 퍼즐

플랫폼 | Android, IOS

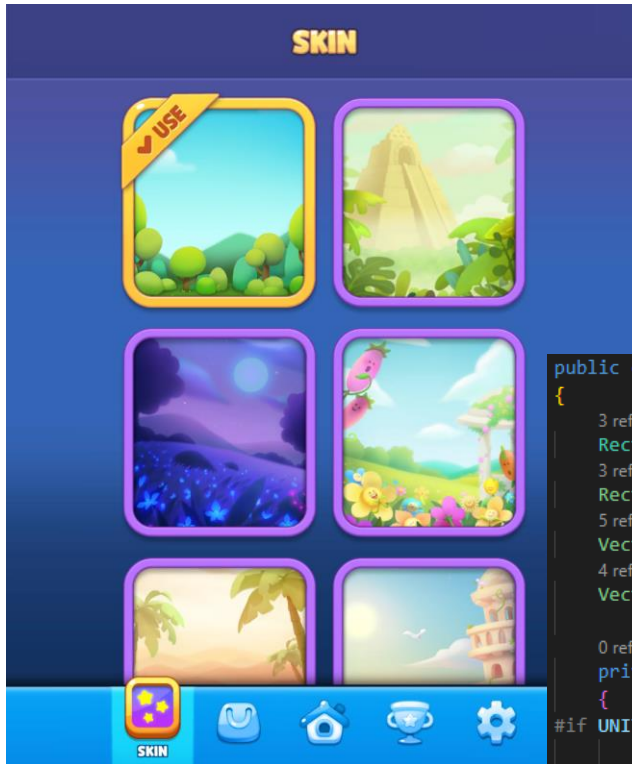
엔진 | Unity3D

완성도 | 신작 개발 -> 서비스 종료

담당 파트 | UI UX개발, 신규 콘텐츠 개발,

- Unity3D를 이용한 2D 캐주얼 퍼즐게임.
- UGUI, Spine 등을 이용하여 UI를 구현.
- Firebase와 Google Cloud를 이용하여 유저 데이터, CDN을 관리.

Onet Adventure



```
public class SafeArea : MonoBehaviour
{
    3 references
    RectTransform rectTransform;
    3 references
    Rect safeArea;
    5 references
    Vector2 minAnchor;
    4 references
    Vector2 maxAnchor;

    0 references
    private void Awake()
    {
        #if UNITY_EDITOR
            return;
        #endif

        // safeArea를 받아서 min앵커와 max앵커에 position을 부여
        rectTransform = GetComponent<RectTransform>();
        safeArea = Screen.safeArea;
        minAnchor = safeArea.position;
        maxAnchor = minAnchor + safeArea.size;

        //인스펙터 프로퍼티에 집어 넣을수 있게 비율로 변환 및 할당
        minAnchor.x /= Screen.width;
        minAnchor.y /= Screen.height;
        maxAnchor.x /= Screen.width;
        maxAnchor.y /= Screen.height;

        rectTransform.anchorMin = minAnchor;
        rectTransform.anchorMax = maxAnchor;
    }
}
```

구현 내용 소개

Safe Area

- Screen.safeArea를 사용하여 기기 별 해상도 대응 및 노치 대응.

Onet Adventure

```
FireBaseManager.Instance.FireStore.GameResult(stage_key, isClear, remainTime, rewardCoin, resultInfo, (msPacketBase p) =>
{
    GameConfig.Set("Poppys_Help_Reset", false);

    resultfunc?.Invoke(resultInfo, p.ERROR == 0 ? true : false);

    UI_Indicator.Instance.CloseIndicator(msPacketType.Packet_ResultGame);
});
```

```
public void GameResult(int stage_key, bool isClear, int remainTime, int rewardCoin, GameInfo
{
    fb_UserInfo info = _ConvertPlayerInfoToUserInfo();
    if(isClear == true)
    {
        info.lastClearStageKey = stage_key;
        info.remainTime = remainTime;

        int callBackCount = 0;
        _QuerySetUser(info.toDictionary(), (success) =>
        {
            if (isClear)
```

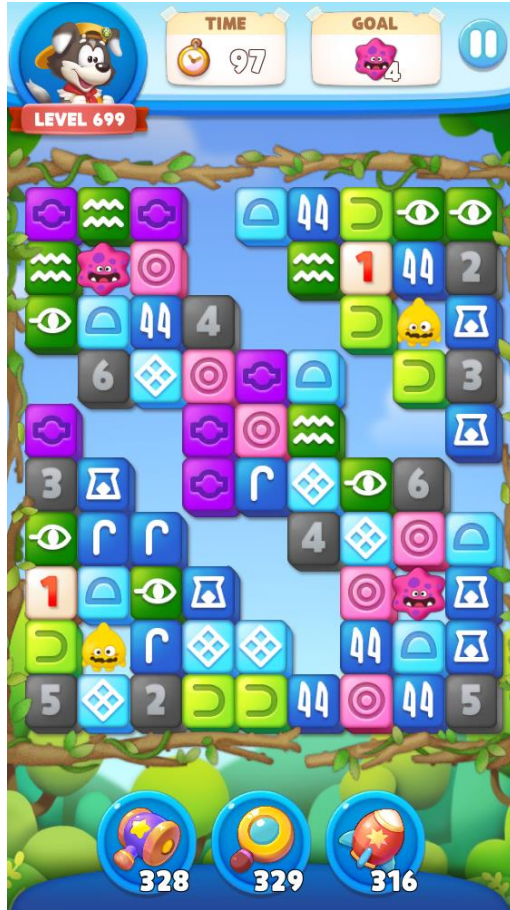
```
internal void _QuerySetUser(DocumentReference user, object data, Action<bool> onComplete)
{
    user.SetAsync(data, SetOptions.MergeAll).ContinueWith(queryTask =>
    {
        SyncContext.RunOnUnityThread(() =>
        {
            if (queryTask.IsFaulted || queryTask.IsCanceled)
            {
                Debug.Log("Not Update UserData : " + queryTask.Exception.Message);
                onComplete?.Invoke(false);
            }
            else
            {
                onComplete?.Invoke(true);
            }
        });
    });
}
```

구현 내용 소개

Firebase 데이터베이스 연동

- Firebase에 유저 정보 및 아이템 등의 정보를 저장하고, 해당 정보를 불러와 UI 표시에 사용하는 기능 구현

Onet Adventure



구현 내용 소개

스테이지 모드 추가

- 테이블로 관리되는 스테이지
데이터를 불러와 해당 스테이지의
모드에 따른 인게임 UI 변경 기능
구현

신규 블록 제작

- 보너스 모드에서 사용하는 코인 블록,
조건 모드에서 사용하는 토템 블록
구현

구현 내용 소개

```
1 reference
void _StartGame(int map_key)
{
    _mapInfo = DataManager.Instance.GetDataInfo<msDataInfoMapInfo>(map_key, E_DATAINFO_TYPE.MapInfo);

    string stageStep = string.Empty;
    if (DataManager.Instance.ConvertMapKeyToStageStepText(_mapInfo.Key, out stageStep) == false)
        return;

    if (_txtStage)
        _txtStage.text = string.Format(msGameText.Instance.GetText("Text_Common_Stage_0"), stageStep);

    if (_txtGoalTitle)
        _txtGoalTitle.text = msGameText.Instance.GetText(string.Format("Text_UI_Ingame_{0}", _mapInfo.Clear_Type == E_Clear_Type.Bonus ? "Reward" : "Goal"));
    //보너스 스테이지의 경우에만 리워드로 바뀜

    if (_goBonusAnimations)
        UIUtil.SetActive(_goBonusAnimations, _mapInfo.Clear_Type == E_Clear_Type.Bonus);
    UIManager.SendMessage(E_UIType.UI_BackgroundPanel, (int)UI_BackgroundPanel.E_Sub_Type.SetBonusCoverActive, _mapInfo.Clear_Type == E_Clear_Type.Bonus);

    if (_txtTimeTitle)
        _txtTimeTitle.text = msGameText.Instance.GetText("Text_UI_Ingame_Time");

    _SetTimeAnimation(false, true);
    _UpdateRemainTime(0);

    msPlayerInfo.Instance.List_IngameUseItemPacket.Clear();

    UIUtil.SetActive(_imgComboBonus, false);
}
```

스테이지 모드 추가

- 테이블로 관리되는 스테이지 데이터를 불러와 해당 스테이지의 모드에 따른 인게임 UI 변경 기능 구현

신규 블록 제작

- 보너스 모드에서 사용하는 코인 블록, 조건 모드에서 사용하는 토템 블록 구현

Onet Adventure

```
int _GetGoalCount()
{
    //보너스 스테이지의 경우 클리어조건 대신 획득 코인 수를 표시
    if (_mapData.Clear_Type == E_Clear_Type.Bonus)
        return InGameManager.Instance.acquiredCoinBlockCoin;

    int goalCount = 0;
    for(int i = 0; i < _map_size_width; ++i)
    {
        for(int j = 0; j < _map_size_height; ++j)
        {
            if(!_lstGameBlock[i, j].IsActive == true && _lstGameBlock[i, j].BlockInfo != null
            && _lstGameBlock[i, j].BlockInfo.BlockInfoData != null)
            {
                if(_mapData.Clear_Type == E_Clear_Type.Poppy)
                {
                    if(_lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Poppy)
                        goalCount += _lstGameBlock[i, j].BlockInfo.BlockInfoData.Key - POPPY_BLOCK_BASE_INDEX;
                }
                else if(_mapData.Clear_Type == E_Clear_Type.Balloon)
                {
                    if(_lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Balloon)
                        ++goalCount;
                }
                else if(_mapData.Clear_Type == E_Clear_Type.Totem)
                {
                    if(_lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Totem)
                        ++goalCount;
                }
                else if(_mapData.Clear_Type == E_Clear_Type.Normal)
                {
                    if (_lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Blink ||
                        _lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Fate ||
                        _lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Hint ||
                        _lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Key ||
                        _lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Lock ||
                        _lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Normal ||
                        _lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Number ||
                        _lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Question ||
                        _lstGameBlock[i, j].BlockInfo.BlockInfoData.BlockType == E_Block_Type.Time)
                        ++goalCount;
                }
            }
        }
    }
    return goalCount;
}
```

구현 내용 소개

스테이지 모드 추가

- 테이블로 관리되는 스테이지 데이터를 불러와 해당 스테이지의 모드에 따른 인게임 UI 변경 기능 구현

신규 블록 제작

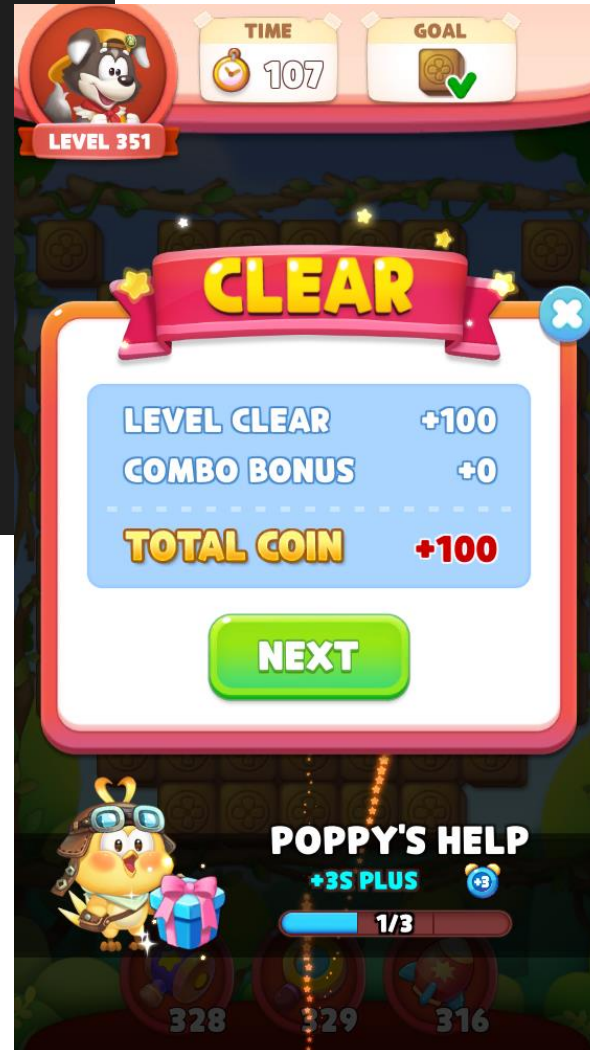
- 보너스 모드에서 사용하는 코인 블록, 조건 모드에서 사용하는 토템 블록 구현

Onet Adventure

```
//Coin Result
_ShowResultCoinText(_txtClearScore, _txtClearScoreValue, InGameManager.Instance.acquiredLevelClearCoin + InGameManag
{
    _ShowResultCoinText(_txtComboScore, _txtComboScoreValue, InGameManager.Instance.acquiredComboBonusCoin, ()=>
    {
        _ShowResultCoinText(_goTotalScore, _txtTotalScoreValue, InGameManager.Instance.GetTotalAcquiredCoin(), ()=>
        {
            if (_poppysHelp)
            {
                _poppysHelp.ShowPoppyStep(false, () =>
                {
                    //Coin Result Finish Callback
                    _isAnimating = false;
                    if (_btnCoverBtn)
                    {
                        _btnCoverBtn.gameObject.SetActive(false);
                    }
                });
            }
            else
            {
                //Coin Result Finish Callback
                _isAnimating = false;
                if (_btnCoverBtn)
                {
                    _btnCoverBtn.gameObject.SetActive(false);
                }
            }
        });
    });
};
```

```
void _ShowResultCoinText(Text nameText, Text valueText, int coinValue, System.Action callback)
{
    if (coinValue >= 1)
        SoundManager.Instance.PlayEFXSound(SOUND_LIST.InGame_Clear_success_score);
    UIUtil.SetActive(nameText, true);
    nameText.color = new Color(1, 1, 1, 0);
    nameText.DOFade(1, 0.3f);
    if (valueText)
        UIUtil.ClimbUpInt(this, valueText, "+", 30, 0.5f, 0, coinValue, callback);
}

1 reference
void _ShowResultCoinText(GameObject nameGO, Text valueText, int coinValue, System.Action callback)
{
    if (coinValue >= 1)
        SoundManager.Instance.PlayEFXSound(SOUND_LIST.InGame_Clear_success_score);
    UIUtil.SetActive(nameGO, true);
    if (valueText)
        UIUtil.ClimbUpInt(this, valueText, "+", 30, 0.5f, 0, coinValue, callback);
}
```



구현 내용 소개

결과 화면 연출 구현

- 플레이 중 결과화면에 따라 성공, 실패 화면 노출 및 해당 화면 연출 처리 구현
- (각 클리어 별 보상 숫자 상승 연출, 결과화면 애니메이션 및 파티클 연출)
- 결과 화면에서 노출되어야 하는 이벤트 연출 기능 구현
- (게이지 상승, 스텝에 따른 지급 효과 이미지 변경 등)

감사합니다.