Міністерство науки і освіти України

Національний університет “Львівська політехніка”

Курсова робота

з дисципліни «Мобільні Інфокомунікаційні платформи» на тему:
«Створення гри для мобільних пристроїв на основі OS Android в середовищі Unity»

Виконав:
ст. гр. ТР – 45
Пилип С.С.
Прийняв:
Бурачок Р.А.

Львів 2021

Зміст

Вступ

Метою курсової роботи є розробка мобільного додатку для операційної системи Android. Для розробки мобільного додатку було вибрано середовище розробки Unity. Написання логіки гри здійснювалось на мові програмування C#.

Розділ 1. Unity

Серед ігрових движків Unity займає далеко не останнє місце. Його використовують і великі розробники, і (набагато частіше) невеликі незалежні студії. У цій статті ми розповімо про особливості, сильні та слабкі сторони двигуна, а також типи проектів, в яких його використання буде найбільш доцільним.

Unity - більше, ніж двигун, це середовище для розробки комп'ютерних ігор, в якій об'єднані різні програмні засоби, що використовуються при створенні програмного забезпечення - текстовий редактор, компілятор, відладчик і так далі. При цьому, завдяки зручності використання, Unity робить створення ігор максимально простим і комфортним, а мультиплатформенність двигуна дозволяє гравцям охопити якомога більшу кількість ігрових платформ та операційних систем.

Переваги Unity:

Насамперед, як ми вже згадували, двигун Unity дає можливість розробляти ігри, не вимагаючи для цього якихось спеціальних знань. Тут використовується компонентно-орієнтований підхід, у якого розробник створює об'єкти (наприклад, головного героя) і до них додає різні компоненти (наприклад, візуальне відображення персонажа та способи управління ним). Завдяки зручному Drag & Drop інтерфейсу і функціональному графічному редактору двигун дозволяє малювати карти і розставляти об'єкти в реальному часі і відразу ж тестувати результат, що вийшов.

Друга перевага движка – наявність величезної бібліотеки ассетів та плагінів, за допомогою яких можна значно прискорити процес розробки гри. Їх можна імпортувати та експортувати, додавати в гру цілі заготівлі – рівні, ворогів, патерни поведінки II тощо. Ніякої метушні з програмуванням . Багато ассет доступні безкоштовно, інші пропонуються за невелику суму, і при бажанні можна створювати власний контент, публікувати його в Unity Asset Store і отримувати від цього прибуток.

Третя сильна сторона Unity 3D – підтримка величезної кількості платформ, технологій, API. Створені на движку ігри можна легко портувати між ОС Windows, Linux, OS X, Android, iOS, на консолі сімей PlayStation, Xbox, Nintendo, на VR-і AR-пристрої. Unity підтримує DirectX та OpenGL, працює з усіма сучасними ефектами рендерингу, включаючи новітню технологію трасування променів у реальному часі.

Фізика жорстких тіл, ragdoll і тканин, система Level of Detail, колізії між об'єктами, складні анімації - все це можна продати силами двигуна. Стереотипна думка про те, що двигун придатний тільки для невеликих інді-ігор і нездатний видавати красиву картинку, давно вже не актуально: достатньо подивитися технодемо ADAM, The Blacksmith і Book of the Dead від творців середовища Unity, щоб переконатися в її видатних здібностях.

Нарешті Unity доступний безкоштовно, що відкриває перед незалежними розробниками двері в ігрову індустрію. Звичайно, існують обмеження: безкоштовна версія двигуна демонструє логотип Unity перед запуском гри, а проект, створений з її допомогою, не повинен приносити розробнику більше $100 тисяч на рік. Втім, тарифи на підписку не спустошать гаманці навіть команди-початківця: Про-версія коштує $125 на місяць, що не так вже й багато в порівнянні з іншими двигунами, причому базова версія містить рівно той же функціонал, що і професійна.

Недоліки Unity:

За всіх своїх переваг, двигун має і свої недоліки. Так, якщо команда захоче розробити щось складніше простого клікера або платформера , то їй доведеться шукати хорошого програміста на C#, який напише скрипти та компоненти, впровадить їх у гру та змусить працювати.

З цього випливає інша проблема двигуна Unity - повільність. Створення масштабних, складних сцен з безліччю компонентів може негативно вплинути на продуктивність гри, внаслідок чого розробникам доведеться витратити додатковий час та ресурси на оптимізацію , а можливо – видалення деяких елементів із проекту.

Крім того, додатки, створені на Unity, досить «важкі»: навіть найпростіша піксельна гра може займати кілька сотень мегабайт на ПК. Так, для жорсткого диска комп'ютерів це невеликий обсяг, але якщо проект розробляється і для мобільних платформ, слід задуматися про оптимізацію його розміру.

Розділ 2. Розробка Гри

Візуальна Складова:

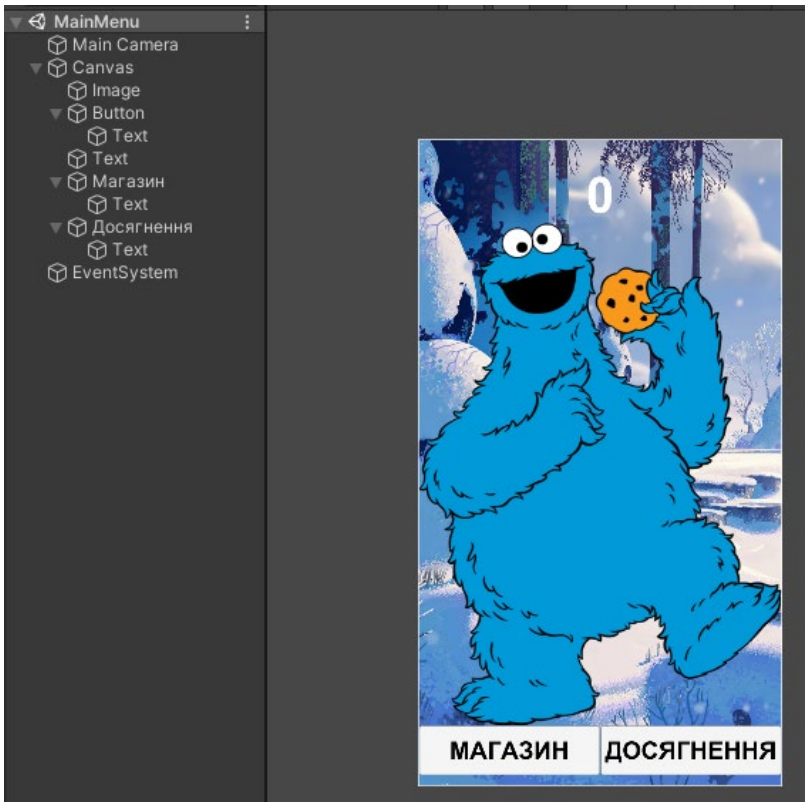
Візуальна складова, з якою буде взаємодіяти користувач, налаштована в середовищі Unity.

При першому запуску середовища створюємо проект нашої гри та налаштовуємо наступні параметри:

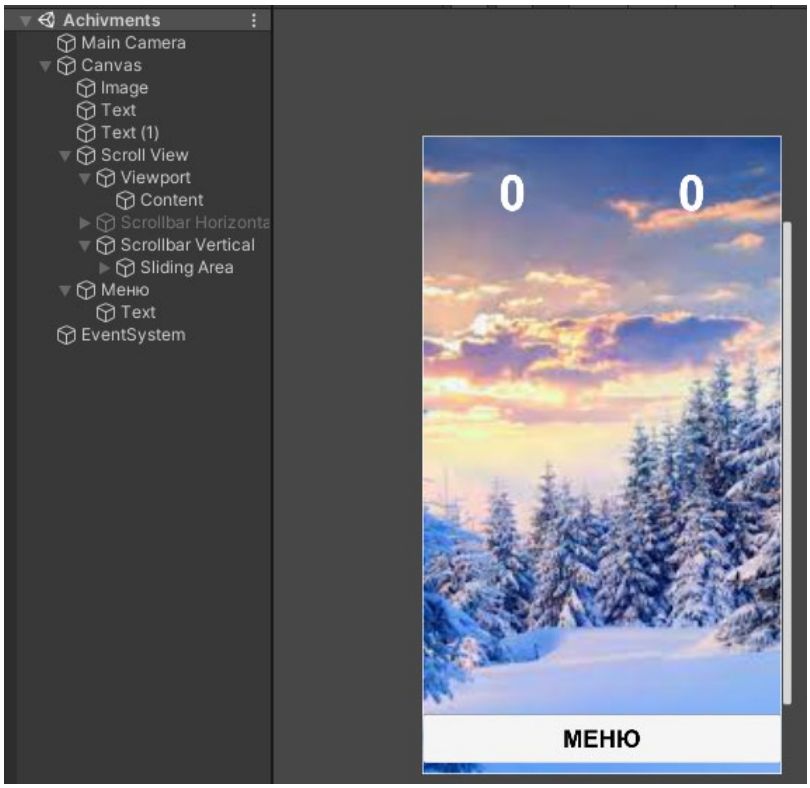
- Тип ОС: Android
- Тип підтримуваної архітектури: ARM64
- Мінімальна версія ОС: Android 4.4 ‘KitKat’
- Назва компанії: SergiosProdaction
- Назва гри: Clicker Game
- Версія: 1.4(Остання встановлена мною версія)

Дана гра має три сцени які виглядають так:

1) Сцена MainMenu:



2) Сцена Achivments:



3) Сцена MainMenu:

User interface (UI) елементи – це частини, які дизайнери використовують для створення програм або веб-сайтів. Вони додають інтерактивність в інтерфейс користувача, надаючи користувачеві точки дотику при навігації по них. Кнопки, смуги прокручування, пункти меню та чекбокси.

В даних Сценах використовувались наступні UI елементи:

1. Canvas – елемент на якому містяться всі основні UI елементи, тобто так зване ‘Полотно’.

2. Text – елемент для роботи з текстом. Використовувались для відображення ігрової валюти, кількості зроблених клікі, ціни в магазині та опису кнопок.

3. Button – основний елемент гри який виконує функції кнопки. Використовується для зчитування дій які гравець може робити в грі(заробіток ігрової валюти, покупки в магазині тощо).

4. Image – елемент для відображення зображень. Використовується для встановлення фону гри.

5. Scroll View – елемент за допомогою якого можна встановлювати безліч елементів які можна переглядати за допомогою ‘Scroll’(прокрутки).

Функціональна складова:

Вся логіка додатку написана в трьох файлах .cs, які написані мовою C#.

Файл MainMenu.cs є головним файлом у якому створені основні змінні для підрахунку зароблених коштів та кількості натисків на кнопу.

```
/*Підключння необхідних бібліотек для роботи з елементами Unity*/
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
/*Основний клас для роботи сцени MainMenu*/
public class MainMenu : MonoBehaviour
```

```
{
    /*Основні змінні даного класу*/
    [SerializeField] int money;
    public int total_money;
    public Text moneyText;
    public int K;
    [SerializeField] int C;
```

Також створені функція задання початкових параметрів Start(). Функції для підрахунку кількості кліків та суми ігрової валюти ButtonClick(). Продубльована функція для авто-кліків IdleFarm(), яка починає працювати пісяля виконання першого досягнення. Функція переходу до меню Досягнень ToAchievements(). Функція переходу до Меню магазину ToShop(). Функція яка оновлюється 20 разів за секунду і відображає зміну клількості зароблених коштів Update().

```
    /*Функція задання параметрів при кожному запуску сцени*/
    private void Start()
    {
        money = PlayerPrefs.GetInt("money");
        total_money = PlayerPrefs.GetInt("total_money");
        K = PlayerPrefs.GetInt("K");
        C = PlayerPrefs.GetInt("C");
        StartCoroutine(IdleFarm());//ферма

    }
    /*Функція роботи Кнопки на яку ми натискаємо*/
    public void ButtonClick()
    {
        money+=1+C;
        total_money++;
        PlayerPrefs.SetInt("money", money);
```

```
        PlayerPrefs.SetInt("total_money", total_money);
    }
    /*Функція авто-кліків*/
    IEnumerator IdleFarm()
    {
        K = PlayerPrefs.GetInt("K");
        yield return new WaitForSeconds(1);
        money+=K;
        PlayerPrefs.SetInt("money", money);
        StartCoroutine(IdleFarm());
    }
    /*Функція переходу до меню Досягнень*/
    public void ToAchievements()
    {
        SceneManager.LoadScene(1);
    }
    /*Функція переходу до меню Магазину*/
    public void ToShop()
    {
        SceneManager.LoadScene(2);
    }
    /*Функція оновлення виводу кількості зароблені ігрової валюти*/
    void Update()
    {
        moneyText.text = money.ToString()+" $";

    }
}
```

Файл AchMenu.cs є файлом сцени Achivments у якому створені основні змінні для підтвердження виконання досягнень та обєкти за допомогою яких можна створити конопки по шаблону і помістити їх в слайдер та призначити дію цим кнопакам.

```
/*Підключння необхідних бібліотек для роботи з елементами Unity*/
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
/*Основний клас для роботи сцени Achivments*/
public class AchMenu : MonoBehaviour
{
   /*Основні змінні даного класу*/
   public Text moneyText, total_moneyText;
   public int money;
   public int total_money;
   [SerializeField] int K;//змінна для роботи авто-кліків
   private bool SET0, SET1, SET2, SET3, SET4, SET5;//змінні виконання досягень

   public string[] arrayTitles;// назва досягнення
   public Sprite arraySprites;// масив картинок для досягнень
   public GameObject button;// шаблон кнопки
   public GameObject content;

   private List<GameObject> list = new List<GameObject>();
   private VerticalLayoutGroup _group;
```

Також створені функція задання початкових параметрів Start(). Допоміжна функція для очистки елементів RemovedList(). Функція для створення кнопок SetAchievs(), які поміщаються в Scroll View -> Content. Функція для опрацювання виконання досягнень GetAchievment(), після виконання яких гравцеві надається доступ до автоматичного заробляння коштів. Функція для авто-кліків IdleFarm(), яка починає працювати після виконання першого досягнення. Функція переходу до Головного Меню ToMenu(). Функція яка оновлюється 20 разів за секунду і відображає зміну клількості зароблених коштів та зроблених кліків Update().

```
/*Функція задання параметрів при кожному запуску сцени*/
void Start()
{
    money = PlayerPrefs.GetInt("money");
    total_money = PlayerPrefs.GetInt("total_money");
    K = PlayerPrefs.GetInt("K");

    SET0 = PlayerPrefs.GetInt("SET0") == 1 ? true : false;
    SET1 = PlayerPrefs.GetInt("SET1") == 1 ? true : false;
    SET2 = PlayerPrefs.GetInt("SET2") == 1 ? true : false;
    SET3 = PlayerPrefs.GetInt("SET3") == 1 ? true : false;
    SET4 = PlayerPrefs.GetInt("SET4") == 1 ? true : false;
    SET5 = PlayerPrefs.GetInt("SET5") == 1 ? true : false;

    RectTransform rectT = content.GetComponent<RectTransform>();
    rectT.transform.localPosition = new Vector3(0.0f, 0.0f, 0.0f);// нульові кординати для кнопок
    _group = GetComponent<VerticalLayoutGroup>();
    setAchievs();
    /*Перевірка виконнання досягнень та вимикання їх повторного отримання*/
    if (SET0 == true)
```

```
        {
            list[0].GetComponent<Button>().interactable = false;
        }
        if (SET1 == true)
        {
            list[1].GetComponent<Button>().interactable = false;
        }
        if (SET2 == true)
        {
            list[2].GetComponent<Button>().interactable = false;
        }
        if (SET3 == true)
        {
            list[3].GetComponent<Button>().interactable = false;
        }
        if (SET4 == true)
        {
            list[4].GetComponent<Button>().interactable = false;
        }
        if (SET5 == true)
        {
            list[5].GetComponent<Button>().interactable = false;
        }
        StartCoroutine(IdleFarm());

    }
    /*Допоміжна функція очистки елементів*/
    public void RemovedList()
    {
```

```
            foreach (var elem in list)
            {
                Destroy(elem);
            }
            list.Clear();
        }
        /*Функція створення кнопок для отримання Досягнень*/
        void setAchievs()
        {
            RectTransform rectT = content.GetComponent<RectTransform>();//зчитування кнопки шаблону
            rectT.transform.localPosition = new Vector3(-360.0f, 10.0f, 0.0f);//встановлення початкових координат кнопки
            RemovedList();
            if (arrayTitles.Length > 0)
            {
                var pr1 = Instantiate(button, transform);// створення образца
                var h = pr1.GetComponent<RectTransform>().rect.height;// запис висоти
                var tr = GetComponent<RectTransform>();// створення змінної
                tr.sizeDelta = new Vector2(tr.rect.width, h * arrayTitles.Length);
                Destroy(pr1);
                for(var i = 0; i < arrayTitles.Length; i++)//цикл досягнень
                {
                    var pr = Instantiate(button, transform);
                    pr.GetComponentInChildren<Text>().text = arrayTitles[i];//запис тексту
                    pr.GetComponentInChildren<Image>().sprite = arraySprites;//запис іконки
                    var i1 = i;
                    pr.GetComponent<Button>().onClick.AddListener(() => GetAchievement(i1));// опрацювання нажиму на кнопки
```

```
            list.Add(pr);// добавити досягнення до списку
        }
    }
}

/*Функція опрацювання виконання Досягень*/
void GetAchievement(int id)
{
    switch(id)
    {
        case 0:
            if (!SET0 && total_money >= 10)
            {
                money += 10;
                K += 1;
                SET0 = true;
                PlayerPrefs.SetInt("money", money);
                PlayerPrefs.SetInt("K", K);
                PlayerPrefs.SetInt("SET0", 1);
                list[0].GetComponent<Button>().interactable = false;
            }
            Debug.Log(id);
            break;
        case 1:
            if (!SET1 && total_money >= 50)
            {
                money += 50;
                K += 5;
                SET1 = true;
```

```
            PlayerPrefs.SetInt("money", money);
            PlayerPrefs.SetInt("K", K);
            PlayerPrefs.SetInt("SET1", 1);
            list[1].GetComponent<Button>().interactable = false;
        }
        Debug.Log(id);
        break;
    case 2:
        if (!SET2 && total_money >= 100)
        {
            money += 100;
            K += 10;
            SET2 = true;
            PlayerPrefs.SetInt("money", money);
            PlayerPrefs.SetInt("K", K);
            PlayerPrefs.SetInt("SET2", 1);
            list[2].GetComponent<Button>().interactable = false;
        }
        Debug.Log(id);
        break;
    case 3:
        if (!SET3 && total_money >= 500)
        {
            money += 500;
            K += 50;
            SET3 = true;
            PlayerPrefs.SetInt("money", money);
            PlayerPrefs.SetInt("K", K);
            PlayerPrefs.SetInt("SET3", 1);
```

```
            list[3].GetComponent<Button>().interactable = false;
        }
        Debug.Log(id);
        break;
    case 4:
        if (!SET4 && total_money>= 1000)
        {
            money += 1000;
            K += 100;
            SET4 = true;
            PlayerPrefs.SetInt("money", money);
            PlayerPrefs.SetInt("K", K);
            PlayerPrefs.SetInt("SET4", 1);
            list[4].GetComponent<Button>().interactable = false;
        }
        Debug.Log(id);
        break;
    case 5:
        if (!SET5 && total_money >= 10000)
        {
            money += 10000;
            K += 1000;
            SET5 = true;
            PlayerPrefs.SetInt("money", money);
            PlayerPrefs.SetInt("K", K);
            PlayerPrefs.SetInt("SET5", 1);
            list[5].GetComponent<Button>().interactable = false;
        }
        Debug.Log(id);
```

```
                break;
            }
        }

        /*Функція авто-кліків*/
        IEnumerator IdleFarm()
        {
            K = PlayerPrefs.GetInt("K");
            yield return new WaitForSeconds(1);
            money+=K;
            PlayerPrefs.SetInt("money", money);
            Debug.Log(K);
            StartCoroutine(IdleFarm());
        }
        /*Функція повернення до сцени MainMenu*/
        public void ToMenu()
        {
            SceneManager.LoadScene(0);
        }

        /*Функція оновлення виводу кількості заробленої ігрової валюти та кількості накисків*/
        void Update()
        {
            total_moneyText.text = total_money.ToString() + " Clics";
            moneyText.text = money.ToString() + " $";
        }
    }
```

Файл ShopMenu.cs є файлом сцени Shop у якому створені основні змінні для можливості здійснювати покупки натискаючи на кнопки, змінні для показу ціни та змінні які збільшують вартість кожної наступної покупки, а також змінна яка збільшує кількість ігрової валюти яку гравець може отримати при одному натиску на кнопку.

```
/*Підключння необхідних бібліотек для роботи з елементами Unity*/
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

/*Основний клас для роботи сцени Shop*/
public class ShopMenu : MonoBehaviour
{
    /*Основні змінні даного класу*/
    public Text moneyText;
    public Text TSET0;
    public Text TSET1;
    public Text TSET2;
    public Text TSET3;
    public Text TSET4;
    public int money;
    public int K;
    private int FSET0, FSET1, FSET2, FSET3, FSET4;
    public int l = 1;
    public int C;
    private int sum0, sum1, sum2, sum3, sum4;
```

Також створені функція задання початкових параметрів Start(). Функція для опрацювання роботи першої кнопки магазину GetShop0(), в якій проходить операція покупки додаткової суми, яку може отримати гравець та перерахунок вартості наступної покупки. Функція для опрацювання роботи другої кнопки магазину GetShop1(). Функція для опрацювання роботи третьої кнопки магазину GetShop2(). Функція для опрацювання роботи четвертої кнопки магазину GetShop3(). Функція для опрацювання роботи п'ятої кнопки магазину GetShop4(). Дублювання функції для авто-кліків IdleFarm. Функція переходу до Головного Меню ToMenu(). Функція яка оновлюється 20 разів за секунду і відображає зміну клількості зароблених коштів та суми кожної наступної покупки в магазині Update().

```
/*Функція задання параметрів при кожному запуску сцени*/
    void Start()
    {
        money = PlayerPrefs.GetInt("money");
        C = PlayerPrefs.GetInt("C");
        //total_money = PlayerPrefs.GetInt("total_money");
        K = PlayerPrefs.GetInt("K");

        FSET0 = PlayerPrefs.GetInt("FSET0");
        FSET1 = PlayerPrefs.GetInt("FSET1");
        FSET2 = PlayerPrefs.GetInt("FSET2");
        FSET3 = PlayerPrefs.GetInt("FSET3");
        FSET4 = PlayerPrefs.GetInt("FSET4");

        sum0 = (int)(Mathf.Pow(1.2f, FSET0) * 10);
        sum1 = (int)(Mathf.Pow(1.2f, FSET1) * 50);
        sum2 = (int)(Mathf.Pow(1.2f, FSET2) * 100);
```

```
    sum3 = (int)(Mathf.Pow(1.2f, FSET3) * 500);
    sum4 = (int)(Mathf.Pow(1.2f, FSET4) * 1000);

    StartCoroutine(IdleFarm());
}

/*Функція роботи кнопки 1*/
public void GetShop0()
{
    float d;
    Debug.Log(sum0);
    if (money >= sum0)
    {
        money -= sum0;
        FSET0 ++;
        C += 1;

        PlayerPrefs.SetInt("money", money);
        PlayerPrefs.SetInt("C", C);
        PlayerPrefs.SetInt("FSET0", FSET0);
    }
    d = 10 * Mathf.Pow(1.2f,FSET0);
    sum0 = (int)d;
    Update();
}
/*Функція роботи кнопки 2*/
public void GetShop1()
{
    float d;
```

```
    if (money >= sum1)
    {

        money -= sum1;
        FSET1 ++;
        C += 5;

        PlayerPrefs.SetInt("money", money);
        PlayerPrefs.SetInt("C", C);
        PlayerPrefs.SetInt("FSET1", FSET1);

    }
    d = 50 * Mathf.Pow(1.2f, FSET1);
    sum1 = (int)d;
    Update();
}
/*Функція роботи кнопки 3*/
public void GetShop2()
{
    float d;

    if (money >= sum2)
    {

        money -= sum2;
        C += 10;
        FSET2 ++;
        PlayerPrefs.SetInt("money", money);
```

```
        PlayerPrefs.SetInt("C", C);
        PlayerPrefs.SetInt("FSET2", FSET2);

    }
    d = 100 * Mathf.Pow(1.2f, FSET2);
    sum2 = (int)d;
    Update();
}
/*Функція роботи кнопки 4*/
public void GetShop3()
{
    float d;

    if (money >= sum3)
    {

        money -= sum3;
        C += 50;
        FSET3 ++;
        PlayerPrefs.SetInt("money", money);
        PlayerPrefs.SetInt("C", C);
        PlayerPrefs.SetInt("FSET3", FSET3);

    }
    d = 500 * Mathf.Pow(1.2f, FSET3);
    sum3 = (int)d;
    Update();
}
/*Функція роботи кнопки 5*/
```

```
public void GetShop4()
{
    float d;

    if (money >= sum4)
    {
        money -= sum4;
        C += 100;
        FSET4 ++;
        PlayerPrefs.SetInt("money", money);
        PlayerPrefs.SetInt("C", C);
        PlayerPrefs.SetInt("FSET4", FSET4);

    }
    d = 1000 * Mathf.Pow(1.2f, FSET4);
    sum4 = (int)d;
    Update();
}

/*Функція авто-кліків*/
IEnumerator IdleFarm()
{
    K = PlayerPrefs.GetInt("K");
    yield return new WaitForSeconds(1);
    money += K;
    PlayerPrefs.SetInt("money", money);

    Debug.Log(C);
    Update();
```

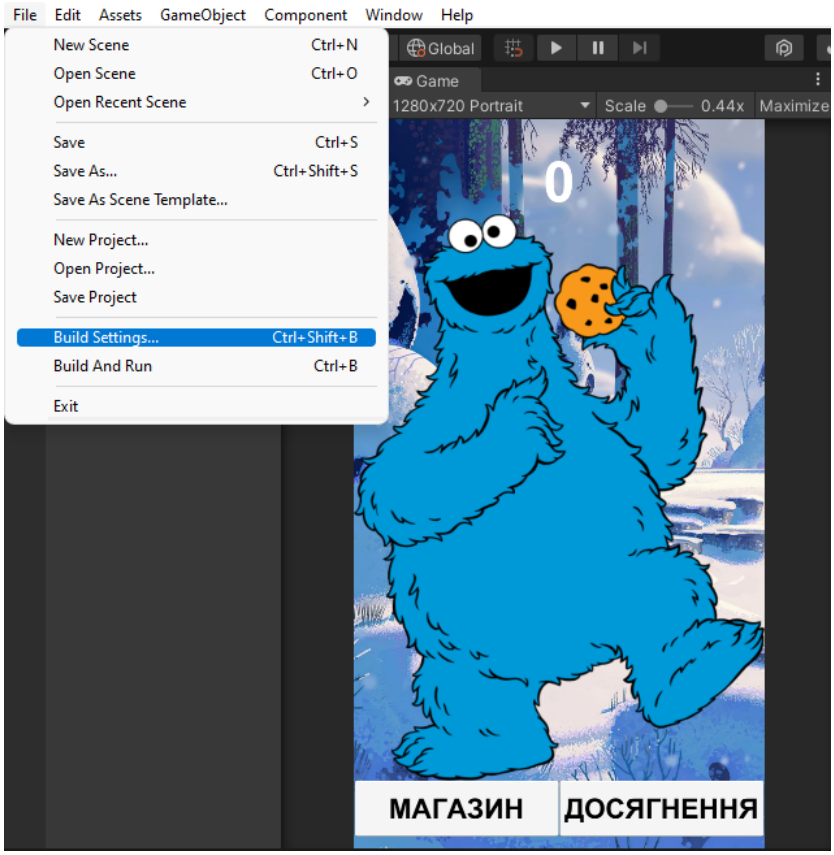
```
            StartCoroutine(IdleFarm());
        }
        /*Функція повернення до сцени MainMenu*/
        public void ToMenu()
        {
            PlayerPrefs.SetInt("FSET0", FSET0);
            PlayerPrefs.SetInt("FSET1", FSET1);
            PlayerPrefs.SetInt("FSET2", FSET2);
            PlayerPrefs.SetInt("FSET3", FSET3);
            PlayerPrefs.SetInt("FSET4", FSET4);
            PlayerPrefs.Save();
            SceneManager.LoadScene(0);
        }
        /*Функція оновлення виводу кількості зароблені ігрової валюти та вартості
покупок*/
        void Update()
        {
            moneyText.text = money.ToString() + " $";

            TSET0.text = sum0.ToString();
            TSET1.text = sum1.ToString();
            TSET2.text = sum2.ToString();
            TSET3.text = sum3.ToString();
            TSET4.text = sum4.ToString();
        }
    }
```

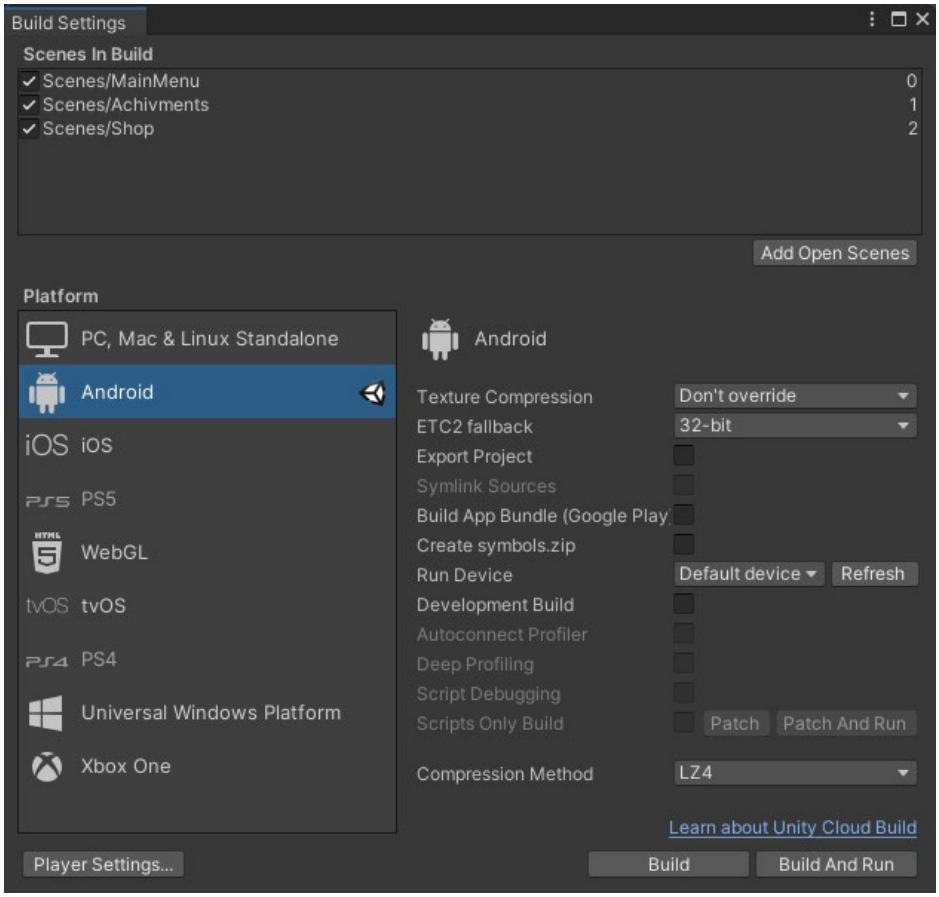
Розділ 3. Збір проекту

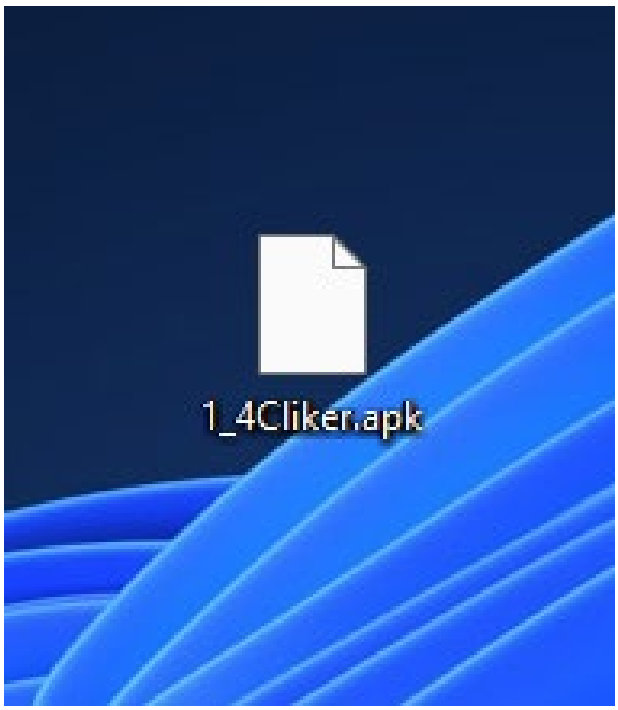
Для встановлення гри на сматрфон потрібно зібрати проект додатку в Unity.

Переходимо в розділ Build Settings, File - Build Settings. Дане меню потрібне для налаштування параметрів проекту та його побудови.

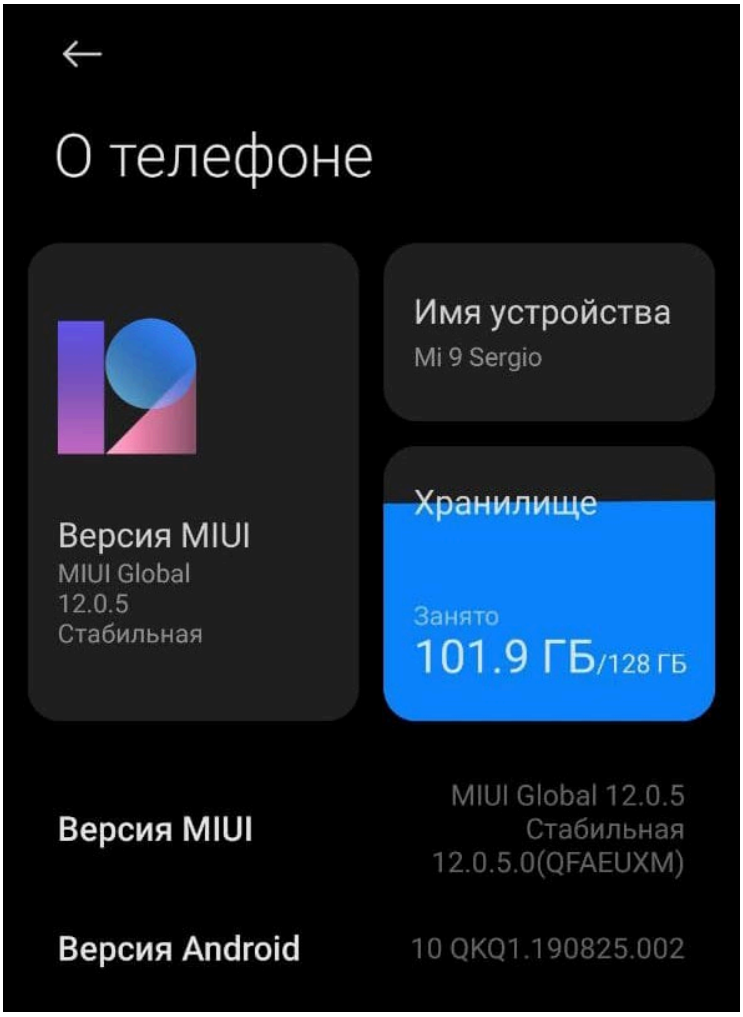


У вкладці Build Settings додаємо всі сцени нашої гри, нажимаємо кнопку Build та вибираємо назву нашого .APK файлу та куди його зберігати. Через декілька хвилин файл для встановлення гри буде готовий та збережений до папки, яку ми задали.



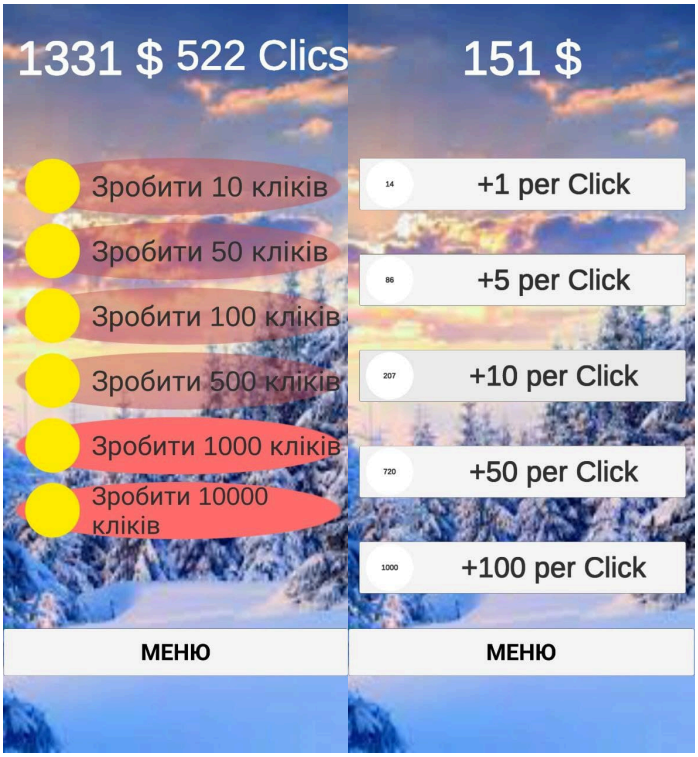


Після того як файл нашої гри був створений потрібно його перекинути на смартфон. Я використовуватиму смартфон Xiaomi Mi 9 з модифікованою ОС Android MIUI Global 12.0.5, яка використовує Android 10.



Далі встановлюємо додаток та заходимо до нього.

Виконання роботи програми продемонстровано на наступних скріншотах:



Також ознайомитись з грою можна за наступними посиланнями:
https://youtu.be/3D9QIOjQKpA
https://drive.google.com/file/d/1xhsIpjGFXGu2oOxfk3y45L9khx9dZFt1/view?usp=sharing

Висновок

Для виконання курсової роботи було використане середовище розробки Unity. Вся розробка велась за допомогою інтерфейсу Unity та мови програмування C#.

Дана гра є типу Клікер. В даній грі Гравець натискає кнопку, число збільшується. Натискає знову, число знову зростає. Гравець продовжує натискати і поступово розблоковує функцію, що збільшує число за гравця. І цей процес повторюється вічно .

По суті, такою є загальна структура гри. Вона здається простою, навіть примітивною, але в ігровому процесі є несподівана глибина та привабливість. Стилі гри можуть бути різними - від комерційно успішних та казуальних Clicker Heroes та AdVenture Capitalist до більш експериментальних або хардкорних прикладів, таких як Candy Box , Cookie Clicker та Sandcastle Builder .

В даній грі я використав 3 фундаментальні аспекти цього жанру. Наявність хоча б однієї валюти чи числа, які збільшуються із заданою швидкістю з мінімальними зусиллями, або взагалі без зусиль і які можна витрачати для збільшення швидкості приросту даної валюти.

Сама гра реалізована в трьох сценах. До кожної сцени було створено свій Скрипт(опис дій записаний за допомогою мови програмування).

Головна сцена. В даній сцені проходить головні дії гри, тобто заробіток валюти та підрахунок кількості зроблених кліків.

Сцена Досягнень. В якій гравець може отримати доступ до автоматичного заробітку валюти після виконання хоча б одного досягнення. Повторно досягення гравець не може виконати. Це зроблено для зберігання балансу гри.

Сцена Магазину. В Якій гравець може витратити свої кошти для збільшення валюти яку він може заробити за один клік. В логіці магазину вкладено те що сума для наступної покупки збільшується за формулою Ціна першої * (1.2)^кількість зроблених покупок.

Список Використаної Літератури

1) https://habr.com/ru/post/335754/

2) https://cubiq.ru/dvizhok-unity/

3) https://bool.dev/blog/detail/32-user-interface-elementov-dlya-ui-dizaynerov

4) https://proglib.io/p/sohranenie-igrovyh-dannyh-v-unity-2020-04-17

5) https://www.programiz.com/c-programming/library-function/math.h/pow

6) https://www.cyberforum.ru/unity/thread2389161.html

7) https://ru.stackoverflow.com/questions/566132/ui-%D0%BA%D0%BD%D0%BE%D0%BF%D0%BA%D0%B8-unity-%D0%BA%D0%B0%D0%BA-%D1%81%D0%B4%D0%B5%D0%BB%D0%B0%D1%82%D1%8C-%D0%B0%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B9-%D0%BD%D0%B5%D0%B0%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B9

8) https://docs.unity3d.com/ru/530/Manual/ExecutionOrder.html

9) https://www.cyberforum.ru/unity/thread2568046.html

10) http://www.c-cpp.ru/content/pow-powl

11) https://stackoverflow.com/questions/2544394/c-floating-point-to-integer-type-conversions

12) https://stackoverflow.com/questions/38459689/how-to-save-dictionary-data-or-array-list-data-to-playerprefs-more-effeciently

13) https://ru.stackoverflow.com/questions/714523/%D0%9A%D0%B0%D0%BA%D0%B8%D0%B5-%D0%B5%D1%81%D1%82%D1%8C-%D1%81%D0%BF%D0%BE%D1%81%D0%BE%D0%B1%D1%8B-

%D0%B4%D0%BE%D0%B1%D0%B0%D0%B2%D0%B8%D1%82%D1%8C-%D1%84%D0%BE%D0%BD%D0%BE%D0%B2%D1%83%D1%8E-%D0%BA%D0%B0%D1%80%D1%82%D0%B8%D0%BD%D0%BA%D1%83-%D0%B2-2%D0%94-%D0%B8%D0%B3%D1%80%D0%B5-%D0%B2-%D1%8E%D0%BD%D0%B8%D1%82%D0%B8

14) https://answers.unity.com/questions/1427421/should-not-be-capturing-when-there-is-a-hotcontrol.html

15) https://www.cyberforum.ru/unity/thread2347241.html

16) http://unity3d.ru/distribution/viewtopic.php?f=105&t=45190

17) https://www.youtube.com/watch?v=RbvZ0HWDi9E&list=PL3V36b1NObb_mrh2qMzcBY1j-17FBgW9L

18) https://www.youtube.com/watch?v=zyGJlWOjITQ&t=1s