Obchodní akademie, Vyšší odborná škola a Jazyková škola s právem státní jazykové zkoušky Uherské Hradiště

Technická dokumentace ke hře Jurské Třešně

Matěj Blaha, Petr Rídl, Jakub Vazan

16.5.2021

Obsah

Úvod

Jurské Třešně je singleplayer tower defense(později v dokumentu „TD“) hra, která je zasazená do pohádkového sporu dinosaurů a třešní. Cílová skupina této hry je 7-17 let, dokáže si ji však užít každá věková skupina. Jelikož jsou TD hry založeny na stejné herní mechanice, liší se naše hra pouze zasazením. Ostatní hry se většinou točí okolo sporu rytířů a monster, vojáků a nemrtvých,… Téma třešní a dinosaurů jsme si zvolili kvůli absurdnímu sporu malého ovoce a velkých bytostí.

Popis gameplaye

Samotný gameplay se odehrává v 10 jednotlivých levelech. Po každém takovém pokořeném levelu, se vám nový odemkne. Náročnost na rozmýšlení položení a kombinace třešní, ale roste s výší levelu. Již zmíněné třešně fungují jako "věže", jak je v TD hře zvykem. Mají různé schopnosti, ceny, slabé i silné stránky. Dinosauři zastávají roli nepřátel, kteří se hrnou k základně třešní. Dinosaurů je více druhů stejně jako třešní, ale narozdíl třešní, které stojí na jednom místě, se dinosauři pohybují po cestičce, která vyplývá z grafiky levelu.

Ovládání

Hra se ovládá pomocí myši a týká se pohybu po menu a kupování/vylepšování třešní. Hra se samozřejmě hraje z pohledu shora, aby hry byla co nejpřehlednější. Bude v režimu fullscreen, tudíž přes celou obrazovku. Nakupování bude fungovat systémem, kdy si kliknete na jednotku, kterou chcete koupit a ta bude cestovat s vaším kurzorem po bojišti, kliknutím ji položíte(koupíte).

Instalace

Po stažení souboru klikněte na odkaz, který naleznete zde. Stačí pouze spustit a hra je plně funkční.

Použité technologie

Na vývoj hry jsme použili:

Visual Studio Code (verze 1.56)

-Prostředí pro psaní kódu hry

Unity (verze 2020.3.0f1)

-Vývojové prostředí

Adobe Illustrator 2021

-Program na vytváření grafiky ke hře

Adobe Photoshop 2021

-Dolaďování maličkostí v oblasti grafiky

Technická stránka produktu

Charakter

Proměnné

```
public float speed;
```

Proměnná udávající rychlost jednotky

```
public int maxHP;
```

Proměnná udávající maximální zdraví jednotky

```
private Waypoints Wpoints;
```
Pomáhá pohybu jednotek po mapě pomocí bodů, po kterých jednotky chodí

```
private int waypointIndex;
```
Realizuje pohyb jednotek po mapě

```
private int myHP;
```
Umožňuje odečítání životů hráče

```
public static int damageGot;
```
Udává kolik životů jednotka hráči ubere

```
public int killIncome;
```
Přidává hráči peníze za eliminaci jednotky

Funkce

```
void Start()
    {
        Wpoints = GameObject.FindGameObjectWithTag("Waypoints").GetComponent<W
aypoints>();
        myHP = maxHP;
    }
```
Wpoints říká charakteru ať jde na bod na mapě

myHP nastavuje životy

```
 void Update()
    {
        if(damageGot != null){
            myHP -= damageGot;
            damageGot = 0;
        }
        transform.position = Vector2.MoveTowards(transform.position, Wpoints.w
aypoints[waypointIndex].position, speed * Time.deltaTime);

        Vector3 smer = Wpoints.waypoints[waypointIndex].position - transform.p
osition;
        float uhel = Mathf.Atan2(smer.y, smer.x) * Mathf.Rad2Deg;
        transform.rotation = Quaternion.AngleAxis(uhel, Vector3.forward);


        if(Vector2.Distance(transform.position, Wpoints.waypoints[waypointInde
x].position) < 0.1f)
        {
            if (waypointIndex < Wpoints.waypoints.Length - 1) waypointIndex++;
            else
            {
                Destroy(gameObject);
                Spawner.zivoty -= damageDo;
            }
        }
        if(myHP<= 0)
```

```
        {
            Destroy(gameObject);
            Spawner.penize += killIncome;


        }


    }
```
Umožňuje odečítání životů, zabíjení se při dosažení 0 životů a nastavování dalších bodů pro pohyb

```
public static void getDamage(int damage)
    {
        damageGot = damage;

    }
```
Realizuje dostávání poškození

Friendly

Proměnné

```
private Transform cil;
```
Zaměřování nepřátel

```
public int damage;
```
Nastavení poškození

```
public float range = 1f;
```
Dosah zaměřování

```
public int shootDelay;
```
Rychlost střelby

```
private int counter;
```
Počítadlo, které usnadňuje fungování jiných funkcí

```
private float rotace;
```
Otáčení přátelské jednotky za nepřátelskou(po označení za cíl)

```
public GameObject pfPecka;
```
Projektil, které střílí

```
public Transform poziceStrelby;
```
Místo, ze kterého projektil vyletí

```
private string nazevNepritele = "Nepritel";
```
Upravuje jméno pro lepší používání

Funkce

```
void upravCil()
    {
        GameObject[] nepratele = GameObject.FindGameObjectsWithTag(nazevNeprit
ele);
```

```
        float nejkratsiVzdalenost = Mathf.Infinity;
        GameObject nejblizsi = null;
        foreach(GameObject duch in nepratele)
        {
            float vzdalenost = Vector3.Distance (transform.position, duch.tran
sform.position);
            if(vzdalenost < nejkratsiVzdalenost)
            {
                nejkratsiVzdalenost = vzdalenost;
                nejblizsi = duch;
            }
        }
        if(nejblizsi != null && nejkratsiVzdalenost <= range)
        {
            cil = nejblizsi.transform;
        }
        if(nejkratsiVzdalenost > range)cil = null;

    }
```

Přepínání cílů

```
  void Update()
    {
        upravCil();
        counter++;
        if(counter >= shootDelay)
        {
            Utok();
            counter = 0;
        }


        Vector3 dir = cil.position - transform.position;

        transform.rotation = Quaternion.LookRotation(Vector3.forward, dir);
    }
```

Realizuje zaměřování cíle, otáčení, střelba a rychlost střelby

```
   private void Utok()
    {

        GameObject PeckaGO = (GameObject) Instantiate(pfPecka, poziceStrelby.p
osition, poziceStrelby.rotation);
        Pecka pecka = PeckaGO.GetComponent<Pecka>();
        if(pecka != null)pecka.nastavCil(cil, damage);
    }
```

Samotný útok

```
   void OnDrawGizmosSelected ()
    {
```

```
        Gizmos.color = Color.blue;
        Gizmos.DrawWireSphere(transform.position, range);
    }
}
```

Pomáhá určení dosahu

Obchod

```
int koupenaTresen;
bool jeKoupenaTresen;
```

Označuje stavy nákupu

```
public GameObject tlacitko;
    public GameObject tresenBasic;
    public GameObject tresenSniper;
    public GameObject tresenRambo;
```

Jednotlivé třešně, které se dají koupit

```
    void Start()
    {
        tlacitko.SetActive(false);
    }
```

Zprovozňuje tlačítka

```
public void polozitTresen()
    {
        if(jeKoupenaTresen)
        {
            Instantiate(tresenBasic, transform.position, transform.rotation);
            tlacitko.SetActive(false);
            jeKoupenaTresen = false;
        }
    }
```

Umožňuje pokládání třešní

Veškeré následující funkce slouží ke kupování jednotlivých druhů třešní

Pecka

Proměnné

```
private Transform cil;
```

následování cíle

```
private int damage;
```

Určuje poškození

```
private float speed = 20f;
```

Nastavuje rychlost projektilu

Funkce

```
public void nastavCil(Transform _cil, int _damage)
    {
        cil = _cil;
```

```
        damage = _damage;
    }
```
Realizuje nastavování cíle

```
private void Update()
    {

        if(cil == null)
        {
            Destroy(gameObject);
            return;
        }
        Vector3 dir = cil.position - transform.position;
        float distanceThisFrame = speed * Time.deltaTime;
        if(dir.magnitude <= distanceThisFrame)
        {
            HitTarget();
            return;
        }
        transform.Translate (dir.normalized * distanceThisFrame, Space.World);
    }
```
Provádí následování cíle a trefování cíle

```
    {
        Destroy(gameObject);
        Character.getDamage(damage);
    }
```
Smaže projektil po trefení cíle

Peníze

```
Text penize;
    private void Start()
    {
        penize = GetComponent<Text>();
    }
    private void Update()
    {
        penize.text = Spawner.penize.ToString();
    }
```
Zobrazuje stav peněz hráče

Spawner

Proměnné

```
public static int zivoty = 100;
public static int penize = 1000;
```
Nastavuje životy a peníze

```
    public GameObject Dinosaurus_basic;
    public GameObject Dinosaurus_fast;
    public GameObject Dinosaurus_strong;
```

```
    public GameObject Dinosaurus_tank;
```
Definuje druhy dinosaurů

```
public Transform SpawnPoint;
```
Nastavuje bod, z kterého vychází nepřátelé

```
public GameObject Vitezstvi;
```
Vítězná obrazovka

```
public Transform spawn_Konec;
```
Přestane tvořit nepřátele

```
 private int wave = 1;
```
Nastavuje aktuální vlnu na 1

```
private int spawn;
```
Zapříčiňuje spuštění vln

```
 public void Spawn()
```
Definuje jednotlivé vlny

Následující funkce pomáhají lepšímu průběhu vln

Waypoints

Nastavuje body pro orientaci nepřátel

Zivoty

Zobrazuje počet životů a stará se o odečítání při průniku nepřátel

Uživatelské rozhraní

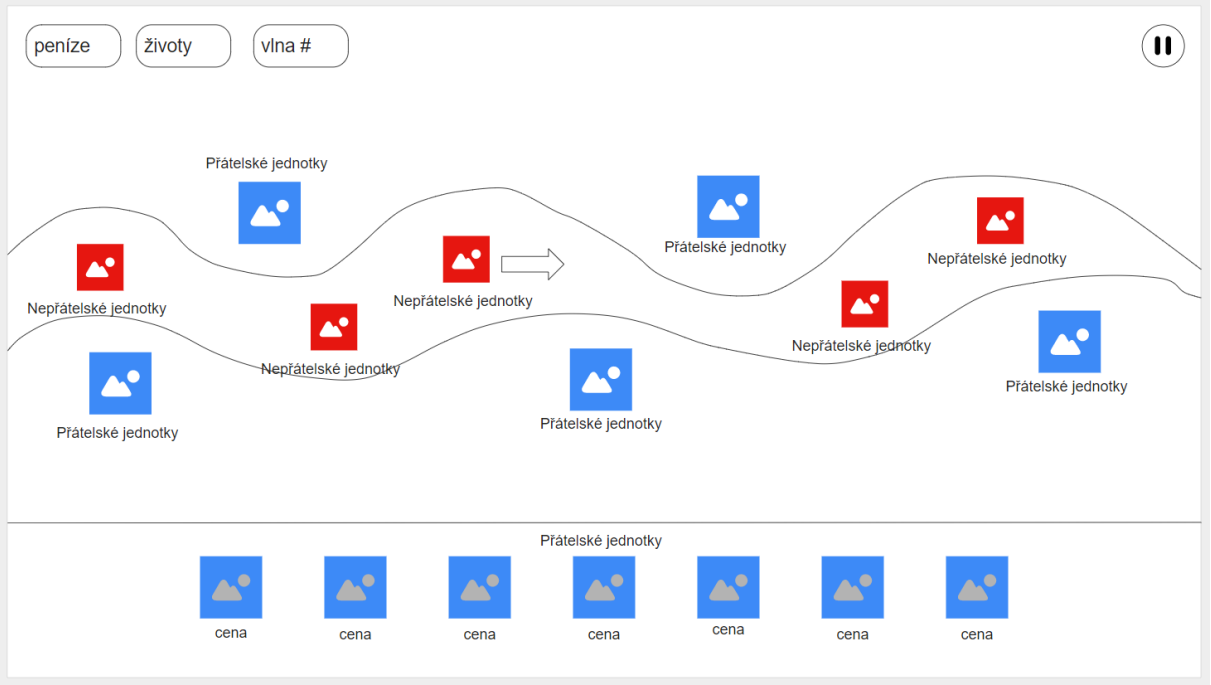
Wireframe úvodní obrazovky

Tento wireframe zachycuje úvodní obrazovku, tlačítko "hrát" vás odešle k výběru mapy, tlačítko "Konec hry" hru ukončí.

Wireframe výběru levelu

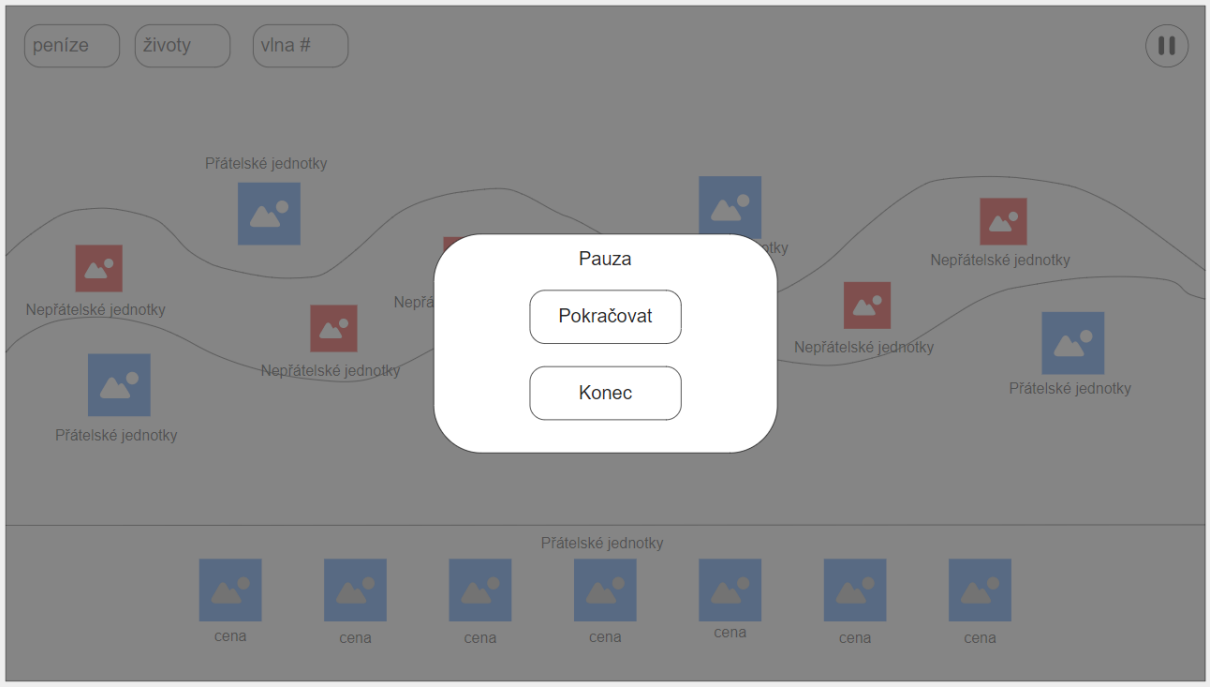
Na wireframu můžete vidět mapy. Když na nějakou kliknete, tak vás to odkáže do hry. Zeleně jsou označeny levely, které jste již dohráli, bíle jsou označeny levely, které máte odemčeny a nedohráli jste je a šedě levely, které jsou stále uzamčené.

Wireframe hry



Zde můžete vidět, jak hra bude vypadat za pochodu. Na spodní straně obrazovky můžete vidět menu s nakupováním jednotek. V levém horním rohu můžete vidět, kolik máte peněz, životů a v jaké jste aktuálně vlně nepřátel. Tlačítkem v pravém horním rohu si hru pozastavíte a dostanete se na wireframePauza.

Wireframe Pauzy



Takhle bude vypadat hra, když si ji pozastavíte. Tlačítkem "pokračovat" se vrátíte do hry, tlačítkem "konec" se vrátíte na úvodní obrazovku.

Testování produktu

Testování produktu jsme prováděli jak na sobě, tak na rodině a pár kamarádech. Nenarazili jsme na žádný zádrhel a testery hra zabavila. Nakonec jsme byli nuceni vyškrtat možnost vylepšování přátelských jednotek z důvodů časté chybovosti.

Zdroje

Při práci na hře jsme využívali tahle videa

Závěr

Při práci na projektu jsme zlepšili své dovednosti v teamové spolupráci, tvoření her, tvoření dokumentace, jazyku C# a grafických programů. Taky jsme si osvojili plánování a následné rozdělování práce.