Goblin Killer

Ročníková práce

Programování – 4.E

Autor: Robert Ambrož

Škola: Gymnázium, Praha 6, Arabská 14

Kraj: Praha

Konzultant: Mgr. Jan Lána

Třídní učitel: Mgr. Barbora Novosadová

Prohlášení

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 4. 4. 2021 ……..

Anotace

Hra z první osoby. Hráč bude vhozen na mapu vytvořenou z místností. Na začátku si bude moct vybrat ze tří základními předměty. Jakmile vstoupíte do další místnosti, dveře se zavřou a jediná možnost jak se dostat ven je, zabít všechny nepřátelé. Mimo místností s nepřáteli bude možnost narazit na obchod, místnost s překvapením nebo s nečekanou událostí. Jedna z místností bude ukrývat těžšího nepřítele, který bude nebezpečnější. Po úmrtí nepřítele se dostanete do dalšího patra, mimo jiné dostanete lepší předmět, který padá pouze z těchto nepřátel. Dohromady budou tři patra a v posledním patře bude čekat nejnebezpečnější nepřítel. Po zabití vyhrajete. Ve hře půjde o optimalizovaní svých předmětů, aby jste mohli hru dokončit.

First person shooter game. Player will appear in map made out of rooms. He can choose out of 3 basic items. After you step into the room, doors will shut and you have to kill everybody in that room. Apart of normal rooms you will have trading room, room with treasure or unexpected event. At the end of the level harder enemy would appear, it will drop special item after death. There are three levels and in the last one there will be hardest enemy. After killing him, you will win the game. You have to manage your items in order to finish the game.

Obsah

1. Úvod 1
   a) Očekávání 1
   b) Děj hry 1
   c) Problematika při tvorbě 1
2. Programování hry 2
   a) Tvoření postavy a pohybu 2
   b) Tvorba zbraně 3
   c) Životy pro hráče a nepřítele 4
   d) Ukazatel životů a nábojů 4
   e) Tvorba nepřátel 5
   f) Tvorba bomby 6
   g) Prostředí pro hru 7
3. Závěr 8
a) Dokončení hry 8

1. Úvod

a) Očekávání

Nejprve jsem si myslel, že dělání her v prostředí Unity bude jednoduché, ale přesvědčil jsem se o opaku. Některé věci se nedali z hlediska časové náročnosti stihnout. Skripty se programují v jazyce C# a většina jiných věcí se následně odehrává v Unity. Stavba místností a rozmístění nepřátel byl v Unity jednoduchý úkol, udělat však skripty jak se bude nepřátel chovat a co bude na hráče házet už byl těžší úkol. Velmi mě inspirovala hra The Binding of Issac, kde podobně jako zde musíte projít místnosti a zabíjet nepřátele.

b) Děj hry

Hra je zasazena do temného sklepení, kam jste byl poslán jakožto jediný, kdo dokáže zastavit hordu příšer. Zachránit svět tedy můžete jedině dosáhnutí poslední místnosti a následné zabití všech přítomných příšer. Než jste vstoupili do podzemí král vám dal vybrat ze tří zbraní, kuše, luk a následně meč. Otec vás kdysi učíval s kuší a v dětství jste běhal po lesích a hrál jste si s ní, proto jste nakonec vybral kuš. Pokud porazíte celé sklepení dostanete od krále velké bohatství a vše co budete chtít, samozřejmě ještě ruku jeho dcery. Vydáváte se tedy na cestu. Šance na úspěch jsou malé, ale přijmul jste tuto nabídku kvůli vaší rodině, kdyby jste nešel vy, musel by jít váš mladší bratr.

c) Problematika při tvorbě

Jelikož hra je ve 3D prostředí, může často nastat problém. Je nutné hru dělat postupně a nejprve mít vše připraveno a následně pak udělat prostředí pro hru, kde už dáváme pouze hotové objekty do pole, ovšem i to může vyvolat mnoho problémů. Při tvorbě hry jsem postupoval následovným způsobem.

1. Vytvoření postavy, kamery a pohybu
2. Vytvoření zbraně
3. Nastavení životů pro hráče a jiné objekty
4. Ukazatel nábojů a životů
5. Tvorba nepřátel
6. Tvorba bomby pro nepřátele
7. Tvorba prostředí pro hru
8. Tvorba NavMesh prostředí pro nepřátele.
9. Dokončování hry

2. Programování hry

a) Tvoření postavy a pohybu

Hned na začátku jsem si vytvořil skript pro pohyb myši. Nastavení základních hodnot bylo vcelku jednoduché. Kurzor myši se musí ukotvit na střed, jelikož zde budou lítat naše projektily a bude zde náš bod pro usnadnění míření. Citlivost myši a počáteční rotace musí být na začátku nastavena. Pozice myši se aktualizuje s časem. Současně s kamerou se musí otáčet i postava. To lze docílit jednoduchým vektorem.

```
float mouseX = Input.GetAxis("Mouse X") * mouseSensitivity * Time.deltaTime;
float mouseY = Input.GetAxis("Mouse Y") * mouseSensitivity * Time.deltaTime;


xRotation -= mouseY;
xRotation = Mathf.Clamp(xRotation, -90f, 90f);

transform.localRotation = Quaternion.Euler(xRotation, 0f, 0f);
//Rotace postavy soucasne s postavou
playerBody.Rotate(Vector3.up * mouseX);
```

Obrázek 1 - Nastavení myši

Následně jsem si udělal jednoduchou postavu, která je tvořena z vysokého válce, na horní podstavu jsem položil kameru, která byla následně trochu zapuštěna do válce aby ho hráč neviděl. Tento válec následně slouží jako kolize s naším hráčem. Chůze postavy se skládá ze základních proměnných jako je třeba rychlost nebo gravitace. Proměnná groundCheck kontroluje zda je postava přímo na zemi. Pokud není stále na postavu působí gravitace, jakmile dopadne na zem gravitace se zruší. Potom zde máme nastavení pohybu a aktualizaci. Skok je udělán obdobně, akorát když jsme ve skoku nemůžeme skočit znovu. Pro groundCheck je v naší postavě neviditelný objekt u spodní podstavy který zaznamenává zda postava stojí na zemi. Toho lze docílit když zemi nastavíme vrstvy zem a náš objekt následně kontrolujeme zda se této vrstvy dotýkáme. (Unity, nedatováno)

```
isGrounded = Physics.CheckSphere(groundCheck.position, groundDistance, groundMask);

if(isGrounded && velocity.y <0){
    velocity.y = -2f;
}
float x = Input.GetAxis("Horizontal");
float z = Input.GetAxis("Vertical");

Vector3 move = transform.right * x + transform.forward * z;
controller.Move(move * rychlost * Time.deltaTime);

if(Input.GetButtonDown("Jump")&& isGrounded){
    velocity.y = Mathf.Sqrt(skok * -2f * gravity);
}
```

Obrázek 2 - Kontrola země

Aby nám nyní postava fungovala stačí jen přidat kameru a neviditelná objekt na naší postavu. Do našeho válce vložíme náš skript s názvem Movement a můžeme si libovolně vybrat jakou bude mít rychlost či gravitaci. Následně do kamery vložíme skript Mouselook a přidáme si citlivost myší a čím se bude při otáčení rotovat, zde tedy naší postavou. V sekci movement ještě vybereme náš neviditelný objekt aby kontroloval zda je na zemi. Teď můžeme projekt zapnout a s postavou se volně pohybovat. Gravitace je normálně nastavena na 9.81. Zde jsem musel nastavit 15 aby byl skok o něco pomalejší a plynulejší.

b) Tvorba zbraně

Vytvoření zbraně bylo docela náročné, mnoho proměnných a hodně tříd, které musí fungovat tak jak mají. Nejprve jsem si určil základní proměnný jako třeba poškození, čas nabíjení, čas za jak dlouho můžete vystřelit nebo dokonce zda je zbraň připravená znovu střílet, to se použije u zbraní které nejsou automatické například naše kuše. Následovné přidaní jiné zbraně by nemělo být složitě. Přidáme objekt a následně bychom mohli udělat například automatickou kuši, která by měla menší poškození, ale za to by rychleji střílela. Hned po nastartování hry musíme nastavit abychom měli plný zásobník a byli schopni střílet. Následně implementujeme ukazatel nábojů na plochu. Přiřadíme klávesy pro střílení, v našem případě je to levé tlačítko myši. Musíme přidat tlačítko pro nabíjení. Pro nabíjení bude tlačítko R, které když zmáčkneme po krátkém intervalu se zbraň nabije a máme opět plný zásobník. Nakonec nám stačí do kláves udělat podmínky kdy je možné začít střílet.

```
if(readyToShoot && shooting && !reloading && bulletsLeft > 0){
    Shoot();
}
```

Obrázek 3 - Podmínka pro střelbu

Třídy Reload a ReloadFinished se starají o správné chování nabíjení zbraně. Ihned po nabití zbraně se musí obnovit množství kulek v zásobníku. Nyní se dostáváme do třídy Shoot, která se stará o výstřely. Vytvořil jsme zde i jednoduchý rozptyl kulek v případě implementace zbraně která například vystřelí více šípů najednou a bude fungovat podobně jako brokovnice. Výstřel se neskládá ze žádného projektilu, ale z raycastu, což je neviditelná kulka. Pokud dopadne na nepřítele způsobí mu poškození a pokud dopadne na zeď, způsobí menší defekt na zemi nebo na zdi. Tento defekt jsem si stáhnul z Unity, kde lze získat nějaké věci zadarmo. Tento balíček se jmenuje Unity Particle Pack 5.x. Obsah tohoto balíčků je velmi rozmanitý od defektu na zemi až po krev nebo výbuch. Rozhodně usnadní práci když nechce animovat nějaké věci, které ani neumíte. Defekt musí po nějaké

době zmizet ze hry aby se hra následně nesekala kvůli velkému množství objektů.

```
//Vystrel
RaycastHit hit;
//Zaznamenani vystrelu
if(Physics.Raycast(fpsCam.transform.position, fpsCam.transform.forward, out hit, range))
{
    Target target = hit.transform.GetComponent<Target>();
    if(target != null){
        target.TakeDamage(damage);
    }
}
//Dopad kulky
GameObject impact = Instantiate(impactEffect, hit.point, Quaternion.LookRotation(hit.normal));
Destroy(impact, 1f);

readyToShoot = false;
bulletsLeft --;
Invoke("ResetShot", casMeziStrelou);
```

Obrázek 4 - Poškození, defekt na objektech

Třída GunSystem slouží pro vytvoření zbraně, ale může poskytnout mnoho dalších zbraní, kde stačí jenom upravit proměnné a ihned můžeme mít například brokovnici nebo něco ve stylu raketometu.

c) Životy pro hráče a nepřítele

Pro nastavení životů jsem si udělal jednoduchou třídu s názvem Target. Target udává životy daného objektu, pro mě je to postava a nepřátelé, dále by šli udělat další objekty které střelením odemkneme. Pro hráče je zde také nastavený indikátor života pomocí textu, který pak vidíme na obrazovce. Tuto možnost mají i nepřátele, ale u nich zůstala prázdná a není důvod abychom viděli jejich životy na obrazovce. Pokud nějaký objekt bude mít nula životů nebo méně, objekt automaticky zmizí ze hry a bude pokořen. To samý se může stát našemu hrdinovi, a tím hra končí a musíme jít od znovu. Počet životů lze měnit v Unity. Je jen na nás zda chceme lehkého protivníka nebo silnějšího.

```
//Poskozeni
public void TakeDamage(float amount){
        zivoty -= amount;
        if(zivoty <= 0f){
            Die();
        }
```

Obrázek 5 - Životy

d) Ukazatel životů a nábojů

Pro ukazatel životů a nábojů jsem použil volně dostupný balíček TextMesh Pro (Unity, 2020), který usnadňuje práci s textem a je možné ho dokonce i upravovat. Poté si jen nastavíme kde chceme na jakém místě se bude text vykreslovat a co bude vykreslovat. Text, který bude vypisovat získáme inicializací ve skriptu jako zde u životů.

```
// Ukazatel zivotu v UI
public void Update(){
      if(ukazatelZivotu != null){
            ukazatelZivotu.SetText(zivoty + "");
      }
```

Obrázek 6 - Ukazatel životů

V Unity nám stačí vytvořit novy TextMesh Pro (Unity, 2020) a vložit do našeho skriptu. Ten pak vložíme na obrazovku kde ho chceme a vidíme naše životy. Podobně to funguje i u nábojů tam, ale musíme pro přehlednost ještě za náboje napsat kolik je kapacita zásobníku abychom věděli zda se nám vyplatí nabít znovu, nebo nám postačí stávající náboje. Podmínka je ovšem stejná. Mezera v SetText je z toho důvodu, protože neumí konvertovat int. Mezera nám tuto chybu odstraní a int může zůstat původní bez jakékoliv změny. Teď se nám bez problému zobrazí náboje a životy v naší hře.

e) Tvorba nepřátel

Nyní přišlo na řadu vytvořit nepřátele, které budeme muset zabít, ze začátku jsem opět pracoval pouze s válcem, jenž představoval nepřítele. Nepřítel je nastaven tak, že když nevidí hráče chodí kolem, pokud zahlédne našeho hráče, ale není v dosahu útoku, začne ho pronásledovat. Pokud doběhne až k nám a má nás v dosahu pro útok, začne házet bomby, které následně poškozují našeho hráče. Nejprve jsem si musel udělat několik proměnných, které sloužily pro útok, a poté ještě nějaké pro pohyb. Vzdálenost od hráče a viditelnost hledá protivník v kouli okolo sebe, kterou neustále kontroluje. Volná chůze fungují na bázi zvolení náhodného bodu, který pak nepřítel bude pronásledovat dokud k bodu nepřijde. Jestliže stále hráče nevidí se tento proces stále opakuje. Stejně jako u hráče musíme stále kontrolovat, zda je nepřítel na zemi a zda vybíra náhodný bod, který není například pod ním nebo nad ním, vše musí vybírat na zemi. (Persoc, 2015)

```
private void SearchWalkPoint(){
    float randomZ = Random.Range(-walkPointRange, walkPointRange);
    float randomX = Random.Range(-walkPointRange, walkPointRange);

    walkPoint = new Vector3(transform.position.x + randomX, transform.position.y, transform.position.z + randomZ);

    if(Physics.Raycast(walkPoint, -transform.up, 2f, whatIsGround)){
        walkPointSet = true;
    }
```

Obrázek 7 - Bod pro přemístění

Pokud je hráč v dosahu zatímco jde k bodu, přestane následovat bod a začne pronásledovat nás. Toho lze docílit jednoduchou metodou, místo náhodného bodu protivník zvolí jeden bod, kterým jsme my a začne nás pronásledovat. Viditelnost je o něco větší než útok aby nás mohl chvíli pronásledovat a my měli chvilku čas utéct. Jakmile jsme v dosahu útoku začne házet malé bomby, které si musíme ještě vytvořit. Nepřítel se musí pohybovat po zemi, a proto zde mám naimportovaný NavMesh (Unity, 2017) balíček, který slouží k zmapování země a udělá mapu, kde následně protivník může chodit.

Nakonec musíme vytvořit třídu, která se postará o poškození naší postavy. Poté nám stačí odstranit bombu po dopadu a máme hotovo.

```
public void ResetAttack(){
    uzUtocil = false;
}

//Protivnik dostava poskozeni
public void TakeDamage(int damage){
    zivoty -= damage;

    if(zivoty <= 0){
        Invoke(nameof(DestroyEnemy), 0.5f);
    }
}
```

Obrázek 8 - Poškození

Protivník už fungoval, ale neměl žádný vzhled. Vzhled nepřítele jsem si zvolil příšeru s názvem „goblin“, její texturu jsem našel zdarma v obchodu s balíčky v Unity (DancingEyebrows, 2021). Následně stačilo této postavě přiřadit skript pro nepřítele a ještě zivoty.

f) Tvorba bomby

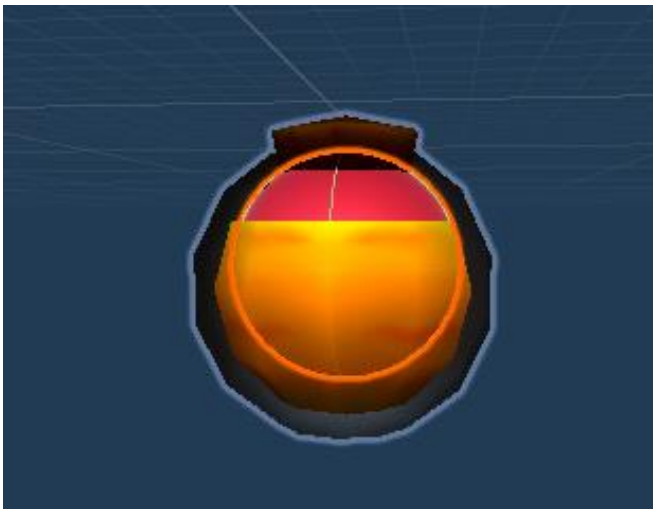
Nepřítel musí mít způsob jak na nás útočit, to lze udělat mnoha způsoby, ale já jsem si nakonec vybral, že po nás bude házet bomby, které po dopadu vybuchují. Skript jsem udělal lehko upravitelný. Snadno můžeme udělat mnohem rychlejší bombu, která se může odrážet od stěn a následně trefit hráče. Pro jednoduchost jsem zvolil obyčejnou bombu. Ze začátku jsem si udělal pouze kouli, která měla sloužit jako naše bomby a začal dělat na skriptu. Skript má zase několik proměnných jako například pro výbuch nebo kolik odrazů může udělat než vybouchne. Musíme neustále hlídat zda projektil nepřesáhl svůj maximální počet dopadů, jestli ano musí bouchnout. To samé platí pro čas, který může ve hře strávit. Bomba musí vědět co trefí, proto jsme si zvolili LayerMask, což se postará o to, že bomba ví kdy trefí hráče a udělí mu poškození.

```
private void OnCollisionEnter(Collision collision){
    collisions ++;
    if(collision.collider.CompareTag("Player") && explozeDotyk){
        Explode();
    }
}
```

Obrázek 9 - Exploze

Pokud trefí něco jiného, bomba pouze vybouchne. Jestliže bomba zasáhne hráče okamžitě zmizí a vyvolá se výbuch. Ten je opět importován z Unity Particle Pack 5.x. Exploze se přehraje vždy po dotyku bomby s jakýmkoliv povrchem. Abychom viděli sílu výbuchu necháme si vykreslit pomocí „Gizmos“ kruh, který bude hodnotu exploze ukazovat. Bombu si nyní musíme uložit jakožto asset, kdybychom tak neudělali nastala by chyba překročení času, který může bomba strávit na hracím poli. Ještě jsem přidal texturu z balíčku Low Poly Weapons (Games,

2016). Balíček je zdarma ke stažení v Unity. Naše hlavní zbraň je také z tohoto balíčku. Nyní můžeme bombu uložit jako asset. Bomba nyní vypadá takto a je připravená k použití.



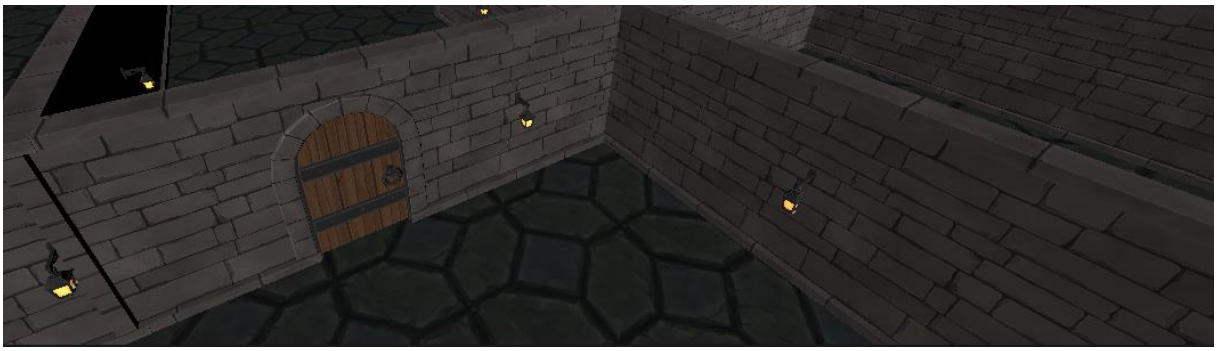
Obrázek 10 - Bomba

Nyní nám stačí přidat bombu k protivníkovi a může na nás vesele útočit. Ještě musíme určit jak velká bude exploze a jak velké poškození bude dávat našemu hráči. V nastavení protivník ještě můžeme upravit rychlost za jak dlouho bude protivník házet jednu bombu. V tomto případě je nastavena na 0.7s.

g) Prostředí pro hru

Vše potřebné už jsme si vytvořili. Nejdůležitější je ale prostředí ve kterém se budeme pohybovat. V Unity jsem si vybral balíček zdarma ke stažení, pro zem jsem použil Stone Floor Texture (3dfancy, 2014) a pro zdi, světlo a dveře balíček s názvem Stylized Hand Painted Dungeon (ARTS, 2020), který byl také zdarma ke stažení. V unity jsem si následně udělal pár místností, jež jsem si následně uložil jakožto předlohu pro ostatní. Vytvořil jsem si čtyři druhy místností: malou, velkou, do tvaru L a jednu obdélníkovou. Ty jsem následně kombinoval abych vytvořil mapu. Ta se skládá ze tří částí, kde první je nejlehčí a poslední nejtěžší. V nejtěžší části je více místností a více nepřátel. Nakonec jsem vložil dveře, které se při výstřelu otevřou a zmizí. Na mapě je také schovaná jedna místnost. Po dokončení prostředí jsem si udělal novou NavMesh (Unity, 2017) metodu, která zmapovala místnosti a umožnuje nepřátelům chodit. Tento balíček kontroluje i zdi okolo a prostředí zmapuje tak, aby nepřítel nechodil do zdí nebo někam jinam kam nemá. Tajná místnost funguje jenom pro hráče, vrstva „SECRET“ je pro

všechny objekty vnímán jako normální, pro hráče je sice zeď viditelná, ale bez problému může zeď projít a získat malý bonus. Lampy jsem rozmístil různě po zdech aby byla hra dobře prosvícená, ale zároveň nesmí být herní pole příliš světlé jelikož jde o podzemí plný „goblinů“. Nyní stačilo dodělat strop, který v Unity není vidět, protože jedna jeho strana je průhledná, ale druhá je plná. To mi umožnilo lepší manipulaci s prostředí jelikož jsem dovnitř stále viděl. Na konci části je nápis nové části nad vstupem. Ty jsou udělané jednoduše z kvádrů, na které je následně dána textura z balíčku Dungeon Stone Textures (3d.rina, 2016). Jinou texturu jsem použil aby byla na zdi vidět a nepřehlédli jsme ji.



Obrázek 11 - Ukázka prostředí

3. Závěr

a) Dokončení hry

V poslední řadě musíme hru optimalizovat. Rovnováha mezi naším životem a životem protivníkem je nejspíše nejdůležitější, následuje poškození, které uděláme nepřítelovi a také i jaké udělí on nám. Následně musíme upravit dosah naší zbraně, protože kdybychom mohli zabít nepřítele přes celou místnost nebylo by to férové, proto jsem nastavil dostřel stejně velký jako viditelnost nepřítele. Nakonec stačila upravit rychlost nepřítele, abychom mu hned neutekli mimo dosah a nezabili. Hra se nemusí na první pohled zdát vyvážená, ale záměrně je těžší, aby hráč měl nějakou výzvu.

4. Seznam obrázků

Obrázek 1 - Nastavení myši 2
Obrázek 2 - Kontrola země 2
Obrázek 3 - Podmínka pro střelbu 3
Obrázek 4 - Poškození, defekt na objektech 4
Obrázek 5 - Životy 4
Obrázek 6 - Ukazatel životů 5
Obrázek 7 - Bod pro přemístění 5
Obrázek 8 - Poškození 6
Obrázek 9 - Exploze 6
Obrázek 10 - Bomba 7
Obrázek 11 - Ukázka prostředí 8

5. Bibliografie

Bibliografie

3d.rina. (12. Červen 2016). *assetstore.unity.com*. Načteno z Dungeon Stone Textures: https://assetstore.unity.com/packages/2d/textures-materials/stone/dungeon-stone-textures-66487#releases

3dfancy. (11. Červen 2014). *assetstore.unity.com*. Načteno z Stonfe Floor Texture Tile: https://assetstore.unity.com/packages/2d/textures-materials/roads/stone-floor-texture-tile-18683#releases

ARTS, L. (9. Červen 2020). *assetstore.unity.com*. Načteno z Stylized Hand Painted Dungeon: https://assetstore.unity.com/packages/3d/environments/stylized-hand-painted-dungeon-free-173934#releases

DancingEyebrows. (3. Únor 2021). *assetstore.unity.com/*. Načteno z Goblin01: https://assetstore.unity.com/packages/3d/characters/goblin01-188119#releases

Games, S. (3. Říjen 2016). *assetstore.unity.com*. Načteno z Low Poly Weapons: https://assetstore.unity.com/packages/3d/props/weapons/low-poly-weapons-71680#releases

Persoc. (2. Duben 2015). *answers.unity.com*. Načteno z Basic Enemy AI in C#: https://answers.unity.com/questions/938221/basic-enemy-ai-in-c.html

Unity. (26. Květen 2017). *docs.unity3d.com*. Načteno z NavMesh building components: https://docs.unity3d.com/Manual/NavMesh-BuildingComponents.html

Unity. (14. Říjen 2020). *learn.unity.com*. Načteno z QuickStart to TextMesh Pro: https://learn.unity.com/tutorial/working-with-textmesh-pro#

Unity. (nedatováno). *docs.unity3d.com*. Načteno z CharacterController.Move: https://docs.unity3d.com/ScriptReference/CharacterController.Move.html