Informatika

Seminarska naloga

Igra “Cryptic Dungeons”

Program: Informatika

Predmet: Programiranje 2

Šolsko leto: 2020/21

Predavatelj: Aleksandar Lazarević

Lokacija: Ljubljana

Maks Jurkas

Vpisna številka: 1S00190598

Ljubljana, 28.4.2021

**Kazalo**

[**1 POVZETEK** 7](#_Toc70870043)

[**2 UVOD** 8](#_Toc70870044)

[**2.1 Opredelitev teme in razlogi za izbiro** 8](#_Toc70870045)

[**2.2 Predstavitev problemov** 9](#_Toc70870046)

[**2.3 Cilji projekta** 10](#_Toc70870047)

[**2.4 Uporabljena sredstva** 10](#_Toc70870048)

[**2.5 Reševanje glavnih problemov** 11](#_Toc70870049)

[**2.5.1 Prenašanje podatkov** 11](#_Toc70870050)

[**2.5.2 Optimizacija** 11](#_Toc70870051)

[**2.5.3 Shranjevanje in nalaganje igre** 12](#_Toc70870052)

[**3 VSEBINA PROGRAMA** 13](#_Toc70870053)

[**3.1 Načrt** 13](#_Toc70870054)

[**3.1.1 Oprema** 13](#_Toc70870055)

[**3.1.2 Zemljevid** 13](#_Toc70870056)

[**3.1.3 Trgovina** 14](#_Toc70870057)

[**3.1.4 Kovač** 14](#_Toc70870058)

[**3.1.5 Misija/Glavni spopad (pridobivanje materialov)** 14](#_Toc70870059)

[**3.1.6 Neskončni spopad (pridobivanje točk izkušenj)** 15](#_Toc70870060)

[**3.1.7 Lov za hrano** 15](#_Toc70870061)

[**3.1.8 Rudarjenje** 15](#_Toc70870062)

[**3.2 Atributi v igri** 16](#_Toc70870063)

[**3.2.1 Igralec in oprema** 16](#_Toc70870064)

[**3.2.2 Nasprotniki** 16](#_Toc70870065)

[**3.3 Opis izdelave aplikacije** 17](#_Toc70870066)

[**3.3.1 Zagon aplikacije** 17](#_Toc70870067)

[**3.3.2 Začetni ekran** 17](#_Toc70870068)

[**3.3.3 Domači zaslon** 19](#_Toc70870069)

[**3.3.3.1 Izgled domačega zaslona** 19](#_Toc70870070)

[**3.3.3.2 Koda domačega zaslona** 21](#_Toc70870071)

[**3.3.3.2.1 Funkcije** 21](#_Toc70870072)

[**3.3.3.2.2 Inicializacija forme** 23](#_Toc70870073)

[**3.3.3.2.3 Gumbi** 23](#_Toc70870074)

[**3.3.4 Trgovina** 26](#_Toc70870075)

[**3.3.4.1 Izgled trgovine** 26](#_Toc70870076)

[**3.3.4.2 Koda za trgovino** 28](#_Toc70870077)

[**3.3.4.2.1 Funkcije** 28](#_Toc70870078)

[**3.3.4.2.2 Inicializacija forme** 29](#_Toc70870079)

[**3.3.4.2.3 Gumbi** 29](#_Toc70870080)

[**3.3.5 Kovač** 31](#_Toc70870081)

[**3.3.5.1 Izgled kovača** 31](#_Toc70870082)

[**3.3.5.2 Koda kovača** 32](#_Toc70870083)

[**3.3.5.2.1 Funkcije** 32](#_Toc70870084)

[**3.3.5.2.2 Inicializacija forme kovač** 33](#_Toc70870085)

[**3.3.5.2.3 Gumbi** 34](#_Toc70870086)

[**3.3.6 Zemljevid** 37](#_Toc70870087)

[**3.3.6.1 Izgled zemljevida** 37](#_Toc70870088)

[**3.3.6.2 Koda zemljevida** 38](#_Toc70870089)

[**3.3.6.2.1 Funkcije** 38](#_Toc70870090)

[**3.3.6.2.2 Inicializacija forme zemljevid** 40](#_Toc70870091)

[**3.3.6.2.3 Gumbi** 40](#_Toc70870092)

[**3.3.7 Glavni spopad oz. misija** 43](#_Toc70870093)

[**3.3.7.1 Izgled glavnega spopada** 43](#_Toc70870094)

[**3.3.7.2 Koda glavnega spopada** 45](#_Toc70870095)

[**3.3.7.2.1 Funkcije** 45](#_Toc70870096)

[**3.3.7.2.2 Inicializacija glavnega spopada** 54](#_Toc70870097)

[**3.3.7.2.3 Gumbi** 54](#_Toc70870098)

[**4 ZAKLJUČEK** 60](#_Toc70870099)

**Kazalo slik**

[Slika 1: Koda za zagon začetnega zaslona 17](#_Toc70870378)

[Slika 2: Začetni zaslon 17](#_Toc70870379)

[Slika 3: Funkcija za gumb "Start" 18](#_Toc70870380)

[Slika 4: Funkcija za gumb "Load Game" 18](#_Toc70870381)

[Slika 5: Funkcija za gumb "Exit" 18](#_Toc70870382)

[Slika 6: Domači zaslon 19](#_Toc70870383)

[Slika 7: Oprema 20](#_Toc70870384)

[Slika 8: Using stavki za domači zaslon 21](#_Toc70870385)

[Slika 9: Funkcija za prikaz izbrane opreme 21](#_Toc70870386)

[Slika 10: Funkcija za izbor opreme 21](#_Toc70870387)

[Slika 11: Funkcija za odklep opreme 21](#_Toc70870388)

[Slika 12: Funkcija za prikaz opreme 22](#_Toc70870389)

[Slika 13: Funkcija za skrivanje opreme 22](#_Toc70870390)

[Slika 14: Funkcija za prikaz igralčevih atributov 22](#_Toc70870391)

[Slika 15: Konstruktor za domači zaslon/formo 23](#_Toc70870392)

[Slika 16: Gumbi za izbor opreme 23](#_Toc70870393)

[Slika 17: Gumb za prikaz/skrivanje opreme 24](#_Toc70870394)

[Slika 18: Gumb za prikaz naslednje strani opreme 24](#_Toc70870395)

[Slika 19: Gumb za nalaganje igre 25](#_Toc70870396)

[Slika 20: Gumb za shranjevanje igre 25](#_Toc70870397)

[Slika 21: Izgled trgovine 26](#_Toc70870398)

[Slika 22: Funkcija za zaklep gumbov za nakup 28](#_Toc70870399)

[Slika 23: Funkcija za pregledovanje razpoložljivosti opreme 28](#_Toc70870400)

[Slika 24: Konstruktor za trgovino 29](#_Toc70870401)

[Slika 25: Gumb za nakup orožja 29](#_Toc70870402)

[Slika 26: Gumb za vrnitev na domači zaslon 30](#_Toc70870403)

[Slika 27: Izgled kovača 31](#_Toc70870404)

[Slika 28: Funkcija za prikaz opreme 32](#_Toc70870405)

[Slika 29: Funkcija za pregledovanje razpoložljivosti opreme za izdelavo pri kovaču 32](#_Toc70870406)

[Slika 30: Funkcija za pregled razpoložljivosti opreme za izdelavo v datoteki s podatki 32](#_Toc70870407)

[Slika 31: Funkcija za posodabljanje igralčevih materialov 33](#_Toc70870408)

[Slika 32: Funkcija za izpis potrebnih materialov v podatkih 33](#_Toc70870409)

[Slika 33: Konstruktor za kovača 33](#_Toc70870410)

[Slika 34: Gumb za izdelavo opreme 34](#_Toc70870411)

[Slika 35: Funkcija za izdelavo opreme 34](#_Toc70870412)

[Slika 36: Funkcija za prikaz igralčeve torbe 34](#_Toc70870413)

[Slika 37: Funkcija za porabo materialov 36](#_Toc70870414)

[Slika 38: Polje z materiali 36](#_Toc70870415)

[Slika 39: Gumb za vrnitev 36](#_Toc70870416)

[Slika 40: Prikaz zemljevida 37](#_Toc70870417)

[Slika 41: Funkcija za odklep nivojev 38](#_Toc70870418)

[Slika 42: Funkciji za zagon nivojev 39](#_Toc70870419)

[Slika 43: Konstruktor za zemljevid 40](#_Toc70870420)

[Slika 44: Gumb za neskončni spopad z vsemi nasprotniki 40](#_Toc70870421)

[Slika 45: Gumb za končnega nasprotnika igre 40](#_Toc70870422)

[Slika 46: Gumbi za neskončni nivojski spopad 42](#_Toc70870423)

[Slika 47: Gumb za izbiro misije oz. glavnega spopada 42](#_Toc70870424)

[Slika 48: Prikaz glavnega spopada - izbira stopnje 43](#_Toc70870425)

[Slika 49: Prikaz glavnega spopada – bitka 44](#_Toc70870426)

[Slika 50: Funkcija za prikaz zaslona za bitko 45](#_Toc70870427)

[Slika 51: Funkcija za preskok stopnje 46](#_Toc70870428)

[Slika 52: Funkcija za skrivanje okna za bojevanje 46](#_Toc70870429)

[Slika 53: Funkcija za prikaz tapravih oken po zmagi – dodeljevanje točk in kovancev 48](#_Toc70870430)

[Slika 54: Funkcija za prikaz tapravih oken po zmagi - odklep naslednje stopnje in pregled zvišanja nivoja igralca 49](#_Toc70870431)

[Slika 55: Okno za nov nivo 49](#_Toc70870432)

[Slika 56: Funkcija za prikaz tapravih oken po zmagi - zmaga nad končnim nasprotnikom 50](#_Toc70870433)

[Slika 57: Funkcija za prikaz tapravih oken pri izgubi 51](#_Toc70870434)

[Slika 58: Funkcija za igralčev napad 51](#_Toc70870435)

[Slika 59: Funkcija za nasprotnikov napad 52](#_Toc70870436)

[Slika 60: Funkcija za prikaz okna za višji igralčev nivo 53](#_Toc70870437)

[Slika 61: Funkciji za izbiro nasprotnika 53](#_Toc70870438)

[Slika 62: Konstruktor za glavni spopad (misijo) 54](#_Toc70870439)

[Slika 63: Gumb za spopad 54](#_Toc70870440)

[Slika 64: Gumb za naključno izbrano stopnjo 55](#_Toc70870441)

[Slika 65: Gumb za preskok stopnje 55](#_Toc70870442)

[Slika 66: Gumb za izbor posebnega nasprotnika 56](#_Toc70870443)

[Slika 67: Gumb za napad 56](#_Toc70870444)

[Slika 68: Gumb za pobeg iz boja 57](#_Toc70870445)

[Slika 69: Gumb za končnega nasprotnika 58](#_Toc70870446)

[Slika 70: Gumb za dodeljevanje atributov 58](#_Toc70870447)

[Slika 71: Gumb za obrambo 59](#_Toc70870448)

# **1 POVZETEK**

Izdelava RPG (Role-Playing Game) igre z uporabo programskega jezika C# in Microsoft-ovega framework-a Windows Forms, ter Newtonsoft JSON framework-a za shranjevanje.

V šolskem letu 2020/21 smo pri predmetu Programiranje 2 delali s programskim jezikom C# in sicer vizualno programiranje preko Windows Forms-ov. Pri odločanju za seminarsko nalogo sem se odločil izdelati nekaj svojega, in sicer igro, kjer se spopadaš z raznimi nasprotniki, za to, da lahko ustvariš njihov oklep in orožje, ter postaneš močnejši. Igralec začne z začetno opremo, kjer se mora spopadati z močnejšimi nasprotniki, da pridobi kovance in točke izkušenj in si s tem izboljša možnosti zmage nad končnim nasprotnikom prvega nivoja. Po prvi zmagi nad njim, igralec pridobi material, katerega lahko po ponovnih spopadih s končnim nasprotnikom uporabi za izdelavo njegovega orožja in oklepa.

Igralec se nato počasi prebija skozi nivoje, nadgrajuje svojo opremo in zvišuje svojo moč in obrambo, dokler ne pride do zadnjega nivoja, kjer se lahko spopade z končnim nasprotnikom igre in odklene način igre, kjer se prikažejo vsi nasprotniki iz prejšnih nivojev. Ko izdela še zadnji meč in oklep, se igralcu odklene »Hard Mode« oz. težki način, kjer postanejo nasprotniki močnejši in igralcu dajo nove materiale, katere mora uporabiti za izdelavo končnega orožja in oklepa.

S tem projektom sem se odločil, da bom nadaljeval razvoj iger in poskušal narediti nekaj, kar bi pritegnilo pozornost publike in mi dalo veselje izdelave, ter nadaljnjih projektov.

Moram priznati da je bil projekt bolj uspešen v določenih področjih in rahlo nezadovoljiv v drugih, ampak bi ga z veseljem ponovil.

# **2 UVOD**

## **2.1 Opredelitev teme in razlogi za izbiro**

Odločil sem se za izdelavo “Role-Playing” igre, ki bo temeljila na igralčevi opremi in delovala na princip “boljša oprema = močnejši igralec”.

Igro sem izdelal s pomočjo programa Visual Basic, v C# programskem jeziku, s pomočjo Windows Form-ov. Visual Basic je eden izmed najbolj priljubljenih progamov za programiranje (vsaj za Microsoftove programske jezike. C# je relativno popularen programski jezik, katerega se uporablja za več popolnoma različnih stvari, z močno podporo razvijanju iger za Windows okolja, poleg tega se je pa v zadnjih letih nabrala podpora za razvijanje iger za pametne telefone. Windows Form-i so v trenutnem stanju že precej stari in se jih uporablja vse manj, ampak so kljub temu še zmeraj precej uporabni in priljubljeni, saj so dokaj preprosti za naučiti.

Za izdelavo igre sem se odločil, ker sem se že mnogo let želel spraviti k izdelavi in razvoju iger, vendar nisem vedel kje začeti. Nato smo pri predmetu Programiranje 2 delali z Windows Form-i in sem si zamislil neko igro, ki bi jo lahko naredil za seminarsko nalogo, po pogovoru s profesorjem sem se pa odločil to igro razširiti in iz nje izdelati diplomsko nalogo.

Že dolga leta sem oboževalec RPG (Role-Playing Game) žanre, kot so na primer: World of Warcraft, Monster Hunter, Slay The Spire, in podobne, ampak glede na to, da nimam nobenih izkušenj iz razivanja iger, sem se odločil za nekaj bolj preprostega, kjer bi iz tistih iger črpal le osnovne lastnosti (izdelovanje opreme, z materiali, ki jih igralec dobi od nasprotnikov, nakupovanje opreme, nadgradnja opreme), brez 3D grafike in sveta, kjer bi se igralec premikal in spopadal z nasprotniki.

Zelo me navdušujejo igre, kjer se igralec premika po raznih različnih nivojih, z različnimi nasprotniki, vse za to, da bi si lahko nadgradil svojo opremo in šel proti težjim in težjim nasprotnikom.

## **2.2 Predstavitev problemov**

Problemi, na katere sem naletel pri pisanju programa, so bili rahlo različni od problemov, ki sem si jih zastavil pred začetkom programiranja.

Pričakoval sem, da bodo problemi pri uporabljanju različnih lastnosti opreme skozi različne nivoje igre in pri shranjevanju podatkov pridobljenih po spopadu z nasprotnikom.

Predvideval sem, da bom naletel na številne probleme glede optimiziranja napisane kode, saj je bilo moje prvotno razmišljanje o uporabi opreme popolnoma različno in neoptimizirano glede na to kar sem kasneje ugotovil in kako sem stvari na koncu zapisal.

Poleg ostalih problemov, sem bil zaskrbljen s tem, kako bom shranjeval podatke, katere bi se kasneje dalo klicati tudi iz drugega sistema. Rabil sem se odločiti med serializacijo podatkov in shranjevanjem podatkov v podatkovno bazo (tu bi se moral odločiti še za ustrezno bazo podatkov, npr. MySQL, SQL, SQLite, itd.).

Manjši problemi, ki sem jih pričakoval so bili uporaba posebnih efektov od določenih nasprotnikov in določene igralčeve opreme, prikaz likov (igralec, orožje, oklep, nasprotniki, ozadja), odklepanje in zaklepanje prisvojene opreme in izvajanje spopada med igralcem in nasprotnikom.

## **2.3 Cilji projekta**

Na začetku projekta sem si zastavil cilj, da ustvarim neko aplikacijo oz. igro, ki bi imela nekatere principe svetovno znanih RPG (Role-Playing Game) iger.

Želel sem uporabiti številne lastnosti iz raznih iger, ki sem jih igral in jih združiti v eno igro, brez 3D grafik, premikanja po svetu, animacij, itd.

Prvotno sem nameraval implementirati pridobivanje opreme na način igre World of Warcraft, kjer bi igralec pridobil opremo, ko bi premagal končnega nasprotnika nivoja, ampak sem se kasneje odločil za drug pristop k pridobivanju opreme, in sicer, na metodo igre Monster Hunter, kjer igralec po zmagi nad končnim nasprotnikom nivoja pridobi materiale za izdelavo orožja in oklepa od njega.

Premikanje skozi nivo, premagovanje nasprotnikov in premikanje do končnega nasprotnika sem si zamislil na način igre Slay the Spire, kjer ima igralec več možnosti pristopa k končnemu nasprotniku (bojevanje, preskakovanje boja…), s tem, da ima vsaka možnost svoje prednosti in slabosti.

Za like sem se odločil, da bodo 8-bitne sličice, katere bi sam izdelal.

## **2.4 Uporabljena sredstva**

Za izdelovanje likov sem uporabil program Adobe Illustrator, kjer sem s pomočjo mreže izdelal vse like, orožja, oklepe, gumbe in ozadja.

Za pisanje programa sem uporabil program Visual Basic in programski jezik C#.

V Visual Basic-u sem uporabil Windows Form-e za prikaz programa in Newtonsoft JSON framework za serializacijo oz. shranjevanje ter nalaganje igre.

## **2.5 Reševanje glavnih problemov**

### **2.5.1 Prenašanje podatkov**

Problemi pri tej nalogi so bili številni, največji problem je bil, ugotoviti kako prenašati podatke o igralcu in nasprotnikih iz ene Windows Form-e v drugo, kajti se nova forma odpre s klicem »New«, ki pa kreira novo instanco Windows Form-a, kar pomeni, da so vsi podatki ponastavljeni, ta problem sem odrešil s pomočjo kazalcev oz. referenc, s tem da sem vse igralčeve podatke shranil v posebej dokument, iz katerega sem jih nato klical, poleg tega sem pa iz ene forme v drugo pošiljal določene podatke (npr. izbran oklep in orožje) in sem tako odpravil problem shranjevanja oz.   
ne-shranjevanja podatkov, tako sem lahko po parih dneh ugotavljanja končno nadaljeval izdelavo svojega programa.

### **2.5.2 Optimizacija**

Ključni problem pri večini programov je optimizacija, z drugimi besedami, kako bi program zapisal s čim manjšim številom vrstic in ponavljanja, da bi delal hitreje in bolje kot prej.

Pri optimizaciji je bila uporaba kazalcev ključna, saj bi bil moj progam dolg več tisoč vrstic kode brez njih. Tako, pa je ponavljanje kode minimalno in uporabljeno le ko je absoluto potrebno.

Kazalci delujejo tako, da v eni datoteki oz. eni formi kličemo podatke iz druge datoteke. V mojem primeru so bili vsi podatki shranjeni v eni datoteki (Data.cs) in vse forme črpajo podatke iz te datoteke.

V objektu oz. class-u Podatki so shranjeni vsi podatki o igralcu, poleg tega pa so tu shranjeni tudi vsi podatki o orožjih, oklepih in zalogi materialov, ki jih je igralec pridobil pri igranju.

### **2.5.3 Shranjevanje in nalaganje igre**

Poleg problema prenašanja podatkov, medtem ko igralec igra, je bil eden izmed večjih problemov še shranjevanje igre, tako da bi lahko igralec nadaljeval svojo prejšno sejo igre in se mu ne bi vsi podatki in pridobljena oprema zbrisala ob zaustavitvi programa.

Tu sem se potreboval odločiti za način shranjevanja in nalaganja igre, imel sem izbiro med serializacijo (shranjevanje objektov v neko datoteko) in uporabo podatkovne baze (MySQL, SQL, SQLite).

Najprej sem se odločil za uporabo podatkovne baze in sem se odloči za MySQL, saj sem bil s to bazo že seznanjen. Po številnih urah ugotavljanja, kako bi svoj progam povezal z bazo podatkov in iz baze bral oz. spreminjal vrednosti, se mi je celoten program obrnil na glavo in sem porabil par ur, da sem vse nazaj zgradil v stanje pred izgubo podatkov. Naslednji dan sem ponovno poskusil vzpostaviti povezavo in uporabo baze, tokrat mi je uspelo, ampak se je pojavil nov problem; veliko kode in malo optimizacije, kar bi pomenilo, da bi bil program veliko bolj počasen, z dodajanjem elementov in objektov v igro.

Po tem, ko sem uspešno vzpostavil povezavo s podatkovno bazo, sem to idejo tudi zavrgel in se odločil za uporabo serializacije.

Serializacija je bila dokaj enostavna, saj je tehnologija pogosto uporabljena in dobro dokumentirana, kar je zelo poenostavilo problem shranjevanja.

Odločil sem se za Newtonsoft JSON serializacijo, kar shranjuje objekt Podatki v dokument z .json končnico.

Newtonsoft JSON je že precej razširjen in uporabljen »framework«, z leti razvijanja in optimiziranja in z dobro opisano dokumentacijo je njegova uporaba relativno enostavna.

# **3 VSEBINA PROGRAMA**

## **3.1 Načrt**

Osnovni načrt aplikacije oz. programa je bil, izdelati igro, ki bi imela določene elemente drugih iger, ampak bi bila narejena na način HTML5 iger (iger, ki so narejene za spletni brskalnik). Aplikacija bi imela neko glavno stran (domača stran), kjer bi lahko igralec izbiral orožje in oklep, imel vpogled v njegov inventar, imel povezavo do zemljevida, kjer bi izbiral kateri nivo bi šel igrati, povezavo do trgovine, kjer bi lahko nakupil opremo, povezavo do kovača, kjer bi si lahko izdelal opremo iz materialov, ki jih je pridobil na misiji in nadgrajevanje opreme ter opcijo hranjenja, kar bi mu zvišalo moč in/ali obrambo.

### **3.1.1 Oprema**

Prvotno sem nameraval narediti izbiro opreme v posebej oknu, ampak sem se na koncu odločil, da bi bilo bolje, če je vse to na domači strani in lahko igralec izbira opremo kadarkoli je na tej strani.

### **3.1.2 Zemljevid**

Sem bi igralec prišel iz domače strani in bi imel možnost izbora nivoja, katerega želi igrati. Novi nivoji se odklenejo, ko igralec izdela orožje in oklep iz prejšnjega področja.

Tu ima igralec izbiro med neskončnim spopadom z nasprotniki, kjer si pridobiva točke izkušenj in kovance, da ima več možnosti priti do končnega nasprotnika nivoja in si pridobi njegove materiale. Vsak nivo ima opcijo neskončnega spopadanja in misije za pridobivanje materialov.

Poleg nivojev, ima igralec tu možnost odhoda na lov za hrano, katero lahko kasneje speče in si zviša moč oz. obrambo, ter rudarjenja, kjer pridobi potrebne materiale za izdelovanje določene opreme.

Ko se igralec prebije skozi vse nivoje, odklene končnega nasprotnika ter neskončni spopad z vsemi nasprotniki v igri.

Ko igralec premaga končnega nasprotnika in izdela njegovo opremo, se mu odklene možnost »Hard Mode« oz. težji način, kjer vsi nasprotniki postanejo močnejši in pustijo nove materiale za sabo, katere na koncu združite za končno orožje in oklep.

### **3.1.3 Trgovina**

Igralec lahko tu s kovanci nakupi opremo, katere ne more izdelati z materiali, ta oprema je sicer šibkejša od opreme, katero igralec izdela, ampak je boljša od začetne opreme in igralcu na določene momente olajša igranje.

### **3.1.4 Kovač**

Narejeno na podoben princip kot trgovina, pri kovaču igralec pridobi novo opremo. Vsak končni nasprotnik nivoja po smrti za sabo pusti določene materiale, katere igralec nato uporabi pri kovaču za izdelavo nove opreme.

Poleg izdelave opreme lahko igralec pri kovaču tudi nadgradi svojo trenutno opremo in tako zviša moč orožja ter obrambo oklepa. Za nadgrajevanje potrebuje določene kamne, ki jih pridobi na lovu.

### **3.1.5 Misija/Glavni spopad (pridobivanje materialov)**

To je glavna funkcija igre, vsaka misija ima svoje nasprotnike in končnega nasprotnika, ki po smrti za sabo pusti svoje materiale, katere igralec potrebuje za izdelovanje opreme in napredovanje.

Želel sem narediti na princip igre »Slay the Spire«, kjer ima igralec več možnosti (spopad, beg pred spopadom, posebni nasprotniki, trgovina…).

Pri moji aplikaciji ima igralec možnost med spopadom, posebnim nasprotnikom (ki je močnejši od ostalih na misiji in za sabo spusti kamne za nadgradnjo opreme) ter opcijo naključnosti, katera lahko igralca vrže proti nasprotnika, posebnega nasprotnika, ali mu pa pusti preskočiti spopad.

Nasprotniki se izberejo naključno (trije nasprotniki na nivo + posebni nasprotnik in končni nasprotnik). Vsak nasprotnik ima svoje atribute – najnižji in najvišji možni napad, točke življenja in svoj izrisan lik.

Pri spopadu ima igralec 3 možnosti. Izbira med napadom, ki nasprotniku zbije določeno količino življenjskih točk, obrambo, ki ima možnost obrambe nasprotnikovega napada in odbitjem napada nazaj na nasprotnika, ter možnost pobega, kjer lahko pred nasprotnikom pobegne, ampak, če mu ne uspe, ga nasprotnik še zmeraj napade.

Če igralcu uspe priti do končnega nasprotnika in ga premaga, dobi njegove materiale, katere lahko uporabi za izdelavo boljšega orožja ali oklepa.

### **3.1.6 Neskončni spopad (pridobivanje točk izkušenj)**

Pri neskončnem spopadu ima igralec na voljo nove funkcije:

* Življenjski napitek, ki igralcu zviša življenjske točke
* Ščit, ki obrani nasprotnikove napade dokler se ne uniči
* Bomba, ki naredi močan napad
* Zvitek moči, ki poveča igralčev napad za 3 napade

Poleg novih funkcij, pa igralec izgubi možnost pobega in obrambe.

Tu se pred igralcem prikazujejo nasprotniki željenega nivoja, torej za 1. nivo bo igralec dobil nasprotnike iz 1. nivoja, da lahko hitreje zbira točke izkušenj in si tako pridobi večjo možnost zmage nad končnim nasprotnikom nivoja.3

Točke izkušenj in kovanci, ki jih igralec pridobi tu so manjše kot pri glavnem spopadu oz. misiji, da ne bi prehitro pridobil preveč točk in postal premočen za nivo.

### **3.1.7 Lov za hrano**

Tu bi igralec imel »mini-game«, torej način igre, ki bi bil drugačen od spopada in sicer, na ekranu se naključno prikaže gumb, katerega mora igralec pritisniti v določenem časovnem roku in tako ujame žival, katero lahko kasneje speče in si zviša atribute.

### **3.1.8 Rudarjenje**

Pri temu načinu igre igralec pridobiva materiale, ki jih ne more pridobiti pri spopadu z končnim nasprotnikom nivoja. Tu pridobi železo, katerega kasneje odnese v trgovino in s tem poleg kovancev kupi novo opremo.

## **3.2 Atributi v igri**

### **3.2.1 Igralec in oprema**

Igralec ima svoje atribute (moč, obramba in življenjske točke), katere si zvišuje s pridobivanjem točk izkušenj in višanjem svojega nivoja.

Ko igralec premaga nasprotnika, pridobi točke izkušenj in kovance, ko pa enkrat pridobi dovolj točk izkušenj, se mu zviša nivo in dobi 2 točki, katere lahko uporabi za zvišanje moči ali obrambe, poleg tega se mu pa tudi z vsakim višjim nivojem, zvišajo življenjske točke.

Igralčeva oprema ima svoje atribute, ki se nato seštevajo z igralčevimi atributi.

Orožje ima najnižji in najvišji možni napad, oklep pa svojo obrambo.

Izračun igralčevega napada vsebuje napad od orožja in moč igralca.

Izračun igralčeve obrambe vsebuje obrambo od oklepa in igralčevo obrambo.

### **3.2.2 Nasprotniki**

Vsak nasprotnik ima neko konstantno moč oz. najnižji in najvišji možni napad ter svoje življenjske točke. Nasprotnikovi atributi so skozi igro enaki, razen če igralec izbere »Hard Mode« oz. težji način, kjer se vsi atributi nasprotnikov zvišajo, da je igra bolj zanimiva.

## **3.3 Opis izdelave aplikacije**

### **3.3.1 Zagon aplikacije**

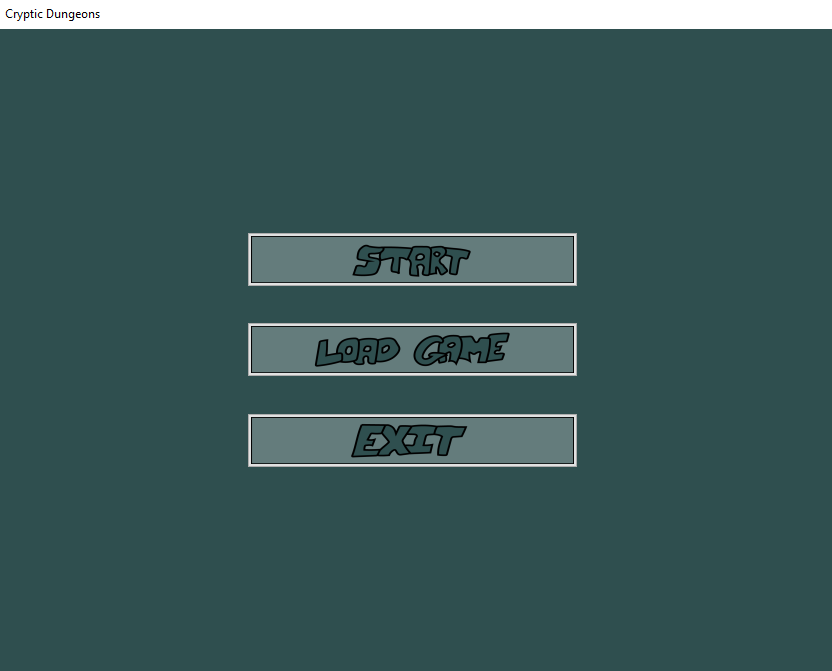


Slika : Koda za zagon začetnega zaslona

Pri zagonu aplikacije se kreirajo novi podatki, ki so shranjeni v dokumentu »Data.cs«, v temu dokumentu so pa te podatki shranjeni v objekt »Podatki«

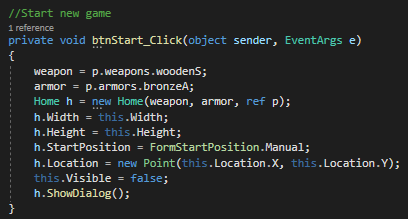
Kot sem že omenil, se na začetku programa kreirajo novi podatki, tu se jih mora referencirat, ko aplikacija požene začetni zaslon »Start.cs«, da program ve katere podatke mora uporabljati.

### **3.3.2 Začetni ekran**



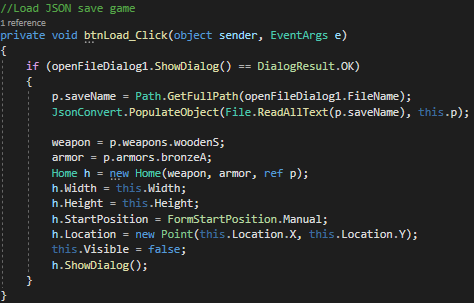
Slika : Začetni zaslon

Na začetnemu zaslonu ima igralec možnost med »Start«, ki zažene novo igro, »Load Game«, kjer lahko igralec izbere datoteko, ki jo je izdelal pri prejšnji igralni seji ali »Exit«, ki zapre aplikacijo.



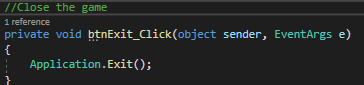
Slika : Funkcija za gumb "Start"

Pri pritisku na gumb »Start« se iz dokumenta s podatki nastavi izbrano opremo na začetno opremo in odpre domače okno z enako velikostjo okna, na isti poziciji, ter zapre trenutno okno.



Slika : Funkcija za gumb "Load Game"

Pri pritisku na gumb »Load Game« se odpre pogovorno okno, kjer lahko izberemo shranjeno igro s končnico .json, to naloži podatke shranjene v tej datoteki in igralca pošlje na domači zaslon z vsemi podatki iz datoteke.

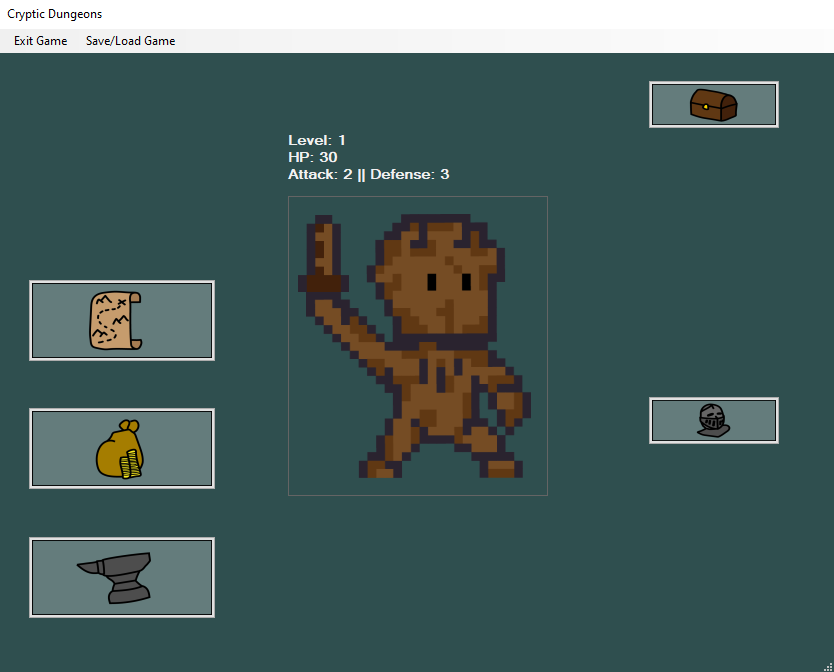


Slika : Funkcija za gumb "Exit"

Ob pritisku na gumb »Exit« se aplikacija zapre.

### **3.3.3 Domači zaslon**

#### **3.3.3.1 Izgled domačega zaslona**



Slika : Domači zaslon

Na domačem zaslonu na sredini vidimo igralca opremljenega s trenutno izbrano opremo, nad njim so podatki o igralčevih atributih (nivo igralca, življenjske točke, moč in obramba igralca).

Na levi strani vidimo gumbe za zemljevid, trgovino in kovača.

Na desni strani vidimo gumbe za inventar in opremo.

Na vrhu okna vidimo »Exit Game« in »Save/Load Game« kar se uporablja za izhod iz igre in shranjevanje oz. nalaganje prejšnje seje igre.



Slika : Oprema

Ko pritisnemo na gumb za opremo, se nam odpre vsa oprema, ki je trenutno v igri. Če je oprema zaklenjena, ima izbrana oprema sivo ozadje, ki se pa spremeni, ko opremo pridobimo.

Na dnu imamo še gumb za naslednjo stran opreme, ki prikaže še preostalo opremo.

#### **3.3.3.2 Koda domačega zaslona**

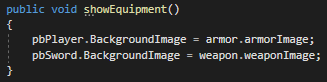
##### **3.3.3.2.1 Funkcije**



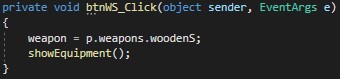
Slika : Using stavki za domači zaslon

»using Newtonsoft.Json« se uporabi za shranjevanje in nalaganje igre

»using System.IO« se uporablja za prikaz pogovornega okna za shranjevanje ali odpiranje



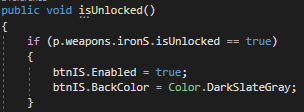
Slika : Funkcija za prikaz izbrane opreme



Slika : Funkcija za izbor opreme

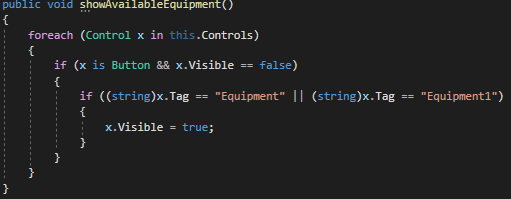
Ob pritisku na željen del opreme, se v spremenljivko »weapon« ali »armor« zapiše željen objekt (vsi objekti so shranjeni v datoteki Data.cs in jih zato kličemo z »p.[željen objekt].[atribut objekta])

Po zapisu objekta v spremenljivko se kliče funkcija »showEquipment«, ki spremeni sliko orožja ali oklepa na željeno orožje oz. oklep.



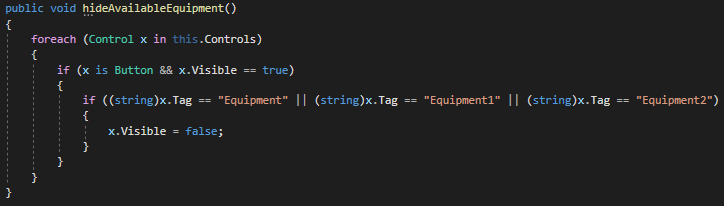
Slika : Funkcija za odklep opreme

Funkcija »isUnlocked« se kliče, ko se odpre domači zaslon in odklene določeno opremo, če je izpolnjen pogoj, da je oprema odklenjena. Poleg odklepa opreme, ta funkcija tudi spremeni ozadje opreme iz sive na barvo ozadja domačega zaslona.



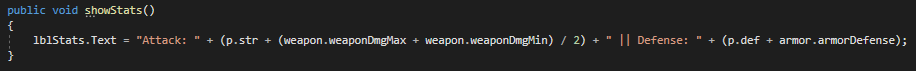
Slika : Funkcija za prikaz opreme

Funkcija, ki gre skozi »foreach« zanko, ki pregleduje če je izbran objekt gumb in če je gumb neviden, potem pregleda, če ima gumb oznako »Equipment« ali »Equipment1« in ga prikaže, če je pogoj izpolnjen.



Slika : Funkcija za skrivanje opreme

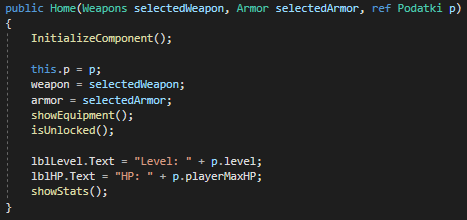
Tako kot funkcija za prikaz opreme gre ta funkcija skozi »foreach« zanko in skrije vse gumbe, ki imajo oznako »Equipment«, »Equipment1« ali »Equipment2«.



Slika : Funkcija za prikaz igralčevih atributov

Funkcija »showStats« prikaže igralčeve atribute nad njegovo sliko. Prikaže moč igralca (izračunana iz igralčeve moči + mediano od napada za izbrano orožje) in obrambo igralca (izračunana iz igralčeve obrambe + obrambo oklepa)

##### **3.3.3.2.2 Inicializacija forme**



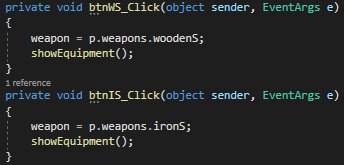
Slika : Konstruktor za domači zaslon/formo

»InitializeComponent()« inicializira formo (prikaže formo), temu sledi »this.p = p«, kar pove programu od kod mora črpati podatke

Po temu se nastavita spremenljivki »weapon« in »armor« na izbrano opremo iz prejšnje forme, ter kliče funkcijo za prikaz opreme in takoj za tem še funkcijo, ki odklene pridobljeno opremo.

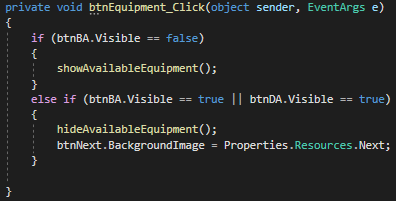
Na koncu še nastavi igralčev nivo in življenjske točke, ter igralčeve atribute.

##### **3.3.3.2.3 Gumbi**



Slika : Gumbi za izbor opreme

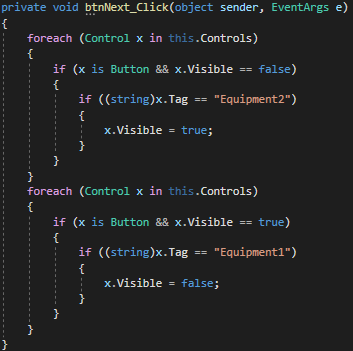
Tako kot sem že omenil prej, so tu funkcije za gumbe, ki nastavijo željeno opremo. Vsak gumb ima svoj klic, ker vsak gumb nastavlja drug del opreme.



Slika : Gumb za prikaz/skrivanje opreme

Ob pritisku na gumb program preveri če je oprema vidna in jo prikaže, če ni, s klicom na funkcijo za prikaz opreme.

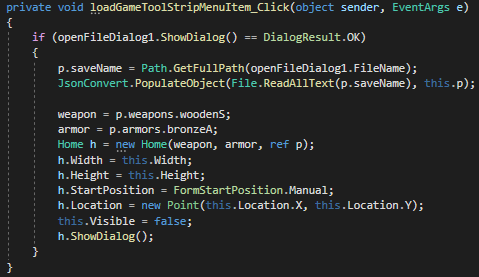
Če je oprema vidna, program kliče funkcijo za skrivanje opreme in skrije opremo.



Slika : Gumb za prikaz naslednje strani opreme

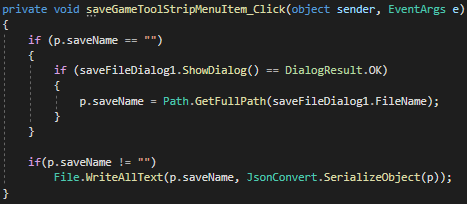
Ko igralec pritisne na gumb za naprej pri opremi se prikaže naslednji set opreme in skrije trenutni set opreme.

Funkcija za gumb za nazaj je le obrnjena funkcija za gumb naprej.



Slika : Gumb za nalaganje igre

Če želimo naložiti igro, izberemo gumb »Load Game«, ki odpre pogovorno okno, kjer izberemo željeno datoteko s končnico .json, nato pa na novo naloži domači zaslon, da se vsi podatki posodobijo.

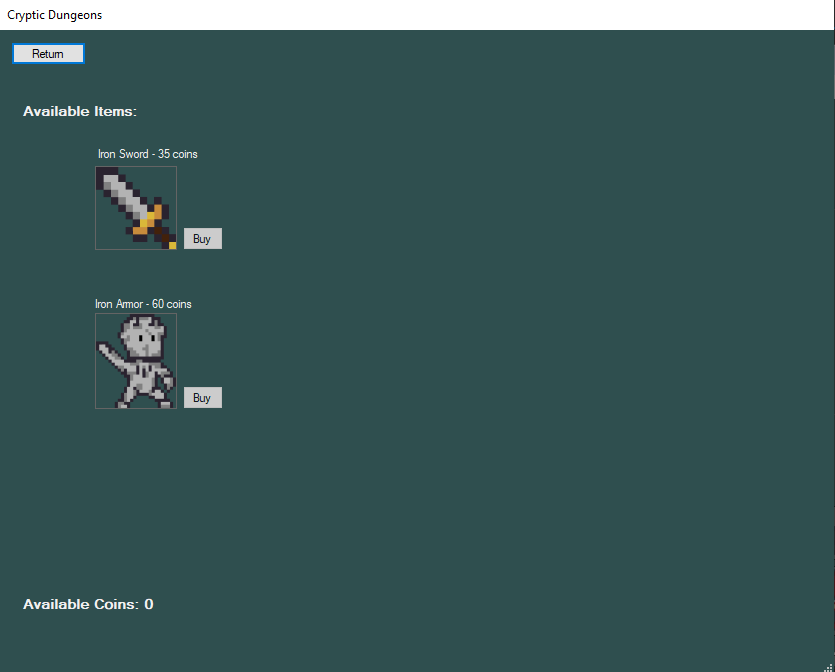


Slika : Gumb za shranjevanje igre

Podobno kot gumb za nalaganje deluje tudi gumb za shranjevanje, le da ta pregleda, če smo igro že naložili, v katerem primeru program avtomatsko shrani trenutne podatke v datoteko, če pa nismo naložili nobene datoteke, se pa odpre pogovorno okno, kjer izberemo ime in lokacijo datoteke.

### **3.3.4 Trgovina**

#### **3.3.4.1 Izgled trgovine**



Slika : Izgled trgovine

Ko pritisnemo na gumb za trgovino na našem domačem zaslonu se odpre nov Windows Form, ki prikazuje orožje in oklep, katerega lahko nakupimo s kovanci in železom.

Na zgornji sliki vidimo opremo, ki se odklene za nakup, ko so izpolnjeni vsi pogoji (pravilna količina kovancev in železa).

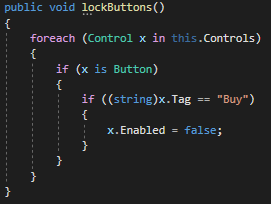
Na vrhu je gumb »Return«, ki nas vrne na domači ekran

Pod njim je oprema, ki jo lahko nakupimo (gumb »Buy« se odklene, ko ima igralec vse materiale)

Na dnu se nam prikaže naše število kovancev.

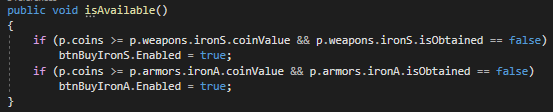
#### **3.3.4.2 Koda za trgovino**

##### **3.3.4.2.1 Funkcije**



Slika : Funkcija za zaklep gumbov za nakup

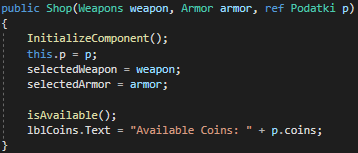
Ta funkcija gre skozi »foreach« zanko s Control parametri (za kontroliranje in nastavljanje delovanja programa), ki gre skozi vse vizualne komponente forme in pregleduje, če je komponenta gumb z oznako »Buy«, ter jo zaklene, če je.



Slika : Funkcija za pregledovanje razpoložljivosti opreme

Pri tej funkciji program pregleda če ima igralec pravilno količino kovancev in odklene gumb za nakup, če igralec dela opreme še nima.

##### **3.3.4.2.2 Inicializacija forme**

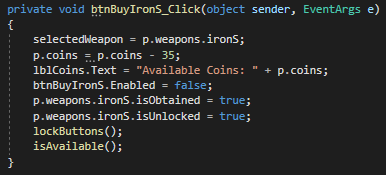


Slika : Konstruktor za trgovino

Tako kot pri prejšnji formi se forma začne s funkcijo »InitializeComponent«, ki je vgrajena v framework, nato se tako kot pri vsaki formi nastavi »this.p = p«, da program ve od koder mora črpati podatke.

Za tem se nastavi izbrano orožje in oklep, se izvede funkcija, ki pregleduje, če lahko igralec kupi del opreme ter posodobi razpoložljive kovance.

##### **3.3.4.2.3 Gumbi**

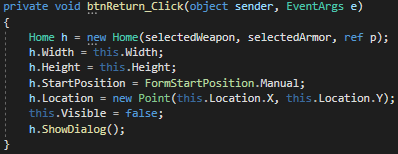


Slika : Gumb za nakup orožja

Ob pritisku na gumb »Buy« pri željenem orožju se nastavi izbrano orožje na nakupljenega

Igralcu se nato odštejejo kovanci, potrebni za nakup in se nastavijo vrednosti »isObtained« in »isUnlocked« v podatkih na »true«, kar pomeni, da bo orožje na voljo na domačem ekranu, poleg tega se pa gumb za nakup tega orožja zdaj ne bo več odklenil, tudi če bi imel igralec dovolj materialov za nakup.

Na koncu se izvedeta še funkciji za zaklep gumbov in pregledovanje razpoložljivosti opreme.

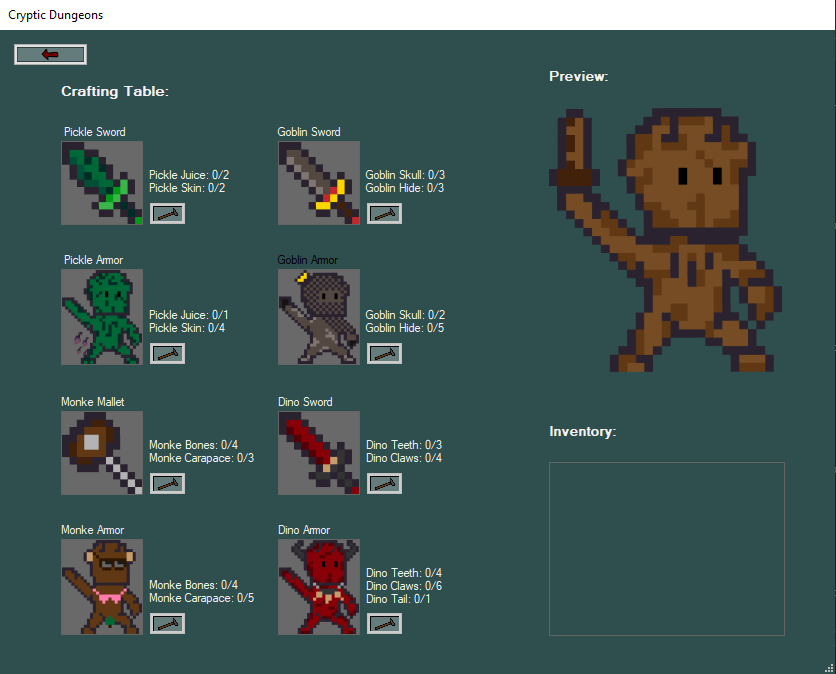


Slika : Gumb za vrnitev na domači zaslon

Ko se želimo vrniti iz trgovine, pritisnemo na gumb »Return«, ki zapre trenutno okno in odpre domači zaslon.

### **3.3.5 Kovač**

#### **3.3.5.1 Izgled kovača**



Slika : Izgled kovača

Ob pritisku na gumb z ikono nakovala na domačemu zaslonu se nam odpre okno, kjer lahko izdelujemo in nadgrajujemo opremo.

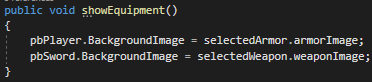
Na vrhu zopet vidimo gumb za povrnitev na prejšnje okno

Pod njim imamo orožja in oklepe, ki jih lahko izdelamo v igri. Oprema je zaklenjena dokler igralec ne pridobi vseh potrebnih materialov za izdelavo. Materiale igralec pridobiva na misijah, ti se shranijo v igralčevo torbo, prikažejo se pa tudi nad gumbom za izdelavo dela opreme, kar nam pove koliko materialov imamo in koliko jih še potrebujemo.

Na desni strani imamo izgled igralca in igralčevo torbo z vso vsebino.

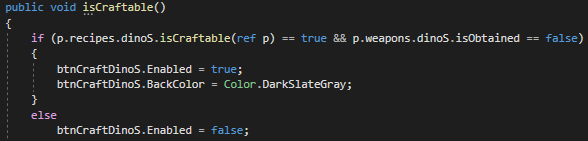
#### **3.3.5.2 Koda kovača**

##### **3.3.5.2.1 Funkcije**

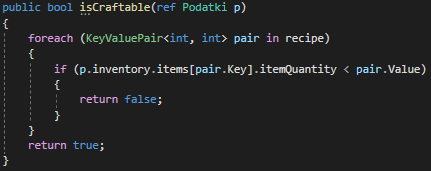


Slika : Funkcija za prikaz opreme

Tako kot na domačem zaslonu, je funkcija za prikaz opreme uporabljena tudi tu.



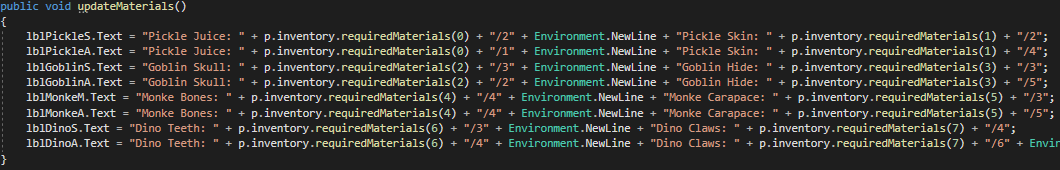
Slika : Funkcija za pregledovanje razpoložljivosti opreme za izdelavo pri kovaču



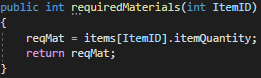
Slika : Funkcija za pregled razpoložljivosti opreme za izdelavo v datoteki s podatki

Podobno kot funkcija za pregledovanje razpoložljivosti opreme pri trgovini, imamo nekaj takega tudi tu. Deluje tako, da v datoteki »Data.cs« pregleda, če so zagotovljeni vsi pogoji (program pregleda v igralčevi torbi, če ima pravilno količino materialov in jo primerja s potrebno količino pri receptu za opremo), ter odklene gumb za izdelavo, če ima igralec dovolj materialov, poleg tega pa spremeni ozadje slike opreme na ozadje forme.

Če je igralec del opreme že izdelal, se gumb ne more več odkleniti.



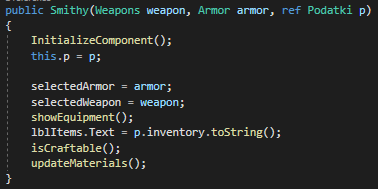
Slika : Funkcija za posodabljanje igralčevih materialov



Slika : Funkcija za izpis potrebnih materialov v podatkih

Ta funkcija deluje tako, da posodobi etiketo za prikaz igralčevih materialov + željene količine materialov za izdelavo s tem da kliče funkcijo »requiredMaterials«, ki se nahaja v datoteki s podatki, ta pa vrne potrebne materiale za določen del opreme, nato v novo vrsto ponovi pregled, še za druge materiale.

##### **3.3.5.2.2 Inicializacija forme kovač**



Slika : Konstruktor za kovača

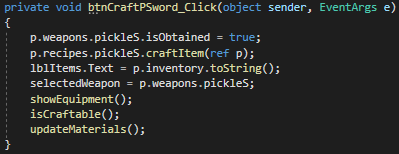
Kovač tako kot vse ostale forme prejme izbrano orožje in oklep poleg podatkov.

Najprej zažene prikaz forme in programu pove od kod naj črpa podatke, nato kot povsod drugod nastavi izbrano opremo.

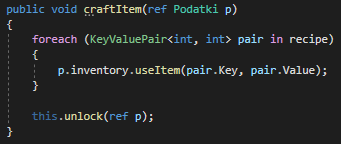
Po temu program kliče funkcijo za prikaz opreme in prikaže igralčevo torbo.

Na koncu se kličeta še funkciji za pregledovanje razpoložljivosti opreme za izdelavo, ter funkcija za posodabljanje igralčevih materialov.

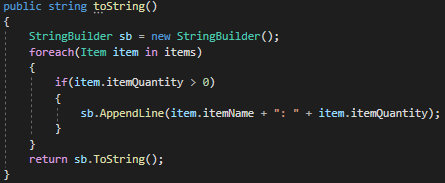
##### **3.3.5.2.3 Gumbi**



Slika : Gumb za izdelavo opreme



Slika : Funkcija za izdelavo opreme



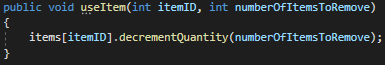
Slika : Funkcija za prikaz igralčeve torbe

Ko ima igralec dovolj materialov za izdelavo željenega dela opreme, pritisne na gumb z ikono kladiva, da ga ustvari.

Ob pritisku na gumb se spremeni vrednost »isObtained« izbranege opreme na »true«, kar pomeni, da je igralec ta del opreme pridobil in je ne more ponovno izdelati.

Za tem se izvede funkcija »craftItem()«, ki je zapisana v datoteki s podatki, ta funkcija porabi zahtevano količino materialov, ter jih odstrani iz igralčeve torbe. Temu sledi posodobitev igralčeve torbe, to stori funkcija »toString()«, ki je shranjena v datoteki s podatki.

Na koncu se še prikaže izdelan del opreme na igralcu, se zaklene gumb za izdelavo in posodobijo materiali, ki so zapisani pri vsakemu delu opreme.



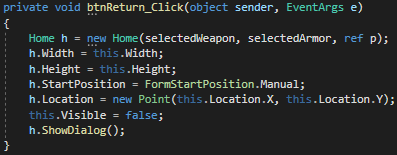
Slika : Funkcija za porabo materialov

Ko se kliče funkcija »craftItem()«, se znotraj te funkcije kliče še ena funkcija, »useItem()«, ki kot parameter prejme številko materiala in željeno količino materialov za odstranitev iz igralčeve torbe.



Slika : Polje z materiali

Vsak material ima svojo zaporedno številko in vsi materiali so zapisani v neko polje, iz katerega se jih nato tudi kliče.



Slika : Gumb za vrnitev

Na koncu imamo še gumb za vrnitev, ki ima enako funkcionalnost kot gumb za vrnitev pri trgovini.

### **3.3.6 Zemljevid**

#### **3.3.6.1 Izgled zemljevida**



Slika : Prikaz zemljevida

S pritiskom na gumb z ikono zemljevida na domačemu zaslonu nas program popelje do zemljevidnega okna, kjer lahko igralec izbira kam bo šel.

Ko začnemo novo igro, imamo odklenjen le prvi nivo »Seabed«, kjer se igralec spopada z morskimi nasprotniki, poleg tega ima pa na voljo še »LowRank Grind«, kjer hitreje pridobiva točke izkušenj in kovance, ter postane močnejši, da gre lahko probati premagati končnega nasprotnika prvega nivoja.

Poleg prvega nivoja ima igralec odklenjen tudi »Mining« oz. rudarjenje in »Hunting«, kjer gre igralec na lov za hrano.

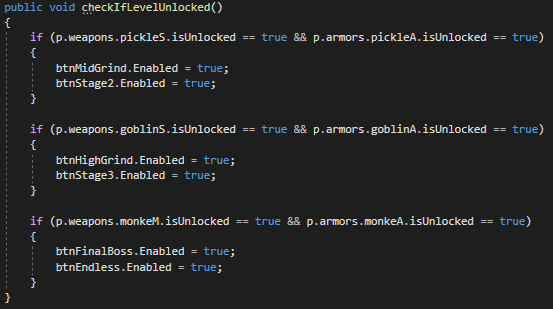
Na vrhu ponovno sedi gumb za povračilo

Pod njim so nivoji, od zgoraj navzdol so: misija (glavni spopad), neskončni spopad nivoja in neskončni spopad vseh nasprotnikov, na desni strani je končni nasprotnik igre.

Na dnu sta pa še gumba za lov in rudarjenje.

#### **3.3.6.2 Koda zemljevida**

##### **3.3.6.2.1 Funkcije**

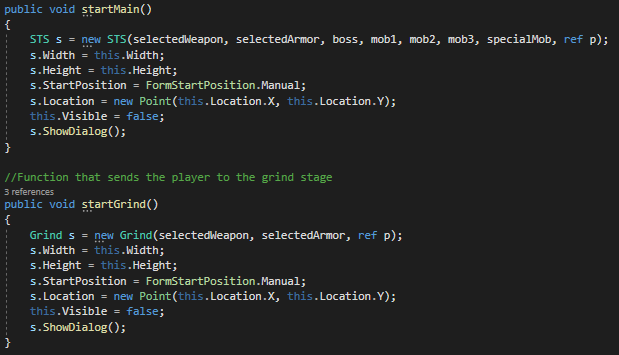


Slika : Funkcija za odklep nivojev

Ta funkcija pregleduje, če je igralec že odklenil oz. izdelal orožje in oklep iz prejšnjega nivoja.

Odklep nivojev deluje na principu igralčevega napredovanja in sicer, ko igralec izdela celotni set opreme od nivoja, se mu odklene naslednja misija in naslednji neskončni nivojski spopad.

Ko igralec premaga končnega nasprotnika džungle in izdela njegovo opremo se odklene končni nasprotnik igre ter neskončni spopad, kjer se prikažejo vsi nasprotniki iz igre.



Slika : Funkciji za zagon nivojev

Te dve funkciji sta namenjeni odprtju različnih form in sicer, glavni spopad (funkcija »startMain()«) in neskončni nivojski spopad (funkcija »startGrind()«).

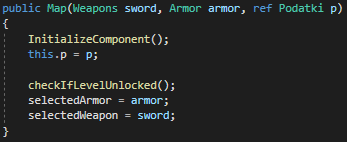
Funkcija »startMain()« pošlje formi za glavni spopad parametre:

* selectedWeapon, kar pošlje izbrano orožje
* selectedArmor, kar pošlje izbran oklep
* boss, kar pošlje končnega nasprotnika za določen nivo
* mob1, mob2 in mob3, kar pošlje navadne nasprotnike (trije na nivo)
* specialMob, kar predstavlja posebnega in močnejšega nasprotnika
* ref p, kar naslednji formi pove od koder mora črpati podatke

Naslednjemu oknu se nato nastavi velikost in širina okna (se nastavi, da je enaka kot pri tej formi), začetna pozicija (pozicija trenutnega okna), nato se še skrije trenutno okno in odpre naslednje.

Funkcija »startGrind()« prejme le parametre: selectedWeapon in selectedArmor, poleg tega pa tudi referenco oz. kazalec na podatke.

##### **3.3.6.2.2 Inicializacija forme zemljevid**



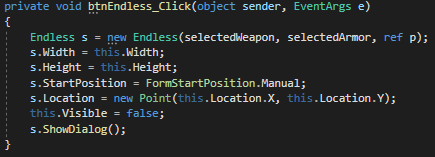
Slika : Konstruktor za zemljevid

Tu se na začetku ponovno zažene slika in vse njene komponente, nato se programu pove kje se nahajajo podatki.

Za tem program preveri katere nivoje mora odkleniti in to stori.

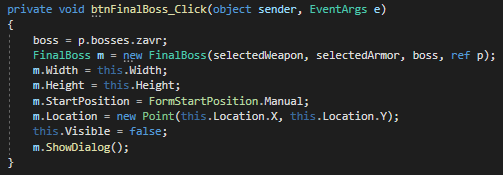
Na koncu se pa še nastavi izbrana oprema igralca.

##### **3.3.6.2.3 Gumbi**



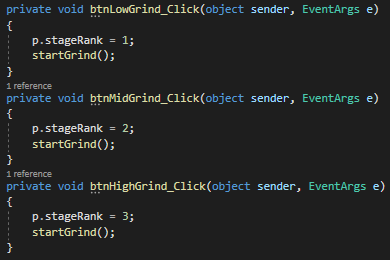
Slika : Gumb za neskončni spopad z vsemi nasprotniki

Tako kot funkciji »startMain()« in »startGrind()«, je zgrajen tudi gumb za neskončni spopad, le da je koda zanj v celotni napisana v gumbu, saj se gumb ne ponavlja in je unikaten.



Slika : Gumb za končnega nasprotnika igre

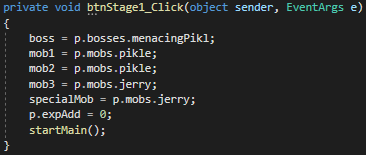
Glede na to da je unikaten tudi gumb za končnega nasprotnika, je tudi pri njemu koda zapisana v celoti v gumbu in ne v posebej funkciji.



Slika : Gumbi za neskončni nivojski spopad

Ob pritisku na gumb (npr. LowRank Grind) se nastavi vrednost »p.stageRank« na 1, kar pomeni, da bo program upošteval le nasprotnike za prvi nivo.

Na koncu gumb še kliče funkcijo »startGrind()«, kar požene okno za neskončni spopad.



Slika : Gumb za izbiro misije oz. glavnega spopada

Ko želi igralec oditi na misijo za materiale, pritisne na željen nivo.

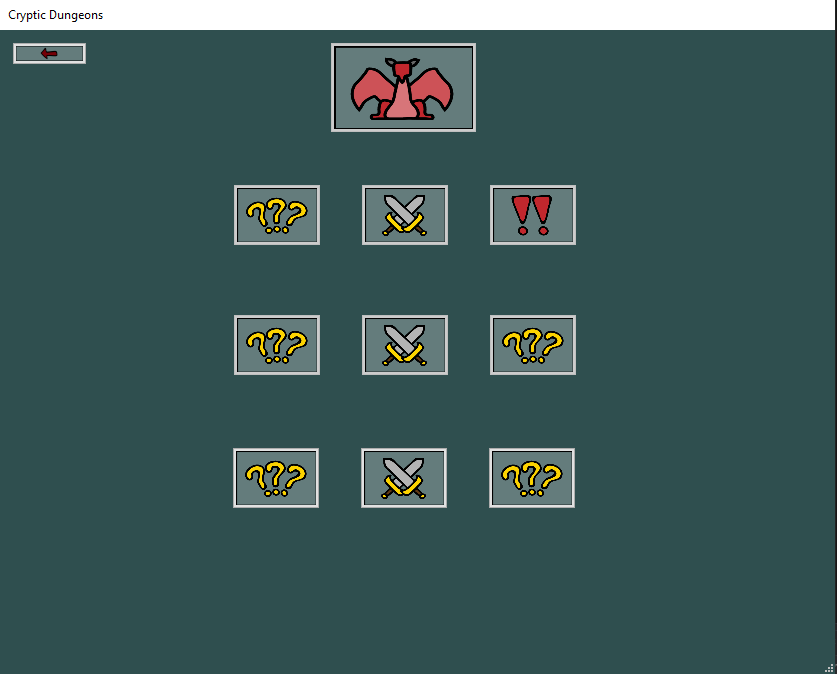
Ob izbiri nivoja se nastavi primerni končni nasprotnik in ostali nasprotniki. Poleg tega se pa še nastavi vrednost »p.expAdd«, kar se uporabi, da igralec dobi primerno količino točk izkušenj za izbran nivo.

Na koncu se izvede še funkcija »startMain()«, kar odpre okno za glavni spopad.

Zemljevid ima tudi gumb za vrnitev na domači zaslon, ampak ga ne bom ponovno prikazal, saj je povsod enak.

### **3.3.7 Glavni spopad oz. misija**

#### **3.3.7.1 Izgled glavnega spopada**



Slika : Prikaz glavnega spopada - izbira stopnje

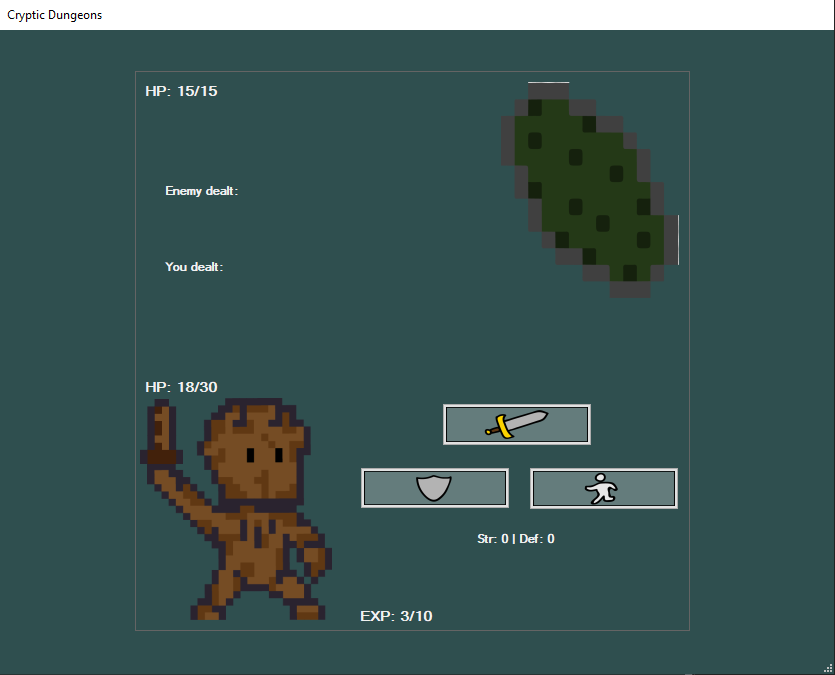
Ko igralec pride na misijo, se mu odpre zgornja slika.

Tu ima nato izbiro med spopadom in naključnim izborom.

Naključni izbor ima 30% možnost preskoka stopnje, 20% možnost da se mu zvišajo življenjske točke za 10 in 50% možnost spopada.

Na tretji stopnji ima igralec možnost spopada s posebnim nasprotnikom, ki je sicer močnejši, ampak je tudi več vreden.

Zadnja stopnja misije je spopad s končnim nasprotnikom, ki pusti za sabo materiale, če ga igralec premaga.



Slika : Prikaz glavnega spopada – bitka

Če igralec izbere bitko, se mu prikaže ta zaslon.

Na vrhu se vidi nasprotnikove življenjske točke, pod njimi se izpisuje igralčev in nasprotnikov napad, pod tem so pa še igralčeve življenjske pike.

Levo spodaj je prikaz igralca, desno zgoraj pa prikaz nasprotnika.

Desno od igralca vidimo 3 gumbe: gumb za napad, gumb za obrambo in gumb za beg iz boja.

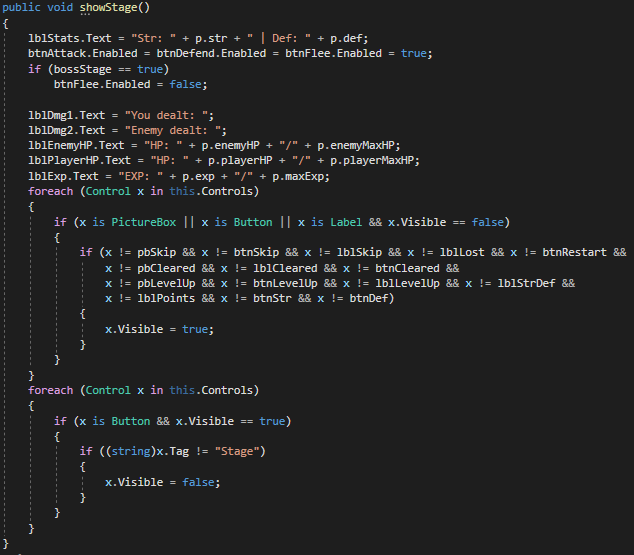
Pod gumbi vidimo še atribute igralca, na dnu zaslona pa nam prikazuje še naše točke izkušenj.

Igra deluje tako, da igralec izbere enega izmed treh gumbov in poskuša zbiti nasprotnikove življenjske točke na 0, medtem ko sam proba preživeti spopad.

Če igralec nasprotnika premaga, se mu odklene naslednja stopnja, vse do končnega spopada, kjer ob zmagi pridobi materiale, katere lahko kasneje uporabi za izdelavo opreme.

#### **3.3.7.2 Koda glavnega spopada**

##### **3.3.7.2.1 Funkcije**



Slika : Funkcija za prikaz zaslona za bitko

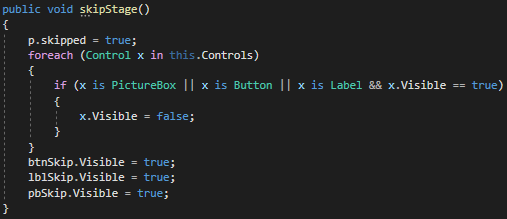
Kar ta funkcija naredi je, prikaže privzete tekstovne oznake (igralčev in nasprotnikov napad in življenjske točke, ter igralčeve atribute in točke izkušenj).

Pred tem še prikaže gumbe za napad, obrambo in pobeg iz boja, ter zaklene gumb za pobeg, če se igralec spopada s končnim nasprotnikom nivoja.

Nato gre skozi »foreach« zanko, ki gre skozi vse gumbe, like in tekstovna polja in prikaže vse gumbe in like za fazo bojevanja.

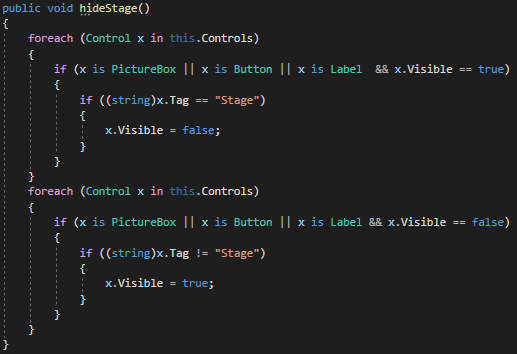
Na koncu gre program skozi še eno »foreach« zanko, kjer pregleda vse gumbe in njihove oznake, ter jih skrije, če nimajo oznake 'Stage'.

Komponente z oznako 'Stage' so tiste, ki se prikažejo ob fazi bojevanja.



Slika : Funkcija za preskok stopnje

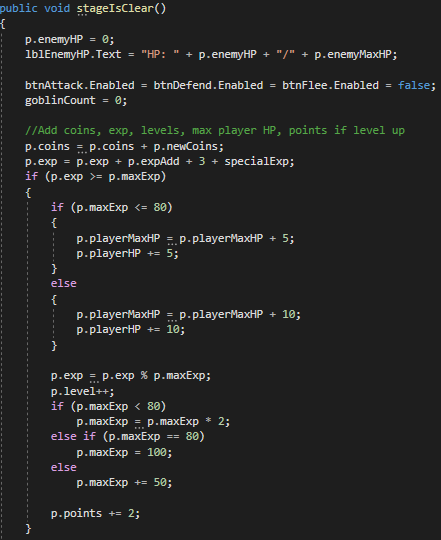
Ob klicu te funkcije gre program skozi zanko, kjer preverja vse slike, gumbe in tekstovna polja v oknu ter jih skrije, če so vidna. Nato prikaže okence za preskok, nakar preskoči stopnjo, ko igralec pritisne na gumb »OK«.



Slika : Funkcija za skrivanje okna za bojevanje

Narejeno na podoben princip kot funkcija za prikaz okna za bojevanje, ta funkcija pošlje program skozi zanko, kjer pregleduje komponente Windows Form-a in jih skrije, če imajo oznako 'Stage'.

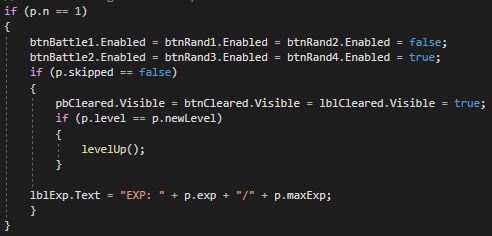
Za tem gre skozi še eno zanko, ki prikaže vse komponente, ki nimajo oznake 'Stage' in nas tako vrne na zaslon za izbiro stopnje.



Slika : Funkcija za prikaz tapravih oken po zmagi – dodeljevanje točk in kovancev

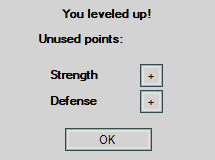
Ko igralec premaga nasprotnika, se izvede funkcija »stageIsClear()«, ki zaklene gumbe za napad, obrambo in pobeg.

Poleg tega igralcu doda kovance v torbo in točke izkušenj, ter preveri, če je že zbral dosti točk, da se mu zviša nivo. Če je zbral dosti točk se mu zviša prag, katerega mora doseči za naslednji nivo, hkrati pa se mu zvišajo skupne življenjske točke, na koncu mu program še dodeli 2 piki, katere lahko dodeli svoji moči ali obrambi.



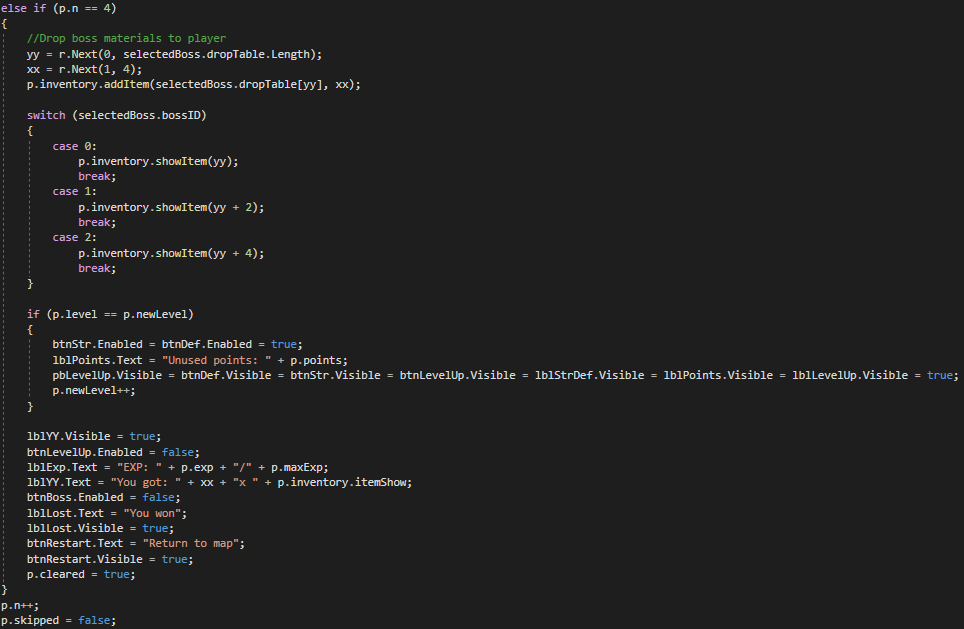
Slika : Funkcija za prikaz tapravih oken po zmagi - odklep naslednje stopnje in pregled zvišanja nivoja igralca

Naslednji del te funkcije preverja katero stopnjo je igralec premagal in odklene naslednjo, ter zaklene stopnjo, ki jo je ravnokar opravil.



Slika : Okno za nov nivo

Nato pregleda še, če je igralec dosegel nov nivo in prikaže zgornjo okence, kjer lahko dodeli svoje točke v moč ali obrambo.



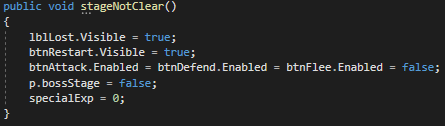
Slika : Funkcija za prikaz tapravih oken po zmagi - zmaga nad končnim nasprotnikom

Če je igralec premagal končnega nasprotnika se bo izvedla koda iz zgornje slike.

Najprej se v spremenljivko »yy« zapiše naključno število med 0 in dolžino nasprotnikove tabele z materiali (če lahko nasprotnik pusti za sabo 2 različna materiala, bo dolžina tabele 2), nato se v spremenljivko »xx« zapiše naključno število med 1 in 3 (to se uporabi za količino spuščenih materialov), na koncu program še dodeli material v igralčevo torbo.

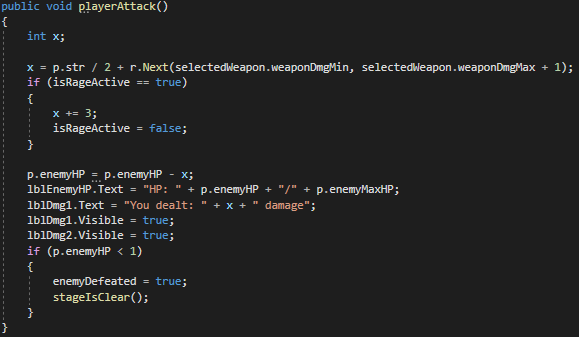
Temu sledi prikaz materiala in količina pridobljenega materiala skozi zanko »switch«.

Za tem se prikaže tekstovno polje, ki izpiše, da je igralec zmagal, hkrati se pa prikaže tudi gumb, ki igralca popelje nazaj na domači zaslon.



Slika : Funkcija za prikaz tapravih oken pri izgubi

Ta funkcija se izvrši le v primeru, da igralčeve življenjske točke pridejo na 0.



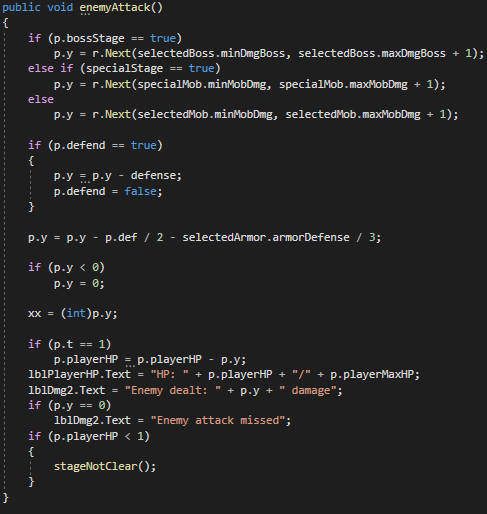
Slika : Funkcija za igralčev napad

Funkcija za igralčev napad se izvrši, ko igralec pritisne gumb za napad.

Tu se nastavi spremenljivka »x«, ki predstavlja napad, izračuna se pa s igralčevo močjo + naključno številko med najnižjim in najvišjim napadom izbranega orožja.

Nasprotniku se nato odbijejo življenjske pike in zapiše napad v tekstovno polje.

Na koncu program še pregleda, če so nasprotnikove življenjske pike pod 1, kar pomeni da je igralec nasprotnika premagal, ter mu prikaže okno za uspešno opravljen boj.



Slika : Funkcija za nasprotnikov napad

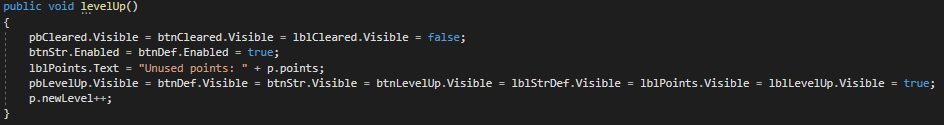
Na začetku funkcije program pregleda kateri nasprotnik je pred igralcem in pravilno nastavi nasprotnikov napad, kar je naključno število med najnižjim in najvišjim možnim napadom nasprotnika.

Za tem preveri, če je igralec pritisnil gumb za obrambo, v katerem primeru se nasprotnikov napad zniža.

Na koncu se od napada odšteje še igralčeva obramba in obramba izbranega oklepa.

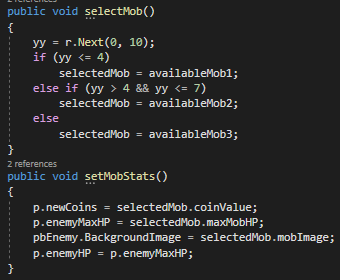
V primeru da je nasprotnikov napad manjši od 0, se izpiše da je nasprotnik zgrešil napad, drugače se pa odbijejo igralčeve življenjske točke.

Na koncu funkcije se še pregleda, če so igralčeve življenjske pike manjše od 1, ter kliče funkcijo za neuspešno izveden boj, če je argument pravilen.



Slika : Funkcija za prikaz okna za višji igralčev nivo

Ta funkcija se kliče, ko igralčeve točke izkušenj presežejo mejo in se mu zviša nivo.



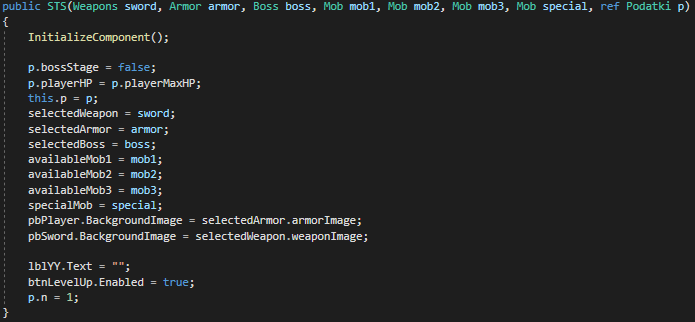
Slika : Funkciji za izbiro nasprotnika

Te dve funkciji delujeta vzporedno in sta klicani ena za drugo.

Delujeta tako da, prva funkcija »selectMob()« izbere enega izmed treh nasprotnikov, ki so bili programu posredovani iz zemljevida in sicer tako, da ima prvi 50% možnost izbora, drugi 30% možnost izbora in tretji, ki je najmočnejši, 20% možnost izbora.

Po izvedbi prve funkcije, se v programu kliče še druga funkcija, ki nastavi spremenljivke »p.newCoins« na število kovancev, ki jih pusti nasprotnik za seboj po smrti, »p.enemyMaxHP« na življenjske točke izbranega nasprotnika in »pbEnemy.BackgroundImage«, ki prikazuje sliko nasprotnika.

##### **3.3.7.2.2 Inicializacija glavnega spopada**



Slika : Konstruktor za glavni spopad (misijo)

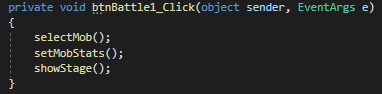
Ta konstruktor prejme parametre objektov orožje, oklep, končni nasprotnik, navadni nasprotniki, poseben nasprotnik in referenco oz. kazalec na podatke.

Na začetku zopet zažene vse vizualne komponente forme ter programu pove od koder naj črpa podatke z »this.p = p«

Nato nastavi spremenljivke na sprejete parametre, da se lahko te objekti uporabljajo v programu.

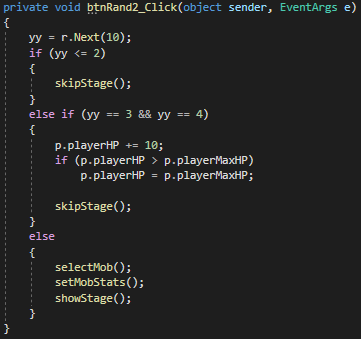
Na koncu pa še nastavi stopnjo misije na 1, da začne igralec na prvi stopnji.

##### **3.3.7.2.3 Gumbi**



Slika : Gumb za spopad

Ob kliku na gumb, z ikono dveh mečev, se kličejo funkcije za izbor nasprotnika in nastavljanje vrednosti nasprotnika, ter funkcija za prikaz faze bojevanja.

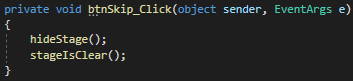


Slika : Gumb za naključno izbrano stopnjo

Če igralec pritisne na gumb z tremi vprašaji (gumb naključnosti) se zgodi ena izmed treh zadev.

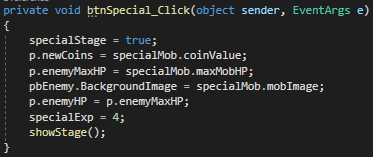
Najprej se v spremenljivko »yy« zapiše naključno število med 0 in 9, temu pa sledi preverjanje tega števila.

* V primeru da je število med 0 in 2 (30% možnost), igralec preskoči stopnjo.
* V primeru da je število 3 ali 4 (20% možnost), igralec pridobi 10 življenjskih pik in gre na naslednjo stopnjo.
* V ostalih primerih se pa pred igralcem prikaže nasprotnik, s katerim se more igralec bojevati.



Slika : Gumb za preskok stopnje

Gumb, ki se uporabi za preskakovanje stopnje.

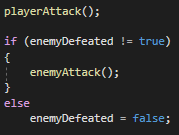


Slika : Gumb za izbor posebnega nasprotnika

Ob pritisku na gumb z ikono dveh rdečih klicajev, se nastavijo vrednosti nasprotnika na vrednosti posebnega nasprotnika.

Posebni nasprotniki so močnejši od navadnih nasprotnikov, ampak imajo možnost, da pustijo za sabo kamne za nadgrajevanje opreme.

Poleg tega pa igralcu po zmagi dajo tudi več točk izkušenj.



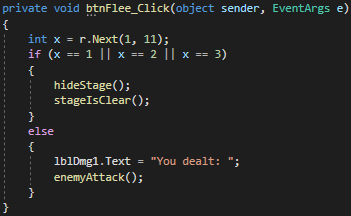
Slika : Gumb za napad

Ko želi igralec napasti nasprotnika se izvede naslednja koda.

Najprej se pokliče funkcijo »playerAttack()«, ta funkcija je za napadanje nasprotnika

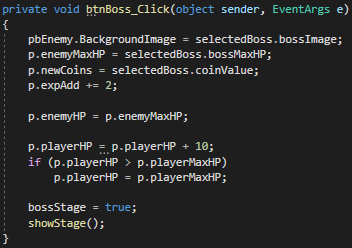
Po izvedeni funkciji program pregleda če je igralec nasprotnika premagal

Če je nasprotnik še živ, se izvede funkcija »enemyAttack()«, kar zbije igralcu določeno število življenjskih pik.

Slika : Gumb za pobeg iz boja

Ob pritisku na gumb za pobeg program preveri če je izbrano naključno število med 1 in 3, kar povzroči, da uporabnik pobegne pred nasprotnikom in gre na naslednjo stopnjo nivoja.

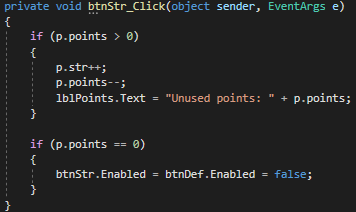
Če število ni med 1 in 3, se izvede funkcija »enemyAttack()« in nasprotnik napade igralca.



Slika : Gumb za končnega nasprotnika

Tako kot funkcija »setMobStats()« se nastavijo atributi nasprotnika na končnega nasprotnika.

Po temu se igralcu prišteje 10 življenjskih točk in preveri če presežejo prag igralčevih življenjskih točk, ter se nastavijo na prag, če je argument pravilen.

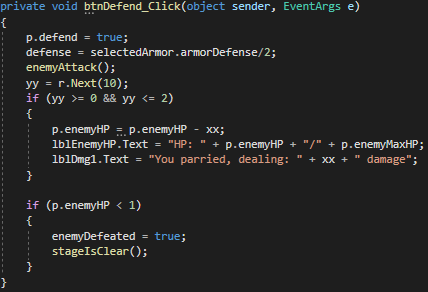


Slika : Gumb za dodeljevanje atributov

Ko uporabnik pritisne na gumb »+« se mu zviša moč ali obramba, glede na kateri »+« gumb je pritisnil.

Ob pritisku na gumb se tudi odbije točka, ki jo je pridobil z zvišanjem nivoja.

Na koncu se še preveri, če ima pozitivno število točk in zaklene gumbe za dodeljevanje atributov, če ni.



Slika : Gumb za obrambo

Ob pritisku na gumb z ikono ščita, se nastavi spremenljivka »p.defend« na »true«, kar v funkciji »enemyAttack()« zniža nasprotnikov napad za polovico izbranega oklepa.

Za tem se v spremenljivko »yy« zapiše naključno število med 0 in 9, kar predstavlja 100%.

Nato preveri če je vrednost spremenljivke med 0 in 2, kar predstavlja 30%.

Če je argument pravilen, igralec vrne nasprotnikov napad.

Na koncu se preveri, če je nasprotnik zgubil vse življenjske točke in kliče funkcijo »stageIsClear()«, če je.

# **4 ZAKLJUČEK**

Projekt sem začel brez kakršnegakoli znanja razvijanja iger, poznal sem le določeno količino programskega jezika in večino kode, ki sestavlja program, sem moral naknadno ugotoviti s pomočjo interneta.

Na srečo je splet že toliko razvit, da sem našel vse z minimalnim trudom in brez pomoči StackOverflow spletne strani (namenjena pomoči pri pisanju programske kode) mi projekta ne bi uspelo narediti, vsaj ne tako optimizirano kot je končni produkt.

Skozi grajenje projekta sem ugotovil, da sem se pravilno odločil, saj mi je projekt razširil obzorje.

Če bi moral projekt ponovno začeti bi spremenil par stvari.

* Spremenil bi program za izrisovanje likov. Za ta projekt sem uporabljal Adobe Illustrator, saj je relativno enostaven za uporabo, mu pa manjkajo pripomočki za izdelovanje 8-bitnih likov. Če bi moral like izdelati na novo, bi uporabil program Aseprite, ki je namenjen za tako grafiko.
* Namesto izdelave programa, ki odpira razna okna, bi igro izdelal za HTML5 (naredil bi spletno aplikacijo), saj je grafika in stil igranja bolj primeren za splet.
* Od samega začetka bi razmišljal bolj v smeri optimizacije, saj sem za projekt porabil število ur, da sem spremenil zadeve, ki bi lahko že prvotno drugače zapisal.
* Pred začetkom izdelave programa, bi se bolj pozanimal o izvedbi programa in bi si zapisal bolj podroben načrt, da bi izdelava projekta potekala bolj tekoče.

Kljub vsem problemov, na katere sem naletel, je bil projekt precej informativen in sem se veliko naučil. Zahvaljujoč predmetu Programiranje 2 in profesorju Aleksandaru Lazareviću bom nadaljeval z razvijanjem iger in v upanju izdelal nekaj, kar bom lahko delil z ostalimi.