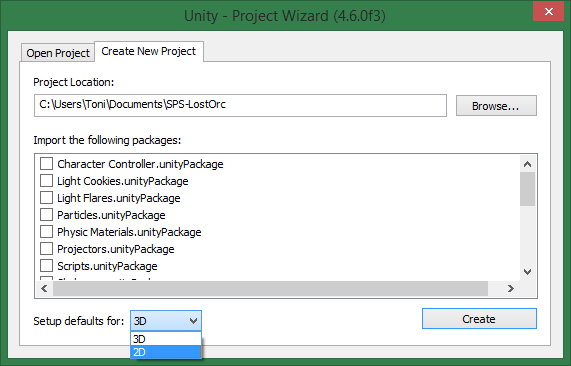
# Student poučava studenta – Day 2

Autor: Toni Steyskal

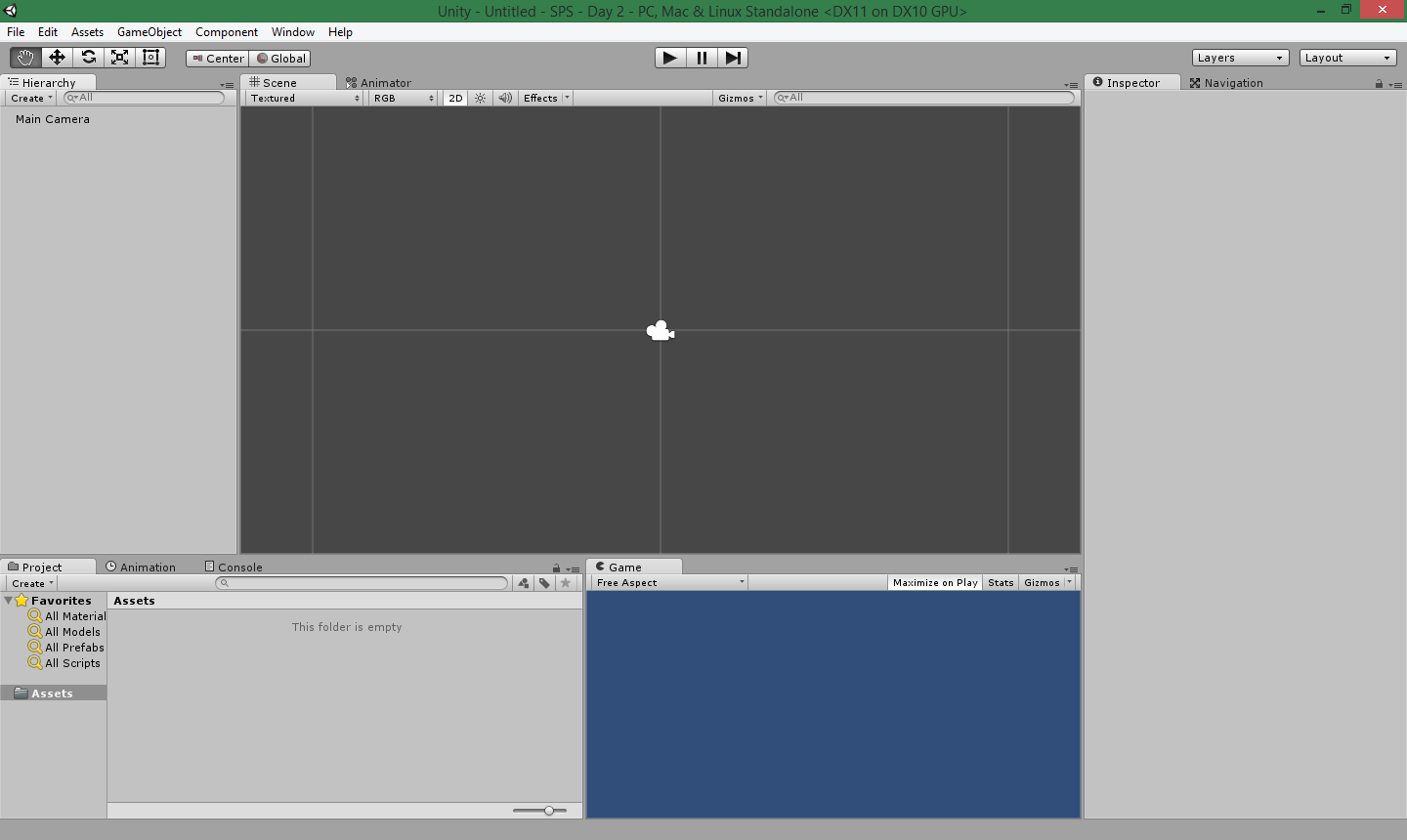
Kako bi započeli s kreiranjem igre, pokrenite Unity i kreirajte novi projekt. Bitno je odabrati **Setup defaults 2D**, kako bi sam Unity editor tretirao naš projekt kao 2D projekt, postavio prikaz scene u 2D prikaz, prebacio kameru na ortografski pogled i svaku novu uvezenu sliku označio kao **sprite[[1]](#footnote-1)** (prikazano na slici 1).



***Slika 1:*** *Kreiranje 2D projekta*

Nakon što ste kreirali projekt otvara se prazan Unity editor u kojem ćemo tijekom današnje vježbe kreirati našeg 2D lika. Za daljnji rad potrebno je preuzeti sve materijale koji se mogu naći na sljedećem linku **LINK**. (\* opis assetsa\*)

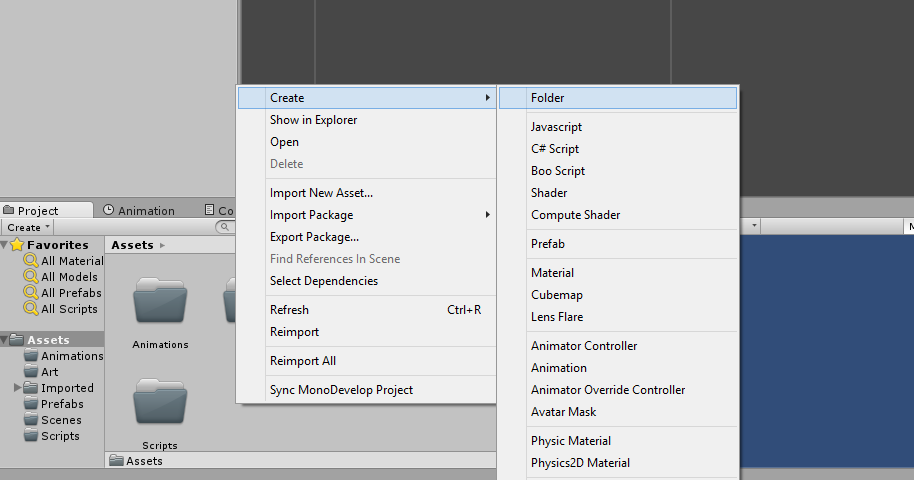
Otvoren editor vidljiv je na slici 2.



***Slika 2:*** *Unity editor*

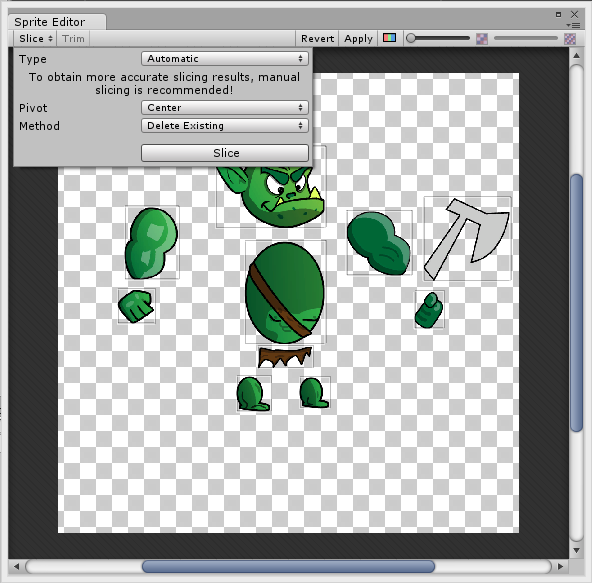
Preimenujte Main Camera u **MainCamera** preko **Inspector-**a, jer će se kroz projekt koristiti **CamelCase[[2]](#footnote-2)** notacija da postoji jedinstven način nazivanja objekata u hijerarhiji ukoliko ih je potrebno kasnije referencirat, a i to predstavlja stvar dobre prakse. ☺

Zbog daljnje preglednosti kreirajte sljedeće datoteke: **Animations, Art, Prefabs, Scenes** i **Scripts**. Kao i što sami nazivi pojedinih datoteka govore, u pojedine će se spremati razne animacije, slike, scene (nivoi), skripte i **prefab[[3]](#footnote-3)**-ovi koji će se koristiti tijekom razvoja igre. Kreiranje nove datoteke je vidljivo na slici 3.



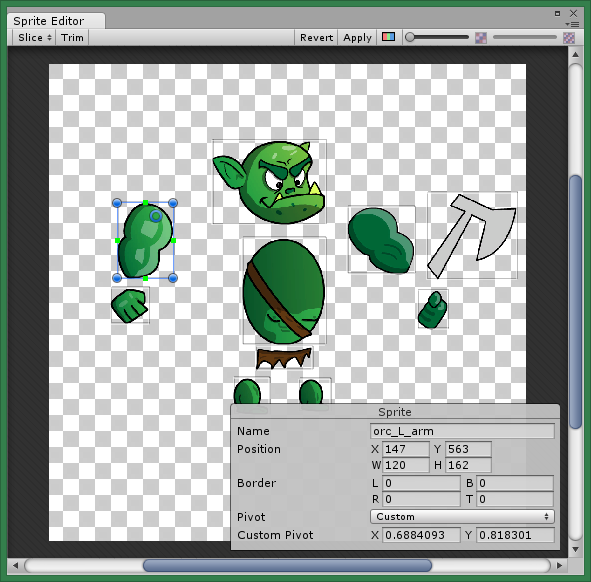
***Slika 3:*** *Kreiranje nove datoteke*

Iz preuzetih materijala kopirajte (ili drag & drop) **OrcArt.png** u kreiranu Art datoteku unutar Unity Editora. Označite uneseni sprite te promijenite njegove postavke **Sprite Mode -> Multiple**  i odznačite **Generate Mip Maps[[4]](#footnote-4)**. Kako se zapravo naša slika sastoji od više dijelova samog lika potrebno ju je urediti unutar **Sprite Editor**-a. Rascijepat ćemo ju na manje dijelove koje ćemo koristiti za kreiranje našeg lika, kao što je vidljivo na slici.



***Slika 4:*** *Rascijepanje sprite-a*

Nakon pritiska na **Apply** sprite OrcArt dobit će zasebne dijelove odjeljenje u slici 4. Pojedine dijelove potrebo je preimenovati (unutar Sprite Editor-a) prema slici 5. I dodijeliti im pozicije pivota prema tablici 1.

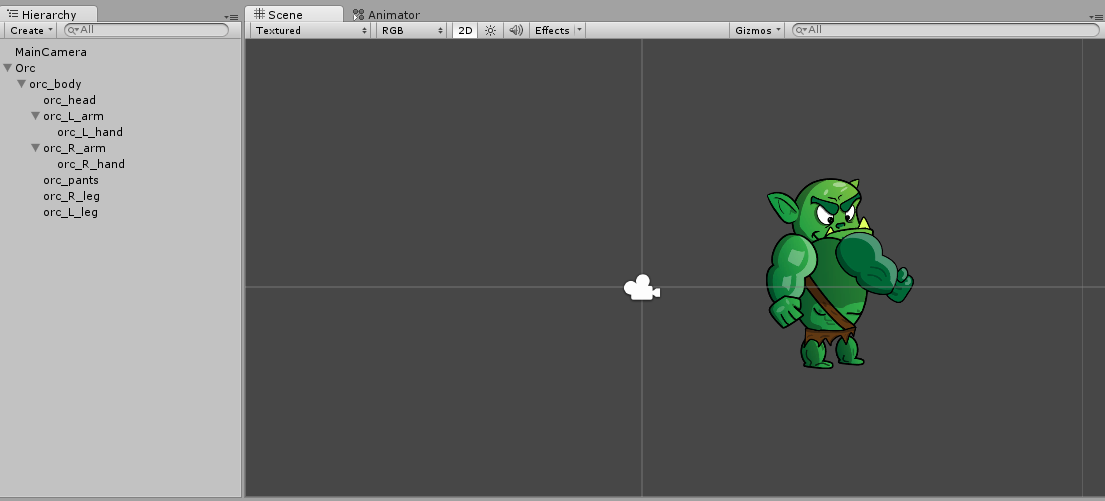
***Slika 5:*** *Imenovanje dijelova*

Izuzetno je bitno da se dijelovi imenuju kako su prikazani, jer se predefinirane animacije za taj preuzeti asset vežu na nazive i hijerarhiju objekta. Tako su nazivi redom: **orc\_body, orc\_head, orc \_L\_arm, orc\_L\_hand, orc \_R\_arm, orc \_R\_hand, orc\_pants, orc \_R\_leg** i **orc \_L\_leg** (kako autor lika očito ne poznaje notacije imenovanja i konzistentnost, postoje razmaci i prije povlake \_, koje je kasno u noć teško uočiti, zato pazite kako dajete nazive i uštedite korisnicima poput meni noć debuggiranja i traženja greške zbog par razmaka…).

|  |  |  |
| --- | --- | --- |
| **Orc** | **Pivot X** | **Pivot Y** |
| **Head** | 0.486188 | 0.1940409 |
| **Body** | 0,5 | 0,5 |
| **LeftArm** | 0.6884093 | 0.818301 |
| **LeftHand** | 0.263753 | 0.7673514 |
| **RightArm** | 0.219031 | 0.8085149 |
| **RightHand** | 0,5 | 0,5 |
| **Pants** | 0,5 | 0,5 |
| **LeftLeg** | 0.3634006 | 0.9244167 |
| **RightLeg** | 0.3787468 | 0.921729 |

***Tablica 1.*** *Pivot pozicije*

Kako bi kreirali vlastitog lika prvo moramo stvoriti prazan objekt u sceni koji će sadržavati sve njegove dijelove. Desni klik unutar hijerarhije pa **CreateEmpty**, kreirani objekt preimenujte u **Orc**. Tom objektu dodajte pojedine dijelove (drag & drop) OrcArt-a kao što je prikazano na slici 6, izuzev sjekire, naš Orc nije ratoboran, samo mrzovoljan i izgubljen (ukoliko želite, kasnije možete dodati sjekiru i mogućnost napadanja, te tako proširiti igru).



***Slika 6:*** *Orc hijerarhija*

Naime dolazi do problema nepravilnog preklapanja dijelova koji čine našeg lika. Ova vrsta problema se može riješiti na 2 načina, preko određivanja Z osi zasebno za svaki dio ili slojeva sortiranja. U ovom primjeru će se koristi **Sorting Layer** pristup. Preko inspector-a našeg Orc objekta pod opcijom **Layer** odaberite **Add Layer**. Pod Sorting Layers dodajte Player pod **Layer 1** i ispod u Layer dodajte također Player pod **User Layer 8**. Zatim se vratite na sam objekt te odaberite **Player** kao njegov sloj i dozvolite Unity-u da učini isto njegovoj djeci (svim podijelovima).

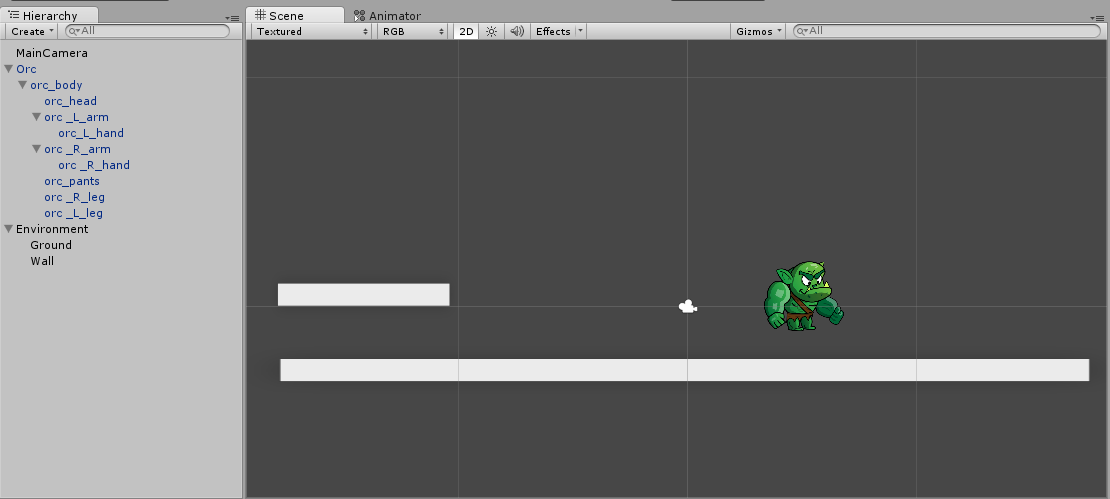
Sad kako smo Orc-a dodali na Player sloj potrebno je njegove dijelove sortirati i pozicionirati kako bi on sam dobio nekakav oblik. Učinite to prema sljedećoj tablici.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Orc** | **Layer** | **Sorting Layer** | **Order** | **X position** | **Y position** | **Z position** |
| **Head** | Player | Player | 1 | 0,1458159 | -0,1245941 | 0 |
| **Body** | Player | Player | 0 | 0 | 1,620405 | 0 |
| **LeftArm** | Player | Player | 4 | -0,7376138 | 0,183571 | 0 |
| **LeftHand** | Player | Player | 5 | -0,4679554 | -1,230647 | 0 |
| **RightArm** | Player | Player | -2 | 0,5498632 | 0,01226807 | 0 |
| **RightHand** | Player | Player | -1 | 1,059995 | -1,041762 | 0 |
| **Pants** | Player | Player | 3 | -0,01526686 | -1,107126 | 0 |
| **LeftLeg** | Player | Player | 2 | -0,2869641 | -0,9984478 | 0 |
| **RightLeg** | Player | Player | 1 | 0,2737674 | -1,022029 | 0 |

***Tablica 2.*** *Sortiranje i pozicija lika*

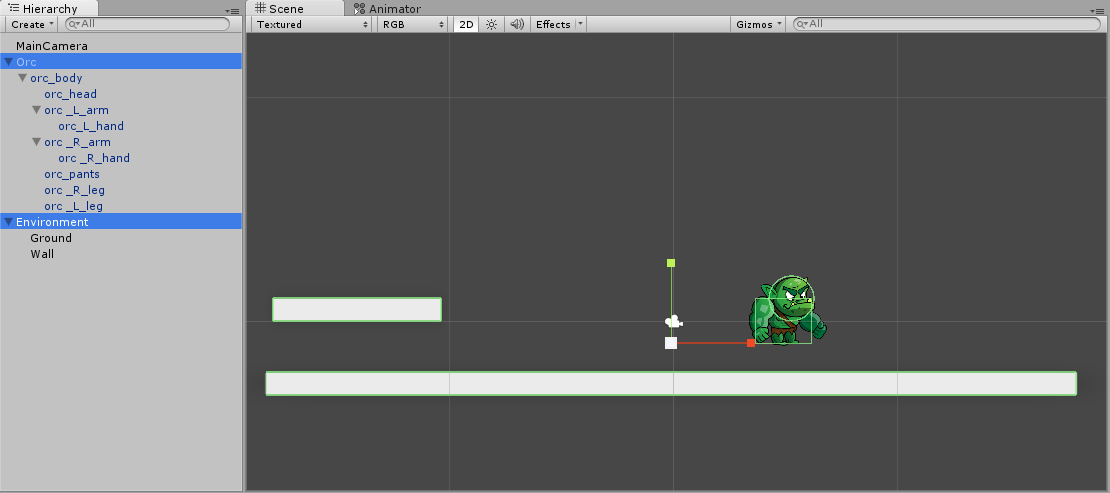
Sada kako smo konačno kreirali lika po našoj mjeri, povučemo gotov objekt iz hijerarhije u datoteku **Prefabs**, te tako stvaramo predložak glavnog lika koji kasnije možemo mijenjati ili koristiti i negdje drugdje.

Budući da sada imamo kreiranog Orc-a, potrebno je kreirati određene skripte kako bi mogli njime upravljati. Da bi mogli testirati dali sve što radimo funkcionira kako treba, kreirat ćemo privremenu testu okolinu pomoću **Wall.png** slike. Izgled prve scene može biti proizvoljan, bitno je da postoji dio na kojem naš lik može stajati i na njemu se kretati. Okvirni izgled scene vidljiv je na slici 7. Također promijenite postavu **Size** kamere na 15.



***Slika 7:*** *Dodavanje okoline*

Pokretanjem igre uočavamo da se zapravo ništa ne događa, već sve stoji kako je i predefinirano. Kako bi na našeg lika utjecala gravitacija i kako bi on mogao biti u interakciji s fizikom 2D svijeta, dodajemo mu komponentu **Physics 2D -> Rigidbody2D[[5]](#footnote-5)**. Također označujemo postavku **Fixed Angle** jer u ovoj igri ne želimo da se lik rotira, a ona nam to omogućuje. No i dalje primjećujemo da nije sve kako treba biti, naime naš igrač prolazi kroz pod, da bi to izbjegli dodanim dijelovima okoline potrebno je dodati komponente **Physics 2D -> Box Collider 2D**. Bitno je također podesiti postavke dodanog collider-a da odgovara veličini sprite-a (Size: X = 0.25 i Y = 2.85). Istu stvar je potrebno učiti i kod igrača, kako bi Unity znao koliko gdje su rubne granice igrača, no njemu dodajemo jedan **Circle Collider 2D** (X = 0.35, Y = 2, Radius = 1) i jedan **Box Collider 2D** (Center: X = 0, Y = 1, Size: X = 2.5, Y = 2). Konačan izgled prikazan je na sljedećoj slici.



***Slika 8:*** *Dodavanje collider-a*

Sad smo definirali okolinu i način interakcije okoline s igračem. Samo kretanje se rješava preko kreiranja skripta (u ovom slučaju koristit ćemo C#) i pridruživanje istih objektu. Unity sadrži ugrađen IDE za kreiranje skripata nazvan **Monodevelop**, no skripte se mogu kreirati u bilo kojem razvojnom okruženju koje podržava jezik u kojem se one kreiraju. Najjednostavniji način je korištenje samog Monodevelop alata kako je on već integriran i prilagođen Unity-u. Kreirajte C# skriptu **CharacterController2D** unutar **Scripts** foldera, dodijelite ju na našeg igrača te ju otvorite. Skripta neće biti detaljno objašnjena već se koriste komentari unutar samog koda kako bi se shvatili pojedini dijelovi. Skripta je vidljiva na slici 9.



***Slika 9:*** *CharacterController2D*

Spremite skriptu te pokrenite igru, sada se Orc može kretati te se okreće prema strani u koju se kreće. No vidimo određenu manu, naime kretanje ne izgleda prirodno, odnosno ne postoje animacije. Animacije ćemo dodati u nastavku.

1. tip teksture koji se koristi u 2D razvojnom okviru [↑](#footnote-ref-1)
2. praksa pisanja složenih riječi ili fraza tako da svaka riječ ili kratica počinje velikim slovom [↑](#footnote-ref-2)
3. predložak iz kojeg se može stvoriti novi objekt u sceni [↑](#footnote-ref-3)
4. Mip Maps predstavljaja listu manjih verzija slike koje se koriste za optimiziranje performansi za objekte koji su udaljeni od kamere [↑](#footnote-ref-4)
5. klasa koja objektu pruža funkcionalnosti fizike i stavlja objekt pod kontrolu sustava za fiziku [↑](#footnote-ref-5)