UNIVERZITET U NOVOM SADU
PRIRODNO-MATEMATIČKI FAKULTET
DEPARTMAN ZA MATEMATIKU I INFORMATIKU

Svemirska pucačina "Andromeda" u Unity platformi

Informatički Projekat

Tojagić Rastko

Novi Sad, 2020.

Predgovor

Razvoj računarskih igara je tokom protekle decenije veoma uznapredovao zahvaljujući pojavi platformi koje taj razvoj znatno olakšavaju i ubrzavaju. Jedna od takvih platformi jeste Unity koji podržava razvoj kako jednostavnijih 2d tako i zahtevnijih 3d igara. U ovom informatičkom projektu pokazaćemo upravo koliko je jednostavno napraviti 2d svemirsku pucačinu u svega nekoliko stotina linija koda, kao i neke često korišćene obrasce (eng. design patterns) u razvoju igara.

Tojagić Rastko

Novi Sad, Septembar 2020.

Sadržaj

1 Platforma za razvoj igara Unity

1.1 MonoBehaviour i pisanje skripti u jeziku C#

MonoBehaviour je osnovna klasa koju svaka Unity skripta nasleđuje. Kada koristimo C# za skriptni jezik ovu klasu moramo eksplicitno da nasledimo [1]. U svakom trenutku moguće je isključiti ili ponovo uključiti skriptu. Isključivanje izuzima skriptu iz aktivnih skripti. Ukoliko je skripta uključena i ima definisane neke od specifičnih funkcija koje se pozivaju u određenom trenutku tokom životnog ciklusa objekta smatra se aktivnom i reaguje na pozive tih funkcija onako kako je korisnik to definisao. Neke od ključnih funkcija su: *Start()*, *Update()*, *FixedUpdate()*, *OnEnable()*, *OnDisable()*. Životni ciklus skripte prikazan je grafikonom 1. Kasnije u tekstu biće data preciznija objašnjenja za najbitnije funkcije.

Drugim rečima MonoBehaviour predstavlja definisano ponašanje nekog živog objekta u svetu igre. Svaki objekat u svetu igre naziva se *GameObject* i tipa je istoimene klase. Svaki MonoBehaviour jeste *GameObject*. Na GameObject je moguće dodati komponente. Treba imati na umu da ovakav sistem koji Unity pruža nije pravi ECS (Entity Component System) te da zbog ovakvog dizajna cela platforma radi sporije. Unity od skoro uvodi pravi ECS uz *Burst Compiler*, ali ovo još uvek nije deo standardne platforme [1].

1.2 Unity Editor

Editor je grafičko okruženje namenjeno za postavljanje sveta igre. Editor pruža vizuelni pregled svih objekata na sceni, kamere i efekata. Prozor je podeljen na nekoliko potprozora čije uloge su da olakšaju podešavnja i pregled igre.

SceneView predstavlja pregled trenutne, aktivne scene. Scena predstavlja skup objekata i kameru koji su u datom trenutku prikazani korisniku. Poželjno je zasebne logičke celine igre odvajati u zasebne scene, tako na primer, možemo imati scenu za *glavni meni* i *scenu igre*, gde bi kroz u meniju bila omogućena neka podešavanja, pregled najboljih rezultata i slično, dok je scena igre sama igra.

Hierarchy View omogućava pregled svih objekata na trenutnoj sceni. Objekti mogu biti u roditelj-dete odnosu, te stoga i ime, jer tako čine hijerarhiju.

Project View daje pregled svih fajlova trenutno otvorenog projekta na disku.

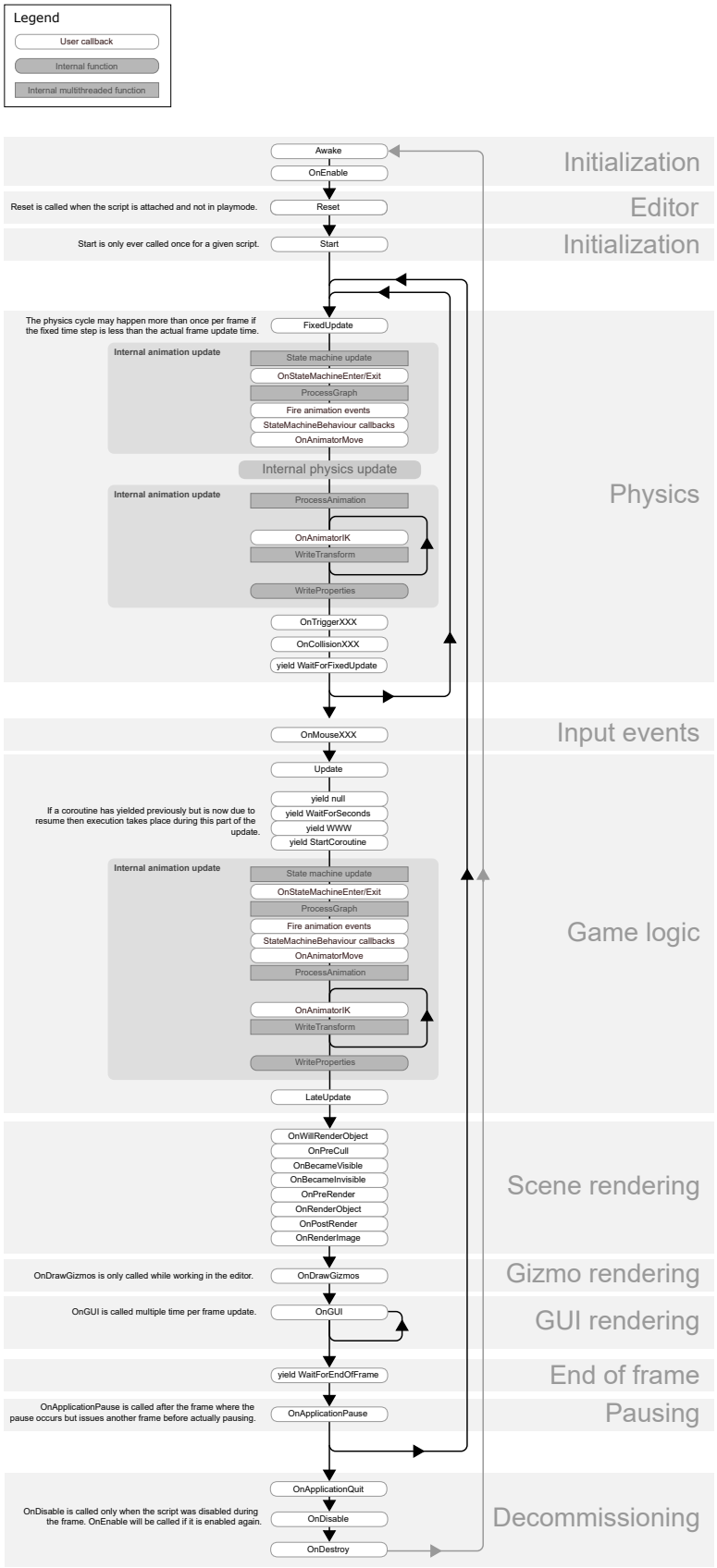

Figure 1: MonoBehaviour lifecycle

Game View obično složen pored pregleda scene prikazuje svet igre onakav kakav će biti prikazan igraču, dakle bez pomoćnih elemenata poput kvadratne mreže ili mogućnosti pomeranja objekata. ** TODO ubaciti sliku editora **.

2 Osnovni pojmovi i sistemi u igrama

2.1 Game Loop

Ključna tačka izvršavanja igre jeste glavna petlja igre (eng. game loop). Jednom kada je igra pokrenuta sve ono što je vidljivo korisniku, grafika, zvuk, interakcija procesira se kroz glavnu petlju. U tom jednom prolazu petlje koji nazivamo *tick* ili *frame* dešava se iscrtavanje grafike, reagovanje na korisnički unos (unos sa tastature ili putem miša ili nekih drugih perifernih uređaja) te reakcija sveta na zadate promene... odnosno, ovde se dešava sve, zato je glavna petlja srce same igre. Modernije arhitekture dozvoljavaju određeni stepen paralelizacije nekih procesa, pa bi na primer iscrtavanje grafike moglo da se izdvoji u paralelan proces i slično.

2.1.1 FPS

FPS skraćeno od *frames per second* je osnovna mera performansi igre. Ova mera kazuje nam koliko je puta po sekundi moguće izvršiti glavnu petlju. Veći broj je i bolji, pa tako igra koja se izvršava na manje od 30 frejmova po sekundi nije prijatna za igranje i događa se takozvano *seckanje*, odnosno objekti se ne pomeraju glatko kako bi trebalo već iscepkano što narušava celokupan doživljaj igranja. Nizak FPS može da bude rezultat loše napisanog koda ili prosto zastarelog hardvera. Moderne igre se trude da se održe na 60 FPS.

2.1.2 Update

Update je funkcija koja se poziva upravo iz glavne petlje igre. To je funkcija koja je može biti definsana u MonoBehavior klasi. Update funkcija se poziva nad svakim objektom tipa MonoBehavior (ukoliko je definiše) tačno jednom svakog frejma [1]. Sva promenljiva ponašanja objekta mogu se definisati ovde.

2.1.3 FixedUpdate

Slično funkciji Update, i ova funkcija se poziva periodično ali ne svakog frejma, već na fiksno vreme svake 0.02 sekunde [1]. Prema uputstvima iz dokumentaicje, u ovoj funkciji se mora raditi sve što ima veze sa bilo kakvim izračunavanjem fizike. U igri se oslanjamo na Unity-jev sistem za fiziku za kretanje, pa ćemo se dalje u tekstu detaljnije baviti time.

2.1.4 Vremena, klasa Time

Jedna od bitnijih stavki u igrama jeste vreme. Merimo dve vrste vremena, koja očitavamo iz klase *Time*.

Time.time je vreme koje je proteklo od pokretanja igre i ono je korisno obično kada želimo da ograničimo izvršavanje nekog dela koda vremenski. Na primer, ne želimo da neprijatelji prebrzo ispaljuju metke na igrača, te možemo meriti vreme proteklo od prethodnog pucanja i proveriti da li je to vreme veće od neke konstante koja nam predstavlja minimalan vremenski interval između dva pucnja.

Time.deltaTime je vreme za koje se izvršio prethodni frejm. Ovo je jako bitna stavka, i potrebno nam je kada imamo bilo kakvo kretanje u igri. ** TODO: Ubaciti grafik koji objasnjava vaznost mnozenja vrednosti kretanja sa deltatime **

2.2 Obrasci (eng. design patterns)

2.2.1 "Singleton" obrazac

2.2.2 "Command" obrazac

3 Andromeda - Svemirska pucačina

3.1 GameManager klasa

3.2 Kontrole

3.3 Ponašanje svemirskog broda i ShipController klasa

3.4 Neprijatelji i osnovno ponašanje

3.5 Poboljšanja i optimizacije

References

[1] Unity. *Unity Documentation*.