

# Laboratorul 4

## Animații

### 1 Asset-uri

În acest laborator vom avea nevoie de caractere și de animații pentru acestea. Un loc bun în care putem găsi caractere și animații pentru acestea este Mixamo.

De asemenea, vom avea nevoie de teren în aplicația pe care o vom realiza. Pentru teren putem folosi un asset pack precum acesta sau acesta. ATENȚIE: Aceste asset pack-uri nu au fost concepute pentru URP, deci materialele obiectelor trebuie schimbată în cazul în care proiectul nostru folosește URP.

### 2 Movement

În aplicație, utilizatorul va trebui să controleze un caracter. Puteti urma metodele prezentate în primele două tutoriale ale acestei serii pentru a implementa movement-ul. De asemenea, puteți porni ca starting point pentru movement de la controlul bilei din jocul realizat în semestrul anterior la materia *Introducere în Programarea Jocurilor pe Calculator*, facând modificările aferente (collider de tip capsulă, freeze rotation pe rigidbody, etc...).

### 3 Animații

Acest laborator explică lucrul cu animații atât în cazurile 2D cât și în cazul 3D. Pe noi ne interesează doar animațiile 3D în acest laborator.

## 4 Cerințe de laborator

### 4.1 Crearea unei scene (0.05p bonus)

Creați o scenă în care să existe un teren cu detalii (copaci, iarba, denivelări, etc) și un caracter. Terenul poate fi luat de pe *Asset Store*, sau creat manual, sau chiar generat procedural (pentru studenții care vor să învețe mai multe).



Figure 1: Un personaj plasat deasupra unui teren.

### 4.2 Controlul caracterului (0.1p bonus)

Definiți un script prin care să puteți controla caracterul adăugat în scenă prin folosirea tastelor WASD sau a săgeților de pe tastatură. Miscarea va avea comportament fizic (caracterul se va adapta la denivelările drumului și se vor realiza coliziunile cu celelalte obiecte din scenă). Orientarea caracterului va fi în concordanță cu direcția de deplasare a acestuia.



Figure 2: Drumul pe care s-a deplasat caracterul controlat de jucător.

### 4.3 Jumping (0.05p bonus)

La apăsarea tastei *spatiu*, caracterul va sări în aer. Acest lucru se poate face prin incrementarea componentei *y* a lui `rigidbody.velocity` cu o valoare pozitivă atunci când jucătorul se află pe pământ. Dacă dorim ca săritura să ajungă până la o înălțime maximă de *jumpHeight* unități, atunci `rigidbody.velocity.y` trebuie să fie incrementat cu  $\sqrt{-2 \times Physics.gravity.y \times jumpHeight}$ .



Figure 3: Sarituri prin scenă.

### 4.4 Animații (0.1p bonus)

Descărcați animații pentru *idle*, *running*, *jumping* și *dying*. Creati un **Animation Controller** care să conțină stări pentru toate animațiile descrise anterior și care să fie legate prin tranziții condiționate de variabile precum *IsRunning*, *IsJumping* sau *IsDying*. Testați manual acest **Animation Controller** folosind o componentă de tip **Animator**.



Figure 4: Caracter care are o animație de alergare.

### 4.5 Legarea animațiilor (0.1p bonus)

Din scriptul de mișcare, modificați valorile variabilelor definite în **Animation Controller** astfel încât animațiile să corespundă cu acțiunile caracterului. Pentru *dying* puteți implementa ca jucătorul să moară după un anumit număr de secunde de la deschiderea jocului.



Figure 5: Caracter mort.