



***אויבים והתנגשויות***

אז אחרי שיצרנו דמות ראשית ומערכת ירי אנחנו רוצים שהדמות שלנו תוכל לירות על משהו ולא רק באוויר.  
אויבים הם גם סוג של prefab- לשחקן אין שליטה עליהם, לרובם יש מאפיינים זהים, וככל הנראה הם מופיעים כמה פעמים במהלך משחק. לרוב האויבים 'יורשים' מאובייקט אב אחד ופשוט משכללים את הנתונים שלו, לכן במשחקים קלאסיים בד"כ נראה קבוצות של שונות אויבים אך עם מערכת פעולה דומה.

תחילה נבנה אב טיפוס לכל האויבים במשחק, ניצור אובייקט משחק פרימיטיבי שישמש כבסיס, היות והשחקן הראשי שבחרנו כדוגמא היה קובייה נראה לי מן הראוי להמשיך באותו קו ונבנה גם את האויב כקובייה.  
כדי להבדיל בין השחקן הראשי לאויב כדאי שניתן לו חומר בצבע שונה ונשנה גם את הגודל במקצת.  
ניתן לו שם ונגדיר אותו כ-prefab ע"י גרירה לתיקיה הייעודית.

## סקריפט בסיסי

ניצור סקריפט חדש ונחבר אותו לדגם של ה'אויב' **בחלון הפרויקט** (כי אם נחבר לאובייקט שמופיע בסצנה החיבור יחשב כ'דריסה' של המאפיינים של הדגם, ואנחנו רוצים לשנות את הדגם דווקא כדי שיוריש לאינסטנסים שלו).  
נרצה שהאויב יקיים את הדברים הבאים: 1) זז כלפי מטה במהירות אחידה. 2)אם הוא הגיע לתחתית העמוד שלא יושמד, אלא יחזור למעלה אך מנקודה אחרת בציר ה-X, כלומר יצוץ באופן רנדומלי מלמעלה.3)ברגע שאובייקט 'יתנגש' בשחקן או בלייזר הוא יושמד. כבר ראינו כיצד ניתן לגרום לאובייקט לנוע בקו ישר ולכן לא נתעכב על זה יותר מידי, רק יש לזכור שלמחלקת vactor3 יש כמשתנה וקטור ייעודי למקרה הספציפי שלנו: Vector3.down.

בשביל לגרום לאובייקט שלנו לצוץ מלמעלה בנקודה אחרת על ציר ה-x לאחר שהוא יצא מהמסגרת של המשחק נכיר מחלקה חדשה- מחלקת Random. אומנם ל-c# יש מחלקה ייעודית להגרלת מספרים באופן רנדומלי עם אותו שם, אך החברה של unity פיתחו גם מחלקה ייעודית בכדי להקל על מפתחי המשחקים. למחלקת Random יש את המתודה range(float start ,float end)   
שמחזירה מספר מוגרל בין שני מספרים שהיא מקבלת כפרמטרים. אנחנו נשתמש במתודה כדי לקבל איזושהי נקודת x שממנה יצוץ האובייקט שלנו. איך נעשה את זה? ראשית נבדוק מה הגבולות של המסגרת שלנו לצדדים וכך נדע את טווח המספרים שממנו אנחנו יכולים להגריל. ברמת העיקרון יש לנו כבר את הטווח הזה מהסקריפט של השחקן הראשי, כאשר רצינו לתת לו טווח תזוזה לצדדים. אח"כ ניצור תנאי בפונקציה update: אם השחקן שלנו יצא מגבולות המסגרת על ציר ה-y, אז נפעיל את המתודה range ונשמור את הערך שקיבלנו במשתנה שייצג את נקודת ה-x הבאה של האובייקט, ובאותו תנאי גם נשנה את ה-position שלו גם לאותה נקודת x וגם את נקודת ה-y כפול מינוס אחד (כי הוא עובר מתחתית המסגרת לראש המסגרת). מבחינת סינטקס (\*\*\* אזהרה – בקוד שלכם אל תשתמשו ב"מספרי קסם, אלא בשדות \*\*\*):

void Update()

{

transform.Translate(Vector3.down \* \_speed \* Time.deltaTime);

if(transform.position.y<-8.0f)

{

float randomized = Random.Range(-8.0f, 8.0f);

transform.position = new Vector3(randomized, 7, 0);

}  
 }

## התנגשויות

נחזור ל-unity. אם נסתכל על חלון ה-inspector של האובייקטים שיצרנו נראה שיש להם box collider שהוא אחראי על היכולת של אובייקט להתנגש במרחב.  
אז מה זה בעצם collider? הוא האחראי על התנגשויות של גופים במרחב. הcolliders הפשוטים (שצורכים פחות זמן עיבוד) הם ה-colliders הפרימיטיביים. בd3 יש את ה-box collider ,shape collider ו- capsule collider. ב-D2 יש אפשר להשתמש ב-boxcollider2d ו- circlecollider2d.  
ישנם שני סוגים של collisions (התנגשויות): 1)hard surface collision- התנגשות בין עצמים מוחשיים במרחב, למשל כדור שפוגע בקיר, או התנגשות בין שתי מכוניות. 2) trigger collision- שנותן תחושה כאילו קרה איזשהו מאורע, למשל לקבל מטבע, לקחת חפצים מהרצפה, ובמקרה שלנו- לייזר שפוגע באובייקט אויב.  
בשביל להפעיל את היכולת להתנגשות של האובייקט שלנו נצטרך להוסיף לו רכיב rigidbody (גוף קשיח). בשביל להוסיף רכיב rigidbody נבחר בחלון ה-inspector של האובייקט שלנו ('אויב') add component-> rigidbody, וב-box collider לסמן את   
 is Trigger. נשים לב שב-rigidbody יש כמה אפשרויות לבחירה, במקרה שלנו אף אחת מהם לא ממש רלוונטי אלינו, אבל נעבור עליהם בקיצור: 1) mass – אחראי על המסה של האובייקט, יותר מסה ההתנגשות יכול לגרום ליותר נזק לגוף המתנגש.  
2)drag- משמש כדי להאט את האובייקט, כלומר כמה כוחות מושכים את האובייקט ומונעים ממנו התקדמות, ככל שה-drag גדול יותר ככה קצב ההתקדמות שלו יורד.  
3) angular drag- משמש כדי להאט את מהירות הסיבוב של אובייקט, ככל שהוא גבוהה יותר ככה מהירות הסיבוב קטנה יותר.  
4)use Gravity – האם מופעל כוח משיכה על האובייקט, כלומר האם הוא ינוע כלפי מטה. אם הוא לא מסומן, כלומר מוגדר כ-false אז האובייקט יתנהג כאילו הוא נע 'בחלל החיצון'.  
5) is kinematic- מגדיר את האובייקט האם הוא מושפע מגרומים נוספים, או רק מהסקריפט ואנימציות.  
**נסו זאת בעצמכם**, ובדקו כיצד האובייקט מתנהג בהתאם לאפשרויות השונות.  
לאחר שהוספנו את ה-rigidbody ושחקנו קצת באפשרויות השונות שלו עתה נתעסק בהתנגשות עצמה של האובייקט.  
אנחנו רוצים ברגע שהאובייקט שלנו יתנגש באובייקט אחר יקרה איזשהו מאורע, למשל כשהלייזר או השחקן הראשי פוגע באויב האויב מושמד. ל-unity יש פונקציה מיוחדת בדיוק למקרה הזה: private void onTriggerEnter(Collider other).  
המתודה פועלת באופן עצמי, כלומר לא צריך להפעיל אותה בפונקציית update() כי היא מופעלת אוטומטית במקרה של התנגשות עם גוף זר (אם מאפשרים is trigger ב- collider (box ומבצעת את הפקודה שמגדירים לה בפונקציה.  
הפונקציה מקבלת כפרמטר איזשהו collider אחר, collider יכול להיות כל אובייקט משחק אחר שמתנגש עם האובייקט שלנו  
ובלבד שנוכל זהות אותו, אבל כיצד נזהה שהאובייקט שהתנגשנו בו הוא לייזר או השחקן? בשביל זה נצטרך להשתמש בתגיות.  
נחזור ל-inspector של השחקן הראשי. אם נסתכל למעלה ,שורה מתחת לשם האובייקט, נראה שמופיע שם tag, התגיות מאפשרות לנו לזהות את שם האובייקט. הן משמשות כמו משתנה nameמסוג מחרוז לאובייקט שלנו. על מנת שנוכל להגדיר לאובייקט שלנו שם ייחודי נבחר איזשהו Tag ,במידה ואין תגית שמתאימה לנו, למשל player עבור השחקן הראשי , נוכל להוסיף תגית חדשה ב- Add tag .נוסיף תגיות לכל אחד מהאובייקטים שלנו ונחזור לקוד של האוייב.  
נרצה שבפונקציה onTriggerEnter יתרחש הדבר הבא: אם התג של הcollider שנכנס הוא כמו של הלייזר אז שיושמד האוייב:

if(other.tag=="Laser" )

{

Destroy(this.gameObject);

}

אם נריץ עכשיו את המשחק נראה שכשאנחנו פוגעים באויב הוא נעלם אבל הלייזר ממשיך לנוע קדימה, זה משום שלא הגדרנו לו collider ו-trigger. נעשה את אותו תהליך שעשינו לאויב גם ללייזר, כלומר נוסיף לו rigidbody ונסמן is trigger. ניכנס לקוד של הלייזר ונוסיף לו המתודה onTriggerEnter כך שאם הוא התנגש באובייקט עם התגית 'אויב' הוא יושמד גם:

private void OnTriggerEnter(Collider other)

{

if (other.tag == "Enemy")

{

Destroy(this.gameObject);

}

}

## מערכת חיים

חלק מהמשחק הוא גם לדעת מתי הוא אמור להיגמר. בדר"כ המשחקים בסגנון המשחק שלנו נגמרים כאשר הגענו לסוף המסלול, או כאשר השחקן הראשי מאבד את כל הנקודות חיים שלו.   
בשביל ליצור לשחקן הראשי מערכת חיים נצטרך להוסיף לו משתנה עצם חדש. נצטרך להוסיף לאובייקט גם מתודה שמורידה לו מהחיים נקודה אחת, לה נקרא כאשר תהיה התנגשות בין השחקן שלנו לאויב, לצורך העניין נקרא למתודה public void Damage():

public void Damage()

{

life--;

if(life<1)

{

Destroy(this.gameObject);

}

}

במשחק שלנו ספציפית אין צורך להוסיף לשחקן הראשי collider ניתן להשתמש ב-collider של ה'אויב'.   
בשביל להפעיל מתודה של אובייקט משחק מתוך סקריפט של אובייקט אחר אנחנו צריכים לקבל את אותו אובייקט, לunity יש מתודה מיוחדת במיוחד בשביל זה: transform.GetComponent<GameObject object Component>().  
אנחנו רוצים שבפונקציה onTriggerEnter של האויב ,מתי שקורת התנגשות בין אויב לשחקן אז תופעל המתודה Damage() של השחקן ונהרוג את האויב שלנו. לשם כך נצטרך לבקש אובייקט 'שחקן' והרכיב שלו הרלוונטי לנו (במקרה שלנו הסקריפט נשמר ב-inspector ברכיב- "Player") ולשמור אותו במשתנה מסוג שחקן, אח"כ נפעיל את המתודה שלו ונשתמש במתודה destroy() כדי להשמיד את האובייקט 'אויב':

if(other.tag=="Player")

{

Player player= other.transform.GetComponent<Player>();

player.Damage();  
 Destroy(this.gameObject);

}