# Unity - מה זה?

## מנועי משחק גרפיים (game engines) - מה הם?

מנועים גרפיים הם כלי לפיתוח תוכנה שנועדו להפחית את העלות, סיבוכיות והזמן שלוקח לפיתוח משחק וידאו.
כלי פיתוח אלו נותנים שכבה של אבסטרקציה (פישוט) מעל המשימות השכיחות ביותר בפיתוח משחקים.
מנועים גרפיים הם תועלת אדירה בכך שהם מפחיתים את כמות הידע הנדרשת על מנת לפתח משחק.
הם מורידים את הכאב ראש שבבניית מעטפת המשחק, למשל בניית לוח המשחק או אנימציית הדמויות והתאמתם לקוד המרכזי, לכדי מינימום, ובכך מאפשרים למפתחי המשחק 'להתמקד' בכתיבת הקוד של לוגיקת המשחק.
מנועים גרפיים מספקים יתרון אדיר למתכנתים שמתחיל לבנות משחק מאפס, או קבוצות שרוצות להתמקד בחוויית המשחק יותר מאשר בחלק הטכני של המשחק.

המנועיים המודרניים עושים גם עבודה טובה בחלוקת הפונקציונליות של מחלקות המשחק-תוכנית המשחק אשר אחראית על לוגיקת המשחק נשמרת בנפרד מהקוד שאחראי לנגן את קובצי ה- 'mp3.' ברקע.
רכיבים המבוססים ארכיטקטורה של מנועים גרפיים הבנויים היטב מאפשרים הרחבה שמעודדת אימוץ של משאבים חדשים, כך שהמשחק לא מוגבל רק לפלטפורמה אחת, למשל אם רוצים לשחרר משחק למחשב, טלפון נייד או קונסולת משחקים המנוע מאפשר לשנות כמה סווייצים כדי להתאים את המשחק לפלטפורמה החדשה.
כמובן שיש להיזהר כאשר רוצים לבנות משחק חוצה פלטפורמות.
למרות שקומפילציה חוצת פלטפורמות היא דבר נהדר, ועדות עד כמה רחוק הגיעה הטכנולוגיה של בניית משחקים בימינו, יש לשים לב שאם בונים משחק כזה צריך להתאים אותו גם למערכת של הפלטפורמה: לספק תמונות בגדלים שונים, ולאפשר לקוד לקבל ציודים היקפיים כמו מקלדת.
יש מנועים שהם כל כך מונחי עיצוב וויזואליות שהם מאפשרים יצירת משחק מבלי אפילו להקליד שורת קוד אחת.
ל-unity יש את היכולת להתאים אישית ממשקי משתמשים שיכולים להיות מוגדרים לשימוש גם ע"י חברי צוות שאינם מתכנתים כגון אנימטורים, מעצבים, במאים וכו' .

יש הרבה סוגים של מנועים גרפיים ואין חוקים שמגדירים איזו פונקציונליות הכרחית למנוע גרפי, אך רוב המנועים הפופולאריים

מכילים כמה מהתכונות הבאות:
*מנוע רינדור גרפי שתומך בגרפיקת D2 ו- D3
*מנוע פיזיקלי שתומך בזיהוי התנגשויות
*מנוע אודיו שמטעין ומנגן קבצי שמע
*תומך סקריפטים (קוד) ליישום לוגיקת משחק
*תפעול אנימציות - לטעינה ותצוגה של אנימציות
* multithreading - להרצה של כמה תהליכונים במקביל
*מנהל זיכרון- garbage collector, בנאי אוטומטי של אובייקטים על הערימה
*בינה מלאכותית
ועוד.

## 60 שניות על unity –

Unity הוא מנוע גרפי פופולרי מאוד שיש לו יתרונות רבים על הרבה מהמנועים הקיימים היום בשוק.
הוא מאפשר עבודה ויזואלית עם זריקה וגרירה של אובייקטים ותמיכה בכתיבת קוד בשפת c#.
למנוע תמיכה גרפית דו-ממדית ותלת-ממדית, וסט כלי עבודה לשניהם שמתפתח בצורה מתוחכמת וידידותית למשתמש עם כל גרסה חדשה שמתפרסמת.
לunity יש כמה שכבות של רישיונות והוא חינמי לפרויקטים עם הכנסות של עד 100 אלף דולר
הוא מספק תמיכה חוצת פלטפורמות לכ- 27 פלטפורמות שונות.
צוות הפיתוח של unity מציע ענן לשיתוף פעולה בין פרויקטים.
מאז הופעת הבכורה שלו ב- 2005, פותחו ב-unity אלפי משחקי desktop, טלפון נייד ,קונסולת משחקים ועוד.
בין כמה מהכותרות המפורסמות שפותחו ע"י unity ניתן למצוא:

Escape plane (2012), temple run (או בכינויו הישראלי 'טמבל רץ'), Deus Ex: The Fall (2013), wasteland2 (2014), Pokémon GO (2016), 'cuphead'(2017), ועוד טובים ורבים

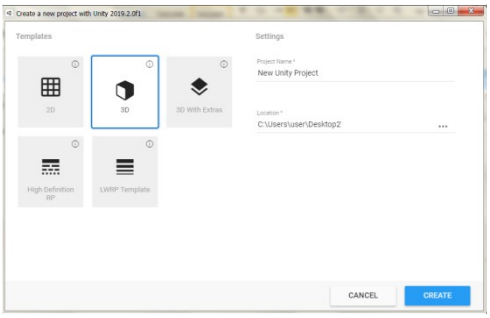
למפתחי משחקים שרוצים להתאים אישית את מהלך העבודה שלהם, unity מאפשר לערוך את העורך הוויזואלי הדיפולטיבי שלהם (default visual editor). המכניזם העוצמתי הזה מאפשר יצירת או ייבוא של כלים, עורכי טקסט ובקרים מתואמים אישית. למשל ניתן ליצור כלי ויזואלי למעצבים שבקלות מקבל ערכים מאובייקטים בתוך המשחק (כמו- hit point , טווח התקפה וכדו') למחלקה של הדמויות מבלי להצריך אותם להיכנס לקוד ולשנות ערכים או שימוש במסד נתונים חיצוני.
עוד 'נכס' (asset ) שיש ל-unity הוא השימוש ב- Unity Asset store.
ה- Asset store היא כמין חנות אינטרנטית שבהם אמנים, מפתחים ויוצרי תוכן יכולים למכור את רעיונותיהם ומוצריהם או לקנות מאחרים.
החנות מכילה אלפי הרחבות ל-unity, מודלים, סקריפטים, טקסטורות ועוד, בתשלום או בחינם שניתן להשתמש בהם כדי להאיץ את זמן הפיתוח של המשחק ולשפר את התוצר הסופי.

## היכרות ראשונית עם Unity –

אם עדיין לא הורדתם unity למחשב מומלץ להסתכל במדריך המקוצר על הורדה והתקנה של unity שנמצא במודל.

### יצירת פרויקט חדש:

במידה ועדיין לא יצרתם משתמש באתר של unity יש ליצור משתמש כדי להפעיל את המנוע הגרפי.
בכדי ליצור פרויקט חדש יש להיכנס ל unity hub ולבחור Projects מצד ימין יופיע לנו כפתור NEW נבחר את הגירסה שבה נרצה להשתמש לפרויקט החדש.
נזכיר שבמהלך הקורס אנו נשתמש בגרסה 2019.1 והלאה .
לאחר שבחרנו גרסה יקפוץ לנו חלון חדש של templates אפשריים(D2, D3...),

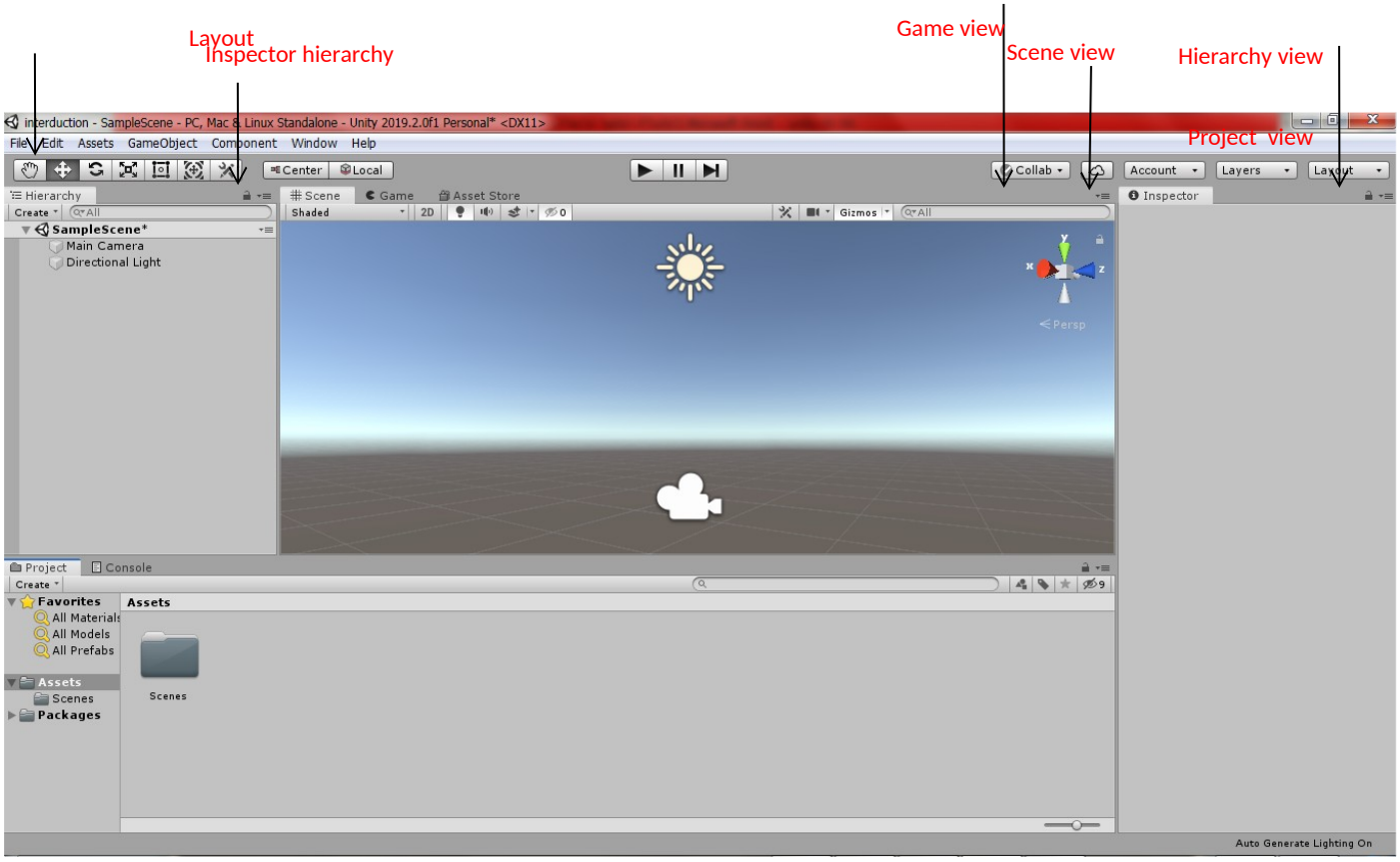


איור 1

שם הפרויקט , והיכן לשמור אותו (איור 1).
לצורך הדוגמא ניצור פרויקט D3 חדש.

## העורך הראשי (editor) :

לאחר שיצרנו את הפרויקט אמור לקפוץ לנו החלון של העורך הראשי (זה יכול לקחת זמן עד שיטען העורך), כאן הולך לקרות כל הקסם. העורך אמור להיראות כך:




Scene view

– היכן שנמקם את האובייקטים של הסצנה (סצנה יכולה להיות שלב במשחק). בחלון הזה נציג את המפה של אותו סצנה במשחק- איפה יהיו הדמויות, איך יראה הרקע של המשחק, חפצים שיש בשטח וכדו'.
באמצעות לחצן ימני בעכבר אנחנו יכולים לטייל בסביבה וע"י הכפתור F אפשר להתמקד באובייקט מהסצנה שנבחר.

Game view- איך תראה הסצנה בזמן אמת, כלומר איך היא תראה כאשר נפעיל את המשחק.

Hierarchy view- ההיררכיה של האובייקטים בפרויקט- נוכל לעבור בין האובייקטים השונים (דמויות, חפצים, מצלמה, תאורה וכדו') שיש באותה סצנה.

Inspector hierarchy- נותן לנו מידע על כל אובייקט שבחרנו, ומאפשר לנו לשנות את אותו (למשל מיקום, סיבוביות וכדו').

Layout – פריסת עמוד, איך שתראה הסביבת עבודה שלנו.

Project view- מכיל את כל ה- assets שיש לנו בפרויקט- חומרים, דמויות, חפצים וכדו'.

# אובייקט משחק

יוניטי היא סביבה מונחת-עצמים. המושג העיקרי ביוניטי הוא אובייקט-משחק – **GameObject**.

אובייקט-משחק יכול להיות דמות הפועלת במשחק, כגון מכונית, חללית, מפלצת וכדו'.

אבל יותר מזה, רוב האלמנטים של unity הם אובייקטים שבאים עם העורך הראשי, למשל מצלמה או תאורה.

יש אובייקטים בסיסיים שבאים עם יוניטי, למשל קובייה, צילינדר, קפסולה וכו', ואנחנו יכולים להרכיב אותם לאובייקטים מורכבים יותר, או ליצור חדשים בעצמנו.

לכל אובייקט יש אחד או יותר **רכיבים** (**Component**): כל רכיב אחראי לחלק מההתנהגות של אותו אובייקט.

בסי שארפ, כל רכיב ממומש כמחלקה.

על מנת ליצור אובייקט חדש יש ללחוץ על GameObject שנמצא בתפריט הראשי, ולבחור בסוג האובייקט שאותו נרצה.

## יצירת אובייקט ריק

לצורך הדוגמה, ניצור קודם-כל אובייקט ריק – Create Empty. אנחנו רואים שגם לאובייקט ריק יש רכיב אחד והוא ה-**Transform**. זה רכיב שיש אוטומטית לכל אובייקט במשחק, והוא קובע את המיקום שלו ב-3 ממדים, הסיבוב שלו ב-3 ממדים, וההגדלה שלו (scaling) ב-3 ממדים.

כרגע, אנחנו בכלל לא רואים את האובייקט שלנו במשחק. מדוע? כי אין שום רכיב שמצייר אותו. הרכיב שמצייר אובייקט נקרא Renderer (המילה render משמעותה – לצייר אובייקט על המסך). אז כדי שנראה את הרכיב שיצרנו, צריך להוסיף לו renderer. איך נעשה את זה? בצד ימין, בחלון הinspector, נלחץ על "הוספת רכיב חדש", נתחיל לכתוב renderer ונראה כמה אפשרויות.

נבחר באפשרות sprite renderer – צייר ע"י "ספרייט" (תמונה). אנחנו רואים שלרכיב יש כמה שדות, ואחד מהם הוא התמונה עצמה. בשלב ראשון השדה הזה ריק – אין בו שום ערך – ולכן עדיין לא רואים את הרכיב. כדי שנוכל לראות אותו, נקח תמונה כלשהי מהמחשב שלנו (או נוריד מהאינטרנט – ניתן למצוא תמונות חינמיות במנוע החיפוש Creative Commons Search), ונשים אותה בתיקיית assets. עכשיו אפשר לגרור את קובץ התמונה מתיקיית Assets על הרכיב, או ללחוץ על העיגולון הקטן ליד השדה של ה sprite ולבחור את קובץ התמונה.

עכשיו סוף סוף רואים את האובייקט שלנו על המסך!

## יצירת קוביה

לצורך דוגמא נוספת, ניצור אובייקט מסוג קובייה (מהתפריט הראשי: GameObject -> 3D -> Cube). כדי שיהיה לנו קל לזהות אותה ניתן לה שם. שימו לב שלקוביה כבר יש Renderer (שנקרא Mesh Renderer), ולכן כבר רואים אותה על המסך.

הדבר הבא שוודאי נרצה לדעת הוא כיצד ניתן ליצור מניפולציות על האובייקט, כלומר לשנות את הגודל של האובייקט, מיקום סיבוב וכדו'. יש שתי דרכים עיקריות ליצירת מניפולציות על אובייקטים:

1) בחלון של האינספקטור (Inspector hierarchy) יש לנו אזור שמיועד למניפולציות על האובייקט עליו אנו עובדים-Transform.

ב- Transform יש לנו את האפשרות לשנות את המיקום של האובייקט יחסית לציר ה-x, y ו-z של המסך הראשי. מתחת לזה יש את האפשרות לסובב את האובייקט, ומתחת לזה ניתן לשנות את גודל האובייקט בהתאם לצירים.

כמובן ששימוש ב transform הוא הרבה פחות אינטואיטיבי, נשתמש בו בעיקר כדי להשוות גדלים/מיקום של אובייקטים, או , כפי שנראה בהמשך, לגרום לאובייקט לנוע בקו אחיד.

2) דרך שניה ליצור מניפולציות על אובייקטים היא ע"י המניפולטורים המופיעים בצד שמאל למעלה:

Hand tool - מזיז את המצלמה בגרירה מהאובייקט שיצרנו.


Move tool - מאפשר להניע את האובייקט לפי צירי ה- x, y ו- z


Rotate tool – מאפשר לסובב את האובייקט.


Scale tool - מאפשר למתוח את האובייקט, המתיחה היא לפי הצירים אך דו כיוונית, כלומר גם לאזור החיוביים וגם השליליים של הצירים בו זמנית.


Rect tool – מתיחה ריבועית. מאפשר למתוח את האובייקט לכיוון ספציפי.


All in one- מאפשר לבצע את כל המניפולציות בו זמנית- מתיחה, סיבוב והזזה.

טיפ: במקלדת הכפתורים Y-Q משמים כקיצורי דרך למניפולטורים השונים בהתאמה.

## יצירת חומרים לאובייקט

כל אובייקט חדש שיוצרים עשוי בד"כ מחומר דיפולטיבי. אם נרצה ליצור לאובייקט שלנו חומר או שנרצה שלפחות יהיה לצבע שונה מהאפור שמגיע כברירת מחדל, נצטרך ליצור אובייקט מסוג חומר (material) שיתאים לאובייקט עליו הוא יולבש.

נהוג ליצור תיקייה של 'חומרים' בחלון של הפרויקט (Project view) וכל חומר חדש לקרוא לו כשם האובייקט שלו, מקף תחתון והמילה mat(קיצור של material),למשל: player_mat.
בשביל ליצור תיקייה יש ללחוץ מקש ימני בחלון ה- Project view-> create -> folder, לאחר שקראנו לה בשם נרצה להוסיף לה חומרים חדשים לרשימה, לשם כך יש ללחוץ בתיקיה מקש ימני -> material -> create.
ובלחיצה על האובייקט חומר ניתן לערוך את החומר בחלון האינספקטור מצד ימין.
אחד היתרונות הגדולים של המנוע הגרפי unity הוא היכולת להזיז ולהתאים דברים באמצעות גרירה, כך שאם יצרנו את החומר בטעות מחוץ לתיקיה ניתן לגרור אותו לתיקייה הייעודית לחומרים, ואם נרצה להתאים בין האובייקט לחומר כל מה שנצטרך זה לגרור את החומר לתוך האובייקט- או לתוך האינספקטור של האובייקט, או לתוך האייקון של האובייקט בחלון ההיררכיה ( Hierarchy view).

**שימו לב!** החומר (Material) הוא למעשה שדה של רכיב ה- Renderer (למרות שבחלון הinspector הוא מוצג בנפרד). לכל אובייקט שיש לו רכיב Renderer, אפשר "להלביש" חומר.

# יצירת סקריפט

בכל משחק הדבר החשוב ביותר, לפני עיצוב השחקנים, התפאורה או מוזיקת רקע הוא יצירת מנגנון שינהיג את המשחק, כלומר מנגנון שיתן למשחק חוקיות: הדמות הזאת עושה ככה, כשמדברים איתו קורה ככה וככה, מתי נגמר המשחק, איך הוא מתחיל וכדו'. כאן בדיוק נכנס התפקיד של הקוד.

בעזרת תיכנות בסי שארפ, אנחנו יכולים ליצור סוגים חדשים של רכיבים (Component), ולהלביש אותם על העצמים שלנו, וכך לגרום להם להתנהג כמו שאנחנו רוצים.

למען הסדר הטוב נהוג ליצור תיקיה ייעודית לכל הסקריפטים של הסצנה, ולכל סקריפט לקרוא על שם האובייקט עליו הוא פועל. בשביל ליצור את הסקריפט יש ללחוץ על התיקייה הייעודית לסקריפטים מקש ימני-> create-> c# Script . בדיוק כמו ביצירת

חומר לאובייקט גם כאן נלביש את הסקריפט לאובייקט ע"י גרירה. אם הגרירה עבדה כמו שצריך אמור להופיע הסקריפט בחלון האינספקטור של האובייקט, ובכדי לערוך את הקוד נצטרך ללחוץ מקש ימני על האייקון של הסקריפט בחלון האינספקטור-> Edit Script.

אם התקנתם את visual studio כמו שצריך זה אמור לפתוח לכם את התוכנה עם הקוד של הסקריפט.

שימו לב: כדי ליצור רכיב חדש, הקוד של הסקריפט אמור להיות בנוי כך:

- מחלקה עם שם הסקריפט שהיא יורשת ממחלקת MonoBehaviour (שהיא מחלקת האב לכל הרכיבים של unity)
- למחלקה שתי מתודות: void start()- מתודה שנקראת ראשונה עם אתחול האובייקט. המתודה נקראת פעם אחת בסקריפט;
- ו- void update() – בכל פריים (עדכון של המסך) נקראת הפונקציה והיא מייצגת את השינויים שמתחוללים באובייקט וכיצד הם נראים על המסך. למשל בשביל להזיז אובייקט במסך נצטרך לעדכן את המיקום שלו (ה- transfom) בפונקציה, ואז ניתן יהיה לראות את השינוי על המסך.

בהמשך הקורס נלמד בהרחבה על בניית רכיבים ותיכנותם באמצעות סקריפטים.

## Layout-

פריסת עמוד: כמו כמעט בכל תוכנת עיצוב תלת ממדית ניתן לשנות את פריסת העמוד הדיפולטיבית למשהו שיהיה לנו יותר קל לעבוד איתו. למשל במקום שהטאב של הסצנה (scene) view והטאב של המשחק (game view) יהיו נפרדים, נרצה שיהיו לנו שני חלונות אחד ליד השני של המשחק ושל עיצוב הסצנה, וכך לא נצטרך לדלג בניהם כל פעם שנשנה קצת את התוכנית.
ע"י גרירה של טאב המשחק לתחתית המסך ניתן ליצור מצב של שתי חלונות נפרדים.
במידה ואנחנו לא מרוצים מהשינוי שעשינו, ואנחנו רוצים לחזור לתצוגה המקורית שקיבלנו ניתן לעשו זאת ע"י לחיצה על layout -> Default .
המלצה לפריסת עמוד: נלחץ על הכפתור ה- layout ונבחר במקום ב default ב- tall, אח"כ נגרור את ה- game view שיהיה מתחת ל- scene view. עוד שיפור שניתן להוסיף לעמוד הוא האפשרות שב-project view נוכל לראות יותר מרק אובייקטים של תיקייה אחת כל פעם: נלחץ מקש ימני על הטאב של ה- one column layout -> project זה יהפוך את התצוגה לתצוגת רשימה.
בכדי לשמור על ה-layout שעשינו נלחץ על החלון ה save layout –> layout.
אנחנו אמורים לקבל תצוגה כזאת: