המנוע הפיזיקלי של unity

על מנת שתהיה התנהגות פיזיקלית משכנעת, אובייקט במשחק צריך להאיץ בצורה נכונה ולהיות מושפע מהתנגשויות, גרביטציה וכוחות נוספים. המנוע הפיזיקלי של unity, המגיע עם התקנת התוכנה, מספק רכיבים המשמשים להדמיה הפיזיקלית בשבילנו.

בעזרת הגדרת כמה פרמטרים פשוטים, נוכל ליצור אובייקטים המתנהגים בצורה פסיבית בדרך ראליסטית, כלומר הם יזוזו כתוצאה מהתנגשויות ויפלו אך לא ינועו בכוחות עצמם.

באמצעות שליטה על הפיזיקליות ע"י סקריפטים, נוכל לתת לאובייקטים דינמיקה של רכבים, מכונות או אפילו חתיכות בד.

במהלך הפרק הקרוב נעשה סקירה קצרה על היכולות של המנוע הפיזיקלי של unity. נגדיר התנהגויות פיזיקליות מחיי היום יום ונראה כיצד נוכל לבטא אותם באמצעות כמה שורות קוד פשוטות או בהגדרת כמה פרמטרים באינספקטור.

כמו כן נפגוש כמה סוגי משחקים (חלקם אף מוכרים), וכיצד המנוע הפיזיקלי בא לידי ביטוי באותם משחקים.

המערכת הפיזיקלית של unity

כשאנחנו חושבים על משחקים מבוססי פיזיקה, לרוב אנחנו חושבים על התנהגויות בין גופים קשיחים שונים - חישוב וביצוע של אינטראקציות ראליסטיות בין קבוצות עצמים. שלושת ההיבטים המרכזיים של מערכת פיזיקלית הם:

- אינטגרציה - איך הפיזיקה מתאימה לעולם המשחק.
- זיהוי חפיפות (Collusion detection) - אילו עצמים בעולם המשחק חופפים זה לזה, ואיך המערכת מבחינה בכך.
- תגובה לחפיפות (collusion reaction) – איך עצמים החופפים זה לזה מגיבים באופן ריאליסטי, למשל כדור גומי שמתנגש בקיר וקופץ.

כפי שכבר יצא לנו לראות בשיעורים הקודמים, השימוש ברכיבים פיזיקאליים ב-unity הוא אופציונלי. אם נרצה להוסיף לאובייקט איזושהי התנהגות פיזיקאלית, נצטרך להוסיף לו את הרכיבים המתאימים לכך – בפרט Rigidbody, מסוג dynamic, וכן collider.

למעשה לunity יש שתי מערכות פיזיקליות שונות: אחת המשמשת למודלים בתלת-ממד, ואחת המשמשת למודלים בדו-ממד. הרעיון המרכזי די זהה בין שתי המערכות השונות (למעט הממד הנוסף שיש ב3D), אך הם מיושמים ע"י רכיבים שונים. כבר יצא לנו לראות בשיעורים הקודמים את ההבדל בין הרכיבים, למשל עבור תלת ממד יש לנו את ה-Rigidbody ואילו בדו ממד נשתמש ברכיב Rigidbody2D בשביל לדמות גוף קשיח.

קצת רקע על המערכת הפיזיקלית של unity

עד לאחרונה השימוש במערכת הפיזיקאלית של unity נעשה באמצעות הספרייה physx, ספריית open source מתפתח שנותנת מענה פיזיקלי להרבה מאוד פלטפורמות שונות.
השימוש בספרייה היה לצורך משחקי תלת-ממד, עבור דו-ממד יש את הספרייה Box2D.
היתרונות הבולטים של השימוש בספריות אלו היה לא רק בשל היותם open source או היכולת להתאים אותן להרבה פלטפורמות, כי אם ניהול הזיכרון המעולה ויכולות ריבוי-תהליכים שלהן.

נכון ל-2019, יוניטי הכריזה בוועידה לפיתוח משחקים (GDC) כי היא תיתן את האפשרות לבחור בין שתי פלטפורמות להדמיית התנהגות פיזיקלית- "unity physics" ו- "Havok" המבוססות על המערכות הפיסיקליות הישנות.

חומרים - Materials

בשיעור הראשון כבר למדנו על חומר – **Material** – שאפשר להוסיף לגוף כדי לקבוע את הצבע שלו ואת המראה שלו בתנאי-תאורה שונים (עד כמה הוא מבריק וכד').

ביוניטי ישנם עוד שני סוגים של חומרים הקשורים לפיסיקה, וקובעים את התכונות הפיסיקליות של הגוף: יש **PhysicsMaterial2D** העובד עם המנוע הפיסיקלי הדו-ממדי, ויש **PhysicsMaterial** העובד עם המנוע הפיסיקלי התלת-ממדי.

כדי ליצור חומר חדש נבחר Asset -> Create -> Physic Material או Physics Material 2D. בחומרים נוכל להגדיר תכונות כגון רמת חיכוך (האם הוא מחליק על המשטח כמו קוביית קרח), ורמת קפיציות של החומר, למשל עבור אובייקט סטטי כמו אבן נעדיף להגדיר רמת קפיציות נמוכה, ועבור אובייקטים עשויים גומי נעדיף רמת קפיציות גבוהה.

Player setting

אם לא יצא לנו להכיר עדיין, Unity Project Setting הוא החלון בו ניתן לשנוי הגדרות מתקדמות של המשחק - רזולוציות, איכות סאונד, הגדרת קלטים, פיזיקה בדו ובתלת ממד ועוד.
ההגדרות כאן הן דיפולטיביות, כלומר אם נשנה אותן נשנה לאורך כל המשחק.
נפתח את החלון ע"י: Edit -> Project setting. שימו לב שיש שני חלונות שונים לשני המנועים הפיסיקליים – דו-ממדי ותלת-ממדי. אנחנו נתמקד בכמה שדות-מפתח מתוך המאפיינים הפיזיקאליים:

Gravity- לפני שנמשיך קצת רקע על גרביטציה:
בהגדרה הגרביטציה (או כוח הכבידה), היא אחת הכוחות הבסיסים בטבע.
כוח משיכה פועל בין גופים בהתאם למכפלת המסות שלהם ומרחקם אחד מהשני המונח "כוח הכובד" מתייחס בדרך כלל לכוח בו כדור הארץ מושך אליו עצמים על פני שטחו) [ויקיפדיה].
גופים קטנים מושפעים ומשפיעים על גופים אחרים, לדוגמא בני אדם הם בעלי מאסות קטנות ולכן ההשפעה תהיה מאד קטנה על הגופים האחרים עד כדי כך שההשפעה זניחה. לעומת זאת, בנושא כדור הארץ והירח האפקט ברור לעין- הירח מקיף את כדור הארץ בגלל השפעת כוח הכבידה. בכדור הארץ קיימות תופעות כמו גאות ושפל הנגרמות מהשפעת כוח הכבידה של הירח.
כאשר אנחנו מדברים על כוח המשיכה או כוח g (על כדור הארץ) של כוכב, אנחנו בעצם מדברים על כוח תאוצה שקול שהוא שילוב של המשיכה מהמסה של כדור הארץ והכוח הצנטריפוגלי (כח תנועה סיבובית) הנוצר מסיבוב הכוכב סביב עצמו.
תאוצה זו נמדדת ביחידות של מטר לשנייה בריבוע (m/s^2).
קרוב לפני שטח כדור הארץ תאוצת הכבידה היא בערך $9.81\ m/s^2$, למעט אזורים כמו מפרץ הדסון בקנדה (אם אתם מתקשים לעמוד בדיאטה כדאי שתשקלו פשוט לעבור למפרץ הדסון, שם תשקלו פחות).

עד כאן הרקע וחזרה לעניינו: אם נסתכל על אחד מההגדרות שקשורות בפיזיקה של המשחק (Physics או Physics2D) נראה שם שדה שקוראים לו Gravity, וכשמו כן הוא- מנסה לדמות את הגרביטציה (או, יותר נכון, כוח המשיכה) על כדור הארץ. הכוח מסומן על פני ציר ה-y כ 9.81- (הוא מתבטא כנפילה "חופשית"). כמובן שהכוח הוא אופציונלי, למשל אם נרצה ליצור משחק שמתרחש במאדים או סתם סימולציה של הכוכב נוכל לשנות את אותו כוח ל -3.711 . אפשר גם ליצור כוח כבידה בציר אחר, למשל אם נרצה שחפצים יפלו הצידה במקום למטה ניצור כוח כבידה בציר x.

Default Material- ניתן להגדיר את סוג חומר שיתווסף לכל אובייקט חדש ששמנו לו Rigidbody כלשהו.

Layer Collision Matrix- שימו לב שבאותו החלון, בלשונית של אחד ה-physics, למטה יש לנו כמין פירמידה כזאת של שכבות. הפירמידה מסמלות אילו שכבות יכולות להתנגש אחת עם השניה. עצם שנמצא בשכבה א יכול להתנגש עם עצם שנמצא בשכבה ב, אם ורק אם יש V במקום המתאים בטבלה. אם אין V, אז העצמים בשכבות אלו פשוט יחלפו אחד על-פני השני בלי להתנגש ובלי ליצור שום תגובה פיסיקלית.

רכיבי פיסיקה דו-ממדיים - תזכורת

כבר יצא לנו להכיר חלק מהרכיבים, אך בכל זאת נעשה סקירה כללית ליישור קו:

Transform

הרכיב הבסיסי ביותר של עצמים מסוג MonoBehavior. הוא מגיע אוטומטית ביצירת אובייקט משחק חדש. אחראי למיקום, סיבוב, וגודל.

RigidBody2D

רכיב המגדיר את העצם כגוף קשיח, ובכך מאפשר לו התנהגות פיזיקלית מסוימת.
הרכיב לא מגיע עם העצם, ויש להוסיף אותו ב-Add component ולחפש בשורת החיפוש Rigidbody2D.
כברירת מחדל האובייקט מסומן עם גרביטציה.
נתמקד על כמה שדות עיקריים ברכיב:
* ***Body type-*** ביוניטי קיימים שלושה סוגים של גופים קשיחים:
1. Dynamic- הנפוץ מבין השלושה. גוף שיש לו מיקום ומהירות, ופועלים עליו כוחות פיסיקליים כגון כבידה.
2. Kinematic- גוף שאמור לזוז אך לא מושפע מכוחות אחרים. יכול להתנגש רק באובייקטים שמוגדרים כדינאמיים.
3. Static- גוף שאמור להישאר במקום אחד כמו קיר. מתאים לייצוג או אובייקטים הקשורים לרקע. לא מגיב לגרביטציה, לא אמור לשנות את מיקומו בצורה ישירה, ואם לא מגדירים לו אחרת, לא אמור להתנגש באובייקטים שאינם דינאמיים.

****Material-*** החומר שממנו עשוי העצם. קובע את רמת החיכוך של העצם ואת רמת הקפיציות שלו בזמן התנגשות, כפי שתיארנו למעלה.

****Drag-*** כוח ה"גרר" הפועל על העצם. גרר הוא הנטייה של עצם להאט כתוצאה מחיכוך עם האוויר או נוזל כלשהו שסובב לו. ישנם שני סוגים של גרר:
1. Angular Drag - גרר זויתי - מייצג את היכולת לסובב את האובייקט. ככל שהוא יותר גבוה ככה קשה יותר לסובב אותו.
2. Linear Drag- גרר ליניארי - משפיע על מיקום האובייקט, ככל שהוא יותר גבוה הוא מתאר אובייקט שקשה יותר להזיז אותו.

****Mass-*** *המסה* של האובייקט. ככל שהמסה גדולה יותר, כך דרוש כוח חזק יותר כדי להאיץ את האובייקט. *הערה*: אין זה אומר שהוא נופל מהר יותר מאובייקט עם מסה פחותה – כל החפצים נופלים באותה תאוצה בדיוק! (ראו ניסוי גלילאו).

Collider2D

רכיב שמגדיר את קווי המתאר של האובייקט לצורך התנגשות, כלומר הוא מסמן את הגבולות שבהם האובייקט ירגיש את ההתנגשות. אם יש לאובייקט Collider ניתן להגדיר לו לוגיקה ספציפית בזמן התנגשות עם אובייקט אחר ע"י מתודות ייעודיות של הרכיב.
הערה: לכל אובייקט יכולים להיות כמה colliders – יכול להיות שהאובייקט בנוי מכמה תתי-אובייקטים שלכל אחד יש הגדרה אחרת להתנגשות. זאת גם אחת הסיבות שיש לנו material ב-collider על אף שכבר יש לנו ב-rigidbody, כדי שנוכל לתת לכל collider חומר אחר אם נרצה.

IsTrigger - לפעמים נרצה לדמות התנגשות עם אובייקט שאינו דינאמי לכן נשתמש בטריגרים שידמו לנו כמין שדה בלתי נראה שמעורר התנהגות כלשהי. כאשר אנחנו מפעילים את הפונקציות הקשורות בטריגר, אנחנו בעצם מגדירים לוגיקה מבלי להתייחס לתוצאות ההתנגשות (אם יש כאלו, יכול להיות שאובייקט שאינו דינאמי התנגש בטריגר ועדיין יופעל הטריגר).

מקורות

- סרטונים ביוטיוב

סיכם: מעוז גרוסמן