 ***GitHub and Unity***

## GitHub Desktop

תוכנה המאפשרת לנו לנהל את חשבון הגיטהאב שלנו משולחן העבודה. היא מקלה עלינו בעיקר בלשלוח מסמכים לאתר לא ע"י שימוש בחלון הפקודה של GitHub, שלא נראה הכי ידידותי בהתחלה. התוכנה מספקת ממשק משתמש נח ומובן יחסית להעלאת פרויקטים, החלפת ענפים, והחלפת מסמכים עדכניים בcommits ישנים.   
למען הנוחות אנחנו נשתמש ב- GitHub Desktop, על אף שקיימות דרכים אחרות לקשר בין unity ל-GitHub משום ש:  
א. קיימים הרבה מדריכים שימושיים לGitHub desktop   
ב. רוב ה-assets של unity שקשורים לגיטהאב לא הכי עדכניים וגורמים לבעיות בזמן העברה של הנתונים.  
ג. הכי קרוב לממשק משתמש של האתר.   
אז לפני שנמשיך במדריך ,אני ממליץ למי שאין עדיין את ה-GitHub desktop להוריד אותו מהאתר: <https://desktop.github.com/>

## קבצי הגדרות

אחרי שיצרתם משחק ביוניטי, המשחק שלכם יושב על המחשב ועדיין לא הגיע לגיטהאב. לפני שנעלה אותו לשם, חשוב שנוסיף לו שני קבצים:

1. **gitignore – להתעלמות מקבצים מיותרים**: רוב הקבצים בתיקיה של יוניטי הם קבצים זמניים שהעורך יוצר. אין טעם להעלות את כולם לגיטהאב – צריך להעלות רק את קבצי-המקור. כדי לעשות זאת, צריך לשים בתיקיה הראשית של המשחק שלכם קובץ בשם "**.gitignore**" שימו לב לשם הקובץ – שם הקובץ מתחיל בנקודה ואחריה “gitignore” בלי סיומת. בקובץ הזה צריך לכתוב את כל הקבצים והתיקיות שיש להתעלם מהם.
2. **gitattributes - לניהול קבצים גדולים**: קבצי קול, תמונה ואנימציה הם בדרך-כלל גדולים ותופסים הרבה מקום על השרת של גיטהאב. ניתן להגיד לגיטהאב לשמור אותם בשרת מיוחד יעיל יותר, שנקרא LFS – Large File Storage. כדי לעשות זאת צריך לשים בתיקיה הראשית של המשחק שלכם קובץ בשם "**.gitattributes**" שימו לב לשם הקובץ – שם הקובץ מתחיל בנקודה ואחריה “gitattributes” בלי סיומת.

ניתן להוריד את שני הקבצים מאתר הקורס:

https://github.com/gamedev-at-ariel/gamedev-5781/blob/master/02-unity-basics/gitignore-and-gitattributes.zip

להוריד את הזיפ, ולהעתיק את שני הקבצים לתיקיה של המשחק על המחשב שלכם (ליד התיקיות Asset, ProjectSettings וכו').

## יצירת מאגר גיט

פיתחו את גיטהאב דסקטופ, וגררו את התיקיה עם המשחק לכם לתוך התוכנה. ייפתח לכם חלון שאומר לכם שהתיקיה הזאת לא נראית כמו מאגר של גיט, ומציע לכם ללחוץ על קישור כדי ליצור מאגר חדש של גיט – לחצו על הקישור.

החלון שייפתח ישאל אתכם אם להוסיף קובץ רידמי README - בחרו "כן". אחר-כך החלון ישאל אתכם אם להוסיף קובץ gitignore - בחרו "לא" כי כבר הוספתם אותו ידנית יחד עם ה gitattributes.

## פירסום המאגר לגיטהאב

עכשיו המאגר שלכם מוכן להעלאה. בתוך התוכנה github desktop, לחצו על הכפתור Publish to github.

בחלון שייפתח, יש לשים לב לשני דברים:

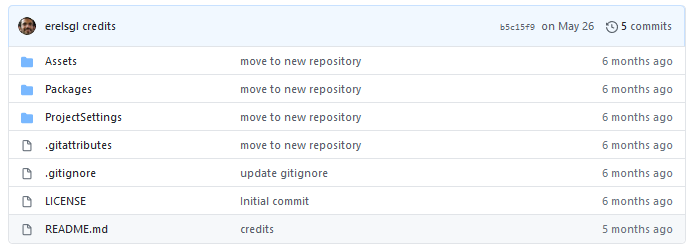
1. למחוק את הסימון ליד Private – כדי שהמאגר ייחשב לציבורי ונוכל לראות אותו;

2. לבחור את הארגון (organization) שלכם – שיצרתם בהתחלה – כדי שהמשחק ייכנס לשם.

## בדיקה

**בדיקה**: אחרי שדחפתם את המשחק לגיטהאב, היכנסו לחשבון שלכם, לחצו על ה"ארגון" שיצרתם, וודאו שהמאגר שלכם נמצא שם (אם בטעות העליתם אותו לחשבון הפרטי שלכם – אפשר להעביר אותו לחשבון הארגון דרך Settings).

חשוב לוודא שהמאגר שלכם נראה בערך כך:

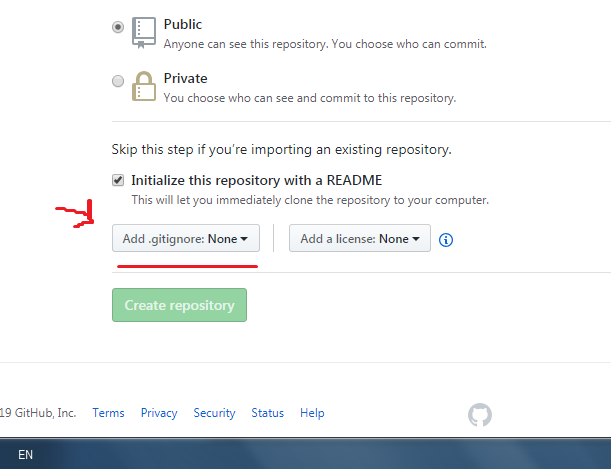
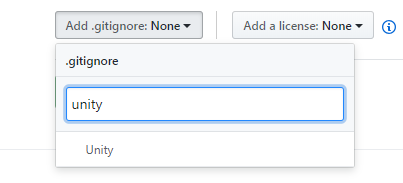


(כמו המאגר בקישור זה: <https://github.com/gamedev-at-ariel/01-unity-basics>)

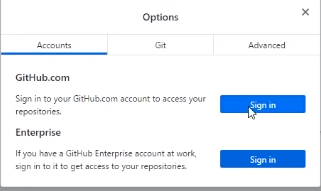
כלומר הוא מכיל רק שלוש תיקיות (Assets, Packages, ProjectSettings) ושלושה-ארבעה קבצים (.gitattributes, .gitignore, Readme.md, LICENSE). אם אתם רואים שם תיקיות נוספות, למשל Library, סימן שטעיתם בשלב כלשהו בדרך. למרבה הצער, כשקבצים גדולים נכנסים למאגר פעם אחת, הם לא יוצאים משם גם כשמוחקים אותם – הם נשארים לנצח בהסטוריה של המאגר. לכן אם טעיתם תצטרכו לפתוח מאגר חדש ולחזור על התהליך.

## תהליך חלופי / מעוז

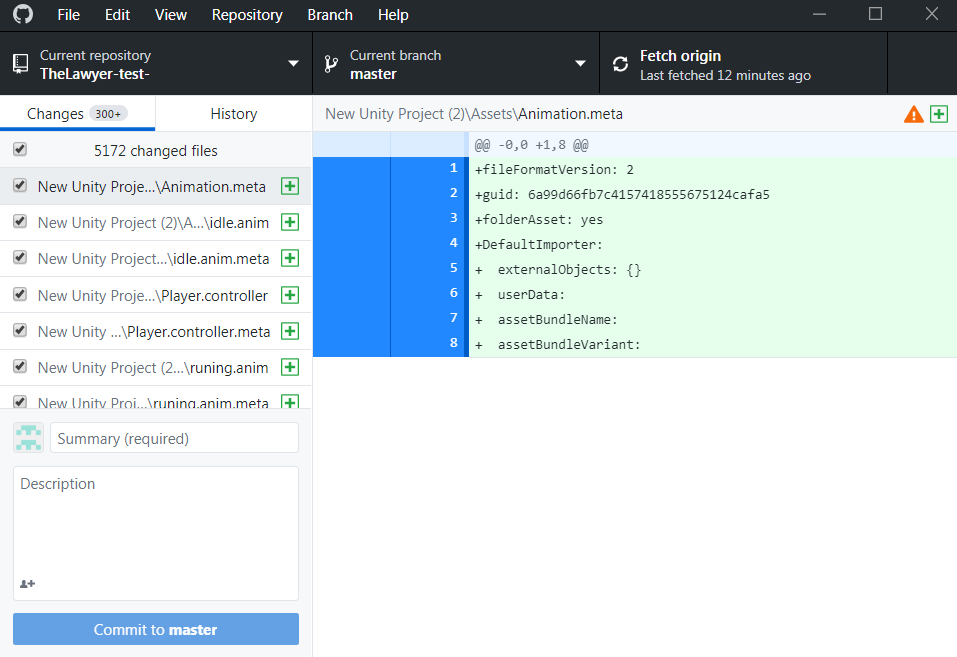
ניצור repository חדש בגיטהאב.  
ניתן לו שם (אם רוצים גם תיאור), ונחליט אם נרצה שהפרויקט יהיה ציבורי או פרטי.  
נלך לתחתית העמוד, נראה שיש Add.gitignore , gitignore הוא למעשה קובץ שאנחנו מגדירים מראש שנותן לנו את האפשרות לסנן בהעלאה קבצים שלא נצרכים לנו. מסתבר שלגיטהאב יש גם gitignore מיוחד בשביל unity, שמסנן קבצי מטה-דאטה שהמנוע מייצר אבל לא ממש מועילים למשהו.   
נבחר ב- Add.gitignore ובשורת חיפוש נכתוב unity:

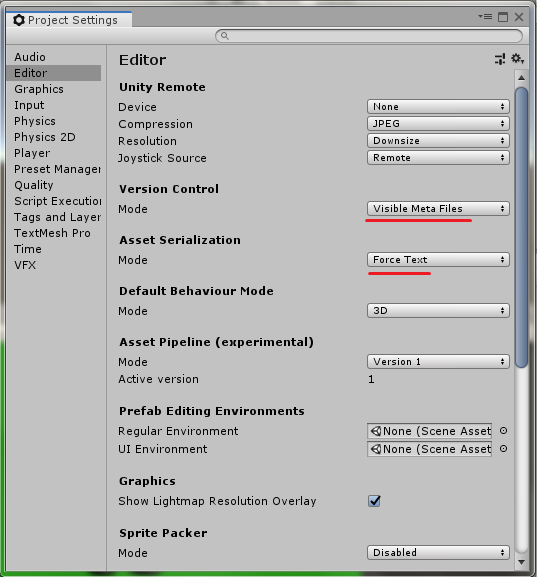


נפתח את ה GitHub Desktop . בפעם הראשונה שניכנס אליו, אם לא הגדרנו עדיין חשבון גיטהאב, נלך ל-file->option וליד ה-GitHib.com נבחר sign in , ושם נזין את השם משתמש והסיסמא שלנו בגיטהאב כדי להתחבר:



כדי לייבא repository מהאתר נבחר באופציה clone a repository , ונכניס את ה-url של אותו repository.  
נבחר היכן אנחנו רוצים שזה ישב על המחשב שלנו(באיזו תיקייה), ו-clone.  
ניצור פרויקט חדש ב-unity, אם נשים לב מתחת לבחירת השם לפרויקט ניתן לבחור באיזו תיקייה במחשב נרצה לשים את המשחק שלנו. נבחר את אותה תיקייה של ה-repository שהרגע ייבאנו דרך ה-GitHub Desktop .  
במידה ואנחנו רוצים לשמור פרויקט שכבר התחלנו לעבוד עליו נעתיק את התיקייה של הפרויקט unity לתיקיה של ה-repository , וב-unity hub כנראה לא יפתח לנו הפרויקט אם נלחץ עליו (כי שינינו את המיקום שלו), לכן נבחר ב-Add -> התיקייה של הפרויקט unity שהעברנו.   
אם נחזור לgithub desktop נראה שהוא מראה לנו את כל השינויים שחלו בתיקייה של הrepository מאז ה-commit האחרון שעשינו:



לפני שנעשה commit אנחנו צריכים להגדיר כמה דברים ב-unity :  
נחזור ל-unity ונבחר Edit-> project setting -> Editor וב- editor נדאג שה-version control יהיה על visible meta file.  
לכל asset שאנחנו יוצרים בפרויקט, unity מייצר meta file שמכיל מידע על אותו אובייקט שממנו הוא נוצר ומתי משתמשים בו בפרויקט, אנחנו רוצים שיהיה ניתן לראות את ה-meta files כדי שהם יבחרו ע"י ה-version control.   
אם נרצה להפוך את הקבצים של unity גם ליותר קראים נבחר Asset serialization -> force text:

אחרי שעשינו את זה נוכל לחזור לGitHub desktop ולעשות commit.  
הcommit שעשינו מעודכן רק בgit- המקומי שלנו, אבל לא באתר. כדי לעלות אותו לאתר נצטרך לעשות גם push to origin . עכשיו כל פעם שנעשה איזשהו שינוי במשחק הוא ישר יתעדכן בתיקייה של ה-repository ונוכל לעשות לו commit ו-push . במידה ועבדנו על הפרויקט ממחשב אחר, או שאנחנו עובדים עם כמה אנשים ואנחנו רוצים לעדכן את התיקייה במחשב שלנו עם ה-commit האחרון שהתבצע נלחץ על Fetch origin, שנמצא בדיוק היכן שהכפתור של ה ,pushואז נלחץ pull (או ctrl+shift+P שעושה pulling אוטומטית).