



GitHub and Unity

תזכורת - מה זה בכלל גיטהאב?

בשביל אלו שעדיין לא יצא להם, או ששכחו בינתיים כמה מהפיצ'רים של גיטהאב, להלן סקירה מקוצרת:

גיטהאב היא מערכת אינטרנטית לניהול גרסאות- כל מסמך שאנחנו מעלים לאתר, אם העלנו אותו כמה פעמים אנחנו יכולים לברור מבין העלאות שהיו לנו (או העלאה האחרונה) איזו גרסה של המסמך אנחנו רוצים.

במה זה עוזר לנו? נניח יש לנו פרויקט גדול הבנוי ממלא מסמכים, סקריפטים וכדו', ואנחנו במקרה עשינו טעות באחד הקבצים, או שניסינו משהו חדש שלא ממש הצליח. גיטהאב מאפשרת לנו לחזור אחורה לאותה נקודת מפנה לפני שהפרויקט קיבל "תפנית שלילית" ולא תחל את הפרויקט מאותה נקודה. כמו בסרט חזרה בזמן, רק בלי התסביכים של נסיעה בזמן.

היופי בגיטהאב שהוא יודע לסנכרן בין מסמכים שונים בפרויקטים גדולים, וכך ניתן לנהל פרויקטים גדולים, לחלק תפקידים מבלי ששניים שעובדים על שני דברים שונים מאותו פרויקט יתנגשו בטעות (אלא אם אנחנו עובדים על אותו מסמך בדיוק). לגיטהאב יש כמה תכונות שימושיות כמו למשל היכולת ליצור ענפים חדשים - אם אנחנו רוצים לקחת את הפרויקט שעשינו עד כה ולנסות לעשות לו איזשהו פיווט (pivot), כלומר לנסות לקחת אותו למקום אחר, אבל מבלי לפגוע במה שעשינו עד עכשיו. ניתן לעשות זאת ע"י יצירת ענף (branch) חדש לפרויקט, ואז נעבוד על אותו ענף, ובקלות ניתן לחזור לאותו לפרויקט הישן שעבדנו עליו ע"י החלפת ענפים.

אירגונים בגיטהאב - Organization

אתם עובדים בצוות, ולכן חשוב שיהיה לכם חשבון-צוות. בגיטהאב אפשר לעשות את זה בקלות. לא צריך לפתוח חשבון חדש עם סיסמה חדשה - צריך רק לפתוח "ארגון" חדש - Organization: לוחצים על סימן ה + בצד ימין למעלה, ובוחרים New Organization; נותנים שם מקורי לצוות שלכם; לוחצים על People ואז על Invite, ומזמינים את כל חברי הצוות להצטרף לארגון, כך שמי שנכנס לדף של הארגון יראה את כולם. מעכשיו והלאה, בכל פעם שאחד מחברי-הצוות ייצור מאגר חדש, הוא יוכל לבחור לשייך אותו לארגון החדש שיצרתם. למה זה חשוב?

- כדי שלכל המשחקים שתפתחו תהיה גישה חופשית לכל חברי-הצוות, וכולם יקבלו הכרה על העבודה המשותפת.
- כי המשחקים ביוניטי מאד גדולים, ואם תשימו אותם בחשבון הרגיל שלכם - הוא עלול להיחסם (מסיבה זו כדאי לכם לפתוח חשבון צוות גם אם אתם עובדים לבד).

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 ל-GitHub 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:

א. **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.

ב. **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 לכתובת התוכנה. יפתח 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". הדבר הראשון שאנשים רואים כשהם נכנסים למאגר שלכם בגיטהאב זה הקובץ **Readme.md**. זה חלון הראווה שלכם – חשוב שהוא יהיה ברור ויפה. שימו לב קישור למשחק שלכם (שתפרסמו בהמשך באתר itch), וכן הסבר על מהות המשחק ואיך משחקים בו. הוסיפו צילומי מסך לפי הצורך. מומלץ גם לשים סרטוני הדגמה של המשחק, שאפשר ליצור בעזרת תוכנות חנימיות כגון <https://www.screentogif.com>. בנוסף, כיוון שאנחנו בקורס תיכנות, חשוב לשים בקובץ רידמי גם הסברים על הקוד: איזה רכיבים יצרתם ואיך תיכנתתם אותם. שימו קישורים מהקובץ רידמי לשורות הקוד הרלבנטיות. לסיום, הוסיפו קישורים לכל המשאבים שנעזרתם בהם – תמונות, קבצי-קול ונכסים נוספים שהורדתם, מדריכים באינטרנט שנעזרתם בהם, וכו'. הכירו טובה לכל מי שעזר לכם.

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:

 erelsgl credits	b5c15f9 on May 26	🕒 5 commits
Assets	move to new repository	6 months ago
Packages	move to new repository	6 months ago
ProjectSettings	move to new repository	6 months ago
.gitattributes	move to new repository	6 months ago
.gitignore	update gitignore	6 months ago
LICENSE	Initial commit	6 months ago
README.md	credits	5 months ago

(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, Library, 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

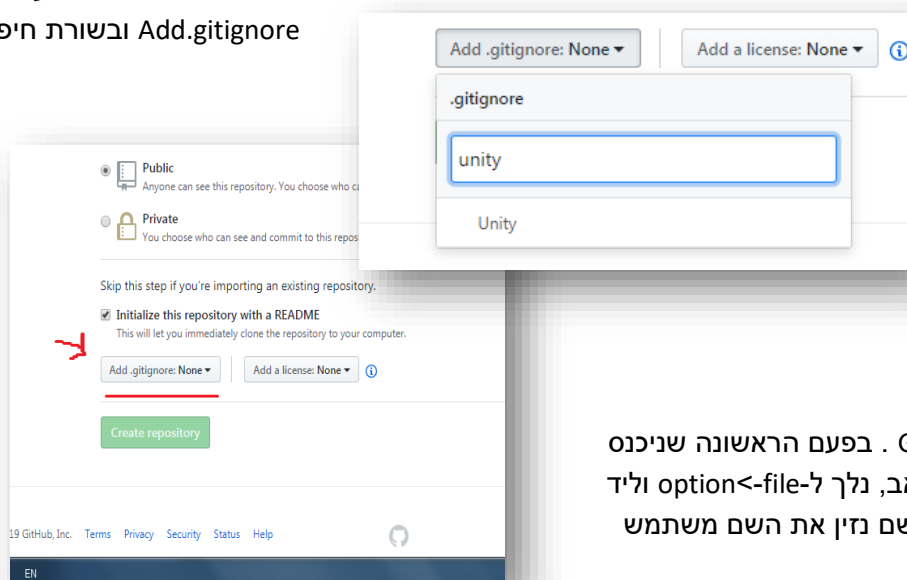
ניצור repository חדש בגיטהאב.

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

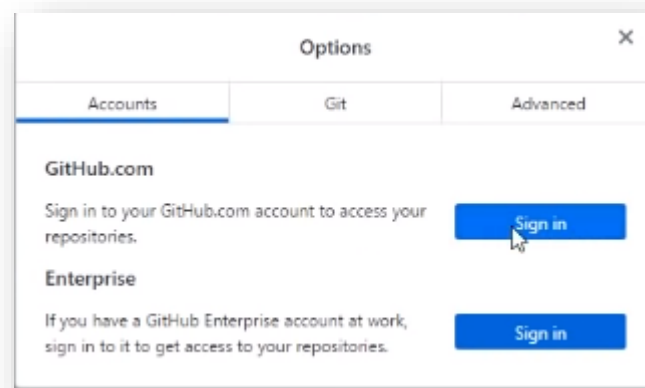
נבחר ב-

unity:



אליו, אם
ה-
והסיסמא

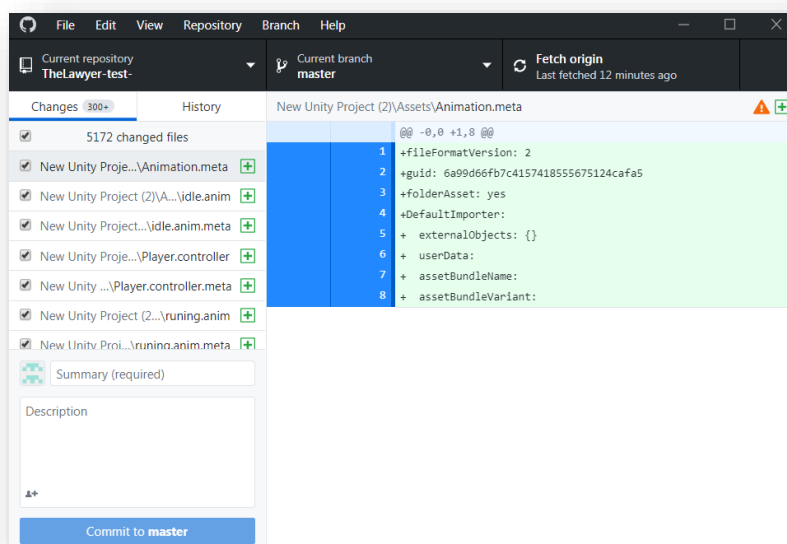
נפתח את ה-GitHub Desktop. בפעם הראשונה שניכנס לא הגדרנו עדיין חשבון גיטהאב, נלך ל-<file option וליד GitHub.com נבחר sign in, ושם נזין את השם משתמש שלנו בגיטהאב כדי להתחבר:



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 ה-unity לתיקייה של ה-repository, וב-unity hub נראה לא יפתח לנו הפרויקט אם נלחץ עליו (כי שינינו את המיקום שלו), לכן נבחר ב-Add-> התיקייה של הפרויקט unity שהעברנו.

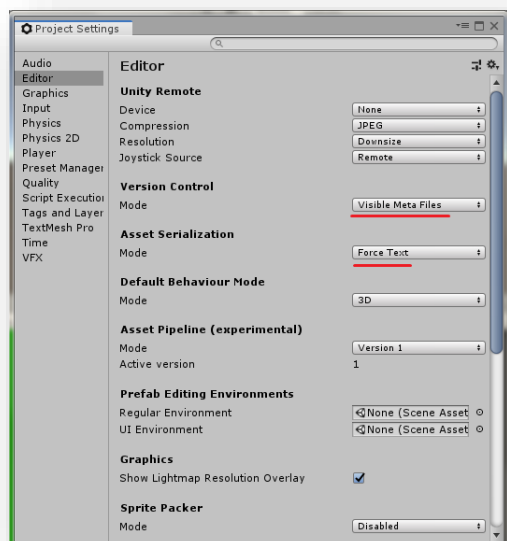
אם נחזור לgithub desktop נראה שהוא מראה לנו את כל השינויים שחלו בתיקייה של repository מאז ה-commit האחרון שעשינו:



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 ל- GitHub desktop ולעשות commit.

ה commit שעשינו מעודכן רק ב- git המקומי שלנו, אבל לא באתר. כדי

לעלות אותו לאתר נצטרך לעשות גם push to origin. עכשיו כל פעם שנעשה איזשהו שינוי במשחק הוא 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- push, ואז נלחץ pull (או ctrl+shift+P שעושה pulling אוטומטית).