***מדריך גיט***

1. **עקרון:**
   1. כל מה שבאופן ישיר, או עקיף קשור לפיתוח הסימולטור, חייב להיות מנוהל במערכת [ניהול גרסאות](http://he.wikipedia.org/wiki/%D7%A0%D7%99%D7%94%D7%95%D7%9C_%D7%92%D7%A8%D7%A1%D7%90%D7%95%D7%AA).
   2. המאגר ( ([**Repository**](http://en.wikipedia.org/wiki/Repository_(revision_control))יכיל את כל קבצי קוד המקור, קבצי הנתונים של הסימולטור, תיעוד, הגדרות וכלים חיצוניים.
   3. המאגר שלנו מנוהל ע"י [גיט](http://he.wikipedia.org/wiki/%D7%92%D7%99%D7%98) ([**Git**](http://en.wikipedia.org/wiki/Git_(software))).
   4. תוכנת ניהול הגיט שנבחרה היא [**SmartGit**](http://en.wikipedia.org/wiki/Comparison_of_Subversion_clients).
   5. המאגר יושב בשרת שנקרא [גיטהאב](http://he.wikipedia.org/wiki/GitHub) ([**GitHub**](http://en.wikipedia.org/wiki/GitHub)).
   6. בגיטהאב יש לנו גם אפשרות לדווח באגים ועוד מספר כלים הקשורים לפיתוח.
2. **מבנה כללי:**
   1. הפרויקט מחולק לשני תתי פרויקטים. פרויקט מקורי ופרויקט חדש.
   2. הפרויקט המקורי הוא גירסת הסימולטור FreeFalcon6. הפרויקט החדש נקרא NegevStorm.
   3. הפרויקט המקורי מכיל קובץ ראשי שנקרא **FreeFalcon.sln.** לפרויקט החדש ישנו קובץ **NegevStorn.sln.**
   4. הפרויקט המקורי מכיל הפניות ל-66 פרויקטים שמרכיבים יחדיו את כל הסימולטור. הפרויקט החדש כרגע הוא בעל מספר קבצים מאוד מצומצם.
   5. בפרויקט המקורי, קובץ ההרצה נוצר מתוך פרויקט הנקרא **FFviper** ולכן זהו גם הפרויקט שיש לו את הגדרות [המקשר](http://he.wikipedia.org/wiki/%D7%9E%D7%A7%D7%A9%D7%A8_(%D7%AA%D7%95%D7%9B%D7%A0%D7%94)) ([**Linker**](http://en.wikipedia.org/wiki/Linker)).
   6. כל קבצי קוד המקור של הפרויקט המקורי נמצאים בתיקיית **SOURCES**. לפרויקט החדש ישנה תיקיה בשם .**NEGEVSTORM**
   7. כל קבצי התיעוד נמצאים בתיקיית **DOCS**.
   8. כל קבצי הנתונים נמצאים בתיקיית **DATA** והיא משותפת לשני הפרויקטים.
   9. לאחר בניית קבצי קוד המקור, קבצי ההרצה ימצאו בתיקיית **BUILD**.
   10. התיקייה הראשית מכילה תיקייה נסתרת שנקראת **GIT**. אין לשנות אותה, או את התוכן שלה, זה בעצם מאגר המידע של גיט.
   11. בתיקייה הראשית נמצא קובץ שנקרא **gitignore**. קובץ זה מכיל רשימת סיומות קבצים שאותם אנו לא רוצים להעלות למאגר. הרי אין סיבה להעלות קבצים זמניים או קבצי הרצה. אין לשנות קובץ זה ללא התייעצות איתי !
   12. בתיקייה הראשית נמצאים גם קובצי ההגדרות לויז'ואל סטודיו 2013 ו-2015 ול-**VisualAssist**.
   13. הפיתוח מורכב משני ענפים:
       1. **Master**: ענף ראשי שרק קבצים בדוקים היטב יעברו אליו.
       2. **הענף הפרטי שלך**: מה שאחליט שמספיק טוב, ימוזג לענף ה-**master**.
   14. להלן תרשים גראפי המראה את התקדמות הפיתוח: <https://github.com/IsraelyFlightSimulator/Negev-Storm/network> כל נקודה מייצגת **Commit** ואם מרחפים עם העכבר עליה, ניתן לראות את ההודעה של אותו **Commit**. ניתן גם להקליק על **Commit** מסוים כדי לראות מה התעדכן.
3. **נוהל עבודה עם גיט:**
   1. הפעל את **SmartGit**.
   2. כל הוספה/מחיקה/שינוי שם או עריכה לאחד או יותר מהקבצים במאגר נרשם ומתועד אוטומטית.
   3. כדי לסנכרן אותו עם המאגר ברשת (ומשם לשאר החברים בפיתוח) יש לבצע שתי פעולות: **Commit** ו-**Push**.
   4. בחלון **Files** מופיעה רשימת הקבצים ששונו.
4. **צפייה בלוג:**
   1. חשוב לראות את היסטוריית השינויים כדי להבין מי שינה מה ולמה.
   2. לחץ על כפתור **Log,** גלגל מטה וקרא את הודעות ה-**commits** השונות ואיך בוצעו מיזוגים בין הענפים והמפתחים השונים.
5. **ביצוע סנכרון:**
   1. לפני שאתה משנה משהו, ייתכן ומישהו אחר כבר שינה אותו. כדי שיהיה בידך את העותק החדש ביותר של המאגר, עליך לבדוק עדכונים לפני תחילת עבודה על אחד הקבצים.
   2. אל דאגה, אתה לא באמת מסוגל לגרום נזק למאגר, כי תמיד אפשר לשחזר לגרסה קודמת, פעולה שנקראת **Revert**.
   3. הקלק על כפתור **Pull** מסרגל הכלים. זה ימשוך את ה-**Commits** האחרונים מהשרת.
6. **ביצוע Commit:**
   1. כל **commit** צריך להיות שינוי בעל נושא מסוים. אין טעם שיהיה קטן ברמה של שינוי שורה בקוד וגם לא גדול מידי לשינויים מאסיביים.
   2. הקלק על כפתור **Commit** מסרגל הכלים.
   3. בחלון שיפתח יש לך אפשרות לבחור את הקבצים אותם אתה רוצה להעלות. לא תמיד תרצה להעלות את כל מה שהשתנה.
   4. בחלון התחתון, רשום את מהות השינוי בעברית.
   5. הקלק על כפתור **Commit**. לא על כפתור **Commit & Push** כי אם יש שגיאה, יהיה הרבה יותר קשה לתקן אותה !
   6. בחלון **Output** מופיע לוג של הפעולות שביצעת, וודא שהכול שם ירוק ולא צהוב או אדום.
   7. בחלון **Outgoing** תופיע רשימת ה-**Commits** שביצעת.
   8. הקבצים עדיין לא נטענו לשרת המרכזי, כלומר ניתן לבצע מספר **Commits** ברמה מקומית ורק אז להעלות את כולם לשרת המרכזי.
7. **ביצוע** **Push:**
   1. הקלק על כפתור **Push** מסרגל הכלים.
   2. בחר את הענף אליו תרצה להעלות את ה-**Commits** שלך. לחץ **Push**.
   3. בחלון **Output** מופיע לוג של הפעולות שביצעת, וודא שהכול שם ירוק ולא צהוב או אדום.
8. **ביצוע Pull Request:**
   1. ברגע שפיתוח של נושא מסוים הסתיים, אתה כמפתח אמור להגיש בקשה לעדכון - **Pull Request** באתר שלך או בתוכנה.
   2. בחלון **Log** בחר את הענף הרצוי בחלון **Branches**, בצע קליק ימני ובחר מהתפריט **Create Pull Request**.
   3. אני, כמפתח ראשי, רואה את כל הבקשות של כלל המפתחים, אעבור על השינויים שביצעת ואחליט האם למזג (**Merge**) את כולם או חלקם עם הענף הראשי של הפיתוח. ניתן לנהל על כך דיון ישירות באתר.
9. **תרגול:**
   1. על מנת לבחון שאתה אכן מבין את תהליך הגיט, אתה מוזמן לערוך את קובץ ה-**license** שנמצא בתיקייה הראשית.
   2. זהו קובץ שמכיל מידע משפטי על הרישיון לכל המידע שבפרויקט.
   3. אני החלטתי שאין שום זכויות יוצרים על כלום ולכן הכול חופשי לחלוטין. זה אומר שאתה יכול לערוך את הקובץ הזה כרצונך.
   4. בצע בו מספר שינויים, בצע **Commit** ו-**Pull**. צור **Pull request** ואני אאשר את השינויים שלך.