

# Git & GitHub





**Git** היא מערכת מבוזרת לניהול גרסאות (DVCS) שיעזרת למפתחים לנוהל שינויים בקוד המקור שלהם לאורך זמן. היא נוצרה על ידי לינוס טורוואלדס, יוצר LINUS,

בשנת 2005



Git מאפשר לעקוב אחר שינויים שבוצעו בקבצים לאורך זמן. זה אומר שניתן לצפות בגרסאות קודמות של הקוד, להשוות שינויים ולהחזיר את הקוד לגרסאות קודמות במידת הצורך.

### מערכת מבוזרת

בניגוד למערכות ניהול גרסאות מרכזיות Git היא מבוזרת. לכל מפתח יש עותק מלא של ההיסטוריה, כולל כל ההיסטוריה שלו. משמעות הדבר היא שגם אם השרת נופל, המפתחים יכולים המשיך לעבוד באופן עצמאי ולסנכרן את השינויים שלהם מאוחר יותר.



## הגדרות הראשונות:

```
git config --global user.name "your_name"          הגדרת שם משתמש גלובלית #
git config --global user.email "your_email@example.com"  הגדרת אימייל גלובלית #
git config --global user.name                      בדיקת שם משתמש נוכחי #
git config --global user.email                     בדיקת אימייל נוכחי #
git config --local user.name "your_name"          שינוי שם משתמש לרייפו ספציפי #
git config --local user.email "your_email@example.com"  שינוי אימייל לרייפו ספציפי #
```

- 
- ב- Git יש שלושה אזורים עיקריים שבהם קבצים יכולים להימצא במהלך העבודה על פרויקט:
1. – **Working Directory** האזור שבו נמצאים כל הקבצים הפיזיים של הפרויקט בפי שהם מאוחסנים במחשב. בלשונו שמקורו בקובץ בלאו ולא נשמר עדין ב- Git נחשב לשינוי שנמצא ב- working directory.
  2. – **Staging Area (Index)** – אזור שבו שומרים שינויים לפני ביצוע commit. commit באשר קובץ מתווסף ל- staging area, אך עדין לא נשמר בהיסטוריה ה- Git commits של commmits.
  3. – **Repository (Local Git History)** – המוקם שבו ה- commmits נשמרים. לאחר ביצוע commit, הקובץ מועבר מה- staging area repository ה- Git. commmits שנזרקם בהיסטוריה של Git, קובץ שנמצא בכך נחשב לשומר באופן קבוע עד ביצוע שינויים נוספים עליו.

## ריפודיזיטורי

Git מachsן את כל הנתונים שלuproיקט בתוך ריפודיזיטורי. זה כולל את ההיסטוריה המלאה של השינויים, הענפים, התגיות ומידע נוסף.

## commit

קומייט הוא תמונה מצב שלuproיקט בנקודת זמן מסוימת. הוא מייצג קבוצה של שינויים שבוצעו בקבצים בתוך הריפודיזיטורי. לכל קומייט יש מזהה ייחודי (hash) והוא מביל מידע כמו שם המחבר, תאריך והודעת הקומייט.



## עבודה עם ריבו (Repository)

```
git init      # חדש Git יצרת ריבוי טור  
git clone URL # GitHub-שיבוט (העתקת) ריבו קיימים מ-  
git status    # Shinoyim, קבצים שלא בוצעו להם) בדיקת מצב הריבו
```

עבודה עם commit

```
git add file_name          # Staging-הוספה קובץ ספציפי ל-  
git add .                  # Staging-הוספה כל הקבצים שונים ל-  
git commit -m "message"   # Um hodaah Commit ביצוע #  
git commit --amend -m "new message" # Um hodaah shel h Commit
```



עובדת עם לוג לראות היסטורייה של commit

```
git log          # הצגת היסטוריית Commits  
git log --oneline # הצגת היסטורייה בצורה מקוצרת  
git diff         # -Staging -הצגת הבדלים בין הקבצים ששונו אך לא נוספו ל  
git diff --staged # Commit -L -Stage -הצגת הבדלים בין קבצים שנמצאים כבר ב
```

```
git reset <commit_id> # Commit -Working Directory -האחרון והשارة השינויים ב  
git reset --hard <commit_id>1 # Commit -Biutol h -Biutol h -hard -מיחיקת השינויים לגמרי
```

ענפים משמשים לעבודה על תכונות חדשות, תיקוני באגים או ניסויים מבלוי להשפייע על קוד הבסיס הראשי. הם מאפשרים למפתחים לבדוק שינויים עד שהם מוכנים להתמזג לתוך הענף הראשי) שלרוב נקרא master או ( main )

```
git branch          # הצגת כל הענפים  
git branch new_branch # יצרת ענף חדש  
git switch new_branch # מעבר לענף חדש ( `checkout`-חלופה ל )  
git checkout -b new_branch # יצרת ענף חדש ומעבר אליו  
git merge new_branch # מיזוג ענף לתוך הענף הנוכחי  
git branch -d new_branch # מחיקת ענף
```

## מיזוג(Merging)

מיזוג הוא התהליך של שילוב שינויים מענף אחד לתוך ענף אחר. לרוב, זה מתבצע באשר פיצ'ר הושלם והוא מוכן להשתלב בקוד הבסיס הראשי.

Git מאפשר לעבוד עם ריפורזיטוריים מרוחקים המאוחסנים בשרת או פלטפורמות כמו Bitbucket, GitHub, GitLab ועוד. ניתן שיתוף פעולה עם מפתחים אחרים ומספק מקום מרכזי לשיתוף הקוד.

```
git remote -v                                     #-הציגת רשימת ה Remotes  
git remote add origin URL                      #GitHub-חיבור רIFO מוקומי ל  
git push -u origin main                         #GitHub-שליחת השינויים הראשונים ל GitHub  
git pull origin main                           #משיכת שינויים מהרIFO המרוחק  
git fetch                                         #בל, למזג GitHub-משיכת עדכונים מ GitHub  
git rebase main                                 #על הענף הראשי Rebase ביצוע #
```

## Pull Requests

Pull Requests הם פיצ'ר בפלטפורמות אירוח של Git כמו GitHub ו-GitLab. הם מאפשרים למפתחים להציג שינויים בריפורזיטורי ולבקש שיבחנו אותם וימצגו אותם לתוך הענף הראשי

## **GitHub** שירות אירוח מבוסס ענן

ריפורזיטורי של Git.

- מספק פלטפורמה לשיתוף קוד ושיתוף פעולה בין מפתחים.

- כולל כלים לניהול פרויקטים, Pull Requests, ביקורת קוד, CI/CD ועוד.

- אפשר עבודה עם ריפורזיטורי מרוחקים, ברגע שפתחים יכולים לעבוד יחד על אותו פרויקט.



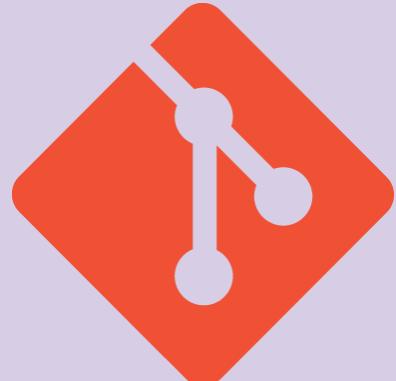
## Git - מערכת מבוזרת לניהול גרסאות (DVCS)

- אפשר לעקוב אחרי שינויים בקוד, לשמר

- היסטוריה, לחזור לגרסאות קודמות ולשתף קוד.

- עובד **מקומית** על מחשב המפתח, ללא תלות באינטרנט.

- אפשר יצרת ענפים, ביצוע מיזוגים וניהול קבצים בשורת הפקודה.



## GitHub שירות אירוח מבוסס ענן

- ריפודיזיטורייז של Git.

- מספק פלטפורמה לשיתוף קוד ושיתוף פעולה בין מפתחים.

- כולל כלים לניהול פרויקטים, ביקורת קוד, CI/CD ועוד.

- אפשר עבודה עם ריפודיזיטורייז מרוחקים, ברגע שפתחים יכולים לעבוד יחד על אותו פרויקט.