

YENİ BAŞLAYANLAR İÇİN GIT EĞİTİMİ

GRUP ÜYELERİ:

PINAR GÜZELÖZ

ZEYNEP ECE KUTLU

YENİ BAŞLAYANLAR İÇİN GIT EĞİTİMİ

Yeni başlayanlar için git eğitimi, git komutlarının nasıl çalıştığı ve nerelerde kullanıldığı konusunda yol gösteren, yeni başlayan seviyesine uygun bir eğitimidir.



Git Nedir?

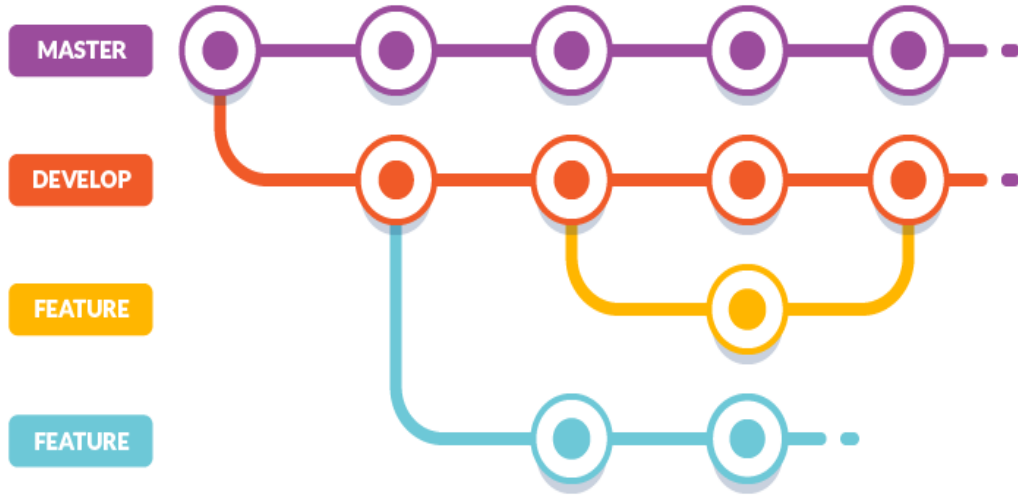
Git yazılım geliştirme süreçlerinde kullanılan, hız odaklı, dağıtık çalışan bir sürüm kontrol ve kaynak kod yönetim sistemidir.

Yazılım projelerini takip etmek, sürüm kontrolü yapmak ve ekip üyeleri arasında işbirliği yapmak için kullanılır.

Git'in Özellikleri Nelerdir?

Dosyalarını güncelleyen ve eski hallerini de saklamak isteyen herkesin bildiği gibi dosyalarımıza verdiğimiz “son, bu son, bu en son, bu best son, son son” gibi terimleri kullanmamıza gerek kalmadan en yeni halini ve diğer zamanlarda yapılan bütün değişiklikleri de kendisinde barındıran bir yapıdır Git.

- Depolarla projeleri yönetir.
- Yerel bir kopya üzerinde çalışmak için bir projeyi klonlar.
- Evreleme ve Taahhüt ile değişiklikleri kontrol eder ve takip eder.
- Projenin en son sürümünü yerel bir kopyaya çeker.
- Ana projeye yerel güncellemeleri iletir.



Neden Git Kullanılmalıdır?

- *Geliştiriciler dünyanın her yerinden birlikte çalışma imkanına sahiptir.*
- *Geliştiriciler projenin geçmiş değişikliklerini ve hatalarını görebilir.*
- *Git, veri yapısı ve algoritması sayesinde hızlıdır ve büyük projelerde bile etkili performans sergiler.*
- *Geliştiricilerin %70'inden fazlası git kullanır. Dolayısıyla geniş bir kullanıcı tabanına sahiptir.*

Git ve GitHub

Github sürüm kontrol sistemi olarak Git kullanan yazılım geliştirme projeleri için web tabanlı bir depolama servsidir.

GitHub servisi sayesinde pek çok kişiden oluşan kalabalık yazılım geliştirici ekipler bile aynı anda aynı proje üzerinde çalışabilir ve ayrı değişiklikler yapabilirler.

GitHub yalnızca geliştirici ekibin değil yazılımı hazırladıkları müşterinin de kullanabileceği bir alana sunar.

Müşteri GitHub servisi üzerinden değişiklik yapmak istediği yerlerde gerekli düzenlemeleri ve güncellemeleri yaparak iletişim sürecini hızlandırabilir.



GitHub Profili

GitHub platformu, portfolyo oluřturmak iin yaygın olarak kullanılan bir sitedir. Veri bilimcilerin projelerini sergilemelerini saęlar ve CV’de iř deneyimleri olarak da sayılabilir. Proje sergilemek ve birlikte alıřmak, bir giriřim bařlatmak ve arařtırma alıřmaları iin fırsatlar yaratır.

Bir GitHub profili rneęi:

Muhammad Adeel
Adeel91
2 followers · 1 following

Achievements
Beta Send feedback

Highlights
Developer Program Member

Skills
JAVASCRIPT, TYPESCRIPT, NODEJS, REACT, VUEJS, PYTHON, PHP, NEXT, REDUX, REACT ROUTER, MATERIAL UI, PROGRESSIVE WEB APP, STORYBOOK, C++, HTML5, CSS3, BOOTSTRAP, MONGODB, REDIS, POSTGRESQL, MYSQL, HEROKU, AMAZON AWS, GIT, GITHUB, GITLAB, BITBUCKET, JQUERY

Top Blog Posts
• Frontend Testing Techniques
• 8 React Open-Source projects to learn and become a React.js Developer
• Understand NFT in Simple Words
• Journey to Become a Blockchain Developer
• Are we in NFT Bubble?

Stats
Muhammad Adeel's GitHub Stats
Total Stars Earned: 5
Total Commits: 334
Total PRs: 0
Total Issues: 0
Contributed to: 0
A+

Most Used Languages
JavaScript 57.97%, CSS 4.00%, Java 30.24%, TypeScript 2.82%, PHP 4.97%

visitors 51

En Temel Git Komutları

- **git config:** e-posta, kullanıcı adı ve dosya formatı gibi kullanıcıya özel yapılandırmaları ayarlamak için kullanılır.
- **git init:** yeni bir depo başlatmak için kullanılır.

- ***git clone:*** mevcut bir URL'den bir havuz elde etmek için kullanılır.
- ***git add:*** repoya bir dosya eklemek için kullanılır.
- ***git push:*** deponun uzak dalına yerel taahhütler gönderir.
- ***git pull:*** proje ana dosyasında yapılan değişikliklerin bilgisayardaki versiyona çekilmesinde kullanılır.
- ***git commit:*** dosyayı sürüm geçmişinde kalıcı olarak kaydeder veya anlık görüntüler.
- ***git status:*** işlenmesi gereken tüm dosyaları listeler.
- ***git branch:*** tüm branchleri görmeyi sağlar.
- ***git merge:*** üzerinde çalışılan iki parçayı birleştirmede kullanılır.
- ***git diff:*** henüz aşamalı olmayan dosya farklılıklarını gösterir.
- ***git reset:*** dosyanın aşamasını kaldırır ancak dosya içeriğini korur.
- ***git status:*** işlenmesi gereken tüm dosyaları listelemede kullanılır.
- ***git rm:*** dosyayı çalışma dizininden siler.
- ***git log:*** geçerli dalın sürüm geçmişini listelemek için kullanılır.

BAŞLANGIÇ

Git Kurulumu : git-scm.com adresine giderek ekrana çıkan işletim sistemi seçeneklerinden kendinize uygun olanı seçerek indirmeyi başlatabilirsiniz.

- *Kurulum tamamlandıktan sonra kullanıcı adı ve e-posta yapılandırılır.*

```
git config --global user.name "your-user-name"
git config --global user.email "your@email.com"
```

- *Yeni klasör oluşturulur ve klasöre girilir.*

```
mkdir DataCamp-Git
cd .\DataCamp-Git
```

- *Proje Git'e dahil edilir.*

```
git init

>>> Initialized empty Git repository in C:/Repository/GitHub/DataCamp-Git/.git/

git remote add origin https://github.com/kingabzpro/DataCamp-Git.git
```

- *Henüz commitlenmeyen değişiklikleri görmek için:*

```
git status

>>> On branch main
>>> No commits yet
>>> Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   README.md
```


- *Dosyaları sürüm geçmişine kalıcı olarak kaydetmek için:*

```
git commit -m "first commit"

>>> [main (root-commit) ed9c886] first commit
>>> 1 file changed, 1 insertion(+)
>>> create mode 100644 README.md
```

- *Geçerli Git dalının sürüm geçmişini listelemek için:*

```
git log [<options>] [<revision-range>] [--] [<path>...]
```

- *İki versiyon arası farkları görmek için:*

```
git diff [<options>] [<commit>] [--] [<path>...]
git diff [<options>] --cached [--merge-base] [<commit>] [--] [<path>...]
git diff [<options>] [--merge-base] <commit> [<commit>...] <commit> [--]
[<path>...]
git diff [<options>] <commit>...<commit> [--] [<path>...]
git diff [<options>] <blob> <blob>
git diff [<options>] --no-index [--] <path> <path>
```

- *Otomatik yeni bir commit üretmeden değişikliği geri almak için:*

```
git restore [<options>] [--source=<tree>] [--staged] [--worktree] [--]
<pathspec>...
git restore [<options>] [--source=<tree>] [--staged] [--worktree] --
pathspec-from-file=<file> [--pathspec-file-nul]
git restore (-p|--patch) [<options>] [--source=<tree>] [--staged] [--]
worktree [--] [<pathspec>...]
```

- *Dizinin sonuna yeni değer eklemek için:*

```
git push origin readme:readme
>>> remote: Resolving deltas: 100% (1/1), completed with 1 local object.
>>> remote: Create a pull request for 'readme' on GitHub by visiting:
>>> remote: https://github.com/kingabzpro/DataCamp-Git/pull/new/readme
>>> remote:
To https://github.com/kingabzpro/DataCamp-Git.git
>>> * [new branch]      readme -> readme
```

- *Dosyayı silmek için:*

```
git rm [-f | --force] [-n] [-r] [--cached] [--ignore-unmatch]
      [--quiet] [--pathspec-from-file=<file>] [--pathspec-file-nul]
      [--] [<pathspec>...]
```

- *Var olan bir Git repository'sinin kopyalanması için :*

```
git clone https://github.com/kingabzpro/DataCamp-Git.git

>>> Cloning into 'DataCamp-Git'...
>>> warning: You appear to have cloned an empty repository.

cd .\DataCamp-Git\
```

- *Branc'ler arası veya commitler arası geçiş yapmak için :*

```
git checkout -b readme
```

Branch kullanımı

Branch oluşturmak kullanıcıya çalıştığı projenin farklı versiyonlarına erişmesini sağlar. Kullanıcı, projesine bir yenilik eklemek istediğinde yaptığı değişiklik projenin çalışmasını olumsuz etkileyebilir. Bu gibi durumlarda projemizin o anki halini bozmamak için branch kullanabiliriz.

```
git branch [--color[=<when>] | --no-color] [--show-current]
    [-v [--abbrev=<n> | --no-abbrev]]
    [--column[=<options>] | --no-column] [--sort=<key>]
    [--merged [<commit>]] [--no-merged [<commit>]]
    [--contains [<commit>]] [--no-contains [<commit>]]
    [--points-at <object>] [--format=<format>]
    [(-r | --remotes) | (-a | --all)]
    [--list] [<pattern>...]
git branch [--track[=(direct|inherit)] | --no-track] [-f]
    [--recurse-submodules] <branchname> [<start-point>]
git branch (--set-upstream-to=<upstream> | -u <upstream>) [<branchname>]
git branch --unset-upstream [<branchname>]
git branch (-m | -M) [<oldbranch>] <newbranch>
git branch (-c | -C) [<oldbranch>] <newbranch>
git branch (-d | -D) [-r] <branchname>...
git branch --edit-description [<branchname>]
```

Git readme kullanımı

readme dosyası web depolama servislerinde birinde bir proje oluşturduğunuzda projenin açıklama kısmıdır. Projenizi ziyaret eden bir kullanıcının projeye

tıkladığında önüne gelen sayfadır.

```
git add README.md
git commit -m "project description and links to blog"

>>> [readme f3b8b9b] project description and links to blog
>>> 1 file changed, 8 insertions(+)
```

KAYNAKÇA :

<https://www.w3schools.com/git/default.asp>

<https://www.datacamp.com/tutorial/github-and-git-tutorial-for-beginners>

linkedin.com

git-scm.com

Dev.to

Webtekno.com

Medium.com

