



GIT & GITHUB



GRUP 3



Neler öğreneceğiz?

01

Versiyon Kontrol Nedir?
Git Nedir?
Tercih Edilme Sebepleri?
Kurulumu Nasıldır?

03

GitHub Nedir?
Neden Kullanılmalıdır?

02

Git Komutları Nelerdir?

04

Github Nasıl Kullanılır?

Versiyon Kontrol Nedir?

Versiyon kontrolü nedir ve bizi neden ilgilendirmeli? Versiyon kontrolünü bir dosya veya bir küme dosyadaki değişiklikleri takip edebilmek için uyguladığımız bir yöntem olarak tanımlayabiliriz.



git



SUBVERSION

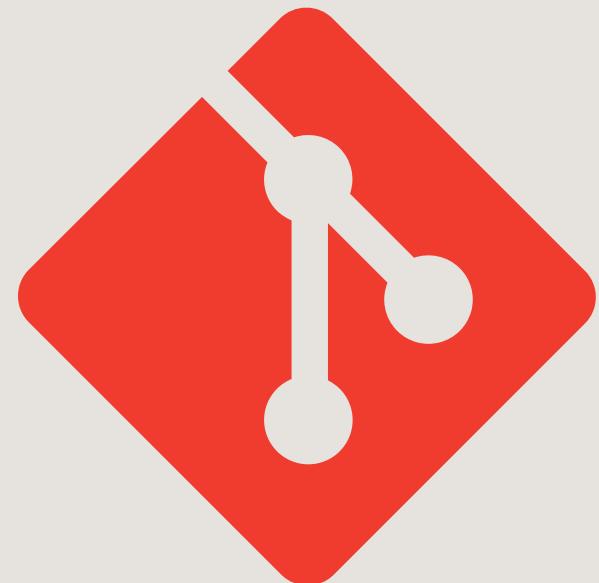


mercurial

Neden Versiyon Kontrol?

- Uyumlu ekip çalışması
- Versiyonların düzgün bir şekilde takip edilebilmesi
- Önceki Versiyonlara Geri Dönebilme
- Dosyalarınızın neden değiştığını anlama
- Yedekleme



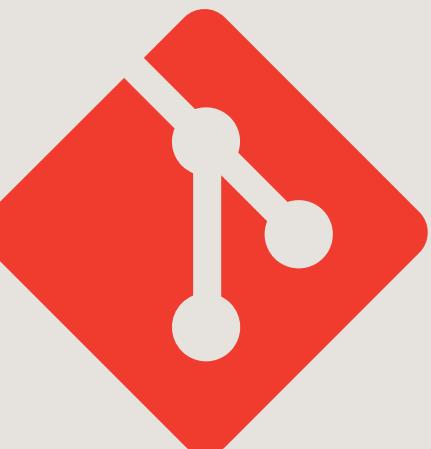


git

Logo ve İsim Anlamı nedir?



git



"git" Açılımı Nedir?

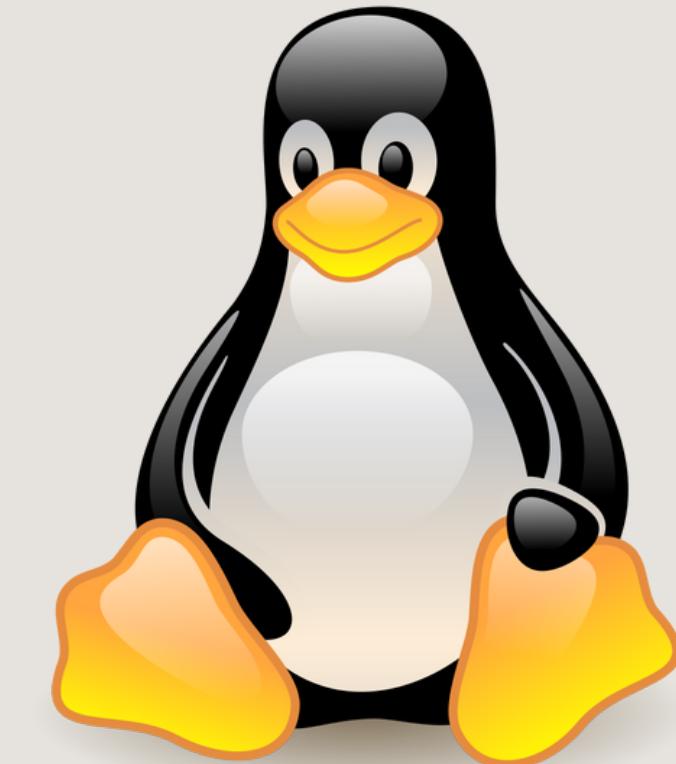
Global Information Tracker (Küresel bilgi takip sistemi)'dır

Logo Neden bu şekildedir?

Bir projeyi aynı anda birden fazla kişi birbirinden bağımsız şekilde geliştirebildiği için "branch (dallar)" simgesi logoda tercih edilmiştir.

Git Tarihçesi

Git 2005 yılında, başta Linus Torvalds olmak üzere Linux çekirdeğini de kodlayan ekip tarafından Linux kaynak kodunu versiyon kontrolü altında tutmak ve kendi iş akışlarını düzenlemek için geliştirilmiştir.



Linux

GIT NEDİR?

Git Versiyon Kontrol Sistemi, bir proje üzerinde birden çok kişinin çalışmasına ve her birinin kendi versiyonunu oluşturmasına, daha sonra değişiklik yapılmak istendiğinde istenilen versiyona dönülüp oradan değişiklik yapılmasına olanak veren bir kontrol sistemidir.

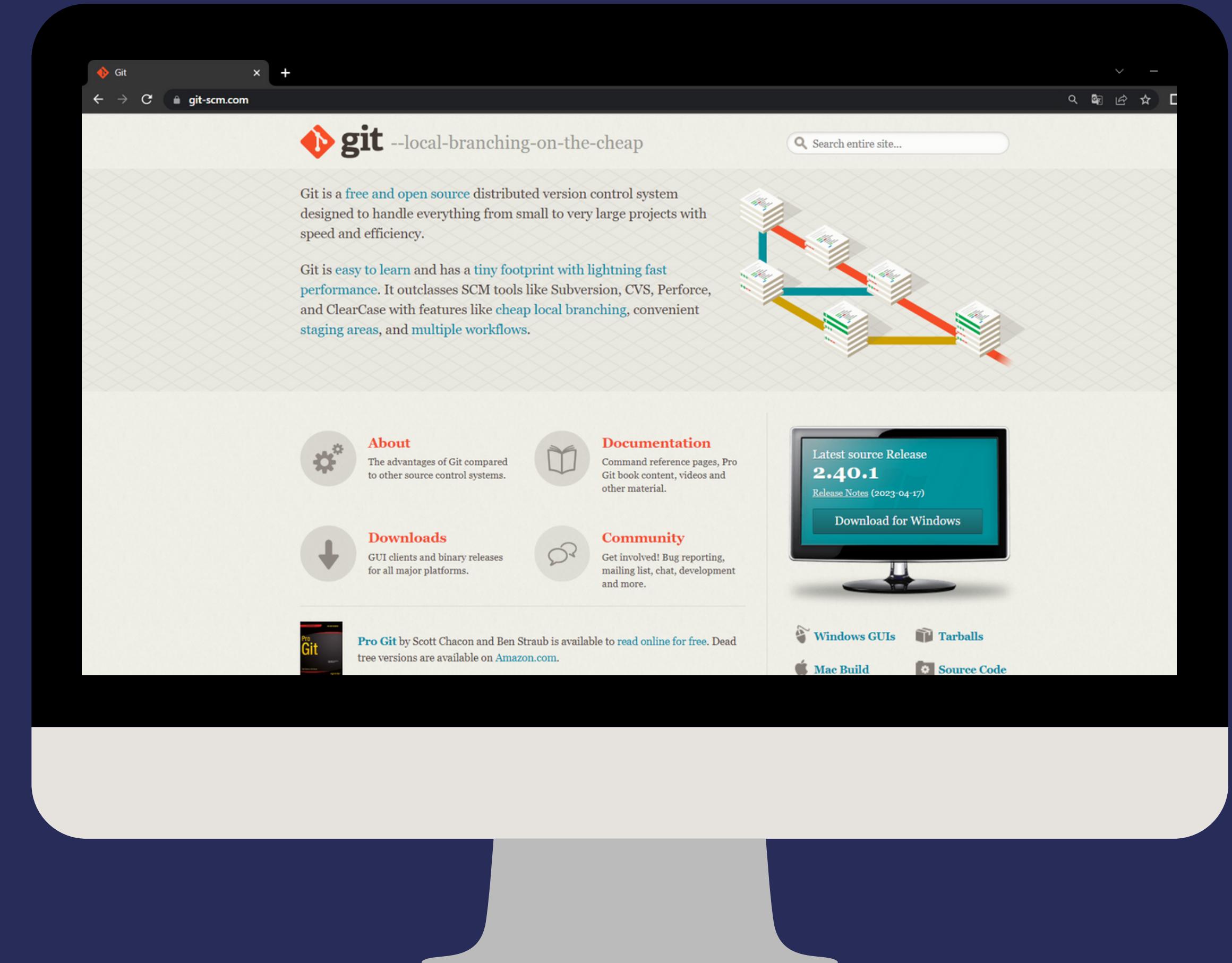


Tercih Edilme Sebepleri?

- Takımların aynı proje ortamında çalışmasını kolaylaştırır.
- Kim düzenledi? Ne değişti? Ne zaman değişti? bilgilerini tutar
- Herhangi bir dosyaya veya versiyona her zaman dönüş sağlayabilirsiniz?
- Yerel veya uzak bilgisayarlarında çalışabilir



KURULUMU NASILDIR?



 git --distributed-is-the-new-centralized

Search entire site...

About

Documentation

Downloads

- GUI Clients
- Logos

Community

The entire [Pro Git book](#) written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Download for Windows

[Click here to download](#) the latest (2.40.1) 64-bit version of Git for Windows. This is the most recent maintained build. It was released about 1 month ago, on 2023-04-25.

Other Git for Windows downloads

- Standalone Installer**
- 32-bit Git for Windows Setup.**
- 64-bit Git for Windows Setup.** 
- Portable ("thumbdrive edition")**
- 32-bit Git for Windows Portable.**
- 64-bit Git for Windows Portable.**

Using winget tool
Install [winget tool](#) if you don't already have it, then type this command in command prompt or Powershell.
`winget install --id Git.Git -e --source winget`

The current source code release is version 2.40.1. If you want the newer version, you can build it from [the source code](#).

Now What?

Now that you have downloaded Git, it's time to start using it.

 [Read the Book](#)
Dive into the Pro Git book and learn at your own pace.

 [Download a GUI](#)
Several free and commercial GUI tools are available for the Windows platform.

 [Get Involved](#)
A knowledgeable Git community is available to answer your questions.



 **Git 2.40.1 Setup**

Select Components
Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

Additional icons

- In the Quick Launch
- On the Desktop

Windows Explorer integration

- Git Bash Here
- Git GUI Here

Git LFS (Large File Support)

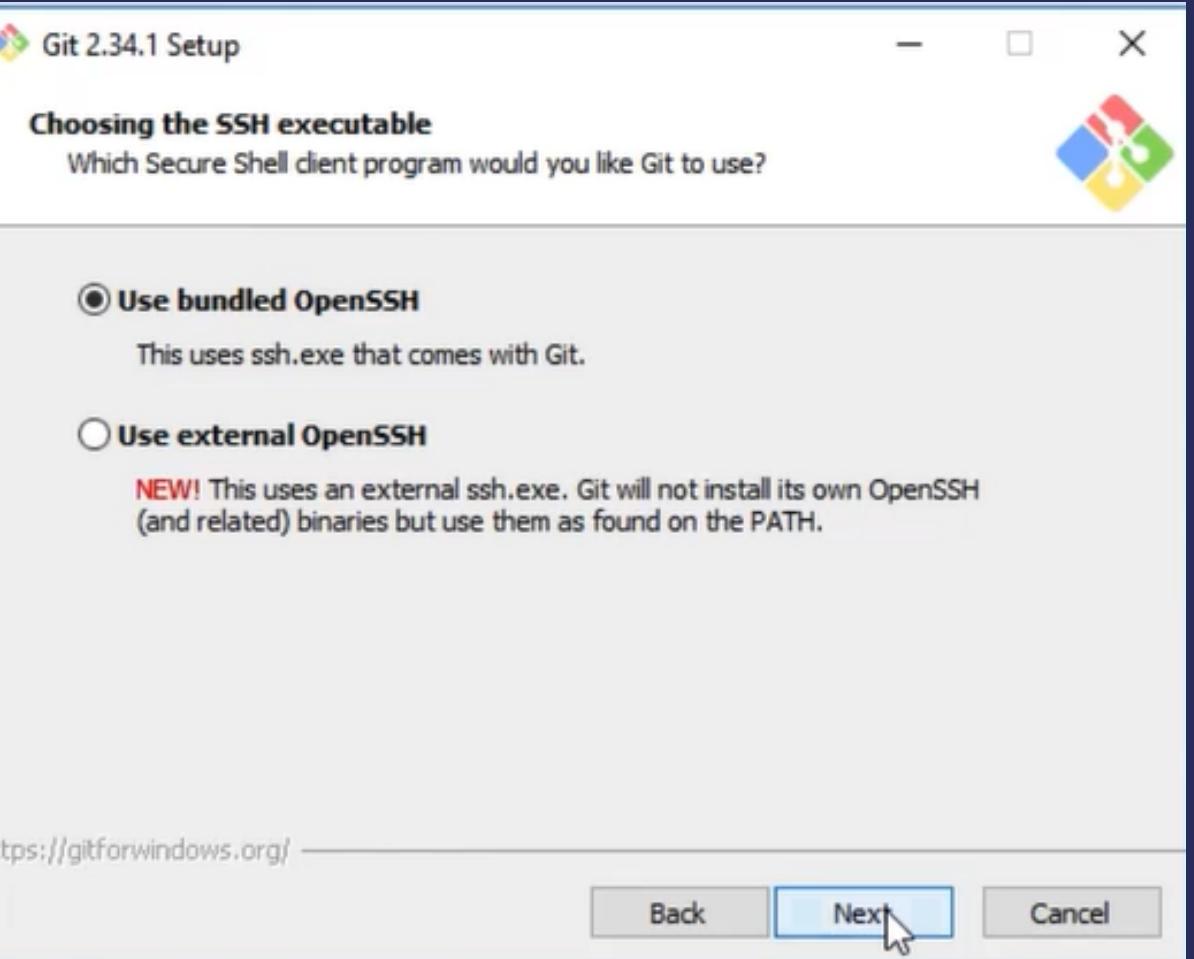
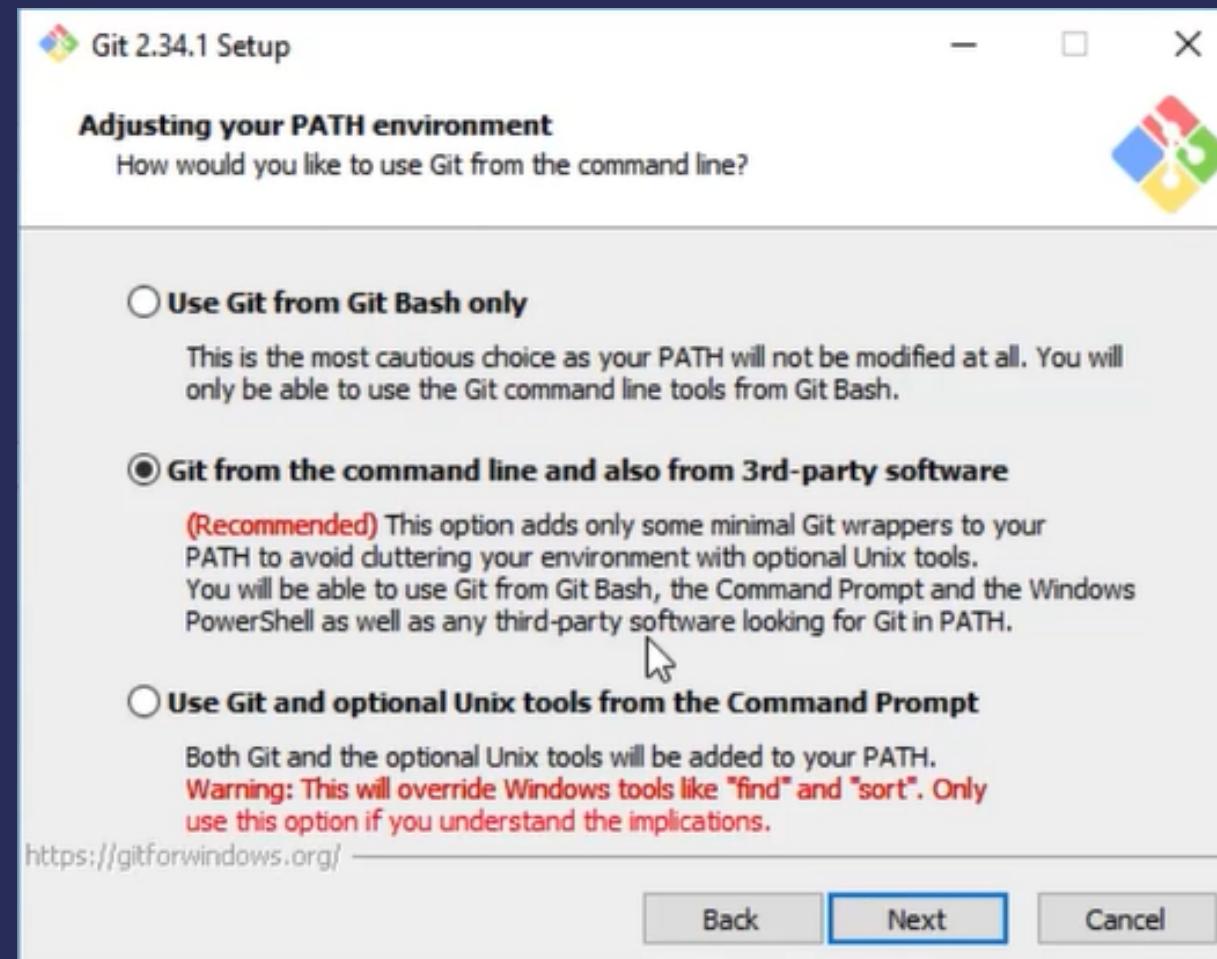
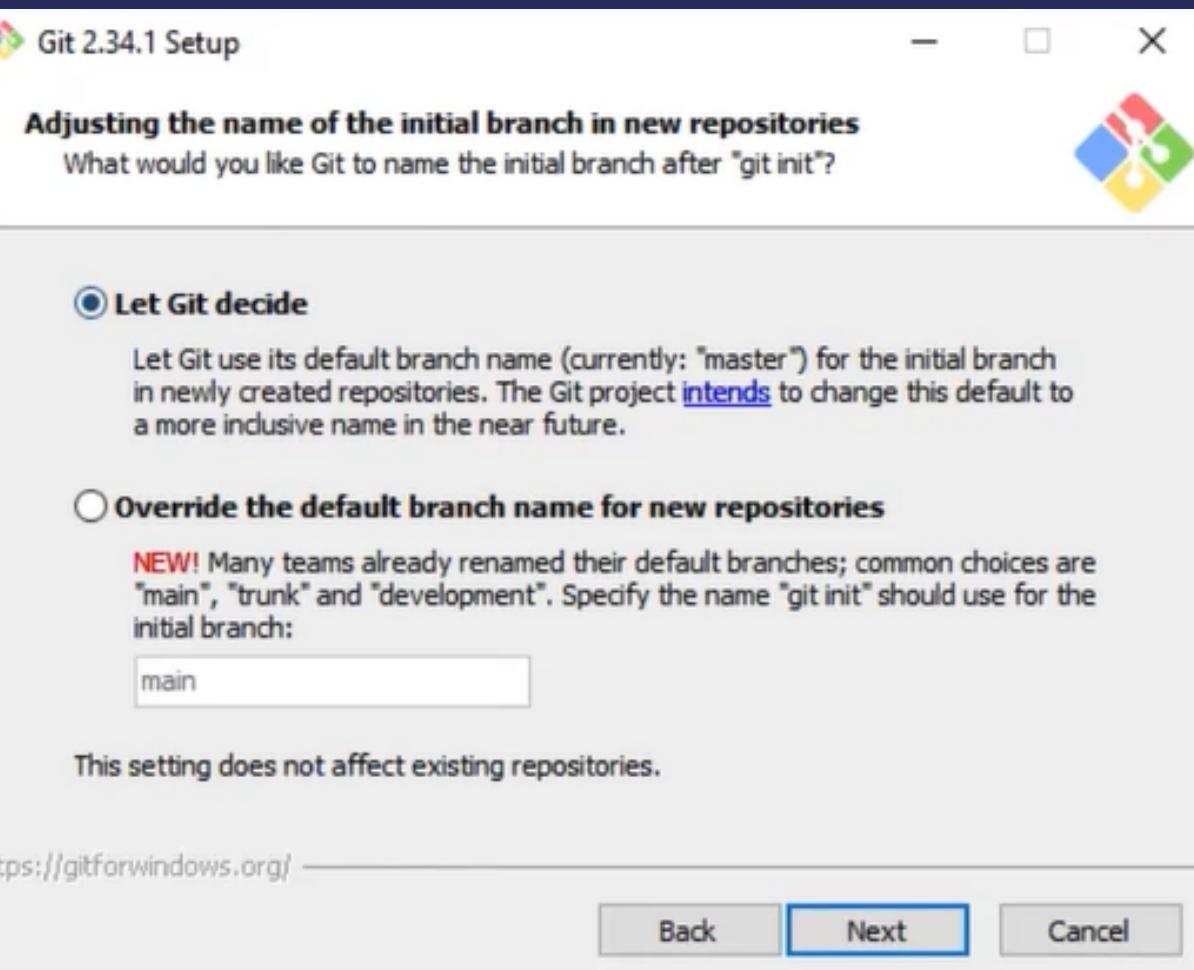
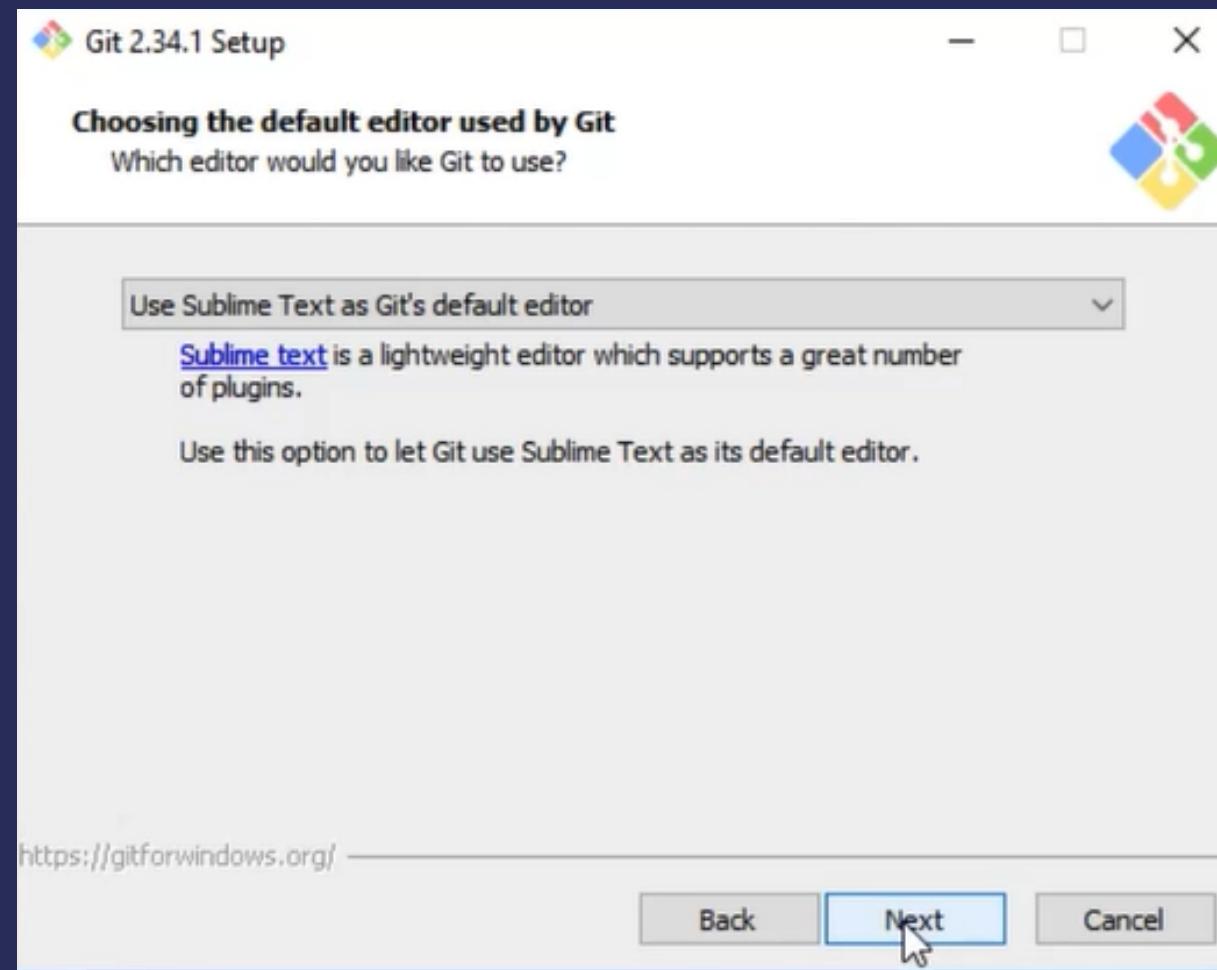
Associate .git* configuration files with the default text editor

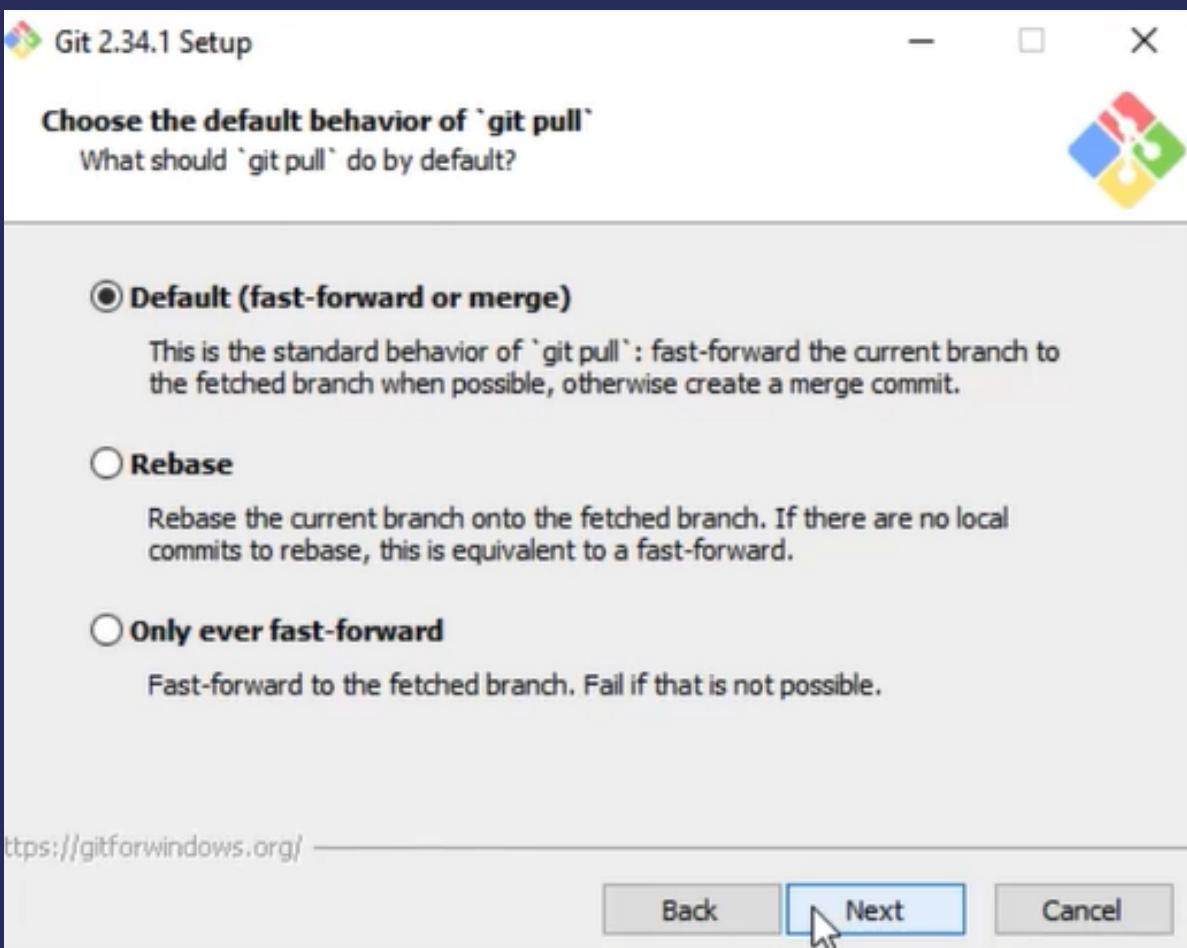
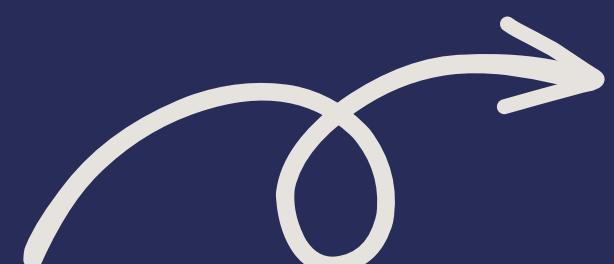
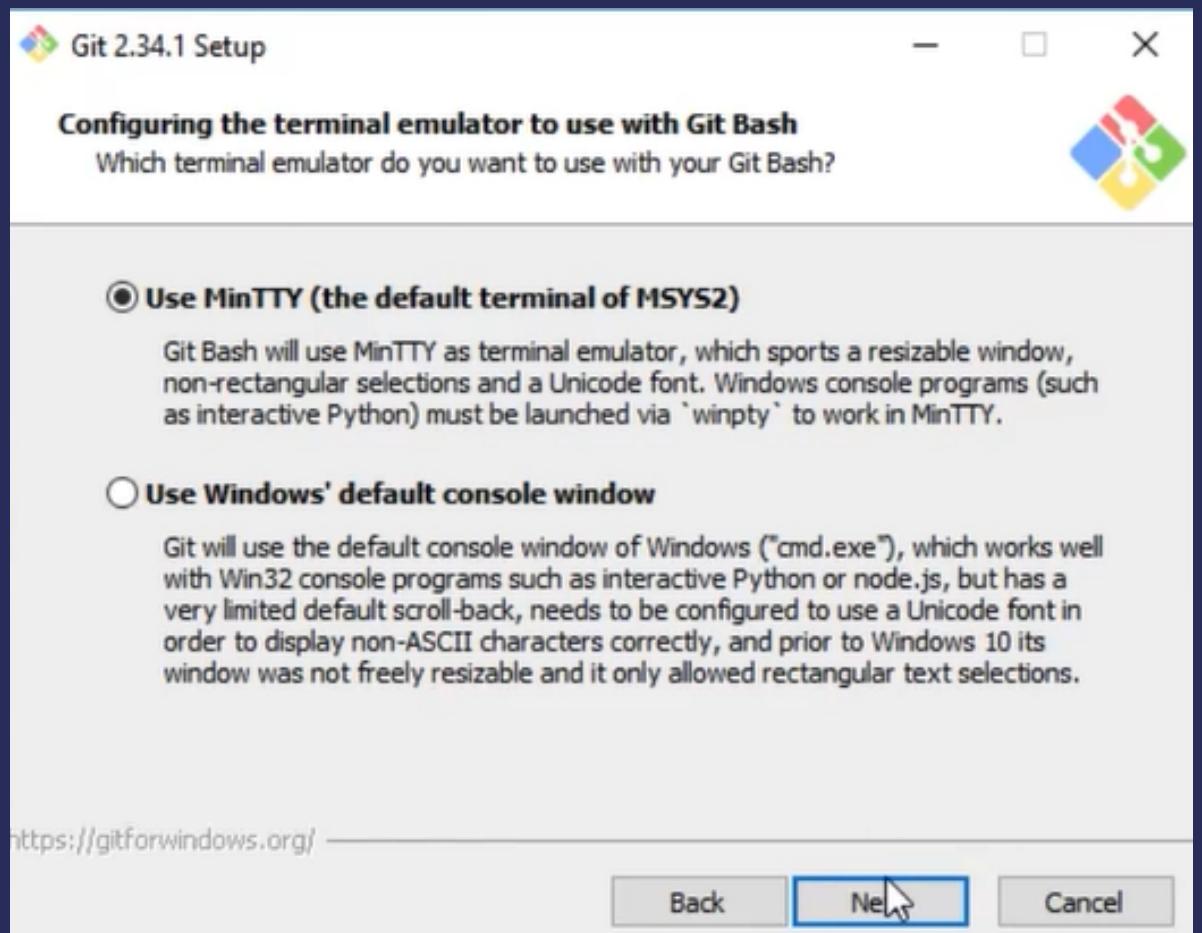
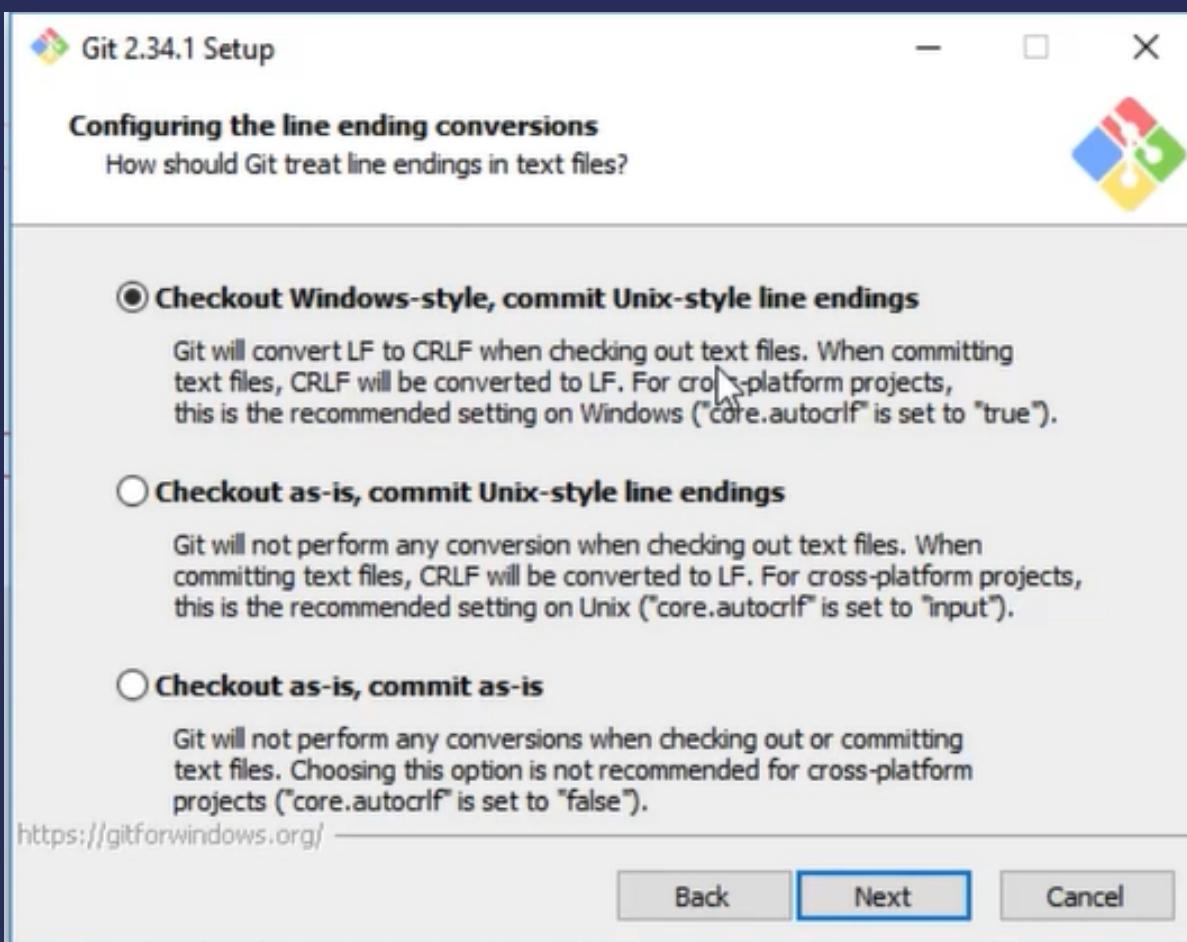
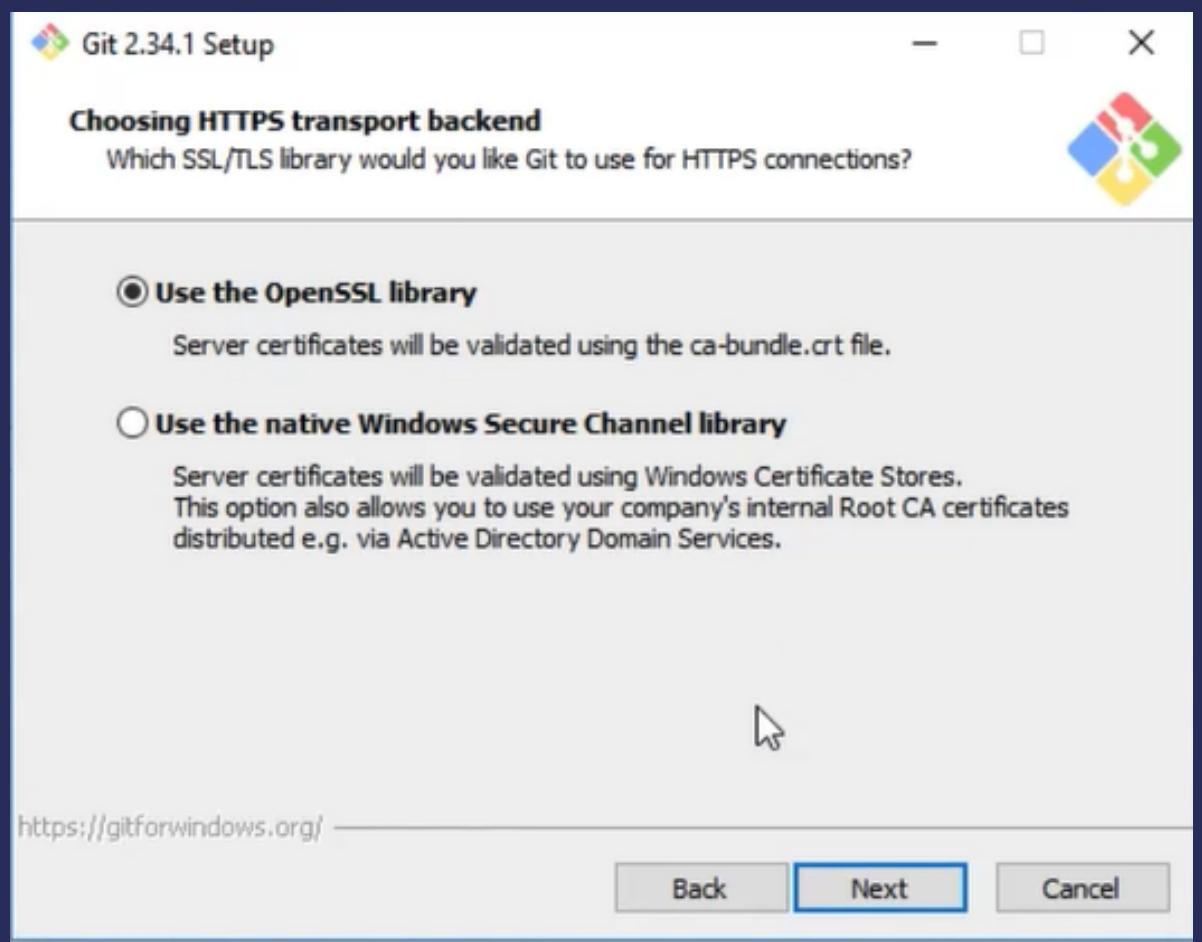
Associate .sh files to be run with Bash

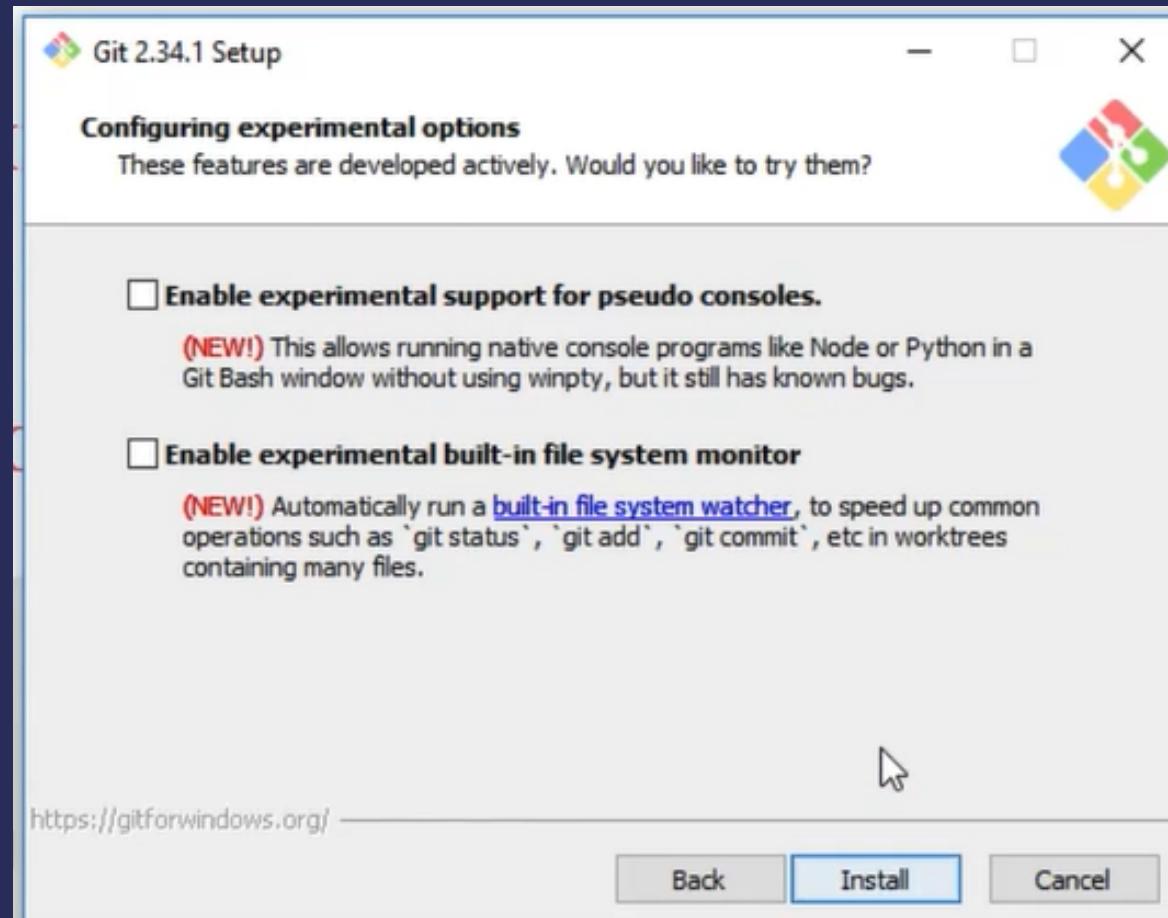
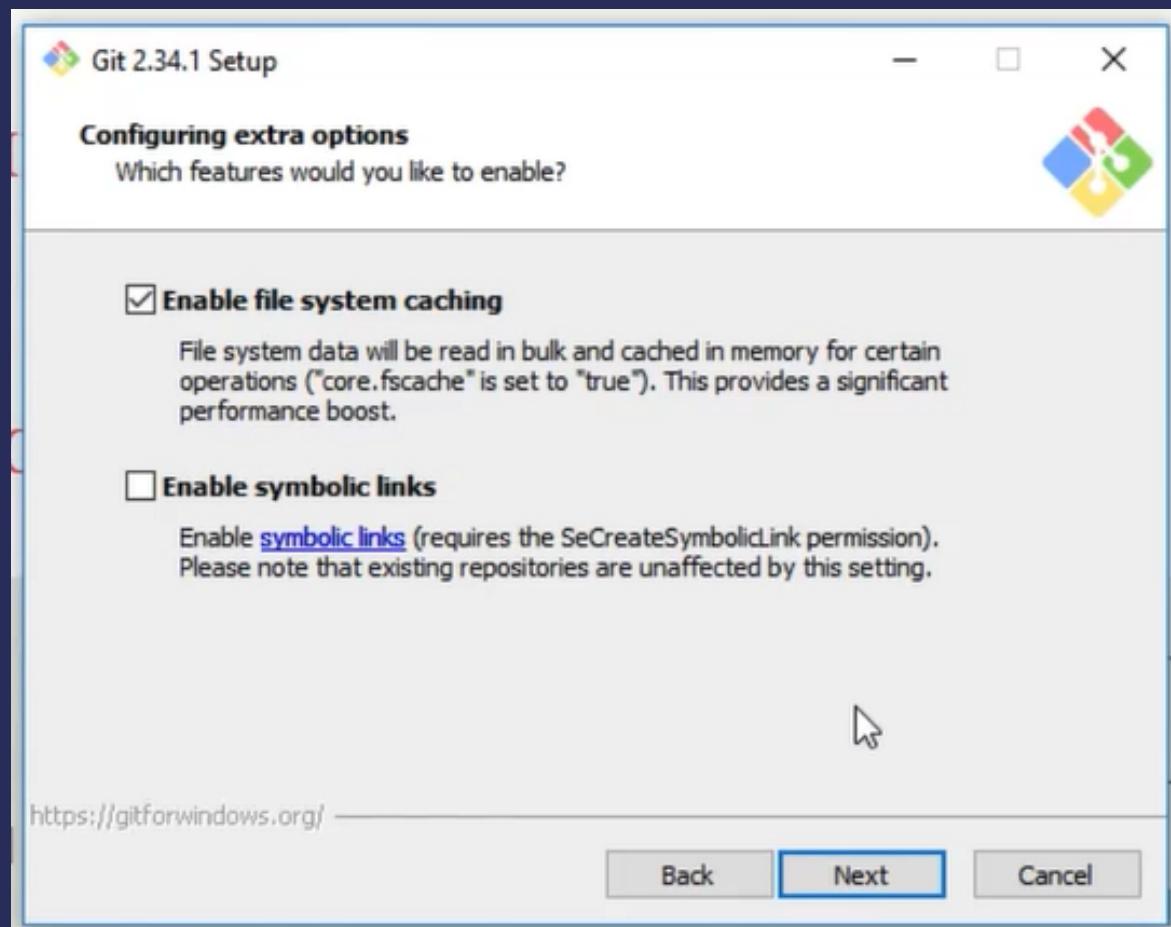
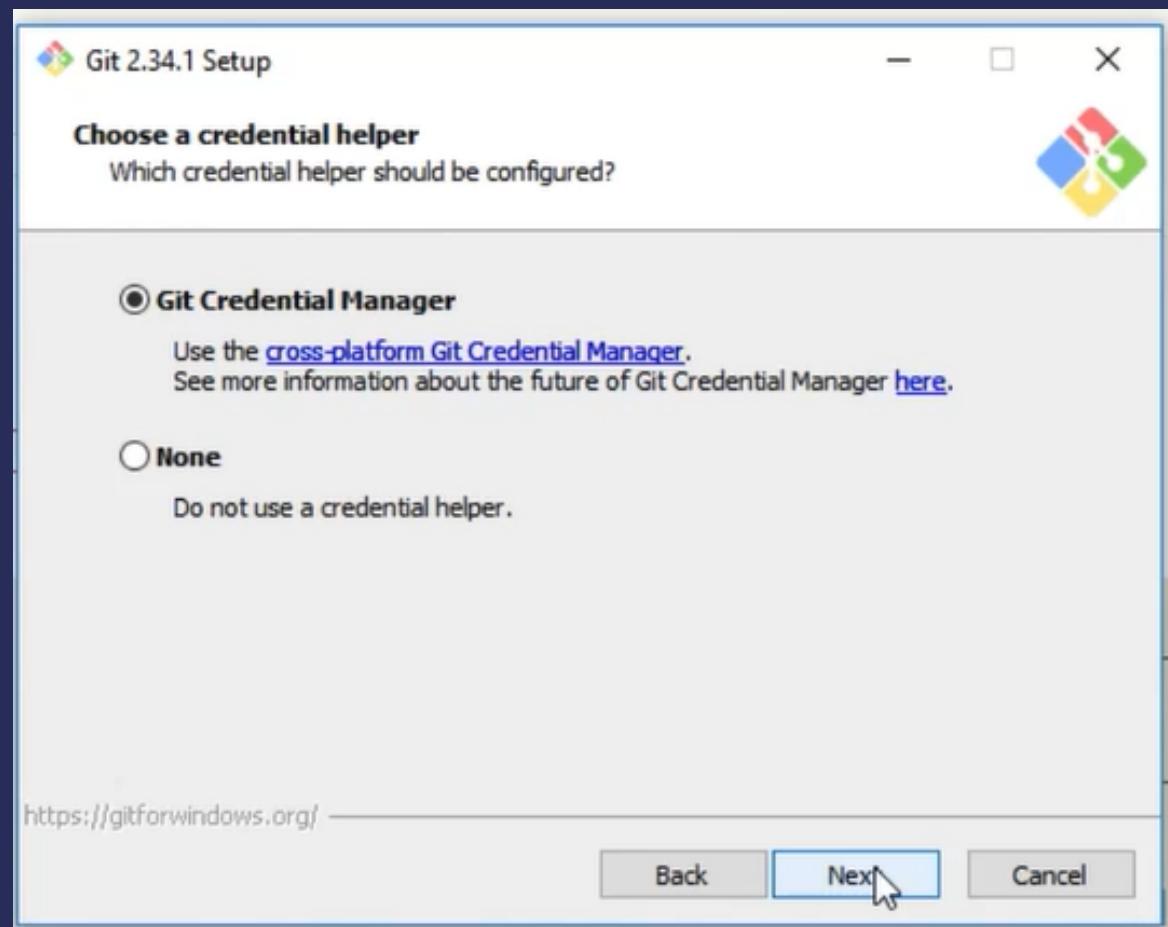
Check daily for Git for Windows updates

Current selection requires at least 292,9 MB of disk space.
<https://gitforwindows.org/>

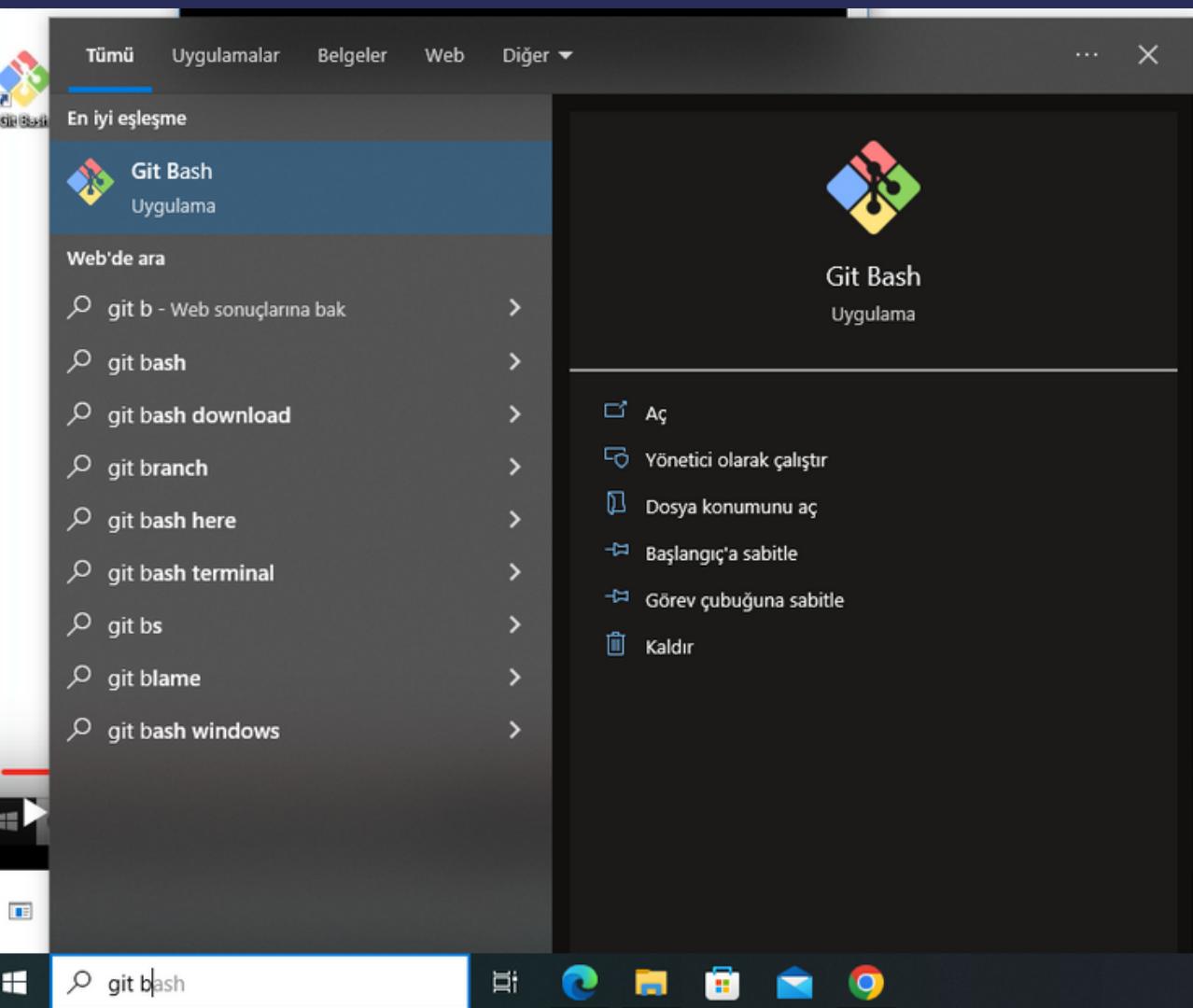
Back Next Cancel







Ve Kullanıma
Hazır..



GIT KOMUTLARI

```
margin: 0;
padding: 0;
font-size: sans-serif;
background: url("https://www.w3schools.com/html/images/html_box.png");
background-size: cover;
}
.box{
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    width: 400px;
    padding: 40px;
    background: #fff;
    box-sizing: border-box;
    box-shadow: 0 15px 30px 0 #ccc;
    border-radius: 10px;
}
.box h2{
    margin: 0 0 30px;
    padding: 0;
    color: #fff;
    text-align: center;
}
.box h3{
    margin: 0 0 10px;
    padding: 0;
    color: #fff;
    text-align: center;
}
.box .inputBox{
    position: relative;
    box-sizing: border-box;
    width: 100px;
    height: 40px;
    margin: 10px auto;
    border: 1px solid #ccc;
    border-radius: 5px;
    padding: 10px;
}
.box .inputBox input{
    width: 100px;
    height: 100%;
    border: none;
    outline: none;
    font-size: 14px;
    color: #333;
}
.box .inputBox label{
    position: absolute;
    top: 0;
    left: 0;
    width: 100px;
    height: 100%;
    background-color: #fff;
    border: 1px solid #ccc;
    border-radius: 5px;
    padding: 10px;
    font-size: 14px;
    color: #333;
    transform: rotate(-90deg);
    transform-origin: left top;
    pointer-events: none;
}
.box .inputBox input::placeholder{
    color: #333;
}
.box .inputBox input:focus{
    border: 1px solid #ccc;
}
.box .inputBox input:valid{
    border: 1px solid #ccc;
}
.box .inputBox input:invalid{
    border: 2px solid #ccc;
}
```

Awesome Web Browser ×



git config:

Bu komut, git etkinliklerinizle ilişkili bir yazar adı ve e-posta yapılandırmak için kullanılır.

- git config --global user.name "Kullanıcı Adı"
- git config --global user.email email@gmail.com

Awesome Web Browser ×



git init:

Mevcut bir klasörü Git deposu olarak başlatmak için kullanılan bir komuttur. Bu komut, Git'in kontrolünü altında olacak yeni bir boş Git deposu oluşturur. İlgili klasörde `.git` adında gizli bir klasör oluşturur ve Git'in depo ile ilgili tüm verilerini içerir.

- git init

Awesome Web Browser X

← → ⌂



:

git status

Çalışma klasöründeki dosyaların ve Git deposundaki durumlarının anlık bir görünümünü sağlar. Bu komutu kullandığınızda, hangi dosyaların değiştirildiğini, hangi dosyaların staging alanında olduğunu veya commit edilmeyi beklediğini ve hangi dosyaların Git tarafından takip edilmediğini görebilirsiniz.

- git status

Awesome Web Browser X

← → ⌂



:

git add:

Git'e çalışma klasöründeki dosyaları staging alanına eklemek için kullanılan bir komuttur. Staging alanı, bir sonraki commit işlemi için hazırlık yapmanızı sağlar ve hangi dosyaların commit'e dahil edileceğini belirlemenize olanak tanır.

- git add klasör adı (sadece belirli klasör)
- git add . (tüm değişiklikler)

Awesome Web Browser X

← → C



:

git commit:

Geçiş bölgesinde yer alan değişiklikleri depoya bildirmek için commit komutu kullanılır. Önce add, sonra commit komutunun kullanılmasıyla proje klasöründeki tüm değişiklikler depoya bildirilmiş yanı kopyalanmış, yedeği alınmış olur.

- git commit -m "Açıklama"

Awesome Web Browser X

← → C



:

git log:

Git deposundaki commit geçmişini görüntülemek için kullanılan bir komuttur. Bu komut, projedeki yapılan commitleri listeleyerek her commit için ayrıntılı bilgileri gösterir.

- git log

```
Awesome Web Browser ×  
← → ⌂ ⌂ :  
  
git clone:  
Git versiyon kontrol sistemindeki  
bir depoyu yerel bir bilgisayara  
kopyalamak için kullanılan bir  
komuttur.  
  
• git clone <repo_url>  
• git clone  
https://github.com/kullanici_adi/proj  
e.git
```

```
Awesome Web Browser ×  
← → ⌂ ⌂ :  
  
git diff:  
Diff komutu proje klasörü ile geçiş  
bölgesinde yer alan dosyalardaki  
farklılıklarını listeler. Diyelim ki  
commit işlemi yaptıktan sonra bazı  
dosyalarda kod yazmaya devam ettiniz.  
İşte aşağıdaki komutu çalıştırırsanız  
proje klasörünüzde yer alan dosyalar  
ile geçiş bölgesinde gönderdiğiniz  
dosyalar arasındaki kod farklılıklarını  
görebilirsiniz.  
  
• git diff
```

Awesome Web Browser X

← → ⌂



:

git reset: Git deposundaki geçmişi değiştirmek veya geri almak için kullanılan bir komuttur. Bu komutla, commitleri geri alabilir, değişiklikleri iptal edebilir veya dosyaları geri yükleyebilirsiniz.

- git reset --soft <commit>
- git reset --mixed <commit>
- git reset --hard <commit>

Awesome Web Browser X

← → ⌂



:

git rm:

Git deposundaki dosyaları silmek için kullanılan bir komuttur. Bu komutla, belirtilen dosyayı hem çalışma dizininden hem de Git deposundan kaldırabilirsiniz.

- git rm <dosya>

git show

Bu komut, herhangi bir git nesnesi hakkında bilgi görüntülemek için kullanılır.

Kullanımı: git show [commit id]

git tag

Bu komut; etiketleme, belirli taahhütleri basit kısımlara işaretlemek için kullanılır.

Kullanımı: git tag [commit id]

git checkout

git checkout komutu, dal oluşturmak veya dallar arasında geçiş yapmak için kullanılır.
Kullanımı: git checkout [branch name]

git merge

Bu komut, belirtilen dalın geçmişini geçerli dalla birleştirir
Kullanımı: git merge [branch name]

git pull

Bu komut, uzak depoda bulunan tüm değişiklikleri yerel çalışma dizinine birleştirmek için kullanılır.

Kullanımı: `git pull [variable name] [remote repository name]`

git stash

Az bilinen temel git komutlarından biridir. Hemen işlenmeyecek değişiklikleri geçici olarak kaydetmeye yardımcı olur.

git stash: Değiştirilen tüm izlenen dosyaları geçici olarak kaydetmek için kullanılır.

git stash list: Tüm öğe listelerini saklamak için kullanılır.

git remote

Bu komut, yerel depoyu uzak sunucuya bağlamak için kullanılır.

Kullanımı: git remote add [variable name] [Uzak Sunucu Linki]

git push

Bu komut; basit bir itme ile, yapılan değişiklikleri çalışma diziniyle ilişkili uzak dizinin ana dalına gönderir.

Kullanımı: git push [variable name] [remote repository name]

git branch

git branch komutu; dalları listelemek, oluşturmak ya da silmek için kullanılabilir.

1-git branch: Dizinde bulunan tüm dalları listelemek için kullanılır:

2-git branch <branch-adı>: Yeni bir branch oluşturmak için kullanılır.

3-`git branch -d <branch-adı>`: Var olan bir branch'i silmek için kullanılır.

git fetch

Bu komut, bir kullanıcının bu nesneleri şu anda yerel çalışma dizininde bulunmayan uzak dizinden almasına izin verir.

Kullanımı: git fetch origin

git ls-tree

Bir ağaç nesnesini, her maddenin adı ve modu ile blob'un SHA-1 değerini birlikte görüntülemek için git ls-tree komutunu kullanın.

Örneğin:

```
git ls-tree HEAD
```

git catfile

SHA-1 değeri ile, git cat-file komutunu kullanarak bir nesnenin türünü görüntüleyin.

Örneğin:

```
git cat-file -p  
d670460b4b4aece5915caf5c68d12f560a9fe3e4
```

git grep

kullanıcıların içerik ağaçları üzerinden cümleler ve/veya kelimeler bulmalarını sağlar. Örneğin, tüm dosyalarda

`www.hostinger.com` aramak için şu kullanılır:

`git grep "www.hostinger.com"`

gitk

`gitk` komutu yerel bir dizin için grafiksel arayüzdür.

git instaweb

git instaweb komutuyla bir web sunucusu yerel depo ile arabirimde çalıştırılabilir. Bir web tarayıcı da kendisine otomatik olarak yönlendirilir. Örneğin:
git instaweb -httpd=webrick

git gc

Depoyu gereksiz dosyaları temizleyerek, çöp toplama yoluyla optimize etmek için aşağıdakini kullanın

git archive

git archive komutu, bir kullanıcının tek bir dizin ağacının bileşenlerini içeren bir zip veya tar dosyası oluşturmasını sağlar. Örneğin:

```
git archive --format=tar  
master
```

git prune

git prune komutu aracılığıyla, gelen işaretçilere sahip olmayan nesneler silin

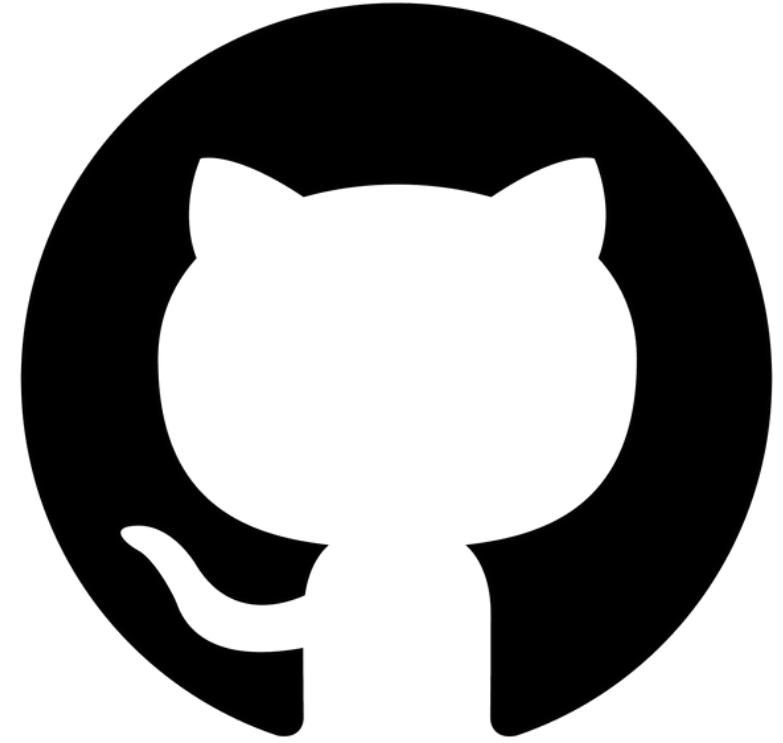
git fsck

GIT dosya sisteminin bütünlüğünü kontrol etmek için ***git fsck*** komutunu kullanın.

Bozuk nesneler tanımlanır

git rebase

git rebase komutu, başka bir şubedeki taahütlerin tekrar uygulanması için kullanılır. Örneğin: ***git rebase master***



GitHub

GitHub Nedir?



Github, dünyanın en büyük geliştirici topluluklarından birisi olup "git versiyon kontrol sistemini" kullanarak yazılım geliştirme projeleri için web tabanlı bir bulut depolama servisidir.

Ayrıca, Github yazılım geliştiricileri için bir sosyal ağ platformudur. Github sayesinde, yazılım geliştiriciler, kendileri gibi yazılımla uğraşan kişilerin projelerine göz atabilir onları takip edebilirler.



GitHub Ne İşe Varar?



GitHub servisi sayesinde aynı anda pek çok kişiden oluşan kalabalık yazılım geliştirici ekipler aynı proje üzerinde çalışmalar olmadan çalışabilir ve ayrı değişiklikler yapabilir. Bu servis üzerinden geliştirilen projelerde ekip büyük bir zaman kazanır.



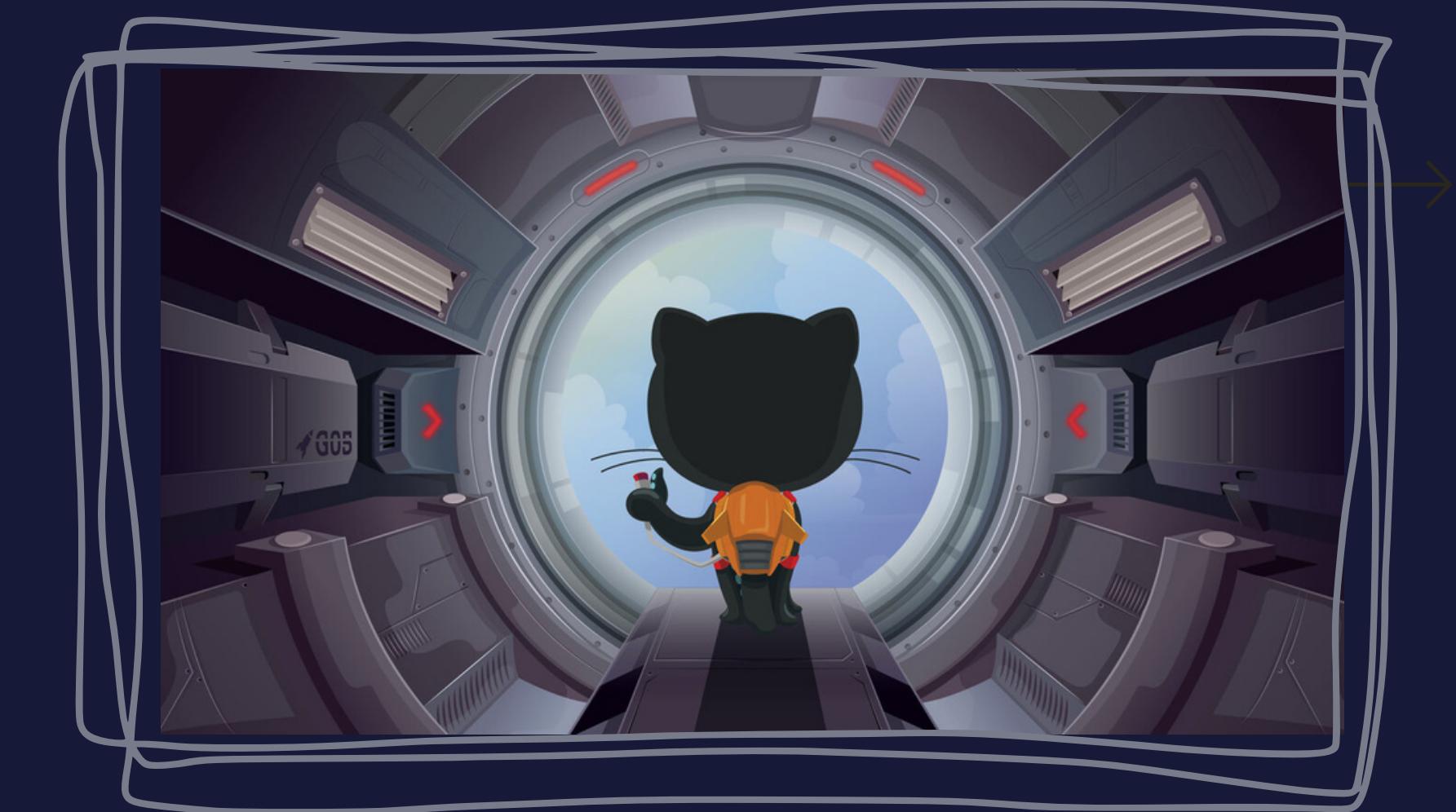
GitHub, yazılımı hazırladıkları müşterinin de kullanabileceği bir alan sunar. Örneğin, ekip yazılımı hazırlayıp sunduktan sonra müşteri eğer isterse GitHub servisi üzerinden gerekli düzenlemeleri ve güncellemeleri yaparak iletişim sürecini hızlandırabilir. Yapılan değişiklikler onaydan geçmeden ana projeye dahil edilmediği için herkes kendi düzenlemesini yaparak fikrini ortaya koyabilir.





GİTHUB
NASIL
KULLANILIR?

GitHub Nasıl Kullanılır?



Kullanışlı bir ara yüze sahip olan GitHub'a bir kullanıcı adı ve e-mail ile üye olmanız gerekiyor. Bu adresten ana sayfaya giderek kendinize bir kullanıcı adı oluşturun. Eğer projelerinizi GitHub'ta açık bir şekilde tutarsınız sistemi kullanmak ücretsizdir. Private (gizli) projeler için ise ücretli bir plan oluşturmak mümkün.

Sisteme ilk defa
giriysanız github-
learning-lab denen
bir eğitim aracına
yönlendiriliyorsunuz
ve burada bir bot
yardımıyla sistemin
nasıl çalıştığını adım
adım
öğrenebiliyorsunuz.

The screenshot shows a GitHub issue page with the following details:

- Title:** GitHub'a Başlarken
- Status:** Açık (Open)
- Assignee:** github-learning-lab | bot
- Created:** 2 saat önce (2 hours ago)
- Comments:** 1 yorum (comment)

GitHub nedir?

Sorduğuna sevindim Birçok kişi açık kaynağa katkıda bulunmak istedikleri için GitHub'a geliyor projeleri veya projeleri için kullanan ekip arkadaşları veya sınıf arkadaşları tarafından davet ediliyorlar. İnsanlar neden bu projeler için GitHub kullanıyor?

GitHub, temelinde bir işbirliği platformudur.

Yazılımdan yasal belgelere kadar, ekibinizin ihtiyaç duyduğu işbirliği ve güvenlik araçlarıyla en iyi çalışmanızı yapmanıza yardımcı olması için GitHub'a güvenebilirsiniz. GitHub ile projeleri tamamen gizli tutabilir, dünyayı işbirliğine davet edebilir ve projenizin her adımını düzene sokabilirsiniz.

GitHub ayrıca güçlü bir sürüm kontrol aracıdır.

GitHub Git kullanır her katkıyı ve katılımcıyı izlemek için en popüler açık kaynaklı sürüm kontrol yazılımı projenize - böylece her kod satırının tam olarak nereden geldiğini bilirsiniz.

GitHub, insanların daha fazlasını yapmasına yardımcı olur.

GitHub, dünyadaki en gelişmiş teknolojilerden bazılarını oluşturmak için kullanılır. İster verileri görselleştiriyor, ister yeni bir oyun oluşturuyor olun, GitHub'da sizi bir sonraki adıma götürecek koca bir topluluk ve bir dizi araç var. Bu kurs temel bilgilerle başlar, ancak geri kalanını daha sonra inceleyeceğiz!

Video: GitHub nedir?

▶ GitHub havuzunu keşfetme

Sorunları kullanma

Bu bir sorun kodunuzdaki hatalar, kod incelemesi ve hemen hemen her şey hakkında konuşabileceğiniz bir yer.

Sorun başlıkları, e-posta konu satırları gibidir. Ortak çalışanlarınıza sorunun ne hakkında olduğunu bir bakışta anlatırlar. Örneğin, bu sorunun başlığı GitHub'a Başlarken.

▶ GitHub Sorunlarını Kullanma

▶ Bildirimleri yönetme

Kilometre taşı
Kilometre taşı yok

Bağılı çekme istekleri
Bir çekme isteğini başarıyla birleştirmek bu sorunu kapatabilir.

Henüz yok

Bildirimler **Özelleştir**
Aboneliği iptal et

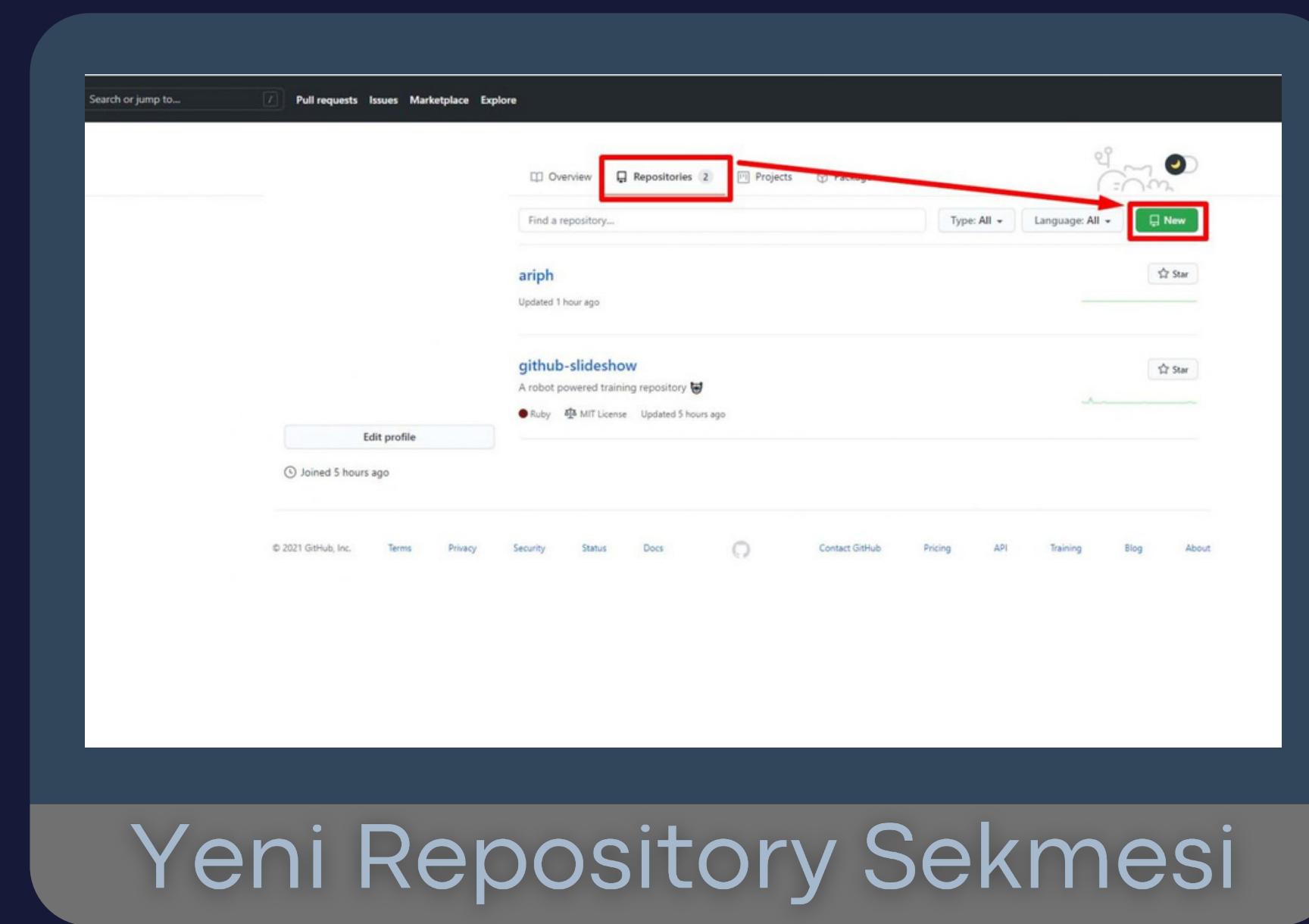
Bu depoyu izlediğiniz için bildirim alıyzsunuz.

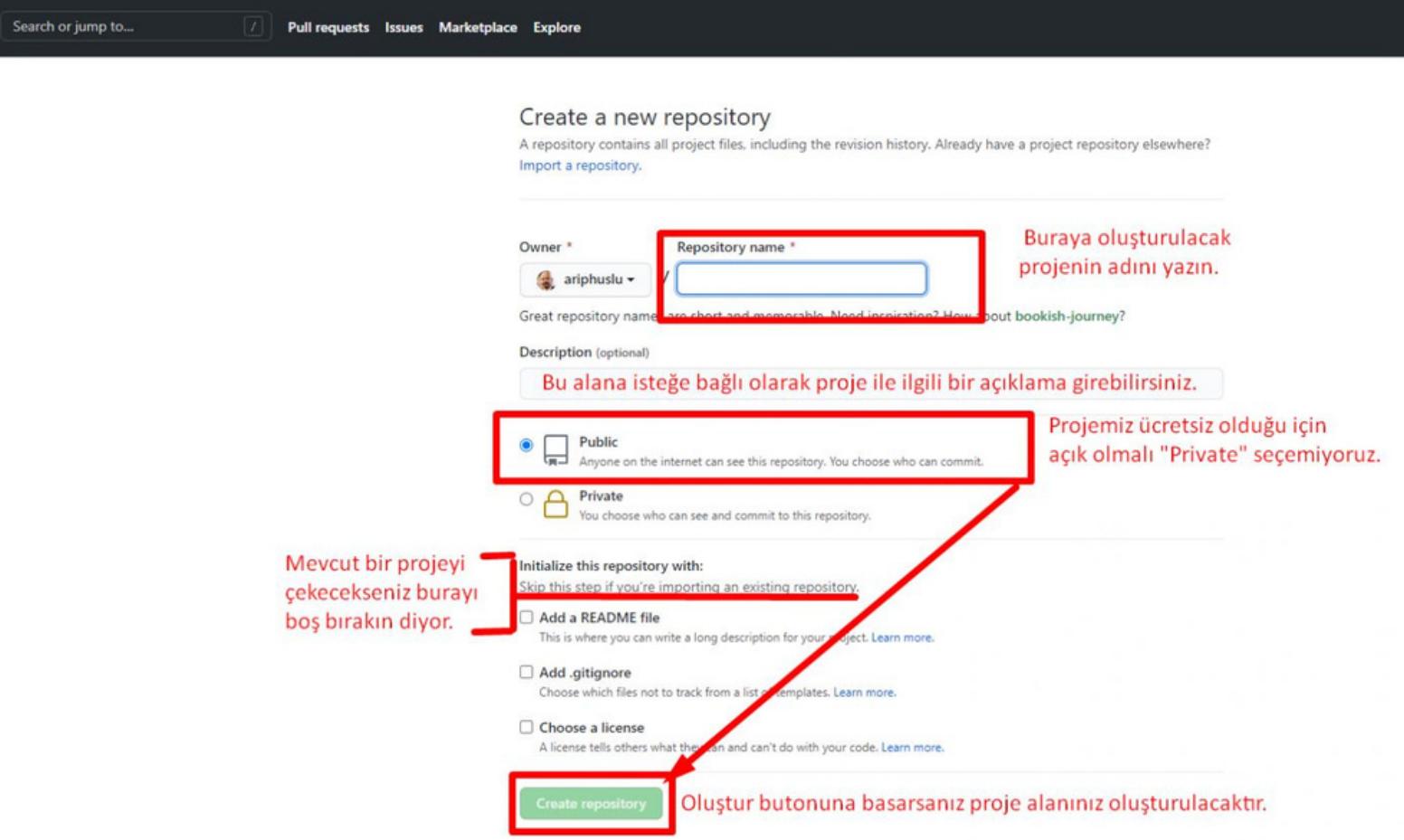
0 katılımcı

Görüsmeyi kapat
Pin sorunu
Aktarım sorunu
Sorunu sil

YENİ REPO (REPOSITORY) OLUŞTURMAK:

GitHub'ta bir proje oluşturmak çok basittir. Yeni bir repository oluşturmak çok kolaydır. Ana sayfada profilinize gelerek “Repositories” sekmesine tıklayarak sağ üstte “New” butonuna basmalısınız. Yeni açılan sayfada sizin kullanıcı adınızın altında bir dizin şeklinde oluşacak bir “Repository” eklemeniz gerekiyor.





Nasıl yeni repository oluşturulur?

Oluşturduğunuz bu çalışma alanınıza bir URL tanımlayacaktır. Bu URL adresine Git terminali üzerinden ulaşarak dizinden dosyaları alabilir veya değiştirebilirsiniz.

Program geliştiricisi ve farklı programlama dillerinde kod yazarı iseniz burada depoladığınız koda istediğiniz an ve istediğiniz yerden ulaşabilir ve oluşturduğunuz versiyonlarla çalışmanızı takip edebilirsiniz. Farklı programlarla entegrasyonu olmakla birlikte Git uygulaması ile kod yükleyebilirsiniz.



**SIK KULLANILAN
GITHUB
TERİMLERİ**



Git

REPOSITORY

Git, bilgisayarınıza yüklenen bir programdır. Windows, MacOs ve Linux sistemlerinde çalışır. Git terminali ile GitHub ve benzeri depo sistemlerine belirli komutlar kullanarak dosya yüklemek, bilgisayarınızda bir çalışma alanı oluşturmak ve bu projeyi GitHub kullanıcı adınıza bağlı bir dizine yüklemek gibi basit veya çok daha kompleks işlemler yapılabilir.

GitHub kullanıcı adınızın altında açacağınız bir dosya dizini ve depodur. İstediğiniz bir isimle adlandırıp projenizi bu depoda kaydedebilirsiniz. Gizli veya diğer kullanıcılarla açık repo oluşturabilirsiniz.

BRANCH

Projeyi bir doğru şekilde düşünelim, ana gövde oluşturuktan sonra, çalışmanın herhangi bir yerinde bir branch (dal, şube) oluşturabilir ve çalışmanızın o noktadan sonraki kısmını siz veya başkasının o branch üzerinden yürütmesini sağlayabilirsiniz.



GitHub Branch ve Master İlişkisi

ORIGIN

Çalışmanın ana dalı Origin (kaynak) olarak ifade edilir. Aktif olan bir branch da origin olabilir. İlk oluşturulan Branch, Master olarak ifade edilir, Origin de genelde ilk oluşturulan Branch'tır.

-

FORK

Başkasının oluşturduğu bir "Repository" yi, yani projeyi kendi hesabınıza kopyalamak çekmektir. Başkasının projesini alıp üzerinde çalışarak kendini istediğiniz hale getirebilirsiniz. Açık kaynak kodlu sistemlerde programa ekleyeceğiniz kodlar ile sizin kabiliyetlerinize bağlı olarak çok farklı özellikler kazandırılabilir.

CLONE

Clone GitHub bulut sisteminde bulunan belirli bir program veya kod dizisini kopyalayıp, bilgisayarınıza veya hesabınıza indirmeye yarar.

COMMIT

- GitHub üzerinden çalışma dosyalarınızı aldınız ve üzerinde çalışmaya başladınız. Projeye eklediğiniz bu yeni kodları da içeren çalışmanın GitHub' taki kopyasına da iletilmesi için öncelikle “Commit” işlemini gerçekleştirmeniz gerek. Bu işlev çalışmanızın GitHub bulut sistemine gönderilmek üzere paketlenmesini sağlar.

IGNORE

Commit işlemi yaparken çalıştığınız bazı dosyaların paketlenip gönderilmesini istemeyebilirsiniz. Özette pakete dahil olması istenmeyen dosyalar, ignore edilerek git tarafından göz ardı edilmesi sağlanır.

- 卷之三

PUSH

Commit işlemiyle
paketlenen yeni
çalışmanın GitHub
sunucusuna Push edilmesi
(ıtılmesi) yanı
gönderilmesi işlemini ifade
eder.

PULL

Başkalarınca
yapılan
değişikliklerin kendi
bilgisayarınız veya
sunucudaki ana
dosya versiyonuna
dahil etmek
(çekmek) işlemidir.

-

ISSUES

Türkçesi durum olarak adlandırılabilen issues kelimesi ile bir durum, sorun veya gelişme ile ilgili bilgilendirme açmak denebilir. Projeye dahil olanlar issues açıldığı zaman isterse konuya dahil olup durumun gerektirdiği işlemi yapar ve issues kapatılabilir.

MERGE

Merge bir catalda (Branch) ta yapılan bir değişikliğin diğer şubelerde de kullanılır hale gelmesi için birleştirilmesi işlemidir. Bu işlem Pull İsteği veya birkaç kod yardımıyla yapılabilir.

CONFLICT

Birkaç farklı Branch'ta yapılan çalışmalar paketlenip gönderilirken bazen iki dosyanın birbirleriyle çelişmesiyle sonuçlanabilir. Bu olaya conflict denir.

The

END

ZEYNEP TEKİN
FATMA HÜMEYRA GÜL
KEVSER SEMİZ

KAYNAKÇA

- <https://www.hostinger.web.tr/rehberler/github-kullanimi-basit-git-komutlari>
- <https://www.hosting.com.tr/blog/en-cok-kullanilan-git-komutlari/>
- <https://www.hosting.com.tr/blog/github-nedir/>
- <https://www.hostinger.web.tr/rehberler/github-kullanimi-basit-git-komutlari>
- <https://bidb.itu.edu.tr/seyir-defteri/blog/2019/02/13/git>
- <https://blog.protein.tech/yeni-ba%C5%9Flayanlar-i%C8%A7in-git-b%C3%B6l%C3%BCm-1-b95762b8d975>
- <https://www.argenova.com.tr/en-cok-kullanilan-git-komutlari-ornekli>
- <https://www.btkakademi.gov.tr/portal/course/player/deliver/versiyon-kontrolleri-git-ve-github-19439>
- <https://stk.bilgi.edu.tr/media/uploads/2015/01/12/git101.pdf>

