

GitHub Actions Kullanımı - İş Akışını Ayarlama



Gerekli tahmini süre: 30 dakika

GitHub Actions Kullanımı - İş Akışını Ayarlama için uygulamalı laboratuvara hoş geldiniz. Bu bölümde, GitHub Actions kullanarak bir GitHub deposunda bir iş akışı oluşturacaksınız. Adım 1'de boş bir iş akışı dosyası oluşturacak ve sonraki adımlarda olaylar ve bir iş çalıştırıcı ekleyeceksiniz. Ardından, iş akışını **GitHub Actions Kullanımı - Bölüm 2** adlı bir sonraki laboratuvarla tamamlayacaksınız. Bölüm 2'ye başlamadan önce bu laboratuvarı tamamen tamamladığınızdan emin olun.

Öğrenme Hedefleri

Bu laboratuvari tamamladıktan sonra şunları yapabileceksiniz:

- CI boru hattınızı çalıştırmak için bir GitHub iş akışı oluşturun
- İş akışını tetiklemek için olaylar ekleyin
- İş akışına bir iş ekleyin
- İşe bir iş çalıştırıcı ekleyin
- İş çalıştırıcısına bir konteyner ekleyin

Ön Koşullar

Bu laboratuvardaki alıştırmaları tamamlamak için aşağıdakilere ihtiyacınız olacak:

- YAML hakkında temel bir anlayış
- Bir GitHub hesabı
- CLI'lar hakkında orta düzey bilgi

GitHub Kişisel Erişim Token'ı Oluştur

Laboratuvara başlamadan önce biraz hazırlık yapmanız gerekiyor.

Kişisel Erişim Token'ı Oluştur

Bu laboratuvar sırasında `gh` CLI aracını kullanarak bir repo fork'layacak ve klonlayacaksınız. Ayrıca, bu laboratuvarın sonunda klonladığınız repoya değişiklikler göndereceksiniz. Bu, GitHub ile bir `kişisel erişim token`'ı kullanarak kimlik doğrulamanızı gerektirir. Bu token'ı oluşturmak ve daha sonra kullanmak üzere kaydetmek için buradaki adımları takip edin:

- Hesabınızın [GitHub Ayarları](#) sayfasına gidin.

2. Kişisel erişim token'ı oluşturmak için **Yeni token oluştur** butonuna tıklayın.

The screenshot shows the GitHub 'Developer settings' page under 'Personal access tokens'. A red box highlights the 'Personal access tokens' button. To the right, a token card is displayed with the title 'Personal access token' and a note about its use for GitHub Actions. The token itself has been redacted.

3. Token'ınıza açıklayıcı bir isim verin ve istege bağlı olarak son kullanma tarihini değiştirin.

The screenshot shows the 'New personal access token' creation form. A red box highlights the 'Note' section containing the text 'GitHub Actions Lab'. Below it, a question 'What's this token for?' is shown. Another red box highlights the 'Expiration *' dropdown, which is set to '30 days'. A note below the dropdown states 'The token will expire on Sat, Sep 10 2022'.

4. Bu laboratuvar için gereken minimum izinleri seçin: repo, read:org ve workflow.

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scope](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:org	Full control of orgs and teams, read and write orgs
<input type="checkbox"/> write:org	Read and write org and team membership, read org projects
<input checked="" type="checkbox"/> read:org	Read org and team membership, read org projects

5. **Token oluştur** butonuna tıklayın.

<input type="checkbox"/> project	Full control of projects
<input type="checkbox"/> read:project	Read access of projects
<input type="checkbox"/> admin:gpg_key	Full control of public user GPG keys
<input type="checkbox"/> write:gpg_key	Write public user GPG keys
<input type="checkbox"/> read:gpg_key	Read public user GPG keys
 Generate token	Cancel

6. Token'ı kopyaladığınızdan ve güvenli bir yere yapıştırıldığınızdan emin olun, çünkü bir sonraki adımda buna ihtiyacınız olacak. **UYARI: Bir daha göremeyecəksiniz.**

Personal access tokens

Tokens you have generated that can be used to access the GitHub API.

Make sure to copy your personal access token now. You won't be able to see it again!

✓ ghp_H

3eyDj8 

github actions course — *read:org, repo, workflow*

Expires on **Fri, Sep 9 2022**.

Personal access tokens function like ordinary OAuth access tokens. They can be used to **authenticate to the API over Basic Authentication**.

Uyarı: Token'larınızı güvende tutun ve şifreler gibi koruyun.

Bu token'ı herhangi bir zamanda kaybederseniz, yukarıdaki adımları tekrarlayarak token'i yeniden oluşturun.

Depoyu Fork'la ve Klonla

Bir Terminal Aç

Editördeki menüyü kullanarak bir terminal penceresi açın: Terminal > Yeni Terminal.

Terminalde, eğer /home/project klasöründe değilseniz, şimdi proje klasörünüze geçin.

```
cd /home/project
```

GitHub ile Kimlik Doğrulama

Öncelikle, GitHub CLI'yi kurmak için aşağıdaki komutları çalıştıralım.

```
sudo apt update  
sudo apt install gh
```

Sonra, terminalde GitHub ile kimlik doğrulamak için aşağıdaki komutu çalıştırın. Önceki adımda oluşturduğunuz GitHub Personal Token'a ihtiyacınız olacak.

```
gh auth login
```

Sizi burada gösterildiği gibi rehberli bir deneyime alacağız:

```
Hangi hesaba giriş yapmak istersiniz? GitHub.com  
Git işlemleri için tercih ettiğiniz protokol nedir? HTTPS  
Git'i GitHub kimlik bilgilerinize doğrulayın.  
GitHub CLI'yi nasıl doğrulamak istersiniz? Bir kimlik doğrulama jetonu yapıştırın.  
Kimlik doğrulama jetonunuzu yapıştırın: *****  
Hesabınız kullanıcısı olarak GitHub'a giriş yapacaksınız.
```

Başarıyla doğruladıktan sonra, terminalde [bu GitHub](#) reposunu fork edip klonlamanz gerekecek. Ardından, depodaki fork edilmiş versiyonunuzda GitHub Actions'ı tetiklemek için bir iş akışı oluşturacaksınız.

Referans Repo'yu Fork ve Klonla

```
gh repo fork ibm-developer-skills-network/wtecc-CICD_PracticeCode
```

Not Komutu çalıştırduğumızda, sizi fork'u klonlamaya yönlendirecektir. Devam etmek için Evet yazın.

Çıktımız aşağıdaki resme benzer görünmelidir:

```
theia@theia-ritikaj:/home/project$ gh repo fork ibm-developer-skills-network/wtecc-CICD_PracticeCode  
✓ Created fork [REDACTED]/wtecc-CICD_PracticeCode  
? Would you like to clone the fork? Yes  
Cloning into 'wtecc-CICD_PracticeCode'...  
remote: Enumerating objects: 139, done.  
remote: Total 139 (delta 0), reused 0 (delta 0), pack-reused 139 (from 1)  
Receiving objects: 100% (139/139), 39.43 KiB | 5.63 MiB/s, done.  
Resolving deltas: 100% (51/51), done.  
Updating upstream  
From https://github.com/ibm-developer-skills-network/wtecc-CICD_PracticeCode  
* [new branch]      Adding-github-actions -> upstream/Adding-github-actions  
* [new branch]      main                  -> upstream/main  
✓ Cloned fork
```

Önemli: Pull Request

Pull request yaparken, isteğinizin fork'unuzla birleştirdiğinden emin olun çünkü bir fork'un pull request'i varsayılan olarak [bu](#) repoya geri dönecektir, fork'unuza değil.

Laboratuvar Klasörüne Geçin

Depoyu klonladıkten sonra, wtecc-CICD_PracticeCode adındaki dizine geçin.

```
cd wtecc-CICD_PracticeCode
```

Bu dizinin içeriğini listeleyerek bu laboratuvar için eserleri görebilirsiniz.

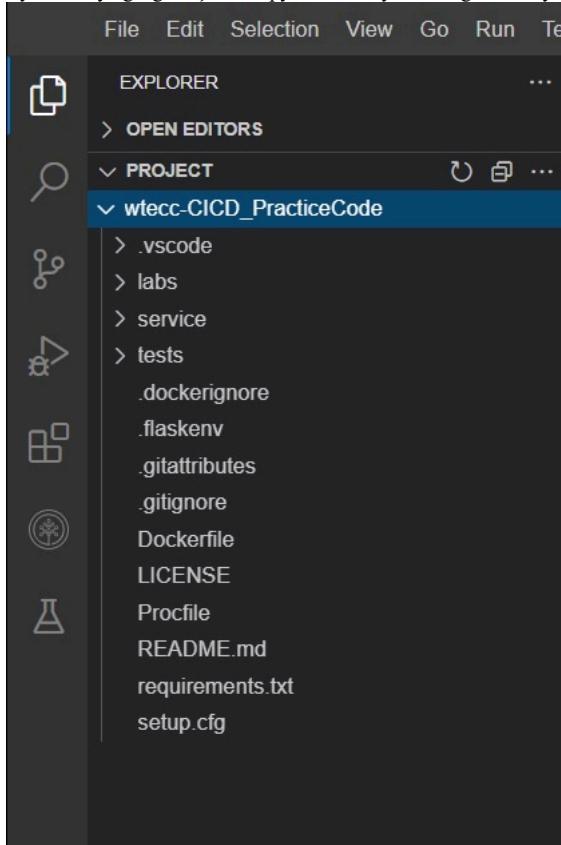
```
ls -l
```

Dizin aşağıdaki listeye benzer görünmelidir:

```
Problems      theia@theia-captainfedo1: /home/project/wtecc-CICD_PracticeCode ×

theia@theia-captainfedo1:/home/project/wtecc-CICD_PracticeCode$ ls -l
total 44
-rw-r--r-- 1 theia users   491 Aug  8 23:34 Dockerfile
drwxr-sr-x 8 theia users  4096 Aug  8 23:34 labs
-rw-r--r-- 1 theia users 11357 Aug  8 23:34 LICENSE
-rw-r--r-- 1 theia users    72 Aug  8 23:34 Procfile
-rw-r--r-- 1 theia users   915 Aug  8 23:34 README.md
-rw-r--r-- 1 theia users   327 Aug  8 23:34 requirements.txt
drwxr-sr-x 3 theia users  4096 Aug  8 23:34 service
-rw-r--r-- 1 theia users   331 Aug  8 23:34 setup.cfg
drwxr-sr-x 2 theia users  4096 Aug  8 23:34 tests
```

Ayrıca dosya gezgini içinde kopyalanan dosyaları da görüntüleyebilirsiniz.



Artık laboratuvara başlamak için hazırlısınız.

İsteğe Bağlı

Terminalde çalışmak zorlaşırsa çünkü komut istemi çok uzunsa, aşağıdaki komutu kullanarak istemi kısaltabilirsiniz:

```
export PS1="\[\033[01;32m\]\u\[033[00m\]: \[\033[01;34m\]\w\[033[00m\]]\$ "
```

Adım 1: Bir İş Akışı Oluşturun

Başlamak için, bir iş akışı yaml dosyası oluşturmanız gerekiyor. Bu dosyadaki ilk satır, depo GitHub Actions sayfasında görünen iş akışının adını tanımlayacaktır.

Göreviniz

- Terminali açın ve `wtecc-CICD_PracticeCode` dizininde olduğunuzdan emin olun.

▼ İpucu için buraya tıklayın.

```
cd /home/project/wtecc-CICD_PracticeCode
```

- `.github/workflows` dizin yapısını oluşturun ve `workflow.yml` adında bir dosya oluşturun.

▼ İpucu için buraya tıklayın.

```
mkdir -p .github/workflows  
touch .github/workflows/workflow.yml
```

- Her iş akışı bir isimle başlar. İsim, Actions sayfasında ve herhangi bir rozet üzerinde görüntülenecektir. Dosyanın ilk satırına `name:` etiketi ekleyerek iş akışınıza `CI workflow` adını verin.

▼ İpucu için buraya tıklayın.

```
name: {insert name here}
```

[Open workflow.yml in IDE](#)

Yaptığınızın aşağıdaki çözümle eşleştiğinden emin olun.

Çözüm

▼ Yanıt için buraya tıklayın.

Aşağıdaki kod parçasıyla `workflow.yml` dosyasını değiştirebilirsiniz. Ayrıca kodun ilgili kısımlarını kopyalayabilirsiniz. Doğru şekilde girinti yaptığınızdan emin olun:

```
name: CI workflow
```

Adım 2: Olay Tetikleyicileri Ekle

Olay tetikleyicileri, iş akışını çalıştırabilecek olayları tanımlar. Aşağıdaki olayları eklemek için `on:` etiketini kullanacaksınız:

- Ana dalda her itme için iş akışını çalıştırın.
- Ana dalda bir çekme isteği oluşturulduğunda iş akışını çalıştırın.

Göreviniz

1. name: ile aynı girinti seviyesinde iş akışına on: anahtar kelimesini ekleyin.

▼ İpucu için buraya tıklayın.

on:

2. İş akışını tetikleyebilecek ilk olay olarak push: olayını ekleyin. Bu, on:'in alt ögesi olarak eklendiği için altında girintili olmalıdır.

▼ İpucu için buraya tıklayın.

on:

{ilk olay adını buraya ekleyin}:

3. İtme olayına "main" dalını ekleyin. İş akışının, birisi ana dala her itme yaptığında başlamasını istiyorsunuz. Bu, birleştirme olaylarını da içerir. Bunu, branches: anahtar kelimesini kullanarak ve bir dizi dal ile ya [] ya da - şeklinde yaparsınız.

▼ İpucu için buraya tıklayın.

on:

push:

branches: [{dal adını buraya ekleyin}]

4. Ana daldaki kullanıcı bir çekme isteği yaptığında tetiklenecek şekilde, az önce tamamladığınız itme olayıyla benzer bir pull_request: olayı ekleyin.

▼ İpucu için buraya tıklayın.

on:

push:

branches: ["main"]

{ikinci olay adını buraya ekleyin}:

branches: [{dal adını buraya ekleyin}]

Yaptığınız işi aşağıdaki çözümle karşılaştırarak kontrol edin.

Çözüm

▼ Yanıt için buraya tıklayın.

workflow.yml dosyasını aşağıdaki kod parçasıyla değiştirin. İlgili kod kısımlarını da kopyalayabilirsiniz. Doğru bir şekilde girintilediğinizden emin olun:

```
name: CI workflow
on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]
```

Adım 3: Bir İş Ekleyin

Artık iş akışı dosyasına build adında bir iş ekleyeceksiniz. Bu iş ubuntu-latest çalıştırıcısında çalışacak. Unutmayın, bir iş, önceki adımda eklediğiniz olaylar üzerinde çalışan adımların bir koleksiyonudur.

Göreviniz

1. Öncelikle bir işe ihtiyacınız var. name ile aynı girinti seviyesinde jobs: bölümünü iş akışına ekleyin (yani, girinti yok).

- ▼ İpucu için buraya tıklayın.

```
jobs:
```

2. Sonra, işe bir isim vermeniz gerekiyor. jobs: bölümünün altına yeni bir satır ekleyerek işinize build: adını verin.

- ▼ İpucu için buraya tıklayın.

```
jobs:  
  {iş adı buraya ekleyin}:
```

3. Son olarak, bir çalıştırıcıya ihtiyacınız var. GitHub Actions'a bu iş için ubuntu-latest çalıştırıcısını kullanmasını söyleyin. Bunu runs-on: anahtar kelimesini kullanarak yapabilirsiniz.

- ▼ İpucu için buraya tıklayın.

```
jobs:  
  build:  
    runs-on: {çalıştırıcı adı buraya ekleyin}
```

Yaptığınız işin aşağıdaki çözümle eşleştiğinden emin olun.

Çözüm

▼ Cevap için buraya tıklayın.

Aşağıdaki kod parçasıyla workflow.yml dosyasını değiştirin. İlgili kod parçalarını da kopyalayabilirsiniz. Girintileri düzgün bir şekilde ayarladığınızdan emin olun:

```
name: CI workflow  
on:  
  push:  
    branches: [ "main" ]  
  pull_request:  
    branches: [ "main" ]  
jobs:  
  build:  
    runs-on: ubuntu-latest
```

Adım 4: Hedef Python 3.9

Tüm geliştirme aşamalarında, CI boru hattı dahil olmak üzere, bağımlılıkların ve işletim sisteminin aynı sürümünü tutarlı bir şekilde kullanmak önemlidir. Bu proje Python 3.9 üzerinde geliştirilmiştir, bu nedenle CI boru hattının da aynı Python sürümünde çalıştığını emin olmalısınız. Bunu, iş akışınızı GitHub eylemi içinde bir konteynerde çalıştırarak başarácaksınız.

Göreviniz

1. Build işinin runs-on: bölümünün altına bir container: bölümü ekleyin ve GitHub Actions'a python:3.9-slim görüntüsünü kullanmasını söyleyin.

İpucu

▼ İpucu için buraya tıklayın.

```
jobs:  
  build:  
    runs-on: ubuntu-latest  
    container: {insert container name here}
```

Yaptığınız işin aşağıdaki çözümle eşleştiğinden emin olun.

Çözüm

▼ Cevap için buraya tıklayın.

workflow.yml dosyasını aşağıdaki kod parçasıyla değiştirin. Kodun ilgili kısımlarını da kopyalayabilirsiniz. Doğru şekilde girinti yaptığınızdan emin olun:

```
name: CI workflow  
on:  
  push:  
    branches: [ "main" ]  
  pull_request:  
    branches: [ "main" ]  
jobs:  
  build:  
    runs-on: ubuntu-latest  
    container: python:3.9-slim
```

Adım 5: Çalışmanızı Kaydedin

Artık çalışmanızı fork'ladığınız GitHub deposuna kaydetme zamanı.

Göreviniz

1. `git config --global user.email` ve `git config --global user.name` komutlarını kullanarak Git hesabınızı e-posta adresiniz ve adınız ile yapılandırın.

▼ İpucu için buraya tıklayın.

Terminali açın ve e-posta adresinizi yapılandırın:

```
git config --global user.email "you@example.com"
```

Terminali açın ve kullanıcı adınızı yapılandırın:

```
git config --global user.name "Your Name"
```

2. Bir sonraki adım, önceki alıştırmalarda yaptığınız tüm değişiklikleri sahneye almak ve bunları GitHub'daki fork'ladığınız depoya göndermektir.

▼ İpucu için buraya tıklayın.

Değişikliklerinizi sahneye almak ve ardından fork'ladığınız depoya göndermek için aşağıdaki komutları kullanabilirsiniz:

```
git add -A  
git commit -m "COMMIT MESSAGE"
```

```
git push
```

Çıktınız aşağıdaki görüntüye benzer olmalıdır:

Çözüm

```
theia@theia-captainfed01:/home/project/wtecc-CICD_PracticeCode$ git config --global user.name "Theia"
theia@theia-captainfed01:/home/project/wtecc-CICD_PracticeCode$ git config --global user.email "theia@example.com"
theia@theia-captainfed01:/home/project/wtecc-CICD_PracticeCode$ git add -A
theia@theia-captainfed01:/home/project/wtecc-CICD_PracticeCode$ git commit -m "finished workflow file"
[main c362045] finished workflow file
 1 file changed, 24 insertions(+)
 create mode 100644 .github/workflows/workflow.yml
theia@theia-captainfed01:/home/project/wtecc-CICD_PracticeCode$ git push
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 746 bytes | 373.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/capfedora/wtecc-CICD_PracticeCode.git
 03977e9..c362045  main -> main
```

Laboratuvarın 1. kısmını tamamladınız, ancak fork’ladığınız depodaki Actions sekmesine bakarsanız, GitHub eyleminin tetiklendiğini ve başarısız olduğunu göreceksiniz. Eylem, deponun `main` dalına kod `push` ettiğiniz için tetiklendi. İş akışını henüz tamamlamadığınız için başarısız oldu. İş akışının başarılı bir şekilde çalışması için laboratuvarın 2. kısmında kalan adımları ekleyeceksiniz. Bu hatayı şu anda göz ardı edebilirsiniz.

⚡ capfedora / wtecc-CI_CD_PracticeCode

Public

forked from [ibm-developer-skills-network/wtecc-CI_CD_PracticeCode](#)

<> Code

Pull requests

Actions

Projects

Wiki

!

Workflows

New workflow

All workflows

.github/workflows/workflow....

All workflows

Showing runs from all wo

Filter workflow runs

2 workflow runs

finished GitHub .github/workflows/w capfedora

Sonuç

Tebrikler! Bu laboratuvara Sürekli Entegrasyon (Continuous Integration) hattınızı oluşturmaya başladınız. Bu hat, kodunuzu GitHub deposuna taahhüt ettiğinizde, iş akışında tanımlanan olaylara dayanarak otomatik olarak çalışacaktır.

Başarıyla bir GitHub Actions iş akışı oluşturduğunuz ve boş bir iş eklediniz. Artık bağımlılıkları oluşturmak, kodunuzu test etmek ve test kapsamını raporlamak için adımlar ekleyerek CI hattını genişletmeye devam edebilirsiniz.

Author(s)

Tapas Mandal

Other Contributor(s)

Captain Fedora
John Rofrano

© IBM Corporation. Tüm hakları saklıdır.