GitHub Actions의 체크아웃(Checkout) 액션으로 코드 내려받기

GitHub Actions에서 가장 많이 사용되는 액션(Action)은 무엇일까요? 바로 저장소로 부터 코드를 내려받기 위한 체크아웃(Checkout) 액션인데요. 이번 포스팅에서는 GitHub Actions를 사용할 때 거의 필수적으로 사용하게 되는 체크아웃 액션에 대해서 알아보겠습니다.

GitHub Actions의 액션(Action)이란?

먼저 GitHub Actions에서 액션(Action)이 무엇을 의미하는지 간단하게 짚고 넘어가겠습니다. GitHub Actions는 일반적으로 CI(Continuous Integration, 지속 통합) 또는 CD(Continuous Deployment, 지속 배포)와 같은 자동화를 위해서 사용되는데요. 이를 위해 워크플로우(workflow)를 구성하다보면 거의 필연적으로 여러 작업(job)에서 반복적으로 처리되야 할 일들이 생겨나기 마련입니다.

액션(Action)은 이렇게 빈번하게 필요한 반복 단계를 재사용하기 용이하도록 GitHub Actions에서 제공되는 일종의 작업 공유 매커니즘인데요. 이 액션은 하나의 코드 저장소 범위 내에서 여러 워크플로우 간에서 공유를 할 수 있을 뿐만 아니라, 공개 코드 저장소를 통해 액션을 공유하면 깃허브 상의 모든 코드 저장소에서 사용이 가능해집니다.

누구나 GitHub Marketplace를 통해 깃허브 뿐만 아니라 다양한 써드 파티 업체들이 공개해놓은 액션을 쉽게 검색하고 써볼 수 있습니다.

GitHub Actions에 대한 소개와 핵심 개념은 별도로 정리해 두었으니 관련 포스팅를 참고 바랍니다.

Git의 체크아웃(Checkout)이란?

다음으로 Git이라는 버전 관리 시스템에서 체크아웃(Checkout)이 어떤 개념인지 가볍게 짚고 넘어가겠습니다. Git에서 체크아웃이라는 행위는 일반적으로 코드 저장소(repository)에서 특정 브랜치(branch)로 전환하는 작업을 뜻하는데요. 이 작업을 위해서는 우선 원격 저장소에 올려둔 코드를 로컬 컴퓨터로 내려받는 작업이 반드시 선행이 되어야 합니다.

GitHub의 체크아웃 액션

Git의 체크아웃을 GitHub Actions 입장에서 바라보면 깃허브의 코드 저장소에 올려둔 코드를 CI 서버로 내려받은 후에 특정 브랜치로 전환하는 행위라고 볼 수 있습니다. 이렇게 자주 필요한 체크아웃 작업 단계를 모든 워크플로우에서 매번 쉘 스크립트로 작성하려면 매우 번거롭겠죠? 뿐만 아니라 CI 서버와 코드 저장소 간에 인증과 같은 까다로운 절차도 신경을 써줘야 할 것입니다.

GitHub Actions에서 처리되는 모든 작업은 이 체크아웃 단계부터 시작한다고 해도 과언이 아닌데요. 그래서 깃허브에서는 체크아웃에 필요한 일련의 과정을 묶어서 액션으로 제공하고 있습니다. 이 GitHub Actions의 Checkout 액션을 사용하면 매우 간편하게 코드 저장소로 부터 CI 서버로 코드를 내려받도록 워크플로우를 구성할 수 있습니다.

지금부터 실습 저장소에 GitHub Actions에 간단한 워크플로우를 구성하고 깃허브에서 제공하는 체크아웃 액션을 사용해보겠습니다.

실습 워크플로우 생성

깃허브 계정이 있으시다면 누구나 무료로 GitHub Actions를 사용해볼 수 있습니다. 실습을 위해서 본인 GitHub에 계정에 새로운 코드 저장소(repository)를 하나를 만드시길 바랍니다. (저는 `github-actions-checkout` 이라는 이름으로 실습 코드 저장소를 만들겠습니다.) 그 다음 저장소에 `.github/workflows/` 라는 폴더를 만든 후, 그 안에 `checkout.yml` 이라는 이름의 YAML 파일을 하나 생성합니다.

`checkout.yml` 파일에는 아래와 같은 기본 템플릿을 복사해서 붙여넣고 실습을 시작해보겠습니다.

.github/workflows/checkout.yml

```yaml
name: Our Workflow
on: push
jobs:
  checkout:
    runs-on: ubuntu-latest
    steps:
      - run: ls -al
```

워크플로우를 만드는 방법은 별도 포스팅에서 자세히 다루었으니 참고 바랍니다.

이 파일을 저장 후 GitHub의 코드 저장소로 올린(push) 후 `Actions` 탭에 들어가면 다음과 같은 워크플로우 실행 로그를 확인할 수 있을 것입니다.

Log

```
✅ Set up Job
✅ Run ls -al
▶ Run ls -al
total 8
drwxr-xr-x 2 runner docker 4096 Apr 23 01:33 .
drwxr-xr-x 3 runner docker 4096 Apr 23 01:33 ..
✅ Complete Job
```

아직 깃허브 저장소로 부터 CI 서버로 코드를 한 번도 내려받은 적이 없기 때문에 `ls -al` 명령어의 실행결과로 비어있는 디렉토리를 확인되는 것을 볼 수 있습니다.

체크아웃 액션 사용

워크플로우 YAML 파일에서는 `steps` 키 하위의 `uses` 키에 사용하고자 하는 액션의 위치를 `{소유자}/{저장소명}@{참조자}` 의 형태로 명시하는데요. GitHub에서 제공하는 체크아웃 액션의 소유자는 `actions` 이고, 저장소 이름은 `checkout` 이며 현재 포스팅 시점에서 사용 가능한 최신 버전은 `v3` 입니다.

따라서 실습 워크플로우에 체크아웃 액션을 추가하려면 다음과 같이 `checkout.yml` 을 수정해줍니다.

.github/workflows/checkout.yml

```yaml
name: Our Workflow
on: push
jobs:
```

```
    checkout:
      runs-on: ubuntu-latest
      steps:
        - run: ls -al
        - uses: actions/checkout@v3
```

이제 다시 Actions 탭에 들어가보면 워크플로우 실행 로그에 체크아웃 액션 관련 부분이 추가되어 있는 것을 볼 수 있습니다.

Log

- ✅ Set up Job
- ✅ Run ls -al
- ✅ Run actions/checkout@v3
- ✅ Post Run actions/checkout@v3
- ✅ Complete Job

Run actions/checkout@v3 항목을 열어보시면 CI 서버에서 어떻게 저장소로부터 코드를 내려받는지 알 수가 있는데요. 대강 훑어보시면 다음과 같은 Git 명령어들이 실행되고 있는 것을 보실 수 있으실 것입니다.

- git init 명령어를 통해 작업 디렉토리를 로컬 저장소로 만든다.
- git config 명령어를 통해 각종 인증 관련 정보를 설정한다
- git fetch 명령어를 통해 원격 저장소로 부터 코드를 받아온다.
- git checkout 명령어를 통해 주(main) 브랜치로 전환한다.
- git log 명령어로 마지막 커밋(commit)의 해시값을 확인한다.

깃허브에서 체크아웃 액션을 제공해주지 않았다면 다 우리가 스스로 해야했던 일이겠죠? 😳 체크아웃 액션 덕분에 우리는 이 모든 일을 워크플로우 파일에서 단 한줄로 처리할 수 있는 것입니다.

내려받은 코드 확인

정말로 코드 저장소에 올려둔 코드가 CI 서버로 잘 내려받아졌는지 확인해볼까요? 우리가 현재 작성 중인 checkout.yml 자체도 GitHub의 코드 저장소에 올려져있기 때문에 작업 디렉토리 내에서 해당 파일이 존재하는지 확인해보면 되겠네요.
체크아웃 액션의 다음 단계로 두 개의 명령어를 추가해보겠습니다. 첫 번째는 작업 디렉토리 내의 파일을 나열하기 위함이고, 두 번째는 워크플로우 YAML 파일의 내용을 출력하기 위함입니다.

.github/workflows/checkout.yml

```
name: Our Workflow
on: push
jobs:
  checkout:
    runs-on: ubuntu-latest
    steps:
      - run: ls -al
      - uses: actions/checkout@v3
      - run: ls -al
      - run: cat .github/workflows/checkout.yml
```

이제 다시 Actions 탭에서 워크플로우 실행 로그를 확인해보면 체크아웃을 액션을 사용하기 전 단계에서는 CI 서버의 작업 디렉토리가 분명 비어있는데 체크아웃 액션을 사용한 다음 단계에서는 README.md 과 LICENSE 파일을 포함한 저장소 내의 코드가 CI 서버의 작업 디렉토리 내로 그대로 복사된 것을 볼 수 있습니다. 워크플로우 YAML 파일인 checkout.yml 도 .github/workflows/ 폴더 아래에 존재하며 우리가 작성한 그대로 설정 내용이 출력이 됩니다.

Log

```
✅ Set up Job
✅ Run ls -al
▶ Run ls -al
total 8
drwxr-xr-x 2 runner docker 4096 Apr 23 01:33 .
drwxr-xr-x 3 runner docker 4096 Apr 23 01:33 ..
✅ Run actions/checkout@v3
✅ Run ls -al
▶ Run ls -al
total 20
drwxr-xr-x 4 runner docker 4096 Apr 23 01:33 .
drwxr-xr-x 3 runner docker 4096 Apr 23 01:33 ..
drwxr-xr-x 8 runner docker 4096 Apr 23 01:33 .git
drwxr-xr-x 3 runner docker 4096 Apr 23 01:33 .github
```

```
-rw-r--r-- 1 runner docker 1067 Apr 23 01:33 LICENSE
-rw-r--r-- 1 runner docker   25 Apr 23 01:33 README.md
 Run cat .github/workflows/checkout.yml
▶ Run cat .github/workflows/checkout.yml
name: Our Workflow
on: push
jobs:
  checkout:
    runs-on: ubuntu-latest
    steps:
      - run: ls -al
      - uses: actions/checkout@v3
      - run: ls -al
      - run: cat .github/workflows/checkout.yml
 Post Run actions/checkout@v3
 Complete Job
```

› 다른 경로에 코드 내려받기

CI 서버의 최상위 작업 디렉토리 대신에 그 하위에 있는 다른 폴더로 코드를 내려받고 싶다면 체크아웃 액션의 `path` 옵션을 사용할 수 있습니다.
워크플로우 YAML 파일에서 어떤 액션에 입력 파라미터를 넘길 때는 `with` 키를 사용하는데요. 파라미터의 이름과 값의 맵핑을 지정해주면 됩니다.
실습 워크플로우에서는 `our-source` 라는 폴더에 코드를 내려받도록 한번 설정해볼께요.

.github/workflows/checkout.yml

```
name: Our Workflow
on: push
jobs:
  checkout:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
        with:
          path: our-source
      - run: ls -al
      - run: ls -al our-source
```

이제 다시 Actions 탭에서 워크플로우 실행 로그를 확인해보면 작업 디렉토리에는 `our-source` 라는 폴더가 생겼고, 이 폴더 안에 저장소로 부터 내려받은 코드가 위치할 것입니다.

Log

```
 Set up Job
 Run actions/checkout@v3
▶ Run ls -al
total 12
drwxr-xr-x 3 runner docker 4096 Apr 23 02:03 .
drwxr-xr-x 3 runner docker 4096 Apr 23 02:03 ..
drwxr-xr-x 4 runner docker 4096 Apr 23 02:03 our-source
 Run ls -al our-source
▶ Run ls -al our-source
total 20
drwxr-xr-x 4 runner docker 4096 Apr 23 02:03 .
drwxr-xr-x 3 runner docker 4096 Apr 23 02:03 ..
drwxr-xr-x 8 runner docker 4096 Apr 23 02:03 .git
drwxr-xr-x 3 runner docker 4096 Apr 23 02:03 .github
-rw-r--r-- 1 runner docker 1067 Apr 23 02:03 LICENSE
-rw-r--r-- 1 runner docker   25 Apr 23 02:03 README.md
 Post Run actions/checkout@v3
 Complete Job
```

› 과거 변경 이력도 불러오기

체크아웃 액션은 기본적으로 저장소로 부터 최신 상태의 코드만을 내려받는데요. 대부분의 CI/CD 작업에서 마지막 커밋(commit)만 있으면 충분하고 성능 측면에서도 마지막 커밋만 가져오는 것이 유리하기 때문입니다. 만약에 과거 변경내역이 필요한 경우라면 `fetch-depth` 옵션을 사용하여 최근 몇개의 커밋을 불러오기 싶은지 설정할 수 있습니다. 기본값은 `1` 이며, `0` 으로 설정하면 전체 변경 이력을 불러옵니다.
실습에서는 최근 3개의 변경이력을 불러온 후, 바로 다음 단계에서 `git log` 명령어를 이용해서 변경 이력을 출력해보겠습니다.

.github/workflows/checkout.yml

```yaml
name: Our Workflow
on: push
jobs:
  checkout:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
        with:
          fetch-depth: 3
      - run: git log --pretty=oneline
```

다시 Actions 탭에서 워크플로우 실행 로그를 확인해보면 3개의 변경 이력이 잘 확인될 것입니다.

```
✅ Set up Job
✅ Run actions/checkout@v3
✅ Run git log --pretty=oneline
Run git log --pretty=oneline
4325676aa19f300a441cca95b6a8d4fde8b9dca4 Update checkout.yml
a9cb009eec4d44d135f027c5432997b2a544afab Update checkout.yml
f09c8dad47aa7f4427bc4b45016d3b7153373f44 Update checkout.yml
✅ Post Run actions/checkout@v3
✅ Complete Job
```

› 다른 저장소의 코드 내려받기

체크아웃 액션은 비단 GitHub Actions 워크플로우가 위치하고 있는 현재 저장소뿐만 아니라 공개된 저장소라면 다른 저장소의 코드를 내려받을 때도 사용할 수 있습니다. repository 옵션으로 {소유자}/{저장소명} 형태로 저장소를 지정해주고, ref 옵션으로 브랜치명이나 태크명 또는 커밋 해시값을 지정해주면 됩니다.

재밌는 예로 체크아웃 액션 자체의 소스 코드를 담고 있는 저장소로 부터 코드를 한 번 내려받도록 워크플로우를 설정해보겠습니다.

.github/workflows/checkout.yml

```yaml
name: Our Workflow
on: push
jobs:
  checkout:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
        with:
          repository: actions/checkout
          ref: v3
      - run: ls
```

액션에 대한 메타 정보를 담고 있는 action.yml 파일이 확인되는 걸로 봐서 해당 저장소가 CI 서버로 잘 내려받진 것 같습니다.

```
✅ Set up Job
✅ Run actions/checkout@v3
✅ Run ls
Run ls
CHANGELOG.md
CODEOWNERS
LICENSE
README.md
__test__
action.yml
adrs
dist
jest.config.js
package-lock.json
package.json
src
tsconfig.json
✅ Post Run actions/checkout@v3
✅ Complete Job
```

참고로 repository 옵션을 생략하고 ref 옵션만 사용하면 현재 저장소 내의 다른 태그나 브랜치, 커밋 해시의 코드를 내려받을 수 있습니다.

› 실습 코드

본 포스팅에서 작성한 YAML 파일과 워크플로우 실행 결과는 아래 코드 저장소에서 확인하실 수 있습니다.

https://github.com/DaleSchool/github-actions-checkout