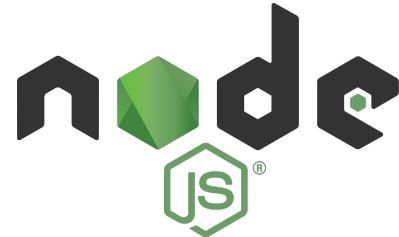






Assignments / 과제

과제를 완료할 때 해결해야 할 3가지 주요 문제가 있다.



01. Access / 과제 받기

모든 과제를 각 수업이 시작될 때부터 [홈페이지](#)에서 찾을 수 있다.

02. Submit / 과제 제출

과제를 제출하는 가장 쉬운 방법은 VS Code 내에서 직접 커밋 및 푸시를 수행하는 것이다.

03. Check / 테스트 결과 확인

터미널에서 로컬로 코드를 확인하거나 GitHub Actions에서 온라인으로 확인할 수 있다.

01

Access

과제 받기



과제를 어떻게 받을 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



1. 깃 클래스룸으로 **VS Code**를 직접 열기
2. 명령줄에서 **git clone**
3. 직접 코드 다운로드





과제를 어떻게 받을 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



1. 깃 클래스룸으로 **VS Code**를 직접 열기
2. 명령줄에서 **git clone**
3. 직접 코드 다운로드



1.

깃 클래스룸으로 **VS Code**를 열기

1. [홈페이지](#),
2. 과제 참여 링크,
3. 개별 저장소,
4. GitHub 조직

New Tab x +

http://

🏠 | 소개 | 일정 | 스라이드 | **[과제]** | 프로젝트 | 시험 | 성적 | 설문 ↗

Web Programming Application 2023

한국교통대학교, 충주 | KNUT (Korea National University of Transportation)

Practice / 과제

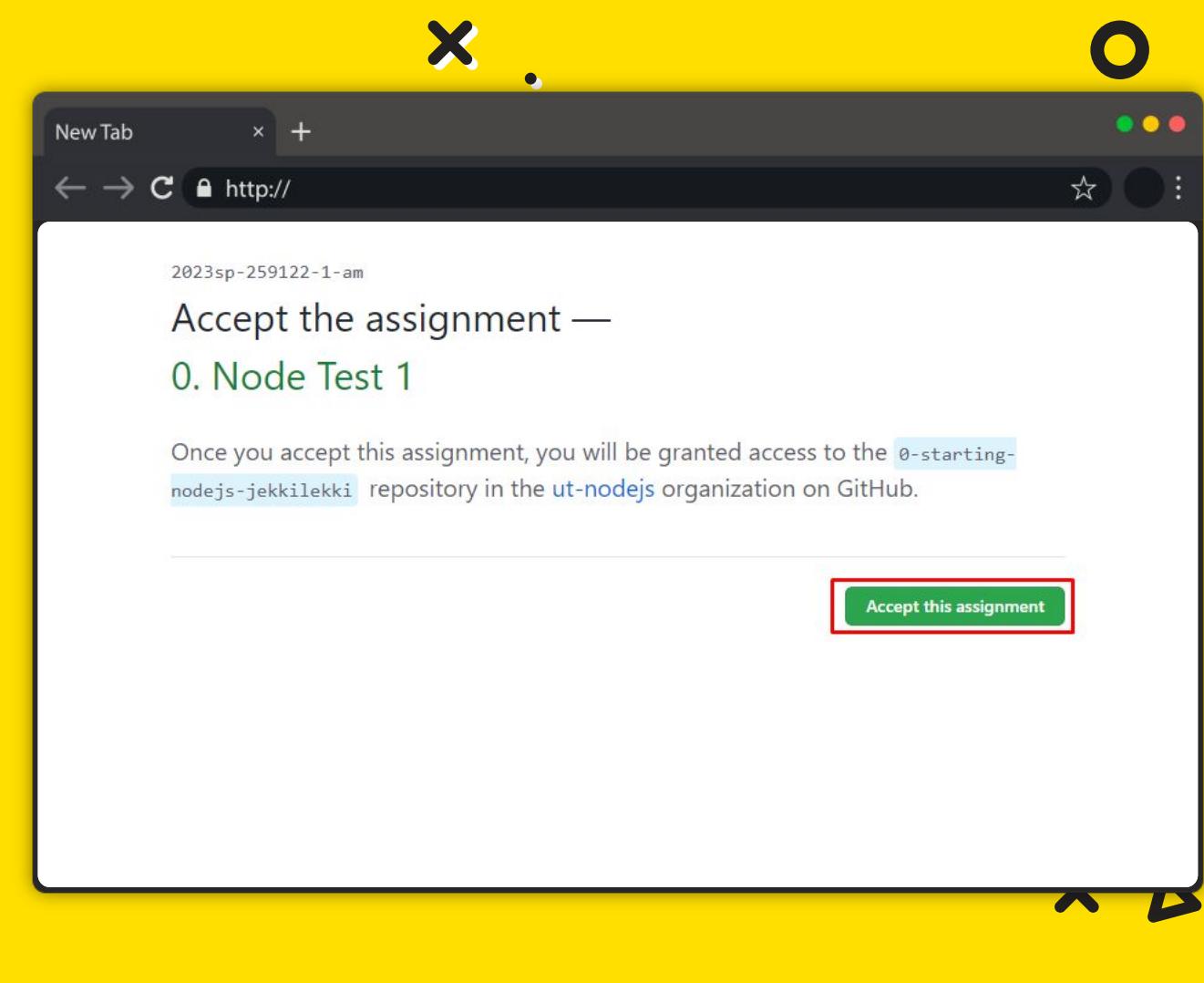
Practice Labs will be assigned and collected in **GitHub** Classroom and accessible there. Follow the instructions below to understand how to access and submit labs.

Week	Date	과제	am	pm
1	3월3일	-	-	-
2	3월10일	0. Starting Node / 노드 시작	오전	오후

1.

깃 클래스룸으로 **VS Code**를 열기

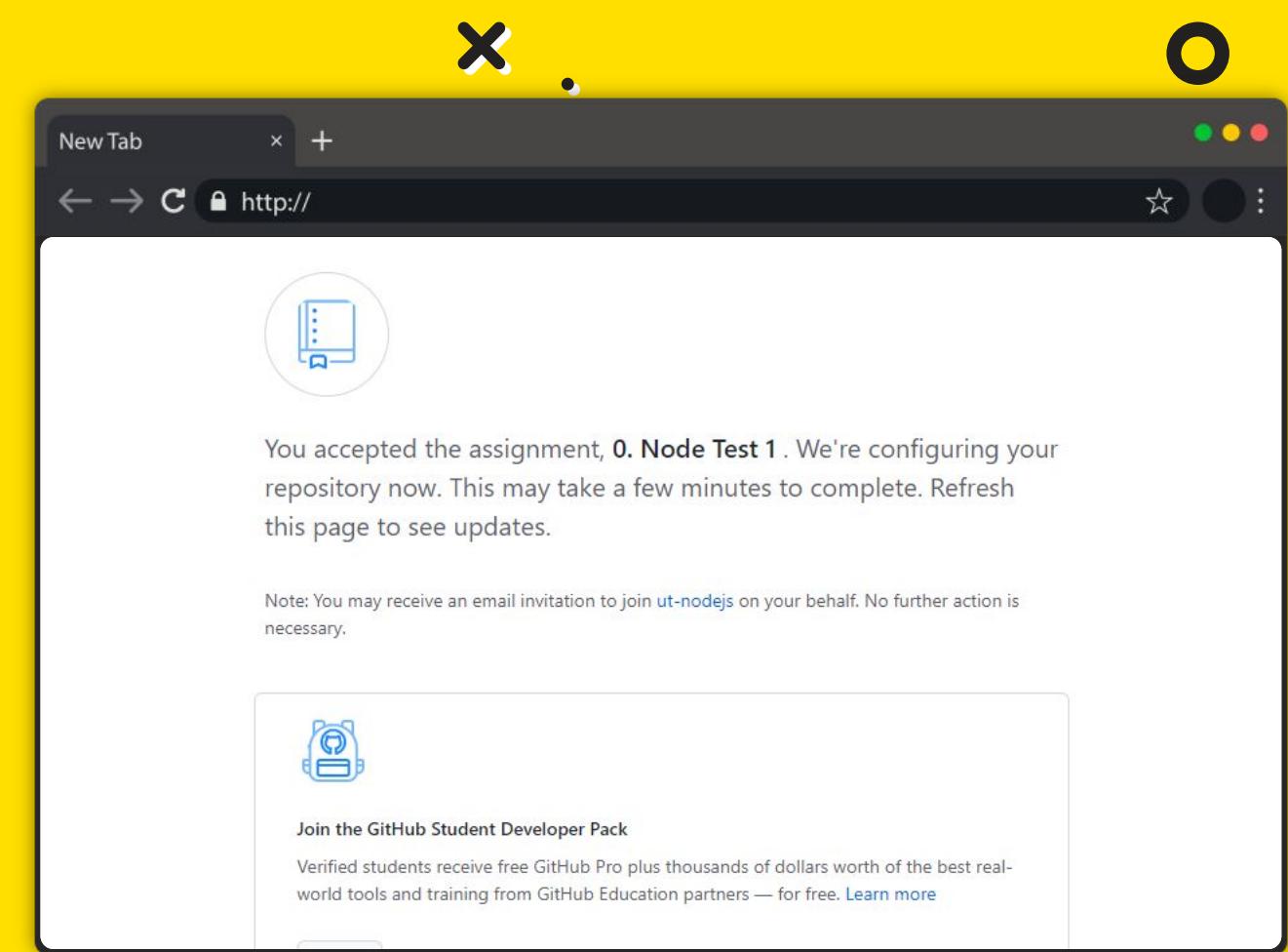
1. 홈페이지,
2. 과제 참여 링크,
3. 개별 저장소,
4. GitHub 조직



1.

깃 클래스룸으로 **VS Code**를 열기

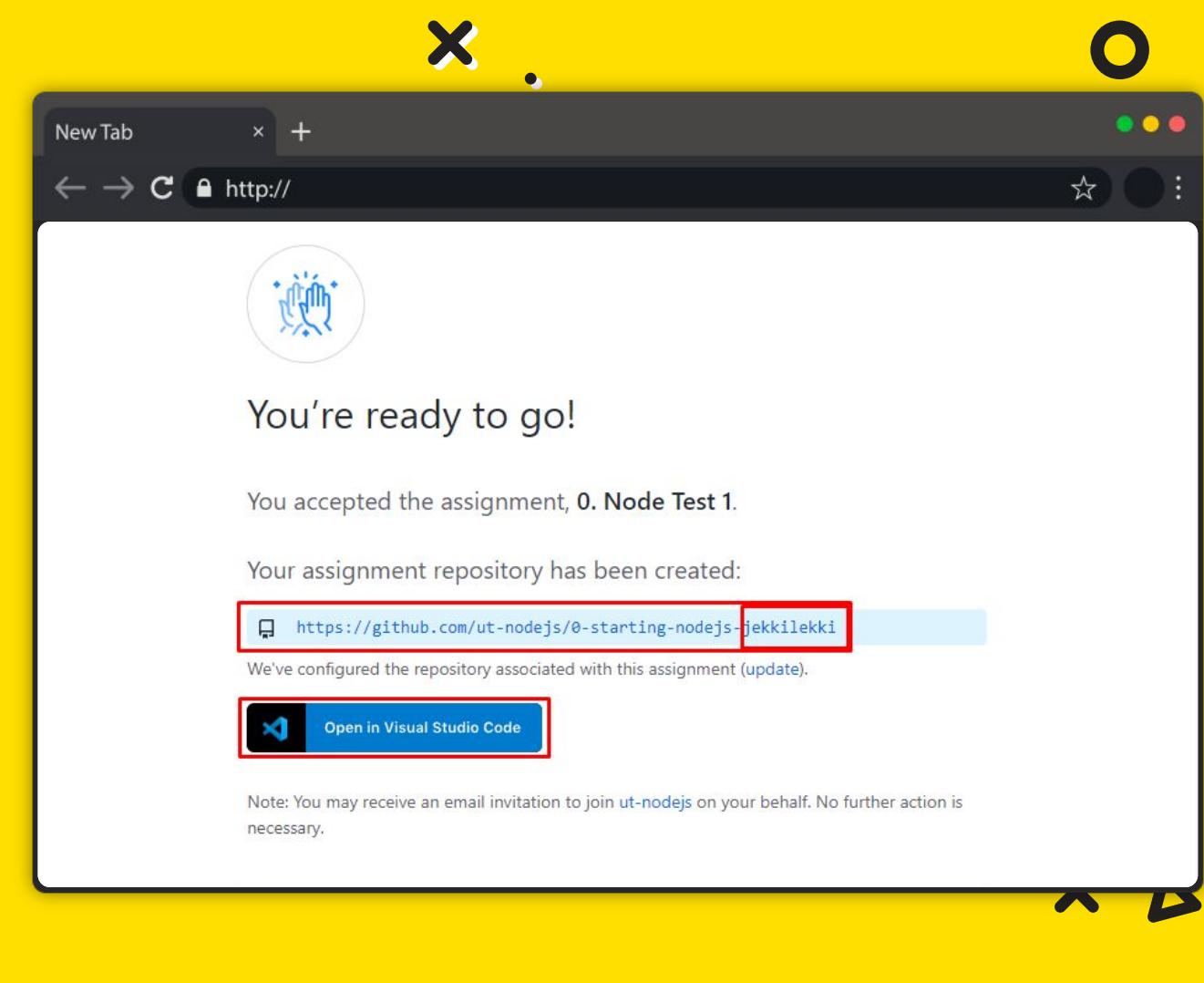
1. 홈페이지,
2. 과제 참여 링크,
3. 개별 저장소,
4. GitHub 조직



1.

깃 클래스룸으로 **VS Code**를 열기

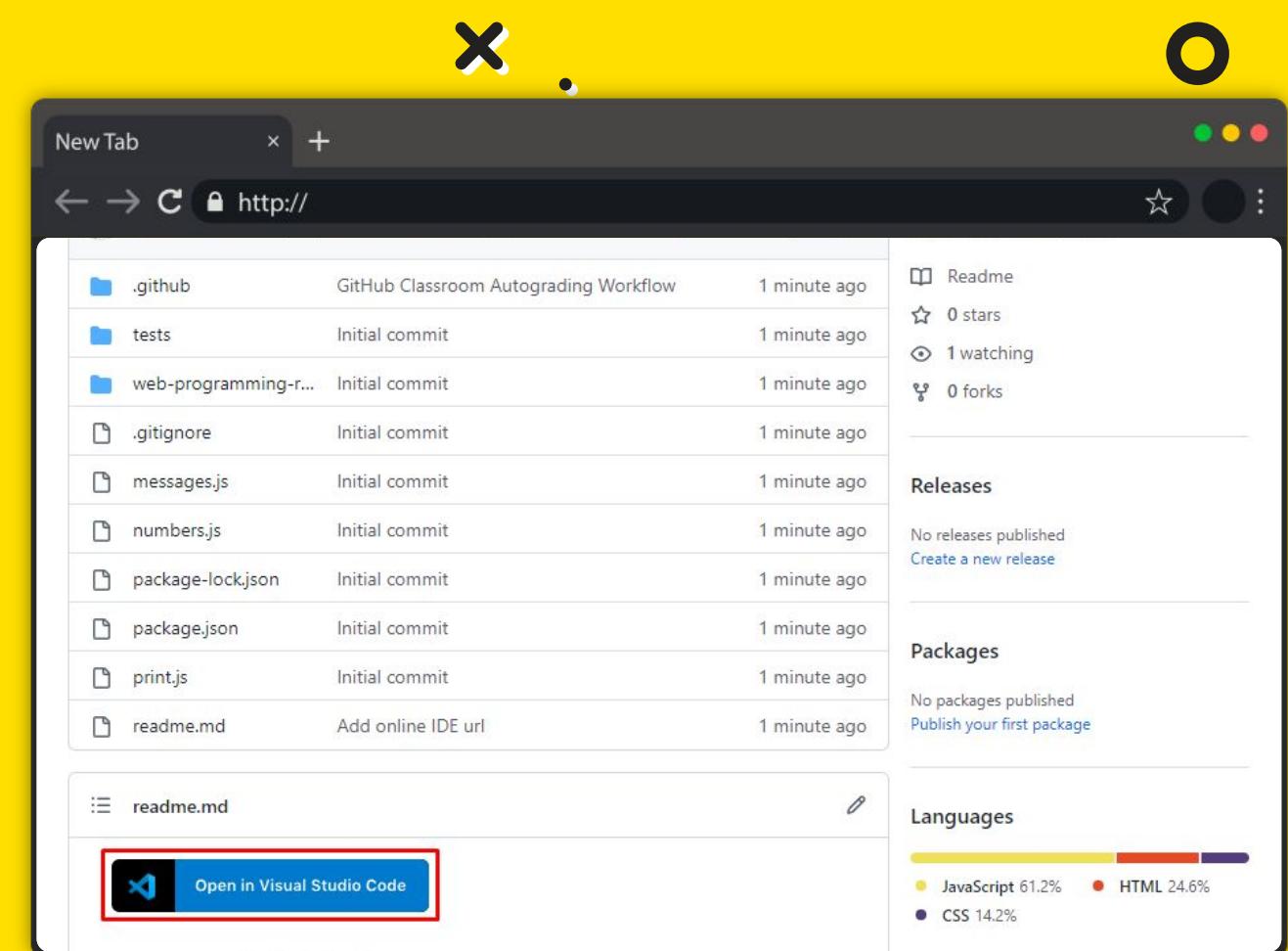
1. 홈페이지,
2. 과제 참여 링크,
3. 개별 저장소,
4. GitHub 조직



1.

깃 클래스룸으로 **VS Code**를 열기

1. 홈페이지,
2. 과제 참여 링크,
- 3. 개별 저장소,**
4. GitHub 조직

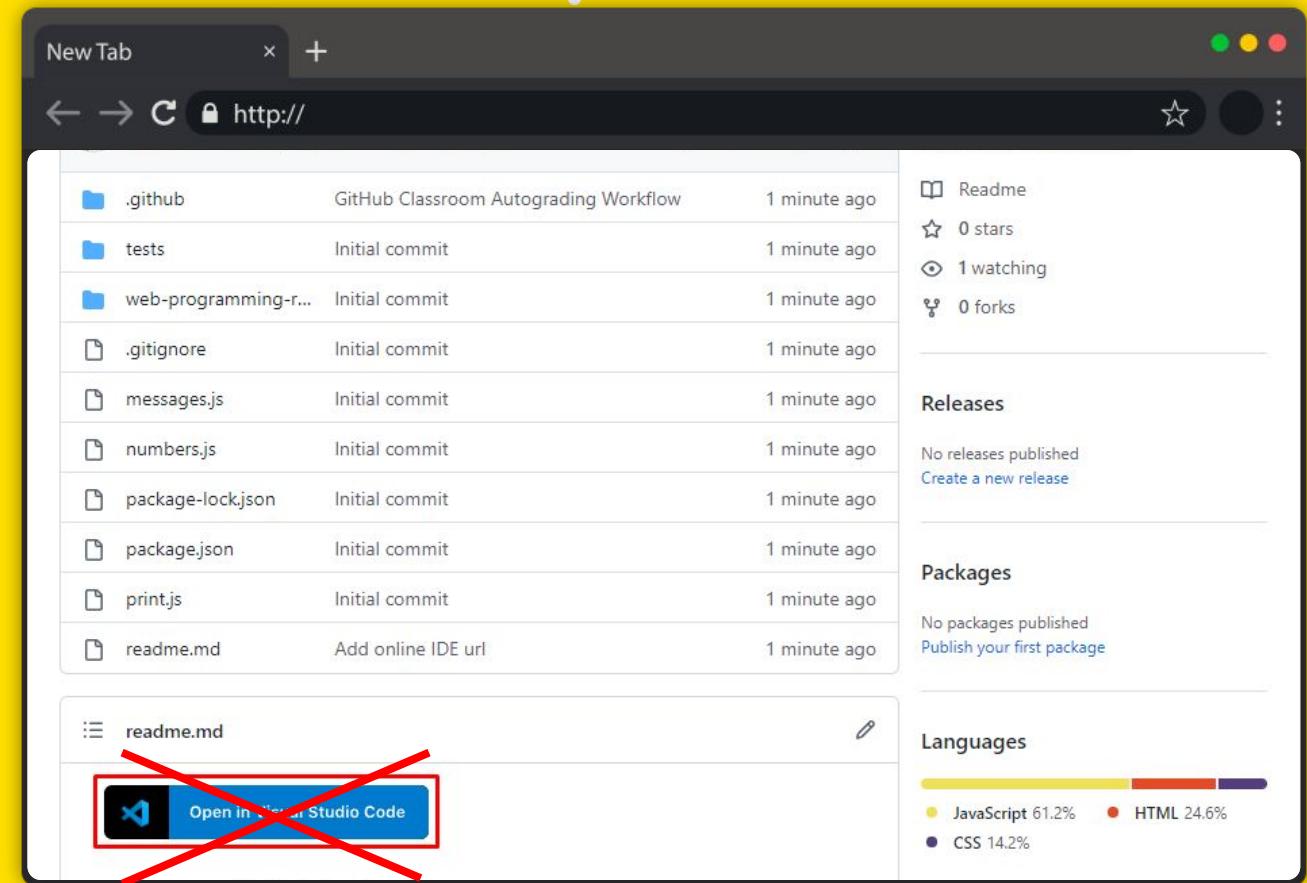


1.

깃 클래스룸으로 **VS Code**를 열기

1. 홈페이지,
2. 과제 참여 링크,
- 3. 개별 저장소,**
4. GitHub 조직

못 열면?





과제를 어떻게 받을 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



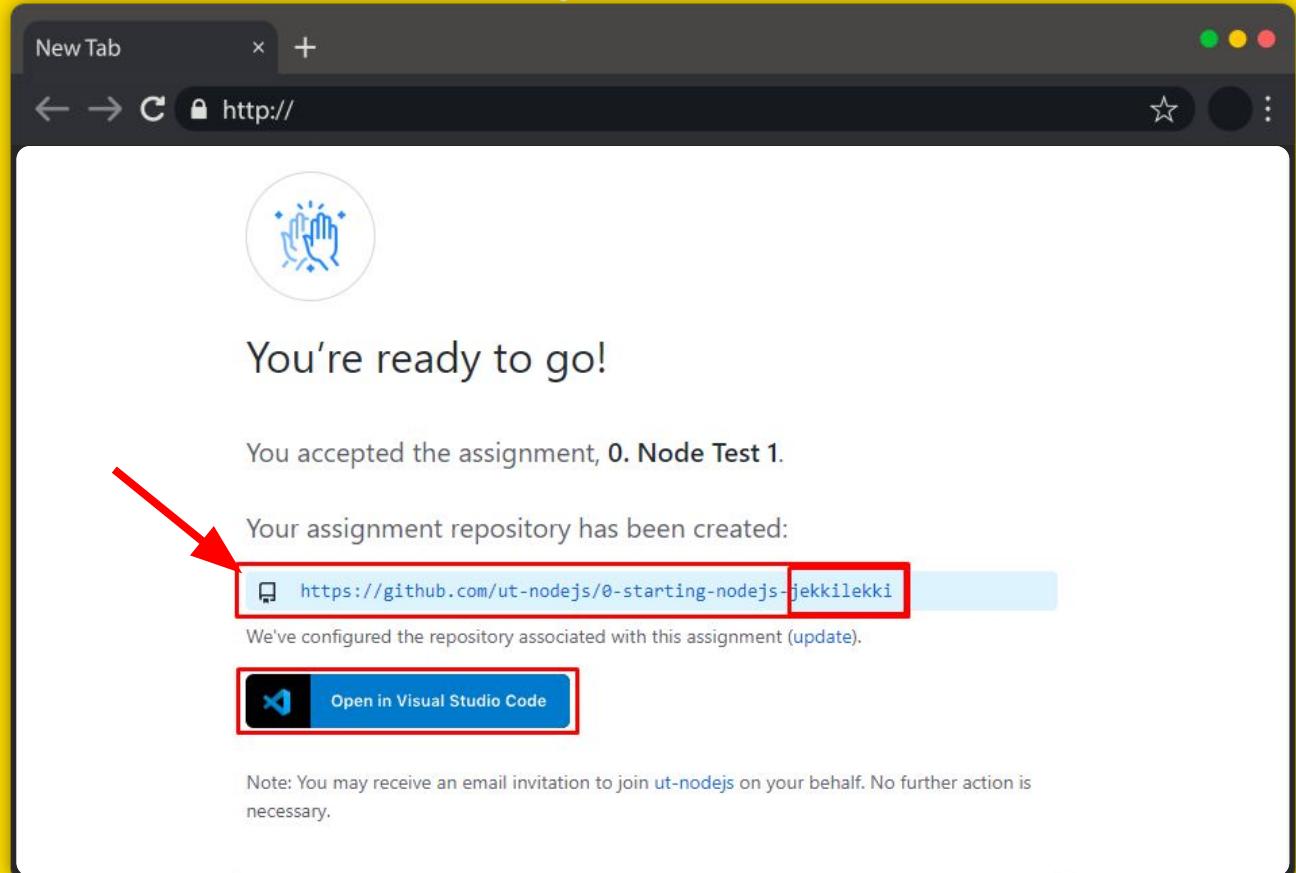
1. 깃 클래스룸으로 **VS Code**를 직접 열기
2. 명령줄에서 **git clone**
3. 직접 코드 다운로드



2.

명령줄에서 **git clone**

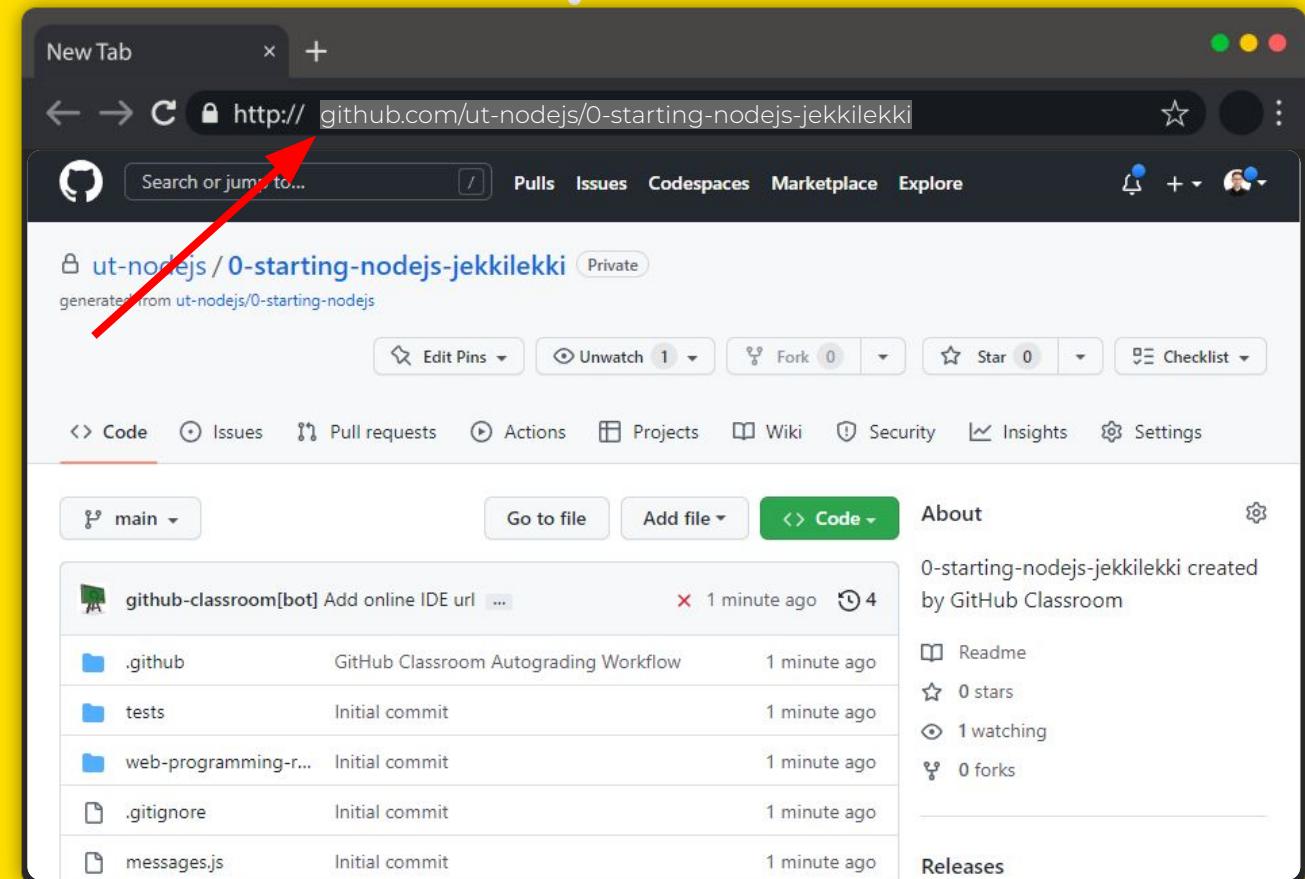
1. 온라인 저장소로 가고
2. URL 복사
3. 바탕화면에서 새 폴더
4. Git Bash에서 폴더로 cd
5. git clone URL



2.

명령줄에서 git clone

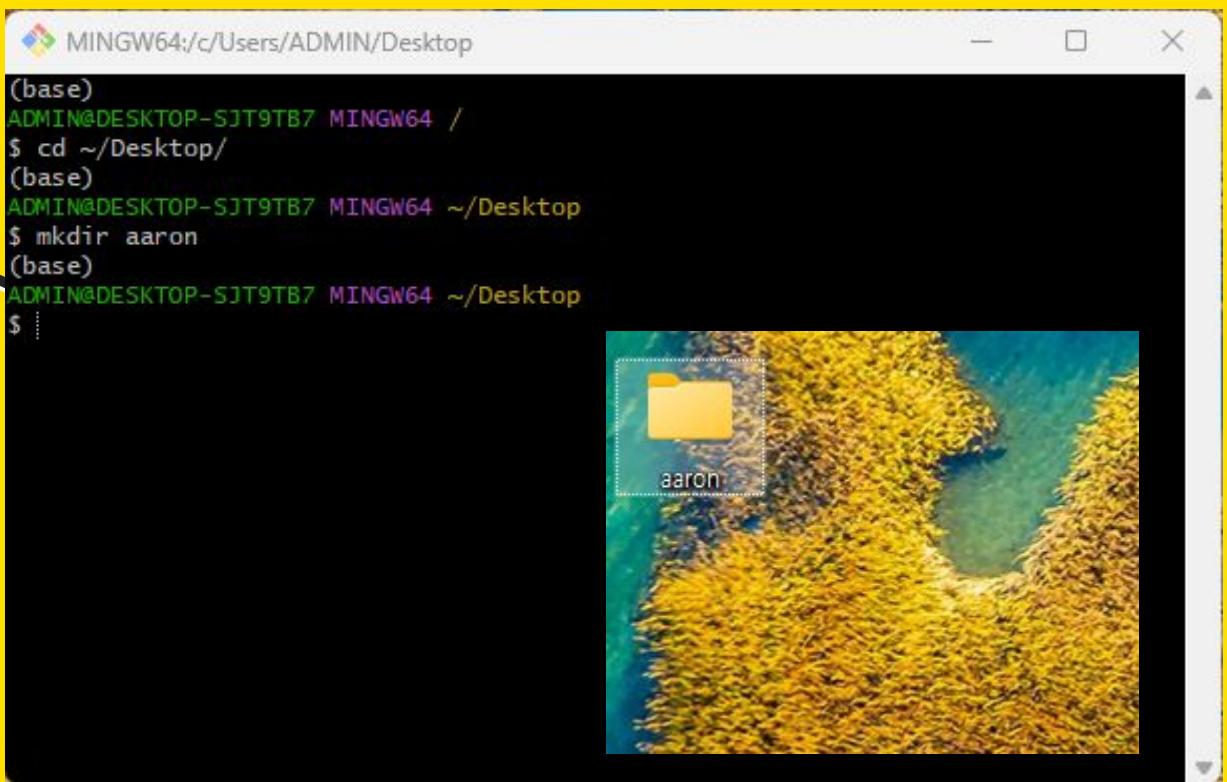
1. 온라인 저장소로 가고
2. URL 복사
3. 바탕화면에서 새 폴더
4. Git Bash에서 폴더로 cd
5. git clone URL



2.

명령줄에서 **git clone**

1. 온라인 저장소로 가고
2. URL 복사
- 3. 바탕화면에서 새 폴더**
4. Git Bash에서 폴더로 cd
5. git clone URL



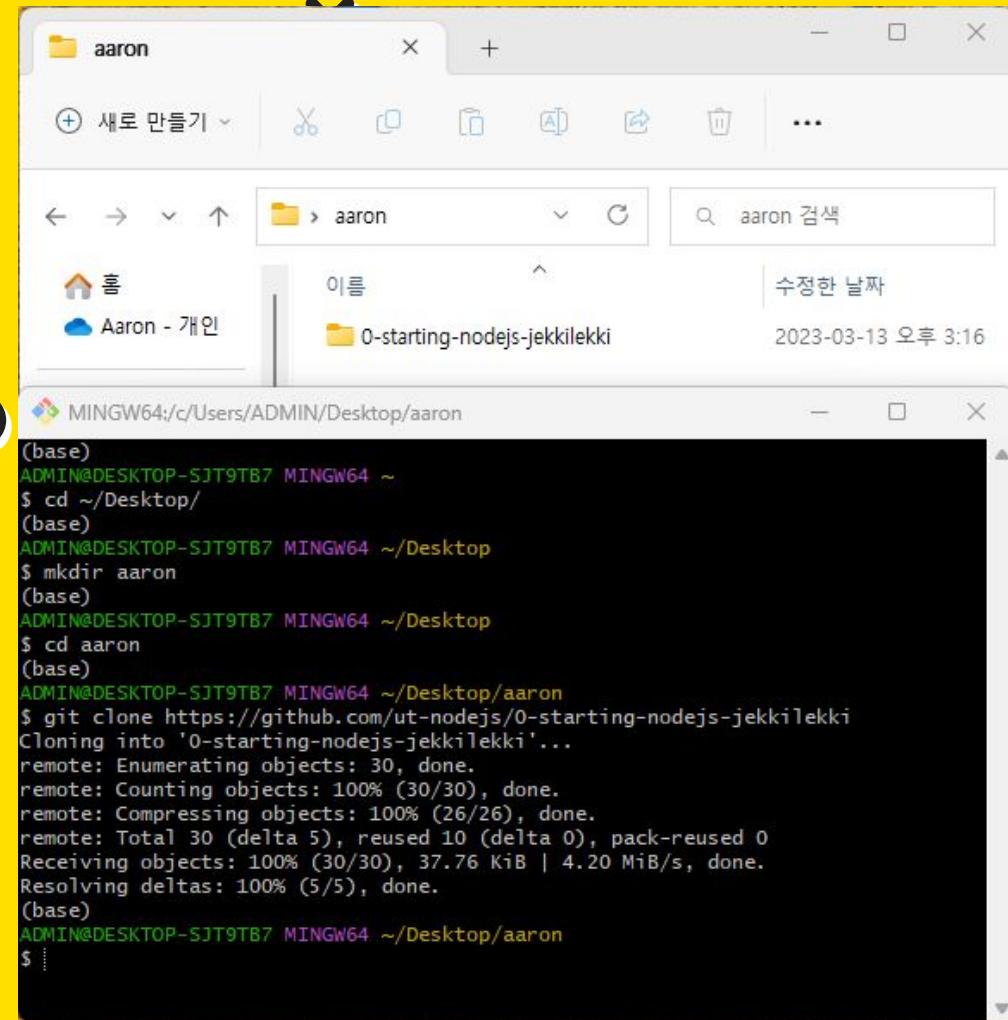
```
MINGW64:/c/Users/ADMIN/Desktop
(base)
ADMIN@DESKTOP-SJT9TB7 MINGW64 /
$ cd ~/Desktop/
(base)
ADMIN@DESKTOP-SJT9TB7 MINGW64 ~/Desktop
$ mkdir aaron
(base)
ADMIN@DESKTOP-SJT9TB7 MINGW64 ~/Desktop
$
```



2.

명령줄에서 **git clone**

1. 온라인 저장소로 가고
2. URL 복사
3. 바탕화면에서 새 폴더
4. Git Bash에서 폴더로 cd
5. git clone URL



```
MINGW64:/c/Users/ADMIN/Desktop/aaron
(base)
ADMIN@DESKTOP-SJT9TB7 MINGW64 ~
$ cd ~/Desktop/
(base)
ADMIN@DESKTOP-SJT9TB7 MINGW64 ~/Desktop
$ mkdir aaron
(base)
ADMIN@DESKTOP-SJT9TB7 MINGW64 ~/Desktop
$ cd aaron
(base)
ADMIN@DESKTOP-SJT9TB7 MINGW64 ~/Desktop/aaron
$ git clone https://github.com/ut-nodejs/0-starting-nodejs-jekkilekki
Cloning into '0-starting-nodejs-jekkilekki'...
remote: Enumerating objects: 30, done.
remote: Counting objects: 100% (30/30), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 30 (delta 5), reused 10 (delta 0), pack-reused 0
Receiving objects: 100% (30/30), 37.76 KiB | 4.20 MiB/s, done.
Resolving deltas: 100% (5/5), done.
(base)
ADMIN@DESKTOP-SJT9TB7 MINGW64 ~/Desktop/aaron
$
```



과제를 어떻게 받을 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



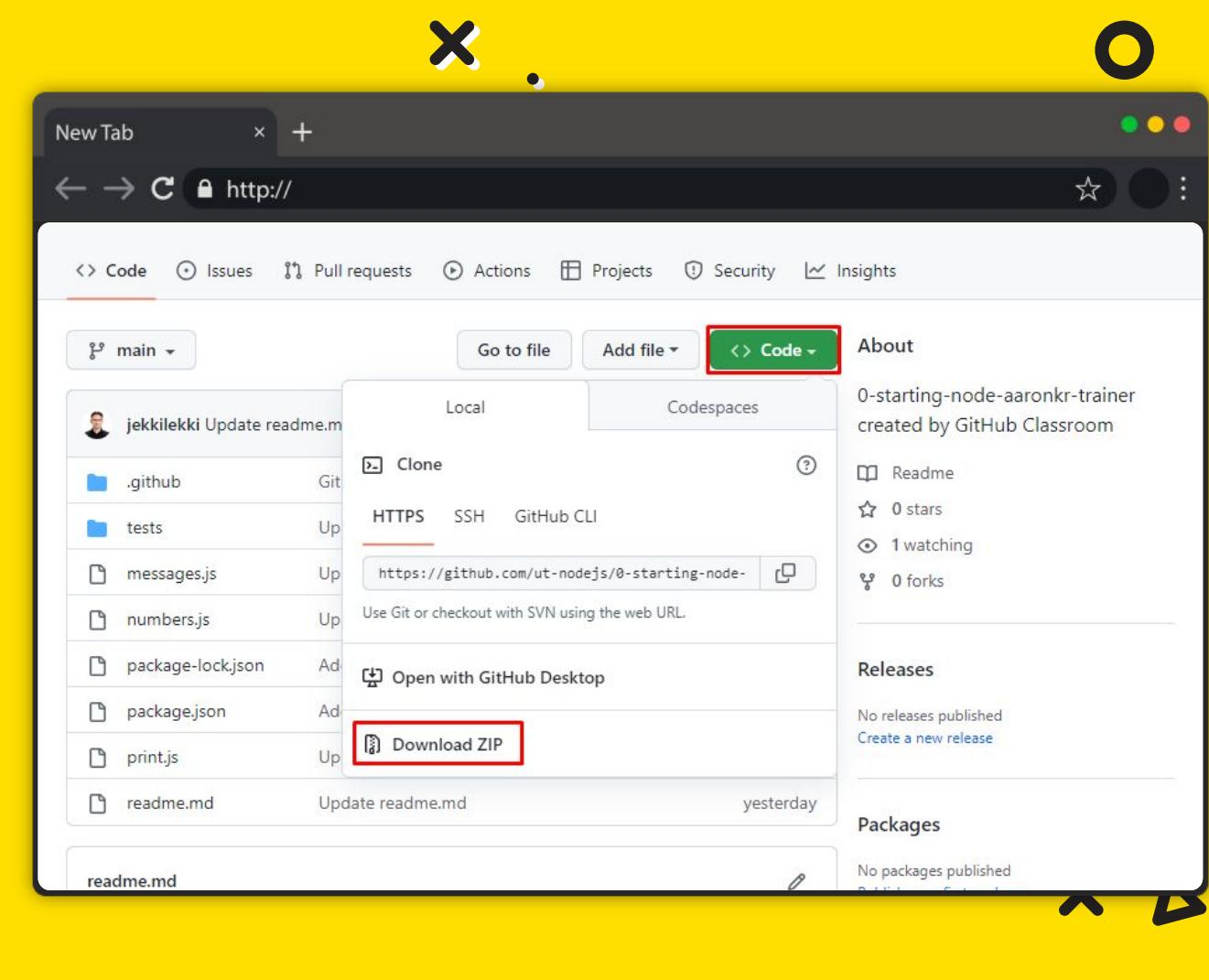
1. 깃 클래스룸으로 **VS Code**를 직접 열기
2. 명령줄에서 **git clone**
3. 직접 코드 다운로드



3.

직접 코드 다운로드

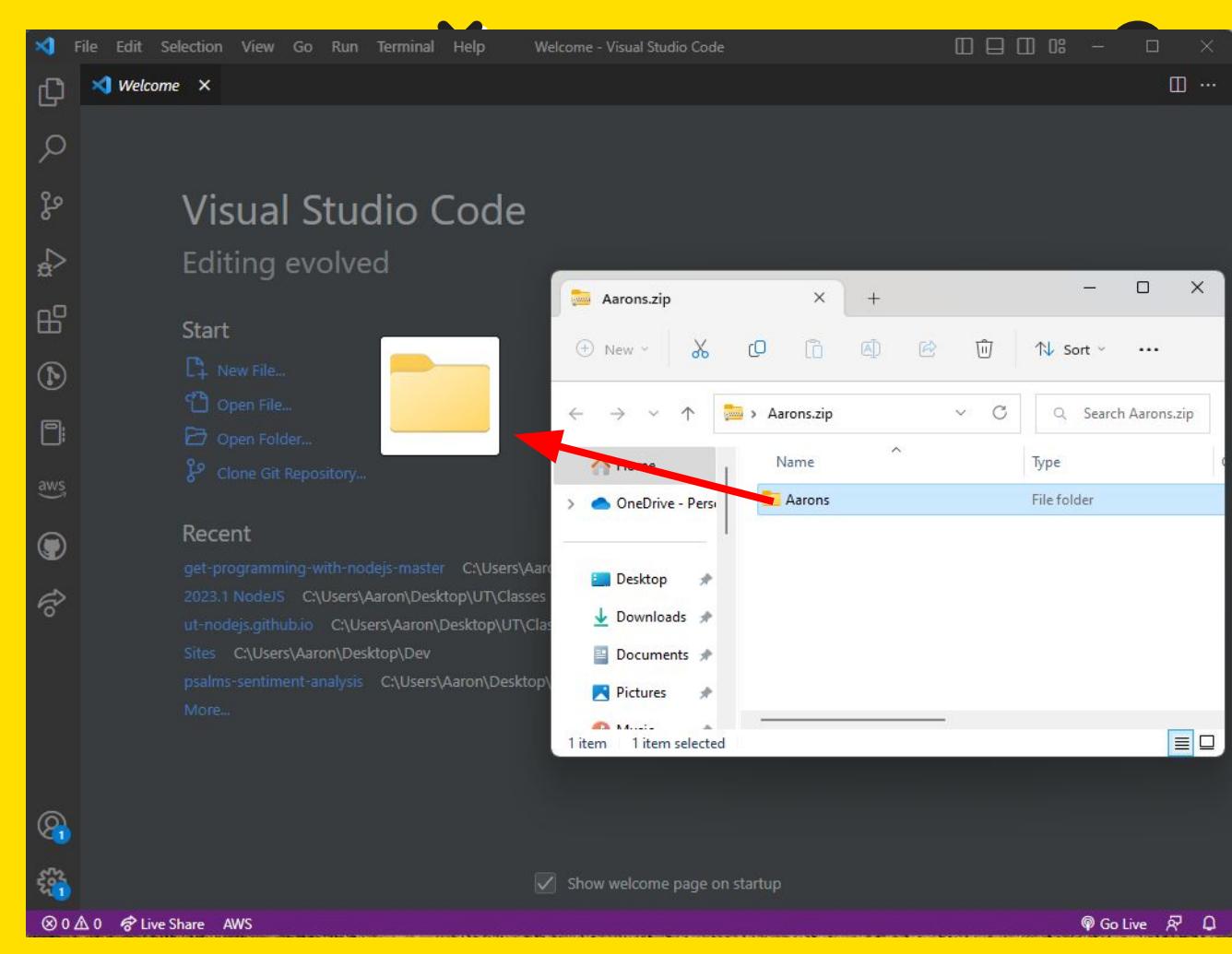
1. 온라인 저장소로 가고
2. 다운로드 ZIP
3. 압축
4. VS Code로 열고
5. VS Code터미널에서
 - a. git remote add URL



3.

직접 코드 다운로드

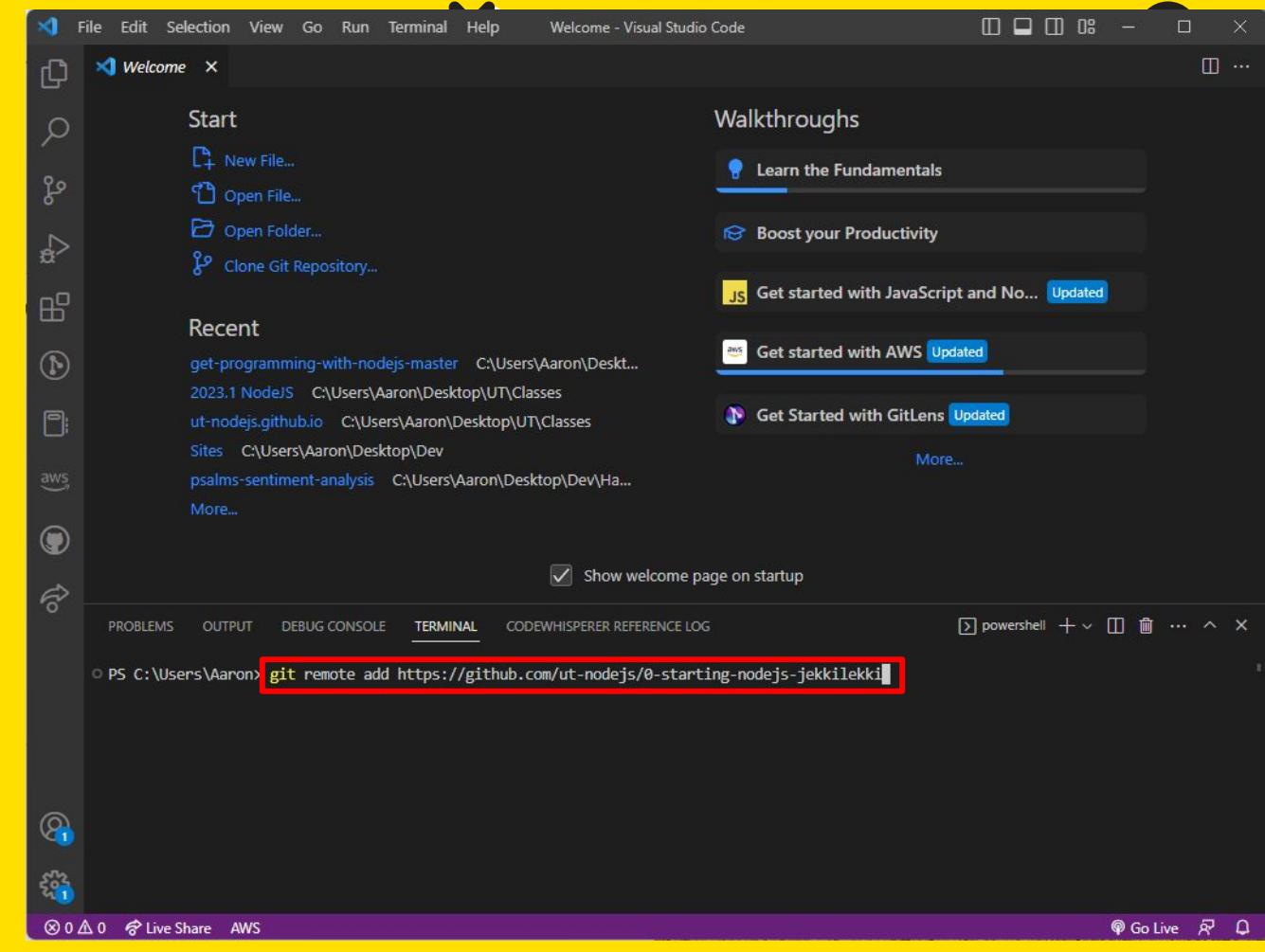
1. 온라인 저장소로 가고
2. 다운로드 ZIP
- 3. 압축**
- 4. VS Code로 열고**
5. VS Code터미널에서
a. git remote add URL



3.

직접 코드 다운로드

1. 온라인 저장소로 가고
2. 다운로드 ZIP
3. 압축
4. VS Code로 열고
5. **VS Code**터미널에서
a. git remote add URL



02

Submit

과제 제출



과제를 어떻게 제출할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



- 1. VS Code**에서 직접 **commit & push**
- 2. 명령줄**에서 **git commit & push**
- 3. 직접 코드 업로드**





과제를 어떻게 제출할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



- 1. VS Code에서 직접 `commit & push`**
- 2. 명령줄에서 `git commit & push`**
- 3. 직접 코드 업로드**





1.

VS Code에서 git commit git push

1. 과제 참여 링크,
코드 받았다
2. 코딩,
3. commit & push
4. 과제 제출

The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows a project structure with files like `node_modules`, `solutions`, `tests` (containing `messages.test.js`, `numbers.test.js`, `printtest.js`), `web-programming-review` (containing `css`, `html`, `js` files), `.gitignore`, `messages.js`, `numbers.js` (selected), `package-lock.json`, `package.json`, `print.js`, and `readme.md`.
- Terminal Tab:** The `TERMINAL` tab is selected.
- Terminal Output:**

```
Counting objects: 100% (10/10), done.
Delta compression using up to 12 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (10/10), 3.36 KiB | 1.12 MiB/s, done.
Total 10 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/ut-nodejs/solutions.git
 * [new branch] main -> main
branch 'main' set up to track 'origin/main'.
(base) PS C:\Users\ADMIN\Desktop\UT\Classes\2023.1-UT-NodeJS\과제\solutions>
```





1.

VS Code에서 git commit git push

1. 과제 참여 링크,
코드 받았다
2. 코딩,
3. **commit & push**

The screenshot shows the VS Code interface with the following details:

- Source Control View:** A red box highlights the "Commit" button in the message input field.
- Terminal View:** Shows the command-line output of a git push operation:

```
Counting objects: 100% (10/10), done.  
Delta compression using up to 12 threads  
Compressing objects: 100% (9/9), done.  
Writing objects: 100% (10/10), 3.36 KiB | 1.12 MiB/s, done.  
Total 10 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/ut-nodejs/solutions.git  
 * [new branch] main -> main  
branch 'main' set up to track 'origin/main'.  
(base) PS C:\Users\ADMIN\Desktop\UT\Classes\2023.1-UT-NodeJS\과제\solutions>  
* History restored.
```

•





1.

VS Code에서 git commit git push

1. 과제 참여 링크,
코드 받았다
2. 코딩,
3. commit & push
4. 오류 생기면
- 5.

The screenshot shows the Visual Studio Code interface with several windows open:

- Source Control**: A message box says "Message (Ctrl+Enter to commit on...)" with a "Commit" button highlighted by a red box.
- Changes**: A list of files including "readme.md".
- Terminal**: A modal window titled "Visual Studio Code" with a red "X" icon. It displays the message "Git에서 'user.name' 및 'user.email'을 구성하세요." (Please configure 'user.name' and 'user.email' in Git) and three buttons: "Git 로그 열기" (Open Git Log), "자세한 정보" (More Information), and "취소" (Cancel).
- Editor**: A file named "readme.md" with Korean text about creating TODO sections and upgrading menus.
- Output**: A list of log entries:
 - 37+ ## 보충 면습 (선택)
 - 38+
 - 39+ 1. GitHub 헤더 생성 완료
 - + (오른쪽)
- Bottom Status Bar**: Shows the command line: "(base) PS C:\Users\ADMIN\Desktop\UT\Classes\2023.1-UT-NodeJS\과제\solutions>" and a note: "* History restored".





VS Code에서 git commit git push

1. 과제 참여 링크,
코드 받았다
2. 코딩,
commit & push
3. 오류 생기면
user.name 입력
4. **user.email** 입력

The screenshot shows the VS Code interface with the following details:

- Source Control** pane: A message box says "Message (Ctrl+Enter to commit on...)" with a red box around it. Below it is a "Commit" button with a dropdown arrow.
- Terminal**:
 - Shows the command: \$ git config --global user.name "Min Seop"
 - Shows the command: \$ git config --global user.email "snsgr3341@gmail.com"
 - An empty line: \$ |
 - At the bottom, it shows the output of a git push command:

```
To https://github.com/ut-nodejs/solutions.git
 * [new branch] main -> main
branch 'main' set up to track 'origin/main'.
(base) PS C:\Users\ADMIN\Desktop\UT\Classes\2023.1-UT-NodeJS\과제\solutions>
 * History restored
```





과제를 어떻게 제출할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



1. **VS Code**에서 직접 **commit & push**
2. 명령줄에서 **git commit & push**
3. 직접 코드 업로드





2.

명령줄에서

git commit git push

1. git init

2. git remote add origin <URL>
3. git add .
4. git commit -m "MSG"
5. git push -u origin master

```
MINGW64:/c/Users/Aaron/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main
Aaron@DESKTOP-PTD9GI3 MINGW64 ~
$ cd ~/Desktop/UT/Classes/2023.1\ NodeJS/0-web-programming-review-node-start-jekkilekki-main
Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main
$ git init
Initialized empty Git repository in C:/Users/Aaron/Desktop/UT/classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main/.git/
Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main (master)
$
```



2.

명령줄에서

git commit git push

1. git init
2. **git remote add origin <URL>**
3. git add .
4. git commit -m "MSG"
5. git push -u origin master



```
MINGW64:/c/Users/Aaron/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main

Aaron@DESKTOP-PTD9GI3 MINGW64 ~
$ cd ~/Desktop/UT/Classes/2023.1\ NodeJS/0-web-programming-review-node-start-jekkilekki-main

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main
$ git init
Initialized empty Git repository in C:/Users/Aaron/Desktop/UT/classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main/.git/

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main (master)
$ git remote add origin https://github.com/ut-nodejs/0-web-programming-review-node-start-jekkilekki

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main (master)
$ |
```



2.

명령줄에서

git commit git push

1. git init
2. git remote add origin <URL>
3. **git add .**
4. git commit -m "MSG"
5. git push -u origin master



```
MINGW64:/c/Users/Aaron/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main
$ git init
Initialized empty Git repository in C:/Users/Aaron/Desktop/UT/classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main/.git/

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main (master)
$ git remote add origin https://github.com/ut-nodejs/0-web-programming-review-node-start-jekkilekki

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main (master)
$ git add .
warning: in the working copy of '.github/classroom/autograding.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of '.github/workflows/classroom.yml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'messages.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'numbers.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package-lock.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'print.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'readme.md', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tests/messages.test.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tests/numbers.test.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tests/print.test.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'web-programming-review/web-programming-review.css', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'web-programming-review/web-programming-review.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'web-programming-review/web-programming-review.js', LF will be replaced by CRLF the next time Git touches it

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main (master)
$ |
```



2.

명령줄에서

git commit git push

1. git init
2. git remote add origin <URL>
3. git add .
- 4. git commit -m "MSG"**
5. git push -u origin master



```
MINGW64:/c/Users/Aaron/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkileksi-main
warning: in the working copy of 'print.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'readme.md', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tests/messages.test.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tests/numbers.test.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tests/print.test.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'web-programming-review/web-programming-review.css', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'web-programming-review/web-programming-review.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'web-programming-review/web-programming-review.js', LF will be replaced by CRLF the next time Git touches it

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkileksi-main (master)
$ git commit -m "Demo"
[master (root-commit) c76aa50] Demo
 15 files changed, 3716 insertions(+)
 create mode 100644 .github/classroom/autograding.json
 create mode 100644 .github/workflows/classroom.yml
 create mode 100644 .gitignore
 create mode 100644 messages.js
 create mode 100644 numbers.js
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 print.js
 create mode 100644 readme.md
 create mode 100644 tests/messages.test.js
 create mode 100644 tests/numbers.test.js
 create mode 100644 tests/print.test.js
 create mode 100644 web-programming-review/web-programming-review.css
 create mode 100644 web-programming-review/web-programming-review.html
 create mode 100644 web-programming-review/web-programming-review.js

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkileksi-main (master)
$
```



2.

명령줄에서

git commit git push

1. git init
2. git remote add origin <URL>
3. git add .
4. git commit -m "MSG"
5. **git push -u origin master**

```
[master (root-commit) c76aa50] Demo
 15 files changed, 3716 insertions(+)
 create mode 100644 .github/classroom/autograding.json
 create mode 100644 .github/workflows/classroom.yml
 create mode 100644 .gitignore
 create mode 100644 messages.js
 create mode 100644 numbers.js
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 print.js
 create mode 100644 readme.md
 create mode 100644 tests/messages.test.js
 create mode 100644 tests/numbers.test.js
 create mode 100644 tests/print.test.js
 create mode 100644 web-programming-review/web-programming-review.css
 create mode 100644 web-programming-review/web-programming-review.html
 create mode 100644 web-programming-review/web-programming-review.js

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main (master)
$ git push -u origin master
Enumerating objects: 22, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 16 threads
Compressing objects: 100% (19/19), done.
Writing objects: 100% (22/22), 35.65 KiB | 2.38 MiB/s, done.
Total 22 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/ut-nodejs/0-web-programming-review-node-start-jekkilekki/pull/new/master
remote:
To https://github.com/ut-nodejs/0-web-programming-review-node-start-jekkilekki
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

Aaron@DESKTOP-PTD9GI3 MINGW64 ~/Desktop/UT/Classes/2023.1 NodeJS/0-web-programming-review-node-start-jekkilekki-main (master)
$ |
```



과제를 어떻게 제출할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



- 1. VS Code**에서 직접 **commit & push**
- 2. 명령줄에서 git push**
- 3. 직접 코드 업로드**

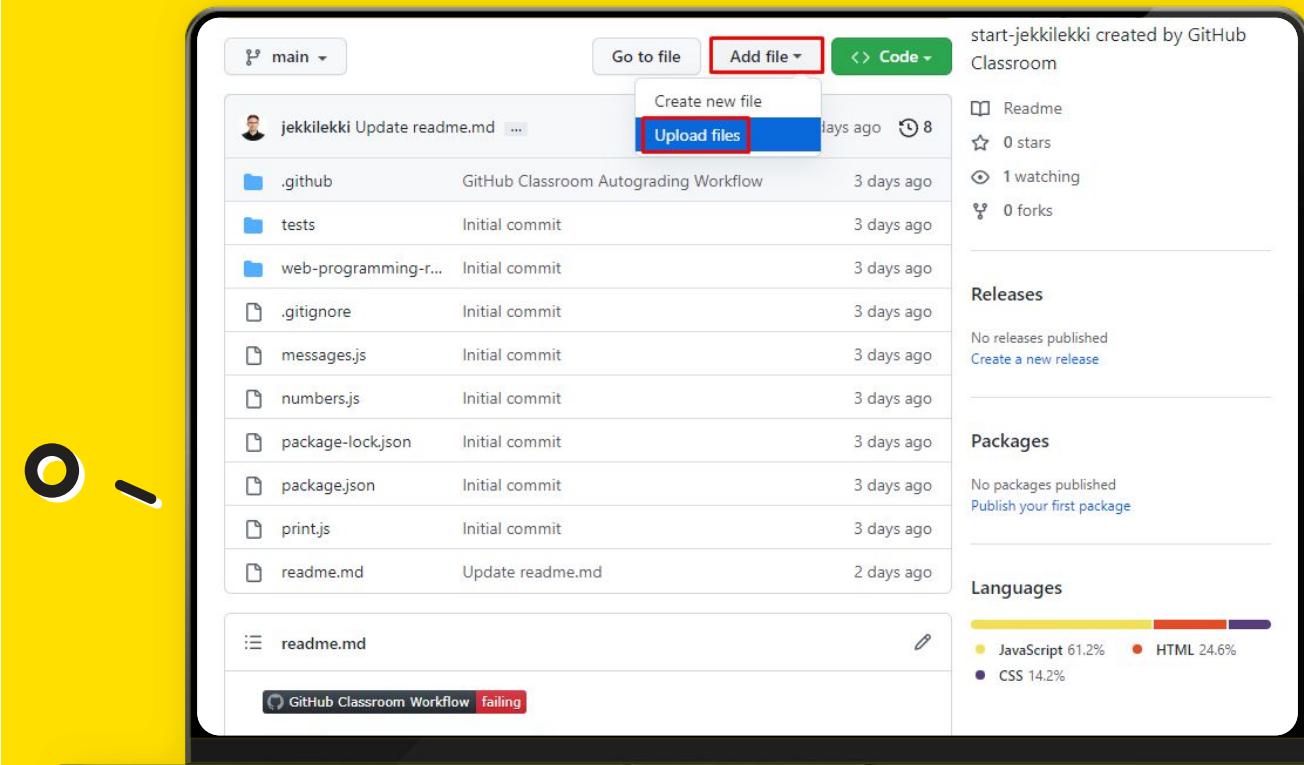




3.

직접 코드 업로드

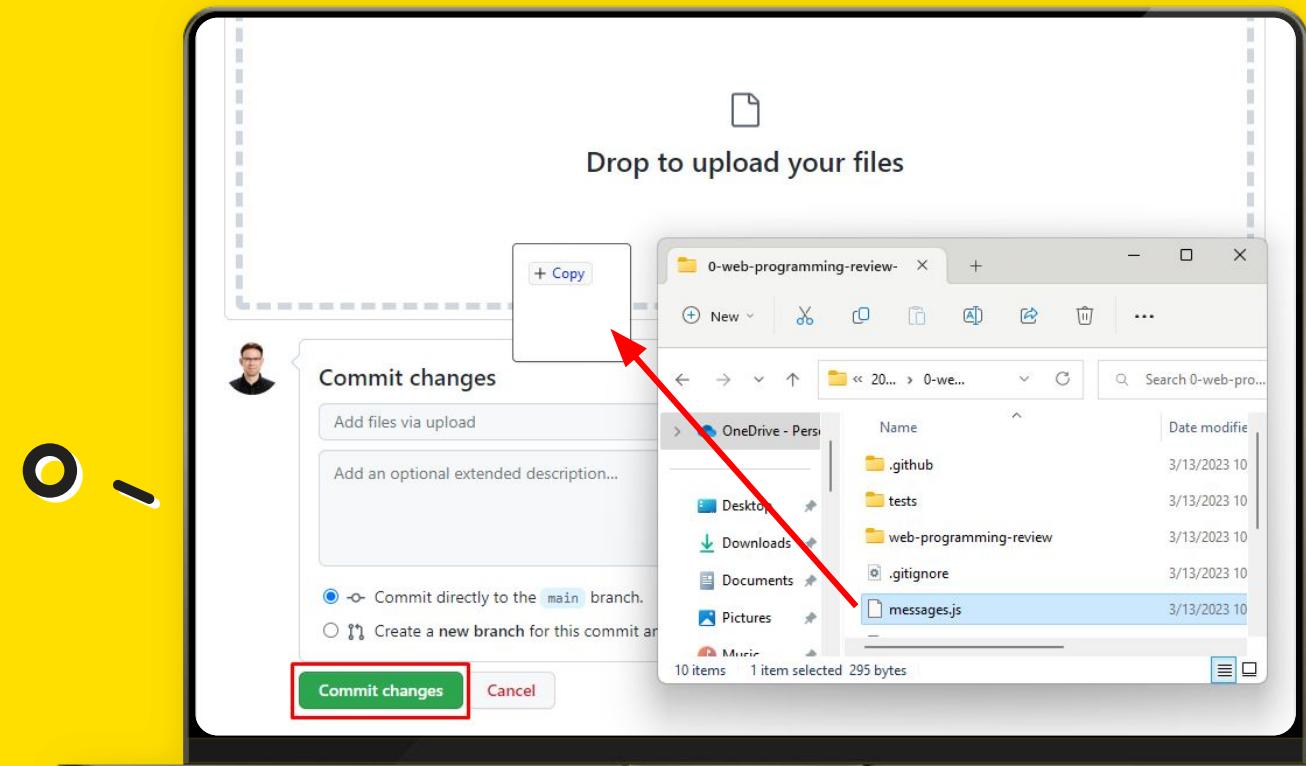
1. 저장소에서 “Add files -> Upload files”
2. Commit changes



3.

직접 코드 업로드

1. 저장소에서 "Add files -> Upload files"
2. Commit changes



03

Check

테스트 결과 확인



과제를 어떻게 확인할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



1. 로컬 컴퓨터 명령줄에서 코드 실행
2. 명령줄에서 **npm install + npm test**
3. 온라인 **GitHub Actions**에서

* **PASS/FAIL** 배지 추가 하면?





과제를 어떻게 확인할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



1. 로컬 컴퓨터 명령줄에서 코드 실행
2. 명령줄에서 **npm install + npm test**
3. 온라인 **GitHub Actions**에서
* **PASS/FAIL** 배지 추가 하면?





1.

로컬 컴퓨터 명령줄에서 코드 실행

- 함수를 생성하기
- 함수를 돌리기





과제를 어떻게 확인할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



1. 로컬 컴퓨터 명령줄에서 코드 실행
2. 명령줄에서 **npm install + npm test**
3. 온라인 **GitHub Actions**에서
* **PASS/FAIL** 배지 추가 하면?





2.

명령줄에서

npm install + npm test

1. npm install는 Jest 테스팅 패키지 설치
2. npm test는 Jest 테스팅 패키지 실행



The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure named "0-STARTING-NODEJS". It includes folders for "node_modules", "solutions", "tests", "web-programming-review", and files like "messages.test.js", "numbers.test.js", "print.test.js", "messages.js", "numbers.js", "package-lock.json", "package.json", "print.js", and "readme.md".
- Code Editor:** Displays the content of "messages.js". The code defines an array "messages" with three strings and a function "printMsgs" that logs each message to the console.
- Terminal:** Shows the command "npm install" being run in a PowerShell terminal. The output indicates that 278 packages were audited and 30 packages are looking for funding.
- Status Bar:** At the bottom, it shows "Ln 22, Col 1" and other settings like "Spaces: 4", "UTF-8", "CRLF", and "JavaScript".





2.

명령줄에서

npm install + npm test

1. npm install는 Jest 테스팅 패키지 설치
2. npm test는 Jest 테스팅 패키지 실행

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure named "0-STARTING-NODEJS" containing files like node_modules, solutions, tests (messages.test.js, numbers.test.js, print.test.js), web-programming-review (css, html, js files), .gitignore, messages.js, numbers.js, package-lock.json, packagejson, printjs, and readme.md.
- Code Editor:** Displays the file "messages.js" with the following content:

```
// messages.js
'use strict';

// @TODO: 3개의 문자열 messages 배열 만들어 주세요.
let messages = [
  'Hello',
  'How are you?',
  'I'm fine thank you, and you?'
];

// @TODO: messages 배열에서 각 메시지를 인쇄하는 함수를 만들어 주세요.
let printMsgs = () => {
  messages.forEach(msg => console.log(msg));
};


```
- Terminal:** Shows the command "npm test" being run in the terminal tab, with the output:

```
30 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
(base) PS C:\Users\ADMIN\Desktop\UT\Classes\2023.1-UT-NodeJS\과제\0-starting-nodejs> npm test

> UT NodeJS 2023 - Web Programming Review@20230307 test
> jest

PASS  tests/messages.test.js
  ● Console

    console.log
      Hello

      at console.<anonymous> (node_modules/jest-mock/build/index.js:709:23)
      at Array.forEach (<anonymous>)


```
- Status Bar:** Shows the file "main*" is open, 0 errors or warnings, and the code is in JavaScript mode.



2.

명령줄에서

**npm install +
npm test**

1. npm install는 Jest 테스팅 패키지 설치
2. **npm test**는 Jest 테스팅 패키지 실행



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "0-STARTING-NODEJS". It includes folders for "node_modules", "solutions", and "tests", and files like "messages.test.js", "numbers.test.js", "print.test.js", "web-programming-review.css", "web-programming-review.html", "web-programming-review.js", ".gitignore", "messages.js", "numbers.js", "package-lock.json", "package.json", "print.js", and "readme.md".
- Terminal:** The title bar indicates "messages.js - 0-starting-nodejs - Visual Studio Code". The terminal tab is active, showing the command "npm test".
- Output:** The terminal output shows the results of the Jest tests:
 - A green "PASS" message for "tests/messages.test.js":
 - Console log: "Hello"
 - At console.<anonymous> (node_modules/jest-mock/build/index.js:709:23)
at Array.forEach (<anonymous>)
 - Console log: "How are you?"
 - At console.<anonymous> (node_modules/jest-mock/build/index.js:709:23)
at Array.forEach (<anonymous>)
 - Console log: "I'm fine thank you, and you?"
 - At console.<anonymous> (node_modules/jest-mock/build/index.js:709:23)
at Array.forEach (<anonymous>)
 - A red "FAIL" message for "tests/numbers.test.js":
 - Console log was called 10 times
 - TypeError: printNumbers is not a function
 - Code snippet from numbers.test.js:

```
4 |   const logSpy = jest.spyOn(global.console, 'log');
5 |
6 |   printNumbers();
7 |   ^
8 |   expect(logSpy).toHaveBeenCalledTimes(10);
9 | });
```
 - At Object.printNumbers (tests/numbers.test.js:6:5)





2.

명령줄에서

npm install + npm test

1. npm install는 Jest 테스팅 패키지 설치
2. npm test는 Jest 테스팅 패키지 실행

File Edit Selection View Go Run ... messages.js - 0-starting-nodejs - Visual Studio Code

EXPLORER PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

powershell + Live Share

4 | const logSpy = jest.spyOn(global.console, 'log');

5 |

6 | printNumbers();

7 |

8 | expect(logSpy).toHaveBeenCalledTimes(10);

9 |});

at Object.printNumbers ([tests/numbers.test.js:6:5](#))

FAIL tests/print.test.js

● Console log was called at least 3 times

expect(jest.fn()).toHaveBeenCalledTimes(expected)

Expected number of calls: 3

Received number of calls: 0

6 | printChar();

7 |

8 | expect(logSpy).toHaveBeenCalledTimes(3);

9 |});

at Object.toHaveBeenCalledTimes ([tests/print.test.js:8:20](#))

Test Suites: 2 failed, 1 passed, 3 total

Tests: 2 failed, 1 passed, 3 total

Snapshots: 0 total

Time: 1.878 s

Ran all test suites.

(base) PS C:\Users\ADMIN\Desktop\UT\Classes\2023.1-UT-NodeJS\과제\0-starting-nodejs>

OUTLINE TIMELINE

main* Live Share

Ln 22, Col 1 Spaces: 4 UTF-8 CRLF {} JavaScript



과제를 어떻게 확인할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



1. 로컬 컴퓨터 명령줄에서 코드 실행
2. 명령줄에서 **npm install + npm test**
3. 온라인 **GitHub Actions**에서

* **PASS/FAIL** 배지 추가 하면?





3.

온라인 GitHub Actions에서

1. 온라인 저장소로 가고
2. “Actions”탭 클릭
3. 최신 Workflow 클릭
4. “Autograding” 클릭
5. “Run autograding” 클릭
6. 테스트 확인

The screenshot shows a GitHub repository interface. At the top, there are tabs for Code, Issues, Pull requests, and Actions. The Actions tab is highlighted with a red box. Below the tabs, there are buttons for main, Go to file, Add file, and Code. On the left, there's a sidebar with a dropdown for main. The main content area displays a list of recent commits:

Commit	Message	Time Ago
github-classroom[bot]	Add online IDE url	1 minute ago
.github	GitHub Classroom Autograding Workflow	1 minute ago
tests	Initial commit	1 minute ago
web-programming-r...	Initial commit	1 minute ago
.gitignore	Initial commit	1 minute ago
messages.js	Initial commit	1 minute ago
numbers.js	Initial commit	1 minute ago
package-lock.json	Initial commit	1 minute ago
package.json	Initial commit	1 minute ago
print.js	Initial commit	1 minute ago
readme.md	Add online IDE url	1 minute ago

On the right side, there are sections for About, Releases, and Packages. The About section shows the repository was created by GitHub Classroom. The Releases section says "No releases published" and "Create a new release". The Packages section says "No packages published" and "Publish your first package".





3.

온라인 GitHub Actions에서

1. 온라인 저장소로 가고
2. “Actions”탭 클릭
3. 최신 Workflow 클릭
4. “Autograding” 클릭
5. “Run autograding” 클릭
6. 테스트 확인

The image shows a smartphone displaying the GitHub Actions interface. The screen displays a list of workflow runs for the "GitHub Classroom Workflow". There are four workflow runs listed:

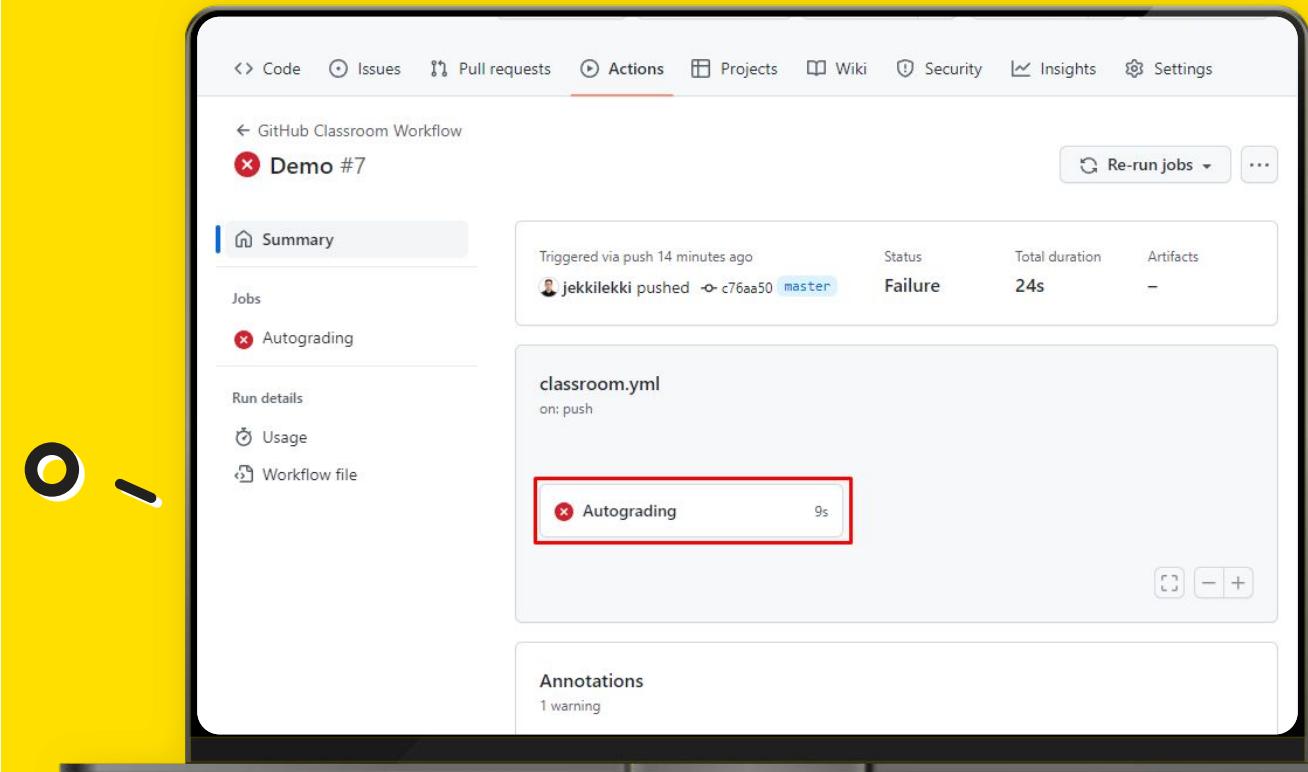
Event	Status	Branch	Actor	Timestamp	Duration
✖ Demo	Success	master	jekkileksi	14 minutes ago	24s
✖ Update readme.md	Success	main	jekkileksi	2 days ago	24s
✖ Update readme.md	Success	main	jekkileksi	2 days ago	27s
✖ Update readme.md	Success	main	jekkileksi	2 days ago	26s



3.

온라인 GitHub Actions에서

1. 온라인 저장소로 가고
2. “Actions”탭 클릭
3. 최신 Workflow 클릭
4. “Autograding” 클릭
5. “Run autograding” 클릭
6. 테스트 확인

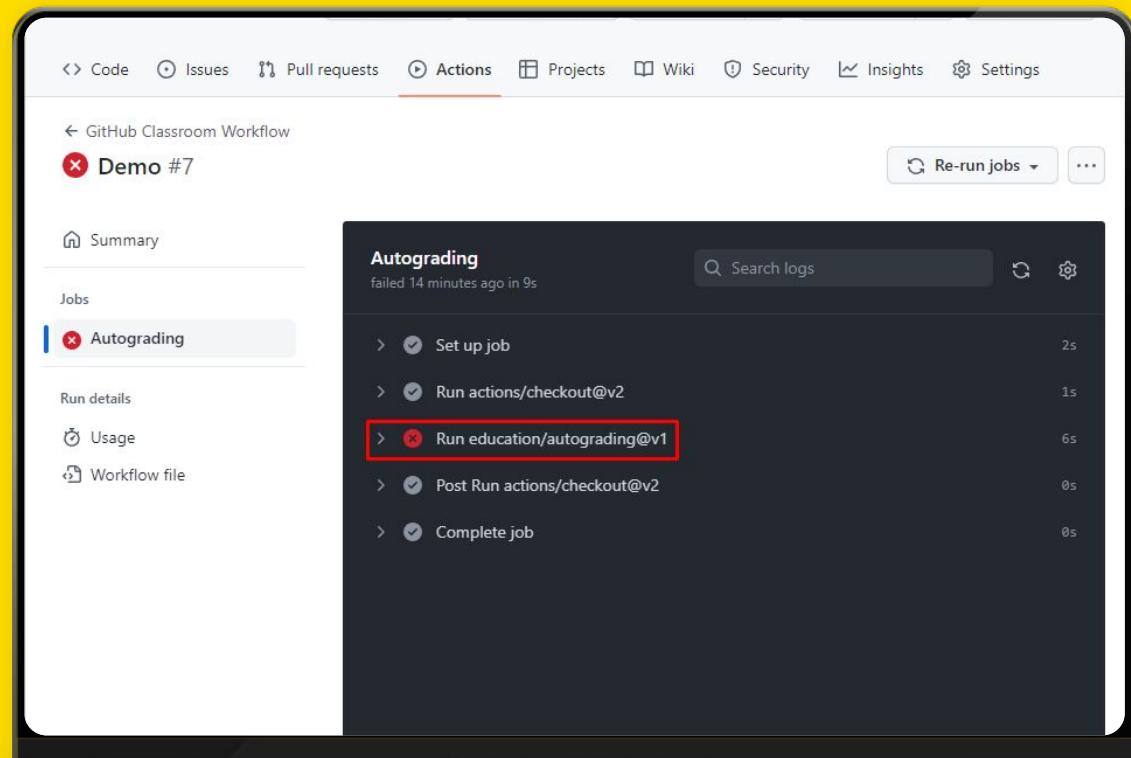




3.

온라인 GitHub Actions에서

1. 온라인 저장소로 가고
2. “Actions”탭 클릭
3. 최신 Workflow 클릭
4. “Autograding” 클릭
5. **“Run autograding”** 클릭
6. 테스트 확인





3.

온라인 GitHub Actions에서

1. 온라인 저장소로 가고
2. “Actions”탭 클릭
3. 최신 Workflow 클릭
4. “Autograding” 클릭
5. “Run autograding” 클릭
6. 테스트 확인

Run details

Usage

Workflow file

```
50      |   ^
51      7 |
52      8 |     expect(logSpy).toHaveBeenCalledTimes(10);
53      9 | });
54
55      at Object.printNumbers (tests/numbers.test.js:6:5)
56
57 FAIL  tests/print.test.js
58   • Console log was called at least 3 times
59
60   expect(jest.fn()).toHaveBeenCalled(expected)
61
62     Expected number of calls: 3
63     Received number of calls: 0
64
65     6 |     printChar();
66     7 |
67     > 8 |     expect(logSpy).toHaveBeenCalledTimes(3);
68     |
69     9 | });
70
71      at Object.toHaveBeenCalledTimes (tests/print.test.js:8:20)
72
73 Test Suites: 3 failed, 3 total
74 Tests:       3 failed, 3 total
75 Snapshots:   0 total
76 Time:        0.56 s
77 Ran all test suites.
78
79 ✘ Test console.log output
```





3.

온라인 GitHub Actions에서

1. 온라인 저장소로 가고
2. “Actions”탭 클릭
3. 최신 Workflow 클릭
4. “Autograding” 클릭
5. “Run autograding” 클릭
6. 테스트 확인

The screenshot shows the GitHub Actions interface. At the top, there are navigation tabs: Code, Issues, Pull requests, Actions (which is underlined in red), Projects, Security, and Insights. Below the tabs, there's a search bar labeled "Filter workflow runs". On the left, there's a sidebar with "Actions" selected, showing "All workflows" and "GitHub Classroom Workflow". Under "Management", there are links for "Caches". The main area is titled "All workflows" and "Showing runs from all workflows". It displays "21 workflow runs" with the following details:

Event	Status	Branch	Actor	Time	Details
Update print.test.js	Success (green checkmark)	main	aaronkr-trainer	yesterday 20s	Github Classroom Workflow #21: Commit 833689e pushed by aaronkr-trainer
Update numbers.test.js	Failure (red X)	main	aaronkr-trainer	yesterday 20s	Github Classroom Workflow #20: Commit 5f41e85 pushed by aaronkr-trainer
Update messages.test.js	Failure (red X)	main	aaronkr-trainer	yesterday 17s	Github Classroom Workflow #19: Commit ed1e10f pushed by aaronkr-trainer
Update messages.test.js	Failure (red X)	main	aaronkr-trainer	yesterday 17s	Github Classroom Workflow #18: Commit 2042cca pushed by aaronkr-trainer





과제를 어떻게 확인할 수 있나요?

문제를 생기면? 다른 방법으로 해 보세요.



1. 로컬 컴퓨터 명령줄에서 코드 실행
2. 명령줄에서 **npm install + npm test**
3. 온라인 **GitHub Actions**에서

* **PASS/FAIL** 배지 추가 하면?

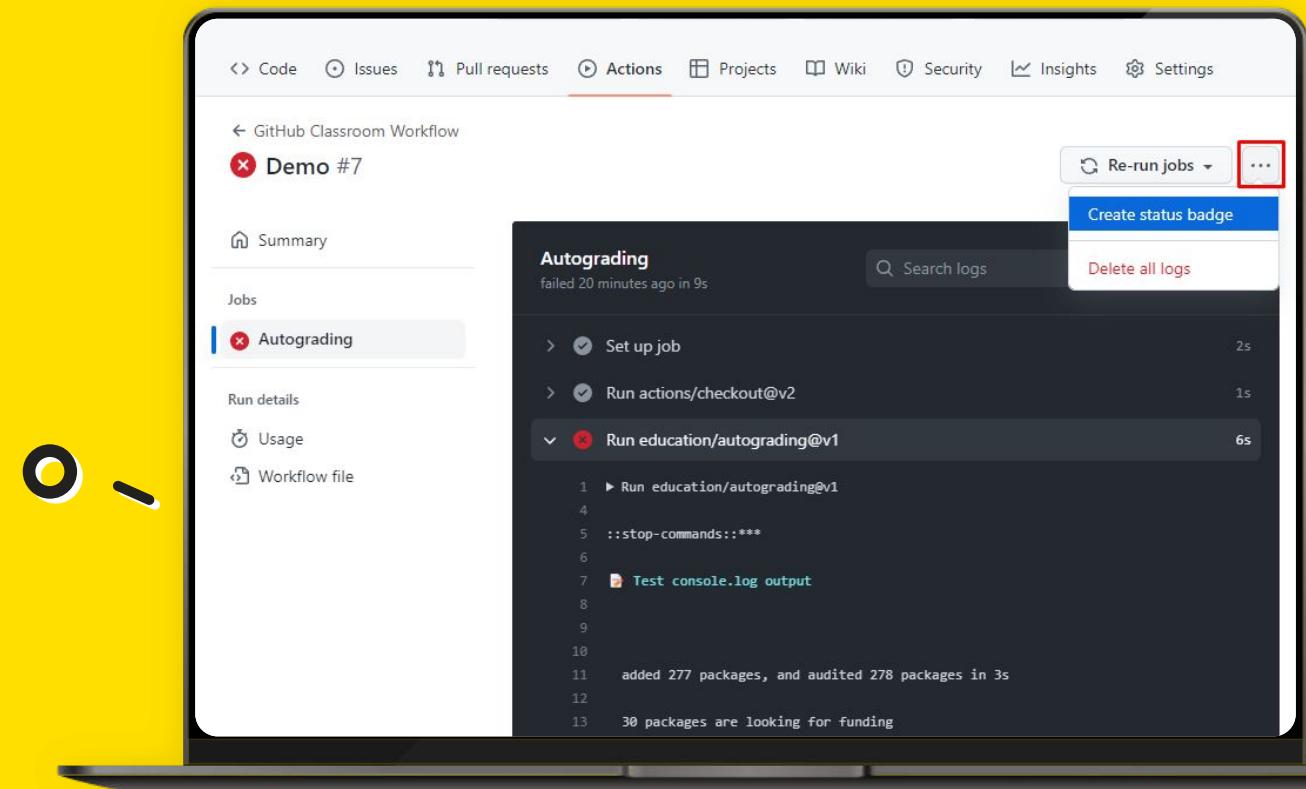




PASS/FAIL

페이지 추가 하면

1. Autograde에서
2. "... ->
Create status badge
3. Markdown 코드 복사
4. 저장소에서
5. readme.md 편집
6. Markdown 코드 부쳐놓기



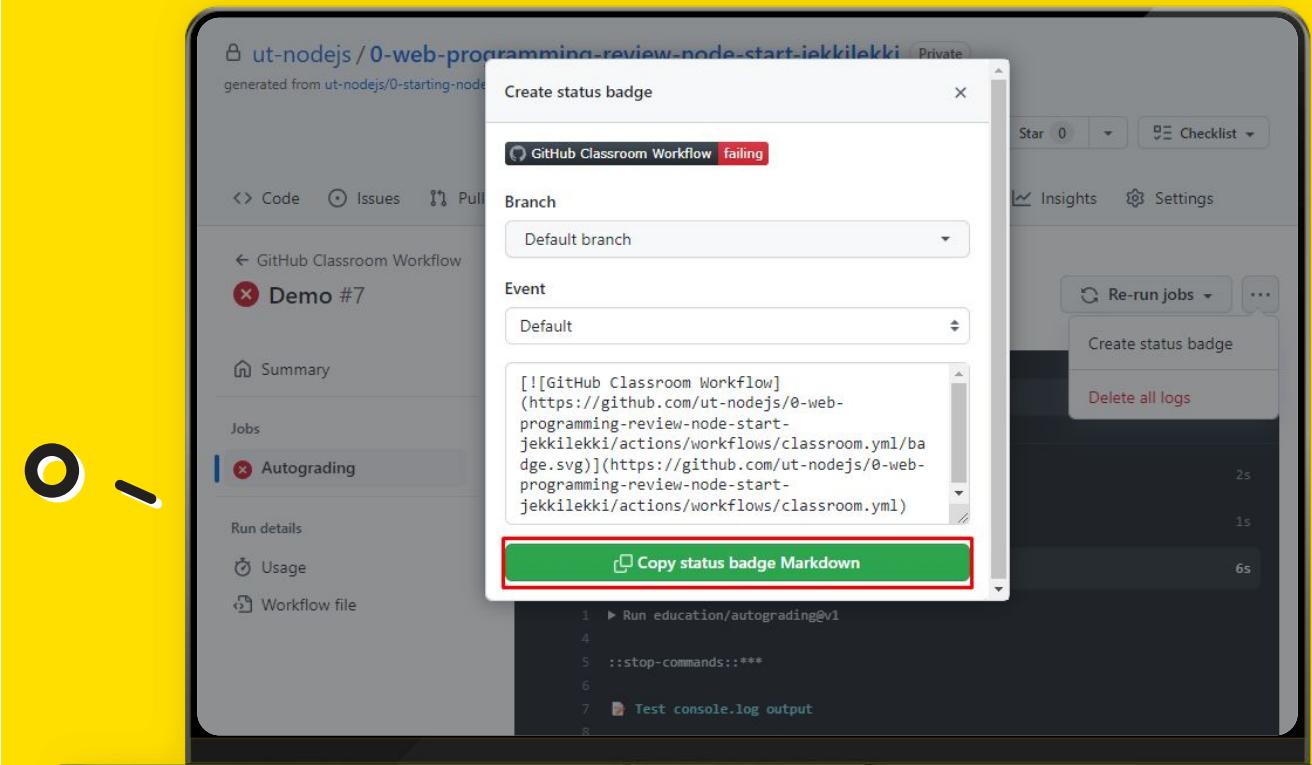


PASS/FAIL

페이지 추가 하면

1. Autograding에서
“...” ->
2. Create status badge
3. **Markdown 코드 복사**

4. 저장소에서
5. readme.md 편집
6. Markdown 코드 부쳐놓기





PASS/FAIL

페이지 추가 하면

1. Autograding에서
“...” ->
2. Create status badge
3. Markdown 코드 복사

4. 저장소에서
readme.md 편집
5. Markdown 코드 부쳐놓기
6. Markdown 코드 부쳐놓기

The screenshot shows a GitHub repository page for a user named jekkilekki. The repository contains several files: .github, tests, web-programming-r..., .gitignore, messages.js, numbers.js, package-lock.json, package.json, print.js, and readme.md. The readme.md file is highlighted with a red border. Below the file list, there is a preview of the README.md content, which includes a GitHub Classroom Workflow badge labeled "failing". At the bottom of the preview, there is a button labeled "Open in Visual Studio Code". To the right of the preview, there is a red-bordered edit icon. On the right side of the repository page, there are sections for "Releases" (no releases published), "Packages" (no packages published), and "Languages" (JavaScript 61.2%, HTML 24.6%, CSS 14.2%).

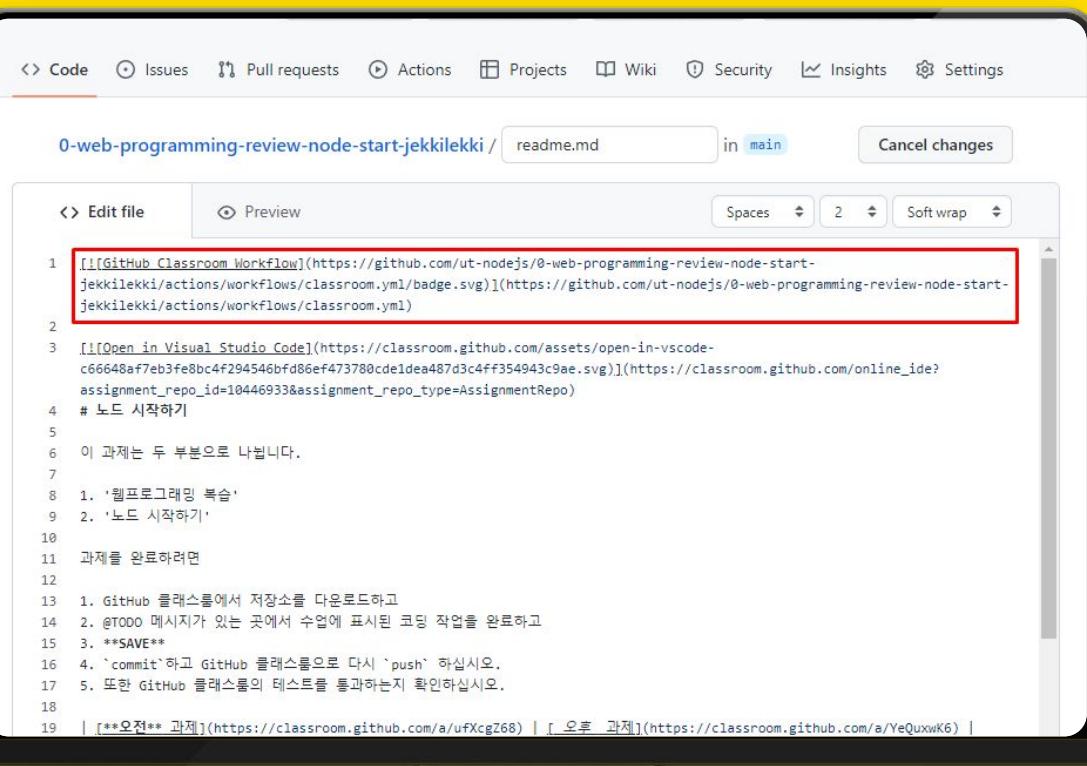




PASS/FAIL

배지 추가 하면

1. Autograding에서
“...” ->
2. Create status badge
3. Markdown 코드 복사
4. 저장소에서
5. readme.md 편집
6. **Markdown** 코드
부쳐놓기



```
1 [![GitHub Classroom Workflow](https://github.com/ut-nodejs/0-web-programming-review-node-start-jekkilekki/actions/workflows/classroom.yml/badge.svg)](https://github.com/ut-nodejs/0-web-programming-review-node-start-jekkilekki/actions/workflows/classroom.yml)
2
3 [![Open in Visual Studio Code](https://classroom.github.com/assets/open-in-vscode-c66648af7eb3fe8bc4f294546bfd86ef473780cde1dea487d3c4ff354943c9ae.svg)](https://classroom.github.com/online_ide?assignment_repo_id=104466933&assignment_repo_type=AssignmentRepo)
4 # 노드 시작하기
5
6 이 과제는 두 부분으로 나뉩니다.
7
8 1. '헬프로그래밍 복습'
9 2. '노드 시작하기'
10
11 과제를 완료하려면
12
13 1. GitHub 클래스룸에서 저장소를 다운로드하고
14 2. @TODO 메시지가 있는 곳에서 수업에 표시된 코딩 작업을 완료하고
15 3. **SAVE**
16 4. `commit`하고 GitHub 클래스룸으로 다시 `push` 하십시오.
17 5. 또한 GitHub 클래스룸의 테스트를 통과하는지 확인하십시오.
18
19 | [**오전** 과제](https://classroom.github.com/a/ufXcgZ68) | [__오후__ 과제](https://classroom.github.com/a/Ye0uxwK6) |
```





PASS/FAIL

페이지 추가 하면

1. Autograding에서
“...” ->
2. Create status badge
3. Markdown 코드 복사
4. 저장소에서
5. readme.md 편집
6. **Markdown** 코드
부쳐놓기

A screenshot of a GitHub Classroom repository for "GitHub Classroom Autograding Workflow". The repository has the following file history:

File	Commit Message	Time
.github	GitHub Classroom Autograding Workflow	last week
tests	Update print.test.js	last week
messages.js	Update messages.js	last week
numbers.js	Update numbers.js	last week
package-lock.json	Add files via upload	last week
package.json	Add files via upload	last week
print.js	Update print.js	last week
readme.md	Update readme.md	2 days ago

The `readme.md` file contains the following content:

```
GitHub Classroom Workflow passing
```

Below the file list, there is a message in Korean: "노드 시작하기" (Starting Node) and a link: "https://github.com/ut-nodejs/0-starting-node-aaronkr-trainer".

On the right side of the repository page, there are sections for "Readme", "Releases", "Packages", and "Contributors".

Readme:
- Readme
- 0 stars
- 1 watching
- 0 forks

Releases:
No releases published
Create a new release

Packages:
No packages published
Publish your first package

Contributors (2):
- aaronkr-trainer (Avatar)
- jekkilekki Aaron Snowberger (Avatar)



Questions?

한번 해 보자!~