

GitHub Tutorial

'24W

송 인 식

Outline

- 원격 저장소
- Github Workflow
- 프로젝트 Fork하기
- 협업하기
- Issue 처리하기

GitHub?

- Git 기반의 버전 관리와 협업 기능을 제공하는 소스 코드 호스팅 플랫폼
- GitHub 주요 기능
 - Git 원격 저장소
 - 코드 리뷰
 - 프로젝트 관리
 - 문서화 위키
 - 오픈 소스 프로젝트 & 커뮤니티 & 이슈 토론

Github Repository를 local로 clone해서 작업하기

- Github에 이미 등록된 원격 저장소가 있다면 local machine으로 clone해 온다.

```
# download a repository on GitHub to our machine
# Replace `owner/repo` with the owner and name of the repository to clone
git clone https://github.com/owner/repo.git

# change into the `repo` directory
cd repo

# create a new branch to store any new changes
git branch my-branch

# switch to that branch (line of development)
git checkout my-branch

# make changes, for example, edit `file1.md` and `file2.md` using the text editor

# stage the changed files
git add file1.md file2.md

# take a snapshot of the staging area (anything that's been added)
git commit -m "my snapshot"

# push changes to github
git push --set-upstream origin my-branch
```

Local Repository 를 GitHub에 Push하기

- 이미 로컬 저장소가 있다면 Github에 push한다.
 - Local 머신에서 remote repository를 등록한 후 git push 명령을 이용하여 변경 내용을 원격 저장소에 반영한다.

```
# create a new directory, and initialize it with git-specific functions
git init my-repo

# change into the `my-repo` directory
cd my-repo

# create the first file in the project
touch README.md

# git isn't aware of the file, stage it
git add README.md

# take a snapshot of the staging area
git commit -m "add README to initial commit"

# provide the path for the repository you created on github
git remote add origin https://github.com/YOUR-USERNAME/YOUR-REPOSITORY-NAME.git

# push changes to github
git push --set-upstream origin main
```

GitHub의 특정 branch에 작업 내용 올리기

- Github과 로컬 저장소가 동시에 존재하는 경우 local에서 작업한 내용을 github의 특정 branch에 반영한다.

```
# change into the `repo` directory
cd repo

# update all remote tracking branches, and the currently checked out branch
git pull

# change into the existing branch called `feature-a`
git checkout feature-a

# make changes, for example, edit `file1.md` using the text editor

# stage the changed file
git add file1.md

# take a snapshot of the staging area
git commit -m "edit file1"

# push changes to github
git push
```

두 가지 협업 모델

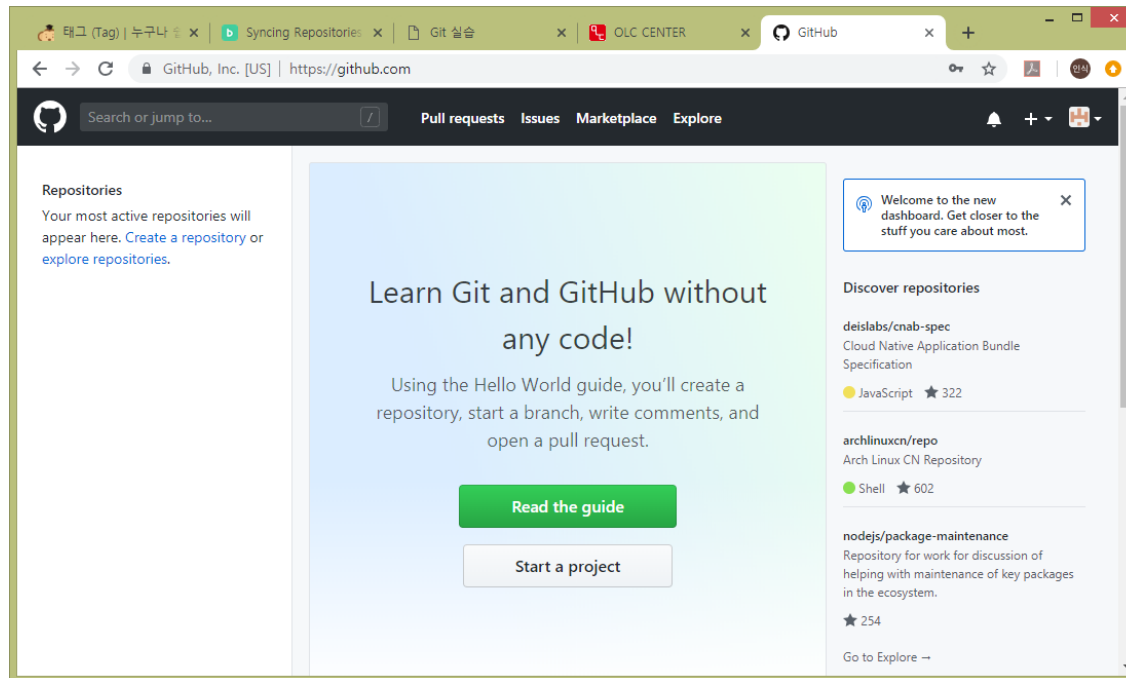
- 공유 repository
 - 폐쇄형 공동 작업 모델로 작업자는 모두 프로젝트에 기여자로 등록되어야 하며 적절한 read, write, 또는 관리자 액세스 권한을 가져야 함
- Fork and pull
 - 오픈 소스 프로젝트에 적합한 모델로 본 프로젝트에 기여하고자 하는 기여자는 본 프로젝트의 복사본 프로젝트를 자신의 계정으로 복제 (fork) 하여 작업한다.
 - 이후 변경 내역을 개인적으로 유지하거나 본 프로젝트에 기여하고 싶을 경우 pull request를 통해 원본 프로젝트 소유자가 검토 후 merge할 수 있도록 한다.

Outline

- 원격 저장소
- Github Workflow
- 프로젝트 Fork하기
- 협업하기
- Issue 처리하기

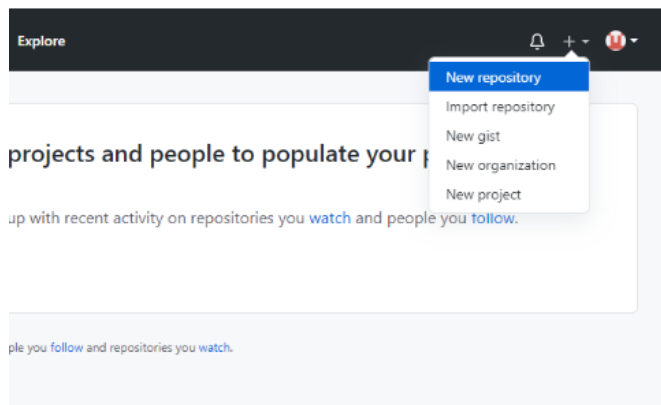
GitHub 계정 등록

- GitHub 사이트에서 계정 등록 (<https://github.com/join>)
 - 이메일 주소와 비밀번호 등록하여 계정 생성
 - Local repository에서 등록한 이메일 계정과 동일한 계정으로 등록할 것)
 - 이메일 인증 후 계정 생성 완료됨



1단계: 새 Repository 만들기

- 로그인 후 오른쪽 위의 + 버튼을 누른 후 **'New repository'** 선택
 - 프로젝트 이름 설정
 - 간단한 설명 입력
 - Public/Private 선택, gitignore 설정, License 설정, README 설정 선택
 - **'Create repository'** 클릭하면 Github에 저장소가 생성됨



Description (optional)

Hello World repository for Git tutorial

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

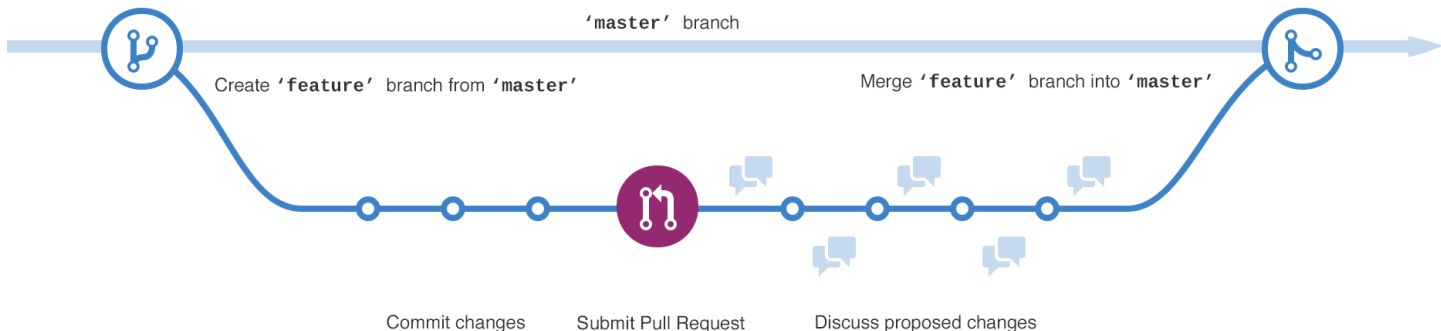
☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

브랜치 만들기

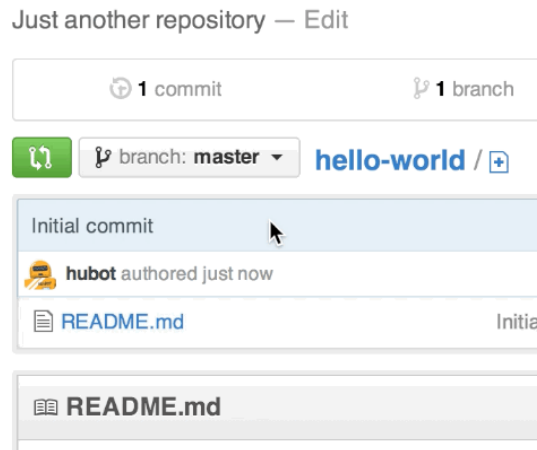
- 브랜치는 특정 시점에 다른 버전의 작업을 할 수 있도록 해 주는 메커니즘
- 저장소의 기본 브랜치는 'master'
- 'master' 브랜치에서 새로운 브랜치를 생성하면 해당 시점의 스냅샷이 생성됨



브랜치 만들기

- 브랜치 생성하기

1. 새로운 저장소 'hello-world'로 이동
2. 파일 리스트의 상단에서 'branch: master' 드롭 다운 항목을 클릭
3. 새 브랜치의 텍스트 상자에 이름 'readme-edits'를 입력
4. 파란 색의 'Create branch' 상자나 클릭하거나 커보드에서 "Enter"를 입력



수정 후 커밋하기

The screenshot shows the GitHub web interface for the repository 'hubot / hello-world'. At the top, there are navigation links for 'Code', 'Issues', 'Pull requests', 'Wiki', 'Pulse', 'Graphs', and 'Settings'. Below these, the file 'hello-world / README.md' is selected. The main area displays the content of the README file, which includes a greeting and a mention of tacos on the moon. At the bottom, the 'Commit changes' dialog is open, showing the commit message 'Finish README' and a description 'And mention moon tacos'. The dialog also shows the option to commit directly to the 'readme-edits' branch.

hubot / hello-world


Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Wiki Pulse Graphs Settings

hello-world / README.md or cancel

Edit file Preview changes Spaces 2 Soft wrap

```
1 # hello-world
2
3 Hi Humans!
4
5 Hubot here, I like Node.js and Coffeescript (that's what I'm made of!).
6 I've had tacos on the moon and find them far superior to Earth tacos.
7
```

 **Commit changes**

Finish README

And mention moon tacos

☒ Commit directly to the `readme-edits` branch


☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)


Commit changes Cancel


Pull Request 열기



- Pull request를 열면 변경을 제안하고 다른 사람들에게 내용을 검토한 후 자신의 기여 내용을 가져다가 병합해 주기를 요청하게 됨
- Pull request에는 두 브랜치의 차이를 나타내는 diff가 보이며 변경, 추가, 삭제 내용이 녹색과 적색으로 보임
- GitHub의 '@mention' 기능을 이용하여 특정인이나 팀의 피드백을 요청할 수 있음
- README의 변경 내역에 대해 pull request 열기
 1. **'Pull request'** 탭을 클릭한 후 Pull request 페이지에서 녹색 **'New pull request'** 버튼을 클릭
 2. 'Example Comparisons' 박스에서 자신이 작성한 브랜치 'readme-edits'를 선택하여 'master'와 비교
 3. Compare 페이지의 diff를 조사하여 변경 내역이 확실한 지 확인
 4. 변경 내역이 확실하면 녹색 큰 버튼 'Create Pull Request'를 클릭
 5. Pull request에 제목과 변경 내용에 대한 간단한 설명을 작성
 6. 메시지 작성이 끝나면 'Create pull request' 를 클릭


Pull Request 열기


 **1** commit


 **1** file changed

 Commits on Oct 27, 2014

  **hubot**

Finish README 

 Showing **1** changed file with **1** addition and **1** deletion.

2  README.md

...	...	@@ -1,4 +1,4 @@
1	1	hello-world
2	2	=====
3	3	
4		-Just another repository
	4	+Hubot here, I like Node.js and Coffee them far superior to Earth tacos.

Pull Request 병합하기

- 'readme-edits' 브랜치를 'master' 브랜치에 병합하기
 1. 변경 내용을 'master' 브랜치에 병합하기 위해 '**Merge pull request**' 버튼을 클릭
 2. 'confirm merge' 를 클릭
 3. 이제 'readme-edits' 브랜치는 더 이상 필요 없으니 보라색 박스의 '**Delete branch**' 버튼을 이용하여 브랜치를 삭제



This branch has no conflicts with the base branch

Merging can be performed automatically.



Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).



Pull request successfully merged and closed

You're all set—the `readme-edits` branch can be safely deleted.



Delete branch

Outline

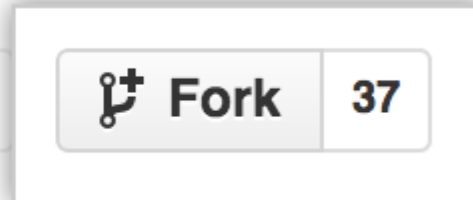
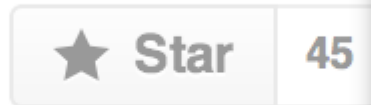
- 원격 저장소
- Github Workflow
- 프로젝트 Fork하기
- 협업하기
- Issue 처리하기

프로젝트에 기여하기

- 다른 프로젝트에 기여하거나 또는 다른 프로젝트를 자신의 시작점으로 삼고자 하는 경우 fork 기능을 이용
- Fork는 다른 사람의 프로젝트를 개인적으로 복사하는 것
- 작업 후 자신의 변경 내역을 적용하도록 pull request를 제출할 수 있음
- Fork는 GitHub을 이용한 협동 코딩의 핵심
- 여기서는 'Spoon-Knife' 프로젝트를 예제로 사용

저장소 포크하기

- Spoon-Knife 저장소를 포크하려면 해당 저장소의 헤더에서 '**Fork**' 버튼을 클릭
- 작업이 끝나면 Spoon-Knife 프로젝트의 사본이 만들어짐



포크를 클론하기

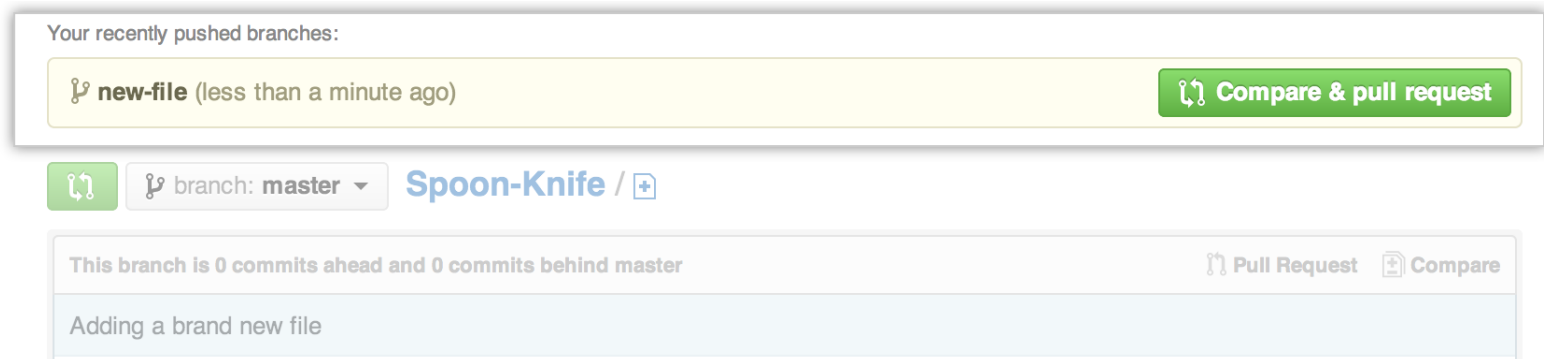
- Fork 만 하면 저장소는 GitHub에만 존재
- 작업을 위해 자신의 컴퓨터로 복제해야 함
- GitHub Desktop을 사용하고 있다면 우측 사이드 바에서 'Clone or Download'를 클릭
- 명령어로 clone할 수도 있음 (`git clone 사용자명@호스트:/원격/저장소/경로`)

수정하고 변경 내용을 Push하기

- 원하는 작업을 마친 후 변경 내용을 스테이징 및 커밋
- 수정이 완료되면 자신의 로컬 저장소의 변경 내용을 원격 저장소(fork한 저장소)에 push (`git push origin master`)
- 여기까지 완료하면 수정 내용을 자신의 GitHub 저장소에 반영한 것

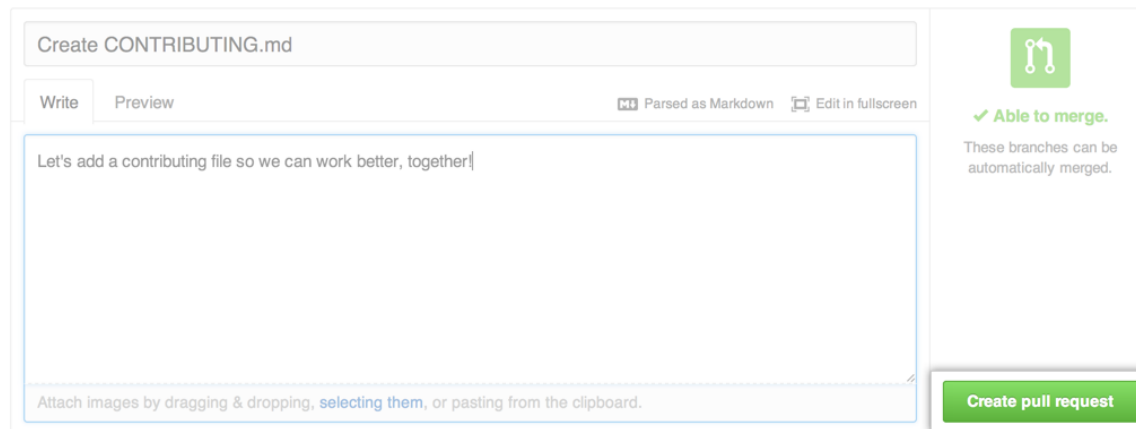
Pull Request 만들기

- 이제 자신의 변경 내용을 메인 프로젝트에 제안할 준비가 된 상태임
- 커뮤니티에 기여하려면 GitHub의 저장소 (https://www.github.com/<your_username>/Spoon-Knife)로 이동
- 최근에 새로운 브랜치를 push했으며 'upstream' 브랜치에 제출할 수 있다는 내용을 확인할 수 있음



Pull Request 만들기

- **'Compare and Pull Request'**를 클릭하여 토론 페이지로 이동
- 여기서 제목과 선택적 설명 입력 가능
- Project owner가 이해할 수 있도록 충분한 정보와 '왜 중요한지'를 제시해야 함
- 준비되면 'Send pull request'를 클릭하면 됨



The screenshot shows the GitHub 'Create pull request' page. At the top, there's a header 'Create CONTRIBUTING.md'. Below it, there are tabs for 'Write' and 'Preview', and links for 'Parsed as Markdown' and 'Edit in fullscreen'. The main text area contains the placeholder text: 'Let's add a contributing file so we can work better, together!'. At the bottom of the text area, there's a note: 'Attach images by dragging & dropping, selecting them, or pasting from the clipboard.' On the right side, there's a green icon of two arrows pointing to each other, followed by the text '✓ Able to merge. These branches can be automatically merged.' At the bottom right, there's a green button labeled 'Create pull request'.

Pull Request 만들기

- Project owner가 바빠서 병합을 못 할 수도 있고, 적당치 않다고 생각하여 거절하거나, 추가적인 정보를 요청할 수도 있음
- 누군가 나중에 도움을 받을 수도 있으니 무조건 '공유'

Outline

- 원격 저장소
- Github Workflow
- 프로젝트 Fork하기
- 협업하기
- Issue 처리하기

협업하기

- GitHub의 사용자나 프로젝트는 굉장히 많고 빠르게 증가하고 있으므로 모든 것을 다 따라갈 수는 없음
- 다른 사용자나 저장소를 관심 있게 지켜 보거나, 별표를 주거나, Explore를 이용하여 새로운 사람이나 프로젝트를 찾을 수 있음

친구 팔로우하기

- 다른 사람을 follow하면 그 사람의 GitHub 활동에 대한 알림을 받을 수 있음
- 'personal dashboard'를 이용하여 친구의 관심 사항을 보거나 Explore 페이지를 이용하여 GitHub 커뮤니티에서 어떤 일이 일어나고 있는 지 파악 가능

친구 팔로우하기

- 원하는 친구를 GitHub에서 찾는다.



cheshire137



charliesome



benbalter



jeffrafter



muan



pifafu

- 원하는 친구의 프로파일에서 'follow' 버튼을 클릭하여 친구가 된다.



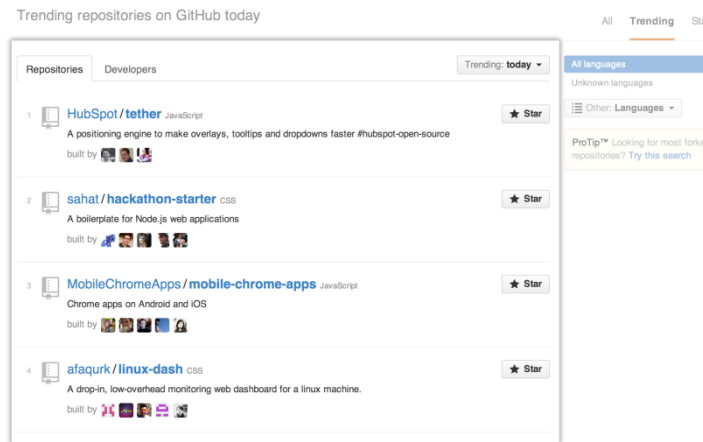
프로젝트 저장소 Watch하기

- 어떤 프로젝트의 최신 상황을 확인하고 싶으면 '사용자 설정'을 통해 특정 저장소의 알림을 이메일 또는 웹으로 확인할 수 있도록 설정 가능
- 보통 pull request에 대한 코멘트가 알림 대상
- 친구 Octocat의 'Hello World' 저장소를 watch하려면 해당 저장소에 가서 페이지 상단의 'watch' 버튼을 클릭
- 이제 'Hello World' 저장소에서 수정 사항이 발생하면 자신의 dashboard에 뜨거나 알림을 받게 됨



추가적인 작업

- 'Stars' 페이지에서 다른 사용자로 이동하기
- 'Discover Repositories'에서 watch, follow, star 관련 개인화된 추천 받기
- 'Explore' 페이지에서 자신이 star 준 사람들이나 GitHub 스타프, 일반적으로 잘 나가는 저장소들을 확인할 수 있고 이들에 대한 정보를 일간, 주간, 월간 뉴스레터로 받을 수도 있음

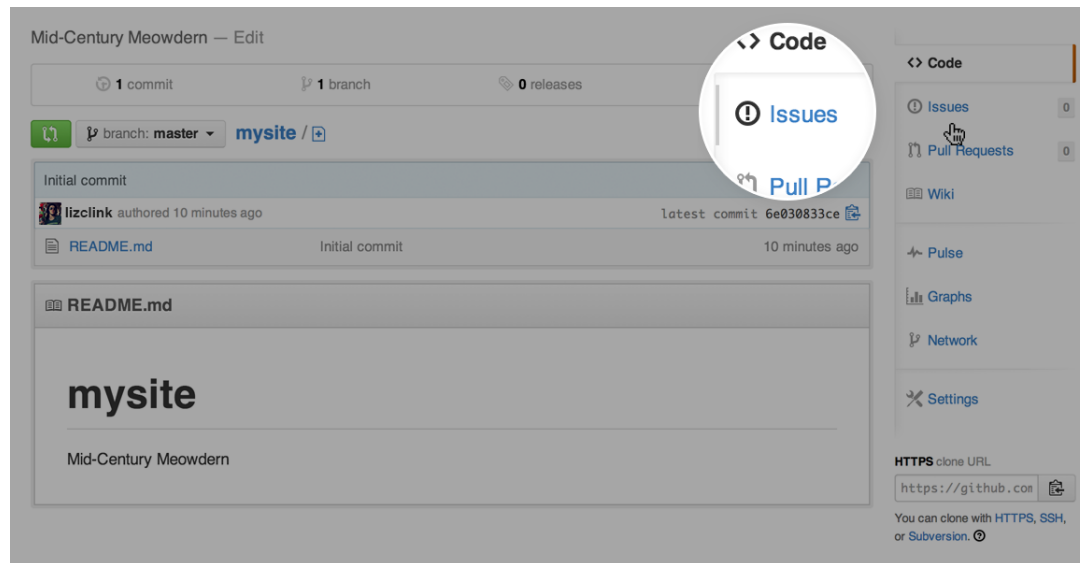


Outline

- 원격 저장소
- Github Workflow
- 프로젝트 Fork하기
- 협업하기
- Issue 처리하기

이슈 처리하기

- 이슈는 작업, 기능 확장, 버그 등을 추적하는 훌륭한 방법
- 이메일 같은 것으로 볼 수 있으나 팀 내 다른 사람들과 공유 및 토론이 가능함
- 대부분의 소프트웨어 프로젝트에는 버그 트래커가 있으며 GitHub의 트래커를 'Issues' 라고 함



Bootstrap의 Issues 섹션

The screenshot displays the GitHub Issues interface for the Bootstrap repository. At the top, there are tabs for 'Issues', 'Pull requests', 'Labels', and 'Milestones'. A search bar contains the text 'is:open is:issue', and a 'New Issue' button is on the right. Below the tabs, a summary bar shows '104 Open' and '9,660 Closed' issues, along with sorting options like 'Author', 'Labels', 'Milestones', 'Assignee', and 'Sort'. The main content area lists several open issues, each with a title, status (e.g., 'confirmed'), category (e.g., 'css', 'js', 'docs'), and a comment count. The issues listed are:

- .form-group-sm .form-group-lg shrink textarea** (confirmed, css) - 4 comments
- Tooltip unnecessarily breaks into multiple lines when positioned to the right** (confirmed, js) - 0 comments
- Tooltip Arrows in Modal example facing wrong way** (css) - 6 comments
- Table improvement** (css) - 0 comments
- docs/dist files** (docs) - 7 comments
- Potential solution to #4647** (js) - 4 comments
- Bootstrap site: right-hand navigation text becomes rasterized after scrolling** (css, docs) - 4 comments
- Dropdown toggle requires two clicks** (js) - 1 comment

GitHub의 전형적인 Issue

◀ The no-conflict mode should be the default behaviour #12395

[Edit](#)[New Issue](#)[Open](#)

thewebdreamer opened this issue 3 days ago · 10 comments



thewebdreamer commented 3 days ago

The no-conflict mode should be the default behaviour. Why would a Bootstrap client need to implement this?



cvrebert commented 3 days ago

I believe no-conflict-is-not-the-default is the norm for jQuery plugins?



thewebdreamer commented 3 days ago

It is true that it is the norm for jQuery plugins.

Couldn't there be a clash with other jQuery plugins with the current implementation of Bootstrap though?

Labels



js

Milestone



No milestone

Assignee



No one assigned

Notifications



Subscribe

3 participants



GitHub의 전형적인 Issue

- Title/description: 제목 및 설명
- 색상이 있는 label: 이슈를 체계적으로 구분
- Milestone: 이슈의 컨테이너, 특정 기능이나 프로젝트 단계와 연관 지어 관리할 때 유용
- 하나의 assignee: 특정 시점에 이슈에 대한 작업 책임자
- Comments: 피드백을 제공하기 위해 액세스 하는 사람 누구나

Milestone, Label, Assignee

- 많은 이슈가 있을 때, milestone, label, assignee를 이용하여 이슈를 분류 및 필터링
- 특정 이슈의 우측 사이드 바에서 해당 기능을 추가할 수 있음

The screenshot shows a GitHub issue page for the title "Open modal is shifting body content to the left #9855". At the top right, there are "Edit" and "New issue" buttons. Below the title, a green "Open" button is followed by the text "mat0r opened this issue 5 months ago · 88 comments". The main content area displays two comments. The first comment, by user "mat0r" (commented 5 months ago), describes a problem where launching a modal component on macOS causes the body content to shift left. The second comment, by user "jamescostian" (commented 5 months ago), confirms the issue on Chrome for Linux. On the right side of the issue, there is a sidebar with three sections: "Labels" (containing "confirmed", "css", and "js"), "Milestone" (set to "v3.1.0"), and "Assignee" (set to "No one assigned").

◀ Open modal is shifting body content to the left #9855 Edit New issue

🔔 Open mat0r opened this issue 5 months ago · 88 comments

mat0r commented 5 months ago

When launching the modal component (<http://getbootstrap.com/javascript/#modals>) the entire content will slightly move to the left on mac OS (haven't tried it on windows yet). With the active modal the scrollbar seem to disappear, while the content width still changes.

You can observer the problem on the bootstrap page

jamescostian commented 5 months ago

I can confirm that I see this also on Chrome for Linux, both on [getbootstrap.com](#) and on the `3.0.0-wip` branch

Labels

- confirmed
- css
- js

Milestone

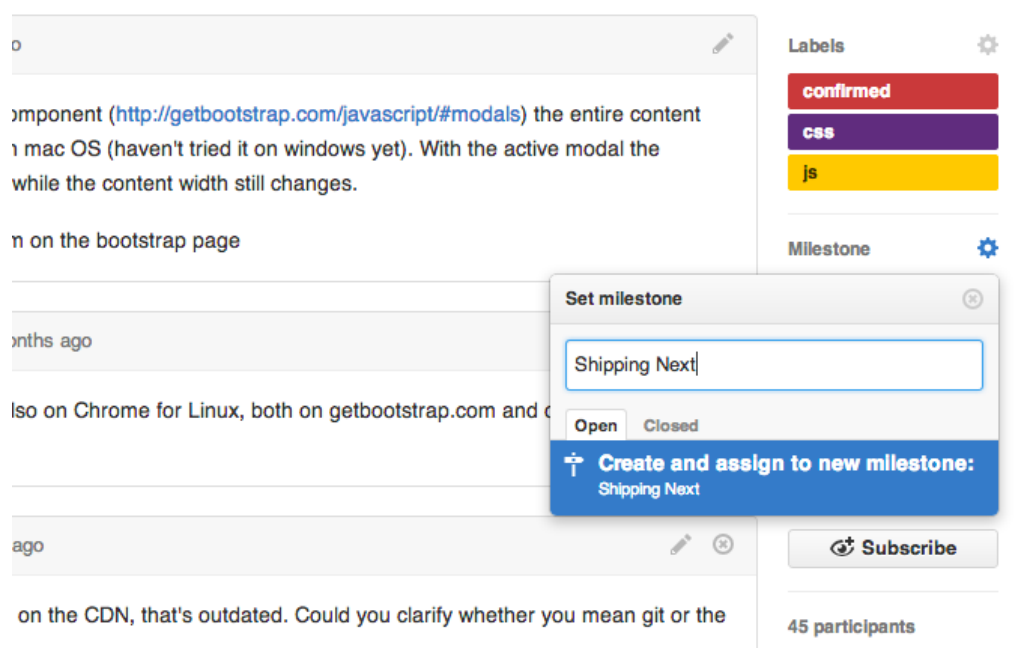
v3.1.0

Assignee

No one assigned























Milestone

- Milestone은 특정 프로젝트, 기능, 시점에 관련된 이슈 집합
- 마일스톤의 예: Beta launch(bug fix), October sprint, Redesign



Label

- 이슈의 종류를 구분
- 임의의 수의 label을 지정할 수 있고 한 번에 하나 또는 여러 label로 필터링 가능

	Open modal is shifting body content to the left   	#9855
Opened by matOr 3 months ago  62 comments		
	Navbar issues   	#11243
Opened by Nugrata a month ago  36 comments		
	Add support of extra styling class on collapse event   	#11350
Opened by ziogaschr 22 days ago  24 comments		
	Redundant responsive utility styles 	#11214
Opened by AlexYursha a month ago  24 comments		
	Select tag not properly styled on stock android browser  	#11055
Opened by ADmad a month ago  19 comments		

Assignee


- 각 이슈는 해당 이슈를 책임 지는 하나의 assignee를 가질 수 있음
- 이슈의 상단에 있는 회색 바를 이용해 milestone 처럼 선택 가능

Notification, @mention, Reference

- 이슈 내의 @mention과 reference를 이용하여 다른 GitHub 사용자나 팀에게 알림을 보낼 수 있고 이슈를 상호 연결할 수 있음
- 이를 통해 적절한 사람을 이슈 해결에 투입시킬 수 있음
- GitHub 내의 모든 텍스트에 적용되며 GitHub의 텍스트 포맷 문법인 Markdown 을 따름
 - Markdown에 대한 자세한 내용은 (<http://guides.github.com/features/mastering-markdown/>) 참조

Notification, @mention, Reference



Please review the [guidelines for contributing](#) to this repository.



This is an example of a new issue

Write

Preview

 Parsed as Markdown  Edit in fullscreen

Text fields on GitHub are parsed with Markdown. This means we can @mention people like @githubstudent or @octocat to reference them into the issue. You can also cross-reference other issues and pull requests with the number of those issues and pull requests like #14020

You can also create tasks list to monitor the progress of smaller tasks as well:

- [] #23
- [] #142
- [] #140

Attach images by dragging & dropping, [selecting them](#), or pasting from the clipboard.

Submit new Issue

Notification

- 이슈의 최신 상태를 알려주는 기능
- 저장소에 관한 새로운 이슈 확인 또는 누군가 이슈 해결에 자신의 도움이 필요할 때 알려주는 기능
- 두 가지 방식: 이메일, 웹 (사용자 설정에서 선택)
 - **Participating**에 대해 web + email, **Watching**에 대해서는 web 알림 설정을 추천
- 'notifications' 화면을 이용하여 많은 알림을 살펴 보고 muted thread (@mention으로 지정되지 않으면 unread 상태로 더 이상 표시되지 않음)로 관리 가능
- 웹과 이메일에서 read/unread 상태 연동됨

@mention

- GitHub 이슈 내에 다른 GitHub 사용자를 참조하는 기능
- 이슈 설명이나 코멘트 내에 다른 사용자의 '@username'을 삽입하면 해당 사용자에게 알림이 전달됨 (Twitter의 @mention 과 유사)
- '/cc'를 사용할 수도 있음

```
It looks like the new widget form is broken on Safari. When I
try and create the widget, Safari crashes. This is
reproducible on 10.8, but not 10.9. Maybe a browser bug?
/cc @kneath @jresig
```

- 특정 사용자를 정확히 모르는 경우에는 GitHub organization의 Team을 사용할 수 있음

```
/cc @acmeinc/browser-bugs
```

Reference

- 이슈가 다른 이슈와 관련이 있을 때 해시태그와 이슈 번호를 이용하여 해당 이슈를 참조 가능

Hey @kneath, I think the problem started in #42

 **mdo** referenced this pull request 2 months ago
Issue #11734: v3.1.0 ship list

Open

- 다른 저장소의 이슈는 저장소 이름과 이슈 번호를 붙여서 쓰면 됨 (kneath/example-project#42)
- 커밋 메시지 안에 이슈 번호를 넣을 수도 있음
- 커밋 메시지에 "Fixes", "Fixed", "Fix", "Closes", "Closed", "Close" 등을 넣어 master에 합병하면 해당 이슈가 자동으로 종료됨

Search

- 각 페이지 상단에 검색 박스가 있어 이슈 검색 가능
 - 키워드, 상태, 책임자 등으로 검색 범위 지정 가능

Issues Pull requests Labels Milestones

Filters New pull request

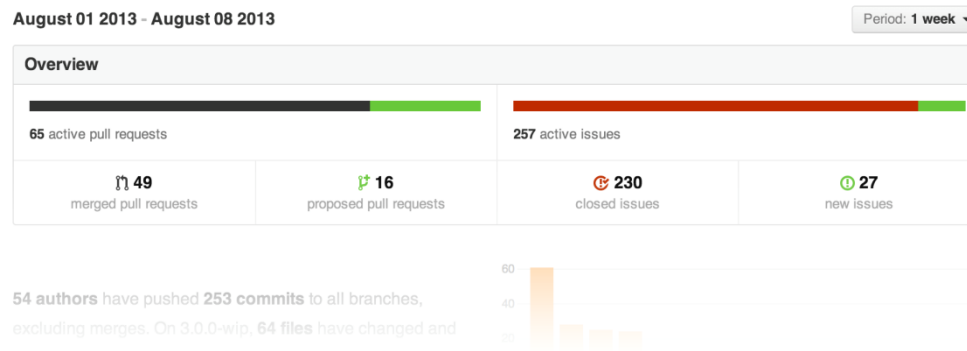
✕ Clear current search query, filters, and sorts

🔗 0 Open ✓ 24 Closed Author ▾ Labels ▾ Milestones ▾ Assignee ▾ Sort ▾

- 🔗 show proper error message when viewing Customizer in IE8 ✓ css customizer 4
#13626 opened on May 19 by cvrebert 🚩 v3.2.0
- 🔗 Fixed affix-bottom positioning ✓ js 4
#13541 opened on May 8 by gpakosz 🚩 v3.2.0
- 🔗 fix background-color on hover of .nav-sidebar link in docs/example/dashboard ✓ css examples 2
#12907 opened on Mar 3 by sevab
- 🔗 Template the customizer's nav sidebar too ✕ customizer grunt 5
#12318 opened on Jan 20 by cvrebert 🚩 v3.1.0
- 🔗 a few javascript.html documentation edits ✓ docs js 14
#10092 opened on Aug 23, 2013 by jodytate

개요 및 보고서

- Issue dashboard를 이용하면 여러 프로젝트의 이슈를 볼 수 있음
 - 자신이 소유 또는 협업 중인 저장소들의 모든 이슈
 - 자신에게 할당된 이슈
 - 자신이 생성한 이슈 등으로 분류
- 각 저장소 아래에 Pulse 섹션이 있어 지난 주, 일, 3개월 등 저장소 에 일어난 모든 내용에 대한 스냅샷을 제공



Questions?