

GITHUB

Hosting Services:-

If we want to start using git, we need to know where to host our repositories.

There are 2 ways you can host your repositories . One is online (on the cloud) and second is offline (self-installed on your server).

There are 3 popular Git hosting services :

Github (owned by Microsoft), Gitlab (owned by Gitlab) and Bitbucket. We will use Github as our hosting service.

Github:- So Github is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

How Github works ?




1. Create a repository

A repository is usually used to organize a single project. Repositories can contain folders and files, images, videos, spreadsheets, and data sets – anything your project needs. We recommend including a README, or a file with information about your project. GitHub makes it easy to add one at the same time you create your new repository. It also offers other common options such as a license file. Your hello-world repository can be a place where you store ideas, resources, or even share and discuss things with others.

To create a new repository:

- In the upper right corner, next to your avatar or identicon, click and then select New repository.
- Name your repository hello-world.
- Write a short description.
- Select Initialize this repository with a README.


Owner **Repository name**


PUBLIC   **hubot** / **hello-world** 

Great repository names are short and memorable. Need inspiration? How about **petulant-shame**.


Description (optional)

Just another repository

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

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

☒ **Initialize this repository with a README**
This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.

Add .gitignore: **None** | Add a license: **None** 

Create repository

Click create repository

2. Create a branch

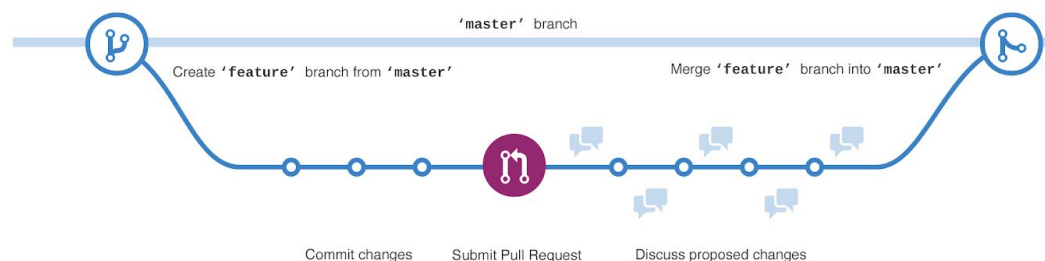
is the way to work on different versions of a repository at one time.

By default your repository has one branch named `main` which is considered to be the definitive branch. We use branches to experiment and make edits before committing them to `main`.

When you create a branch off the `main` branch, you're making a copy, or snapshot, of `main` as it was at that point in time. If someone else made changes to the `main` branch while you were working on your branch, you could pull in those updates.

This diagram shows:

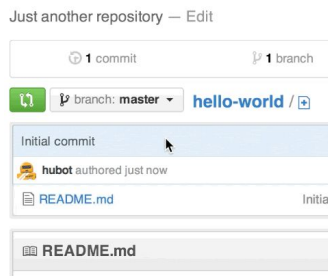
- The `main` branch
- A new branch called `feature` (because we're doing 'feature work' on this branch)
- The journey that `feature` takes before it's merged into `main`



Here at GitHub, our developers, writers, and designers use branches for keeping bug fixes and feature work separate from our main (production) branch. When a change is ready, they merge their branch into main.

To create a new branch :

- Go to your new repository hello-world.
- Click the drop down at the top of the file list that says branch: main.
- Type a branch name, readme-edits, into the new branch text box.
- Select the blue Create branch box or hit “Enter” on your keyboard



Now you have two branches, main and readme-edits. They look exactly the same, but not for long! Next we'll add our changes to the new branch.

3. Make and commit changes

Now, you're on the code view for your readme-edits branch, which is a copy of main. Let's make some edits.

On GitHub, saved changes are called commits. Each commit has an associated commit message, which is a description explaining why a particular change was made. Commit messages capture the history of your changes, so other contributors can understand what you've done and why.

Make and commit changes:

Click the README.md file.

Click the pencil icon in the upper right corner of the file view to edit.

In the editor, write a bit about yourself.

Write a commit message that describes your changes.

Click Commit changes button.

hubot / hello-world

Unwatch 1Star 0Fork 0


CodeIssues 0Pull requests 0WikiPulseGraphsSettings

hello-world / README.md or cancel

Edit filePreview changes

Spaces2Soft 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 changesCancel

These changes will be made to just the README file on your readme-edits branch, so now this branch contains content that's different from main.

4. Open a pull request

orid

Unwatch 1Star 0Fork 0

95 0Pull requests 0WikiPulseGraphsSettings

hello-world / README.mdFind fileCopy path

DME7106852 3 minutes ago

EXAMPLE COMPARISONS	
	4 minutes ago
	24 hours ago

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.

base: master ▼ ... compare: readme-ed

Create pull request Discuss and review the

base: master ▼ ... compare: readme-edits ▼

DT

Readme edits

Write Preview

Content for non-telepathic human.

When you're done with your message, click Create pull request!

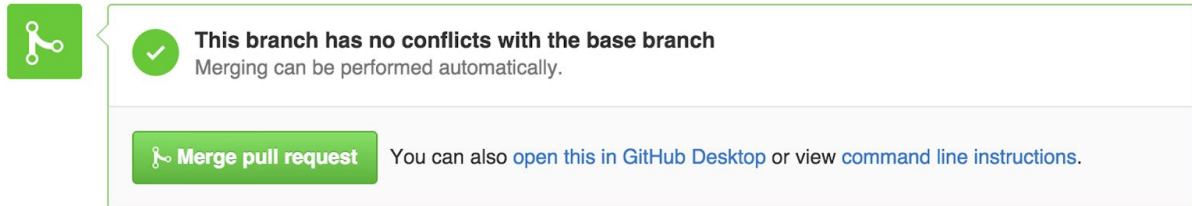
5. Merge your pull request

In this final step, it's time to bring your changes together – merging your readme-edits branch into the main branch.

Click the green Merge pull request button to merge the changes into main.

Click Confirm merge.

Go ahead and delete the branch, since its changes have been incorporated, with the Delete branch button in the purple box.



Celebrate!

In Short:

- Fork the repository
- Clone the repository

(For example: `git clone https://github.com/sanscript-tech/orphan_support-php.git`)

- Create a branch

(For example: `git checkout -b dev_username`)

- Do the necessary changes.
- Add those changes to the branch you just created and commit those changes.

(`git commit -m "<Add your message here>"`)

- Push changes to GitHub

(`git push origin <branch-name>`)

- Submit your changes for review

Soon we will be merging all your changes into the master branch of this project. You will get a notification email once the changes have been merged.