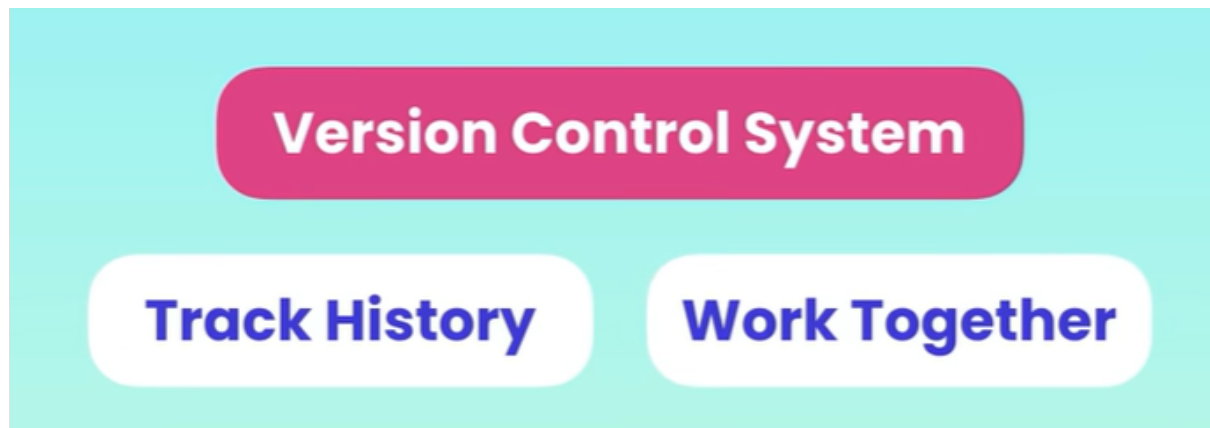


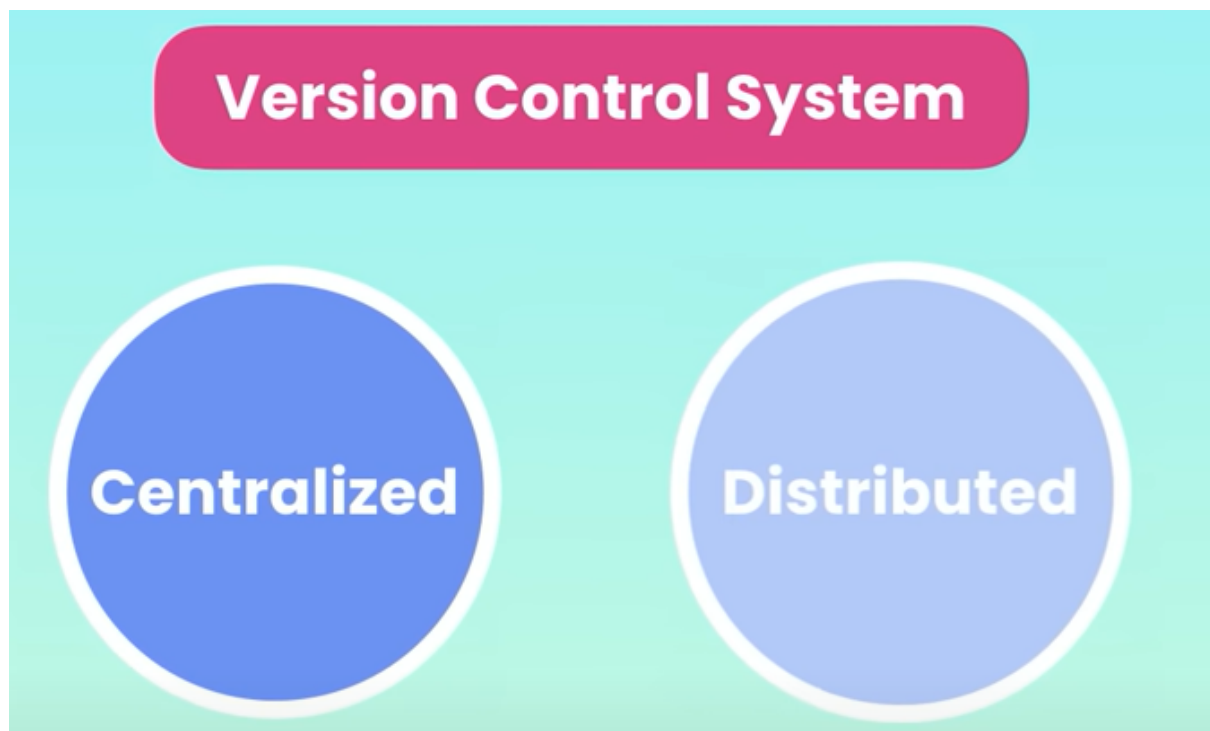
Git is the most popular and widely used version control system in the world and it is free and open source. Version control system is used to keep track of the changes that we have made to code in a special database called “repository”.

We can see the project history, who has made particular changes and why did he make changes like that. If we have made a mistake in check in then we can easily revert back our changes into earlier state.

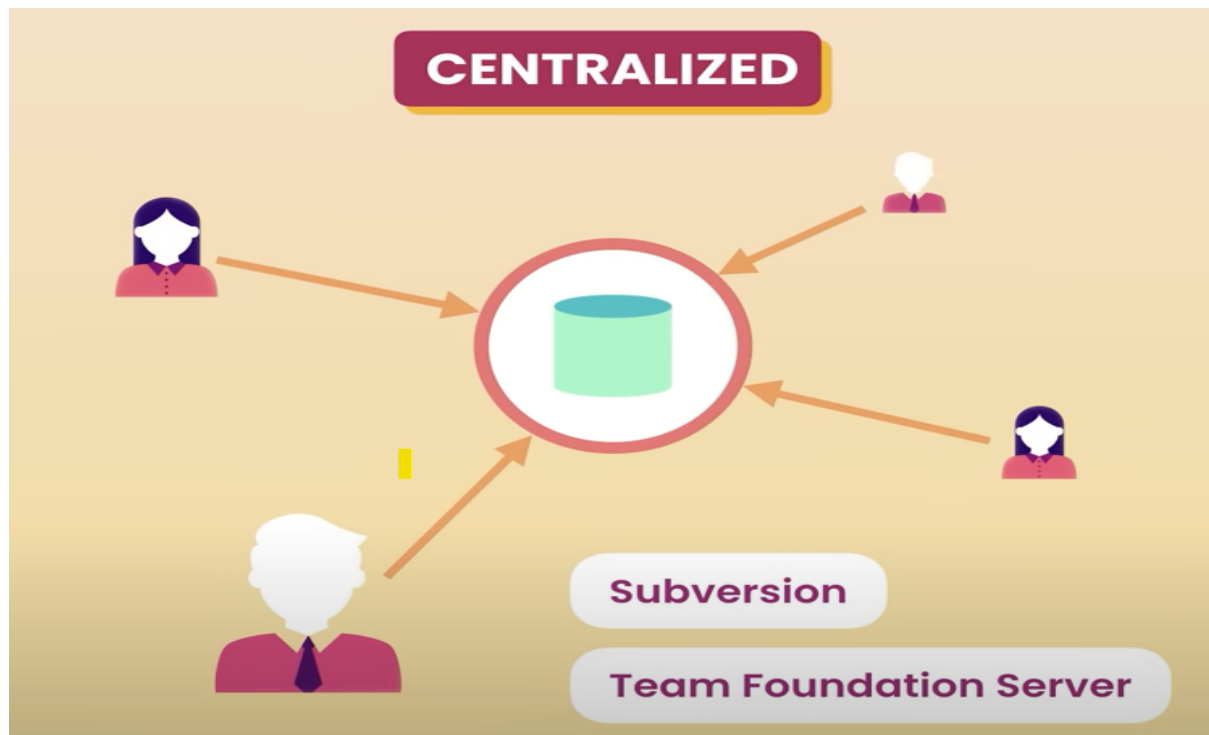


The version control system falls into two categories:

1. Centralised
2. Distributed

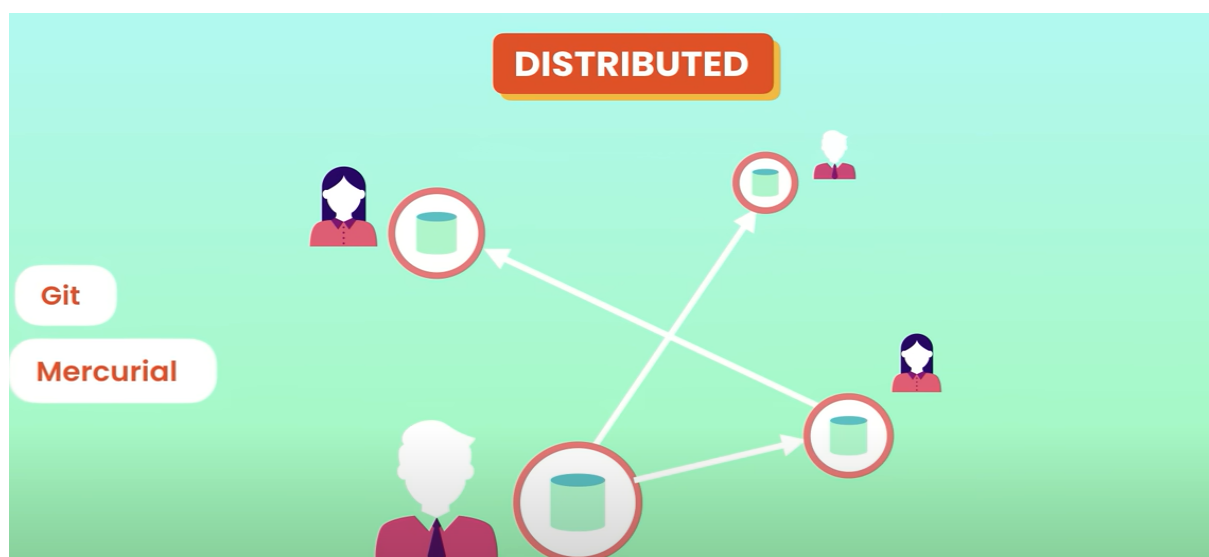


In a **centralized** system all the team members will be connected to a centralised server to get the latest copy of the code and to let everyone know about the changes in the code.



The problem with the centralized architecture is that if the server goes offline we can not merge our code or take a snapshot or get the latest project till the server comes back online.

In the distributed architecture every team member will be having a copy of the project with the history on their machine, so we do not have this problem. If the server is offline we can synchronize the changes directly with the teammates.



Git and Mercurial are the examples of the distributed version control system however the Git is the most popular version control system in the world because of the following reasons:

1. It is free to use.
2. It is open source.
3. It is super fast.
4. It is Scalable.

How to know whether the Git is installed on our machine ?

Open a command prompt and type in **git --version**. If it is not installed then we will get a error message like below

```
C:\>git --version
'git' is not recognized as an internal or external command,
operable program or batch file.

C:\>
```

Some of the terms that we should be familiar with while using the Git.

- Directory -> **Folder**
- Terminal or Command Line -> **Interface for Text Commands**
- CLI -> **Command Line Interface**
- cd -> **Change Directory**
- Code Editor -> **Word Processor for Writing Code**
- Repository -> **Project, or the folder/place where your project is kept**
- Github -> **A website to host your repositories online**
 - **Github** - Frequent confusion arises between Git and GitHub. To clarify, Git is a version control tool utilised to monitor code changes over time, effectively managing the development process. On the other hand, GitHub functions as a web-based platform where we can host and manage our Git repositories, providing a collaborative and centralized environment for teams to work together on projects.

Frequently utilized terminologies and Git commands include:

- Clone -> **Bring a repository that is hosted somewhere like Github into a folder on your local machine**
- add -> **Track your files and changes in Git**
- commit -> **Save your files in Git**
- push -> **Upload Git commits to a remote repo, like Github**
- pull -> **Download changes from remote repo to your local machine, the opposite of push**

1. **Clone** - This command is employed when a repository exists on GitHub but not on the local machine. It allows users to retrieve the repository and its contents to their local environment, enabling them to make changes and updates.

2. **Add** - When modifications are made to files, or new files are created or deleted, the "add" command informs Git to track these changes. It prepares the changes to be included in the next commit.

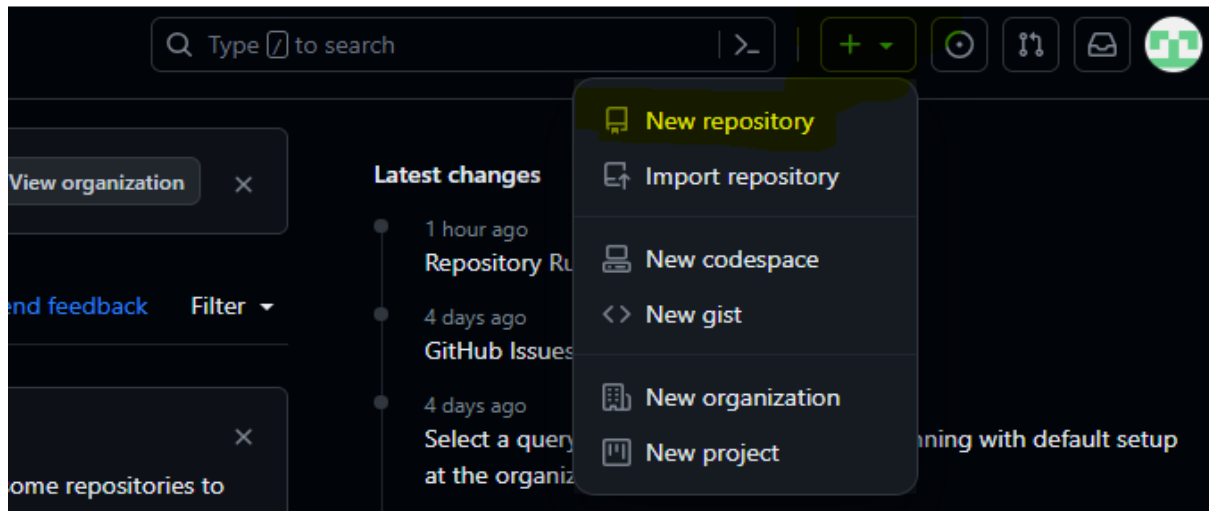
3. **Commit** - As mentioned earlier, Git's purpose is to record changes made to the codebase. The "commit" command is used to create a snapshot of the changes made since the last commit. This action effectively saves the changes to the local repository.

4. **Push** - Once changes have been committed locally and are ready to be uploaded to a remote location, such as GitHub, the "push" command is employed. It transmits the committed changes to the remote repository, making them available for others to access.

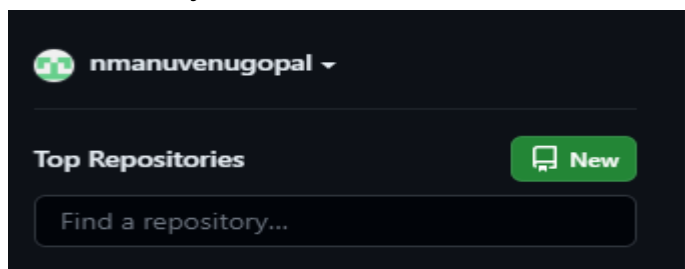
5. **Pull** - When changes have been made to the remote repository, for example, by a teammate, and these changes need to be brought into the local repository, the "pull" command is used. This action fetches and merges the changes from the remote repository into the local workspace.

The next thing we need to do is to create a repository. We can create a directory in Github using following ways:

First way is to use the “+” symbol located near to the user profile picture.



Second way is to use the “New” button on the top left corner.



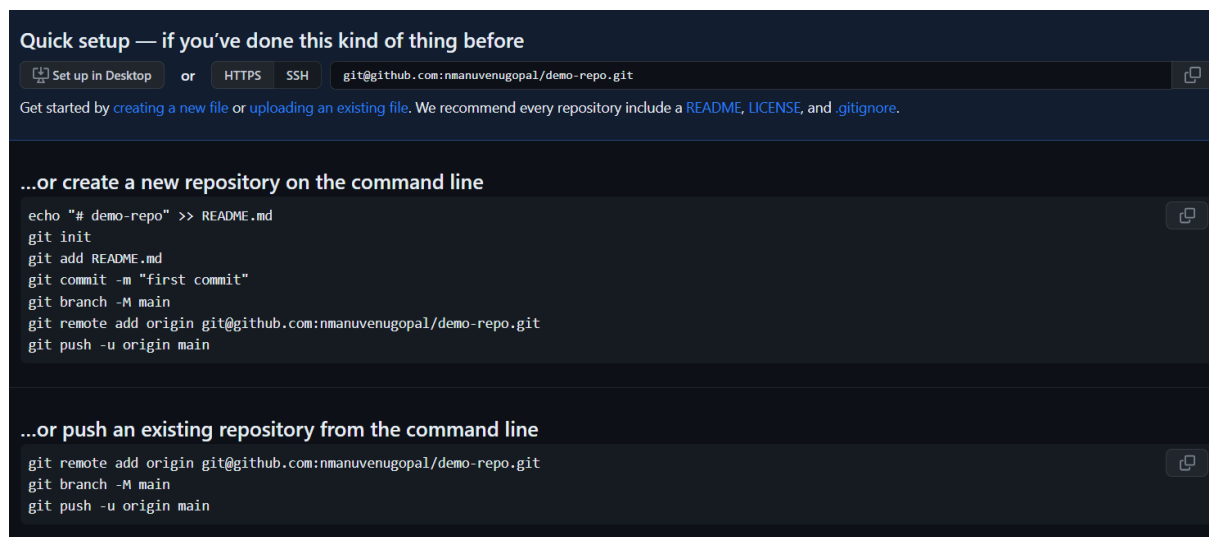
Click on the add button, fill in the details and click “Create repository” button.

A screenshot of the 'Create new repository' form in GitHub. The form has two main sections: 'Owner' and 'Repository name'. The 'Owner' is set to 'nmanuvenugopal'. The 'Repository name' is 'demo-repo', and a green checkmark indicates it is available. Below this, there is a suggestion: 'Great repository names are short and memorable. Need inspiration? How about silver-system?'. The 'Description' field is optional and contains 'Demo for the Glthub intro'. At the bottom, there are two radio buttons for visibility: 'Public' (selected) and 'Private'. The 'Public' option is described as 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option is described as 'You choose who can see and commit to this repository.'

Repository is our project, it contains all the files and subfolders that project utilises to work properly.

We can create the files and folders for the repository locally on our machine or we can create it through the online editors on the Github website.

After we click on the create repository button, I am going to create a basic readme markdown file. Readme.md is the most basic file that contains the basic information about the project, what it does or any other specific information or details.

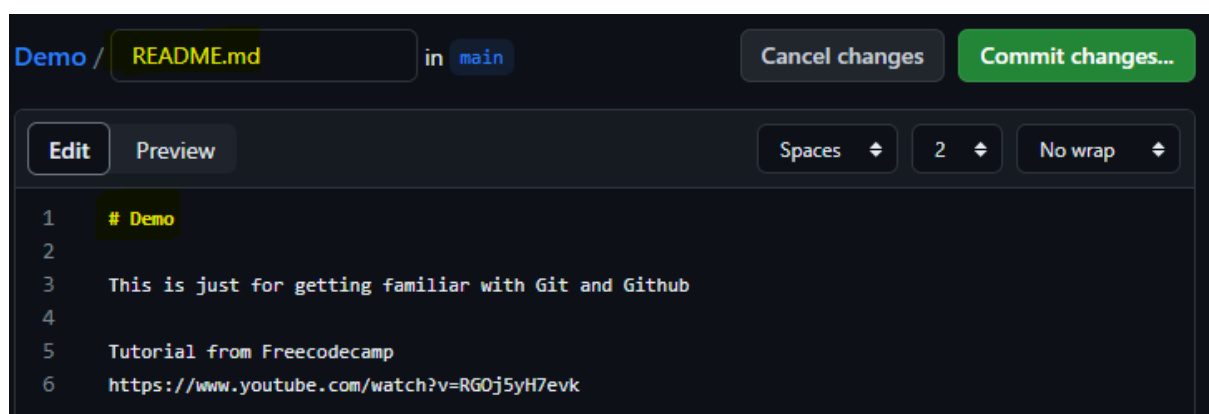


```
Quick setup — if you've done this kind of thing before
Set up in Desktop or HTTPS SSH git@github.com:nmanuvenugopal/demo-repo.git
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line
echo "# demo-repo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:nmanuvenugopal/demo-repo.git
git push -u origin main

...or push an existing repository from the command line
git remote add origin git@github.com:nmanuvenugopal/demo-repo.git
git branch -M main
git push -u origin main
```

I have clicked on creating a new file and I need to name it as README.md file. Markdown file have many shortcuts like “#” for the main header.



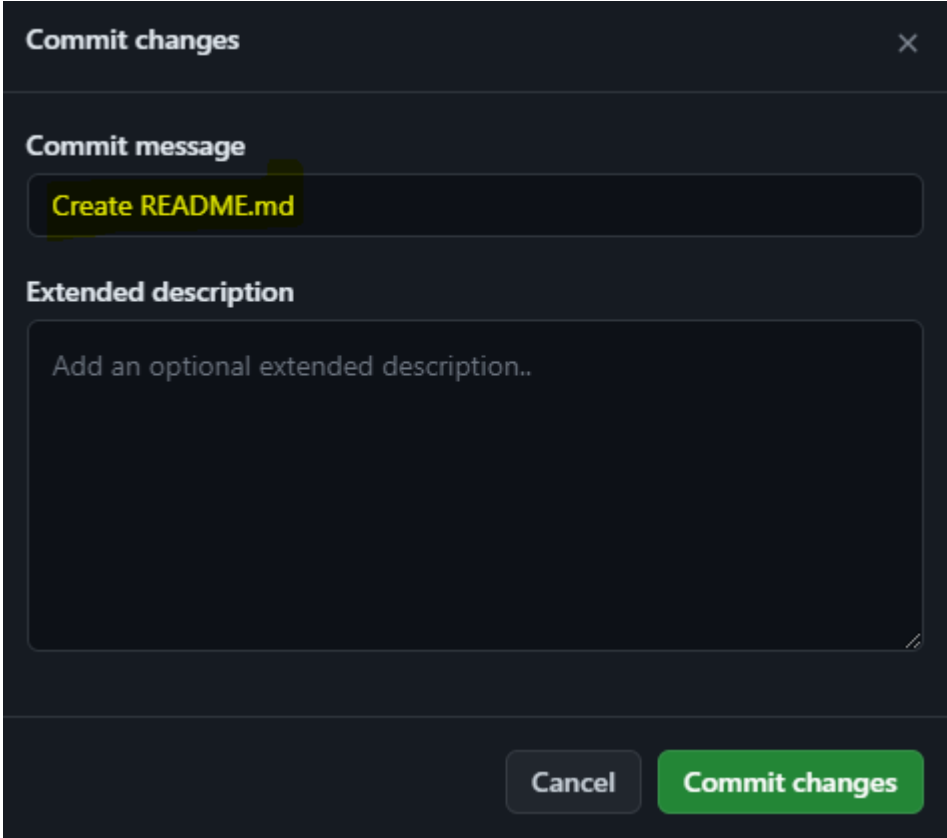
```
Demo / README.md in main
Cancel changes Commit changes...

Edit Preview Spaces 2 No wrap

1 # Demo
2
3 This is just for getting familiar with Git and Github
4
5 Tutorial from Freecodecamp
6 https://www.youtube.com/watch?v=RG0j5yH7evk
```

Once I enter the details about the project in the README.md file, the next thing will be committing the changes we have made. For that we need to click on the commit change button.

Upon clicking the commit changes button, a dialog box will appear:





A dark-themed dialog box titled "Commit changes" with a close button (X) in the top right corner. It contains two input fields: "Commit message" with the text "Create README.md" and "Extended description" with the placeholder text "Add an optional extended description..". At the bottom, there are two buttons: "Cancel" and "Commit changes".

We can add or update the details and click on the commit changes button for saving our changes.



A dark-themed view showing commit details. It has a table with three columns: "Name", "Last commit message", and "Last commit date". The first row shows a file icon, "README.md", "Create README.md", and "now". Below the table, there is a section for "README.md" with a pencil icon for editing. The content of the README.md file is displayed below, starting with a large heading "Demo", followed by a horizontal line, and then two paragraphs of text.

Name	Last commit message	Last commit date
 README.md	Create README.md	now

README.md 

Demo

This is just for getting familiar with Git and Github

Tutorial from Freecodecamp <https://www.youtube.com/watch?v=RGOj5yH7evk>

Reference

Git and Github for beginners - freeCodeCamp

<https://www.youtube.com/watch?v=RGQj5yH7evk>