https://www.padok.fr/en/blog/devops-process#:~:text=Most%20of%20the%20DevOps%20projects,push%20a%20new%20feature%20easily.

If you wished to turn your team into DevOps but you got confused, this article is for you.

Here I explain the DevOps, its principles, its process and

how to implement this methodology in your software development process.

Summary

What is DevOps? The DevOps principles The DevOps process Implementing DevOps in your team

What is DevOps?

If you are following our blog, you already know what the DevOps is.

If not here is a quick sum up and an article that you can read about the DevOps culture:

The DevOps methodology improves communication between your developers and Ops on projects. The objectives of DevOps are as follows:

save time on incident resolution

launch new features faster

reduce risks through process automation

significantly increase the satisfaction of your customers, and your developers at the same time.

To do so, we are talking about the feedback loop. It is a DevOps concept that relies on measuring

the impact of the development to know its effectiveness and continue to learn. This feedback loop is

the key to achieve DevOps objectives and successfully improve your software and your team’s process.

It will also help your teams delivering quality code.

feedback-loop-devops

Today, according to IDC, 20% of application development projects benefit from a DevOps approach.

By 2021, this figure will increase to between 35% and 40%. It is no coincidence that so many

IT teams are adopting the DevOps methodology for the project.

The DevOps principles

Using the DevOps methodology within your team for software development is going to

change radically your organization. It will change the way your developer team is coding and deploying releases.

It is based on 6 key principles:

Automation

Iteration

Self-service

Continuous improvement

Continuous testing

Collaboration

Automation allows developers and Ops to simplify the process and allows teams to be more productive.

Since automation reduces the number of manual actions, it improves the development quality and

security of the software. While iteration accelerates the development process thanks to faster

end-users feedback. The self-service also accelerates releases by enabling developers to deploy

applications on-demand by themselves. Continuous improvement is there to make the process more fluid.

Indeed, after each incident, it is part of the DevOps process to do a post mortem. A post mortem is

recording each incident, its impact, the actions taken to fix it, the cause of the problem and the

actions that have been taken to prevent it to happen again. Continuous testing allows faster releases

and higher quality at the same time. And finally but not least, the collaboration between developers

and Ops is key to combine efforts and reach success faster.

The DevOps process

The DevOps process consists of a series of steps. Here is a diagram to help you visualize:

devops-process

In the order, the DevOps process steps are:

Plan: This is the part of the project where you are organizing the tasks, schedules and set up your project

management tools. The idea is to plan tasks using the user story process from the agile methodology.

Write tickets in the form of a user story will allow developers and Ops to understand what development needs

to be done and why. A perfect user story as a what (who, where, trigger), a why and acceptance criteria.

(Example: As a user, in my customer account, when I click on "option" I open a popup and I can change my name and first name).

Code: Here the developers are doing code development and code review. When the code is ready, they merge it.

In DevOps practice, it is important to share a code tool between Ops and developers teams like Github or Gitlab.

Build: This step is the first one toward automation. The goal here is to build the source code into one desired format,

compiling, testing and deploying in a particular place of the infrastructure. Once this step sets up the continuous

integration (CI) and delivery (CD) tools can check and verify the source code from Source Code Management and build it.

Test: The continuous testing process reduces risks. Automatic tests ensure that no bugs will be implemented in production.

You have to implement testing tools in your workflow to ensure the best development quality for your software.

Release: The code has passed the testing (continuous integration) process and is ready to be deployed.

Deploy: The operational team is deploying the new feature in production. But as automation is one of the DevOps principles,

it is possible to set up continuous deployment.Operate/ configure infrastructure: The Ops build or maintain a scalable

infrastructure, infrastructure as code and check security issues and log management.

Monitor: The monitoring is an important step as it allows to fix incidents faster and to create a better experience for your end-user.

As DevOps aims to significantly increase the satisfaction of your customers, naturally your teams start again the

steps with a new feature for your software or application. That is why we always draw the DevOps as an endless loop.

Implementing DevOps in your team

If your team has not yet implemented a DevOps process or even an Agile one, it can seem confusing, especially because it

means shift not only the organization but also the culture of your team.

To introduce DevOps in your team you have to take your time, go step by step, don’t rush it. Here is how to gradually

implement this new culture and organization:

Start by changing the mindset of your teams. Without tools, the DevOps process can’t exist, but it is not enough, developers

and Ops must understand and be interested in the work of each other. In short, developers and Ops must learn to collaborate.

Implement an agile development process. Most of the DevOps projects use the Scrum or Kanban methodologies from Agile.

To be able to do the fourth step, you should adopt cloud computing (Private, Public or Hybrid Cloud).

Automate your software deployment thanks to continuous integration (CI) so developers can push a new feature easily.

Use software continuous testing to reduce risks.

Implement continuous deployment (CD), less manual actions there are, faster you can deploy new releases in production.

Keep in mind that adding those processes will change your organization and create new needs. For instance, your team may require

training for the new DevOps tools that have been implemented.

You will reach the true DevOps environment when your development process will be as automated and agile as possible.

1.\* What is Version Control System?

Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later

What is GIT?

GIT is the most popular and widely used distributed version control system (DVCS) and source code management (SCM) tool with lightning speed, data integrity and efficiency.

Git snapshots each and every commit you do, it becomes easy to track the version you saved, and in case you can revert to whichever version you need.

2.\* Why we need any Version Control System (v.C.S)

What is a Repository in GIT?

Git repository is nothing but a location where you can store all files related to your project, when you commit your code. Normally a repository contains a

directory named ".git" along with your project files, where git keeps all of its metadata of the repository. The content in the ".git" is private to git.

For ‘bare-repository’ ".git" directory will not be there but git’s metadata store in the same directory where you store your projects files.

What are the advantages of using GIT?

a) Git is Open source b) Better branching system b) High availability, very reliable c) Only one .git directory per repository d) Superior disk utilization

and network performance f) Light weight and Lighting speed e) Collaboration friendly and compatible with previous version control system f) Any sort of projects can use GIT.

What is the command to write a commit message?

Command to write a commit message is "git commit –m < commit message >". The "–m" on the command line instructs git to commit the new content of all files that have been modified or created. You can use "git add

" before git commit –m if new files need to be committed for the first time. To commit only modified files, use this command "git commit –am < commit message >"

What language is used in GIT?

Its ‘C’ language, GIT is fast as ‘C’ language makes this possible by reducing the overhead of runtimes associated with high level languages.

Difference between GIT and SVN are

GIT SVN

Git is a Distributed Version Control tool SVN is not a Distributed Version Control tool

Git Commits are unchangeable A tag can be treated as a branch and multiple versions can be committed

Clients can clone entire repositories Can't Clone Version history

on their local systems

Commits are possible even if offline Offline Commits are not possible

Push/pull operations are faster Push/pull operations are slower

Git is less preferred in case of extremely

large files or frequently changing binary SVN allows to multiple projects with in a same repository

files

Works are shared automatically by a commit Nothing is shared automatically

Why is GIT better than Subversion?

Git and SVN are each viable workflow and version control systems, Git will allow you to store versions of a project, and it has stage area where you can store your uncommitted changes and you can also do patch staging in case you and your co-devel

oper are working on same file then you can patch your work to the file by using ‘-p’ flag while git add. And Git has a better branching system and git stash too.

In GIT, What is “Index” or “Staging Area”?

Before committing your changes to git you can save changes to git using git add <file\_name>, it is reviewed as an intermediate area known as ‘Staging Area’ or ‘Index’.

What is the function of ‘GIT PUSH’ in GIT?

‘GIT PUSH’ publishes local repository changes into remote repositories so that others can take those changes.

What is GIT stash?

If you are in the middle of a job and you need to jump to another job, use the 'git stash' command. git stash records the current state of your working directory & index, and gives you a clean working directory which matches the HEAD. So you can jump to another job, by this you won't lose your changes/edits. And you can un-stash to get your previous changes back to the working directory.

To see what changes you stashed use 'git stash list' this will list you the changes and 'git stash pop' will get back your changes to the working directory

What is GIT stash drop?

When you don't need the stashed changes and want to remove it from the stashed list, run the git 'git stash drop' command. By default it will remove

the last added stash item, and to remove a specific item you include it as an argument 'git stash drop <stash\_name>'.

How will you know if the Git branch has merged to master or not?

This command 'git branch --merged master' will list the branches that have been merged into the master. Use the '-a' flag to show local and remote branches and '-r' to view only remote branches.

What is a Git Clone?

The git clone creates a copy of an existing remote repository into a local, command to clone a git repository is "git clone <remote\_repo\_url>".

What is the function of Git Config?

As we know git is a version control tool it keep track of all commits done by each and every user, so git should have user data like name and email, git never allow user to commit without this details, so user need to configure git using this command,

git config --global user.name "your\_name"

git config --global user.email "your\_email"

What does the commit object contain?

A 'commit' points to the specific version of the project looked like at a certain point of time.

A set of files/directories, representing the state of a project at a given point of time.

Reference to parent commit(s) objects

Time stamp.

With all these info git commit has a unique id known as SHA id or SHA value.

How to create a local repository in Git?

Create a directory for the project if it does not exist, and go into the directory then run the command ‘git init’. A directory by name .git will

be created in the project directory.

What is ‘HEAD’ in git and how many heads can be created in a repository?

HEAD is simply a reference to the last commit in the current working branch. In every repository, there will be a default head known as "Master". A repository can contain any number of heads. When you checkout to any branch the HEAD revision

changes to the latest commit of the new branch. We directly will not create head but when we commit, git creates a reference to that commit known as head.

What is the purpose of branching in GIT?

The purpose of branching in GIT is that you can create any no.of branches and jump between those branches. It will allow you to go to your previous work in one branch keeping your recent work intact in another branch. For example you can create a branch for main release and a branch for bug fix after

completion of job in one branch. You can merge the code to master by this. You can work parallely with multiple jobs and you will have a safe code in master.

What is the common branching pattern in GIT?

A common way to create a branch in GIT is to maintain one as "Main" branch and create another branch to develop new features.

Generally used branches are Feature branch, Release branch and Hotfix branch. This pattern is useful for multiple developers working on the same project.

4.\* Which VCS you prefer? SVN Or Git? Why?

5.\* What are the advantages of Git over SVN?

6. Why we call Git as Distributed VCS?

7. Can you explain Git's End-to-End work flow?

8. How do you clone the code using git?

Git clone your url

9.\* What is the difference between Commit & Push?

Well, basically git commit puts your changes into your local repo, while git push sends your changes to the remote location.

Local repository: repository on your computer.

Remote repository: repository on a server (Github).

10.\* What is the difference bet'n Push and Pull?

Push - pushing sends the recent commit history from your local repository up to GitHub. ... Pull - a pull grabs any changes from the GitHub repository and merges them into your local repository.

11. Can you explain Git architecture?

12.\* What is the diff. bet'n Centralized and Distributed VCS.

13. Have you ever created Remote repositories in Git? How?

14. What happens if I delete .git folder?

15. How do you configure username, email and editor first time

in Git?

16. Where Git stores configuration details?

17.\* What is the advantage of STAGE in Git?

18. Git log options related questions

--author

--grep

--oneline

--since/until

-n2

19. What is SHA-1? How Git uses this?

20.\* I have a file modified in my Working directory. How do you

show the content diff?

21.\* How do you show the content diff of a file which is staged?

22. How do you delete and rename a file in Git?

23.\*\* What is your branching stratogy?

Can you explain your release process/Stratogy?

24.\*\* What branching model you suggest for parellel development?

25. Developer fixes a bug. How do you take the change to

production?

25.\*\* Explain defferent branching models that you have worked-on.

26. Did you work on merging the code in Git?

27.\* How do you merge the code in Git?

28.\* What is merge? What is conflict?

29. When do we get conflict?

30.\* What is fast-forward merge in Git?

31.\* What is the difference between Merge and Rebase?

32.\* How do you resolve the conflit in Git?

34.\* What kind of conflicts you have seen?

35. Who resolves the conflicts?

36.\*\* What is the difference between branch and tag?

When do you create a branch and tag?

37. How do you create a branch and switch to that using single

command?

38. What is HEAD pointer in Git? Where Git store HEAD info.

39. Can we store binary files in Git?

40. Can skip the staging? How? what are the caveats?

41.\* How do you list files/folders modified as part of a commit?

42.\* How do you ignore: ex:

all files ending with .class

all files having alphanumeric

all log files but not build.log

43. How do you add ignore list for all users?

44.\* What are the different files you ignore in your project?

45. How to remove a committed change? Or can we remove?

$ git reset --hard HEAD~1

$ git reset --soft HEAD~1

46. How do you lock the branch

47. How do you clone the code from a particular SHA?

48. How do you restore a deleted file? Or previous changes of

a file?

49. How do you list the diff. of a file between two different

branches.

$ git diff dev\_1.2.4...master -- LoginUser.java

50. How do you list the changes which are going to be fetched?

method:1

$ git fetch

$ git log origin/master ^master

method:2

$ git fetch && git diff master origin/master --name-only

51. What is Git Stash?

52. How do you add a new remote to git? Or How do you attach

your local repo with remote?

53. What is git ls-tree?

git ls-tree <sha> --> Lists files committed as part of

a commit.

54.How do you clone the repository with a single/particular branch?

$ git clone -b dev\_1.2.4 --single-branch https://github.com/nageshvkn/flipkart899.git

55.

How to compare two branches?

$ git diff master..dev\_1234 [compare local branches]

$ git diff origin/master..origin/dev\_1234 [compare remote branches]

Qns:

1. How do you revert the code which is already committed in the repository?

2. User A has deleted the file in local repository. User B modified the same file and pushed to remote. Now, when user A push'es the file what will happen?

3. How do you make local repository as remote?

4. How do you push a new branch to remote repository?

5. How do you clone a single branch?

6. How do you search a commit based on time?

7. How do you clone a single folder / file? Or is it possible in Git?

7. How do you list the changes which are fetched?

$ git diff origin/master

8. How do you list the changes before pull/fetch?

$ git checkout master

$ git fetch

$ git diff origin/master

$ git remote rm origin

$ git remote add origin https://github.com/nageshvkn/jinglegurus.git

9. Push a particular commit to remote repository

$ git push origin 7d662c54a4e0367c:master