Goals

In modern software development age, it is almost impossible to develop a software without the help of a source version control. Especially in term of large-scale development. The invention of source version control is a remarkable milestone to make software development no long a lone programmer’s job but instead a team effort. A clear understanding and the capabilty of using a source version control is an essential requirement for any developers. There are a lot of source version control systems available in the market. However, GitHub manages to hold it top spot above its competitors for a significant period of time. This proves that GitHub always delivers the best utilities, efficiency for which it is trusted from many prestiguos software companies worldwide. This learning package aim is to provide the organization developers a deep understanding about GitHub. It also introduces the skills needed to apply GitHub from fundamental to advance level for the company actual project.

Target Audience

This learning package is going to be made available to all employees in the company and compulsory to be studied by all developers, especially for new ones who have not had any prior experience using GitHub before. Nonetheless, other skilled developers are encouraged to follow through the package to strenghthen their knowledge about source version control. Staff members apart from developers are also welcomed since this is a good opportunity to embrace an important concept of software engineering.

Learning Plan

The organization expects every developer to be familiar with GitHub before actually starting the project to avoid any confusions later on which may affect the team productivity. There are going to be 4 separate sessions needed to upskill all the developers which scheduled as following:

* Session 1: Introduce the concept of source version control.

1. Objective: make sure that the audience embrace the concept of using a source version control system in software development.
2. Content: explain the definition of source version control, also talking about the background history. Differentiate the concept of Centralized Version Control System and Decentralized Version Control System. State the benefits of having a source version control system and the overall information of some popoular source version control systems recently.
3. Duration: 30 minutes.
4. Time: 2 weeks before actual development.

* Session 2: Introduce Git and GitHub.

1. Objective: convince all the employees that GitHub is the most suitable choice for the firm project. Remove any possibly doubts about GitHub usage.
2. Content: provide background history, current state and achievement of GitHub. List out the benefits and the advance features that set GitHub above other similar source version control system.
3. Duration: 30 minutes.
4. Time: 3 days after the first session.

* Session 3: Basic feature/functionalities of GitHub.

1. Objective: ensure that all attendee can confidently use GitHub for their project after the session.
2. Content: teaching GitHub basic feature/functionalities. Provide demos in parallel with learning activities for each of them. Every attendee is expected to bring their own laptop to do the activity.
3. Duration: 1 hour 30 minutes.
4. Time: 3 days after the second session.

* Session 4: Advance feature/functionalities of GitHub.

1. Objective: this session is made for one who wants to master GitHub with its advance features. Specifically targets the team managers.
2. Content: teaching GitHub advance feature/functionalities. Provide demos in parallel with learning activities for each of them. Every attendee is expected to bring their own laptop to do the activity.
3. Duration: 1 hour 30 minutes
4. Time: 3 days after the third session

These sessions should be lectured by a version control expert. The company can invite external guess or appoint one of the experienced software developers who is nicely capable of transmitting ideas to become the instructor. Printed learning materials will be provided to all attendants after the session ended. Employees are reminded to revise their contents before attending the next session. The total period of time for employees to complete this learning guide is around 2 weeks.

Learning resources

The session slides are encouraged to extract and display information from the following resources. They are all empirical studies and published from credible source. The contents of these resources are closely related to what the organization wants the attendee to embrace for each session. However, since the amount of information is quite huge while the session time is limited, instructors are advised to carefully go through all of them and pick up appropriate pieces of information. Any less critical/extra knowledge should be handed out in form of printed learning materials.

* Session 1: this session focus on purely providing information theory without having any actual practices to the audiences. Some information extracted from famous conference proceedings is sufficient enough for this session.
* Ren, Y., Xing, T., Quan, Q., & Zhao, Y. (2010). Software Configuration Management of Version Control Study Based on Baseline. *2010 3rd International Conference on Information Management, Innovation Management and Industrial Engineering* (pp. 118 - 121). Kunming: IEEE.
* Alwis, B. d., & Sillito, J. (2009). Why are software projects moving from centralized to decentralized version control systems? *Cooperative and Human Aspects on Software Engineering, 2009. CHASE '09. ICSE Workshop on* (pp. 36 - 39). Vancouver: IEEE.
* Brindescu, C., Codoban, M., Shmarkatiuk, S., & Dig, D. (2014). How do centralized and distributed version control systems impact software changes? *ICSE 2014 Proceedings of the 36th International Conference on Software Engineering* (pp. 322-333 ). Hyderabad: ACM.
* Session 2: the second session supplies information about GitHub without any actual activities. These learning resources below are conducted from trusted studies about GitHub would be helpful for this session.
* Kalliamvakou, E., Damian, D., Blincoe, K., Singer, L., & German, D. M. (2015). Open Source-Style Collaborative Development Practices in Commercial Projects Using GitHub. *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering (Volume:1 )* (pp. 574 - 585). Florence: IEEE.
* Vasilescu, ., Yu, Y., Wang, H., Devanbu, P., & Filkov, . (2015). Quality and productivity outcomes relating to continuous integration in GitHub. *ESEC/FSE 2015 Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering* (pp. 805-816). New York: ACM New York, NY, USA ©2015.
* Yu, Y., Yin, G., Wang, H., & Wang, T. (2014). Exploring the patterns of social behavior in GitHub. *CrowdSoft 2014 Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies* (pp. 31-36 ). Hong Kong: ACM.
* Session: 3 & Session 4: these last 2 sessions is more practical than previous ones. They focus on demonstrating GitHub features and creating learning activities to help attendants become more familiar and comfortable using GitHub to develop softwares. The organization instructor should relate to the following this suggested resources list about GitHub guides, online tutorials, etc.
* Git Real: this is an online Git course operated by Code School which includes seven levels and dozen individual exercises in a fun game format: Pollack, G., Lacan, O., McGavern, J., Mezzell, J., Denney, D., Rensel, A., . . . Films, L. (2016, May 14). *Git Real*. Retrieved from Code School: https://www.codeschool.com/courses/git-real
* ProGit: this book contains deeper knowledge about Git and GitHub. This is a suitable choice for Session 4 content: Chacon, S., & Straub , B. (2014). *ProGit 2nd Edition.* New York: Apress.
* Git - The Simple Guide: this single-page site gently offers the basics of Git: Dudler , R. (2016, May 14). *git - the simple guide*. Retrieved from rogerdudler.github: http://rogerdudler.github.io/git-guide/
* Git Workflows: exhaustively describes workflow used at GitHub: Chacon, S. (2011, August 31). *GitHub Flow*. Retrieved from scottchacon: http://scottchacon.com/2011/08/31/github-flow.html

Learning activity

This is an example of a learning activities that the session instructor should hold in Session 3 and 4.

* Content: create and clone a new Git repository using Git Bash.
* Prerequisite: audiences need to have their accounts already registered on GitHub.
* Outcomes: learners should know how create their own respository or create/clone someone else Git repository.

**Basic guide to create and clone a repository on GitHub**

**1) Create a new repository**

Firstly, learners have to go to https://github.com/ and sign in their registered accounts. In order to create a new repository, user can click the green button "New Repository" in the right corner of the web page. (Figure 1)

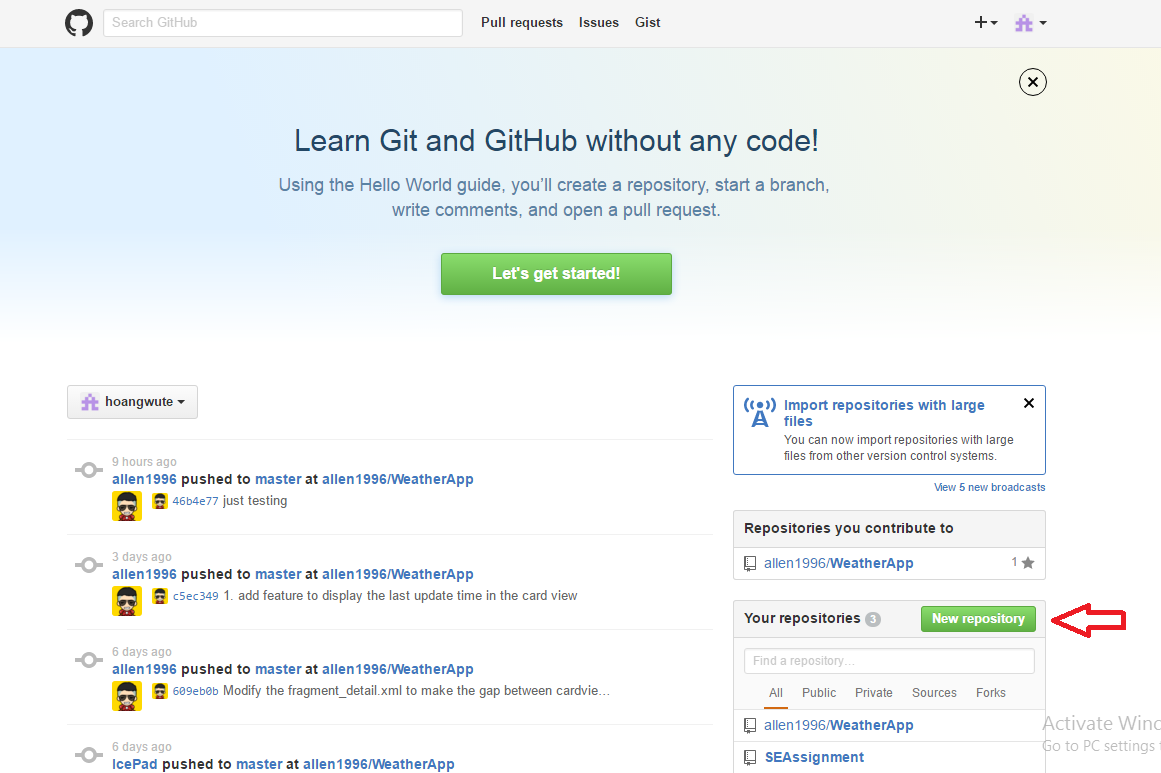


Figure 1

The new repository is named basicTutorial and the "**Initialize this repository with a README"** option is checked. (Figure 2). Then click on the green button "Create repository" to finish the process. (Figure 2)

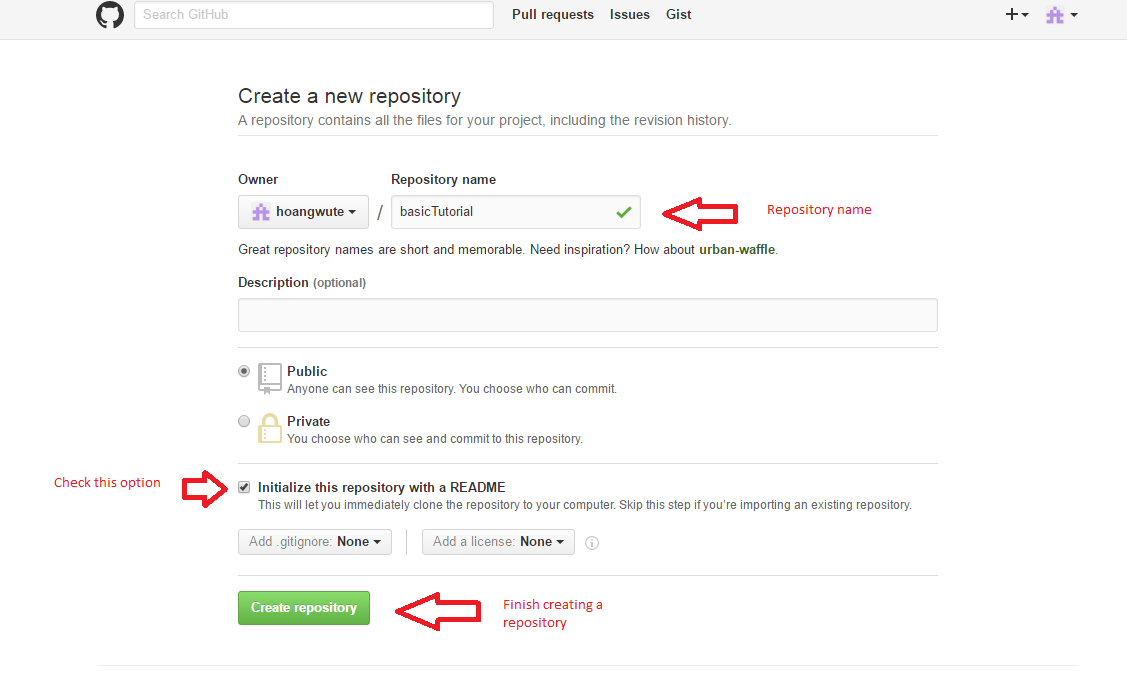


Figure 2

This is the user newly created repository page. It only has 1 file which is README.md. This file is supposed to contain instructions to whom uses the repository and it will automatically display its contents when someone access the repository. Up to now, we have known how to create a new repository on GitHub webiste. It is quite simple. Let’s move to the next section.

**2) Clone a repository**

The clone URL is already highlighted and has to be copied in order to clone the repository. It can be easily done by clicking the button right next to it. (Figure 3). The same concept happens if users wants to clone someone else repository, just go to their repository and copy the clone URL.

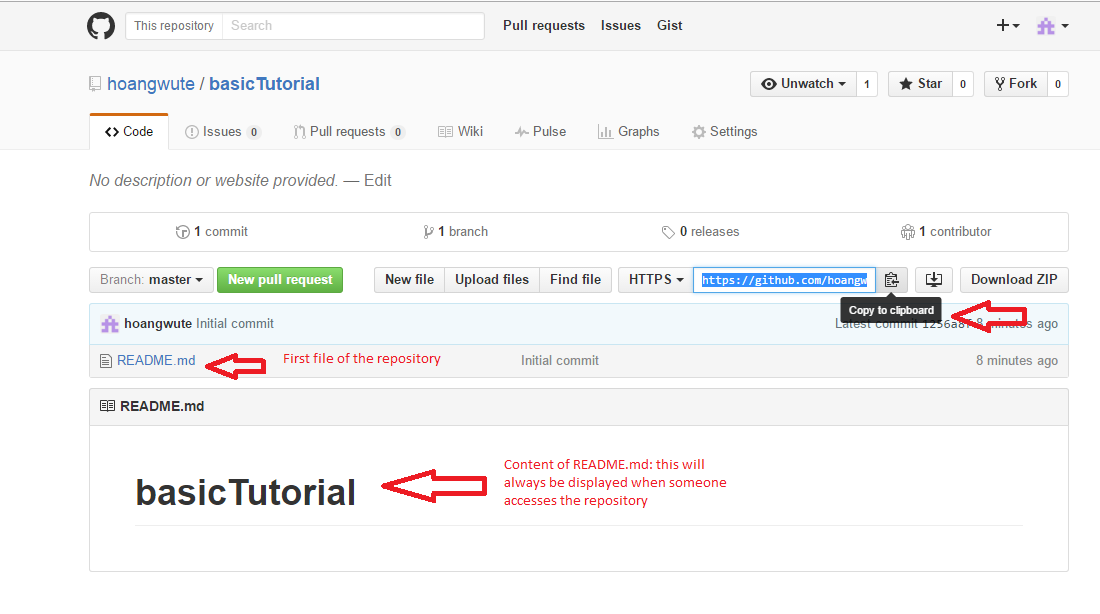


Figure 3

Next step, go the computer command line interface. For Windows users, open the Command Prompt, for Mac users it is the Terminal. In this tutorial, we use the Command Prompt of Windows. Here are some basic commands user should be familiar with to use the command line interface:

* cd (Windows/Unix) + directory path: to move to a particular directory.
* cd.. (Windows/Unix): goes back one directory.
* dir (Windows), ls (Unix): shows all the files in the current directory
* mkdir (Windows/Unix) + directory name: creates a directory with a specified name.

Now user should choose a directory to clone the Git repository. In our example, it will be stored in D:\GitExample. First, we have to create and move to the new directory (Figure 4).

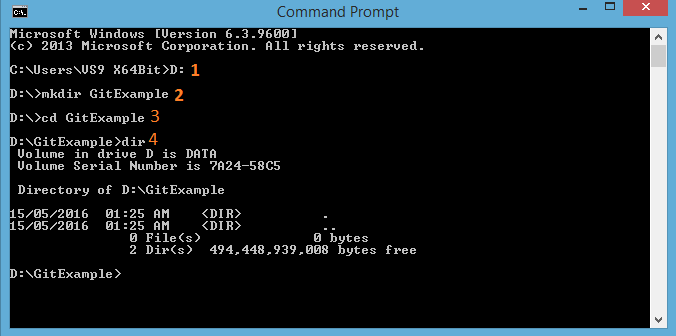


Figure 4

Command explanations:

1. This command changes the root directory from C: to D: - where we want to clone our repository.
2. This command create a new directory called "GitExample".
3. We move to the "GitExample" directory we have just created.
4. The "dir" lists all the files in "GitExample" directory. There is no file yet.

Now user should be in the directory which he/she wants to clone the repository in. To do it, use the command "git clone " + the copied clone URL. (Figure 5)

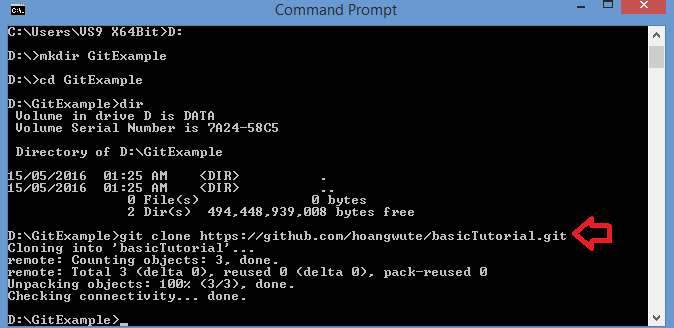


Figure 5

To check if the the repository is already cloned or not. User can use "dir" command. Observably, we have successfully cloned it since the new repository called "basicTutorial" is already included in the "GitExample" directory. (Figure 6)

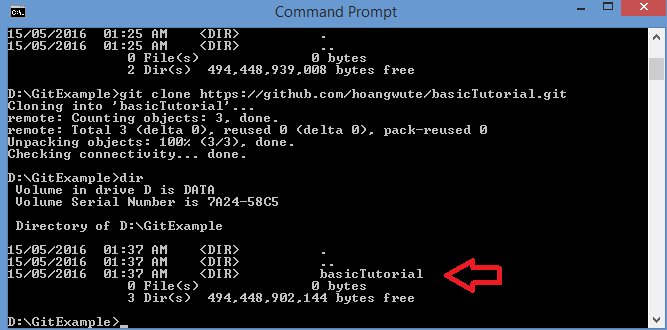


Figure 6

User can move to the cloned repository and start working on it (Figure 7).

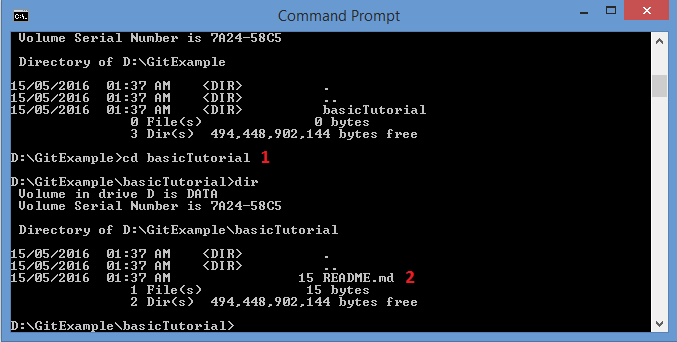


Figure 7

1. Move to the cloned repository.
2. "dir" command to check what files are inside the repository. Only the README.md file is there as it is supposed to be.

Here is what the new repository looks like on the personal computer (Figure 8).

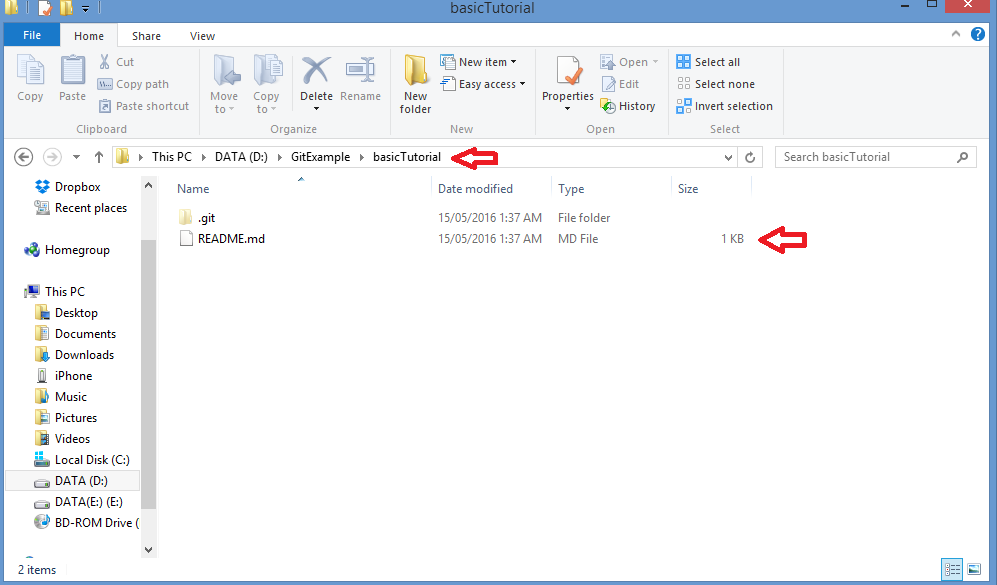


Figure 8

We have just gone through the process of creating and cloning a repository on GitHub using the command line interface. The commands which help users to interact with the cloned repository will be introduced in later tutorials.