TECHNO LOGS

01 - GITHUB

DÉROULEMENT

- Part 1 : Cours
  - Problématique
  - Git & GitHub
  - Les commande utiles
  - Les outils

- Part 2 : Pratique
  - Installation du poste de travail
  - Cas

LA PROBLÉMATIQUE

LES ENJEUX DU DÉVELOPPEMENT WEB

- Le travail en équipe
- L'historisation du travail
- La diffusion du code

UNE SOLUTION : LE VERSIONNING

- "Si le code n'est pas enregistré dans un logiciel de gestion de version, il n'existe pas. »
  - Jeff Atwood

- La notion de **commit**
  - Etape de modification
  - L'ensemble de commit == l'histoire de votre code

DIFFÉRENTES SOLUTIONS

- SVN / CVS / GIT / Bazaar …

- Diffèrent modèle
  - Centralisé => un serveur qui contrôle le tout
  - Distribué => toutes les machines ont la totalité du code

- Avantage du distribué
  - Eviter de tout perdre
  - Offline

GIT & GITHUB

GIT

- Création en 2005 par Linus Torvalds
- Modèle distribué
- https://git-scm.com/
- Mise en place d'un serveur distant
- *git remote*
  - Sauvegarde sur un service distant

GITHUB

- De nombreux services existant … comme BitBucket
- Hébergement -> numéro 1 mondial !
- Racheté par Microsoft
- *Tour du propriétaire*

WORKFLOWS

COMMANDES

GIT --INSTALL SERVER

- install git-core

- git init –bare
  - **git init** est la commande d'initialisation d'un dépôt GIT qui est utilisée en local (sur un client) et qui ne doit pas être utilisée seule pour initialiser un dépôt sur le serveur ;
  - **git init –bare** permet d'initialiser un dépôt GIT sur le serveur : cela créé un ensemble de fichiers (*config*, *description*, *HEAD*) et de dossiers (*branches*, *looks*, *info*, *objects*, *refs*) utiles à GIT ;

GIT --COMMIT

- git log
- git checkout SHADuCommit
- git revert SHADuCommit
- git commit --amend -m "Votre nouveau message »
- git reset --hard

GIT --PLAY

- git clone
- git stash
- git status
  - Vérifier l'etat de son espace de travail.
- git fetch
  - Verifier l'etat du groupe de travail.

GIT --BRANCH

- git checkout -b brancheA
- git merge brancheB
- git pull
- git push
- git rebase

GIT -- MANAGEMENT

- .gitignore
  - Fichier de configuration pour ignorier des fichiers

- git rm --cached “un fichier”
- git rm --cached -r “un repertoire”

- git branch --merged master | grep -v "* master" | xargs -n 1 git branch -d
  - Nettoyer toutes les branches ayant déjà été mergé avec master > Nettoyage de votre poste

GIT --CONFLIT

- git blame nomdufichier.extension
- git show 05b1233

- CONFLICT (content): Merge conflict in hello.md
  Automatic merge failed; fix conflicts and then commit the result.
- git diff

LES OUTILS

UTILISATION DE GIT VIA DES OUTILS

- Tour propriétaire WebStorm
- Check GITLAB

LIGNE DE COMMANDE == LA VIE !

INSTALLATION DU POSTE

GIT -INSTALL

- Download
  - Mac / Linux : https://git-scm.com/downloads
  - Windows : https://gitforwindows.org/

- Default Config :
  - git config --global user.name "Votre nom ou pseudo"
  - git config --global user.email "votre@email.com"

CLEF SSH

- https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/
- https://help.github.com/articles/adding-a-new-ssh-key-to-your-github-account/
- Eviter le HTTPS ! https://help.github.com/articles/adding-a-new-ssh-key-to-your-github-account/

RÉCUPÉRATION D'UN REPO

- git clone <url>
- git checkout –b <banch_name>
- touch myFile.txt
- vi myFile.txt
- git add myFile.txt
- git commit –a –m 'My commit'
- git push origin <branch_name>