

Workshop GIT

Par Kristina Matrosova

Présentation des outils

Installation

Configuration

Nouveau projet

GitHub Desktop

Projet existant en local

Introduction au travail collaboratif

PLAN

Principales commandes terminal (Windows)

- → cd nom_de_dossier accéder à un dossier
- → cd .. revenir un dossier en arrière
- → cd revenir à la racine
- → dir afficher les fichiers du dossier actuel
- → dir nom_de_dossier afficher les fichiers d'un dossier
- → md nom_de_dossier créer un dossier
- → cls nettoyer le terminal

Principales commandes terminal (Mac / UNIX)

- → cd nom_de_dossier accéder à un dossier
- → cd .. revenir un dossier en arrière
- → cd revenir à la racine
- → Is afficher les fichiers du dossier actuel
- → Is nom_de_dossier afficher les fichiers d'un dossier
- → mkdir nom_de_dossier créer un dossier
- → clear nettoyer le terminal

Principales commandes Git

- → git clone copier un repository GitHub en local
- → git init initialiser un dossier avec les fichiers de configuration git
- → git add fichier.ext ajouter un fichier
- → git add -A ajouter tous les fichiers
- → git commit -m "message" soumettre les modifications
- → git branch M branchname sélectionner la branche pour appliquer les modification (main = branche principale)
- → git remote add origin https://github.com/username/projectname.git rattacher le dossier local avec le repository git
- → git push appliquer les modifications
- → git push -u origin main appliquer les modifications à la branche d'origine
- → git pull récupérer la dernière version du projet
- → git status afficher la liste des fichiers modifiés

PRÉSENTATION DES OUTILS



Qu'est-ce que Git?

Git est un **logiciel de versionning** qui permet de **stocker un ensemble de fichiers** en conservant la **chronologie de toutes les modifications** qui ont été effectuées dessus. Il fait partie de la famille des VCS dit décentralisés car dans son fonctionnement chaque développeur va avoir en local une copie complète de l'historique de son code source (dépôt ou repository en anglais). Pourquoi utiliser un logiciel de versioning?

- 1. **Conserver un historique des modifications** effectuées sur un projet afin de pouvoir rapidement identifier les changements effectués et de revenir à une ancienne version en cas de problème.
- 2. Faciliter la **gestion de projet en équipe** de manière à ce que :
 - a. Les fonctionnalités sur lesquelles travaille un membre du projet ne rentrent pas en conflit avec les fonctionnalités sur lesquelles travaillent les autres
 - b. Chacun sache sur quoi travaillent les autres membres du projet afin de travailler correctement
 - c. Chacun possède une version actualisée du projet pour tester et implémenter ses fonctionnalités

Dans ce premier workshop nous nous intéresserons uniquement à l'utilisation individuelle de Git. Une deuxième session pourra être organisée sur demande pour étudier les spécificités du travail collaboratif.

Qu'est-ce que Github?

Github est un service en ligne qui permet entre autres d'héberger des dépôts Git. Il permet notamment de :

- Partager du code source avec d'autres développeurs et chercheurs.
- Signaler et gérer les problèmes ou bugs de votre code source via les issues.
- Proposer des évolutions pour un projet open source.

D'autres plateformes de ce type existent (GitLab, Bitbucket etc.), mais GitHub reste la plus utilisée à ce jour, et est notamment disponible pour la plupart des OS (Windows, Mac, Linux) et est compatible avec de nombreux IDE. Nous avons donc choisi de travailler avec Github dans le cadre de ce workshop.

GitLab	=	Q Search (GitLab		1]			• •	D n ~ G @•~	🏵 ~
,	Kristina Mat	rosova			Ç	Kris @kma (§ 12:	t ina Matrosov trosova · User ID: 60 21 AM	/a ୦4 _{ପ୍ରି} · Member sinco	New project/repository Create ne New group New snippet	Ø.	
			Overview	Activity	Groups	Contributed projects	Personal projects	Starred projects S	nippets Followers 0 Followin	ng o	
			Ma M W	y Jun	Jul	Aug Sep	Oct Nov	Dec Jan	Feb Mar Apr N	Лау	
			S		Ģ	Q Search or jump to		Pull requests issues	Codespaces Marketplace Explore Repositories 17 E Projects 😚	Packages 🏠 Stars	A New repository Import repository New codespace New gist
								Jeu-de-Dames Jeu de dames en console (algo min-max) © C	(Public) avec possibilité de jouer contre l'ordinateur	CreationMusicale	New organization New project
						Kristina Matro kmatrosova	sova	textgenrnn Forked from minimaxir/tex Easily train your own text- complexity on any text da Python	(Public) tgenrnn generating neural network of any size and taset with a few lines of code.	EID University labs/projects Jupyter Notebook	Public
						Achievements	rotile	iqdbms Python	Public	test_Records Jupyter Notebook 	Public)
						Beta Send feedback		11 contributions in the	last year		
					ttps://github.	com/new		May Jun Mon Wed	Jul Aug Sep Oct	Nov Dec Jan	Feb Mar Apr

Q + - 💿 -

L'instance GitLab d'HumaNum

- Demandez l'ouverture d'un compte HumanID : https://humanid.huma-num.fr/
- Vous pouvez ensuite demander l'accès aux différents services d'Huma-Num

La création du compte est très rapide si vous possédez un email institutionnel de l'ESR



isidore N ShareDocs Votre assistant de Plateforme de stockage Partager, publier et et de partage de fichiers valoriser vos données recherche en Sciences Humaines et Sociales (Web et clients WebDAV) scientifiques GitLab _Stylo_ () Mattermost Un éditeur de texte Plateforme de forge Service de discussion simplifiant la rédaction et basé sur git d'équipes l'édition d'articles scientifiques en SHS

Gérer mes services Huma-Num

INSTALLATION



A DESCRIPTION OF THE OWNER OWNER

Prérequis

Avant de commencer le tutoriel, assurez-vous d'avoir créé un compte sur le site <u>https://github.com/</u>.

Installation de Git sous Windows :

- 1. Téléchargez la dernière version sur <u>https://gitforwindows.org/</u>.
- 2. Lancez l'installation et suivez les instructions.
- 3. Ouvrez un terminal, entrez la commande git version pour vérifier que l'installation s'est bien passée.

Installation de Git sous MacOS :

- 1. La plupart du temps, Git est déjà installé sur MacOS. Ouvrez un terminal et entrez la commande git version afin de vérifier si c'est le cas.
- 2. Si ce n'est pas le cas, téléchargez la dernière version de Git en suivant <u>ce lien</u>.
- 3. Lancez l'installation et suivez les instructions.
- 4. Dans le terminal, entrez la commande git version pour vérifier que l'installation s'est bien passée.



Installation de Git sous Linux :

- 1. Sous Debian/Ubuntu :
 - a. Dans le terminal, entrez la commande sudo apt-get update afin de s'assurer que l'installateur apt est à jour.
 - b. Installez Git avec la commande sudo apt-get install git-all.
- 2. Sous Fedora :
 - a. Installez Git en entrant la commande sudo dnf install git-all dans le terminal.
- 3. Entrez la commande git version pour vérifier que l'installation s'est bien passée.

CONFIGURATION



Une fois que Git est installé, il va falloir faire quelques configurations.

Tout d'abord, nous allons **configurer le username et l'email**. Dans le terminal, entrez les commandes suivantes (en remplaçant les variables entre guillemets) :

- → git config --global user.name "mon username"
- → git config --global user.email "mon email"

Vous pouvez ensuite taper git config --global --list pour vérifier que la configuration a bien été prise en compte.

(base) kmatrosova@FRLPMC2823 ~ % git config --global --list user.name=kmatrosova user.email=tina.matrossova@gmail.com Nous allons utiliser Git depuis le terminal. Par défaut, à chaque interaction entre votre machine locale et GitHub, il vous sera demandé de vous authentifier avec vos identifiants. Depuis 2021, GitHub a remplacé (pour les opérations en terminal uniquement) le mot de passe par un **Personal access token** : c'est une sorte de mot de passe temporaire pour l'authentification durant l'utilisation en ligne de commande. Il permet d'avoir une sécurité plus élevée que celle du mot de passe, car il est généré aléatoirement et renouvelé régulièrement.

Néanmoins, cela peut être pénible d'entrer ses identifiants plusieurs fois dans la journée. Pour éviter cela, il est possible de configurer une paire de clés **SSH**. Ce moyen permet de créer une connexion durable et sécurisée entre votre machine locale et votre compte GitHub.

Les clés SSH servent à authentifier les utilisateurs sans utiliser de noms d'utilisateur ni de mots de passe. Au lieu de cela, les utilisateurs disposent d'une paire de clés SSH de confiance qui les authentifie comme la personne qu'ils disent être.

La **clé publique** permet de chiffrer un message ou un document, tandis que la **clé privée** permet de le déchiffrer. Ceci garantit que seul le destinataire choisi peut déchiffrer et lire le contenu.

Création d'une clé SSH

- 1. Vérifiez si vous avez déjà configuré une clé SSH. Pour cela, allez dans le dossier ~/.ssh/ avec la commande cd ~/.ssh/, puis vérifiez si les fichiers ~/.ssh/id_rsa and ~/.ssh/id_rsa.pub existent en entrant la commande ls .
- 2. Si ce n'est pas le cas, générez une paire de clés avec la commande suivante (en remplaçant votre email) : ssh-keygen -t rsa -C "mon email".
- 3. On vous demandera d'entrer l'emplacement où vous voulez stocker votre clé. Vous pouvez entrer le chemin vers le dossier de votre choix, ou cliquer 🖉 pour sauvegarder la clé à la racine.
- 4. Ensuite on vous demandera d'entrer une passphrase. Celle-ci représente une deuxième couche de sécurité : si une autre personne à accès à votre ordinateur, elle ne pourra pas utiliser la connexion par ssh sans connaître la passphrase. Cette étape n'est pas obligatoire, mais fortement recommandée.

(base) kmatrosova@FRLPMC2823 ~ % ssh-keygen -t rsa -C tina.matrossova@gmail.com Generating public/private rsa key pair. Enter file in which to save the key (/Users/kmatrosova/.ssh/id_rsa): Created directory '/Users/kmatrosova/.ssh'. Enter passphrase (empty for no passphrase): Enter same passphrase again: [Your identification has been saved in /Users/kmatrosova/.ssh/id_rsa Your public key has been saved in /Users/kmatrosova/.ssh/id_rsa.pub The key fingerprint is: SHA256:s5M+KX0TPFxPq0yIBA1UwNQzyl/wdxHEZrRBM4CqKQA tina.matrossova@gmail.com The key's randomart image is: +---[RSA 3072]----+ +=*. ..*0. 0 * . ___ . 0 * 0.. 0 0 0 0 0 =S= + + .. 0 0+* . 0 . .+. = . ..+.0 0 ..0 . ----[SHA256]----+

Afficher votre clé publique avec la commande cat ~/.ssh/id_rsa.pub .

[(base) kmatrosova@FRLPMC2823 ~ % cat ~/.ssh/id_rsa.pub] ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCy7j/FjWImzOkEfD02LFEgjVGpkHqS+bFj+Lui&nTMJPhEnVQojyxkkhHg mkD1eEgy9vAskAnsR8LFYUC3jQQtbOrlZuZ9KVT1PkRFUDjTJbRiT1ymN8XrFw2Cg8X10H57iSG4IQ&24Zo2&qARam2vX3fW tYFVUZOLHCHyVCjhAF2pXxVvZtZnALczXhAMbJwW7sX051sfoLlgd/Byp&zlUXugfYee&ysIjG10+B0zUvRzvnIJAVK/jtmD DiVXFMCBVYhKhmpI7RzgM0Q2ZwT/uL1&Z29DCqSsnoLLqGigUOGvPmAAgqkrjheqpE+S4ZtzXUqeAHdWN6XDKkBnNp1WP8hN RNE0gKHPDA3BMNs47heG0g1Zqtif9HUcxA51iYNjF10UEp4eGOnQSgy6dhWY4McgLCE30JNXhNDV2j5txX8h1XqN7G14mKHU WH0nMDNGjoexS06yAgcj10v46qfBzmgHq58IepeiaI5C6/3fTueMcliW5UjXwNwmAqQOgoM= tina.matrossova@gmail.c om

Connectez-vous à GitHub, puis allez dans Settings \rightarrow SSH and GPG Keys \rightarrow New SSH key.



Donnez un nom à votre clé ("Machine perso" par exemple) et copiez-collez votre clé publique dans le champ dédié.

SSH keys / Add new	
Title	
machine perso	
Key type Authentication Key	
Кеу	
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCy7i/FiWImzOkEfD02LFEgiVGpkHqS+bFj+Lui8nTMJPhEnVQojyxkkhHgm	

AAAAB3NzaC1yc2EAAAADAQABAAABgQCy7i/FiWImzOkEfD02LFEgiVGpkHqS+bFj+Lui8nTMJPhEnVQojyxkkhHgm kD1eEgy9vAskAnsR8LFYUC3jQQtbOrlZuZ9KVTIPkRFUDjTJbRiTIymN8XrFw2Cg8X10H57iSG4IQ824Zo28qARam2 vX3fWtYFVUZ0LHCHyVCjhAF2pXxVvZtZnALczXhAMbJwW7sX051sfoLlgd/Byp8zIUXugfYee8ysIiGl0+B0zUvRzvnI JAVK/itmDDiVXFMCBVYhKhmpI7RzgM0Q2ZwT/uL18Z29DCqSsnoLLqGigU0GvPmAAgqkriheqpE+S4ZtzXUqeAHd WN6XDKkBnNp1WP8hNRNE0gKHPDA3BMNs47heG0glZqtif9HUcxA51iYNjFloUEp4eG0nQSgy6dhWY4McgLCE30 JNXhNDV2j5txX8hIXqN7G14mKHUWH0nMDNGjoexS06yAgcjI0v46qfBzmgHq58lepeial5C6/3fTueMcliW5UiXwNw mAqQQgoM= tina.matrossova@gmail.com

Add SSH key

Tapez la commande ssh -T git@github.com puis suivez les instructions (entrez yes, puis votre passcode ssh).

Le message suivant devrait s'afficher :

Hi username! You've successfully authenticated, but Github does not provide shell access.

[(base) kmatrosova@FRLPMC2823 ~ % ssh -T git@github.com The authenticity of host 'github.com (140.82.121.4)' can't be established. ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU. This key is not known by any other names Are you sure you want to continue connecting (yes/no/[fingerprint])? yes Warning: Permanently added 'github.com' (ED25519) to the list of known hosts. [Enter passphrase for key '/Users/kmatrosova/.ssh/id_rsa': Hi kmatrosova! You've successfully authenticated, but GitHub does not provide shell access.

Connexion avec un token HTTPS

- 1. Connectez vous à GitHub, puis allez dans Settings → Developer settings → Tokens (classic)
- 2. Cliquez sur Generate new token (classic)
- 3. Donnez un nom à votre token, sélectionnez une durée de validité et cochez les cases correspondantes au cadre d'utilisation du token. Dans ce cours, nous allons uniquement faire de la gestion de repository, donc nous avons besoin de cocher la case repo.
- 4. Copiez ensuite le token généré et sauvegardez-le dans un lieu sûr. Vous pouvez désormais l'utiliser comme mot de passe lorsqu'il vous sera demandé en ligne de commande sur votre machine.

New person	al access token (classic)
Personal access to for Git over HTTPS,	kens (classic) function like ordinary OAuth access tokens. They can be used instead of a passwor , or can be used to authenticate to the API over Basic Authentication.
Note	
repo access	
What's this token for?	
Expiration *	
30 days 🛛 🌲	The token will expire on Thu, Feb 16 2023
Select scopes Scopes define the a	access for personal tokens. Read more about OAuth scopes.
🗹 repo	Full control of private repositories

GitLab - Connexion en SSH

∭ GitLab ≡ Q Search GitLab		□ ~ D•11) 1ì ~ ⊠10) @•~ 🤇
💭 User Settings	User Settings > SSH Keys	
8 Profile		
8* Account	Q Search page	
88 Applications	SSH Keys	Add an SSH key
🖓 Chat	SSH keys allow you to establish a secure connection	Add an SSH key for secure access to GitLab. Learn more.
Access Tokens	between your computer and GitLab.	Кеу
🖾 Emails	SSH Eingerprints	
	SSH fingerprints verify that the client is connecting to the	
₽ SSH Keys	correct host. Check the current instance configuration.	
🖉 GPG Keys		
₽ Preferences		Begins with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-
豆 Active Sessions		sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'.
Authentication Log		Title
		Example: MacBook key

Key titles are publicly visible.

Usage type

Authentication & Signing

Expiration date

2024-05-12

Optional but recommended. If set, key becomes invalid on the specified date.

8 🖻

V

GitLab - Connexion en SSH

Tapez la commande ssh -T <u>git@gitlab.huma-num.fr</u> puis suivez les instructions...

GitLab - Connexion avec un token HTTPS

	UNCAD .	
💭 User Settings	User Settings > Access Tokens	
Profile	Q Search page	
 ³⁹ Account ³⁹ Account ³⁰ Applications ³¹ Chat ³² Emails ³² Emails ⁴ Notifications ⁴ SSH Keys ⁴ GPG Keys ³² Preferences ³² Active Sessions ³³ Authentication Log 	Q. Search page Parsonal Access Tokens Sucan generate a personal access token for each application you use that needs access to the GitLab application you use that needs access to the GitLab application you use that needs access to the GitLab application you use that needs access tokens to authenticate against Git over HTTP. They are the only accepted password when you have Two-Factor Authentication (2FA) enabled.	Add a personal access token Enter the name of your application, and we'll return a unique personal access token. Token name

NOUVEAU PROJET

04

Créer un projet depuis GitHub

Nous allons maintenant créer un nouveau projet. Allez dans New repository dans le menu déroulant + en haut à droite.



Créer un projet depuis GitHub

Nous allons appeler notre projet tuto-git. Ce projet sera publique (visible par tous les utilisateurs).

Cochez la case Add a README file - nous allons générer un fichier qui servira de descriptif de notre projet. On pourra y mettre des informations utiles à toute personne qui serait intéressée par notre projet.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Repository template

Start your repository with a template repository's contents.

Owner *		Repository name *	
💮 kmatrosova 👻	1	tuto-git	
		🛇 tuto-git is available.	
Great repository names	are	short and memorable. Nee	d inspiration? How about suprem

Public

0

Description (optional)

Anyone on the internet can see this repository. You choose who can commit

Private

You choose who can see and commit to this repository.

Initialize this repository with:

🗹 Add a README file

is is where you can write a long description for your project. Learn more about README

e-octo-lamp?

Créer un projet depuis GitHub

Une fois le projet créé, allez dans votre terminal. Déplacez vous dans le dossier de votre choix (Documents par exemple) avec la commande cd, puis tapez git clone git@github.com:username/tuto-git (en remplaçant username par le votre). Tapez ls (dir sous Windows) - un dossier tuto-git est apparu sur votre machine. Si vous tapez ls tuto-git (dir tuto-git sous Windows) vous verrez que le fichier README est bien dans le dossier.

(base) kmatrosova@FRLPMC2823 ~ % cd Documents (base) kmatrosova@FRLPMC2823 Documents % git clone git@github.com:kmatrosova/tuto-git Cloning into 'tuto-git'... [Enter passphrase for key '/Users/kmatrosova/.ssh/id_rsa': remote: Enumerating objects: 3, done. remote: Counting objects: 100% (3/3), done. remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 Receiving objects: 100% (3/3), done. (base) kmatrosova@FRLPMC2823 Documents % ls 1MMMM tuto-git Docs PhD Zoom (base) kmatrosova@FRLPMC2823 Documents % cd tuto-git (base) kmatrosova@FRLPMC2823 tuto-git %

Faire des modifications en local

Notre projet contiendra un fichier texte test.txt. Il contiendra la phrase "Ceci est un fichier test.". Créez ce fichier dans le repository tuto-git sur votre machine.

Soumettre les modifications

Nous venons de modifier le projet en ajoutant un fichier et voulons maintenant soumettre ces modifications.

Ajoutez le fichier créé avec git add test.txt (utiliser git add -A s'il y à plusieurs fichiers à ajouter).

Si vous voulez systématiquement ajouter tous les nouveaux fichiers, sauf quelques exceptions, vous pouvez créer un fichier .gitignore. Cela peut-être le cas par exemple si vous mettez un dataset dans votre dossier pour tester votre code en local, mais que vous n'avez pas envie de publier le dataset sur GitHub. Les fichiers qui commencent par un point sont des fichiers cachés, pour les voir vous devez entrer la commande ls -a, le -a veut dire qu'on veut voir tous les fichiers qu'il y a dans le dossier actuel. Vous pouvez créer votre fichier .gitignore dans n'importe quel éditeur de texte, et vous pouvez y mettre les noms des fichiers/dossiers concernés (un par ligne).

Exemple d'utilisation d'un fichier .gitignore

Par exemple, vous avez plusieurs fichiers csv qui contiennent des données sensibles. Vous pouvez les ignorer en écrivant *.csv. Le symbole * signifie qu'on veut exclure tous les fichiers.

Si les données sont toutes contenues dans le dossier nommé data, vous pouvez les exclure avec data/*. Vous pouvez être plus précis en indiquant le dossier et l'extension en même temps comme ceci : data/*.csv.

Supposons que vous voulez exclure tous les fichiers du même type sauf un, qui s'appelle par exemple special_data.csv. Vous pouvez d'abord mettre *.csv, puis à la ligne !special_data.csv. Le symbole ! désigne la négation.

Vous pouvez également ajouter des commentaires, en commençant la ligne par #.

Ce type de syntaxe repose sur le principe des **expressions régulières**. Voilà un bon tuto si vous voulez en savoir plus : <u>https://www.lucaswillems.com/fr/articles/25/tutoriel-pour-maitriser-les-expressions-regulieres</u>

Exemple d'utilisation d'un fichier .gitignore

Voilà un exemple de fichier .gitignore.

# Fichiers à ignorer :	← commentaire
requirements.txt	← le fichier requirements.txt
*.log	← tous les fichiers d'extension .log contenus à la racine du projet
log/*.log	← tous les fichiers d'extension .log contenus dans le dossier log
data/*	← tous les fichiers contenus dans le dossier data
!data/subset.csv	← sauf le fichier subset.csv

Soumettre les modifications

Soumettez les modifications avec git commit -m "add test.txt" ("add test.txt" ici est le nom du commit). Il est préférable de choisir un nom cohérent avec les modifications faites afin de pouvoir facilement retracer les changements dans le futur et car il s'agit également d'un nom que tout le monde verra si le dépôt est public.

Prenez l'habitude de faire un commit à chaque fois que vous avez fini de travailler sur une partie de votre code, c'est un bon moyen d'organiser son travail et vous pourrez facilement retracer l'historique des modifications dans le futur. C'est d'autant plus important d'adopter cette habitude dans les projets collaboratifs.

(base) kmatrosova@FRLPMC2823 tuto-git % git commit -m "add test.txt"
[main bc0977b] add test.txt
1 file changed, 1 insertion(+)
create mode 100644 test.txt

Soumettre les modifications

Appliquez les modifications avec git push. Notez que si vous avez choisi l'authentification par token, il faudra le rentrer en tant que mot de passé à cette étape, précédé de votre username.

(base) kmatrosova@FRLPMC2823 tuto-git % git push Enter passphrase for key '/Users/kmatrosova/.ssh/id_rsa': Enumerating objects: 4, done. Counting objects: 100% (4/4), done. Delta compression using up to 8 threads Compressing objects: 100% (2/2), done. Writing objects: 100% (3/3), 301 bytes | 150.00 KiB/s, done. Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 To github.com:kmatrosova/tuto-git dele1fd..bc0977b main -> main (base) kmatrosova@FRLPMC2823 tuto-git % Vous pouvez voir que le fichier test.txt est apparu dans le projet sur GitHub.

🖟 kmatro	osova/tuto	-git Public							Sty Pin	⊙U
<> Code	 Issues 	ີ່ 1 Pull requests	Actions	Η Projects	🛱 Wiki	Security	🗠 Insights	龄 Settin	gs	
	₽ main -	រះ 1 branch 🔊	0 tags				Go to file	Add file -	<> Code	-
	kmatro	osova first commit					dfbdb85 14	minutes ago	🕑 1 comm	iit
	🗋 test.txt		firs	st commit				1	4 minutes ag	0

Faire des modifications depuis GitHub

Vous avez également la possibilité de modifier des fichiers directement sur GitHub. Allez dans le fichier README et ajoutez la phrase "Ceci est un projet test git."

📮 kmatrosova / tuto-git 🖓 Public						☆ Star 0 -
↔ Code ⊙ Issues 1 Pull requests	▹ Actions	Commit changes				
Code	tuto-git / READMI	Commit message		Cancel cha	inges	Commit changes
° main → + Q	Edit Preview	Update <u>README.md</u>		Spaces 🗢		¢ Soft wrap ¢
	1 # tuto-gi	Extended description				
🕒 README.md	2 3 Ceci est	Add an optional extended description				
『 test.txt						
		Commit directly to the main branch				
		Create a new branch for this commit and start a pull requ	est			
		Learn more about pull requests				
		Cancel Commit cha	inges			

Mettre à jour le projet en local

Pour que ces modifications soient prises en compte sur votre machine locale, entrez la commande git pull dans votre terminal.

En général, cette commande est essentielle pour s'assurer que le projet est à jour sur votre machine. Il est recommandé de l'utiliser régulièrement, par exemple une fois par jour avant de procéder à de nouvelles modifications sur le projet.

[(base) kmatrosova@FRLPMC2823 tuto-git % git pull [Enter passphrase for key '/Users/kmatrosova/.ssh/id_rsa': remote: Enumerating objects: 5, done. remote: Counting objects: 100% (5/5), done. remote: Compressing objects: 100% (2/2), done. remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 Unpacking objects: 100% (3/3), 699 bytes | 174.00 KiB/s, done. From github.com:kmatrosova/tuto-git bc0977b..3da033f main -> origin/main Updating bc0977b..3da033f Fast-forward README.md | 4 +++-1 file changed, 3 insertions(+), 1 deletion(-) (base) kmatrosova@FRLPMC2823 tuto-git %

GITHUB DESKTOP



GitHub propose également une interface graphique GitHub Desktop, disponible sous Windows, Mac et Linux. Vous pouvez la télécharger ici <u>https://desktop.github.com/</u>. Installez en suivant les instructions. Lancez GitHub Desktop et ajoutez le projet en cliquant sur Add \rightarrow Add Existing Repository.

Let's get started Add a repository to GitHub Desktop to start coll	laborating		
Create a Tutorial Repository		Filter your repositories	
Add	Local Repository	×	e
Loca	l Path ers/kmatrosova/Documents/tu	to-git Choose	
Add an Existing Repository fro	Cancel	Add Repository	

Ajoutez une phrase au fichier test.txt. Vous allez voir que le changement apparaît sur l'interface. Vous pouvez désormais cliquer sur Commit to main (n'oubliez pas de donner un nom qui fait du sens à votre commit), puis cliquez sur Push origin. Cliquez sur Fetch origin pour récupérer la dernière version du projet.

٠		-				
Current Repository tuto-git	÷	P Current Branch main	~	C Fetch origin Last fetched just	t now	
Changes 1	History	test.txt				
✓ 1 char	ged file	1 1	@@ -1 +1,3 @@	hier test.		
✓ test.txt		2	+ +Ceci est une mo	dification.⊘≁		
		Curren و main	t Branch	•	Push origin Last fetched	a minute ago
Update test.txt						
Description	to main					
Commi						

AJOUTER UN PROJET EXISTANT



Initialiser un projet depuis votre machine

Il est possible que vous ayez déjà des projets en cours sur votre machine pour lesquels vous aurez envie d'utiliser Git. Vous pouvez alors suivre les étapes suivantes :

- 1. Créez un repository vide sur GitHub (ne l'initialisez pas avec un README et/ou autres fichiers pour éviter les conflits).
- 2. Sur votre machine, allez dans le dossier du projet et entrez les commandes suivantes
 - a. git init (initialiser le dossier avec les fichiers de configuration git)
 - b. git add -A
 - c. git commit -m "first commit"
 - d. git branch M main
 - e. git remote add origin git@github.com:username/projectname.git (en changeant username et projectname par votre identifiant GitHub et le nom du repository sur GitHub)
 - f. git push --set-upstream origin main

Dans le cas de l'erreur remote origin already exists, utilisez la commande git remote remove origin, puis reprenez à l'étape e.

3. Vérifiez que vos fichiers sont apparus dans le repository sur GitHub. Vous pouvez désormais utiliser les commandes git habituelles.

Créez projet Quarto avec Rstudio : File → New Project → New Directory → Quarto Project



Créez un nouveau dépôt sur GitHub.





Ajouter le projet (quarto) existant dans le nouveau répertoire créé :

- Allez dans le dossier du projet. Exemple : cd .\Documents\Mon_doc_quarto\
- Entrez les commandes suivantes :
 - a. git init (initialiser le dossier avec les fichiers de configuration git)
 - b. git add -A
 - c. git commit -m "first commit"
 - d. git branch -M main
 - e. git remote add origin git@github.com:username/projectname.git
 - f. git push --set-upstream origin main

Dans le cas de l'erreur remote origin already exists, utilisez la commande git remote remove origin, puis reprenez à l'étape e.

Vérifiez que vos fichiers sont apparus dans le repository sur GitHub.

ę	main 🗸 🦻 1 branch 🔊 0 tags	1	Go to file Add file - <> Code -
0	HuguesPecout first commit		f4f2e59 15 minutes ago 🕥1 commit
	.Rproj.user	first commit	15 minutes ago
	.quarto	first commit	15 minutes ago
	Mon_doc_quarto_files/libs	first commit	15 minutes ago
ß	Mon_doc_quarto.Rproj	first commit	15 minutes ago
Ľ	Mon_doc_quarto.html	first commit	15 minutes ago
ß	Mon_doc_quarto.qmd	first commit	15 minutes ago
ß	_quarto.yml	first commit	15 minutes ago

Help people interested in this repository understand your project by adding a README.

Add a README

(SERVIR) DÉPLOYER UN FICHIER HTML



Il existe plusieurs façons pour déployer un fichier HTML via GitHub (ou GitLab). Nous présentons ici la solution la plus simple et basique.

- 1. Commencez par renommer votre fichier quarto index.qmd.
- 2. Compiler le notebook quarto en format HTML en cliquant sur Render



Le document Quarto en format HTM s'affiche dans le Viewer et le fichier index.html a été enregistré à la racine du répertoire projet.

Le fichier HTML à déployer doit obligatoirement se nommer index.html



Ajouter ces nouveaux fichiers (et les modifications) sur le dépôt GitHub :

- a. git add -A
- b. git commit -m "add render result of my quarto doc"
- c. git push

Console	Terminal	Background Jobs	
•	Terminal 1 🝷	MINGW64:/c/Users/	HP/Documents/Mon_doc_quarto
HP@DESK \$ git p Enumera Countin Delta c Compres Writing Total 2 remote: To http f4f2	TOP-8CKOQ8 ush ting objects: ompression sing objects: 8 (delta 5 Resolving s://github e596d6ac	P MINGW64 /c/Us ts: 48, done. 100% (48/48), using up to 16 ts: 100% (19/19 100% (28/28), 2), reused 0 (de deltas: 100% (com/HuguesPecc 59 main -> mai	done. threads), done. 558 KiB 264.00 KiB/s, done. 11a 0), pack-reused 0 5/5), completed with 4 local objects. but/delete.git n

1. Sur la page du dépôt GitHub, allez dans Settings → pages, sélectionnez la branche main, enregistrez.

C Visit site

...



2. Rafraichissez la page quelques instants après... Votre page est en ligne !

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.



Vous pouvez ajouter le lien de la page déployée par GitHub en ajoutant l'URL dans le fichier README.md pour faciliter l'accès depuis le dépôt du code source du projet.

0	HuguesPecout update		✓ e4b43df 12 minutes ago 🕚 12 commits
	.Rproj.user	update	12 minutes ago
	.quarto	first commit	47 minutes ago
	index_files/libs	update	12 minutes ago
Ľ	.gitignore	first commit	1 hour ago
ß	.nojekyll	first commit	1 hour ago
ľ	Mon_doc_quarto.Rproj	first commit	2 hours ago
C	README.md	Create README.md	1 hour ago
ľ	_quarto.yml	update	12 minutes ago
ľ	index.html	update	12 minutes ago
ľ	index.qmd	update	12 minutes ago

DÉPLOYER UN FICHIER HTML

README.md

Mon document

https://huguespecout.github.io/delete/

0

Déployer un HTML avec GitLab

.

La <u>méthode de déploiement de documents HTML avec GitLab</u> est différente. Il suffit d'ajouter un fichier appelé .gitlab-ci.yml à la racine de votre dépôt, qui contient la configuration de l'intégration (ou déploiement continu, CI/CD). Exemple :

Cl/CD configuration Add LICENSE Add CHANGELOG Add CONTRIBUTING Auto DevOps enabled Add Kubernetes cluster Add Kubernetes clus	de_pres_paris / + ~			Find file	Web IDE		
Image Last commit ing final version index_files/libs final version • .gitlab-ci.yml 1 pages: stage: deplo 3 script: • .gitlab-ci.yml 1 • .gitlab-ci.yml<	CI/CD configuration	NSE 🗄 Add CHANGELOG 🕀 Add CONTRIBUTING	Auto DevOps enabled	🕀 Add K	ubernetes cluster 🕀 Add Wiki		
NameLast commitD imgfinal versionD index_files/libsfinal versionC index_files/libsfinal versionI index_files/libsfinal versionI initialgitlab-ci.ymlI initialgitlab-ci.ymlI initialupdate.gitlab-ci.ymlI initial- mkdir .pulI initial- cp - r * .jI index.htmlfinal versionI index.htmlfinal versionI index.ntmlfinal versionI index.qmdfinal versionI index.qmdfinal versionI index.ntmlfinal versionI index.qmdfinal version </td <td>egrations</td> <td></td> <td></td> <td></td> <td></td>	egrations						
E indg final version 1 pages: E index_files/libs final version 2 stage: deplet Image: stage: deplet 3 script: 3 Image: stage: deplet 3 script: 3 <		Last commit	Last commit 🦊		.gitlab-ci.yml 🕻 147 bytes		
Implicit in the second seco		final version		1	pages:		
 gitignore initial gitignore gitignore update .gitlab-ci.yml mw .public public update .gitlab-ci.yml update .gitlab-ci.yml update .gitlab-ci.yml gitlab-ci.yml gitlab-ci.yml<td>ibs</td><td>final version</td><td></td><td>2</td><td>stage: deploy</td>	ibs	final version		2	stage: deploy		
 gitignore initial gitignore gitignore update .gitlab-ci.yml update .gitlab-ci.yml cp -r * .g - mkdir .pul - cp -r * .g - mv .public - mv .public - public - public - only: 				3	script:		
Update .gitlab-ci.yml 5 - Cp - r * .j Memory README.md 0 - mv .public Image: Index.html final version 8 Image: Index.qmd final version 9 Image: Index.qmd final version 10		initial	initial		- mkdir .public		
Mit README.md 0 - mv .public Mit README.md 7 artifacts: I index.html final version 8 paths: I index.qmd final version 9 - public	ıl	Update .gitlab-ci.yml	Update .gitlab-ci.yml		- cp -r * .publi		
Pre-README.md Update README.md 7 artifacts: I index.html final version 8 paths: I index.qmd final version 9 - public					- mv .public pub		
index.ntml final version 8 paths: index.qmd final version 9 - public only: 10 only:	1	Update README.md	Update README.md		artifacts:		
index.qmd final version 9 - public 0 only:		final version		8	paths:		
Bindex.qmd final version 10 only:				9	- public		
		final version		10	only:		
🕒 slide_pres_p.a.r.i.s.Rproj initial 11 - main	o.a.r.i.s.Rproj	initial	initial		- main		

INTÉGRATION CONTINUE ET COLLECTIVE



Intégration continue collective - Mise en pratique

Nous allons apporter, tour à tour, des modifications à un document Quarto déployé via Github.

Pour cela, les droits de modifications d'un dépôt contenant un document Quarto vous ont été ouverts.

- 1. Cloner ce dépôt <u>https://github.com/HuguesPecout/delete</u> sur votre machine :
 - a. cd Documents
 - b. git clone git@github.com:HuguesPecout/delete.git
- 2. Chaque participant.es, à tour de rôle :
 - a. Si des modifications ont eu lieu sur le dépôt depuis votre clone : git pull
 - b. Apportez une modification au fichier index.qmd
 - c. Render le fichier index.qmd
 - d. git add -A
 - e. git commit -m "modification du titre"
 - f. git push

Les modifications sont rapidement répercutées sur le fichier HTML déployé en ligne.

INTRODUCTION AU TRAVAIL COLLABORATIF



INTRODUCTION AU TRAVAIL COLLABORATIF



Branches

Créer une branche signifie diverger de la ligne principale de développement et continuer à travailler sans impacter cette ligne. Cela se fait lors d'un travail en groupe, mais il est aussi possible de créer sa propre branche sur n'importe quel repository publique, afin de l'utiliser pour ses propres besoins, ou bien pour apporter une contribution au projet initial.



Someone Else's Work

Une PullRequest - PR est une demande de relecture de code par un pair développeur avant le merge d'une branche sur une autre.

On appelle merge le fait de réunir le code de deux branches sur une unique branche.

Lorsque deux personnes apportent des modifications sur le même fichier dans deux branches différentes, puis essayent de les merge, cela peut créer un conflit.