**GIT**

**Définition :**

Git est un [logiciel de gestion de versions](https://fr.wikipedia.org/wiki/Logiciel_de_gestion_de_versions) [décentralisé](https://fr.wikipedia.org/wiki/Gestion_de_versions#Gestion_de_versions_d.C3.A9centralis.C3.A9e).

utilise un système de connexion [pair à pair](https://fr.wikipedia.org/wiki/Pair_%C3%A0_pair). Le code informatique développé est stocké non seulement sur l’ordinateur de chaque contributeur du projet, mais il peut également l'être sur un serveur dédié

la principale tâche est de gérer l'évolution du contenu d'une [arborescence](https://fr.wikipedia.org/wiki/Arborescence).

Git possède deux [structures de données](https://fr.wikipedia.org/wiki/Structure_de_donn%C3%A9es) : une base d'objets et un cache de [répertoires](https://fr.wikipedia.org/wiki/R%C3%A9pertoire_(informatique)). Il existe cinq types d'objets :

**Objets GIT**

* 'objet blob (pour [binary large object](https://fr.wikipedia.org/wiki/Binary_large_object" \o "Binary large object) désignant un ensemble de données brutes[7](https://fr.wikipedia.org/wiki/Git#cite_note-:0-7)), qui représente le contenu d'un [fichier](https://fr.wikipedia.org/wiki/Fichier_informatique) ;
* l'objet tree (mot anglais signifiant arbre), qui décrit une arborescence de fichiers. Il est constitué d'une liste d'objets de type blobs et des informations qui leur sont associées, tel que le nom du fichier et les [permissions](https://fr.wikipedia.org/wiki/Droits_sur_un_fichier). Il peut contenir [récursivement](https://fr.wikipedia.org/wiki/Type_r%C3%A9cursif) d'autres trees pour représenter les sous-répertoires ;
* l'objet commit (résultat de l'opération du même nom signifiant « valider une transaction »[8](https://fr.wikipedia.org/wiki/Git#cite_note-8)), qui correspond à une arborescence de fichiers (tree) enrichie de métadonnées comme un message de description, le nom de l'auteur, etc. Il pointe également vers un ou plusieurs objets commit parents pour former un graphe d'historiques[7](https://fr.wikipedia.org/wiki/Git#cite_note-:0-7) ;
* l'objet tag (étiquette) qui est une manière de nommer arbitrairement un commit spécifique pour l'identifier plus facilement. Il est en général utilisé pour marquer certains commits, par exemple par un numéro ou un nom de version (2.1 ou bien [Lucid Lynx](https://fr.wikipedia.org/wiki/Liste_des_versions_d%27Ubuntu" \l "Ubuntu_10.04_LTS_.28Lucid_Lynx.29" \o "Liste des versions d'Ubuntu));
* l'objet branch (branche) qui contient une partie de l'avancement du projet. Les branches sont souvent utilisées pour avancer dans une partie du projet sans impacter une autre partie.