**DESCRIPTION**

Récupérer des branches et/ou des étiquettes (collectivement, "réfs") depuis un ou plusieurs autres dépôts, ainsi que les objets nécessaires pour compléter leur historique. Les branches de suivi à distance sont mises à jour (voir la description de <spéc-de-réf> ci-dessous pour les moyens de contrôler ce comportement).

Par défaut, toute étiquette qui pointe vers les historiques recherchés est également recherchée ; l’effet est de rechercher les étiquettes qui pointent vers les branches qui vous intéressent. Ce comportement par défaut peut être modifié en utilisant les options --tags ou --no-tags ou en configurant remote.<nom>.tagOpt. En utilisant une spécification de référence qui récupère les étiquettes explicitement, vous pouvez également récupérer les étiquettes qui ne pointent pas sur les branches qui vous intéressent.

**git fetch** peut aller chercher à partir d’un seul dépôt nommé ou d’une seule URL, ou à partir de plusieurs dépôts à la fois si <groupe> est donné et qu’il y a une entrée remotes.<groupe> dans le fichier de configuration. (Voir [git-config[1]](https://git-scm.com/docs/git-config/fr)).

Lorsqu’aucun distant n’est spécifié, par défaut le distant origin sera utilisé, à moins qu’il n’y ait une branche amont configurée pour la branche courante.

Les noms des refs qui sont récupérés, ainsi que les noms des objets qu’ils pointent, sont écrits dans .git/FETCH\_HEAD. Ces informations peuvent être utilisées par des scripts ou d’autres commandes git, telles que [git-pull[1]](https://git-scm.com/docs/git-pull/fr).

**OPTIONS**

**--all**

Récupérer tous les distants.

**-a**

**--append**

Ajouter les noms de références et les noms d’objets des références récupérées au contenu existant de .git/FETCH\_HEAD. Sans cette option, les anciennes données dans .git/FETCH\_HEAD seront écrasées.

**--depth=<profondeur>**

Limiter la récupération au nombre spécifié de commits à partir du sommet de l’historique de chaque branche distante. Si vous allez chercher dans un dépôt "superficiel" créé par git clone avec l’option --depth=<profondeur> (voir [git-clone[1]](https://git-scm.com/docs/git-clone/fr)), approfondir ou raccourcir l’historique jusqu’au nombre spécifié de validations. Les étiquettes pour les commits approfondis ne sont pas récupérées.

**--deepen=<profondeur>**

Semblable à --depth, sauf que cela précise le nombre de commits à partir de la limite actuelle superficielle au lieu du sommet de l’historique de chaque branche distante.

**--shallow-since=<date>**

Approfondir ou raccourcir l’historique d’un dépôt superficiel pour inclure tous les commits accessibles après <date>.

**--shallow-exclude=<révision>**

Approfondir ou raccourcir l’historique d’un dépôt superficiel afin d’exclure les commits accessibles depuis une branche ou une étiquette distante spécifiée. Cette option peut être spécifiée plusieurs fois.

**--unshallow**

Si le dépôt de sources est complet, convertir un dépôt superficiel en un dépôt complet, en supprimant toutes les limitations imposées par les dépôts superficiels.

Si le dépôt source est superficiel, il faut en extraire le plus possible afin que le dépôt actuel ait le même historique que le dépôt source.

**--update-shallow**

Par défaut, lors de la récupération d’un dépôt superficiel, git fetch refuse les références qui nécessitent une mise à jour de .git/shallow. Cette option met à jour le fichier .git/shallow et accepte de telles références.

**--negotiation-tip=<commit|glob>**

Par défaut, Git signalera au serveur les commits accessibles à partir de toutes les références locales pour trouver les commits communs afin de réduire la taille du fichier de paquet à recevoir. Si ceci est spécifié, Git ne signalera que les commits accessibles à partir des sommets donnés. Ceci est utile pour accélérer les recherches lorsque l’utilisateur sait quelle réf locale est susceptible d’avoir des commits en commun avec la réf amont qui est recherchée.

Cette option peut être spécifiée plus d’une fois ; si c’est le cas, Git signalera les commits accessibles à partir de l’un des commits donnés.

L’argument de cette option peut être un glob sur les noms de référence, une référence ou le SHA-1 (éventuellement abrégé) d’un commit. La spécification d’un glob équivaut à spécifier cette option plusieurs fois, une pour chaque nom de référence correspondant.

Voir aussi la variable de configuration fetch.negotiationAlgorithm documentée dans [git-config[1]](https://git-scm.com/docs/git-config/fr).

**--dry-run**

Montrer ce qui serait fait, sans faire de changements.

**-f**

**--force**

Lorsque **git fetch** est utilisé avec la spécification de référence <src>:<dst>, il peut refuser de mettre à jour la branche locale comme cela a été discuté dans la partie <spécificateur-de-référence> ci-dessous. Cette option permet de passer outre à ce contrôle.

**-k**

**--keep**

Conserver le paquet téléchargé.

**--multiple**

Permettre de spécifier plusieurs arguments <dépôt> et <groupe>. Aucun <spécificateur-de-référence> ne peut être spécifié.

**--[no-]auto-gc**

Exécuter git gc --auto à la fin pour effectuer le ramassage des ordures si nécessaire. Ceci est activé par défaut.

**--[no-]write-commit-graph**

Écrire un graphe de commit après avoir récupéré. Ceci remplace le paramètre de configuration fetch.writeCommitGraph.

**-p**

**--prune**

Avant de récupérer, supprimer toutes les références de suivi à distance qui n’existent plus sur le dépôt distant. Les étiquettes ne sont pas sujettes à l’élagage si elles ne sont récupérées qu’en raison du suivi automatique de l’étiquette par défaut ou en raison d’une option --tags. Cependant, si les étiquettes sont récupérées en raison d’un spécificateur de référence explicite (soit en ligne de commande, soit dans la configuration distante, par exemple si le dépôt distant a été cloné avec l’option --mirror), alors elles sont également sujettes à l’élagage. La fourniture de --prune-tags est une abréviation pour la fourniture du spécificateur de référence d’étiquette.

Voir la section ÉLAGAGE ci-dessous pour plus de détails.

**-P**

**--prune-tags**

Avant de récupérer, supprimer toutes les étiquettes locales qui n’existent plus sur le distant si --prune est activé. Cette option doit être utilisée avec plus de précaution, car contrairement à --prune, elle supprime toutes les références locales (étiquettes locales) qui ont été créées. Cette option est un raccourci pour la fourniture du spécificateur de référence d’étiquette explicite avec --prune, voir la discussion à ce sujet dans sa documentation.

Voir la section ÉLAGAGE ci-dessous pour plus de détails.

**-n**

**--no-tags**

Par défaut, les étiquettes qui pointent sur des objets téléchargés à partir du dépôt distant sont récupérées et stockées localement. Cette option désactive le suivi automatique des étiquettes. Le comportement par défaut d’un distant peut être spécifié avec le paramètre remote.<nom>.tagOpt. Voir [git-config[1]](https://git-scm.com/docs/git-config/fr).

**--refmap=<spécificateur-de-référence>**

Lors de la récupération des références listées en ligne de commande, utiliser la spécification de référence (qui peut être donnée plusieurs fois) pour mapper les références sur les branches de suivi à distance, au lieu des valeurs des variables de configuration remote.\*.fetch pour le dépôt distant. Fournir un <spécificateur-de-référence> vide à l’option --refmap fait que Git ignore les spécification de référence configurées et se fie entièrement aux spécifications de référence fournies comme arguments de la ligne de commande. Voir la section sur les "Branches de suivi à distance configurées" pour plus de détails.

**-t**

**--tags**

Récupérer toutes les étiquettes à distance (c’est-à-dire, récupérer les étiquettes refs/tags/\* dans les étiquettes locales avec le même nom), en plus de tout ce qui serait récupéré autrement. L’utilisation de cette seule option ne soumet pas les étiquettes à un élagage, même si --prune est utilisé (bien que les étiquettes puissent être élaguées de toute façon si elles sont aussi la destination d’une spécification de référence explicite ; voir --prune).

**--recurse-submodules[=yes|on-demand|no]**

Cette option permet de contrôler si et dans quelles conditions les nouveaux engagements de sous-modules peuplés doivent également être récupérés. Elle peut être utilisée comme une option booléenne pour désactiver complètement la récursivité lorsqu’elle est définie sur **no** ou pour visiter récursivement sans condition tous les sous-modules peuplés lorsqu’elle est définie sur **yes**, qui est la valeur par défaut lorsque cette option est utilisée sans aucune valeur. Utiliser **on-demand** pour ne visiter récursivement un sous-module peuplé que lorsque le superprojet récupère un commit qui met à jour la référence du sous-module à un commit qui n’est pas déjà dans le clone local du sous-module. Par défaut, **on-demand** est utilisé à moins que fetch.recurseSubmodules ne soit paramétré (voir [git-config[1]](https://git-scm.com/docs/git-config/fr)).

**-j**

**--jobs=<n>**

Nombre d’enfants parallèles à utiliser pour toutes les formes d’extraction.

Si l’option --multiple a été spécifiée, les différents distants seront récupérés en parallèle. Si plusieurs sous-modules sont récupérés, ils seront récupérés en parallèle. Pour les contrôler indépendamment, utilisez les paramètres de configuration fetch.parallel et submodule.fetchJobs (voir [git-config[1]](https://git-scm.com/docs/git-config/fr)).

Généralement, les recherches récursives parallèles et sur des distants multiples seront plus rapides. Par défaut, les recherches sont effectuées de manière séquentielle, et non en parallèle.

**--no-recurse-submodules**

Désactiver la récupération récursive des sous-modules (cela a le même effet que d’utiliser l’option --recurse-submodules=no).

**--set-upstream**

Si le distant est récupéré avec succès, tirer et ajouter la référence (de suivi) amont , utilisée par les commandes sans argument [git-pull[1]](https://git-scm.com/docs/git-pull/fr) et autres. Pour plus d’informations, voir branch.<nom>.merge et branch.<nom>.remote dans [git-config[1]](https://git-scm.com/docs/git-config/fr).

**--submodule-prefix=<chemin>**

Préfixer <chemin> aux chemins affichés dans les messages informatifs tels que "Récupération du sous-module foo". Cette option est utilisée en interne lors de la récursion sur les sous-modules.

**--recurse-submodules-default=[yes|on-demand]**

Cette option est utilisée en interne pour fournir temporairement une valeur par défaut non négative pour l’option --recurse-submodules. Toutes les autres méthodes de configuration de la récupération récursive des sous-module (comme les paramètres de [gitmodules[5]](https://git-scm.com/docs/gitmodules/fr) et [git-config[1]](https://git-scm.com/docs/git-config/fr)) remplacent cette option, tout comme le fait de spécifier directement --[no-]recurse-submodules.

**-u**

**--update-head-ok**

Par défaut, **git fetch** refuse de mettre à jour la tête qui correspond à la branche en cours. Ce drapeau désactive la vérification. C’est purement pour l’usage interne de **git pull** pour communiquer avec **git fetch**, et à moins que vous n’implémentiez votre propre Porcelaine, vous n’êtes pas censé l’utiliser.

**--upload-pack <upload-pack>**

Lorsqu’il est donné, et que le dépôt à récupérer est géré par **git fetch-pack**, --exec=<upload-pack> est passé à la commande pour spécifier le chemin par défaut pour la commande exécutée à l’autre bout.

**-q**

**--quiet**

Passer — quiet pour git-fetch-pack et faire taire toute autre commande git utilisée en interne. La progression n’est pas signalée dans le flux d’erreurs standard.

**-v**

**--verbose**

Mode bavard.

**--progress**

L’état d’avancement est affiché sur la sortie d’erreur standard quand elle est attachée à un terminal, à moins que -q soit spécifié. Ce drapeau force l’état d’avancement même si le flux d’erreur standard n’est pas dirigé vers un terminal.

**-o <option>**

**--server-option=<option>**

Transmettre la chaîne donnée au serveur lors d’une communication utilisant la version 2 du protocole. La chaîne donnée ne doit pas contenir de caractère NUL ou LF. La gestion par le serveur des options du serveur, y compris les options inconnues, est spécifique au serveur. Lorsque plusieurs ‘---server-option=<option>’ sont donnés, ils sont tous envoyés à l’autre côté dans l’ordre indiqué sur la ligne de commande.

**--show-forced-updates**

Par défaut, git vérifie si une branche est mise à jour de force pendant la récupération. Cela peut être désactivé via fetch.showForcedUpdates, mais l’option --show-forced-updates garantit que cette vérification a lieu. Voir [git-config[1]](https://git-scm.com/docs/git-config/fr).

**--no-show-forced-updates**

Par défaut, git vérifie si une branche est mise à jour de force pendant la récupération. Passer --no-show-forced-updates ou régler fetch.showForcedUpdates à false pour sauter cette vérification pour des raisons de performance. Si elle est utilisée pendant **git-pull**, l’option --ff-only vérifiera toujours les mises à jour forcées avant de tenter une mise à jour rapide. Voir [git-config[1]](https://git-scm.com/docs/git-config/fr).

**-4**

**--ipv4**

Utiliser uniquement les adresses IPv4, en ignorant les adresses IPv6.

**-6**

**--ipv6**

Utiliser uniquement les adresses IPv6, en ignorant les adresses IPv4.

**<dépôt>**

Le dépôt "distant" qui est la source d’une opération de récupération ou de tirage. Ce paramètre peut être soit une URL (voir la section [URLS GIT](https://git-scm.com/docs/git-fetch/fr#URLS) ci-dessous) soit le nom d’un remote (voir la section [DISTANTS](https://git-scm.com/docs/git-fetch/fr#REMOTES) ci-dessous).

**<groupe>**

Un nom faisant référence à une liste des dépôts comme la valeur de remotes.<groupe> dans le fichier de configuration. (voir [git-config[1]](https://git-scm.com/docs/git-config/fr)).

**<spécificateur-de-référence>**

Préciser les références à récupérer et les références locales à mettre à jour. Lorsqu’aucun <spéc-de-réf> n’apparaît sur la ligne de commande, les références à récupérer sont lues à partir des variables remote.<dépôt>.fetch à la place (voir [BRANCHES DE SUIVI À DISTANCE CONFIGURÉES](https://git-scm.com/docs/git-fetch/fr#CRTB) ci-dessous).

Le format d’un paramètre <spéc-de-réf> est un plus + optionnel, suivi de la source <src>, suivi de deux points :, suivi de la destination ref <dst>. Les deux points peuvent être omis lorsque <dst> est vide. <src> est typiquement une réf, mais cela peut aussi être un nom d’objet hexadécimal entier.

tag <étiquette> signifie la même chose que refs/tags/<tag>:refs/tags/<tag> ; cela demande de tout récupérer jusqu’à l’étiquette donnée.

La référence distante qui correspond à <src> est récupérée, et si <dst> n’est pas une chaîne vide, une tentative est faite pour mettre à jour la référence locale qui lui correspond.

Le fait que cette mise à jour soit autorisée sans ---force dépend de l’espace de noms de référence vers lequel elle est récupérée, du type d’objet récupéré, et si la mise à jour est considérée comme une avance rapide. Généralement, les mêmes règles s’appliquent pour la récupération que pour la poussée, voir la section <spéc-de-réf>... de [git-push[1]](https://git-scm.com/docs/git-push/fr) pour les connaître. Les exceptions à ces règles particulières à **git fetch** sont notées ci-dessous.

Jusqu’à la version 2.20 de Git, et contrairement à ce qui se passe avec [git-push[1]](https://git-scm.com/docs/git-push/fr), toute mise à jour de refs/tags/\* serait acceptée sans + dans la spéc-de-réf (ou --force). Lors de la récupération, nous considérons sans distinction toutes les mises à jour d’étiquettes depuis un dépôt distance comme des récupérations forcées. Depuis la version 2.20 de Git, la récupération pour mettre à jour les refs/tags/\* fonctionne de la même manière que lors de la poussée. C’est-à-dire que toute mise à jour sera rejetée sans "+" dans le spécificateur de référence (ou --force).

Contrairement à une poussée avec [git-push[1]](https://git-scm.com/docs/git-push/fr), toute mise à jour en dehors de refs/{tags,heads}/\* sera acceptée sans + dans le spéc-de-réf (ou --force), que ce soit en échangeant par exemple un objet arbre pour un blob, ou un commit pour un autre commit qui n’a pas le commit précédent comme ancêtre etc.

Contrairement à une poussée avec [git-push[1]](https://git-scm.com/docs/git-push/fr), il n’y a pas de configuration qui modifie ces règles, et rien de tel qu’un crochet pré-récupération pre-fetch analogue à celui de pré-réception`pre-receive`.