GIN CLI Usage Cheat Sheet

Logging, setup and information

Checking setup and help

```
gin --version
```

If everything is setup properly, you will receive an output with the installed versions of GIN, git and git-annex.

Logging in and out of GIN services

```
gin login
```

Login to the GIN services. If no username is specified on the command line, you will be prompted for it. The login command always prompts for a password.

```
gin logout
```

Logout of the GIN services. This command takes no arguments.

Listing remote repositories

```
gin repos [<username>]
```

List repositories on the server that provide read access. If no argument is provided, it will list the repositories owned by the logged in user.

Either a flag `--shared` or `--all` may be used.

Specific username and repository information

```
gin info [<username>]
```

Print user information. If no argument is provided, it will print the information of the currently logged in user.

```
gin repoinfo [<repopath>]
```

Show the information for a specific repository on the server. This can be used to check if the logged in user has access to a specific repository.

Creating and cloning repositories

Creating a new repository

```
gin create <reponame> <description>
```

Create a new repository on the GIN server and optionally clone it locally or initialise working directory using the flags `--here`, `--no-clone`, `--server`.

Initialising a local repository as a GIN repository

```
gin init
```

Initialise a local repository in the current directory with the default options.

Retrieving a repository from remote server

```
gin get <repopath>
```

Download a remote repository to a new directory and initialise the directory with the default options. The local directory is referred to as the 'clone' of the repository.

Updating, fetching and removing repository contents

Retrieving a repository from remote server

```
gin download
```

Downloads changes from the remote repository to the local clone.

Optionally downloads the content of all files in the repository. If '`--content`' is not specified, new files will be empty placeholders.

Upload local changes to a remote repository

```
gin upload [<filenames>]...
```

Uploads all changes made in a local repository clone to the remote repository on the GIN server. This command must be called from within the local repository clone.

You can specify which remote(s) the content will be uploaded to using the `--to` flag. If the keyword '`all`' is specified as a remote, the data is uploaded to all configured remotes.

If no arguments are specified, only changes to files already being tracked are uploaded.

List the sync status of files in the local repository

```
gin ls [<filenames>]...
```

List files or the contents of directories and the status of the files within it.

- OK: The file is part of the GIN repository and its contents are synchronised with the server.
- TC: The file has been locked or unlocked and the change has not been recorded yet (and it is unmodified).
- NC: The local file is a placeholder and its contents have not been downloaded.
- MD: The file has been modified locally and the changes have not been recorded yet.
- LC: The file has been modified locally, the changes have been recorded but they haven't been uploaded.
- RM: The file has been removed from the repository.
- ??: The file is not under repository control.

Download the content of files from a remote repository

```
gin get-content [<filenames>]...
```

GIN CLI Usage Cheat Sheet

Download the annexed content of the listed placeholder files. This command must be called from within the local repository clone. With no arguments, downloads the content for all files under the working directory, recursively.

Remove the content of local files that have already been uploaded

```
gin remove-content [<filenames>]...
```

Remove the content of local files. This command will not remove the content of files that have not been already uploaded to a remote repository, even if the user specifies such files explicitly. Removed content can be retrieved from the server by using the 'get-content' command. With no arguments, removes the content of all files under the current working directory, as long as they have been safely uploaded to a remote repository.

Sync all new information bidirectionally

```
gin sync [--content]
```

Synchronises changes bidirectionally between remote repositories and the local clone.

Optionally downloads and uploads the content of all files in the repository. If 'content' is not specified, new files will be empty placeholders.

Managing changes in a repository

Locking and unlocking files

```
gin lock <filenames>...
```

Lock one or more files to prevent editing.

Locked files are replaced by pointer files in the working directory (or symbolic links where supported by the filesystem).

```
gin unlock <filenames>...
```

Unlock one or more files to allow editing.

A 'commit' command is required to save the locking or unlocking change. Locked and unmodified unlocked files that have not yet been committed are marked as 'Lock status changed' (short TC).

Locking and unlocking a file takes longer depending on its size.

Record changes in local repository

```
gin commit [--message] [<filenames>]
```

Record changes made in a local repository. This command must be called from within the local repository clone.

All changes made to the files and directories that are specified will be recorded, including addition of new files, modifications and renaming of existing files, and file deletions.

If no arguments are specified, no changes are recorded.

Managing remote repositories

Adding a remote to the current repository

```
gin add-remote [<name>] [<location>]
```

Add a remote to the current repository for uploading and downloading. The name of the remote can be any word except the reserved keyword 'all'.

The location must be of the form alias:path or server:path. Currently supported aliases are 'gin' for the default configured gin server, and 'dir' for directories. If neither is specified, it is assumed to be the address of a git server. For gin remotes, the path is the location of the repository on the server, in the form user/repositoryname. For directories, it is the path to the storage directory.

When a remote is added, if it does not exist, the client will offer to create it. This is only possible for 'gin' and 'dir' type remotes and any other GIN servers the user has configured.

A new remote is set as the default for uploading if no other remotes are configured. To set any new remote as the default, use the –default option.

Removing a remote from the current repository

```
gin remove-remote [<name>]
```

Remove a remote from the current repository. It takes as argument the name of the remote repository.

Set the repository's default upload remote

```
gin use-remote [<name>]
```

Set the default remote for uploading. The default remote is used when 'gin upload' is invoked without specifying the –to option.

With no arguments, this command simply prints the currently configured default remote.

List the repository's configured remotes

```
gin remotes
```

List configured remotes and their information.

NOTE: Most gin commands take the flag `--json` to print the output in JSON format. Check `gin help <command>` for more information on flags and arguments of each command.

GIN Web Service
https://gin.g-node.org/
GIN Information Page
https://gin.g-node.org/G-Node/Info/wiki