KENZAN
An Amdocs Company

Git Cheat Sheet

0. Installation

```
$ sudo apt-get install git
```

Install Git using the GNU/Linux `apt-get` command. To install from the source, download from **https://git-scm.com/downloads**

1. Setup

```
$ git config --global user.name “Your Name”
```

Set the name to be associated with commits.

```
$ git config --global user.email “email@example.com”
```

Set the email to be associated with commits.

2. Getting Started

```
$ git init [project_name]
```

Create a new local repository in the current directory. If **project_name** is provided, a new directory will be created.

```
$ git clone project_url [dir]
```

Clone repository. If **dir** is provided, a the repo will be in a new directory.

3. Day to Day

```
$ git status
```

See the status of your files.

```
$ git diff [options]
```

Show changes between files and branches.

```
$ git checkout [<branch>] -- <file> ...
```

Change branches or discard changes to the provided files.
Note: discarding changes can not be undone.

```
$ git add <file> ...
```

Stage the provided files.

```
$ git reset <file> ...
```

Reset staged file(s) to working directory.

```
$ git commit <file> ... -m <message>
```

Commit files with a message. **Note:** A message is required in this form.

```
$ git rm <file> ...
```

Remove **<file>** from repository.

```
$ git stash [pop]
```

Put changes into stash. Include **pop** to retrieve that latest stashed items.

KENZAN
An Amdocs Company

Git Cheat Sheet, pg2

4. Branching

```
$ git branch [-a]
```

List all branches. Include **-a** to list remote branches.

```
$ git branch <branch_name>
```

Create a new branch named **<branch_name>**.

```
$ git checkout -b <branch_name>
```

Create a new branch named **<branch_name>,** as well as check out the new branch.

```
$ git branch -d <branch_name>
```

Deletes the branch named **<branch_name>**.

5. Tagging

```
$ git tag
```

Lists all tags.

```
$ git tag <tag_name> [commit]
```

Create a new tag reference named **<tag_name>** for the latest commit. Include a **commit** sha to tag specific commit.

```
$ git tag -d <tag_name>
```

Delete tag named **<tag_name>**.

6. Undoing

```
$ git reset [--hard] <commit>
```

Reset working directory to provided **commit**. Providing **--hard** will discard changes.

```
$ git revert <commit> ...
```

Revert existing commits.

7. Sharing Files

```
$ git fetch [remote]
```

Download refs from the provided **remote** name. When a remote is not given, refs will be downloaded from the `origin`.

```
$ git fetch --prune [remote]
```

Remove remote refs from the provided **remote** name which were removed from the remote repo.

```
$ git pull [remote] [branch]
```

Retrieve changes from **remote** and merge into the current branch. When **branch** is provided, remote branch changes are merged.

```
$ git push [--tags] [--follow-tags] [remote]
```

Update **remote** refs. Providing **--tags** will update tag refs, but not commits. Providing **--follow-tags** will update both commits and tags