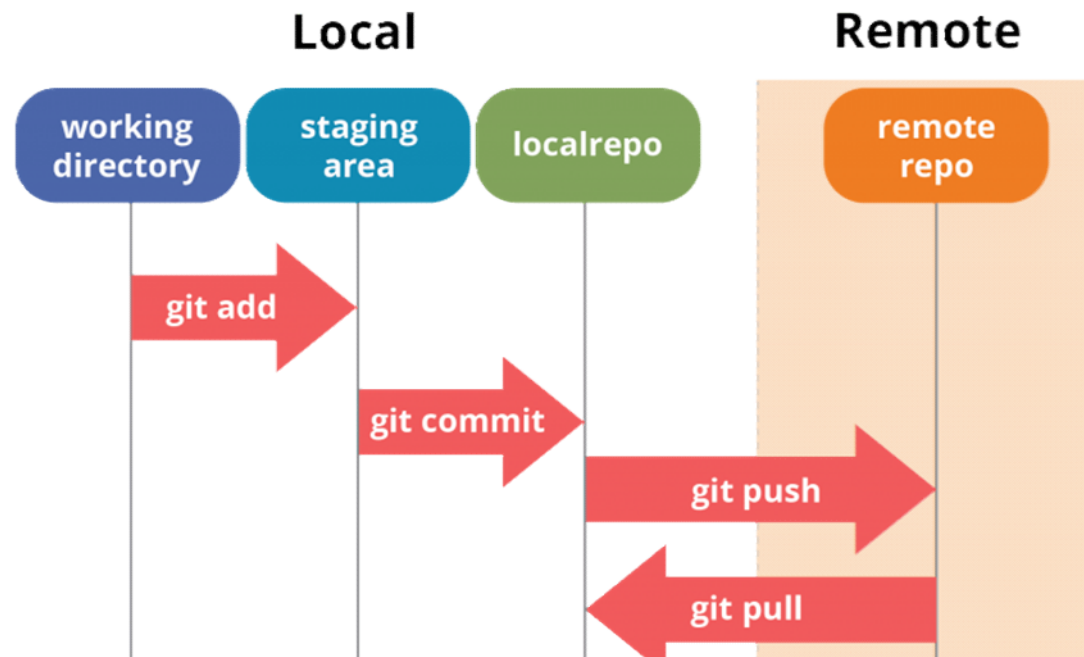


# git cheat sheet

Formador > Ezequiel Llarena Borges



WORK FLOW

<b>crear nuevo</b> repositorio	<code>git init</code>
<b>configurar</b>	<code>git config [--global] user.name &lt;nombre_usuario&gt;</code>
	<code>git config [--global] user.email &lt;email_usuario&gt;</code>
<b>mostrar configuración</b>	<code>git config -l   --list</code>

CREATE

ver <b>estado</b> del repo local	<code>git status</code>
<b>clonar</b> repositorio <b>local</b>	<code>git clone /path/to/repository</code>
<b>clonar</b> repositorio <b>remoto</b>	<code>git clone http[s]://host[:port]/path/to/repo.git/</code>
<b>clonar rama</b> repositorio <b>remoto</b>	<code>git clone -b   --branch &lt;branch_name&gt; &lt;repoURL&gt;</code>
<b>clonar</b> repo con <b>SSH</b>	<code>git clone ssh://[user@]host[:port]/path/to/repo.git/</code>
<b>clonar</b> repo con <b>Git SSH</b> protocol	<code>git clone git@hostname:username/reponame.git</code>

# CREATE & CLONE

<b>añadir</b> cambios a Staging	<code>git add &lt;filename&gt;</code>
<b>añadir todos</b> los cambios a Staging	<code>git add .</code>
<b>eliminar</b> archivos del Staging. <code>--cached</code> solo elimina del index/staging. Los ficheros siguen en el directorio.	<code>git rm --cached &lt;filename&gt;</code>

# ADD & REMOVE

<b>registrar cambios</b> add + commit	<code>git commit -m "Commit message/description"</code> <code>git commit -am "mensaje"</code>
<b>vincular</b> repo local con repo remoto	<code>git remote add origin &lt;url-repo&gt;</code>
<b>subir</b> cambios al repositorio <b>remoto</b>	<code>git push origin master</code>
<b>desvincular</b> repo local de repo remoto	<code>git remote remove origin</code>
<b>actualizar</b> repo local con los cambios repo remoto	<code>git pull</code>

# COMMIT & SYNCHRONIZE

<b>crear nueva rama</b>	<code>git checkout -b &lt;branch&gt;</code>
<b>cambiar a rama master</b>	<code>git checkout master</code>
<b>eliminar rama</b>	<code>git branch -d &lt;branch&gt;</code>
<b>subir rama al repositorio remoto</b>	<code>git push origin &lt;branch&gt;</code>

# BRANCHES

**fusionar** cambios desde otra rama

```
git merge <branch>
```

**revisar** cambios entre dos ramas

```
git diff <source_branch> <target_branch>
```

# MERGE



**crear etiqueta**

```
git tag <tag> <commit ID>
```

**obtener IDs** de los commits

```
git log
```

# TAGGING

shows the <b>committed history</b>	<code>git log [--online]</code> <code>git reflog</code>
<b>undo and create</b> new commits which contain an inverse of the specified commits changes*	<code>git revert &lt;hash-commit-to-undo&gt;</code>
<b>undo changes**</b>	<code>git reset -soft --mixed --hard &lt;hash-commit-to-undo&gt;</code>

# UNDOING CHANGES

**reemplazar** cambios locales

```
git checkout -- <filename>
```

**deshacer todos** los cambios locales y  
commits

```
git fetch origin  
git reset --hard origin/master
```

# RESTORE

# ENLACES Y RECURSOS

## **Clientes gráficos**

- SourceTree
- GitHub for Mac
- GitBox
- [Comparativa clientes gráficos git](#)

## **Guías**

- [Documentación Oficial Git](#)
- [Git, la guía sencilla](#)
- [Tutorial “Learn Git” de Bitbucket](#)
- [Git Community Book](#)
- [Think like a git](#)
- [GitHub Help](#)
- [A Visual Git Guide](#)