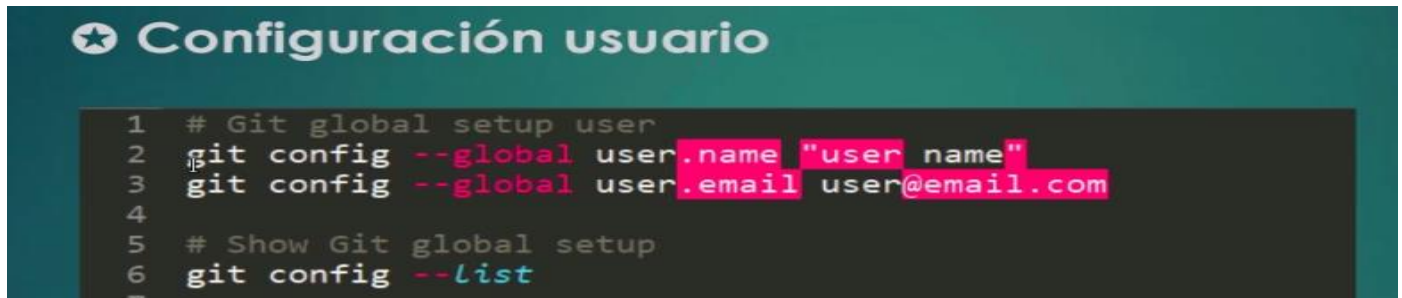


Muy bien el día de hoy lo que vamos a realizar es conocer algunos comandos GIT mas, para reforzar tus conocimientos. Recuerda si quieres ser un administrador de proyectos web, GIT es fundamental, practica siempre los comandos ya sea en una maquina o incluso en tu cuaderno, pero lo importante es recordar la secuencia de los comando y saber qué hacer.

Las **capturas** son de propiedad: **Juan Carlos Montoya. (odooerpcloud.com)**

Recordando los pasos:

PRIMERO: CONFIGURAR un USUARIO. Si vas a clonar un proyecto PÚBLICO no hay necesidad, pero si vas a ser colaborador y tienes que gestionar cambios tienes que realizarlo.



```
1 # Git global setup user
2 git config --global user.name "user name"
3 git config --global user.email user@email.com
4
5 # Show Git global setup
6 git config --list
```

SEGUNDO: aprendimos como usar nuestro primer comando GIT, pero conocer algunos parámetros. Que son importantes.



- ★ **Comandos imprescindibles**
- ★ **git clone**
Clonar, descargar un repositorio
- ★ **Parámetros**
 - * -b (indica el Branch a descargar)
 - * --depth (N.º commits a descargar)

```
25 git clone https://github.com/oca/project -b 8.0 --depth 10
```

TERCERO: ahora es ver siempre como se encuentra nuestro proyecto si hubo algún cambio o no?.



- ★ **git status**
 - * **Mostrar el estado actual del repositorio**
 - * **Indica los cambios o modificaciones de los ficheros del repositorio**

```
On branch 11.0
Your branch is up-to-date with 'origin/11.0'.
nothing to commit, working directory clean
```

CUARTO: Bien como es de suponerse también es necesario saber en que RAMA estamos trabajando. Para ello les presento a otro comando GIT.

★ **git branch**

Muestra la rama actual seleccionada

★ **Parámetros**

* **-a** (muestra ramas locales y remotas)

```
* 11.0
remotes/origin/11.0
remotes/origin/HEAD -> origin/11.0
```

QUINTO: bien ahora vamos a crear una rama, fuera de la que tenemos en el proyecto. (desechar cambios ó crear)

★ **git checkout**

Para crear una nueva rama (Branch) o para resetear los cambios de uno o varios ficheros (se puede hacer un reset desde local o desde remoto) ejemplo:

Reset cambios en el fichero main.py

Local:

git checkout ruta/main.py

Remoto:

git checkout origin/rama ruta/main.py

★ **Parámetros**

* **-b** (crear una nueva rama)

SEXTO: Aquí recordando un comando que ya vimos en la clase anterior y su usu.

★ **git add**

Agregar cambios actuales en el working directory

```
27 git add module_name|
```

SETIMO: Aquí vemos un comando el cual, cuando realizamos un cambios GIT nos recomienda ejecutarlos si queremos guarda los cambios efectuados.

★ git commit

Guardar los cambios actuales en local (Area/ Stage)

★ Parámetros

* -m (Mensaje o texto)

```
27 git commit -m "update values function get_area in module_name"
```

OCTAVO: Bien una vez que estemos seguros de ello lo único que nos queda por hacer es subir nuestros cambios a nuestro GITHUB Y GITLAB.

★ git push

Subir los cambios actuales desde local a remoto (remote)

```
29 git push origin branch_name
```

NOVENO: Bien es necesario saber cómo quedo, y commit's tenemos hasta el momentos fuera del default.

★ git log

Muestra el histórico de commits ordenamos del mas nuevo al mas antiguo

```
commit e958eafb6ef98ea4bb1bbf0d90a347a5c8cc867a
Author: Stéphane Bidoul (ACSONE) <stephane.bidoul@acsone.eu>
Date:   Fri Jun 15 23:33:34 2018 +0200

    remove obsolete .pot files [ci skip]

commit 9c09dbc6c35b5b6a07abf722e804f482f46edd29
Author: Jairo Llopis <yajo.sk8@gmail.com>
Date:   Mon Jun 11 12:07:51 2018 +0200

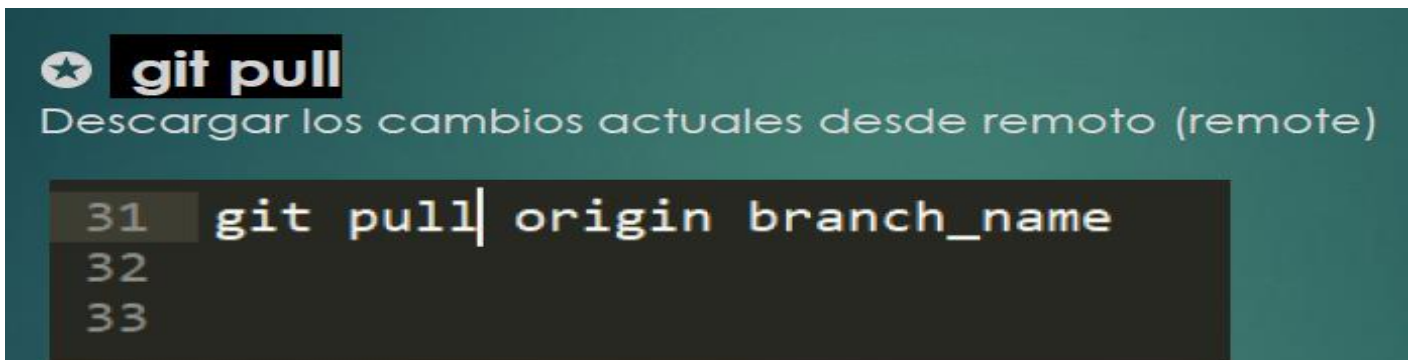
    [FIX] partner_firstname: Mixed content error, move to HTTPS

    Fix https://github.com/OCA/partner-contact/pull/487#issuecomment-396026511

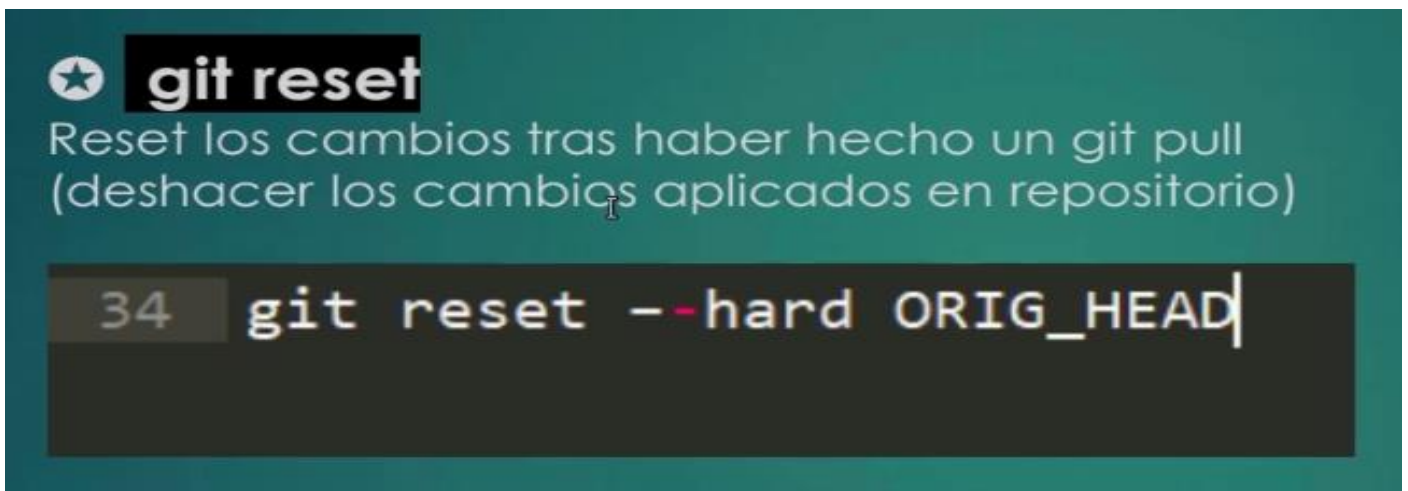
commit 60828335454b04b4b75e8e2723abd926212a3e4f
Author: OCA Git Bot <oca-git-bot@odoo-community.org>
Date:   Sat May 19 05:12:51 2018 +0200

    [ADD] setup.py
```


DECIMO: Ahora conocemos un nuevo comando de GIT, el cual nos permite recuperar los últimos cambios efectuados en nuestro proyecto. **(OJO ANTES DE REALIZAR ESTO, POR MI PARTE ES RECOMENDABLE SACAR UN BACKUP AL QUE YA TIENES EN CORRECTO FUNCIONAMIENTO, ESTO POR SI TIENES CORRIENDO PROYECTOS FUNCIONLES Y NO QUERRAS DAÑARLOS)**



ONCEAVO: Bien si ahora no, nos gusta el commit que realizamos, entonces podemos borrar el commit.



Solución si NO hemos subido el commit a nuestro repositorio remoto (no hemos realizado push):

1ª opción:

```
git reset --hard HEAD~1
```

--head: Con esta opción estamos indicando que retrocedemos a el comit HEAD~1 y perdemos todas las confirmaciones posteriores. HEAD~1 es un atajo para apuntar al commit anterior al que nos encontramos. CUIDADO, con la opcion – head, ya que como he dicho se borran todos los commits posteriores al commit al que indicamos.

2ª opción:

```
git reset --soft HEAD~1
```

--soft: con esta opción estamos indicando que retrocedemos a el commit HEAD~1 y no perdemos los cambios de los commits posteriores. Todos los cambios aparecerán como pendientes para realizar un commit.

Solución si hemos subido el commit a nuestro repositorio remoto (hemos realizado push):

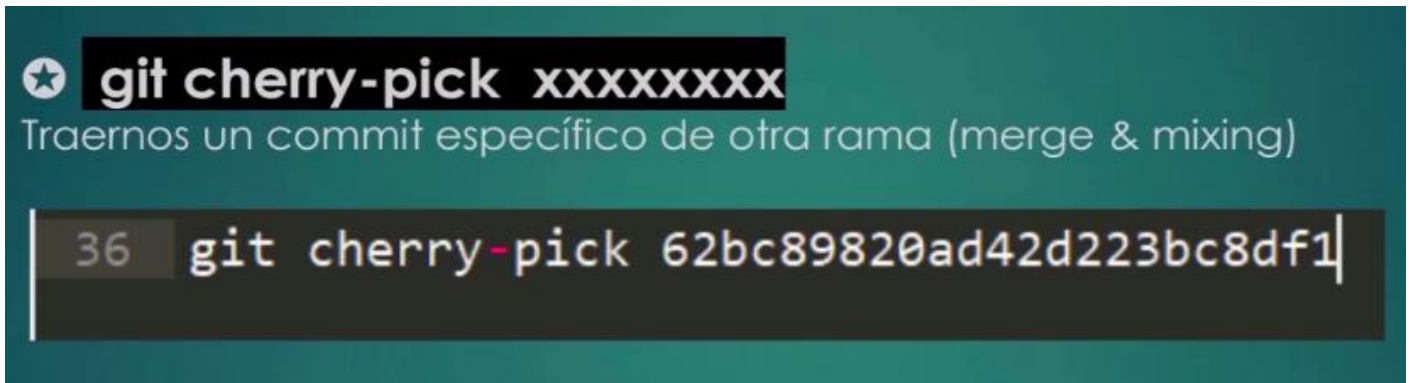
En caso de que queramos borrar un commit que ya hemos subido al servidor remoto, la mejor opcion es realizar un nuevo commit que borre el commit que queremos eliminar utilizando el comando **revert**. De esta forma cualquier usuario que se tenga actualizado el contenido del repositorio remoto puede obtener el cambio simplemente haciendo pull. Por lo tanto para borrar el ultimo commit teniendo en cuenta que el commit esta subido en un repositorio remoto debemos usar:

```
git revert HEAD
```

Espero que os sirva de ayuda en vuestro día a día con git.

<https://www.solucionex.com/blog/borrar-ultimo-commit-con-reset-y-revert-en-git>

DOCEAVO: Bien ahora si queremos traernos algo especifico como un commit. Ejecutamos lo siguiente.



TRECEAVO: Vamos a utilizar este comando GIT, para poder nivelar nuestras ramas.

GIT MERGE

Git merge 1.0