Soluciones: Ejercicios básicos de Git

1. *Configuración*

```
git config --global user.name 'Tu nombre'
git config --global user.email tuemail@alum.uca.es
git config --global core.editor emacs
```

2. *Creando un proyecto "Hello, world!"*

```
mkdir ProyectoWorld
cd ProyectoWorld
git init
git add hello.txt
git commit -m "Primer Commit"
git status
```

3. *Haciendo y añadiendo cambios al proyecto*

Tras modificar el fichero "hello.txt!

```
git status
git add hello.txt
git status
git commit -m "Cambios en hello.txt"
```

4. *Commits separados*

```
git add hola.txt
git add ciao.txt
git commit -m "Saludos latinos"
git status
git add bonjour.txt
git status
```

Tras la modificación del fichero "bonjour.txt"

```
git status
git add bonjour.txt
git status
git commit -m "Saludo frances"
git add hallo.txt
```

5. *Cambios en varias líneas*

Tras añadir las dos primeras líneas en el fichero.

```
git status
```

Tras añadir las siguientes dos líneas en el fichero y crear el nuevo fichero "hei.txt".

```
git diff
git add .
git status
git commit -m "Más saludos europeos"
```

6. *El historial*

```
git log --all --pretty=oneline --since='7 days ago'
git log --all --pretty=oneline --since='7 days ago'
--author=tunombre
```

7. *Opciones del historial*

```
git log --all --pretty=oneline --grep='Saludo'
```

8. *Formatos en el historial*

```
git log --pretty=format:"%h %ad | %s%d [%an]" --graph --
date=short
```

9. *Alias*

```
[alias]
  ci = commit
  st = status
  hist = log --pretty=format:\"%h %ad | %s%d [%an]\" --graph --
date=short
```

10. *Alias del shell*

No hay diferencia.

11. *Etiquetando*

```
git tag -a v1.0.0 -m "Creando la primera version oficial"
git show v1.0.0
```

12. *Volviendo al estado original*

```
git hist
git checkout <hash>
```

Tras ver el contenido del fichero y del directorio de trabajo.

```
git checkout master
git checkout <hash>
```

13. *Etiquetando una version anterior*

```
git tag -a v1.0.0-beta -m "Creando la primera version beta"
git checkout master
git tag
git hist
```

14. *Nuevo fichero para la siguiente versión*

```
git describe --tags
git add .
git ci –m "Fichero hej.txt"
git describe --tags
```

Significa que estamos un commit adelantados de la última etiqueta. Nos sirve para controlar cuánto hemos trabajado desde la última etiqueta, ya que se va incrementando el número que aparece por cada commit que realizamos.

15. *Etiquetando desde master*

```
git tag -a v0.0.0 -m "Versión de prueba" <hash_del_1er_commit>
```

16. *¿Quién hizo un commit de esto?*

```
git blame hallo.txt
```

17. *Escondiendo los cambios*

Tras guardar los cambios realizados en el fichero "hola.txt".

```
git st
git stash save "Añadiendo saludos en hola.txt
git st
```

Tras guardar los cambios realizados en el fichero "hello.txt".

```
git stash save "Añadiendo saludos en hello.txt
git stash list
```

Tras guardar los cambios realizados en el fichero "bonjour.txt", añadirlos al repositorio y guardarlos en el repositorio.

```
Git stash apply stash@{Num}
*Num = número del segundo cambio guardado
```

Tras guardar los cambios realizados en el fichero "hello.txt".

```
git add hello.txt
git ci –m "Añadiendo más saludos en hello.txt"
Git stash apply stash@{Num}
```

Tras guardar los cambios realizados en el fichero "hola.txt".

```
git add hola.txt
git ci –m "Añadiendo más saludos en hola.txt"
git stash clear
```

18. *Deshaciendo cambios locales*

Tras guardar los cambios realizados en el fichero "ciao.txt".

```
git st
git checkout ciao.txt
git st
```

19. *Deshaciendo cambios antes del commit*

Tras guardar los cambios realizados en el fichero "ciao.txt".

```
git add ciao.txt
git st
git reset HEAD ciao.txt
git checkout ciao.txt
git st
```

20. *Deshaciendo cambios después del commit*

Tras guardar los cambios realizados en el fichero "ciao.txt".

```
git add ciao.txt
git ci –m "Modificaciones en ciao.txt"
git revert HEAD
```

Tras guardar los cambios realizados en el fichero emergente.

```
git hist
git st
```

21. *Eliminar ficheros del área de preparación*

```
echo "El fichero de prueba a borrar" > erase.txt
git add .
git rm –f erase.txt
git status / ls (para el directorio de trabajo)
rm erase.txt (en el directorio de trabajo)
git rm erase.txt (para el área de preparación)
git status / ls (para el directorio de trabajo)
```

22. *Eliminar y renombrar de ficheros*

```
git rm --cached hóla.txt
ls (para comprobar que el fichero sigue en el directorio de trabajo)
mv hóla.txt óla.txt
git add óla.txt
git commit –m "Saludo portugués"
```

23. *Moviendo ficheros*

```
mkdir España
git mv hola.txt España
git status / ls (para el directorio de trabajo)
git add España/hola.txt
git commit –m "Directorio España"
```

Como resultado el fichero hola.txt ha sido borrado del directorio de trabajo "ProyectoWorld". Y se ha creado un nuevo fichero hola.txt dentro del directorio "España".

Tras crear el fichero kaixo.txt

```
git add kaixo.txt
```

```
git commit –m “Saludo euskera”
mv kaixo España
git add España/kaixo.txt
git rm kaixo.txt
git commit –m “kaixo.txt en España”
```

24. *Eliminando etiquetas*

```
git tag -a v1.0.1 -m "Idiomas castellanos"
git help tag
git tag –d v1.0.1
git hist
```

Tras crear el fichero holaCat.txt dentro de la carpeta España

```
git add España/holaCat.txt
git commit –m “saludo catalán”
git tag -a v1.0.1 -m "Idiomas castellanos"
```

25. *Arreglando commits*

Tras crear el fichero halo.txt

```
git add halo.txt
git commit –m “saludo”
Tras añadir otra frase en el fichero halo.txt
git add halo.txt
git help commit
git commit ––amend –m “saludo croata”
git hist
```

26. *Referencia a objetos (diferencias)*

```
git diff HEAD HEAD^^
git diff master~4 HEAD
```

27. *Referencia a objetos (visualización)*

```
git show ––stat @{Fecha.semana_pasada}
git show master~3
git show master@{Fecha_hace.6días} | master@{6.days}
```

28. *Referencia a objetos (historial)*

```
git log -7
git log ––since=”Mes día” ––until=”Mes día”
git log hello.txt
```

29. *Búsquedas*

```
git grep (-n –H) mundo (segunda pregunta)
```

30. *Visualización del repositorio*

```
git instaweb
```