Capacitación Git
Módulo 2

Repositorios

por Gonzalo Barro Gil, fuente https://git-scm.com/book/es/v2

https://github.com/GonzaloBarroGil/git-course

Crear repositorio

- Los repositorios se inicializan en la carpeta que va a contener (o ya contiene) los archivos y carpetas que Git va a gestionar

```
cd /home/user/repos
cd repositorio
git init
```

- Un repositorio puede clonarse desde otro existente en una carpeta raíz ya que la carpeta contenedora será creada por Git

```
cd /home/user2/projects
git clone /home/user/repos/repositorio
cd /home/user2/projects/repositorio
```

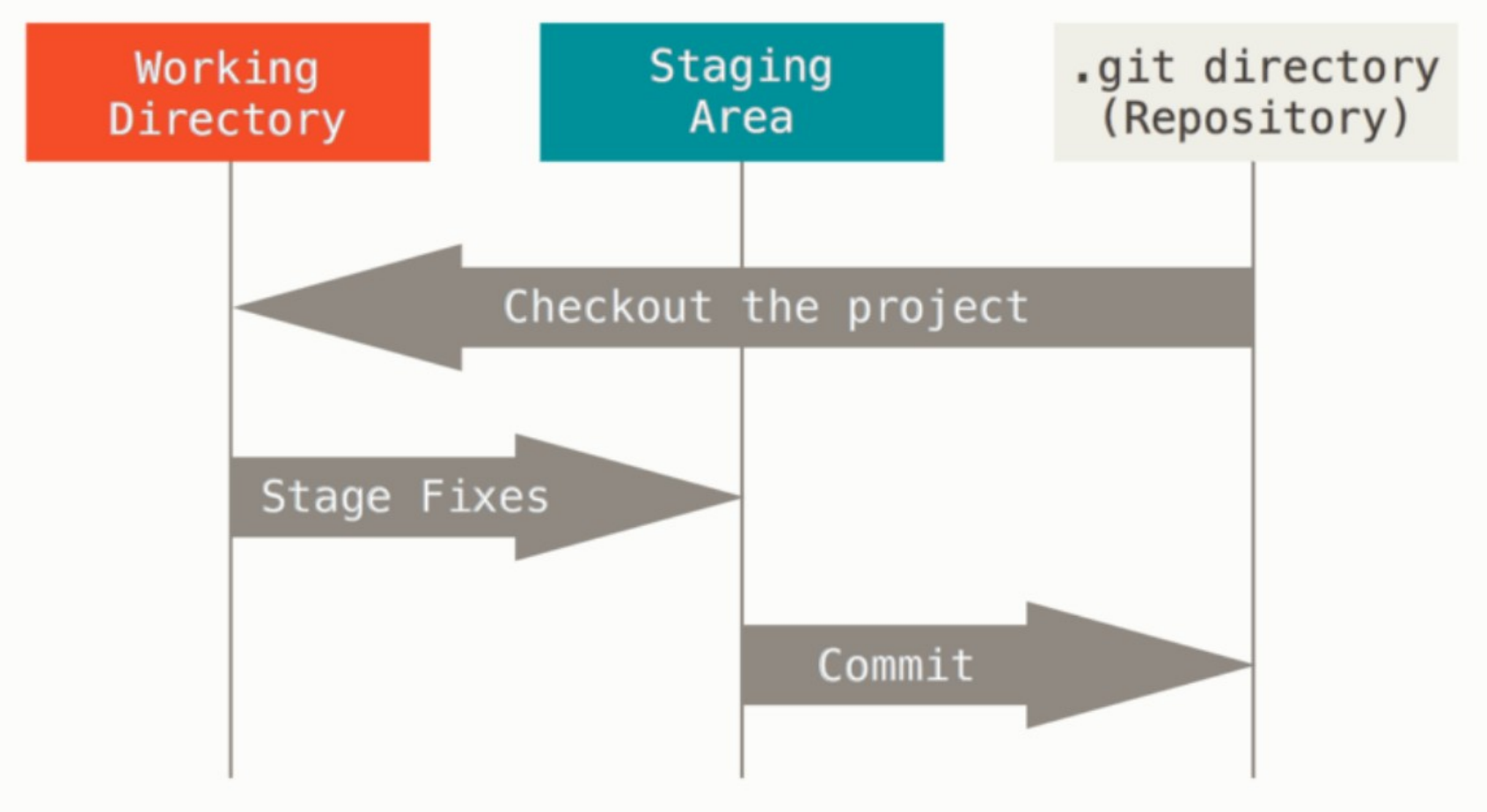
Trabajar con archivos

- Un repositorio puede ser inicializado vacio o con archivos y carpetas existentes

- A partir de ese momento se pueden realizar las siguientes operaciones:

  - Crear, modificar o eliminar archivos y subcarpetas
  - Seleccionar uno, algunos o todos los elementos para ser confirmados
  - Confirmarlos como versión

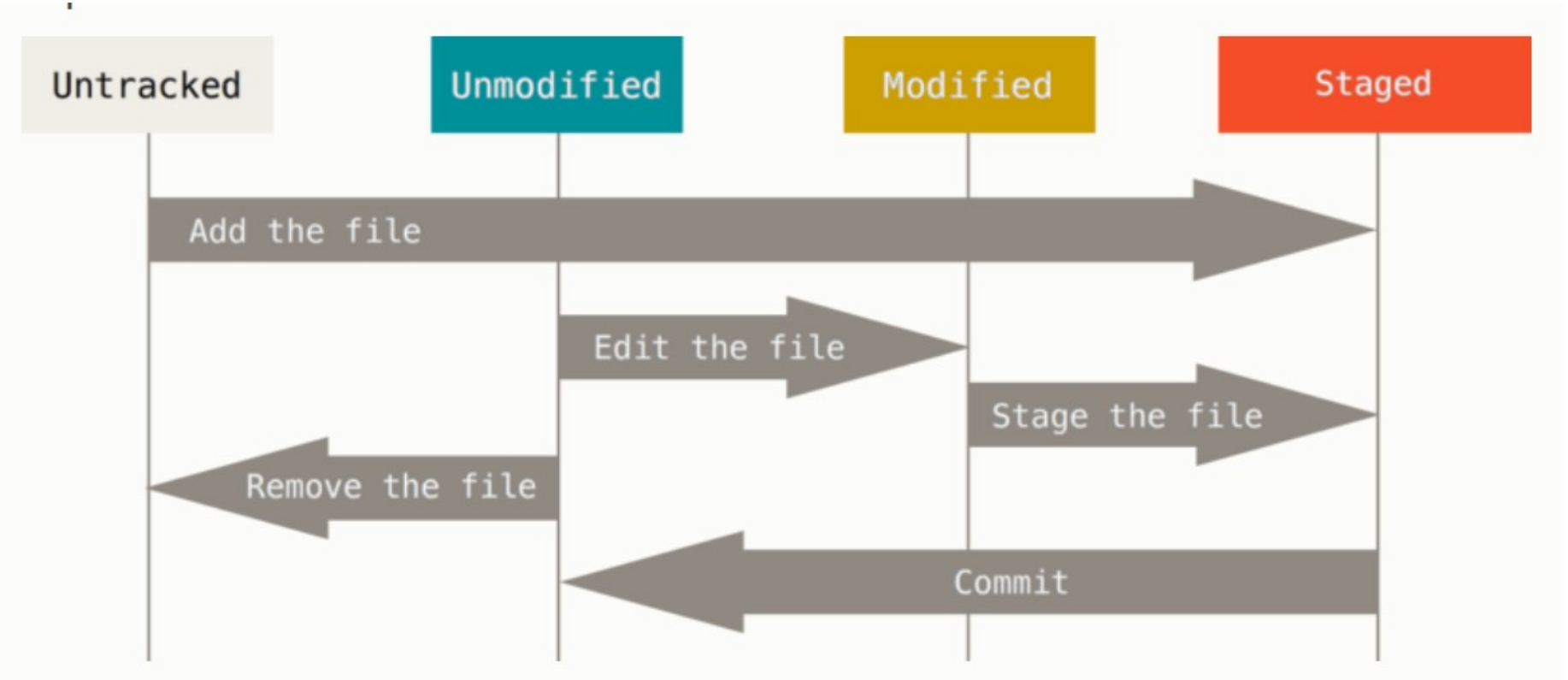
Áreas


Working Directory
Staging Area
.git directory (Repository)
Checkout the project
Stage Fixes
Commit

Áreas

- Directorio Git: repositorio/.git
  - Carpeta oculta donde entre otras cosas, se guardarán las copias de las versiones confirmadas
- Directorio de trabajo: repositorio
  - Carpeta contenedora donde el sistema de archivos expondrá los archivos con los que trabajaremos
- Área de preparación:
  - Archivo dónde se registran los elementos a ser incluidos en la próxima confirmación

Estados


Untracked
Unmodified
Modified
Staged
Add the file
Edit the file
Stage the file
Remove the file
Commit

Estados

- **Sin seguimiento de cambios**: Archivos nuevos o eliminados.
  - **No rastreados** (Untracked): Cambios sobre archivos nuevos o eliminaciones de archivos exitentes desde la última confirmación o creación del repo.
    - Agregar (git add): Pasan a ser **preparados**
- **Con seguimiento de cambios**: Archivos existentes desde la última confirmación.
  - **Preparados** (Staged): Cambios (creación, edición o eliminación) que fueron incluidos para la próxima confirmación.
    - Confirmar (git commit): Pasan a ser **no modificados**
  - **No modificados** (Unmodified): Estado de archivos sobre los que no hubo cambios desde la última confirmación.
    - Modificar: Pasan a ser **modificados**
    - Eliminar: Pasan a ser **sin trackear** (git rm --cached), o **preparados** (git rm)
  - **Modificados**: Cambios (sólo edición) efectuados después de la última confirmación.
    - Agregar (git add): Pasan a ser **preparados**

Verificar estados

- Estos estados se pueden verificar con

```
git status
```

- Como primera medida nos informará en qué branch estamos. Por ejemplo:

```
On branch master
...
```

- Seguidamente nos agrupará por estado todos los cambios:
  - No rastreados:

```
Untracked files:
...
```

  - Preparados:

```
Changes to be committed:
...
```

  - Modificados:

```
Changes not staged for commit:
...
```

  - No modificados:

```
nothing to commit, working directory clean
```

Estado inicial

- Todos los archivos (si el repositorio no está vacío) son **no modificados**:

```
> git status

On branch master
nothing to commit, working directory clean
```

Crear archivos

- Al crear un archivo que no existía previamente en el repositorio, pasa a ser **no rastreado**:

```
> git status

On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

  nuevoarchivo.jpg

nothing added to commit but untracked files present (use "git add" to
track)
```

- Si se elimina el archivo desde el file system el directorio de trabajo volverá a estar limpio y se volverá al estado anterior (**no modificado**)
- Si se edita el archivo el estado no cambia ya que sigue siendo **no rastreado**.

Agregar archivos

- Al agregar un archivo **no rastreado** al área de preparación (stage o index), pasa a ser **preparado**:

```
> git add nuevoarchivo.jpg
> git status

On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

  new file:    nuevoarchivo.jpg
```

- Si se edita o elimina el archivo habrá dos copias del archivo, una como **modificado** (o **eliminado**) y otra como se ve arriba (**preparado**), por lo que las únicas acciones recomendadas en este punto son las sugeridas volver al estado anterior (git reset HEAD) o confirmar los cambios (git commit).

Editar archivos del repo

- Al modificar o eliminar desde el FS un archivo existente desde la última confirmación, clonado o inicialización pasa a ser **modificado** (o **eliminado**):

```
> git status

On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working
  directory)

  deleted:    index.html
  modified:    home.html
```

- Si se apunta el archivo a la última confirmación (git checkout --) vuelve a **no modificado**.
- Si se prepara el archivo (git add) pasa a ser **preparado**.
- Cualquier confirmación directa (git commit) que se haga sin preparar el archivo no lo incluirá en la misma.

Verificar Cambios

- Además de verificar el estado (git status), se pueden consultar las modificaciones específicas de un archivo respecto a la versión confirmada en el repo:
  - Archivos modificados:

    ```
    > git diff
    ```

  - Archivos preparados:

    ```
    > git diff --cached
    ```

- Esto mostrará las diferencias línea por línea entre ambas versiones (siempre que el archivo contenga texto plano como texto o source code).

Confirmar Cambios

- Teniendo cambios agregados al stage, se puede aplicarlos a través de una confirmación:

```
> git commit
```

  - Tras lo que nos solicitará incluir un mensaje descriptivo, por ejemplo:

```
"add title to html."
```

- Una vez completado el commit el estado pasará a ser **no modificado**.

```
> git status

On branch master
nothing to commit, working directory clean
```

Excluir del repositorio

- Hay archivos y carpetas sobre los que, por distintos motivos no deben ser considerados por Git como parte del repositorio:
  - Archivos o carpetas con datos de configuración local
  - Archivos o carpetas con datos de autenticación
  - Archivos o carpetas temporales
  - Otros …

- Archivo **.gitignore**:

```
# comentarios
carpetaexcluida/
archivoexcluido
carpeta/archivoexcluido
carpeta/carpetaexcluida/
!noexcluir

# cualquier cadena de carácteres:
*

# cualquier carácter:
?

# cualquiera de estos caracteres:
[añb]

# cualquier carácter entre estos:
[a-z]
```

Excluir archivos existentes

- Si se agregan a .gitignore archivos que ya estaban incluidos en el directorio de Git no serán excluidos si además no lo elimino de la caché:

```
git rm -r --cached carpeta/
git rm --cached archivo
```

- Tras lo cual se debe preparar y efectuar una nueva confirmación:

```
git add .
git commit -m "apply gitignore for committed files"
```

Eliminar o renombrar

- Si se elimina o renombra un archivo desde el file system Git incluirá tanto el archivo eliminado como el antecedor del renombrado como eliminaciones dentro del grupo no preparado para el commit, y mostrará al renombrado como un archivo sin seguimiento:

```
rm archivoaeliminar
mv archivoarenombrar nuevoarchivo
```

- Pero si la elminación se hace a través de Git incluirá la eliminación y el cambio de nombre en el stage:

```
git rm archivoaeliminar
git mv archivoarenombrar nuevoarchivo
```

Clonar un repositorio

- Recordemos que Git es un VCS **distribuido** enfocado en el trabajo **colaborativo**

```
git clone https://github.com/GonzaloBarroGil/git-course
```