Version 1.0 13/08/2014

Configuraciones Iniciales

Cuentas en bitbucket.org

Como solo podemos contar con 5 cuentas, las dividi por area de trabajo, por lo cual tendríamos las siguientes cuentas:

User: implementador1
User: implementador2
User: testing
User: demas-usuarios
User: snieves (Administrador del Repositorio, capote también va a tener esta cuenta)

---Descargar Repositorio

git clone https://snieves@bitbucket.org/snieves/pis-2014.git
| ---> nombre del usuario, en mi caso snieves

---Configurar datos del usuario

git config --global user.name “nombre”
(En nombre va su nombre real, no tiene que ver con el usuario de bitbucket)
git config –global user.email “<email>”

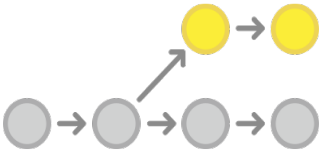
---Entorno de Trabajo

Directorio de Trabajo **(**Es donde realizamos los cambios locales)

Stagin index **(**Area de trabajo temporal)

- Para agregar, un archivo al Stagin index, debemos usar **git add nombreArchivo**, luego para ver que archivos estan listos para ser agregados en el proximo commit podemos usar el comando **git status**.
- Para realizar los cambios en el repositorio se debe realizar un **git commit**.
  Cada commit debe tener un mensaje chico y resumido del objetivo del commit o algun dato que guie que se cambio, los mensajes tienen un formato determinado, por lo cual para simplificar se podria realizar de esta manera, **git commit -m “mensaje”**
- Para borrar un archivo del stagin index, se debe utilizar **git rm nombreArchivo**, luego se deberia realizar un commit para que efectivamente borre el archivo, tambien puede realizarse como **git rm * -r** (El asterisco significa todos los archivos de ese directorio y el -r que sea recursivo a otros directorios dentro del mismo).
- Esto se realiza simplemente moviendo el archivo con los comando del SO “mv”, el git detectara que moviste el archivo (se borro el actual y se creo uno igual en una carpeta que no esta incluida), por lo cual se debera realizar un add de dicha carpeta.
- Para descartar el commit de un archivo se debe realizar
  git reset HEAD nombreArchivo
- Si queremos descartar todos los cambios y dejar el archivo como estaba anteriormente debemos realizar **git checkout -- nombreArchivo**
- Para ver el historial de commit se puede utilizar el comando **git log**

Ramas (Punto Importante)



- Como primer punto destacamos que la rama **master** es la linea base del proyecto, todo cambio realizado en dicha rama debera ser tomado con cuidado.
  La idea de una rama es que a partir de cierto punto de la rama master o cualquier otra, se ramifique una nueva rama creando una nueva copia del repositorio en ese punto y en paralelo, con el objetivo de realizar trabajaos independientes de los demas y donde deseamos que los commit de otros no nos afecten. A posterior se podria realizar un marge de las ramas y descartarla o simplemente descartarla.

- Para crear una rama, se ejecuta el siguiente comando, **git branch nombreRama**
- Para saber en que rama estoy parado, se utiliza el comando **git branch**, la rama marcada con un asterisco es la rama actual.
- Para cambiar de rama se ejecuta el siguiente comando **git checkout nombreRama**
- Para fucionar ramas utilizams el comando **git merge nombreRama**, por ejemplo si tenemos la rama master y una rama B, nos paramaos en la rama master (**checkout**) y aplicamos el comando **git merge B**, de esto pueden salir conflictos, git hace el merge de manera inteligente por lo cual existen estas 3 posibilidades:
  1- Haces un merge de una rama en otra sin haber tocado los mismos archivos, entonces el merge se integra bien.
  2- Se toco el mismo archivo en ambas ramas pero editando partes separadas, bloques dijuntos de codigo, entonces el merge se integra bien.
  3- Si modificaste el mismo archivo en ambas ramas cambiando las mismas regiones de codigo, entonces hay un conflicto. (cuando sucede git agrega marcas especiales)
- Para b borrar una rama **git branch -d ramaBorrar**

- Para volver a un commit anterior simplemente ejecutamos **git log** y luego **git checkout numeroCommit.**
- Los tags, sirben para enmascarar commits de forma mas entendible, esto se realiza con el comando **git tag nombreTag numeroDeCommit**
- Con **git tag** podremos ver todos los tags importantes.

---Repositorios Remotos

- **git push repositorio nombreRama** (Envia nuestra rama al repo remoto)
  Ejemplo: git push https://snieves@bitbucket.org/snieves/pis-2014.git master
  Nota: hay que destacar que el snieves antes del arroba es el nombre de usuario el cual de debera cambiar al que corresponda
- **git pull repositorio nombreRama** (Descarga los commits de otros en el repositorio)