Argentina programa

Instituto Nacional de Tecnología Industrial
INTI

Ministerio de Desarrollo Productivo
Argentina

#YoPrograMo

(Programador Full Stack Web Jr.)

Ministerio de Desarrollo Productivo Argentina

Instituto Nacional de Tecnología Industrial
INTI

Argentina programa

Argentina Programa

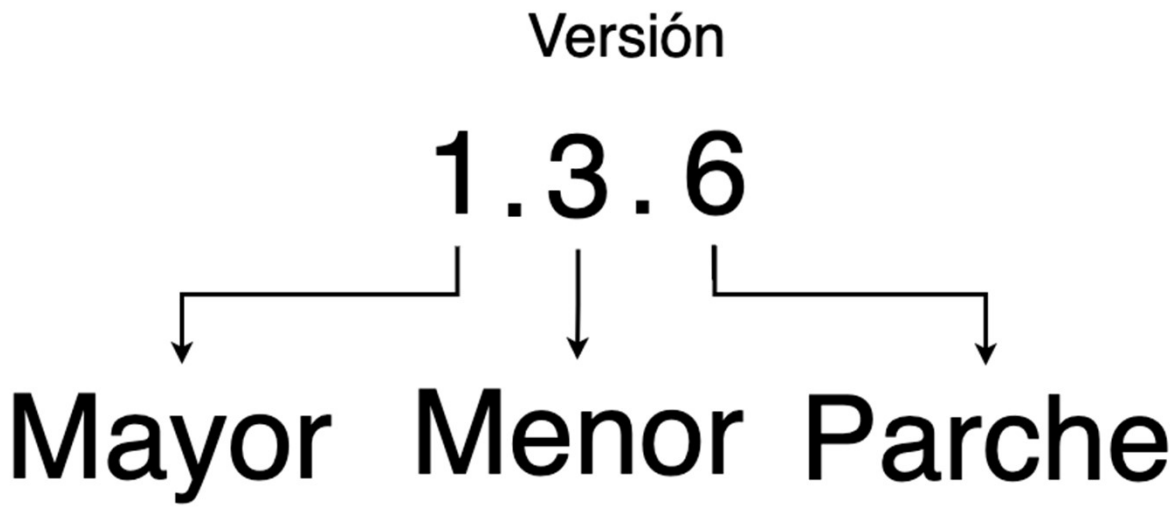
Módulo 1 - Introducción a Desarrollo Web y Aplicaciones

Guión de teoría

¿Qué veremos de teoría ?

- Que es una versión de software
- Conceptos de Git (repositorio local) y Github (repositorio remoto)
- Te guiaremos para la instalación
- Cómo funciona internamente Git
- Comandos básicos
- Crear un Repositorio local
- Creación de un proyecto en repositorio local
- Creación de ramas
- Crearemos un repositorio remoto
- Subir los cambios del Git (repositorio local) hacia Github (repositorio remoto)

¿Cómo leer una versión?



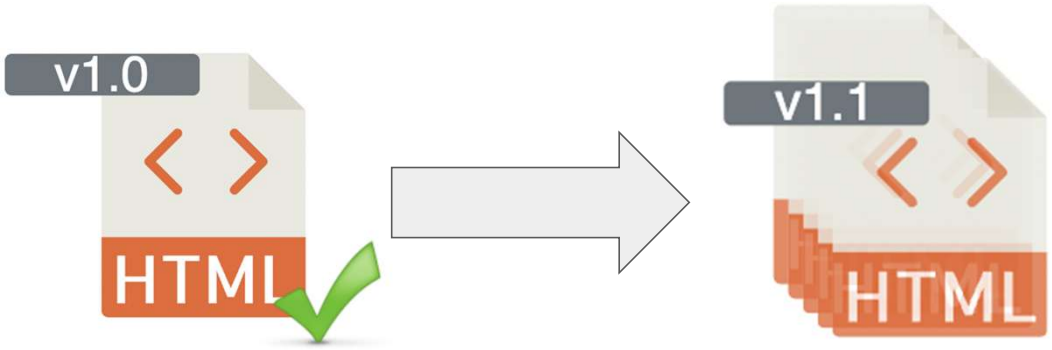
- 1er valor - **Mayor**: Incrementa cuando realizas un cambio incompatible en el API.
- 2do valor - **Menor**: Incrementa cuando añades funcionalidad que es compatible con versiones anteriores.
- 3er valor - **Parche**: Incrementa cuando añades cambios referidos a correcciones y mantienen la compatibilidad con la versión anterior.

https://semver.org/lang/es/

https://gitversion.net/docs/

Control de versiones

- Hacer un cambio en el código

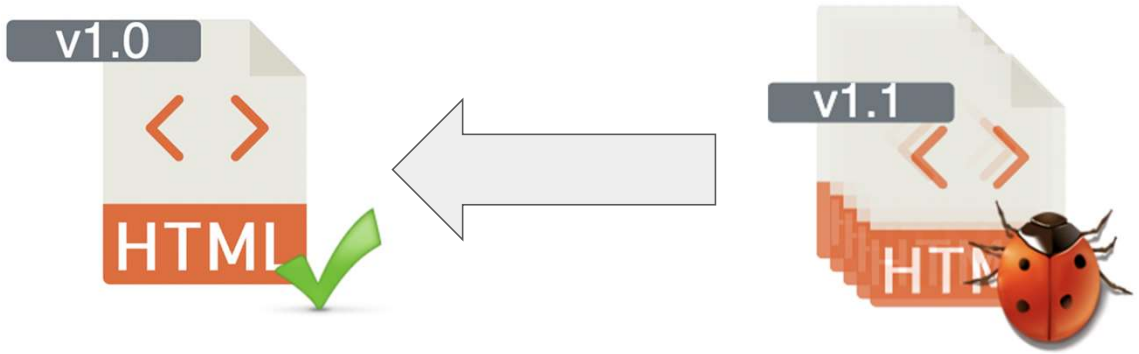


Control de versiones

- Se introdujo un error o bug en el código

Control de versiones

- Volver a la versión anterior o rollback



Argentina Programa

Introducción Git

¿Cómo usamos Git?

- Historial de cambios
- Identificar cuando se introdujo un error o bug
- Recuperar versiones anteriores
- Etiquetar cambios
- Almacenar los archivos de código de nuestro proyecto
- Trabajar en equipo y de forma descentralizada

Argentina Programa

Git - Introducción

¿Cómo compartimos código?

1. **Instalamos Git en la PC**

Repositorio Local

2. **Creamos una cuenta en**

Repositorio Remoto

Argentina Programa

Git - Introducción

Instalamos Git en la PC

Repositorio Local

Argentina Programa

Github - Repositorio Remoto

Creemos una cuenta

Argentina Programa

Git - Repositorio Local

¿Cómo trabajamos con Git?

git + **Comandos**

Repositorio Local

Consolas

Otras opciones

Argentina Programa
Git - Repositorio Local

¿Cómo trabajamos con Git?

- Comandos básicos

GIT CONFIG

`git config --global user.name <name>`	Define the author name to be used for all commits by the current user.
`git config --global user.email <email>`	Define the author email to be used for all commits by the current user.
`git config --global alias. <alias-name> <git-command>`	Create shortcut for a Git command. E.g. `alias.glog "log --graph --oneline"` will set `"git glog"` equivalent to `"git log --graph --oneline.`
`git config --system core.editor <editor>`	Set text editor used by commands for all users on the machine. `<editor>` arg should be the command that launches the desired editor (e.g., vi).
`git config --global --edit`	Open the global configuration file in a text editor for manual editing.

GIT BASICS

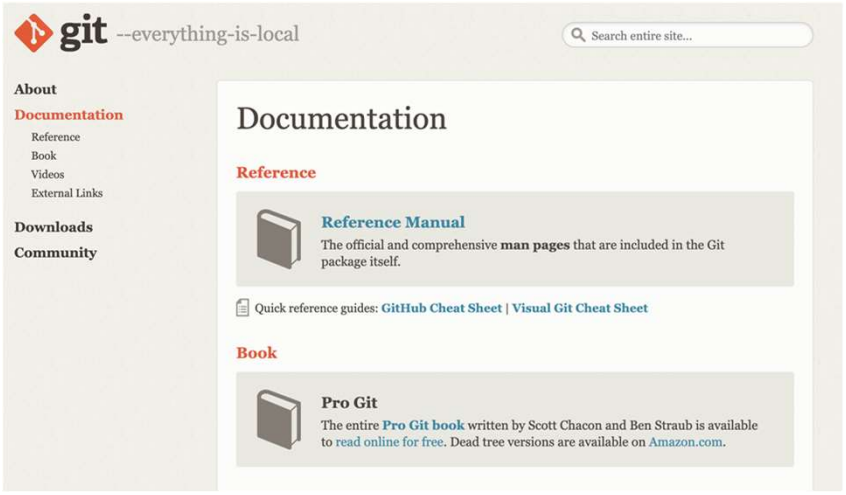
`git init <directory>`	Create empty Git repo in specified directory. Run with no arguments to initialize the current directory as a git repository.
`git clone <repo>`	Clone repo located at `<repo>` onto local machine. Original repo can be located on the local filesystem or on a remote machine via HTTP or SSH.
`git config user.name <name>`	Define author name to be used for all commits in current repo. Devs commonly use `--global` flag to set config options for current user.
`git add <directory>`	Stage all changes in `<directory>` for the next commit. Replace `<directory>` with a `<file>` to change a specific file.
`git commit -m "<message>"`	Commit the staged snapshot, but instead of launching a text editor, use `<message>` as the commit message.
`git status`	List which files are staged, unstaged, and untracked.
`git log`	Display the entire commit history using the default format. For customization see additional options.
`git diff`	Show unstaged changes between your index and working directory.

https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet

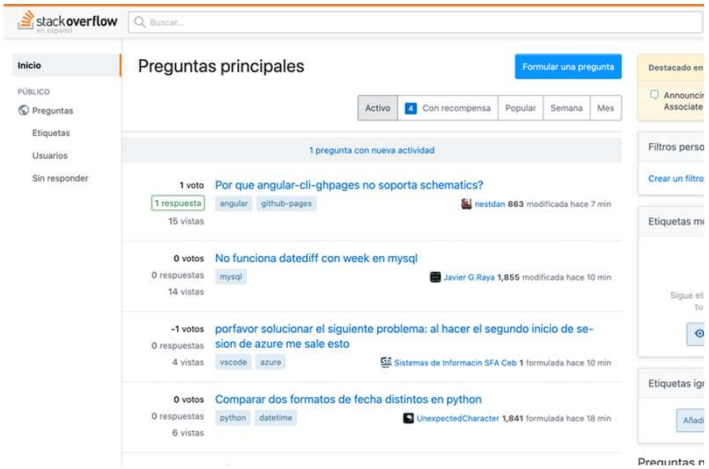
Argentina Programa
Git - Comandos

¿Dónde profundizar más?

https://git-scm.com/doc



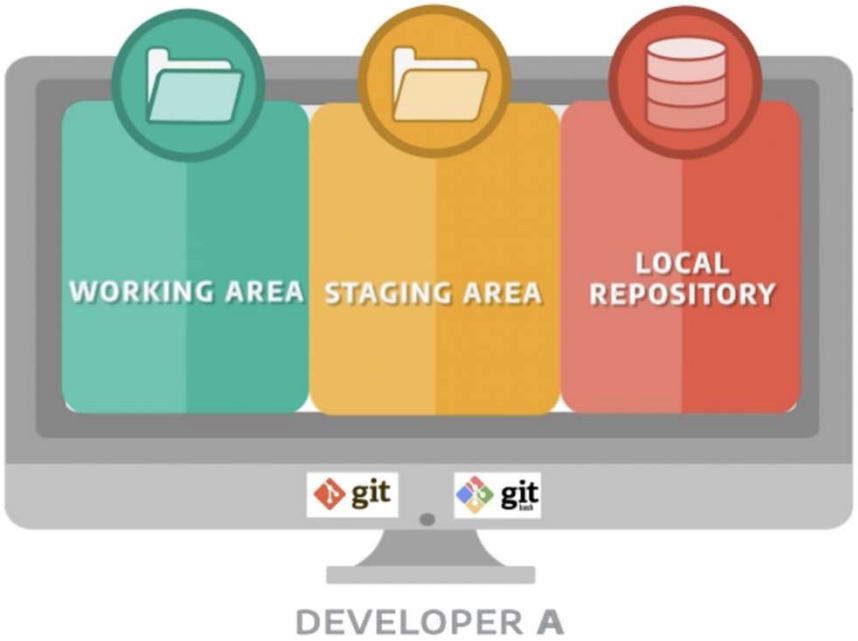
https://es.stackoverflow.com/



Argentina Programa

Git - Repositorio Local

¿Cómo es por dentro un Repositorio Local?



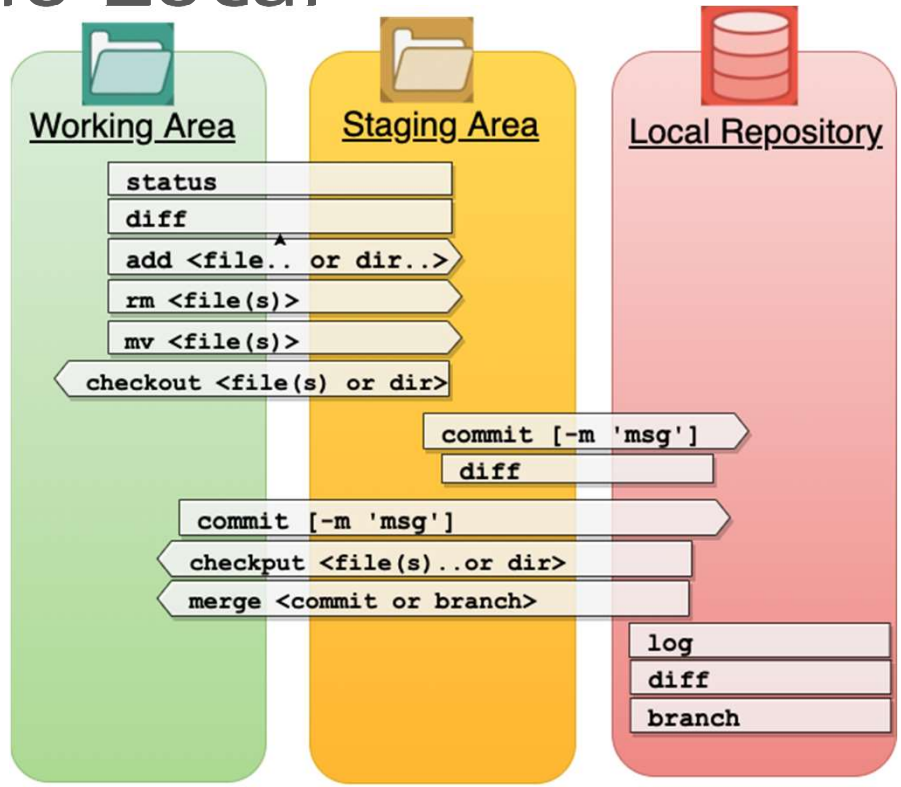
- **Working Area o Espacio de trabajo:** que puedes ver en el catálogo de ordenadores.
- **Staging Area o Área Temporal:** Inglés llamado staging, o index. Generalmente se almacena en el archivo de índice del "directorio git" (.git / index), por lo que a veces lo llamamos índice de área temporal (índice).
- **Local repository o Repositorio local:** Este espacio de trabajo tiene un directorio oculto .git, este no es el área de trabajo, sino el repositorio de Git.

Argentina Programa

Git - Repositorio Local

Comandos en el Repositorio Local

Consola de comandos



Argentina Programa

Github - Repositorio Remoto

¿Cómo se trabaja de forma colaborativa?

Repositorio Local

en tu equipo

+

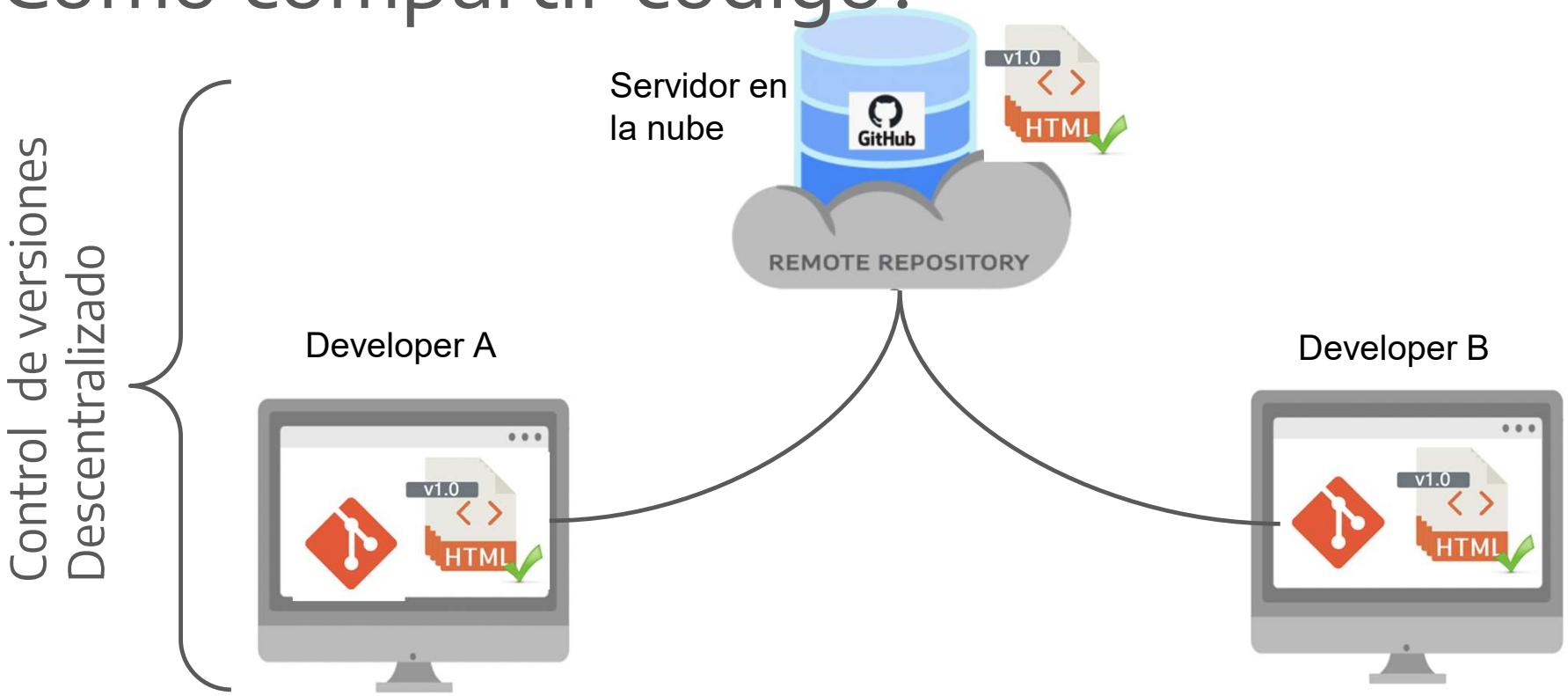
Repositorio Remoto

en la nube

Argentina Programa

Github - Repositorio Local y Remoto

¿Cómo compartir código?



Argentina Programa
Github - Repositorio Local y Remoto

¿Cómo trabajamos con Github?

- Comandos básicos

REMOTE REPOSITORIES

`git remote add <name> <url>`	Create a new connection to a remote repo. After adding a remote, you can use `<name>` as a shortcut for `<url>` in other commands.
`git fetch <remote> <branch>`	Fetches a specific `<branch>`, from the repo. Leave off `<branch>` to fetch all remote refs.
`git pull <remote>`	Fetch the specified remote's copy of current branch and immediately merge it into the local copy.
`git push <remote> <branch>`	Push the branch to `<remote>`, along with necessary commits and objects. Creates named branch in the remote repo if it doesn't exist.

REWRITING GIT HISTORY

`git commit --amend`	Replace the last commit with the staged changes and last commit combined. Use with nothing staged to edit the last commit's message.
`git rebase <base>`	Rebase the current branch onto `<base>`. `<base>` can be a commit ID, branch name, a tag, or a relative reference to `HEAD`.
`git reflog`	Show a log of changes to the local repository's `HEAD`. Add `--relative-date` flag to show date info or `--all` to show all refs.

GIT BRANCHES

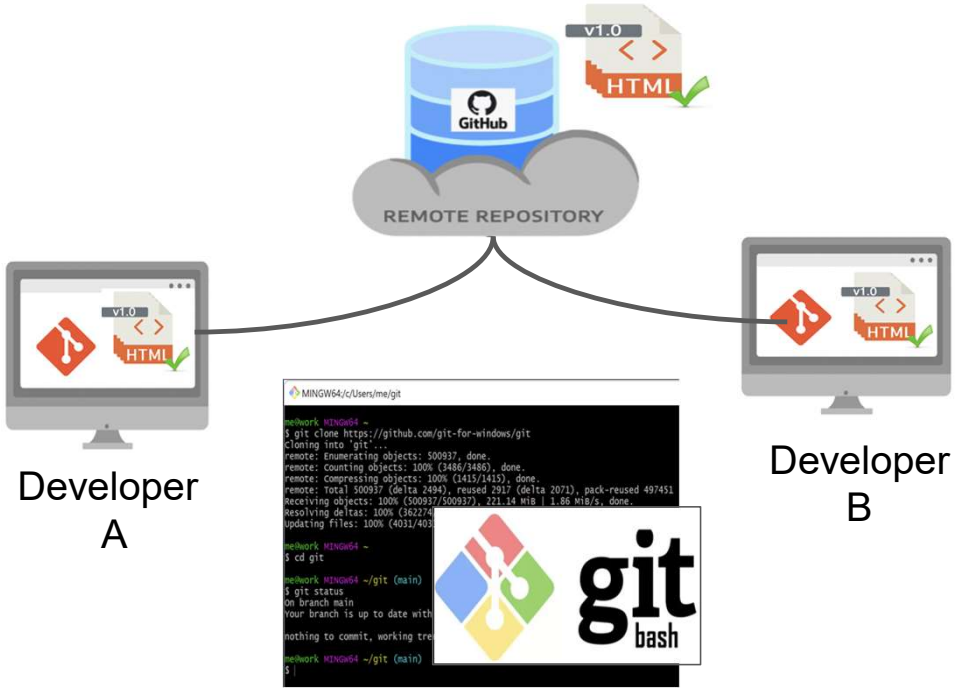
`git branch`	List all of the branches in your repo. Add a `<branch>` argument to create a new branch with the name `<branch>`.
`git checkout -b <branch>`	Create and check out a new branch named `<branch>`. Drop the `-b` flag to checkout an existing branch.
`git merge <branch>`	Merge `<branch>` into the current branch.

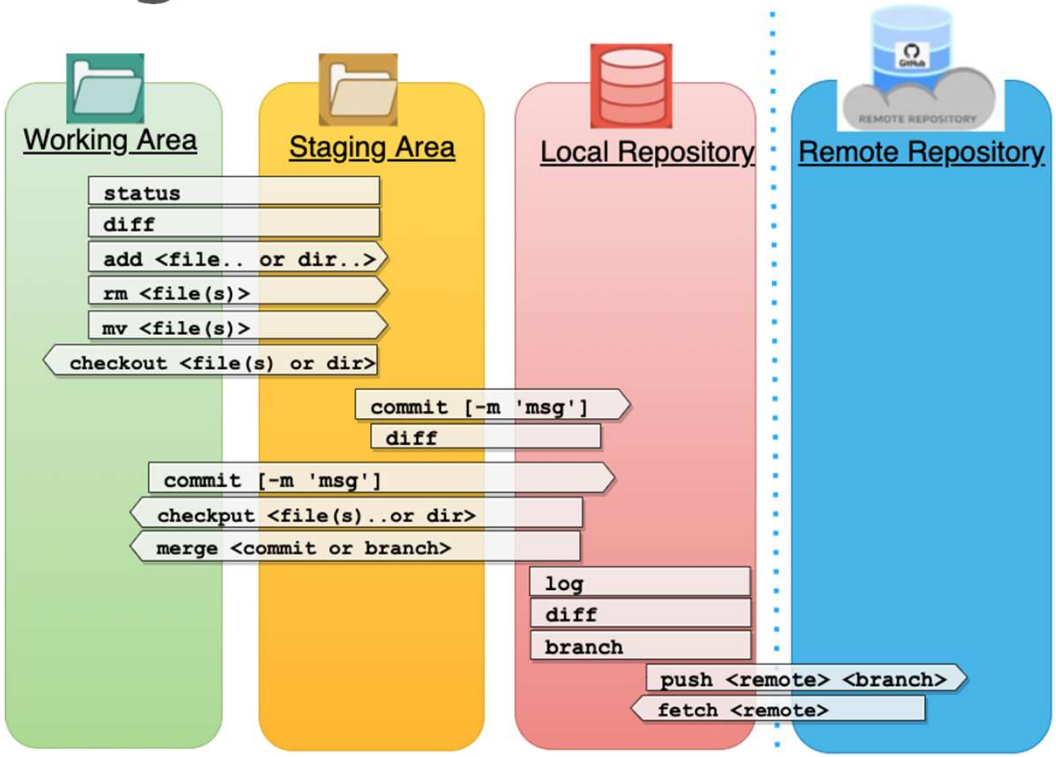
https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet

Argentina Programa

Github - Repositorio Local y Remoto

¿Cómo se comparte el código?





Argentina Programa

Github - Repositorio Remoto

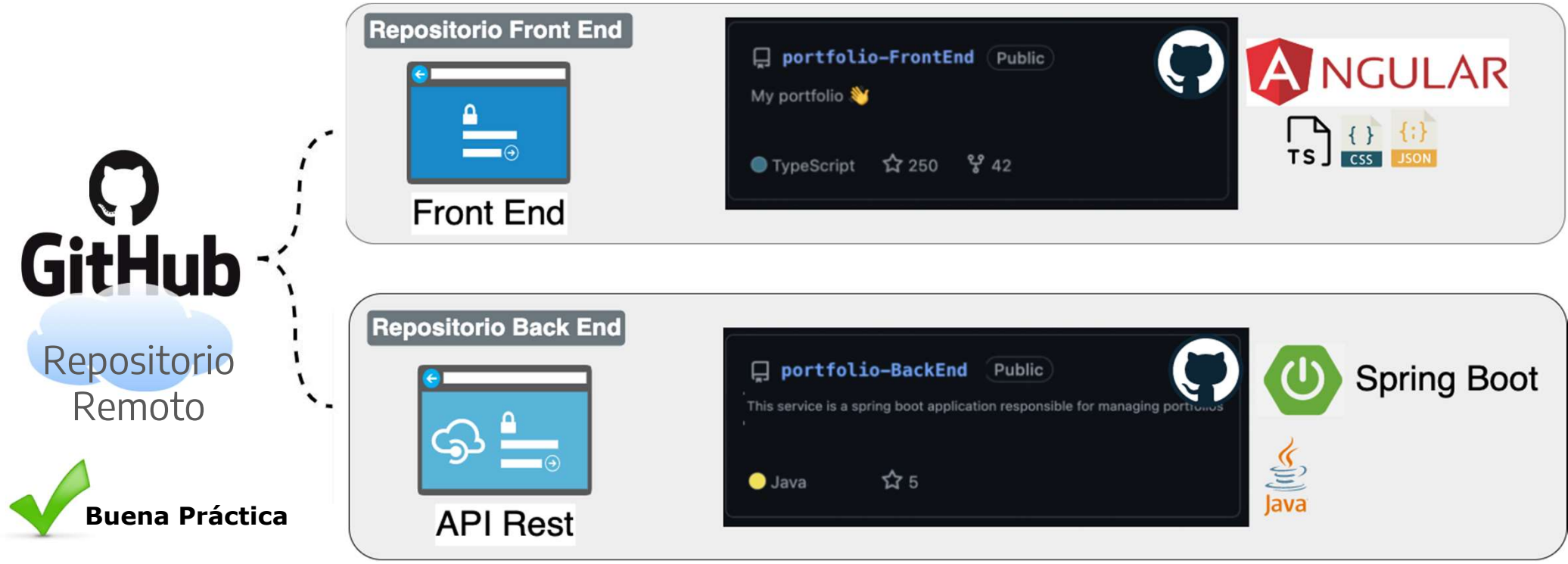
¿Cómo crear un Repositorio Remoto?

1. **Tener una cuenta:** Contar con una cuenta en la web de Github con un correo de forma gratuita creamos una cuenta.
2. **Crear un repositorio remoto:** Desde la web de Github podemos crear uno o varios repositorios dependiendo de cómo organicemos nuestro proyecto.
3. **Crear el access token:** Desde la web de Github debemos crear el token para que nos permita conectarnos desde el Repositorio Local
4. **Vincular el Repositorio Local y Remoto:** Debemos seguir los pasos que nos indica la web de Github, ejecutando los comandos que nos permiten configurar y vincular el repositorio remoto

Argentina Programa

Github - Repositorio Remoto

¿Cómo organizar el repositorio?



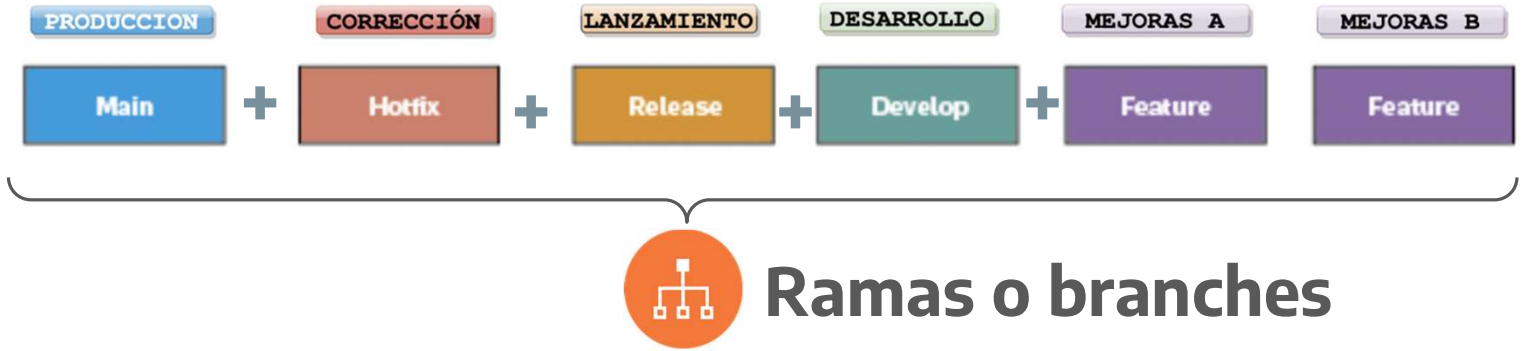
Buena Práctica

Argentina Programa

Git y Github - Flujo de trabajo Gitflow

¿Cómo organizar el flujo de trabajo?

Gitflow =



¿Qué es?

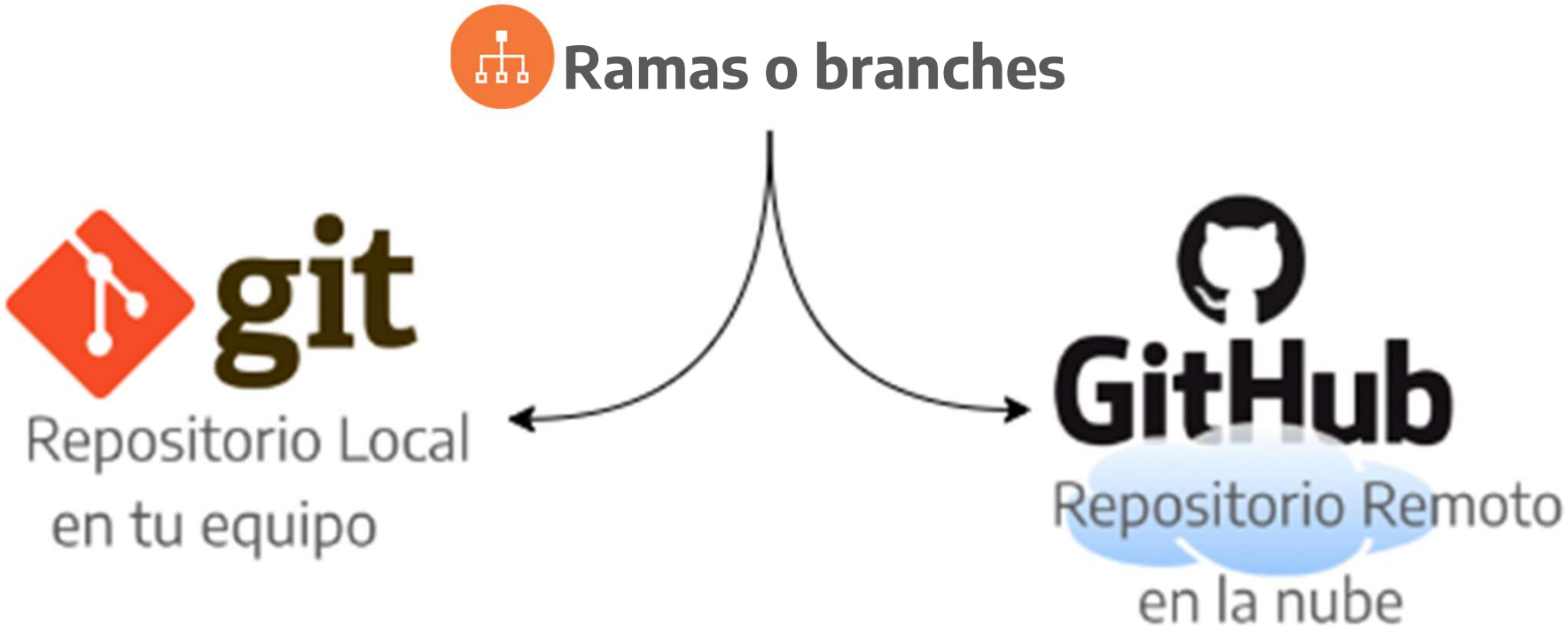
- **Modelo de ramificación:** basado en ramas que propone una forma ordenada para el desarrollo y mantenimiento de un proyecto en constante cambio.
- **Flujo de trabajo:** propone un criterio en la utilización de ramas para que los equipos de un proyecto de desarrollo cuente con un flujo de trabajo.

- **Desarrollo paralelo**
- **Colaboración**
- **Separación de código basado en ramas**

https://git-scm.com/download/win

Git y Github - Flujo de trabajo Gitflow

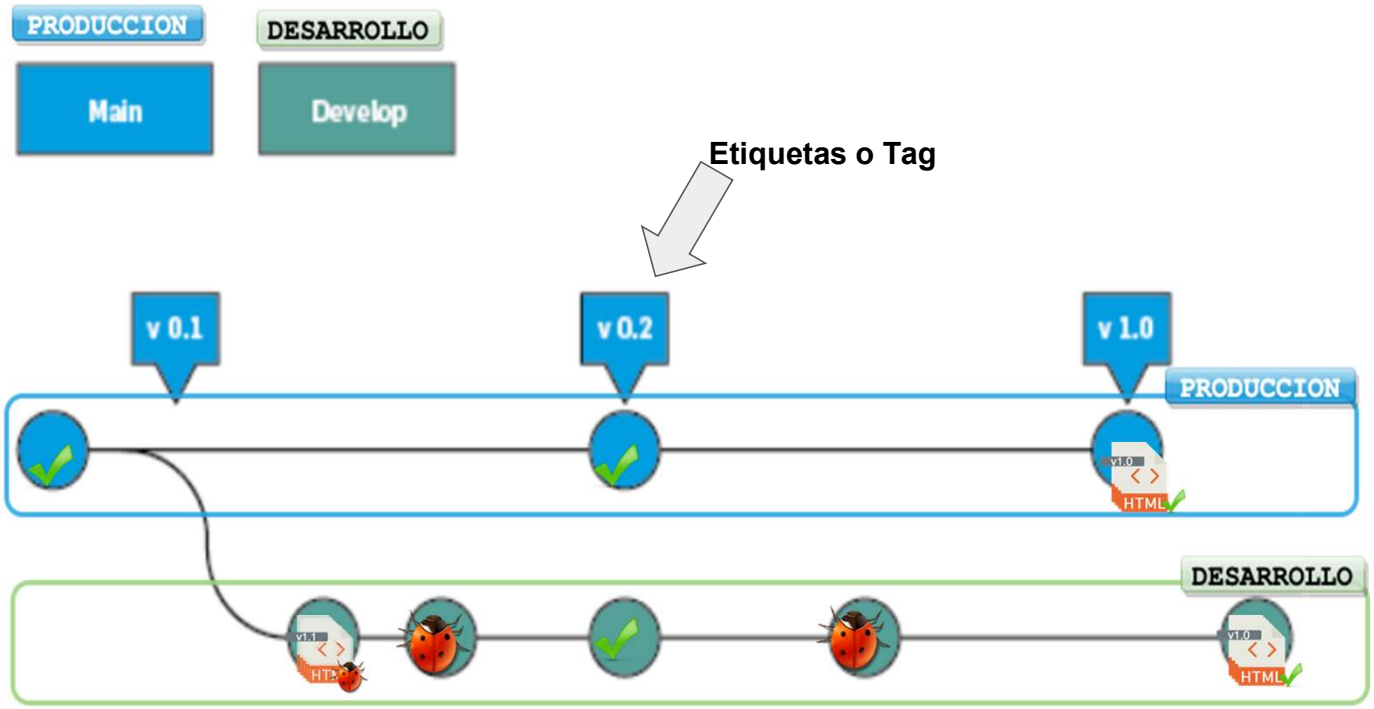
¿Dónde podemos usar las ramas?



Argentina Programa

Git y Github - Flujo de trabajo Gitflow

¿Qué son las ramas o branches?

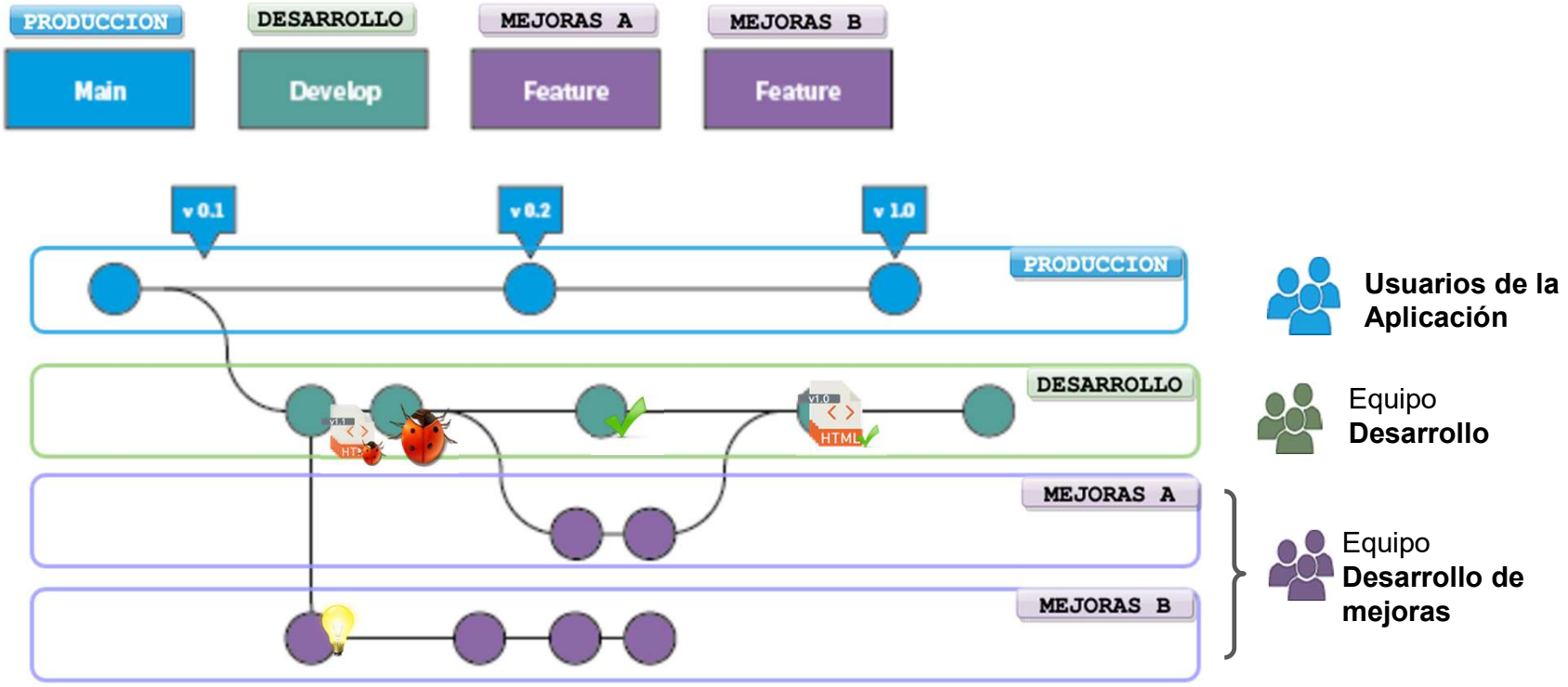


- **Las ramas en Git:** Son espacios independientes que el desarrollador puede crear para no afectar el código de la rama principal (**Master** o **Main**), teniendo el código funcionando y en la rama **Develop** se utiliza para hacer cambios sin afectar la rama main (principal). Los programadores pueden crear las ramas que necesite o puede adoptar metodologías de trabajo como Gitflow.
- **Rama main o master:** Esta rama es la principal.
- **Rama Develop:** Esta rama es un espacio paralelo y separado para que los desarrolladores trabajen en agregar nuevas funciones o corregir un error.

Argentina Programa

Git y Github - Flujo de trabajo Gitflow

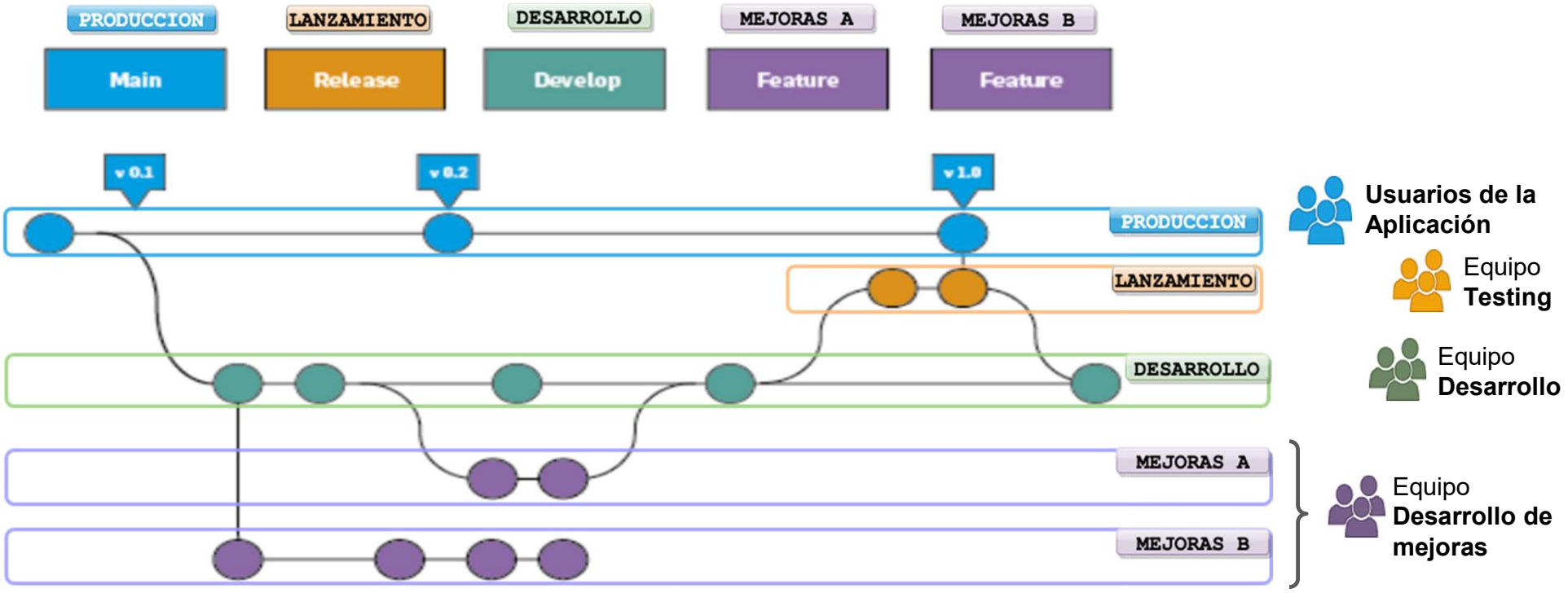
¿Por qué usar Ramas o branches?



Argentina Programa

Git y Github - Flujo de trabajo Gitflow

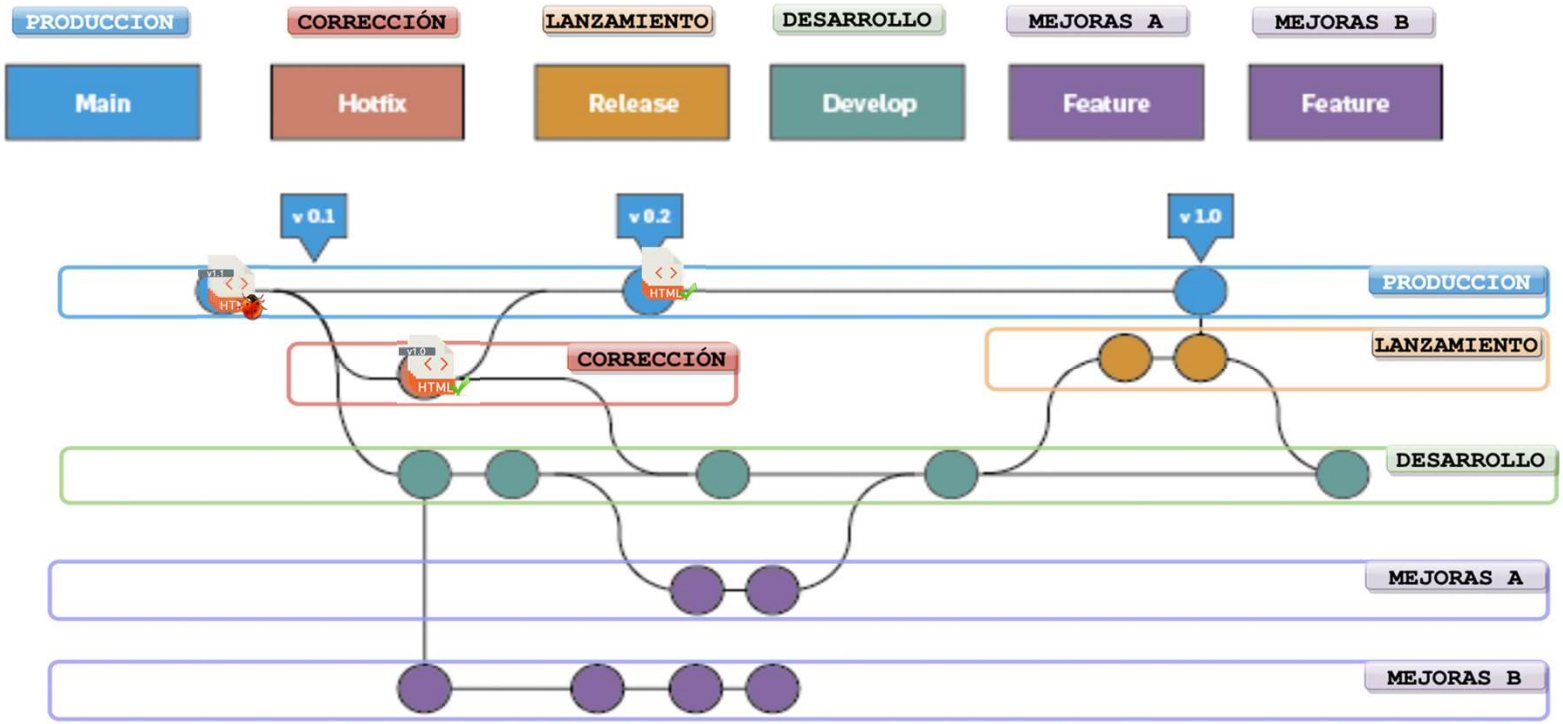
Rama release o Lanzamiento



Argentina Programa

Git y Github - Ramas o branches

¿Rama hotfix o Corrección?



Argentina Programa

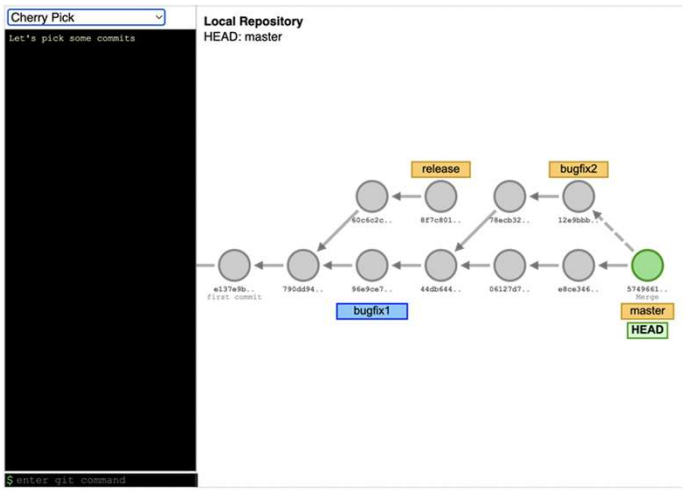
Git y Github - Prácticas

Te dejamos algunas web para practicar

- Comandos de Git

https://learngitbranching.js.org/?locale=es_AR&demo=

http://git-school.github.io/visualizing-git/#cherry-pick



Muchas gracias.

Ministerio de Desarrollo Productivo Argentina

Instituto Nacional de Tecnología Industrial
INTI

Argentina programa