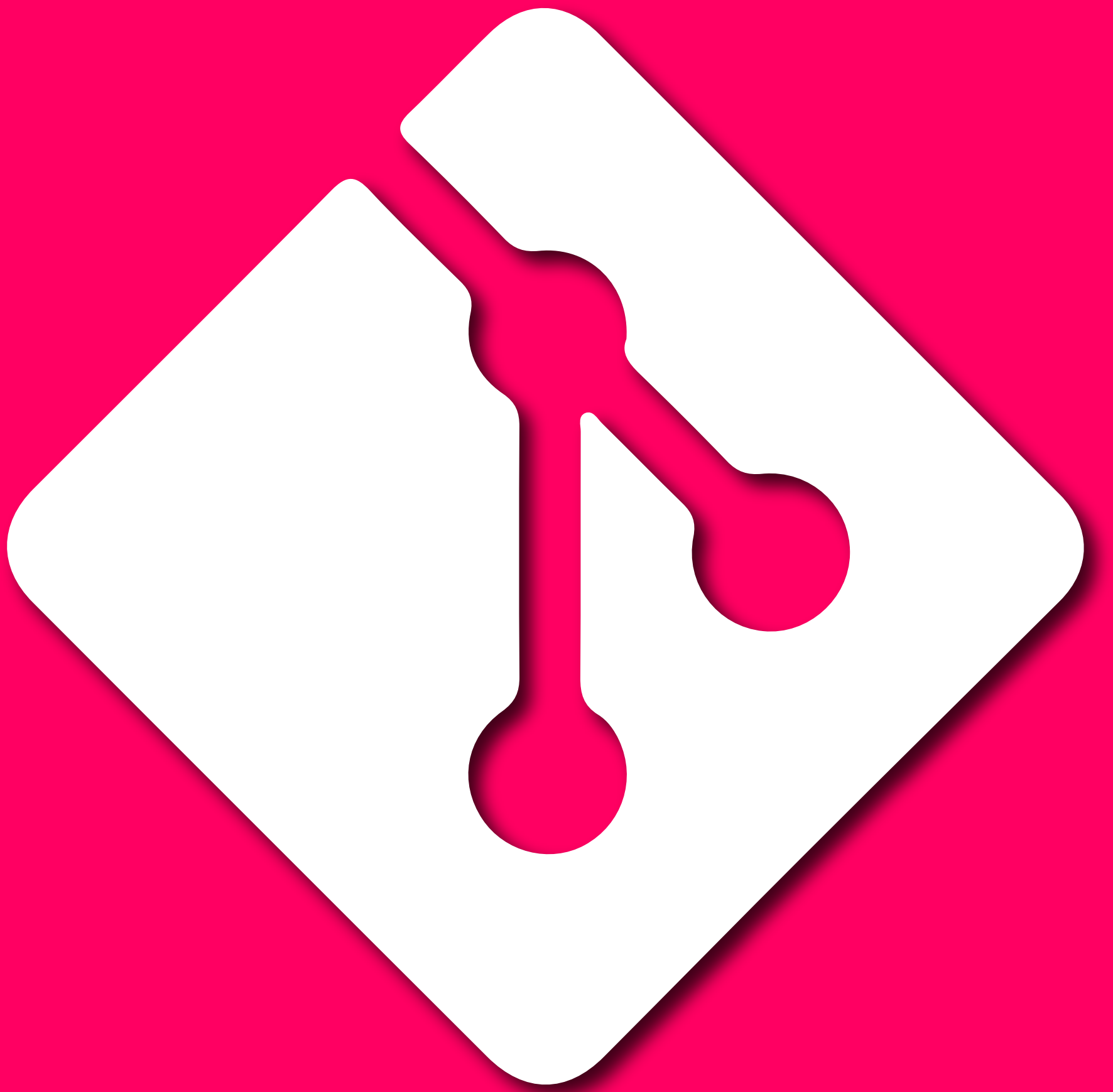
El futuro digital es de todos

MinTIC



Introducción a Git

Universidad de Caldas

Hola

Git nos permite realizar la gestión de versiones de nuestro código, de igual manera nos puede salvar el día cuando cometemos errores, cuando nuestro computador se bloquee o el disco duro se dañe.
¿Cómo generan ustedes las copias de seguridad de sus archivos? ¿O acaso no hacen copias de seguridad? Posiblemente la mayoría de ustedes no hacen copias de seguridad, y si lo hacen, puede ser de manera manual. Un método de control de versiones, usado por muchas personas, que es copiar los archivos a otra carpeta, tal vez agregando la fecha y la hora en que lo hicieron. Este método es muy común porque es muy sencillo, pero también es propenso a errores. Es fácil olvidar en qué directorio se encuentran los archivos, y por accidente guardar archivos nuevos o sobrescribir archivos importantes.

Entonces ¿Qué es un control de versiones?, ¿por qué debería ser tenido en cuenta por un desarrollador de software? Un control de versiones es un sistema que registra los cambios realizados en un archivo o conjunto de archivos a lo largo del tiempo, de modo que se pueda recuperar versiones específicas posteriormente. Aunque no solo se controlan archivos con código fuente, en realidad se puede hacer lo mismo con casi cualquier tipo de archivo que se encuentre en una computadora.

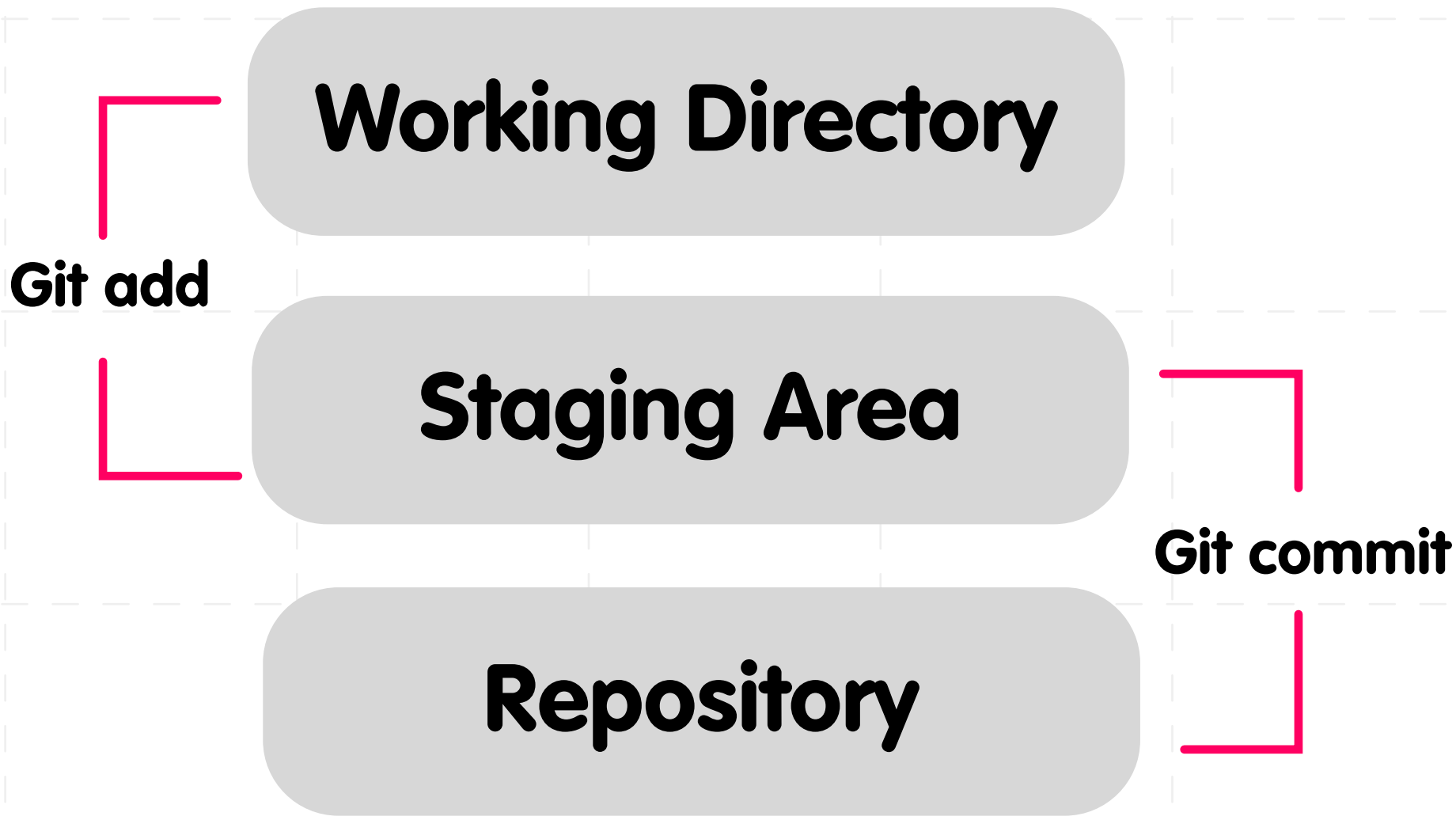
¿Qué pasa con Git? Desde su nacimiento en el 2005 ha evolucionado y madurado para ser fácil de usar y conservar sus características iniciales basadas en trabajo en equipo y versionamiento instantáneo de archivos. Es tremendamente rápido, muy eficiente con grandes proyectos y tiene un incremento en el sistema de ramificación para desarrollo no lineal, es decir, me permite trabajar en equipo dividiendo tareas a varios desarrolladores en un mismo proyecto.

Git se basa en tres estados principales de un archivo: confirmado *(committed)*, modificado *(modified)*, y preparado *(staged)*.

Confirmado: significa que los datos están almacenados de manera segura en la base de datos local de Git.
Modificado: significa que se ha modificado el archivo pero todavía no se ha confirmado en la base de datos.
Preparado: significa que se ha marcado un archivo modificado en su versión actual para que vaya en la próxima confirmación.

Estos tres (3) estados son la base para definir las tres (3) secciones principales de trabajo en un versionamiento con Git. La primera sección se centra en el directorio de trabajo local, donde continuamente estamos escribiendo código, agregando archivos o eliminándolos.

El segundo estado se llama *Staging area* (área de preparación), encargada de almacenar los cambios que se agregan a medida que se escribe nuevo código. Finalmente se tiene el directorio en la base de datos local de Git, la parte principal del versionado, ya que posee todos los metadatos del proyecto.



El flujo de trabajo básico en Git es algo así:

1. Se modifica una serie de archivos en el directorio de trabajo.
2. Se preparan los archivos, añadiendoles al área de preparación.
3. Se confirman los cambios, lo que toma los archivos tal y como están en el área de preparación, y almacena esa copia instantánea de manera permanente en el directorio de Git.

En conclusión, si una versión concreta de un archivo está en el directorio de Git se considera confirmada *(committed)*. Si ha sufrido cambios desde que se obtuvo del repositorio, pero ha sido añadida al área de preparación, está preparada *(staged)*. Y si ha sufrido cambios desde que se obtuvo del repositorio, pero no se ha preparado, está modificada *(modified)*.

Uso de Git

Existen muchas formas de usar Git. Por un lado se tienen las herramientas originales de línea de comandos; y, por otro lado, se cuenta con una gran variedad de interfaces de usuario con distintas capacidades. Durante el curso utilizaremos la línea de comandos. Que es es el único lugar en donde se puede ejecutar todos los comandos de Git, debido a que la mayoría de interfaces gráficas de usuario solo implementan una parte de las características de Git por motivos de simplicidad.

Para conocer la forma de instalar Git en los diferentes sistemas operativos les recomendamos visitar el siguiente enlace:

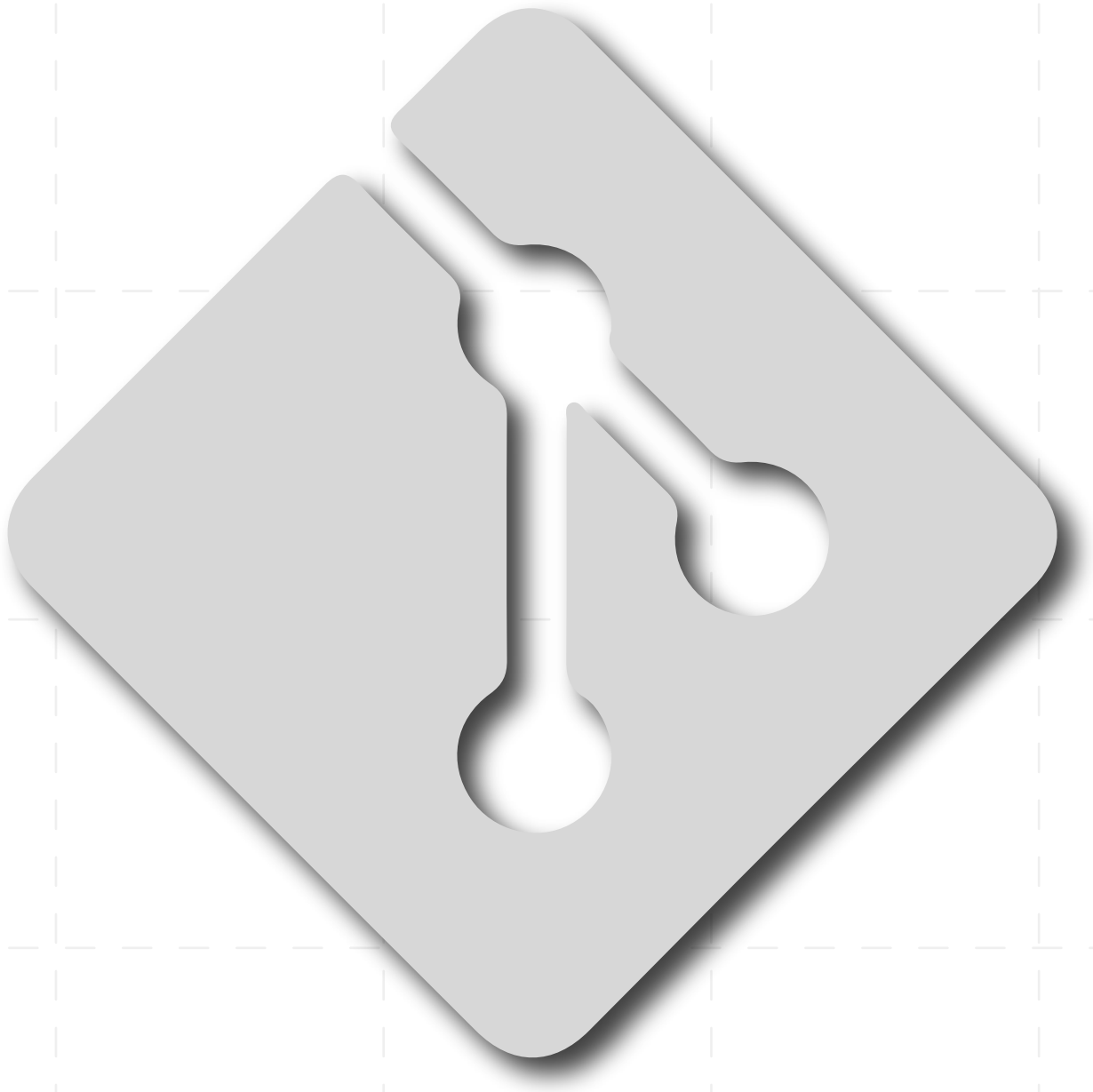
https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Instalación-de-Git

Instalación del software y ver la consola de comandos (https://git-scm.com/download/win)

Video de instalación de Git:

1. Descarga del instalador de *Windows*.
2. Instalación paso a paso.
3. Lanzamiento de la consola de Git bash.

Ahora que hemos visto cómo se instala Git y cómo se lanza la consola de Git bash debemos practicar el proceso, además de profundizar a través de la documentación de Git que se tiene a disposición en el material disponible en el aula.



Universidad de Caldas