AE-2.

Control de versiones con GIT, GitHub y un IDE.



Realizado por: Gonzalo Sanz López-Peña

Daniel Del Río Pérez

INDICE.

1. Introducción
2. Trabajo realizado
3. Pasos Realizados
4. Conclusión
5. INTRODUCCION.

Como ya se citó en el anterior trabajo, los componentes del grupo únicamente somos Gonzalo y Daniel, por su parte Jorge Zuloaga González y Luis Javier Álvarez Delgado no han participado en la actividad a resolver.

Se pide trabajar con Git para la segunda parte de la entrega, que tiene que ver con el control de versiones.

La actividad consiste en trabajar con la clase calculadora que puede realizar una serie de operaciones, suma, resta, multiplicación y cociente, en la que no se pueden introducir números negativos, sino saltarán una serie de avisos, en la que la operación no puede devolver dicho resultado y se mantiene a 0.

1. TRABAJO REALIZADO:

Se ha creado un repositorio en GitHub denominado GrupoOcho cuyo enlace es el siguiente:

https://github.com/GrupoOcho/ActiviadadCalculadora

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Como vemos hay 3 ramas con diferentes versiones, en la última de ellas versión 1.2, en la rama denominada documentación se encuentra la clase calculadora con todos los métodos realizados, adaptado a las características que se necesitaban, que no se pueda pasar como parámetros números negativos y en las clases resta que no pueda arrojar un negativo.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Además, se han tenido en cuenta todas las consideraciones sobre los métodos cuando es infinito, o casos especiales, como en el calculo del cociente o inverso.

Se ha subido un readme, para que aquellas personas que visiten la web sepan de que va la aplicación creada de la calculadora.

En el main, se han metido excepciones y algunos métodos para comprobar que los resultados son correctos y funciona correctamente la aplicación

Se ha trabajado tanto con los comandos Git, como con Eclipse. Al ser únicamente dos compañeros, ha sido fácil la compenetración y el trabajo en equipo.

Vemos el readme:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Aquí tenemos el main:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. PASOS REALIZADOS.

Trabajo realizado por Gonzalo:

1. Se ha trabajado tanto por GIT BASH como por Eclipse, aunque más por Eclipse que es lo que se solicitaba en la actividad.
2. Se ha seleccionado el proyecto desde la ventana Packet Explorer y se ha seleccionado la opción de team y share Project.
3. Se ha configurado el repositorio local donde tenemos los archivos con los que se han trabajado.
4. Ya tenemos todo listo para trabajar en local, donde se han realizado diferentes acciones para realizar sobre los ficheros que necesitemos.
5. Se ha creado un repositorio remoto en GitHub, donde se han ido subiendo las diferentes versiones y proyectos.
6. En Eclipse, en la perspectiva Git se han añadido los archivos al Índex y se ha realizado el push y el commit
7. Como hemos visto, se han subido los archivos correctamente, una vez que se han creado todos los métodos y se ha realizado el main. En el Javadoc, y en el código vienen las partes que ha realizado Daniel y las partes realizadas por mí (cociente y resta), además de ayudar a elaborar el main en conjunto.
8. Se ha trabajado también por ramas, y diferentes versiones para ver la operativa de Git, añadiendo modificaciones al proyecto con sus respectivos commits.
9. Ver en anexo la parte realizada por comandos.

Trabajo realizado por Daniel:

1. Se ha trabajado tanto por GIT BASH como por Eclipse, aunque más por Eclipse que es lo que se solicitaba en la actividad.
2. Se ha seleccionado el proyecto desde la ventana Packet Explorer y se ha seleccionado la opción de team y share Project.
3. Se ha configurado el repositorio local donde tenemos los archivos con los que se han trabajado.
4. Ya tenemos todo listo para trabajar en local, donde se han realizado diferentes acciones para realizar sobre los ficheros que necesitemos.
5. Se ha creado un repositorio remoto en GitHub, donde se han ido subiendo las diferentes versiones y proyectos.
6. En Eclipse, en la perspectiva Git se han añadido los archivos al Índex y se ha realizado el push y el commit
7. Como hemos visto, se han subido los archivos correctamente, una vez que se han creado todos los métodos y se ha realizado el main. En el Javadoc, y en el código vienen las partes que ha realizado Daniel y las partes realizadas por mí (cociente y resta), además de ayudar a elaborar el main en conjunto.
8. Se ha trabajado también por ramas, y diferentes versiones para ver la operativa de Git, añadiendo modificaciones al proyecto con sus respectivos commits.
9. Ver en anexo la parte realizada por comandos.

No hemos tenido demasiados problemas al ser únicamente dos personas, se ha trabajado lo mismo por ambas partes.

1. Conclusión

Como hemos visto Git y GitHub son herramientas diferentes, pero complementarias. GitHub es una plataforma que permite almacenar los respaldos creados mediante Git en la nube. De esta manera se ha conseguido que las copias de las versiones del proyecto de la calculadora no estén únicamente en nuestro disco duro, previniendo posibles problemas que puedan surgir al trabajar en local.

Gracias a Git hemos llevado un registro de los cambios en los archivos utilizados, pudiendo coordinar el trabajo entre dos personas en este caso compartidos en el repositorio de código.

En el repositorio de GitHub el usuario podrá ver la clase Main, con los métodos realizados y comprobar que la aplicación funciona correctamente, en base a lo descrito en el JavaDoc que también se encuentra en el repositorio, con las explicaciones oportunas de cada método, parámetros y resultados.

**En resumen podemos decir que Git es una herramienta:**

* **Muy potente**
* **Muy rápida**
* **Multiplataforma**
* **Que se puede utilizar a través de la línea de comando o con múltiples clientes**
* **Es la base o el primer eslabón de herramientas de estructura a plataforma tipo GitHub que son las plataformas que se utilizan, de manera masiva, para albergar proyectos de software libre y otro tipo de proyectos que pueda tener una organización, y que quieran delegar en GitHub como su Servicio de Gestión de Control de Versiones Distribuidas.**

Anexo Gonzalo

Un pequeño ejemplo del trabajo realizado con los comandos, aunque luego se realizó la actividad con Eclipse.

**GIT:**

1. **Crea un repositorio nuevo en una carpeta con un proyecto.**

Se ha creado la carpeta .git, en el repositorio en el cual tenemos la clase calculadora.

Como vemos nos situamos en la rama master y como vemos, no hemos realizado ningún commit aún.

Texto

Descripción generada automáticamente

Nos situamos sobre la clase calculadora, que es sobre la que vamos a trabajar.

Texto

Descripción generada automáticamente

Realizamos la identificación de usuarios, ya que se va a realizar cambios y que cuando uno de los integrantes realice un cambio haya un feedback hacia el resto. Los parámetros name y email, nos ayudarán a identificar al usuario y recibir notificaciones en un correo común que hemos puesto para el grupo.

Texto

Descripción generada automáticamente

1. **Crea una copia local en el repositorio.**

Añadiremos el fichero especificado al staging área del repositorio.

Como vemos estamos en la rama master, aún no hemos commitado, a pesar de que hay un fichero en el index que se puede commitar.

Texto

Descripción generada automáticamente

Al commitar Git asigna un identificador al commit, que es el que se invocará luego para movernos por los commits.

Texto

Descripción generada automáticamente

Realizamos un git log que muestra un histórico de los commits que se han realizado, el identificador que tienen, el usuario que que lo ha realizado y la fecha. Además, vemos como viene la rama y ubicación actual del puntero.

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza media

1. **Realiza un cambio en el código y añádelo al repositorio.**

Accedemos al código:

Texto

Descripción generada automáticamente

Salimos del código y comprobamos que los cambios efectuados en la clase suma se han registrado con éxito.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Volvemos a realizar el git status y comprobamos que:

* Seguimos en la rama master
* Hay cambios que se han realizado sobre el fichero calculadora, y que no se han commitado.
* Nos sugiere dos comandos, uno para añadir y el otro para mover el fichero del repositorio al index

Texto

Descripción generada automáticamente

Añadimos y commitamos y al consultar el log ahora vemos que tenemos dos entradas, una con el primer commit y otra para el segundo tras cumplimentar la clase suma todos sus parámetros.

1. **Crear una rama y situarse en ella**

Consultamos el id del commit:

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza media

Creamos la nueva ramo documentación:

Texto

Descripción generada automáticamente

Pasamos de una rama a otra con git checkout

Texto

Descripción generada automáticamente

Ahora mismo, tenemos dos ramas, el puntero Head esta en la rama documentación.

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza media

Lo subimos con push:

Texto

Descripción generada automáticamente