**Atajos**

Ctrl. + E : Borra línea de codigo donde estoy posicionado.Ctrl. + S : Guarda los cambios en la clase que estoy posicionado.Shift. + F6 : RUN main proyect (actual).Alt. + Shift + F : Le da formato al código.(queda mas legible, ordenado)Alt + Enter : Se usa sobre el texto subrayado para ver la sugerencia que nos hace el IDE cuando hayalgún error de sintaxis o se necesita realizar algún import o surround.Ctrl. + Shift + : Hace una copia de la línea de código que estabas posicionado.Shift + DELETE : Borra toda la línea de código donde estoy posicionada.Ctrl. + Space : Completa el código que estamos escribiendo.Ctrl. + U U : Convierte el código a mayúsculas.Ctrl. + U L : Convierte el código a minúsculas.Alt + Insert : Agrega constructores, métodos accesotes, propiedades.(codigos predeterminados)Ctrl. + G : Ir numero de línea concreto(introducida x ventana auxiliar).Ctrl. +[NOMBRECLASE] : Me despliega todas las características de la clasePara completar formatosif. + TAB : Genera los bloques if.sout. + TAB : Genera los System.out.printlnfori + TAB : Genera el codigo necesario para hacer un for.sw + TAB : Genera el switch.whilexp/whilen/whileit + TAB : Genera los bloques while.dowhile + TAB : Genera los bloques dowhile.PSVM + TAB : Genera la función main.Para crear propios ATAJOSTools / Options / Editor / CodeTemplates / New :-Primero voy a poner la abreviatura para llamar a mi “código”-Luego escribo el código a utiliza

GIT - GITHUB

Comando Explicación

Git Init

Para iniciar el trackeo de los archivos de una carpeta.

Git Status

Nos da toda la información necesaria sobre la rama actual.

Git Add

Para incluir los cambios del o de los archivos en tu siguiente commit.

Git Commit

Es como establecer un punto de control en el proceso de desarrollo al cual puedes volver más tarde si necesario. Es como un punto de guardado en un videojuego.

Git Push

Es Enviar archivos incluidos en el commit al repositorio local.

Git Push Origin

Enviar archivos incluidos en el commit al repositorio remoto.

git branch

Para ver tanto las ramas de seguimiento remoto como las ramas locales, ejecuta el comando

**POO**

Si el nombre es compuesto, entonces cada palabra debe empezar por mayúscula. La definición de la clase se pone entre las llaves de apertura y cierre.

public class NombreClase {

// atributos

// constructores

// métodos propios

}

El estado o información de un objeto se almacena en atributos.

<modificador>\* <tipo> <nombre>

<modificador>: si bien hay varios valores posibles para el <modificador>, por el momento solo usaremos modificadores de visibilidad: public, protected, private.

<tipo>: indica la clase a la que pertenece el atributo definido.

<nombre>: puede ser cualquier identificador válido y denomina el atributo que está siendo declarado.

Toda clase debe definir un método especial denominado constructor para instanciar los objetos de la clase.

[<modificador>] <nombre de clase> ( <argumento>\* ) {

<sentencia>\*

{

<nombre de clase>: mismo que el de la clase. El nombre del constructor debe ser siempre el mismo que de la clase.

<modificador>: Actualmente, los únicos modificadores válidos para los constructores son public, protected y private.

<argumentos>: es una lista de parámetros que tiene la misma función que en los métodos.

El operador **new** instancia el objeto y reserva espacio en memoria para los atributos y devuelve una referencia que se guarda en la variable.

nombreObjeto = new nombreClase();

Para lograr el uso correcto del encapsulamiento vamos a utilizar los modificadores de acceso, estos, van a dejarnos elegir como se accede a losdatos y a través de que se accede a dichos datos

Public: Este modificador permite acceder a los elementos desde cualquier clase , independientemente de que esta pertenezca o no al paquete en que se encuentra el elemento.

Private: Es el modificador más restrictivo y especifica que los elementos que lo utilizan sólo pueden ser accedidos desde la clase que se encuentra.

Protected: Este modificar indica que los elementos solo pueden ser accedidos desde su mismo paquete y desde cualquier clase que extienda la clase en que se encuentra.

**Visibilidad Public Private Protected Default**

Desde la misma clase SI SI SI SI

Desde cualquier clase SI NO SI SI

del mismo Paquete

Desde una Subclase del SI NO SI SI

Mismo paquete

Desde una Subclase fuera SI NO SI, a través de la NO

Del mismo Paquete

Desde cualquier Clase fuera del SI NO NO NO

Paquete

Para definir los métodos, el lenguaje de programación Java toma la siguiente forma básica:

<modificador>\* <tipo de retorno> <nombre> ( <argumento>>\* ) {

<sentencias>\*

Return valorRetorno;

}

¿Cómo se crean los getter y setter?

Dentro de la clase, para agregar automáticamente los getter & setter podemos presionar el botón derecho > Insert Code > Getter & Setter y selecciono todos los atributos.