

Desarrollo de Interfaces

Documentación

Calculadora



Índice

I.	Introducción	3
II.	Metodología del proyecto	3
III.	Parte gráfica	3
IV.	Diseño de experiencia del usuario	6
V.	Fases de desarrollo	10
	A. Sketch	10
	B. Wireframe	11
	C. MockUp	22
	D. Prototipo	23
VI.	Fases	24
VII.	Testeos	26
VIII.	Versiones	28
IX.	Problemas de desarrollo	30
X.	Código sacado de internet	35
XI.	Librerías	35

Introducción

A petición del profesor vamos a hacer un proyecto de una calculadora con un diseño óptimo, tendrá las funciones básicas de la misma como sumar, restar, multiplicar, dividir, porcentaje y cambiar el signo de la operación, las operaciones se ejecutan al instante (no almacena operaciones) inspirado en el funcionamiento de la calculadora de windows, a parte está conectada a una base de datos donde almacena el historial de la calculadora donde podrás coger esas operaciones.

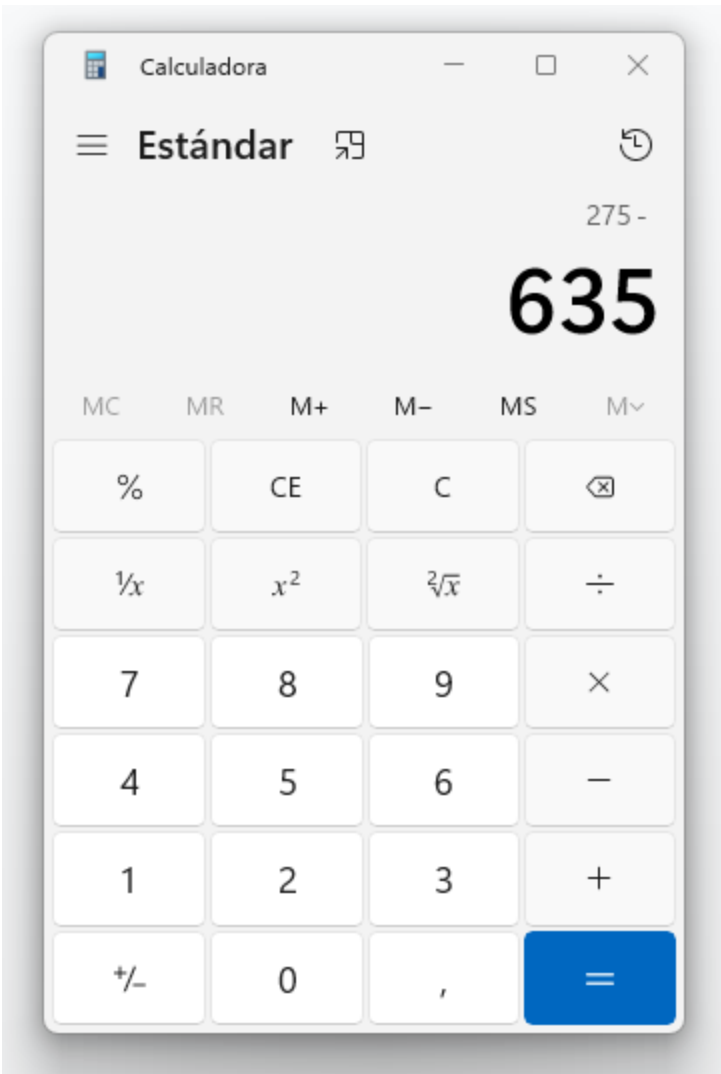
Metodología del proyecto

Voy a utilizar una metodología lineal donde primero mirare los requerimientos necesarios para hacer la calculadora y el tiempo necesario para cada uno de ellos, después me voy a centrar en el diseño donde voy a buscar la más óptima y estética posible que se me adecue a mis necesidades , cuando termine el diseño implementare toda la programación necesaria para que todo funcione correctamente y haré testeos buscando errores

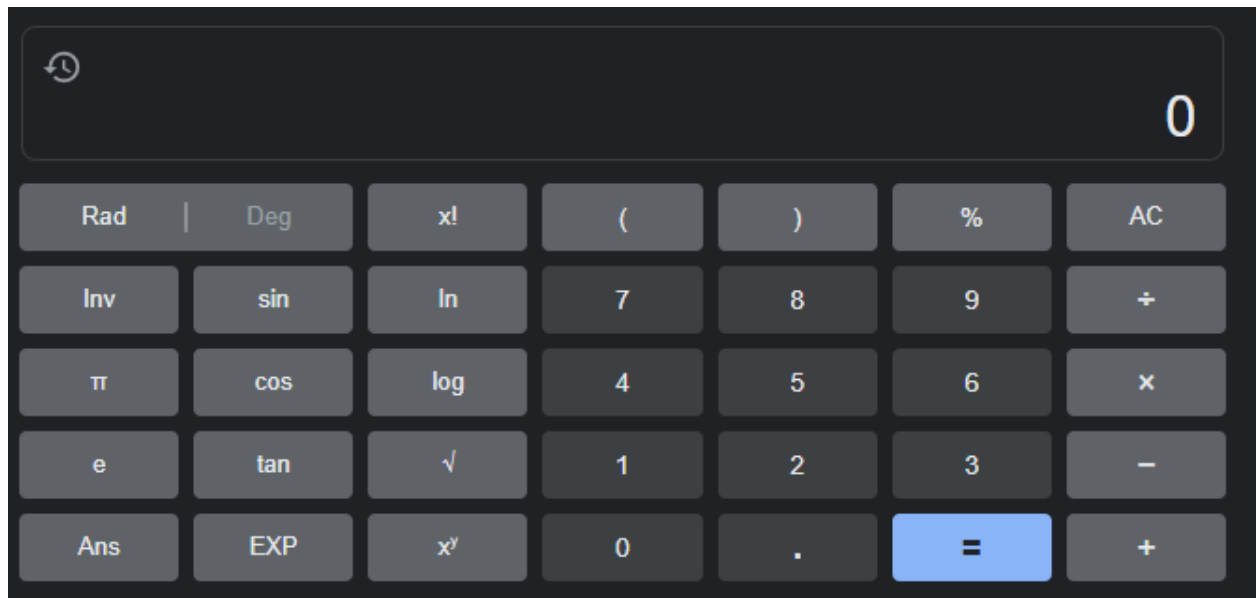
Parte gráfica

Lo primero que he hecho ha sido empaparme de diferentes tipos de calculadoras con sus diferentes diseños y buscar la más óptima para inspirarse sobre la misma

Me ha gustado la de windows por su diseño y su forma de operar ya que puedes hacer el igual si le vuelves a las a los signos de +-x/ y me voy a inspirar en esta metodología



El Problema de esta calculadora es que tiene más operaciones de las que voy a usar yo y he visto que google utiliza una forma del diseño que solucionaba mi problema de diseño



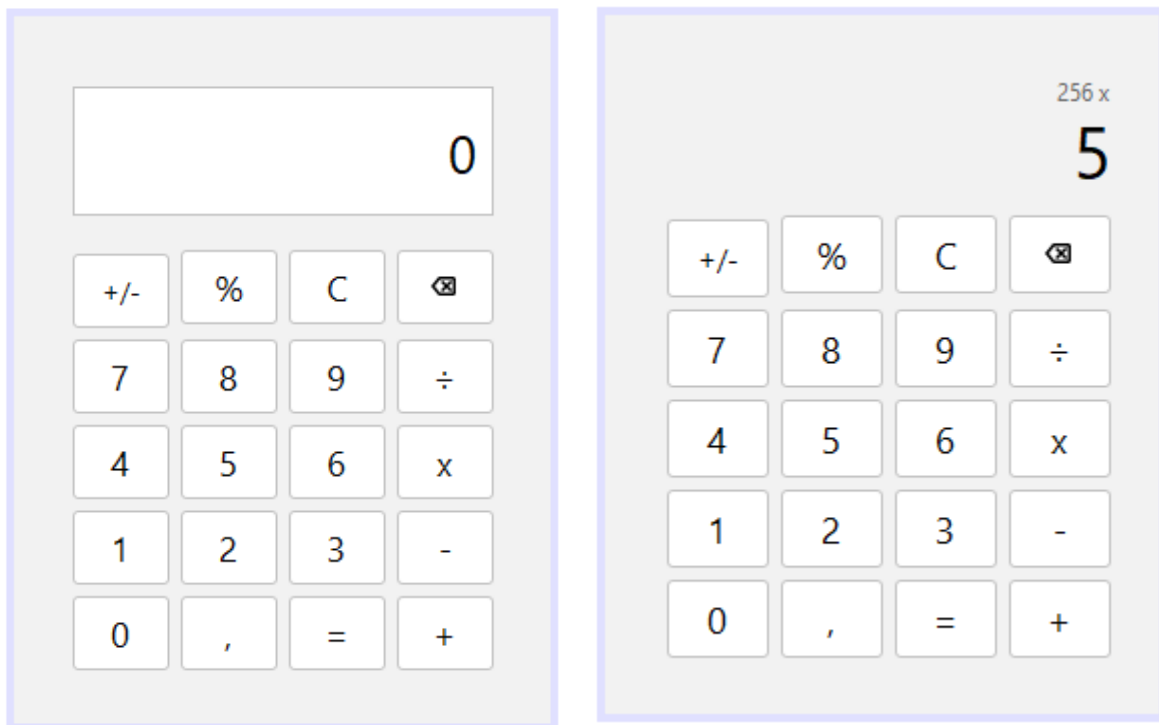
Esta calculadora no pone el igual en la derecha del todo sino que la corre a la izquierda junto con el 0 y la coma.

Me he inspirado en estos dos diseños para hacer mi Sketch ya que los veo los más óptimos.

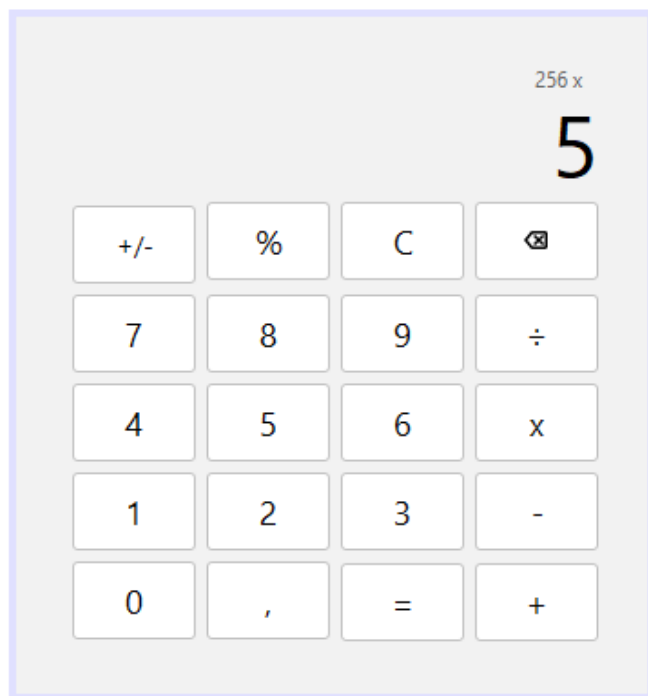
Diseño de experiencia del usuario

El primer dilema de diseño ha sido si crear una calculadora con un Jtext o un dos labels para poner los números.

Ya que al hacerlo con un Jtext se ve más antigua y con labels en cambio se ve más estética y mas comoda a la hora de hacer muchos cálculos ya que para encadenar sumas o cualquier tipo de signos es más eficaz hacerlo con dos labels; uno en la parte inferior más grande y de color negro que tendrá el solo en numero sin ningun signo y otro label superior que tendrá el numero más el signo que se va a operar.



Algunos usuarios se quejan de que los botones son pequeños por eso he decidido agrandarlos un poco para que no te equivoques pulsando y tenga un mejor diseño

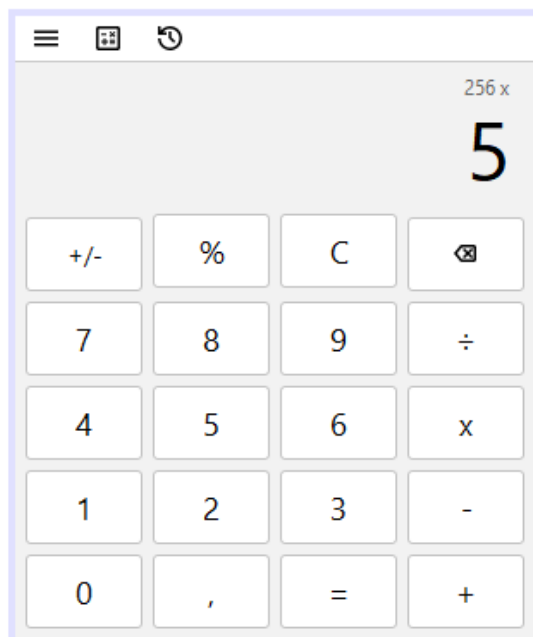


En la etapa inicial del proyecto quería poner un botón que cambiará cada vez que se le pulse para abrir el panel de historial o el panel de operaciones pero si ponía el botón a nivel estético quedaría el menú y un botón en medio haciendo fuera innecesariamente grande y con muchos espacios sin aprovechar así que he puesto dichos botones en la parte del menú quedando todo más óptimos y limpio



Por lo que estuve viendo en las app de calculadoras pocas tienen un diseño con grandes bordes, la mayoría optimiza el espacio al máximo posible como por ejemplo la de windows

Así que en base a esto he acertado los bordes de mi calculadora

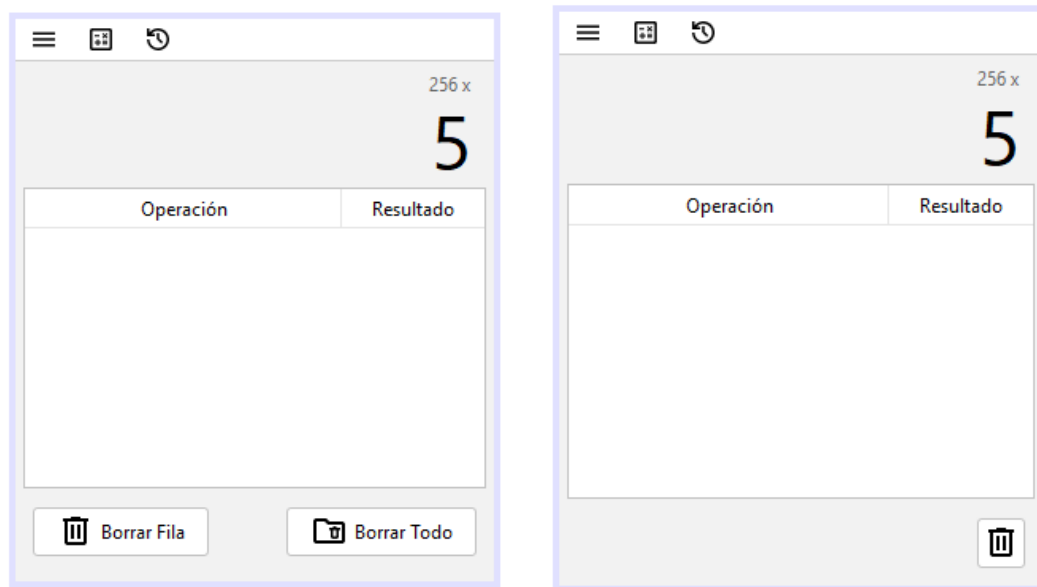


Uno o dos botones de eliminar Historial:

un problema de diseño que he tenido es si meter un toton que elimine todo el historial o si en cambio meter uno para eliminar la fila seleccionada y otro para borrar todas las filas.

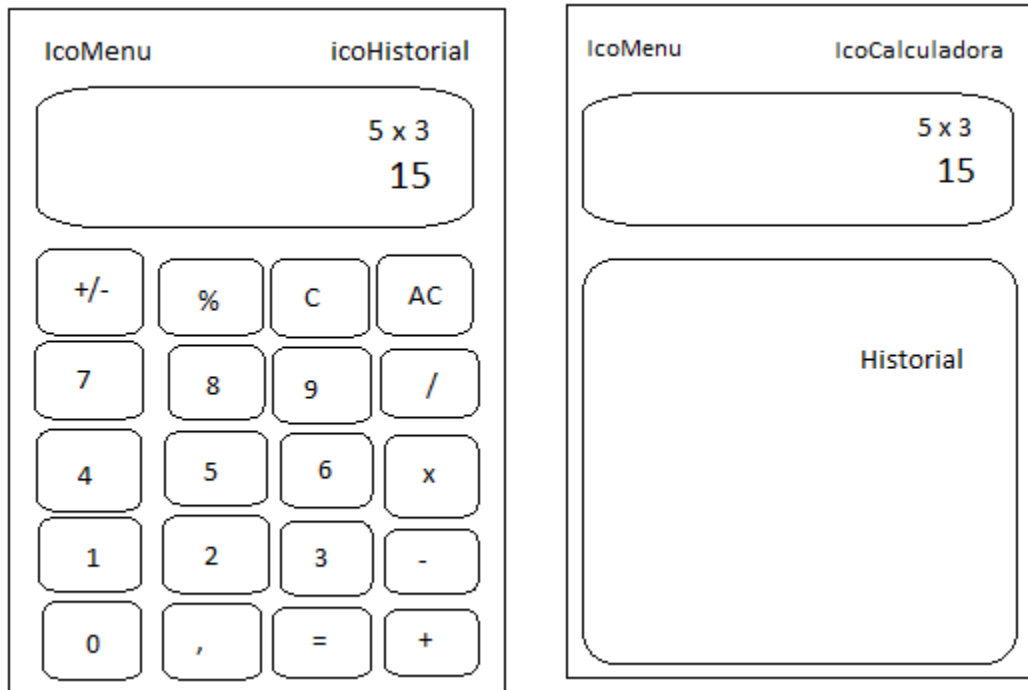
También he pensado si crear un botón que si no seleccionas ninguna fila elimina todas y si seleccionas una te elimina esa fila, esto elimina mi problema estético pero creo que no es orientativo y puede llevar a confusión

Así que he estado viendo algunas calculadoras con historial y lo borran todo directamente con un solo botón y yo también pienso que no tiene mucha utilidad de borrar una sola línea y he optado por una más estética que sería poner un botón que elimine todo el historial.



Fases de desarrollo

Sketch



No he puesto el = a la derecha del todo ya que al no poner muchos componentes me entraba en conflicto con el borrar, en cambio he optado por poner el +/- en la parte superior correr en 0 a la izquierda junto con la "," como el diseño de la calculadora de google, teniendo las operaciones básicas "+, -, x, /" juntas, el borrar arriba junto con la C, el % y +/-.

Quedando un diseño limpio y estético con un menú en la parte superior donde podrás salir, hay un icono de Historial que al pulsarlo se ocultará el panel actual, el cual abrirá otro panel donde se podrá ver el historial, hacer click y obtener el resultado.

Manera de representar el resultado:

He optado por hacer algo similar a la calculadora de windows donde hay dos Jtext una que guarda la operación y otra que solo el cálculo actual, es decir yo pulso un número y se

almacena en el TextNumero en el momento que yo le de a un cálculo como sumar se guardará en el TextCalculo.

de esta forma queda mas estetico y a mi parecer fas fácil de verlo el cálculo realizado y poder concatenar operaciones rápidamente y de forma óptima



Wireframe

Menú:

En la parte superior habrá un menú el cual al pulsarlo te saldrá un submenú donde podrás salir de la aplicación, modificar el diseño de la calculadora o cambiar el panel para abrir el historial e ir a la calculadora si ya estás en el historial.

Botón jMenuHistorial:

Ocultar el panelOperaciones y lo deshabilita, enseña y habilita el panelHistorial y actualiza los datos de el jTable tableHistorial

Botón MenuCalcu

Ocultar el panel panelHistorial y lo deshabilita, enseña y habilita el panelOperaciones

jMenu

Abre un submenú donde puedes salir de la aplicación o elegir un diseño

jMenuSalir

Ejecuta System.exit(0) para cerrar la app

jMenuDisenyo

abre cuatro botones del menú donde puedes elegir el diseño que más te guste

JMIDisenyoStandard / JMIDisenyoClaro / JMIDisenyoOsucro / JMIDisenyoCreem

aplican un diseño al Frame con el uso de un método disenyo(String nombre)

Historial:

En la parte derecha del menú pretendo poner un botón que tenga un icono de historial que al pulsarlo esconda el panel de las operaciones y que haga visible otro donde se cargará el historial en un Jtable que al hacerle doble click se ponga el resultado de la operación en el txtOperacion.

Al pulsar el botón historial se desactivará, se hará invisible y activará otro en la misma posición, que al pulsarlo hará que se oculte el panel del historial, que este se desactive y a su vez activa y hace visible el panel de las operaciones

JText:

TextCalculo:

Aquí se almacena la operación hecha, es decir si yo pongo un número se pondrá en TextNumero y le doy a sumar o alguna operación, se almacenará aquí dejando el TextNumero vacío

TextNumero:

Almacena el texto de la número actual sin ninguna operación

Base de datos

La base de datos tiene una tabla, esta tabla se llama operaciones y está compuesta por un id que es ope_codope es un valor int que además es primary key, not null y auto_increment, también tenemos ope_operacion donde se registra la operación sin el resultado, es un varchar(30) y también es not null, por último guardamos ope_resultado que es un double donde guardamos el resultado.

Botones:

(accedo a estos botones desde el controlador teniendo el diseño y los procesos separados)

Botones numéricos:

Estos botones solamente introducirán al texto actual el número del botón, llamando a la función `meterNumero(int num);`

Botón btnComa:

Introduce una coma llamando al método `meterComa()`, este tiene un algoritmo que mira si ya hay una coma en el texto para que no se pueda poner más de una coma por operación

Botón btnIgual:

Llama al método `hacerOperacion(String operacion)` llevando un null que indica que es un igual, este método se encarga de hacer la operación y ponde el resultado de la operación en el `lviNumero` y añade al historial de la última operación en la parte superior `"lbiCalculo"`

Botón btnmultiplicar, btndividir rbtnestar y btnsumar:

Estos botones llaman al método `hacerOperacion(String operacion);` con la operación pertinente este método hará toda la operación actualizará los resultados en los `lbi` y añade en la BBDD la operación (explico más detalladamente este método más adelante)

Botón btnMasMenos :

Este botón llama al método `cambiarSigno();` el cual Primero detecta si el número es positivo o negativo buscando `"-"` y si es negativo añadirá al principio de la operación un `"-"` y si es negativo quitara el `"-"` haciendo que cuando se opere un cálculo salga el número positivo o negativo

Botón btnPorcentaje:

Llama al método `porcentaje();` el cual coge el número con el que se está operando y lo divide por 100 sacando el porcentaje poniendo en el text el número que nos a dado y quitando el anterior automáticamente

Botón btnC:

Se encarga de limpiar todo el text del número que se va a operar y la el text donde está el resto de la operación

Ej 243 * pasa a

56 0

Botón de btnAC o ico.restarNumero:

Coje el TextCalculo que el el texto donde reside el número actual sin ninguna operación y le elimina el último número, si va a eliminar el último número pondrá un 0 y si solo hay un 0 no podrá eliminar nada

Ej 243 * pasa a 243 *

56 5

Botón btnBorrarHistorial

Si no hay nada seleccionado borrará todo el historial con el método borrarHistorial() y si del jTable tableHistorial y si hay alguno seleccionado ejecutará el método borrarFila()

Métodos:

public void meterNumero(int num):

Los botones llaman a este método para meter un numero en el lblNumero

Variables:

int num: El número que meteremos en lblNumero

Que hace:

- Mira si es 0 para no poner un 0 de primer numero
- miro si es el número de la operación anterior ya que al pulsar una operación no te borra el número hasta que hagas click en otro número (utilizando la variable booleana borrar)

public void hacerOperacion(String operacion)

Variables:

String operación: indica la operación que se quiere realizar (si es un null significa que se pulsó el botón =)

Que hace:

Primero mirará si hay algo en el lblCalculo.

Si este está vacío lo meterá ahí con el signo y se acabará .

Si ese tiene algo lo cogerá con su signo y llamará al método calcular pasándole los datos pertinentes.

Una vez tenga el resultado lo mete en el lblCalculo con el signo de la siguiente operación y en el lbl Número el número.

Después de esto llama al método guardar(String operacion, double resultado) que lo guardara en la base de datos

Métodos para obtener los números de la operación

public double obtenerNumerosLblCalcular():

este método cogerá número que tiene el lblCalculo

public static Double calcular(double num1, double num2, String operacion)

Variables:

double num1: el primer número que queremos calcular

double num2: el segundo número que queremos calcular

String operación: indica la operación

Devuelve un double con el resultado de la operación

Que hace:

Se encarga de calcular los dos valores que le metas con la operación pertinente "+ - x ÷"

Transforma los double en BigDecimal para operar y devuelve un double

Métodos para obtener los números de la operación

public static double obtenerNumerosLblCalcular()

Que hace:

Devuelve un double del número que esté en la parte superior almacenado (lblCalculo)

public static double obtenerNumerosLblNumero()

Que hace:

Este método cogerá número que tiene el lblnumero

(se han creado estos métodos para ahorrar código y poder acceder fácilmente a estos números en todo momento)

public static String quitar0(double num)

Variables:

double num: un número que puede tener decimales o no

Devuelve un String con un número

Que hace:

Se le pasa un numero y le devuelve un string con el mismo número si tiene 15.0 pasa a 15 y si tiene 15.5 se queda igual

(se creó este método para que el lblcalculo no tuviera decimales con números como 15.0)

public static void setTextNumeroIgual(double numero1, double numero2, String op)

Variables:

double numero1: el primer número con el que se va a operar

double numero2: el segundo número con el que se va a operar

String op: solo se utiliza para indicar si es un = que irá con null

Que hace:

Se le pasan los valores al pulsar = y se encarga de hacer la operación, prorratar los doubles y poner el resultado en los labels

(se creó para evitar duplicar código)

public static void meterComa()

Que hace:

mete coma en el lblNumero

si ya tiene coma no meterá la coma

También tiene una función que mira si son iguales lblCalculo y lblNumero y si se ha hecho una operación para poner directamente 0.

public static void borrar(boolean borrarTodo)

Variables:

boolean borrarTodo: si esta variable es true borrara todo y pondrá por defecto las variables de comprobación como borrar y borrarTodo que se activan después de ejecutar una operación, si es false borrará uno

Que hace:

Borra texto. según como sea la variable borrará de una forma o otra

public void porcentaje()

Que hace:

Saca el porcentaje del número del lblNumero y lo pone por pantalla. Si en el lblcalcular no hay nada se pone un 0 como en el funcionamiento de la calculadora de windows Se tiene en cuenta el número de decimales para poner la escala de la división del bigDecimal correctamente porque sino saldría 0.002500000000000016541654165

public void cambiarSigno()

Que hace:

Cambia el signo del lblNumero

public static void anyadirOperacionesTabla()

Que hace:

Pone los datos de la línea seleccionada en el lblCalculo y el lblNumero

public static void disenyo(String nombre)

Variables:

String nombre: Es un string donde le dice el nombre del diseño que quieres y lo pone en todo el Frame

Que hace:

Recibe un parámetro, según este parámetro establecerá un diseño cambiando el UIManager.setLookAndFeel y aplicando unos cambios manualmente como el fondo o el btnIgual

public static void colorFondo(Color colorFondo, Color colorIblNumero, Color colorIblCalculo, Color colorBtnIgual)

Variables:

Color colorFondo: le decimos el color que queremos para el fondo

Color colorIblNumero: le decimos el color que queremos para el lblNumero

Color colorIblCalculo: le decimos el color que queremos para el lblCalculo

Color colorBtnIgual: le decimos el color que queremos para el btnIgual

Que hace:

Le introducimos los colores que queremos para el diseño del frame

(se ha creado para ahorrar código en el método disenyo(String nombre))

Paneles:

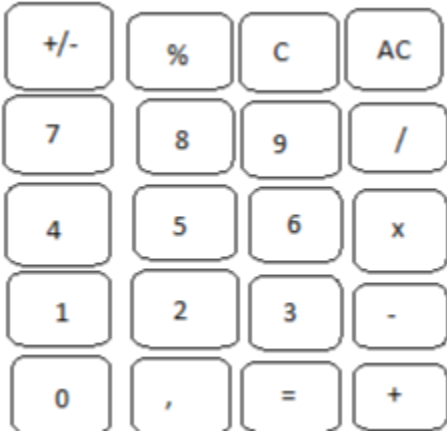
El proyecto se divide en el panel general donde estará todos los botones t text de la parte superior del programa



después se divide en dos paneles que se acceden mediante el botón IcoHistorial que al pulsarlo se oculta u saldrá otro botón con forma de operación para acceder a las operaciones

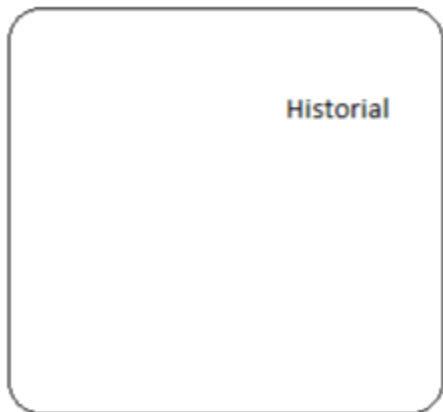
PanelOperaciones:

Está todos los botones y operaciones de la calculadora



PanelHistorial:

Hay un Jtable donde se almacena todos los registros del historial y un botón donde se puede eliminar el historial



Clases

Operacion

private String operacion: se guarda la operación

private Double resultado: se guarda el resultado de la operación

métodos de la clase

public static void insertarOperacion(String operacion, double resultado)

Variables:

String operacion: se guarda la operación

Double resultado: se guarda el resultado de la operación

Que hace:

Inserta la operación en la BBDD

public static LinkedList<Operacion> selectOperaciones()

Que hace:

carga todos los datos de la BBDD en la tabla y los mete en una LinkedList de operaciones

public static void borrarCeldas()

Que hace:

Borra todas las celdas de la BBDD

JTable:

JtableHistorial:

Que hace:

Este Jtable se actualizará cada vez que pulsemos el botón de IcoHistorial y se cargaran todos los datos de operaciones anteriores, cuando hagamos clic en uno se nos pondrán los datos de la operación anterior en en TextNumero y TextOperacion

OnMouseClicked:

private void jTableHistorialMouseClicked(java.awt.event.MouseEvent evt)

Que hace:

Cogerá el String de la row seleccionada y se cogerá el resultado de la operación, luego se actualizará el JText TextCalculo poniendo el resultado de la operación antigua para poder usarla en cálculos posteriores

private void jMenuItemHistorialMouseClicked(java.awt.event.MouseEvent evt)

Que hace:

oculta el panel de cálculo, hace visible el de historial y rellena los datos

private void jMenuItemCalcuMouseClicked(java.awt.event.MouseEvent evt)

Que hace:

Oculto el panel del historial y hace visible el panel de cálculos

private void JKeyListenerKeyPressed(java.awt.event.KeyEvent evt)

Que hace:

al hacer click te pone el resultado en el lblnumero para poder usarlo posteriormente y hacer operaciones

private void JKeyListenerKeyPressed(java.awt.event.KeyEvent evt)

Que hace:

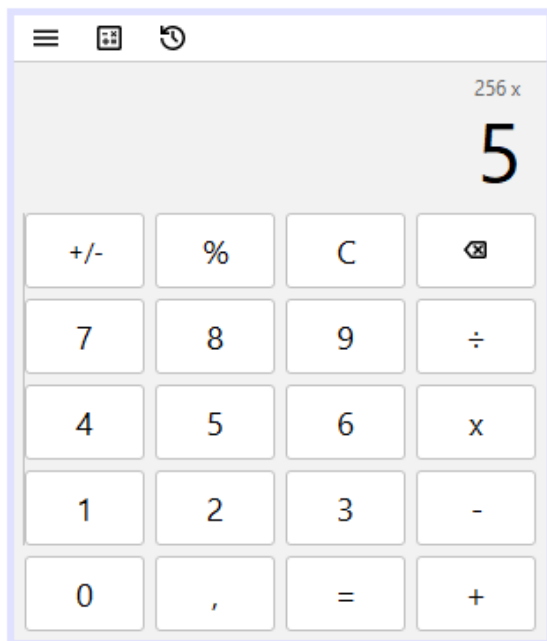
Detecta los botones del teclado como los botones de la pantalla para que sea mas eficiente

MockUp

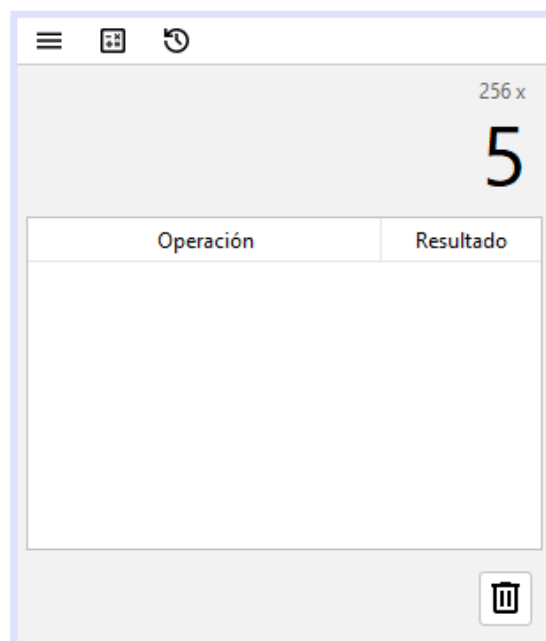
El diseño actual es un diseño óptimo, tanto en espacio como en utilidad, con una interfaz sencilla e intuitiva.

se tiene en cuenta posibles modificaciones estéticas del fondo, a un todo de blanco un poco oscuro pero no gris.

Parte de la calculadora

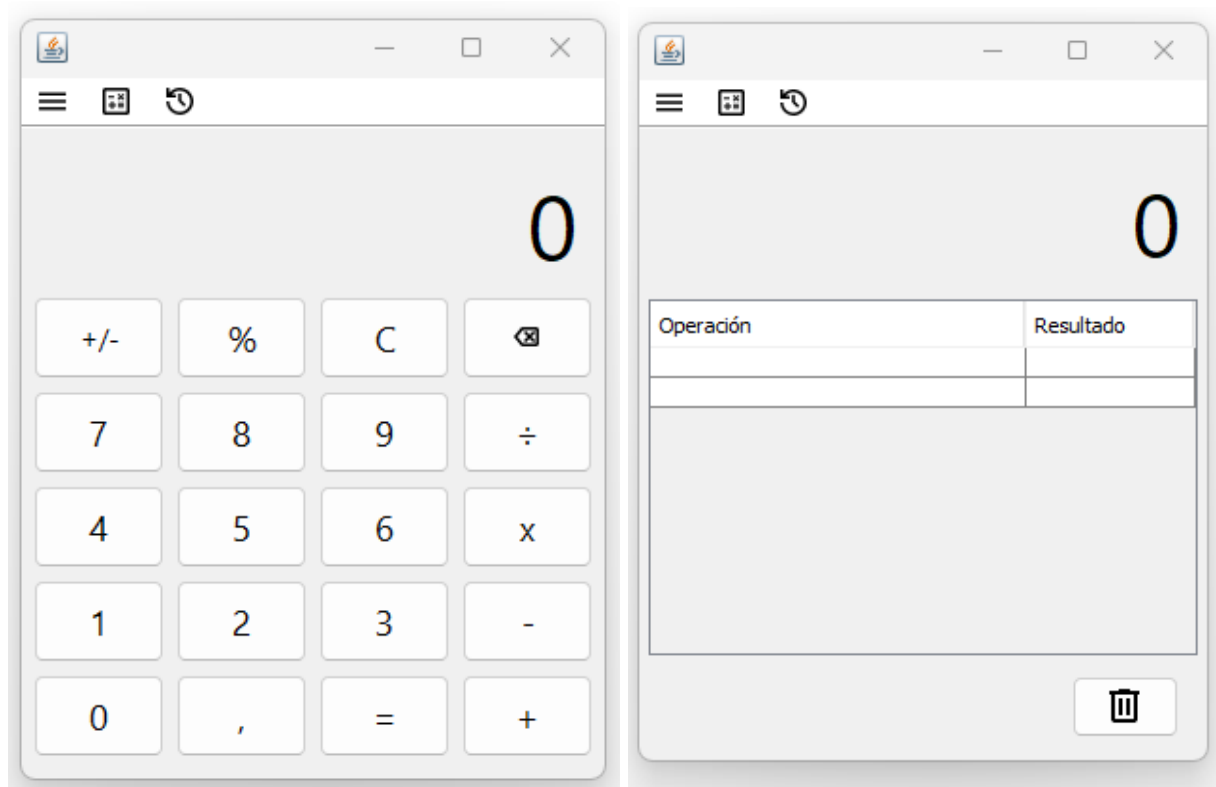


Parte del Historial



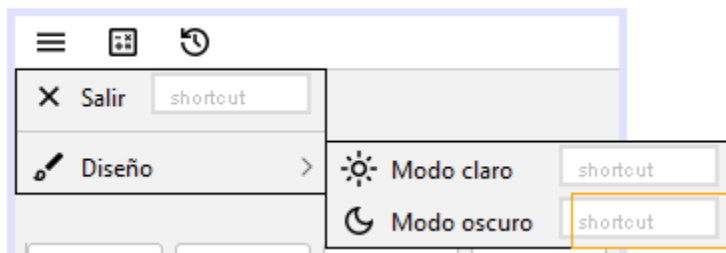
Se pasa de un panel a otro con los botones de calcular y de historial del menú.

Prototipo



El prototipo cuenta con el frame principal y un menú y dos paneles más los lbl donde se reflejan las operaciones.

los paneles se dividen en panel cálculo donde están todos los botones para hacer cálculo en esta fase ya están todos los botones funcionales y hacen todas las operaciones correctamente “La parte de la calculadora está hecha”, luego tenemos un menú funcional donde podemos cambiar el diseño o ir al panel del historial



en la parte del historial está un panel donde aparece el historial y un botón el cual borrará todo el historial

Fases

He empezado buscando que tenía que tener una calculadora para que tuviera un buen diseño de experiencia del usuario ya sea viendo comentarios de la gente viendo calculadoras populares como la de windows o la de google.

Seguidamente he hecho un Sketch para tener una imagen clara de cómo sería mi calculadora.

Después he estado creando un Wireframe donde especificaba todos los componentes de mi proyecto como botones, métodos...

Al tener el Wireframe listo me he puesto con el MockUp donde he tenido que crear todo el diseño de la calculadora y resolver algunos problemas con los paneles que especifico en Problemas de desarrollo,

El siguiente paso ha sido crear un prototipo en el cual he tenido que pasar todos los action listener y todas las operaciones lógicas de la calculadora en la carpeta de controlador para tener separada toda la parte lógica de la de diseño.

Una vez separado me he puesto a hacer funciones para poder hacer todas las operaciones más rápido y ahorrando mucho código

al hacer que se pueda hacer operaciones he hecho un pequeño testeo donde la gran parte del programa funcionaba bien menos una función de los botones que explico en problemas de desarrollo la cual he optado por quitar

He estado dedicando unos días para implementar correctamente todo el funcionamiento de hacer operaciones continuas con los operadores de $+-*/$, que al darle igual después de una operación, te cargue todo correctamente y que si le vuelves a dar te multiplique el número del resultado por el número de calculo de la operación anterior, además de arreglar todos los fallos que pueden dar estas funciones y hacer testeos para ver que funcionan correctamente

Una vez terminado todo el tema de los cálculos de la calculadora he creado el código necesario para pasar de panel a panel y añadir iconos al menú para que quede mejor.

Ya encaminando la fase final del proyecto he empezado a crear la base de datos la cual tiene una tabla que es operaciones, después he creado la conexión y he creado métodos para insertar, borrar y seleccionar los datos.

He creado los botones % +- c y borrar un num creado sus métodos y testeado que vaya correctamente

Luego he añadido una librería de diseños y también he editado yo un poco los diseños para que quede bien.

He añadido una nueva funcionalidad los keylistener para poder usar la calculadora solo con teclado haciendo la calculadora infinitamente más funcional.

Por último me he encargado de testear la calculadora y mientras lo hacía vi que $2 \times =$ daba null y hice que $5 \times =$ multiplicará por su mismo número como hace la calculadora de windows

Testeos

10 / 3:

Da infinito y peta hay que poner en la división `numero1.divide(numero2, 4, RoundingMode.HALF_UP);`

el segundo valor del divide define el número de decimales que saldrán .

el tercer valor del divide redondeado al alza.

10 / 3 pulsamos igual 3.3333 pero si luego le damos a otra operación coje el número anterior es decir que saldría 10 + en el lblCalculo

Solución: En el método public void hacerOperacion(String operacion){

En la parte de código que mira si hay una operación, antes tenía puesto que llamara a `obtenerNumerosLblCalculo()` en vez de `obtenerNumerosLblNumero()`

Uno coge el texto de `lblCalculo` y otro del `lblNumero`

} else if (borrar) {

`ventana.lblCalculo.setText(obtenerNumerosLblNumero() + " " + operacion);`

} else {

0=

Te ponía en el `lblcalculo` 0 null y lo he arreglado diciéndole que cuando sea 0 `lbl.setText("");`

0.88 x 4 salta un error en la bbdd diciendo que no puede almacenar un número tan grande.

El Problema sucede ya que en el momento de crear el big decimal con un número con decimales como 0.88 se guarda como 0.88000000000000014534335

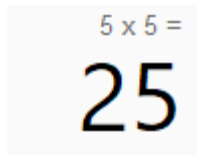
Solución si a la hora de calcular el número 1 o el num2 tienen una distancia de más de 15 "ya que cuando pasa tiene una distancia de 45 " le decimos `setScale` a 2

5 x = me da null

para arreglar esto he buscado la parte donde sucedía ya que poner texto en los lbl lo tengo mediante if else

yle he dicho que mirase si `operación == null` que significa que se pulsó un = y que no tenía segundo número ya que sino se habría ejecutado antes y decir que ejecute el método `setTextNumerolIgual(Double num1, Double num2, String operacion);`

le paso el mismo número dos veces ya que la calculadora de windows si pasa esto detecta que quieres multiplicar por su mismo número $5 \times 5 = 5 \times 5$ y de operación null para que detecte que quiero que ponga los lbl como un =

A screenshot of a calculator interface. At the top, the text "5 x 5 =" is displayed in a small, light blue font. Below it, the number "25" is shown in a large, bold, black font.

también le ponemos `borrarDespuesIgual = true;`

para que el siguiente número que insertemos borre el texto actual "si se inserta una operación calculará el número de lblnumero en este caso 25 x

A screenshot of a calculator interface. At the top, the text "25 x" is displayed in a small, light blue font. Below it, the number "25" is shown in a large, bold, black font.

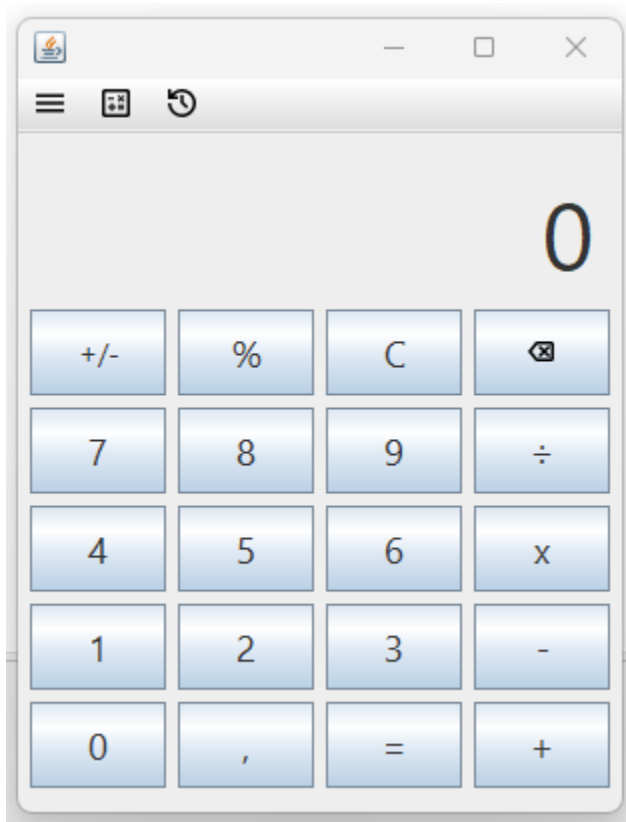
$6 * \% = da 6 * 6 = 36$

realmente está haciendo esto $6 * 0.06$

Después de estar buscando el error me he dado cuenta que la variable borrar estaba en true y me borraba el número y detectaba que quería hacer $6 * =$ y me metia el 6 que es lo que tengo puesto que haga, poniendo esta variable en false calcula bien.

Versiones

1 Versión



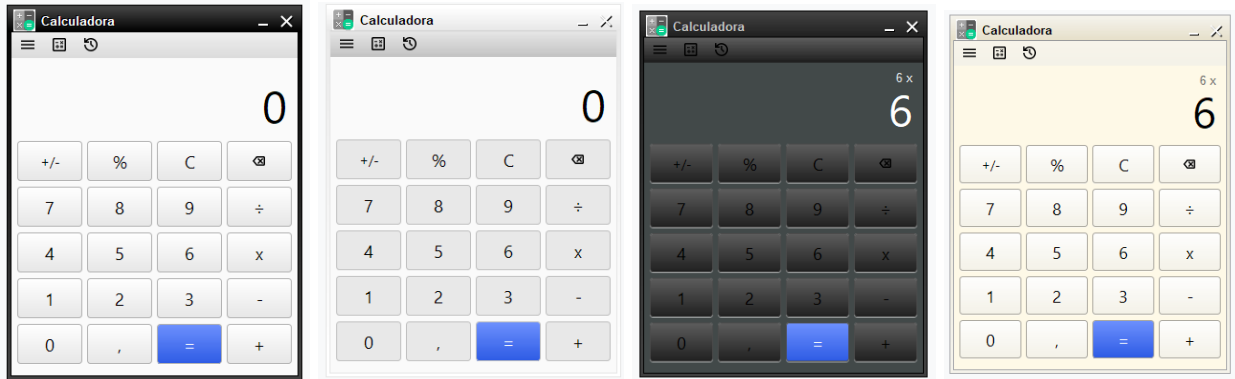
Esta primera versión tiene las operaciones básicas de la calculadora más el porcentaje y el +/-, aparte de esto puede hacer operaciones continuas con los operadores \times / $+$ - y tiene un historial donde puedes ver las operaciones realizadas borrar dicho historial y coger las operaciones que quieras

Versión 1.2

En esta versión he añadido una funcionalidad nueva, al hacer una operación y pulsar el botón igual, si volvemos a darle igual antes no hacía nada pero al mirar que hacian otras calculadoras vi que algunas como la de windows cogen el resultado y te lo vuelven a multiplicar por el número anterior.

Versión 2.0

En esta versión se ha corregido algunos problemas de los cálculos, se ha añadido la funcionalidad usar la calculadora por teclado y se le han aplicado 4 tipos de diseños



(no se puso imágenes en los botones para poder tener varios diseños sin tener que cambiar el color de todas las imágenes ya que necesitaría mucho tiempo)

Problemas de desarrollo

Al querer crear dos paneles para poder separar la parte de cálculo y la de la BBDD he tenido problemas ya que no podía poner los dos paneles en la misma posición sin que se solapen.

Después de estar intentando hacerlo invisible, probando de deshabilitar el panel sin poco resultado ya que lo podía deshabilitar pero una vez que ya estuviera en ejecución y entonces no podía programar encima de él, o incluso querer quitar el bloqueo que te pone Netbeans por defecto para poder comentar el código y una vez creará todo el código necesario para pasar de un panel a otro descomentarlo.

Al final opte por pegar un panel en la esquina y modificar la posición manualmente desde las propiedades del panel y enviar el otro panel a una posición más atrás y así poder programar tranquilamente un panel y para poder trabajar sobre el otro solo tendría que cambiar la preferencia de orden.

Problema a la hora de meter un número después de que el lbl cálculo ya tenga ese número

He estado intentando replicar la función de la calculadora de windows que al meter un número darle a sumar me lo pone en la parte superior



El problema que mi función meter numero para hacer esta operación mediante los métodos obtenerNumerosLblCalcular() y obtenerNumerosLblNumero(), miro si son iguales estos números para borrarlos, pero claro esto me borrará siempre el lblnumero cuando el los dos números sean iguales creando un fatal error.

Después de estar pensando alguna manera de hacerlo se me ocurrió que podría a la hora de meter un número en la parte superior borrarlo directamente, pero esto para el usuario

final no sería intuitivo ya que es mejor tener en pantalla el número seleccionado anteriormente hasta que metas otro.

Así que mi solución fue crear una variable global booleana llamada borrar y que cuando se añadiese texto en la parte del lblCalculo ésta entrase en true y que a la hora de meter un número llamado a la función meterNumero() mirase si está true para directamente borrar todo el texto y poner solo en número arreglando el error

```
else if (obtenerNumerosLblCalcular() == obtenerNumerosLblNumero() && borrar == true)
```

Que el los botones de operaciones hicieran cálculos si les dabas 2 veces seguidas

Al hacer una operación por ejemplo 2×2 si le volviéramos a dar a la \times se multiplica y te da un resultado de 4 con la operación \times para hacer otra operación, lo cual está bien.

Pero el problema es que podíamos volver a darle a la \times haciendo esta operación $2 \times 2 \times \times$

Lo cual multiplicaba como si fuera un porcentaje dando 8 lo cual aunque pareciera funcional no lo era ya que para que una calculadora sea funcional al concatenar operaciones no se tenían que seguir multiplicando con darle al botón ya que el usuario podría querer cambiar de signo de operación y no podría hacerlo.

Así que mi solución ha sido poner aplicarle la variable booleana error que valida que después de un cambio se cambie algún número evitando errores

“Se hizo este cambio a parte de por lo dicho porque en una pequeña fase de testeos que hice causaba conflictos como por ejemplo al hacer cálculos consecutivos utilizando esta función y querer hacer una resta provoca que se haga otra operación y luego te permita hacer la resta $5 \times 5 \times = 25 \times$ (esto sale por pantalla y al darle a $-$ te saliera 625 $-$ ”

Se ve un botón del panel de historial en el panel de cálculo

Solución: en la vista tenemos que ocultar con `panel.setVisible(false)` y tenemos que hacer que no esté disponible con `Panel.setenable(false);`

Al hacer una operación y pulsar el botón igual, si volvemos a darle igual coger el resultado y multiplicarlo por el número anterior.

A la hora de implementar esta función me di cuenta que sería más difícil de lo que pensaba ya que tenía que sacar los números del lblCalculo y también tendría que hacer que se ejecutará solo cuando pulse igual y ya haya una operación

Para obtener el segundo número "42 x **13** =" vi que la solución más fácil era quitar el "=" de la operación y a partir de ahí hacer un lastIndexOf y sacar el número

```
String operacion = op.substring(0, op.indexOf("=")-1);
```

```
num2=Double.valueOf(operacion.substring(operacion.lastIndexOf(" "), operacion.length()));
```

también implemente el método setTextNumeroligual(numero1, numero2, operacion);

que hace el resto del código; prorratar, meter el texto en los lbl para evitar la duplicidad de código y que quede más bonito

Al meter porcentajes se pone infinito

Al meter un porcentaje y volver a buscar el porcentaje de ese número salía
0.0002000000000000002425352545546464664

Para arreglar esto he tenido que prorratar el bigDecimal

Crear un action listener en el menú

Para pasar de panelCalculo a panelHistorial he creado en el menú unos botones pero el problema es que quería meter un menú ítem para que al pulsar se fuera directamente no tener que crear un menú ítem en el menú que lo hace menos óptimo y funcional.

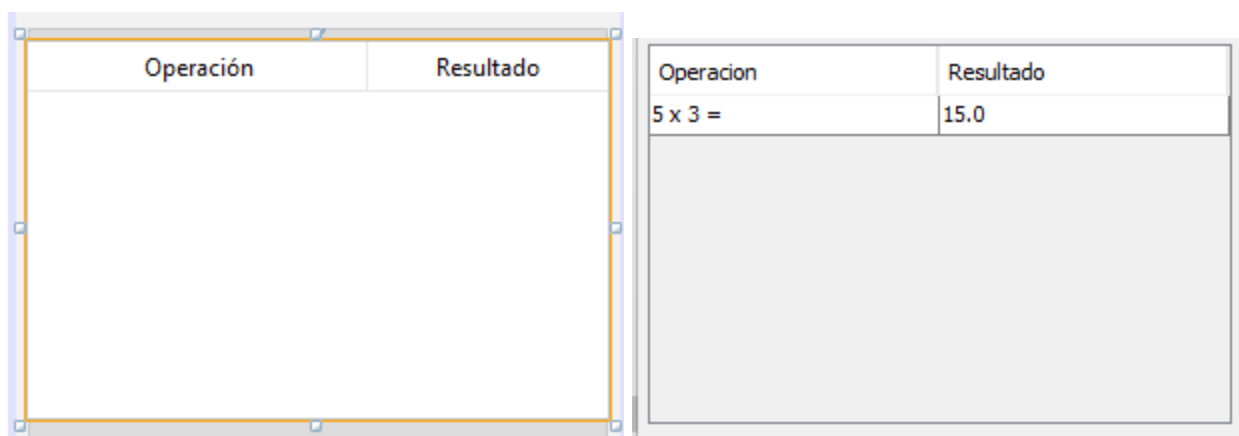
Al no poder crearlo decidí buscar como hacerlo y después de probar con varios tipos de eventos como action listener keyPressed menuSelected vi que mouseClicked funcionaba correctamente

Hacer la conexión a la base de datos

He tenido algunos problemas al acceder a la BBDD, primero y creado la conexión y he hecho pruebas ya que no entraba, y me salía un error inusual, ya que tenía todo el código bien, he estado revisando la contraseña, también he revisado si me faltaba algo para importar y nada, al final buscando he visto que el error era del portatil y he tenido que ir a administrador de tareas / servicios y allí buscar MySQL80 deshabilitarlo y activar MySQL desde el XAMPP

Configuración del JTable

Quiero cambiar el tamaño de las columnas y no me deja, he probado con muchas cosas pero el problema no es que lo ponga porque está bien puesto y en la vista se ve modificado, el problema es que al crearlo se pone por defecto



Operación	Resultado
5 x 3 =	15.0

Pref. Width: ▼

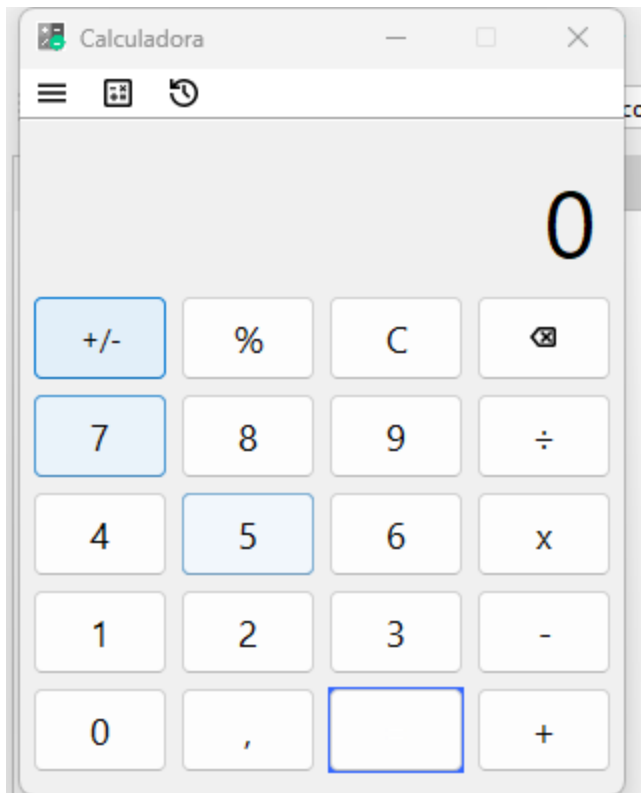
Min. Width: ▼

Max. Width: ▼

Poner el color a los botones y hacer el diseño

he puesto el color a un botón y me sale como se ve en la pantalla al estar buscando un buen rato el motivo me di cuenta que es del ui manager pero si cambiaba a otro o si lo

quitaba se quedaba muy feo el diseño así que opte por importar unas librerías que me cambian el diseño automáticamente haciendo que este problema se acabe



Añadiendo un `KeyListener`

he añadido un `KeyListener` para poder usarla solo con el teclado sin tener que darle a los botones pero no me lo detecta, he probado de todo durante un largo tiempo y nada hasta que en clase me di cuenta que era necesario tener un `JText` para poder tener un `KeyListener` y creando el `JText` funcionaba correctamente (también he tenido que crear un método que me posicione este `JText` cada vez que pulso un botón para que siga funcionando el `KeyListener` ya que si no está el cursor encima no funciona)

Código sacado de internet

quitar de una variable double el .0 si es entero

<https://es.stackoverflow.com/questions/62846/quitar-la-coma-en-un-double-android-studio>

Librerías

Jtatto

Diseños predefinidos importados de la librería:

<http://www.jtattoo.net/>

(No se aplican tal cual ya que modifíco el fondo y los labels para que se vea mas bonito)

se puede ver en el método disenyo(String nombre) del controlador