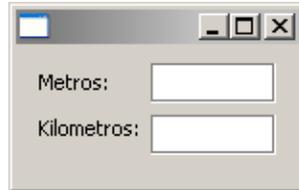




PRACTICO N° 16

EJERCICIO 1. Implemente una interfaz gráfica para un conversor de metros a kilómetros.



El usuario ingresa la distancia en metros y debe aparecer la distancia convertida a kilómetros en el rectángulo que se muestra justo debajo luego de apretar ENTER.

Para ver cómo lograr este comportamiento consultar la documentación de la clase JTextField en: <https://docs.oracle.com/javase/7/docs/api/javax/swing/JTextField.html>

EJERCICIO 2. Implemente la interfaz gráfica de un surtidor de combustible. Inicialmente solo están habilitados los botones Comprar y Recargar.

- El panel derecho muestra siempre el estado actual del surtidor. Los tipos de combustible se muestran a través etiquetas y los litros se muestran usando cuadros de texto no editables.
- Si el usuario elige Comprar se activa la selección de tipo de combustible y se desactivan Comprar y Recargar.
- Al presionar el botón Recargar se recargan todos los depósitos de combustible al máximo posible.
- Con los JRadioButton se elige con qué tipo de combustible se desea cargar.
- En el cuadro de texto <valor> se ingresa cuántos litros de desea comprar. Luego de ingresarse el valor se modifica el estado interno del surtidor, se habilitan los botones Comprar y Recargar y se deshabilita el resto.

Gui_Surtidor
<<atributos de instancia>> surtidor : Surtidor //botones, etiquetas, etc.
GuiSurtidor() ...

Surtidor
<<atributos de clase>> maximaCarga: entero
<<atributos de instancia>> cantGasoil: entero cantSuper: entero cantPremium: entero
<<Constructor>> Surtidor()
<<comandos>> llenarDepositoGasoil() llenarDepositoSuper() llenarDepositoPremium() extraerGasoil(litros: entero) extraerSuper(litros: entero) extraerPremium(litros: entero)
<<consultas>> obtenerLitrosGasoil(): entero obtenerLitrosSuper(): real obtenerLitrosPremium(): entero





Introducción a la Programación Orientada a Objetos DCIC - UNS 2019



EJERCICIO 3. Una estación de servicio dispone de n surtidores representados en una clase Playa que encapsula un arreglo con n elementos de clase Surtidor. La clase Playa brinda operaciones para llenar el depósito de Gasoil de todos los surtidores, llenar el depósito de Super de todos los surtidores, y llenar el de Premium de todos los surtidores.

Implemente una interface GUI_Playa con dos paneles, el panel superior muestra un objeto de clase JComboBox con los números de surtidor (1 a n). El segundo panel tiene el mismo diagramado que en el ejercicio anterior. Inicialmente las componentes del segundo panel no están visibles, recién cuando el usuario selecciona el surtidor, se recupera el surtidor de la playa, se hace visible el segundo panel con los botones Comprar y Recargar activos. Cuando se completa la operación, se vuelve a habilitar la selección de surtidor y los componentes gráficos del segundo panel dejan de estar visibles.

<p>Playa</p> <p>surtidores: Surtidor[] n: entero</p> <hr/> <p><<Constructor>> Playa (cap:entero)</p> <p><<Comandos>> LlenarDepositosGasoil() LlenarDepositosSuper() LlenarDepositosPremium()</p> <p><<Consultas>> obtenerSurtidor(p:entero):Surtidor obtenerCantidad():entero</p>	<p>GUI_Playa</p> <p>panelSuperior:JPanel panelInferior: JPanel comboNumeros: JComboBox comprar, recargar: JButton</p> <hr/> <p><<Constructor>> GUI_Playa (p: Playa)</p>
---	---

EJERCICIO 4. Modifique el ejercicio 1 para que cuando se cliquee en el cuadro de texto inferior, se borre su contenido (este efecto se logra capturando el evento mouseClicked sobre el JTextField). ¿Qué ocurre con las demás operaciones que provee la interfaz?

EJERCICIO 5. Dado el siguiente diagrama de clases

<p>ReservaPasaje</p> <p>codigo:String pasajero: String butaca:entero confirmada:boolean</p> <hr/> <p><<Constructor>> ReservaPasaje (c:Codigo, p:String, b : entero)</p> <p><<Comandos>> confirmar()</p> <p><<Consultas>> obtenerPasajero():String obtenerButaca() : entero obtenerConfirmada():boolean</p>	<p>PasajesReservados</p> <p>T [] ReservaPasaje cantPasajes:entero</p> <hr/> <p><<Constructor>> PasajesReservados(n:entero)</p> <p><<Comandos>> insertar(r:ReservaPasaje) eliminar(r:ReservaPasaje)</p> <p><<Consultas>> estaLlena():boolean hayReservas():boolean obtenerElemento(p:entero): ReservaPasaje recuperarElemento(p:String): ReservaPasaje</p>	<p>GUI_LA</p> <p>panelCaja, panelEtiqueta, panelBotones:JPanel texto:JTextField etiqueta:Jlabel bConf,bCanc,bSalir:JButton</p> <hr/> <p>GUI_LA(PR: PasajesReservados)</p>
--	---	--



Introducción a la Programación Orientada a Objetos DCIC - UNS 2019



Implemente las dos clases y una GUI que incluya tres paneles. En el primero aparece una caja de texto, en el segundo una etiqueta y en el tercero tres botones Confirmar, Anular y Salir, inicialmente deshabilitados.

Cuando el usuario completa el cuadro de texto con el código de una reserva y oprime enter, el oyente de la caja busca una reserva con ese código y muestra el nombre del pasajero y la butaca en la etiqueta del segundo panel. Si la reserva no está confirmada habilita los botones del tercer panel. Si la reserva está confirmada o no existe se muestran mensajes adecuados.

Si el usuario oprime el botón Confirmar se envía el mensaje confirmar() a la reserva. Si oprime Anular se elimina la reserva. Si oprime Salir se vuelve al estado inicial.

Implemente una clase LineaAerea que cree e inicialice un objeto PR de clase PasajesReservados con 8 pasajes y cree un objeto de clase GUI_LA con PR como parámetro.

EJERCICIO 6: ESTACIONAMIENTO

Dado el siguiente diagrama de clases:

Estacionamiento	GUI_Estacionamiento
Lugares: boolean[] cant:entero	Botones: JButton[] estacionarAuto: JButton estacionamiento: Estacionamiento
<<Constructor>> Estacionamiento (capacidad:entero) <<Comandos>> ocupar(i: entero) liberar(i:entero) <<Consultas>> capacidad(): entero estaOcupado(i:entero):boolean estaLleno():boolean primerDisponible():entero	<<Constructor>> GUI_Estacionamiento(e: Estacionamiento)

Se desea implementar una aplicación que facilite la administración de un estacionamiento. Para esto implemente las clases Estacionamiento y GUI_Estacionamiento. En la GUI, los lugares para estacionar se representan con botones, que cuando están habilitados indican que el lugar está ocupado y cuando están deshabilitados que el lugar está disponible. La GUI incluye dos paneles, en el primer panel aparecerán tantos botones como capacidad tenga el estacionamiento que se quiere representar. Inicialmente todos los botones estarán deshabilitados, indicando que están disponibles y mostrando una imagen similar a la de la figura.

En el segundo panel, se mostrará el botón "estacionarAuto", que al ser oprimidos simulará el estacionamiento de un auto en el primer lugar disponible. Al estacionar un auto en un lugar i, se habilitará el botón de la posición i cambiando la imagen del botón a una que indique que el lugar está ocupado. Si se llena el estacionamiento, el botón "estacionarAuto" se deshabilitará.

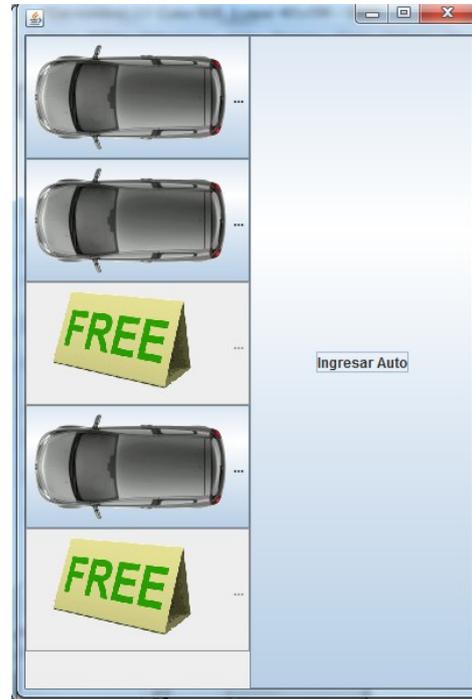
Al oprimir el botón correspondiente a un lugar ocupado, este lugar se liberará, deshabilitando el botón, cambiando la imagen, y habilitando el botón "estacionarAuto" si es que se encontraba deshabilitado.



Introducción a la Programación Orientada a Objetos DCIC - UNS 2019



Estado inicial: botones deshabilitados indicando los lugares disponibles.



Lugares ocupados intercalados con lugares disponibles