Libreria Grafica SwingGL

MANUALE D'USO

Autore : Giovanni Alestra

Introduzione

La libreria grafica SwingGL permette la creazione di interfacce grafiche in linguaggio Java. L'approccio della libreria rispetto a quelle utilizzate in ambito professionale è un approccio didattico all'argomento. Si vuole consentire facilmente di creare applicazioni desktop con interfacce grafiche attraverso una più semplice gestione dei componenti grafici e degli eventi cui questi componenti sono soggetti nell'interazione con l'utente. La libreria è composta da classi che estendono le classi dei componenti grafici del package grafico Swing, da classi di supporto e da una classe principale. Per creare una interfaccia grafica non è necessario conoscere tutte le classi che compongono la libreria ma è necessario conoscere solamente l'utilizzo delle funzioni che la classe principale, la classe SwingGL, mette a disposizione. Per la gestione degli eventi è necessario avere il concetto di Interfaccia tipico del linguaggio Java. La gestione degli eventi risulta essere semplificata.
La seguente guida ha lo scopo di rendere abile lo sviluppatore nella creazione di interfacce grafiche per applicazioni desktop anche se questi non ha un background sull'argomento.

INDICE

Aggiungere la Libreria al tuo Progetto

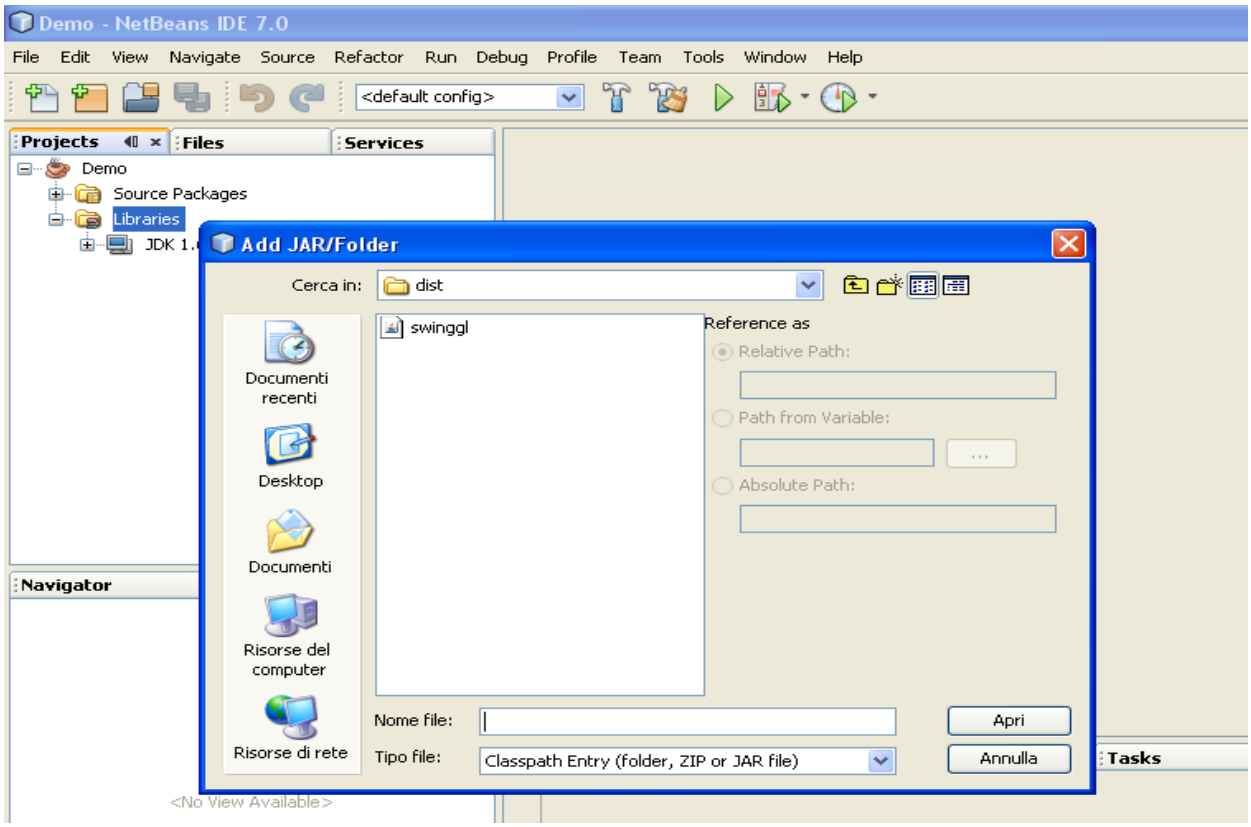
Per aggiungere la libreria al tuo progetto se si stà sviluppando in un ambiente di sviluppo quale Netbeans o Eclipse è necessario seguire i seguenti passi.

- Procurarsi il file “swinggl.jar”

Per prima cosa dobbiamo disporre del file swinggl.jar che contiene tutte le classi della libreria. A questo punto bisogna aggiungere questo file alle librerie.

Aggiungere la Libreria in Netbeans

Se si stà utilizzando netbeans, bisogna cliccare sulla cartella “Libraries” del pannello progetto e successivamente cliccare su “Add Jar/Folder”.

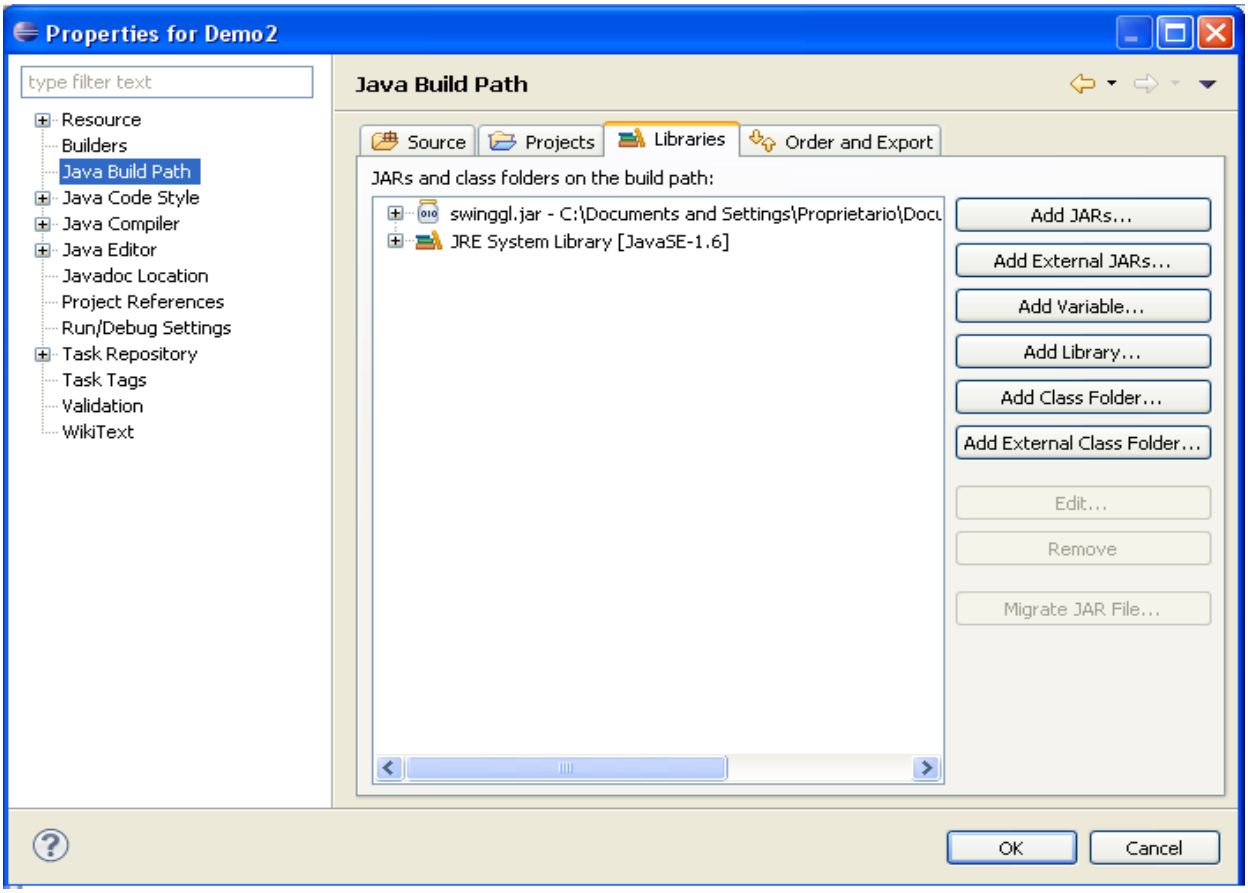


E caricare il file tra le librerie. Dopo questa operazione sarà possibile prima della definizione delle classi del nostro progetto importare il package “swinggl” tramite l'operazione :

import swinggl.*

Aggiungere la Libreria in Eclipse

Attraverso un procedimento del tutto simile a quello visto è possibile aggiungere la libreria nell'ambiente di sviluppo Eclipse. Se disponiamo del file “swinggl.jar” dobbiamo selezionare “properties” dal menù “project”, cliccare su “Java Build Path” e successivamente su “Add External JARs”.



A questo punto prima della definizione delle classi sarà possibile importare il package con :

import swinggl.*;

Creare una Interfaccia Grafica

Per creare una interfaccia grafica basta istanziare un oggetto della classe principale, la classe SwingGL scrivendo all'interno di una funzione :

SwingGL swgl = new SwingGL();

ESEMPIO :

```
import swinggl.*;

public class Demo {

   public static void main(String [ ] args) {
      SwingGL swgl = new SwingGL();
   }

}
```

A questo punto tramite l'oggetto di tipo SwingGL e le sue funzioni saremo in grado di creare una interfaccia grafica.

Creare una Finestra

Per creare una finestra è possibile utilizzare la funzione della classe principale **SwingGL.newWindow**, esplicitando il nome (una Stringa) che vogliamo dare alla finestra per poterci riferire ad essa in futuro, ed anche il titolo, la posizione, la larghezza e l'altezza.

newWindow

```
public void newWindow(java.lang.String name,
                      java.lang.String title,
                      int x,
                      int y,
                      int width,
                      int height)
```

Permette la creazione e visualizzazione di una finestra

Parameters:

`name` - Stringa contenente il nome che si vuole assegnare alla finestra
`title` - Stringa che indica il titolo che verrà visualizzato nella finestra
`x` - Posizione X nello schermo
`y` - Posizione Y nello schermo
`width` - Larghezza della finestra
`heightAltezza` - della finestra

ESEMPIO:

```
import swinggl.*;

public class Demo {

   public static void main(String [ ] args) {
     SwingGL swgl = new SwingGL();
     try {
      swgl.newWindow("Main", "Hello World!", 100, 100, 200, 200);
         } catch ( SwingGLException e ) { System.out.println ( e.toString() ); }
   }
 }
```

RISULTATO ESEMPIO:

Aggiungere un Panello ad una Finestra

Dopo aver creato una finestra se vogliamo in essa visualizzare dei componenti dobbiamo aggiungere un pannello. Per far ciò possiamo utilizzare la funzione **SwingGL.newPanel**, la quale prevede di esplicitare : il nome (una Stringa) che si vuole dare al pannello, il nome del contenitore (tipicamente il nome della finestra), la larghezza del pannello e l'altezza. Se il pannello ha dimensioni maggiori di quelle della finestra verranno visualizzate in modo automatico dell scrollbars.

newPanel

```
public void newPanel(java.lang.String name,
                     java.lang.String container,
                     int width,
                     int height)
```

Permette di visualizzare un pannello sul quale creare altri oggetti grafici

Parameters:

`name` - Stringa che contine il nome che si vuole assegnare al pannello
`container` - Contenitore del pannello tipicamente una finestra
`width` - Esprime la Larghezza del Pannello in pixel
`height` - Esprime l'Altezza del Pannello in pixel

ESEMPIO :

```
import swinggl.*;

public class Demo {

  [..]
 try {
     swgl.newWindow("Main", "Hello World!", 100, 100, 200, 200);
     swgl.newPanel("Panel", "Main", 500, 500);
   } catch ( SwingGLException e ) { System.out.println ( e.toString() ); };
 }
```

RISULTATO ESEMPIO:

Aggiungere una Etichetta ad un Pannello

Dopo aver creato un pannello possiamo visualizzare in esso molti componenti grafici. Ora trattiamo la visualizzazione delle etichette. Per visualizzare una etichetta possiamo utilizzare la funzione **SwingGL.newLabel**, la quale prevede di esplicitare : il nome (una Stringa) che si vuole dare al componente, il nome del contenitore (tipicamente il nome di un pannello), il testo, la posizione x ed y, la larghezza del pannello e l'altezza.

newLabel

```
public void newLabel(java.lang.String name,
                     java.lang.String container,
                     java.lang.String text,
                     int x,
                     int y,
                     int width,
                     int height)
```

Cosente la visualizzazione di una etichetta ed il suo posizionamento relativo al contenitore

Parameters:

`name` - Stringa contenente il nome che si vuole assegnare all'etichetta
`container` - Stringa che indica il nome del contenitore tipicamente un pannello
`text` - Stringa Testo dell'etichetta
`x` - Posizione X relativa al contenitore
`y` - Posizione Y relativa la contenitore
`width` - Larghezza dell'etichetta
`height` - Altezza dell'etichetta

ESEMPIO :

[...]

```
swgl.newPanel("Panel", "Main",500,500);
swgl.newLabel("Label", "Panel", "Hello world!", 10, 10, 200, 20);
```

[...]

RISULTATO ESEMPIO:

Aggiungere un TextField ad un Pannello

I text field sono campi nei quali si può scrivere del testo. Per inserirli possiamo usare la funzione **SwingGL.newTextField** esplicitando il nome da dare al componente, il contenitore tipicamente un pannello, il testo di default, la posizione x ed y, la larghezza e l'altezza.

newTextField

```
public void newTextField(java.lang.String name,
                         java.lang.String container,
                         java.lang.String text,
                         int x,
                         int y,
                         int width,
                         int height)
```

Permette la visualizzazione di un Text Field

Parameters:

`name` - Nome da dare al componente
`container` - Nome del contenitore tipicamente un pannello
`text` - Testo di default
`x` - Posizione X
`y` - Posizione Y
`width` - Larghezza del componente
`height` - Altezza del Componente

ESEMPIO :

[...]

swgl.newPanel("Panel", "Main",500,500);

swgl.newTextField("TextField", "Panel", "Hello world!", 10, 10, 200, 20);

[...]

RISULTATO ESEMPIO :

Aggiungere una Text Area ad un Pannello

E' possibile aggiungere una Text Area ad un pannello attraversa la funzione **SwingGL.newTextArea** esplicitando il nome che si vuole dare al componente, il nome del contenitore tipicamente un pannello, il testo di default, la posizione x ed y, la larghezza e l'altezza del componente.

newTextArea

```
public void newTextArea(java.lang.String name,
                        java.lang.String container,
                        java.lang.String text,
                        int x,
                        int y,
                        int width,
                        int height)
```

Permette la visualizzazione di una Text Area

Parameters:

`name` - Nome da dare al componente
`container` - Nome del contenitore tipicamente un pannello
`text` - Testo di default
`x` - Posizione X
`y` - Posizione Y
`width` - Larghezza del componente
`height` - Altezza del Componente

ESEMPIO :

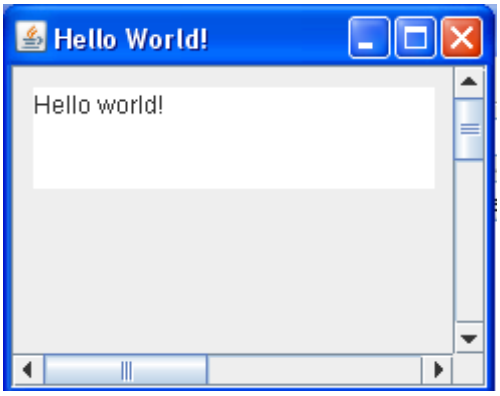
[...]

swgl.newPanel("Panel", "Main",500,500);

swgl.newTextArea ("TextArea", "Panel", "Hello world!", 10, 10, 200, 50);

[…]

RISULTATO ESEMPIO :



Aggiungere un Campo Password ad un Pannello

Per aggiungere un campo Password si può utilizzare la funzione **SwingGL.newPasswordField**, esplicitando il nome del componente, il nome del contenitore tipicamente un pannello, la posizione x ed y, la larghezza e l'altezza del componente.

newPasswordField

```
public void newPasswordField(java.lang.String name,
                             java.lang.String container,
                             int x,
                             int y,
                             int width,
                             int height)
```

Permette la visualizzazione di un campo Password

Parameters:

`name` - Nome da dare al componente
`container` - Nome del contenitore tipicamente un pannello
`x` - Posizione x
`y` - Posizione y
`width` - Larghezza del componente
`height` - Altezza del componente

ESEMPIO :

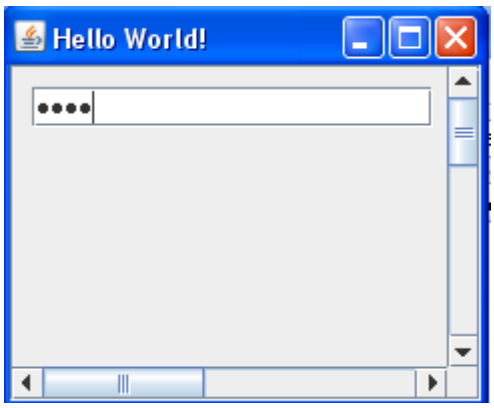
[...]

swgl.newPanel("Panel", "Main",500,500);

swgl.newPasswordField ("Password", "Panel", 10, 10, 200, 20);

[…]

RISULTATO ESEMPIO :



Aggiungere un ComboBox ad un Pannello

Per aggiungere un combobox ad un pannello possiamo utilizzare la funzione **SwingGL.newComboBox**, esplicitando il nome del componente, il nome del contenitore tipicamente un pannello, un vettore di stringhe che rappresentano i valori selezionabili, la posizione x ed y, la larghezza e l'altezza del componente.

newComboBox

```
public void newComboBox(java.lang.String name,
                        java.lang.String container,
                        java.lang.String[] items,
                        int x,
                        int y,
                        int width,
                        int height)
```

Permette la visualizzazione di un ComboBox

Parameters:

`name` - Nome da dare al Componente
`container` - Nome del contenitore tipicamente un pannello
`items` - Lista dei valori selezionabili
`x` - Posizione x
`y` - Posizione y
`width` - Larghezza del componente
`height` - Altezza del componente

ESEMPIO :

[…]

```
swgl.newPanel("Panel", "Main",500,500);
String[] Items = { "Bird", "Cat", "Dog", "Rabbit", "Pig" };
swgl.newComboBox("ComboBox", "Panel", Items, 10, 10, 100, 20);
```

[…]

RISULTATO ESEMPIO :

Aggiungere un CheckBox ad un Pannello

Per aggiungere un checkbox possiamo utilizzare la funzione **SwingGL.newCheckBox** esplicitando il nome del componente, il nome del contenitore tipicamente un pannello, il testo inerente alla selezione, la posizione x ed y, la larghezza, l'altezza ed il valore di default.

newCheckBox

```
public void newCheckBox(java.lang.String name,
                        java.lang.String container,
                        java.lang.String text,
                        int x,
                        int y,
                        int width,
                        int height,
                        boolean selected)
```

Permette di visualizzare un CheckBox

Parameters:

`name` - Nome del componente
`container` - Nome del contenitore tipicamente un pannello
`text` - Testo inerente alla selezione
`x` - Posizione x
`y` - Posizione y
`width` - Larghezza del componente
`height` - Altezza del componente
`selected` - Selezionato vero/falso

ESEMPIO:

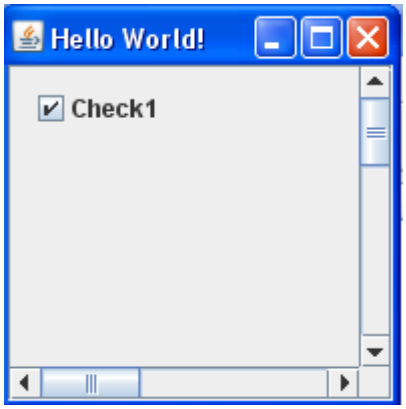
[…]

swgl.newPanel("Panel", "Main",500,500);

swgl.newCheckBox("CheckBox","Panel", "Check1", 10, 10, 100, 20, true);

[…]

RISULTATO ESEMPIO :



Aggiungere dei RadioButton ad un Pannello

Prima di aggiungere dei radiobutton dobbiamo creare un button group mediante la funzione **SwingGL.newButtonGroup :**

newButtonGroup

```
public void newButtonGroup(java.lang.String name)
```

Successivamente possiamo creare i radio button mediante la funzione :

newRadioButton

```
public void newRadioButton(java.lang.String name,
                           java.lang.String container,
                           java.lang.String Group,
                           java.lang.String text,
                           int x,
                           int y,
                           int width,
                           int height,
                           boolean selected)
```

Permette la visualizzazione di un radio button associato ad un button group

Parameters:

`name` - Nome del componenete
`container` - Nome del contenitore tipicamente un pannello
`Group` - Nome del button group
`text` - Testo inerente alla selezione
`x` - Posizione x
`y` - Posizione y
`width` - Larghezza del componente
`height` - Altezza del componente
`selected` - Selezionato Vero/Falso

ESEMPIO E RISULTATO:

swgl.newPanel("Panel", "Main",500,500);

swgl.newButtonGroup("BG");

swgl.newRadioButton("RButton", "Panel", "BG","Uno", 10, 10, 80, 20, true);

swgl.newRadioButton("RButton2", "Panel", "BG","Due", 10, 30, 80, 20, false);

Aggiungere un TabbedPane ad un Pannello

Per aggiungere un TabbedPane possiamo utilizzare la funzione **SwingGL.newTabbedPane**, esplicitando il nome che si vuole dare al tabbedpane, il nome del contenitore tipicamente un pannello, la posizione x ed y, la larghezza e l'altezza del tabbedpane.

newTabbedPane

```
public void newTabbedPane(java.lang.String name,
                          java.lang.String container,
                          int x,
                          int y,
                          int width,
                          int height)
```

Permette la visualizzazione di un TabbedPane

Parameters:

`name` - Nome da dare al componente
`container` - Nome del contenitore tipicamente un pannello
`x` - Posizione x
`y` - Posizione y
`width` - Larghezza del componente
`height` - Altezza del componente

ESEMPIO :

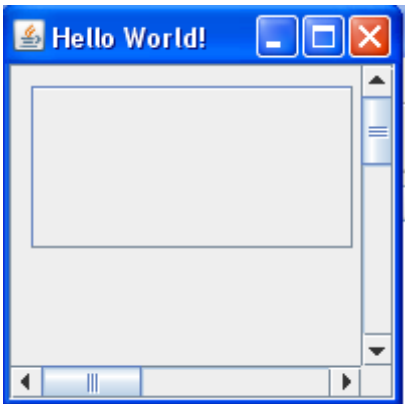
[...]

swgl.newPanel("Panel", "Main",500,500);

swgl.newTabbedPane("TPane", "Panel",10, 10, 160, 80);

[…]

RISULTATO ESEMPIO:



Aggiungere una Tab ad un TabbedPane

Per aggiungere una o più tab ad un tabbedpane possiamo utilizzare la funzione **SwingGL.addTabToTabbedPane**, esplicitando il nome del tabbedpane, il nome da dare al nuovo pannello che si stà creando ed il titolo del pannello.

addTabToTabbedPane

```
public void addTabToTabbedPane(java.lang.String tabbedpane,
                               java.lang.String name,
                               java.lang.String title)
```

Permette di aggiugere un pannello ad un TabbedPane

Parameters:

`tabbedpane` - Nome del TabbedPane
`name` - Nome del pannello che si vuole aggiungere
`title` - Titolo del pannello

ESEMPIO:

[…]

```
swgl.newPanel("Panel", "Main",500,500);
swgl.newTabbedPane("TPane", "Panel",10, 10, 160, 80);
swgl.addTabToTabbedPane("TPane", "LibrettoTab", "Libretto");
swgl.addTabToTabbedPane("TPane", "TesiTab", "Tesi");
```

[…]

RISULTATO ESEMPIO:

Aggiungere un Bottone ad un Pannello

Per aggiungere un bottone ad un pannello possiamo utilizzare la funzione **SwingGL.newButton**, esplicitando il nome da dare al componente, il nome del contenitore tipicamente un pannello, il testo del bottone, la posizione x ed y, la larghezza, l'altezza e la classe gestore dell'evento click. Se non è previsto un gestore dell'evento scriveremo **null**.

newButton

```
public void newButton(java.lang.String name,
                      java.lang.String container,
                      java.lang.String text,
                      int x,
                      int y,
                      int width,
                      int height,
                      SwingGLEvent e)
```

Consente la creazione e visualizzazione di un Bottone su cui si può cliccare

Parameters:

`name` - Stringa contenente il nome che si vuole assegnare al Bottone
`container` - Stringa contente il nome del contenitore tipicamente un pannello
`text` - Testo del Bottone
`x` - Posizione X relativa al contenitore
`y` - Posizione Y relativa al contenitore
`width` - Larghezza del Bottone
`height` - Altezza del bottone
`e` - Classe che implementa SwingGLEvent per la gestione dell'evento Click settabile su null se non si vuole gestire l'evento

ESEMPIO :

[…]

```
swgl.newPanel("Panel", "Main",500,500);
swgl.newButton("Button", "Panel", "OK", 10, 10, 30, 20, null);
```

[…]

RISULTATO ESEMPIO :

Scrivere un Gestore dell'Evento Click per un Bottone

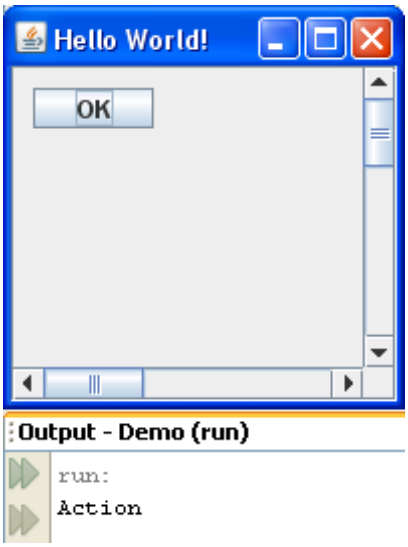
Per scrivere un gestore dell'evento click per un bottone è necessario scrivere una classe che implementi la classe **SwingGLEvent,** istanziare un oggetto di tale classe e passarlo alla funzione **SwingGL.newButton**. Un semplice gestore dell'evento click potrebbe stampare sul prompt dei comandi una stringa. Ecco un esempio :

```
public class Click implements SwingGLEvent {
   public void Event ( String ActionCommand ) {
      System.out.println ( "Action" );
   }
}
```

E scrivere nella funzione che richiama **newButton :**

```
[...]
        swgl.newPanel("Panel", "Main",500,500);
        Click click = new Click();
        swgl.newButton("Button", "Panel", "OK", 10, 10, 60, 20, click);
[…]
```

RISULTATO ESEMPIO :



Consiglio Operativo

Al fine di poter avere accesso all'interfaccia grafica quando si gestisce un evento è opportuno che la classe gestore dell'evento abbia un riferimento all'oggetto **SwingGL** e che il riferimento a questo oggetto venga settato dal costruttore della classe.

ESEMPIO :

```
import swinggl.*;
public class Click implements SwingGLEvent {
   SwingGL swgl;
   public Click (SwingGL swinggl ) {
      swgl = swinggl;
   }

   public void Event ( String ActionCommand ) {
      System.out.println ( "Action" );
      try {
         // swgl. ...
         } catch ( SwingGLException e ) { System.out.println ( e.toString() ); };
}
}
```

E nella funzione che crea l'oggetto :

```
SwingGL swgl = new SwingGL();
Click click = new Click(swgl);
```

In questo modo possiamo avere accesso all'interfaccia durante la gestione degli eventi.

Aggiungere i Menù ad una Finestra

Per aggiungere una menù bar ad una finestra, dobbiamo utilizzare la funzione **SwingGL.AddMenuBar** esplicitando il nome del componente ed il nome della finestra.

addMenuBarToWindow

```
public void addMenuBarToWindow(java.lang.String name,
                               java.lang.String window)
```

Aggiunge una menù bar ad una finestra

Parameters:

`name` - Nome da dare al componente
`window` - Nome della finestra

Per aggiungere i menù ad una menùbar dobbiamo utilizzare la funzione **SwingGL.AddMenuToWindow** esplicitando il nome della menùbar, il nome del menù, i campi visualizzati del menù ed il gestore dell'evento click.

addMenuToWindow

```
public void addMenuToWindow(java.lang.String menubar,
                            java.lang.String name,
                            java.lang.String[] menuItems,
                            SwingGLEvent e)
```

Aggiunge un menù alla finestra

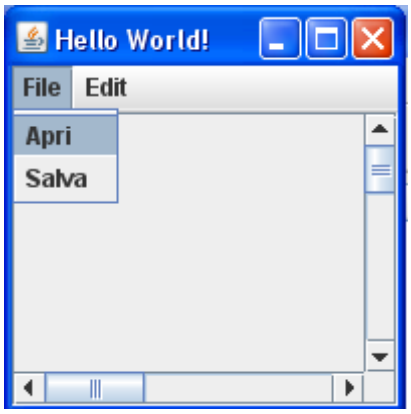
Parameters:

`menubar` - Nome della menùbar
`name` - Nome del menu che si stà aggiungendo
`menuItems` - Campi del menù
`e` - Gestore dell'evento click sul menù

ESEMPIO :

```
[..]
swgl.newWindow("Main", "Hello World!", 100, 100, 200, 200);
Click click = new Click(swgl);
String[] Items = { "Apri", "Salva" };
String[] Items2 = { "Copia" };
swgl.addMenuBarToWindow("Menu", "Main");
swgl.addMenuToWindow("Menu", "File", Items, click );
swgl.addMenuToWindow("Menu", "Edit", Items2, click );
```

RISULTATO ESEMPIO :



Per la gestione degli eventi, nella funzione della classe che implementa **SwingGLEvent** bisogna con un **if** seguito da un **equals**, gestire l'evento relativo al menù che è stato realmente cliccato, analizzando la stringa **ActionCommand** che ritorna il testo del menuItem che è stato cliccato.

```
import swinggl.*;

public class MenuClick implements SwingGLEvent {

   SwingGL swgl;

   public MenuClick (SwingGL swinggl ) {
      swgl = swinggl;
   }

   public void Event ( String ActionCommand ) {
     if (ActionCommand.equals("Apri")) { /* try … swgl. … catch */ };
   }
}
```

Visualizzare Open/Save File Dialog

Per visualizzare una openfiledialog e per visualizzare una savefiledialog è possibile utilizzare le funzioni : **SwingGL.ShowOpenFileDialog** e la funzione **SwingGL.ShowSaveFileDialog**.

ShowOpenFileDialog

```
public java.lang.String ShowOpenFileDialog(java.lang.String title)
```

Visualizza una OpenFileDialog

Parameters:
`title` - Titolo della finestra
Returns:
String Nome del file selezionato

ShowSaveFileDialog

```
public java.lang.String ShowSaveFileDialog(java.lang.String title)
```

Visualizza una FileSaveDialog

Parameters:
`title` - Titolo della finestra di dialogo
Returns:
String Nome del file sul quale salvare salvare

ESEMPIO :

```
[…]
    String file = swgl.ShowOpenFileDialog("Apri");
    System.out.println (file);
    String file2 = swgl.ShowSaveFileDialog("Salva");
    System.out.println (file2);
[…]
```

Visualizzare Message ed Input Dialogs

Per visualizzare un messaggio possiamo utilizzare la funzione **SwingGL.ShowMessageDialog**, passando alla funzione il messaggio da visualizzare.

ShowMessageDialog

```
public void ShowMessageDialog(java.lang.String message)
```

Permette la visualizzazione di un messaggio

Parameters:

`message` - Messaggio da visualizzare

Per visualizzare una Input Dialog possiamo utilizzare la funzione **SwingGL.ShowInputDialog**, fornendo la domanda cui l'utente deve rispondere.

ShowInputDialog

```
public java.lang.String ShowInputDialog(java.lang.String message)
```

Permette la visualizzazione di una finestra di input

Parameters:

`message` - Messaggio da visualizzare

Returns:

Stringa di risposta da parte dell'utente

Ottenere il testo di un TextField o TextArea

Per ottenere il testo di una text field o text area tipicamente durante la gestione di un evento possiamo utilizzare la funzione **SwingGL.GetText.**

GetText

public java.lang.String **GetText**(java.lang.String name)

Applicabile a TextField e TextArea ritorna una stringa contenente il testo digitato sul componente

Parameters:
name - Nome del Componente
Returns:
Testo Digitato

Ottenere il testo selezionato in un ComboBox

Per ottenere il testo selezionato in un combo box tipicamente durante la gestione di un evento possiamo utilizzare la funzione **SwingGL.GetSelection**.

GetSelection

public java.lang.String **GetSelection**(java.lang.String name)

Applicabile a ComboBox ritorna il testo selezionato

Parameters:
name - Nome del Componenete
Returns:
Testo Selezionato

Verificare se una CheckBox o un Radio Button è stato Selezionato

Per verificare se una CheckBox o un Radio Button è stato selezionato possiamo utilizzare la funzione **SwingGL.GetSelected**.

GetSelected

public boolean **GetSelected**(java.lang.String name)

Applicabile a CheckBox e RadioButton ritorna se l'elemento è stato selezionato

Parameters:
name - Nome del Componente
Returns:
Booleano Vero/Falso

Settare il testo di un textfield o textarea

Per settare il testo ti un text field o text area possiamo utilizzare la funzione **SwingGL.SetText.**

SetText

public void **SetText**(java.lang.String name,
java.lang.String text)

Applicabile a TextField e TextArea setta il testo del componente

Parameters:
name - Nome del componente
text - testo da settare

Settare la selezione di un combobox

Per settare il la selezione di un combo box possiamo utilizzare la funzione **SwingGL.SetSelection**.

SetSelection

public void **SetSelection**(java.lang.String name,
int index)

Applicabile a ComboBox setta il testo della possibile selezione

Parameters:
name - Nome del componente
index - Intero che indica la posizione del testo da settare nel vettore String[] Items

Settare la selezione di una CheckBox o Radio Button

Per settare il la selezione di una CheckBox o di un Radio Button possiamo utilizzare la funzione **SwingGL.SetSelected.**

SetSelected

public void **SetSelected**(java.lang.String name,
boolean value)

Applicabile a CheckBox e RadioButton setta la selezione

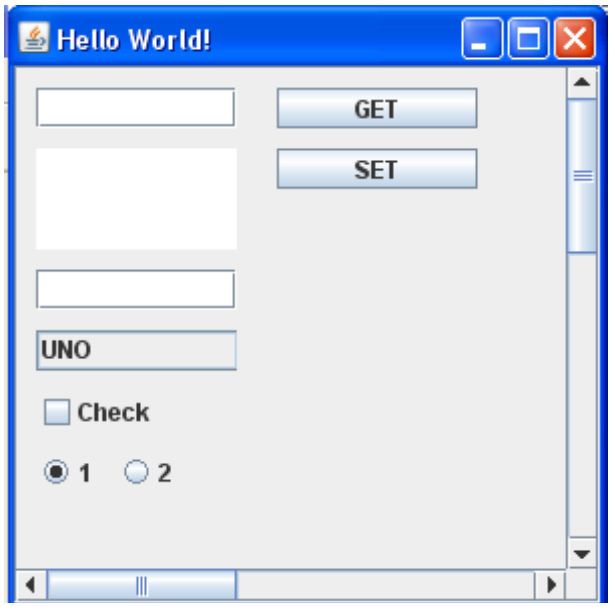
Parameters:
name - Nome del Componente
value - Booleano Vero/Falso

Esempio Completo Funzioni di Get e Set

```
import swinggl.*;

public class Demo {

   public static void main(String [ ] args) {
     SwingGL swgl = new SwingGL();
    try {
     swgl.newWindow("Main", "Hello World!", 100, 100, 300, 300);
     swgl.newPanel("Panel", "Main",700,700);

     swgl.newTextField("TextField", "Panel", "", 10, 10, 100, 20 );
     swgl.newTextArea("TextArea", "Panel", "", 10, 40, 100, 50);
     swgl.newPasswordField("Password", "Panel", 10, 100, 100, 20);
     String[] Items = { "UNO" , "DUE" };
     swgl.newComboBox("Combo", "Panel", Items, 10, 130, 100, 20);
     swgl.newCheckBox("Check", "Panel", "Check", 10, 160, 100, 20, false);
     swgl.newButtonGroup("BG");
     swgl.newRadioButton("Radio1", "Panel", "BG", "1", 10, 190, 50, 20, true);
     swgl.newRadioButton("Radio2", "Panel", "BG", "2", 50, 190, 50, 20, false);
```

```
      ClickGet Get = new ClickGet (swgl);
      ClickSet Set = new ClickSet (swgl);

      swgl.newButton("GET", "Panel", "GET", 130, 10, 100, 20, Get);
      swgl.newButton("SET", "Panel", "SET", 130, 40, 100, 20, Set);
     } catch ( SwingGLException e ) {
        System.out.println(e.toString());
     }
   }
 }

import swinggl.*;

public class ClickGet implements SwingGLEvent {

   SwingGL swgl;

   public ClickGet (SwingGL swinggl ) {
      swgl = swinggl;
   }

   public void Event ( String ActionCommand ) {
     try {
      String textfield = swgl.GetText("TextField");
      String textarea = swgl.GetText("TextArea");
      char[] password = swgl.GetPassword("Password");
      String combobox = swgl.GetSelection("Combo");
      boolean checkbox = swgl.GetSelected("Check");
      boolean radiobutton1 = swgl.GetSelected("Radio1");
      boolean radiobutton2 = swgl.GetSelected("Radio2");

      System.out.println (textfield);
      System.out.println (textarea);
      System.out.println (password);
      System.out.println (combobox);
      System.out.println (checkbox);
```

```
        System.out.println (radiobutton1);
        System.out.println (radiobutton2);
       } catch ( SwingGLException e ) { System.out.println ( e.toString()); }
    }
}

import swinggl.*;

public class ClickSet implements SwingGLEvent {

    SwingGL swgl;

    public ClickSet (SwingGL swinggl ) {
        swgl = swinggl;
    }

    public void Event ( String ActionCommand ) {
    try {
        swgl.SetText("TextField", "Setted");
        swgl.SetText("TextArea", "Setted");
        swgl.SetSelection("Combo",1);
        swgl.SetSelected("Check", true);
        swgl.SetSelected("Radio2",true);
    } catch ( SwingGLException e ) { System.out.println ( e.toString()); };
    }
}
```


Hello World!
GET
SET
UNO
Check
1
2

Uso della Funzione GetObject

Al fine di poter sfruttare tutte le potenzialità di Swing, la classe **SwingGL,** mette a disposizione pubblicamente la funzione **SwingGl.GetObject.**

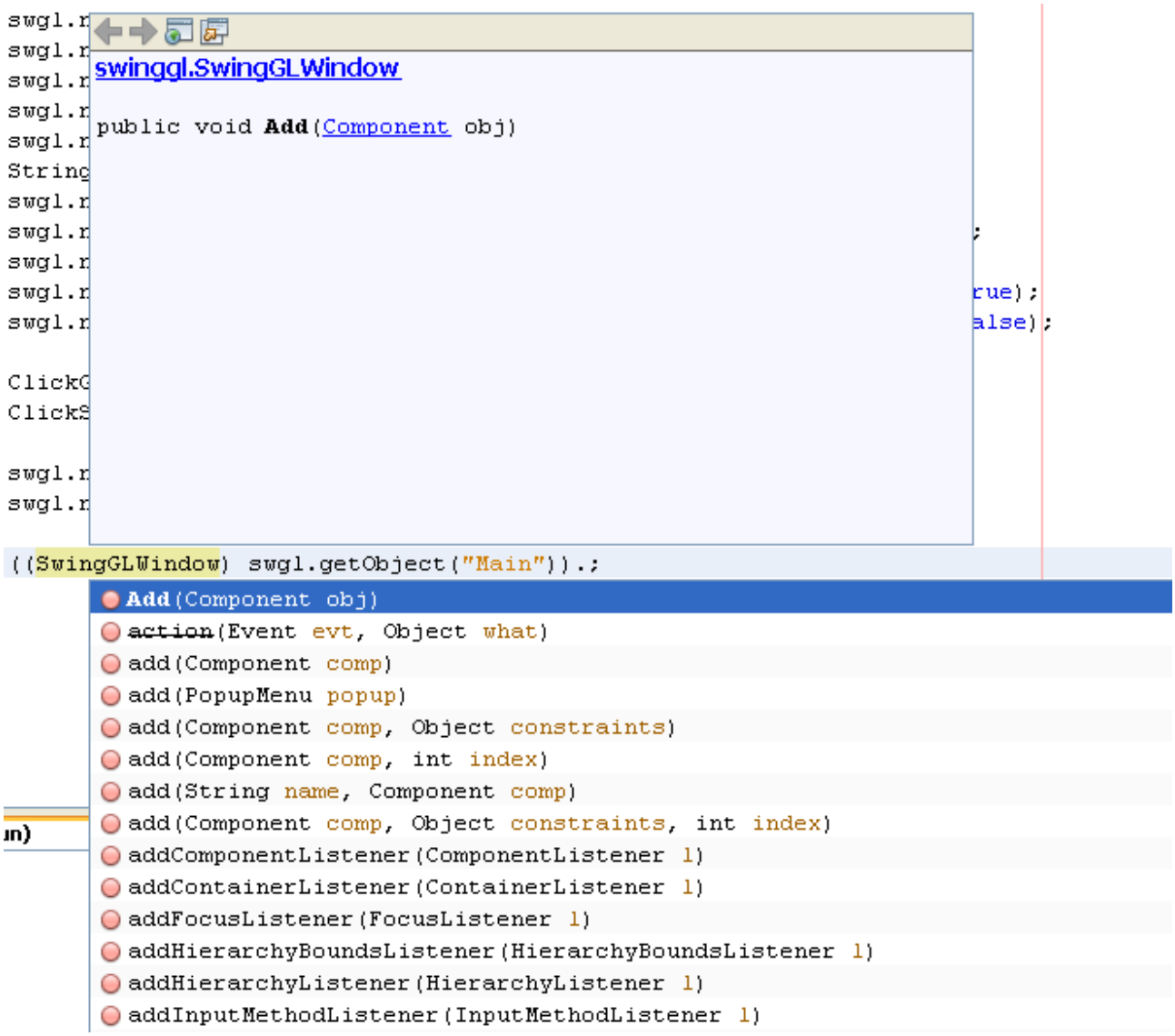
getObject

public java.lang.Object **getObject**(java.lang.String name)

Restituisce un riferimento di tipo Object all'oggetto (componente grafico) a cui ci si riferisce mediante una stringa che è il nome dell'oggetto

Parameters:
name -
Returns:
Object

La funzione permette di ottenere l'oggetto (componente grafico) dal suo nome. Dopo un cast alle classi della libreria, o a quelle di swing è possibile utilizzare tutte le funzioni di Swing sul componente. L'immagine della seguente pagina mostra il concetto. Per la documentazione sulle classi della libreria vedere il JavaDoc.


swinggl.SwingGLWindow
public void Add(Component obj)
((SwingGLWindow) swgl.getObject("Main")).;
Add(Component obj)
action(Event evt, Object what)
add(Component comp)
add(PopupMenu popup)
add(Component comp, Object constraints)
add(Component comp, int index)
add(String name, Component comp)
add(Component comp, Object constraints, int index)
addComponentListener(ComponentListener l)
addContainerListener(ContainerListener l)
addFocusListener(FocusListener l)
addHierarchyBoundsListener(HierarchyBoundsListener l)
addHierarchyListener(HierarchyListener l)
addInputMethodListener(InputMethodListener l)

Aggiungere un Desktop Pane ad una Finestra

Per creare un contenitore di finestre che tipicamente rappresentano documenti bisogna aggiungere un Desktop Pane ad una finestra. Possiamo farlo utilizzando la funzione della classe **SwingGL**:

AddDesktopPaneToWindow

public void **AddDesktopPaneToWindow**(java.lang.String name,
java.lang.String window)
throws SwingGLException

Aggiunge un DeskTop Pane alla Finestra

Parameters:
name - Nome del Desktop Pane
window - Nome della Finestra

Aggiungere un Internal Frame ad un Desktop Pane

Per creare una finestra docuemento, ovvero un internal frame possiamo utilizzare la funzione della classe **SwingGL** :

newInternalFrame

public SwingGL **newInternalFrame**(java.lang.String name,
java.lang.String desktoppane,
java.lang.String title,
int x,
int y,
int width,
int height)
throws SwingGLException

Aggiunge un InternalFrame ad un Desktop Pane

Parameters:
name - Nome dell'Internal Frame
desktoppane - Nome del Desktop Pane
title - Titolo dell'Internal Frame
x - Posizione x
y - Posizione y
width - Larghezza del Frame
height - Altezza del Frame
Returns:
Oggetto di tipo SwingGL

Ottenere l'oggetto SwingGL di un Internal Frame

Per ottenere l'oggetto **SwingGL** di un internal frame possiamo utilizzare la funzione della classe **SwingGL :**

getGraphicInteface

public SwingGL **getGraphicInteface**(java.lang.String internalFrame)
throws SwingGLException

Restituisce l'oggeto SwingGL di un Internal Frame

Parameters:
internalFrame - Nome dell'Internal Frame
Returns:

Per comprendere come utilizzare i Desktop Pane e gli Internal Frame studiare l'esempio qui proposto :

Esempio Funzioni di Get e Set con Internal Frames

```
import swinggl.*;

public class Demo2 {

  /**
   * @param args the command line arguments
   */
  public static void main(String[] args) {

    SwingGL swgl = new SwingGL();
    try {
      swgl.newWindow("Main", "Demo Internal Frames", 100, 100, 500, 400);
      swgl.addMenuBarToWindow("Menu", "Main");
      String [] items = { "Demo" };
      MenuClick mclick = new MenuClick ( swgl );
      swgl.addMenuToWindow("Menu", "Demo",items, mclick);
      swgl.AddDesktopPaneToWindow("Desktoppane", "Main");

    } catch ( SwingGLException e ) { System.out.println ( e.toString()); }
```

```
    }
}

import swinggl.*;

public class MenuClick  implements SwingGLEvent {
    SwingGL swgl;
    int count;

    public MenuClick ( SwingGL swinggl ) {
        swgl = swinggl;
        count = 1;
    }

    @Override
    public void Event ( String ActionCommand ) {
        if (ActionCommand.equals("Demo")) {
            String frameName = "Demo " + count; count++;
            try {
                    SwingGL iframe = swgl.newInternalFrame(frameName, "Desktoppane", frameName,
10+count*20, 10+count*20, 300,300);
                iframe.newPanel("Panel", frameName,700,700);
                iframe.newTextField("TextField", "Panel", "", 10, 10, 100, 20 );
                iframe.newTextArea("TextArea", "Panel", "", 10, 40, 100, 50);
                iframe.newPasswordField("Password", "Panel", 10, 100, 100, 20);
                String[] Items = { "UNO" , "DUE" };
                iframe.newComboBox("Combo", "Panel", Items, 10, 130, 100, 20);
                iframe.newCheckBox("Check", "Panel", "Check", 10, 160, 100, 20, false);
                iframe.newButtonGroup("BG");
                iframe.newRadioButton("Radio1", "Panel", "BG", "1", 10, 190, 50, 20, true);
                iframe.newRadioButton("Radio2", "Panel", "BG", "2", 50, 190, 50, 20, false);

                ClickGet Get = new ClickGet (iframe);
                ClickSet Set = new ClickSet (iframe);

                iframe.newButton("GET", "Panel", "GET", 130, 10, 100, 20, Get);
```

```
 iframe.newButton("SET", "Panel", "SET", 130, 40, 100, 20, Set);
          } catch ( SwingGLException e ) { System.out.println (e.toString()); };
       }
    }
}

import swinggl.*;

public class ClickGet implements SwingGLEvent {

    SwingGL swgl;

    public ClickGet (SwingGL swinggl ) {
       swgl = swinggl;
    }

    public void Event ( String ActionCommand ) {
      try {
       String textfield = swgl.GetText("TextField");
       String textarea = swgl.GetText("TextArea");
       char[] password = swgl.GetPassword("Password");
       String combobox = swgl.GetSelection("Combo");
       boolean checkbox = swgl.GetSelected("Check");
       boolean radiobutton1 = swgl.GetSelected("Radio1");
       boolean radiobutton2 = swgl.GetSelected("Radio2");

       System.out.println (textfield);
       System.out.println (textarea);
       System.out.println (password);
       System.out.println (combobox);
       System.out.println (checkbox);
       System.out.println (radiobutton1);
       System.out.println (radiobutton2);
      } catch ( SwingGLException e ) { System.out.println ( e.toString()); }
    }
}
```

```
import swinggl.*;

public class ClickSet implements SwingGLEvent {

   SwingGL swgl;

   public ClickSet (SwingGL swinggl ) {
      swgl = swinggl;
   }

   public void Event ( String ActionCommand ) {
   try {
      swgl.SetText("TextField", "Setted");
      swgl.SetText("TextArea", "Setted");
      swgl.SetSelection("Combo",1);
      swgl.SetSelected("Check", true);
      swgl.SetSelected("Radio2",true);
   } catch ( SwingGLException e ) { System.out.println ( e.toString()); };
   }
}
```

RISULTATO

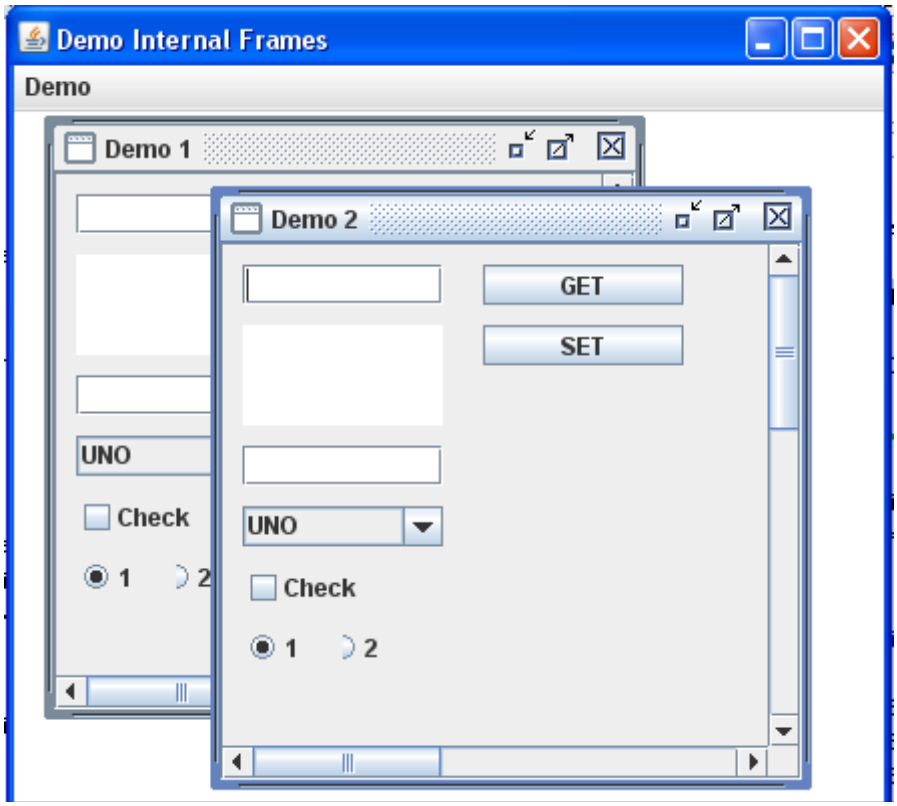


TABELLA CODICI ED ERRORI ECCEZIONI

Codice	Tipo Errore
1	Nome Oggetto Esistente e Non Univoco
2	Nome Oggetto Errato
3	Get da un Oggetto di tipo Errato
4	Set su di un Oggetto di tipo Errato