Cpp2Java API

For CSC254 students and lab assistants

NOTE: Before you read this guide, I would recommend looking at the included code examples and seeing if you can understand the basic gist of Cpp2Java by inspection. Copy and paste them into Visual Studio and try making small changes. If you are still confused, refer to this more detailed guide for help.

Quick Links

Cpp2Java class

Description

Use this class to make changes to the Cpp2Java JFrame, which is running in the background.

As with all of the other Cpp2Java classes, you will need to include “**Cpp2Java.h”** in your C++ code to use this class.

Initialization

Use the following line for initialization:

```
Cpp2Java myJFrame;
```

Methods

Return Value	*Syntax / Parameters*	*Description / Example*
`void`	**`removeAll ();`**	Remove all components from the JFrame.
`void`	**`update ();`**	Signal Java to update your GUI based on commands

		you have written.
`void`	**`finish ();`**	Signal Java to execute all commands in your C++ code, and start listening for action events from Java.
`void`	**`pause`** `( double seconds );`	Causes your program to pause for the number of seconds indicated. Use this in conjunction with the paint methods of JComponent to create animations.
`void`	**`add`** `( JComponent jc );`	Place a JComponent within the hierarchy of this JFrame (for example, JPanels or JButtons).
`void`	**`add`** `( JComponent jc, string BorderLayout );`	Add a JComponent with a specified BorderLayout type. ***`Cpp2Java.add(myPanel, “BorderLayout.NORTH”)`***
`void`	**`setLayout`** `( BorderLayout bl );`	Set the Layout of the JFrame’s Content Pane. ***`Cpp2Java.setLayout( new BorderLayout() )`***
`void`	**`setLayout`** `( GridLayout gl );`	Set the Layout of the JFrame’s Content Pane. ***`Cpp2Java.setLayout( new GridLayout(2,2) )`***

JComponent

Description

This is the base class of all the components that you will add to your JFrame. Just as in Java, all of the types of JComponent inherit from this class (for example, JPanel, JLabel, JButton, etc).

The student should note that just as in Java, JComponents can have their own paint elements. In Cpp2Java however, instead of overriding the paint method for each component as you did in Java, you can simply call the paint methods directly on the component. For example:

```
myPanel.drawRect ( arguments );
myPanel.drawString ( arguments );
```

These new graphics will be displayed upon calling **`repaint()`**.

Subclasses

`JPanel, JButton, JLabel, JTextArea, JTextField`

Methods

Most of the methods look exactly identical to the corresponding Java methods, but they are listed in the following table in case you need a refresher.

Return Value	*Syntax / Parameters*	*Description / Example*

`void`	`add ( JComponent jc );`	Place another JComponent within the hierarchy of this JComponent. For example, you could add a JButton to a JPanel. `myPanel.add( myButton );`
`void`	`repaint ();`	Signal Java to repaint this JComponent immediately.

`void`	`drawLine ( int xStart,`
	`           int yStart,`
	`           int xEnd,`
	`           int yEnd );`
		Draw a line on this component.

`void`	`drawRect ( int x, int y,`
	`           int width,`
	`           int height );`
		Draw a rectangle on this component.

`void`	`fillRect ( int x, int y,`
	`           int width,`
	`           int height );`
		Fill a rectangle on this component.

`void`	`clearRect( int x, int y,`
	`           int width,`
	`           int height );`
		Clear a rectangle on this component.

`void`	`drawOval ( int x, int y,`
	`           int width,`
	`           int height );`
		Draw an oval on this component.

`void`	`fillOval ( int x, int y,`
	`           int width,`
	`           int height );`
		Fill an oval on this component.

`void`	`drawString ( string text,`
	`             int x, int y );`
		Draw a string on this component.

JButton

Description

One of the types of JComponents you can make. It makes the same JButtons you saw in Java.

Initialization

Use this line of code in your C++ program:

```
JButton myBtn1 ( “This is my first button in C++” );
```

Methods

In addition to having all of the methods of JComponent, JButton has the following additional method:

Return Value	*Syntax / Parameters*	*Description / Example*
`void`	`addActionListener ( ActionListener aL );`	Add an actionlistener to this button. Works just like Java.

JPanel

Description

JPanels work the same way as in Java.

Initialization

```
JPanel myPanel;
```

Methods

See JComponent. JPanel is a subclass of JComponent, and therefore can use methods from JComponent.

ActionListener

Description

Use this class to create Button Handlers and other ActionListeners as you did in Java. It has one method you can override: actionPerformed (just as in Java). You will then initialize this class and attach it to your JComponents.

Example

Use this template to create your own ActionHandler in your C++ code:

```
class ActionHandler : public ActionListener
{
      public:
            void actionPerformed(ActionEvent ae)
            {
                  if (ae.getSource() == myBtn1)
                        // do something

                  else if (ae.getSource() == myBtn2)
                        // do something different
            };
};
```

Initialization

After you have written your Action Handler using the template above, use this code for initialization:

```
ActionHandler myAH;
```

Methods

Return Value	*Syntax / Parameters*	*Description / Example*
void	**actionPerformed (ActionEvent** ae **);**	Override this method and add your own commands. It will be called when actions are performed in Java.

KeyListener

Description

Use this to create a key listener as you did in Java. Some “strange” characters may simply return a space.

Example

Use this template to create your own KeyHandler in your C++ code:

```
class KeyHandler : public KeyListener
{
      public:
            void keyReleased(KeyEvent ke)
            {
                  // do stuff
            };
};
```

Initialization

After you have written your Key Handler using the template above, use this code for initialization:

```
KeyHandler myKH;
```

Methods

Return Value	*Syntax / Parameters*	*Description / Example*
void	**keyReleased (KeyEvent** ke **);**	Override this method and add your own commands. It will be called when keys are pressed in Java.

ItemListener

Description

Use this to create a item listener as you did in Java.

Example

Use this template to create your own ItemHandler in your C++ code:

```
class ItemHandler : public ItemListener
{
     public:
          void itemStateChanged(ItemEvent ie)
          {
               if (ie.getSource() == myItem1)
               {
                    if (ie.getStateChange() == ie.SELECTED)
                         // do something
               }
               else if (ie.getSource() == myItem2)
               {
                    if (ie.getStateChange() == ie.SELECTED)
                         // do something else
               }

          };
};
```

Initialization

After you have written your Item Handler using the template above, use this code for initialization:

```
ItemHandler myIH;
```

Methods

Return Value	*Syntax / Parameters*	*Description / Example*
`void`	**`itemStateChanged ( ItemEvent ie );`**	Override this method and add your own commands. It will be called when items are changed in Java.