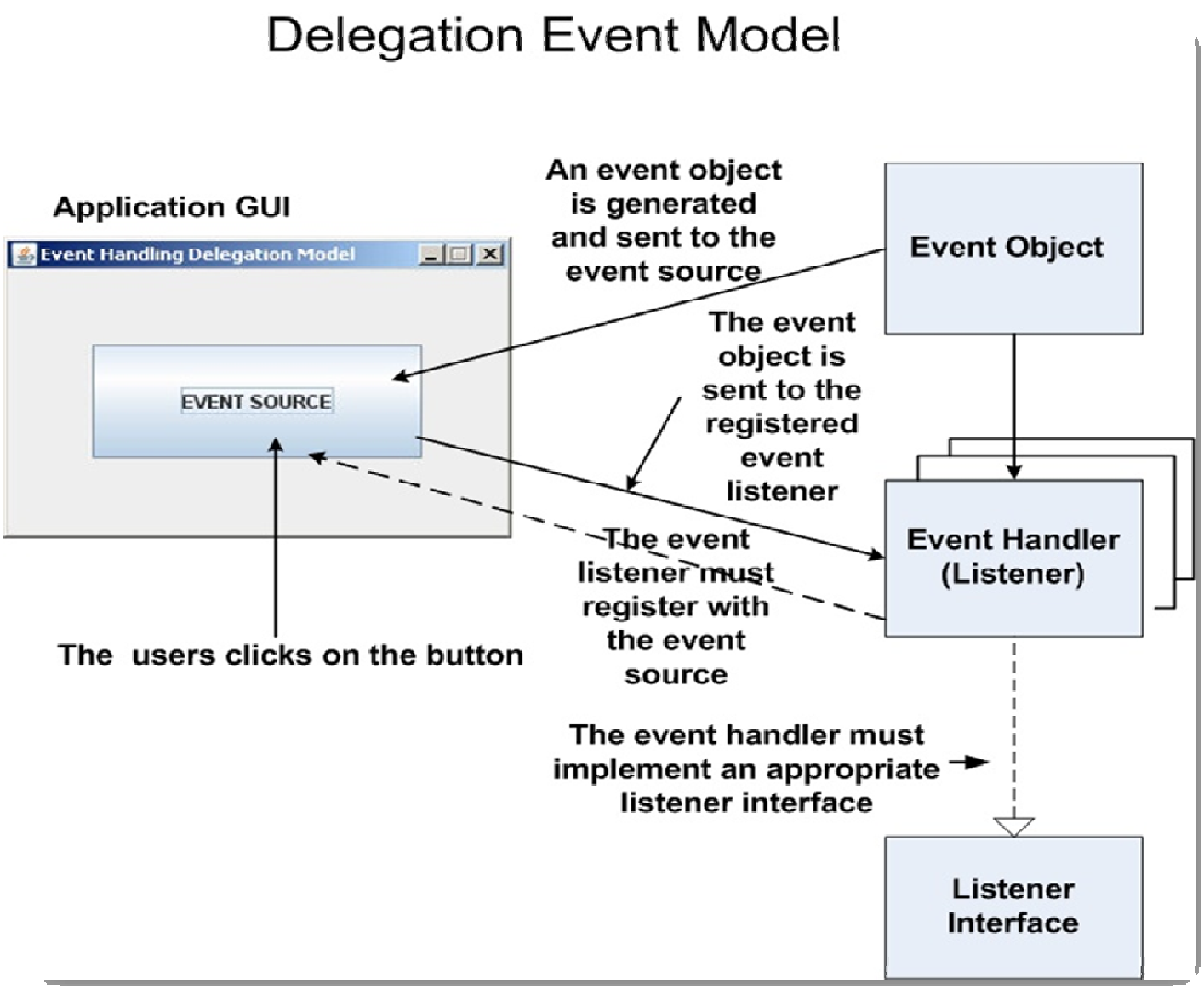
CST8221 – Java Application Programming

Unit 3 Part 2 – Event Handling

When a user performs an action at the user interface level (clicks a mouse or presses a key), this causes an event to be generated by the virtual machine. Events are of objects that contain information about what has happened. A number of different types of event classes exist to describe different categories of user action.



When a user clicks on a GUI component (for example JButton), an event object is created by the JVM and the component sends this event to the registered event handler (listener) for that component by calling an appropriate method (for example actionPErformed()).

Event Sources

An event source is the generator (originator) of the event. For example, a mouse click on *JButton* component generates an *ActionEvent* (and *MouseEvent*) instance with the button as a source. The *ActionEvent* instance is an object that contains information about the events that just took place. Part of this information is a reference to the source the event.

Event Handlers

An event handler is a method that receives an event object, gets information from the event object and reacts to the user's action.

Delegation Event Model

With the delegation event model, events are sent to the component from which the event originated, but it is a responsibility of the component to transmit the event to one or more registered classes which called ***listeners***. Listeners contain one or many event handlers that receive and process the event. In this way, the event handler can be in an object completely separate from the component.. Events are reported only to registered listeners. If no listener is registered with the component the event will not be processed. Every event has a corresponding listener interface that determines which methods must be defined in a class designated as a listener. Any class that implements the interface which defines the event handler methods can be registered as a listener. The objects that are scheduled to listen to particular events on a particular GUI component register themselves with that component. When an event occurs, only the objects that are registered with the component receive a message that the event occurred.

Each component which can be an event source has a suitable method for registering a specific event listener. In Swing the name of the method is addXXXXListener, where XXXX is the name of the event. For example, ***addWindowListener***, ***addMouseListener***, ***addActionListener***, ***addKeyListener***, and so on.

JavaFX is using the same event delegation model, but different API classes are involved in the implementation. The methods used to register an event listener are named ***setOnXXX***, where the XXX is the name of the event. For example:

final void setOnAction(EventHndler<ActionEvent> *handler*)

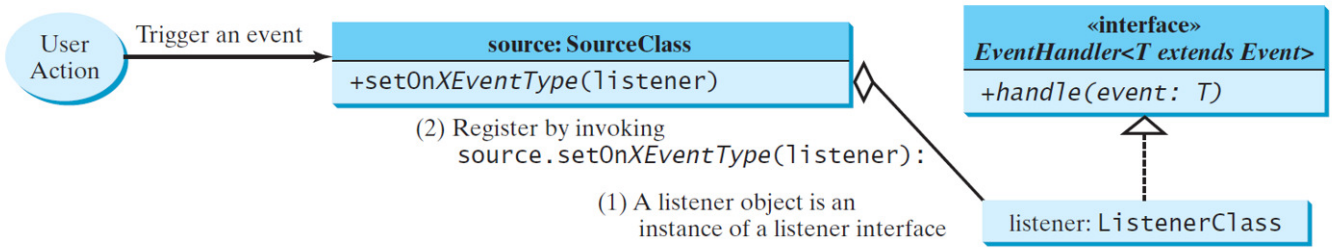
where, *handler* is the handler being registered. Events are handled by implementing ***EventHandler*** interface

interface EventHandler <T extends Event>

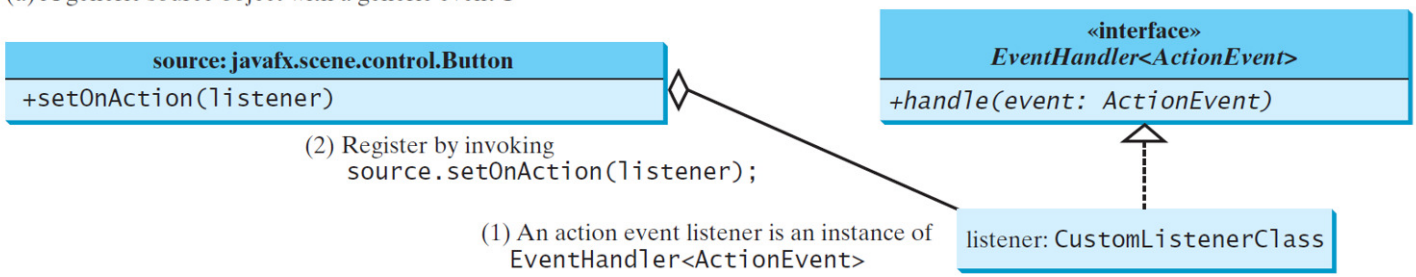
which defines one single method called

void handler(T *eventObject*).

JavaFX Delegation Event Model



(a) A generic source object with a generic event T



(b) A `Button` source object with an `ActionEvent`

JavaFX Event Classes