INTRODUCTION TO JAVAFX EVENT

Using JavaFX, several types of applications such as desktop, web, and graphical applicationscan be developed. Most of the applications in the current world need user interaction to work. For that, the event concept is used from JavaFX. An event is said to have happened inthe situations where the user interacts with the application nodes. These events can be triggered using mouse movements, button press, page scrolling, etc. That is, these events are able to give a notification that something has happened from the user end.

Syntax of JavaFX Event

There are several events that JavaFX supports. For Event class, the package used isjavafx.event which is considered as the base class.

Following are the different types of events supported by JavaFX:

1. MouseEvent: This event occurs in the situation where the mouse is clicked.

- **Represented Class:** MouseEvent.
- **Actions:** Clicking mouse, pressing the mouse, releasing the mouse, moving mouse,target entering, target exiting etc.

Syntax:

```
EventHandler<MouseEvent> eh = new EventHandler<MouseEvent>()
```

2. KeyEvent: This event occurs in the situation where a keystroke happens at the node.

- **Represented Class:** KeyEvent.
- **Actions:** Typing key, pressing a key, releasing the key.

Syntax:

```
EventHandler<KeyEvent> eh = new EventHandler<KeyEvent>()
```

3. DragEvent: This event occurs in the situation where dragging of the mouse is done.

- **Represented Class:** DragEvent.
- **Actions:** Entering drag, dropping drag, entering target, exiting target, drag over.

Syntax:

```
EventHandler<DragEvent> eh = new EventHandler<DragEvent>()
```

4. WindowEvent: This event occurs in the situation where a keystroke happens at the node.

- **Represented Class:** WindowEvent.
- **Actions:** Hiding window, showing window.

Syntax:

```
EventHandler<WindowEvent> eh = new EventHandler<WindowEvent>()
```

How JavaFX Event Handling works?

- Event Handling is the process in which the decision to determine what

has to have happened when an event occurs and how to control that particular event.

- For this, a code is used as an event handler that gets executed at the time which theevent has occurred.
- JavaFX offers several handlers as well as filters for handling the events.
- That is, for every event in JavaFX, it has a target which is the node where the event has occurred (these nodes can be scene, window, or node), a source where the event has generated (mouse, keys, etc.), type of the event (mouse event, key event,etc.).

Events

An event represents an occurrence of something of interest to the application, such as amouse being moved or a key being pressed. In JavaFX, an event is an instance of
the javafx.event.Event class or any subclass of Event. JavaFX provides several events, including DragEvent, KeyEvent, MouseEvent, ScrollEvent, and others. You can define yourown event by extending the Event class.

Every event includes the information described

in table .Table Event Properties

Property	Description
Event type	Type of event that occurred.
Source	Origin of the event, with respect to the location of the event in the event dispatch chain. The source changes as the event is passed along the chain.
Property	Description

Target	Node on which the action occurred and the end node in the event dispatch chain. The target does not change, however if an event filter consumes the event during the event capturing phase, the target will not receive the event.

Event subclasses provide additional information that is specific to the type of event. For example, the MouseEvent class includes information such as which button was pushed, thenumber of times the button was pushed, and the position of the mouse.

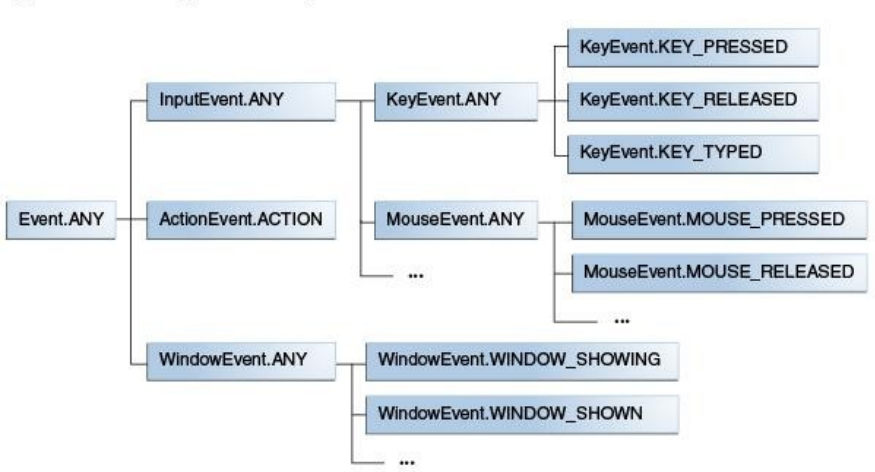
Event Types

An event type is an instance of the EventType class. Event types further classify the events of a single event class. For example, the KeyEvent class contains the following event types:

- KEY_PRESSED
- KEY_RELEASED
- KEY_TYPED

Event types are hierarchical. Every event type has a name and a super type. For example,the name of the event for a key being pressed is KEY_PRESSED, and the super type
is KeyEvent.ANY. The super type of the top-level event type is null. Figure 1-1 shows asubset of the hierarchy.

Figure Event Type Hierarchy



The top-level event type in the hierarchy is Event.ROOT, which is equivalent to Event.ANY. Inthe subtypes, the event type ANY is used to mean any event type in

the event class. For
example, to provide the same response to any type of key event, use KeyEvent.ANY as the event type for the event filter or event handler. To respond only when a key is released, use the KeyEvent.KEY_RELEASED event type for the filter or handler.

Event Targets

The target of an event can be an instance of any class that implements
the EventTarget interface. The implementation of the buildEventDispatchChain creates theevent dispatch chain that the event must travel to reach the target.

The Window, Scene, and Node classes implement the EventTarget interface and subclasses of those classes inherit the implementation. Therefore, most of the elements in your user interface have their dispatch chain defined, enabling you to focus on responding to the events and not be concerned with creating the event dispatch chain.

If you create a custom UI control that responds to user actions and that control is a subclass of Window, Scene, or Node, your control is an event target through inheritance. If your control or an element of your control is not a subclass of Window, Scene, or Node, you must implement the EventTarget interface for that control or element. For example,
the MenuBar control is a target through inheritance, but the MenuItem element of a menu bar must implement the EventTarget interface so that it can receive events.