Listeners

pm jat @ daiict

Listeners

- What are?
- How do you implement?

What are listeners?

- Listeners allow to track key events in web applications through event listeners.

- This functionality allows more efficient resource management and automated processing based on event status.

- Following are events that can be tracked –
  - ServletContext Startup and shutdown
  - ServletContext Attribute changes - setAttribute
  - HttpSession Creation and invalidation
  - HttpSession Changes in attributes - setAttribute

Usage Examples of Servlet Listeners

- Monitor start and stop of Web modules to perform startup and shutdown tasks.
  - When application starts, listener is notified and creates a connection to the database.
  - Connection is stored in servlet context attribute so that servlets access the connection as needed from the servlet context
  - When the Web application is shutdown, listener is notified and closes database connection
- Add attributes to **ServletContext** or **HttpSession** objects on creation
- Monitor creation and destruction of sessions
- Log important application events

How do you implement?

- Decide which scope object you want to receive an event notification
- Implement appropriate interface
- Override methods that need to respond to the events of interest
- Obtain access to important Web application objects and use them
- Configure web.xml accordingly
- Provide any needed initialization parameters

Listener Registration

- Web container
  - creates an instance of each listener class
  - registers it for event notifications before processing first request by the application
  - Registers the listener instances according to
    - the interfaces they implement
    - the order in which they appear in the deployment descriptor web.xml
- Listeners are invoked on occurrence of appropriate event, if there are multiple listeners for an event they are invoked in the in which they appear in the deployment descriptor web.xml

Context Listener Events

- Context Initialized
- Context Destroyed
- Attribute Added
- Attribute Removed
- Attribute Replaced

HTTP Session Listener Events

- Session Created
- Session Destroyed
- Attributed Added
- Attributed Removed
- Attributed Replaced

Java API (interfaces)

- **`ServletContextListener`**
  - contextInitialized/Destroyed(ServletContextEvent)
- **`ServletContextAttributeListener`**
  - attributeAdded/Removed/Replaced(ServletContextAttributeEvent)
- **`HttpSessionListener`**
  - sessionCreated/Destroyed(HttpSessionEvent)
- **`HttpSessionAttributeListener`**
  - attributedAdded/Removed/Replaced(HttpSessionBindingEvent)


Dhirubhai Ambani Institute of Information and Communication Technology
DA-IICT

- Example

Example: Context Listener

```
public final class ContextListener
      implements ServletContextListener {
      private ServletContext context = null;

      public void contextInitialized(ServletContextEvent event)
            context = event.getServletContext();

            try {
               BookDB bookDB = new BookDB();
               context.setAttribute("bookDB", bookDB);
            } catch (Exception ex) {
               context.log("Couldn't create bookstore
                           database bean: " + ex.getMessage());
            }

            Counter counter = new Counter();
            context.setAttribute("hitCounter", counter);
            counter = new Counter();
            context.setAttribute("orderCounter", counter);
      }
```

Example: Context Listener

```
    public void contextDestroyed(ServletContextEvent event) {
          context = event.getServletContext();
          BookDB bookDB = (BookDB)context.getAttribute
   ("bookDB");
          bookDB.remove();
          context.removeAttribute("bookDB");
          context.removeAttribute("hitCounter");
          context.removeAttribute("orderCounter");
    }
}
```

Listener Configuration

```
<web-app>
  <display-name>Bookstore1</display-name>
  <description>no description</description>

  <filter>..</filter>
  <filter-mapping>..</filter-mapping>
  <listener>
    <listener-class>listeners.ContextListener</listener-class>
  </listener>
  <servlet>..</servlet>
  <servlet-mapping>..</servlet-mapping>
  <session-config>..</session-config>
  <error-page>..</error-page>
  ...
</web-app>
```

Listeners and Filters

- Filters are queued and always executed for every HTTP request

- Listeners responds to events and can be better for event based tracking, and can be used for
  - logging,
  - auditing,
  - life cycle based initialization and clean-ups

Additional readings

http://docs.oracle.com/cd/B14099_19/web.1012/b14017/filters.htm