Use Cases and Controllers

Use Case

- Describe an interaction between a user and a system
  - Spans multiple commands
  - Is intended to perform a task
- Examples:
  - Buy a book online
  - Buy gas at a pay-at-the-pump service station.
Terminology

- Actor – an active entity in the use case
  - Generally a user
  - Can include other entities external to the program – printer, database, gas pump, ...

Example Use Case - Pay at the Pump

1. Customer inserts credit card.
2. Customer removes credit card quickly.
3. System authorizes credit card.
4. Customer remove nozzle from pump.
5. Customer selects grade of gasoline.
7. Customer pumps fuel.
8. Customer returns nozzle to pump.
9. System charges credit card.
10. System prints receipt.
Example Use Case Diagram

- Buy gas with cash
- Buy gas with credit card
- Credit card company
- Cashier
- Pump
- Gas pump software

Use Case Controller

- Manages the control flow through a use case
- Localizes knowledge of the control flow, making it easy to change.
- Coordinates actions taken by the user, the user interface and system responses.
- Can be customized by applying the Strategy pattern.
Controller

- Receives events from other components
- Determines what action the system should take in response
- Typically create a new instance of a controller each time its use case starts
- Controller classes sound like active entities, like GasPumpManager or HandPlayer
- Often, each instance executes in its own thread.

Pay with Credit Card Controller

public class PayAtPumpController implements Runnable {
    public void run() {
        authorizeCreditCard();
        pumpGas();
        chargeCreditCard();
        printReceipt();
    }
}

new PayAtPumpController().start();
Java Servlet Model

- Apache - Web server
- Tomcat - servlet container inside Apache
- Install your servlet where Tomcat can find it, accessible from the network via a URL
- When your servlet URL is used, Tomcat creates a thread to run your servlet to handle one user request
- Use session to manage data that needs to persist between requests, like an order that is being processed

Example

<table>
<thead>
<tr>
<th>Function</th>
<th>Servlet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter the bookstore</td>
<td>BookStoreServlet</td>
</tr>
<tr>
<td>Create the bookstore banner</td>
<td>BannerServlet</td>
</tr>
<tr>
<td>Browse the bookstore catalog</td>
<td>CatalogServlet</td>
</tr>
<tr>
<td>Put a book in a shopping cart</td>
<td>CatalogServlet, BookDetailsServlet</td>
</tr>
<tr>
<td>Get detailed information on a specific book</td>
<td>BookDetailsServlet</td>
</tr>
<tr>
<td>Display the shopping cart</td>
<td>ShowCartServlet</td>
</tr>
<tr>
<td>Remove one or more books from the shopping cart</td>
<td>ShowCartServlet</td>
</tr>
<tr>
<td>Buy the books in the shopping cart</td>
<td>CashierServlet</td>
</tr>
<tr>
<td>Receive an acknowledgement for the purchase</td>
<td>ReceiptServlet</td>
</tr>
</tbody>
</table>