

Final Exam

EE/CIS 694T

Spring 2000

A chemistry lab uses a **tank** to collect liquids produced by various **experiments**. This **tank** is controlled by a **valve controller**, which **monitors** the volume of the liquid in the **tank**, and **drains** the tank if the volume of the liquid reaches a certain value.

In this exam you will *implement* an abstraction for a **tank**, a **valve controller**, and an **experiment**. The **tank** and the **valve controller**, will be implemented as **JavaBean** components. The **experiment** will *assemble* the system, *wire* the **valve controller** to the **tank**, and start *filling* the tank with liquid.

- *Fully implement* the three classes involved, **experiment**, **valve controller**, and **tank** (details given below) in *Java*. Also implement any required *Interfaces*, and *classes* to handle all the *events*, and *exceptions*.
- Show *all* your code.

Here are the **requirements** in more detail:

Tank. Implemented as a JavaBean

- Contains a **volume** of liquid in it. Use gallons for the unit of volume.
- You can **add** liquid to the tank, one gallon at a time.
- Every time the volume of the liquid changes in the tank, interested *listeners* should be informed that the volume has changed. The interested listeners should be able to determine the volume of liquid in the tank.
- If the volume of the liquid in the tank reaches, or goes above **maxVolume**, interested *listeners* should be informed about this. It is the responsibility of a *listener* to take appropriate action in response to this event.
- If the volume of the liquid reaches **fullVolume**, then the tank **shuts** itself down, and throws an *exception* every time there is an attempt to add more liquid to it. The volume should not increase above **fullVolume**.
- The tank can be **drained**. The volume of the liquid decreases to **minVolume** when the tank is **drained**. Note that since this is a change of volume of the liquid, *listeners* should be informed of this event.
- Note that **minVolume < maxVolume < fullVolume**.

- Use only *one* **EventListener Interface**, and one **Event Object class** to handle all the events.

Valve Controller. Implemented as a JavaBean.

- One should be able to **wire** a **Valve Controller** to a **Tank**.
- Every time the liquid level of the tank (it is associated with) changes, **Valve Controller** writes the new volume to a **log** (standard output).
- The **Valve Controller** also writes any *exceptional* behavior associated with the **Tank** to the **log** (standard output).
- If the volume of the **Tank** it is associated with reaches **maxVolume** or above, it **drains** the Tank.

Experiment.

- **Experiment** is a client of the **Tank** and **Valve Controller**.
- **Experiment**, in its **main()** method:
 - Creates a **Tank** named **mainTank**, with **fullVolume = 110** gallons, **maxVolume = 100** gallons, and **minVolume = 10** gallons.
 - **mainTank** starts out being **empty**.
 - Creates a **Valve Controller** named **mainValveController**.
 - **Wires** the **mainValveController** to the **mainTank**. Note that this wiring should automatically make **mainValveController** a listener to events emitted by **mainTank**. **Experiment** should not have to take care of event registration etc.
 - Starts filling **mainTank** with liquid (in a loop) in one gallon increments. The loop should iterate **150** times.