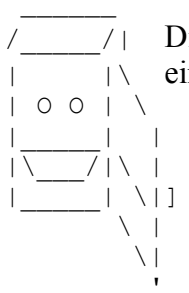
Übung 7

Hilfe, mein Kühlschrank spricht zu mir – Teil 4 (15 Punkte)



Diesmal erweitern wir den Kühlschrank um einen Logger, ein Ant buildfile sowie eine Netzwerk-Fernbedienung über XML-RPC.

Aufgabe 1: Log4J (Log4J-Doku: http://logging.apache.org/log4j/1.2/manual.html)

Erweitere den Kühlschrank-Code aus dem Template-file (Musterlösung Übung 4) um log4j-Logging. Es soll im INFO-Level das Starten und Beenden des Kühlschrankprogramms, im DEBUG-Level das Verändern des Kühlschrankbestands geloggt werden. Bei Benutzer-Eingabe-Fehlern, Java Errors und Exceptions soll eine Meldung im ERROR-Level ausgegeben werden.

Definiere eine Logging-Konfigurationsdatei, welche log4j veranlasst, alle Log-Meldungen außer den DEBUG-Meldungen in eine HTML-Datei und außerdem gleichzeitig **alle** Meldungen auf einen Chainsaw-Socket auszugeben (Tip: Wie man die Log-Levels filtert, steht in den log4j FAQs!). Beim Testen muss Chainsaw zuerst gestartet werden (in einer anderen JVM). Die Logger-Konfigurationsdatei bitte ebenfalls als Teil der Lösung einschicken.

Aufgabe 2: Apache Ant Buildfile (Ant-Dokumentation: http://ant.apache.org/manual/index.html)

Definiere eine ant-build-Datei (build.xml), welche jeweils in einem eigenen target

- **javacc** den javacc-Parser mittels des `<javacc>`-Tasks erzeugt,
- **compile** alle Klassen compiliert (Patternset für Quelldateien definieren),
- **javadoc** Javadoc für die Klassen erzeugt (patternset), und anschließend die erzeugte javadoc-HTML-Start-Datei im Web-Browser anzeigt,
- **xpathtest** das XSLT-Stylesheet aus Übung 3, Aufgabe 1 auf der dort angegebenen Testdatei testet (`<xslt>`-task verwenden),
- **chainsaw** die log4j-GUI Chainsaw startet (mit Hilfe des `<java>`-Tasks; vgl. Aufgabe 1),
- **start_fridge** den Kühlschrank-Code startet.

Aufgabe 3: XML-RPC (XML-RPC download + Dokumentation: http://ws.apache.org/xmlrpc/)

Verwende den XML-RPC-Server-Code aus der Vorlesung, um den Kühlschrank mittels des XML-RPC-Protokolls fernbedienbar zu machen. Erweitere dazu FridgeEx2 um eine public-Methode `getInventoryAsString()`, die den Kühlschrankinhalt als String zurück gibt (`Properties.toString()` genügt), sowie eine Methode `getWeatherAsString()`, die die Wettervorhersage als String liefert. Schreibe außerdem einen Test-Client, der die beiden Methoden über XML-RPC aufruft und ihre Rückgaben auf den Bildschirm ausgibt.