InterBit
Training & Consulting Ltd.

Chain of responsibility

Chapter Content

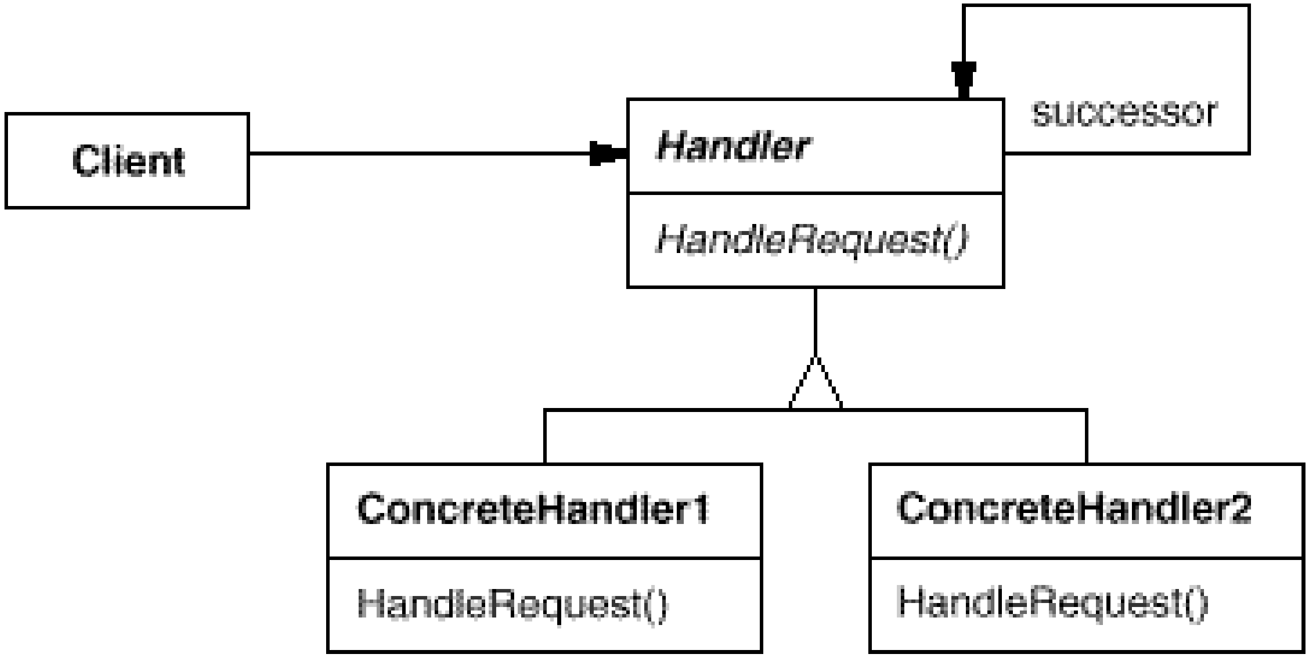
- Chain of Responsibility Overview
- CoR UML Diagram
- CoR Examples
- Chain Or Tree?

Chain of Responsibility Overview

- A request is passed through a chain of objects:
  - Each will decide whether to handle it or pass it on.
  - Request will be passed until it encounters object that can handle it, or until chain er

- Loose-coupling: chained classes only communicate through *handle(request).*
- May be considered a fancy, dynamically-built switch statement.

CoR UML Diagram



Each handler points to its successor (next handler in the chain).

Each handler should decide whether it can handle a request, or pass in on to its successor.

Handler objects may be of various concrete types, but they all need to implement the *handleRequest* method.

Example: class loaders

Java class loaders are chained, but in reverse order: first let your parent locate the class; if it fails, search for it yourself.

```
// Simplified code (consult jdk code for full implementation !)
protected synchronized Class loadClass(String classname)
throws ClassNotFoundException{
     try {
          // First let parent attempt to locate class:
          return parent.loadClass(classname);
     }catch(ClassNotFoundException e){
          // Parent failed, so try and locate class myself:
          return this.findClass(classname);
     }
}
```

.)Example: class loaders (cont

- Class-loading example:
  - The Application-class-loader loads classes from CLASSPATH.
  - However, it only does so if its parent (Extension-class-loader) fails to find such a class in the jre/lib/ext directory.
  - But extension-class-loader first lets it’s parent (Bootstrap-class-loader) locate the class in the standard java installation.
  - How does this model promote security ?

Example: Servlet chaining

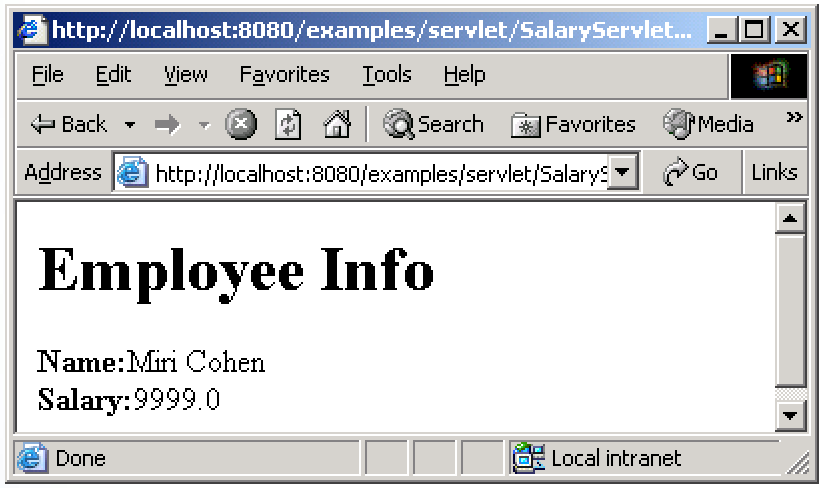
- Chained servlets, looking for employee details:
  - Our company’s servlet will look for employee details in its local DB.
  - If It fails, request will be forwarded to the parent company’s servlet (different DB, different reply structure).
  - If parent company fails, it could forward to a national DB.

Example: servlet chaining (cont.)

```
public class SalaryServlet extends HttpServlet{
  public void doGet(HttpServletRequest req, HttpServletResponse res)  throws ServletException{
      try{
          String employeeName =req.getParameter("employeeName");
          Employee emp = findEmployee(employeeName);
         if ( emp != null ) {
              String htmlReply = formatEmployeeHtml(emp);
              Writer out = res.getWriter();
              out.write(htmlReply);
              out.close();
          } else {
              String forwardUrl="/servlet/ParentCompanySalary";
             ServletContext ctx = getServletConfig().getServletContext();
             ctx.getRequestDispatcher(forwardUrl).forward(req, res);
         }
      }
      catch(Exception e){  ... }
```

:Output

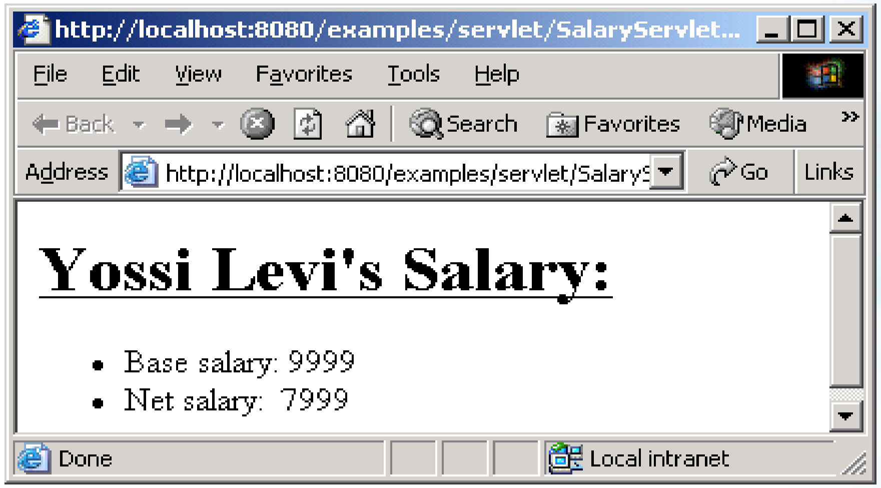
1. Local
(daughter company
servlet):


http://localhost:8080/examples/servlet/SalaryServlet...
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:8080/examples/servlet/SalaryS
Go Links
Done
Local intranet

Employee Info

Name: Miri Cohen
Salary:9999.0

2. Forwarded to
parent company servlet:


http://localhost:8080/examples/servlet/SalaryServlet...
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:8080/examples/servlet/SalaryS
Go Links
Done
Local intranet

Yossi Levi's Salary:

- Base salary: 9999
- Net salary: 7999

3. Forward to
servlet that contacts national
DB...

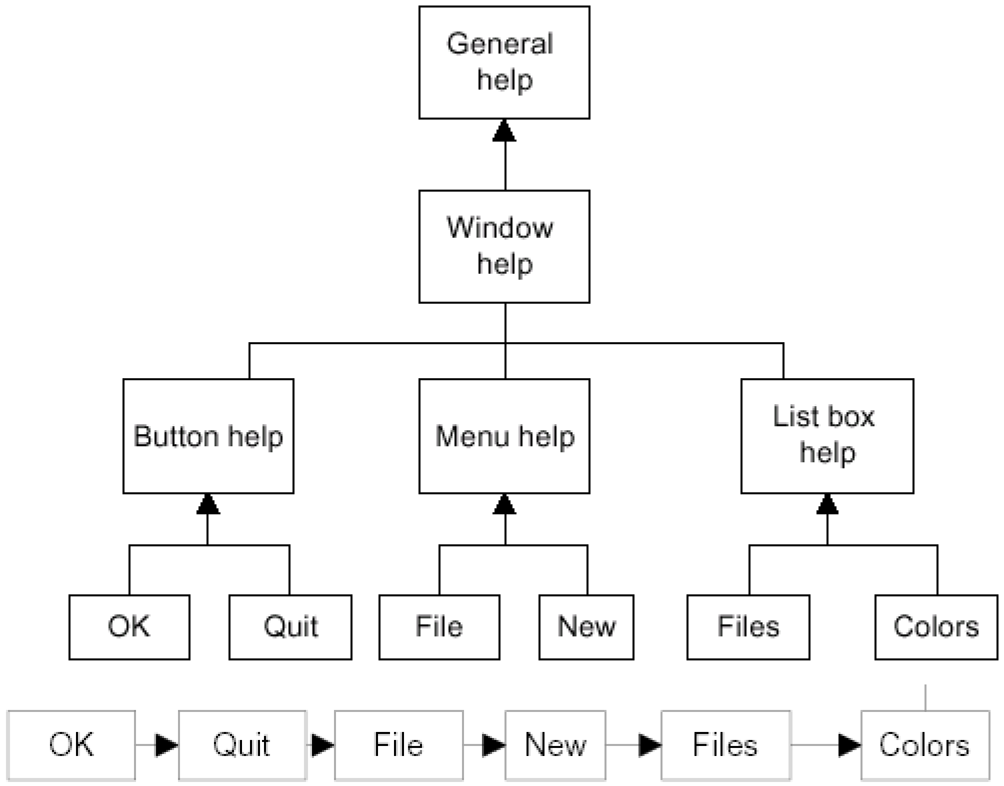
Variations

- Possible Twists to our servlet chain:
  - Servlet may modify the request (e.g. add attributes) before forwarding it.
  - We're likely to have a tree rather than a chain
  - Note the difference between *forward* & *include.*
  - You may also send an *html redirection* to the browser (for a completely different machine).

Example: old awt events

- JDK 1.0 events used to rely on a chain of responsibility.
  - When an awt component discovered an event (e.g. mouse clicked on this component's area)
    it would either:
    - Decide how to handle the event.
    - Forward the event to it container.
    - Both (handle the event but not consume it) – this is somewhat of a deviation from the classic pattern.

Chain Or Tree?



- The *Smalltalk companion for Design Patterns* suggests that, at times, a Tree of Responsibility may be a more generalized solution, albeit it may be more costly.