**Filter**

* Với MVC2 DP dùng Servlet làm Controller
  + Servlet sẵn có trong bộ nhớ, khi Request tới Container, tạo ra Request Object, Request Object được đưa tới Servlet. Dữ liệu có sai thì Servlet vẫn phải chạy (chỉ để thông báo dữ liệu sai) 🡪 phí phạm
* Filter tương ứng với Request Dispatcher 🡪 Pre-process request và post-process response
* Trước Filter 🡪 ServletRequest, sau Filter 🡪 HttpServletRequest
  + Khi vào Container có 2 loại:
    - Request thô: không tồn tại Parameter
    - RequestObject đã qua xử lí: có Parameter, Properties
* Đối với người dùng, filter là invisible, tự động kích hoạt khi đến thành phần xử lí
* Sử dụng Filter trong việc xử lí password mã hóa:

1. Khi Request chạm Container, Filter kích hoạt. Kiểm tra code mã hóa có đúng chuẩn không, nếu không thì không cho vô, nếu có thì qua bước 2
2. Thực hiện cơ chế giải mã
3. Sau khi kiểm tra, đẩy toàn bộ username password vào Login, Login thực hiện nhiệm vụ checkLogin

|  |  |
| --- | --- |
| **Filter**   * Invisible, tự động được gọi * Request vào phải qua filter xử lí thì Response ra cũng được filter xử lí | **Request Dispatcher**   * Gọi tường minh |

* chain.doFilter(): nếu chồng nhiều Filter lên nhau, khi hàm được gọi sẽ kiểm tra xem sau filter này còn filter nào không, nếu có chuyển từ filer 1 sang filter 2, và tương tự… nếu không thì chuyển cho thành phần resource xử lí
  + Khi response ra phải theo thứ tự filter ngược lại lúc vào
  + Tương ứng với call FilterServlet, thực thi code trong FilterServlet xong thì trở về lại vị trí call
* Ứng dụng:
  + Chia tải, mỗi filter làm 1 chức năng khác nhau
  + Modify header và data của request và response
  + Xác thực người dùng
  + Nén dữ liệu
  + Mã hóa
  + Convert ảnh
  + …
* Life cycle: giống life cycle của Servlet
  + Servlet kích hoạt dựa trên thành phần của phương thức trong Request
    - doHttpMethod() truyền vào HttpServletRequest, HttpServletResponse
  + Filter kích hoạt tự động bởi container
    - doFilter() truyền vào ServletRequest, ServletResponse, FilterChain
* Khai báo trong web.xml: giống khai báo Servlet

<filter>

<filter-name>…</filter-name>

<filter-class>…</filter-class>

</filter>

<filter-mapping>

<filter-name>…</filter-name>

<url-pattern>/context</url-pattern>

</filter-mapping>

* + url-pattern: tên resource mà filter sẽ apply (không có filter apply filter), có thể là servlet, jsp, xml…
  + Thứ tự khai báo filter cũng là thứ tự apply
  + url-pattern là /\* 🡪 apply toàn bộ ứng dụng, kể cả request yêu cầu html

**Sử dụng Filter như Controller**

* Chuyển Request thô thành Request Object
* getRequestURI() 🡪 RequestURI là toàn bộ đường truyền contextpath/…?queryString

**Assignment**

* Làm lại toàn bộ bài trong lớp với 9 chức năng

Login

Search

Delete

Update

Add to cart

View cart

Remove cart

Checkout

Sign up

* Bắt buộc mô hình MVC2 dùng Filter làm Controller
  + Tất cả mọi thứ trong Assignment phải dùng label/logical name để mapping
  + Click here to try again 🡪 rê vô hiện đường link “login.html” 🡪 0đ
    - href=“try”
  + Thay đổi phần code cứng (dòng 111 🡪 118 slide trang 49) thành file txt, key=value, chuyển thành Map, đặt trên ContextScope
  + printStackTrace() biến thành log(…)
  + CartServlet apply Filter hoặc Map, không được dùng if else, không được khai báo hằng số
* Trang shopping phải tạo bảng sản phẩm dưới db và load lên để lựa chọn add to cart (giữ nguyên giao diện cũ, hoặc giao diện lưới)
  + Nếu làm lưới, mỗi dòng có nút add to cart