發現java ee 有個東西叫 WebFilter

因此我寫了個filter如下

1. 使用WebFilter annonation
2. 因為是annonation 因此web.xml可以不用在寫相關設定
3. 可以過濾想要的網址 但因為使用者如果刻意多加/faces 也是會無法對應 因此我這邊就/faces/\* 下面都看
4. 解析URI

EX. [/ems/faces/mtbf/query.xhtml](http://localhost:8080/ems/faces/mtbf/query.xhtml)

      我先把/faces後面的取出來

      如果使用者惡意輸入網址 [/ems/faces/faces/mtbf/query.xhtml](http://localhost:8080/ems/faces/mtbf/query.xhtml)

      我第一次拆出來的字串 如果還有/faces就代表不是正常網址

1. 非正常網址就看要redirect or forward somewhere
2. 附件是我寫的測試程式 請自行參考 還沒有美化就是了
3. 有甚麼更好的心得建議或想法 再請提出來討論

=====code start=====

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package com.tcci.facilityManagement.controller.filter;

import java.io.IOException;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.annotation.WebFilter;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.slf4j.LoggerFactory;

/\*\*

\*

\* @author Jason.Chang

\*/

//@WebFilter("/faces/\*")

public class AuthorizationFilter implements Filter {

org.slf4j.Logger logger = LoggerFactory.getLogger(AuthorizationFilter.class);

@Override

public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws ServletException, IOException {

logger.debug("dofilter");

HttpServletRequest req = (HttpServletRequest) request;

HttpServletResponse resp = (HttpServletResponse) response;

try {

String uriStr = req.getRequestURI();

logger.debug("uriStr=" + uriStr);

//check /faces/ evil try multi

int firstPos = uriStr.indexOf("/faces");

if (firstPos != -1) {

String rootStr = uriStr.substring(0, firstPos);

String tempStr = uriStr.substring(firstPos + 6, uriStr.length());

logger.debug("rootStr=" + rootStr);

logger.debug("tempStr=" + tempStr);

if (tempStr.indexOf("/faces") != -1) {

logger.debug("get u and error forward");

String processStr = "";

//回到系統首頁 start

// processStr = req.getContextPath();

// processStr = processStr + "/";

//回到系統首頁 end

//回到正常頁面URL 過濾掉多餘的/faces start

tempStr = tempStr.replaceAll("/faces", "");

logger.debug("tempStr fix=" + tempStr);

processStr = "/faces" + tempStr;

//回到正常頁面URL 過濾掉多餘的/faces end

logger.debug("processStr=" + processStr);

//===無效 請勿使用 forward start===

//forward實測有問題 security constraint 沒有用...

// RequestDispatcher rd = req.getRequestDispatcher(processStr);

// rd.forward(req, resp);

//=== forward end===

//=== redirect start===

resp.sendRedirect(rootStr + processStr);

//=== redirect end===

}

}

chain.doFilter(request, response);

} catch (Exception e) {

logger.error("exception filter");

String forwardURI = request.getServletContext().getContextPath();

forwardURI = forwardURI + "/";

resp.sendRedirect(forwardURI);

}

}

// You need to override init() and destroy() as well, but they can be kept empty.

@Override

public void init(FilterConfig filterConfig) throws ServletException {

//

logger.debug("INIT");

}

@Override

public void destroy() {

//

logger.debug("destroy");

}

}

=====code end======

* 另外一種解法就是 直接針對/faces/faces/\*



直接就轉導到系統首頁去