public final class ICalWorker

Line 56-57

Not usefull class Javadoc specification: insufficient and too general.

Line 61

public static final String module = ICalWorker.class.getName();

It does not follow the common standard regarding variable naming: for static final variables one has to follow this type of pattern: all uppercase with words separated by an underscore In these cases it should be:

public static final String MODULE = ICalWorker.class.getName();

Line 65

Missing Javadoc for class “public static final class ResponseProperties” and for constructor method “public ResponseProperties(int statusCode, String statusMessage)”.

Line 112 and 116

Missing Javadoc for “public static ResponseProperties createNotFoundResponse(String statusMessage)” and for “public static ResponseProperties createOkResponse(String statusMessage)”.

Literal “workEffortId” is duplicated eight times. Define a constant instead. Lines affected 132, 134, 147, 167, 236, 255, 303, 318.

Line 134

GenericValue publishProperties = EntityQuery.use(delegator).from("WorkEffort").where("workEffortId", workEffortId).queryOne();

This line’s length exceeds the maximum 120 allowed.

Line 143

Missing Javadoc for “public static void handleGetRequest(HttpServletRequest request, HttpServletResponse response, ServletContext context) throws ServletException, IOException”.

This line’s length exceeds the maximum 120 allowed.

Line 163

Missing Javadoc for “public static void handlePropFindRequest(HttpServletRequest request, HttpServletResponse response, ServletContext context) throws ServletException, IOException”.

This line’s length exceeds the maximum 120 allowed.

ServletContext context is never used in this method.

Line 182

Element etagElement = helper.createElementSetValue("D:getetag", String.*valueOf*(System.*currentTimeMillis*()));

This line’s length exceeds the maximum 120 allowed.

Line 188

Element lmElement = helper.createElementSetValue("D:getlastmodified", WebDavUtil.*formatDate*(WebDavUtil.*getRFC1123DateFormat*(), lastModified));

This line’s length exceeds the maximum 120 allowed.

Line 192

unSupportedProps.add(responseDocument.createElementNS(propElement.getNamespaceURI(), propElement.getTagName()));

This line’s length exceeds the maximum 120 allowed.

Line 196 and 200

if (supportedProps.size() > 0)

It should be used “if(supportedProps.isEmpty())” instead of “if (supportedProps.size() > 0)” indeed “supportedProps” is a List.

Line 208

Debug.logVerbose("[handlePropFindRequest] PROPFIND response:\r\n" + UtilXml.writeXmlDocument(responseDocument), module);

This line’s length exceeds the maximum 120 allowed.

Line 226

Missing Javadoc for “public static void handlePutRequest(HttpServletRequest request, HttpServletResponse response, ServletContext context) throws ServletException, IOException”.

This line’s length exceeds the maximum 120 allowed.

ServletContext context is useless in this method because the it is present here only for method “writeResponse(responseProps, request, response, context)” (line 246). But in writeResponse method context it is never used.

Line 263

This line’s length exceeds the maximum 120 allowed.

Line 269

Useless assignment of “userLogin” to null value. userLogin declaration should be placed at line 276 when it is assigned to a not null value.

Line 279

VisitHandler.getVisitor(request, response);

Return value of get method is not assigned to any variable.

Line 298

Replace the double quotes with the single quotes to use “indexOf(char)” that is faster than “indexOf(String)”.

Line 311

private static void writeResponse(ResponseProperties responseProps, HttpServletRequest request, HttpServletResponse response, ServletContext context) throws IOException

This line’s length exceeds the maximum 120 allowed.

ServletContext context is never used in this method.

General consideration about this class

This class handle the request generated by the front-end of the application. In particular, handle the WebDAV requests and responses about a calendar manager.

WebDAV is an extension of the HTTP that allows clients to perform remote Web content authoring operations. Infact in the class, we find a method “isValidRequest” that checks if the request is legit throw “setupRequest” and if the user is authorized throw “logInUser”.

We understood the behavior of the class, by analyzing all its public methods. We identify Get and Put requests that are typical of the HTTP and also the PropFind request that is typical of WebDAV. All this requests are related to a work effort calendar like reported in the small Javadoc related to this class and the attribute’s name of the requests. “handleGetRequest” manages to retrieve data from the calendar; “handlePutRequest” manages to insert data in the calendar and store it for future use; “handlePropFindRequest” is used to retrieve properties from the calendar and store it in a XML file.

In this class it is present also a nested class, called “Response Properties”, that represent the properties of the response like “statusCode” and “statusMessage”. This class have a problem with these attributes: they are public attributes and they do not expose any getter. It is better to avoid this behavior.  
Our suggest is to make all the attributes private and to expose getters.