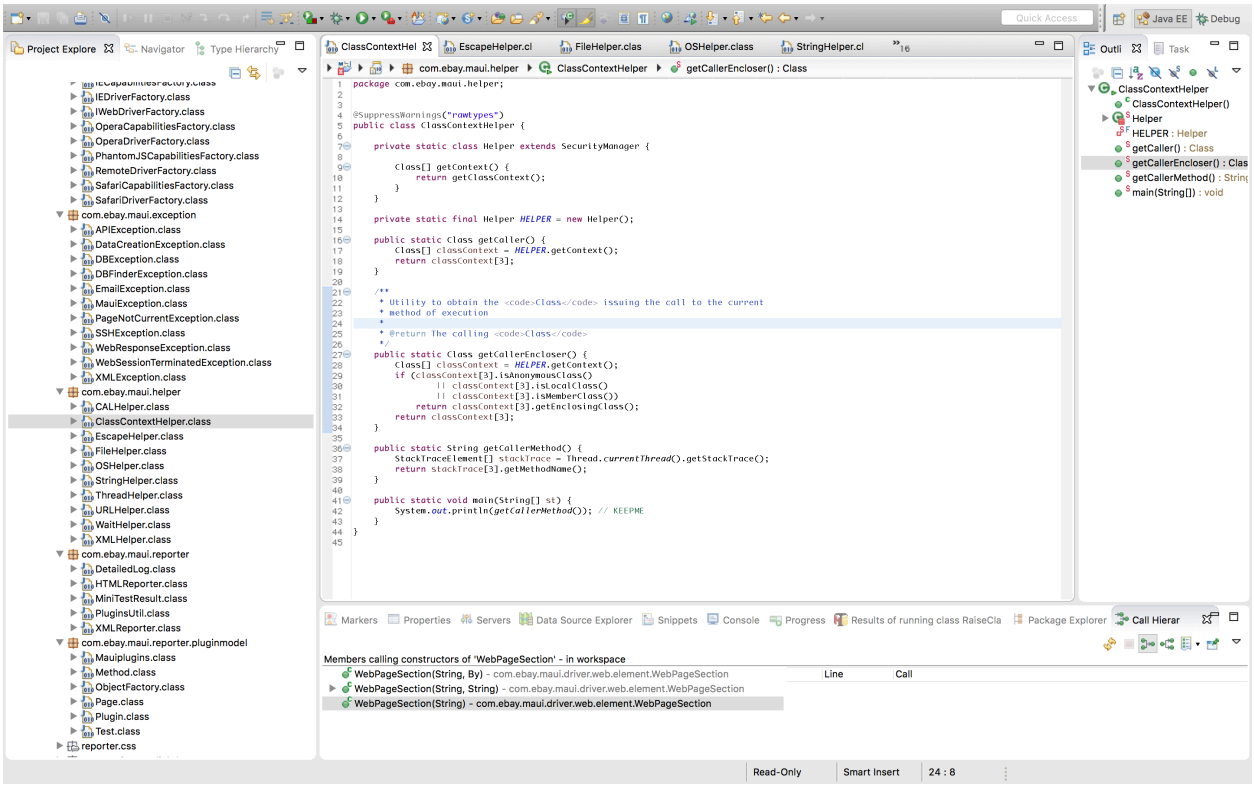
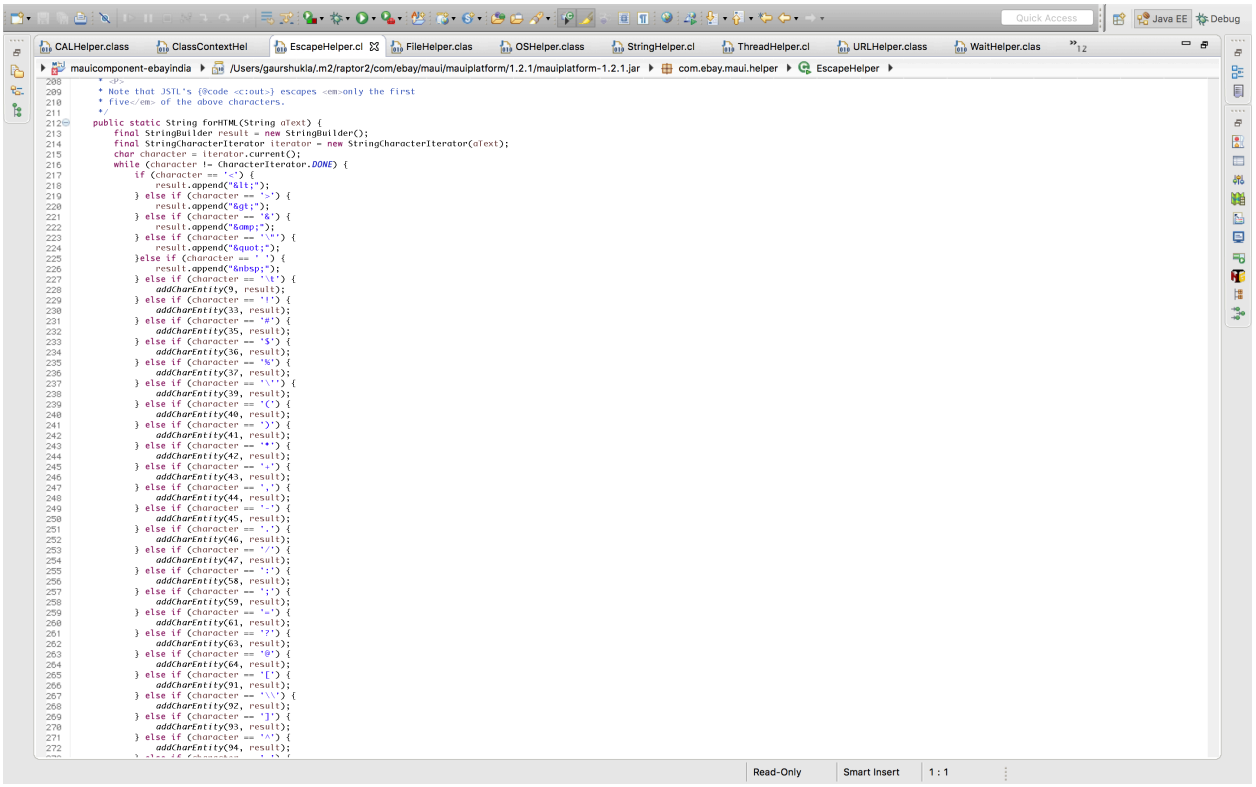
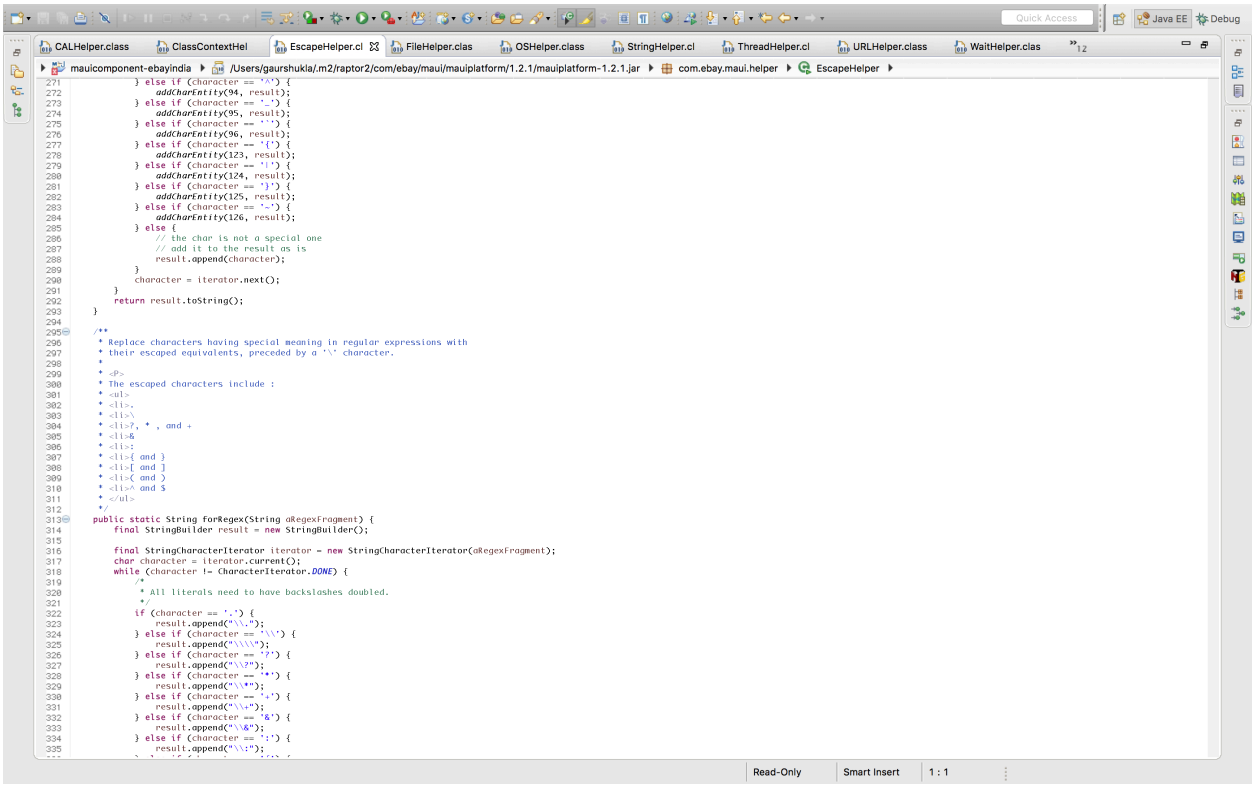
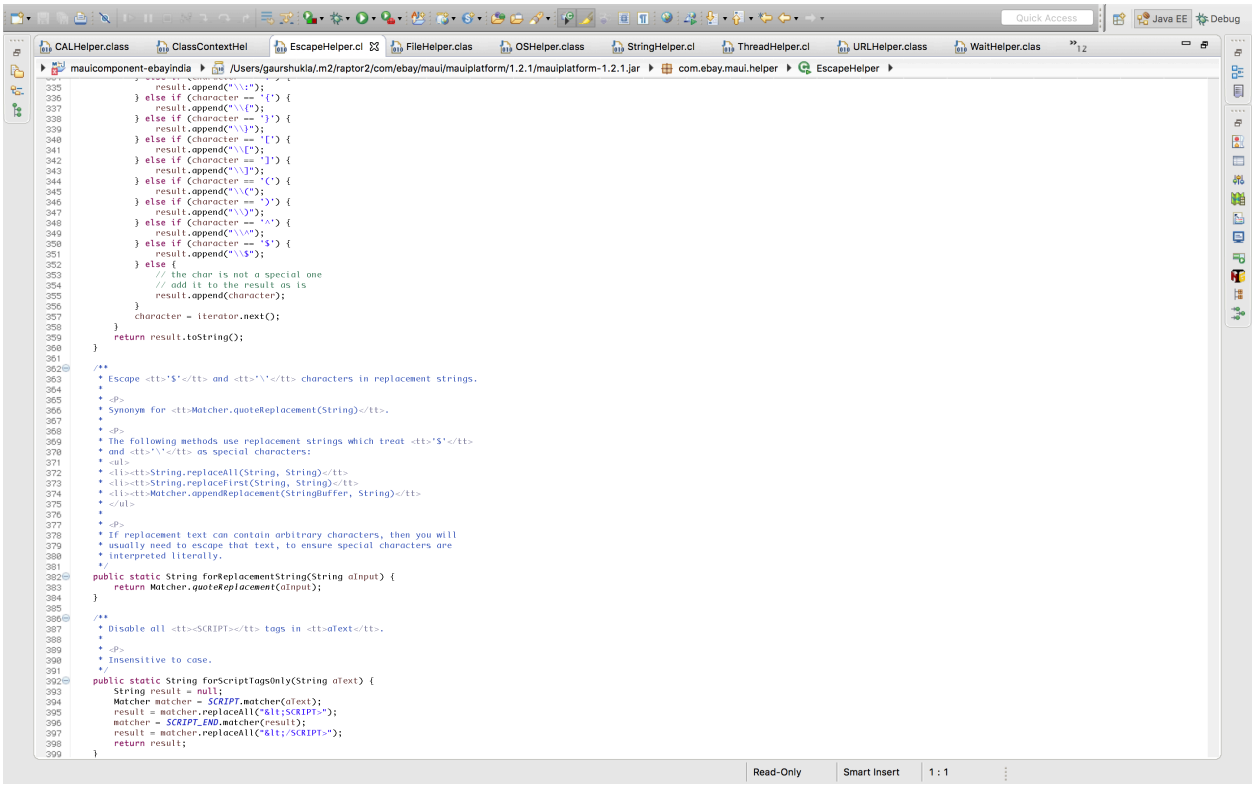
HELPER:

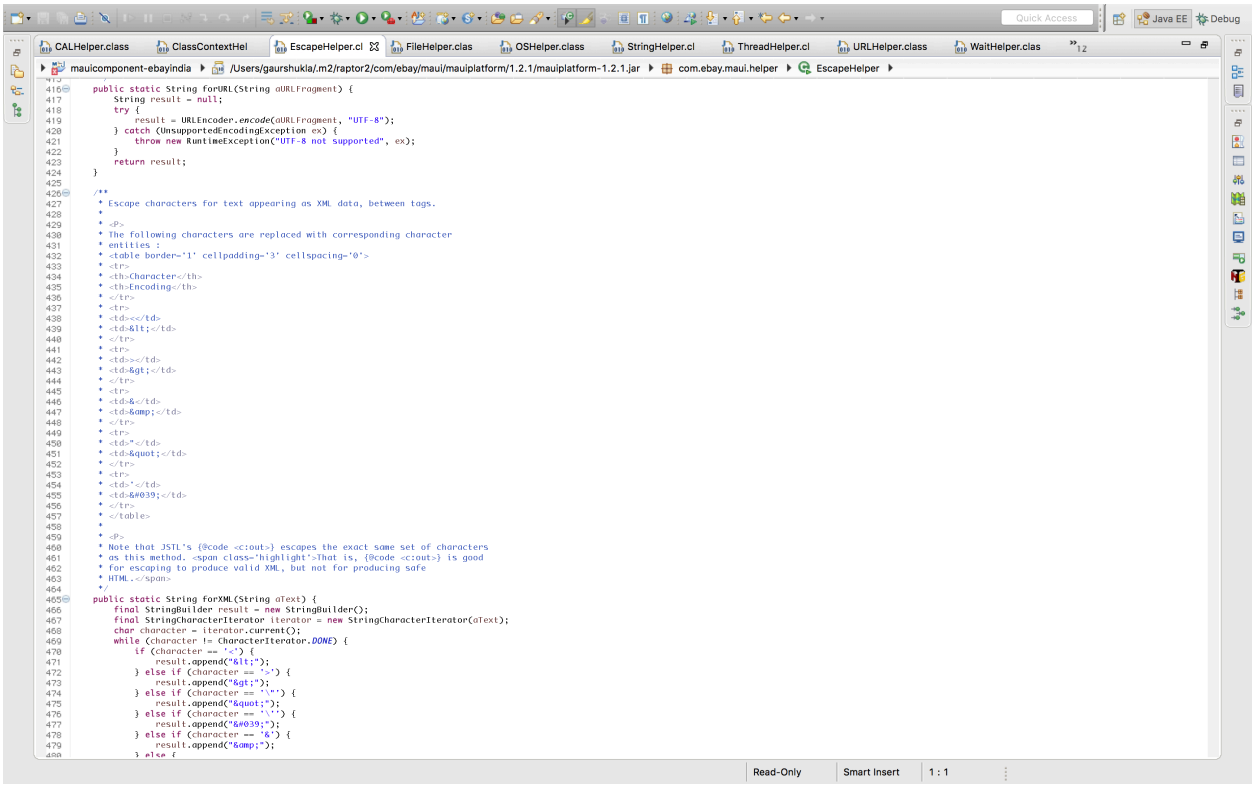

package com.ebay.maui.helper;
@SuppressWarnings("rawtypes")
public class ClassContextHelper {
private static class Helper extends SecurityManager {
Class[] getContext() {
return getClassContext();
}
}
private static final Helper HELPER = new Helper();
public static Class getCaller() {
Class[] classContext = HELPER.getContext();
return classContext[3];
}
/**
* Utility to obtain the <code>Class</code> issuing the call to the current
* method of execution
*
* @return The calling <code>Class</code>
*/
public static Class getCallerEncloser() {
Class[] classContext = HELPER.getContext();
if (classContext[3].isAnonymousClass()
|| classContext[3].isLocalClass()
|| classContext[3].isMemberClass())
return classContext[3].getEnclosingClass();
return classContext[3];
}
public static String getCallerMethod() {
StackTraceElement[] stackTrace = Thread.currentThread().getStackTrace();
return stackTrace[3].getMethodName();
}
public static void main(String[] st) {
System.out.println(getCallerMethod()); // KEEPME
}
}
Members calling constructors of 'WebPageSection' - in workspace
WebPageSection(String, By) - com.ebay.maui.driver.web.element.WebPageSection
WebPageSection(String, String) - com.ebay.maui.driver.web.element.WebPageSection
WebPageSection(String) - com.ebay.maui.driver.web.element.WebPageSection
Line
Call
Read-Only
Smart Insert
24 : 8

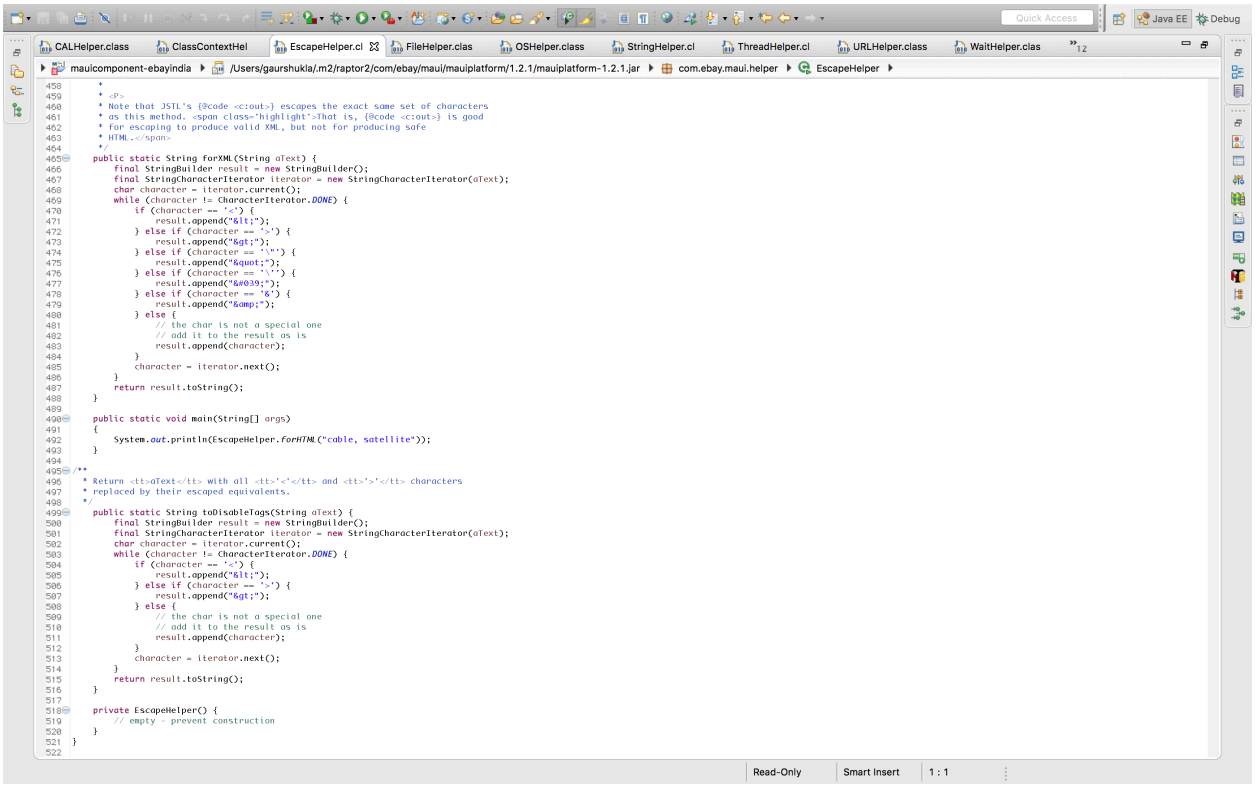

public class EscapeHelper {
private static final Pattern SCRIPT = Pattern.compile("<SCRIPT>", Pattern.CASE_INSENSITIVE);
private static final Pattern SCRIPT_END = Pattern.compile("</SCRIPT>", Pattern.CASE_INSENSITIVE);
private static void addCharEntity(Integer aIdx, StringBuilder aBuilder) {
String padding = "";
if (aIdx <= 9) {
padding = "00";
} else if (aIdx <= 99) {
padding = "0";
} else {
// no prefix
}
String number = padding + aIdx.toString();
aBuilder.append("&#" + number + ";");
}
/**
* Escape all ampersand characters in a URL.
*
* <P>
* Replaces all <tt>'&'</tt> characters with <tt>'&'</tt>.
*
* <P>
* An ampersand character may appear in the query string of a URL. The
* ampersand character is indeed valid in a URL.
* However, URLs usually appear as an <tt>HREF</tt> attribute, and
* such attributes have the additional constraint that ampersands
* must be escaped.
*
* <P>
* The JSTL <c:url> tag does indeed perform proper URL encoding of query
* parameters. But it does not, in general, produce text which is valid as
* an <tt>HREF</tt> attribute, simply because it does not escape the
* ampersand character. This is a nuisance when multiple query parameters
* appear in the URL, since it requires a little extra work.
*/
public static String forHrefAmpersand(String aURL) {
return aURL.replace("&", "&");
}
/**
* Escape characters for text appearing in HTML markup.
*
* <P>
* This method exists as a defence against Cross Site Scripting (XSS) hacks.
* The idea is to neutralize control characters commonly used by scripts,
* such that they will not be executed by the browser. This is done by
* replacing the control characters with their escaped equivalents.
*
* <P>
* The following characters are replaced with corresponding HTML character
* entities :
* <table border='1' cellpadding='3' cellspacing='0'>
* <tr>
* <th>Character</th>
* <th>Replacement</th>
* </tr>
* <tr>
* <td><</td>
* <td><</td>
* </tr>
* <tr>
Read-Only
Smart Insert
1 : 1

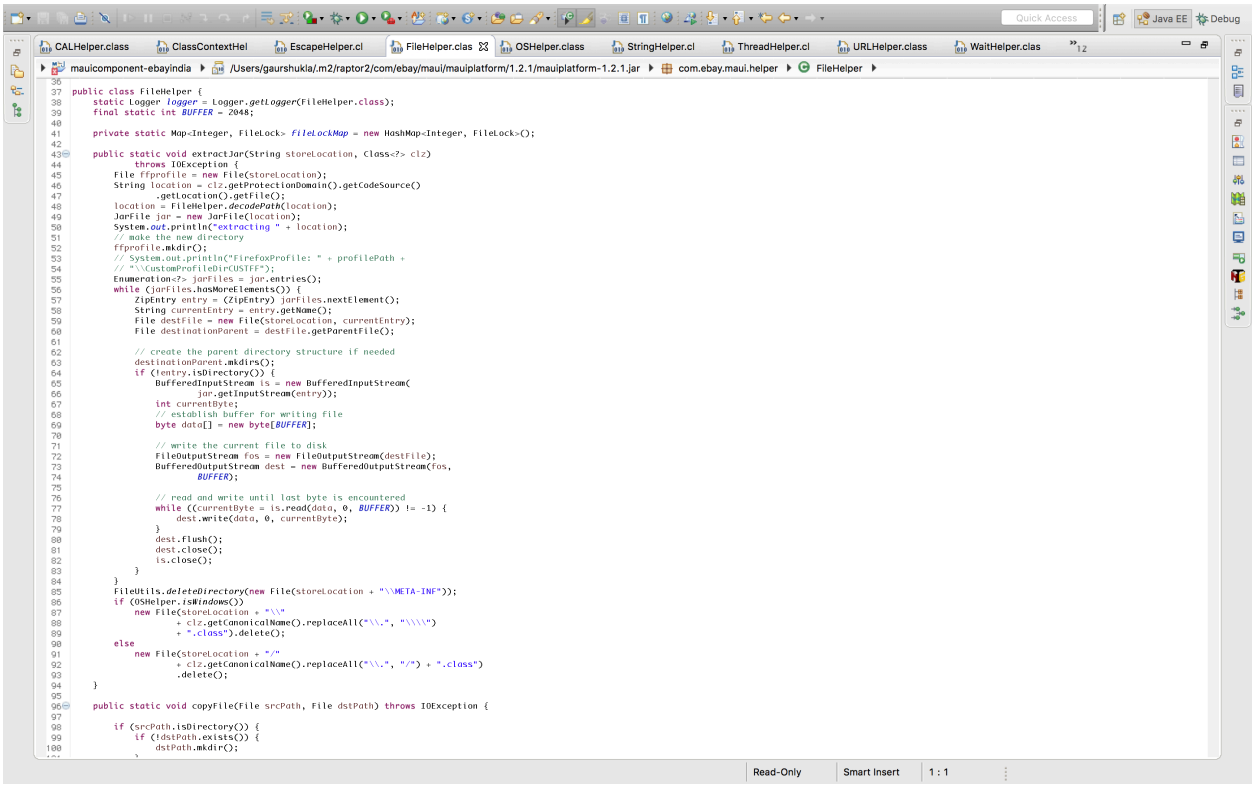

CALHelper.class
ClassContextHel
EscapeHelper.cl
FileHelper.clas
OSHelper.class
StringHelper.cl
ThreadHelper.cl
URLHelper.class
WaitHelper.clas
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
EscapeHelper
Read-Only
Smart Insert
1 : 1

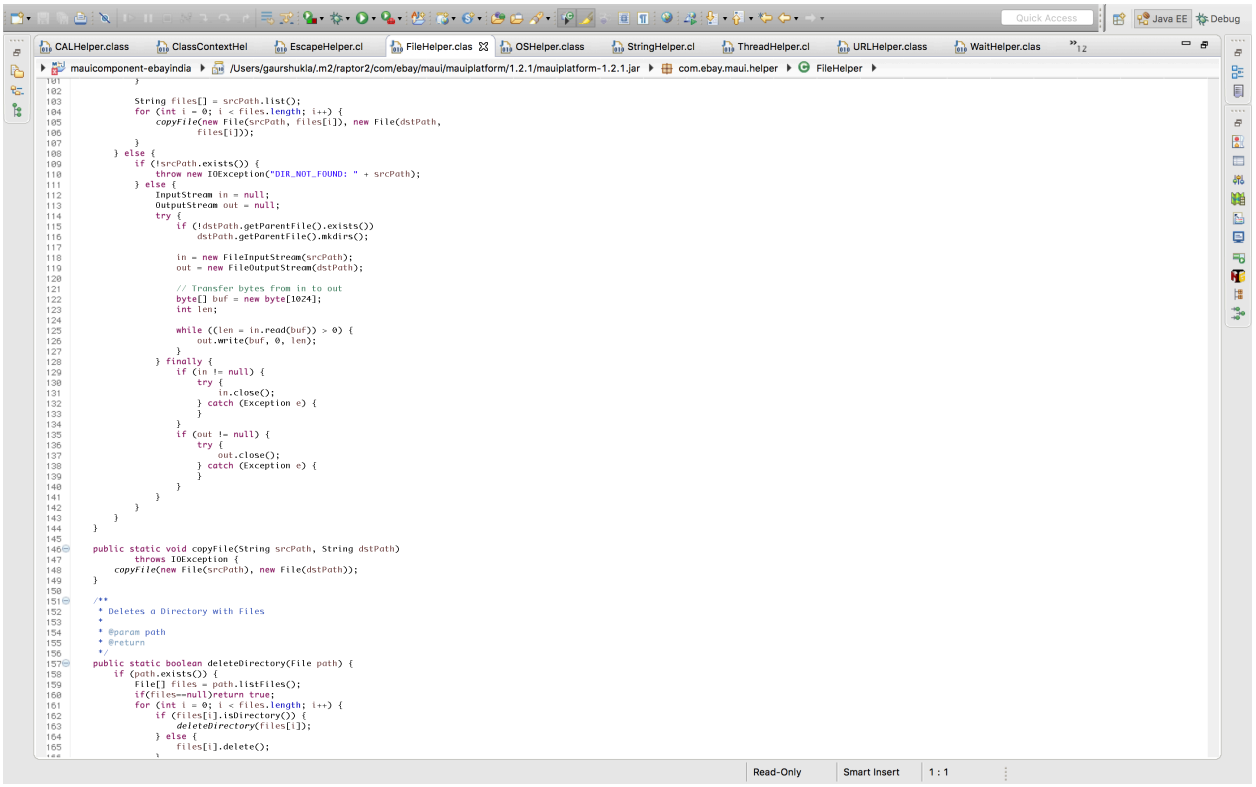

CALHelper.class
ClassContextHel
EscapeHelper.cl
FileHelper.clas
OSHelper.class
StringHelper.cl
ThreadHelper.cl
URLHelper.class
WaitHelper.clas
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
EscapeHelper
Read-Only
Smart Insert
1 : 1


CALHelper.class
ClassContextHel
EscapeHelper.cl
FileHelper.clas
OSHelper.class
StringHelper.cl
ThreadHelper.cl
URLHelper.class
WaitHelper.clas
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
EscapeHelper
Read-Only
Smart Insert
1 : 1


CALHelper.class
ClassContextHel
EscapeHelper.cl
FileHelper.clas
OSHelper.class
StringHelper.cl
ThreadHelper.cl
URLHelper.class
WaitHelper.clas
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
EscapeHelper
Read-Only
Smart Insert
1 : 1

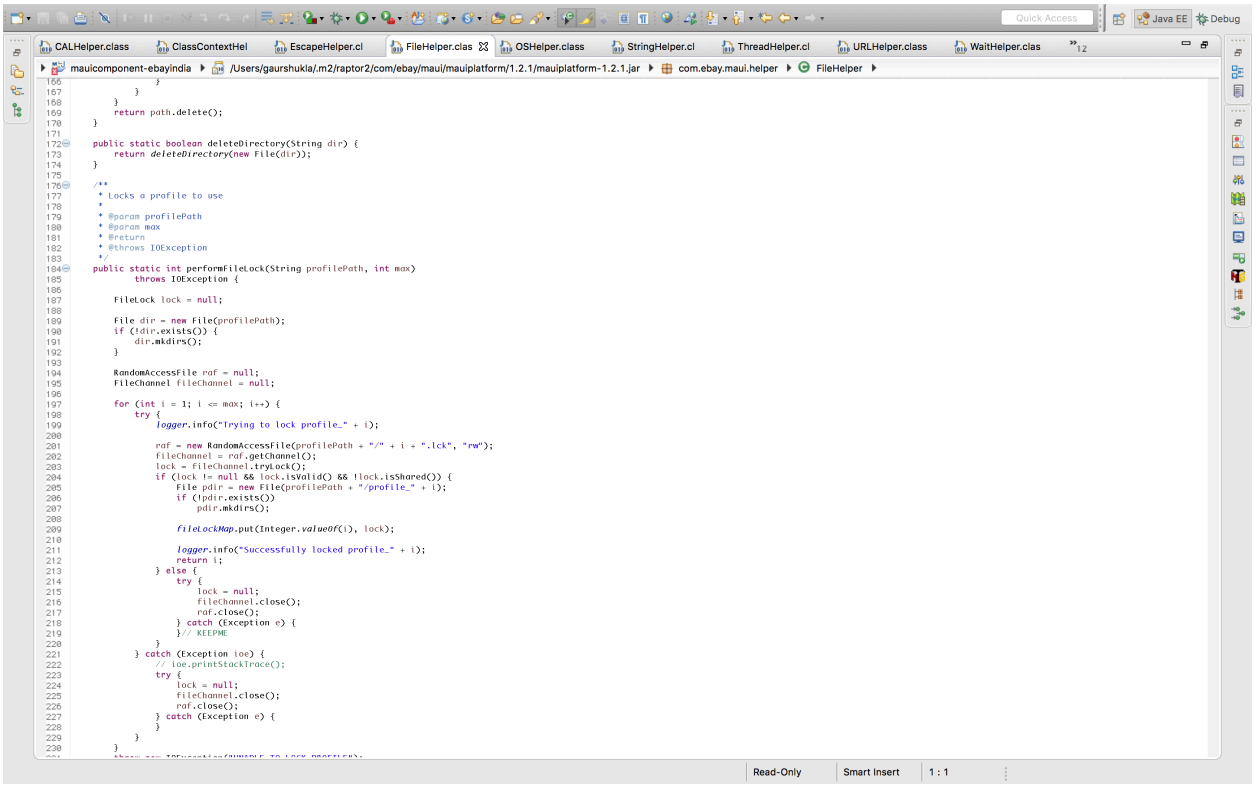

CALHelper.class
ClassContextHel
EscapeHelper.cl
FileHelper.clas
OSHelper.class
StringHelper.cl
ThreadHelper.cl
URLHelper.class
WaitHelper.clas
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
EscapeHelper
Read-Only
Smart Insert
1 : 1


CALHelper.class
ClassContextHel
EscapeHelper.cl
FileHelper.clas
OSHelper.class
StringHelper.cl
ThreadHelper.cl
URLHelper.class
WaitHelper.clas
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
FileHelper
Read-Only
Smart Insert
1 : 1



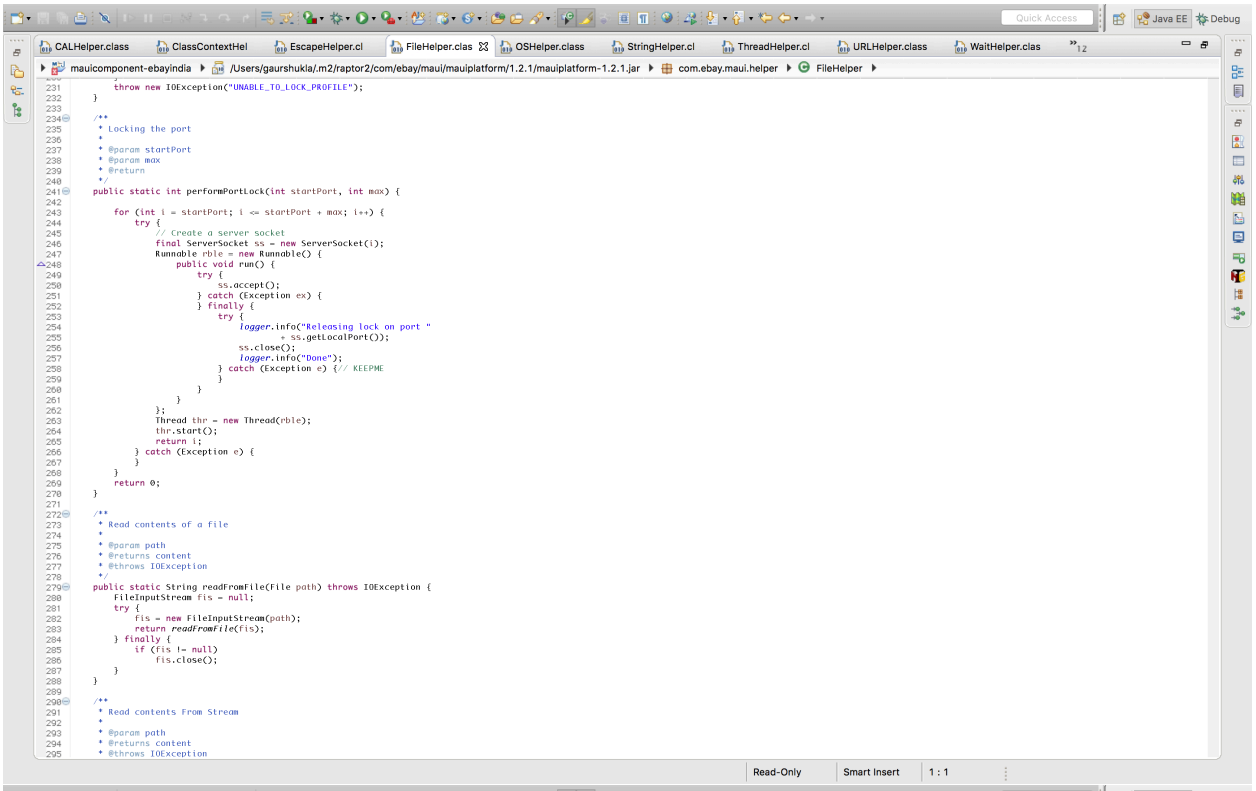
```
            }

            String files[] = srcPath.list();
            for (int i = 0; i < files.length; i++) {
                copyFile(new File(srcPath, files[i]), new File(dstPath,
                        files[i]));
            }
        } else {
            if (!srcPath.exists()) {
                throw new IOException("DIR_NOT_FOUND: " + srcPath);
            } else {
                InputStream in = null;
                OutputStream out = null;
                try {
                    if (!dstPath.getParentFile().exists())
                        dstPath.getParentFile().mkdirs();

                    in = new FileInputStream(srcPath);
                    out = new FileOutputStream(dstPath);

                    // Transfer bytes from in to out
                    byte[] buf = new byte[1024];
                    int len;

                    while ((len = in.read(buf)) > 0) {
                        out.write(buf, 0, len);
                    }
                } finally {
                    if (in != null) {
                        try {
                            in.close();
                        } catch (Exception e) {
                        }
                    }
                    if (out != null) {
                        try {
                            out.close();
                        } catch (Exception e) {
                        }
                    }
                }
            }
        }
    }

    public static void copyFile(String srcPath, String dstPath)
            throws IOException {
        copyFile(new File(srcPath), new File(dstPath));
    }

    /**
     * Deletes a Directory with Files
     *
     * @param path
     * @return
     */
    public static boolean deleteDirectory(File path) {
        if (path.exists()) {
            File[] files = path.listFiles();
            if(files==null)return true;
            for (int i = 0; i < files.length; i++) {
                if (files[i].isDirectory()) {
                    deleteDirectory(files[i]);
                } else {
                    files[i].delete();
                }
```

Read-Only | Smart Insert | 1 : 1

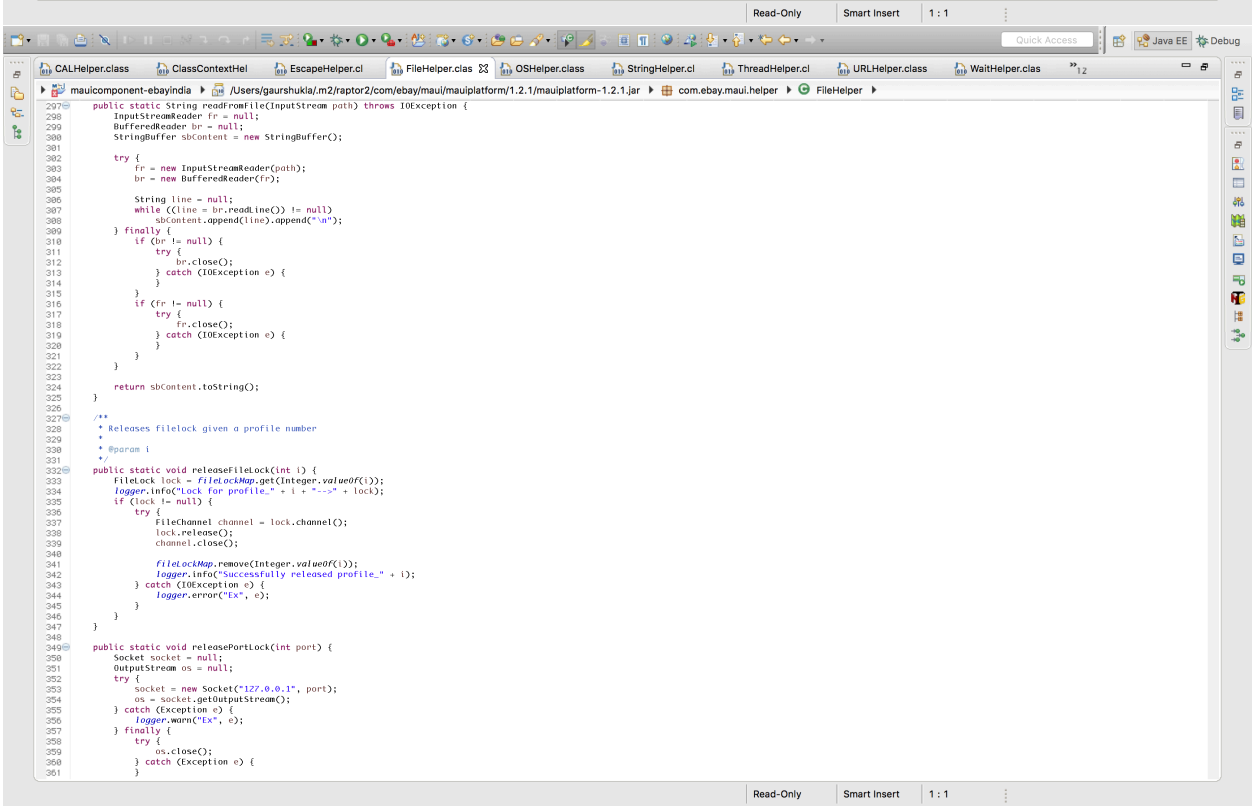


```
                }
            }
        }
        return path.delete();
    }

    public static boolean deleteDirectory(String dir) {
        return deleteDirectory(new File(dir));
    }

    /**
     * Locks a profile to use
     *
     * @param profilePath
     * @param max
     * @return
     * @throws IOException
     */
    public static int performFileLock(String profilePath, int max)
            throws IOException {

        FileLock lock = null;

        File dir = new File(profilePath);
        if (!dir.exists()) {
            dir.mkdirs();
        }

        RandomAccessFile raf = null;
        FileChannel fileChannel = null;

        for (int i = 1; i <= max; i++) {
            try {
                logger.info("Trying to lock profile_" + i);

                raf = new RandomAccessFile(profilePath + "/" + i + ".lck", "rw");
                fileChannel = raf.getChannel();
                lock = fileChannel.tryLock();
                if (lock != null && lock.isValid() && !lock.isShared()) {
                    File pdir = new File(profilePath + "/profile_" + i);
                    if (!pdir.exists())
                        pdir.mkdirs();

                    fileLockMap.put(Integer.valueOf(i), lock);

                    logger.info("Successfully locked profile_" + i);
                    return i;
                } else {
                    try {
                        lock = null;
                        fileChannel.close();
                        raf.close();
                    } catch (Exception e) {
                    }// KEEPME
                }
            } catch (Exception ioe) {
                // ioe.printStackTrace();
                try {
                    lock = null;
                    fileChannel.close();
                    raf.close();
                } catch (Exception e) {
                }
            }
        }
```

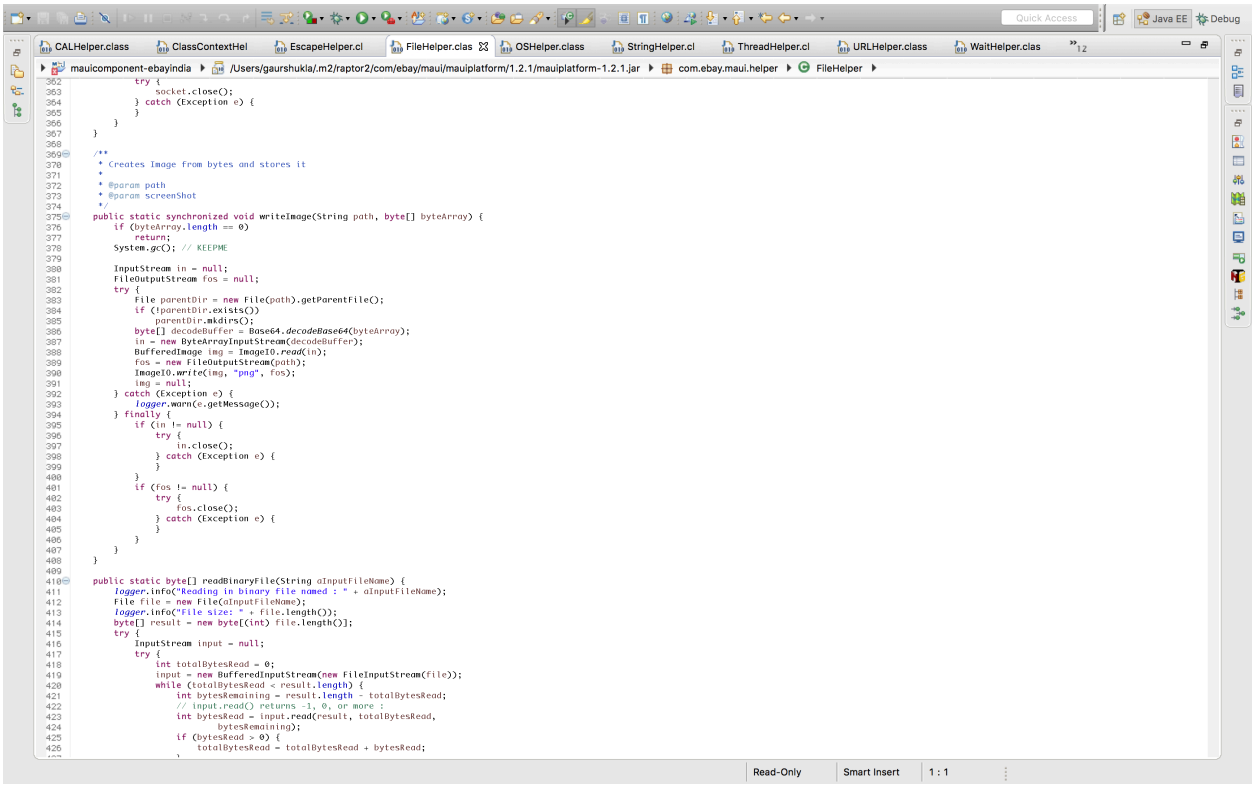
Read-Only | Smart Insert | 1 : 1



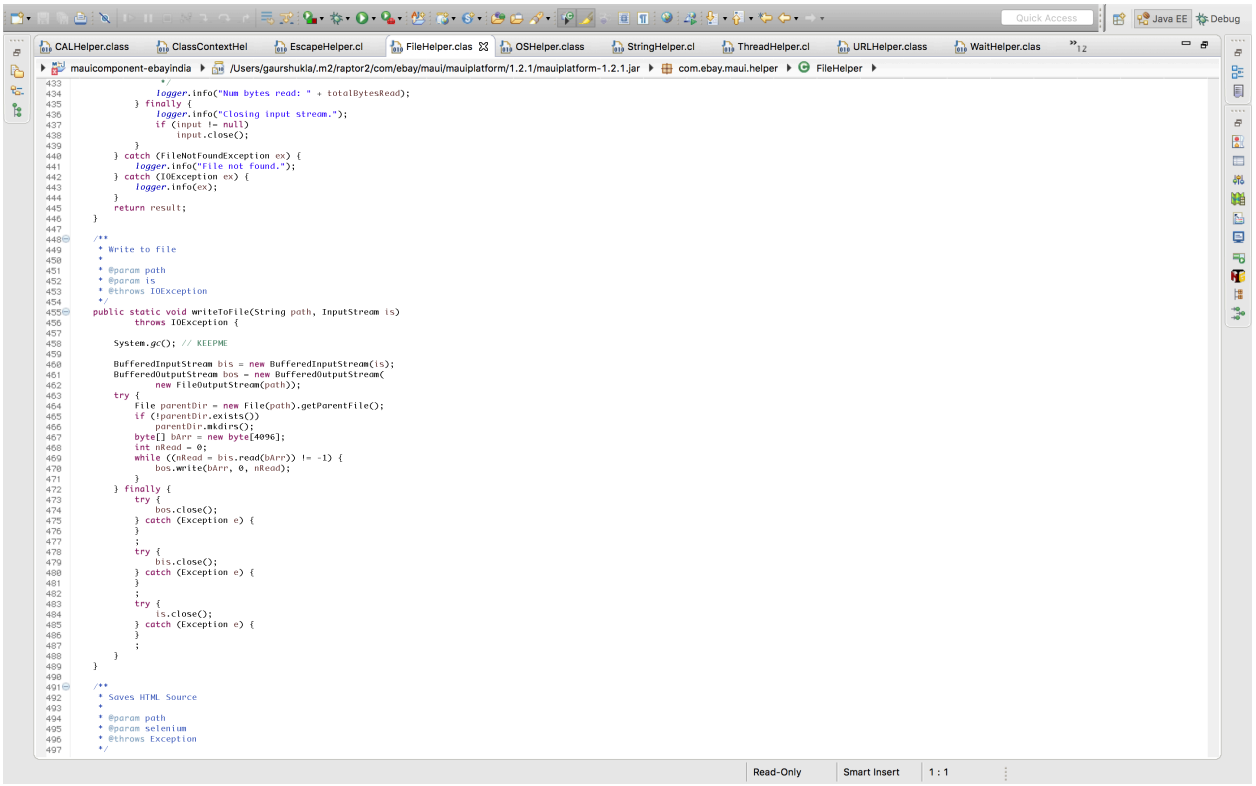
```
        throw new IOException("UNABLE_TO_LOCK_PROFILE");
    }

    /**
     * Locking the port
     *
     * @param startPort
     * @param max
     * @return
     */
    public static int performPortLock(int startPort, int max) {

        for (int i = startPort; i <= startPort + max; i++) {
            try {
                // Create a server socket
                final ServerSocket ss = new ServerSocket(i);
                Runnable rble = new Runnable() {
                    public void run() {
                        try {
                            ss.accept();
                        } catch (Exception ex) {
                        } finally {
                            try {
                                logger.info("Releasing lock on port "
                                        + ss.getLocalPort());
                                ss.close();
                                logger.info("Done");
                            } catch (Exception e) {// KEEPME
                            }
                        }
                    }
                };
                Thread thr = new Thread(rble);
                thr.start();
                return i;
            } catch (Exception e) {
            }
        }
        return 0;
    }

    /**
     * Read contents of a file
     *
     * @param path
     * @returns content
     * @throws IOException
     */
    public static String readFromFile(File path) throws IOException {
        FileInputStream fis = null;
        try {
            fis = new FileInputStream(path);
            return readFromFile(fis);
        } finally {
            if (fis != null)
                fis.close();
        }
    }

    /**
     * Read contents from Stream
     *
     * @param path
     * @returns content
     * @throws IOException
```



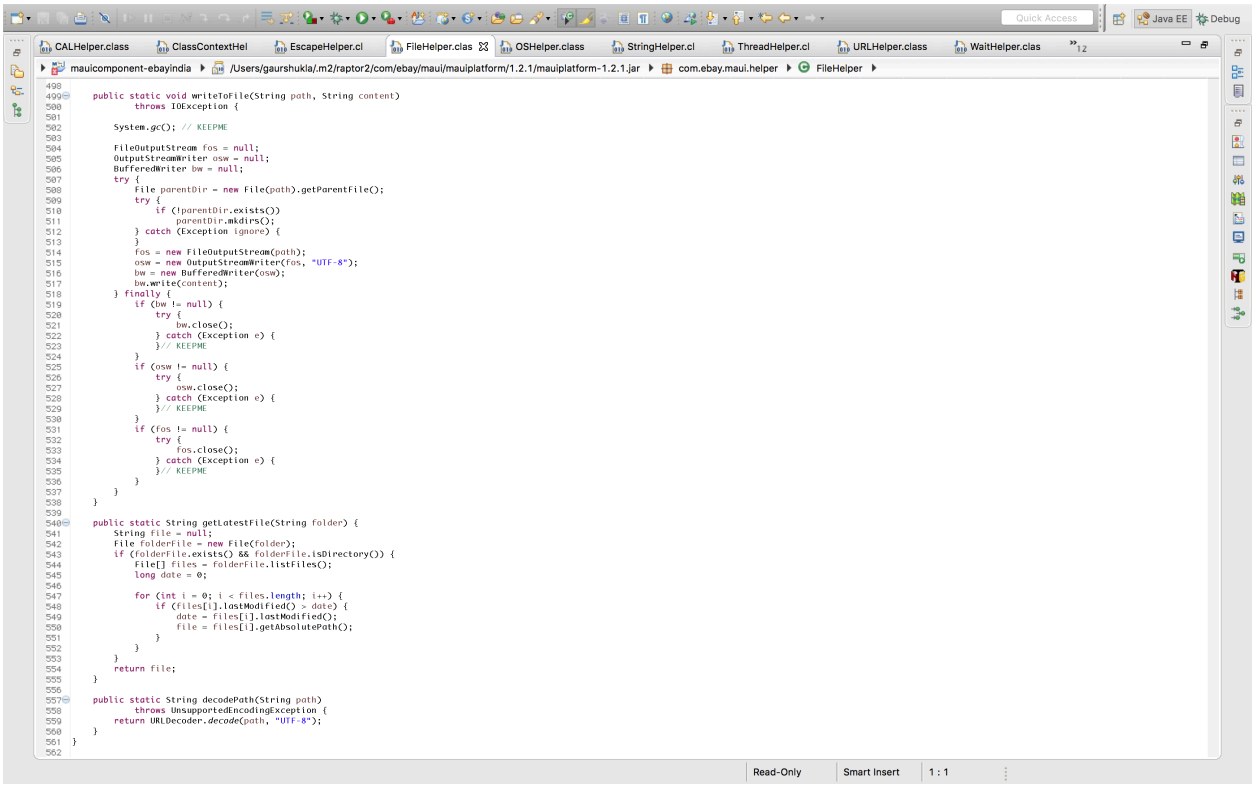
```
    public static String readFromFile(InputStream path) throws IOException {
        InputStreamReader fr = null;
        BufferedReader br = null;
        StringBuffer sbContent = new StringBuffer();

        try {
            fr = new InputStreamReader(path);
            br = new BufferedReader(fr);

            String line = null;
            while ((line = br.readLine()) != null)
                sbContent.append(line).append("\n");
        } finally {
            if (br != null) {
                try {
                    br.close();
                } catch (IOException e) {
                }
            }
            if (fr != null) {
                try {
                    fr.close();
                } catch (IOException e) {
                }
            }
        }

        return sbContent.toString();
    }

    /**
     * Releases filelock given a profile number
     *
     * @param i
     */
    public static void releaseFileLock(int i) {
        FileLock lock = fileLockMap.get(Integer.valueOf(i));
        logger.info("lock for profile_" + i + "-->" + lock);
        if (lock != null) {
            try {
                FileChannel channel = lock.channel();
                lock.release();
                channel.close();

                fileLockMap.remove(Integer.valueOf(i));
                logger.info("Successfully released profile_" + i);
            } catch (IOException e) {
                logger.error("Ex", e);
            }
        }
    }

    public static void releasePortLock(int port) {
        Socket socket = null;
        OutputStream os = null;
        try {
            socket = new Socket("127.0.0.1", port);
            os = socket.getOutputStream();
        } catch (Exception e) {
            logger.warn("Ex", e);
        } finally {
            try {
                os.close();
            } catch (Exception e) {
            }
```

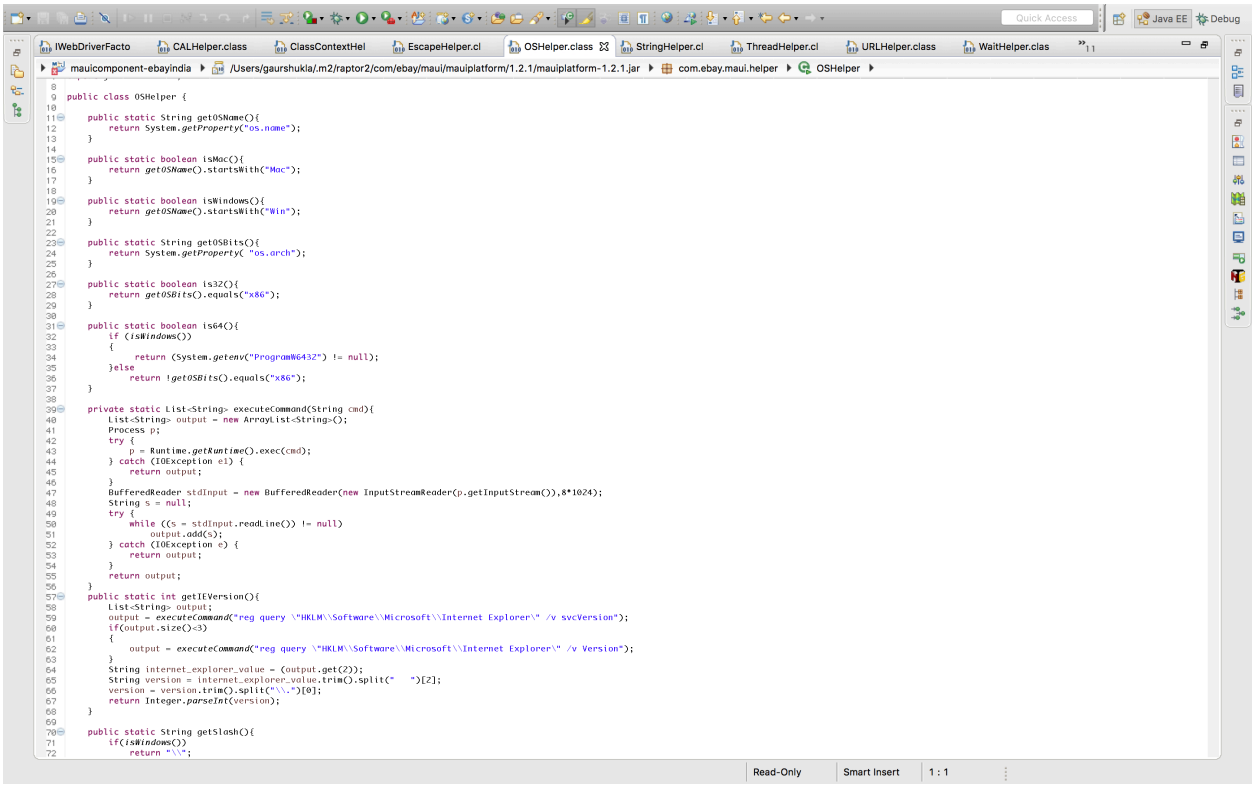


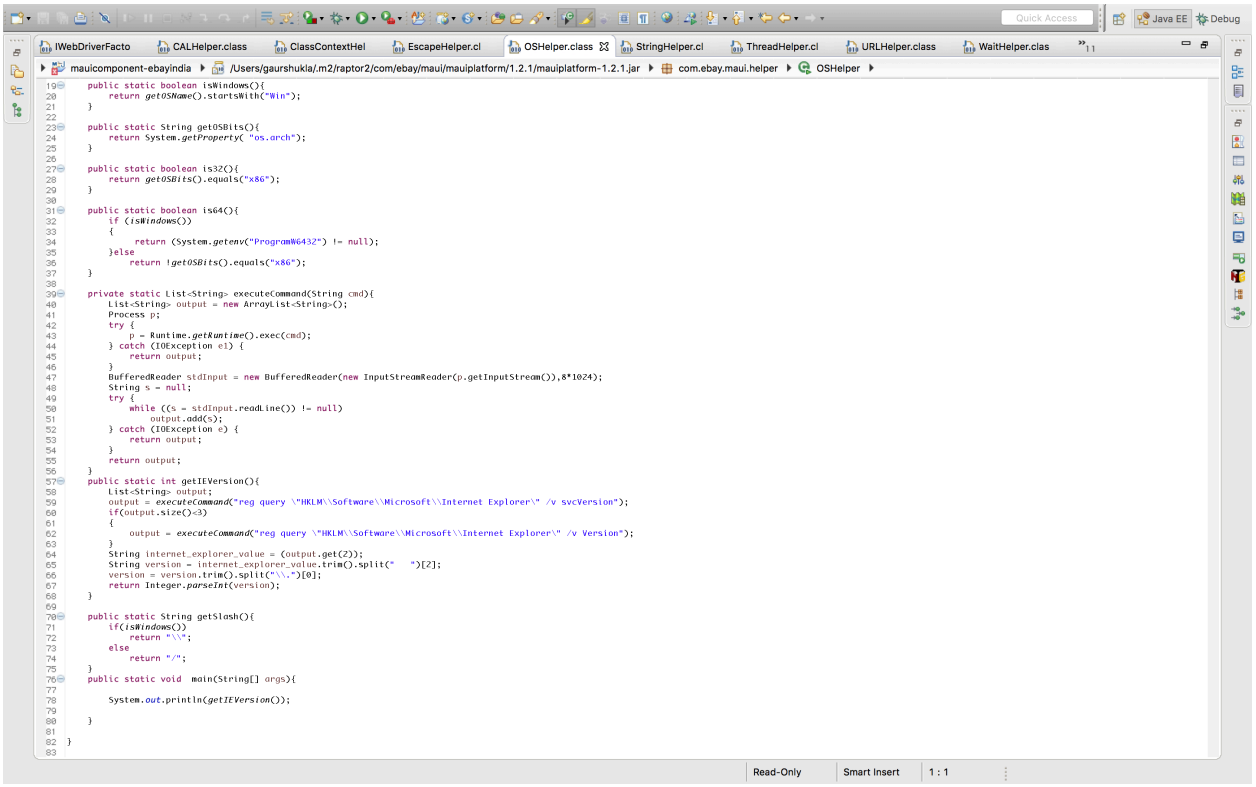

```
            try {
                socket.close();
            } catch (Exception e) {
            }
        }
    }

    /**
     * Creates Image from bytes and stores it
     *
     * @param path
     * @param screenShot
     */
    public static synchronized void writeImage(String path, byte[] byteArray) {
        if (byteArray.length == 0)
            return;
        System.gc(); // KEEPME

        InputStream in = null;
        FileOutputStream fos = null;
        try {
            File parentDir = new File(path).getParentFile();
            if (!parentDir.exists())
                parentDir.mkdirs();
            byte[] decodeBuffer = Base64.decodeBase64(byteArray);
            in = new ByteArrayInputStream(decodeBuffer);
            BufferedImage img = ImageIO.read(in);
            fos = new FileOutputStream(path);
            ImageIO.write(img, "png", fos);
            img = null;
        } catch (Exception e) {
            logger.warn(e.getMessage());
        } finally {
            if (in != null) {
                try {
                    in.close();
                } catch (Exception e) {
                }
            }
            if (fos != null) {
                try {
                    fos.close();
                } catch (Exception e) {
                }
            }
        }
    }

    public static byte[] readBinaryFile(String aInputFileName) {
        logger.info("Reading in binary file named : " + aInputFileName);
        File file = new File(aInputFileName);
        logger.info("File size: " + file.length());
        byte[] result = new byte[(int) file.length()];
        try {
            InputStream input = null;
            try {
                int totalBytesRead = 0;
                input = new BufferedInputStream(new FileInputStream(file));
                while (totalBytesRead < result.length) {
                    int bytesRemaining = result.length - totalBytesRead;
                    // input.read() returns -1, 0, or more :
                    int bytesRead = input.read(result, totalBytesRead,
                            bytesRemaining);
                    if (bytesRead > 0) {
                        totalBytesRead = totalBytesRead + bytesRead;
                    }
```



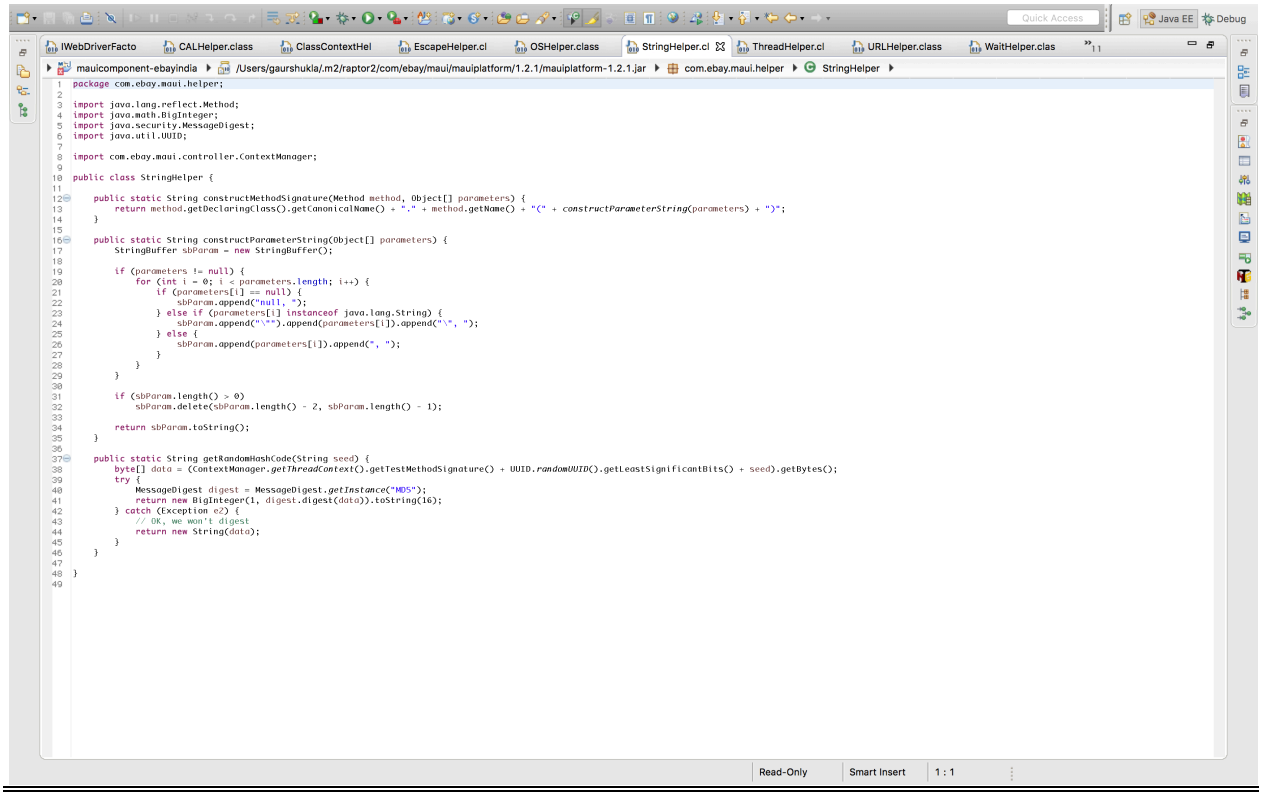

```
                }
                logger.info("Num bytes read: " + totalBytesRead);
            } finally {
                logger.info("Closing input stream.");
                if (input != null)
                    input.close();
            }
        } catch (FileNotFoundException ex) {
            logger.info("File not found.");
        } catch (IOException ex) {
            logger.info(ex);
        }
        return result;
    }

    /**
     * Write to file
     *
     * @param path
     * @param is
     * @throws IOException
     */
    public static void writeToFile(String path, InputStream is)
            throws IOException {

        System.gc(); // KEEPME

        BufferedInputStream bis = new BufferedInputStream(is);
        BufferedOutputStream bos = new BufferedOutputStream(
                new FileOutputStream(path));
        try {
            File parentDir = new File(path).getParentFile();
            if (!parentDir.exists())
                parentDir.mkdirs();
            byte[] bArr = new byte[4096];
            int nRead = 0;
            while ((nRead = bis.read(bArr)) != -1) {
                bos.write(bArr, 0, nRead);
            }
        } finally {
            try {
                bos.close();
            } catch (Exception e) {
            }
            ;
            try {
                bis.close();
            } catch (Exception e) {
            }
            ;
            try {
                is.close();
            } catch (Exception e) {
            }
            ;
        }
    }

    /**
     * Saves HTML Source
     *
     * @param path
     * @param selenium
     * @throws Exception
     */
```


CALHelper.class
ClassContextHel
EscapeHelper.cl
FileHelper.clas
OSHelper.class
StringHelper.cl
ThreadHelper.cl
URLHelper.class
WaitHelper.clas
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
FileHelper
Read-Only
Smart Insert
1 : 1

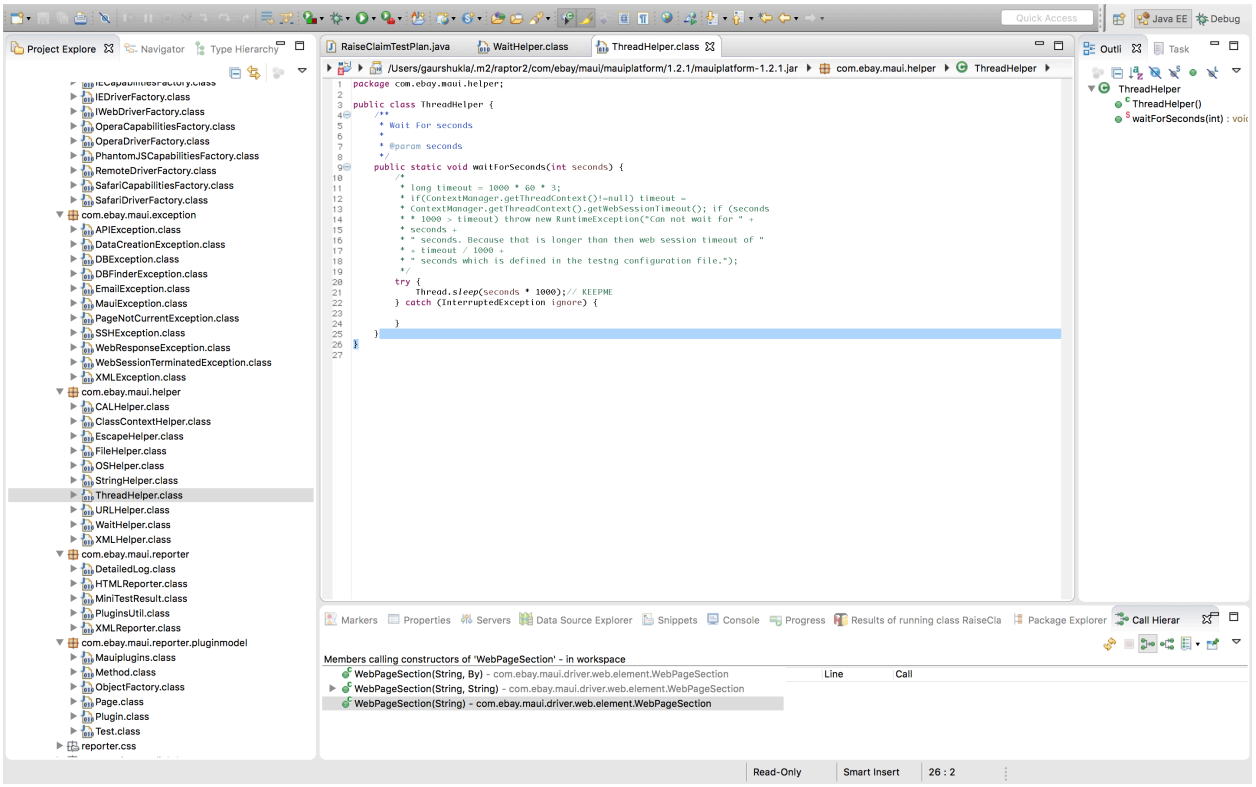

IWebDriverFacto
CALHelper.class
ClassContextHel
EscapeHelper.cl
OSHelper.class
StringHelper.cl
ThreadHelper.cl
URLHelper.class
WaitHelper.clas
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
OSHelper
Read-Only
Smart Insert
1 : 1



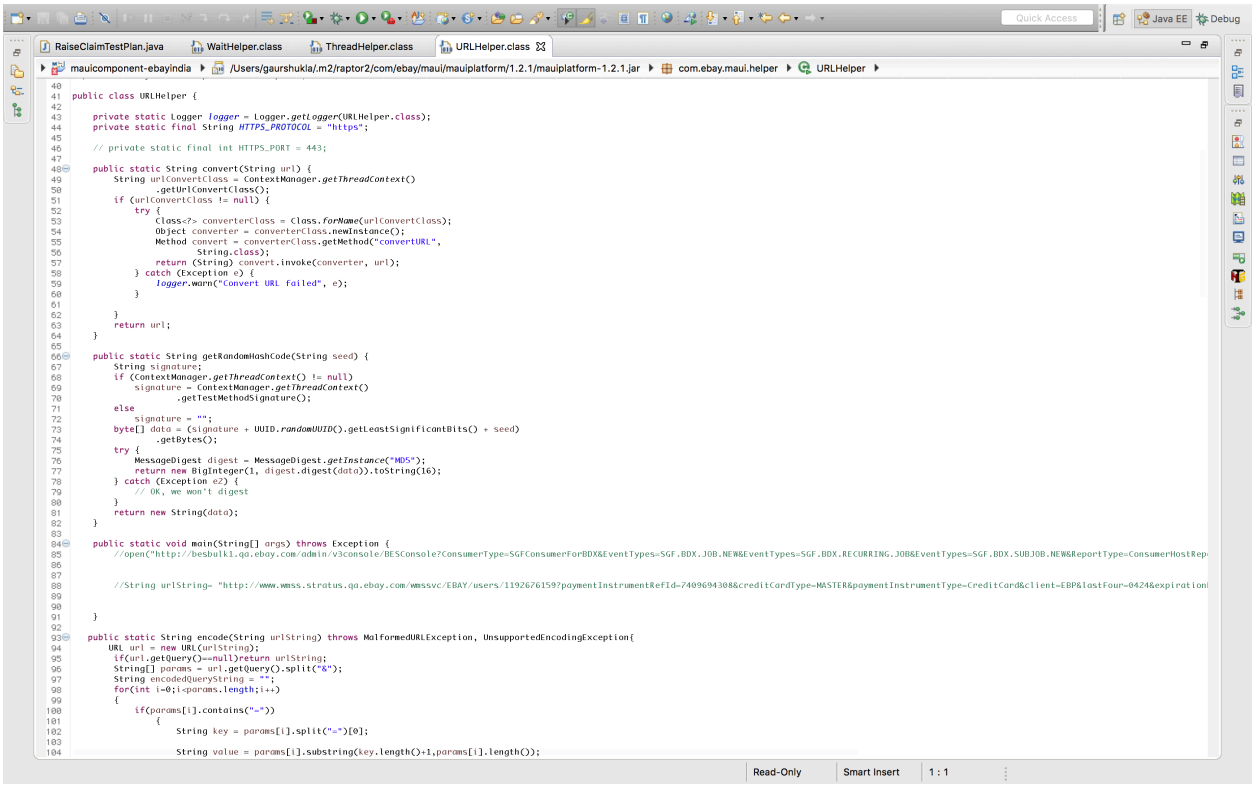
```
    public static boolean isWindows(){
        return getOSName().startsWith("Win");
    }

    public static String getOSBits(){
        return System.getProperty( "os.arch");
    }

    public static boolean is32(){
        return getOSBits().equals("x86");
    }

    public static boolean is64(){
        if (isWindows())
        {
            return (System.getenv("ProgramW6432") != null);
        }else
            return !getOSBits().equals("x86");
    }

    private static List<String> executeCommand(String cmd){
        List<String> output = new ArrayList<String>();
        Process p;
        try {
            p = Runtime.getRuntime().exec(cmd);
        } catch (IOException e1) {
            return output;
        }
        BufferedReader stdInput = new BufferedReader(new InputStreamReader(p.getInputStream()),8*1024);
        String s = null;
        try {
            while ((s = stdInput.readLine()) != null)
                output.add(s);
        } catch (IOException e) {
            return output;
        }
        return output;
    }
    public static int getIEVersion(){
        List<String> output;
        output = executeCommand("reg query \"HKLM\\Software\\Microsoft\\Internet Explorer\" /v svcVersion");
        if(output.size()<3)
        {
            output = executeCommand("reg query \"HKLM\\Software\\Microsoft\\Internet Explorer\" /v Version");
        }
        String internet_explorer_value = (output.get(2));
        String version = internet_explorer_value.trim().split("    ")[2];
        version = version.trim().split("\\.")[0];
        return Integer.parseInt(version);
    }

    public static String getSlash(){
        if(isWindows())
            return "\\";
        else
            return "/";
    }
    public static void  main(String[] args){

        System.out.println(getIEVersion());

    }
}
```



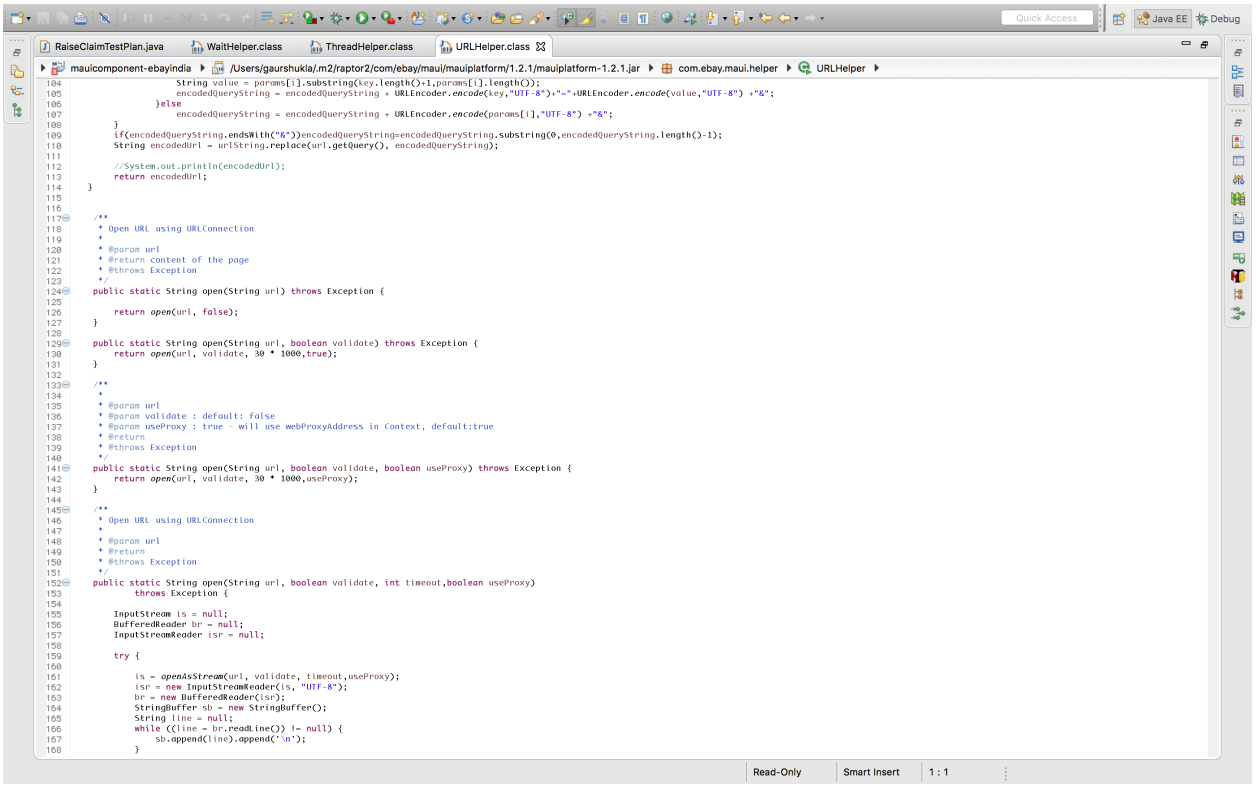
```
package com.ebay.maui.helper;

import java.lang.reflect.Method;
import java.math.BigInteger;
import java.security.MessageDigest;
import java.util.UUID;

import com.ebay.maui.controller.ContextManager;

public class StringHelper {

    public static String constructMethodSignature(Method method, Object[] parameters) {
        return method.getDeclaringClass().getCanonicalName() + "." + method.getName() + "(" + constructParameterString(parameters) + ")";
    }

    public static String constructParameterString(Object[] parameters) {
        StringBuffer sbParam = new StringBuffer();

        if (parameters != null) {
            for (int i = 0; i < parameters.length; i++) {
                if (parameters[i] == null) {
                    sbParam.append("null, ");
                } else if (parameters[i] instanceof java.lang.String) {
                    sbParam.append("\"").append(parameters[i]).append("\", ");
                } else {
                    sbParam.append(parameters[i]).append(", ");
                }
            }
        }

        if (sbParam.length() > 0)
            sbParam.delete(sbParam.length() - 2, sbParam.length() - 1);

        return sbParam.toString();
    }

    public static String getRandomHashCode(String seed) {
        byte[] data = (ContextManager.getThreadContext().getTestMethodSignature() + UUID.randomUUID().getLeastSignificantBits() + seed).getBytes();
        try {
            MessageDigest digest = MessageDigest.getInstance("MD5");
            return new BigInteger(1, digest.digest(data)).toString(16);
        } catch (Exception e2) {
            // OK, we won't digest
            return new String(data);
        }
    }

}
```

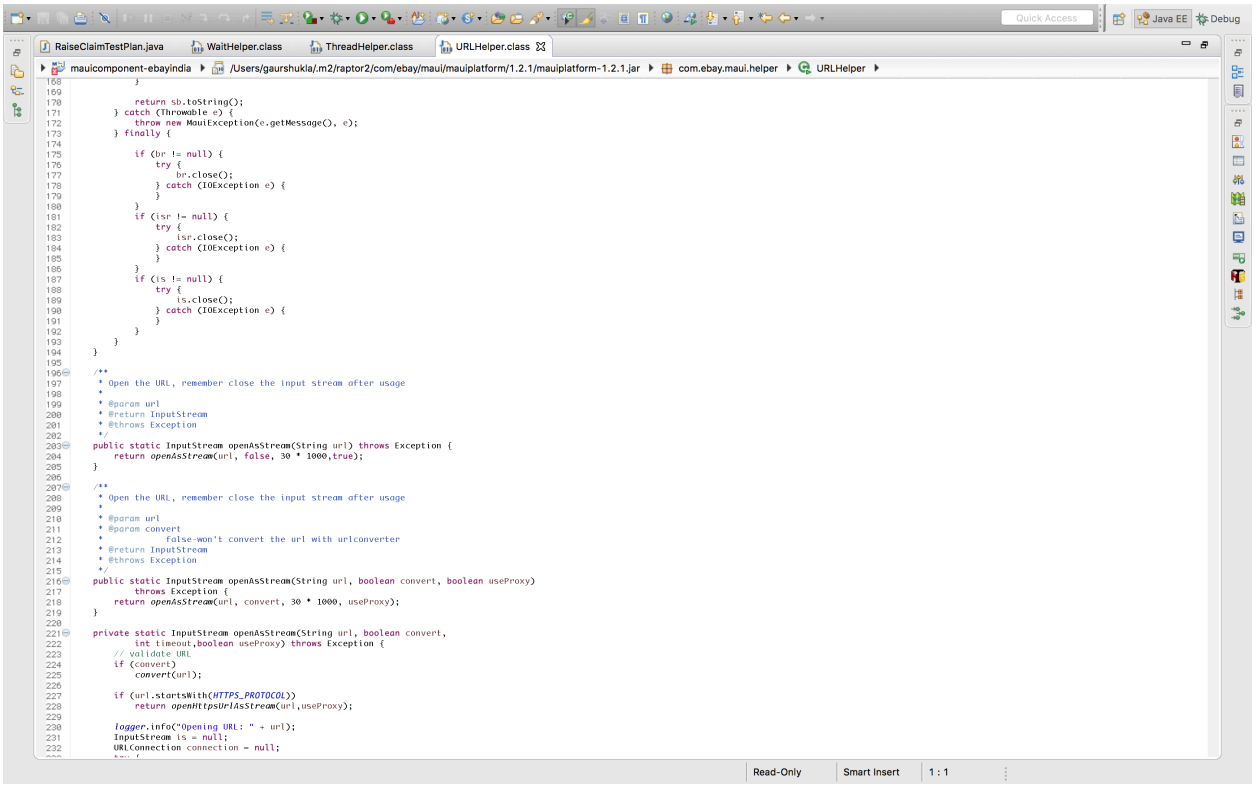


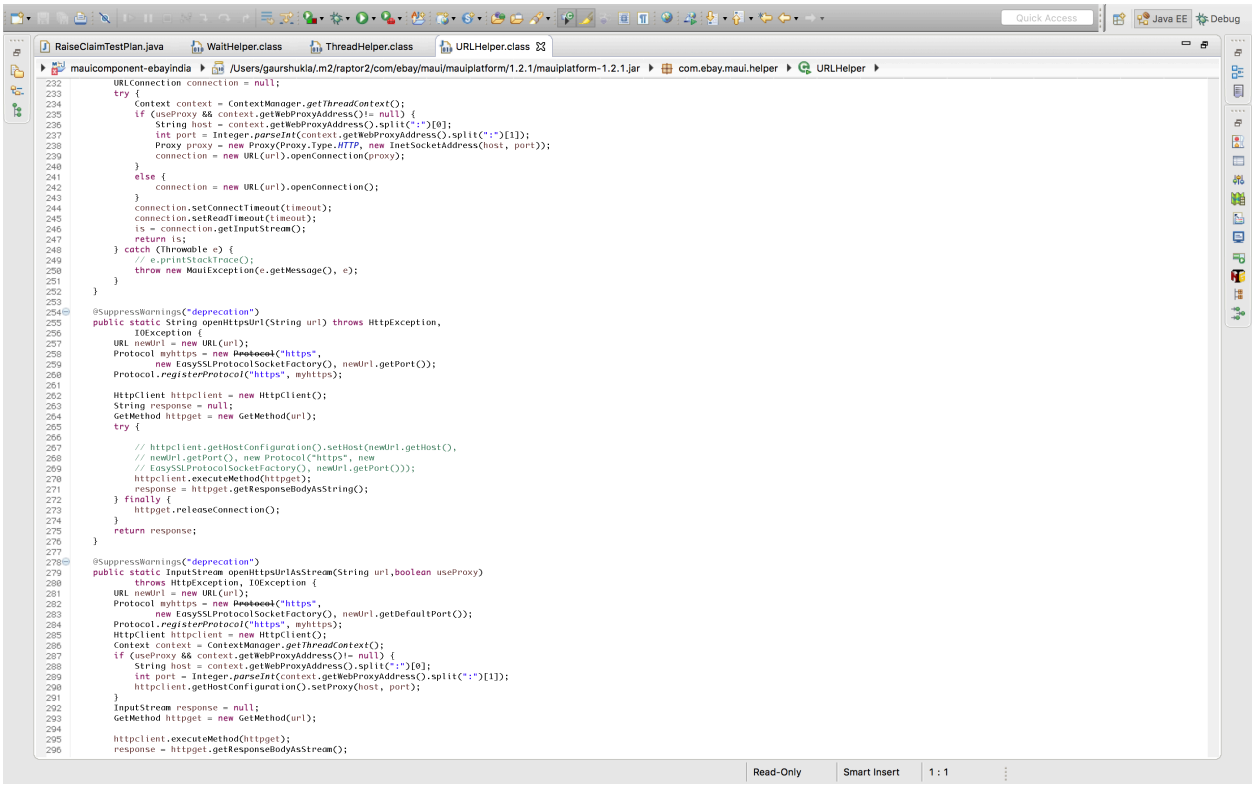
```
package com.ebay.maui.helper;

public class ThreadHelper {
    /**
     * Wait For seconds
     *
     * @param seconds
     */
    public static void waitForSeconds(int seconds) {
        /*
         * long timeout = 1000 * 60 * 3;
         * if(ContextManager.getThreadContext()!=null) timeout =
         * ContextManager.getThreadContext().getWebSessionTimeout(); if (seconds
         * * 1000 > timeout) throw new RuntimeException("Can not wait for " +
         * seconds +
         * " seconds. Because that is longer than then web session timeout of "
         * + timeout / 1000 +
         * " seconds which is defined in the testng configuration file.");
         */
        try {
            Thread.sleep(seconds * 1000);// KEEPME
        } catch (InterruptedException ignore) {

        }
    }
}
```



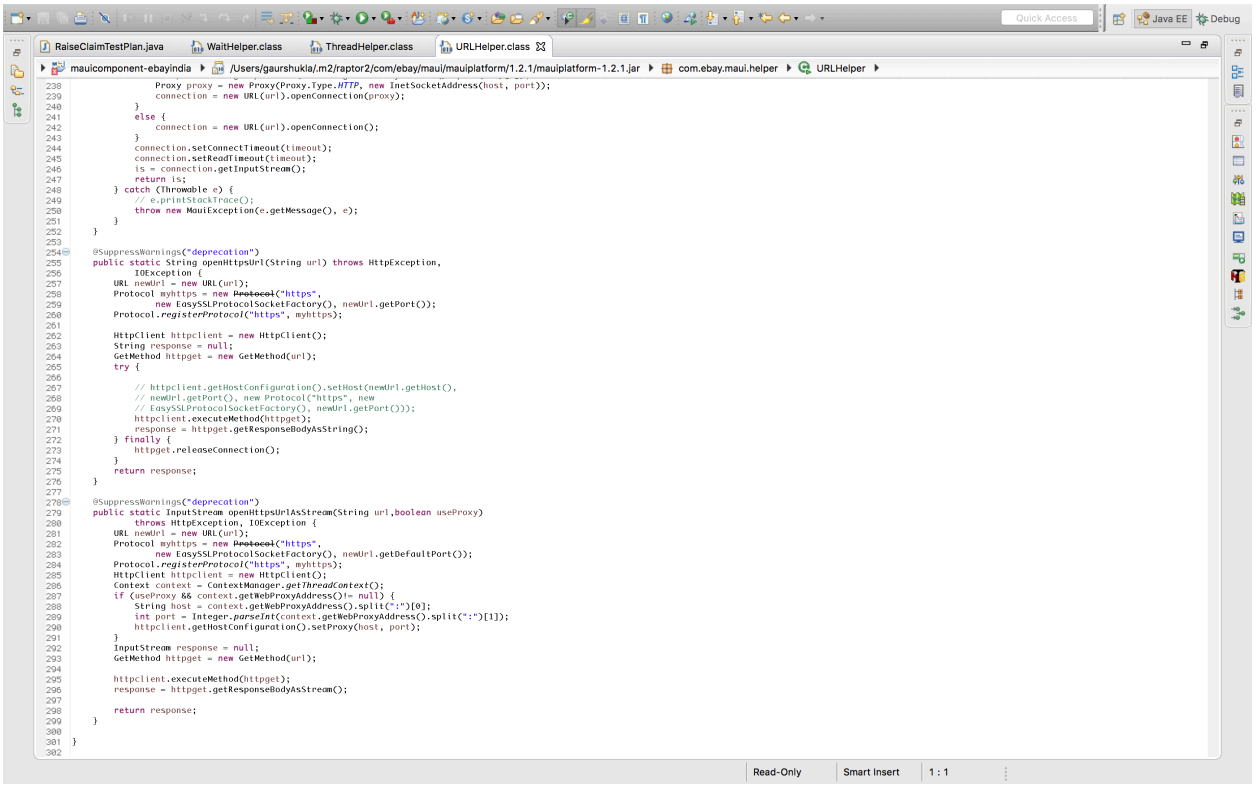
```
public class URLHelper {

    private static Logger logger = Logger.getLogger(URLHelper.class);
    private static final String HTTPS_PROTOCOL = "https";

    // private static final int HTTPS_PORT = 443;

    public static String convert(String url) {
        String urlConvertClass = ContextManager.getThreadContext()
                .getUrlConvertClass();
        if (urlConvertClass != null) {
            try {
                Class<?> converterClass = Class.forName(urlConvertClass);
                Object converter = converterClass.newInstance();
                Method convert = converterClass.getMethod("convertURL",
                        String.class);
                return (String) convert.invoke(converter, url);
            } catch (Exception e) {
                logger.warn("Convert URL failed", e);
            }

        }
        return url;
    }

    public static String getRandomHashCode(String seed) {
        String signature;
        if (ContextManager.getThreadContext() != null)
            signature = ContextManager.getThreadContext()
                    .getTestMethodSignature();
        else
            signature = "";
        byte[] data = (signature + UUID.randomUUID().getLeastSignificantBits() + seed)
                .getBytes();
        try {
            MessageDigest digest = MessageDigest.getInstance("MD5");
            return new BigInteger(1, digest.digest(data)).toString(16);
        } catch (Exception e2) {
            // OK, we won't digest
        }
        return new String(data);
    }

    public static void main(String[] args) throws Exception {
        //open("http://besbulk1.qa.ebay.com/admin/v3console/BESConsole?ConsumerType=SGFConsumerForBDX&EventTypes=SGF.BDX.JOB.NEW&EventTypes=SGF.BDX.RECURRING.JOB&EventTypes=SGF.BDX.SUBJOB.NEW&ReportType=ConsumerHostRep

        //String urlString= "http://www.wmss.stratus.qa.ebay.com/wmssvc/EBAY/users/1192676159?paymentInstrumentRefId=740969430&creditCardType=MASTER&paymentInstrumentType=CreditCard&client=EBP&lastFour=0424&expiration

    }

    public static String encode(String urlString) throws MalformedURLException, UnsupportedEncodingException{
        URL url = new URL(urlString);
        if(url.getQuery()==null)return urlString;
        String[] params = url.getQuery().split("&");
        String encodedQueryString = "";
        for(int i=0;i<params.length;i++)
        {
            if(params[i].contains("="))
                {
                    String key = params[i].split("=")[0];

                    String value = params[i].substring(key.length()+1,params[i].length());
```


RaiseClaimTestPlan.java
WaitHelper.class
ThreadHelper.class
URLHelper.class
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
URLHelper
Read-Only
Smart Insert
1 : 1

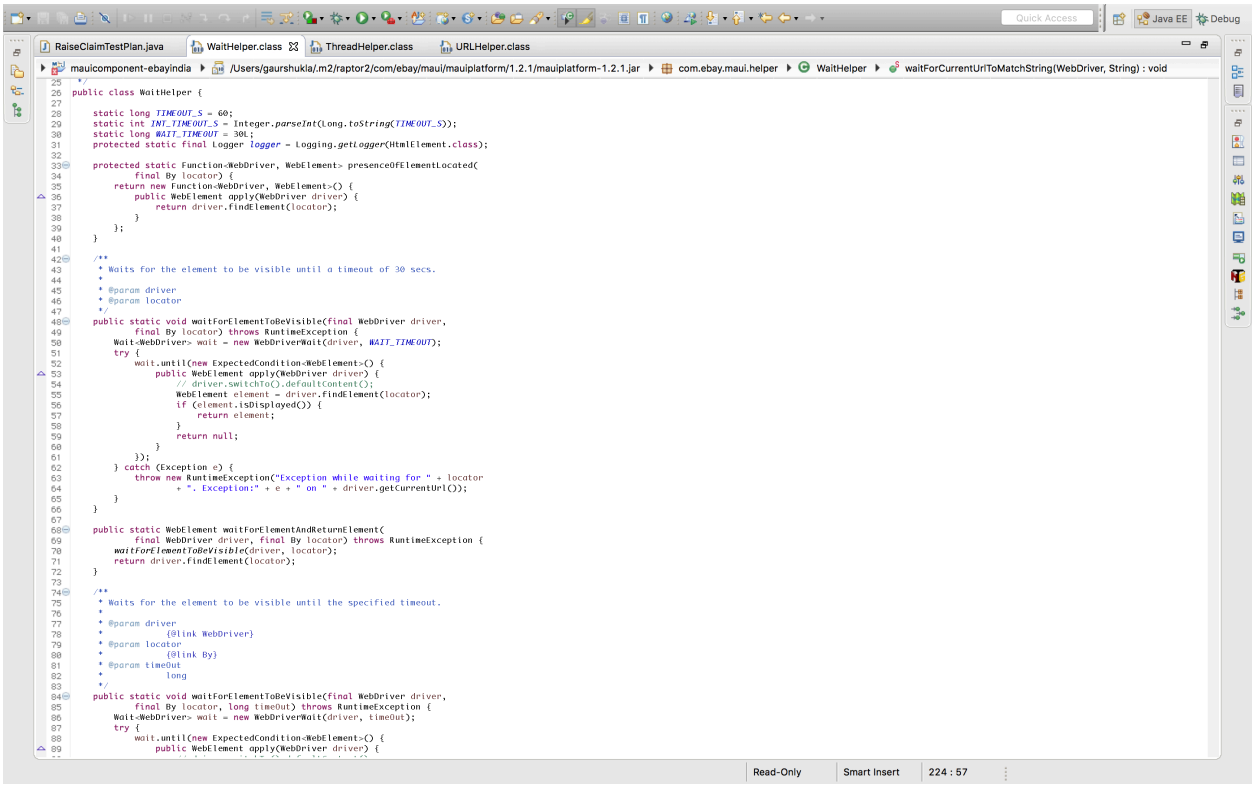

RaiseClaimTestPlan.java
WaitHelper.class
ThreadHelper.class
URLHelper.class
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
URLHelper
Read-Only
Smart Insert
1 : 1



```
        URLConnection connection = null;
        try {
            Context context = ContextManager.getThreadContext();
            if (useProxy && context.getWebProxyAddress()!= null) {
                String host = context.getWebProxyAddress().split(":")[0];
                int port = Integer.parseInt(context.getWebProxyAddress().split(":")[1]);
                Proxy proxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(host, port));
                connection = new URL(url).openConnection(proxy);
            }
            else {
                connection = new URL(url).openConnection();
            }
            connection.setConnectTimeout(timeout);
            connection.setReadTimeout(timeout);
            is = connection.getInputStream();
            return is;
        } catch (Throwable e) {
            // e.printStackTrace();
            throw new MauiException(e.getMessage(), e);
        }
    }

    @SuppressWarnings("deprecation")
    public static String openHttpsUrl(String url) throws HttpException,
            IOException {
        URL newUrl = new URL(url);
        Protocol myhttps = new Protocol("https",
                new EasySSLProtocolSocketFactory(), newUrl.getPort());
        Protocol.registerProtocol("https", myhttps);

        HttpClient httpclient = new HttpClient();
        String response = null;
        GetMethod httpget = new GetMethod(url);
        try {

            // httpclient.getHostConFiguration().setHost(newUrl.getHost(),
            // newUrl.getPort(), new Protocol("https", new
            // EasySSLProtocolSocketFactory(), newUrl.getPort()));
            httpclient.executeMethod(httpget);
            response = httpget.getResponseBodyAsString();
        } finally {
            httpget.releaseConnection();
        }
        return response;
    }

    @SuppressWarnings("deprecation")
    public static InputStream openHttpsUrlAsStream(String url,boolean useProxy)
            throws HttpException, IOException {
        URL newUrl = new URL(url);
        Protocol myhttps = new Protocol("https",
                new EasySSLProtocolSocketFactory(), newUrl.getDefaultPort());
        Protocol.registerProtocol("https", myhttps);
        HttpClient httpclient = new HttpClient();
        Context context = ContextManager.getThreadContext();
        if (useProxy && context.getWebProxyAddress()!= null) {
            String host = context.getWebProxyAddress().split(":")[0];
            int port = Integer.parseInt(context.getWebProxyAddress().split(":")[1]);
            httpclient.getHostConfiguration().setProxy(host, port);
        }
        InputStream response = null;
        GetMethod httpget = new GetMethod(url);

        httpclient.executeMethod(httpget);
        response = httpget.getResponseBodyAsStream();
```

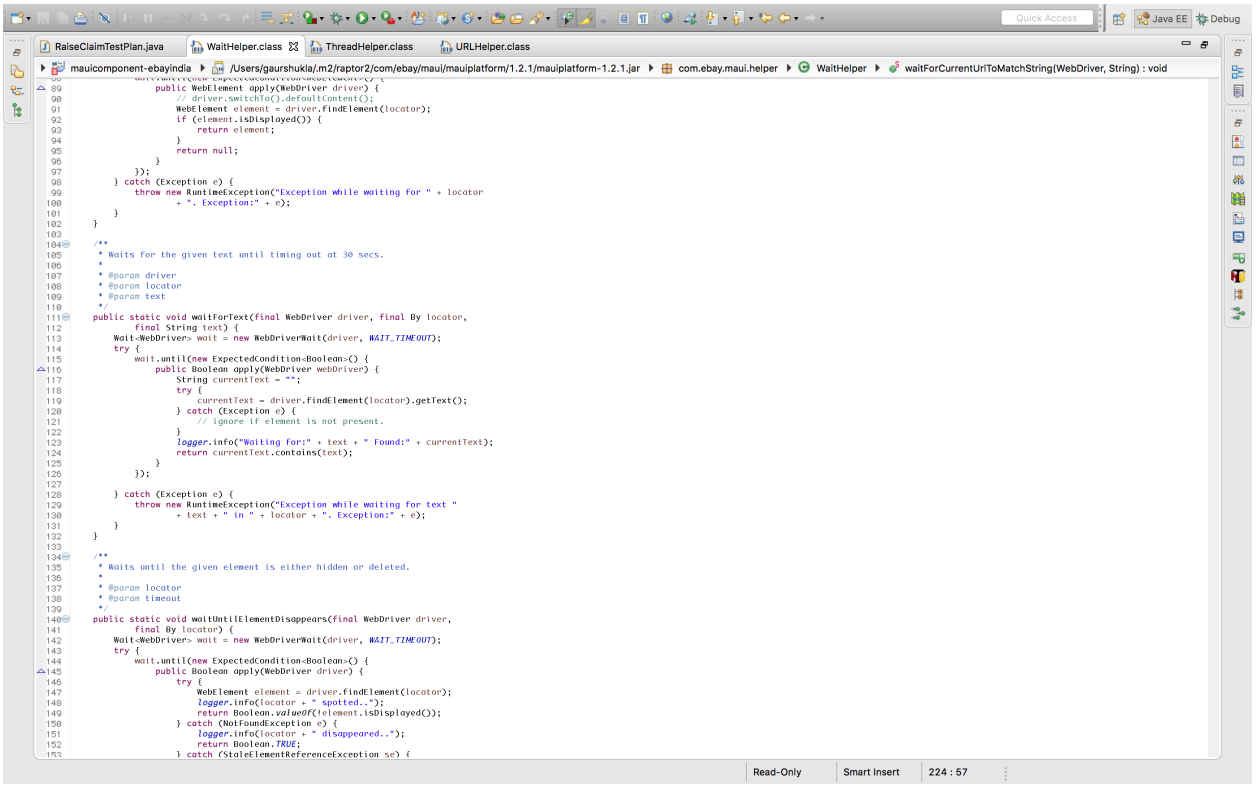


```
                Proxy proxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(host, port));
                connection = new URL(url).openConnection(proxy);
            }
            else {
                connection = new URL(url).openConnection();
            }
            connection.setConnectTimeout(timeout);
            connection.setReadTimeout(timeout);
            is = connection.getInputStream();
            return is;
        } catch (Throwable e) {
            // e.printStackTrace();
            throw new MauiException(e.getMessage(), e);
        }
    }

    @SuppressWarnings("deprecation")
    public static String openHttpsUrl(String url) throws HttpException,
            IOException {
        URL newUrl = new URL(url);
        Protocol myhttps = new Protocol("https",
                new EasySSLProtocolSocketFactory(), newUrl.getPort());
        Protocol.registerProtocol("https", myhttps);

        HttpClient httpclient = new HttpClient();
        String response = null;
        GetMethod httpget = new GetMethod(url);
        try {

            // httpclient.getHostConFiguration().setHost(newUrl.getHost(),
            // newUrl.getPort(), new Protocol("https", new
            // EasySSLProtocolSocketFactory(), newUrl.getPort()));
            httpclient.executeMethod(httpget);
            response = httpget.getResponseBodyAsString();
        } finally {
            httpget.releaseConnection();
        }
        return response;
    }

    @SuppressWarnings("deprecation")
    public static InputStream openHttpsUrlAsStream(String url,boolean useProxy)
            throws HttpException, IOException {
        URL newUrl = new URL(url);
        Protocol myhttps = new Protocol("https",
                new EasySSLProtocolSocketFactory(), newUrl.getDefaultPort());
        Protocol.registerProtocol("https", myhttps);
        HttpClient httpclient = new HttpClient();
        Context context = ContextManager.getThreadContext();
        if (useProxy && context.getWebProxyAddress()!= null) {
            String host = context.getWebProxyAddress().split(":")[0];
            int port = Integer.parseInt(context.getWebProxyAddress().split(":")[1]);
            httpclient.getHostConfiguration().setProxy(host, port);
        }
        InputStream response = null;
        GetMethod httpget = new GetMethod(url);

        httpclient.executeMethod(httpget);
        response = httpget.getResponseBodyAsStream();

        return response;
    }

}
```



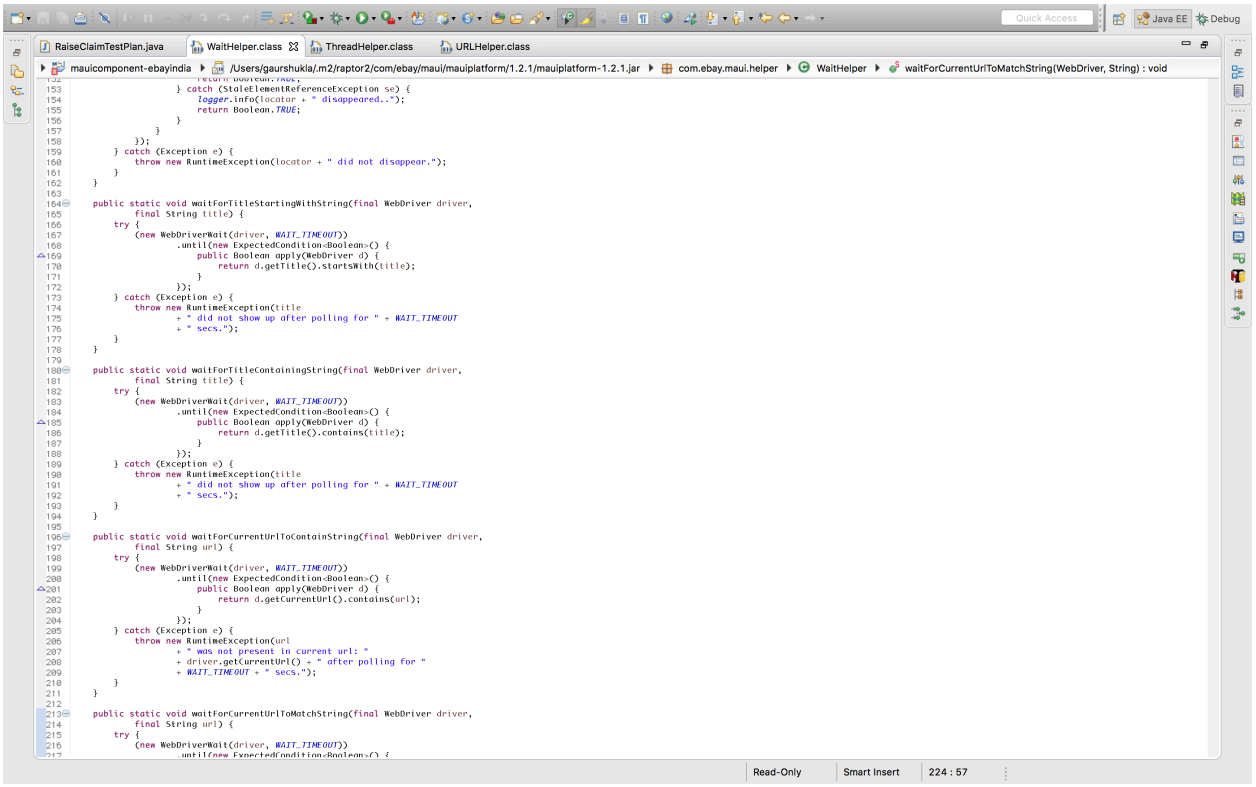
```
*/
public class WaitHelper {

    static long TIMEOUT_S = 60;
    static int INT_TIMEOUT_S = Integer.parseInt(Long.toString(TIMEOUT_S));
    static long WAIT_TIMEOUT = 30L;
    protected static final Logger logger = Logging.getLogger(HtmlElement.class);

    protected static Function<WebDriver, WebElement> presenceOfElementLocated(
            final By locator) {
        return new Function<WebDriver, WebElement>() {
            public WebElement apply(WebDriver driver) {
                return driver.findElement(locator);
            }
        };
    }

    /**
     * Waits for the element to be visible until a timeout of 30 secs.
     *
     * @param driver
     * @param locator
     */
    public static void waitForElementToBeVisible(final WebDriver driver,
            final By locator) throws RuntimeException {
        Wait<WebDriver> wait = new WebDriverWait(driver, WAIT_TIMEOUT);
        try {
            wait.until(new ExpectedCondition<WebElement>() {
                public WebElement apply(WebDriver driver) {
                    // driver.switchTo().defaultContent();
                    WebElement element = driver.findElement(locator);
                    if (element.isDisplayed()) {
                        return element;
                    }
                    return null;
                }
            });
        } catch (Exception e) {
            throw new RuntimeException("Exception while waiting for " + locator
                    + ". Exception:" + e + " on " + driver.getCurrentUrl());
        }
    }

    public static WebElement waitForElementAndReturnElement(
            final WebDriver driver, final By locator) throws RuntimeException {
        waitForElementToBeVisible(driver, locator);
        return driver.findElement(locator);
    }

    /**
     * Waits for the element to be visible until the specified timeout.
     *
     * @param driver
     *            {@link WebDriver}
     * @param locator
     *            {@link By}
     * @param timeOut
     *             long
     */
    public static void waitForElementToBeVisible(final WebDriver driver,
            final By locator, long timeOut) throws RuntimeException {
        Wait<WebDriver> wait = new WebDriverWait(driver, timeOut);
        try {
            wait.until(new ExpectedCondition<WebElement>() {
                public WebElement apply(WebDriver driver) {
```

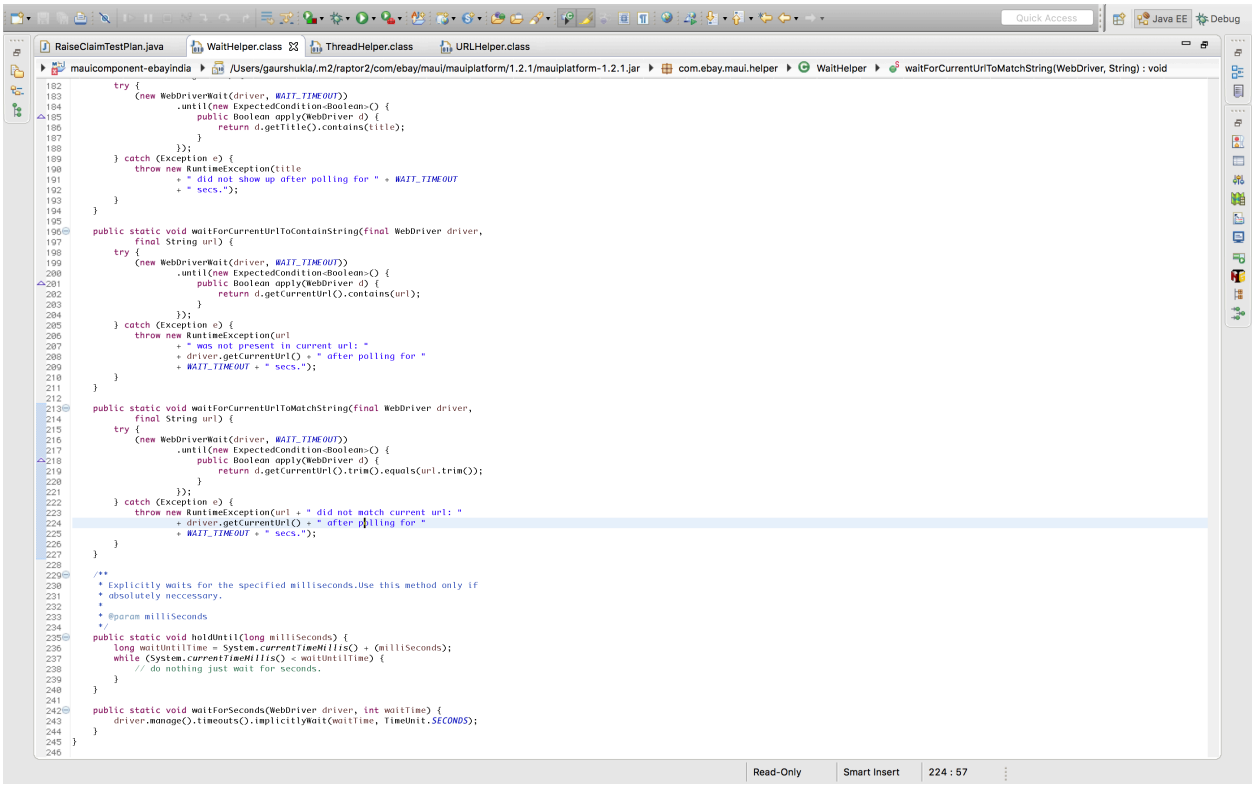
Read-Only | Smart Insert | 224 : 57

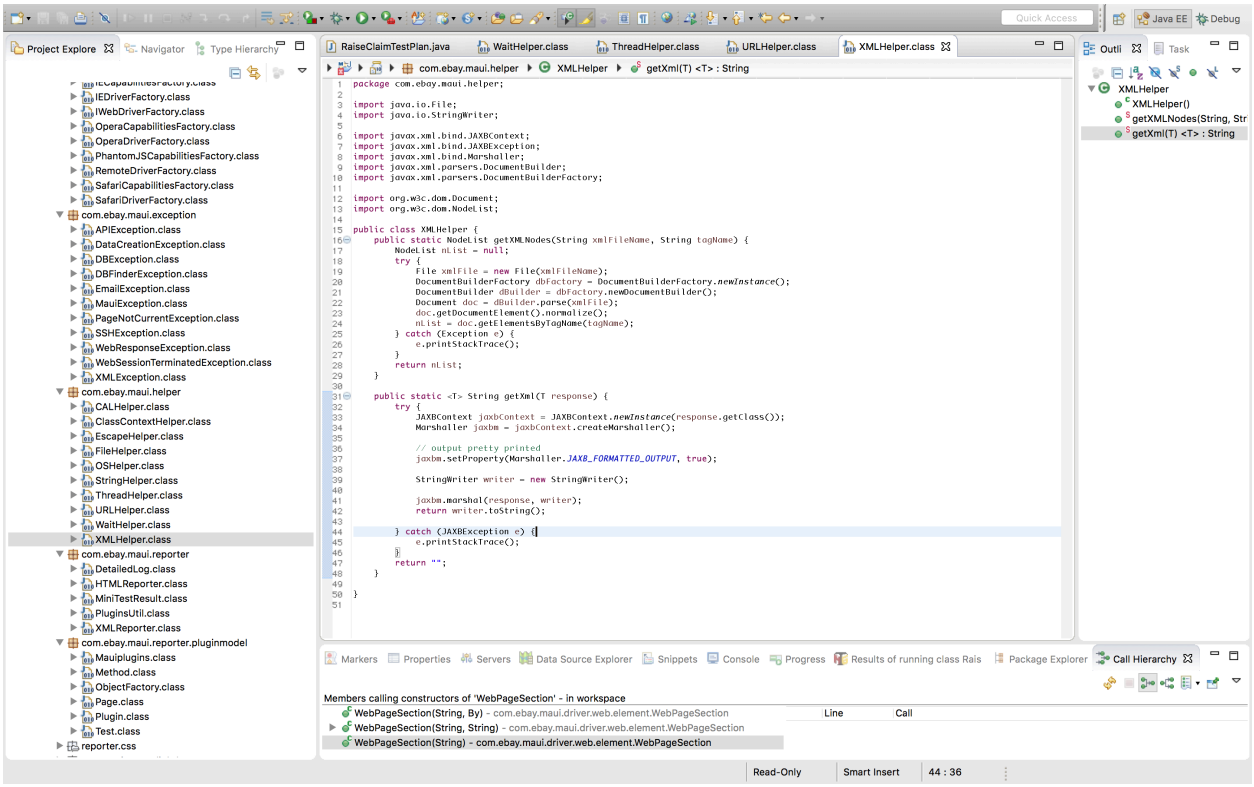


```
                public WebElement apply(WebDriver driver) {
                    // driver.switchTo().defaultContent();
                    WebElement element = driver.findElement(locator);
                    if (element.isDisplayed()) {
                        return element;
                    }
                    return null;
                }
            });
        } catch (Exception e) {
            throw new RuntimeException("Exception while waiting for " + locator
                    + ". Exception:" + e);
        }
    }

    /**
     * Waits for the given text until timing out at 30 secs.
     *
     * @param driver
     * @param locator
     * @param text
     */
    public static void waitForText(final WebDriver driver, final By locator,
            final String text) {
        Wait<WebDriver> wait = new WebDriverWait(driver, WAIT_TIMEOUT);
        try {
            wait.until(new ExpectedCondition<Boolean>() {
                public Boolean apply(WebDriver webDriver) {
                    String currentText = "";
                    try {
                        currentText = driver.findElement(locator).getText();
                    } catch (Exception e) {
                        // ignore if element is not present.
                    }
                    logger.info("Waiting for:" + text + " Found:" + currentText);
                    return currentText.contains(text);
                }
            });

        } catch (Exception e) {
            throw new RuntimeException("Exception while waiting for text "
                    + text + " in " + locator + ". Exception:" + e);
        }
    }

    /**
     * Waits until the given element is either hidden or deleted.
     *
     * @param locator
     * @param timeout
     */
    public static void waitUntilElementDisappears(final WebDriver driver,
            final By locator) {
        Wait<WebDriver> wait = new WebDriverWait(driver, WAIT_TIMEOUT);
        try {
            wait.until(new ExpectedCondition<Boolean>() {
                public Boolean apply(WebDriver driver) {
                    try {
                        WebElement element = driver.findElement(locator);
                        logger.info(locator + " spotted..");
                        return Boolean.valueOf(!element.isDisplayed());
                    } catch (NotFoundException e) {
                        logger.info(locator + " disappeared..");
                        return Boolean.TRUE;
                    } catch (StaleElementReferenceException se) {
```

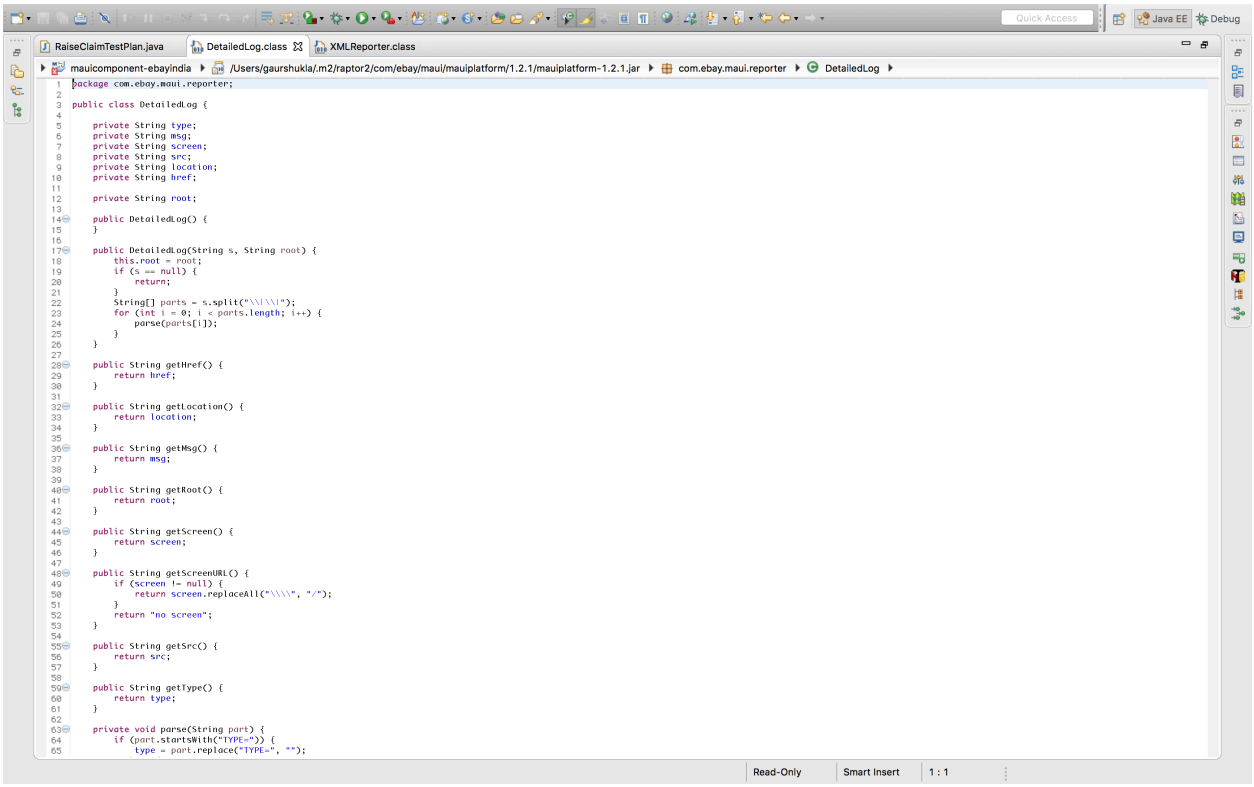
Read-Only | Smart Insert | 224 : 57


RaiseClaimTestPlan.java
WaitHelper.class
ThreadHelper.class
URLHelper.class
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
WaitHelper
waitForCurrentUrlToMatchString(WebDriver, String) : void
Quick Access
Java EE
Debug
Read-Only
Smart Insert
224 : 57

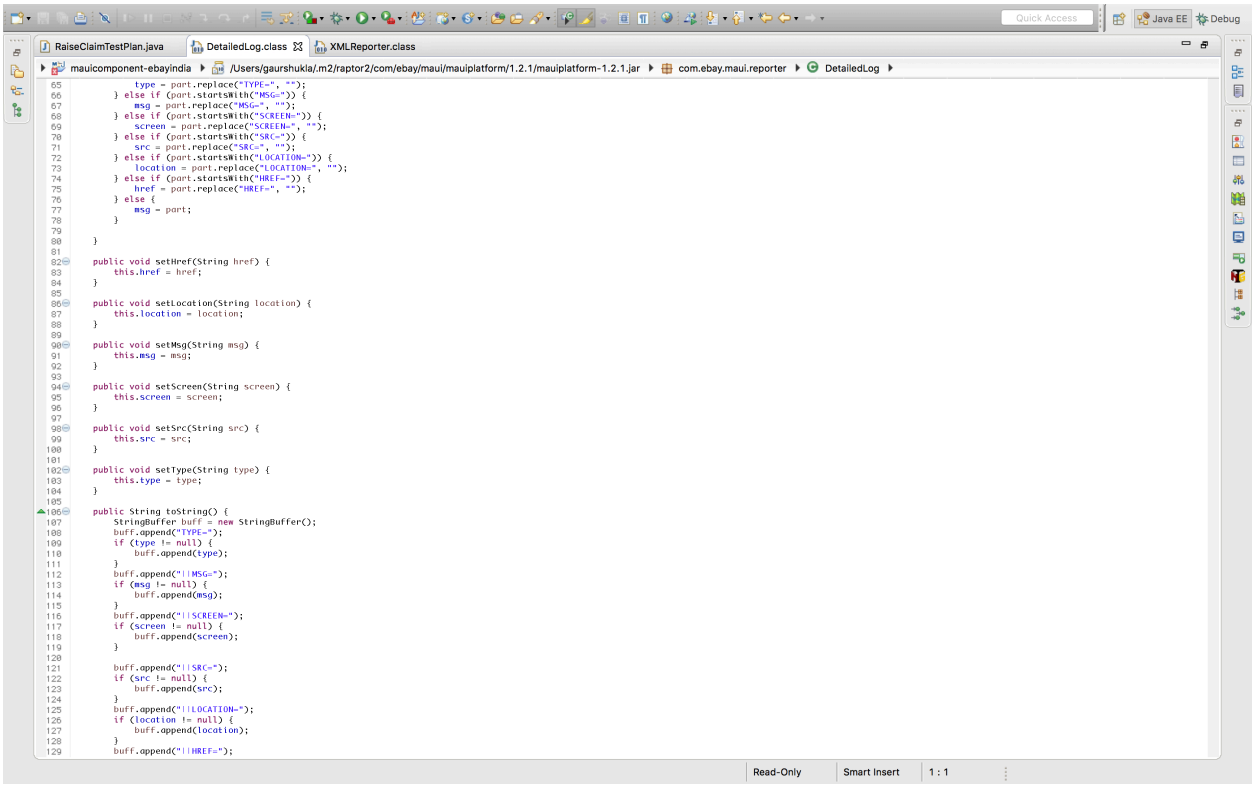

RaiseClaimTestPlan.java
WaitHelper.class
ThreadHelper.class
URLHelper.class
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.helper
WaitHelper
waitForCurrentUrlToMatchString(WebDriver, String) : void
Quick Access
Java EE
Debug
Read-Only
Smart Insert
224 : 57


Project Explore
Navigator
Type Hierarchy
IEDriverFactory.class
IWebDriverFactory.class
OperaCapabilitiesFactory.class
OperaDriverFactory.class
PhantomJSCapabilitiesFactory.class
RemoteDriverFactory.class
SafariCapabilitiesFactory.class
SafariDriverFactory.class
com.ebay.maui.exception
APIException.class
DataCreationException.class
DBException.class
DBFinderException.class
EmailException.class
MauiException.class
PageNotCurrentException.class
SSHException.class
WebResponseException.class
WebSessionTerminatedException.class
XMLException.class
com.ebay.maui.helper
CALHelper.class
ClassContextHelper.class
EscapeHelper.class
FileHelper.class
OSHelper.class
StringHelper.class
ThreadHelper.class
URLHelper.class
WaitHelper.class
XMLHelper.class
com.ebay.maui.reporter
DetailedLog.class
HTMLReporter.class
MiniTestResult.class
PluginsUtil.class
XMLReporter.class
com.ebay.maui.reporter.pluginmodel
Mauiplugins.class
Method.class
ObjectFactory.class
Page.class
Plugin.class
Test.class
reporter.css
RaiseClaimTestPlan.java
WaitHelper.class
ThreadHelper.class
URLHelper.class
XMLHelper.class
com.ebay.maui.helper
XMLHelper
getXml(T) <T> : String
package com.ebay.maui.helper;
import java.io.File;
import java.io.StringWriter;
import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Marshaller;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
public class XMLHelper {
public static NodeList getXMLNodes(String xmlFileName, String tagName) {
NodeList nList = null;
try {
File xmlFile = new File(xmlFileName);
DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
Document doc = dBuilder.parse(xmlFile);
doc.getDocumentElement().normalize();
nList = doc.getElementsByTagName(tagName);
} catch (Exception e) {
e.printStackTrace();
}
return nList;
}
public static <T> String getXml(T response) {
try {
JAXBContext jaxbContext = JAXBContext.newInstance(response.getClass());
Marshaller jaxbm = jaxbContext.createMarshaller();
// output pretty printed
jaxbm.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
StringWriter writer = new StringWriter();
jaxbm.marshal(response, writer);
return writer.toString();
} catch (JAXBException e) {
e.printStackTrace();
}
return "";
}
}
Outli
Task
XMLHelper
XMLHelper()
getXMLNodes(String, Str
getXml(T) <T> : String
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
Progress
Results of running class Rais
Package Explorer
Call Hierarchy
Members calling constructors of 'WebPageSection' - in workspace
WebPageSection(String, By) - com.ebay.maui.driver.web.element.WebPageSection
WebPageSection(String, String) - com.ebay.maui.driver.web.element.WebPageSection
WebPageSection(String) - com.ebay.maui.driver.web.element.WebPageSection
Line
Call
Read-Only
Smart Insert
44 : 36

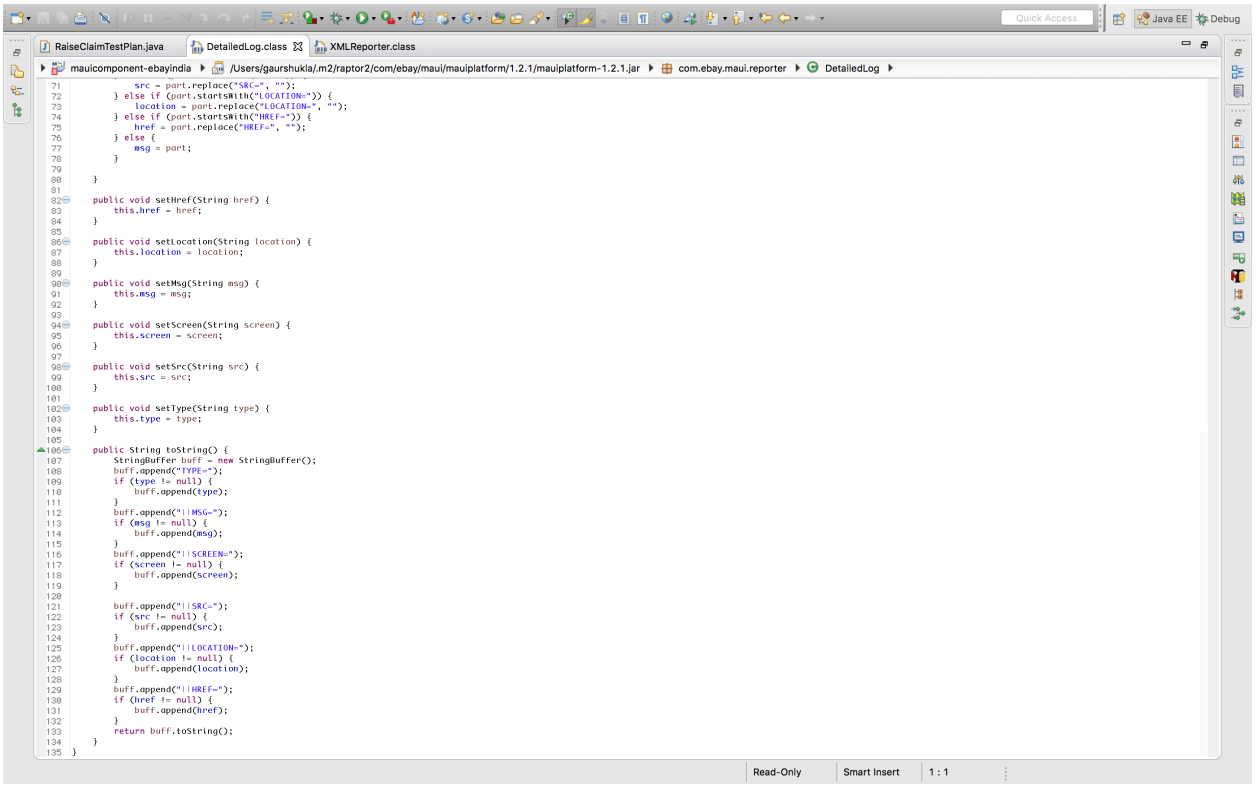
REPORTER:

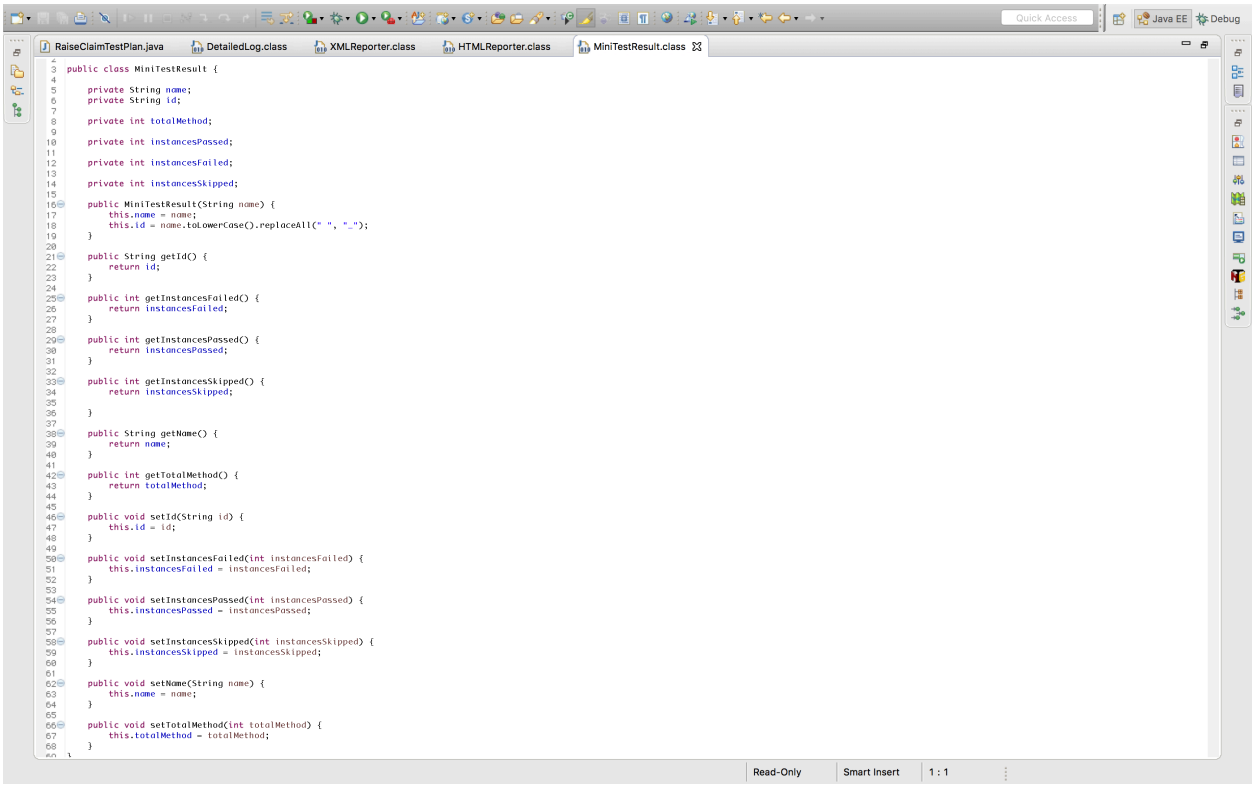


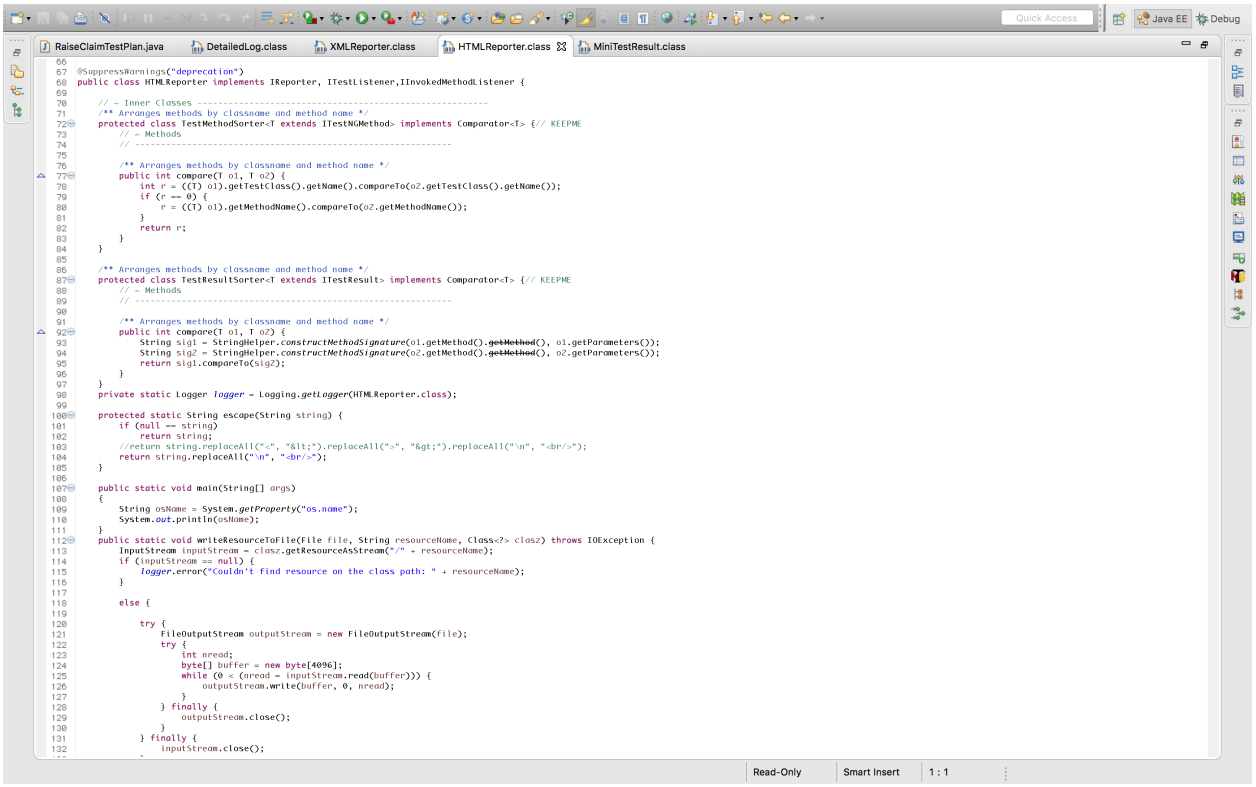
```
package com.ebay.maui.reporter;

public class DetailedLog {

    private String type;
    private String msg;
    private String screen;
    private String src;
    private String location;
    private String href;

    private String root;

    public DetailedLog() {
    }

    public DetailedLog(String s, String root) {
        this.root = root;
        if (s == null) {
            return;
        }
        String[] parts = s.split("\\|\\|");
        for (int i = 0; i < parts.length; i++) {
            parse(parts[i]);
        }
    }

    public String getHref() {
        return href;
    }

    public String getLocation() {
        return location;
    }

    public String getMsg() {
        return msg;
    }

    public String getRoot() {
        return root;
    }

    public String getScreen() {
        return screen;
    }

    public String getScreenURL() {
        if (screen != null) {
            return screen.replaceAll("\\\\", "/");
        }
        return "no screen";
    }

    public String getSrc() {
        return src;
    }

    public String getType() {
        return type;
    }

    private void parse(String part) {
        if (part.startsWith("TYPE=")) {
            type = part.replace("TYPE=", "");
```

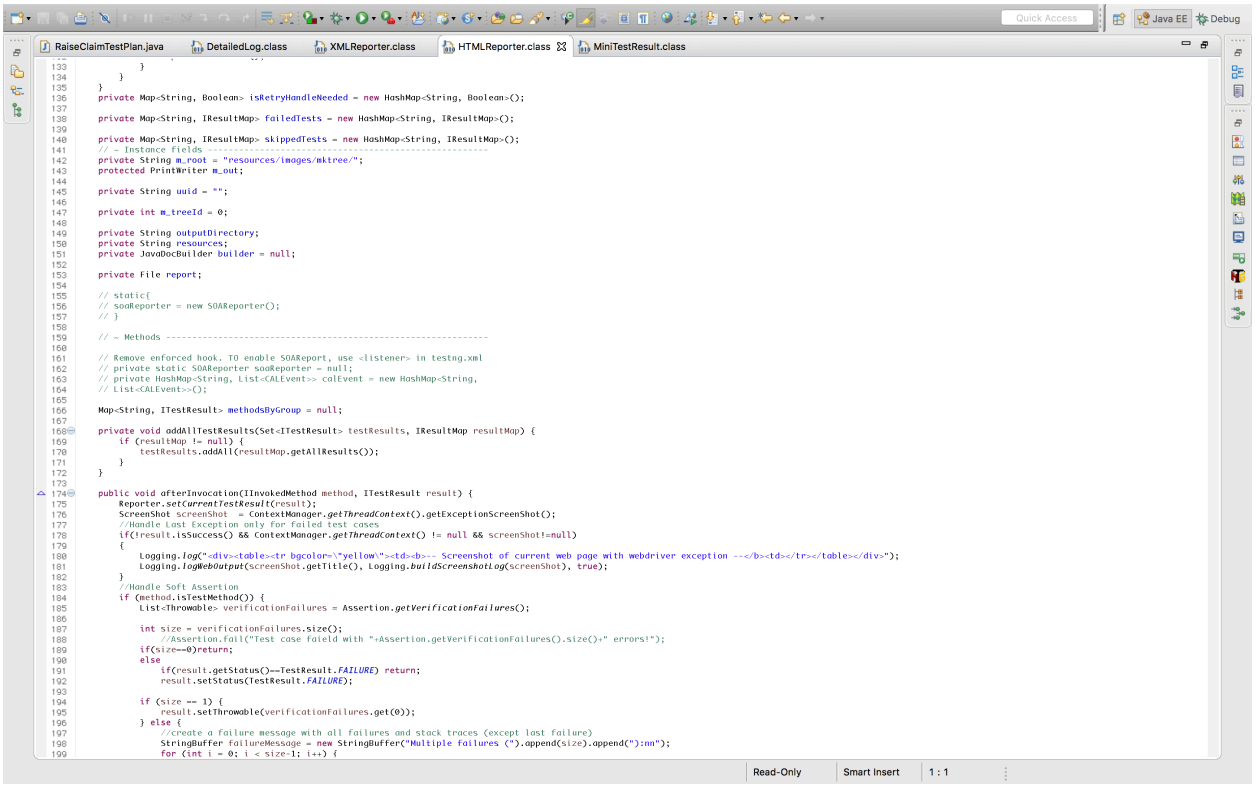


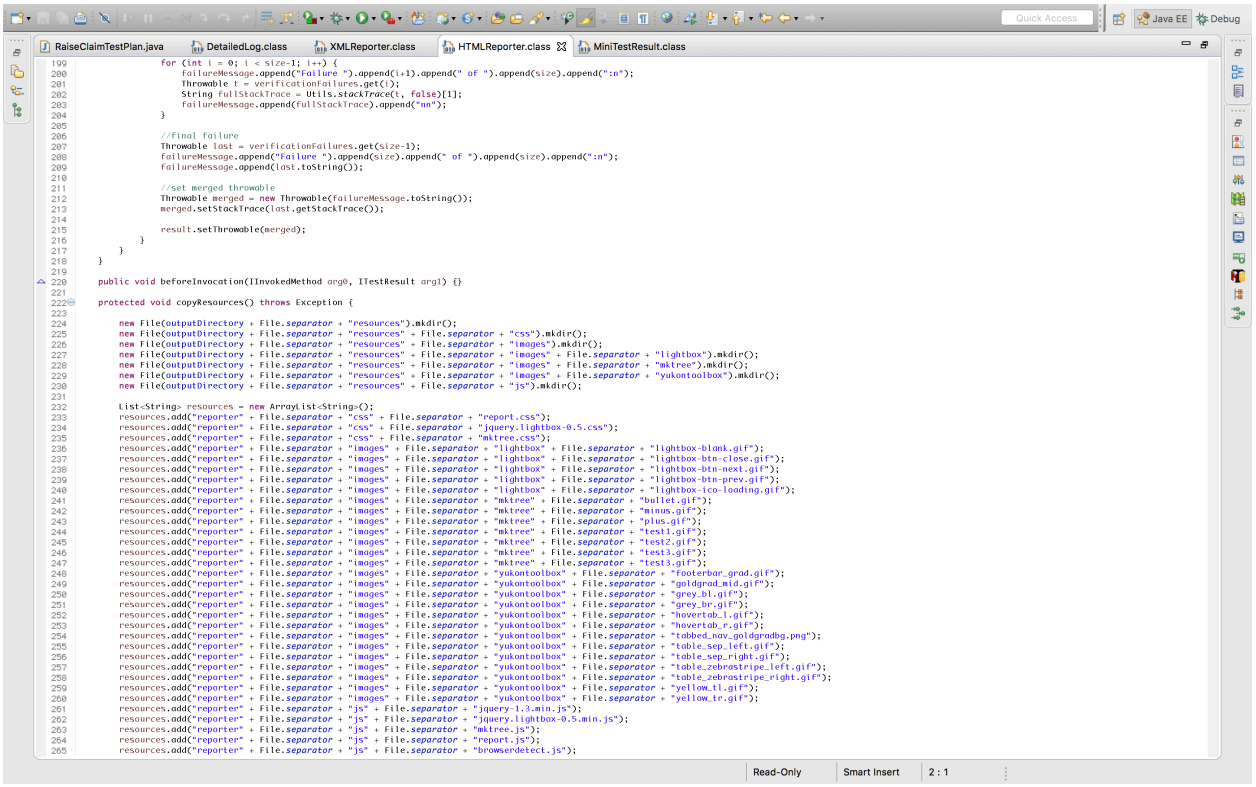
```
            type = part.replace("TYPE=", "");
        } else if (part.startsWith("MSG=")) {
            msg = part.replace("MSG=", "");
        } else if (part.startsWith("SCREEN=")) {
            screen = part.replace("SCREEN=", "");
        } else if (part.startsWith("SRC=")) {
            src = part.replace("SRC=", "");
        } else if (part.startsWith("LOCATION=")) {
            location = part.replace("LOCATION=", "");
        } else if (part.startsWith("HREF=")) {
            href = part.replace("HREF=", "");
        } else {
            msg = part;
        }

    }

    public void setHref(String href) {
        this.href = href;
    }

    public void setLocation(String location) {
        this.location = location;
    }

    public void setMsg(String msg) {
        this.msg = msg;
    }

    public void setScreen(String screen) {
        this.screen = screen;
    }

    public void setSrc(String src) {
        this.src = src;
    }

    public void setType(String type) {
        this.type = type;
    }

    public String toString() {
        StringBuffer buff = new StringBuffer();
        buff.append("TYPE=");
        if (type != null) {
            buff.append(type);
        }
        buff.append("||MSG=");
        if (msg != null) {
            buff.append(msg);
        }
        buff.append("||SCREEN=");
        if (screen != null) {
            buff.append(screen);
        }

        buff.append("||SRC=");
        if (src != null) {
            buff.append(src);
        }
        buff.append("||LOCATION=");
        if (location != null) {
            buff.append(location);
        }
        buff.append("||HREF=");
```


RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
mauicomponent-ebayindia
/Users/gaurshukla/.m2/raptor2/com/ebay/maui/mauiplatform/1.2.1/mauiplatform-1.2.1.jar
com.ebay.maui.reporter
DetailedLog
src = part.replace("SRC=", "");
} else if (part.startsWith("LOCATION=")) {
location = part.replace("LOCATION=", "");
} else if (part.startsWith("HREF=")) {
href = part.replace("HREF=", "");
} else {
msg = part;
}
}
public void setHref(String href) {
this.href = href;
}
public void setLocation(String location) {
this.location = location;
}
public void setMsg(String msg) {
this.msg = msg;
}
public void setScreen(String screen) {
this.screen = screen;
}
public void setSrc(String src) {
this.src = src;
}
public void setType(String type) {
this.type = type;
}
public String toString() {
StringBuffer buff = new StringBuffer();
buff.append("TYPE=");
if (type != null) {
buff.append(type);
}
buff.append("||MSG=");
if (msg != null) {
buff.append(msg);
}
buff.append("||SCREEN=");
if (screen != null) {
buff.append(screen);
}
buff.append("||SRC=");
if (src != null) {
buff.append(src);
}
buff.append("||LOCATION=");
if (location != null) {
buff.append(location);
}
buff.append("||HREF=");
if (href != null) {
buff.append(href);
}
return buff.toString();
}
}
Read-Only
Smart Insert
1 : 1

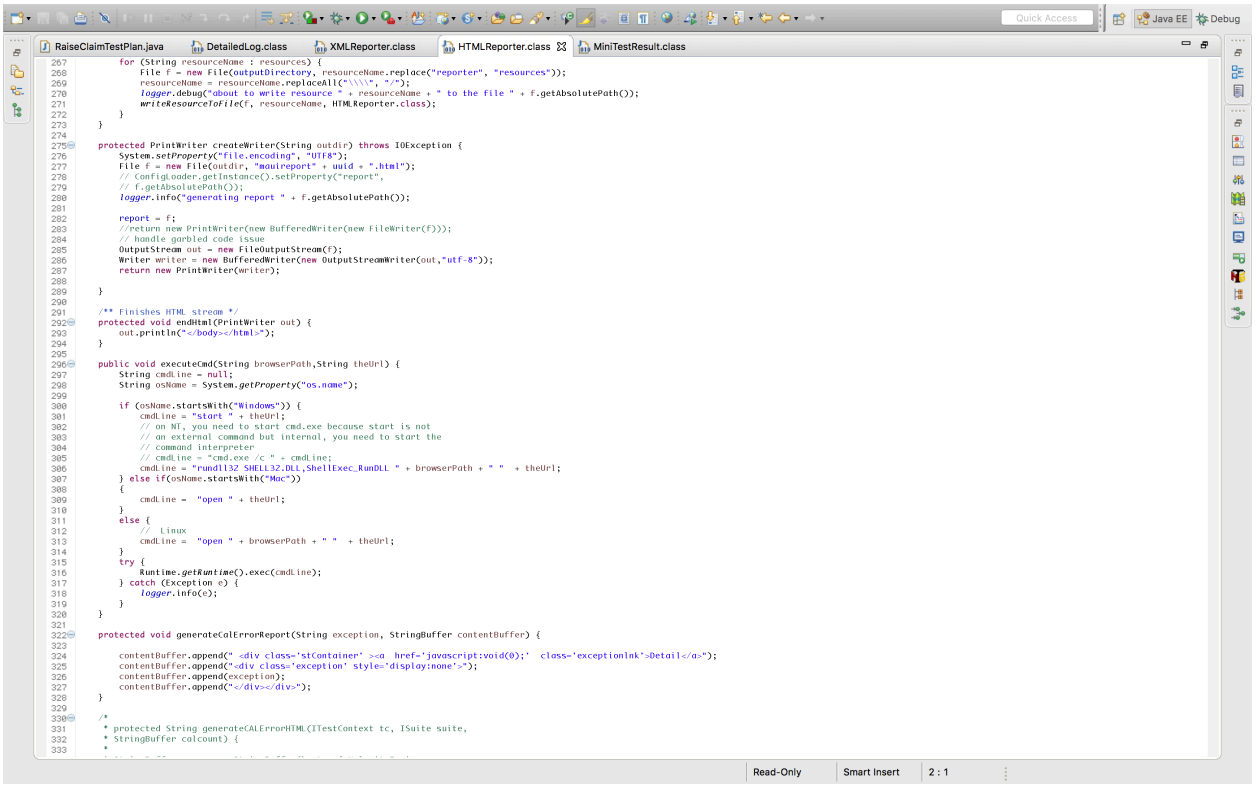

RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
public class MiniTestResult {
private String name;
private String id;
private int totalMethod;
private int instancesPassed;
private int instancesFailed;
private int instancesSkipped;
public MiniTestResult(String name) {
this.name = name;
this.id = name.toLowerCase().replaceAll(" ", "_");
}
public String getId() {
return id;
}
public int getInstancesFailed() {
return instancesFailed;
}
public int getInstancesPassed() {
return instancesPassed;
}
public int getInstancesSkipped() {
return instancesSkipped;
}
public String getName() {
return name;
}
public int getTotalMethod() {
return totalMethod;
}
public void setId(String id) {
this.id = id;
}
public void setInstancesFailed(int instancesFailed) {
this.instancesFailed = instancesFailed;
}
public void setInstancesPassed(int instancesPassed) {
this.instancesPassed = instancesPassed;
}
public void setInstancesSkipped(int instancesSkipped) {
this.instancesSkipped = instancesSkipped;
}
public void setName(String name) {
this.name = name;
}
public void setTotalMethod(int totalMethod) {
this.totalMethod = totalMethod;
}
Read-Only
Smart Insert
1 : 1


Quick Access
Java EE
Debug
RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Read-Only
Smart Insert
1 : 1

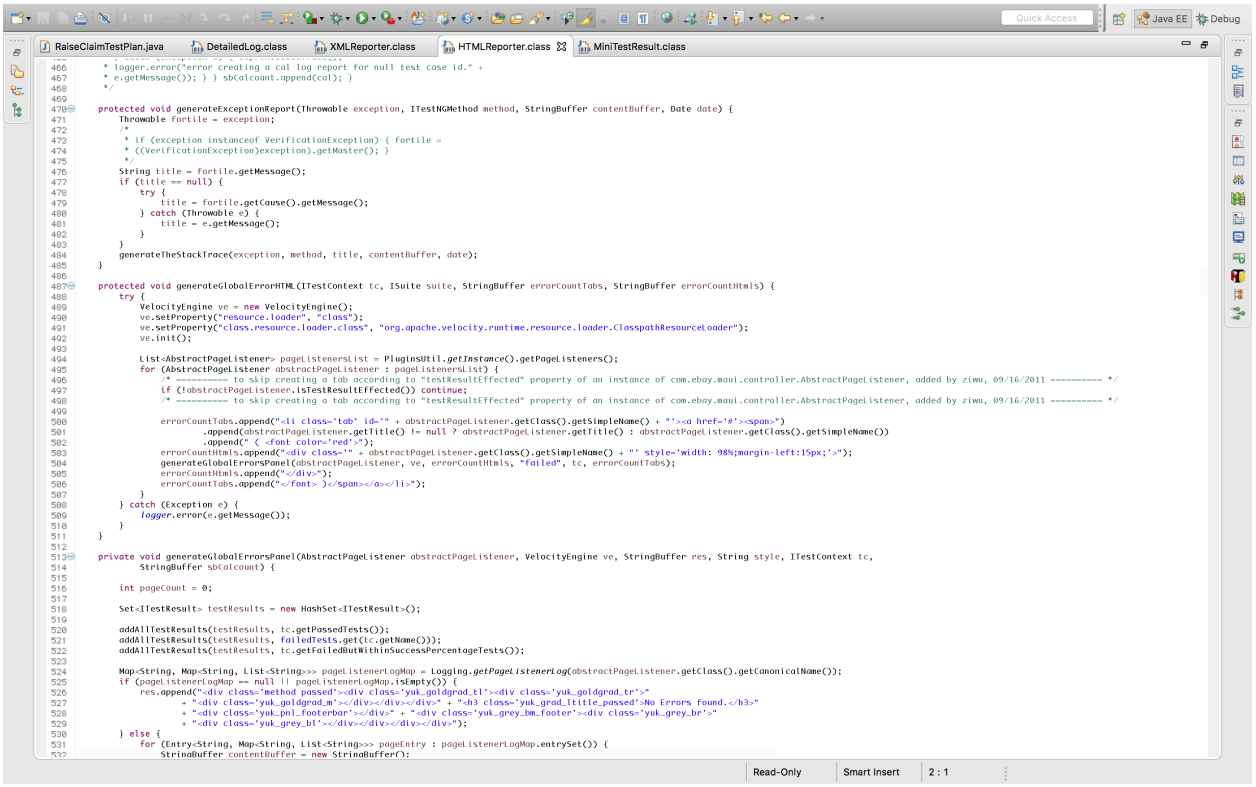

Quick Access
Java EE
Debug
RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Read-Only
Smart Insert
1 : 1



```
                for (int i = 0; i < size-1; i++) {
                    failureMessage.append("Failure ").append(i+1).append(" of ").append(size).append(":n");
                    Throwable t = verificationFailures.get(i);
                    String fullStackTrace = Utils.stackTrace(t, false)[1];
                    failureMessage.append(fullStackTrace).append("nn");
                }

                //final failure
                Throwable last = verificationFailures.get(size-1);
                failureMessage.append("Failure ").append(size).append(" of ").append(size).append(":n");
                failureMessage.append(last.toString());

                //set merged throwable
                Throwable merged = new Throwable(failureMessage.toString());
                merged.setStackTrace(last.getStackTrace());

                result.setThrowable(merged);
            }
        }
    }

    public void beforeInvocation(IInvokedMethod arg0, ITestResult arg1) {}

    protected void copyResources() throws Exception {

        new File(outputDirectory + File.separator + "resources").mkdir();
        new File(outputDirectory + File.separator + "resources" + File.separator + "css").mkdir();
        new File(outputDirectory + File.separator + "resources" + File.separator + "images").mkdir();
        new File(outputDirectory + File.separator + "resources" + File.separator + "images" + File.separator + "lightbox").mkdir();
        new File(outputDirectory + File.separator + "resources" + File.separator + "images" + File.separator + "mktree").mkdir();
        new File(outputDirectory + File.separator + "resources" + File.separator + "images" + File.separator + "yukontoolbox").mkdir();
        new File(outputDirectory + File.separator + "resources" + File.separator + "js").mkdir();

        List<String> resources = new ArrayList<String>();
        resources.add("reporter" + File.separator + "css" + File.separator + "report.css");
        resources.add("reporter" + File.separator + "css" + File.separator + "jquery.lightbox-0.5.css");
        resources.add("reporter" + File.separator + "css" + File.separator + "mktree.css");
        resources.add("reporter" + File.separator + "images" + File.separator + "lightbox" + File.separator + "lightbox-blank.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "lightbox" + File.separator + "lightbox-btn-close.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "lightbox" + File.separator + "lightbox-btn-next.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "lightbox" + File.separator + "lightbox-btn-prev.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "lightbox" + File.separator + "lightbox-ico-loading.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "mktree" + File.separator + "bullet.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "mktree" + File.separator + "minus.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "mktree" + File.separator + "plus.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "mktree" + File.separator + "test1.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "mktree" + File.separator + "test2.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "mktree" + File.separator + "test3.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "mktree" + File.separator + "test3.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "footerbar_grad.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "goldgrad_mid.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "grey_bl.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "grey_br.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "hovertab_l.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "hovertab_r.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "tabbed_nav_goldgradbg.png");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "table_sep_left.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "table_sep_right.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "table_zebrastripe_left.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "table_zebrastripe_right.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "yellow_tl.gif");
        resources.add("reporter" + File.separator + "images" + File.separator + "yukontoolbox" + File.separator + "yellow_tr.gif");
        resources.add("reporter" + File.separator + "js" + File.separator + "jquery-1.3.min.js");
        resources.add("reporter" + File.separator + "js" + File.separator + "jquery.lightbox-0.5.min.js");
        resources.add("reporter" + File.separator + "js" + File.separator + "mktree.js");
        resources.add("reporter" + File.separator + "js" + File.separator + "report.js");
        resources.add("reporter" + File.separator + "js" + File.separator + "browserdetect.js");
```

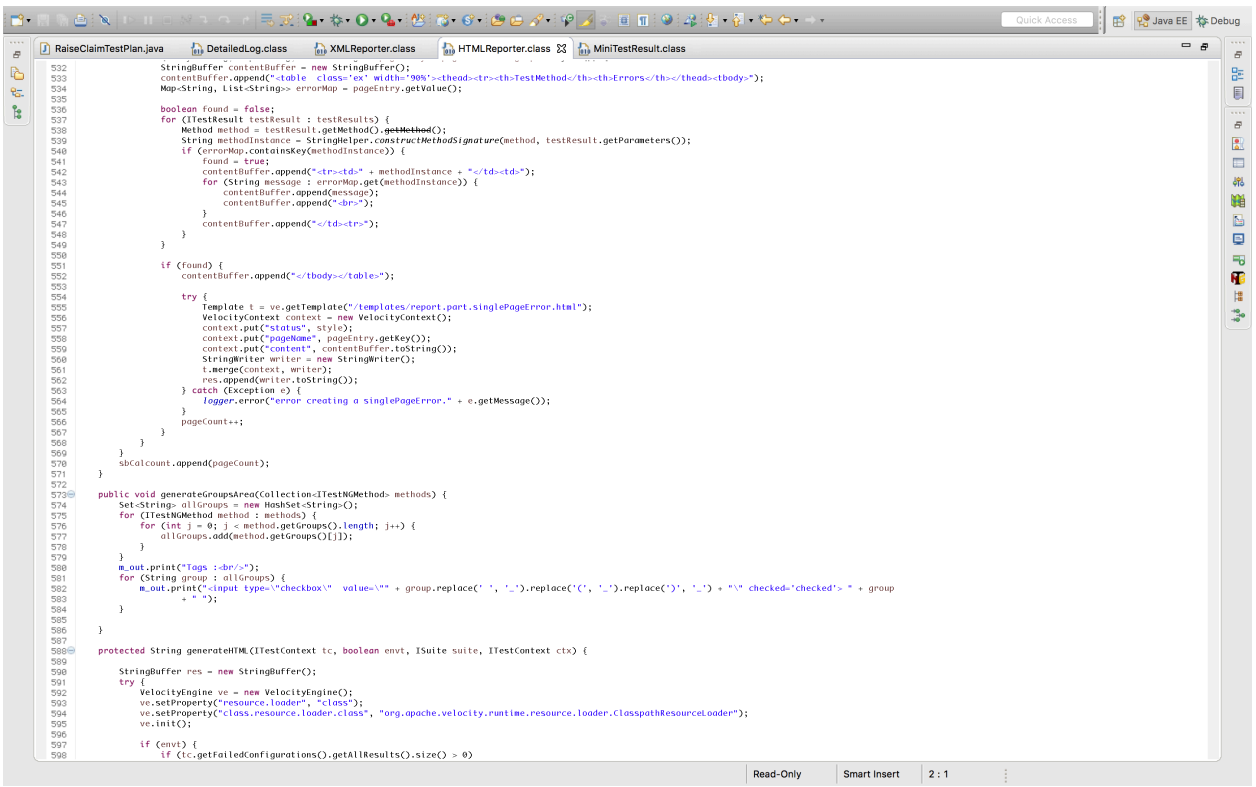


```
        for (String resourceName : resources) {
            File f = new File(outputDirectory, resourceName.replace("reporter", "resources"));
            resourceName = resourceName.replaceAll("\\\\", "/");
            logger.debug("about to write resource " + resourceName + " to the file " + f.getAbsolutePath());
            writeResourceToFile(f, resourceName, HTMLReporter.class);
        }
    }

    protected PrintWriter createWriter(String outdir) throws IOException {
        System.setProperty("file.encoding", "UTF8");
        File f = new File(outdir, "maureport" + uuid + ".html");
        // ConfigLoader.getInstance().setProperty("report",
        // f.getAbsolutePath());
        logger.info("generating report " + f.getAbsolutePath());

        report = f;
        //return new PrintWriter(new BufferedWriter(new FileWriter(f)));
        // handle garbled code issue
        OutputStream out = new FileOutputStream(f);
        Writer writer = new BufferedWriter(new OutputStreamWriter(out,"utf-8"));
        return new PrintWriter(writer);

    }

    /** Finishes HTML stream */
    protected void endHtml(PrintWriter out) {
        out.println("</body></html>");
    }

    public void executeCmd(String browserPath,String theUrl) {
        String cmdLine = null;
        String osName = System.getProperty("os.name");

        if (osName.startsWith("Windows")) {
            cmdLine = "start " + theUrl;
            // on NT, you need to start cmd.exe because start is not
            // an external command but internal, you need to start the
            // command interpreter
            // cmdLine = "cmd.exe /c " + cmdLine;
            cmdLine = "rundll32 SHELL32.DLL,ShellExec_RunDLL " + browserPath + " "  + theUrl;
        } else if(osName.startsWith("Mac"))
        {
            cmdLine =  "open " + theUrl;
        }
        else {
            // Linux
            cmdLine =  "open " + browserPath + " "  + theUrl;
        }
        try {
            Runtime.getRuntime().exec(cmdLine);
        } catch (Exception e) {
            logger.info(e);
        }
    }

    protected void generateCalErrorReport(String exception, StringBuffer contentBuffer) {

        contentBuffer.append(" <div class='stContainer' ><a  href='javascript:void(0);'  class='exceptionlnk'>Detail</a>");
        contentBuffer.append("<div class='exception' style='display:none'>");
        contentBuffer.append(exception);
        contentBuffer.append("</div></div>");
    }

    /*
     * protected String generateCALErrorHTML(ITestContext tc, ISuite suite,
     * StringBuffer calcount) {
     *
```



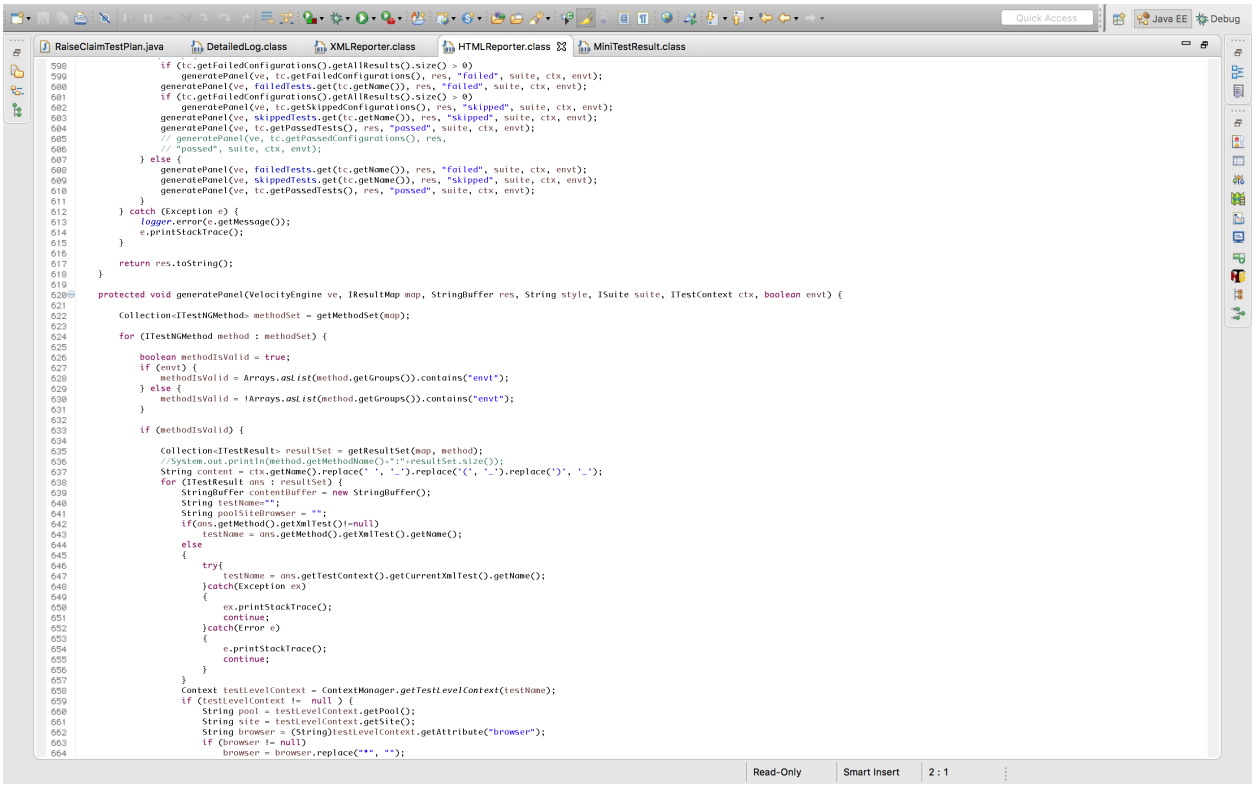
```
 * logger.error("error creating a cal log report for null test case id." +
 * e.getMessage()); } } sbCalcount.append(col); }
 */

protected void generateExceptionReport(Throwable exception, ITestNGMethod method, StringBuffer contentBuffer, Date date) {
    Throwable fortile = exception;
    /*
     * if (exception instanceof VerificationException) { fortile =
     * ((VerificationException)exception).getMaster(); }
     */
    String title = fortile.getMessage();
    if (title == null) {
        try {
            title = fortile.getCause().getMessage();
        } catch (Throwable e) {
            title = e.getMessage();
        }
    }
    generateTheStackTrace(exception, method, title, contentBuffer, date);
}

protected void generateGlobalErrorHTML(ITestContext tc, ISuite suite, StringBuffer errorCountTabs, StringBuffer errorCountHtmls) {
    try {
        VelocityEngine ve = new VelocityEngine();
        ve.setProperty("resource.loader", "class");
        ve.setProperty("class.resource.loader.class", "org.apache.velocity.runtime.resource.loader.ClasspathResourceLoader");
        ve.init();

        List<AbstractPageListener> pageListenersList = PluginsUtil.getInstance().getPageListeners();
        for (AbstractPageListener abstractPageListener : pageListenersList) {
            /* ---------- to skip creating a tab according to "testResultEffected" property of an instance of com.ebay.maui.controller.AbstractPageListener, added by ziwu, 09/16/2011 ---------- */
            if (!abstractPageListener.isTestResultEffected()) continue;
            /* ---------- to skip creating a tab according to "testResultEffected" property of an instance of com.ebay.maui.controller.AbstractPageListener, added by ziwu, 09/16/2011 ---------- */

            errorCountTabs.append("<li class='tab' id='" + abstractPageListener.getClass().getSimpleName() + "'><a href='#'><span>")
                    .append(abstractPageListener.getTitle() != null ? abstractPageListener.getTitle() : abstractPageListener.getClass().getSimpleName())
                    .append(" ( <font color='red'>");
            errorCountHtmls.append("<div class='" + abstractPageListener.getClass().getSimpleName() + "' style='width: 98%;margin-left:15px;'>");
            generateGlobalErrorsPanel(abstractPageListener, ve, errorCountHtmls, "failed", tc, errorCountTabs);
            errorCountHtmls.append("</div>");
            errorCountTabs.append("</font> )</span></a></li>");
        }
    } catch (Exception e) {
        logger.error(e.getMessage());
    }
}

private void generateGlobalErrorsPanel(AbstractPageListener abstractPageListener, VelocityEngine ve, StringBuffer res, String style, ITestContext tc,
        StringBuffer sbCalcount) {

    int pageCount = 0;

    Set<ITestResult> testResults = new HashSet<ITestResult>();

    addAllTestResults(testResults, tc.getPassedTests());
    addAllTestResults(testResults, failedTests.get(tc.getName()));
    addAllTestResults(testResults, tc.getFailedButWithinSuccessPercentageTests());

    Map<String, Map<String, List<String>>> pageListenerLogMap = Logging.getPageListenerLog(abstractPageListener.getClass().getCanonicalName());
    if (pageListenerLogMap == null || pageListenerLogMap.isEmpty()) {
        res.append("<div class='method passed'><div class='yuk_goldgrad_tl'><div class='yuk_goldgrad_tr'>"
                + "<div class='yuk_goldgrad_m'></div></div></div>" + "<h3 class='yuk_grad_ltitle_passed'>No Errors found.</h3>"
                + "<div class='yuk_pnl_footerbar'></div>" + "<div class='yuk_grey_bm_footer'><div class='yuk_grey_br'>"
                + "<div class='yuk_grey_bl'></div></div></div></div>");
    } else {
        for (Entry<String, Map<String, List<String>>> pageEntry : pageListenerLogMap.entrySet()) {
            StringBuffer contentBuffer = new StringBuffer();
```

Read-Only | Smart Insert | 2 : 1

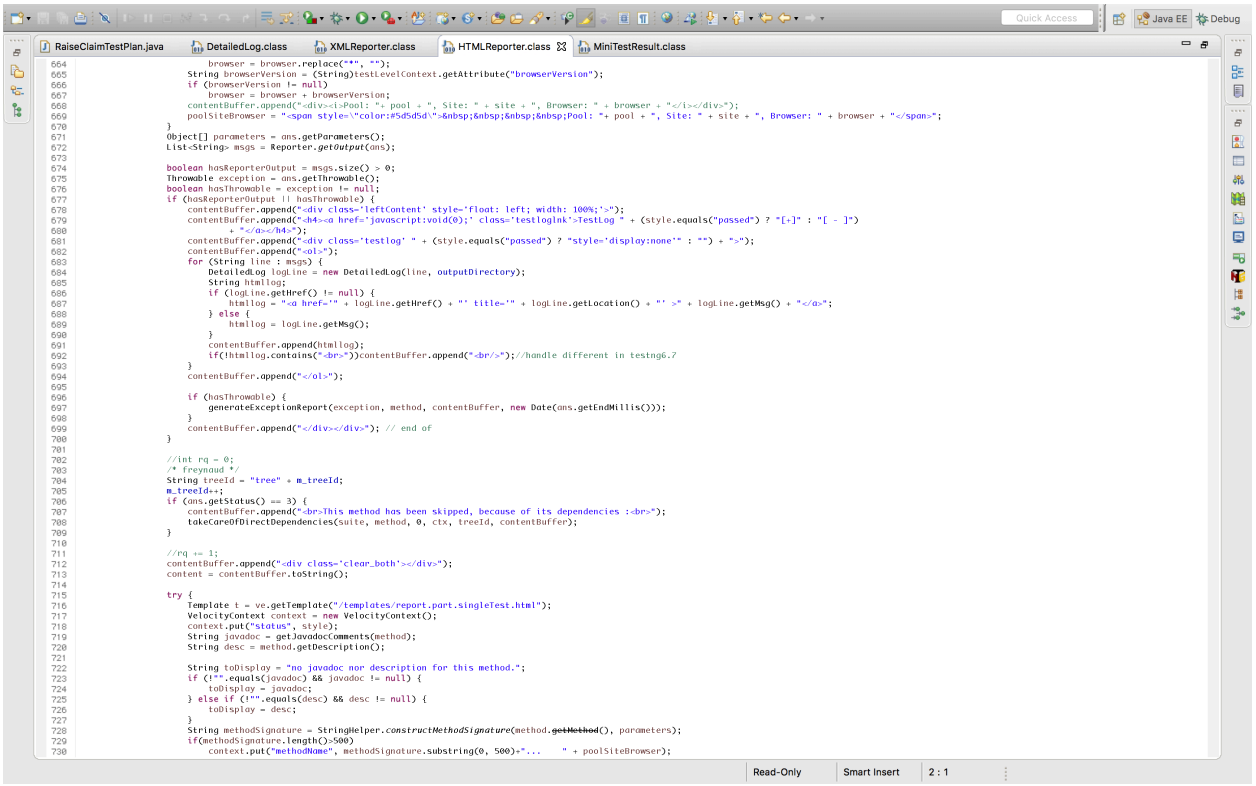


```
            StringBuffer contentBuffer = new StringBuffer();
            contentBuffer.append("<table  class='ex' width='90%'><thead><tr><th>TestMethod</th><th>Errors</th></thead><tbody>");
            Map<String, List<String>> errorMap = pageEntry.getValue();

            boolean found = false;
            for (ITestResult testResult : testResults) {
                Method method = testResult.getMethod().getMethod();
                String methodInstance = StringHelper.constructMethodSignature(method, testResult.getParameters());
                if (errorMap.containsKey(methodInstance)) {
                    found = true;
                    contentBuffer.append("<tr><td>" + methodInstance + "</td><td>");
                    for (String message : errorMap.get(methodInstance)) {
                        contentBuffer.append(message);
                        contentBuffer.append("<br>");
                    }
                    contentBuffer.append("</td><tr>");
                }
            }

            if (found) {
                contentBuffer.append("</tbody></table>");

                try {
                    Template t = ve.getTemplate("/templates/report.part.singlePageError.html");
                    VelocityContext context = new VelocityContext();
                    context.put("status", style);
                    context.put("pageName", pageEntry.getKey());
                    context.put("content", contentBuffer.toString());
                    StringWriter writer = new StringWriter();
                    t.merge(context, writer);
                    res.append(writer.toString());
                } catch (Exception e) {
                    logger.error("error creating a singlePageError." + e.getMessage());
                }
                pageCount++;
            }
        }
    }
    sbCalcount.append(pageCount);
}

public void generateGroupsArea(Collection<ITestNGMethod> methods) {
    Set<String> allGroups = new HashSet<String>();
    for (ITestNGMethod method : methods) {
        for (int j = 0; j < method.getGroups().length; j++) {
            allGroups.add(method.getGroups()[j]);
        }
    }
    m_out.print("Tags :<br/>");
    for (String group : allGroups) {
        m_out.print("<input type=\"checkbox\"  value=\"" + group.replace(' ', '_').replace('(', '_').replace(')', '_') + "\" checked='checked'> " + group
                + " ");
    }

}

protected String generateHTML(ITestContext tc, boolean envt, ISuite suite, ITestContext ctx) {

    StringBuffer res = new StringBuffer();
    try {
        VelocityEngine ve = new VelocityEngine();
        ve.setProperty("resource.loader", "class");
        ve.setProperty("class.resource.loader.class", "org.apache.velocity.runtime.resource.loader.ClasspathResourceLoader");
        ve.init();

        if (envt) {
            if (tc.getFailedConfigurations().getAllResults().size() > 0)
```

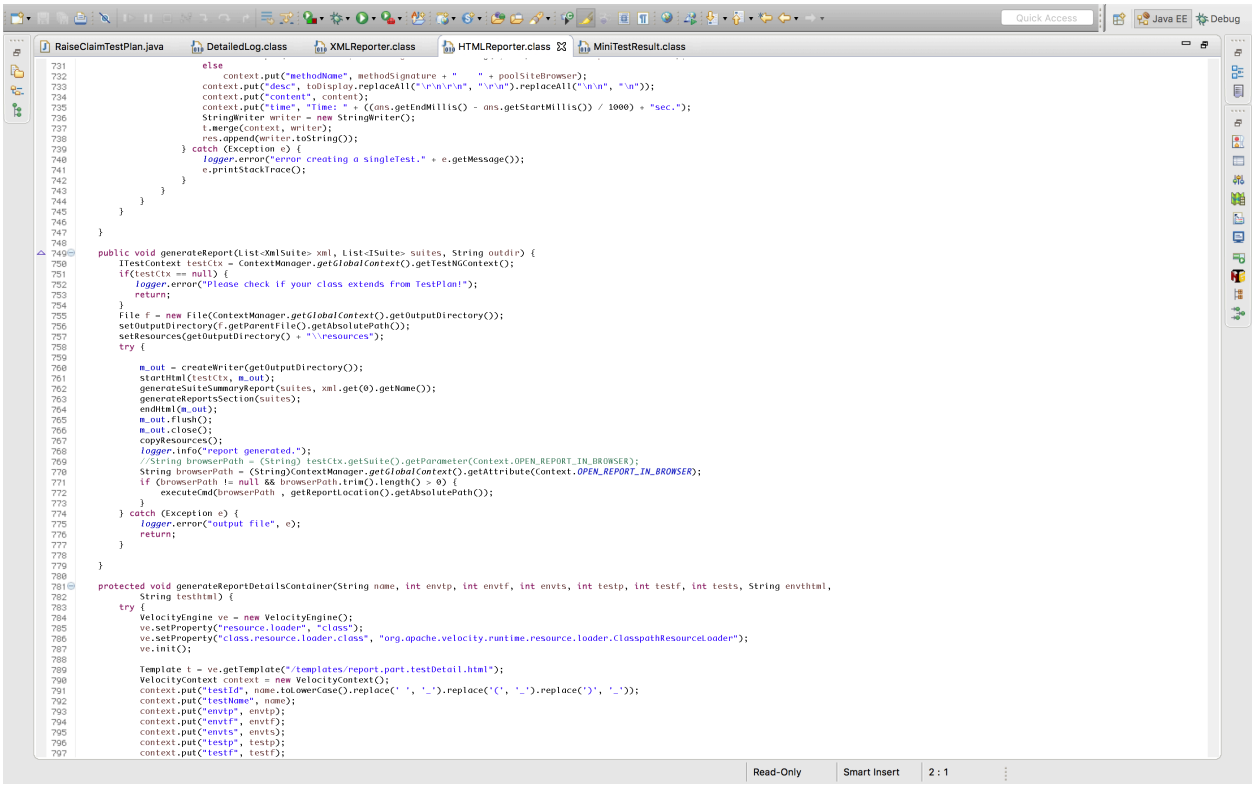
Read-Only | Smart Insert | 2 : 1

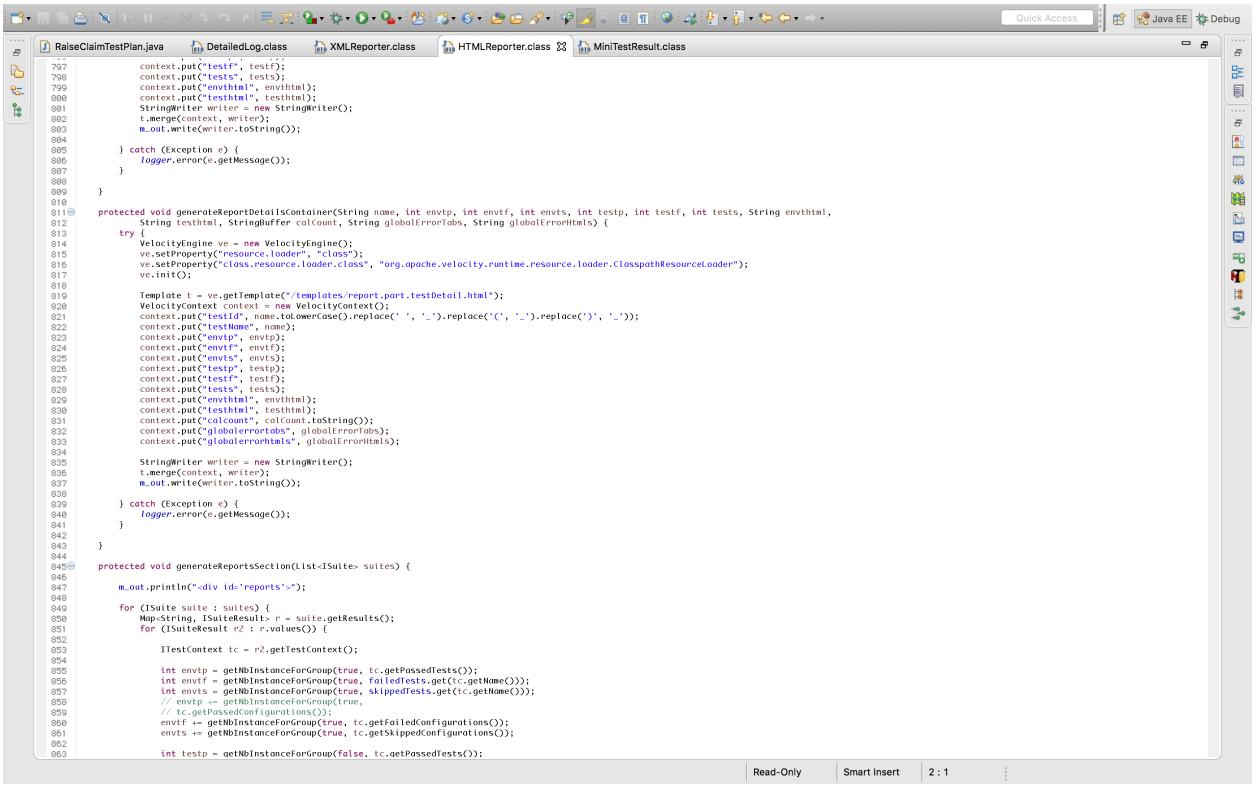


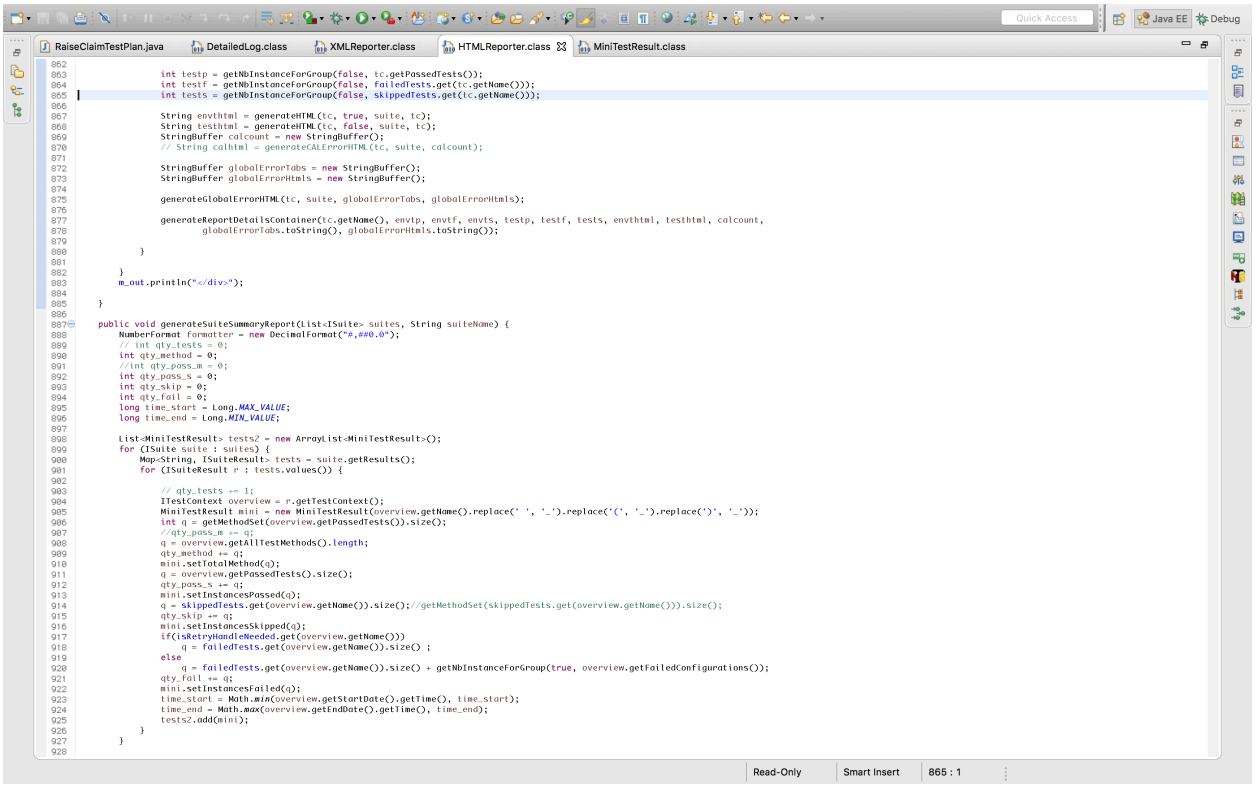
```
                if (tc.getFailedConfigurations().getAllResults().size() > 0)
                    generatePanel(ve, tc.getFailedConfigurations(), res, "failed", suite, ctx, envt);
                generatePanel(ve, failedTests.get(tc.getName()), res, "failed", suite, ctx, envt);
                if (tc.getFailedConfigurations().getAllResults().size() > 0)
                    generatePanel(ve, tc.getSkippedConfigurations(), res, "skipped", suite, ctx, envt);
                generatePanel(ve, skippedTests.get(tc.getName()), res, "skipped", suite, ctx, envt);
                generatePanel(ve, tc.getPassedTests(), res, "passed", suite, ctx, envt);
                // generatePanel(ve, tc.getPassedConfigurations(), res,
                // "passed", suite, ctx, envt);
            } else {
                generatePanel(ve, failedTests.get(tc.getName()), res, "failed", suite, ctx, envt);
                generatePanel(ve, skippedTests.get(tc.getName()), res, "skipped", suite, ctx, envt);
                generatePanel(ve, tc.getPassedTests(), res, "passed", suite, ctx, envt);
            }
        } catch (Exception e) {
            logger.error(e.getMessage());
            e.printStackTrace();
        }

        return res.toString();
    }

    protected void generatePanel(VelocityEngine ve, IResultMap map, StringBuffer res, String style, ISuite suite, ITestContext ctx, boolean envt) {

        Collection<ITestNGMethod> methodSet = getMethodSet(map);

        for (ITestNGMethod method : methodSet) {

            boolean methodIsValid = true;
            if (envt) {
                methodIsValid = Arrays.asList(method.getGroups()).contains("envt");
            } else {
                methodIsValid = !Arrays.asList(method.getGroups()).contains("envt");
            }

            if (methodIsValid) {

                Collection<ITestResult> resultSet = getResultSet(map, method);
                //System.out.println(method.getMethodName()+":"+resultSet.size());
                String content = ctx.getName().replace(' ', '_').replace('(', '_').replace(')', '_');
                for (ITestResult ans : resultSet) {
                    StringBuffer contentBuffer = new StringBuffer();
                    String testName="";
                    String poolSiteBrowser = "";
                    if(ans.getMethod().getXmlTest()!=null)
                        testName = ans.getMethod().getXmlTest().getName();
                    else
                    {
                        try{
                            testName = ans.getTestContext().getCurrentXmlTest().getName();
                        }catch(Exception ex)
                        {
                            ex.printStackTrace();
                            continue;
                        }catch(Error e)
                        {
                            e.printStackTrace();
                            continue;
                        }
                    }
                    ContextManager.getTestLevelContext(testName);
                    if (testLevelContext !=  null ) {
                        String pool = testLevelContext.getPool();
                        String site = testLevelContext.getSite();
                        String browser = (String)testLevelContext.getAttribute("browser");
                        if (browser != null)
                            browser = browser.replace("*", "");
```

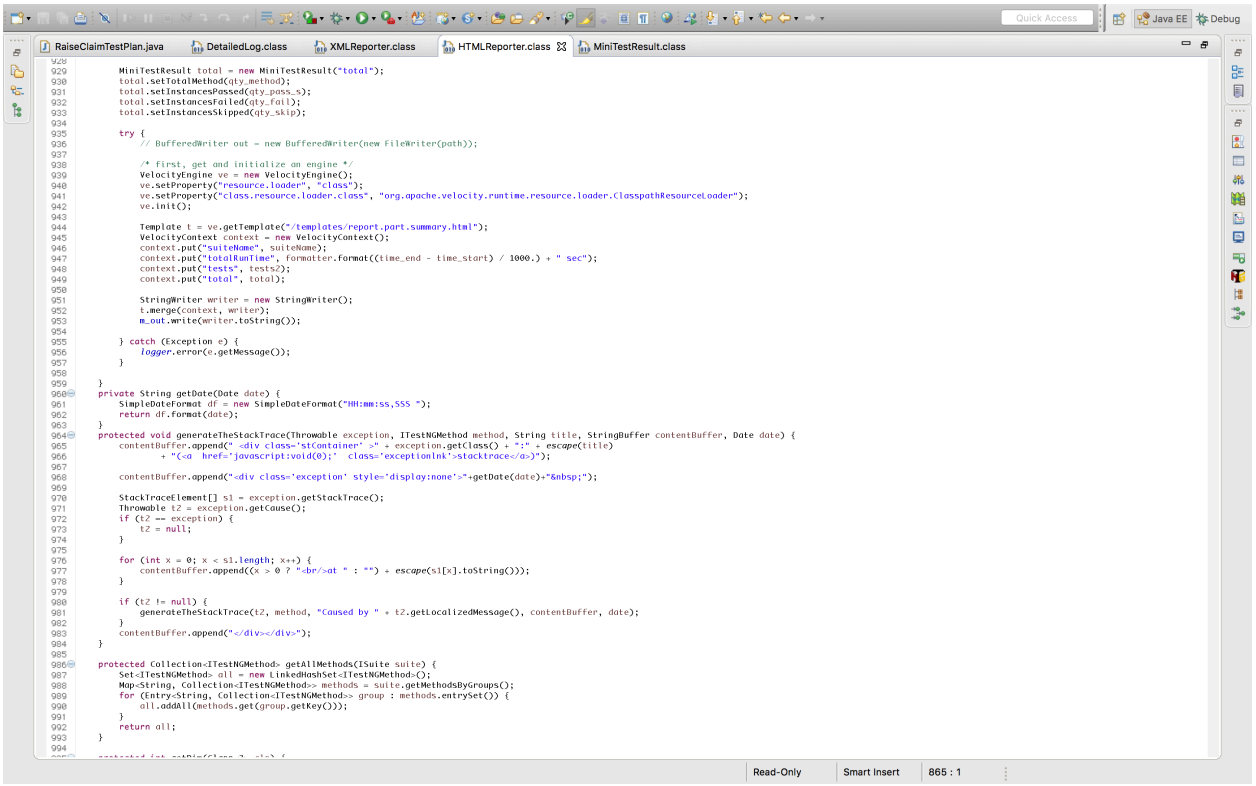


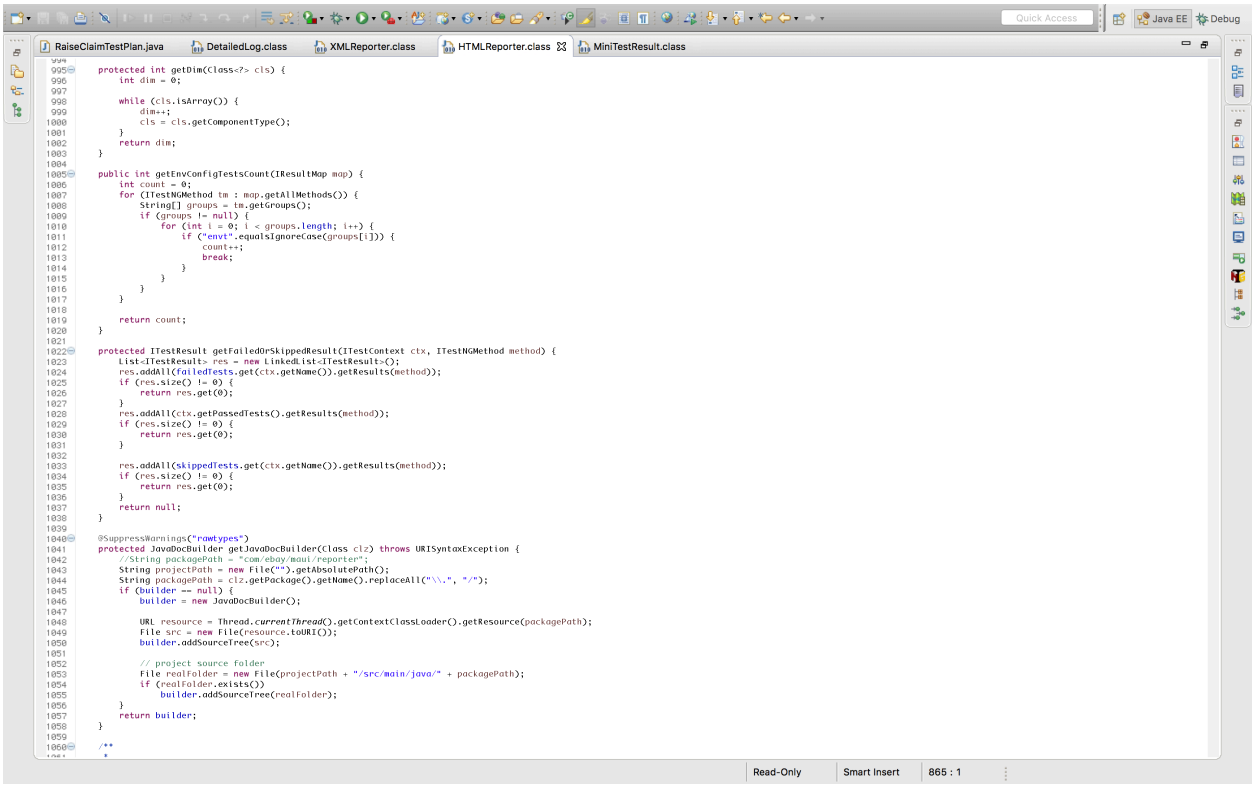
```
                            browser = browser.replace("*", "");
                        String browserVersion = (String)testLevelContext.getAttribute("browserVersion");
                        if (browserVersion != null)
                            browser = browser + browserVersion;
                        contentBuffer.append("<div><i>Pool: "+ pool + ", Site: " + site + ", Browser: " + browser + "</i></div>");
                        poolSiteBrowser = "<span style=\"color:#5d5d5d\">    Pool: "+ pool + ", Site: " + site + ", Browser: " + browser + "</span>";
                    }
                    Object[] parameters = ans.getParameters();
                    List<String> msgs = Reporter.getOutput(ans);

                    boolean hasReporterOutput = msgs.size() > 0;
                    Throwable exception = ans.getThrowable();
                    boolean hasThrowable = exception != null;
                    if (hasReporterOutput || hasThrowable) {
                        contentBuffer.append("<div class='leftContent' style='float: left; width: 100%;'>");
                        contentBuffer.append("<h4><a href='javascript:void(0);' class='testloglnk'>TestLog " + (style.equals("passed") ? "[+]" : "[ - ]")
                                + "</a></h4>");
                        contentBuffer.append("<div class='testlog' " + (style.equals("passed") ? "style='display:none'" : "") + ">");
                        contentBuffer.append("<ol>");
                        for (String line : msgs) {
                            DetailedLog logLine = new DetailedLog(line, outputDirectory);
                            String htmllog;
                            if (logLine.getHref() != null) {
                                htmllog = "<a href='" + logLine.getHref() + "' title='" + logLine.getLocation() + "' >" + logLine.getMsg() + "</a>";
                            } else {
                                htmllog = logLine.getMsg();
                            }
                            contentBuffer.append(htmllog);
                            if(!htmllog.contains("<br>"))contentBuffer.append("<br/>");//handle different in testng6.7
                        }
                        contentBuffer.append("</ol>");

                        if (hasThrowable) {
                            generateExceptionReport(exception, method, contentBuffer, new Date(ans.getEndMillis()));
                        }
                        contentBuffer.append("</div></div>"); // end of
                    }

                    //int rq = 0;
                    /* freynaud */
                    String treeId = "tree" + m_treeId;
                    m_treeId++;
                    if (ans.getStatus() == 3) {
                        contentBuffer.append("<br>This method has been skipped, because of its dependencies :<br>");
                        takeCareOfDirectDependencies(suite, method, 0, ctx, treeId, contentBuffer);
                    }

                    //rq += 1;
                    contentBuffer.append("<div class='clear_both'></div>");
                    content = contentBuffer.toString();

                    try {
                        Template t = ve.getTemplate("/templates/report.part.singleTest.html");
                        VelocityContext context = new VelocityContext();
                        context.put("status", style);
                        String javadoc = getJavadocComments(method);
                        String desc = method.getDescription();

                        String toDisplay = "no javadoc nor description for this method.";
                        if (!"".equals(javadoc) && javadoc != null) {
                            toDisplay = javadoc;
                        } else if (!"".equals(desc) && desc != null) {
                            toDisplay = desc;
                        }
                        String methodSignature = StringHelper.constructMethodSignature(method.getMethod(), parameters);
                        if(methodSignature.length()>500)
                            context.put("methodName", methodSignature.substring(0, 500)+"...    " + poolSiteBrowser);
```


Quick Access
Java EE
Debug
RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Read-Only
Smart Insert
2 : 1

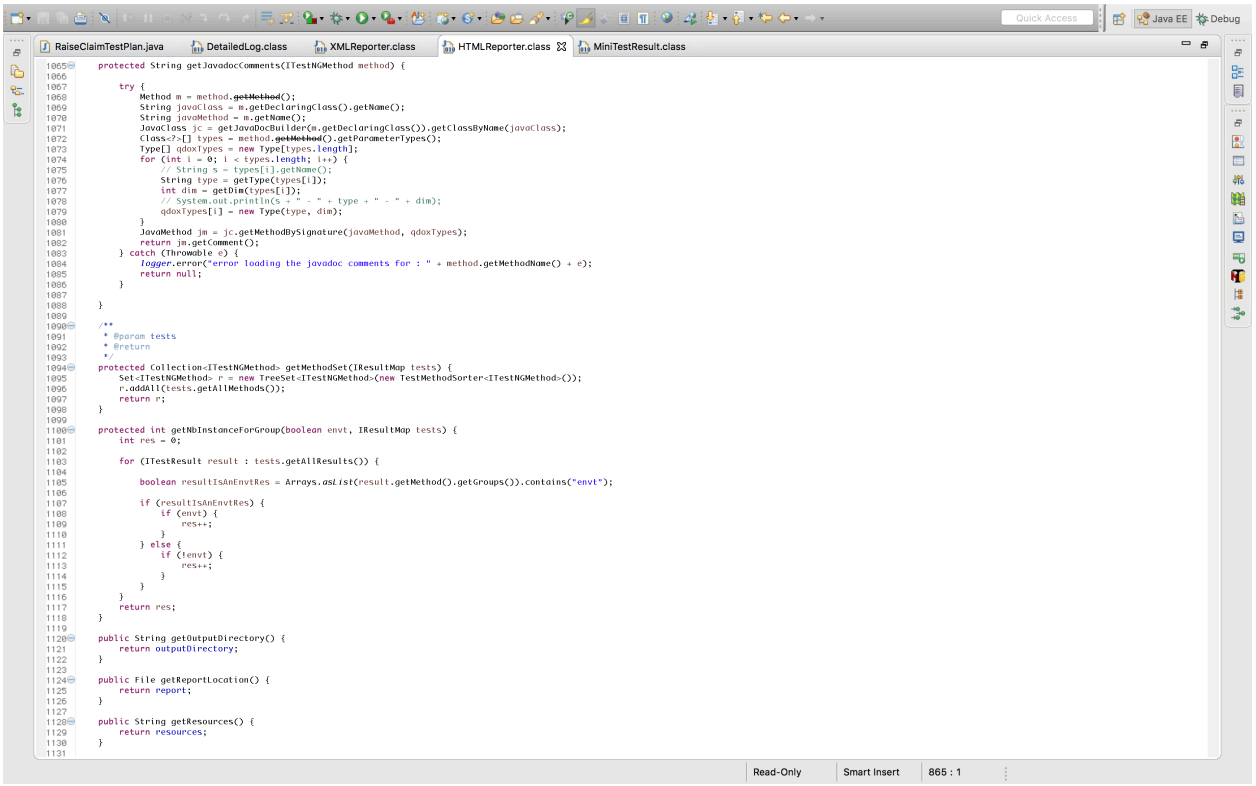

Quick Access
Java EE
Debug
RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Read-Only
Smart Insert
2 : 1


Quick Access
Java EE
Debug
RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Read-Only
Smart Insert
865 : 1

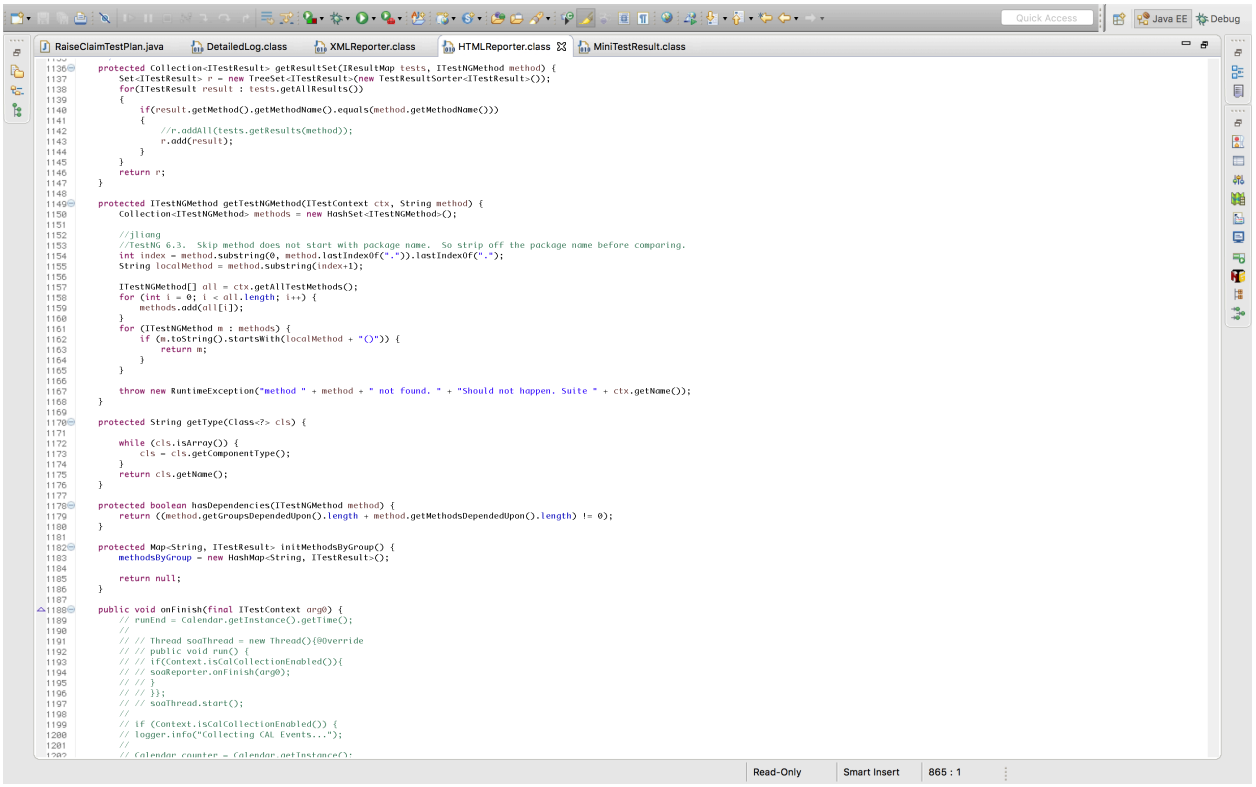

Quick Access
Java EE
Debug
RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Read-Only
Smart Insert
865 : 1




```
    protected int getDim(Class<?> cls) {
        int dim = 0;

        while (cls.isArray()) {
            dim++;
            cls = cls.getComponentType();
        }
        return dim;
    }

    public int getEnvConfigTestsCount(IResultMap map) {
        int count = 0;
        for (ITestNGMethod tm : map.getAllMethods()) {
            String[] groups = tm.getGroups();
            if (groups != null) {
                for (int i = 0; i < groups.length; i++) {
                    if ("envt".equalsIgnoreCase(groups[i])) {
                        count++;
                        break;
                    }
                }
            }
        }

        return count;
    }

    protected ITestResult getFailedOrSkippedResult(ITestContext ctx, ITestNGMethod method) {
        List<ITestResult> res = new LinkedList<ITestResult>();
        res.addAll(failedTests.get(ctx.getName()).getResults(method));
        if (res.size() != 0) {
            return res.get(0);
        }
        res.addAll(ctx.getPassedTests().getResults(method));
        if (res.size() != 0) {
            return res.get(0);
        }

        res.addAll(skippedTests.get(ctx.getName()).getResults(method));
        if (res.size() != 0) {
            return res.get(0);
        }
        return null;
    }

    @SuppressWarnings("rawtypes")
    protected JavaDocBuilder getJavaDocBuilder(Class clz) throws URISyntaxException {
        //String packagePath = "com/ebay/maui/reporter";
        String projectPath = new File("").getAbsolutePath();
        String packagePath = clz.getPackage().getName().replaceAll("\\.", "/");
        if (builder == null) {
            builder = new JavaDocBuilder();

            URL resource = Thread.currentThread().getContextClassLoader().getResource(packagePath);
            File src = new File(resource.toURI());
            builder.addSourceTree(src);

            // project source folder
            File realFolder = new File(projectPath + "/src/main/java/" + packagePath);
            if (realFolder.exists())
                builder.addSourceTree(realFolder);
        }
        return builder;
    }

    /**
     *
```

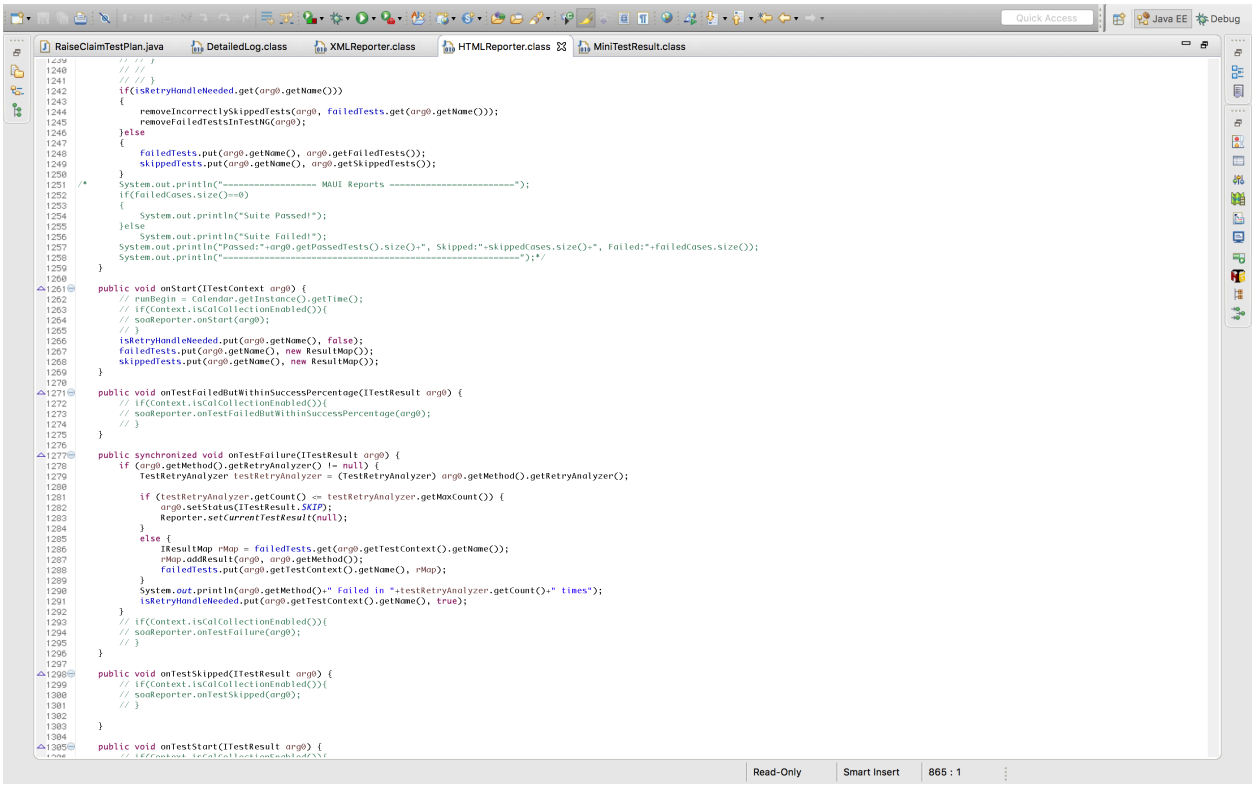



```
    protected String getJavadocComments(ITestNGMethod method) {

        try {
            Method m = method.getMethod();
            String javaClass = m.getDeclaringClass().getName();
            String javaMethod = m.getName();
            JavaClass jc = getJavaDocBuilder(m.getDeclaringClass()).getClassByName(javaClass);
            Class<?>[] types = method.getMethod().getParameterTypes();
            Type[] qdoxTypes = new Type[types.length];
            for (int i = 0; i < types.length; i++) {
                // String s = types[i].getName();
                String type = getType(types[i]);
                int dim = getDim(types[i]);
                // System.out.println(s + " - " + type + " - " + dim);
                qdoxTypes[i] = new Type(type, dim);
            }
            JavaMethod jm = jc.getMethodBySignature(javaMethod, qdoxTypes);
            return jm.getComment();
        } catch (Throwable e) {
            logger.error("error loading the javadoc comments for : " + method.getMethodName() + e);
            return null;
        }

    }

    /**
     * @param tests
     * @return
     */
    protected Collection<ITestNGMethod> getMethodSet(IResultMap tests) {
        Set<ITestNGMethod> r = new TreeSet<ITestNGMethod>(new TestMethodSorter<ITestNGMethod>());
        r.addAll(tests.getAllMethods());
        return r;
    }

    protected int getNbInstanceForGroup(boolean envt, IResultMap tests) {
        int res = 0;

        for (ITestResult result : tests.getAllResults()) {

            boolean resultIsAnEnvtRes = Arrays.asList(result.getMethod().getGroups()).contains("envt");

            if (resultIsAnEnvtRes) {
                if (envt) {
                    res++;
                }
            } else {
                if (!envt) {
                    res++;
                }
            }
        }
        return res;
    }

    public String getOutputDirectory() {
        return outputDirectory;
    }

    public File getReportLocation() {
        return report;
    }

    public String getResources() {
        return resources;
    }
```



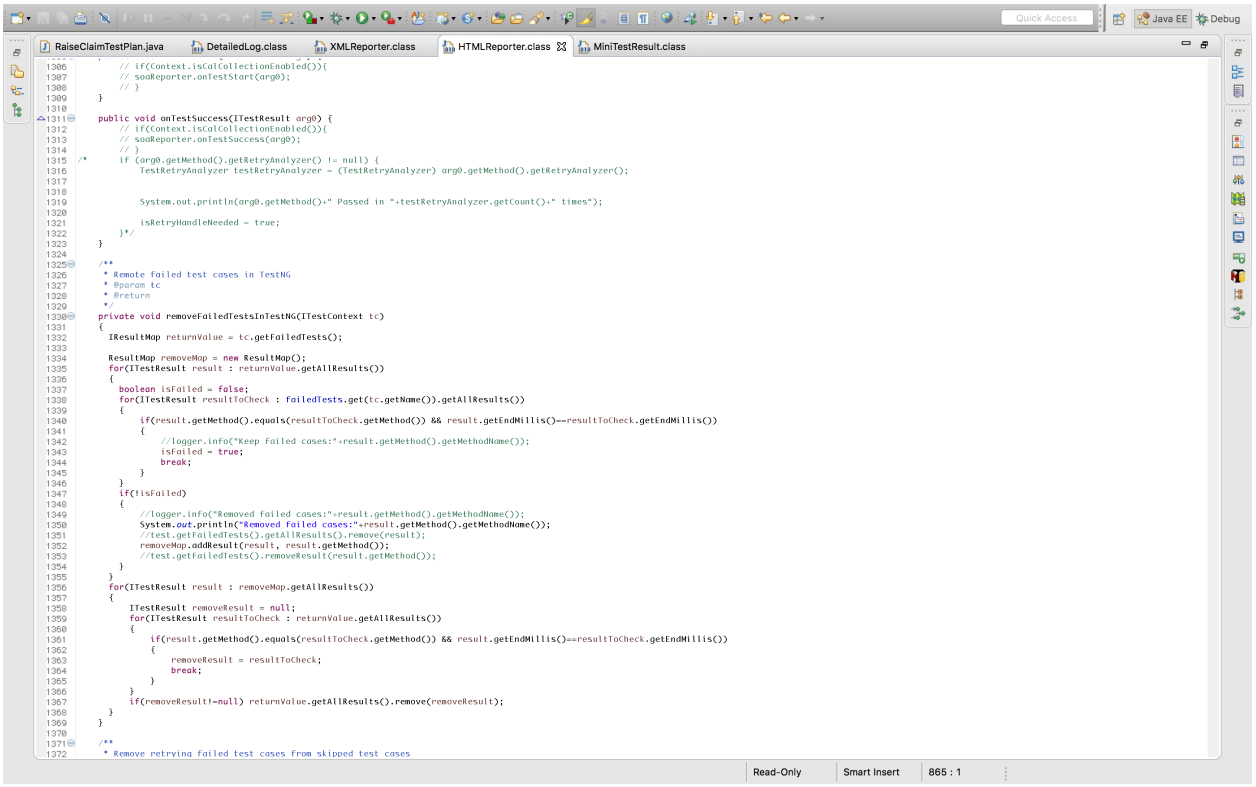
```
    protected Collection<ITestResult> getResultSet(IResultMap tests, ITestNGMethod method) {
        Set<ITestResult> r = new TreeSet<ITestResult>(new TestResultSorter<ITestResult>());
        for(ITestResult result : tests.getAllResults())
        {
            if(result.getMethod().getMethodName().equals(method.getMethodName()))
            {
                //r.addAll(tests.getResults(method));
                r.add(result);
            }
        }
        return r;
    }

    protected ITestNGMethod getTestNGMethod(ITestContext ctx, String method) {
        Collection<ITestNGMethod> methods = new HashSet<ITestNGMethod>();

        //jliang
        //TestNG 6.3.  Skip method does not start with package name.  So strip off the package name before comparing.
        int index = method.substring(0, method.lastIndexOf(".")).lastIndexOf(".");
        String localMethod = method.substring(index+1);

        ITestNGMethod[] all = ctx.getAllTestMethods();
        for (int i = 0; i < all.length; i++) {
            methods.add(all[i]);
        }
        for (ITestNGMethod m : methods) {
            if (m.toString().startsWith(localMethod + "()")) {
                return m;
            }
        }

        throw new RuntimeException("method " + method + " not found. " + "Should not happen. Suite " + ctx.getName());
    }

    protected String getType(Class<?> cls) {

        while (cls.isArray()) {
            cls = cls.getComponentType();
        }
        return cls.getName();
    }

    protected boolean hasDependencies(ITestNGMethod method) {
        return ((method.getGroupsDependedUpon().length + method.getMethodsDependedUpon().length) != 0);
    }

    protected Map<String, ITestResult> initMethodsByGroup() {
        methodsByGroup = new HashMap<String, ITestResult>();

        return null;
    }

    public void onFinish(final ITestContext arg0) {
        // runEnd = Calendar.getInstance().getTime();
        //
        // // Thread soaThread = new Thread(){@Override
        // // public void run() {
        // // if(Context.isCalCollectionEnabled()){
        // // soaReporter.onFinish(arg0);
        // // }
        // // }};
        // // soaThread.start();
        //
        // if (Context.isCalCollectionEnabled()) {
        // logger.info("Collecting CAL Events...");
        //
        // Calendar counter = Calendar.getInstance();
```

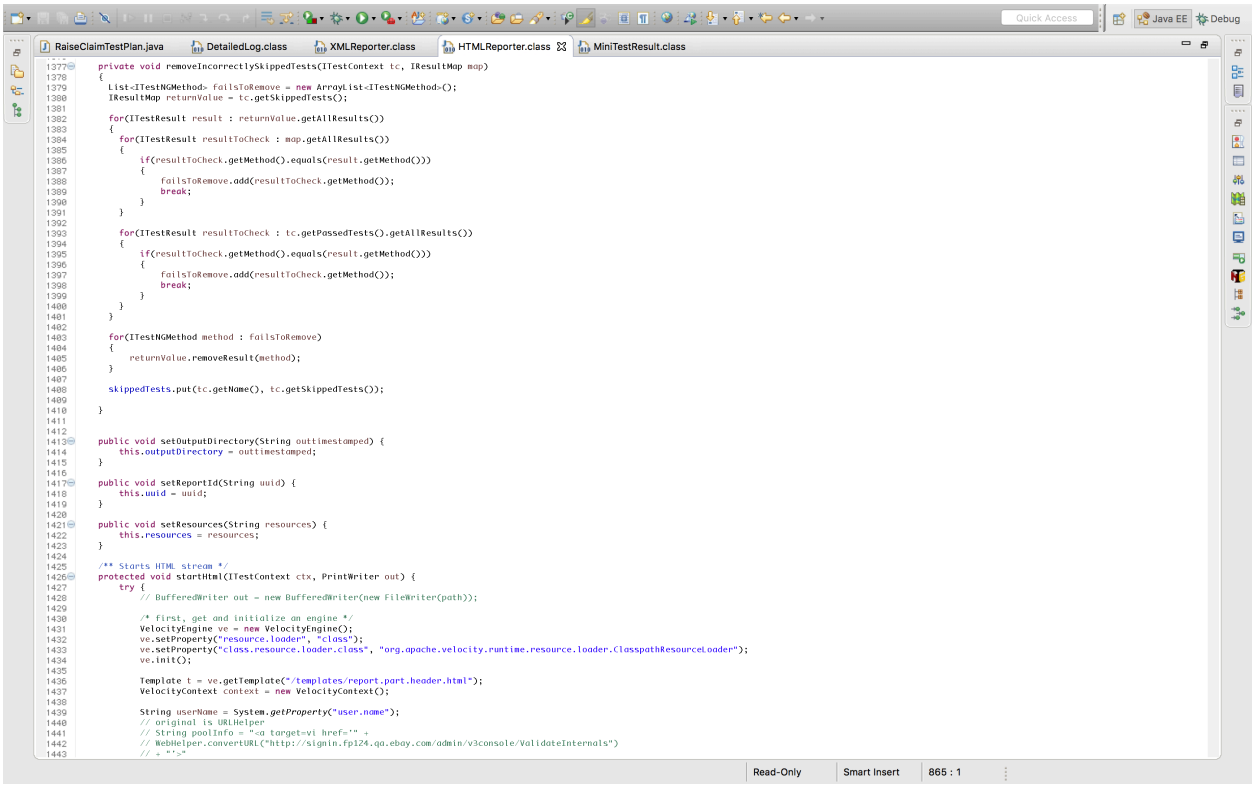
Read-Only | Smart Insert | 865 : 1

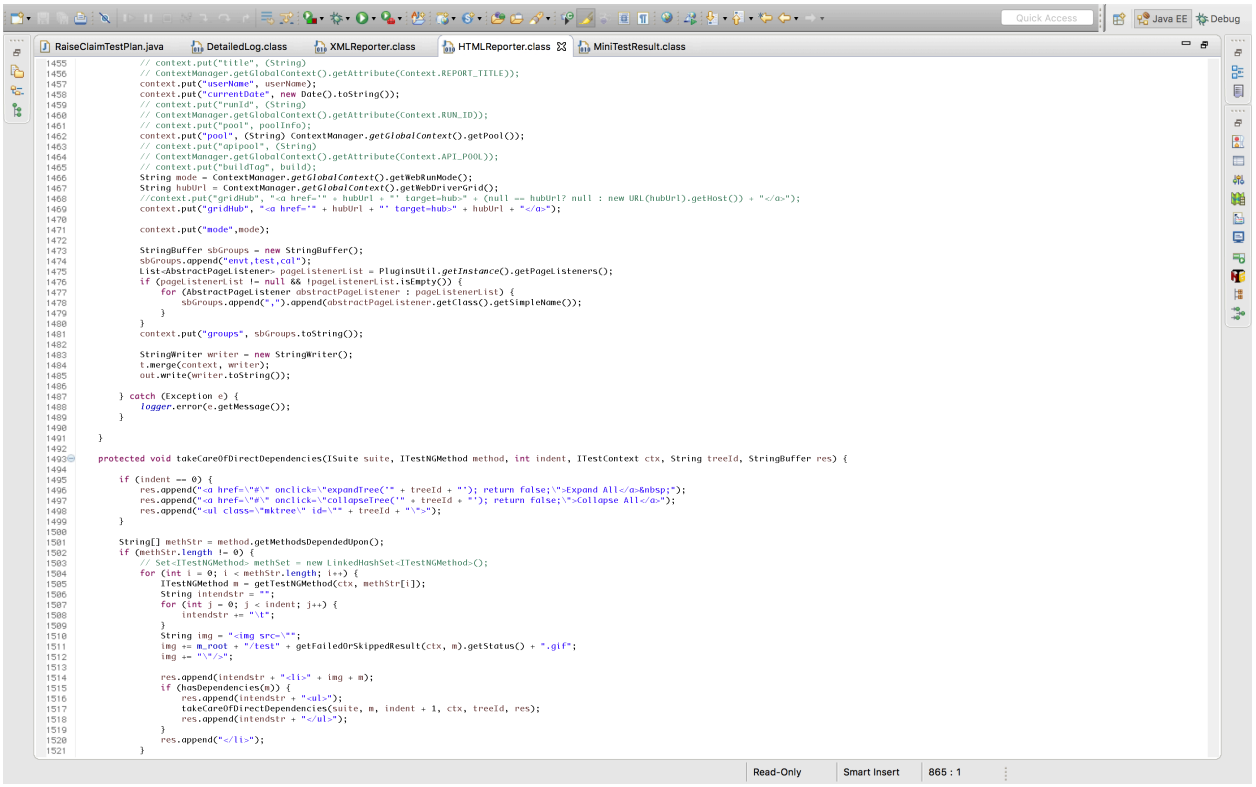


```
        // // }
        //
        // }
        if(isRetryHandleNeeded.get(arg0.getName()))
        {
            removeIncorrectlySkippedTests(arg0, failedTests.get(arg0.getName()));
            removeFailedTestsInTestNG(arg0);
        }else
        {
            failedTests.put(arg0.getName(), arg0.getFailedTests());
            skippedTests.put(arg0.getName(), arg0.getSkippedTests());
        }
/*      System.out.println("================== MAUI Reports ========================");
        if(failedCases.size()==0)
        {
            System.out.println("Suite Passed!");
        }else
            System.out.println("Suite Failed!");
        System.out.println("Passed:"+arg0.getPassedTests().size()+", Skipped:"+skippedCases.size()+", Failed:"+failedCases.size());
        System.out.println("=========================================================");*/
    }

    public void onStart(ITestContext arg0) {
        // runBegin = Calendar.getInstance().getTime();
        // if(Context.isCalCollectionEnabled()){
        // soaReporter.onStart(arg0);
        // }
        isRetryHandleNeeded.put(arg0.getName(), false);
        failedTests.put(arg0.getName(), new ResultMap());
        skippedTests.put(arg0.getName(), new ResultMap());
    }

    public void onTestFailedButWithinSuccessPercentage(ITestResult arg0) {
        // if(Context.isCalCollectionEnabled()){
        // soaReporter.onTestFailedButWithinSuccessPercentage(arg0);
        // }
    }

    public synchronized void onTestFailure(ITestResult arg0) {
        if (arg0.getMethod().getRetryAnalyzer() != null) {
            TestRetryAnalyzer testRetryAnalyzer = (TestRetryAnalyzer) arg0.getMethod().getRetryAnalyzer();

            if (testRetryAnalyzer.getCount() <= testRetryAnalyzer.getMaxCount()) {
                arg0.setStatus(ITestResult.SKIP);
                Reporter.setCurrentTestResult(null);
            }
            else {
                IResultMap rMap = failedTests.get(arg0.getTestContext().getName());
                rMap.addResult(arg0, arg0.getMethod());
                failedTests.put(arg0.getTestContext().getName(), rMap);
            }
            System.out.println(arg0.getMethod()+" Failed in "+testRetryAnalyzer.getCount()+" times");
            isRetryHandleNeeded.put(arg0.getTestContext().getName(), true);
        }
        // if(Context.isCalCollectionEnabled()){
        // soaReporter.onTestFailure(arg0);
        // }
    }

    public void onTestSkipped(ITestResult arg0) {
        // if(Context.isCalCollectionEnabled()){
        // soaReporter.onTestSkipped(arg0);
        // }

    }

    public void onTestStart(ITestResult arg0) {
```

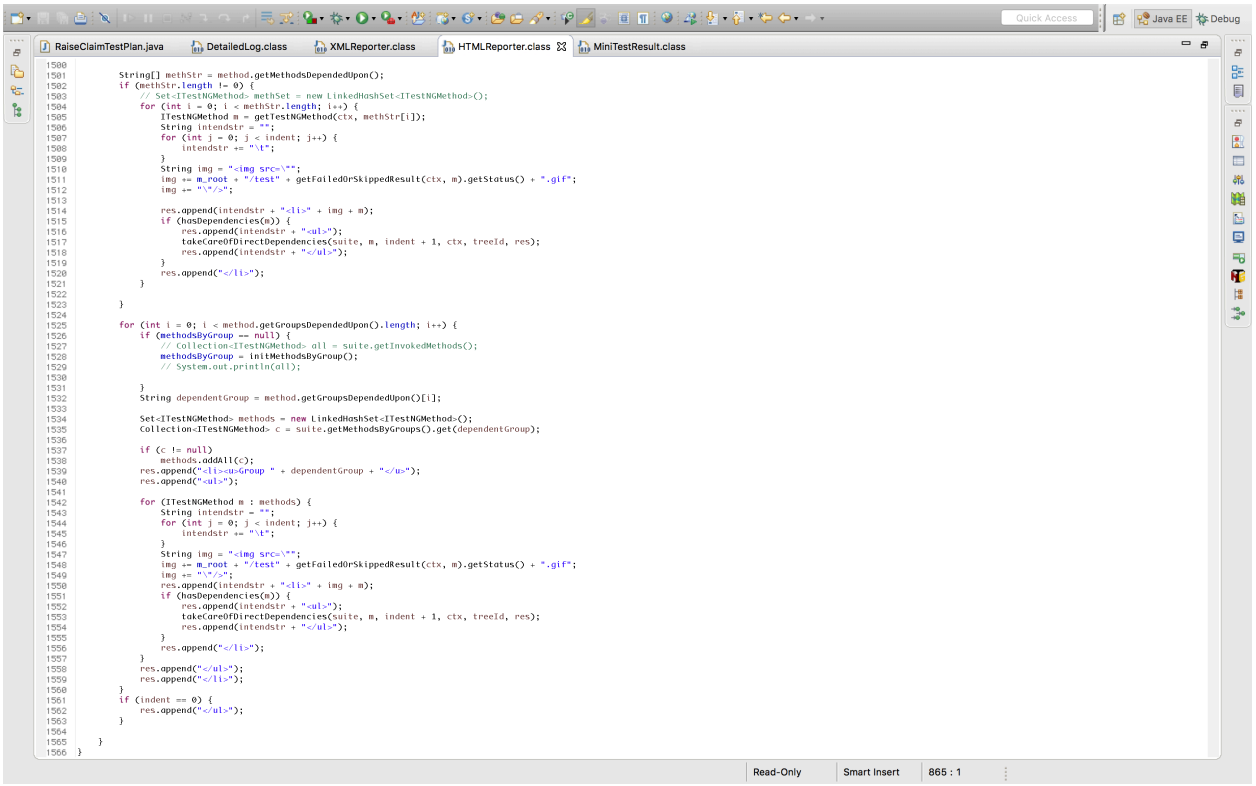
Read-Only | Smart Insert | 865 : 1


RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Quick Access
Java EE
Debug
// if(Context.isCalCollectionEnabled()){
// soaReporter.onTestStart(arg0);
// }
}
public void onTestSuccess(ITestResult arg0) {
// if(Context.isCalCollectionEnabled()){
// soaReporter.onTestSuccess(arg0);
// }
/* if (arg0.getMethod().getRetryAnalyzer() != null) {
TestRetryAnalyzer testRetryAnalyzer = (TestRetryAnalyzer) arg0.getMethod().getRetryAnalyzer();
System.out.println(arg0.getMethod()+" Passed in "+testRetryAnalyzer.getCount()+" times");
isRetryHandleNeeded = true;
}*/
}
/**
* Remote failed test cases in TestNG
* @param tc
* @return
*/
private void removeFailedTestsInTestNG(ITestContext tc)
{
IResultMap returnValue = tc.getFailedTests();
ResultMap removeMap = new ResultMap();
for(ITestResult result : returnValue.getAllResults())
{
boolean isFailed = false;
for(ITestResult resultToCheck : failedTests.get(tc.getName()).getAllResults())
{
if(result.getMethod().equals(resultToCheck.getMethod()) && result.getEndMillis()==resultToCheck.getEndMillis())
{
//logger.info("Keep failed cases:"+result.getMethod().getMethodName());
isFailed = true;
break;
}
}
if(!isFailed)
{
//logger.info("Removed failed cases:"+result.getMethod().getMethodName());
System.out.println("Removed failed cases:"+result.getMethod().getMethodName());
//test.getFailedTests().getAllResults().remove(result);
removeMap.addResult(result, result.getMethod());
//test.getFailedTests().removeResult(result.getMethod());
}
}
for(ITestResult result : removeMap.getAllResults())
{
ITestResult removeResult = null;
for(ITestResult resultToCheck : returnValue.getAllResults())
{
if(result.getMethod().equals(resultToCheck.getMethod()) && result.getEndMillis()==resultToCheck.getEndMillis())
{
removeResult = resultToCheck;
break;
}
}
if(removeResult!=null) returnValue.getAllResults().remove(removeResult);
}
}
/**
* Remove retrying failed test cases from skipped test cases
Read-Only
Smart Insert
865 : 1

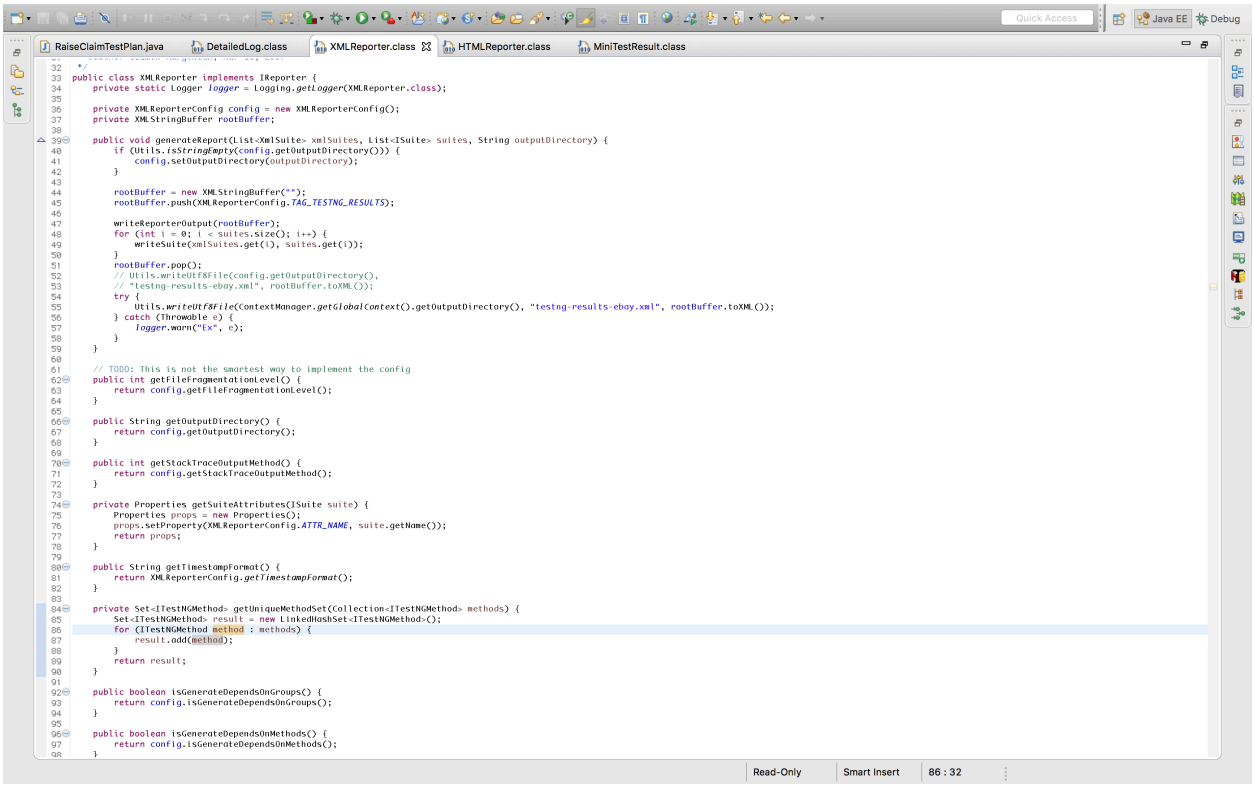

RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Quick Access
Java EE
Debug
private void removeIncorrectlySkippedTests(ITestContext tc, IResultMap map)
{
List<ITestNGMethod> failsToRemove = new ArrayList<ITestNGMethod>();
IResultMap returnValue = tc.getSkippedTests();
for(ITestResult result : returnValue.getAllResults())
{
for(ITestResult resultToCheck : map.getAllResults())
{
if(resultToCheck.getMethod().equals(result.getMethod()))
{
failsToRemove.add(resultToCheck.getMethod());
break;
}
}
for(ITestResult resultToCheck : tc.getPassedTests().getAllResults())
{
if(resultToCheck.getMethod().equals(result.getMethod()))
{
failsToRemove.add(resultToCheck.getMethod());
break;
}
}
}
for(ITestNGMethod method : failsToRemove)
{
returnValue.removeResult(method);
}
skippedTests.put(tc.getName(), tc.getSkippedTests());
}
public void setOutputDirectory(String outtimestamped) {
this.outputDirectory = outtimestamped;
}
public void setReportId(String uuid) {
this.uuid = uuid;
}
public void setResources(String resources) {
this.resources = resources;
}
/** Starts HTML stream */
protected void startHtml(ITestContext ctx, PrintWriter out) {
try {
// BufferedWriter out = new BufferedWriter(new FileWriter(path));
/* First, get and initialize an engine */
VelocityEngine ve = new VelocityEngine();
ve.setProperty("resource.loader", "class");
ve.setProperty("class.resource.loader.class", "org.apache.velocity.runtime.resource.loader.ClasspathResourceLoader");
ve.init();
Template t = ve.getTemplate("/templates/report.part.header.html");
VelocityContext context = new VelocityContext();
String userName = System.getProperty("user.name");
// original is URLHelper
// String poolInfo = "<a target=vi href='" +
// WebHelper.convertURL("http://signin.fp124.qa.ebay.com/admin/v3console/ValidateInternals")
// + "'>"
Read-Only
Smart Insert
865 : 1



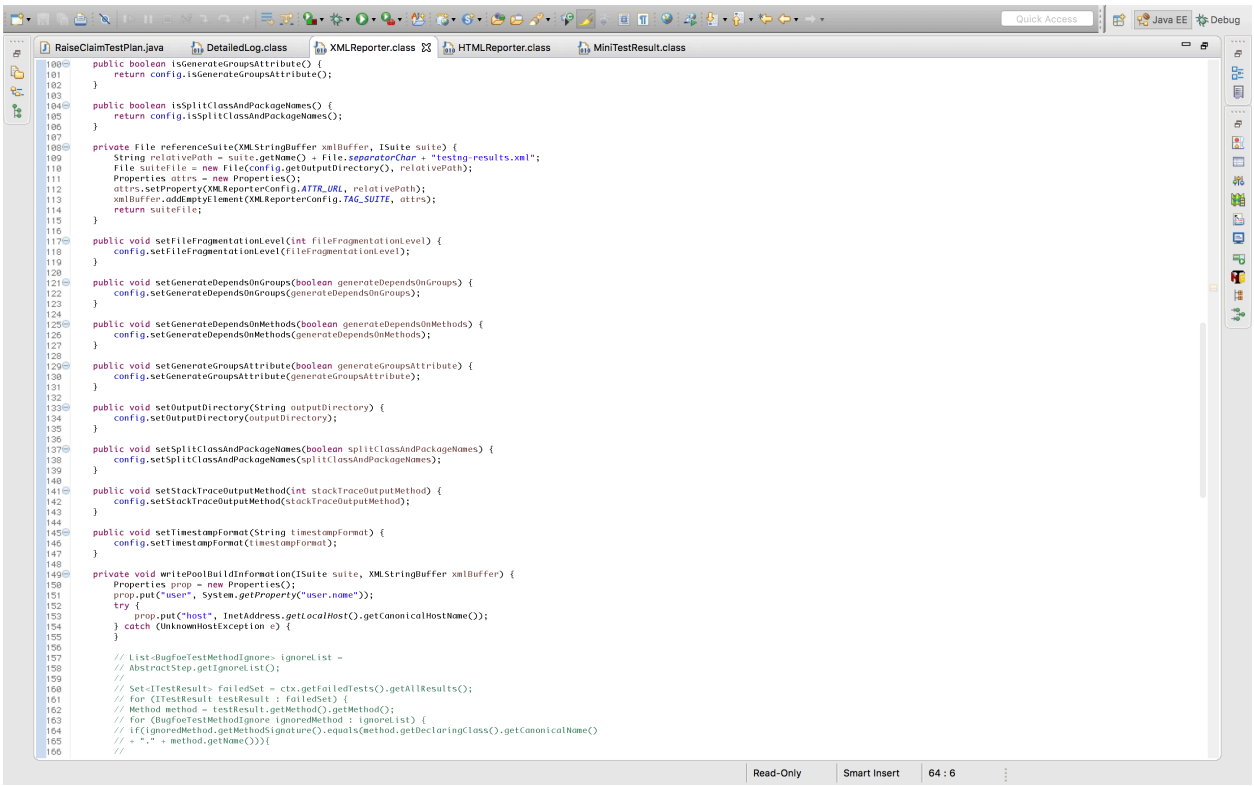
```
            // context.put("title", (String)
            // ContextManager.getGlobalContext().getAttribute(Context.REPORT_TITLE));
            context.put("userName", userName);
            context.put("currentDate", new Date().toString());
            // context.put("runId", (String)
            // ContextManager.getGlobalContext().getAttribute(Context.RUN_ID));
            // context.put("pool", poolInfo);
            context.put("pool", (String) ContextManager.getGlobalContext().getPool());
            // context.put("apipool", (String)
            // ContextManager.getGlobalContext().getAttribute(Context.API_POOL));
            // context.put("buildTag", build);
            String mode = ContextManager.getGlobalContext().getWebRunMode();
            String hubUrl = ContextManager.getGlobalContext().getWebDriverGrid();
            //context.put("gridHub", "<a href='" + hubUrl + "' target=hub>" + (null == hubUrl? null : new URL(hubUrl).getHost()) + "</a>");
            context.put("gridHub", "<a href='" + hubUrl + "' target=hub>" + hubUrl + "</a>");

            context.put("mode",mode);

            StringBuffer sbGroups = new StringBuffer();
            sbGroups.append("envt,test,cal");
            List<AbstractPageListener> pageListenerList = PluginsUtil.getInstance().getPageListeners();
            if (pageListenerList != null && !pageListenerList.isEmpty()) {
                for (AbstractPageListener abstractPageListener : pageListenerList) {
                    sbGroups.append(",").append(abstractPageListener.getClass().getSimpleName());
                }
            }
            context.put("groups", sbGroups.toString());

            StringWriter writer = new StringWriter();
            t.merge(context, writer);
            out.write(writer.toString());

        } catch (Exception e) {
            logger.error(e.getMessage());
        }

    }

    protected void takeCareOfDirectDependencies(ISuite suite, ITestNGMethod method, int indent, ITestContext ctx, String treeId, StringBuffer res) {

        if (indent == 0) {
            res.append("<a href=\"#\" onclick=\"expandTree('" + treeId + "'); return false;\">Expand All</a> ");
            res.append("<a href=\"#\" onclick=\"collapseTree('" + treeId + "'); return false;\">Collapse All</a>");
            res.append("<ul class=\"mktree\" id=\"" + treeId + "\">");
        }

        String[] methStr = method.getMethodsDependedUpon();
        if (methStr.length != 0) {
            // Set<ITestNGMethod> methSet = new LinkedHashSet<ITestNGMethod>();
            for (int i = 0; i < methStr.length; i++) {
                ITestNGMethod m = getTestNGMethod(ctx, methStr[i]);
                String intendstr = "";
                for (int j = 0; j < indent; j++) {
                    intendstr += "\t";
                }
                String img = "<img src=\"";
                img += m_root + "/test" + getFailedOrSkippedResult(ctx, m).getStatus() + ".gif";
                img += "\"/>";

                res.append(intendstr + "<li>" + img + m);
                if (hasDependencies(m)) {
                    res.append(intendstr + "<ul>");
                    takeCareOfDirectDependencies(suite, m, indent + 1, ctx, treeId, res);
                    res.append(intendstr + "</ul>");
                }
                res.append("</li>");
            }
```



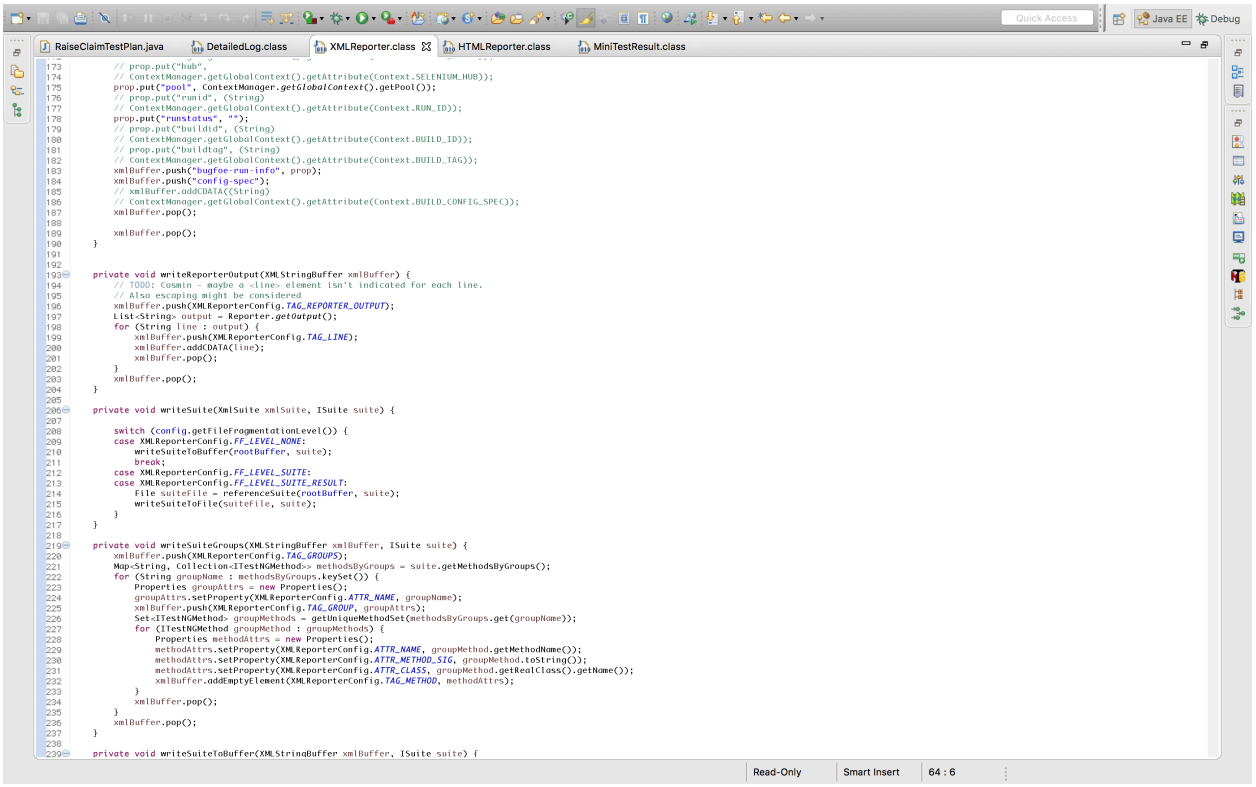
```
        String[] methStr = method.getMethodsDependedUpon();
        if (methStr.length != 0) {
            // Set<ITestNGMethod> methSet = new LinkedHashSet<ITestNGMethod>();
            for (int i = 0; i < methStr.length; i++) {
                ITestNGMethod m = getTestNGMethod(ctx, methStr[i]);
                String intendstr = "";
                for (int j = 0; j < indent; j++) {
                    intendstr += "\t";
                }
                String img = "<img src=\"";
                img += m_root + "/test" + getFailedOrSkippedResult(ctx, m).getStatus() + ".gif";
                img += "\"/>";

                res.append(intendstr + "<li>" + img + m);
                if (hasDependencies(m)) {
                    res.append(intendstr + "<ul>");
                    takeCareOfDirectDependencies(suite, m, indent + 1, ctx, treeId, res);
                    res.append(intendstr + "</ul>");
                }
                res.append("</li>");
            }

        }

        for (int i = 0; i < method.getGroupsDependedUpon().length; i++) {
            if (methodsByGroup == null) {
                // Collection<ITestNGMethod> all = suite.getInvokedMethods();
                methodsByGroup = initMethodsByGroup();
                // System.out.println(all);

            }
            String dependentGroup = method.getGroupsDependedUpon()[i];

            Set<ITestNGMethod> methods = new LinkedHashSet<ITestNGMethod>();
            Collection<ITestNGMethod> c = suite.getMethodsByGroups().get(dependentGroup);

            if (c != null)
                methods.addAll(c);
            res.append("<li><u>Group " + dependentGroup + "</u>");
            res.append("<ul>");

            for (ITestNGMethod m : methods) {
                String intendstr = "";
                for (int j = 0; j < indent; j++) {
                    intendstr += "\t";
                }
                String img = "<img src=\"";
                img += m_root + "/test" + getFailedOrSkippedResult(ctx, m).getStatus() + ".gif";
                img += "\"/>";
                res.append(intendstr + "<li>" + img + m);
                if (hasDependencies(m)) {
                    res.append(intendstr + "<ul>");
                    takeCareOfDirectDependencies(suite, m, indent + 1, ctx, treeId, res);
                    res.append(intendstr + "</ul>");
                }
                res.append("</li>");
            }
            res.append("</ul>");
            res.append("</li>");
        }
        if (indent == 0) {
            res.append("</ul>");
        }

    }
}
```



```
*/
public class XMLReporter implements IReporter {
    private static Logger logger = Logging.getLogger(XMLReporter.class);

    private XMLReporterConfig config = new XMLReporterConfig();
    private XMLStringBuffer rootBuffer;

    public void generateReport(List<XmlSuite> xmlSuites, List<ISuite> suites, String outputDirectory) {
        if (Utils.isStringEmpty(config.getOutputDirectory())) {
            config.setOutputDirectory(outputDirectory);
        }

        rootBuffer = new XMLStringBuffer("");
        rootBuffer.push(XMLReporterConfig.TAG_TESTNG_RESULTS);

        writeReporterOutput(rootBuffer);
        for (int i = 0; i < suites.size(); i++) {
            writeSuite(xmlSuites.get(i), suites.get(i));
        }
        rootBuffer.pop();
        // Utils.writeUtf8File(config.getOutputDirectory(),
        // "testng-results-ebay.xml", rootBuffer.toXML());
        try {
            Utils.writeUtf8File(ContextManager.getGlobalContext().getOutputDirectory(), "testng-results-ebay.xml", rootBuffer.toXML());
        } catch (Throwable e) {
            logger.warn("Ex", e);
        }
    }

    // TODO: This is not the smartest way to implement the config
    public int getFileFragmentationLevel() {
        return config.getFileFragmentationLevel();
    }

    public String getOutputDirectory() {
        return config.getOutputDirectory();
    }

    public int getStackTraceOutputMethod() {
        return config.getStackTraceOutputMethod();
    }

    private Properties getSuiteAttributes(ISuite suite) {
        Properties props = new Properties();
        props.setProperty(XMLReporterConfig.ATTR_NAME, suite.getName());
        return props;
    }

    public String getTimestampFormat() {
        return XMLReporterConfig.getTimestampFormat();
    }

    private Set<ITestNGMethod> getUniqueMethodSet(Collection<ITestNGMethod> methods) {
        Set<ITestNGMethod> result = new LinkedHashSet<ITestNGMethod>();
        for (ITestNGMethod method : methods) {
            result.add(method);
        }
        return result;
    }

    public boolean isGenerateDependsOnGroups() {
        return config.isGenerateDependsOnGroups();
    }

    public boolean isGenerateDependsOnMethods() {
        return config.isGenerateDependsOnMethods();
    }
```

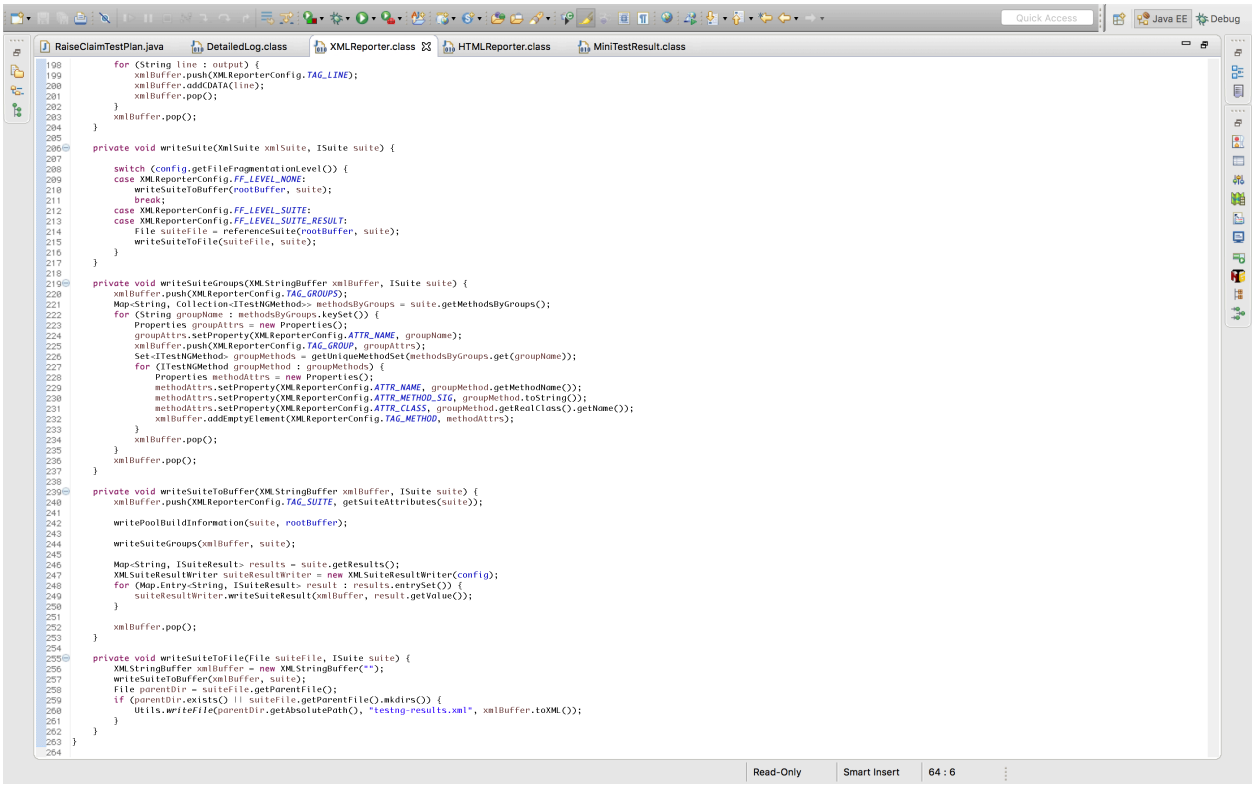


```
    public boolean isGenerateGroupsAttribute() {
        return config.isGenerateGroupsAttribute();
    }

    public boolean isSplitClassAndPackageNames() {
        return config.isSplitClassAndPackageNames();
    }

    private File referenceSuite(XMLStringBuffer xmlBuffer, ISuite suite) {
        String relativePath = suite.getName() + File.separatorChar + "testng-results.xml";
        File suiteFile = new File(config.getOutputDirectory(), relativePath);
        Properties attrs = new Properties();
        attrs.setProperty(XMLReporterConfig.ATTR_URL, relativePath);
        xmlBuffer.addEmptyElement(XMLReporterConfig.TAG_SUITE, attrs);
        return suiteFile;
    }

    public void setFileFragmentationLevel(int fileFragmentationLevel) {
        config.setFileFragmentationLevel(fileFragmentationLevel);
    }

    public void setGenerateDependsOnGroups(boolean generateDependsOnGroups) {
        config.setGenerateDependsOnGroups(generateDependsOnGroups);
    }

    public void setGenerateDependsOnMethods(boolean generateDependsOnMethods) {
        config.setGenerateDependsOnMethods(generateDependsOnMethods);
    }

    public void setGenerateGroupsAttribute(boolean generateGroupsAttribute) {
        config.setGenerateGroupsAttribute(generateGroupsAttribute);
    }

    public void setOutputDirectory(String outputDirectory) {
        config.setOutputDirectory(outputDirectory);
    }

    public void setSplitClassAndPackageNames(boolean splitClassAndPackageNames) {
        config.setSplitClassAndPackageNames(splitClassAndPackageNames);
    }

    public void setStackTraceOutputMethod(int stackTraceOutputMethod) {
        config.setStackTraceOutputMethod(stackTraceOutputMethod);
    }

    public void setTimestampFormat(String timestampFormat) {
        config.setTimestampFormat(timestampFormat);
    }

    private void writePoolBuildInformation(ISuite suite, XMLStringBuffer xmlBuffer) {
        Properties prop = new Properties();
        prop.put("user", System.getProperty("user.name"));
        try {
            prop.put("host", InetAddress.getLocalHost().getCanonicalHostName());
        } catch (UnknownHostException e) {
        }

        // List<BugfoeTestMethodIgnore> ignoreList =
        // AbstractStep.getIgnoreList();
        //
        // Set<ITestResult> failedSet = ctx.getFailedTests().getAllResults();
        // for (ITestResult testResult : failedSet) {
        // Method method = testResult.getMethod().getMethod();
        // for (BugfoeTestMethodIgnore ignoredMethod : ignoreList) {
        // if(ignoredMethod.getMethodSignature().equals(method.getDeclaringClass().getCanonicalName()
        // + "." + method.getName())){
        //
```



```
        // prop.put("hub",
        // ContextManager.getGlobalContext().getAttribute(Context.SELENIUM_HUB));
        prop.put("rpool", ContextManager.getGlobalContext().getPool());
        // prop.put("runid", (String)
        // ContextManager.getGlobalContext().getAttribute(Context.RUN_ID));
        prop.put("runstatus", "");
        // prop.put("buildid", (String)
        // ContextManager.getGlobalContext().getAttribute(Context.BUILD_ID));
        // prop.put("buildtag", (String)
        // ContextManager.getGlobalContext().getAttribute(Context.BUILD_TAG));
        xmlBuffer.push("bugfoe-run-info", prop);
        xmlBuffer.push("config-spec");
        // xmlBuffer.addCDATA((String)
        // ContextManager.getGlobalContext().getAttribute(Context.BUILD_CONFIG_SPEC));
        xmlBuffer.pop();

        xmlBuffer.pop();
    }


    private void writeReporterOutput(XMLStringBuffer xmlBuffer) {
        // TODO: Cosmin - maybe a <line> element isn't indicated for each line.
        // Also escaping might be considered
        xmlBuffer.push(XMLReporterConfig.TAG_REPORTER_OUTPUT);
        List<String> output = Reporter.getOutput();
        for (String line : output) {
            xmlBuffer.push(XMLReporterConfig.TAG_LINE);
            xmlBuffer.addCDATA(line);
            xmlBuffer.pop();
        }
        xmlBuffer.pop();
    }

    private void writeSuite(XmlSuite xmlSuite, ISuite suite) {

        switch (config.getFileFragmentationLevel()) {
        case XMLReporterConfig.FF_LEVEL_NONE:
            writeSuiteToBuffer(rootBuffer, suite);
            break;
        case XMLReporterConfig.FF_LEVEL_SUITE:
        case XMLReporterConfig.FF_LEVEL_SUITE_RESULT:
            File suiteFile = referenceSuite(rootBuffer, suite);
            writeSuiteToFile(suiteFile, suite);
        }
    }

    private void writeSuiteGroups(XMLStringBuffer xmlBuffer, ISuite suite) {
        xmlBuffer.push(XMLReporterConfig.TAG_GROUPS);
        Map<String, Collection<ITestNGMethod>> methodsByGroups = suite.getMethodsByGroups();
        for (String groupName : methodsByGroups.keySet()) {
            Properties groupAttrs = new Properties();
            groupAttrs.setProperty(XMLReporterConfig.ATTR_NAME, groupName);
            xmlBuffer.push(XMLReporterConfig.TAG_GROUP, groupAttrs);
            Set<ITestNGMethod> groupMethods = getUniqueMethodSet(methodsByGroups.get(groupName));
            for (ITestNGMethod groupMethod : groupMethods) {
                Properties methodAttrs = new Properties();
                methodAttrs.setProperty(XMLReporterConfig.ATTR_NAME, groupMethod.getMethodName());
                methodAttrs.setProperty(XMLReporterConfig.ATTR_METHOD_SIG, groupMethod.toString());
                methodAttrs.setProperty(XMLReporterConfig.ATTR_CLASS, groupMethod.getRealClass().getName());
                xmlBuffer.addEmptyElement(XMLReporterConfig.TAG_METHOD, methodAttrs);
            }
            xmlBuffer.pop();
        }
        xmlBuffer.pop();
    }

    private void writeSuiteToBuffer(XMLStringBuffer xmlBuffer, ISuite suite) {
```

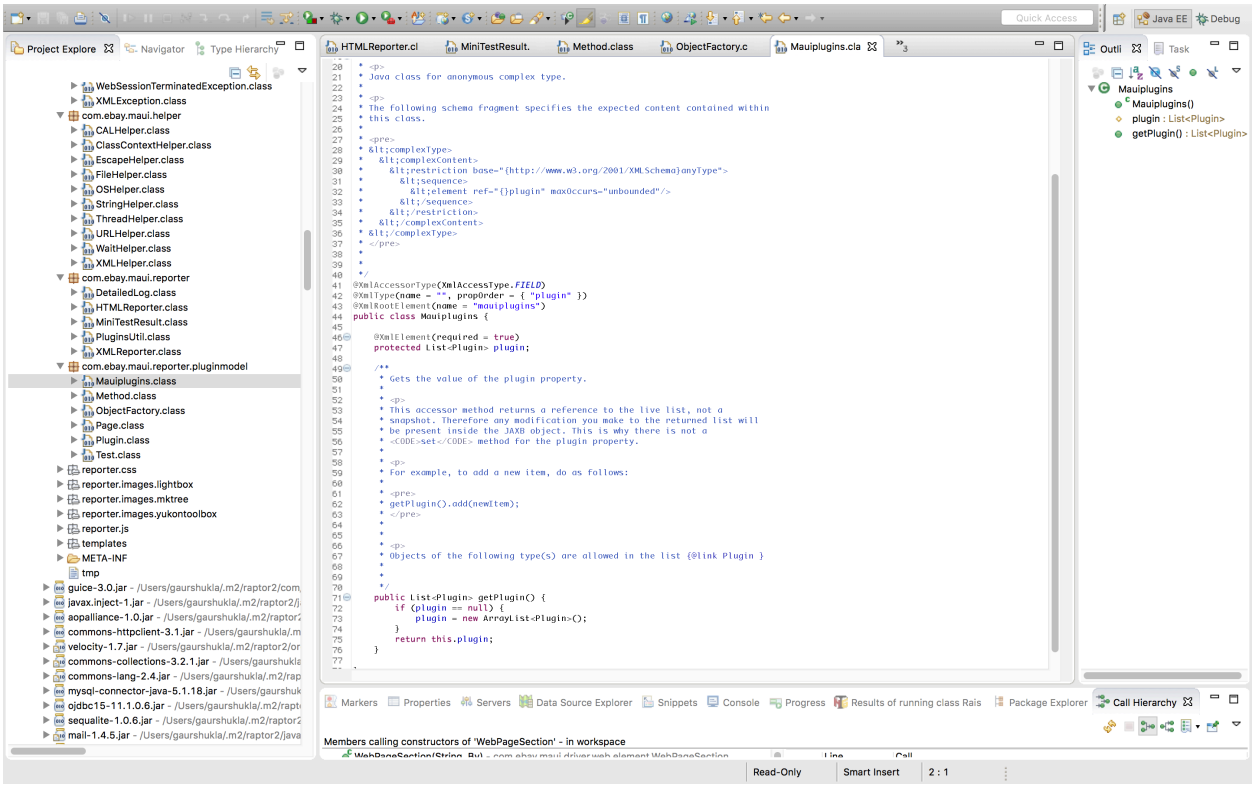
Read-Only | Smart Insert | 64 : 6

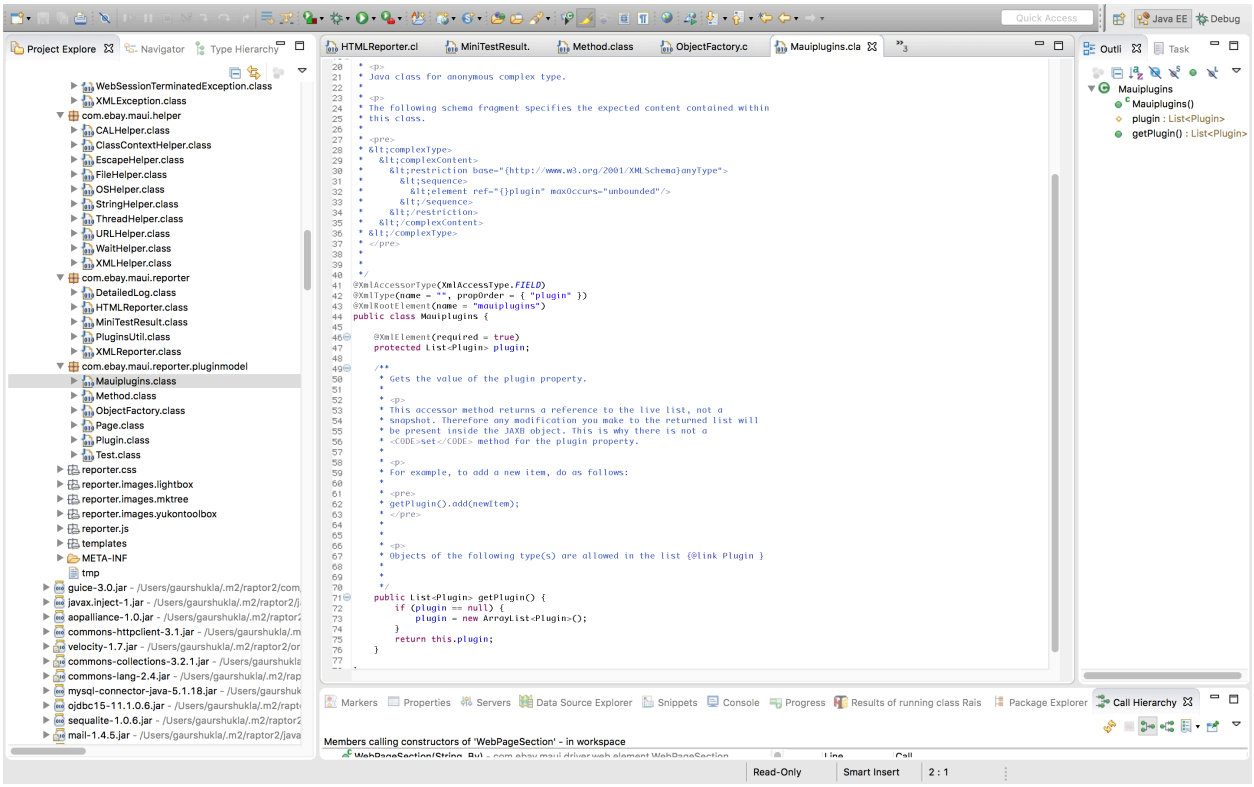


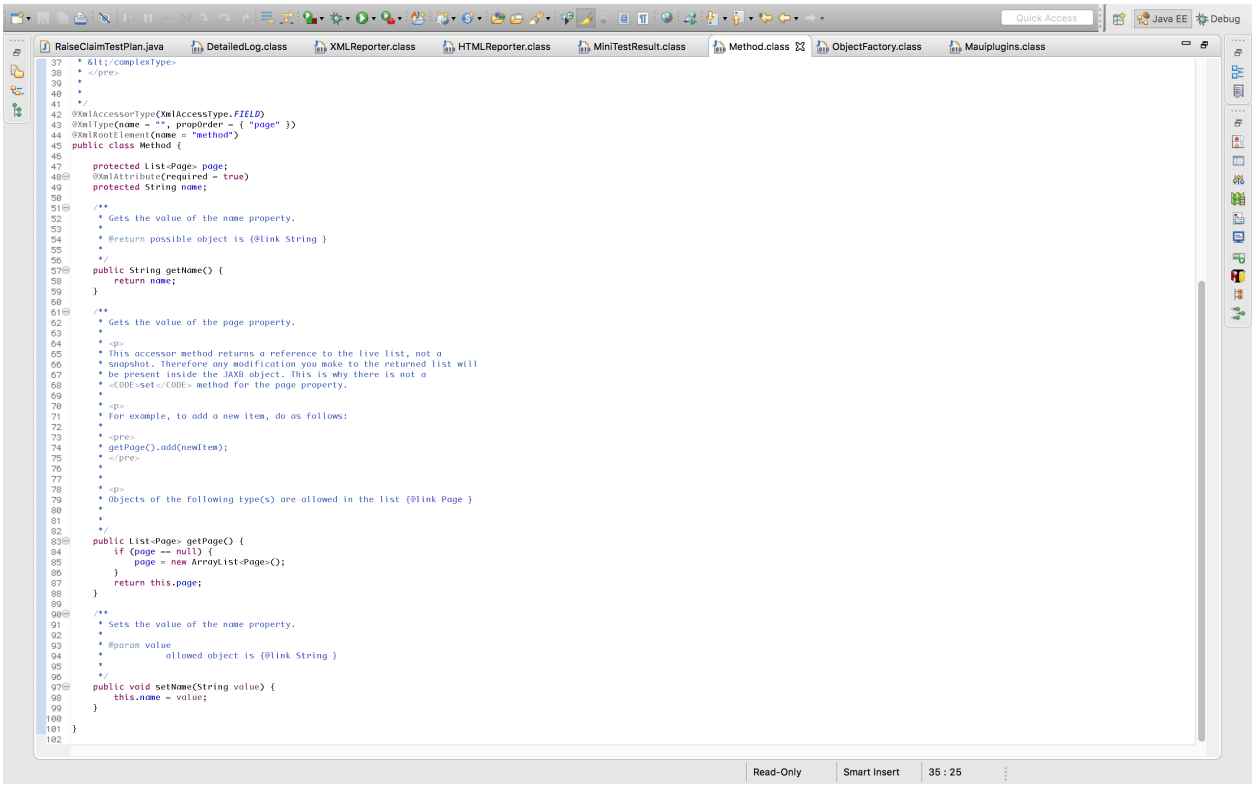
```
        for (String line : output) {
            xmlBuffer.push(XMLReporterConfig.TAG_LINE);
            xmlBuffer.addCDATA(line);
            xmlBuffer.pop();
        }
        xmlBuffer.pop();
    }

    private void writeSuite(XmlSuite xmlSuite, ISuite suite) {

        switch (config.getFileFragmentationLevel()) {
        case XMLReporterConfig.FF_LEVEL_NONE:
            writeSuiteToBuffer(rootBuffer, suite);
            break;
        case XMLReporterConfig.FF_LEVEL_SUITE:
        case XMLReporterConfig.FF_LEVEL_SUITE_RESULT:
            File suiteFile = referenceSuite(rootBuffer, suite);
            writeSuiteToFile(suiteFile, suite);
        }
    }

    private void writeSuiteGroups(XMLStringBuffer xmlBuffer, ISuite suite) {
        xmlBuffer.push(XMLReporterConfig.TAG_GROUPS);
        Map<String, Collection<ITestNGMethod>> methodsByGroups = suite.getMethodsByGroups();
        for (String groupName : methodsByGroups.keySet()) {
            Properties groupAttrs = new Properties();
            groupAttrs.setProperty(XMLReporterConfig.ATTR_NAME, groupName);
            xmlBuffer.push(XMLReporterConfig.TAG_GROUP, groupAttrs);
            Set<ITestNGMethod> groupMethods = getUniqueMethodSet(methodsByGroups.get(groupName));
            for (ITestNGMethod groupMethod : groupMethods) {
                Properties methodAttrs = new Properties();
                methodAttrs.setProperty(XMLReporterConfig.ATTR_NAME, groupMethod.getMethodName());
                methodAttrs.setProperty(XMLReporterConfig.ATTR_METHOD_SIG, groupMethod.toString());
                methodAttrs.setProperty(XMLReporterConfig.ATTR_CLASS, groupMethod.getRealClass().getName());
                xmlBuffer.addEmptyElement(XMLReporterConfig.TAG_METHOD, methodAttrs);
            }
            xmlBuffer.pop();
        }
        xmlBuffer.pop();
    }

    private void writeSuiteToBuffer(XMLStringBuffer xmlBuffer, ISuite suite) {
        xmlBuffer.push(XMLReporterConfig.TAG_SUITE, getSuiteAttributes(suite));

        writePoolBuildInformation(suite, rootBuffer);

        writeSuiteGroups(xmlBuffer, suite);

        Map<String, ISuiteResult> results = suite.getResults();
        XMLSuiteResultWriter suiteResultWriter = new XMLSuiteResultWriter(config);
        for (Map.Entry<String, ISuiteResult> result : results.entrySet()) {
            suiteResultWriter.writeSuiteResult(xmlBuffer, result.getValue());
        }

        xmlBuffer.pop();
    }

    private void writeSuiteToFile(File suiteFile, ISuite suite) {
        XMLStringBuffer xmlBuffer = new XMLStringBuffer("");
        writeSuiteToBuffer(xmlBuffer, suite);
        File parentDir = suiteFile.getParentFile();
        if (parentDir.exists() || suiteFile.getParentFile().mkdirs()) {
            Utils.writeFile(parentDir.getAbsolutePath(), "testng-results.xml", xmlBuffer.toXML());
        }
    }
}
```

Read-Only | Smart Insert | 64 : 6

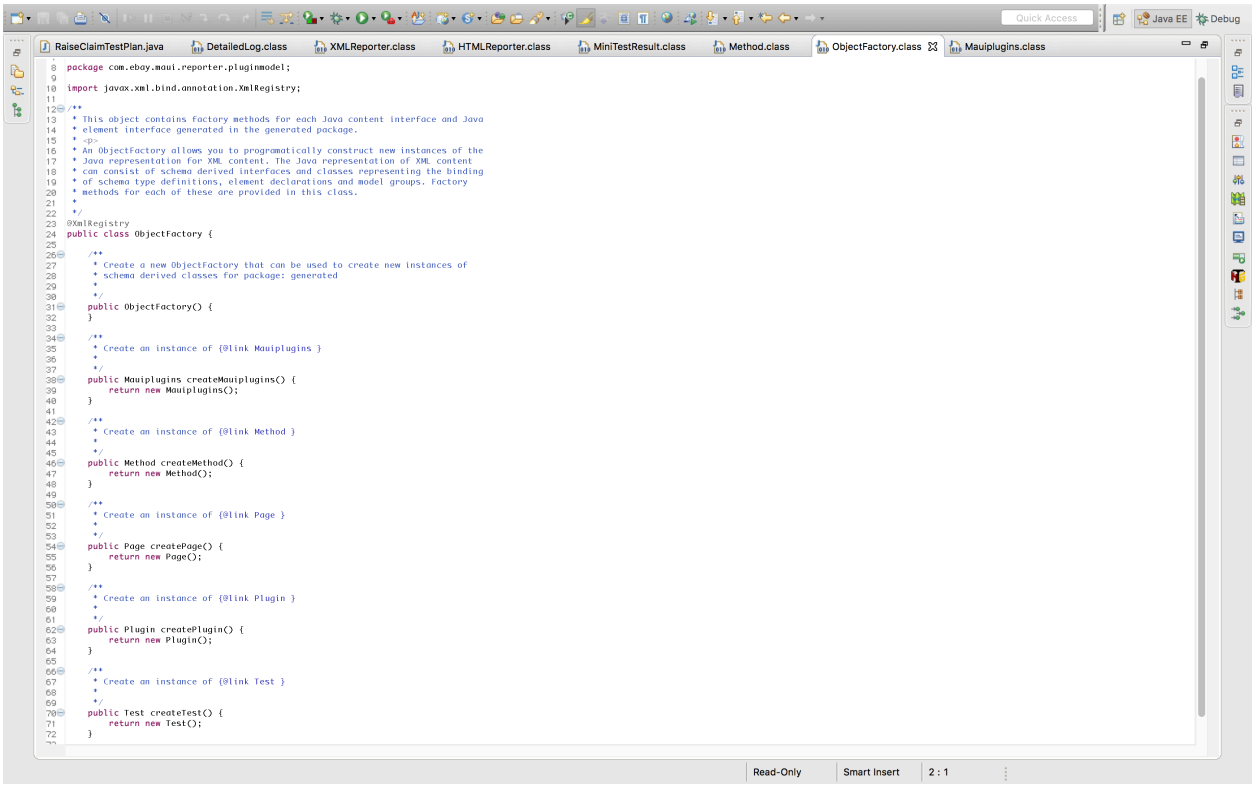
PLUGIN:



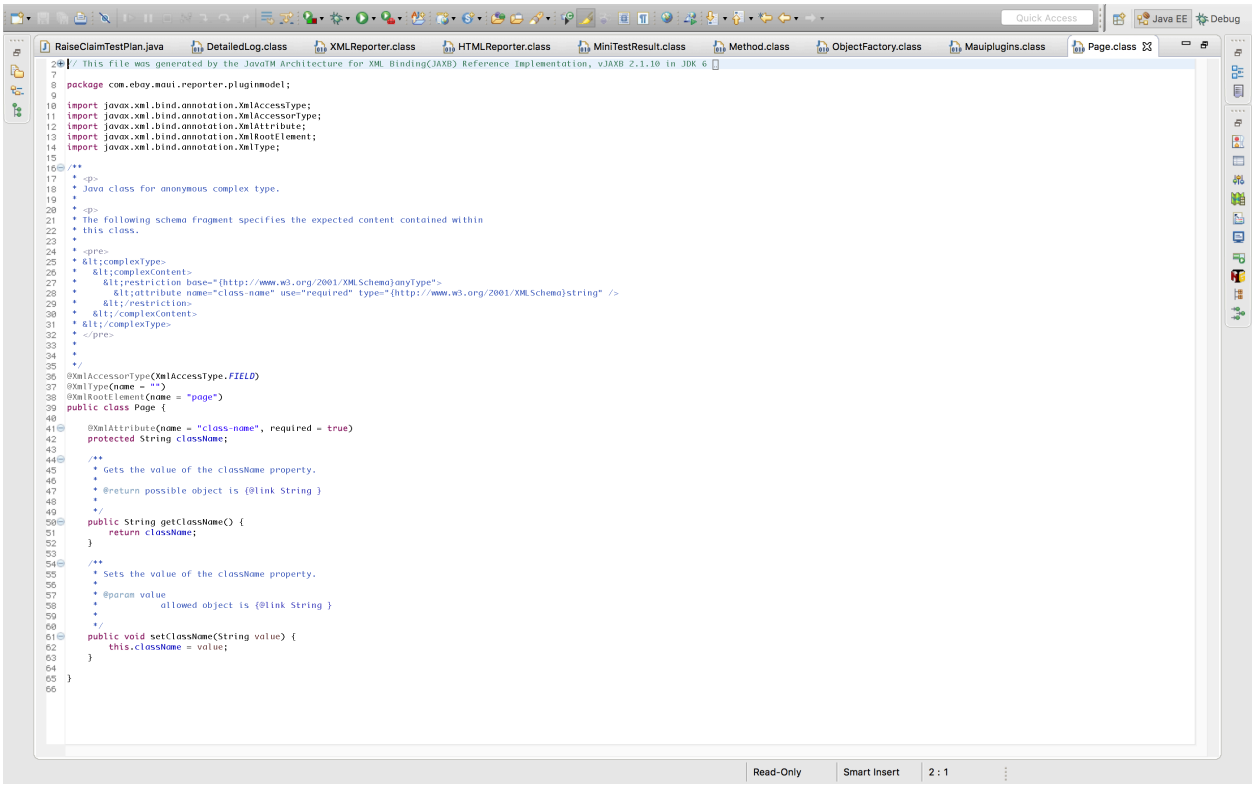


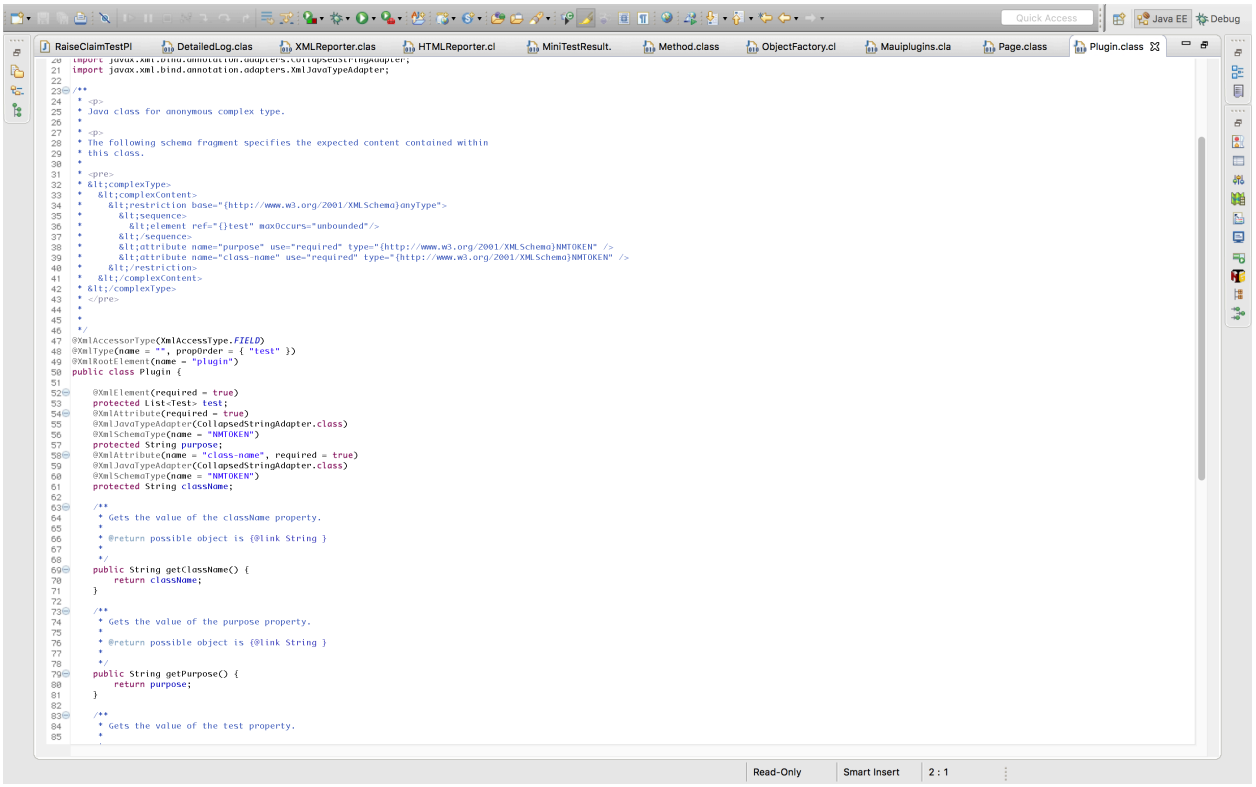


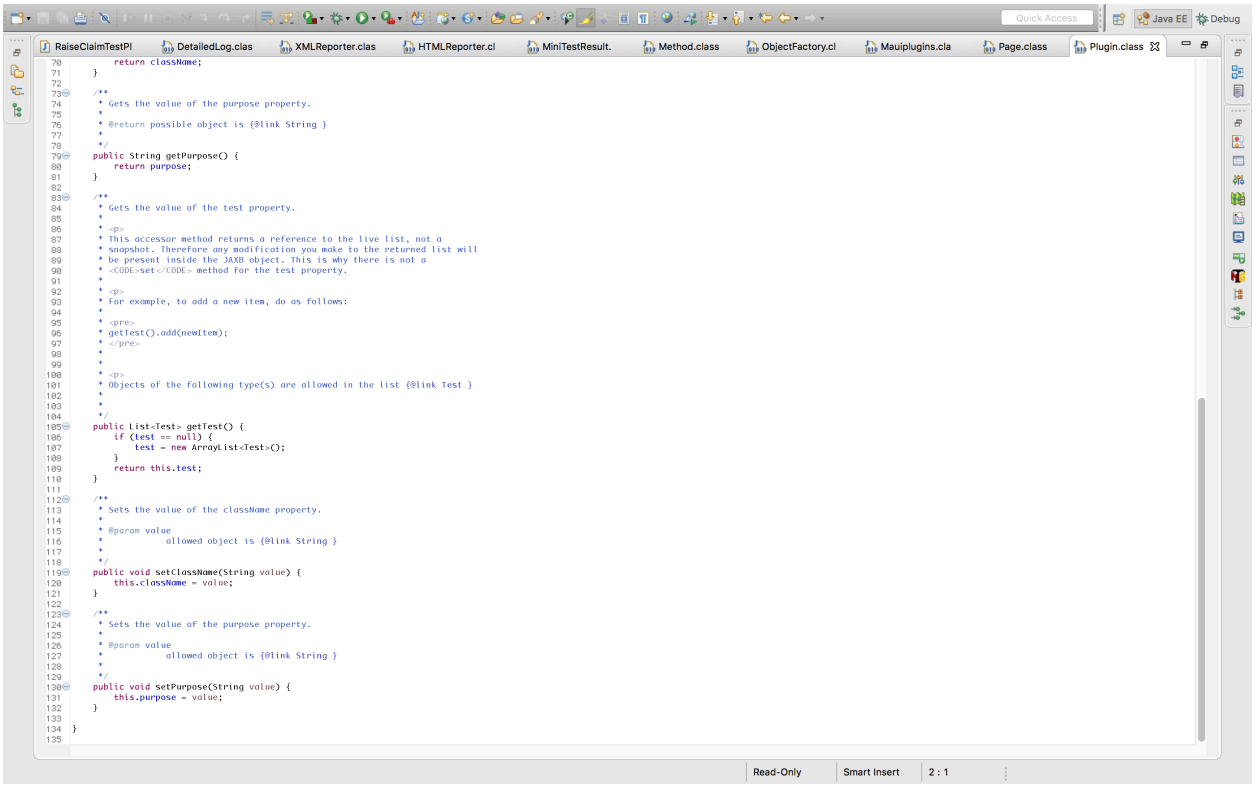

```
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "", propOrder = { "page" })
@XmlRootElement(name = "method")
public class Method {

    protected List<Page> page;
    @XmlAttribute(required = true)
    protected String name;

    /**
     * Gets the value of the name property.
     *
     * @return possible object is {@link String }
     *
     */
    public String getName() {
        return name;
    }

    /**
     * Gets the value of the page property.
     *
     * <p>
     * This accessor method returns a reference to the live list, not a
     * snapshot. Therefore any modification you make to the returned list will
     * be present inside the JAXB object. This is why there is not a
     * <CODE>set</CODE> method for the page property.
     *
     * <p>
     * For example, to add a new item, do as follows:
     *
     * <pre>
     * getPage().add(newItem);
     * </pre>
     *
     *
     * <p>
     * Objects of the following type(s) are allowed in the list {@link Page }
     *
     *
     */
    public List<Page> getPage() {
        if (page == null) {
            page = new ArrayList<Page>();
        }
        return this.page;
    }

    /**
     * Sets the value of the name property.
     *
     * @param value
     *            allowed object is {@link String }
     *
     */
    public void setName(String value) {
        this.name = value;
    }

}
```



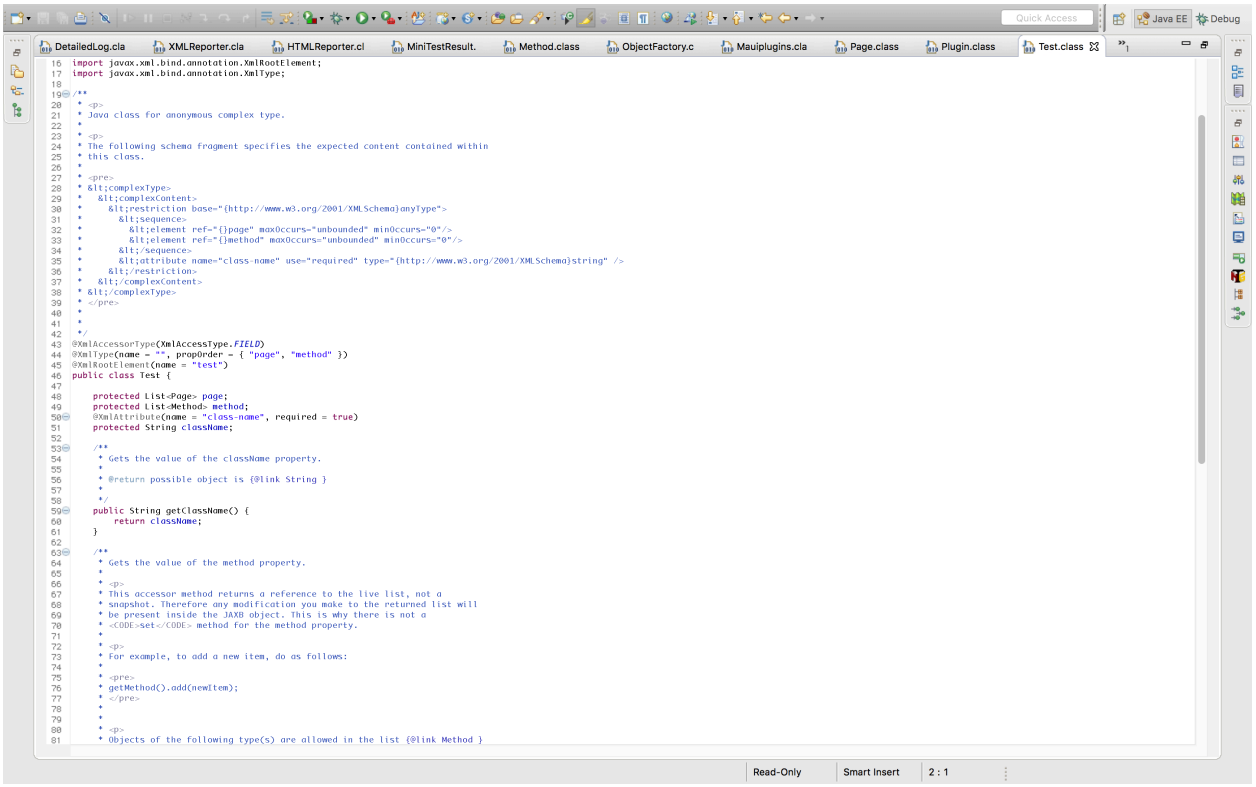

```
package com.ebay.maui.reporter.pluginmodel;

import javax.xml.bind.annotation.XmlRegistry;

/**
 * This object contains factory methods for each Java content interface and Java
 * element interface generated in the generated package.
 * <p>
 * An ObjectFactory allows you to programatically construct new instances of the
 * Java representation for XML content. The Java representation of XML content
 * can consist of schema derived interfaces and classes representing the binding
 * of schema type definitions, element declarations and model groups. Factory
 * methods for each of these are provided in this class.
 *
 */
@XmlRegistry
public class ObjectFactory {

    /**
     * Create a new ObjectFactory that can be used to create new instances of
     * schema derived classes for package: generated
     *
     */
    public ObjectFactory() {
    }

    /**
     * Create an instance of {@link Mauiplugins }
     *
     */
    public Mauiplugins createMauiplugins() {
        return new Mauiplugins();
    }

    /**
     * Create an instance of {@link Method }
     *
     */
    public Method createMethod() {
        return new Method();
    }

    /**
     * Create an instance of {@link Page }
     *
     */
    public Page createPage() {
        return new Page();
    }

    /**
     * Create an instance of {@link Plugin }
     *
     */
    public Plugin createPlugin() {
        return new Plugin();
    }

    /**
     * Create an instance of {@link Test }
     *
     */
    public Test createTest() {
        return new Test();
    }
```


RaiseClaimTestPlan.java
DetailedLog.class
XMLReporter.class
HTMLReporter.class
MiniTestResult.class
Method.class
ObjectFactory.class
Mauiplugins.class
Page.class
Quick Access
Java EE
Debug
// This file was generated by the JavaTM Architecture for XML Binding(JAXB) Reference Implementation, vJAXB 2.1.10 in JDK 6
package com.ebay.maui.reporter.pluginmodel;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlType;
/**
* <p>
* Java class for anonymous complex type.
*
* <p>
* The following schema fragment specifies the expected content contained within
* this class.
*
* <pre>
* <complexType>
*   <complexContent>
*   <restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
*   <attribute name="class-name" use="required" type="{http://www.w3.org/2001/XMLSchema}string" />
*   </restriction>
*   </complexContent>
* </complexType>
* </pre>
*
*
*/
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "")
@XmlRootElement(name = "page")
public class Page {
@XmlAttribute(name = "class-name", required = true)
protected String className;
/**
* Gets the value of the className property.
*
* @return possible object is {@link String }
*
*/
public String getClassName() {
return className;
}
/**
* Sets the value of the className property.
*
* @param value
*   allowed object is {@link String }
*
*/
public void setClassName(String value) {
this.className = value;
}
}
Read-Only
Smart Insert
2 : 1

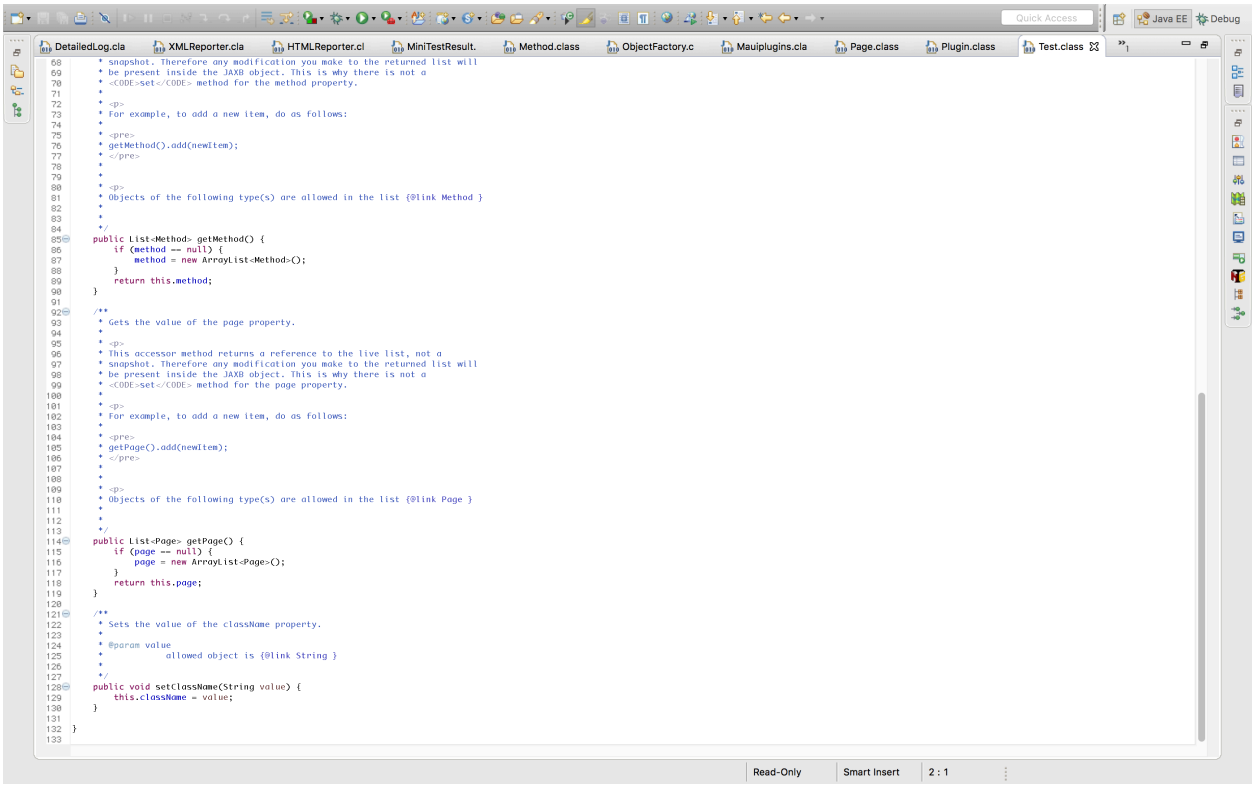

RaiseClaimTestPl
DetailedLog.clas
XMLReporter.clas
HTMLReporter.cl
MiniTestResult.
Method.class
ObjectFactory.cl
Mauiplugins.cla
Page.class
Plugin.class
Quick Access
Java EE
Debug
import javax.xml.bind.annotation.adapters.CollapsedStringAdapter;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;
/**
* <p>
* Java class for anonymous complex type.
*
* <p>
* The following schema fragment specifies the expected content contained within
* this class.
*
* <pre>
* <complexType>
*   <complexContent>
*   <restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
*   <sequence>
*   <element ref="{}test" maxOccurs="unbounded"/>
*   </sequence>
*   <attribute name="purpose" use="required" type="{http://www.w3.org/2001/XMLSchema}NMTOKEN" />
*   <attribute name="class-name" use="required" type="{http://www.w3.org/2001/XMLSchema}NMTOKEN" />
*   </restriction>
*   </complexContent>
* </complexType>
* </pre>
*
*
*/
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "", propOrder = { "test" })
@XmlRootElement(name = "plugin")
public class Plugin {
@XmlElement(required = true)
protected List<Test> test;
@XmlAttribute(required = true)
@XmlJavaTypeAdapter(CollapsedStringAdapter.class)
@XmlSchemaType(name = "NMTOKEN")
protected String purpose;
@XmlAttribute(name = "class-name", required = true)
@XmlJavaTypeAdapter(CollapsedStringAdapter.class)
@XmlSchemaType(name = "NMTOKEN")
protected String className;
/**
* Gets the value of the className property.
*
* @return possible object is {@link String }
*
*/
public String getClassName() {
return className;
}
/**
* Gets the value of the purpose property.
*
* @return possible object is {@link String }
*
*/
public String getPurpose() {
return purpose;
}
/**
* Gets the value of the test property.
*
Read-Only
Smart Insert
2 : 1




```
        return className;
    }

    /**
     * Gets the value of the purpose property.
     *
     * @return possible object is {@link String }
     *
     */
    public String getPurpose() {
        return purpose;
    }

    /**
     * Gets the value of the test property.
     *
     * <p>
     * This accessor method returns a reference to the live list, not a
     * snapshot. Therefore any modification you make to the returned list will
     * be present inside the JAXB object. This is why there is not a
     * <CODE>set</CODE> method for the test property.
     *
     * <p>
     * For example, to add a new item, do as follows:
     *
     * <pre>
     * getTest().add(newItem);
     * </pre>
     *
     *
     * <p>
     * Objects of the following type(s) are allowed in the list {@link Test }
     *
     *
     */
    public List<Test> getTest() {
        if (test == null) {
            test = new ArrayList<Test>();
        }
        return this.test;
    }

    /**
     * Sets the value of the className property.
     *
     * @param value
     *            allowed object is {@link String }
     *
     */
    public void setClassName(String value) {
        this.className = value;
    }

    /**
     * Sets the value of the purpose property.
     *
     * @param value
     *            allowed object is {@link String }
     *
     */
    public void setPurpose(String value) {
        this.purpose = value;
    }

}
```




```
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlType;

/**
 * <p>
 * Java class for anonymous complex type.
 *
 * <p>
 * The following schema fragment specifies the expected content contained within
 * this class.
 *
 * <pre>
 * &lt;complexType>
 *   &lt;complexContent>
 *     &lt;restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
 *       &lt;sequence>
 *         &lt;element ref="{}page" maxOccurs="unbounded" minOccurs="0"/>
 *         &lt;element ref="{}method" maxOccurs="unbounded" minOccurs="0"/>
 *       &lt;/sequence>
 *       &lt;attribute name="class-name" use="required" type="{http://www.w3.org/2001/XMLSchema}string" />
 *     &lt;/restriction>
 *   &lt;/complexContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "", propOrder = { "page", "method" })
@XmlRootElement(name = "test")
public class Test {

    protected List<Page> page;
    protected List<Method> method;
    @XmlAttribute(name = "class-name", required = true)
    protected String className;

    /**
     * Gets the value of the className property.
     *
     * @return possible object is {@link String }
     *
     */
    public String getClassName() {
        return className;
    }

    /**
     * Gets the value of the method property.
     *
     * <p>
     * This accessor method returns a reference to the live list, not a
     * snapshot. Therefore any modification you make to the returned list will
     * be present inside the JAXB object. This is why there is not a
     * <CODE>set</CODE> method for the method property.
     *
     * <p>
     * For example, to add a new item, do as follows:
     *
     * <pre>
     * getMethod().add(newItem);
     * </pre>
     *
     *
     * <p>
     * Objects of the following type(s) are allowed in the list {@link Method }
```


Quick Access
Java EE
Debug
DetailedLog.cla
XMLReporter.cla
HTMLReporter.cl
MiniTestResult.
Method.class
ObjectFactory.c
Mauiplugins.cla
Page.class
Plugin.class
Test.class
68 * snapshot. Therefore any modification you make to the returned list will
69 * be present inside the JAXB object. This is why there is not a
70 * <CODE>set</CODE> method for the method property.
71 *
72 * <p>
73 * For example, to add a new item, do as follows:
74 *
75 * <pre>
76 * getMethod().add(newItem);
77 * </pre>
78 *
79 *
80 * <p>
81 * Objects of the following type(s) are allowed in the list {@link Method }
82 *
83 *
84 */
85 public List<Method> getMethod() {
86 if (method == null) {
87 method = new ArrayList<Method>();
88 }
89 return this.method;
90 }
91
92 /**
93 * Gets the value of the page property.
94 *
95 * <p>
96 * This accessor method returns a reference to the live list, not a
97 * snapshot. Therefore any modification you make to the returned list will
98 * be present inside the JAXB object. This is why there is not a
99 * <CODE>set</CODE> method for the page property.
100 *
101 * <p>
102 * For example, to add a new item, do as follows:
103 *
104 * <pre>
105 * getPage().add(newItem);
106 * </pre>
107 *
108 *
109 * <p>
110 * Objects of the following type(s) are allowed in the list {@link Page }
111 *
112 *
113 */
114 public List<Page> getPage() {
115 if (page == null) {
116 page = new ArrayList<Page>();
117 }
118 return this.page;
119 }
120
121 /**
122 * Sets the value of the className property.
123 *
124 * @param value
125 * allowed object is {@link String }
126 *
127 */
128 public void setClassName(String value) {
129 this.className = value;
130 }
131
132 }
133
Read-Only
Smart Insert
2 : 1