W3C mobileOK Basic Tests 1.0 Checker (Beta Release)
User Manual

Version of this document: 0.8 (Draft)

Date of Release: 2008-02-04

Editors: Ignacio Marín, Abel Rionda (Fundación CTIC)

Contents

Illustration Index

1 Introduction

1.1 Purpose and Audience

This document is meant to serve as a User Manual for the W3C mobileOK Basic Tests 1.0 Checker (Beta Release). Current version of this User Manual is a draft, so the editors intend to write new versions correcting typos and errors and inserting new and more complete and accurate information.

The version of the mobileOK Checker subject to comments in this document is the second public release of the checker (a Beta version after the first Alpha version). A final version of this software, of this User Manual and of the Developer Manual are expected later in 2008.

The audience of this document is the general public being aware of the need to check conformance of web resources (and also of the way in which they are delivered) against W3C mobileOK Basic Tests 1.0 and the need to implement conformance checking tools based on those tests. The W3C mobileOK Checker is a software tool, created by means of collaboration among different organizations (see Acknowledgements). The intention of such Task Force is to create a Reference Implementation so the W3C mobileOK Basic Tests 1.0 document can become a W3C Recommendation. This checker allows users to pass mobileOK Tests on a given web resource.

W3C mobileOK Basic tests implement the way in which some of the Best Practices in the W3C Mobile Web Best Practices (Basic Guidelines) 1.0 document can be checked against web resources. The term “some” refers to all those Best Practices that can be verified by a machine (i.e., which can be described as an algorythm). Those Best Practices which can not be described as an algorythm (needing the assessment of a human expert) are covered by the future W3C Mobile OK Pro Tests document, still in development.

This document is accompanied by the W3C mobileOK Basic Tests 1.0 Checker Developer Manual. While this User Manual tries to explain the current version of the W3C mobileOK Checker software tool as a black box and to cover all the subjects of interest to anyone trying to use the tool as-is, the Developer Manual describes the internal architecture, design and implementation details of this tool as it would be interesting for developers willing to modify, extend or enhance the functionality of the tool or even willing to integrate it with their own software or services.

The W3C mobileOK Basic Tests 1.0 Checker, its User Manual and Developer Manual, the W3C mobileOK Tests 1.0 and the W3C Mobile Web Best Practices (Basic Guidelines) 1.0 are efforts taking place under the framework of the W3C Mobile Web Best Practices working group (part of the W3C Mobile Web Initiative).

1.2 Copyright issues

The W3C mobileOK Basic Tests 1.0 Checker software tool and related documents (User Manual and Developer Manual) are released under the W3C License, in its current version at the very moment of the release of this document.

Several external Java libraries and software tools are used by the W3C mobileOK Basic Tests 1.0 Checker. Their use is subject to different licenses and copyright clauses. The binary distribution for the W3C mobileOK Basic Tests 1.0 Checker and its source code in W3C's CVS repository contains the licenses and copyright claims to which each of them are subjected.

Eclipse, the IDE used as example in this documentation, is released under the Eclipse Public License v1.0 in the version used for the screenshots taken.

1.3 Structure

This document is divided into several sections, which are now briefly described:

- Introduction: Current section, where the general purpose of this document and of the target software tool (W3C mobileOK Checker 1.0a) is commented, including related W3C activities and documents.
- Download and installation: This section describes how to download the Java executable binary file (in JAR format) for the W3C mobileOK Checker, how to optionally download the source files for the software and the library dependencies for the Checker.
- Description of the mobileOK Checker: This section explains the prerequisites needed for the software to work, how to execute the W3C mobileOK Checker JAR file from command line or from Eclipse IDE and the format of the output documents that the Checker generates.
- Examples: This section includes several examples meant to illustrate how to execute the Checker and how to interpret the documents produced.
- Known issues: Issues regarding known bugs and future improvements of the Checker, and missing information (that will be included in future versions) in this User Manual.
- References: Documents interesting for readers, which might help them to understand this document, the W3C mobileOK Checker 1.0a software tool and all related work of the W3C Mobile Web Best Practices working group.
- Acknowledgements: Credits to the people who helped developing this User Manual and the W3C mobileOK Checker 1.0a software tool.

2 Download and installation

2.1 Binary archive download.

The Java binary executable JAR file is downloadable at:

http://dev.w3.org/cvsweb/2007/mobileok-ref/

The existing version of the W3C mobileOK Tests 1.0 Checker when releasing this document is available at:

http://dev.w3.org/cvsweb/~checkout~/2007/mobileok-ref/mobileOK-Basic-RI-1.0-deploy.jar?rev=1.8&content-type=text/plain&only_with_tag=HEAD

2.2 Source code download (from CVS repository)

The source files corresponding to the JAR file mentioned in 2.1 are available in W3C's CVS repository at :pserver:anonymous@dev.w3.org/public/sources/2007/mobileok-ref (password for anonymous user is *anonymous*). If the previous URL does not work, the specific data for CVS access are commented below:

- Host: dev.w3.org
- Repository path: /sources/public
- User: anonymous
- Password: anonymous
- Connection type: pserver
- Port: default pserver port (2401)
- Specified module name: 2007/mobileok-ref.
- Available branches: Only HEAD. No other branches expected in the future.

Please take into account that, in order to be able to deal with the source code as a developer, you will need to read the W3C mobileOK Checker Developer Manual document (still in development).

3 Description of the mobileOK Checker

This section is the core of this document and tries to help users to be able to execute the Checker and perform a mobileOK Basic test (or a set of tests, or even all mobileOK tests) on a given web resource specified by a URI.

For a more in-depth analysis of the source code, the W3C mobileOK Checker 1.0a Developer Manual will be available when next version of this document is published.

3.1 Execution

In addition to command-line execution, an example using Eclipse IDE is shown. Before using any IDE, it is important to read the subsection 3.1.2 (Command line usage) of this document explaining how to execute the W3C mobileOK Checker by using the command line.

3.1.1 Prerequisites

This library was made using Java 1.5. In order to avoid compatibility problems, Java 1.5 or a higher version should be installed in your system and made available to the Checker JAR file.

3.1.2 Command line usage

In order to execute the library from command line, the format to launch the Checker is as shown below:

```
java -jar  mobileOK-Basic-RI-1.0-deploy.jar URL [tests] [options]
```

where:

- URL: The web resource to check against mobile OK Basic Tests. For example http://www.example.org.
- [tests] : An optional argument to check the web resource against a list of mobileOK Basic Test. By default, if this argument is not present, all mobileOK Basic Tests are checked. The name of each test is the same one shown in the mobile OK Basic Tests 1.0 document (whose naming convention for tests is based on the naming in the Mobile Web Best Practices 1.0 Basic Guidelines document).

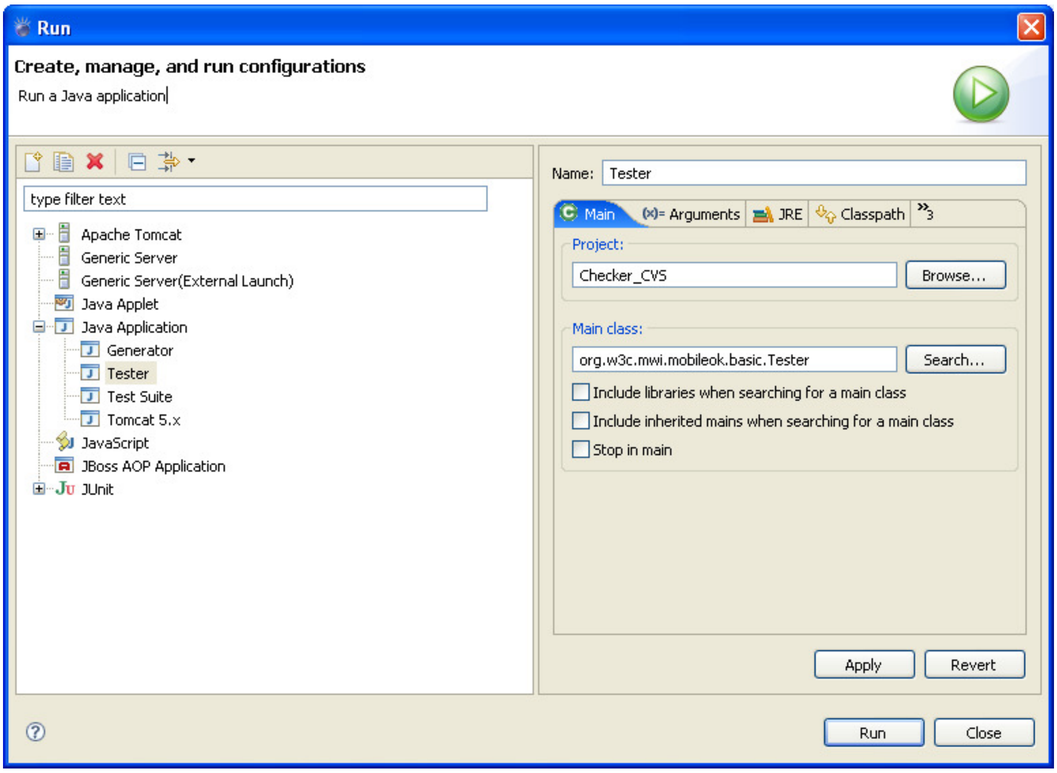
  For example, this would be the way to check the example site against the tests MEASURES and CACHING:

  ```
  java -jar  mobileOK-Basic-RI-1.0-deploy.jar http://www.example.org MEASURES CACHING
  ```
- [options]: Some options related to the output can be set. By default the standard output stream is used.
  - -p *host*:*port* : DNS name or IP and port number of an HTTP intermediate proxy.
  - -r *filename* : Results are saved in the specified file.
  - -m *filename* : The (mOKI, mobileOK Intermediate) document is saved in the specified file.

3.1.3 Eclipse IDE usage

After source code is downloaded from W3C's CVS repository and a new Eclipse Java project is created, it is possible to define a run configuration for the library.

- Launch a new run configuration (Run option in Run menu).
- Introduce as main class `org.mwi.mobileok.basic.Tester`



Run

Create, manage, and run configurations

Run a Java application

Name: Tester

Project: Checker_CVS

Main class: org.w3c.mwi.mobileok.basic.Tester

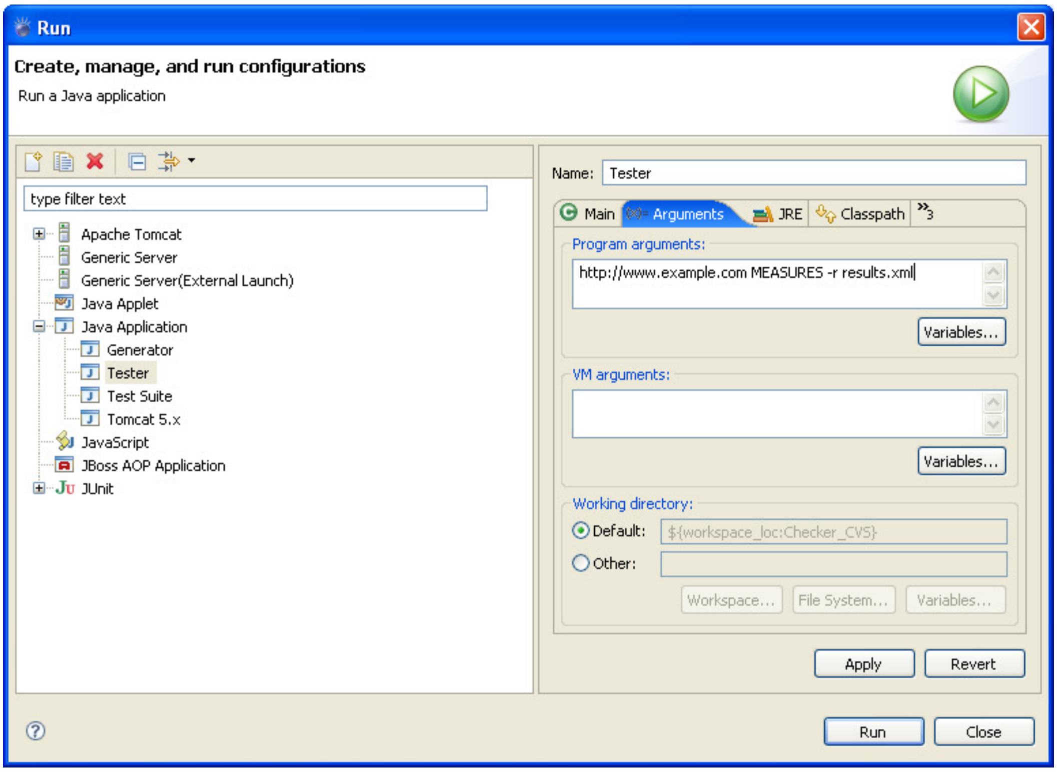
□ Include libraries when searching for a main class

□ Include inherited mains when searching for a main class

□ Stop in main

Illustration 1: Setting a configuration to execute the Checker in Eclipse IDE

- Introduce, as program arguments, the URI of the web resource to be checked and the rest of the options (as shown in 3.1.2). For example in the following screenshot, MEASURES test is tested against http://www.example.com. Furthemore, the results are saved in a file called **results.xml**.



Run

Create, manage, and run configurations

Run a Java application

Name: Tester

Program arguments:

```
http://www.example.com MEASURES -r results.xml
```

VM arguments:

Working directory:

● Default: ${workspace_loc:Checker_CVS}

○ Other:

Illustration 2: Passing parameters to the Checker in Eclipse IDE

Finally, the *Run* button must be activated to launch the new configuration.

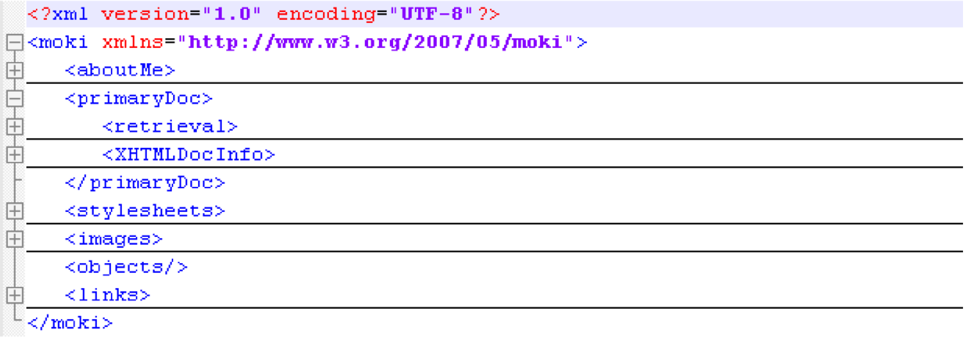
3.2 Output of the mobileOK Checker

When tests are performed by the W3C mobileOK Checker on a web resource, two documents are generated. After retrieving the web resource to be examined, a preprocessing stage is performed by the Checker and an intermediate document is generated. This document is then processed in order to generate the final results document, where the output of the tests is stated. The internal architecture for the W3C mobileOK Checker 1.0a will be detailed in the Developer Manual.

3.2.1 Intermediate (mOKI) document

The format for the intermediate document generated by the Checker (named as mOKI format by the authors of the library, which stands for mobileOK Intermediate format) is explained in this subsection. This document tries to describe the context in which the web resource to be checked has been retrieved. This context is basically the HTTP dialog taking place between the Checker and the server in order to retrieve the web resource, including HTTP requests, HTTP responses, the content of the web resource as delivered by the server in one of the responses and, of course, all the HTTP headers included in requests/responses -which are really important for some tests like CACHING-.

After, the XML declaration, a *moki* element is the root element of this XML document dialect. Inside the root element, a sequence of four elements appears in a mOKI document: *primaryDoc*, stylesheets, *images* and *links*. Note that there is an XML Schema for the mOKI document format available so anyone can validate mOKI documents generated by any software tool, although current version of the Checker generates these documents without referencing the Schema.

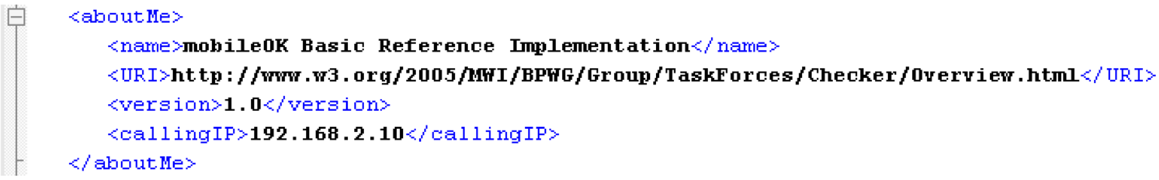


```
<?xml version="1.0" encoding="UTF-8"?>
<moki xmlns="http://www.w3.org/2007/05/moki">
   <aboutMe>
   <primaryDoc>
      <retrieval>
      <XHTMLDocInfo>
   </primaryDoc>
   <stylesheets>
   <images>
   <objects/>
   <links>
</moki>
```

Illustration 3: Main elements in mOKI documents

3.2.1.1 aboutMe element

The first element in a mOKI document serves as a container for information related to the tool that generated the mOKI document. It contains the *name* of the software tool that created the document, a *URI* where to find information about that tool, the version of the tool used when generating the document and the IP of the machine that executed the software (*callingIP*). The example in Illustration 4 shows the information filled by the W3C mobileOK Checker 1.0a inside the *aboutMe* element.

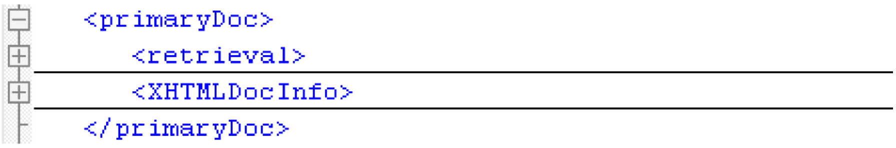


```
<aboutMe>
   <name>mobileOK Basic Reference Implementation</name>
   <URI>http://www.w3.org/2005/MWI/BPWG/Group/TaskForces/Checker/Overview.html</URI>
   <version>1.0</version>
   <callingIP>192.168.2.10</callingIP>
</aboutMe>
```

Illustration 4: aboutMe element

3.2.1.2 primaryDoc element

This element intends to represent the primary document that the Checker needs to retrieve in order to perform the test. This representation includes retrieval information (element *retrieval*, child of *primaryDoc*) and information related to the XHTML document retrieved -when the URI on which to pass the test references an XHTML document- (*XHTMLDocInfo* child element).



```
<primaryDoc>
   <retrieval>
   <XHTMLDocInfo>
</primaryDoc>
```

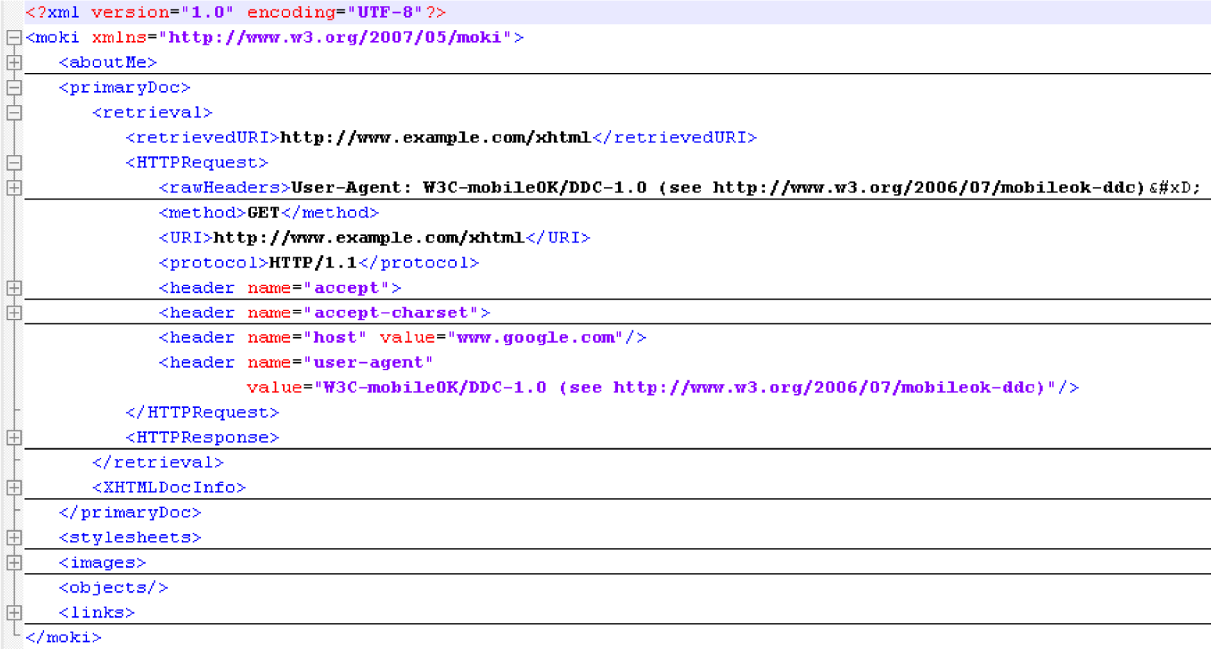
Illustration 5: Children elements of primaryDoc

3.2.1.2.1 *retrieval* element

This element includes the URI of the retrieved web resource (child element *retrievedURI*) and a sequence of pairs of *HTTPRequest* and *HTTPResponse* elements, which represents the HTTP dialog that took place in order to obtain the web resource being target of the validation against mobileOK Basic Tests 1.0.

3.2.1.2.1.1 HTTPRequest element

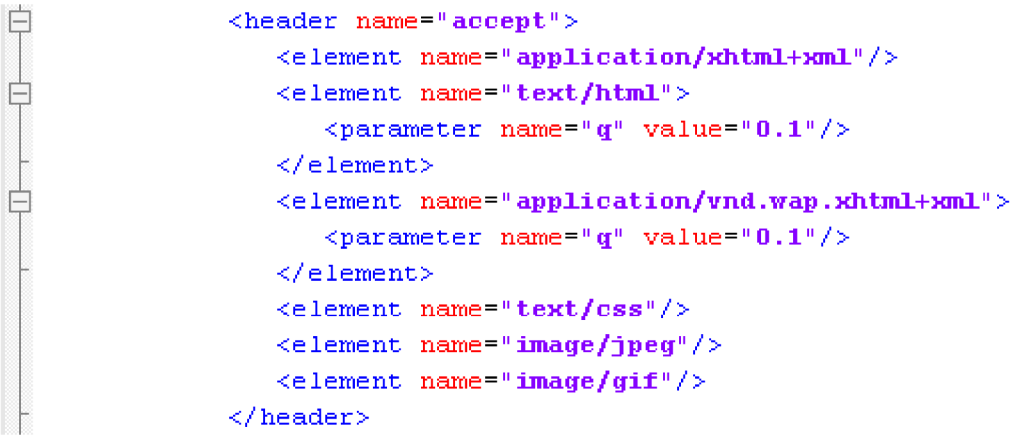
An *HTTPRequest* element holds the raw headers of a request, the HTTP method used for the request (GET, POST, HEAD), the URI requested, the HTTP protocol version, and a list of *header* elements represting the HTTP headers included in the request. Instead of describing each child element of *HTTPRequest*, a visual example is shown below in order to clarify the contents of an *HTTPRequest* element.



```
<?xml version="1.0" encoding="UTF-8"?>
<moki xmlns="http://www.w3.org/2007/05/moki">
   <aboutMe>
   <primaryDoc>
      <retrieval>
         <retrievedURI>http://www.example.com/xhtml</retrievedURI>
         <HTTPRequest>
            <rawHeaders>User-Agent: W3C-mobileOK/DDC-1.0 (see http://www.w3.org/2006/07/mobileok-ddc)&#xD;
            <method>GET</method>
            <URI>http://www.example.com/xhtml</URI>
            <protocol>HTTP/1.1</protocol>
            <header name="accept">
            <header name="accept-charset">
            <header name="host" value="www.google.com"/>
            <header name="user-agent"
                    value="W3C-mobileOK/DDC-1.0 (see http://www.w3.org/2006/07/mobileok-ddc)"/>
         </HTTPRequest>
         <HTTPResponse>
      </retrieval>
      <XHTMLDocInfo>
   </primaryDoc>
   <stylesheets>
   <images>
   <objects/>
   <links>
</moki>
```

Illustration 6: Information in an HTTPRequest element

Note that, although a header element keeps the information of an HTTP Request header by means of a *name* attribute and a *value* attribute in Illustration 6, this is not true in all cases, as shown in Illustration 7.



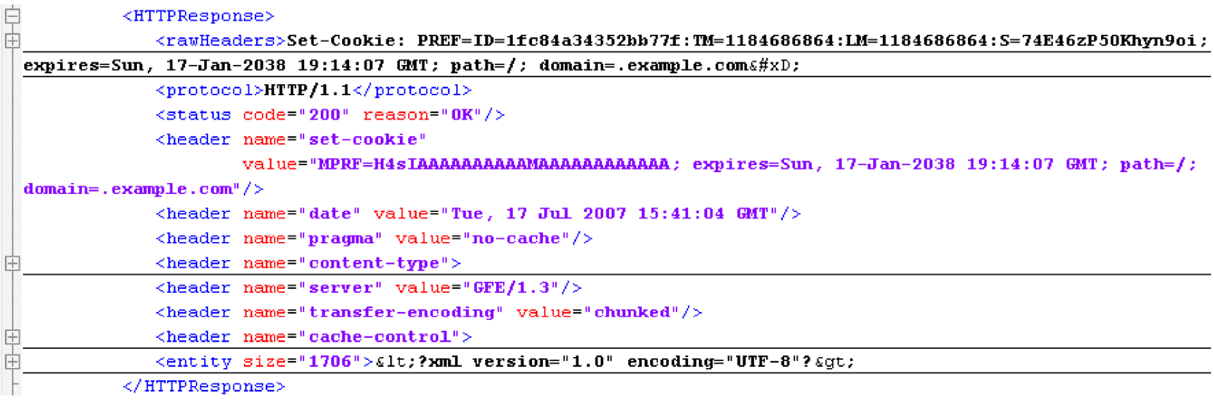
```
<header name="accept">
   <element name="application/xhtml+xml"/>
   <element name="text/html">
      <parameter name="q" value="0.1"/>
   </element>
   <element name="application/vnd.wap.xhtml+xml">
      <parameter name="q" value="0.1"/>
   </element>
   <element name="text/css"/>
   <element name="image/jpeg"/>
   <element name="image/gif"/>
</header>
```

Illustration 7: Example of representation of elements in an HTTP Request header

Illustration 7 shows how the use of *name/value* is not sufficient to express the information contained in HTTP Request headers as the *Accept* header, so a different model with *element* children and *parameter* children to *element* is used. This way, headers including a set of values and a qualifier for each value can be easily expressed.

3.2.1.2.1.2 HTTPResponse element

This element is analogous to the *HTTPRequest* element in mOKI. It includes the raw version of the information contained in the HTTP Response, the *protocol* version in use, the *status* code for the response (and its corresponding reason string), a sequence of *header* elements (analogous in purpose and format to the element of the same name which is children of *HTTPRequest*) and an *entity* element including the body of the HTTP Response which replaces different characters ('<','>',...) by their corresponding XML entities.



```
        <HTTPResponse>
           <rawHeaders>Set-Cookie: PREF=ID=1fc84a34352bb77f:TM=1184686864:LM=1184686864:S=74E46zP50Khyn9oi;
expires=Sun, 17-Jan-2038 19:14:07 GMT; path=/; domain=.example.com&#xD;
           <protocol>HTTP/1.1</protocol>
           <status code="200" reason="OK"/>
           <header name="set-cookie"
                   value="MPRF=H4sIAAAAAAAAAAMAAAAAAAAAAAA; expires=Sun, 17-Jan-2038 19:14:07 GMT; path=/;
domain=.example.com"/>
           <header name="date" value="Tue, 17 Jul 2007 15:41:04 GMT"/>
           <header name="pragma" value="no-cache"/>
           <header name="content-type">
           <header name="server" value="GFE/1.3"/>
           <header name="transfer-encoding" value="chunked"/>
           <header name="cache-control">
           <entity size="1706">&lt;?xml version="1.0" encoding="UTF-8"?&gt;
        </HTTPResponse>
```

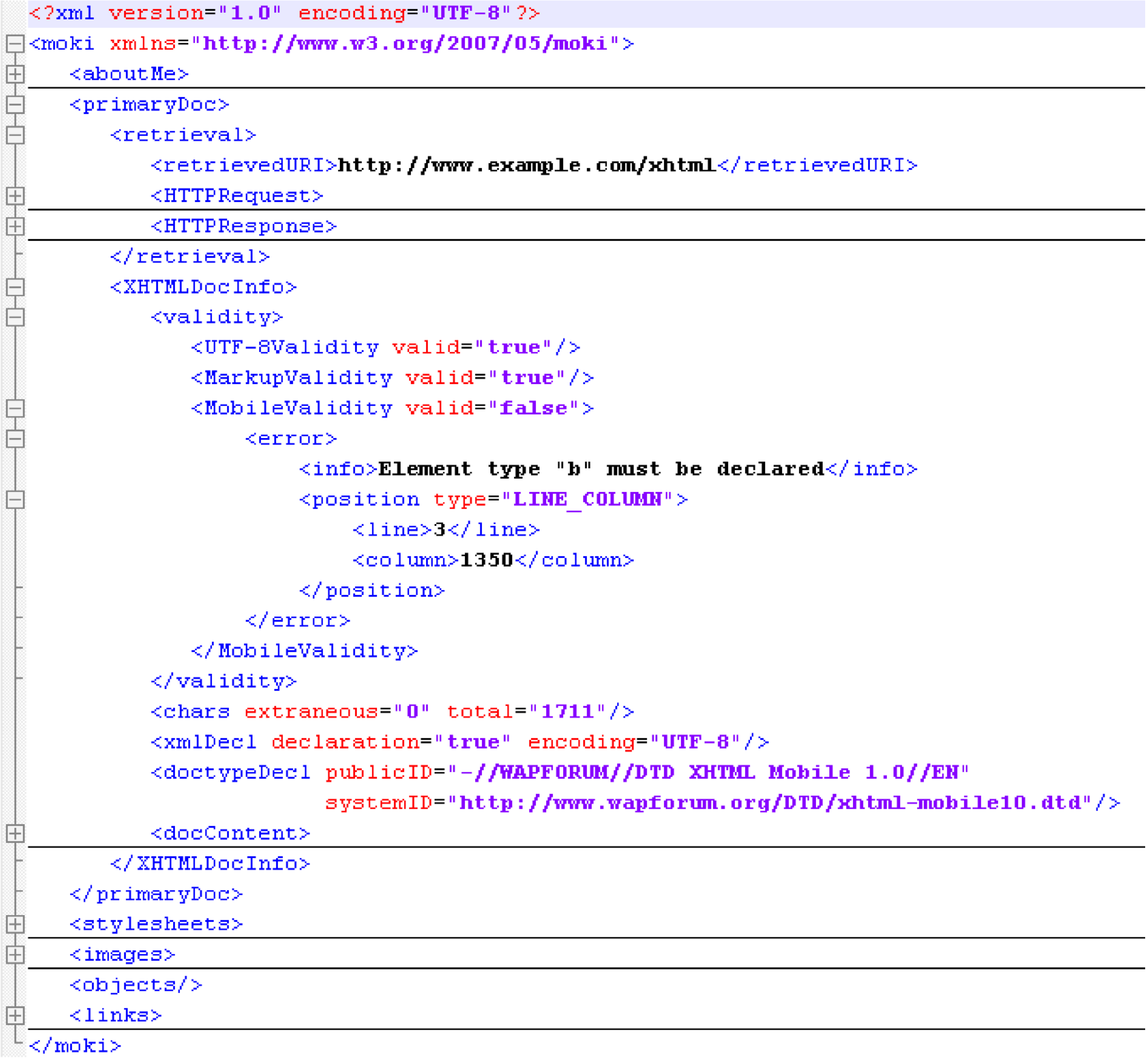
Illustration 8: Example of HTTPResponse *element*

3.2.1.2.2 *XHTMLDocInfo* element

This element includes information about the XHTML document targetted by the Checker for its validation against W3C mobileOK Tests 1.0. As shown in Illustration 10, it contains information about the *validation* of the document:

- **UTF-8 validity**: true, if all characters in the document are expressed in UTF-8, or false (if not).
- **Markup validity**: true, if the document validates against its DTD; false, if not.
- **Mobile validity**: true, if the document validates against XHTML Basic or XHTML Mobile Profile DTDs; false, if not.

It also provides the number of extraneous characters (extra spacing characters) and the total number of characters, information about the XML declaration and the DOCTYPE declaration and, finally, a *docContent* element containing the XHTML document itself.



```
<?xml version="1.0" encoding="UTF-8"?>
<moki xmlns="http://www.w3.org/2007/05/moki">
   <aboutMe>
   <primaryDoc>
      <retrieval>
         <retrievedURI>http://www.example.com/xhtml</retrievedURI>
         <HTTPRequest>
         <HTTPResponse>
      </retrieval>
      <XHTMLDocInfo>
         <validity>
            <UTF-8Validity valid="true"/>
            <MarkupValidity valid="true"/>
            <MobileValidity valid="false">
                <error>
                    <info>Element type "b" must be declared</info>
                    <position type="LINE_COLUMN">
                        <line>3</line>
                        <column>1350</column>
                    </position>
                </error>
            </MobileValidity>
         </validity>
         <chars extraneous="0" total="1711"/>
         <xmlDecl declaration="true" encoding="UTF-8"/>
         <doctypeDecl publicID="-//WAPFORUM//DTD XHTML Mobile 1.0//EN"
                      systemID="http://www.wapforum.org/DTD/xhtml-mobile10.dtd"/>
         <docContent>
      </XHTMLDocInfo>
   </primaryDoc>
   <stylesheets>
   <images>
   <objects/>
   <links>
</moki>
```

Illustration 9: Example of XHTMLDocInfo *element*

The example above also shows how the mOKI format expresses errors when validity against given criteria is not accomplished. Errors are expressed as a sequence of *error* elements (children of the *UTF-8Validity*, *MarkUpValidity* or *MobileValidity* elements) and include the reason of the error (*info* element) and its *position* in the document (and the format in which position is described).

3.2.1.3 *stylesheets* element

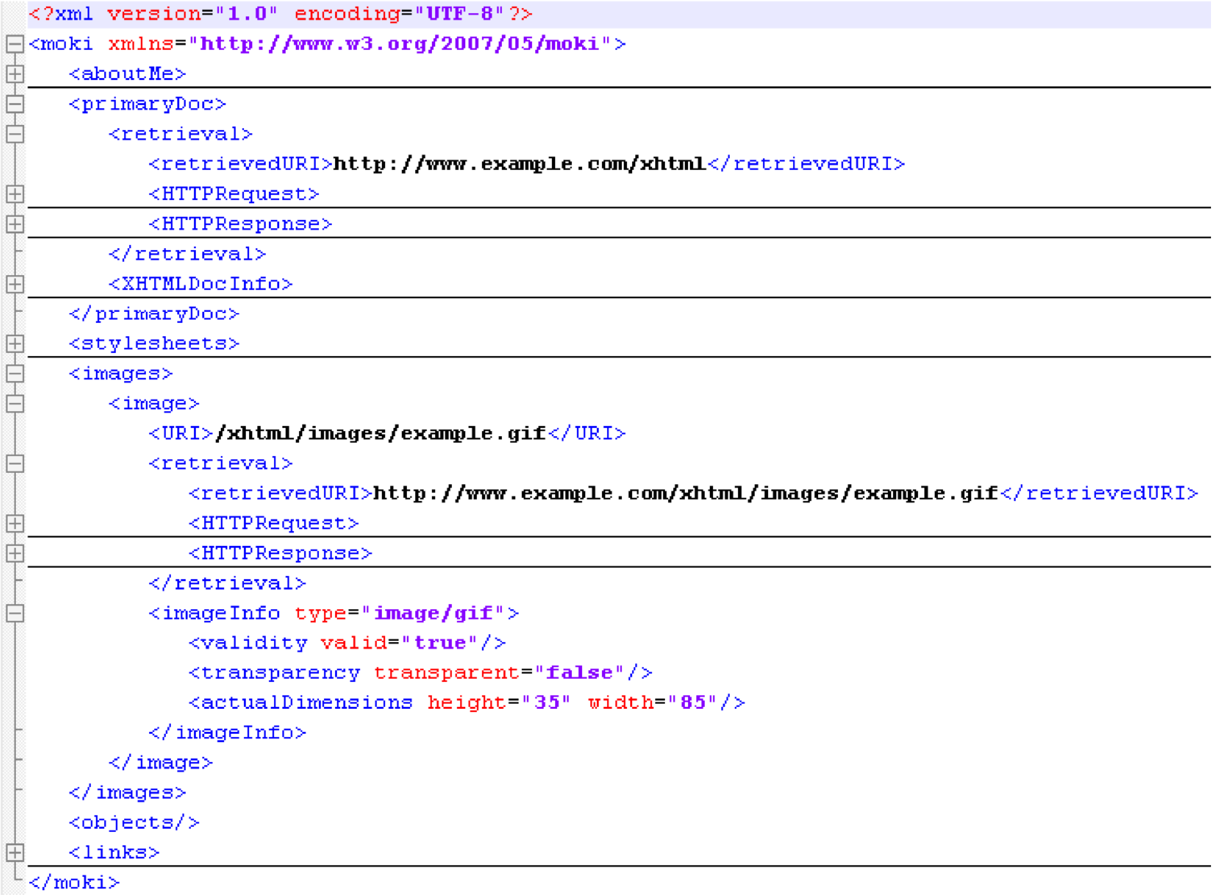
This element includes CSS information relative to the XHTML document target of the mobileOK Basic 1.0 tests. It consists of a sequence of *stylesheet* elements, each of them containing a URI element (representing the URI for a external stylesheet, or the URI of the document itself if the CSS is inline or embedded), a CSSValidity element indicating if the style information is CSS-valid and a styles element.

The styles element is a sequence of style elements: each style element represents a CSS style sentence in XML format.There is a project in the CVS of W3C dealing with the Representation of CSS in XML used in the Checker.

If the *CSSValidity* element indicates that the stylesheet is not valid, the element will include a sequence of *error* elements, analogous to the *error* elements seen in XHTMLDocInfo element.

3.2.1.4 *images* element

This element intends to express information related to the images referenced by the XHTML document. It consists of a sequence of *image* elements, each of them including the URI of the image, information about the process of *retrieval* of the image (*retrievedURI*, *HTTPRequest* and *HTTPResponse* mOKI elements already explained in previous sections), and information of the image of special interest to mobile OK Basic Tests 1.0 (*imageInfo* element).

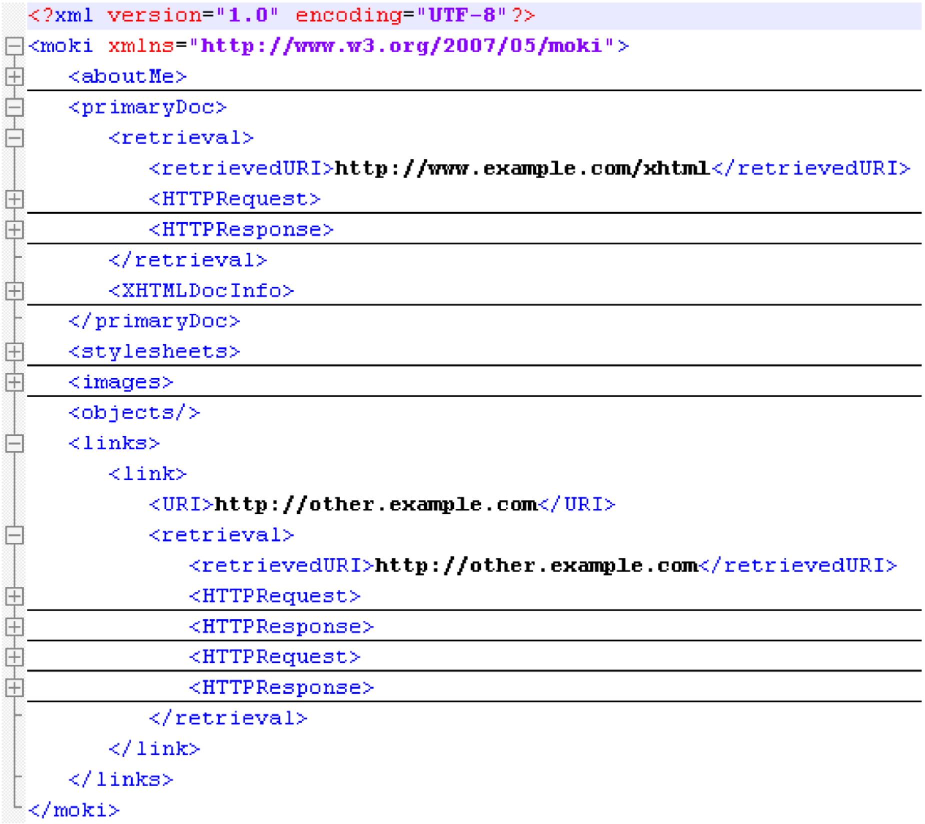


```
<?xml version="1.0" encoding="UTF-8"?>
<moki xmlns="http://www.w3.org/2007/05/moki">
   <aboutMe>
   <primaryDoc>
      <retrieval>
         <retrievedURI>http://www.example.com/xhtml</retrievedURI>
         <HTTPRequest>
         <HTTPResponse>
      </retrieval>
      <XHTMLDocInfo>
   </primaryDoc>
   <stylesheets>
   <images>
      <image>
         <URI>/xhtml/images/example.gif</URI>
         <retrieval>
            <retrievedURI>http://www.example.com/xhtml/images/example.gif</retrievedURI>
            <HTTPRequest>
            <HTTPResponse>
         </retrieval>
         <imageInfo type="image/gif">
            <validity valid="true"/>
            <transparency transparent="false"/>
            <actualDimensions height="35" width="85"/>
         </imageInfo>
      </image>
   </images>
   <objects/>
   <links>
</moki>
```

Illustration 10: Example of information contained in images *element*

3.2.1.5 *links* element

This element includes information relative to other web resources (others than external stylesheets and images) referenced in the XHTML document. This element has as many *link* children elements as external web resources are referenced in the document. Each link element contains the URI of the link and retrieval information (HTTP dialog to retrieve the web resource corresponding to the URI).

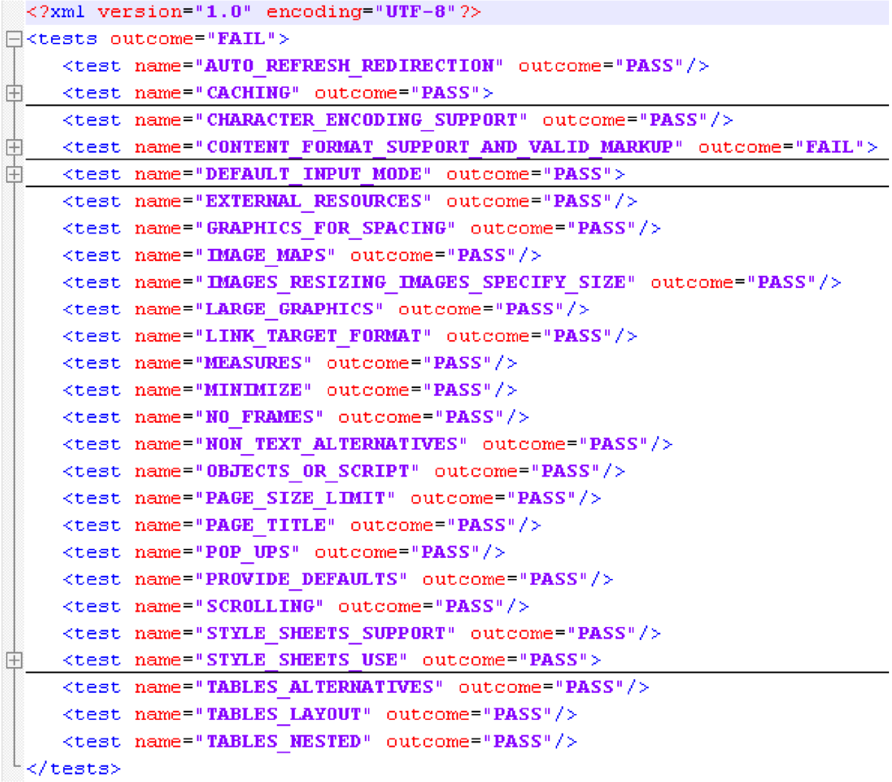


```
<?xml version="1.0" encoding="UTF-8"?>
<moki xmlns="http://www.w3.org/2007/05/moki">
   <aboutMe>
   <primaryDoc>
      <retrieval>
         <retrievedURI>http://www.example.com/xhtml</retrievedURI>
         <HTTPRequest>
         <HTTPResponse>
      </retrieval>
      <XHTMLDocInfo>
   </primaryDoc>
   <stylesheets>
   <images>
   <objects/>
   <links>
      <link>
         <URI>http://other.example.com</URI>
         <retrieval>
            <retrievedURI>http://other.example.com</retrievedURI>
            <HTTPRequest>
            <HTTPResponse>
            <HTTPRequest>
            <HTTPResponse>
         </retrieval>
      </link>
   </links>
</moki>
```

Illustration 11: Example of links *element*

3.2.2 Results document

results.xml is the final document generated by W3C mobileOK Checker 1.0a after processing the previously generated mOKI document. It consists of an XML root element named *tests* which contains a sequence of *test* elements. tests element will be qualified by an *outcome* attribute. The value of *outcome* can be *“FAIL”* (if any of the *test* children elements have an *outcome* attribute with a value of *“FAIL”*) or *“PASS”* (if all the *test* children elements have an *outcome* attribute with a value of *“PASS”*). For an explanation of the formal meaning given to *“FAIL”* and*“PASS”* (and *“WARN”*), and for the definition and naming of each test, it is recommended to visit the W3C mobileOK Basic Tests 1.0 document. Thus, *results.xml* will indicate in *tests* element that all the tests passed (or that there is a warning for the user to take into account, at most) or that any of the tests failed. Remember that the tests performed by the Checker should have been given to the JAR binary file as an optional second argument, just after the URI to be checked (see subsection Execution of this document). If no test is passed as argument to the Checker, all the tests will be performed against the URI.

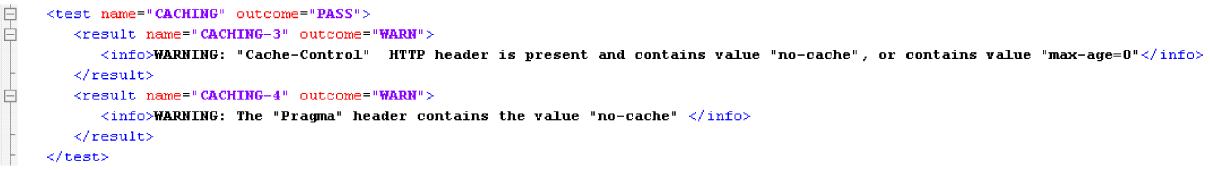


```
<?xml version="1.0" encoding="UTF-8"?>
<tests outcome="FAIL">
   <test name="AUTO_REFRESH_REDIRECTION" outcome="PASS"/>
   <test name="CACHING" outcome="PASS">
   <test name="CHARACTER_ENCODING_SUPPORT" outcome="PASS"/>
   <test name="CONTENT_FORMAT_SUPPORT_AND_VALID_MARKUP" outcome="FAIL">
   <test name="DEFAULT_INPUT_MODE" outcome="PASS">
   <test name="EXTERNAL_RESOURCES" outcome="PASS"/>
   <test name="GRAPHICS_FOR_SPACING" outcome="PASS"/>
   <test name="IMAGE_MAPS" outcome="PASS"/>
   <test name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE" outcome="PASS"/>
   <test name="LARGE_GRAPHICS" outcome="PASS"/>
   <test name="LINK_TARGET_FORMAT" outcome="PASS"/>
   <test name="MEASURES" outcome="PASS"/>
   <test name="MINIMIZE" outcome="PASS"/>
   <test name="NO_FRAMES" outcome="PASS"/>
   <test name="NON_TEXT_ALTERNATIVES" outcome="PASS"/>
   <test name="OBJECTS_OR_SCRIPT" outcome="PASS"/>
   <test name="PAGE_SIZE_LIMIT" outcome="PASS"/>
   <test name="PAGE_TITLE" outcome="PASS"/>
   <test name="POP_UPS" outcome="PASS"/>
   <test name="PROVIDE_DEFAULTS" outcome="PASS"/>
   <test name="SCROLLING" outcome="PASS"/>
   <test name="STYLE_SHEETS_SUPPORT" outcome="PASS"/>
   <test name="STYLE_SHEETS_USE" outcome="PASS">
   <test name="TABLES_ALTERNATIVES" outcome="PASS"/>
   <test name="TABLES_LAYOUT" outcome="PASS"/>
   <test name="TABLES_NESTED" outcome="PASS"/>
</tests>
```

Illustration 12: results.xml *example*

The example in Illustration 12 shows test results for the mOKI document used in the previous subsection of this document: Intermediate (mOKI) document. Note how the *name* given to each *test* is the same name given to tests in the W3C mobileOK Basic Tests 1.0 document.

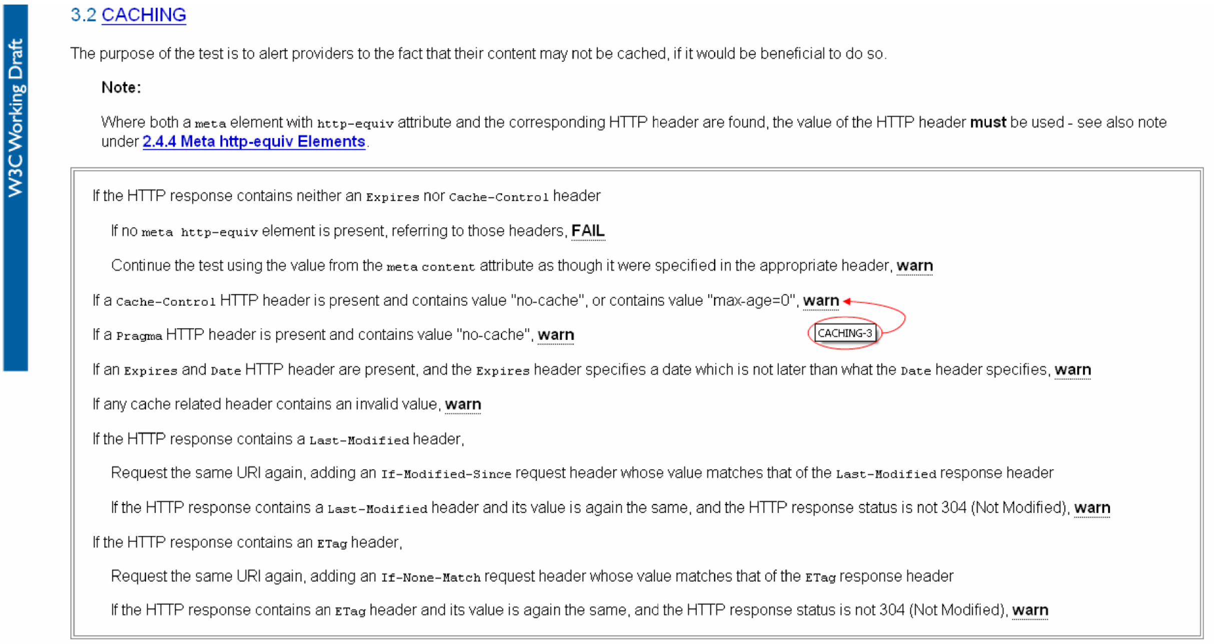
Each *test* element will include a sequence of *result* children elements if its *outcome* is “FAIL” or if it is “PASS” but the checker needs to warn the user about any interesting issue related to the test.



```
<test name="CACHING" outcome="PASS">
   <result name="CACHING-3" outcome="WARN">
      <info>WARNING: "Cache-Control"  HTTP header is present and contains value "no-cache", or contains value "max-age=0"</info>
   </result>
   <result name="CACHING-4" outcome="WARN">
      <info>WARNING: The "Pragma" header contains the value "no-cache" </info>
   </result>
</test>
```

Illustration 13: Details for CACHING *test which* PASS*es with* WARN*ings*

Illustration 13 shows how a test that passed successfully includes extra results information to want the user about certain interesting issues. It is important to note that the *name* attribute for each *result* element with a *test* element references each subtest being part of a test, as mentioned in W3C mobileOK Basic Tests 1.0 document (see Illustration 14).



3.2 CACHING

The purpose of the test is to alert providers to the fact that their content may not be cached, if it would be beneficial to do so.

Note:

Where both a `meta` element with `http-equiv` attribute and the corresponding HTTP header are found, the value of the HTTP header **must** be used - see also note under 2.4.4 Meta http-equiv Elements.

If the HTTP response contains neither an `Expires` nor `Cache-Control` header

If no `meta` `http-equiv` element is present, referring to those headers, FAIL

Continue the test using the value from the `meta` `content` attribute as though it were specified in the appropriate header, warn

If a `Cache-Control` HTTP header is present and contains value "no-cache", or contains value "max-age=0", warn

If a `Pragma` HTTP header is present and contains value "no-cache", warn

If an `Expires` and `Date` HTTP header are present, and the `Expires` header specifies a date which is not later than what the `Date` header specifies, warn

If any cache related header contains an invalid value, warn

If the HTTP response contains a `Last-Modified` header,

Request the same URI again, adding an `If-Modified-Since` request header whose value matches that of the `Last-Modified` response header

If the HTTP response contains a `Last-Modified` header and its value is again the same, and the HTTP response status is not 304 (Not Modified), warn

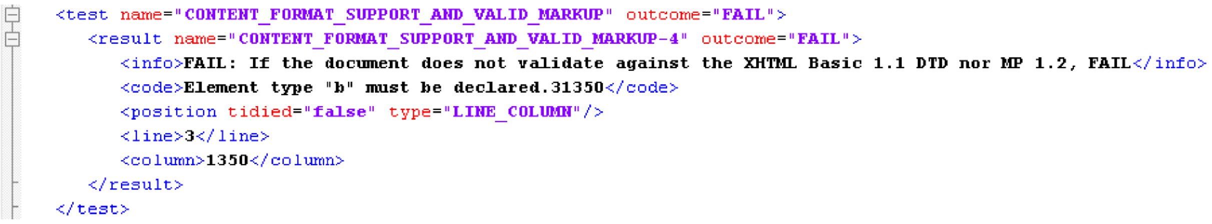
If the HTTP response contains an `ETag` header,

Request the same URI again, adding an `If-None-Match` request header whose value matches that of the `ETag` response header

If the HTTP response contains an `ETag` header and its value is again the same, and the HTTP response status is not 304 (Not Modified), warn

Illustration 14: Naming of subtests in mobileOK Basic Tests 1.0 document

Results information related to failing tests or tests with warnings include general information (info element containing the text of the subtest in the W3C mobileOK Basic Tests 1.0 document), the origin of the error (code element, which generally reports the markup causing problems and/or the output error of an underlying library used by the checker), and the position of the error (as seen in Illustration 15).



```
<test name="CONTENT_FORMAT_SUPPORT_AND_VALID_MARKUP" outcome="FAIL">
   <result name="CONTENT_FORMAT_SUPPORT_AND_VALID_MARKUP-4" outcome="FAIL">
      <info>FAIL: If the document does not validate against the XHTML Basic 1.1 DTD nor MP 1.2, FAIL</info>
      <code>Element type "b" must be declared.31350</code>
      <position tidied="false" type="LINE_COLUMN"/>
      <line>3</line>
      <column>1350</column>
   </result>
</test>
```

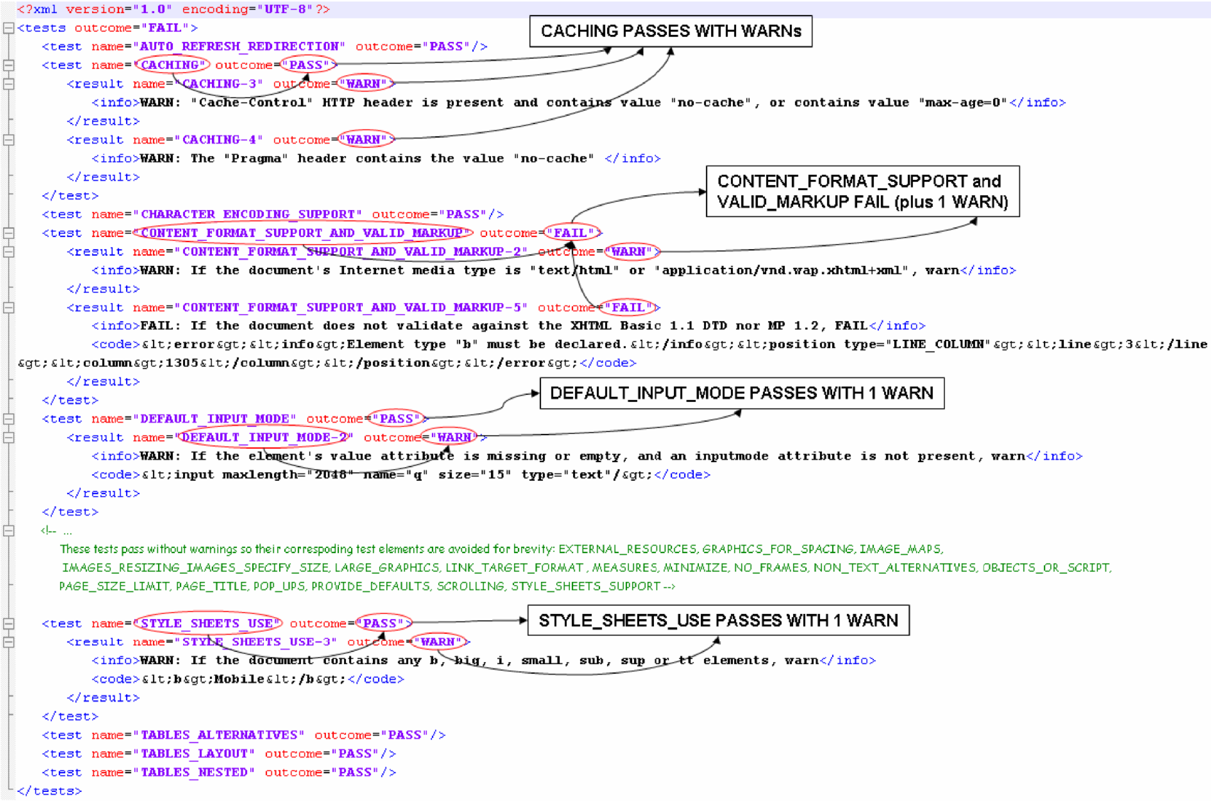
Illustration 15: Additional information for FAILing results

4 Examples

This section tries to illustrate the use of the W3C mobileOK Checker 1.0a with real mobile sites (although real URIs have been changed following the example.domain model). The only intention of this section is to help readers to execute the Checker and interpret its *results.xml* output document.

4.1 Example 1

First example is a test that fails on CONTENT_FORMAT_SUPPORT and VALID_MARKUP, while it warns about some issues regarding CACHING, DEFAULT_INPUT_MODE and STYLE_SHEETS_USE. Firstly, it would be necessary to examine the *results.xml* document (shown in Illustration 17) to gather all the information needed to infer the previous conclusions.



CACHING PASSES WITH WARNs

CONTENT_FORMAT_SUPPORT and VALID_MARKUP FAIL (plus 1 WARN)

DEFAULT_INPUT_MODE PASSES WITH 1 WARN

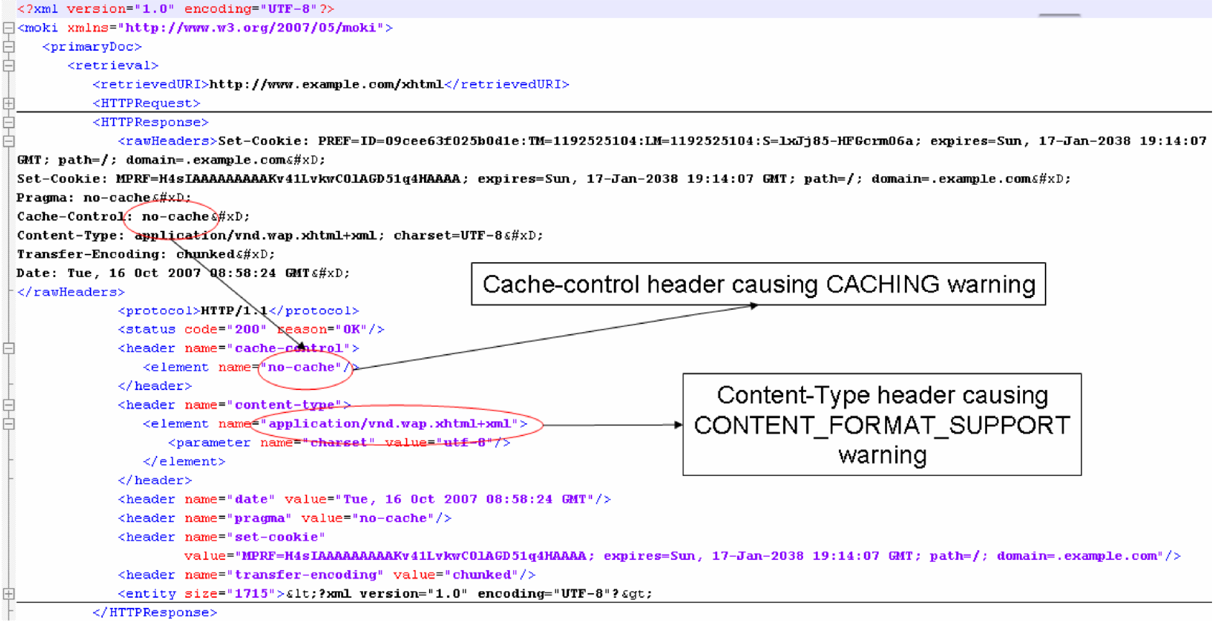
STYLE_SHEETS_USE PASSES WITH 1 WARN

```
<?xml version="1.0" encoding="UTF-8"?>
<tests outcome="FAIL">
   <test name="AUTO_REFRESH_REDIRECTION" outcome="PASS"/>
   <test name="CACHING" outcome="PASS">
      <result name="CACHING-3" outcome="WARN">
         <info>WARN: "Cache-Control" HTTP header is present and contains value "no-cache", or contains value "max-age=0"</info>
      </result>
      <result name="CACHING-4" outcome="WARN">
         <info>WARN: The "Pragma" header contains the value "no-cache" </info>
      </result>
   </test>
   <test name="CHARACTER_ENCODING_SUPPORT" outcome="PASS"/>
   <test name="CONTENT_FORMAT_SUPPORT_AND_VALID_MARKUP" outcome="FAIL">
      <result name="CONTENT_FORMAT_SUPPORT_AND_VALID_MARKUP-2" outcome="WARN">
         <info>WARN: If the document's Internet media type is "text/html" or 'application/vnd.wap.xhtml+xml", warn</info>
      </result>
      <result name="CONTENT_FORMAT_SUPPORT_AND_VALID_MARKUP-5" outcome="FAIL">
         <info>FAIL: If the document does not validate against the XHTML Basic 1.1 DTD nor MP 1.2, FAIL</info>
         <code>&lt;error&gt;&lt;info&gt;Element type "b" must be declared.&lt;/info&gt;&lt;position type="LINE_COLUMN"&gt;&lt;line&gt;3&lt;/line
&gt;&lt;column&gt;1305&lt;/column&gt;&lt;/position&gt;&lt;/error&gt;</code>
      </result>
   </test>
   <test name="DEFAULT_INPUT_MODE" outcome="PASS">
      <result name="DEFAULT_INPUT_MODE-2" outcome="WARN">
         <info>WARN: If the element's value attribute is missing or empty, and an inputmode attribute is not present, warn</info>
         <code>&lt;input maxlength="2048" name="q" size="15" type="text"/&gt;</code>
      </result>
   </test>
   <!-- ...
     These tests pass without warnings so their correspoding test elements are avoided for brevity: EXTERNAL_RESOURCES, GRAPHICS_FOR_SPACING, IMAGE_MAPS,
     IMAGES_RESIZING_IMAGES_SPECIFY_SIZE, LARGE_GRAPHICS, LINK_TARGET_FORMAT , MEASURES, MINIMIZE, NO_FRAMES, NON_TEXT_ALTERNATIVES, OBJECTS_OR_SCRIPT,
     PAGE_SIZE_LIMIT, PAGE_TITLE, POP_UPS, PROVIDE_DEFAULTS, SCROLLING, STYLE_SHEETS_SUPPORT -->

   <test name="STYLE_SHEETS_USE" outcome="PASS">
      <result name="STYLE_SHEETS_USE-3" outcome="WARN">
         <info>WARN: If the document contains any b, big, i, small, sub, sup or tt elements, warn</info>
         <code>&lt;b&gt;Mobile&lt;/b&gt;</code>
      </result>
   </test>
   <test name="TABLES_ALTERNATIVES" outcome="PASS"/>
   <test name="TABLES_LAYOUT" outcome="PASS"/>
   <test name="TABLES_NESTED" outcome="PASS"/>
</tests>
```

Illustration 16: Gathering essential information from results.xml *for example 1*

Now, *moki.xml* would allow the user to confirm what *results.xml* indicates:



Cache-control header causing CACHING warning

Content-Type header causing CONTENT_FORMAT_SUPPORT warning

```
<?xml version="1.0" encoding="UTF-8"?>
<moki xmlns="http://www.w3.org/2007/05/moki">
   <primaryDoc>
      <retrieval>
         <retrievedURI>http://www.example.com/xhtml</retrievedURI>
         <HTTPRequest>
         <HTTPResponse>
            <rawHeaders>Set-Cookie: PREF=ID=09cee63f025b0d1e:TM=1192525104:LM=1192525104:S=1xJj85-HFGcrm06a; expires=Sun, 17-Jan-2038 19:14:07
GMT; path=/; domain=.example.com&#xD;
Set-Cookie: MPRF=H4sIAAAAAAAAAKv41LvkwC0lAGD51q4HAAAA; expires=Sun, 17-Jan-2038 19:14:07 GMT; path=/; domain=.example.com&#xD;
Pragma: no-cache&#xD;
Cache-Control: no-cache&#xD;
Content-Type: application/vnd.wap.xhtml+xml; charset=UTF-8&#xD;
Transfer-Encoding: chunked&#xD;
Date: Tue, 16 Oct 2007 08:58:24 GMT&#xD;
</rawHeaders>
            <protocol>HTTP/1.1</protocol>
            <status code="200" reason="OK"/>
            <header name="cache-control">
               <element name="no-cache"/>
            </header>
            <header name="content-type">
               <element name="application/vnd.wap.xhtml+xml">
                  <parameter name="charset" value="utf-8"/>
               </element>
            </header>
            <header name="date" value="Tue, 16 Oct 2007 08:58:24 GMT"/>
            <header name="pragma" value="no-cache"/>
            <header name="set-cookie"
                    value="MPRF=H4sIAAAAAAAAAKv41LvkwC0lAGD51q4HAAAA; expires=Sun, 17-Jan-2038 19:14:07 GMT; path=/; domain=.example.com"/>
            <header name="transfer-encoding" value="chunked"/>
            <entity size="1715">&lt;?xml version="1.0" encoding="UTF-8"?&gt;
         </HTTPResponse>
```

Illustration 17: Finding reasons for CACHING *and* CONTENT_FORMAT_SUPPORT *warns in* moki.xml *for example 1*

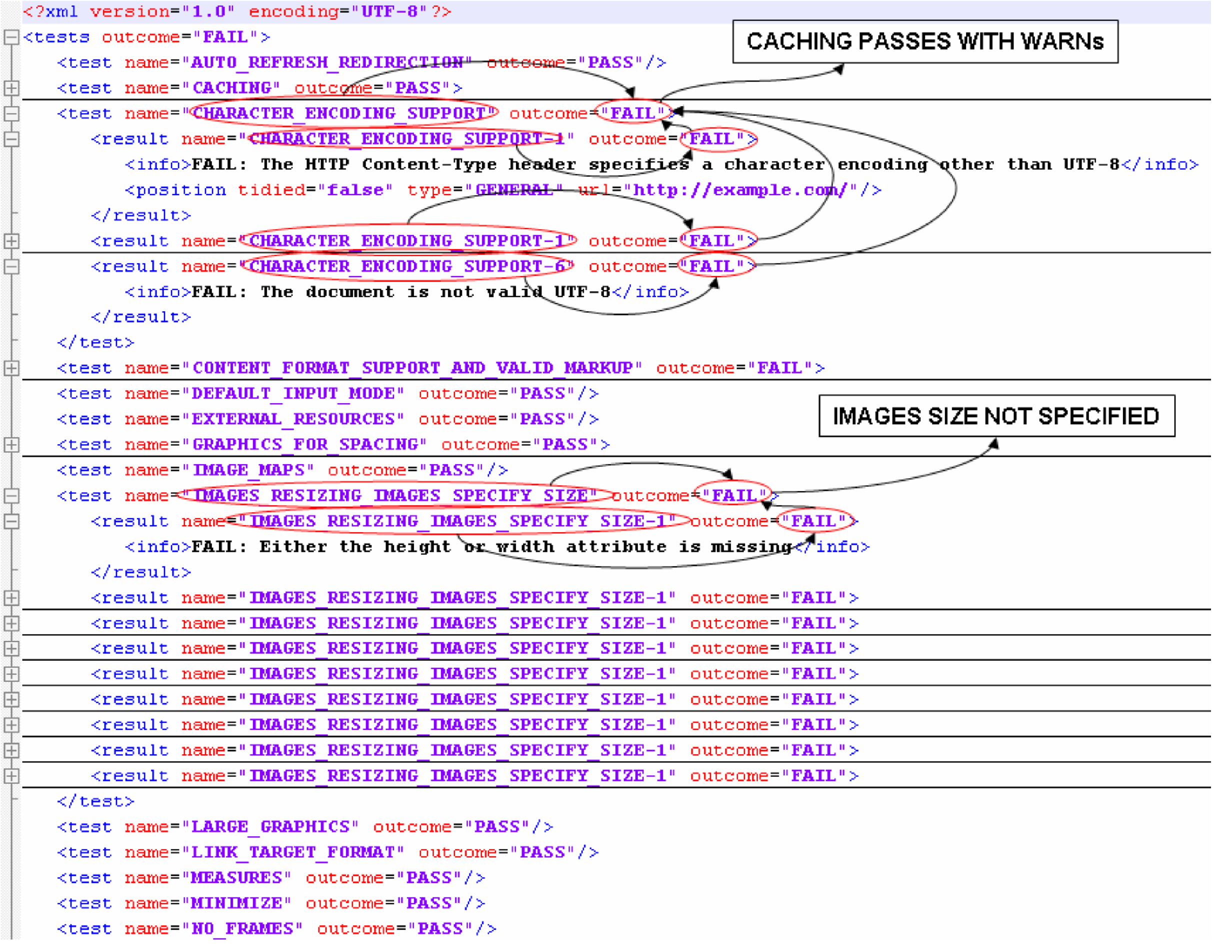
4.2 Example 2

The second example shows the results and intermediate documents created after a test on a page that fails on:

- CHARACTER_ENCODING_SUPPORT
- CONTENT_FORMAT_SUPPORT_AND_VALID_MARKUP
- IMAGES_RESIZING_IMAGES_SPECIFY_SIZE
- NON_TEXT_ALTERNATIVES
- PAGE_TITLE

For the sake of brevity, the overview of this example will not take into account all the issues related to Content Format Support and Valid Markup (as it was already commented in the Example 1) nor repeated fails of the same nature on different resources.

The mOKI document for this example is shown divided in two illustrations due to its size.



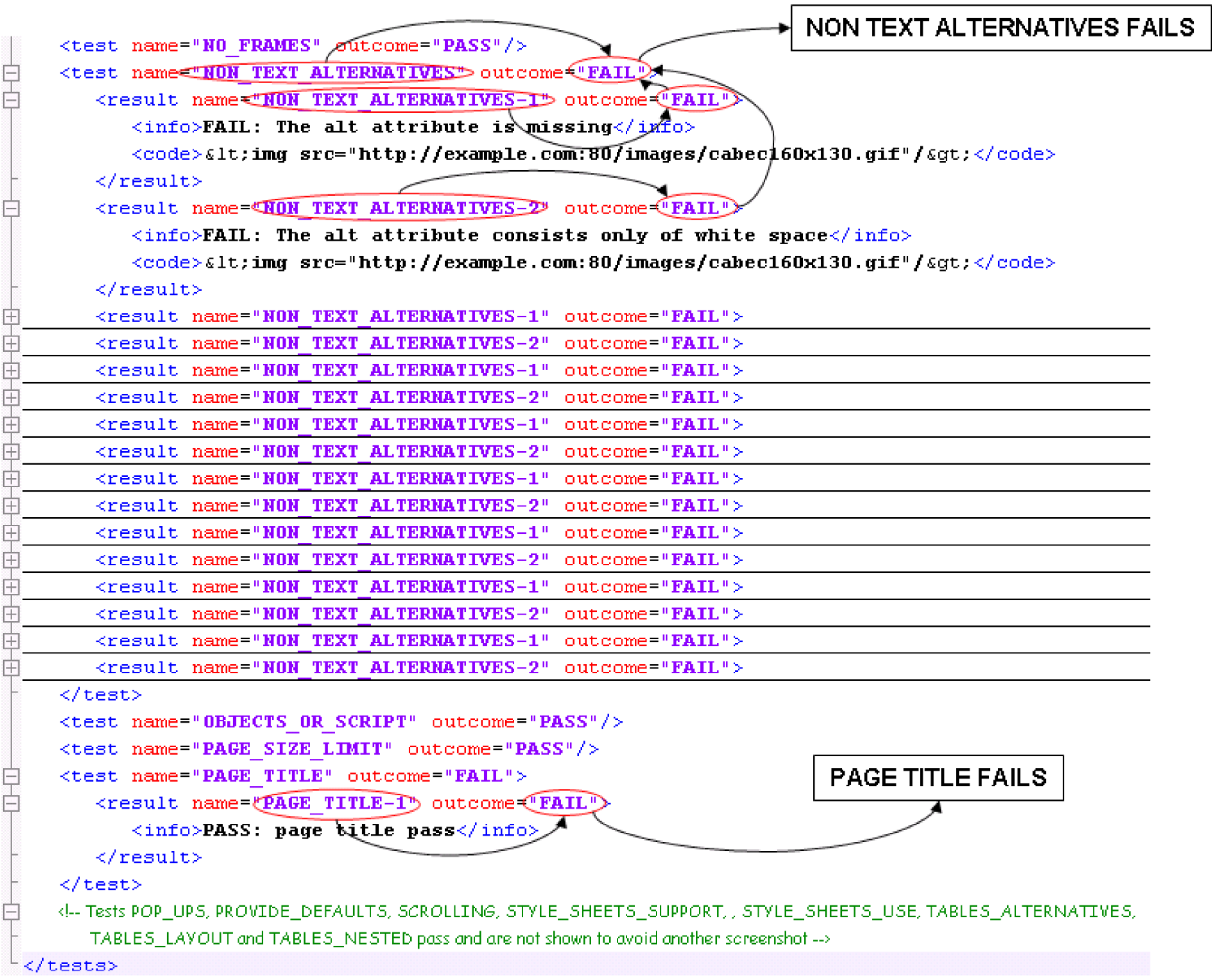
CACHING PASSES WITH WARNs

IMAGES SIZE NOT SPECIFIED

```
<?xml version="1.0" encoding="UTF-8"?>
<tests outcome="FAIL">
    <test name="AUTO_REFRESH_REDIRECTION" outcome="PASS"/>
    <test name="CACHING" outcome="PASS">
    <test name="CHARACTER_ENCODING_SUPPORT" outcome="FAIL">
        <result name="CHARACTER_ENCODING_SUPPORT-1" outcome="FAIL">
            <info>FAIL: The HTTP Content-Type header specifies a character encoding other than UTF-8</info>
            <position tidied="false" type="GENERAL" url="http://example.com/"/>
        </result>
        <result name="CHARACTER_ENCODING_SUPPORT-1" outcome="FAIL">
        <result name="CHARACTER_ENCODING_SUPPORT-6" outcome="FAIL">
            <info>FAIL: The document is not valid UTF-8</info>
        </result>
    </test>
    <test name="CONTENT_FORMAT_SUPPORT_AND_VALID_MARKUP" outcome="FAIL">
    <test name="DEFAULT_INPUT_MODE" outcome="PASS"/>
    <test name="EXTERNAL_RESOURCES" outcome="PASS"/>
    <test name="GRAPHICS_FOR_SPACING" outcome="PASS">
    <test name="IMAGE_MAPS" outcome="PASS"/>
    <test name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE" outcome="FAIL">
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
            <info>FAIL: Either the height or width attribute is missing</info>
        </result>
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
        <result name="IMAGES_RESIZING_IMAGES_SPECIFY_SIZE-1" outcome="FAIL">
    </test>
    <test name="LARGE_GRAPHICS" outcome="PASS"/>
    <test name="LINK_TARGET_FORMAT" outcome="PASS"/>
    <test name="MEASURES" outcome="PASS"/>
    <test name="MINIMIZE" outcome="PASS"/>
    <test name="NO_FRAMES" outcome="PASS"/>
```

Illustration 18: Gathering essential information from results.xml *for example 2 (part 1)*

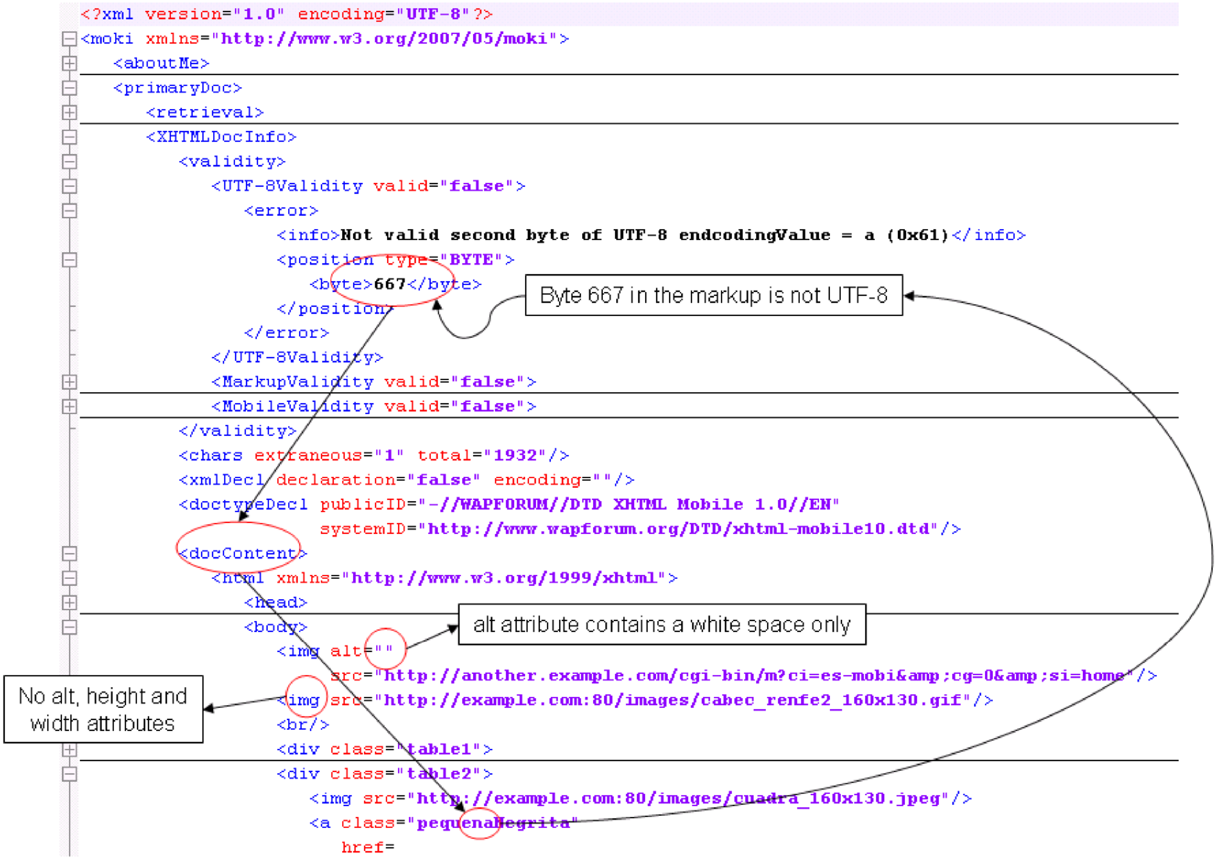
Illustration 18 shows the first part of the results.xml for this second example. It shows how the HTML document requested claimed to contain UTF-8 characters and that at least one character is not UTF-8. It also reports that some images have their *width* and/or *height* attributes missing.Note that this applies to several images and that is the reason why so many results with a name of IMAGES_RESIZING_IMAGES_SPECIFY_SIZE occur in the document.



```
    <test name="NO_FRAMES" outcome="PASS"/>
    <test name="NON_TEXT_ALTERNATIVES" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-1" outcome="FAIL">
          <info>FAIL: The alt attribute is missing</info>
          <code>&lt;img src="http://example.com:80/images/cabec160x130.gif"/&gt;</code>
       </result>
       <result name="NON_TEXT_ALTERNATIVES-2" outcome="FAIL">
          <info>FAIL: The alt attribute consists only of white space</info>
          <code>&lt;img src="http://example.com:80/images/cabec160x130.gif"/&gt;</code>
       </result>
       <result name="NON_TEXT_ALTERNATIVES-1" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-2" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-1" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-2" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-1" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-2" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-1" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-2" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-1" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-2" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-1" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-2" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-1" outcome="FAIL">
       <result name="NON_TEXT_ALTERNATIVES-2" outcome="FAIL">
    </test>
    <test name="OBJECTS_OR_SCRIPT" outcome="PASS"/>
    <test name="PAGE_SIZE_LIMIT" outcome="PASS"/>
    <test name="PAGE_TITLE" outcome="FAIL">
       <result name="PAGE_TITLE-1" outcome="FAIL">
          <info>PASS: page title pass</info>
       </result>
    </test>
    <!-- Tests POP_UPS, PROVIDE_DEFAULTS, SCROLLING, STYLE_SHEETS_SUPPORT, , STYLE_SHEETS_USE, TABLES_ALTERNATIVES,
       TABLES_LAYOUT and TABLES_NESTED pass and are not shown to avoid another screenshot -->
 </tests>
```

Illustration 19: Gathering essential information from results.xml *for example 2 (part 2)*

Illustration 19 represents the second part of the results document for this second example. It shows how the original HTML document does not include an alt attribute for images or it is included but with a string value containing a single blank space. Finally, results.xml indicates that the page does not includes a title.



```
<?xml version="1.0" encoding="UTF-8"?>
<moki xmlns="http://www.w3.org/2007/05/moki">
   <aboutMe>
   <primaryDoc>
      <retrieval>
      <XHTMLDocInfo>
         <validity>
            <UTF-8Validity valid="false">
               <error>
                  <info>Not valid second byte of UTF-8 endcodingValue = a (0x61)</info>
                  <position type="BYTE">
                     <byte>667</byte>
                  </position>
               </error>
            </UTF-8Validity>
            <MarkupValidity valid="false">
            <MobileValidity valid="false">
         </validity>
         <chars extraneous="1" total="1932"/>
         <xmlDecl declaration="false" encoding=""/>
         <doctypeDecl publicID="-//WAPFORUM//DTD XHTML Mobile 1.0//EN"
                      systemID="http://www.wapforum.org/DTD/xhtml-mobile10.dtd"/>
         <docContent>
            <html xmlns="http://www.w3.org/1999/xhtml">
               <head>
               <body>
                  <img alt=" "
                     src="http://another.example.com/cgi-bin/m?ci=es-mobi&amp;cg=0&amp;si=home"/>
                  <img src="http://example.com:80/images/cabec_renfe2_160x130.gif"/>
                  <br/>
                  <div class="table1">
                  <div class="table2">
                     <img src="http://example.com:80/images/cuadra_160x130.jpeg"/>
                     <a class="pequenaNegrita"
                        href=
```

Byte 667 in the markup is not UTF-8

alt attribute contains a white space only

No alt, height and width attributes

Illustration 20: mOKI document for example 2

Illustration 20 marks all the parts of the mOKI document for this second example where to find all the FAIL conditions reported by results.xml. Firstly, the element XHTMLDocInfo in primaryDoc shows in its validity child element that the document is not UTF-8 valid because there is an invalid character in the byte 667 of the markup. It is possible to take a look in the docContent element to examine the original markup of the page and find the erroneous byte (peque**ñ**aNegrita).

The mOKI document sample also shows how img elements in the page do not include any width and height attributes, thus not passing the IMAGES_RESIZING_IMAGE_SPECIFY_SIZE test.

Although it is not shown in Illustration 20, the page did not include a *title* element within its *head* and this is the reason for the PAGE_TITLE test to fail.

4.3 Example 3

[Yet to be developed]

5 Known issues

5.1 About W3C mobileOK Checker 1.0a

The development of the tool is alive. Please do not hesitate to send your comments to the mobileOK Checker Task Force mailing list (public-mobileok-checker@w3.org). Your feedback is very important for the authors of W3C mobileOK Checker 1.0a so this software can be improved. Optionally, you can access the bug tracker of the Checker at http://www.w3.org/Bugs/Public/ and help W3C developing this tool.

5.2 About this User Manual

- **General issues:** Review and improve wording all across the document.
- **Subsection 3.2**: More detailed description of mOKI and result formats.
- **Section 4 (Examples)**: Detail Examples 1 and 2 in more depth. Add Example 3.

6 References

NOTE: When referring a W3C document, the version of the document that has been considered at this version of the W3C mobileOK Checker User Manual appers in this section. Additionally, the URI to the latest version of such document is offered between square brackets.

(1) W3C mobileOK Basic Tests 1.0, http://www.w3.org/TR/2007/WD-mobileOK-basic10-tests-20070928/. Editors: Sean Owen (Google) and Jo Rabin (dotMobi [latest version: http://www.w3.org/TR/mobileOK-basic10-tests/].

(2) W3C Mobile Web Best Practices (Basic Guidelines) 1.0, http://www.w3.org/TR/2006/PR-mobile-bp-20061102/. Editors: Jo Rabin (dotMobi) and Charles McCathieNevile -early drafts- (Opera Software) [latest version: http://www.w3.org/TR/mobile-bp/].

(3) W3C License, http://www.w3.org/Consortium/Legal/2002/copyright-software-20021231.

(4) W3C Mobile Web Best Practices working group, http://www.w3.org/2005/MWI/BPWG/.

(5) W3C Mobile Web Initiative, http://www.w3.org/Mobile/.

(6) XML Schema for the mOKI document format, http://dev.w3.org/cvsweb/2007/mobileok-ref/moki/schema/moki.xsd. Editor: Jo Rabin (dotMobi). HTML version available at http://dev.w3.org/cvsweb/2007/mobileok-ref/moki/schema/moki.xsd.html.

(7) Representation of CSS in XML, http://dev.w3.org/cvsweb/2007/cssxml/. Editor: Jo Rabin (dotMobi).

7 Acknowledgements

W3C mobileOK Checker 1.0a is the result of the effort of several organizations. In alphabetical order, the members of the W3C mobileOK Checker Task Force (taking place within the W3C Mobile Web Best Practices working group) who contributed to the creation of this software tool and its documentation are:

- Roland Gülle (Sevenval)
- Miguel García (Fundación CTIC)
- Dominique Hazäel-Massieux (W3C)
- Laura Holmes (Google)
- Ignacio Marín (Fundación CTIC)
- Ruadhan O'Donoghue (dotMobi)
- Sean Owen (Google, Task Force leader)
- Jo Rabin (dotMobi)
- Abel Rionda (Fundación CTIC)