



WML Transformations

Proposed Version 21-June-2001

Wireless Application Protocol
WAP-244-WMLTR-20010621-p

This document is a work in process and is not an approved WAP Forum™ specification. This document is subject to revision or removal without notice. No part of this document may be used to claim conformance or interoperability with the WAP Forum specifications.

© 2001, Wireless Application Protocol Forum, Ltd. All Rights Reserved. Terms and conditions of use are available from the WAP Forum™ Web site (<http://www.wapforum.org/what/copyright.htm>).

© 2001, Wireless Application Protocol Forum, Ltd. All rights reserved.

Terms and conditions of use are available from the WAP Forum™ Web site at <http://www.wapforum.org/what/copyright.htm>.

You may use this document or any part of the document for internal or educational purposes only, provided you do not modify, edit or take out of context the information in this document in any manner. You may not use this document in any other manner without the prior written permission of the WAP Forum™. The WAP Forum authorises you to copy this document, provided that you retain all copyright and other proprietary notices contained in the original materials on any copies of the materials and that you comply strictly with these terms. This copyright permission does not constitute an endorsement of the products or services offered by you.

The WAP Forum™ assumes no responsibility for errors or omissions in this document. In no event shall the WAP Forum be liable for any special, indirect or consequential damages or any damages whatsoever arising out of or in connection with the use of this information.

This document is not a WAP Forum™ specification, is not endorsed by the WAP Forum and is informative only. This document is subject to revision or removal without notice. No part of this document may be used to claim conformance or interoperability with the WAP Forum specifications.

WAP Forum™ members have agreed to use reasonable endeavors to disclose in a timely manner to the WAP Forum the existence of all intellectual property rights (IPR's) essential to the present document. The members do not have an obligation to conduct IPR searches. This information is publicly available to members and non-members of the WAP Forum and may be found on the "WAP IPR Declarations" list at <http://www.wapforum.org/what/ipr.htm>. Essential IPR is available for license on the basis set out in the schedule to the WAP Forum Application Form.

No representations or warranties (whether express or implied) are made by the WAP Forum™ or any WAP Forum member or its affiliates regarding any of the IPR's represented on this list, including but not limited to the accuracy, completeness, validity or relevance of the information or whether or not such rights are essential or non-essential.

This document is available online in PDF format at <http://www.wapforum.org/>.

Known problems associated with this document are published at <http://www.wapforum.org/>.

Comments regarding this document can be submitted to the WAP Forum™ in the manner published at <http://www.wapforum.org/>.

Document History	
WAP-244-WMLTR-20010621-p	Current

Contents

- 1. SCOPE 4
- 2. REFERENCES 5
 - 2.1. NORMATIVE REFERENCES 5
 - 2.2. INFORMATIVE REFERENCES 5
- 3. TERMINOLOGY AND CONVENTIONS 6
 - 3.1. CONVENTIONS 6
 - 3.2. ABBREVIATIONS 6
- 4. INTRODUCTION (INFORMATIVE) 7
 - 4.1. OVERVIEW OF CHANGES BETWEEN WML VERSION 1.3 AND 2.0 7
- 5. THE WML TRANSFORMATION PROCESSOR (NORMATIVE) 9
- 6. TRANSFORMATION RULES (NORMATIVE) 10
 - 6.1. XSLT TRANSFORMATION SHEET 10
 - 6.2. THE ORG.WAPFORUM.WML.WMLEXPRESSION JAVA CLASS 21
- 7. USING THE XSLT TRANSFORMATION SHEET (INFORMATIVE) 25
- APPENDIX A. STATIC CONFORMANCE REQUIREMENTS (NORMATIVE) 26
- APPENDIX B. CHANGE HISTORY (INFORMATIVE) 27

1. Scope

Specifies transformation rules for transforming WML 1.3 documents into WML 2.0 documents.

2. References

2.1. Normative References

- [CREQ] “Specification of WAP Conformance Requirements”. WAP Forum™.
WAP-221-CREQ-20010425-a. [URL:http://www.wapforum.org/](http://www.wapforum.org/)
- [CXML] “Canonical XML version 1.0”.
W3C Recommendation 15 March 2001.
URL: <http://www.w3.org/TR/2001/REC-xml-c14n-20010315>
- [RFC2119] “Key words for use in RFCs to Indicate Requirement Levels”. S. Bradner. March 1997.
[URL:http://www.ietf.org/rfc/rfc2119.txt](http://www.ietf.org/rfc/rfc2119.txt)
- [WML2] “WML 2.0”, WAP-238-WML.
[URL:http://www.wapforum.org/](http://www.wapforum.org/)
- [WML1] “WML 1.3”, WAP-191-WML-20000219-a.
[URL:http://www.wapforum.org/](http://www.wapforum.org/)
- [WTA] “WTA”, WAP-169-WTA.
[URL:http://www.wapforum.org/](http://www.wapforum.org/)

2.2. Informative References

- [WAPARCH] “WAP Architecture”. WAP Forum™. WAP-210-WAPArch-20001130-p.
[URL:http://www.wapforum.org/](http://www.wapforum.org/)
- [XSLT] “XSL Transformations (XSLT) Version 1.1”,
W3C Working Draft 12 December 2000.
[URL:http://www.w3.org/TR/xslt11](http://www.w3.org/TR/xslt11)
- [XMLNS] “Namespaces in XML”,
W3C Recommendation 14 January 1999.
URL: <http://www.w3.org/TR/1999/REC-xml-names>
- [XHTMLBasic] “XHTML™ Basic”,
W3C Recommendation 19 December 2000.
[URL:http://www.w3.org/TR/2000/REC-xhtml-basic-20001219](http://www.w3.org/TR/2000/REC-xhtml-basic-20001219)
- [XHTMLMod] “Modularization of XHTML™”,
W3C Recommendation 10 April 2001.
[URL:http://www.w3.org/TR/xhtml-modularization](http://www.w3.org/TR/xhtml-modularization)

3. Terminology and Conventions

3.1. Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

Definitions are written as

Definition: A definition is a term or expression that is used a lot.

Conformance requirements are written as

Conformance requirement: A user agent that conforms to this part of the specification MUST implement this requirement.

Notes are written as

Note: An informational message to the reader.

3.2. Abbreviations

WAP	Wireless Application Protocol
WML	Wireless Markup Language
WTA	Wireless Telephony Applications
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformations

4. Introduction (Informative)

This document assumes the reader is familiar with XSLT [XSLT], WML1 [WML1], and WML2 [WML2].

Although WML 2.0 [WML2] supports all features of WML 1.3 [WML1], since WML 2.0 is based on XHTML and WML 1.3 is a language of its own, the WML 2.0 user agent cannot read WML 1.3 documents. The element names and how they are combined is different. In WML 2.0, for example, the root element is `html` in WML 1.3 it is `wml`, and WML 2.0 does not have the WML 1.3 `template` element.

It is possible, however, to transform a valid WML 1.3 document into a valid WML 2.0 document. Older versions of WML 1.3 that are valid WML 1.3 documents can also be transformed. It is also possible to transform WTA-WML 1.2 [WTA] documents. The transformation can be made transparent to the user. Users can access WAP 1 services from a WML 2.0 user agent without detecting any difference compared to real WAP 2 services.

This specification does not mandate, or even recommend, where or when the transformation should take place. It may be in one of the following places:

- In a proxy server that enables WML 2.0 user agents to access WML 1.3 documents.
- In a server that automatically transforms WML 1.3 documents into WML 2.0 document when requested by a WML 2.0 user agent.

The WAP 2.0 client may, as an additional alternative, support WML 1.3 by transforming such documents into WML 2.0 before the WML 2.0 user agent processes the document.

The transformation rules are specified in section 6 as an XSLT [XSLT] transformation sheet. The use of XSLT in order to specify the transformation rules as formal and complete as possible, does not suggest that a XSLT transformation sheet is the only possible implementation, not even that it is an efficient one. Depending on where the transformation takes place, it may be implemented in many different ways.

4.1. Overview of Changes Between WML Version 1.3 and 2.0

Here is an overview of the changes that have been made to WML since version 1.3 and a few words about the transformation.

- Instead of being a document type of its own, WML 2.0 is an extension of the following XHTML modules [XHTMLMod]: every XHTML module supported by XHTML Basic [XHTMLBasic], the Block level presentation module, the Inline Stylesheets module, and the Stylesheets module. This means that some elements now have more attributes, and there are many new elements as well.
- Each element declares an XML Namespace [XMLNS]. The default namespace is XHTML. The WML namespace is bound to the 'wml' prefix.
- The root element is `html` instead of `wml`.
- The `head` element is mandatory. If there is none, the transformation inserts one.
- The XHTML `title` element in the document header is required. In the transformation the title of the first card of the WML 1 document as the document title.
- The `template` element has been removed. Instead template-level elements may be located directly after the head element. The transformation moves the elements to their new location.
- Shadowing of `do` elements is not supported. The transformation distributes the template-level `do` elements into the cards according to the WML 1.3 shadowing algorithm.
- The "ordered" attribute on the `card` element has been removed. Ignored by the transformation.

- Nesting of `optgroup` is illegal. Only the first level is transformed. Deeper levels are collapsed into a flat list of `optgroup` elements.
- The new “label” attribute on the `optgroup` is mandatory. Transformation inserts an empty label.
- The WML “type” attribute of the `do` element is named “role”, and also has some new values.
- The “optional” attribute on the `do` element has been removed. Ignored by the transformation.
- A new `widget` element can be used inside the `do` element to specify preferred widget types for presentation.
- The `do` element can be used in more places than in WML 1.3. For example, inside a `td` element.

WML 2.0 extends XHTML elements with WML attributes and extends the XHTML document model with WML elements. In some cases, WML 2.0 also extends the XHTML document model with XHTML elements:

- Inline tables. In WML 2.0 the `table` element can be inside the `p` element. This is illegal in XHTML. In order to transform from WML 1.3, however, it is allowed in WML 2.0.
- The form control elements `input` and `select` can be inside the `pre` element. This is illegal in XHTML. In order to transform from WML 1.3, however, it is allowed in WML 2.0.

5. The WML Transformation Processor (Normative)

This section defines the WML transformation processor, a *black box*, called *E* in the rest of this section.

Definition: Transformation processor *X* implements exactly the XSLT transformation sheet defined in section 6 of this document. (It may be any conforming XSLT 1.1. [XSLT] processor that takes the transformation sheets from section 6 as input.)

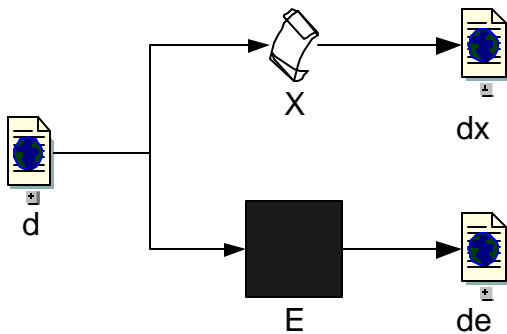
Definition: Document *d* is any valid WML 1.3 [WML1] or WTA-WML 1.2 [WTA] document.

Documents older than WML 1.3 may also be valid WML 1.3 documents.

Definition: Document *dx* is the result of transforming *d* using *X*. ($X(d) \rightarrow dx$).

Definition: Document *de* is the result of transforming *d* using *E*. ($E(d) \rightarrow de$).

This is illustrated in the following figure:



Conformance Rule: Transformation processor *E* conforms to this specification if the documents *dx* and *de* are identical after *XML canonicalization* [CXML] has been applied.

An effect of this conformance clause is that the two documents, *dx* and *de*, may look lexically different, depending on the implementation of *E* and *X*.

6. Transformation Rules (Normative)

This section is an XSLT 1.1 [XSLT] transformation sheet that defines all transformation rules of this specification.

The transformation sheet is organised in the following way. There is one template for each WML 1.3 element in the source document. The template creates the corresponding WML 2.0 element in the result document, plus any additional attributes. It goes on to apply appropriate templates on all element types that may be children of the current WML 1.3 element.

Because in order transform the WML 1.3 `do` element the WML 1.3 shadowing algorithm must be applied, the transformation of the `do` element is the most complex. It takes two templates.

The resulting document is a valid WML 2.0 document with the XHTML and WML namespaces and, since the document may be part of a larger WAP 1 service, the 'use-xml-fragment' [WML2] attribute set to *false*. It also gets the WML 2.0 document type declaration before the root element.

The transformation sheet uses an auxiliary Java class (*org.wapforum.wml.WMLExpression*) in order to extract WML variables from text.

6.1. XSLT Transformation Sheet

This transformation sheet is available at

<http://www.wapfourm.org/xslt/wap-244-wmltr.xsl>

<!--

@Wireless Application Protocol Forum, Ltd. 2001.

Terms and conditions of use are available from the
Wireless Application Protocol Forum Ltd.

Web site (<http://www.wapforum.org/what/copyright.htm>).

-->

<!--

This XSLT transformation sheet transforms valid WML 1.3 and WTA-WML 1.2 documents into valid WML 2.0 documents.

Support for XSLT 1.1 is required.

-->

```
<xsl:transform
  version="1.1 "
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform "
  xmlns:wml="http://www.wapforum.org/2001/wml"
  xmlns:wml:expr="http://www.wapforum.org/2001/wml-expression "
  exclude-result-prefixes="wml:expr"
  xmlns="http://www.w3.org/1999/xhtml"
>
```

<!-- NOTE: This requires support for XSLT 1.1. <http://www.w3.org/TR/xslt11/>-->

```
<xsl:script
  implements-prefix="wml:expr"
  language="java"
```

```

    src="java:org.wapforum.wml.WMLExpression"
  />

<xsl:template match="select">
  <select>
    <xsl:copy-of select="@id | @class | @xml:lang | @title | @name | @tabindex" />

    <xsl:if test="boolean(@multiple='true')">
      <xsl:attribute name="multiple">multiple</xsl:attribute>
    </xsl:if>

    <xsl:if test="boolean(@iname)" >
      <xsl:attribute name="wml:iname">
        <xsl:value-of select="@iname" />
      </xsl:attribute>
    </xsl:if>

    <xsl:if test="boolean(@ivalue)" >
      <xsl:attribute name="wml:ivalue">
        <xsl:value-of select="@ivalue" />
      </xsl:attribute>
    </xsl:if>

    <xsl:if test="boolean(@value)" >
      <xsl:attribute name="wml:value">
        <xsl:value-of select="@value" />
      </xsl:attribute>
    </xsl:if>

    <xsl:if test="boolean(@name)" >
      <xsl:attribute name="wml:name">
        <xsl:value-of select="@name" />
      </xsl:attribute>
    </xsl:if>

    <xsl:apply-templates select="option|optgroup" />
  </select>
</xsl:template>

<xsl:template match="optgroup">
  <optgroup>
    <xsl:copy-of select="@id | @class | @title | @xml:lang" />
    <xsl:attribute name="label"> </xsl:attribute>
    <!--NOTE: Nested optgroup elements are ignored-->
    <xsl:apply-templates select="//option" />
  </optgroup>
</xsl:template>

<xsl:template match="option" >

```

```

<option>
  <xsl:copy-of select="@id | @class | @xml:lang | @title | @value" />

  <xsl:if test="boolean(@onpick)" >
    <xsl:attribute name="wml:onpick">
      <xsl:value-of select="@onpick" />
    </xsl:attribute>
  </xsl:if>

  <xsl:apply-templates select="text() | onevent" />
</option>
</xsl:template>

<xsl:template match="input" >
  <input>
    <xsl:copy-of select="@id | @class | @xml:lang | @title | @type | @size | @maxlength |
@tabindex | @value | @accesskey" />

    <xsl:if test="boolean(@name)" >
      <xsl:attribute name="wml:name">
        <xsl:value-of select="@name" />
      </xsl:attribute>
    </xsl:if>

    <xsl:if test="boolean(@format)" >
      <xsl:attribute name="wml:format">
        <xsl:value-of select="@format" />
      </xsl:attribute>
    </xsl:if>

    <xsl:if test="boolean(@emptyok)" >
      <xsl:attribute name="wml:emptyok">
        <xsl:value-of select="@emptyok" />
      </xsl:attribute>
    </xsl:if>
  </input>
</xsl:template>

<xsl:template match="fieldset" >
  <fieldset>
    <xsl:copy-of select="@id | @class | @xml:lang | @title" />

    <xsl:apply-templates
select="do|input|select|fieldset|text()|em|strong|b|i|u|big|small|br|img|anchor|a|table" />

  </fieldset>
</xsl:template>

<xsl:template match="img">

```

```

<img>
  <xsl:copy-of select="@id | @class | @xml:lang | @height | @width | @align | @hspace |
@vspace | @alt | @src" />

  <xsl:if test="boolean(@localsrc)" >
    <xsl:attribute name="wml:localsrc">
      <xsl:value-of select="@localsrc" />
    </xsl:attribute>
  </xsl:if>

</img>
</xsl:template>

<xsl:template match="a">
  <a>
    <xsl:copy-of select="@*" />
    <xsl:apply-templates select="text() | br | img" />
  </a>
</xsl:template>

<xsl:template match="anchor">
  <wml:anchor>
    <xsl:copy-of select="@id | @class | @xml:lang | @accesskey | @title" />
    <xsl:apply-templates select="text() | br | img | go | prev | refresh" />
  </wml:anchor>
</xsl:template>

<xsl:template match="table">
  <table>
    <xsl:copy-of select="@id | @class | @xml:lang | @title" />

    <xsl:if test="boolean(@columns)" >
      <xsl:attribute name="wml:columns">
        <xsl:value-of select="@columns" />
      </xsl:attribute>
    </xsl:if>

    <xsl:if test="boolean(@align)" >
      <xsl:attribute name="wml:align">
        <xsl:value-of select="@align" />
      </xsl:attribute>
    </xsl:if>

    <xsl:apply-templates select="tr" />
  </table>
</xsl:template>

<xsl:template match="tr">
  <tr>

```

```

        <xsl:copy-of select="@id | @class" />

        <xsl:apply-templates select="td" />

    </tr>
</xsl:template>

<xsl:template match="td" >
    <td>
        <xsl:copy-of select="@*" />

        <xsl:apply-templates select="text()|br|em|strong|b|i|u|big|small|img|anchor|a " />
    </td>
</xsl:template>

<xsl:template match="b">
    <b>
        <xsl:copy-of select="@*" />
        <xsl:apply-templates select="text() | em | strong |b |i |u |big |small | br | img | anchor |a
|table " />
    </b>
</xsl:template>
<xsl:template match="u">
    <u>
        <xsl:copy-of select="@*" />
        <xsl:apply-templates select="text() | em | strong |b |i |u |big |small | br | img | anchor |a
|table " />
    </u>
</xsl:template>
<xsl:template match="i">
    <i>
        <xsl:copy-of select="@*" />
        <xsl:apply-templates select="text() | em | strong |b |i |u |big |small | br | img | anchor |a
|table " />
    </i>
</xsl:template>
<xsl:template match="big">
    <big>
        <xsl:copy-of select="@*" />
        <xsl:apply-templates select="text() | em | strong |b |i |u |big |small | br | img | anchor |a
|table " />
    </big>
</xsl:template>
<xsl:template match="small">
    <small>
        <xsl:copy-of select="@*" />
        <xsl:apply-templates select="text() | em | strong |b |i |u |big |small | br | img | anchor |a
|table " />
    </small>

```

```

</xsl:template>

<xsl:template match="em">
  <em>
    <xsl:copy-of select="@*" />
    <xsl:apply-templates select="text() | em | strong | b | i | u | big | small | br | img | anchor | a
|table" />
  </em>
</xsl:template>

<xsl:template match="strong">
  <strong>
    <xsl:copy-of select="@*" />
    <xsl:apply-templates select="text() | em | strong | b | i | u | big | small | br | img | anchor | a
|table" />
  </strong>
</xsl:template>

<xsl:template match="br">
  <br>
    <xsl:copy-of select="@*" />
  </br>
</xsl:template>

<xsl:template match="pre">
  <pre>
    <xsl:copy-of select="@id | @class"/>
    <xsl:apply-templates select="text() | a | anchor | do | u | br | i | b | em | strong | input |
select" />
  </pre>
</xsl:template>

<xsl:template match="p">
  <p>
    <xsl:copy-of select="@id | @class | @xml:lang " />

    <xsl:if test="boolean(@mode)" >
      <xsl:attribute name="wml:mode">
        <xsl:value-of select="@mode" />
      </xsl:attribute>
    </xsl:if>

    <xsl:if test="boolean(@align)" >
      <xsl:attribute name="align">
        <xsl:value-of select="@align" />
      </xsl:attribute>
    </xsl:if>
  </p>

```

```

    <xsl:apply-templates
select="do|input|select|fieldset|text()|em|strong|b|i|u|big|small|br|img|anchor|a|table" />
  </p>
</xsl:template>

<xsl:template match="onevent" >
  <wml:onevent>
    <xsl:choose>
      <xsl:when test="starts-with(@type, 'on')">
        <xsl:attribute name="type"><xsl:value-of select="substring-after(@type, 'on')"/></xsl:attribute>
      </xsl:when>
      <xsl:otherwise>
        <xsl:attribute name="type"><xsl:value-of select="@type" /></xsl:attribute>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:copy-of select="@id | @class" />
    <xsl:apply-templates select="prev | noop | refresh | go" />
  </wml:onevent>
</xsl:template>

<xsl:template name="DO">
  <wml:do>
    <xsl:choose>
      <xsl:when test="@type='accept'">
        <xsl:attribute name="role">positive</xsl:attribute>
      </xsl:when>
      <xsl:when test="@type='prev'">
        <xsl:attribute name="role">back</xsl:attribute>
      </xsl:when>
      <xsl:when test="@type='help'">
        <xsl:attribute name="role">help</xsl:attribute>
      </xsl:when>
      <xsl:when test="@type='options'">
        <xsl:attribute name="role">options</xsl:attribute>
      </xsl:when>
      <xsl:when test="@type='reset' or @type='delete'">
        <xsl:attribute name="role">negative</xsl:attribute>
      </xsl:when>
      <xsl:otherwise>
        <xsl:attribute name="role"><xsl:value-of select="@type"/></xsl:attribute>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:copy-of select="@id | @class | @xml:lang" />
    <xsl:if test="@label">
      <xsl:element name="wml:widget">
        <xsl:choose>
          <xsl:when test="function-available('wml:expr:WMLExpression')">
            <xsl:value-of select="wml:expr:WMLExpression(string(@label))" />
          </xsl:when>
        </xsl:choose>
      </xsl:element>
    </xsl:if>
  </wml:do>
</xsl:template>

```

```

                disable-output-escaping="yes" />
            </xsl:when>
            <xsl:otherwise>
                <xsl:copy-of select="." />
            </xsl:otherwise>
        </xsl:choose>
    </xsl:element>
</xsl:if>
<xsl:apply-templates select="prev|noop|go|refresh" />
</wml:do>
</xsl:template>

<xsl:template match="do" >

    <xsl:param name="shadow" select="" />
    <xsl:choose>
        <xsl:when test="@name">
            <xsl:if test="not(contains(string($shadow), @name) )">
                <xsl:call-template name="DO" />
            </xsl:if>
        </xsl:when>
        <xsl:otherwise>
            <xsl:if test="not(contains(string($shadow), @type) )">
                <xsl:call-template name="DO" />
            </xsl:if>
        </xsl:otherwise>
    </xsl:choose>
</xsl:template>

<xsl:template match="timer">
    <wml:timer>
        <xsl:copy-of select="@*" />
    </wml:timer>
</xsl:template>

<xsl:template match="meta">
    <meta>
        <xsl:copy-of select="@http-equiv | @name | @content | @scheme" />

        <xsl:if test="boolean(@forua) " >
            <xsl:attribute name="wml:forua">
                <xsl:value-of select="@forua" />
            </xsl:attribute>
        </xsl:if>
    </meta>
</xsl:template>

<xsl:template match="access">
    <wml:access>

```

```

    <xsl:copy-of select="@*" />
  </wml:access>
</xsl:template>

<xsl:template match="wml | wta-wml">
  <html
    wml:use-xml-fragments="false">

    <xsl:call-template name="Template.events" />

    <!-- NOTE: Ignore id and class on html and head and template element -->

    <head>
      <!-- NOTE: Use title of first card as document Title -->
      <title>
        <xsl:value-of select="card[position()=1]/@title" />
      </title>
      <xsl:apply-templates select="head/access | head/meta" />
    </head>

    <xsl:apply-templates select="template/onevent | card" />

  </html>
</xsl:template>

<xsl:template match="card">

  <wml:card>
    <xsl:copy-of select="@id | @class | @xml:lang | @title | @onenterforward |
@onenterbackward | @ontimer | @newcontext"/>

    <xsl:apply-templates select="onevent | timer | p | pre | do" />

    <xsl:if test="not(p | pre)">
      <p/>
    </xsl:if>

    <!-- Get non-shadowed template DO elements
      Create a space separated list of shadowing tokens (either name or type attribute)
    -->
    <xsl:apply-templates select="//template/do" >
      <xsl:with-param name="shadow">
        <xsl:for-each select="do">
          <xsl:choose>
            <xsl:when test="@name">
              <xsl:value-of select="@name" />
            </xsl:when>
            <xsl:otherwise>

```

```

        <xsl:value-of select="@type" />
      </xsl:otherwise>
    </xsl:choose>
    <xsl:text> </xsl:text>
  </xsl:for-each>
</xsl:with-param>
</xsl:apply-templates>

</wml:card>

</xsl:template>

<xsl:template name="Template.events">

  <xsl:if test="template/@ontimer">
    <xsl:attribute name="wml:ontimer">
      <xsl:value-of select="template/@ontimer"/>
    </xsl:attribute>
  </xsl:if>

  <xsl:if test="template/@onenterforward">
    <xsl:attribute name="wml:onenterforward">
      <xsl:value-of select="template/@onenterforward"/>
    </xsl:attribute>
  </xsl:if>

  <xsl:if test="template/@onenterbackward">
    <xsl:attribute name="wml:onenterbackward">
      <xsl:value-of select="template/@onenterbackward"/>
    </xsl:attribute>
  </xsl:if>

</xsl:template>

<xsl:template match="go">
  <wml:go>
    <xsl:copy-of select="@id | @class | @href | @sendreferer | @method | @cache-control
| @accept-charset" />

    <xsl:choose>
      <xsl:when test="@enctype='multipart/form-data'">
        <xsl:attribute name="enctype">application/vnd.wap.wml.form.data</xsl:attribute>
      </xsl:when>
      <xsl:otherwise>
        <xsl:attribute
name="enctype">application/vnd.wap.wml.form.urlencoded</xsl:attribute>
      </xsl:otherwise>
    </xsl:choose>
  </wml:go>
</xsl:template>

```

```

        <xsl:apply-templates select="postfield | setvar" />
    </wml:go>
</xsl:template>

<xsl:template match="postfield">
    <wml:postfield>
        <xsl:copy-of select="@*" />
    </wml:postfield>
</xsl:template>

<xsl:template match="prev">
    <wml:prev>
        <xsl:copy-of select="@*" />
        <xsl:apply-templates select="setvar" />
    </wml:prev>
</xsl:template>

<xsl:template match="noop">
    <wml:noop>
        <xsl:copy-of select="@*" />
    </wml:noop>
</xsl:template>

<xsl:template match="refresh">
    <wml:refresh>
        <xsl:copy-of select="@*" />
        <xsl:apply-templates select="setvar" />
    </wml:refresh>
</xsl:template>

<xsl:template match="setvar">
    <wml:setvar>
        <xsl:copy-of select="@*" />
    </wml:setvar>
</xsl:template>

<xsl:template match="text()">

    <xsl:choose>
        <xsl:when test="function-available('wml:expr:WMLExpression')">
            <xsl:value-of select="wml:expr:WMLExpression(string(.))" disable-output-
escaping="yes" />
        </xsl:when>
        <xsl:otherwise>
            <xsl:copy-of select="." />
        </xsl:otherwise>
    </xsl:choose>

```

```
</xsl:template>
```

```
<xsl:output
  method="xml"
  indent="yes"
  doctype-system="wml20.dtd"
  doctype-public="-//WAPFORUM//DTD WML 2.0//EN"
  media-type="application/wml+xml"
/>
```

```
</xsl:transform>
```

6.2. The org.wapforum.wml.WMLExpression Java Class

```
/*
  @Wireless Application Protocol Forum, Ltd. 2001.

  Terms and conditions of use are available from the Wireless Application Protocol Forum
  Ltd.
  Web site (http://www.wapforum.org/what/copyright.htm).
```

```
*/
package org.wapforum.wml;

public class WMLExpression {

  public static final String WMLExpression (String text)
  {
    StringBuffer str = new StringBuffer(); //Return string
    StringBuffer varname = new StringBuffer(); //Variable name
    String conv = null; //Conversion attrib.
    String startvar = null; //either $( or $

    int len = text.length();
    int i = 0;
    char ch;
    while (i < len) {
      ch = text.charAt(i);
      switch (ch) {
        case '<':
          str.append("&lt;");i++;
          break;
        case '&':
          str.append("&amp;");i++;
          break;
        case '$':
          i++;
          if (i == len ) {
            str.append('$');
          }
        }
      }
    }
  }
}
```

```

        return str.toString();
    }

    startvar = "$";
    conv = null;

    ch = text.charAt(i);
    switch(ch) {
        case '$':
            //Escaped $
            str.append('$');
            i++; //Skip one $
            if (i == len)
                return str.toString();
            break;
        case '(':
            //Starts with $(
            startvar = "$(";
            i++;
            if (i == len)
                return str.append(startvar).toString();
            ch = text.charAt(i);
            //Fall through
        default:
            //read var name
            varname = new StringBuffer();
            if ( !(Character.isLetter(ch) | ch == '_' ) )
            { //Invalid character
                str.append(startvar).append(ch);
                i++;
                break;
            }
            varname.append (ch);
            i++;
            for (;;)
            { //Loop to read varname
                if (i >= len) {
                    if (!startvar.endsWith("("))
                        return str.append(createGetvarElement(varname,
conv)).toString();
                    else
                        return str.append(startvar).append(varname).toString();
                }

                ch = text.charAt(i);

                if (ch == ':')

```

```

    { //Conversion

        if (!startvar.endsWith("(") ) {
            //Invalid if variable did not start with $(
            str.append(startvar).append(varname) ;
            break;
        }

        i++;//Skip ':'
        int lpar = text.indexOf(")",i);
        if (lpar != -1) {
            conv = text.substring(i,lpar).toLowerCase();
            if ( !( conv.equals("noesc") | conv.equals("unesc") |
conv.equals("escape"))) {
                str.append(startvar).append(varname).append(':');
                break;
            }
        } else {
            str.append(startvar).append(varname).append(':');
            break;
        }
        i += conv.length() ;//Skip escape directive

        if (i >= len ) {
            return
str.append(startvar).append(varname).append(conv).toString();
        }
        ch = text.charAt(i);
        if (ch != ')') {

            str.append(startvar).append(varname).append(conv).append(ch);
            break;
        }
        i++;//Skip ')'

        str.append (createGetvarElement(varname, conv));
        break;
    } else if (!(Character.isLetterOrDigit(ch) | (ch == '_')))
    {
        if (startvar.endsWith("(") ) {
            //Variable must end with )
            if ( ch == ')')
                i++;
            else {
                str.append(startvar).append(varname);
                break;
            }
        }
    }
    str.append (createGetvarElement(varname, conv));

```

```
                break;
            }

            varname.append(ch);
            i++;
        }
        break;
    }
    break;
default:
    str.append (ch);i++;
    break;
}
}
return str.toString();
}

static String createGetvarElement(StringBuffer varname, String conv)
{
    String getvar1 = "<wml:getvar name=\"";
    String getvar2 = "\" conversion=\"";
    String getvar3 = "\" />";

    StringBuffer getvar = new StringBuffer();

    getvar.append(getvar1);
    getvar.append(varname);
    if (conv != null) {
        getvar.append(getvar2);
        getvar.append(conv);
    }
    getvar.append (getvar3);

    return getvar.toString();
}
}
```

7. Using the XSLT Transformation Sheet (Informative)

The transformation rules are defined as an XSLT transformation sheet, so any of the XSLT processors available on the Web can be used to transform WML 1 documents into WML 2 documents. The Web sites www.xml.com and www.xmlhack.com have references to XSLT resources.

Since the transformation sheet in section 6 relies on the `xsl:script` element, in order to call the Java class extension function, an XSLT version 1.1 compliant processor is required. For example, the SAXON (users.iclway.co.uk/mhkay/saxon/) processor can be used.

It is possible to use an XSLT version 1.0 processor as well. Most XSLT 1.0 processors provide a way to call extension functions such as Java classes. Refer to the documentation. For example, the XT (www.jclark.com/xml) processor and the Microsoft MSXML (www.microsoft.com/xml) can be used.

Appendix A. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [CREQ].

Item	Function	Reference	Status	Requirement
WMLTR-1	WML Transformation Processor	5	M	

Appendix B. Change History (Informative)

Type of Change	Date	Section	Description
Class 0	21-June-2001		The initial version of this document.