Enabler Test Report Browsing v2.1/v2.2

OMA Test Fest (Mar 2006)
Version 05-Apr-2006

Open Mobile Alliance
OMA-Enabler_Test_Report-Browsing-21_22-20060405

Contents

1. Scope

This report describes the results from the testing carried out at OMA TestFest14 March 2006 concerning Browsing version 2.1/2.2.

2. References

2.1 Normative References

[OMAIOPPROC]	OMA Interoperability Policy and Process, http://www.openmobilealliance.org/
[XHTMLMP]	"XHTML Mobile Profile 1.1". OMA-WAP-XHTMLMP-V1_1-20040816-C http://www.openmobilealliance.org/
[WCSS]	"Wireless CSS Specification Version 1.1", http://www.openmobilealliance.org/
[WML1]	"Wireless Markup Language Version 1.3", Open Mobile Alliance™. WAP-191-WML. URL:http://www.openmobilealliance.org/
[ESMP]	"WAP - ECMAScript Specification". Open Mobile Alliance™. OMA-ESMP-V1_0_0-20040816-C.
[EPTR]	Enabler Product Test Report
[ETP]	Enabler Test Plan
[XHML_ETS]	Enabler Test Specification for XHTML 1.1 Approved Version 1.1, 18th November 2004
[WCSS_ETS]	Enabler Test Specification for WCSS 1.1 Approved Version 1.1, 5th January 2006
[WML1_ETS]	Enabler Test Specification for WML 1.3 Approved Version 1.3, 18th Novemer 2004
[ESMP_ETS]	Enabler Test Specification for ESMP 1.0 Approved Version 1.0, 18th Novemer 2004
[WMLS]	Enabler Test Specification for WML Script Language OMA-ETS-WMLS-20041118-A.doc
[WMLSLib]	Enabler Test Specification for WML Script Standard Libraries OMA-ETS-WMLSStdLib-V1_0-20041118-A.doc
[WML2]	Enabler Test Specification for WML 2.0 OMA-ETS-WML-V2_0-20041118-A.doc

2.2 Informative References

[OMADICT]	Dictionary for OMA Specification, OMA-Dictionary http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

This is an informative document, i.e. the document does not intend to contain normative statements.

3.2 Definitions

none

3.3 Abbreviations

ECMA	European Computers Manufacturers Association
ESMP	ECMAScript Mobile Profile
ETR	Enabler Test Requirements
ETS	Enabler Test Specification
SCR	Static Conformance Statement
UAProf	User Agent profile
WCSS	Wireless Cascading Style Sheets
WML	Wireless Markup Language
XHTML	Extensible HyperText Markup Language
ECMA	European Computers Manufacturers Association

4. Summary

This report gives details of the testing carried out during the OMA TestFest12 (January 2006) for Browsing v2.1/v2.2.

The report is compiled on behalf of OMA by the OMA Trusted Zone.

The work and reporting has followed the OMA IOP processes and policies [OMAIOPPROC].

5. Test Details

5.1 Documentation

This chapter lists the details of the enabler and any documentation, tools or test suites used to prove the enabler.

Date:	20th – 28th March 2006
Location:	Montréal, Canada
Enabler:	Browsing v2.1/v2.2
Process:	OMA Interoperability Policy and Process [OMAIOPPROC]
Type of Testing	Interoperability Testing
Products tested:	Client-to-server, Client-to-Client
Test Plan:	-
Test Specification:	OMA-ETS-XHTML-V1_1-20041118-A OMA-ETS-WCSS-(Browsing2.2)-V1_1-20060105-A OMA-ETS-WML-V1_3-20041118-A OMA-ETS-WML-V2_0-20041118-A OMA-ETS-ESMP-V1_0-20041118-A OMA-ETS-WMLS-V1_0-20041118-A OMA-ETS-WMLStdLib-V1_0-20041118-A
Test Tool:	None
Test Code:	None
Type of Test event:	TestFest
Participants:	Openwave Systems + *two other participants*
Number of Client Products:	3
Participating Technology Providers for clients:	Openwave Systems + *two other participants*
Number of Server Products:	n/a
Participating Technology Providers for servers:	n/a
Number of test sessions completed:	3 of 3

5.2 Test Case Statistics

5.2.1 Test Case Summary

This chapter gives an overview of the result for all test cases included in [ETS].

The following status is used in the tables below:

- **Total number of TCs**: Used in the summary to indicate how many test cases there are in total.
- **Number of passed:** Used in the summary to indicate how many of the total test cases successfully passed.
- **Number of failed:** Used in the summary to indicate how many of the total test cases failed.
- **Number of N/A:** Used in the summary to indicate how many of the total test cases have not been run due to one of the implementations not supporting the functionality required to run this test case.
- **Number of OT:** Used in the summary to indicate how many of the total test cases have not been run due to no time to run the test case.
- **Number of INC:** Used in the summary to indicate how many of the total test cases have not been run due to functionality not being tested due to an error in the implementation or other functionality that is required to run this test case.

Test Section:	**Number of test sessions:**	**Total number of TCs:**	**Number of Passed:**	**Number of Failed:**	**Number of N/A:**	**Number of OT:**	**Number of INC:**	**Total:**
xHTML v1.1 (-int-)	3	11	19	2	0	0	12	33
xHTML v1.1 (-con-)	3	20	42	4	2	0	12	60
WCSS v1.1 (-int-)	3	23	62	7	0	0	0	69
WCSS v1.1 (-con-)	3	36	69	23	7	0	9	108
ESMP v1.0 (-int-)	3	16	11	10	0	9	18	48
ESMP v1.0 (-con-)	3	68	35	38	0	51	80	204
WML 1.3	3	201	561	42	0	0	0	603
WML 2.0	3	336	283	48	0	5	672	1008
WAE-2.2	3	34	0	0	0	0	102	102
WMLS v1.0	3	148	437	7	0	0	0	444
WMLSLib v1.0	3	235	682	23	0	0	0	705
Total	**3**	**1128**	**2201**	**204**	**9**	**65**	**905**	**3384**

Table 1. Test Summary Table

5.2.2 Test Case List

This chapter lists the statistics for all test cases included in [ETS].

The following status is used in the tables below:

- **No. of runs(R):** Used to indicate how many times the test cases have been run in total.
- **No. of passed(P):** Used to indicate how many times the test case has been run with successful result.
- **No. of failed(F):** Used to indicate how many times the test case has been run with failed result
- **No. of OT(O):** Used to indicate how many times the test case has not been run due to no time available.
- **No. of INC(I):** Used to indicate how many times the test case has not been run due to errors being found in other functionality required for running this test case.
- **PR:** Used to indicate if any PRs (Problem Reports) have been issued during testing.
- **Note:** Used to indicate the cause of Inconclusive or Fail verdicts.

Tests for Browsing Enabler TestFest From ETSs

Test Case:	Test Case Description:	R	P	F	O	I	PR:	Note:
xHTML-1.1-int-1	To test basic form Module. Test input element with type set to text but with the illegal attribute checked set		0	0	0	3	54	
xHTML-1.1-int-2	To test *optgroup* element in Forms module		3	0	0	0		
xHTML-1.1-int-3	To test Basic Tables		0	0	0	3	80	
xHTML-1.1-int-4	To test *hr* element in Presentation module		0	0	0	3	47	
xHTML-1.1-int-5	To test base Module. The base element specifies a base URL. The purpose is to check that this base URL is used in a correct way		2	1	0	0	81	
xHTML-1.1-int-6	To test base Module. The base element specifies a base URL. The purpose is to check that this base URL is used in a correct way when linking to a WML page		2	1	0	0	81	
xHTML-1.1-int-7	To test if the interaction between WML and XHTML is handled correctly		3	0	0	0		
xHTML-1.1-int-8	To test if the interaction between WML and XHTML is handled correctly using links		0	0	0	3	49	
xHTML-1.1-int-9	To test List Module		3	0	0	0		

xHTML-1.1-int-10	To test that none terminated tags generate error messages		3	0	0	0		
xHTML-1.1-int-11	To test that tags terminated in wrong order generate error messages		3	0	0	0		
xHTML-1.1-con-1	To test Structure Module.		2	1	0	0		
xHTML-1.1-con-2	To test Text Module.		0	1	0	0		
xHTML-1.1-con-3	To test Hypertext Module.		2	1	0	0		
xHTML-1.1-con-4	To test List Module.		3	0	0	0		
xHTML-1.1-con-5	To test Basic Forms.		3	0	0	0		
xHTML-1.1-con-6	To test Basic Tables.		0	0	0	3	80	
xHTML-1.1-con-7	To test Image Module.		3	0	0	0		
xHTML-1.1-con-8	To test object Module. The Object Module is ignored, but objects with an image, which is possible to show, shall work as an image tag.		0	0	0	3	55	
xHTML-1.1-con-9	The Metainformation Module is ignored. This test checks that the element is ignored in a correct way..		0	0	0	3	56	
xHTML-1.1-con-10	To test Link Module.		3	0	0	0		
xHTML-1.1-con-11	To test Base Module.		2	1	0	0	81	
xHTML-1.1-con-12	To test *fieldset* element in Forms module.		3	0	0	0		
xHTML-1.1-con-13	To test *optgroup* element in Forms module.		3	0	0	0		
xHTML-1.1-con-14	To test *start* element on *ol*.		3	0	0	0		
xHTML-1.1-con-15	To test *value* attribute on *li*.		3	0	0	0		
xHTML-1.1-con-16	To test *b* element in Presentation module.		3	0	0	0		
xHTML-1.1-con-17	To test *big* element in Presentation module.		3	0	0	0		
xHTML-1.1-con-18	To test *hr* element in Presentation module..		0	0	0	3	47	
xHTML-1.1-con-19	To test *I* element in Presentation module.		3	0	0	0		
xHTML-1.1-con-20	To test *Small* element in Presentation module..		3	0	0	0		
WCSS-1.1-int-1	To test Cascade and Inheritance		3	0	0	0		
WCSS-1.1-int-2	Supported Media		2	1	0	0		
WCSS-1.1-int-3	Unsupported Media		3	0	0	0		

WCSS-1.1-int-4	To test Margin Shorthand		3	0	0	0		
WCSS-1.1-int-5	To test Padding Shorthand		3	0	0	0		
WCSS-1.1-int-6	To test Border Shorthand		3	0	0	0		
WCSS-1.1-int-7	To test Color property		3	0	0	0		
WCSS-1.1-int-8	To test Color property(2)		3	0	0	0		
WCSS-1.1-int-9	Background color		3	0	0	0		
WCSS-1.1-int-10	To test when the WHITE-SPACE property is set to 'pre'		1	2	0	0		
WCSS-1.1-int-11	To test when the TEXT-DECORATION property is set to 'underline' and/or 'blink'		3	0	0	0		
WCSS-1.1-int-12	To test Images as Markers		3	0	0	0		
WCSS-1.1-int-13	To test when VISIBILITY property is set to hidden		3	0	0	0		
WCSS-1.1-int-14	To test when the DISPLAY property is set to 'none'		2	1	0	0		
WCSS-1.1-int-15	To test when the CLEAR property is set to 'left'		2	1	0	0		
WCSS-1.1-int-16	To test when the VERTICAL-ALIGN property is set to 'super'.		3	0	0	0		
WCSS-1.1-int-17	To test style slide		3	0	0	0		
WCSS-1.1-int-18	To test when the -wap-marquee-dir property value is set to 'ltr'		3	0	0	0		
WCSS-1.1-int-19	To test margue speeds		3	0	0	0		
WCSS-1.1-int-20	To test when Space and Comma are separated		3	0	0	0		
WCSS-1.1-int-21	To test Escape characters		3	0	0	0		
WCSS-1.1-int-22	To test Alignment		3	0	0	0		
WCSS-1.1-int-23	To test shine through		1	2	0	0		
WCSS-1.1-con-1	To test pattern matching.		2	1	0	0		
WCSS-1.1-con-2	To test syntax and parsing.		0	0	0	3	84	
WCSS-1.1-con-3	To test data types.		1	2	0	0		
WCSS-1.1-con-4	To test assigning property values, cascading and inheritance.		1	2	0	0		
WCSS-1.1-con-5	To test media types.		3	0	0	0		
WCSS-1.1-con-6	To test associating style sheets with XML documents.		0	0	0	0		

WCSS-1.1-con-7	To test pattern matching.		3	0	0	0		
WCSS-1.1-con-8	To test padding properties.		3	0	0	0		
WCSS-1.1-con-9	To test border width.		3	0	0	0		
WCSS-1.1-con-10	To test border colour.		3	0	0	0		
WCSS-1.1-con-11	To test border style.		3	0	0	0		
WCSS-1.1-con-12	To test border shorthand property		3	0	0	0		
WCSS-1.1-con-13	To test foreground colour.		3	0	0	0		
WCSS-1.1-con-14	To test background colour.		3	0	0	0		
WCSS-1.1-con-15	To test background images.		0	0	0	3	89	
WCSS-1.1-con-16	To test background shorthand property.		3	0	0	0		
WCSS-1.1-con-17	To test font family.		3	0	0	0		
WCSS-1.1-con-18	To test font style.		3	0	0	0	50	
WCSS-1.1-con-19	To test font variant.		1	2	0	0		
WCSS-1.1-con-20	To test font weight.		3	0	0	0		
WCSS-1.1-con-21	To test font size.		3	0	0	0		
WCSS-1.1-con-22	To test font shorthand property.		1	2	0	0		
WCSS-1.1-con-23	To test lists.		2	1	0	0		
WCSS-1.1-con-24	To test text indentation.		3	0	0	0		
WCSS-1.1-con-25	To test text alignment.		1	0	0	0		
WCSS-1.1-con-26	To test text decoration.		3	0	0	0		
WCSS-1.1-con-27	To test text transformation.		0	0	0	3	51	
WCSS-1.1-con-28	To test white space.		1	2	0	0		
WCSS-1.1-con-29	To test visual effects.		1	0	0	0		
WCSS-1.1-con-30	To test display properties.		2	1	0	0		
WCSS-1.1-con-31	To test float positioning.		1	2	0	0		
WCSS-1.1-con-32	To test float flow control.		1	2	0	0		
WCSS-1.1-con-33	To test content width and height.		2	1	0	0		
WCSS-1.1-con-34	To test CSS Extension: Marquee.		2	1	0	0	88	
WCSS-1.1-con-35	To test CSS Extension: Access keys.		2	1	0	0		
WCSS-1.1-con-36	To test CSS Extension: Input.		0	3	0	0	53	

ESMP-1.0-int-1	To test the support for Inline Script Execution		0	0	0	0		
ESMP-1.0-int-2	To test the support for Deferred Script Execution		2	0	0	0		
ESMP-1.0-int-3	To test font the support for File Based Script Execution		0	2	0	0		
ESMP-1.0-int-4	To test the semicolon support at end of statements		2	0	0	0		
ESMP-1.0-int-5	To test the support for EvalError exceptio		2	0	0	0		
ESMP-1.0-int-6	To test the Native Object Support – String Object		1	1	0	0		
ESMP-1.0-int-7	To test the Native Object Support – Error Object		0	2	0	0		
ESMP-1.0-int-8	To test the support for DOM2 compliant event binding		0	2	0	0		
ESMP-1.0-int-9	To test the Host Object Support – parent global Object		0	2	0	0		
ESMP-1.0-int-10	To test the Host Object Support – navigator Object		0	0	0	2	91	
ESMP-1.0-int-11	To test the Host Object Support – history Object.		0	1	0	1		
ESMP-1.0-int-12	To test the Host Object Support – location Object		0	0	0	2	92	
ESMP-1.0-int-13	To test the UTF-16 Code point support		0	0	0	2	93	
ESMP-1.0-int-14	To test the UTF-8 Code point support		2	0	0	0		
ESMP-1.0-int-15	To test the semicolon support at end of statements		2	0	0	0		
ESMP-1.0-int-16	To test the ECMAScript Language Syntax and semantics		0	0	0	2	57	
ESMP-1.0-con-1	To test the ECMAscript Type Support.		0	0	0	2	57	
ESMP-1.0-con-2	To test the UTF-16 Code point support.		0	0	0	2	93	
ESMP-1.0-con-3	To test the UTF-8 Code point support.		2	0	0	0		
ESMP-1.0-con-4	To test the character indexing support.		1	1	0	0		
ESMP-1.0-con-5	To test the semicolon support at end of statements.		2	0	0	0		

ESMP-1.0-con-6	To test the IEEE 754 64 Bit Float Support with accuracy to at least 14 digits		0	2	0	0		
ESMP-1.0-con-7	To test the ECMAScript Language Syntax and semantics.		0	0	0	2	60	
ESMP-1.0-con-8	To test the support for eval().		2	0	0	0		
ESMP-1.0-con-9	To test the support for EvalError exception.		2	0	0	0		
ESMP-1.0-con-10	To test the support for dynamic function creation.		0	0	0	0		
ESMP-1.0-con-11	To test the support for the with statement.		2	0	0	0		
ESMP-1.0-con-12	To test the support for dynamic modification of built-in objects.		0	0	0	0		
ESMP-1.0-con-13	To test the version properties for all native objects		0	2	0	0		
ESMP-1.0-con-14	To test that all non-native, built-in objects are enumerable.		0	0	0	2	94	
ESMP-1.0-con-15	To test that All native, built-in objects are enumerable.		0	2	0	0		
ESMP-1.0-con-16	To test the Native Object Support – Global Object.		0	2	0	0		
ESMP-1.0-con-17	To test the Native Object Support – Array Object.		0	0	0	2	64	
ESMP-1.0-con-18	To test the Native Object Support – String Object.		1	1	0	0		
ESMP-1.0-con-19	To test the Native Object Support – Regular Expression Object.		0	0	0	2	66	
ESMP-1.0-con-20	To test the Native Object Support – Boolean Object.		2	0	0	0		
ESMP-1.0-con-21	To test the Native Object Support – Number Object.		0	2	0	0		
ESMP-1.0-con-22	To test the Native Object Support – Math Object		0	1	0	1	69	
ESMP-1.0-con-23	To test the Native Object Support – Date Object.		0	2	0	0		
ESMP-1.0-con-24	To test the Native Object Support – Error Object.		0	2	0	0		
ESMP-1.0-con-25	To test the Native Object Support – Object Creation.		2	0	0	0		
ESMP-1.0-con-26	To test the Native Object Support – Function Object - dynamic function construction.		0	0	0	0		

ESMP-1.0-con-27	To test the support for Native Error Types.		0	2	0	0		
ESMP-1.0-con-28	To test the support for Inline Script Execution.		0	0	0	0		
ESMP-1.0-con-29	To test the support for Deferred Script Execution.		2	0	0	0		
ESMP-1.0-con-30	To test font the support for File Based Script Execution.		0	2	0	0		
ESMP-1.0-con-31	To test the abnormal termination error reporting.		2	0	0	0		
ESMP-1.0-con-32	To test the support for Aborted Script completion.		0	1	0	0		
ESMP-1.0-con-33	To test the support for XHTML Events.		2	0	0	0		
ESMP-1.0-con-34	To test the support for DOM2 compliant event binding.		0	2	0	0		
ESMP-1.0-con-35	To test the Host Object Support – parent global Object.		0	2	0	0		
ESMP-1.0-con-36	To test the Host Object Support – navigator Object.		0	0	0	2	91	
ESMP-1.0-con-37	To test the Host Object Support – history Object.		0	2	0	0		
ESMP-1.0-con-38	To test the Host Object Support – location Object.		0	0	0	2	92	
ESMP-1.0-con-39	To test the Host Object Support – Basic document Object.		0	0	0	2	95	
ESMP-1.0-con-40	To test the XML DOM Support.		0	0	0	2	70	
ESMP-1.0-con-41	To test the DOM Object Support – Exception Object.		0	0	0	2	70	
ESMP-1.0-con-42	To test the DOM Object Support – Node Object – Property Support.		0	0	0	2	96	
ESMP-1.0-con-43	To test the DOM Object Support – Node Object – Data Interrogation Methods.		2	0	0	0		
ESMP-1.0-con-44	To test the DOM Object Support – Structural Mutation.		0	0	0	2	97	
ESMP-1.0-con-45	To test the DOM Object Support – Node Object – Structural Modification Mutation Methods.		0	0	0	2	97	
ESMP-1.0-con-46	To test the DOM Object Support – document Object - Data Interrogation Methods.		0	2	0	0		

ESMP-1.0-con-47	To test the DOM Object Support – document Object – Structural Modification Mutation Methods.		1	0	0	0		
ESMP-1.0-con-48	To test the DOM Object Support – NodeLlist Object.		0	0	0	2	98	
ESMP-1.0-con-49	To test the DOM Object Support – Element Object – Property Support.		0	0	0	2	99	
ESMP-1.0-con-50	To test the DOM Object Support – Element Object – Data Interrogation Methods.		0	2	0	0		
ESMP-1.0-con-51	To test the DOM Object Support – Element Object – Data Modification Methods.		0	0	0	2	71	
ESMP-1.0-con-52	To test the DOM Object Support – Element Object – Structural Modification Mutation Methods.		0	0	0	0		
ESMP-1.0-con-53	To test the DOM Object Support – Text Object.		0	0	0	2	100	
ESMP-1.0-con-54	To test the XHTML DOM Support.		0	0	0	2	76	
ESMP-1.0-con-55	To test the XHTML DOM Support – XHTML Document Object.		2	0	0	0	75	
ESMP-1.0-con-56	To test the XHTML DOM Support – Link Object.		0	0	0	2	75	
ESMP-1.0-con-57	To test the XHTML DOM Support – Image Object.		0	0	0	2	73	
ESMP-1.0-con-58	To test the XHTML DOM Support – Form Object.		0	2	0	0		
ESMP-1.0-con-59	To test the XHTML DOM Support – Text Input Object.		0	2	0	0		
ESMP-1.0-con-60	To test the XHTML DOM Support – Textarea Input Object.		0	0	0	2	72	
ESMP-1.0-con-61	To test the XHTML DOM Support – Password Input Object.		0	0	0	2	74	
ESMP-1.0-con-62	To test the XHTML DOM Support – Radio Input Object.		2	0	0	0		
ESMP-1.0-con-63	To test the XHTML DOM Support – Checkbox Input Object.		2	0	0	0		
ESMP-1.0-con-64	To test the XHTML DOM Support – Submit Object.		0	0	0	2	78	
ESMP-1.0-con-65	To test the XHTML DOM Support – Reset Object.		2	0	0	0		
ESMP-1.0-con-66	To test the XHTML DOM Support – Select Element Object.		0	2	0	0		

ESMP-1.0-con-67	To test the XHTML DOM Support – Option Element Object.		2	0	0	0		
ESMP-1.0-con-68	To test the XHTML DOM Support – Screen Object.		0	0	0	2	78	
WML-1.3-con-1	When a URL does not include a fragment identifier (a hash mark followed by a name), the URL references a deck.		2	1	0	0		
WML-1.3-con-2	When a URL contains a fragment identifier (a hash mark followed by a name), the URL references an individual card within a WML deck.		3	0	0	0		
WML-1.3-con-3	When a relative URL is encountered, it is resolved against the Base URL of the deck.		3	0	0	0		
WML-1.3-con-4	When no character encoding is specified, the default character encoding can be assumed (UTF-8).		3	0	0	0		
WML-1.3-con-5	When a named character entity is encountered (e.g. &), its associated element from the document character set replaces the character entity in the text.		3	0	0	0		
WML-1.3-con-6	When a decimal numeric character entity is encountered, its associated element from the document character set replaces the character entity in the text.		3	0	0	0		
WML-1.3-con-7	When a hexadecimal numeric character entity is encountered, its element from the document character set replaces the character entity in the text.		3	0	0	0		
WML-1.3-con-8	When a numeric character entity is enountered, it is translated into the corresponding character in the document character set.		3	0	0	0		
WML-1.3-con-9	When a named character entity is enountered, it is translated into the corresponding character in the document character set.		3	0	0	0		
WML-1.3-con-10	When an element is declared EMPTY in the WML DTD, it may be specified using the shorthand notation <ELEMENT/>. User agents shall treat documents that use this convention and are otherwise conforming as well formed.		3	0	0	0		

WML-1.3-con-11	When an element is not declared EMPTY in the WML DTD, and such an element has no contents, it may be specified as <ELEMENT/> or as <ELEMENT></ELEMENT>. User agents shall treat documents that use either convention and are otherwise conforming as well formed.		3	0	0	0		
WML-1.3-con-12	Element attribute values must be enclosed in quotation marks. When an attribute value is enclosed in single quotation marks ('), the value may contain double-quotation marks (").		3	0	0	0		
WML-1.3-con-13	Element attribute values must be enclosed in quotation marks. When an attribute value is enclosed in double quotation marks ("), the value may contain single quotation marks (').		3	0	0	0		
WML-1.3-con-14	Element attribute values may contain character entities. Entity transformation into the document character set shall not effect the interpretation of quotation marks enclosing attribute values, nor shall it effect the parsing of attribute values.		3	0	0	0		
WML-1.3-con-15	When an XML comment (<!-- comment -->) is included in a deck, the contents of that comment shall not be presented to the user.		3	0	0	0		
WML-1.3-con-16	When the symbol "$", followed by an identifier, is encountered in a %PCDATA or %vdata context, it is a variable reference. The variable reference is replaced with the value of the variable thus named.		3	0	0	0		
WML-1.3-con-17	When the symbol "$", followed by an identifier enclosed in parenthesis, is encountered in a %PCDATA or %vdata context, it is a variable reference. The variable reference is replaced with the value of the variable thus named.		3	0	0	0		

WML-1.3-con-18	When the symbol "$", followed by an identifier, a colon ":" and a conversion specifier, enclosed in parenthesis, is encountered in a %PCDATA or %vdata context, it is a variable reference and an indicator as to how the variable value should be converted. The variable reference is replaced with the value of the variable thus named, converted as specified.		3	0	0	0		
WML-1.3-con-19	When the symbol "$$" is encountered in a %PCDATA or %vdata context, it is an escaped dollar-sign. The escaped dollar-sign is replaced with a dollar-sign.		2	1	0	0		
WML-1.3-con-20	When data is enclosed within a CDATA block (<![CDATA[data]] >), the data is handled as literal text and presented to the user "as is".		3	0	0	0		
WML-1.3-con-21	When a processing instruction (<? instruction ?>) is encountered, it is ignored unless it is defined in the XML 1.0 Recommendation.		3	0	0	0		
WML-1.3-con-22	The NAME attribute of the POSTFIELD element defines the name of a field, and the VALUE attribute defines the value to associate with that NAME. When a GO task is executed the name and value of the field are sent to the server.		3	0	0	0		
WML-1.3-con-23	When a SETVAR element contains a NAME attribute that evaluates to a legal variable name, the value of that variable (in the current browser context) is set to the value of the VALUE attribute.		3	0	0	0		
WML-1.3-con-24	When the HREF attribute of the GO element specifies a URL that names a WML card or deck, prior to displaying the card (or deck), the location of the current card is pushed onto the history stack.		2	1	0	0		
WML-1.3-con-25	When the SENDREFERER attribute of the GO element is set to true, the user agent must send the URL of the deck containing this task to the server. The URL sent must be the smallest relative URL possible if it can be relative at all.		2	1	0	0		
WML-1.3-con-26	When the METHOD attribute of the GO element is set to GET, it defines the submission method for the request.		3	0	0	0		

WML-1.3-con-27	When the METHOD attribute of the GO element is not set, the submission method for the request defaults to GET.		3	0	0	0		
WML-1.3-con-28	When the SENDREFERER attribute of the GO element is set to "false" or is not set, the user agent must not send the URL of the deck containing this task to the server.		3	0	0	0		
WML-1.3-con-29	When the METHOD attribute of the GO element is set to POST, it defines the submission method for the request.		2	1	0	0		
WML-1.3-con-30	When the target of a GO element of a card is contained within the current deck the user agent must ignore all postfield elements.		3	0	0	0		
WML-1.3-con-31	When the cache-control attribure is present within the GO element and its value is set to "no-cache" the client must reload the URL from the origin server.		3	0	0	0		
WML-1.3-con-32	When the HREF attribute value of a GO element is an HTTP URI, the METHOD attribute has a value of "post" and the ENCTYPE attribute is "application/x-www-form-urlencoded" the request must be performed according to these attributes.		3	0	0	0		
WML-1.3-con-33	When the ENCTYPE attributes value of a GO element is "APPLICATION/X-WWW-FORM-URLENCODED", the field names and values must be encoded using URI-escaping, and listed in the order in which the postfields are presented.		2	1	0	0		
WML-1.3-con-34	When the ENCTYPE attributes value is "APPLICATION/X-WWW-FORM-URLENCODED", the field names and values must be encoded by the name seperated from the value by "=" and name/value pairs seperated from each other by "&".		2	1	0	0		
WML-1.3-con-35	When the user agent only supports data submission as "APPLICATION/X-WWW-FORM-URLENCODED" content type, it may ignore the ENCTYPE attribute.		3	0	0	0		

WML-1.3-con-36	When the HREF attribute value of a GO element is an HTTP URI, the METHOD attribute has a value of "get" and the ENCTYPE attribute is "application/x-www-form-urlencoded" the request must be performed according to these attributes.		3	0	0	0		
WML-1.3-con-37	When the HREF attribute value of a GO element is an HTTP URI, the METHOD attribute has a value of "post" and the ENCTYPE attribute is "multipart/form-data" the request must be performed according to these attributes.		2	1	0	0		
WML-1.3-con-38	When a PREV element is executed, the card referenced by the previous URI in the history stack is loaded (if necessary) and executed.		3	0	0	0		
WML-1.3-con-39	When a PREV element is executed, the previous URI on the history stack is removed from the stack.		3	0	0	0		
WML-1.3-con-40	When a REFRESH element is executed, the current card is redisplayed.		3	0	0	0		
WML-1.3-con-41	When a REFRESH element is executed, any variables specified through enclosed SETVAR elements are set in the browser context prior to the card being redisplayed.		3	0	0	0		
WML-1.3-con-42	When an event-handling element is specified within a TEMPLATE element, that event-handler is available in each card of the deck.		3	0	0	0		
WML-1.3-con-43	When an event-handling element is specified within a TEMPLATE element, and an event-handling element identifying the same event is specified within a CARD element, and that handler specifies its task as NOOP, there is no event handler for the event within the card. The user agent shall not provide a control for this event in this card.		3	0	0	0		
WML-1.3-con-44	When a card level element binds a noop task and is not shadowed by another element the event will be masked and ignored.		3	0	0	0		
WML-1.3-con-45	When a DO element is specified within a TEMPLATE element, the user agent shall behave as if the element is specified at the end of the CARD element.		3	0	0	0		

WML-1.3-con-46	When a DO element is specified within a TEMPLATE element, and a DO element with the same NAME attribute is specified within a CARD element, the CARD level element overrides the TEMPLATE level element.		3	0	0	0		
WML-1.3-con-47	When a DO element has a NOOP task, it shall not be presented to the user.		2	1	0	0		
WML-1.3-con-48	When a DO element has a task other than NOOP, it shall be presented to the user.		3	0	0	0		
WML-1.3-con-49	When a DO element from a TEMPLATE is overridden by a DO element in a CARD, the DO elements from the TEMPLATE shall not be presented to the user.		3	0	0	0		
WML-1.3-con-50	When a NAME attribute of a DO element is specified, its value names the DO element. All uniquely named, active DO elements in a DECK shall be presented to the user.		3	0	0	0		
WML-1.3-con-51	User agents must accept any TYPE, but may treat any unrecognised type as the equivalent of UNKNOWN.		3	0	0	0		
WML-1.3-con-52	When the OPTIONAL attribute of a DO element is set to "false" or is omitted, the DO event must be presented to the user.		3	0	0	0		
WML-1.3-con-53	When the TYPE attribute of the DO element is "ACCEPT" the user agent must display the event as an option.		3	0	0	0		
WML-1.3-con-54	When the TYPE attribute of the DO element is "PREV" the user agent must display the event as an option.		3	0	0	0		
WML-1.3-con-55	When the TYPE attribute of the DO element is "HELP" the user agent must display the event as an option.		3	0	0	0		
WML-1.3-con-56	When the TYPE attribute of the DO element is "RESET" the user agent must display the event as an option.		3	0	0	0		
WML-1.3-con-57	When the TYPE attribute of the DO element is "DELETE" the user agent must display the event as an option.		3	0	0	0		
WML-1.3-con-58	When the TYPE attribute of the DO element is "OPTIONS" the user agent must display the event as an option.		3	0	0	0		

WML-1.3-con-59	When the TYPE attribute of the DO element is "UNKNOWN" the user agent must display the event as an option.		3	0	0	0		
WML-1.3-con-60	When an ANCHOR element is specified, its contents define a link (via a task element such as GO, PREV, or REFRESH) that is activated when the link is selected.		3	0	0	0		
WML-1.3-con-61	When the HREF attribute of the A element is specified, it defines the URL of a resource to be associated with the items within the A element. Selecting the contents of the A element causes the resource referenced to be loaded.		3	0	0	0		
WML-1.3-con-62	If the event binding element specifies an intrinsic event type which applies to its parent element, it must be ignored by the user agent.		3	0	0	0		
WML-1.3-con-63	When an ONEVENT element occurs NOT immediately within a TEMPLATE or CARD element, and the TYPE attribute of the ONEVENT element is specified using the value "onenterbackward", the ONEVENT element is ignored.		3	0	0	0		
WML-1.3-con-64	When an ONEVENT element occurs immediately within a TEMPLATE or CARD element, and the TYPE attribute of the ONEVENT element is specified using the value "onenterforward", the ONEVENT element defines a task to be executed when the user causes the card to be entered via the GO element (or actions with identical semantics).		3	0	0	0		
WML-1.3-con-65	When an ONEVENT element occurs NOT immediately within a TEMPLATE or CARD element, and the TYPE attribute of the ONEVENT element is specified using the value "onenterforward", the ONEVENT element is ignored.		3	0	0	0		
WML-1.3-con-66	When an ONEVENT element occurs immediately within an OPTION element, and the TYPE attribute of the ONEVENT element is specified using the value "onpick", the ONEVENT element defines a task to be executed when the user selects or deselects the item described by the OPTION element.		3	0	0	0		

WML-1.3-con-67	When an ONEVENT element occurs NOT immediately within an OPTION element, and the TYPE attribute of the ONEVENT element is specified using the value "onpick", the ONEVENT element is ignored.		3	0	0	0		
WML-1.3-con-68	When an ONEVENT element occurs immediately within a TEMPLATE or CARD element, and the TYPE attribute of the ONEVENT element is specified using the value "ontimer", the ONEVENT element defines a task to be executed when the TIMER expires.		3	0	0	0		
WML-1.3-con-69	When an ONEVENT element occurs NOT immediately within a TEMPLATE or CARD element, and the TYPE attribute of the ONEVENT element is specified using the value "ontimer", the ONEVENT element is ignored.		3	0	0	0		
WML-1.3-con-70	The user agent must ignore any ONEVENT element specifying a type that does not correspond to a legal intrinsic event for the immediately enclosing element.		3	0	0	0		
WML-1.3-con-71	When an undefined variable is referenced within any element or attribute with a content type of %vdata, it results in the substitution of the empty string.		3	0	0	0		
WML-1.3-con-72	Variable names are case sensitive.		3	0	0	0		
WML-1.3-con-73	When a variable is referenced within any element or attribute with a content type of %vdata, and that variable reference is enclosed in parenthesis using the pattern '(' name ':' conversion ')', the value of the variable is converted before it is substituted. Legal conversion values are "noesc", "escape", and "unesc" - meaning "no change", "URL-escape the value", and "URL-unescape the value" respectively.		2	1	0	0		
WML-1.3-con-74	When a variable is referenced within any element or attribute with a content type of %vdata, and that variable reference specifies no conversion suffix, the default conversion is context sensitive. In the case of ONPICK="onenterforward", ONPICK="onenterbackward", HREF, and SRC attributes the default is escape conversion.		3	0	0	0		

WML-1.3-con-75	When variables are placed in the text(#PCDATA)of a card or into %vdata and %HREF attributes in WML elements they are replaced with the variables value.		2	1	0	0		
WML-1.3-con-76	When a string substitution occurs, the current value of the variable must not be altered.		3	0	0	0		
WML-1.3-con-77	WML variable names consist of an US-ASCII letter or underscore followed by zero or more letters, digits or underscores. Any other characters are illegal. Any variable containing illegal characters shall not be processed by the encoder. The remainder of the card should be processed and displayed.		2	1	0	0		
WML-1.3-con-78	When a variable is referenced within any element or attribute with a content type of %vdata, and that variable reference specifies no conversion suffix, the default is no conversion.		3	0	0	0		
WML-1.3-con-79	When a variable is referenced within any element or attribute with a content type of %HREF, and that variable reference specifies no conversion suffix, the default is ESCAPE conversion.		3	0	0	0		
WML-1.3-con-80	All XML parsing is complete prior to variable substitution. When a dollar-sign in any form (a character entity, for example) is followed by a variable name, it is interpreted as a variable reference.		3	0	0	0		
WML-1.3-con-81	When 2 dollar-signs in a row are encountered in a deck, they are interpreted as a single literal dollar-sign.		3	0	0	0		
WML-1.3-con-82	When a user commits the input in a SELECT element, the associated variable value is updated.		3	0	0	0		
WML-1.3-con-83	When a user commits the input in an INPUT element, the associated variable value is updated.		3	0	0	0		
WML-1.3-con-84	Whenever a user agent navigates to a resource that was not the result of an interaction with the content in the currentcontext, the user agent must establish a fresh context for that navigation.		0	3	0	0		

WML-1.3-con-85	When an ACCESS element is in the HEAD element of a deck and that deck is entered, the user agent checks to ensure that the new deck can be accessed from the current deck.		3	0	0	0		
WML-1.3-con-86	When the DOMAIN attribute of the ACCESS element (of a new deck) is not set, it defaults to the domain of the current deck.		2	1	0	0		
WML-1.3-con-87	When the PATH attribute of the ACCESS element (of a new deck) is not set, it defaults to the value "/".		3	0	0	0		
WML-1.3-con-88	When the DOMAIN attribute of the ACCESS element (of a new deck) is set to a period separated domain suffix, the domain of the current deck is compared to this suffix - where each element specified must match exactly for access to be granted.		3	0	0	0		
WML-1.3-con-89	When the PATH attribute of the ACCESS element (of a new deck) is specified as an absolute path, it defines the path prefix that is permitted access to the deck - where each element of the prefix must match that of the current deck exactly for access to be granted.		3	0	0	0		
WML-1.3-con-90	When a deck contains more than one ACCESS element the user agent shall display an error, along with a means to continue browser navigation.		3	0	0	0		
WML-1.3-con-91	When a user agent encounters a relative path in an ACCESS element it shall convert it to an absolute path. Then it should perform prefix matching against the PATH attribute		2	1	0	0		
WML-1.3-con-92	When an event binding (e.g. a DO or ONEVENT element) is specified within a TEMPLATE element, that specification applies to all cards in the deck.		3	0	0	0		
WML-1.3-con-93	When a DO element is specified within a TEMPLATE element, and a DO element with the same NAME attribute value is specified within a CARD element, the DO element within the CARD element takes precedence.		3	0	0	0		

WML-1.3-con-94	When the ONENTERFORWARD attribute of the CARD element is specified as a URL, it defines a resource to navitage to when the CARD is entered via a GO event or the equivalent.		3	0	0	0		
WML-1.3-con-95	When the ONENTERBACKWARD attribute of the CARD element is specified as a URL, it defines a resource to navigate to when the CARD is entered via a PREV event or the equivalent.		3	0	0	0		
WML-1.3-con-96	When the ONTIMER attribute of the CARD element is specified as a URL, it defines a resource to navigate to when the timer for the CARD expires.		3	0	0	0		
WML-1.3-con-97	When the ID attribute of the CARD element is specified as a string, it defines a name for the card. This name can then be used in the fragment portion of a URI to access the card directly.		3	0	0	0		
WML-1.3-con-98	When the NEWCONTEXT attribute of the CARD element is specified as "true", the current browser context is reinitialized when the card is entered.		3	0	0	0		
WML-1.3-con-99	When the NEWCONTEXT attribute of the CARD element is specified as "false" or is not specified, the current browser context is not initialized when the card is entered.		3	0	0	0		
WML-1.3-con-100	When the TABINDEX attribute of an element is specified as a number, it defines the relative position in the "tabbing order" within the card - where numerically greater numbers indicate a later position than numerically lesser numbers.		0	3	0	0		
WML-1.3-con-101	When the TABINDEX attribute of an input element is not specified, the user agent assigns it a position in the "tabbing order" that is later than all user-specified TABINDEX attributes within the card.		0	3	0	0		
WML-1.3-con-102	When the INAME attribute of the SELECT element is specified, it defines the name of a variable in which the index of the user selected option(s) will be stored.		3	0	0	0		

WML-1.3-con-103	When the IVALUE attribute of the SELECT element is set, it defines a default index value with which to select options within the SELECT element. If the INAME attribute is also specified, and the variable named by INAME already contains a value, the IVALUE attribute value is ignored.		3	0	0	0		
WML-1.3-con-104	When the INAME attribute of the SELECT element is set to a string, it defines the name of a variable in which to record the user's selection(s). When the MULTIPLE attribute of this SELECT element is set to TRUE, and multiple options are selected, the value of the variable named by INAME is a semi-colon separated list of selected option indices.		3	0	0	0		
WML-1.3-con-105	When the INAME attribute of the SELECT element is set to a string, it defines the name of a variable in which to record the user's selection(s). If the variable named by the INAME attribute is set when the element is displayed, the value of the variable is used as indices to pre-select option(s) within the SELECT element.		3	0	0	0		
WML-1.3-con-106	When the INAME attribute of the SELECT element is specified with a variable name, and the value of that variable is 0, it indicates that no options within the SELECT element are selected.		2	1	0	0		
WML-1.3-con-107	When the INAME attribute of the SELECT element is specified with a variable name, and that variable is not set, and the VALUE or IVALUE attribute is not specified, and the MULTIPLE attribute is set to "TRUE", the selected index defaults to 0.		2	1	0	0		
WML-1.3-con-108	When the INAME attribute of the SELECT element is specified with a variable name, and that variable is set to a value greater than the largest index of the enclosed OPTION elements, and the MULTIPLE attribute is set to "TRUE", no option is selected.		2	1	0	0		

WML-1.3-con-109	When the INAME attribute of the SELECT element is not specified, and the NAME attribute is specified as a variable name, and that variable is set, the value of that variable is used to select option(s).		3	0	0	0		
WML-1.3-con-110	When the INAME attribute of the SELECT element is not specified, and the NAME attribute is specified as a variable name, and that variable is set, but the value to which it is set does not match the VALUE attribute of an enclosed OPTION element, and the MULTIPLE attribute is set to "TRUE", no option is selected.		3	0	0	0		
WML-1.3-con-111	When the INAME attribute of the SELECT element is not specified, and the NAME attribute is specified as a variable name, and that variable is not set, and the MULTIPLE attribute is set to FALSE, the first option is selected.		3	0	0	0		
WML-1.3-con-112	When the INAME attribute of the SELECT element is not specified, and the NAME attribute is specified as a variable name, and that variable is set, but the value to which it is set does not match the VALUE attribute of an enclosed OPTION element, and the MULTIPLE attribute is set to "FALSE", the first option is selected.		3	0	0	0		
WML-1.3-con-113	When the INAME attribute of the SELECT element is specified with a variable name, and that variable is set to a value greater than the largest index of the enclosed OPTION elements, and the MULTIPLE attribute is set to "FALSE", the first option is selected.		3	0	0	0		
WML-1.3-con-114	When the NAME and INAME variables are updated the user agent must not display any side effects.		3	0	0	0		
WML-1.3-con-115	When the IVALUE attribute of the SELECT element is set, it defines a default value with which to select options within the SELECT element. If the MULTIPLE attribute of the SELECT element is set to TRUE, this default value is interpreted as a semi-colon separated list of option indices to select.		3	0	0	0		

WML-1.3-con-116	When the MULTIPLE attribute of the SELECT element is set to TRUE, the user is permitted to select multiple options from within the element.		3	0	0	0		
WML-1.3-con-117	When the MULTIPLE attribute of the SELECT element is set to FALSE, the user is permitted to select only a single option from within the element.		3	0	0	0		
WML-1.3-con-118	When the MULTIPLE attribute of the SELECT element is not specified, the user is permitted to select only a single option from within the element.		3	0	0	0		
WML-1.3-con-119	When the NAME attribute of the SELECT element is set to a legal variable name, it defines the name of a variable in which to record the user's selection(s).		3	0	0	0		
WML-1.3-con-120	When the NAME attribute of the SELECT element is set to a string, it defines the name of a variable in which to record the user's selection(s). When the MULTIPLE attribute of this SELECT element is set to TRUE, and multiple options are selected, the value of the variable named by NAME is a semi-colon separated list of values.		3	0	0	0		
WML-1.3-con-121	When the NAME attribute of the SELECT element is set to a string, it defines the name of a variable in which to record the user's selection(s). If the variable named by the NAME attribute is set when the element is displayed, the value of the variable is used to pre-select option(s) within the SELECT element.		3	0	0	0		
WML-1.3-con-122	When the VALUE attribute of the SELECT element is set, it defines a default value with which to select options within the SELECT element.		3	0	0	0		
WML-1.3-con-123	When the VALUE attribute of the SELECT element is set, it defines a default value with which to select options within the SELECT element. If the NAME attribute is also specified, and the variable named by NAME already contains a value, the VALUE attribute value is ignored.		3	0	0	0		

WML-1.3-con-124	When the VALUE attribute of the SELECT element is set, it defines a default value with which to select options within the SELECT element. If the MULTIPLE attribute of the SELECT element is set to TRUE, this default value is interpreted as a semi-colon separated list of values to select.		3	0	0	0		
WML-1.3-con-125	When the VALUE attribute of the OPTION element is set to a string, it defines the value to be used when setting the variable named by the NAME attribute of the enclosing SELECT element.		3	0	0	0		
WML-1.3-con-126	When the ONPICK attribute of the OPTION element is set to a URL, and the MULTIPLE attribute of the enclosing SELECT element is set to TRUE, it defines a task that is to be executed whenever the option is selected or deselected.		3	0	0	0		
WML-1.3-con-127	When the ONPICK attribute of the OPTION element is set to a URL, and the MULTIPLE attribute of the enclosing SELECT element is set to FALSE or not specified, it defines a task that is to be executed whenever the option is selected.		3	0	0	0		
WML-1.3-con-128	When the OPTION element contains PCDATA, that data is presented as the description of the option.		3	0	0	0		
WML-1.3-con-129	When the OPTION element contains a value with a variable reference, the variable is evaluated before the name variable is set.		3	0	0	0		
WML-1.3-con-130	When the OPTGROUP element is encountered, the implementation must allow the elements to be grouped and displayed in a hierarchical manor.		3	0	0	0		
WML-1.3-con-131	When a user agent encounters an OPTGROUP element it must display the elements children in readable manner even if OPTGROUP is not supported.		3	0	0	0		
WML-1.3-con-132	All INPUT objects that represent the input elements within the card must be initialized when the card is loaded.		3	0	0	0		

WML-1.3-con-133	When the MAXLENGTH attribute of the INPUT element is specified as a number, it defines the maximum number of characters that can be entered in the input field.		3	0	0	0		
WML-1.3-con-134	When the EMPTYOK attribute of the INPUT element is specified as "true", and the FORMAT attribute of the INPUT element is also specified, the input field may be left blank by the user.		3	0	0	0		
WML-1.3-con-135	When EMPTYOK is false, input is required even if the format mask would otherwise not require it.		3	0	0	0		
WML-1.3-con-136	When EMPTYOK is true, input is not required even if the format mask would otherwise require it.		3	0	0	0		
WML-1.3-con-137	When the FORMAT attribute of the INPUT element is specified as "*A", the text input is limited to upper-case letter, symbol or punctuation characters. Numeric characters are excluded		3	0	0	0		
WML-1.3-con-138	When the FORMAT attribute of the INPUT element is specified as "*a", the text input is limited to lower-case letter, symbol or punctuation characters.		3	0	0	0		
WML-1.3-con-139	When the FORMAT attribute of the INPUT element is specified as "*N", the text input is limited to numeric characters.		3	0	0	0		
WML-1.3-con-140	When the FORMAT attribute of the INPUT element is specified as "*X", the text input is limited to entry of any uppercase letter, numeric character, symbol, or punctuation character.		3	0	0	0		
WML-1.3-con-141	When the FORMAT attribute of the INPUT element is specified as "*x", the text input is limited to entry of any lowercase letter, numeric character, symbol, or punctuation character.		3	0	0	0		
WML-1.3-con-142	When the FORMAT attribute of the INPUT element is specified as a backslash "\" followed by a character c, that character is displayed in the input field at the position indicated.		2	1	0	0		

WML-1.3-con-143	When the FORMAT attribute of the INPUT element is specified as a series of format characters, the input of each position in the input field is limited as specified.		3	0	0	0		
WML-1.3-con-144	When the FORMAT attribute of the INPUT element is specified as "*M", the text input is limited to any characters, but the user-agent may assume that the input is in upper-case.		3	0	0	0		
WML-1.3-con-145	When the FORMAT attribute of the INPUT element is specified as "*m", the text input is limited to any characters, but the user-agent may assume that the input is in lower-case.		3	0	0	0		
WML-1.3-con-146	When the FORMAT attribute of the INPUT element is specified as "nM", where n is a digit from 1-9 and f is one of the format characters "aAmMNXx", the number of input characters is limited to n and the type is limited to f.		3	0	0	0		
WML-1.3-con-147	When the FORMAT attribute of the INPUT element is specified as an invalid mask it must be ignored by the user agent.		3	0	0	0		
WML-1.3-con-148	The NAME attribute of the INPUT element specifies the name of a variable in which the input value is stored. If this variable is set when the INPUT element is displayed, its value is used to initialize the content of the input field.		2	1	0	0		
WML-1.3-con-149	When the TYPE attribute of the INPUT element is specified as "text", it indicates that the input field is a text entry box, and that the input is to be displayed to the user in readable form.		3	0	0	0		
WML-1.3-con-150	When the TYPE attribute of the INPUT element is specified as "password", it indicates that the input field is a password entry box, and that the input is to be displayed to the user in an illegible form.		3	0	0	0		
WML-1.3-con-151	When the TYPE attribute of the INPUT element is not specified, it defaults to "text". This indicates that the input field is a text entry box, and that the input is to be displayed to the user in readable form.		3	0	0	0		

WML-1.3-con-152	When the VALUE attribute of the INPUT element is specified, it defines the default value for the variable named by the NAME attribute. When the INPUT element is displayed and the variable named by the NAME attribute is not set, that variable is initialized with the value of the VALUE attribute.		3	0	0	0		
WML-1.3-con-153	When the VALUE attribute of the INPUT element is specified, it defines the default value for the variable named by the NAME attribute. When the INPUT element is displayed and the variable named by the NAME attribute is already set, the value of the VALUE attribute is ignored.		3	0	0	0		
WML-1.3-con-154	When the FIELDSET element is used the user agent shall display all input fields even if it does not support FIELDSETS.		3	0	0	0		
WML-1.3-con-155	When the FIELDSET element is nested, the user agent must display all of the fields.		3	0	0	0		
WML-1.3-con-156	When the VALUE attribute of the TIMER element is specified as a positive integer value and the NAME value is not specified, the VALUE attribute defines the default starting value for the timer. Upon entering the card, the timer is started with this value. If the card is not exited prior to the expiration of the timer, the ONTIMER intrinsic event is delivered.		3	0	0	0		
WML-1.3-con-157	When the NAME attribute of the TIMER element is specified as a string, it defines a variable in which the timer value is to be stored.		3	0	0	0		
WML-1.3-con-158	When the NAME attribute of the TIMER element is specified as a string, it defines a variable name. If that variable is set, the timer is initialized with the value of the variable.		3	0	0	0		
WML-1.3-con-159	When the NAME attribute of the TIMER element is specified as a string, it defines a variable name. If that variable is not set and the VALUE attribute of the TIMER element is specified, the variable is set to the value of the VALUE attribute, and then the variable is used to initialize the timer.		3	0	0	0		

WML-1.3-con-160	When the NAME attribute of the TIMER element is specified as a string, it defines a variable name. If that variable is set to zero, the timer is disabled.		3	0	0	0		
WML-1.3-con-161	When the value of the timeout is not a positive integer, the user agent must ignore the TIMER element.		3	0	0	0		
WML-1.3-con-162	When the user agent encounters white space before or after an element or attribute it should be ignored.		3	0	0	0		
WML-1.3-con-163	When the B inline element is used, the text enclosed within the B element is displayed in a bold typeface.		3	0	0	0		
WML-1.3-con-164	When the BIG inline element is used, the text enclosed within the BIG element is displayed in a larger typeface.		3	0	0	0		
WML-1.3-con-165	When the EM inline element is used, the text enclosed within the EM element is displayed in an alternative typeface to give emphasis to the text.		3	0	0	0		
WML-1.3-con-166	When the I inline element is used, the text enclosed within the I element is displayed in an italicized typeface.		3	0	0	0		
WML-1.3-con-167	When the SMALL inline element is used, the text enclosed within the SMALL element is displayed in a smaller typeface.		3	0	0	0		
WML-1.3-con-168	When the STRONG inline element is used, the text enclosed within the STRONG element is displayed in an alternative typeface to give strong emphasis to the text.		3	0	0	0		
WML-1.3-con-169	When the U inline element is used, the text enclosed within the U element is displayed underlined.		3	0	0	0		
WML-1.3-con-170	When a non-breaking space entity is encountered, the user agent does not treat it as an inter-word space.		1	2	0	0		
WML-1.3-con-171	When a P element is empty or contains only insignificant white space, it must not impact line wrap mode.		2	1	0	0		
WML-1.3-con-172	When the ALIGN attribute of the P element is specified as "left", the text of the line is aligned on the left side of the display.		3	0	0	0		

WML-1.3-con-173	When the ALIGN attribute of the P element is specified as "center", the text of the line is centered on the display.		3	0	0	0		
WML-1.3-con-174	When the ALIGN attribute of the P element is specified as "right", the text of the line is aligned on the right side of the display.		3	0	0	0		
WML-1.3-con-175	When the ALIGN attribute of the P element is not specified, text is aligned with the left side of the display.		3	0	0	0		
WML-1.3-con-176	When the MODE attribute of the P element is specified as "wrap", the text of the line is broken onto multiple display lines at whitespace characters as needed.		2	1	0	0		
WML-1.3-con-177	When the MODE attribute of the P element is specified as "nowrap", the text of the line is displayed on a single line.		2	1	0	0		
WML-1.3-con-178	When the MODE attribute of the P element is not specified, it defaults to the MODE of the previous paragraph.		1	2	0	0		
WML-1.3-con-179	When the MODE attribute of the P element is not specified, and the P element is the first P element of the CARD, it defaults to "wrap" mode.		2	1	0	0		
WML-1.3-con-180	When a BR element is encountered, the user agent breaks the current line. Subsequent text in the paragraph is continued on the following line.		3	0	0	0		
WML-1.3-con-181	When the COLUMNS attribute of the TABLE element is specified as a non-negative integer, it defines the number of columns that each row of the table shall contain. Rows with fewer than the specified number of columns have empty columns added to their ends.		3	0	0	0		
WML-1.3-con-182	When the COLUMNS attribute of the TABLE element is specified as a non-negative integer, it defines the number of columns that each row of the table shall contain. Rows with more than the specified number of columns have any additional data collected into the last defined column of the row, with each column's data separated from the others by a single inter-word space.		3	0	0	0		

WML-1.3-con-183	When the ALIGN attribute of the TABLE element is specified as a sequence of the letters "R", "L", or "C", one for each column in the table, it defines the default alignment for data in each column as right, left, or center, respectively.		1	2	0	0		
WML-1.3-con-184	When the ALIGN attribute of the TABLE element is specified as a sequence of the letters "R", "L", or "C", but there are fewer letters than columns, the contents of the unspecified columns get the default alignment (left for left-to-right languages, right for right-to-left languages).		1	2	0	0		
WML-1.3-con-185	When the ALIGN attribute of the TABLE element is not specified, the contents of the columns get the default alignment (left for left-to-right languages, right for right-to-left languages).		3	0	0	0		
WML-1.3-con-186	When the ALIGN attribute of the TABLE element is specified as "D" the contents of that column will be default aligned (left for left-to-right languages, right for right-to-left languages). All extra designators should be ignored.		3	0	0	0		
WML-1.3-con-187	When the ALIGN attribute of the TABLE element is specified as an unknown designator the contents of that column will be default aligned (left for left-to-right languages, right for right-to-left languages).		3	0	0	0		
WML-1.3-con-188	When the TD element is specified, it defines the contents of a single cell of a row of a table. The contents of the element are rendered in the cell.		3	0	0	0		
WML-1.3-con-189	When the TD element is specified, it defines the contents of a single cell of a row of a table. The contents of the element are rendered in the cell.		3	0	0	0		
WML-1.3-con-190	When the TD element is specified and but has no contents, the element must still be counted as a cell for rendering purposes.		3	0	0	0		
WML-1.3-con-191	When the ALT attribute of the IMG element is specified as a string, it defines alternate text that is displayed when the image cannot be loaded.		3	0	0	0		

WML-1.3-con-192	When the SRC attribute of the IMG element is specified as a URL, it defines the location of the image data.		3	0	0	0		
WML-1.3-con-193	When the VSPACE attribute of the IMG element is specified as a length, it defines the vertical whitespace to be inserted above and below the image.		3	0	0	0		
WML-1.3-con-194	When the HSPACE attribute of the IMG element is specified as a length, it defines the horizontal whitespace to be inserted to the left and right of the image.		2	1	0	0		
WML-1.3-con-195	When the VSPACE attribute of the IMG element is not specified, it defaults to zero length.		3	0	0	0		
WML-1.3-con-196	When the HSPACE attribute of the IMG element is not specified, it defaults to zero length.		3	0	0	0		
WML-1.3-con-197	When the ALIGN attribute of the IMG element is specified as TOP, it means that the top of the image is aligned vertically with the top of the current text line.		3	0	0	0		
WML-1.3-con-198	When the ALIGN attribute of the IMG element is specified as BOTTOM, it means that the bottom of the image is aligned vertically with the current baseline.		3	0	0	0		
WML-1.3-con-199	When the ALIGN attribute of the IMG element is specified as MIDDLE, it means that the vertical center of the image is aligned vertically with the center of the current text line.		3	0	0	0		
WML-1.3-con-200	When the ALT attribute of the IMG element is specified as a string, it defines alternate text that is displayed when the user agent does not support the image format.		3	0	0	0		
WML-1.3-con-201	When a WML deck is encoded with an alternate DTD that includes elements or attributes beyond those in the WML 1.3 specification, the user agent will render the content of the unrecognized elements.		2	1	0	0		
WML-2.0-con-1	When navigation history is stored within the history stack, the newest entry must be stored at the top and the oldest entry must be stored at the bottom.		1	0	0	0		

WML-2.0-con-2	When forward navigation occurs, a new entry must be pushed onto the history stack as a result of this.		1	0	0	0		
WML-2.0-con-3	When backward navigation occurs, a new entry must be popped from the history stack as a result of this.		1	0	0	0		
WML-2.0-con-4	When an explicitly specified card is loaded into the history stack, a minimum of the absolute url, method, value of any postfields, and any request headers must be stored.		1	0	0	0		
WML-2.0-con-5	When an entry is loaded into the history stack , variable references in the resource request information must be replaced with the current value of the variable before loading takes place.		1	0	0	0		
WML-2.0-con-6	When the WML:GO task is executed, and it contains WML:SETVAR elements, the variable name and value in each WML:SETVAR element must be converted into a simple string by substituting all referenced variables.		1	0	0	0		
WML-2.0-con-7	When the WML:GO task is executed, the target URI must be identified and fetched by the user agent.		1	0	0	0		
WML-2.0-con-8	When the WML:GO task is executed, the fragment identifier of the target URI must be the card fetched.		1	0	0	0		
WML-2.0-con-9	When the WML:GO task is executed, and a fragment identifier is not specified as part of the URI, the first card in the document must be the destination.		1	0	0	0		
WML-2.0-con-10	When the WML:GO task is executed, and the WML:USE-XML-FRAGMENTS attribute is 'true', and the document contains a matching element ID attribute, the fragment identifier of the target URI must be the card that holds that element with the matching ID.		1	0	0	0		

WML-2.0-con-11	When the WML:GO task is executed(to navigate to a card in the same document), and the WML:USE-XML-FRAGMENTS attribute is 'true', and the document contains a matching element ID attribute, the fragment identifier of the target URI must be the card that holds that element with the matching ID and that element must be visible to the user.		1	0	0	0		
WML-2.0-con-12	When the WML:GO task is executed(to navigate to a card in the same document), and the WML:USE-XML-FRAGMENTS attribute is 'true', and the document contains no matching element ID attribute, the fragment identifier of the target URI must be the first card in that document.		1	0	0	0		
WML-2.0-con-13	When the WML:GO task is executed, and the WML:USE-XML-FRAGMENTS attribute in the destination document is set to 'false', and the document contains a matching card ID attribute, the fragment identifier of the target URI must be the card with the matching card ID attribute.		1	0	0	0		
WML-2.0-con-14	When the WML:GO task is executed, and the WML:USE-XML-FRAGMENTS attribute is 'false', and the document contains no matching card ID attribute, the fragment identifier of the target URI must be the first card in that document.		1	0	0	0		
WML-2.0-con-15	When the WML:GO task is executed, and the WML:USE-XML-FRAGMENTS attribute is 'false', and the document contains non-card element ID "x", the fragment identifier of the target URI must be the first card in that document.		1	0	0	0		
WML-2.0-con-16	When the WML:GO task is executed, and it contains WML:SETVAR elements, the variable assignments resulting from the processing must be assigned to the current user agent context.		1	0	0	0		
WML-2.0-con-17	When the WML:GO task is executed, and the destination document contains a WML:NEWCONTEXT attribute, a new user agent context must be created.		1	0	0	0		

WML-2.0-con-18	When the WML:GO task is executed to take the user to another card, an entry referring to the destination must be pushed onto the history stack.		1	0	0	0		
WML-2.0-con-19	When the WML:GO task is executed, and the destination document contains a ENTERFORWARD intrinsic event binding, the task associated with the event binding must be executed and processing must then stop.		1	0	0	0		
WML-2.0-con-20	When the WML:GO task is executed, and the destination document contains a WML:TIMER element, the timer must be started.		0	1	0	0		
WML-2.0-con-21	When the WML:GO task is executed, the destination card must be displayed using the current variable state.		1	0	0	0		
WML-2.0-con-22	When the WML:PREV task is executed, and it contains WML:SETVAR elements, the variable name and value in each WML:SETVAR element must be converted into a simple string by substituting all referenced variables.		1	0	0	0		
WML-2.0-con-23	When the WML:PREV task is executed, the history stack must be popped.		1	0	0	0		
WML-2.0-con-24	When the WML:PREV task is executed and the target URI is a card in the same document, the fragment identifier of the target URI must be the card fetched.		1	0	0	0		
WML-2.0-con-25	When the WML:PREV task is executed and the target URI is a separate document, the fragment identifier of the target URI must be the part of the document fetched.		1	0	0	0		
WML-2.0-con-26	When the WML:PREV task is executed, and a fragment identifier is not specified as part of the URI, the first card in the document must be the destination.		1	0	0	0		
WML-2.0-con-27	When the WML:PREV task is executed, and the WML:USE-XML-FRAGMENTS attribute is 'true', and the document contains a matching element ID attribute, the fragment identifier of the target URI must be the card that holds that element with the matching ID.		1	0	0	0		

WML-2.0-con-28	When the WML:PREV task is executed(to navigate to a card in the same document), and the WML:USE-XML-FRAGMENTS attribute is 'true', and the document contains a matching element ID attribute, the fragment identifier of the target URI must be the card that holds that element with the matching ID, and that element must be visible to the user.		1	0	0	0		
WML-2.0-con-29	When the WML:PREV task is executed(to navigate to a card in the same document), and the WML:USE-XML-FRAGMENTS attribute is 'true', and the document contains no matching element ID attribute, the fragment identifier of the target URI must be the first card in that document.		0	1	0	0		
WML-2.0-con-30	When the WML:PREV task is executed, and the WML:USE-XML-FRAGMENTS attribute in the destination document is set to 'false', and the document contains a matching card ID attribute, the fragment identifier of the target URI must be the card with the matching card ID attribute.		1	0	0	0		
WML-2.0-con-31	When the WML:PREV task is executed, and the WML:USE-XML-FRAGMENTS attribute is 'false', and the document contains no matching card ID attribute, the fragment identifier of the target URI must be the first card in that document.		0	1	0	0		
WML-2.0-con-32	When the WML:PREV task is executed, and it contains WML:SETVAR elements, the variable assignments resulting from the processing must be assigned to the current user agent context.		1	0	0	0		
WML-2.0-con-33	When the WML:PREV task is executed, and the destination document contains a ENTERBACKWARD intrinsic event binding, the task associated with the event binding must be executed and processing must then stop.		1	0	0	0		
WML-2.0-con-34	When the WML:PREV task is executed, and the destination document contains a WML:TIMER element, the timer must be started.		1	0	0	0		

WML-2.0-con-35	When the WML:PREV task is executed, the destination card must be displayed using the current variable state.		1	0	0	0		
WML-2.0-con-36	When the WML:REFRESH task is executed, and it contains WML:SETVAR elements, the variable name and value in each WML:SETVAR element must be converted into a simple string by substituting all referenced variables.		1	0	0	0		
WML-2.0-con-37	When the WML:REFRESH task is executed, and it contains WML:SETVAR elements, the variable assignments resulting from the processing must be assigned to the current user agent context.		1	0	0	0		
WML-2.0-con-38	When the WML:REFRESH task is executed, and the document contains a WML:TIMER element, the timer must be started.		0	1	0	0		
WML-2.0-con-39	When the WML:REFRESH task is executed, the current card is re-displayed using the current variable state and processing stops.		1	0	0	0		
WML-2.0-con-40	When a task fails to fetch it's target resource, the invoking card must remain the current card.		1	0	0	0		
WML-2.0-con-41	When a task fails to fetch it's target resource, there must be no changes to the browser context.		1	0	0	0		
WML-2.0-con-42	When a task fails to fetch it's target resource, there must be no intrinsic event bindings executed.		1	0	0	0		
WML-2.0-con-43	When a forms initial values have been determined, they must be committed to form control variables.		1	0	0	0		
WML-2.0-con-44	When displaying a card with form controls, the user agent must process the controls in the order in which they appear in the card.		0	1	0	0		
WML-2.0-con-45	When setting up a form control, the user agent must store the initial value of each control for use during form reset.		1	0	0	0		
WML-2.0-con-46	When a card is reached through either forward or backward navigation, the user agent must perform form initialisation.		1	0	0	0		

WML-2.0-con-47	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", and the WML:NAME attribute is assigned, the WML:NAME attribute must name the form control variable.		1	0	0	0		
WML-2.0-con-48	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", the WML:NAME attribute is assigned, and the form control variable conforms to the input mask, the initial value of the control must be the value of the control variable.		1	0	0	0		
WML-2.0-con-49	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", the WML:NAME attribute is assigned, and the value of a form control variable does not conform to the input mask, it must be ignored and the initial value of the control must be determined by the VALUE attribute.		1	0	0	0		
WML-2.0-con-50	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", the WML:NAME attribute is assigned, and the value of the form control variable and the VALUE attribute do not conform to the input mask, they must be ignored and the initial value of the control must be set to an empty string.		1	0	0	0		
WML-2.0-con-51	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", the WML:NAME attribute is assigned, the value of a form control variable does not conform to the input mask, and the VALUE attribute is not set, the initial value of the text input control must be an empty string.		1	0	0	0		
WML-2.0-con-52	When using a text input control, the control is an INPUT element and the TYPE attribute is set to "text", the initial value must be pre-loaded into the control, when the WML:NAME control variable is set.		1	0	0	0		
WML-2.0-con-53	When using a text input control, and the control is a TEXTAREA element, the WML:NAME attribute must name the form control variable.		1	0	0	0		

WML-2.0-con-54	When using a text input control, the control is a TEXTAREA element, the WML:NAME attribute is assigned, and the form control variable conforms to the input mask, the initial value of the control must be the value of the control variable.		1	0	0	0		
WML-2.0-con-55	When using a text input control, the control is a TEXTAREA element, the WML:NAME attribute is assigned, the form control variable does not conform to the input mask and the content of the TEXTAREA element conforms to the input mask, the initial value of the control must be the content of the element.		1	0	0	0		
WML-2.0-con-56	When using a text input control, the control is a TEXTAREA element, the WML:NAME attribute is assigned and both the form control variable and the content of the TEXTAREA element do not conform to the input mask, the initial value of the control must be an empty string.		1	0	0	0		
WML-2.0-con-57	When using a text input control, the control is a TEXTAREA element, the WML:NAME attribute is assigned, the form control variable does not conform to the input mask and the TEXTAREA element is an empty element, the initial value of the control must be an empty string.		1	0	0	0		
WML-2.0-con-58	When using a text input control, and the control is a TEXTAREA element, the initial value must be pre-loaded into the control, when the WML:NAME control variable is set.		1	0	0	0		
WML-2.0-con-59	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", the NAME attribute is assigned, and no input mask is specified, the initial value of the control must be the value of the VALUE attribute.		1	0	0	0		
WML-2.0-con-60	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", the NAME attribute is assigned, and the VALUE attribute is not specified, the initial value of the control must be an empty string.		1	0	0	0		

WML-2.0-con-61	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", the NAME attribute is assigned, and the value of the VALUE attribute conforms to the input mask, the initial value of the control must be the value of the VALUE attribute.		1	0	0	0		
WML-2.0-con-62	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", the NAME attribute is assigned, and the value of the VALUE attribute does not conform to the input mask, the initial value of the control must be an empty string.		1	0	0	0		
WML-2.0-con-63	When using a text input control, the control is an INPUT element and the TYPE attribute is set to "text", the initial value must be pre-loaded into the control, when the NAME control variable is set.		1	0	0	0		
WML-2.0-con-64	When using a text input control, the control is a TEXTAREA element, the NAME attribute is assigned, and no input mask is specified, the initial value of the control must be the content of the element.		1	0	0	0		
WML-2.0-con-65	When using a text input control, the control is a TEXTAREA element, the NAME attribute is assigned, no input mask is specified, and the TEXTAREA element is an empty element, the initial value of the control must be an empty string.		1	0	0	0		
WML-2.0-con-66	When using a text input control, the control is a TEXTAREA element, the NAME attribute is assigned and the content of the TEXTAREA element conforms to the input mask, the initial value of the control must be the content of the element.		1	0	0	0		
WML-2.0-con-67	When using a text input control, the control is a TEXTAREA element, the NAME attribute is assigned and the content of the TEXTAREA element does not conform to the input mask, the initial value of the control must be an empty string.		1	0	0	0		
WML-2.0-con-68	When using a text input control, and the control is a TEXTAREA element, the initial value must be pre-loaded into the control, when the NAME control attribute is set.		0	1	0	0		

WML-2.0-con-69	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", and both the WML:NAME and NAME attributes are assigned, the WML:NAME attribute takes precedence, and the NAME attribute must be ignored.		1	0	0	0		
WML-2.0-con-70	When using a text input control, the control is a TEXTAREA element, and both the WML:NAME and NAME attributes are assigned, the WML:NAME attribute takes precedence, and the NAME attribute must be ignored.		1	0	0	0		
WML-2.0-con-71	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", neither the WML:NAME or NAME attribute is assigned, and the VALUE attribute is, the value of the VALUE attribute must be pre-loaded into the form control.		1	0	0	0		
WML-2.0-con-72	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", neither the WML:NAME or NAME attribute is assigned, and the VALUE attribute is not set, the initial value of the control must be an empty string.		1	0	0	0		
WML-2.0-con-73	When using a text input control, the control is a TEXTAREA element, neither the WML:NAME or NAME attribute is assigned, and the form control has a content, the initial value must be the content of the element.		1	0	0	0		
WML-2.0-con-74	When using a text input control, the control is a TEXTAREA element, neither the WML:NAME or NAME attribute is assigned, and the form control has no content, the initial value must be an empty string.		1	0	0	0		
WML-2.0-con-75	When using a text input control, the control is an INPUT element, and the TYPE attribute is set to "password", the WML:NAME attribute must name the form control variable.		1	0	0	0		

WML-2.0-con-76	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", the WML:NAME attribute is assigned, and the form control variable conforms to the input mask, the initial value of the control must be the value of the control variable.		1	0	0	0		
WML-2.0-con-77	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", the WML:NAME attribute is assigned, and the value of a form control variable does not conform to the input mask, it must be ignored and the initial value of the control must be determined by the VALUE attribute.		1	0	0	0		
WML-2.0-con-78	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", the WML:NAME attribute is assigned, and the value of the form control variable and the VALUE attribute do not conform to the input mask, they must be ignored and the initial value of the control must be set to an empty string.		1	0	0	0		
WML-2.0-con-79	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", the WML:NAME attribute is assigned, the value of a form control variable does not conform to the input mask, and the VALUE attribute is not set, the initial value of the text input control must be an empty string.		1	0	0	0		
WML-2.0-con-80	When using a text input control, the control is an INPUT element and the TYPE attribute is set to "password", the initial value must be pre-loaded into the control, when the WML:NAME control variable is set.		1	0	0	0		
WML-2.0-con-81	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", the NAME attribute is assigned, and no input mask is specified, the initial value of the control must be the value of the VALUE attribute.		0	1	0	0		

WML-2.0-con-82	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", the NAME attribute is assigned, and the VALUE attribute is not specified, the initial value of the control must be an empty string.		1	0	0	0		
WML-2.0-con-83	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", the NAME attribute is assigned, and the value of the VALUE attribute conforms to the input mask, the initial value of the control must be the value of the VALUE attribute.		0	1	0	0		
WML-2.0-con-84	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", the NAME attribute is assigned, and the value of the VALUE attribute does not conform to the input mask, the initial value of the control must be an empty string.		1	0	0	0		
WML-2.0-con-85	When using a text input control, the control is an INPUT element and the TYPE attribute is set to "password", the initial value must be pre-loaded into the control, when the NAME control variable is set.		0	1	0	0		
WML-2.0-con-86	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", and both the WML:NAME and NAME attributes are assigned, the WML:NAME attribute takes precedence, and the NAME attribute must be ignored.		1	0	0	0		
WML-2.0-con-87	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", neither the WML:NAME or NAME attribute is assigned, and the VALUE attribute is, the value of the VALUE attribute must be pre-loaded into the form control.		1	0	0	0		
WML-2.0-con-88	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", neither the WML:NAME or NAME attribute is assigned, and the VALUE attribute is not set, the initial value of the control must be an empty string.		1	0	0	0		

WML-2.0-con-89	When using a hidden input control, and the WML:NAME attribute is assigned, the WML:NAME attribute must name the form control variable.		1	0	0	0		
WML-2.0-con-90	When using a hidden input control, and the WML:NAME attribute is assigned, the initial value of the control is the value of the control variable.		1	0	0	0		
WML-2.0-con-91	When using a hidden input control, the WML:NAME attribute is assigned but not set, and the VALUE attribute is specified, the initial value of the control is the value of the VALUE attribute.		1	0	0	0		
WML-2.0-con-92	When using a hidden input control, the WML:NAME attribute is assigned but not set, and the VALUE attribute is not specified, the initial value of the control is an empty string.		1	0	0	0		
WML-2.0-con-93	When using a hidden input control, the WML:NAME attribute is not assigned, and the NAME attribute is assigned, the name of the control variable must be the value of the NAME attribute.		0	1	0	0		
WML-2.0-con-94	When using a hidden input control, the WML:NAME attribute is not assigned, the NAME attribute is assigned, and the VALUE attribute is specified, the initial value of the control is the value of the VALUE attribute.		0	1	0	0		
WML-2.0-con-95	When using a hidden input control, the WML:NAME attribute is not assigned, the NAME attribute is assigned, and the VALUE attribute is not specified, the initial value of the control is an empty string.		1	0	0	0		
WML-2.0-con-96	When using a hidden input control, and both the WML:NAME and NAME attributes are assigned, the WML:NAME attribute takes precedence and the NAME attribute must be ignored.		1	0	0	0		
WML-2.0-con-97	When the control only permits a single selection, the WML:INAME attribute of the SELECT element is specified, it must name the variable to use to determine the default option index.		1	0	0	0		

WML-2.0-con-98	When the control only permits a single selection, the default option index has not yet been determined and the WML:IVALUE attribute of the SELECT element is specified, the default option index must be the value of the WML:IVALUE attribute.		1	0	0	0		
WML-2.0-con-99	When the control only permits a single selection, the default option index has not yet been determined and the WML:NAME attribute of the SELECT element is specified, it must name the variable to use to determine the default option index.		1	0	0	0		
WML-2.0-con-100	When the control only permits a single selection, the default option index has not yet been determined and the WML:VALUE attribute of the SELECT element is specified, the default option index must be the index of the option whose VALUE attribute equals the value of the WML:VALUE attribute.		1	0	0	0		
WML-2.0-con-101	When the control only permits a single selection, no option has been pre-selected within the SELECT element, the default option index is one.		1	0	0	0		
WML-2.0-con-102	When the control only permits a single selection and the WML:INAME attribute of the SELECT element is specified, the variable named by the WML:INAME attribute must be set with the default option index.		1	0	0	0		
WML-2.0-con-103	When the control only permits a single selection and the WML:NAME attribute of the SELECT element is specified, the variable named by the WML:NAME attribute must be set with the value of the VALUE attribute on the OPTION element as the default option index.		1	0	0	0		
WML-2.0-con-104	When the control only permits a single selection, the WML:NAME attribute of the SELECT element is specified, and the VALUE attribute of the OPTION element is not, the variable named by the WML:NAME attribute must be set to a empty string.		0	1	0	0		

WML-2.0-con-105	When the WML:INAME attribute of the SELECT element is specified and multiple selections are permitted, the WML:INAME attribute must name the variable to use to determine the default option index and the value of the named variable must be a semicolon-delimited list of the indices of the pre-selected OPTION elements.		1	0	0	0		
WML-2.0-con-106	When multiple selections are permitted, the default option index has not yet been determined, the WML:IVALUE attribute of the SELECT element is specified, the value of the WML:IVALUE attribute must be used to determine the default option index and this value must be a semicolon-delimited list of the indices of the pre-selected OPTION elements.		1	0	0	0		
WML-2.0-con-107	When multiple selections are permitted, the default option index has not yet been determined, the WML:NAME attribute of the SELECT element is specified, the value of the WML:NAME variable must be used to determine the default option index and this value must be a semicolon-delimited list of the values of the pre-selected OPTION elements VALUE attributes.		1	0	0	0		
WML-2.0-con-108	When multiple selections are permitted, the default option index has not yet been determined, the WML:VALUE attribute of the SELECT element is specified, the WML:VALUE'S value is used to determine the default option indices and must be a semicolon-delimited list of the values of the pre-selected OPTION elements VALUE attributes.		1	0	0	0		
WML-2.0-con-109	When the WML:INAME attribute of the SELECT element is specified and multiple selections are permitted, the variable named by the WML:INAME attribute must be set with the default option index.		1	0	0	0		
WML-2.0-con-110	When the WML:INAME attribute of the SELECT element is specified, multiple selections are permitted, and there are no default options, the variable named by the WML:INAME attribute must be set to an empty string.		1	0	0	0		

WML-2.0-con-111	When the WML:NAME attribute of the SELECT element is specified and multiple selections are permitted, for each index in the default indices which is greater than zero, the variable named by the WML:NAME attribute must be set to a semicolon delimited string of the VALUE attribute on the OPTION elements.		1	0	0	0		
WML-2.0-con-112	When the WML:NAME attribute of the SELECT element is specified, multiple selections are permitted, and there are no default options, the variable named by the WML:NAME attribute must be set to an empty string.		1	0	0	0		
WML-2.0-con-113	When the WML:NAME attribute of the SELECT element is specified, multiple selections are permitted, and all selected OPTION elements contain an empty VALUE attribute, the variable named by the WML:NAME attribute must be set to an empty string.		1	0	0	0		
WML-2.0-con-114	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, the control only permits a single selection and the NAME attribute is assigned, it must name the control variable.		1	0	0	0		
WML-2.0-con-115	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, the control only permits a single selection the NAME attribute is assigned, and exactly one OPTION has the SELECTED attribute assigned, that option is the pre-selected option.		1	0	0	0		
WML-2.0-con-116	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, the control only permits a single selection the NAME attribute is assigned, and more than one OPTION has the SELECTED attribute assigned, the first such option is the pre-selected option.		1	0	0	0		

WML-2.0-con-117	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, the control only permits a single selection and the NAME attribute is assigned, the initial value of the control variable must be the value of the pre-selected OPTION element's value attribute.		0	1	0	0		
WML-2.0-con-118	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, the control only permits a single selection, the NAME attribute is assigned, and the VALUE attribute of the pre-selected OPTION element is not specified, the initial value of the control variable must be the content of the pre-selected OPTION element.		0	1	0	0		
WML-2.0-con-119	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, the control only permits a single selection, the NAME attribute is assigned, and both the VALUE attribute and the content of the pre-selected OPTION element are not specified, the initial value of the control variable must be an empty string.		1	0	0	0		
WML-2.0-con-120	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, multiple selections are permitted, and the NAME attribute is assigned, the pre-selected options are those option elements with their SELECTED attribute assigned.		1	0	0	0		
WML-2.0-con-121	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, multiple selections are permitted, the NAME attribute is assigned, and no OPTION has the SELECTED attribute assigned, there must be no pre-selected options.		1	0	0	0		
WML-2.0-con-122	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, multiple selections are permitted, and the NAME attribute is assigned, the initial value of the form control must be a string consisting of a semicolon-delimited list of the values of the pre-selected OPTION element's VALUE attributes.		0	1	0	0		

WML-2.0-con-123	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, multiple selections are permitted, the NAME attribute is assigned, and the VALUE attribute of the pre-selected OPTION element are not specified, the initial value of the control variable must be a string consisting of a semicolon-delimited list of the content of the pre-selected OPTION element.		0	1	0	0		
WML-2.0-con-124	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, multiple selections are permitted, the NAME attribute is assigned, and both the VALUE attribute and the content of the pre-selected OPTION element are not specified, the initial value of the control variable must be an empty string.		1	0	0	0		
WML-2.0-con-125	When neither the WML:NAME, the WML:INAME, the WML:VALUE nor the WML:INAME is assigned, multiple selections are permitted, the NAME attribute is assigned, and there are no pre-selected options, the initial value of the control variable must be an empty string.		1	0	0	0		
WML-2.0-con-126	When the WML:INAME, attribute of the SELECT element is assigned, the SELECTED attribute of the OPTION element must be ignored.		1	0	0	0		
WML-2.0-con-127	When the WML:NAME, attribute of the SELECT element is assigned, the SELECTED attribute of the OPTION element must be ignored.		1	0	0	0		
WML-2.0-con-128	When the WML:IVALUE, attribute of the SELECT element is assigned, the SELECTED attribute of the OPTION element must be ignored.		1	0	0	0		
WML-2.0-con-129	When the WML:VALUE, attribute of the SELECT element is assigned, the SELECTED attribute of the OPTION element must be ignored.		1	0	0	0		
WML-2.0-con-130	When the WML:INAME, attribute of the SELECT element is assigned, the NAME attribute of the SELECT element must be ignored.		1	0	0	0		
WML-2.0-con-131	When the WML:NAME, attribute of the SELECT element is assigned, the NAME attribute of the SELECT element must be ignored.		1	0	0	0		

WML-2.0-con-132	When the WML:IVALUE, attribute of the SELECT element is assigned, the NAME attribute of the SELECT element must be ignored.		1	0	0	0		
WML-2.0-con-133	When the WML:VALUE, attribute of the SELECT element is assigned, the NAME attribute of the SELECT element must be ignored.		1	0	0	0		
WML-2.0-con-134	When the WML:NAME attribute of the INPUT element is specified, and the TYPE attribute is set to 'checkbox', the variable named by the WML:NAME attribute must hold a semicolon-delimited list of the values of the INPUT elements VALUE attribute, for all checkboxes that have been pre-selected.		1	0	0	0		
WML-2.0-con-135	When the WML:NAME attribute of the INPUT element is specified, the TYPE attribute is set to 'checkbox', and the variable named by the WML:NAME attribute has not been pre-set, there must be no pre-selected checkboxes.		0	1	0	0		
WML-2.0-con-136	When the WML:NAME attribute of the INPUT element is specified, and the TYPE attribute is set to 'radio', the variable named by the WML:NAME attribute must hold the value of the INPUT elements VALUE attribute, for the radio button that has been pre-selected.		1	0	0	0		
WML-2.0-con-137	When the WML:NAME attribute of the INPUT element is specified, the TYPE attribute is set to 'radio', and the variable named by the WML:NAME attribute has not been pre-set, there must be no pre-selected radio buttons.		0	1	0	0		
WML-2.0-con-138	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", and the user attempts to commit input, the user agent must validate that input against the input mask, as defined by the WML:FORMAT and WML:EMPTYOK attributes or their CSS equivalents.		1	0	0	0		
WML-2.0-con-139	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", and the user enters data into that text control, for which there is no input mask set, the user agent must set the control variable to the control's current value.		1	0	0	0		

WML-2.0-con-140	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", and the user attempts to commit input which conforms to the input mask set, the user agent must set the control variable to that value.		1	0	0	0		
WML-2.0-con-141	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", and the value input by the user into the control does not conform to the input mask, the user agent must not commit that input and must notify the user that the input was rejected, and allow the user to re-submit new input.		1	0	0	0		
WML-2.0-con-142	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "text", and the value input by the user into the control does not conform to the input mask, the control variable must not be modified.		1	0	0	0		
WML-2.0-con-143	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", and the user attempts to commit input, the user agent must validate that input against the input mask, as defined by the WML:FORMAT and WML:EMPTYOK attributes or their CSS equivalents.		1	0	0	0		
WML-2.0-con-144	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", and the user enters data into that text control, for which there is no input mask set, the user agent must set the control variable to the control's current value.		1	0	0	0		
WML-2.0-con-145	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", and the user attempts to commit input which conforms to the input mask set, the user agent must set the control variable to that value.		1	0	0	0		

WML-2.0-con-146	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", and the value input by the user into the control does not conform to the input mask, the user agent must not commit that input and must notify the user that the input was rejected, and allow the user to re-submit new input.		1	0	0	0		
WML-2.0-con-147	When using a text input control, the control is an INPUT element, the TYPE attribute is set to "password", and the value input by the user into the control does not conform to the input mask, the control variable must not be modified.		1	0	0	0		
WML-2.0-con-148	When using a text input control, the control is a TEXTAREA element, and the user attempts to commit input, the user agent must validate that input against the input mask, as defined by the WML:FORMAT and WML:EMPTYOK attributes or their CSS equivalents.		1	0	0	0		
WML-2.0-con-149	When using a text input control, the control is a TEXTAREA element, and the user enters data into that text control, for which there is no input mask set, the user agent must set the control variable to the control's current value.		1	0	0	0		
WML-2.0-con-150	When using a text input control, the control is a TEXTAREA element, and the user attempts to commit input which conforms to the input mask set, the user agent must set the control variable to that value.		1	0	0	0		
WML-2.0-con-151	When using a text input control, the control is a TEXTAREA element, and the value input by the user into the control does not conform to the input mask, the user agent must not commit that input and must notify the user that the input was rejected, and allow the user to re-submit new input.		1	0	0	0		
WML-2.0-con-152	When using a text input control, the control is a TEXTAREA element, and the value input by the user into the control does not conform to the input mask, the control variable must not be modified.		1	0	0	0		

WML-2.0-con-153	When the control only permits a single selection, the WML:NAME attribute of the SELECT element is specified, the named variable must be set to the value of the VALUE attribute of the selected OPTION element.		1	0	0	0		
WML-2.0-con-154	When the control only permits a single selection, the WML:NAME attribute of the SELECT element is specified, the value of the VALUE attribute of the selected OPTION element is not specified, the named variable must be set to an empty string.		0	1	0	0		
WML-2.0-con-155	When the control only permits a single selection, the WML:INAME attribute of the SELECT element is specified, the named variable must be set to the index of the selected OPTION element.		1	0	0	0		
WML-2.0-con-156	When the control only permits a single selection, the WML:INAME attribute of the SELECT element is specified, the index numbering must begin at one and increase monotonically.		1	0	0	0		
WML-2.0-con-157	When the WML:NAME attribute of the SELECT element is specified, and multiple selections are permitted, the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected OPTION elements.		1	0	0	0		
WML-2.0-con-158	When the WML:NAME attribute of the SELECT element is specified, and multiple selections are permitted, the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected OPTION elements that contain a VALUE attribute.		0	1	0	0		
WML-2.0-con-159	When the WML:NAME attribute of the SELECT element is specified, and multiple selections are permitted, the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected OPTION elements that do not contain an empty VALUE attribute.		1	0	0	0		

WML-2.0-con-160	When the WML:NAME attribute of the SELECT element is specified, and multiple selections are permitted, but the user commits without selecting any options, the control variable must be set to an empty string.		1	0	0	0		
WML-2.0-con-161	When the WML:NAME attribute of the SELECT element is specified, and multiple selections are permitted, but all OPTION element VALUE attributes are set to an empty string, the SELECT elements control variable must also be set to an empty string.		1	0	0	0		
WML-2.0-con-162	When the WML:INAME attribute of the SELECT element is specified, and multiple selections are permitted, the named variable must be set to a semicolon-delimited string of the indices of the selected OPTION elements.		1	0	0	0		
WML-2.0-con-163	When the WML:INAME attribute of the SELECT element is specified, and multiple selections are permitted, but the user commits without selecting any options, the control variable must be set to an empty string.		1	0	0	0		
WML-2.0-con-164	When the control only permits a single selection, the NAME attribute of the SELECT element is used to name the control variable, the named variable must be set to the value of the VALUE attribute of the selected OPTION element.		0	1	0	0		
WML-2.0-con-165	When the control only permits a single selection, the NAME attribute of the SELECT element is used to name the control variable, the value of the VALUE attribute of the selected OPTION element is not specified but the content is set, the named variable must be set to the content of the selected OPTION element.		0	1	0	0		
WML-2.0-con-166	When the control only permits a single selection, the NAME attribute of the SELECT element is used to name the control variable, the value of the VALUE attribute of the selected OPTION element is not specified and the content of the OPTION element is unset, the named variable must be set to an empty string.		1	0	0	0		

WML-2.0-con-167	When multiple selections are permitted, the NAME attribute of the SELECT element is used to name the control variable, the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected OPTION elements.		0	1	0	0		
WML-2.0-con-168	When multiple selections are permitted, the NAME attribute of the SELECT element is used to name the control variable, the VALUE attribute of some selected OPTION elements is not specified but the content of these selected OPTION elements is specified, the content of these selected OPTION elements must be substitued to their VALUE attribute in the semicolon-delimited string, result, of the control variable.		0	1	0	0		
WML-2.0-con-169	When multiple selections are permitted, the NAME attribute of the SELECT element is used to name the control variable, the VALUE attributes of a selected OPTION elements is not specified and the content of this selected elements is not specified, this option must be ignored in the semicolon-delimited string, result, of the control variable.		0	1	0	0		
WML-2.0-con-170	When multiple selections are permitted, the NAME attribute of the SELECT element is used to name the control variable, but the user commits without selecting any options, the control variable must be set to an empty string.		1	0	0	0		
WML-2.0-con-171	When the WML:INAME attribute of the SELECT element is specified, and multiple selections are permitted, the user agent must ensure that the WML:INAME variable value contains no duplicate index values.		1	0	0	0		
WML-2.0-con-172	When the WML:NAME attribute of the SELECT element is specified, and multiple selections are permitted, the user agent must contain duplicate values in the situation where multiple selected option elements have the same value.		1	0	0	0		

WML-2.0-con-173	When the WML:NAME attribute of the SELECT element is specified, and multiple selections are permitted, the user agent must not allow the WML:NAME control variable to contain empty values (e.g. 'cat;"";dog').		1	0	0	0		
WML-2.0-con-174	When the WML:NAME variable of the SELECT element is changed, due to a user selecting a option, the user agent must not exhibit display side effects, except when there is an explicit refresh task.		1	0	0	0		
WML-2.0-con-175	When the WML:INAME variable of the SELECT element is changed, due to a user selecting an option, the user agent must not exhibit display side effects, except when there is an explicit refresh task.		1	0	0	0		
WML-2.0-con-176	When the variable named by the WML:NAME of the SELECT element is changed, the user agent must update it each time, before each and every task invocation.		1	0	0	0		
WML-2.0-con-177	When the variable named by the WML:INAME of the SELECT element is changed, the user agent must update it each time, before each and every task invocation.		1	0	0	0		
WML-2.0-con-178	When the WML:NAME attribute of the INPUT element is assigned, the TYPE attribute of the INPUT element is set to 'checkbox', the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected INPUT elements.		1	0	0	0		
WML-2.0-con-179	When the WML:NAME attribute of the INPUT element is assigned, the TYPE attribute of the INPUT element is set to 'checkbox', the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected INPUT elements that contain a VALUE attribute.		1	0	0	0		
WML-2.0-con-180	When the WML:NAME attribute of the INPUT element is assigned, the TYPE attribute of the INPUT element is set to 'checkbox', the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected INPUT elements that do not contain an empty VALUE attribute.		0	1	0	0		

WML-2.0-con-181	When the WML:NAME attribute of the INPUT element is assigned, the TYPE attribute of the INPUT element is set to 'checkbox', but the user commits without selecting any checkboxes, the control variable must be set to an empty string.		1	0	0	0		
WML-2.0-con-182	When the WML:NAME attribute of the INPUT element is assigned, the TYPE attribute of the INPUT element is set to 'checkbox', but all INPUT element VALUE attributes are set to an empty string, the named variable must also be set to an empty string.		1	0	0	0		
WML-2.0-con-183	When the NAME attribute of the INPUT element is used to name the control variable, the TYPE attribute of the INPUT element is set to 'checkbox', the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected INPUT elements.		0	1	0	0		
WML-2.0-con-184	When the NAME attribute of the INPUT element is used to name the control variable, the TYPE attribute of the INPUT element is set to 'checkbox', the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected INPUT elements that contain a VALUE attribute.		0	1	0	0		
WML-2.0-con-185	When the NAME attribute of the INPUT element is used to name the control variable, the TYPE attribute of the INPUT element is set to 'checkbox', the named variable must be set to a semicolon-delimited string of the VALUE attributes of the selected INPUT elements that do not contain an empty VALUE attribute.		0	1	0	0		
WML-2.0-con-186	When the NAME attribute of the INPUT element is used to name the control variable, the TYPE attribute of the INPUT element is set to 'checkbox', but the user commits without selecting any checkboxes, the control variable must be set to an empty string.		1	0	0	0		

WML-2.0-con-187	When the NAME attribute of the INPUT element is used to name the control variable, the TYPE attribute of the INPUT element is set to 'checkbox', but all INPUT element VALUE attributes are set to an empty string, the named variable must also be set to an empty string.		1	0	0	0		
WML-2.0-con-188	When the WML:NAME attribute of the INPUT element is assigned, the TYPE attribute of the INPUT element is set to 'radio', the named variable must hold the value of the VALUE attribute of the selected INPUT element.		1	0	0	0		
WML-2.0-con-189	When the WML:NAME attribute of the INPUT element is assigned, the TYPE attribute of the INPUT element is set to 'radio', the value of the VALUE attribute of the selected INPUT element is not specified, the named variable must be set to an empty string.		1	0	0	0		
WML-2.0-con-190	When the NAME attribute of the SELECT element is used to name the control variable, the TYPE attribute of the INPUT element is set to 'radio', the named variable must hold the value of the VALUE attribute of the selected INPUT element.		0	1	0	0		
WML-2.0-con-191	When the NAME attribute of the SELECT element is used to name the control variable, the TYPE attribute of the INPUT element is set to 'radio', the value of the VALUE attribute of the selected INPUT element is not specified, the named variable must be set to an empty string.		1	0	0	0		
WML-2.0-con-192	When the reset button is launched, the user agent must reset the current value of each control to its initial value.		1	0	0	0		
WML-2.0-con-193	When navigating away from a card containing a form, the user agent must commit the current value of each form control to its form control variable.		1	0	0	0		
WML-2.0-con-194	When submitting a form, and the ACTION attribute of the FORM element is set to 'get', it must define the method used to submit the data.		1	0	0	0		

WML-2.0-con-195	When submitting a form, and the ACTION attribute of the FORM element is set to 'post', it must define the method used to submit the data.		1	0	0	0		
WML-2.0-con-196	When submitting a form, any form control variables must be sent.		1	0	0	0		
WML-2.0-con-197	When calling a document of the type WML2(application/wml+xml), the user agent must process the document according to the WML variable attribute expression language.		1	0	0	0		
WML-2.0-con-198	When using the WML variable attribute expression language, the variable names must consist of a US-ASCII letter or underscore, followed by zero or more letters, digits or underscores.		1	0	0	0		
WML-2.0-con-199	When using the WML variable attribute expression language, and the 'noesc' conversion is used, the user agent must not apply any escapement to the variable.		1	0	0	0		
WML-2.0-con-200	When using the WML variable attribute expression language, and 'escape' conversion is used, the user agent must apply URL encoding to the string before using it in the current context.		1	0	0	0		
WML-2.0-con-201	When using the WML variable attribute expression language, the 'unesc' conversion is used, the user agent must reverse URL encoding from the string before using it in the current context.		1	0	0	0		
WML-2.0-con-202	When a WML2.0 user agent loads an XHTML Basic document, and that document content contains the '$', it must treat is as an ordinary text character, and not a variable.		1	0	0	0		
WML-2.0-con-203	When a WML2.0 user agent loads an XHTML Basic document, and that document content contains the '$$', it must treat them as ordinary text characters, and show both dollar signs.		1	0	0	0		

WML-2.0-con-204	When using the WML variable attribute expression language, in the context of variable parsing, all XML syntax must have a higher precedence than the variable syntax(e.g. entity substitution occurs before the variable substitution syntax is parsed).		0	1	0	0		
WML-2.0-con-205	When a author wishes to include a '$' character in an attribute value in a WML2 document, they must escape the dollar sign by using two dollar signs(e.g. $$).		1	0	0	0		
WML-2.0-con-206	When a variable is declared within a form control element using the WML:NAME attribute, the variable must be globally scoped.		1	0	0	0		
WML-2.0-con-207	When a form control element is placed inside of a FORM element, and that control names a variable with the WML:NAME attribute, that form control variable must be globally scoped.		1	0	0	0		
WML-2.0-con-208	When a variable is declared with the WML:SETVAR element, the variable must be globally scoped, regardless of the positioning of the WML:SETVAR element relative to a FORM element.		1	0	0	0		
WML-2.0-con-209	When the value of an attribute is not to be evaluated as part of the WML variable expression language, and it is to contain a literal '$' character, it must use an escaped literal dollar sign to prevent the creation.		1	0	0	0		
WML-2.0-con-210	When a variable reference uses invalid syntax, the document must be treated as being in error.		1	0	0	0		
WML-2.0-con-211	When a WML:ONEVENT element has its TYPE attribute set to 'timer' and the timer expires, the associated event must occur.		1	0	0	0		
WML-2.0-con-212	When a card is entered and a WML:ONEVENT elements TYPE attribute is set to 'enterforward', the associated event must occur.		1	0	0	0		
WML-2.0-con-213	When a card is entered via backward navigation (e.g. a WML:PREV task), and an ENTERBACKWARD event handler is set, the associated event must occur.		1	0	0	0		

WML-2.0-con-214	When an OPTION element contains a WML:ONEVENT element with the TYPE attribute set to 'pick', and the option is chosen, the event must be processed.		1	0	0	0		
WML-2.0-con-215	When a WML:CARD element has its ONTIMER attribute set to a URL, when the timer expires the user agent must automatically navigate to the given URL.		1	0	0	0		
WML-2.0-con-216	When a WML:CARD element has its ONENTERFORWARD attribute set to a URL, and the card is entered, the user agent must automatically navigate to the given URL.		1	0	0	0		
WML-2.0-con-217	When a WML:CARD has its ONENTERBACKWARD attribute set to a URL, and the card is entered via backward navigation, the user agent must navigate to the URL stated.		0	1	0	0		
WML-2.0-con-218	When an OPTION element contains a WML:ONPICK attribute set to a URL, and the option is chosen, the event must be processed.		1	0	0	0		
WML-2.0-con-219	When the user agent handles an event, it must treat the attribute syntax as an abbreviated form of the WML:ONEVENT element, where the attribute name is mapped to the TYPE attribute of the WML:ONEVENT element.		1	0	0	0		
WML-2.0-con-220	When an event binding is used, it must be scoped to the element in which it is declared.		0	1	0	0		
WML-2.0-con-221	When an event binding element specifies an event type that does not apply to its parent element, the user agent must ignore the event binding.		1	0	0	0		
WML-2.0-con-222	When conflicting event bindings are used within an element, they must be treated as an error.		1	0	0	0		
WML-2.0-con-223	When the WML:DO element is used, the only way to bind a task to the associated event is to place the task element within the WML:DO element.		1	0	0	0		
WML-2.0-con-224	When a card level binding is set, the binding must appear inside a WML:CARD element and specify event-processing behaviour for that particular card.		1	0	0	0		

WML-2.0-con-225	When a document level binding is set, the binding must appear inside the HTML element and specify event-processing behaviour for each card in the document.		0	1	0	0		
WML-2.0-con-226	When a document holds an event binding at both document level and card level, the card level event binding must override the document level event binding.		1	0	0	0		
WML-2.0-con-227	When a card level-binding shadows a document-level binding, and the card-level binding specifies the NOOP task, the event binding for that event must be completely masked.		0	0	0	1	63	
WML-2.0-con-228	When a card-level binding specifies a NOOP task but does not override another binding, then the binding for that event must be masked and similarly ignored with no side effects.		1	0	0	0		
WML-2.0-con-229	When a document-level binding specifies a NOOP task but does not override and is not overridden by another binding, then the binding for that event must be masked and similarly ignored with no side effects.		1	0	0	0		
WML-2.0-con-230	The WML2 user agent must provide the end user access to a back key at all times.		1	0	0	0		
WML-2.0-con-231	When the back key is activated by the end user, the user agent must execute a prev task.		1	0	0	0		
WML-2.0-con-232	When the user activates a WML:DO element, the user agent must execute the associated task.		1	0	0	0		
WML-2.0-con-233	When the WML:DO element is set, the user agent must use the WML:DO element's TYPE attribute to provide a suitable mapping onto a physical user interface construct.		1	0	0	0		
WML-2.0-con-234	When the WML:DO element's TYPE attribute is set to 'accept', the user agent must select the default presentation associated with that role.		1	0	0	0		
WML-2.0-con-235	When the WML:DO element's TYPE attribute is set to 'prev', the user agent must select the default presentation associated with that role.		1	0	0	0		

WML-2.0-con-236	When the WML:DO element's TYPE attribute is set to 'help', the user agent must select the default presentation associated with that role.		1	0	0	0		
WML-2.0-con-237	When the WML:DO element's TYPE attribute is set to 'reset', the user agent must select the default presentation associated with that role.		1	0	0	0		
WML-2.0-con-238	When the WML:DO element's TYPE attribute is set to 'options', the user agent must select the default presentation associated with that role.		1	0	0	0		
WML-2.0-con-239	When the WML:DO element's TYPE attribute is set to 'delete', the user agent must select the default presentation associated with that role.		1	0	0	0		
WML-2.0-con-240	When the WML:DO element's TYPE attribute is set to 'unknown', the user agent must select the default presentation associated with that role.		1	0	0	0		
WML-2.0-con-241	When the WML:DO element's TYPE attribute is set to 'prev', and the default presentation is selected, the user agent must override the default behavior of the BACK key with the task associated with the WML:DO element.		1	0	0	0		
WML-2.0-con-242	When there is more than one WML:DO element with it's TYPE attribute is set to 'prev', the first WML:DO element in the document order must override the BACK key.		1	0	0	0		
WML-2.0-con-243	When a card is entered and the card contains a WML:TIMER element, the timer must be initialised and started.		1	0	0	0		
WML-2.0-con-244	When a timer is started, it must decrement from its initial value, triggering the delivery of a WML-ONTIMER event on transition from a value of one to zero.		1	0	0	0		
WML-2.0-con-245	When a timer is running within a card, and the timer expires before the user exits the card, a TIMER event must occur.		1	0	0	0		

WML-2.0-con-246	When a WML:TIMER element's VALUE attribute is not set to a positive number, the user agent must ignore the WML:TIMER element.		1	0	0	0		
WML-2.0-con-247	When a refresh task is invoked while a timer is running, the task must stop the timer, commit it's value to the context, and update the user agent accordingly.		1	0	0	0		
WML-2.0-con-248	When a refresh task is invoked while a timer is running, on completion of the refresh task, the user agent must treat it as an entry to a card and the timer must be restarted.		1	0	0	0		
WML-2.0-con-249	When a user agent encounters an element it does not recognize, it must continue to process the children of that element.		1	0	0	0		
WML-2.0-con-250	When a user agent encounters an element it does not recognize, and that element contains content that is text, the text must be presented to the user.		1	0	0	0		
WML-2.0-con-251	When a user agent encounters an attribute it does not recognize, it must ignore the entire attribute specification(i.e. the attribute and it's value).		1	0	0	0		
WML-2.0-con-252	When a user agent encounters an attribute value it does not recognize, it must use the default attribute value(if there is one for that element).		0	1	0	0		
WML-2.0-con-253	When a document is entered, and the WML:NEWCONTEXT attribute is set to 'true', the user agent context must be initialised.		1	0	0	0		
WML-2.0-con-254	When a document is constructed, multiple WML:CARD's must be allowed.		1	0	0	0		
WML-2.0-con-255	When a document is constructed, each WML:CARD in the document must represent an individual presentation and/or interaction with the user.		1	0	0	0		
WML-2.0-con-256	When a card is entered, and the NEWCONTEXT attribute is set to 'true', the user agent must reinitialise the browser context.		1	0	0	0		

WML-2.0-con-257	When using a text input control, the WML:FORMAT attribute must specify an input mask for the control.		1	0	0	0		
WML-2.0-con-258	When processing a text input control that contains a WML:FORMAT attribute, the user agent must ignore invalid masks.		1	0	0	0		
WML-2.0-con-259	When using a text input control, the user agent must be able to support ASCII graphic characters of the Unicode Basic Latin block.		1	0	0	0		
WML-2.0-con-260	When the WML:FORMAT attribute of the INPUT element is specified as "*A", the text input must be limited to upper-case letters, symbols or punctuation characters. Numeric characters are excluded.		1	0	0	0		
WML-2.0-con-261	When the FORMAT attribute of the INPUT element is specified as "*a", the text input must be limited to lower-case letters, symbols or punctuation characters.		1	0	0	0		
WML-2.0-con-262	When the FORMAT attribute of the INPUT element is specified as "*N", the text input must be limited to numeric characters.		1	0	0	0		
WML-2.0-con-263	When the FORMAT attribute of the INPUT element is specified as "*n", the text input must be limited to numeric, symbol, or punctuation characters.		1	0	0	0		
WML-2.0-con-264	When the FORMAT attribute of the INPUT element is specified as "*X", the text input must be limited to uppercase letters, numeric characters, symbols, or punctuation characters.		1	0	0	0		
WML-2.0-con-265	When the FORMAT attribute of the INPUT element is specified as "*x", the text input must be limited to lowercase letters, numeric characters, symbols, or punctuation characters.		1	0	0	0		
WML-2.0-con-266	When the FORMAT attribute of the INPUT element is specified as "nf", where n must be a natural number and f is one of the format characters "aANXx", the number of input characters is limited to n and the type is limited to f.		1	0	0	0		

WML-2.0-con-267	When the FORMAT attribute of the INPUT element contains a backslash "\" followed by a character c, that character must be displayed in the input field at the position indicated.		1	0	0	0		
WML-2.0-con-268	When WML:EMPTYOK is true, input is not required even if the format mask would otherwise require it.		1	0	0	0		
WML-2.0-con-269	When EMPTYOK is false, input is required even if the format mask would otherwise not require it.		1	0	0	0		
WML-2.0-con-270	When a FORMAT attribute of the INPUT elmement is specified such that it requires input, e.g. "M*M", and the WML:EMPTYOK attribute is not specified, the FORMAT attribute must fully define the input requirement(the user agent must treat the WML:EMPTYOK as the implied value of 'false').		1	0	0	0		
WML-2.0-con-271	When the FORMAT attribute of the INPUT element is specified such that it allows empty input, e.g. "*f", and the WML:EMPTYOK attribute is not specified, the format attribute must fully define the input requirement(the user agent must treat the WML:EMPTYOK as the implied value of 'true'.		1	0	0	0		
WML-2.0-con-272	When a multiple-selection list generates a pick event, due to the user selecting an option, user must be taken to the card specified by the URI, when the option is selected.		1	0	0	0		
WML-2.0-con-273	When an multiple-selection list generates a pick event, due to the user de-selecting an option, user must be taken to the card specified by the URI, when the option is de-selected.		1	0	0	0		
WML-2.0-con-274	When an multiple-selection list generates a pick event, due to the user selecting an option, user must be taken to the card specified by the URI, when the option is selected.		1	0	0	0		
WML-2.0-con-275	When the IMG element is used, and there is no internal representation for the image, the image must be downloaded from the URI specified in the SRC attribute.		1	0	0	0		
WML-2.0-con-276	When a WML:ONEVENT element is used, it must be scoped to the element in which it is declared.		1	0	0	0		

WML-2.0-con-277	When the WML:ONEVENT element is used and the TYPE attribute does not correspond to a legal event for the immediately enclosing element, the user agent must ignore the WML:ONEVENT element.		1	0	0	0		
WML-2.0-con-278	When the user activates the WML:ANCHOR element, the user agent must execute the associated task.		1	0	0	0		
WML-2.0-con-279	When the WML:ANCHOR element is used, the user agent must accept the WML:ANCHOR element anywhere within the text flow.		1	0	0	0		
WML-2.0-con-280	When a document does not contain a WML:ACCESS element, access control must be disabled.		1	0	0	0		
WML-2.0-con-281	When the WML:ACCESS element is used, and the DOMAIN attribute is set, the entire element of each sub-domain must match each element exactly(e.g., www.wapforum.org shall match wapforum.org).		1	0	0	0		
WML-2.0-con-282	When the WML:ACCESS element is used, and the PATH attribute is set, the entire path elements must match each element exactly to gain access(e.g., /X/Y matches path="/X attribute, but does not match path="/XZ" attribute).		0	1	0	0		
WML-2.0-con-283	When the WML:ACCESS element is used without specifying the DOMAIN attribute, the DOMAIN attribute must default to the current document's domain.		0	1	0	0		
WML-2.0-con-284	When the WML:ACCESS element is used without specifying the PATH attribute, the PATH attribute must default to the value '/'.		1	0	0	0		
WML-2.0-con-285	When the WML:ACCESS element is used, and the PATH attribute is set to a relative URI, and the URI matches the path to be accessed (once the user agent has converted the relative URI into an absolute path), the user agent must allow access to the destination requested.		0	1	0	0		
WML-2.0-con-286	When the WML:ACCESS element is used, and the DOMAIN and PATH attribute are set, the user agent must verify that the referring document's URI matches the values of the attributes.		0	1	0	0		

WML-2.0-con-287	When the SENDREFERER attribute of the WML:GO element is set to true, the user agent must send the URL of the deck containing this task to the server.		0	1	0	0		
WML-2.0-con-288	When the cache-control attribure is present within the WML:GO element and its CACHE-CONTROL attribute value is set to "no-cache", the client must reload the URL from the origin server.		1	0	0	0		
WML-2.0-con-289	When the cache-control attribute is present within the WML:GO element, the HTTP cache-control header must be added to the request with the same value as specified in the attribute.		1	0	0	0		
WML-2.0-con-290	When the WML:GO element is specified, it must declare a GO task, indicating navigation to a URI.		1	0	0	0		
WML-2.0-con-291	When the WML:GO element's HREF attribute names a WML card, the user agent must display the destination card.		1	0	0	0		
WML-2.0-con-292	When the WML:GO element is specified, the METHOD attribute is set to 'get', and the ENCTYPE attribute is set to 'application/vnd.wap.wml.form.urlencoded', all field names and values sent must be escaped using URI-escaping and assembled into an application/x-www-form-urlencoded content type.		0	1	0	0		
WML-2.0-con-293	When the WML:GO element is specified, the METHOD attribute is set to 'get', the ENCTYPE attribute is set to 'application/vnd.wap.wml.form.urlencoded', the WML:POSTFIELD element is used to name a field, and the value of the WML:POSTFIELD's VALUE attribute references the control variable for a multiple-selection control, the field VALUE must be submitted as a semicolon-separated list of assigned values.		1	0	0	0		
WML-2.0-con-294	When the WML:GO element is specified, the METHOD attribute is set to 'get', and the ENCTYPE attribute is set to 'application/vnd.wap.wml.form.urlencoded', data sent to the server must be a valid query component with the original query part and the postfields combined.		1	0	0	0		

WML-2.0-con-295	When the WML:GO element is specified, the METHOD attribute is set to 'get', and the ENCTYPE attribute is set to 'application/x-www-form-urlencoded', all field names and values sent must e escaped using URI-escaping and assembled into an application/x-www-form-urlencoded content type.		0	1	0	0		
WML-2.0-con-296	When the WML:GO element is specified, the METHOD attribute is set to 'get', the ENCTYPE attribute is set to 'application/x-www-form-urlencoded', the WML:POSTFIELD element is used to name a field, and the value of the WML:POSTFIELD's VALUE attribute references the control variable for a multiple-selection control, the field value must be submitted as specified in [HTML4] where one name/value pair is included for each item in the list.		1	0	0	0		
WML-2.0-con-297	When the WML:GO element is specified, the METHOD attribute is set to 'get', and the ENCTYPE attribute is set to 'application/x-www-form-urlencoded', data sent to the server must be a valid query component with the original query part and the postfields combined.		1	0	0	0		
WML-2.0-con-298	When the WML:GO element is specified, the METHOD attribute is set to 'get', and the ENCTYPE attribute is set to 'multipart/form-data', the user agent must ignore the WML:GO element.		1	0	0	0		
WML-2.0-con-299	When the WML:GO element is specified, the METHOD attribute is set to 'post', and the ENCTYPE attribute is set to 'application/vnd.wap.wml.form.urlencoded', all field names and values sent must be escaped using URI-escaping and assembled into an application/x-www-form-urlencoded content type.		0	1	0	0		

WML-2.0-con-300	When the WML:GO element is specified, the METHOD attribute is set to 'post', the ENCTYPE attribute is set to 'application/vnd.wap.wml.form.urlencoded', the WML:POSTFIELD element is used to name a field, and the value of the WML:POSTFIELD's VALUE attribute references the control variable for a multiple-selection control, the field VALUE must be submitted as a semicolon-separated list of assigned values.		1	0	0	0		
WML-2.0-con-301	When the WML:GO element is specified, the METHOD attribute is set to 'post', and the ENCTYPE attribute is set to 'application/vnd.wap.wml.form.urlencoded', any submission data must be sent as the body of the HTTP POST request.		1	0	0	0		
WML-2.0-con-302	When the WML:GO element is specified, the METHOD attribute is set to 'post', and the ENCTYPE attribute is set to 'application/vnd.wap.wml.form.urlencoded', the CONTENT-TYPE header must include the CHARSET parameter to indicate the character encoding.		0	1	0	0		
WML-2.0-con-303	When the WML:GO element is specified, the METHOD attribute is set to 'post', and the ENCTYPE attribute is set to 'application/x-www-form-urlencoded', all field names and values sent must be escaped using URI-escaping and assembled into an application/x-www-form-urlencoded content type.		0	1	0	0		
WML-2.0-con-304	When the WML:GO element is specified, the METHOD attribute is set to 'post', the ENCTYPE attribute is set to 'application/x-www-form-urlencoded', the WML:POSTFIELD element is used to name a field, and the value of the WML:POSTFIELD's VALUE attribute references the control variable for a multiple-selection control, the field value must be submitted as specified in [HTML4] where one name/value pair is included for each item in the list.		1	0	0	0		

WML-2.0-con-305	When the WML:GO element is specified, the METHOD attribute is set to 'post', and the ENCTYPE attribute is set to 'application/x-www-form-urlencoded', any submission data must be sent as the body of the HTTP POST request.		1	0	0	0		
WML-2.0-con-306	When the WML:GO element is specified, the METHOD attribute is set to 'post', and the ENCTYPE attribute is set to 'multipart/form-data', all field names and values sent must be encoded as a 'multipart/form-data' content type.		0	1	0	0		
WML-2.0-con-307	When the WML:GO element is specified, the METHOD attribute is set to 'post', and the ENCTYPE attribute is set to 'multipart/form-data', any submission data must be sent as the body of the HTTP POST request.		0	0	0	1		
WML-2.0-con-308	When the WML:GO element is specified, the METHOD attribute is set to 'post', and the ENCTYPE attribute is set to 'multipart/form-data', the WML:POSTFIELD element is used to name a field, and the value of the WML:POSTFIELD's VALUE attribute references the control variable for a multiple-selection control, the field value must be submitted as specified in [HTML4] where one part is included for each item in the list.		0	0	0	1		
WML-2.0-con-309	When the WML:NOOP element is specified, no operation should be carried out when the task is called.		1	0	0	0		
WML-2.0-con-310	When the WML:POSTFIELD is used, the NAME attribute must specify the field name.		1	0	0	0		
WML-2.0-con-311	When the WML:POSTFIELD is used, the VALUE attribute must specify the field value.		1	0	0	0		
WML-2.0-con-312	When the WML:SETVAR element is used, the NAME attribute must specify the name of the variable.		1	0	0	0		
WML-2.0-con-313	When the WML:SETVAR element is used, the VALUE attribute must specify the expression to evaluate and assign to the variable.		1	0	0	0		
WML-2.0-con-314	When a WML:SETVAR element contains a NAME attribute that does not evaluate to a legal variable name, the element is ignored.		1	0	0	0		

WML-2.0-con-315	When a WML:SETVAR element contains a VALUE attribute that does not evaluate to a legal expression at runtime, the element is ignored.		0	1	0	0		
WML-2.0-con-316	When a WML:SETVAR element contains a NAME attribute and it's value is a variable expression, the user agent must correctly evaluate the variable expression specified.		1	0	0	0		
WML-2.0-con-317	When a WML:SETVAR element contains a VALUE attribute and it's value is a variable expression, the user agent must correctly evaluate the variable expression specified.		1	0	0	0		
WML-2.0-con-318	When the WML:GETVAR element is used, the required NAME attribute must specify the name of the variable whose value is the substitution text.		1	0	0	0		
WML-2.0-con-319	When the WML:GETVAR element is used, and the variable is not set, the substitution text must be an empty string.		1	0	0	0		
WML-2.0-con-320	When the WML:GETVAR element is used, and the NAME attribute does not evaluate to a legal variable name, the substitution text must be a empty string.		1	0	0	0		
WML-2.0-con-321	When the WML:GETVAR element is specified, the user agent must replace the element with the substitution text.		1	0	0	0		
WML-2.0-con-322	When the CONVERSION attribute of the WML:GETVAR element is specified, then it must specify the conversion to be applied to the value of the variable.		1	0	0	0		
WML-2.0-con-323	When the CONVERSION attribute of the WML:GETVAR element is set to 'noesc', the user agent must not apply any escapement to the variable.		1	0	0	0		
WML-2.0-con-324	When the CONVERSION attribute of the WML:GETVAR element is set to 'escape', the user agent must apply URL encoding to the string before using it in the current context.		1	0	0	0		

WML-2.0-con-325	When the CONVERSION attribute of the WML:GETVAR element is set to 'unesc', the user agent must must reverse URL encoding from the string before using it in the current context.		1	0	0	0		
WML-2.0-con-326	When the WML:GETVAR element is used, no substitution of elements must be possible.		1	0	0	0		
WML-2.0-con-327	When the WML:GETVAR element is used, no substitution of attributes must be possible.		0	0	0	1		
WML-2.0-con-328	When the WML:GETVAR element references a variable, the user agent must not attempt to process the resulting string as XML markup.		1	0	0	0		
WML-2.0-con-329	When the NAME attribute of the WML:TIMER element is specified, it must define the name of the variable to be set with the value of the timer.		1	0	0	0		
WML-2.0-con-330	When the NAME attribute of the WML:TIMER element is specified, the named variable must be used to set the timeout period upon timer initialisation.		1	0	0	0		
WML-2.0-con-331	When the NAME attribute of the WML:TIMER element is specified, the named variable must be set with the current timer value when the card is exited.		0	0	0	1		
WML-2.0-con-332	When the NAME attribute of the WML:TIMER element is specified, the named variable must be set to the value of '0' when the timer expires.		0	1	0	0		
WML-2.0-con-333	When the VALUE attribute of the WML:TIMER element is specified, it must hold the default value of the variable named in the NAME attribute.		1	0	0	0		
WML-2.0-con-334	When the NAME attribute of the WML:TIMER element is specified, and the variable value is not already set, the variable must be set with the value of the VALUE attribute.		1	0	0	0		
WML-2.0-con-335	When the NAME attribute of the WML:TIMER element is specified, and the variable value is already set, the VALUE attribute must be ignored.		1	0	0	0		

WML-2.0-con-336	When the NAME attribute of the WML:TIMER element is not specified, the timeout is always initialised to the value specified in the VALUE attribute.		1	0	0	0		
WAE-2.2-con-101	To test support of the required content types		0	0	0	0		
WAE-2.2-con-102	To test support for graphical images		0	0	0	0		
WAE-2.2-con-103	To test Vcard functionality		0	0	0	0		
WAE-2.2-con-104	To test Vcal functionality		0	0	0	0		
WAE-2.2-con-105	To test Multipart Message Encoding functionality		0	0	0	0		
WAE-2.2-con-106	To test support for minimum URI length		0	0	0	0		
WAE-2.2-con-107	To test the support for charactersets UTF-8 and UTF-16		0	0	0	0		
WAE-2.2-con-108	Support for HTTP/1.1 Basic Authentication		0	0	0	0		
WAE-2.2-con-109	To test the Navigation History Model		0	0	0	0		
WAE-2.2-con-110	To test the Navigation History Back Key functionality.		0	0	0	0		
WAE-2.2-con-111	To test the conversion from standard internet Multipart to WAP specific Multipart via WSP		0	0	0	0		
WAE-2.2-con-112	To test the support for the minimum URI length.		0	0	0	0		
WAE-2.2-con-113	Server support for UTF-8 and UTF-16 characterset.		0	0	0	0		
WAE-2.2-con-114	Unknown character behaviour.		0	0	0	0		
WAE-2.2-con-115	Support for Crypto Lib encoding		0	0	0	0		
WAE-2.2-con-116	No TLS / WTLS support available		0	0	0	0		
WAE-2.2-con-117	Support for WML2		0	0	0	0		
WAE-2.2-con-201	To test support of the required content types		0	0	0	0		
WAE-2.2-con-202	Support for maintaining context between WML1, and WML2		0	0	0	0		
WAE-2.2-con-203	To test support for graphical images		0	0	0	0		
WAE-2.2-con-104	To test Vcard functionality		0	0	0	0		
WAE-2.2-con-205	To test Vcal functionality		0	0	0	0		

WAE-2.2-con-106	To test Multipart Message Encoding functionality		0	0	0	0		
WAE-2.2-con-207	To test support for minimum URI length		0	0	0	0		
WAE-2.2-con-108	To test the support for charactersets UTF-8 and UTF-16		0	0	0	0		
WAE-2.2-con-209	Support for HTTP/1.1 Basic Authentication		0	0	0	0		
WAE-2.2-con-210	To test the Navigation History Model		0	0	0	0		
WAE-2.2-con-211	To test the Navigation History Back Key functionality.		0	0	0	0		
WAE-2.2-con-212	To test the conversion from standard internet Multipart to WAP specific Multipart via WSP		0	0	0	0		
WAE-2.2-con-213	To test the support for the minimum URI length.		0	0	0	0		
WAE-2.2-con-214	Server support for UTF-8 and UTF-16 characterset.		0	0	0	0		
WAE-2.2-con-215	Unknown character behaviour.		0	0	0	0		
WAE-2.2-con-216	Support for Crypto Lib encoding		0	0	0	0		
WAE-2.2-con-217	No TLS / WTLS support available		0	0	0	0		
WMLS-2.0-con-1	WMLScript variables are case-sensitive.		3	0	0	0		
WMLS-2.0-con-2	WMLScript function names are case-sensitive.		3	0	0	0		
WMLS-2.0-con-3	Ensure that whitespace tokens <TAB>, <VT>, <FF>, <SP>, <LF> and <CR> are ignored between program tokens.		3	0	0	0		
WMLS-2.0-con-4	When a line starts with a double-slash (//), it is a single line comment. This comment is ignored by the script processor.		3	0	0	0		
WMLS-2.0-con-5	When a line starts with a multi-line comment delimiter (/*), the block between that delimiter and the closing comment delimiter (*/) is ignored by the script processor.		3	0	0	0		
WMLS-2.0-con-6	Floating point literals can be defined using the syntax "DecimalIntegerLiteral. DecimalDigits ExponentPart".		3	0	0	0		
WMLS-2.0-con-7	Floating point literals can be defined using the syntax ". DecimalDigits ExponentPart".		3	0	0	0		

WMLS-2.0-con-8	Floating point literals can be defined using the syntax "DecimalIntegerLiteral ExponentPart".		3	0	0	0		
WMLS-2.0-con-9	Floating point literals can be defined using the syntax "DecimalIntegerLiteral . ExponentPart".		3	0	0	0		
WMLS-2.0-con-10	When an escape sequence (as described in the table in 6.1.5.3) is encountered within a string literal, it is replaced with its character equivalent in the literal.		3	0	0	0		
WMLS-2.0-con-11	String literals of length 0 can be defined.		3	0	0	0		
WMLS-2.0-con-12	Boolean literals of true and false can be defined using "true" and "false".		3	0	0	0		
WMLS-2.0-con-13	An invalid literal can be defined using "invalid".		3	0	0	0		
WMLS-2.0-con-14	Function names have their own name space.		3	0	0	0		
WMLS-2.0-con-15	Function parameters have their own name space.		3	0	0	0		
WMLS-2.0-con-16	Variables have their own name space.		3	0	0	0		
WMLS-2.0-con-17	Pragmas have their own name space.		3	0	0	0		
WMLS-2.0-con-18	Within the function where it is declared, a WMLScript variable is visible outside of the block statement where it has been declared.		3	0	0	0		
WMLS-2.0-con-19	A WMLScript variable has a value from its declaration up to the end of the function.		3	0	0	0		
WMLS-2.0-con-20	A WMLScript variable can hold integer values ranging from Lang.minInt() to Lang.maxInt().		3	0	0	0		
WMLS-2.0-con-21	If an operation results in an integer value outside the range of supported integer values then the result is 'invalid'.		3	0	0	0		
WMLS-2.0-con-22	A WMLScript variable can hold floating-point positive values ranging from Float.minFloat() to Float.maxFloat().		3	0	0	0		

WMLS-2.0-con-23	If an operation results in a floating-point number outside the range of supported floating-point values then the result is 'invalid'.		3	0	0	0		
WMLS-2.0-con-24	A WMLScript variable which is initialized to, or assigned, a positive floating-point value lower than Float.minFloat() gets set to 0.0.		3	0	0	0		
WMLS-2.0-con-25	A WMLScript variable of value '-0.0' is equal to a variable of value '+0.0'.		3	0	0	0		
WMLS-2.0-con-26	The '=' operator assigns the rhs value to the WMLScript variable on the lhs.		3	0	0	0		
WMLS-2.0-con-27	The '+=' operator applies the '+' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-28	The '-=' operator applies the '-' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-29	The '*=' operator applies the '*' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-30	The '/=' operator applies the '/' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-31	The 'div=' operator applies the 'div' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-32	The '%=' operator applies the '%' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		

WMLS-2.0-con-33	The '<<=' operator applies the '<<' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-34	The '>>=' operator applies the '>>' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-35	The '>>>=' operator applies the '>>>' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-36	The '&=' operator applies the '&' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-37	The '^=' operator applies the '^' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-38	The '|=' operator applies the '|' operator with the value of the WMLScript variable on the lhs as the first operand and the value on the rhs as the second operand and assigns the result to this variable.		3	0	0	0		
WMLS-2.0-con-39	An assignment operator of one variable does not change the binding of any other variable.		3	0	0	0		
WMLS-2.0-con-40	When both operands are numbers (integer or floating-point values), the '+' operator adds the value of the first operand to the value of the second operand.		3	0	0	0		
WMLS-2.0-con-41	The '-' operator subtracts the value of the second operand from the value of the first operand.		3	0	0	0		
WMLS-2.0-con-42	The '*' operator multiplies the value of the first operand by the value of the second operand.		3	0	0	0		
WMLS-2.0-con-43	The '/' operator divides the value of the first operand by the value of the second operand.		3	0	0	0		

WMLS-2.0-con-44	The 'div' operator divides the value of the first operand by the value of the second operand and returns the integral part of the result.		3	0	0	0		
WMLS-2.0-con-45	The '%' operator integer divides the value of the first operand by the value of the second operand and returns the remainder part of the result.		3	0	0	0		
WMLS-2.0-con-46	The '<<' operator left shifts the bits of the value of the first operand by the number of bits specified by the value of the second operand, filling the shifted bits with a zero bit.		3	0	0	0		
WMLS-2.0-con-47	The '>>' operator right shifts the bits of the value of the first operand by the number of bits specified by the value of the second operand, filling the shifted bits with the sign bit of the first operand.		3	0	0	0		
WMLS-2.0-con-48	The '>>>' operator right shifts the bits of the value of the first operand by the number of bits specified by the value of the second operand, filling the shifted bits with a zero bit.		3	0	0	0		
WMLS-2.0-con-49	The '&' operator performs a bitwise logical AND operation on the bits of the value of the first operand and the bits of the value of the second operand.		3	0	0	0		
WMLS-2.0-con-50	The '|' operator performs a bitwise logical OR operation on the bits of the value of the first operand and the bits of the value of the second operand.		3	0	0	0		
WMLS-2.0-con-51	The '^' operator performs a bitwise logical XOR operation on the bits of the value of the first operand and the bits of the value of the second operand.		3	0	0	0		
WMLS-2.0-con-52	The '~' operator performs a bitwise logical NOT operation on the bits of the value of the operand.		3	0	0	0		
WMLS-2.0-con-53	The unary '+' operator does not change the value of its operand.		3	0	0	0		
WMLS-2.0-con-54	The unary '-' operator changes the sign bit of its operand.		3	0	0	0		
WMLS-2.0-con-55	When a prefix '++' operator is used, the value of the variable is incremented by 1 before the evaluation of the complete expression.		3	0	0	0		

WMLS-2.0-con-56	When a postfix '++' operator is used, the value of the variable is incremented by 1 after the evaluation of the complete expression.		3	0	0	0		
WMLS-2.0-con-57	When a prefix '--' operator is used, the value of the variable is decremented by 1 before the evaluation of the complete expression.		3	0	0	0		
WMLS-2.0-con-58	When a postfix '--' operator is used, the value of the variable is decremented by 1 after the evaluation of the complete expression.		3	0	0	0		
WMLS-2.0-con-59	When both operands are strings, the '+' operator concatentates the strings together.		3	0	0	0		
WMLS-2.0-con-60	If the first operand of the '&&' operator evaluates to 'true' then the result of the operation is the result of the evaluation of the second operand.		3	0	0	0		
WMLS-2.0-con-61	If the first operand of the '&&' operator evaluates to 'false' then the result of the operation is 'false' and the second operand is not evaluated.		3	0	0	0		
WMLS-2.0-con-62	If the first operand of the '||' operator evaluates to 'false' then the result of the operation is the result of the evaluation of the second operand.		3	0	0	0		
WMLS-2.0-con-63	If the first operand of the '||' operator evaluates to 'true' then the result of the operation is 'true' and the second operand is not evaluated.		3	0	0	0		
WMLS-2.0-con-64	If the operand of the unary '!' operator evaluates to 'true', then the result of the operation is 'false'. If the operand evaluates to 'false', then the result is 'true'.		3	0	0	0		
WMLS-2.0-con-65	If the first operand of the '&&' operator evaluates to 'invalid' then the result of the operation is 'invalid' and the second operand is not evaluated.		3	0	0	0		
WMLS-2.0-con-66	If the first operand of the '||' operator evaluates to 'invalid' then the result of the operation is 'invalid' and the second operand is not evaluated.		3	0	0	0		

WMLS-2.0-con-67	When the '+' operator is applied to string operands, the result of the operation is the concatenation of both operands.		3	0	0	0		
WMLS-2.0-con-68	When the '+=' operator is applied to string operands, the result of the operation is the concatenation of both operands.		3	0	0	0		
WMLS-2.0-con-69	The boolean literal 'true' is larger than 'false'.		3	0	0	0		
WMLS-2.0-con-70	Comparison of integer values is based on the given integer values.		3	0	0	0		
WMLS-2.0-con-71	Comparison of floating-point values is based on the given floating-point values.		3	0	0	0		
WMLS-2.0-con-72	Comparison of string operands is based on the order of character codes of the given string values, as defined by the character set supported by the device.		3	0	0	0		
WMLS-2.0-con-73	When at least one of the operands of a comparison operator is 'invalid' then the result is 'invalid'.		3	0	0	0		
WMLS-2.0-con-74	The result of the comma operator is the result of the evaluation of the second operand.		3	0	0	0		
WMLS-2.0-con-75	If the first operand of a conditional operator '?:' evaluates to 'true', then the result of the operation is the result of the evaluation of the second operand and the third operand is not evaluated.		3	0	0	0		
WMLS-2.0-con-76	If the first operand of a conditional operator '?:' evaluates to 'false' or 'invalid', then the result of the operation is the result of the evaluation of the third operand and the second operand is not evaluated.		3	0	0	0		
WMLS-2.0-con-77	If the operand of the 'typeof' operator evaluates to an integer value, then the result of the operation is '0'.		3	0	0	0		
WMLS-2.0-con-78	If the operand of the 'typeof' operator evaluates to a floating-point value, then the result of the operation is '1'.		3	0	0	0		
WMLS-2.0-con-79	If the operand of the 'typeof' operator evaluates to a string value, then the result of the operation is '2'.		3	0	0	0		

WMLS-2.0-con-80	If the operand of the 'typeof' operator evaluates to a boolean value, then the result of the operation is '3'.		3	0	0	0		
WMLS-2.0-con-81	If the operand of the 'typeof' operator evaluates to the 'invalid' value, then the result of the operation is '4'.		3	0	0	0		
WMLS-2.0-con-82	When the typeof operator is used to identify the type of an expression an integer value is returned.		3	0	0	0		
WMLS-2.0-con-83	If the operand of the 'isvalid' operator evaluates to the 'invalid' value, then the result of the operation is the boolean literal 'false'.		3	0	0	0		
WMLS-2.0-con-84	If the operand of the 'isvalid' operator does not evaluate to the 'invalid' value, then the result of the operation is the boolean literal 'true'.		3	0	0	0		
WMLS-2.0-con-85	The WMLScript operators respect the operator precedence rules defined in the table of section 6.3.12.		3	0	0	0		
WMLS-2.0-con-86	The WMLScript operators return 'invalid' when one of the operands is 'invalid'.		3	0	0	0		
WMLS-2.0-con-87	The WMLScript operators return 'invalid' when type conversions fail.		3	0	0	0		
WMLS-2.0-con-88	The result type of the WMLScript operators is correct according to the table in section 7.3.12.		3	0	0	0		
WMLS-2.0-con-89	All parameters to functions are passed by value. A modification to the value of a parameter does not affect the value of the calling argument.		3	0	0	0		
WMLS-2.0-con-90	Function parameters are local variables that have been initialized before the execution of the function body. As for any other variable, they loose their value at the end of the function.		3	0	0	0		
WMLS-2.0-con-91	When a function call returns a return value is always returned to the calling function.		3	0	0	0		
WMLS-2.0-con-92	Functions defined in the same compilation unit can be called before they have been declared.		3	0	0	0		

WMLS-2.0-con-93	A call to a function using the 'external#function(arguments)' syntax must be resolved by calling the function defined in the external compilation unit used as prefix.		3	0	0	0		
WMLS-2.0-con-94	A function returns an empty string "" by default.		3	0	0	0		
WMLS-2.0-con-95	A function can be called as a statement.		3	0	0	0		
WMLS-2.0-con-96	When a variable statement does not contain a variable initializer, the variable is initialized to the empty string "" by default.		3	0	0	0		
WMLS-2.0-con-97	The variable initializer expression is evaluated every time the variable statement is executed.		3	0	0	0		
WMLS-2.0-con-98	If the condition expression of an if statement evaluates to 'true', then only the first statement is executed.		3	0	0	0		
WMLS-2.0-con-99	If the condition expression of an if statement evaluates to 'false' or 'invalid', then only the second optional 'else' statement is executed.		3	0	0	0		
WMLS-2.0-con-100	When 'if' statements are nested, an 'else' statement is always tied to the closest 'if' statement.		3	0	0	0		
WMLS-2.0-con-101	The condition expression of a 'while' statement is evaluated at each iteration of the statement.		2	1	0	0		
WMLS-2.0-con-102	The loop statement of a 'while' statement is executed as long as the condition expression evaluates to 'true'.		2	1	0	0		
WMLS-2.0-con-103	When the condition expression of a 'while' statement evaluates to 'false' or 'invalid', the execution continues with the statement following the loop statement.		3	0	0	0		
WMLS-2.0-con-104	The initializer expression of a 'for' statement is evaluated only once for each execution of the statement.		3	0	0	0		
WMLS-2.0-con-105	The optional condition expression of a 'for' statement is evaluated at each iteration of the statement.		2	1	0	0		
WMLS-2.0-con-106	The loop statement of a 'for' statement is executed as long as the optional condition expression evaluates to 'true'.		2	1	0	0		

WMLS-2.0-con-107	When the optional condition expression of a 'for' statement is omitted it always evaluates to 'true'.		2	1	0	0		
WMLS-2.0-con-108	When the condition expression of a 'for' statement evaluates to 'false' or 'invalid', the execution continues with the statement following the loop statement.		3	0	0	0		
WMLS-2.0-con-109	The optional increment expression is evaluated each time and after the loop statement is executed. It is evaluated before the condition expression for the next iteration.		2	1	0	0		
WMLS-2.0-con-110	When a 'break' statement is executed as part of the loop statement of a 'while' statement, the execution continues with the statement after the closest enclosing loop statement of the 'while' statement.		3	0	0	0		
WMLS-2.0-con-111	When a 'break' statement is executed as part of the loop statement of a 'for' statement, the execution continues with the statement after the closest enclosing loop statement of the 'for' statement.		3	0	0	0		
WMLS-2.0-con-112	When a 'continue' statement is executed as part of the loop statement of a 'while' statement, the execution continues with the condition expression of the closest enclosing 'while' statement.		3	0	0	0		
WMLS-2.0-con-113	When a 'continue' statement is executed as part of the loop statement of a 'for' statement, the execution continues with the increment expression of the closest enclosing 'for' statement.		3	0	0	0		
WMLS-2.0-con-114	When a return statement is executed, the execution of the function terminates and its return value is the result of the evaluation of the expression specified in the statement.		3	0	0	0		
WMLS-2.0-con-115	When a return statement is executed and there is no return expression, the execution of the function terminates and its return value is the empty string "".		3	0	0	0		
WMLS-2.0-con-116	Relative URLs defined without a hash or fragment identifier can be used as the URL for a pragma declaration.		3	0	0	0		

WMLS-2.0-con-117	When an external function is invoked, an access control check is performed to ensure that the destination compilation unit can be accessed from the current compilation unit.		3	0	0	0		
WMLS-2.0-con-118	When the 'domain' attribute of the 'access' pragma is not set in the destination compilation unit, it defaults to the domain of the current compilation unit.		3	0	0	0		
WMLS-2.0-con-119	When the 'path' attribute of the 'access' pragma is not set in the destination compilation unit, it defaults to the value "/".		3	0	0	0		
WMLS-2.0-con-120	When the 'domain' attribute of the 'access' pragma of a destination compilation unit is set to a period separated domain suffix, the domain of the current compilation unit is compared to this suffix - where each element specified must match exactly for access to be granted.		3	0	0	0		
WMLS-2.0-con-121	When the 'path' attribute of the 'access' pragma of a destination compilation unit is specified as an absolute path, it defines the path prefix that is permitted access to the compilation unit - where each element of the prefix must match that of the current compilation unit exactly for access to be granted.		3	0	0	0		
WMLS-2.0-con-122	When the 'path' attribute of the 'access' pragma of a destination compilation unit is specified as a relative path, it is converted to an absolute path by the user agent. It then defines the path prefix that is permitted access to the compilation unit - where each element of the prefix must match that of the current compilation unit exactly for access to be granted.		3	0	0	0		
WMLS-2.0-con-123	When there is no 'access' pragma in a destination compilation unit, this is equivalent to having an 'access' pragma with the 'public' attribute, and access is granted to any current compilation unit.		3	0	0	0		
WMLS-2.0-con-124	The 'typeof' operator does not perform any type conversion and returns the exact type of a variable or expression.		3	0	0	0		

WMLS-2.0-con-125	An integer value is converted to a string containing the digits that form the decimal representation of its value.		3	0	0	0		
WMLS-2.0-con-126	A floating-point value is converted to an implementation-dependent string containing the digits that form the decimal representation of its value. The resulting string representation must be equal to the original value.		3	0	0	0		
WMLS-2.0-con-127	The boolean value 'true' is converted to string "true". The boolean value 'false' is converted to string "false".		3	0	0	0		
WMLS-2.0-con-128	The value 'invalid' cannot be converted to a string value.		3	0	0	0		
WMLS-2.0-con-129	A string can be converted to an integer value only if it contains a decimal representation of an integer number.		3	0	0	0		
WMLS-2.0-con-130	A floating-point value cannot be converted to an integer value.		3	0	0	0		
WMLS-2.0-con-131	The boolean value 'true' is converted to integer value '1'. The boolean value 'false' is converted to integer value '0'.		3	0	0	0		
WMLS-2.0-con-132	The value 'invalid' cannot be converted to an integer value.		3	0	0	0		
WMLS-2.0-con-133	A string can be converted to a floating-point value only if it contains a valid decimal representation of a floating-point number.		3	0	0	0		
WMLS-2.0-con-134	An integer value is converted to a corresponding floating-point value.		3	0	0	0		
WMLS-2.0-con-135	The boolean value 'true' is converted to floating-point value '1.0'. The boolean value 'false' is converted to floating-point value '0.0'.		3	0	0	0		
WMLS-2.0-con-136	The value 'invalid' cannot be converted to a floating-point value.		3	0	0	0		
WMLS-2.0-con-137	The empty string is converted to boolean value 'false'. All other strings are converted to boolean value 'true'.		3	0	0	0		
WMLS-2.0-con-138	An integer value '0' is converted to boolean value 'false'. All other integer numbers are converted to boolean value 'true'.		3	0	0	0		

WMLS-2.0-con-139	A floating-point value '0.0' is converted to boolean value 'false'. All other floating-point numbers are converted to boolean value 'true'.		3	0	0	0		
WMLS-2.0-con-140	The value 'invalid' cannot be converted to a boolean value.		3	0	0	0		
WMLS-2.0-con-141	When an operator expects operands of type boolean, if the operand is of type boolean or can be converted to a boolean value, then the operation is performed on the boolean values and its result is returned.		3	0	0	0		
WMLS-2.0-con-142	When an operator expects operands of type Integer, if the operand is of type Integer or can be converted to an Integer value, then the operation is performed on the Integer values and its result is returned.		3	0	0	0		
WMLS-2.0-con-143	When an operator expects operands of type Float, if the operand is of type Float or can be converted to a Float value, then the operation is performed on the Float values and its result is returned.		3	0	0	0		
WMLS-2.0-con-144	When a unary operator expects operands of either Integer or Floating Point, if the operand is of type Integer or Floating Point or can be converted to a Integer or Floating Point value, then the operation is performed on the Integer or Floating Point values and its result is returned.		3	0	0	0		
WMLS-2.0-con-145	The interpreter charset can be requested by using the Lang.characterSet() library function.		3	0	0	0		
WMLS-2.0-con-146	When a call to an external WMLScript function is made via the use of a URL fragment anchor the call will fail if the function does not exist in the external compliation unit.		2	1	0	0		
WMLS-2.0-con-147	When a call to an external WMLScript function is made via the use of a URL fragment anchor the parameters list is passed to the external function.		3	0	0	0		
WMLS-2.0-con-148	When a call to an external WMLScript function is made via the use of a relative URL the calling scripts base URL is used as the base URL to identify the external compliation unit location.		3	0	0	0		

WMLSLib-con-1	When the Lang.abs function is called with a value as its argument, it returns the absolute value of that argument.		3	0	0	0		
WMLSLib-con-2	When the Lang.abs function is called with an integer as its argument, it returns an integer type reflecting the absolute value of the argument.		3	0	0	0		
WMLSLib-con-3	When the Lang.abs function is called with a floating-point value as its argument, it returns a floating-point type reflecting the absolute value of the argument.		3	0	0	0		
WMLSLib-con-4	When the Lang.abs function is called with an argument that is not of type integer or floating-point, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-5	When the Lang.min function is called with two values as arguments, it returns the smaller of the two values.		3	0	0	0		
WMLSLib-con-6	When the Lang.min function is called with two values as arguments, and the two values are equal, the first value is returned.		3	0	0	0		
WMLSLib-con-7	When the Lang.min function is called with any non-value argument(s), it returns 'invalid'.		3	0	0	0		
WMLSLib-con-8	The value and type returned by the Lang.min function must be the va value of the smallest value and of type 'integer' when that value is an integer.		3	0	0	0		
WMLSLib-con-9	The value and type returned by the Lang.min function must be the va value of the smallest value and of type 'floating point' when that value is a floating point.		3	0	0	0		
WMLSLib-con-10	When the Lang.max function is called with two values as arguments, it returns the larger of the two values.		3	0	0	0		
WMLSLib-con-11	When the Lang.max function is called with two values as arguments, and the two values are equal, the first value is returned.		3	0	0	0		
WMLSLib-con-12	When the Lang.max function is called with any non-value argument(s), it returns 'invalid'.		3	0	0	0		

WMLSLib-con-13	The value and type returned by the Lang.max function must be the value of the largest value and of type 'integer' when that value is an integer.		3	0	0	0		
WMLSLib-con-14	The value and type returned by the Lang.max function must be the value of the largest value and of type 'floating point' when that value is a floating point.		3	0	0	0		
WMLSLib-con-15	When the Lang.parseInt function is called with a string as an argument, it returns an integer reflecting the value of the contents of the string from its beginning up to the first character that is not '+', '-', or a decimal digit.		3	0	0	0		
WMLSLib-con-16	When the Lang.parseInt function is called with a string as an argument, and that string does not contain the representation of an integer at its beginning, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-17	When the Lang.parseInt function detects an error when parsing its input argument, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-18	When the Lang.parseFloat function is called with a string as an argument, it returns a floating-point value reflecting the value of the contents of the string from its beginning up to the first character that is not a legal part of a floating-point representation.		3	0	0	0		
WMLSLib-con-19	When the Lang.parseFloat function is called with a string as an argument, and that string contains no floating-point value representation, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-20	When the Lang.parseFloat function detects an error when parsing its input argument, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-21	When the Lang.isInt function is called with an object, it returns a boolean indicating whether the object can be converted to type integer.		3	0	0	0		
WMLSLib-con-22	When the Lang.isInt function is called with an object of type invalid, it returns 'invalid'.		3	0	0	0		

WMLSLib-con-23	When the Lang.isFloat function is called with an object, it returns a boolean indicating whether the object can be converted to type floating-point.		3	0	0	0		
WMLSLib-con-24	When the Lang.isFloat function is called with an object of type invalid, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-25	When the Lang.maxInt function is called, it returns the largest integer value that can be represented (2147483647).		2	1	0	0		
WMLSLib-con-26	When the Lang.minInt function is called, it returns the smallest integer value that can be represented (-2147483648).		3	0	0	0		
WMLSLib-con-27	When the Lang.float function is called, it returns true if floating point operations are supported.		3	0	0	0		
WMLSLib-con-28	When the Lang.random function is called with a value as the parameter, it returns an integer value that is greater than or equal to 0 but less than or equal to the value of the parameter.		3	0	0	0		
WMLSLib-con-29	When the Lang.random function is called with a non-value as the parameter, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-30	When the Lang.random function is called with a value of zero, it returns zero.		3	0	0	0		
WMLSLib-con-31	When the Lang.random function is called with a negative value, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-32	When the Lang.random function is called with a floating point value as the parameter, it returns an integer value that is greater than or equal to 0 but less than or equal to the value of the parameter.		3	0	0	0		
WMLSLib-con-33	When the Lang.seed function is called with an integer as the parameter, it initializes the pseudo-random sequence using that integer and returns the empty string.		3	0	0	0		
WMLSLib-con-34	When the Lang.seed function is called with a floating-point value as the parameter, it first converts the value to an integer and then initializes the pseudo-random sequence using that integer and returns the empty string.		3	0	0	0		

WMLSLib-con-35	When the Lang.seed function is called with a non-numeric value as the parameter, invalid must be returned.		3	0	0	0		
WMLSLib-con-36	When a seed value of greater than or equal to zero is used, a repeatable sequence of pseudo-random numbers must be produced.		3	0	0	0		
WMLSLib-con-37	When the Lang.characterSet function is called, it returns an IANA-assigned integer representing the character set supported by the WMLScript Interpreter.		3	0	0	0		
WMLSLib-con-38	When the Float.int function is called with a number, it returns the integer portion of that number.		3	0	0	0		
WMLSLib-con-39	When the Float.int function is called with a non-number, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-40	When the Float.int function is called with an integer, the result is that integer.		3	0	0	0		
WMLSLib-con-41	When the Float.floor function is called with a number, it returns the greatest integer value that is not greater than the given number.		3	0	0	0		
WMLSLib-con-42	When the Float.floor function is called with a non-number, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-43	When the Float.floor function is called with an integer, it returns that integer.		3	0	0	0		
WMLSLib-con-44	When the Float.ceil function is called with a number, it returns the smallest integer value that is not less than the given number.		3	0	0	0		
WMLSLib-con-45	When the Float.ceil function is called with a non-number, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-46	When the Float.ceil function is called with an integer, it returns that integer.		3	0	0	0		
WMLSLib-con-47	When the Float.pow function is called with two numbers val1 and val2, it returns the floating point result of raising val1 to the val2 power.		3	0	0	0		

WMLSLib-con-48	When the Float.pow function is called with two numbers val1 and val2, and val1 = 0 and val2 is less than 0, then the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-49	When the Float.pow function is called with two numbers val1 and val2, and val1 is less than 0 and val2 is not an integer, then the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-50	When the Float.pow function is called with either parameter not a number, then the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-51	When the Float.round function is called with a number val1, it returns the number value that is closest to val1 and is equal to an integer.		3	0	0	0		
WMLSLib-con-52	When the Float.round function is called with a number val1, and that number is equally close to two integer values, the function returns the larger number value.		3	0	0	0		
WMLSLib-con-53	When the Float.round function is called with a number val1, and that number is already an integer, the function returns that number.		3	0	0	0		
WMLSLib-con-54	When the Float.round function is called with a non-number val1, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-55	When the Float.sqrt function is called with a number val1, the function returns the square-root of val1.		3	0	0	0		
WMLSLib-con-56	When the Float.sqrt function is called with a number val1, and that number is less than zero, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-57	When the Float.maxFloat function is called, it returns the maximum floating point value supported (3.40282347E+38).		2	1	0	0		
WMLSLib-con-58	When the Float.minFloat function is called, it returns the smallest non-zero floating point value supported (1.17549435E-38 or smaller).		2	1	0	0		
WMLSLib-con-59	When the String.length function is called with a string parameter String, it returns the length of that String in characters.		3	0	0	0		

WMLSLib-con-60	When the String.length function is called with a non-string parameter String, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-61	When the String.isEmpty function is called with a string parameter String, it returns true if the string is of zero length, and false otherwise.		3	0	0	0		
WMLSLib-con-62	When the String.isEmpty function is called with a non-string parameter, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-63	When the String.charAt function is called with a string parameter String and a number parameter Index, it returns a string of length one reflecting the character at position Index within string String.		3	0	0	0		
WMLSLib-con-64	When the String.charAt function is called with a string parameter String and a number parameter Index, and the Index is a floating-point value, it is converted to an integer. Then the function returns a string of length one reflecting the character at position Index within string String.		3	0	0	0		
WMLSLib-con-65	When the String.charAt function is called with a string parameter String and a number parameter Index, and the Index value is out of range, the function returns an empty string.		3	0	0	0		
WMLSLib-con-66	When the String.charAt function is called with a non-string parameter String or a non-number parameter Index, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-67	When the String.subString function is called with parameters string String, number StartIndex, and number Length, the function returns a string reflecting the portion of String beginning at StartIndex and continuing for Length characters.		3	0	0	0		
WMLSLib-con-68	When the String.subString function is called with parameters string String, number StartIndex, and number Length, and StartIndex is less than zero, the function returns a string reflecting the portion of String beginning at position zero and continuing for Length characters.		3	0	0	0		
WMLSLib-con-69	When the String.subString function is called with parameters string String, number StartIndex, and number Length, and StartIndex is larger than the last index of String, the function returns the empty string.		3	0	0	0		

WMLSLib-con-70	When the String.subString function is called with parameters string String, number StartIndex, and number Length, and Length is larger than the remaining number of characters in String, Length is replaced with the number of remaining characters.		3	0	0	0		
WMLSLib-con-71	When the String.subString function is called with parameters string String, number StartIndex, and number Length, and Length less than or equal to zero, the function returns an empty string.		3	0	0	0		
WMLSLib-con-72	When the String.subString function is called with parameters string String, number StartIndex, and number Length, and either StartIndex or Length or of type floating-point, they are converted to integers. Then the function returns a string starting at the StartIndex-th position of String and continuing for Length characters.		3	0	0	0		
WMLSLib-con-73	When the String.subString function is called with parameters non-string String, non-number StartIndex, or non-number Length, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-74	When the String.find function is called with parameters string String and string SubString, it returns an integer reflecting the index of the first character in String that identically matches the case and encoding of the requested SubString.		3	0	0	0		
WMLSLib-con-75	When the String.find function is called with parameters string String and string SubString, and the SubString does not occur within String, the function returns -1.		3	0	0	0		
WMLSLib-con-76	When the String.find function is called with parameters string String and string SubString, and the SubString is an empty string, the function returns 'invalid'.		2	1	0	0		
WMLSLib-con-77	When the String.replace function is called with parameters string String, string OldSubString, and string NewSubString, it returns an updated version of String in which all substrings that identically match the case and encoding of OldSubString are replaced with NewSubString.		3	0	0	0		

WMLSLib-con-78	When the String.replace function is called with parameters string String, string OldSubString, and string NewSubString, and OldSubString is an empty string. The String.replace function returns invalid.		2	1	0	0		
WMLSLib-con-79	When the String.elements function is called with parameters string String and string Separator, it returns an Integer reflecting the number of elements in String that are separated by the first character of the string Separator.		3	0	0	0		
WMLSLib-con-80	When the String.elements function is called with parameters string String, and string Separator, and Separator is an empty string, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-81	When the String.elements function is called with parameters string String and string Separator, where String is an empty string, it returns 1 (because an empty string is still a valid element).		3	0	0	0		
WMLSLib-con-82	When the String.elementAt function is called with parameters string String, number Index, and string Separator, it returns a string reflecting Index-th element of String, where elements are separated by the first character of Separator.		3	0	0	0		
WMLSLib-con-83	When the String.elementAt function is called with parameters string String, number Index, and string Separator, and Index is less than zero, it returns a string reflecting the first element of String, where the elements is separated by the first character of Separator.		3	0	0	0		
WMLSLib-con-84	When the String.elementAt function is called with parameters string String, number Index, and string Separator, and Index is larger than the number of elements in String, it returns a string reflecting the last element of String, where element is separated by the first character of Separator.		3	0	0	0		
WMLSLib-con-85	When the String.elementAt function is called with parameters string String, number Index, and string Separator, and String is an empty string, then an empty string is returned.		3	0	0	0		

WMLSLib-con-86	When the String.elementAt function is called with parameters string String, number Index, and string Separator, and Separator is an empty string, then the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-87	When the String.elementAt function is called with parameters string String, number Index, and string Separator, and Index is of type floating-point, then it is first converted to a type integer. Then the function returns a string reflecting the Index-th element of String, where elements are separated by the first character of Separator.		3	0	0	0		
WMLSLib-con-88	When the String.removeAt function is called with parameters string String, number Index, and string Separator, it returns a string where the Index-th element and separator have been removed, where elements are separated by the first character of Separator.		3	0	0	0		
WMLSLib-con-89	When the String.removeAt function is called with parameters string String, number Index, and string Separator, and Index is less than 0, it returns a string where the first element and separator have been removed, where elements are separated by the first character of Separator.		3	0	0	0		
WMLSLib-con-90	When the String.removeAt function is called with parameters string String, number Index, and string Separator, and Index is larger than the number of elements, it returns a string where the last element and separator have been removed, where elements are separated by the first character of Separator.		3	0	0	0		
WMLSLib-con-91	When the String.removeAt function is called with parameters string String, number Index, and string Separator, and String is empty, the function returns an empty string.		3	0	0	0		
WMLSLib-con-92	When the String.removeAt function is called with parameters string String, number Index, and string Separator, and Separator is an empty string, then the function returns 'invalid'.		3	0	0	0		

WMLSLib-con-93	When the String.removeAt function is called with parameters string String, number Index, and string Separator, and Index is of type floating-point, then it is first converted to a type integer. Then the function returns a string reflecting the removal of the Index-th element of String, where elements are separated by the first character of Separator.		3	0	0	0		
WMLSLib-con-94	When the String.replaceAt function is called with parameters string String, string Element, number Index, and string Separator, it returns a string where the Index-th element is replaced with the contents of Element (where elements are separated by the first character of Separator).		3	0	0	0		
WMLSLib-con-95	When the String.replaceAt function is called with parameters string String, string Element, number Index, and string Separator, and Index is less than zero, it returns a string where the first element is replaced with the contents of Element (where elements are separated by the first character of Separator).		3	0	0	0		
WMLSLib-con-96	When the String.replaceAt function is called with parameters string String, string Element, number Index, and string Separator, and Index is larger than the number of elements, it returns a string where the last element is replaced with the contents of Element (where elements are separated by the first character of Separator).		3	0	0	0		
WMLSLib-con-97	When the String.replaceAt function is called with parameters string String, string Element, number Index, and string Separator, and String is empty the function returns a new string with the given Element.		3	0	0	0		
WMLSLib-con-98	When the String.replaceAt function is called with parameters string String, string Element, number Index, and string Separator, and Index is of type floating-point, Index is first converted to an integer. Then the function returns a string in which the Index-th element is replaced with the contents of Element.		3	0	0	0		

WMLSLib-con-99	When the String.replaceAt function is called with parameters string String, string Element, number Index, and string Separator, and Separator is an empty string, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-100	When the String.insertAt function is called with parameters string String, string Element, number Index, and string Separator, it returns a string where the string Element and a Separator are inserted at the Index-th element (where elements are separated by the first character of Separator).		3	0	0	0		
WMLSLib-con-101	When the String.insertAt function is called with parameters string String, string Element, number Index, and string Separator, and Index is less than zero, it returns a string where the Element and a Separator are inserted prior to the first element (where elements are separated by the first character of Separator).		3	0	0	0		
WMLSLib-con-102	When the String.insertAt function is called with parameters string String, string Element, number Index, and string Separator, and Index is larger than the number of elements, it returns a string where Element and a Separator are appended at the end of the String (where elements are separated by the first character of Separator).		3	0	0	0		
WMLSLib-con-103	When the String.insertAt function is called with parameters string String, string Element, number Index, and string Separator, and String is empty the function returns a string containing only Element.		3	0	0	0		
WMLSLib-con-104	When the String.insertAt function is called with parameters string String, string Element, number Index, and string Separator, and Index is of type floating-point, Index is first converted to an integer. Then the function returns a string in which Element and a Separator are inserted at the Index-th element of String.		3	0	0	0		
WMLSLib-con-105	When the String.insertAt function is called with parameters string String, string Element, number Index, and string Separator, and Separator is an empty string, the function returns 'invalid'.		3	0	0	0		

WMLSLib-con-106	When the String.squeeze function is called with parameter string String, it returns a string where all consecutive series of white space characters are reduced to a single white space character.		3	0	0	0		
WMLSLib-con-107	When the String.squeeze function is called with a parameter 'invalid', it returns 'invalid'.		3	0	0	0		
WMLSLib-con-108	When the String.trim function is called with parameter string String, it returns a string where all trailing and leading white space characters have been removed.		3	0	0	0		
WMLSLib-con-109	When the String.trim function is called with a parameter 'invalid', it returns 'invalid'.		3	0	0	0		
WMLSLib-con-110	When the String.compare function is called with parameters string String1 and string String2, and String1 is lexicographically equal to String2, the function returns 0.		3	0	0	0		
WMLSLib-con-111	When the String.compare function is called with parameters string String1 and string String2, and String1 is lexicographically less than String2, the function returns -1.		3	0	0	0		
WMLSLib-con-112	When the String.compare function is called with parameters string String1 and string String2, and String1 is lexicographically greater than String2, the function returns 1.		3	0	0	0		
WMLSLib-con-113	When the String.compare function is called with an invalid parameter, it returns 'invalid'.		3	0	0	0		
WMLSLib-con-114	When the String.toString function is called with a parameter Value, the function returns a string representing Value as defined in the WMLScript specification for type conversion.		3	0	0	0		
WMLSLib-con-115	When the String.toString function is called with an 'invalid' Value, the function returns the string "invalid".		3	0	0	0		
WMLSLib-con-116	When the String.format function is called with parameters string Format and Any Value, and Format contains more than one format specifier, subsequent specifiers are replaced with the empty string in the string the function returns.		3	0	0	0		

WMLSLib-con-117	When the String.format function is called with parameters string Format and Any Value, and Format contains two percent signs (%) in sequence, the function returns a string in which those signs are converted into a single percent sign.		3	0	0	0		
WMLSLib-con-118	When the String.format function is called with parameters string Format and Any Value, and Format contains an invalid format specifier, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-119	When the String.format function is called with parameters string "Format", a decimal integer "Value" and the "Format" contains a format specifier of the form "%d", the function replaces the "%d" with a representation of the value of the integer using the form [-]d (where d represents 1 or more decimal digits) in the string it returns.		3	0	0	0		
WMLSLib-con-120	When the String.format function is called with parameters string "Format", a decimal integer "Value" and the "Format" contains a format specifier of the form "%wd" and the number of digits in "Value" is less than the specified width "w", the function replaces the "%wd" with a representation of the value of the integer using the form [-]d (where d represents 1 or more decimal digits) with blanks added to the left until the minimum width is reached in the string it returns.		3	0	0	0		
WMLSLib-con-121	When the String.format function is called with parameters string "Format", a decimal integer "Value" and the "Format" contains a format specifier of the form "%wd" and the number of digits in "Value" is greater than or equal to the specified width "w", the function replaces the "%wd" with a representation of the value of the integer using the form [-]d (where d represents 1 or more decimal digits) in the string it returns.		3	0	0	0		
WMLSLib-con-122	When the String.format function is called with parameters string "Format", a decimal integer "Value" and the "Format" contains a format specifier of the form "%wd" and "w" is a negative decimal integer, the function returns 'invalid'.		3	0	0	0		

WMLSLib-con-123	When the String.format function is called with parameters string "Format", a decimal integer "Value" and the "Format" contains a format specifier of the form "%w.pd" and the number of digits in "Value" is less than the specified precision "p", the function replaces the "%w.pd" with a representation of the value of the integer using the form [-]d (where d represents 1 or more decimal digits) with zeros added to the left until the minimum precision is reached in the string it returns.		3	0	0	0		
WMLSLib-con-124	When the String.format function is called with parameters string "Format", a decimal integer "Value" and the "Format" contains a format specifier of the form "%w.pd" and the number of digits in "Value" is greater than or equal to the specified precision "p", the function replaces the "%w.pd" with a representation of the value of the integer using the form [-]d (where d represents 1 or more decimal digits) in the string it returns.		3	0	0	0		
WMLSLib-con-125	When the String.format function is called with parameters string "Format", a decimal integer "Value" and the "Format" contains a format specifier of the form "%w.pd" and "p" is a negative decimal integer, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-126	When the String.format function is called with parameters string Format and Any Value, and Format contains a format specification of the pattern "%.0d" and the value of Value is zero, the function replaces the "%.0d" with an empty string in the string it returns.		3	0	0	0		
WMLSLib-con-127	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%d", the function replaces the "%d" with a representation of the value of the string converted to an integer using the form [-]d (where d represents 1 or more decimal digits) in the string it returns.		3	0	0	0		

WMLSLib-con-128	When the String.format function is called with parameters string "Format", a floating point "Value" and the "Format" contains a format specifier of the form "%d", the function replaces the "%d" with a representation of the value of the floating point converted to an integer using the form [-]d (where d represents 1 or more decimal digits) in the string it returns.		2	1	0	0		
WMLSLib-con-129	When the String.format function is called with parameters string "Format", a floating point "Value" and the "Format" contains a format specifier of the form "%f", the function replaces the "%f" with a representation of the value of the floating point using the form [-]d.nnnnnn (where nnnnnn represents the default of six digits after the decimal point in the string it returns).		3	0	0	0		
WMLSLib-con-130	When the String.format function is called with parameters string "Format", a floating point "Value" and the "Format" contains a format specifier of the form "%wf" and the number of digits in "Value" is less than the specified width "w", the function replaces the "%wf" with a representation of the value of the floating point using the form [-]d.d (where d represents 1 or more decimal digits) with blanks added to the left until the minimum width is reached in the string it returns.		3	0	0	0		
WMLSLib-con-131	When the String.format function is called with parameters string "Format", a floating point "Value" and the "Format" contains a format specifier of the form "%wf" and the number of digits in "Value" is greater than or equal to the specified width "w", the function replaces the "%wf" with a representation of the value of the floating point using the form [-]d.d (where d represents 1 or more decimal digits) in the string it returns.		3	0	0	0		
WMLSLib-con-132	When the String.format function is called with parameters string "Format", a floating point "Value" and the "Format" contains a format specifier of the form "%wf" and "w" is a negative decimal integer, the function return 'invalid'.		3	0	0	0		

WMLSLib-con-133	When the String.format function is called with parameters string "Format", a floating point "Value" and the "Format" contains a format specifier of the form "%w.pf" and the number of digits after the decimal point in "Value" is less than the specified precision "p", the function replaces the "%w.pf" with a representation of the value of the floating point using the form [-]d.d (where d represents 1 or more decimal digits) with zeros padding until the minimum precision is reached in the string it returns.		3	0	0	0		
WMLSLib-con-134	When the String.format function is called with parameters string "Format", a floating point "Value" and the "Format" contains a format specifier of the form "%w.pf" and the number of digits after the decimal point in "Value" is greater than or equal to the specified precision "p", the function replaces the "%w.pf" with a representation of the value of the floating point using the form [-]d.d (where d represents 1 or more decimal digits) where the digits following the decimal point are rounded to the number of digits specified by precison "p" in the string it returns.		3	0	0	0		
WMLSLib-con-135	When the String.format function is called with parameters string "Format", a floating point "Value" which has no digits preceeding the decimal point and the "Format" contains a format specifier of the form "%f", the function replaces the "%f" with a representation of the value of the floating point using the form [-]d.d (where d represents 1 or more decimal digits) where one digit appears before the decimal point in the string it returns.		3	0	0	0		
WMLSLib-con-136	When the String.format function is called with parameters string Format, a floating point Value, and Format contains a format specification of the pattern "%.0f", the function replaces the "%.0f" with a representation of Value rounded to the nearest integer with no trailing decimal point in the string it returns.		3	0	0	0		

WMLSLib-con-137	When the String.format function is called with parameters string Format, a floating point Value, and Format contains a format specification of the pattern "%.f", the function replaces the "%.f" with a representation of Value rounded to the nearest integer with no trailing decimal point in the string it returns.		3	0	0	0		
WMLSLib-con-138	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%f", the function replaces the "%f" with a representation of the value of the string converted to a floating point using the form [-]d.d (where d represents 1 or more decimal digits) in the string it returns.		3	0	0	0		
WMLSLib-con-139	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%s", the function replaces the "%s" with a representation of the string "Value" in the string it returns.		3	0	0	0		
WMLSLib-con-140	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%ws" and the number of characters in the string "Value" is less than the specified width "w", the function replaces the "%ws" with a representation of the string "Value" with blanks added to the left until the minimum width is reached in the string it returns.		3	0	0	0		
WMLSLib-con-141	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%ws" and the number of characters in "Value" is greater than or equal to the specified width "w", the function replaces the "%ws" with a representation of the string "Value" in the string it returns.		3	0	0	0		
WMLSLib-con-142	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%ws" and "w" is a negative decimal integer, the function returns 'invalid'.		3	0	0	0		

WMLSLib-con-143	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%w.ps", the function replaces the "%w.ps" with a representation of the string "Value" using the characters up to the end of the string "Value" or until the precision value is reached in the string it returns.		3	0	0	0		
WMLSLib-con-144	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%w.ps" and w is larger than p, the function replaces the "%w.ps" with a representation of the string "Value" ignoring the value of w in the string it returns.		3	0	0	0		
WMLSLib-con-145	When the String.format function is called with parameters string "Format", a string "Value" and the "Format" contains a format specifier of the form "%w.ps" and "p" is a negative decimal integer, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-146	When the String.format function is called with parameters string Format and Any Value, and Format contains a percent sign (%) followed by a period (.) and a non-negative decimal precision p followed by an "s", the function interprets Value as a string value and replaces the "%.ps" with up to p characters from Value in the string it returns.		3	0	0	0		
WMLSLib-con-147	When the String.format function is called with parameters string "Format", an integer "Value" and the "Format" contains a format specifier of the form "%s", the function replaces the "%s" with a representation of the integer converted to a string "Value" in the string it returns.		3	0	0	0		
WMLSLib-con-148	When the String.format function is called with parameters string "Format", a floating point "Value" and the "Format" contains a format specifier of the form "%s", the function replaces the "%s" with a representation of the floating point converted to a string "Value" in the string it returns.		3	0	0	0		

WMLSLib-con-149	When the URL.isValid function is called with a parameter string URL that contains an absolute URL, it returns 'true' if the URL is well formed.		3	0	0	0		
WMLSLib-con-150	When the URL.isValid function is called with a parameter string URL that contains a relative URL, it returns 'true' if the URL is well formed.		3	0	0	0		
WMLSLib-con-151	When the URL.isValid function is called with a parameter string URL that does not contain a well formed absolute or relative URL, it returns 'false'.		3	0	0	0		
WMLSLib-con-152	When the URL.isValid function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-153	When the URL.getScheme function is called with a parameter string URL that contains an absolute URL, it returns a string containing the scheme portion of the URL.		3	0	0	0		
WMLSLib-con-154	When the URL.getScheme function is called with a parameter string URL that contains a relative URL, it returns a string containing the scheme portion of the relative URL.		3	0	0	0		
WMLSLib-con-155	When the URL.getScheme function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-156	When the URL.getScheme function is called with a parameter string URL that contains invalid URL syntax, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-157	When the URL.getHost function is called with a parameter string URL that contains an absolute URL, it returns a string containing the hostname portion of the URL.		3	0	0	0		
WMLSLib-con-158	When the URL.getHost function is called with a parameter string URL that contains a relative URL, it returns a string containing the hostname portion of the relative URL.		3	0	0	0		
WMLSLib-con-159	When the URL.getHost function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		

WMLSLib-con-160	When the URL.getHost function is called with a parameter string URL that contains invalid URL syntax, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-161	When the URL.getPort function is called with a parameter string URL that contains an absolute URL, it returns a string containing the port-number portion of the URL.		3	0	0	0		
WMLSLib-con-162	When the URL.getPort function is called with a parameter string URL that contains a relative URL, it returns a string containing the port-number portion of the relative URL.		3	0	0	0		
WMLSLib-con-163	When the URL.getPort function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-164	When the URL.getPort function is called with a parameter string URL, and that string does not contain a port number, the function returns the empty string.		3	0	0	0		
WMLSLib-con-165	When the URL.getPort function is called with a parameter string URL that contains invalid URL syntax, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-166	When the URL.getPath function is called with a parameter string URL that contains an absolute URL, it returns a string containing the pathname portion of the URL.		3	0	0	0		
WMLSLib-con-167	When the URL.getPath function is called with a parameter string URL that contains a relative URL, it returns a string containing the pathname portion of the relative URL.		3	0	0	0		
WMLSLib-con-168	When the URL.getPath function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-169	When the URL.getPath function is called with a parameter string URL that contains invalid URL syntax, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-170	When the URL.getParameters function is called with a parameter string URL that contains an absolute URL, it returns a string containing the parameter portion of the URL.		3	0	0	0		

WMLSLib-con-171	When the URL.getParameters function is called with a parameter string URL that contains a relative URL, it returns a string containing the parameter portion of the relative URL.		3	0	0	0		
WMLSLib-con-172	When the URL.getParameters function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-173	When the URL.getParameters function is called with a parameter string URL that contains invalid URL syntax, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-174	When the URL.getParameters function is called with a parameter string URL that contains an absolute URL that has no parameter portion, the function returns an empty string.		3	0	0	0		
WMLSLib-con-175	When the URL.getParameters function is called with a parameter string URL that contains a relative URL that contains no parameter portion, the function returns an empty string.		3	0	0	0		
WMLSLib-con-176	When the URL.getQuery function is called with a parameter string URL that contains an absolute URL, it returns a string containing the query portion of the URL.		3	0	0	0		
WMLSLib-con-177	When the URL.getQuery function is called with a parameter string URL that contains a relative URL, it returns a string containing the query portion of the relative URL.		3	0	0	0		
WMLSLib-con-178	When the URL.getQuery function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-179	When the URL.getQuery function is called with a parameter string URL that contains invalid URL syntax, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-180	When the URL.getQuery function is called with a parameter string URL that contains an absolute URL that has no query portion, the function returns an empty string.		3	0	0	0		
WMLSLib-con-181	When the URL.getQuery function is called with a parameter string URL that contains a relative URL that does not contain a query portion, the function returns the enpty string.		3	0	0	0		

WMLSLib-con-182	When the URL.getFragment function is called with a parameter string URL that contains an absolute URL, it returns a string containing the fragment portion of the URL.		3	0	0	0		
WMLSLib-con-183	When the URL.getFragment function is called with a parameter string URL that contains a relative URL, it returns a string containing the fragment portion of the relative URL.		3	0	0	0		
WMLSLib-con-184	When the URL.getFragment function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-185	When the URL.getFragment function is called with a parameter string URL that contains invalid URL syntax, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-186	When the URL.getFragment function is called with a parameter string URL that contains an absolute URL that does not have a fragment portion, the function returns an empty string.		3	0	0	0		
WMLSLib-con-187	When the URL.getFragment function is called with a parameter string URL that contains a relative URL that does not have a fragment portion, the function returns an empty string.		3	0	0	0		
WMLSLib-con-188	When the URL.getBase function is called, it returns a string reflecting the absolute URL (without the fragment) of the current Script file.		3	0	0	0		
WMLSLib-con-189	When the URL.getReferer function is called, it returns a string reflecting the smallest relative URL (relative to the base URL of the current compilation unit) to the resource that called the current compilation unit.		1	2	0	0		
WMLSLib-con-190	When a compilation unit calls another local function, and that function calls the URL.getReferer function, the result of url.getReferer will be same as that of calling it from the parent function.		1	2	0	0		
WMLSLib-con-191	When the URL.resolve function is called with parameters string BaseURL and string EmbeddedURL, where BaseURL is an absolute URL, the function returns an absolute URL derived from the combination of BaseURL and EmbeddedURL.		3	0	0	0		

WMLSLib-con-192	When the URL.resolve function is called with parameters string BaseURL and string EmbeddedURL, where EmbeddedURL is an absolute URL, the function returns EmbeddedURL.		3	0	0	0		
WMLSLib-con-193	When the URL.resolve function is called with parameters string BaseURL and string EmbeddedURL, where either parameter's values contain invalid URL syntax, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-194	When the URL.resolve function is called with parameters string BaseURL and string EmbeddedURL, and the BaseURL's path component is an empty string. The function returns an absolute URL derived from the combination of BaseURL and EmbeddedURL with a single slash character ("/") assumed as the path.		2	1	0	0		
WMLSLib-con-195	When the URL.escapeString function is called with a parameter string String, it returns a string in which the special characters in String are replaced with their hexadecimal escape sequences (e.g. %ff).		3	0	0	0		
WMLSLib-con-196	When the URL.escapeString function is called with a parameter string String that contains characters not from the US-ASCII character set, the function returns 'invalid'.		2	1	0	0		
WMLSLib-con-197	When the URL.unescapeString function is called with a parameter string String, it returns a string in which the hexadecimal escape sequences in String are replaced with the characters they represent.		3	0	0	0		
WMLSLib-con-198	When the URL.unescapeString function is called with a parameter string String that contains characters above hexidecimal FF, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-199	When the URL.loadString function is called with the parameters string URL and string ContentType, where URL is an absolute URL and ContentType is a single content type specifier beginning with "text/", the function returns a string that contains the resource.		1	2	0	0		

WMLSLib-con-200	When the URL.loadString function is called with the parameters string URL and string ContentType, where URL is an absolute URL that cannot be loaded, the function returns an integer error code.		2	1	0	0		
WMLSLib-con-201	When the URL.loadString function is called with the parameters string URL and string ContentType, where URL is an absolute URL for which the content type does not match ContentType, the function returns an integer error code.		2	1	0	0		
WMLSLib-con-202	When the URL.loadString function is called with the parameters string URL and string ContentType, where the ContentType is not well formed, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-203	When the URL.loadString function is called with the parameters string URL and string ContentType, where the ContentType contains leading spaces, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-204	When the URL.loadString function is called with the parameters string URL and string ContentType, where the ContentType contains trailing spaces, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-205	When the URL.loadString function is called with the parameters string URL and string ContentType, where more than one ContentType has been specified the function returns invalid		3	0	0	0		
WMLSLib-con-206	When the WMLBrowser.getVar function is called with a parameter string Name, it returns the value of the browser context variable identified by Name.		3	0	0	0		
WMLSLib-con-207	When the WMLBrowser.getVar function is called with a parameter string Name, and a variable Name does not exist in the current browser context, the function returns the empty string.		3	0	0	0		
WMLSLib-con-208	When the WMLBrowser.getVar function is called with non-string parameter, the function returns 'invalid'.		3	0	0	0		

WMLSLib-con-209	When the WMLBrowser.getVar function is called with a parameter string Name, and a variable Name is not formatted as defined in the WML 1.1 specification, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-210	When the WMLBrowser.setVar function is called with the parameters string Name and string Value, the browser context variable Name is set to have the value Value and the function returns true.		3	0	0	0		
WMLSLib-con-211	When the WMLBrowser.setVar function is called with either parameter a non-string, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-212	When the WMLBrowser.setVar function is called with the parameters string Name and string Value, and the value of Name does not conform to the definition of variable names in the WML 1.1 specification, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-213	When the WMLBrowser.setVar function is called with the parameters string Name and string Value, and the value of Value is not legal CDATA as defined in the XML specification, the function returns 'invalid'.		2	1	0	0		
WMLSLib-con-214	When the WMLBrowser.setVar function is called with the parameters string Name and string Value. The function returns false if the browser context variable Name cannot be successfully set.		3	0	0	0		
WMLSLib-con-215	When the WMLBrowser.go function is called with a parameter string URL, the WML browser queues a "GO" task to the URL and the function returns the empty string. When control returns to the WML Browser, the GO task is executed.		3	0	0	0		
WMLSLib-con-216	When the WMLBrowser.go function is called with a parameter string URL, where URL is the empty string, the WML browser cancels any previously queued GO or PREV task in the WML browser and the function returns the empty string.		2	1	0	0		
WMLSLib-con-217	When the WMLBrowser.go function is called with a non-string parameter, the function returns 'invalid'.		3	0	0	0		

WMLSLib-con-218	When the WMLBrowser.go function is called with a parameter string URL, and a fatal error occurs, the WML browser cancels any pending go() requests.		3	0	0	0		
WMLSLib-con-219	When the WMLBrowser.prev function is called, the WML browser queues a "PREV" task and the function returns the empty string. When control returns to the WML Browser, the PREV task is executed.		3	0	0	0		
WMLSLib-con-220	When the WMLBrowser.go function is called with a URL of the empty string, it cancels any previous call to WMLBrowser.prev.		2	1	0	0		
WMLSLib-con-221	When the WMLBrowser.prev function is called and a fatal error occurs, the WML browser cancels any pending prev() requests.		3	0	0	0		
WMLSLib-con-222	When the WMLBrowser.newContext function is called, it clears the current WML Browser context and returns an empty string.		2	1	0	0		
WMLSLib-con-223	When the WMLBrowser.getCurrentCard function is called, and the current compilation unit has the same base URL as the calling WML deck, the function returns the smallest relative URL (to the base of the current compilation unit) specifying the card currently being processed by the WML Browser.		2	1	0	0		
WMLSLib-con-224	When the WMLBrowser.refresh function is called, the WML Browser updates its context and the function returns an empty string.		2	1	0	0		
WMLSLib-con-225	When the WMLBrowser.refresh function is called, the WML Browser updates its context and the function returns an empty string. The refresh must not restart any suspended timers.		2	1	0	0		
WMLSLib-con-226	When the Dialogs.prompt function is called with the parameters string Message and string DefaultInput, the system displays the message Message supplies the string DefaultInput as the default content. When the user has changed the content, the function returns that content as a string.		3	0	0	0		

WMLSLib-con-227	When the Dialogs.prompt function is called with non-string parameters, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-228	When the Dialogs.prompt function is called with the parameters string Message and string DefaultInput, the system displays the message Message supplies the string DefaultInput as the default content.		3	0	0	0		
WMLSLib-con-229	When the Dialogs.confirm function is called with the parameters string Message, string Ok, and string Cancel, the system displays the message Message and two reply alternatives from Ok and Cancel. If the user selects the Ok string, the function returns true.		3	0	0	0		
WMLSLib-con-230	When the Dialogs.confirm function is called with the parameters string Message, string Ok, and string Cancel, the system displays the message Message and two reply alternatives from Ok and Cancel. If the user selects the Cancel string, the function returns false.		3	0	0	0		
WMLSLib-con-231	When the Dialogs.confirm function is called with the parameters string Message, string Ok, and string Cancel, where Ok and Cancel are empty strings, the system displays the message Message and two implementation-defined reply alternatives. If the user selects the "positive" choice, the function returns true.		3	0	0	0		
WMLSLib-con-232	When the Dialogs.confirm function is called with the parameters string Message, string Ok, and string Cancel, where Ok and Cancel are empty strings, the system displays the message Message and two implementation-defined reply alternatives. If the user selects the "negative" choice, the function returns false.		3	0	0	0		
WMLSLib-con-233	When the Dialogs.confirm function is called with any non-string parameters, the function returns 'invalid'.		3	0	0	0		
WMLSLib-con-234	When the Dialogs.alert function is called with a parameter string Message, it displays the message to the user, waits for the user to confirm that they have seen the message, and returns the empty string.		3	0	0	0		

WMLSLib-con-235	When the Dialogs.alert function is called with a non-string parameter, it returns 'invalid'.		3	0	0	0		

Table 2. Test Case Counts

5.2.3 Problem Reports

During the activities for TestFest14, the following problem reports were generated relating to the test materials and test process: (please see http://www.opengroup.org:8000/OMA-PR/ for more detail).

PR Number	Affecting	Description	Test Case reference / Specification reference
0046		Different Reference. It is not in accord with Test Page's document and Test Guide's document.	
0047		contains an invalid assertion , Not supported attribute used.	
0048		Don't existing Test Page file.	
0049		Don’t existence policies for Interaction with XHTML MP pages and WML Pages	
0050		In WCSS-1.1-con-18, [45.21.01], [45.21.01] need more detail and accuracy assertion.	
0051		Duplicated Test Procedure ([45.23.11], [45.23.12], [45.23.13]) in WCSS- 1.1-con-27 and WCSS-1.1-con-28.	
0052		Assertion need to more detail and accurate description.	
0053		Assertion need to more detail and accurate description.	
0054		TC148.html test page not found.	
0055		Test page (t140.html) not found	
0056		Test page (t063.html) not found	
0057		Test Page (49.07.11)not found	
0058		Function error in Test Pages	
0059		do not exist next page	
0060		Test page 49.07.11 not found	
0061		do not exist next page	
0062		Script function error in Test Page	
0063		do not exist test page.	
0064		Script function error in Test Page	
0065		second parameter required.	
0066		Test Page 49.11.11 not found	
0067		script error	
0068		function not found in Test Page 49.13.26, 49.13.49	

0069		Rererence Error in 49.14.01, 49.14.02	
0070		Page not found, test page path not found	
0071		Page not found, test page path not found	
0072		no TCs available in ESMP-1.0-con-60	
0073		no TCs available in ESMP-1.0-con-57	
0074		no TCs available in ESMP-1.0-con-61	
0075		no TCs available in ESMP-1.0-con-56	
0076		no TCs available in ESMP-1.0-con-57,60,61 404 no such page in 49.37.02	
0077		no TCs available in ESMP-1.0-con-68	
0078		404 no such page in ESMP-1.0-con-64	
0079		Test XHTML 45.06.04 (XHTML_Content/xhtml_List/nested_ul.xhtml) in xHTML-1.1-con-4 uses invalid markup.	
0080		Test 45.08.22 (XHTML_Content/xhtml_tables/illegal_width_percent.xhtml) in xHTML-1.1-con-6 contradicts is ambiguous.	
0081		Test 45.11.06 (XHTML_Content/xhtml_base/rel_base.xhtml) in xHTML-1.1-con-11 is invalid, as it contradicts HTML 4.01 specification.	
0082		Test file t144.html for xHTML-1.1-con-21 does not exist.	
0083		Test file t145.html for xHTML-1.1-con-22 can not be located.	
0084		All utf-16 tests in the WCSS test suite use invalid utf-16 encoding for utf-16 content.	
0085		xHTML-1.1-con-14, XHTML test case 45.13.10 has wrong description in test document	
0086		xHTML-1.1-con-15 :(45.13.09) possible mistyping in test document	
0087		xHTML-1.1-con26 : (test file 45.08.26) test markup page has error in it.	
0088		CSS test suite expects wrong ending state for -wap-marquee tests.	
0089		Test 45.20.22 in WCSS-1.1-con-15 uses CSS stylesheet which contradicts the test assertion.	
0090		Many CSS tests utilize optional attribute value selectors to test other properties; if a device doesn't support attr value selectors, the test will be inconclusive.	

0091		Test 49.26.02 (esmp_Sec9/objNavigator/navigator_enum.xhtml) refers to non-existent script file	
0092		Test 49.28.02 (esmp_Sec9/objLocation/location_enum.xhtml) refers to non-existent script file	
0093		Test 49.01.02 in ESMP Test Suite uses invalid utf-16 test script.	
0094		Test 49.25.01 in ESMP Test Suite uses script that can't be located	
0095		Test 49.29.02 (esmp_Sec9/objDocument/doc_enum.xhtml) in ESMP Test Suite uses script that can't be located	
0096		Test 49.42.02 (esmp_Sec11/objNode/node_enum.xhtml) in ESMP Test Suite uses script that can't be located	
0097		Test 49.42.55 (esmp_Sec11/objNode/removeChild2.xhtml) can't be found	
0098		Test 49.44.02 (esmp_Sec11/objNodeList/nodeList_enum.xhtml) refers to a non-existent script file.	
0099		Test 49.45.02 (esmp_Sec11/objElement/element_enum.xhtml) refers to a non-existent script file.	
0100		Test 49.46.02 (esmp_Sec11/objText/text_enum.xhtml) refers to a non-existent script file.	

Full details of all Problem Reports can be found at: http://www.opengroup.org:8000/OMA-PR/

6. Confirmation

This signature states that the included information is true and valid.

[illegible]

____

OMA Trusted Zone

Appendix A. Change History (Informative)

Type of Change	Date	Section	Description