DR 14-0011 — SML: Attribute Value of “none”

Status: Closed; in COR4

Subject: SML: Attribute Value of “none”

Qualifier: Request for clarification

Submitter: Charlie Clark Organization: Ecma/Clark Consulting & Research

Contact Information: [charlie.clark@clark-consulting.eu](file:///E%3A%5CStd%5CSC%2034%5CWG4%5CDRs%5CLog%5C2014%5Ccharlie.clark%40clark-consulting.eu)

Submitter’s Defect Number: None

Supporting Document(s): None

Date Circulated by Secretariat: 2014-09-25

Deadline for Response from Editor: 2014-11-25

IS 29500 Reference(s): 29500:2016, Part 1, §18.8.32, “patternFill (Pattern)”, p. 1792

Related DR(s): None

Nature of the Defect:

I've come across some areas where some guidance would be welcome. Specifically, several attributes allow "none" as a possible value, but do not specify what this refers to, though I take it to mean the same as if the attribute is unset, i.e.,

<patternFill patternType="none">

 <fgColor rgb="FFFFFF00"/>

 <bgColor indexed="64"/>

</patternFill>

is the same as

<patternFill>

 <fgColor rgb="FFFFFF00"/>

 <bgColor indexed="64"/>

</patternFill>

Which according the description is the same as <patternFill /> with all other attributes and children ignored.

Is this true? I know this is an extreme example, but there are quite a few occurrences. I assume, for reasons of backwards compatibility, that it is not possible to remove "none" from the possible values but it would be good to have a note explaining how it should be interpreted and possibly even discouraging its use.

Solution Proposed by the Submitter:

None

Schema Change(s) Needed:

Yes

**Editor’s Response:**

**2014-09-25 Charlie Clark:**

If <element attrib="none><element> is the same as <element><element> as I contend the cases where "none" is not the default should be clearly identified, i.e., where <element /> actually equates to <element attrib="default\_value" />

**2015-02-09 Chris Rae:**

This DR discusses the attribute value of “none”, whether it is a worthwhile feature and whether behaviour is correctly specified in all cases. There is some cross-over with the other DRs which cover optional attributes with no behaviour described when they are missing.

There are 70 types in schema that declare an enumeration value of “none”. Those are then used by a large number of elements in Part 1. I’ve looked through all of these and made some changes in the attached document – a summary of those changes and my investigation is in the table shown here:

|  |  |  |
| --- | --- | --- |
| **attribute name** | **Of type** | **Chris notes** |
| vertical | ST\_VerticalAlignment | Microsoft implementer note exists; added default ('bottom') in schema. |
| flip | ST\_TileFlipMode | Added default 'none' in schema (from practical testing) |
| flip | ST\_TileFlipMode | Added default 'none' in schema (from practical testing) |
| type | ST\_LineEndType | Added default 'none' in schema (from practical testing) |
| wrap | ST\_TextWrappingType | Behaviour adequately described |
| u | ST\_TextUnderlineType | Behaviour adequately described |
| cap | ST\_TextCapsType | Microsoft implementer note exists; added default ('none') in schema. Also corrected poor wording in prose. |
| nextAc | ST\_TLNextActionType | Behaviour adequately described |
| prevAc | ST\_TLPreviousActionType | Behaviour adequately described |
| additive | ST\_TLBehaviorAdditiveType | Covered in DR 13-0014 |
| accumulate | ST\_TLBehaviorAccumulateType | Covered in DR 13-0014 |
| transition | ST\_TLAnimateEffectTransition | Checked imp notes |
| patternType | ST\_PatternType | Behaviour adequately described |
| val | ST\_Underline | Covered in DR 13-0013 |
| combineBrackets | ST\_CombineBrackets | Behaviour adequately described |
| wrap | ST\_Wrap | Behaviour adequately described |
| dropCap | ST\_DropCap | Behaviour adequately described |
| leader | ST\_TabTlc | Behaviour adequately described |
| val | ST\_Zoom | Behaviour adequately described |
| edit | ST\_DocProtect | Behaviour adequately described |
| numFmt | ST\_NumberFormat | Behaviour adequately described |
| clear | ST\_BrClear | Behaviour adequately described |
| edGrp | ST\_EdGrp | Covered in DR 13-0013 |
| themeColor | ST\_ThemeColor | Covered in DR 13-0013 |
| themeColor | ST\_ThemeColor | Covered in DR 13-0013 |
| themeColor | ST\_ThemeColor | Covered in DR 13-0013 |
| themeColor | ST\_ThemeColor | Covered in DR 13-0013 |
| themeFill | ST\_ThemeColor | Covered in DR 13-0013 |
| themeColor | ST\_ThemeColor | Covered in DR 13-0013 |

While one could argue that it’s not necessary to have a enumerated value on an attribute that causes the same behaviour as when the attribute is missing, I think the value of removing this doesn’t make it worthwhile losing compatibility with existing Office applications.

The specific example given by the submitter (@patternType on patternFill) is, I think, already covered adequately by the standard. The fill style of ‘none’ is defined in §18.18.55:

| none (None) | The fill style is none (no fill). When foreground and/or background colors are specified, a pattern of 'none' overrides and means the cell has no fill.[Example:cid:image001.png@01D04460.8A8D0030end example] |
| --- | --- |

… and the behaviour when the attribute is missing is described in §18.8.32 (patternFill):

| patternType (Pattern Type) | Specifies the fill pattern type (including solid and none) Default is none, when missing.The possible values for this attribute are defined by the ST\_PatternType simple type (§18.18.55). |
| --- | --- |

However, here is the proposal to address the general problem:

**Part 1: §20.1.10.64, “ST\_TextCapsType (Text Cap Types)”, enum value none, pp. 3063–3064**

…

|  |  |
| --- | --- |
| Enumeration Value | Description |
| none (Text Caps Enum ( None )) | ~~The reason we cannot implicitly have noCaps be the scenario where capitalization is not specified is because not being specified implies deriving from a particular style and the user might want to override that and make some text not have a capitalization scheme even though the style says otherwise.~~No capitalization effects should be applied during text display, overriding capitalization effects specified in styles. |

**Part 1: §A.2, “SpreadsheetML”, p. 3935, Lines 3,425–3,435**

 <xsd:complexType name="CT\_CellAlignment">

 …

 <xsd:attribute name="vertical" type="[ST\_VerticalAlignment](#XSD_S_sp_ST_VerticalAlignment)" default="bottom" use="optional"/>

 …

 </xsd:complexType>

**Part 1: §A.3, “PresentationML”, p. 3965, Lines 521–529**

 <xsd:complexType name="CT\_TLAnimateEffectBehavior">

 …

 <xsd:attribute name="transition" type="[ST\_TLAnimateEffectTransition](#XSD_S_ppt_ST_TLAnimateEffectTransition)" default="in" use="optional"/>

 …

 </xsd:complexType>

**Part 1: §A.4.1, “DrawingML – Main”, p. 4013, Lines 1443–1451**

 <xsd:complexType name="CT\_GradientFillProperties">

 …

 <xsd:attribute name="flip" type="[ST\_TileFlipMode](#XSD_S_a_ST_TileFlipMode)" use="optional" default="none"/>

 …

 </xsd:complexType>

**Part 1: §A.4.1, “DrawingML – Main”, p. 4013, Lines 1452–1459**

 <xsd:complexType name="CT\_TileInfoProperties">

 …

 <xsd:attribute name="flip" type="[ST\_TileFlipMode](#XSD_S_a_ST_TileFlipMode)" use="optional" default="none"/>

 …

 </xsd:complexType>

**Part 1: §A.4.1, “DrawingML – Main”, p. 4026, Lines 2135–2139**

 <xsd:complexType name="CT\_LineEndProperties">

 <xsd:attribute name="type" type="[ST\_LineEndType](#XSD_S_a_ST_LineEndType)" use="optional" default="none"/>

 …

 </xsd:complexType>

**Part 1: §A.4.1, “DrawingML – Main”, p. 4041, Lines 2904–2940**

 <xsd:complexType name="CT\_TextCharacterProperties">

 …

 <xsd:attribute name="cap" type="[ST\_TextCapsType](#XSD_S_a_ST_TextCapsType)" use="optional" default="none"/>

 …

 </xsd:complexType>

**Part 1: §B.2, “SpreadsheetML”, p. 4249, Lines 3756–3765**

sml\_CT\_CellAlignment =

…

 ## default value: bottom

 attribute vertical { sml\_ST\_VerticalAlignment }?,

…

 attribute readingOrder { xsd:unsignedInt }?

**Part 1: §B.3, “PresentationML”, p. 4279, Lines 304–309**

p\_CT\_TLAnimateEffectBehavior =

 ## default value: in

 attribute transition { p\_ST\_TLAnimateEffectTransition }?,

…

 element progress { p\_CT\_TLAnimVariant }?

**Part 1: §B.4.1, “DrawingML – Main”, p. 4321, Lines 1195–1200**

a\_CT\_GradientFillProperties =

 ## default value: none

 attribute flip { a\_ST\_TileFlipMode }?,

…

 element tileRect { a\_CT\_RelativeRect }?

**Part 1: §B.4.1, “DrawingML – Main”, p. 4321, Lines 1201–1207**

a\_CT\_TileInfoProperties =

…

 ## default value: none

 attribute flip { a\_ST\_TileFlipMode }?,

 attribute algn { a\_ST\_RectAlignment }?

**Part 1: §B.4.1, “DrawingML – Main”, p. 4331, Lines 1702–1705**

a\_CT\_LineEndProperties =

 ## default value: none

 attribute type { a\_ST\_LineEndType }?,

…

 attribute len { a\_ST\_LineEndLength }?

**Part 1: §B.4.1, “DrawingML – Main”, p. 4340, Lines 2206–2220**

a\_CT\_TextCharacterProperties =

…

 ## default value: none

 attribute cap { a\_ST\_TextCapsType }?,

…

 attribute noProof { xsd:boolean }?,

**Part 4: §A.2, “SpreadsheetML”, p. 999, Lines 3451–3461**

 <xsd:complexType name="CT\_CellAlignment">

 …

 <xsd:attribute name="vertical" type="[ST\_VerticalAlignment](#XSD_S_sp_ST_VerticalAlignment)" default="bottom" use="optional"/>

 …

 </xsd:complexType>

**Part 4: §A.3, “PresentationML”, p. 1029, Lines 523–531**

 <xsd:complexType name="CT\_TLAnimateEffectBehavior">

 …

 <xsd:attribute name="transition" type="[ST\_TLAnimateEffectTransition](#XSD_S_ppt_ST_TLAnimateEffectTransition)" default="in" use="optional"/>

 …

 </xsd:complexType>

**Part 4: §A.4.1, “DrawingML – Main”, p. 1079, Lines 1464–1472**

 <xsd:complexType name="CT\_GradientFillProperties">

 …

 <xsd:attribute name="flip" type="[ST\_TileFlipMode](#XSD_S_a_ST_TileFlipMode)" use="optional" default="none"/>

 …

 </xsd:complexType>

**Part 4: §A.4.1, “DrawingML – Main”, p. 1079, Lines 1473–1480**

 <xsd:complexType name="CT\_TileInfoProperties">

 …

 <xsd:attribute name="flip" type="[ST\_TileFlipMode](#XSD_S_a_ST_TileFlipMode)" use="optional" default="none"/>

 …

 </xsd:complexType>

**Part 4: §A.4.1, “DrawingML – Main”, p. 1092, Lines 2156–2160**

 <xsd:complexType name="CT\_LineEndProperties">

 <xsd:attribute name="type" type="[ST\_LineEndType](#XSD_S_a_ST_LineEndType)" use="optional" default="none"/>

 …

 </xsd:complexType>

**Part 4: §A.4.1, “DrawingML – Main”, pp. 1106–1107, Lines 2937–2973**

 <xsd:complexType name="CT\_TextCharacterProperties">

 …

 <xsd:attribute name="cap" type="[ST\_TextCapsType](#XSD_S_a_ST_TextCapsType)" use="optional" default="none"/>

 …

 </xsd:complexType>

**Part 4: §B.2, “SpreadsheetML”, p. 1346, Lines 3778–3787**

sml\_CT\_CellAlignment =

…

 ## default value: bottom

 attribute vertical { sml\_ST\_VerticalAlignment }?,

…

 attribute readingOrder { xsd:unsignedInt }?

**Part 4: §B.3, “PresentationML”, p. 1375, Lines 304–309**

p\_CT\_TLAnimateEffectBehavior =

 ## default value: in

 attribute transition { p\_ST\_TLAnimateEffectTransition }?,

…

 element progress { p\_CT\_TLAnimVariant }?

**Part 4: §B.4.1, “DrawingML – Main”, p. 1419, Lines 1204–1209**

a\_CT\_GradientFillProperties =

 ## default value: none

 attribute flip { a\_ST\_TileFlipMode }?,

…

 element tileRect { a\_CT\_RelativeRect }?

**Part 4: §B.4.1, “DrawingML – Main”, p. 1419, Lines 1210–1216**

a\_CT\_TileInfoProperties =

…

 ## default value: none

 attribute flip { a\_ST\_TileFlipMode }?,

 attribute algn { a\_ST\_RectAlignment }?

**Part 4: §B.4.1, “DrawingML – Main”, p. 1429, Lines 1711–1714**

a\_CT\_LineEndProperties =

 ## default value: none

 attribute type { a\_ST\_LineEndType }?,

…

 attribute len { a\_ST\_LineEndLength }?

**Part 4: §B.4.1, “DrawingML – Main”, p. 1438, Lines 2219–2233**

a\_CT\_TextCharacterProperties =

…

 ## default value: none

 attribute cap { a\_ST\_TextCapsType }?,

…

 attribute noProof { xsd:boolean }?,

**2015-02-24/26 Seattle F2F Meeting:**

Agreed with Chris’ proposal, as detailed above. Closed in COR4.

**2015-07-01 Murata-san:**

Here are some corrections:

"Part 1: §A.2, “SpreadsheetML”, p. 3,955, Lines 521–529" in DR 14-0011 is for PresentationML

"Part 1: §A.2, “SpreadsheetML”, p. 4,003, Lines 1,443–1,451" is for DrawingML.

Ditto for "Part 1: §A.2, “SpreadsheetML”, p. 4,003, Lines 1,452–1,459"

Ditto for "Part 1: §A.2, “SpreadsheetML”, p. 4,016, Lines 2,135–2,139"

Ditto for "Part 1: §A.2, “SpreadsheetML”, p. 4,031, Lines 2,904–2,940"

We have the same errors for the RELAX NG schemas.

*[Editor: Agreed; I’ve corrected the schema headings in-place, above.]*

**2016-12-10 Rex Jaeschke (in private mail to Murata-san):**

As I was revising the DR log to move to 2016 from 2012, I came across a DR that we closed for the next COR, but which is missing schema changes. It is 14-0011.

**2016-12-10 Murata-san (in private mail to Rex Jaeschke):**

It appears that the schema change for 14-0011 was mistakenly incorporated in the 2016 edition (schema attachments) and 13-0014 has never been addressed. Do we need a DR only to cancel the schema change (14-0011) made in the 2016 edition?  Or, can we simply be silent about it?

**2016-12-10 Rex Jaeschke (in private mail to Murata-san):**

Let’s handle that editorially.

**2017-02-24 Murata-san:**

Commits for the schema changes are

* <https://app.assembla.com/spaces/IS29500/subversion/commits/313>
* <https://app.assembla.com/spaces/IS29500/subversion/commits/314>
* <https://app.assembla.com/spaces/IS29500/git/commits/19cce95fac5ca35b25c879acd9693ce42033fe20>

Hit "Expand all diffs" (at the right bottom) in these pages and you will get schema diffs.

**2017-03-02 Seattle Meeting:**

Rex will add the schema changes to the proposed resolution.

**2017-04-06 Rex Jaeschke:**

I incorporated the Strict and Transitional RelaxNG schema changes, in-place, above.

On 2016-12-10, Murata-san wrote, “It appears that the schema change for 14-0011 was mistakenly incorporated in the 2016 edition”. This appears to be true for the electronic annexes only. The text in the 2016 PDF for these annexes does NOT have these changes. As such, the printed version and the electronic version of the schemas are out of synch in this regard, but I don’t see that as requiring any action on our part. Eventually, when we incorporate COR4, they’ll match.

Changes to Part 1: Y Part 2: N Part 3: N Part 4: Y