1. Display and interface structure

The display and interface section of this specification form levels 2 and 3 of XMILE compatibility respectively. The underlying assumptions for how to render the data presented in these two levels are very similar therefore we discuss their structure and assumptions together. The display and interface data must appear in child tags of their owning model tags.

Each XMILE model has one child <display> tag and one child <interface> tag. The <display> tag describes the visual representation of the stock and flow diagram while the <interface> tag describes the layout, content and appearance of the user interface.

Simple XMILE level 3 tag hierarchy example:

<xmile>

<model>

<display/>

<interface/>

</model>

<model>

<display/>

<interface/>

</model>

</xmile>

* 1. Introduction to the display level

The display level (level 2) is used to describe the visual attributes of any object appearing in the stock and flow diagram. Typical examples of these objects are Stocks, Flows, Auxiliaries, Aliases, Connectors, Sliders, Graphs and Tables. This aforementioned list is not meant to be exhaustive and a full list of all valid level 2 display objects can be found **XXX**.

Any object appearing in level 1 (Stock, Flow, Auxiliary) must have a child <display> tag in order to be level 2 compatible. Display properties on any level 1 object must appear in the child <display> tag of the level 1 object. For any object not present in level 1 (Alias, Connector, etc) the object must appear within the child <display> tag of the model.

Simple XMILE level 2 tag hierarchy example:

<xmile>

<model>

<stock>

<display>

</stock>

<display>

<alias/>

</display>

</model>

</xmile>

* 1. Introduction to the interface level

The interface level (level 3) is used to describe the layout, content and appearance of the user interface and all objects appearing within the user interface. Typical examples of these objects are: Graphs, Tables, Sliders etc. This aforementioned list is not meant to be exhaustive and a full list of all valid level 3 display objects can be found **XXX**.

Any object appearing on the user interface of an XMILE model must appear as a child tag of that model’s <interface> tag.

Simple XMILE level 3 tag hierarchy example:

<xmile>

<model>

<display/>

<interface>

<graph/>

</interface>

</model>

</xmile>

* 1. Common interface and display assumptions and attributes

All visual objects contained within an XMILE model are laid out on a 2D Cartesian coordinate space measured in pixels, where 0,0 is the top left of the screen and height runs down while width runs right. An example coordinate space map for a model which has a size of 1000px by 1000px would look like:

0,0 ------------------------------------------------- 1,000

|

|

|

|

|

|

|

|

|

1,000

All visual objects contained within an XMILE model must have the following attributes:

* Position x=”<double>”, y=”<double>”
* Size – predefined OR arbitrary
  + Predefined
    - size=”large|medium|small|name\_only\*” – default=medium
  + Arbitrary
    - width=”<double>”, height=”<double>”

\*name\_only is only applicable for display objects based on level 1 objects

The position referred to by the x and y attributes refers to the center of the object when using a predefined size. When using an arbitrary size the x and y attributes refer to the top left corner of the object. All locations and sizes of XMILE objects are represented using double precision numbers.

The predefined sizes are dependent on the object on which they are present and vary by implementation. All objects of the same type though must use the same actual width and height for each of the listed options with the exception of name\_only. The name\_only predefined size is used only on level 1 based display objects and their size is determined by the text metrics of their name plate. This option allows for the representation of level 1 objects as text only without an associated symbol.

* + - 1. Referring to specific XMILE display objects

Display objects do not have names or any other way to specifically refer to individual objects. Therefore any display object which is referred to anywhere else in the XMILE file provides a uid=”<int>” attribute. This attribute is a unique linearly increasing integer which gives each display object a way to be referred to specifically. Objects requiring a uid are listed in **XXX**. UIDs restart from 0 in each <display> or <interface> tag.

* + 1. Common styles associated with all visual XMILE objects

All level 2 and above XMILE objects provide attributes which describe their look and feel or style. Styles applied to visual XMILE objects are composed of attributes of the following core style objects plus any specific attributes available to that specific type of object.

Border

* border-width=”thick|thin|thick-thin|<double>” –default=1px
* border-color=”<hex code>|predefined color\*”
* border-style=”none|solid” –default=none

Text-Style

* font-family=”<string>”
* font-style=”normal|italic” –default=normal
* font-weight=”normal|bold” –default=normal
* text-decoration=”normal|underline” –default=normal
* text-align=”left|right|center”
* font-color=”<hex code>|predefined color\*”
* text-background=”<hex code>|predefined color\*”
* font-size=”<double>pt”
* padding=”<comma separated list of no more then 4 doubles and no fewer then 1 double>”\*\*
* any attributes of a Border object

All visual XMILE objects allow control over the following style attributes:

* color=”<hex code>|predefined color\*”
* background=”<hex code>|predefined color\*”
* z-index=”<int>” –default=-1 (-1 is bottom-most, top-most is maximum)
* any attributes of a Text-Style object

\* The list of predefined colors and their definitions appear in section 5.4.1.2

\*\* The specification for the padding attributes appears in section 5.4.1.1

* + - 1. Specification of padding

Padding is specified as the amount of area from an object’s boundary to the start of its content. Padding is broken down into four components: top, left, bottom, right and is always measured in pixels. Padding always appears as a single attribute on a visual object and can be written in any of the following forms:

1. padding=”2”
2. padding=”2,3”
3. padding=”2,3,1”
4. padding=”2,3,4,5”

The first form translates into a padding of 2px from all 4 edges. The second form translates into a padding of 2 pixels from the top and bottom edges and 3 pixels from the left and right edges. The third form translates into a padding of 2 pixels from the top edge, 3 pixels from the left and right edges and 1 pixel from the bottom edge. The fourth form translates into 2 pixels from the top edge, 3 pixels from the right edge, 4 pixels from the bottom edge and 5 pixels from the left edge.

5.4.2.2 Specification of color

Color is specified using CSS style hex codes or by using a color from the list of predefined colors. When specified using a CSS style hex code all colors must be prefixed by a # followed by up to 6 digits from 0 to F. (How much detail is really needed here, can I refer to something else)

List of predefined colors

|  |  |
| --- | --- |
| Name | Hex Code |
| aqua | #00FFF |
| black | #000000 |
| blue | #0000FF |
| fuchsia | #FF00FF |
| gray | #808080 |
| green | #008000 |
| lime | #00FF00 |
| maroon | #800000 |
| navy | #000080 |
| olive | #808000 |
| purple | #800080 |
| red | #FF0000 |
| silver | #C0C0C0 |
| teal | #008080 |
| white | #FFFFFF |
| yellow | #FFFF00 |

* + 1. The cascading style system

In order to save space in XMILE files styles can cascade onto objects using a <style> tag. A <style> tag contains style information separated into children tags by object type. A special type ‘model’ is provided to refer to all other child tags of this <style> tag. Styles specified in a style tag apply to all related objects of the specified types

Simple XMILE style tag example:

<style>

<model font-color=”blue” font-size=”10”/>  
 <stock font-weight=”bold” font-color=”black”/>

<flow font-size=”12”/>

</style>

In this example any related Stock objects would have a bold, black, size 10 font and any related Flow objects would have a normal weight, blue, size 10 font.

A <style> tag may be placed in any of the following locations which change its related objects:

* As a child of a <model> tag
* As a child of an <interface> tag
* As a child of a <display> tag

Simple XMILE level 3 style tag hierarchy example:

<xmile>

<model>

<style/>

<display>

<style/>

</display>

<interface>

<style/>

<graph/>

</interface>

</model>

</xmile>

As a child of a <model> tag the <style> tag contains all of the default and highest level styles which apply across both the stock and flow diagram as well as the user interface. The <style> tags which are children of the <display> and <interface> tags override any specifications of the ‘root’ <style> tag.

When reading in any XMILE object any attribute present on that object takes ultimate precedence. For any attribute which is missing the cascading style system is used, first looking to the nearest <display>, <interface>, or <model> tag’s child <style> tag and then if the object is inside of a <display> or <interface> tag to up to the <model> tag. The same principal holds true when writing out style attributes. If the object’s attribute contains no difference from the specified cascading style there is no reason to write out that attribute.