教育部教育信息化专项
交互式电子白板教学功能规范和教学资源通用文件格式标准研发项目

交互式电子白板教学资源
通用文件格式

中央电化教育馆

2015 年 12 月

目录

交互式电子白板数字资源通用文件格式标准

1. 概要

1.1 范围

本格式标准主要用于设计在交互式电子白板显示的内容，它们以交互对象形式存在，能够在页面上被交互操作。本标准参考英国 BECTA IWB 交互式电子白板文件格式标准，结合中国交互式电子白板行业的实践现状和发展趋势，以及引用国际上本领域相关技术规范制定而成，以推动中国交互式电子白板行业产品的标准化。

本标准适用于交互式电子白板提供厂商、交互式设备配套软件供应商、数字资源服务提供商。

1.2 目的

随着交互式电子白板在教学中日益广泛的应用，研制交互式电子白板教学资源通用文件格式标准对于促进交互式电子白板更好地为教学服务，加强交互式电子白板资源的交流和共享具有重要意义。当前，大量交互式电子白板资源和相关服务进入了课堂，广大教师对交互式电子白板在提高教学质量方面的作用给予了充分肯定，但是交互式电子白板在使用过程中存在着教学资源难以共享，尤其是教学过程中生成性资源无法在不同品牌交互式电子白板之间共享使用，这对资源共享和教学研讨造成了较大困难，本标准就是为这一实践问题的解决，提高交互式电子白板数字资源的跨品牌互操作共享的能力。

本格式标准主要目的有如下两个方面：

（1） 建立能够打开、编辑、存储交互式电子白板数字资源通用文件；

（2） 鼓励和引导行业中各厂家电子白板数字资源格式文件参照本标准，支持不同厂家电子白板数字资源文件进行交换和共享，使教学内容能够在不同厂家产品中交换使用。

本标准主要是为第二个目的服务。为了达到上述目的，目标格式标准必须是简约的，并且可扩展的，以能够确保兼容性。

这个格式被称为“交互式电子白板数字资源通用文件格式”，本格式标准延续 BECTA IWB 格式定义规则，以 SVG 规范为基础，以 IWB 标签及属性进行拓展，文件的格式后缀简写“IWB”。

2. 术语与引用

2.1 术语

2.1.1 交互式电子白板

Interactive Whiteboard
交互式电子白板是硬件电子感应白板（White Board）和软件白板操作系统的集成，能够实现使用者与系统之间的信息交流。

2.1.2 交互式电子白板文件阅读器

Reader
显示文档的应用程序。

2.1.3 交互式电子白板文件编辑器

Write
保存文档的应用程序，无论是最初的创造或重新保存现有的文件。

2.1.4 文件

File
计算机上的实际文件。

2.1.5 文档

Document
文档是对个例的交互式电子白板数字资源通用文件通称。一个文档是有一个或多个页面的集合。

2.1.6 页面

Page

是交互式电子白板数字资源通用文件的构成单位。页面是承载交互式电子白板各类元素（elements）的载体。

2.1.7 元素

Elements
出现在一个页面上的对象实例（objects）。

2.1.8 交互

Interaction

用户与系统之间的信息交流。

2.1.9 页面坐标系

Page Coordinates
用户页面内元素图形变换的起始坐标系，支持缩放、平移、旋转、变形等操作。

2.1.10 元素坐标系

Element Coordinates
元素内子元素图形变换的起始坐标系，支持缩放、平移、旋转、变形等操作。

2.1.11 标量矢量图形格式

SVG
一种国际上标准化的图形格式。

2.1.12 SVG 微小规范

SVGT
SVG1.0 规范基础上的精简版。

2.1.13 渲染服务器

Paint server
这是一个 SVG 术语，用渐变或图案模式填充页面中的对象。IWB 交互电子白板格式中允许填充纯色。

2.2 引用

- XML 1.0 - http://www.w3.org/TR/REC-xml/
- SVG 1.1 - http://www.w3.org/TR/SVG11/
- SVG 1.2 (draft) - http://www.w3.org/TR/2004/WD-SVG12-20041027/
- SVG Tiny 1.2 (draft) - http://www.w3.org/TR/SVGMobile12/
- CSS2 - http://www.w3.org/TR/REC-CSS2/cover.html
- Becta Interactive Whiteboard Common File Format. Version1.1.1 May 2009. http://www.becta.org.uk （BECTA，British Educational Communications and Technology Agency）

3. 标准中的标签及其属性定义

在 IWB 文件中出现的标签和属性列表如下：

3.1 BECTA IWB 标签

BECTA IWB v1.1.1 中拓展的 IWB 标签有 6 类，具体描述如下表：

- **iwb:element**
- **iwb:group**
- **iwb:iwb**
- **iwb:link**
- **iwb:meta**
- **iwb:tspan**

3.2 拓展的 IWB 标签

本标准的拓展 IWB 元素标签有 4 类。

- **iwb:Layout** 布局标签
- **iwb:Placeholders** 布局的占位符标签
- **iwb:TextPlaceholder** 文本占位符标签
- **iwb:PicturePlaceholder** 图片占位符标签

本标准的拓展对象标签有 3 类。

- **iwbx:** stroke 画笔对象
- **iwbx:** table 表格对象
- **iwbx:** artword 艺术字对象

表 1: 标签的详细说明

iwb:element		元素的附加属性
属性	ref	强制性的。关联它所引用SVG元素的ID。
	background	可选。确定引用的图像或矩形是否被用作背景。
	background-fill	可选。引用的文本区域或文本的背景颜色。
	background-posture	可选。确定引用图像如何在背景中显示。
	flip	可选。所引用图像的方向。
	freehand	可选。指出被引用的折线是手写创建的。
	highlight	可选。指出被引用的折线是为了突出某页面的一部分。
	highlight-fill	可选。在引用的文本区域或文本的背景加亮的颜色。
	list-style-type	可选。所引用的文本区域中列表标记的外观。
	life-style-type-fill	可选。所引用的文本区域中列表标记的颜色。

	locked	可选。表明所引用的元素是否被锁定在页面上。
	replicate	可选。表明所引用的元素是否是自复制状态，而不是移动。
	revealer	可选。表明所引用的矩形是否用于揭示某页面的特定区域。
	stroke-lineshape-start	可选。添加一个形状至引用线或折线的起始端。
	stroke-lineshape-end	可选。增加了一个形状至引用线或折线的结束端。
	class	强制性的。拓展对象的类型，如：三角形triangle、平行四边行parallelogram、菱形diamond、梯形trapezia、箭头upArrow、角angle、 弧arc；
	bound	可选。对象外框结点，值为左上角和右下角点的坐标x,y序列；
	rotate-base	可选。旋转基点，值为基点的坐标x,y序列；
	radius	可选。弧的半径，单位像素；
	start-angle	可选。弧的起始角度，从 x 轴到弧线的起始点沿顺时针方向度量的角度，单位度；
	sweep-angle	可选。弧的划过角度，从 startAngle 参数到弧线的结束点沿顺时针方向度量的角，单位度；
例子	<iwb:element ref=”rect1” locked=”true” />	
也可参见	SVG元素（例如svg:rect，svg:text和svg:image）	

iwb:group	元素的集合
子标签	<iwb:element>
例子	<iwb:group></iwb:group>

iwb:iwb		根元素
属性	xmlns:iwb	强制性的。IWB命名空间。
	xmlns:svg	强制性的。SVG命名空间。
	xmlns:xlink	强制性的链接。 XLINK命名空间。
	version	强制性的。正在使用的IWB格式版本。
子 标签	<iwb:element>, <iwb:group>, <iwb:link>, <iwb:meta>, <iwb:tspan>, <svg:svg>	
例子	<iwb xmlns:iwb="http://www.becta.org.uk/iwb" xmlns:svg=http://www.w3.org/2000/svg version=”1.0”> </iwb>	

iwb:link		到文件的链接
属性	ref	强制性的。保存指向svg:a的ID。
	file	强制性的。说明是否链接到某文件，该文件可以是ZIP包的内部或是外部的。
例子	<iwb:link ref=”link1” file=”external” />	

iwb:meta		文件的附加描述元数据
属性	name	附加的元数据名字
	content	附加的元数据
例子	<iwb:meta name=”owner” content=”John Smith” />	

iwb:tspan		tspans的附加属性
属性	ref	强制性的。关联它所引用SVG元素的ID。
	background-fill	可选。表明tspan所关联元素的背景色。
	highlight-fill	可选。表明tspan所关联元素的背景高亮颜色。
	list-style-type	可选。列表标记的外观。
	list-style-type-fill	可选。列表标记的颜色。
	type	可选。该tspan所引用的类型。
例子	<iwb:tspan ref=”tspan1” type=”list” />	
也可参见	svg:tspan	

iwb:layout		布局标签
子标签		<iwb:Placeholders>
属性	name	可选。表明布局的名称。
	type	可选。表明布局的类型。
例子	<iwb:layout name=”标题和内容” type=”titleAndContent”> <iwb:placeholders> <!--这里是具体的占位符--> </iwb:placeholders> </iwb:layout>	

iwb:placeholders		布局的占位符标签
子标签		<iwb:textplaceholder>、<iwb:pictureplaceholder>
例子	<iwb:placeholders> <iwb:textplaceholder> <!--这里是文本占位符内容--> <iwb:textplaceholder> <iwb:pictureplaceholder> <!--这里是图片占位符内容--> <iwb:pictureplaceholder> </iwb:placeholders>	

iwb:textplaceholder		文本占位符标签
子标签		<iwb:textplaceholder>、<iwb:pictureplaceholder>
属性	x	可选。占位符的水平位置。

	y	可选。占位符的垂直位置。
	width	可选。占位符的宽度。
	height	可选。占位符的高度。
	type	可选。占位符的类型。可选值为title|subtitle|body
	content	可选。占位符的内容。其值为<svg:text>。
例子	<iwb:TextPlaceholder x=”120” y=”60” width=”1040” height=”120” type=”title”> <content> <svg:text font-family="Microsoft YaHei UI" font-size=”48”> 双击此处添加标题 </svg:text> </content> </ iwb:TextPlaceholder>	

iwb:picturePlaceholder		图片占位符标签
子标签		<iwb:TextPlaceholder>、<iwb:PicturePlaceholder>
属性	x	可选。占位符的水平位置。
	y	可选。占位符的垂直位置。
	width	可选。占位符的宽度。
	height	可选。占位符的高度。
	type	可选。占位符的类型。
例子	<iwb:TextPlaceholder x=”120” y=”192” width=”1040” height=”468”>	

iwbx:stroke		扩展画笔对象
子标签		<iwb:element> <iwb:transform> <iwb:data>
属性	id	强制性的。扩展对象的标识。
	type	可选。表明对象类型。
	locked	可选。表明对象的锁定状态。
	visible	可选。表明对象的显隐状态。
	uclone	可选。表明对象是否能无限克隆。
	width	可选。对象所采用的线宽。
	color	可选。对象的颜色
	stroke-type	强制性。线型
	stroke-start	强制性。线头
	Stroke-end	强制性。线尾
例子	<iwbx:stroke id=”” type=”curve”locked=””visible=””uclone=””width=””color=”” stroke-start=”” stroke-end=”” stroke-type=””> <iwbx:element > //线头、线尾、线型 </iwbx:element> <iwbx:transform>	

	</iwbx:transform> <iwbx:data> <iwb:vertex points="x1,y1,x2,y2,……,xn,yn"/> </iwbx:data> </iwbx:stroke>

iwbx:table		扩展表格对象
子标签		<iwb:element> <iwb:transform> <iwb:data>
属性	id	强制性的。扩展对象的标识。
	type	可选。表明对象类型。
	locked	可选。表明对象的锁定状态。
	visible	可选。表明对象的显隐状态。
	uclone	可选。表明对象是否能无限克隆。
	width	可选。对象所采用的线宽。
	color	可选。对象的颜色。
例子	<iwbx:table id="" type="table" locked=""visible=""uclone="" layer=""> <iwbx:element> </iwbx:element> <iwbx:transform> </iwbx:transform> <iwbx:data> <rect points="x1,y1,x2,y2"/> <!--网格信息--> <columns> <column width=""/> <!--width 列宽--> </columns> <rows> <row heigh=""> <!--height 行高--> <cell rowSpan="" columnSpan="" horzMerge="" vertMerge=""> <!--单元格属性 rowSpan y 方向上的合并的跨度 columnSpan x 方向上合并的跨度，horzMerge 水平合并标签 vertMerge 竖直合并标签 -->	

	<object id=""> </object> <!--单元格内的对象--> </cell> </row> </rows> </iwbx:data> </iwbx:table>

iwbx:artword		扩展艺术字对象
子标签		<iwb:element> <iwb:transform> <iwb:data>
属性	id	强制性的。扩展对象的标识。
	type	可选。表明对象类型。
	locked	可选。表明对象的锁定状态。
	visible	可选。表明对象的显隐状态。
	uclone	可选。表明对象是否能无限克隆。
	width	可选。对象所采用的线宽。
	color	可选。对象的颜色。
例子	<iwbx:artword uclone="" id="" layer="" visible="" type=" " locked="0"> <iwbx:transform> </iwbx:transform> <iwbx:element> <artpath type=" "/> <artoutline color=" " type=" " width=""/> <artfill color="" type=" "/> </iwbx:element> <iwbx:data> <*svg:text x=”30” y=”10” font-size=”20”>Hello world!</svg:text*> <rect points=""/> </iwbx:data> </iwbx:artword>	

3.3 标准中的 SVG 标签引用

本标准中，引用了 SVG 中的标签，具体描述如下表：

- **svg:a**
- **svg:circle**
- **svg:ellipse**
- **svg:g**
- **svg:image**
- **svg:line**
- **svg:video**
- **svg:polygon**
- **svg:polyline**
- **svg:rect**
- **svg:svg**
- **svg:switch**
- **svg:tbreak**
- **svg:text**
- **svg:textarea**
- **svg:tspan**

表2: 对所引用SVG标签的说明

svg:a		创建一个链接
属性	id	如果被引用，则需要强制。id标识此链接。
	xlink:href	强制性的。该链接的URI。
子标签	<svg:circle>, <svg:ellipse>, <svg:image>, <svg:line>, <svg:video>, <svg:polygon>, <svg:polyline>, <svg:rect>, svg:tbreak>, <svg:text>, <svg:textarea>, <svg:tspan>	
例子	<svg:a xlink:href=”http://schools.becta.org.uk/”> </svg:a>	
也可参见	<iwb:link>	

svg:circle		在页面上绘制一个圆。
属性	id	如果被引用，则需要强制。ID标识圆。
	cx	强制性的。圆心的x坐标。
	cy	强制性的。圆心的y坐标。
	r	强制性的。圆的半径。
	fill	可选。圆的填充颜色。
	fill-opacity	可选。填充的透明度。
	stroke	可选。圆边缘的颜色。
	stroke-dasharray	可选。逗号分隔的短平线和空白间距相间模式的圆圈边缘图案。
	stroke-linecap	可选。线的末端如何呈现。只有在stroke-dasharray也设置

		的时候才有效。
	stroke-opacity	可选。圆形边缘的透明度。
	stroke-width	圆圈边缘的宽度。
	transform	可选。圆的旋转和/或翻转
例子	<svg:circle cx=”50” cy=”50” r=”30” />	
也可参见	iwb:element的锁定和复制属性。	

svg:**ellipse**		在页面上画一个椭圆
Attributes属性	id	强制性的。标识该椭圆的ID。
	cx	强制性的。椭圆中心的x坐标位置。
	cy	强制性的。椭圆中心的y坐标位置。
	rx	强制性的。在x方向上椭圆的半径。
	ry	强制性的。在y方向上椭圆的半径。
	fill	可选。椭圆的填充颜色。
	fill-opacity	可选。填充的透明度。
	stroke	可选。椭圆边缘的颜色。
	stroke-dasharray	可选。逗号分隔的短平线和空白间距相间模式的边缘图案。
	stroke-linecap	可选。线条的末端如何呈现。只有在stroke-dasharray也设置的情况下有效。
	stroke-opacity	可选。椭圆边缘的透明度。
	stroke-width	可选。椭圆边缘的宽度。
	transform	可选。椭圆的旋转与/或翻转。
Example例子	*<svg:ellipse cx=”700” cy=”350” rx=”350” ry=”175”/>*	
See also也可参见	iwb:element的“locked”和“replicate”属性。	

svg:**g**		包容其他SVG元素的容器标签。如果该容器中某个特定属性对一个特定元素有效的话，它将应用到所有包含的元素。
Attributes 属性	fill	可选。该容器包含的任何形状和文本元素的填充颜色。
	fill-opacity	可选。填充颜色的透明度。
	stroke	可选。包含的任何线条和形状的线条颜色。
	stroke-dasharray	可选。逗号分隔的短平线和空白间距相间模式的边缘图案。
	stroke-linecap	可选。线条的末端如何呈现。
	stroke-linejoin	可选。连接的交叉线如何呈现。
	stroke-opacity	可选。包含的线条和形状的透明度。
	stroke-width	可选。包含的线条和形状的宽度。
	transform	可选。包含的所有元素的旋转和翻转。
Sub tags 子标签	*<svg:a>, <svg:circle>, <svg:ellipse>, <svg:g>, <svg:image>, <svg:line>, <svg:video>, <svg:polygon>, <svg:polyline>, <svg:rect>, <svg:switch>, <svg:text>, <svg:textarea>*	

Example 例子	*<svg:g fill=”red” transform=”rotate(10)”> <!-- SVG elements here -- > </svg:g>*

svg:**image**		页面中放置的图片。
Attributes 属性	id	如果被引用，则需要强制。Id标识该图片。
	xlink:href	强制性的。指向图片的内部URL。
	x	强制性的。图片的左边位置。
	y	强制性的。图片的上边位置。
	width	强制性的。图片的宽度。
	height	强制性的。图片的高度。
	fill-opacity	可选。图片的透明度。
	requiredExtensions	可选。扩展信息，以方便阅读器显示此图片。
	transform	可选。图片的旋转和/或翻转。
Example 例子	*<svg:image xlink:href=”images/enterprise.png” x=”0” y=”0” width=”300” height=”300” />*	
See also 也可参见	iwb:element的“*flip*”、“*locked*”和“*replicate*”属性。	

svg:**line**		页面中绘制的线条
Attributes 属性	id	如果被引用，则需要强制。id标识该线条。
	x1	强制性的。线条始端的x坐标。
	y1	强制性的。线条始端的y坐标。
	x2	强制性的。线条末端的x坐标。
	y2	强制性的。线条末端的y坐标。
	stroke	可选。线条的颜色。
	stroke- dasharray	可选。逗号分隔的短平线和空白间距相间模式的边缘图案。
	stroke-linecap	可选。线条末端如何呈现。
	stroke-opacity	可选。线条的透明度。
	stroke-width	可选。线条的宽度。
	transform	可选。线条的旋转和/或翻转。
Example 例子	*<svg:rect x1=”150” y1=”50” x2=”100” y2=”50” />*	
See also 也可参见	iwb:element的“*stroke-lineshape-start*”、“*stroke-lineshape-end*”、“*locked*”和“*replicate*”属性。	

svg:**video**		页面上的视频（或flash动画）
Attributes 属性	id	如果被引用，则需要强制。ID标识该视频。
	xlink:href	强制性的。指向视频的内部URL。
	x	强制性的。视频的左边位置。
	y	强制性的。视频的顶部位置。
	width	强制性的。视频的宽度。
	height	强制性的。视频的高度。
	requiredExtensions	可选。扩展信息，以方便阅读器显示此视频。

	transform	可选。视频的旋转和/或翻转。
Example 例子	*<svg:video xlink:href="videos/tardis.mpeg" x="0" y="0" width="300" height="300" />*	
See also 也可参见	iwb:element的“*flip*”、“*locked*”和“*replicate*”属性。	

svg:**polygon**		绘制页面上的多边形
Attributes 属性	id	如果被引用，则需要强制。Id标识该多边形。
	points	强制性的。表示该多边形形状边缘的x和y点列表。
	ill	可选。多边形的填充颜色。
	fill-opacity	可选。填充颜色的透明度。
	stroke	可选。多边形边缘的颜色。
	stroke-dasharray	可选。逗号分隔的短平线和空白间距相间模式的边缘图案。
	stroke-linecap	可选。线条末端如何呈现。只有在stroke-dasharray设置的时候有效。
	stroke-linejoin	可选。边缘的节点如何呈现。
	stroke-opacity	可选。多边形边缘的透明度。
	stroke-width	可选。多边形边缘的宽度。
	transform	可选。多边形的旋转和/或翻转。
Example 例子	*<svg:polygon points="50,375 150,375 150,325 250,325" />*	
See also 也可参见	iwb:element的“*locked*”和“*replicate*”属性。	

svg:**polyline**		在页面上绘制折线。
Attributes 属性	id	如果被引用，则需要强制。id标识该折线。
	points	强制性的。表示一条或多条连接在一起的线条的x和y点的列表。
	stroke	可选。折线的颜色。
	stroke-dasharray	可选。逗号分隔的短平线和空白间距相间模式的边缘图案。
	stroke-linecap	可选。折线末端如何呈现。
	stroke-opacity	可选。折线的透明度。
	stroke-width	可选。折线的宽度。
	transform	可选。折线的旋转和/或翻转。
Example 例子	*<svg:polyline points="50,375 150,375 150,325 250,325" />*	
See also 也可参见	iwb:element的“*stroke-lineshape-start*”、“*stroke-lineshape-end*”、“*freehand*”、“*highlight*”、“*locked*”和“*replicate*”属性。	

svg:**rect**		在页面上绘制矩形。
Attributes 属	id	如果被引用，则需要强制。id标识该矩形。

性	x	强制性的。矩形的左边位置。
	y	强制性的。矩形的顶部位置。
	width	强制性的。矩形的宽度。
	height	强制性的。矩形的高度。
	fill	可选。矩形的填充颜色。
	fill-opacity	可选。填充颜色的透明度。
	stroke	可选。矩形边缘的颜色。
	stroke-dasharray	可选。逗号分隔的短平线和空白间距相间模式的边缘图案。
	stroke-linecap	可选。线条的末端如何呈现。只有在stroke-dasharray也设置的情况下有效。
	stroke-linejoin	可选。矩形的中心如何呈现。
	stroke-opacity	可选。矩形边缘的透明度。
	stroke-width	可选。矩形边缘的宽度。
	transform	可选。矩形的旋/翻转。
Example 例子	*<svg:rect x=”0” y=”0” width=”300” height=”300” />*	
See also 也可参见	iwb:element的“*revealer*”、“*locked*”和“*replicate*”属性。	

svg:**svg**		标识文件的SVG部分。
Attributes 属性	viewbox	强制性的。页面上的显示为“幻灯片”的区域。
	width	可选。以像素为单位，屏幕的原始宽度。
	height	可选。以像素为单位，屏幕的原始高度。
Sub tags 子标签	*<svg:a>, <svg:circle>, <svg:ellipse>, <svg:g>, <svg:image>, <svg:line>, <svg:video>, <svg:polygon>, <svg:polyline>, <svg:rect>, <svg:switch>, <svg:text>, <svg:textarea>*	
Example 例子	*<svg:svg width=”800” height=”600” viewbox=”0 0 8000 6000”> </svg:svg>*	

svg:**switch**	开关，以支持从选项中选择元素。
Sub tags 子标签	*<svg:circle>, <svg:ellipse>, <svg:g>, <svg:image>, <svg:line>, <svg:video>, <svg:polygon>, <svg:polyline>, <svg:rect>, <svg:text>, <svg:textarea>*
Example 例子	*<svg:switch> </svg:switch>*

svg:**tbreak**	在文本区域（Textarea）内添加一条线。
Example 例子	*<svg:textarea x=”0” y=”0” width=”100” height=”50”>Hello<tbreak />world!</svg:textarea>*
See also 也可参见	*svg:textarea*

svg:**text**	单行文本。

Attributes	id	如果被引用，则需要强制，id标识文本。
	x	强制性的。文本基线的x位置。
	y	强制性的。文本基线的y位置。
	fill	可选。该文本的默认颜色。
	font-family	可选。该文本的默认字体。
	font-size	可选。该文本的默认字号。
	font-style	可选。该文本的默认样式。
	font-weight	可选。该文本的默认权重。
	font-stretch	可选。默认只有横向伸展此文。
	transform	可选。文本的旋转和/或翻转。
Sub tags 子标签	*<svg:tspan>*	
Example 例子	*<svg:text x="30" y="10" font-size="20">Hello world!</svg:text>*	
See also 也可参见	*iwb:tspan*	

svg:**textarea**		多行段落文本区域。
Attributes 属性	id	如果被引用，则需要强制。ID标识文本区域。
	x	强制性的。文本区域的左边缘。
	y	强制性的。文本区域的上边缘。
	width	强制性的。文本区域的宽度。文字会自动换行到此宽带的下一行。
	height	强制性的。文本区域的高度。
	fill	可选。文本区域的默认文字颜色。
	font-family	可选。文本区域的默认字体。
	font-size	可选。文本区域的默认字号。
	font-stretch	可选。默认只有横向伸展此文。
	font-style	可选。文本区域的默认样式。
	font-weight	可选。文本区域的默认权重。
	text-align	可选。文本区域的默认对齐方式。
	transform	可选。文本区域的旋转和/或翻转。
Sub tags 子标签	*<svg:tbreak>, <svg:tspan>*	
Example 例子	*<svg:textarea x="0" y="0" width="100" height="50" fill="red">Hi Earthlings!</svg:textarea>*	
See also 也可参见	*iwb:tspan*	

svg:**tspan**		单行文本格式标签。
Attributes	id	如果被引用，则需要强制。标识tspan的id

. . .	fill	可选。tspan的文本颜色。
	font-family	可选。tspan的文本字体。
	font-size	可选。tspan的文本字号。
	font-stretch	可选。在tspan中只有横向伸展此文。
	font-style	可选。tspan的文本样式。
	font-weight	可选。tspan的文字权重。
	text-align	可选。在tspan中文本的对齐方式。仅用于当一个textarea的子标签。
Sub tags 子标签	*<svg:tbreak>, <svg:tspan>*	
Example 例子	*<svg:text x=”50” y=”0” fill=”black”>Greetings <tspan fill=”green”>Gaia</tspan>!</svg:text>*	
See also 也可参见	*svg:text, svg:textarea* 和*iwb:tspan*	

4. 标准中标签元素的必选集和可选集

标签和属性的“必选集”是应用程序做IWB兼容测试所必须支持的最小集合，“可选集”是必须集的超集，包括本标准所包含的所有属性和标签。

不能由阅读器处理的属性可能被跳过或是忽略，尽管这取决于跳过了什么，但如果其处理过程是阅读器有过多的属性或标签不能对原始文件进行解析，用户可能需要被告知。

这里没有明确列出的属性是必选集的一部分。

表 3: 对各标签元素必选和可选的说明。

Tag 标签	Attributes 属性	必选或者可选 Core or Full
iwb:element		必选
	background	必选
	background-fill	可选
	background-posture	可选
	flip	必选
	freehand	可选
	highlight	可选
	highlight-fill	可选
	list-style-type	可选
	list-style-type-fill	可选
	locked	必选
	replicate	可选
	revealer	可选
	stroke-lineshape-start	可选
	stroke-lineshape-end	可选
iwb:group		必选
iwb:iwb		必选
	xmlns:iwb	必选
	xmlns:svg	必选
	xmlns:xlink	必选
	version	必选
iwb:link		必选
iwb:meta		必选
iwb:tspan		必选
	background-fill	可选
	highlight-fill	可选
	list-style-type	可选
	list-style-type-fill	可选
	type	必选
iwbx:stroke		必选

iwbx:table		必选
iwbx:artwork		必选
svg:a		必选
	xlink:href	必选
	xlink:href=”*.wav”	必选
	xlink:href=”*.mp3”	可选
svg:a		必选
	fill	必选
	fill-opacity	必选
	stroke	必选
	stroke-dasharray	可选
	stroke-linecap	可选
	stroke-opacity	可选
	stroke-width	必选
	transform	必选
svg:cirvle		必选
	fill	必选
	fill-opacity	必选
	stroke	必选
	stroke-dasharray	可选
	stroke-linecap	可选
	stroke-opacity	可选
	stroke-width	必选
	transform	必选
svg:ellipse		必选
	fill	必选
	fill-opacity	必选
	stroke	必选
	stroke-dasharray	可选
	stroke-linecap	可选
	stroke-opacity	可选
	stroke-width	必选
	transform	必选
svg:g		必选
	fill	必选
	fill-opacity	必选
	stroke	必选
	stroke-dasharray	可选
	stroke-linecap	可选
	stroke_linejoin	可选
	stroke-opacity	可选
	stroke-width	必选
	transform	必选

svg:image		必选
	xlink:href=”*.jpg”	必选
	xlink:href=”*.bmp”	必选
	xlink:href=“*.gif”	必选
	xlink:href=”*.png”	必选
	xlink:href=”*.wnf”	可选
	xlink:href=”*.emf”	可选
	xlink:href=”*.tif”	可选
	fill-opacity	可选
	requiredExtensions	必选
	transform	必选
svg:line		必选
	stroke	必选
	stroke-dasharray	可选
	stroke-linecap	可选
	stroke-opacity	可选
	stroke-width	必选
svg:video	transform	必选
		必选
	xlink:href=”*.swf”	可选
	xlink:href=”*.mpeg”	必选
	requiredExtensions	必选
	transform	必选
svg:page		必选
svg:pagest		必选
svg:polygon		必选
	fill	必选
	fill-opacity	必选
	stroke	必选
	stroke-dasharray	可选
	stroke-linecap	可选
	stroke_linejoin	可选
	stroke-opacity	可选
	stroke-width	必选
	transform	必选
svg:polyline		必选
	stroke	必选
	stroke-dasharray	可选
	stroke-linecap	可选
	stroke-opacity	可选
	stroke-width	必选
	transform	必选
svg:rect		必选

	stroke	必选
	stroke-dasharray	可选
	stroke-linecap	可选
	stroke-opacity	可选
	stroke-width	必选
	transform	必选
svg:svg		必选
	viewbox	必选
	height	可选
	width	可选
svg:switch		必选
svg:tbreak		必选
svg:text		必选
	fill	必选
	font-family	必选
	font-size	必选
	font-style	必选
	font-weight	必选
	font-stretch	可选
	transform	必选
svg:textarea		必选
	fill	必选
	font-family	必选
	font-size	必选
	font-stretch	可选
	font-style	必选
	font-weight	必选
	text-align	必选
	transform	必选
svg:textarea		必选
	fill	必选
	font-family	必选
	font-size	必选
	font-stretch	可选
	font-style	必选
	font-weight	必选
	text-align	必选

5. 文档概述

5.1 文档构成信息

此文件使用可扩展标签语言（XML）1.0 规范。

所有 IWB 文件在它的名字之后带有“IWB”扩展名（例如:“afile.iwb”）。该文件是一个 Zip 压缩文件，这个 zip 压缩文件可以存储内部自身的多个文件，当其他媒体需要时可以使用这些文件夹进行存储，如附加的图像等。压缩文件将包含一个文本文件 content 来描述这个 IWB 文档。

5.2 IWB 压缩文件

IWBX 压缩文件由以下文件和文件夹组成。

压缩文件必须包含 1 个 XML 文件来描述文档，带有至少 4 个文件夹。为每个媒体设置一个文件夹，一个文件夹放置文档的缩略图，一个文件夹存放其他的附加文件。

文件：“content.xml”——文件的主要内容；

文件夹：“pages”——存放页面文件的独立文件夹。为保障交互式电子白板的交互响应性，所有的页面文件都单独存储；

文件夹：“media”——如果在文件中有用到包括视频、声音、动画等媒体文件就需要创建该文件夹，且依据各自类型使用相对应的独立存储空间；

文件夹：“thumbnails”——如果在文件中有用到缩略图就需要创建；

文件夹：“layouts”——如果在文件中有页面模板格式文件存在；

文件夹：“additional”——如果在文件中包括附加的文件需要创建。

媒体文件夹可以添加子文件夹，这对于包含大量媒体或动画的文档比较有用。为防止在同一个文件夹中文件名冲突，实际媒体文件必须唯一命名，并且必须在名字中保留文件的扩展名，以便能够明确确认其文件格式。

缩略图文件包括文档中关键页面的可选图像。通常第一页具有缺损的缩略图文件，其它页也可依据需要而设定缩略图文件。文件名应该以“thumbnail”开头，最好是一个可移植网络图形文件或位图文件，例如“thumbnailpag1.png”，缩略图文件主要是给应用程序和操作系统显示预览文件。

表 4： IWB 压缩文件文档结构

文件/文件夹	路径	描述
content.xml	/	描述主要内容
pages	/	存放页面内容文件

page(index).svg	/pages/	描述页面主要内容
thumbnails	/	文件缩略图存放文件夹
media	/	存放引用到的媒体文件
images	/media/	存放引用到的图片
videos	/media/	存放引用到的视频
audio	/media/	存放引用到的声音
flash	/media/	存放引用到的动画
layouts	/	存放页面模板文件格式
additional	/	存放附加的文件

5.3 XML 文档文件

该文件标准格式采用矢量图形格式 SVG 来表示大部分内容，大部分 SVG 特性参考完整版 SVG1.0、SVG1.1 和 SVG1.2 规范进行简化，以支持 IWB 格式的简便性和可操作性；另一方面，为保证规范遵从性，除非有明确的定义规则，在这个文件格式中的 SVG 标签和属性必须遵循 SVG 标准规范中的定义。SVG 标签及其属性值的描述请参考国际标准定义，在本标准中不再一一说明。

SVG 规范之外附加属性由拓展的 IWB 标签完成，**本标准是在英国 BECTA IWB（2009）版本上进行了进一步修订**，扩展了当前交互式电子白板对复杂交互性的支持。**本标准将对 BECTA IWB 的拓展进行详细描述。**

所有 IWB 文件需要按照如下的序列组合，主要的 XML 和 IWB 声明放在最前面，其次是元数据标签，接下来是 SVG 部分，最后是 IWB 标签部分。文件的 SVG 部分包含在<svg:svg> 标签中，IWB 元标签放在 SVG 部分之前，附加的 IWB 属性放在其之后。

XML 注释可以出现在文件任何位置，用“<!--”和“-->”进行标识，注释会被阅读器应用程序所忽视，只在读取当前文件时有所帮助。

下面是基本的 IWB 文件格式结构：

```
<?xml version=”1.0”?>
  <iwb xmlns:iwb="http://www.becta.org.uk/iwb"
    xmlns:svg="http://www.w3.org/2000/svg"
    version=”1.0”>
    <!-- Meta data describing file -->
    <!—文档的元数据描述部分 -->
    <svg:svg>
      <!-- Contents of document, the svg tags -->
      <!—以 SVG 标签标识的文档内容 -->
```

```
    </svg:svg>
      <!-- Additional properties of elements, the iwb tags -->
      <!—以 iwb 标签标识的附件元素属性-->
  </iwb>
```

文档的元数据描述部分包含文档基本信息，包括以下几个方面：

- 拥有者——本文档创建者的名字；
- 描述——有关本文档包含内容的描述；
- 生成程序——创建本文档时应用程序的名称及其版本信息。

“拥有者”和“描述”是可选项，取决于文档生成程序是否允许设置这些功能。

“生成程序”也是可选的，大部分情况下需要添加。

元数据描述是由标签<iwb:meta>支持添加的，包括两个属性“name”、“content”,示例如下：

```
<iwb:meta name=”description” content=”A description of the file” />
```

依据需要，上述元数据标签可以被扩展来涵盖其他信息，如一个基本文件处于中心并由一个红色矩形框锁定，示例如下：

```
<?xml version=”1.0”?>
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
    xmlns:svg="http://www.w3.org/2000/svg"
    version=”1.0”>
  <iwb:meta name=”owner” content=”BECTA” />
  <iwb:meta name=”description” content=”A little red box” />
  <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:rect id=”rect1” fill=”red” x=”450” y=”450”
      width=”100” height=”100” />
  </svg:svg>
  <iwb:element ref=”rect1” locked=”true” />
</iwb>
```

6. 文档布局

文档由一系列页面（Slide）组成，页面是容纳交互对象的容器，它包含所有元素集合和基本属性，这些页面在文档中的位置即为其显示的页码顺序。

一个文档可以包含零个或多个页面。当文档包含多个页面时，页面内容存放在文档文件/pages/文件夹下，并按照一定规则进行命名，每个页面需要唯一命名。当文档包含零个页面（即不包含任何页面）时，交互式电子白板软件可拒绝对此文档进行演示，但允许进行编辑。

一个页面所具有的属性主要为：

- 名称及顺序——包含一个在此文档中不重复的名字，用来标识该页面；
- 页面模板——带有预先设定的布局和样式；
- 页面填充效果——包括页面的背景填充类型和方式；
- 视图——创建此页面的编辑区域大小，显示此文件时可据此进行缩放；
- 一个页面可以包含零个或多个元素。

6.1 页面

6.1.1 页面名称及顺序

content.xml 主文件中<iwb:resource>标签里<iwb:file>子节点顺序即为页面顺序。

页面在创建时，根据创建顺序，会生成 page(x).svg 文件，如第一个被创建的页面会生成 page0.svg，第二个被创建的页面会生成 page1.svg，页码顺序则按照子节点顺序确定。

在如下的代码中，生成了共三个页面，依次分别为 page0，page1，page2。然后将 page0 调整为最后一页：

```
<iwb:resource identifier="pages">
    <iwb:file href="pages\page1.svg"/>
    <iwb:file href=" pages\page2.svg"/>
    <iwb:file href=" pages\page0.svg"/>
</iwb:resource>
```

注：

1） content.xml 作为主文件，不仅是页面文件的索引，也可以作为资源、操作记录等文件的索引，因此通过 identifiler 属性来区分，其中 identifiler 值为 pages 的 resource 标签记录了页面。

2） iwb:file 标签定义文件的存储位置，其中 href 属性值是文件存储的相对路径。

页码的显示和隐藏通过实现软件来控制，不记录到文件中。

6.1.2 页面模板

页面模板是页面预设的布局和样式。每个页面可以有不同的模板，但最多只能应用一个模板。可以通过<iwb:resource>标签里<iwb:file>子节点引用模板，例如。

```
<iwb:resource identifier=”layouts”>
<iwb:file href=”layouts\layout1.xml”>
</iwb:resource>
```

latyout1.xml 定义了一组内容占位符和样式。

当页面应用了某个模板时，该页面的内容就应用了对应的布局和样式；没有引用模板时，则应用原有的布局和样式。

页面模版布局可以通过如下结构的元素来定义：
布局文件通过定义一系列占位符来描述页面布局。下面是基本的 Layout 文件格式结构。

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
  xmlns:svg="http://www.w3.org/2000/svg"
  version=”1.0”>
   <iwb:Layout name=”” type=””>
     <iwb:Placeholders>
        <!—占位符信息-->
     </iwb:Placeholders>
   </iwb:Layout>
</iwb>
```

name 属性表示布局名称，type 表示布局类型，一个布局可以包含不同类型的占位符，但至少包含一个。不同类型的占位符可以放置不同的内容。比如文本占位符可以放置文本，但不能放置图片。

通用占位符属性

占位符包含以下基本属性：

- x：占位符 x 坐标。
- y：占位符 y 坐标。
- width：占位符宽度。
- height：占位符高度。

占位符的定位相对于页面。

文本占位符

使用<iwb:TextPlaceholder>描述一个文本占位符，文本占位符包含以下属性：

- type：文本占位符的类别。
  值：[title|subtitle|body]
- content：文本占位符的内容。
  值：<svg:text>

一个布局最多只允许一个 title 和 subtitle，body 可以有多个。

下面是一个标题文本占位符的例子。

```
<iwb:TextPlaceholder type=”title”>
    <content>
      <svg:text font-family="Verdana" font-size=”45”>
          双击此处添加标题
       </svg:text>
    </content>
</iwb:TextPlaceholder>
```

图片占位符

使用<iwb:PicturePlaceholder>描述一个图片占位符，图片占位符包含以下属性：

- content：图片占位符的内容。
  值：<svg:image>
- tip：图片占位符提示信息。

下面是一个图片占位符的例子。

```
<iwb:PicturePlaceholder tip=”请在此处添加图片”>
    <content>
      <svg:image flip=”both”/>
    </content>
</iwb:PicturePlaceholder >
```

示例

下面是一个完整的标题和内容的布局例子：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
  xmlns:svg="http://www.w3.org/2000/svg"
  version=”1.0”>
   <iwb:Layout name=”标题和内容” type=”titleAndContent”>
     <iwb:Placeholders>
        <iwb:TextPlaceholder x=”120” y=”60” width=”1040” height=”120” type=”title”>
          <content>
            <svg:text font-family="Microsoft YaHei UI" font-size=”48”>
                双击此处添加标题
             </svg:text>
          </content>
        </ iwb:TextPlaceholder>
        <iwb:TextPlaceholder x=”120” y=”192” width=”1040” height=”468” type=”body”>
            <content>
              <svg:text font-family="Verdana" font-size=”24”>
                  双击此处添加内容
               </svg:text>
            </content>
```

```
            </ iwb:TextPlaceholder>
        </iwb:Placeholders>
    </iwb:Layout>
</iwb>
```

6.1.3 页面填充

在页面中，页面属性<svg:rect>中的 fillStyle 为填充类型。填充属性及值的说明详见表格2，以下为取值示例：

- 单色

  <svg:rect id=”page1” width="100%" height="100%" fill="#009300"/>

- 渐变

  < svg:rect id=”page1” width="100%" height="100%" fill="#40e0d0"/>
  <iwb:element ref=”page1” colorDes="#ffff00" fillStyle="1" fillStyleDetails="0"/>

  fillStyle 为 1，颜色从 fill 值渐变到 colorDes 的值，fillStyleDetails 表示渐变方向，从 0 开始依次表示为横向（从左到右），纵向（从上到下），主对角线（从左上到右下），副对角线（从右上到左下）。

- 纹理

  < svg:rect id=”page1” width="100%" height="100%" fill="#8b8b00"/>
  <iwb:element ref=”page1” colorDes="#00ff00" fillStyle="2" fillStyleDetails="12"/>

  fillStyle 为 2，fill 值为背景色，colorDes 值为条纹颜色，fillStyleDetails 表示纹理类型。

- 图片

  < svg:rect id=”page1” width="100%" height="100%" fill="#8b8b00"/>
  <iwb:element ref=”page1”
  colorDes="#00ff00" fillStyle="3"
  fillImagePath="images\Image.jpg" fillImageDetails="1"/>

  fillStyle 为 3，fill 值无效，colorDes 值无效，fillStyleDetails 表示填充类型（平铺，中间，拉伸），fillImagePath 为图片链接信息

表 5：页面填充属性及值的说明

取值	意义	附加属性	附加属性取值	附加属性意义	备注
	单色填充	fill	颜色值	填充的颜色	忽略附加属性值
1	渐变填充	fillStyleDetails	0	横向渐变	fill 属性仍需要使用
			1	纵向渐变	
			2	主对角线渐变	

			3	副对角线渐变	
		colorDes	颜色值	渐变的另一个颜色	
2	纹理填充	fillStyleDetails	[0,52]	纹理样式	fill 属性仍需要使用
		colorDes	颜色值	纹理的另一个颜色	
3	图像填充	fillImagePath	字符串	图片路径	忽略 fill 的值
		fillImageDetails	0	平铺	
			1	居中	
			2	拉伸	

xlink:href 为模板图片链接。

Image-width，图片宽度。

Image-height，图片高度。

6.1.4 页面视图

文档应在多种不同分辨率屏幕中呈现出一样效果，文档中的定位系统必须是相对的。本标准采用 SVG 规范中视窗（viewbox）概念来支持这一效果的实现。

在 SVG 规范中，视窗使用起始位置、宽度、高度这三个变量来定义，使用空格或者逗号分隔。（0,0）位置定义为左上角位置，x 值是指与页面左边缘的距离，y 值是指与页面上边缘的距离，例如一个起始位置为（0,0），宽度为 200，高度为 100 的视窗表达如下：

<svg:svg viewbox=”0 0 200 100”>

一个页面的顶部边缘是 y 位置为 0 的地方或者是 y 值小于 0 的视窗 y 值；一个页面的底部边缘是视窗 y 位置加上它的高度，或者是最底层元素的底部，如果该边缘超出了视窗。一个页面的左边边缘就是 x 位置为 0 的地方，或者是值小于 0 视窗的 x 值；一个页面的右边边缘是使用视窗 x 位置加上它的宽度，或者是最右边元素的右边，如果该边缘超出了视窗。

作为用户坐标系统，不一定直接关系到实际屏幕尺寸或宽高比，需要一种方法来记录这一点。因此，除了视窗可以包括创建/编辑的文件中宽和高的分辨率。如果省略这些信息，那么视窗需要匹配最后一次保存文件时屏幕的宽高比。虽然 SVG 规范中的宽度和高度可以用不同单位来定义（如英尺或者厘米），但在 IWB 中这些值使用像素来定义。

下面的两例效果是相同的：

<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>

```
<svg:svg width=”800px” height=”600px” viewbox=”0 0 1000 1000”>
```

下面的这个示例是错误的：

```
<svg:svg width=”5cm” height=”3cm” viewbox=”0 0 1000 1000”>
```

下面的示例是在屏幕中心显示一个矩形。这个文件首次保存于 800*600 分辨率的屏幕上。

```
<?xml version=”1.0”?>
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg=”http://www.w3.org/2000/svg”
    version=”1.0”>
  <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:rect x=”450” y=”450” width=”100” height=”100”>
  </svg:svg>
</iwb>
```

注意宽度和高度是指含滑动时的屏幕面积，而不是仅仅整个物理屏幕的大小。例如在屏幕的一角中包括一些边框和边框中显示幻灯片，那么宽度和高度就是边框内的值。比如，在一个 800*600 分辨率屏幕中，有一个覆盖屏幕每个边缘 10 像素的边，则设置如下：

```
<svg:svg width=”780” height=”580” viewbox=”0 0 1000 1000”>
```

在 IWB 视框中定义了一个页面的大小，显示为“幻灯片（Slide）”。于文档中的元素可以存在于视窗之外，从而在幻灯片（Slide）中是不可见的（这可作为滚动的示例，或者打开一个注释，其方式取决于阅读器的处理）。部分存在于视窗中的元素在幻灯片中也是部分可见。

在下面的这个示例中，幻灯片区域的中心呈现了 ID 为“rect1”的矩形框，紧随着呈现的是 ID 号为”rect2”的矩形上半部分。如果阅读器能够支持滚动，那么另外一半的”rect2”矩形也是可见的。矩形“rect3”不在幻灯片中显示，但是它仍然存在于幻灯片区域之下的文档中。

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
    version=”1.0”>
  <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:rect id=”rect1” x=”450” y=”450”
      width=”100” height=”100”/>
    <svg:rect id=”rect2” x=”450” y=”950”
      width=”100” height=”100”/>
    <svg:rect id=”rect3” x=”450” y=”1200”
      width=”100” height=”100”/>
  </svg:svg>
</iwb>
```

6.2 页面结构

6.2.1 组

组标签<svg:g>能够用来包含其它元素或组，以将这些元素或组当作一个整体来进行操作或属性设置。这种方式将一个页面中的对象组合成一种树形结构：

```
<svg:svg>
  <svg:g>
    <svg:rect></svg:rect>
    <svg:text></svg:text>
  </svg:g>
  <svg:rect></svg:rect>
  <svg:g>
    <svg:rect>
    <svg:g:>
      <svg:rect></svg:rect>
    </svg:g>
    </svg:g>
  <svg:g>
</svg:svg>
```

如果组中的某一个元素和组本身的属性冲突，则采用组的属性。如，设置元素列表并使用红色填充和黑色描绘勾线，以如下方式设置属性：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
  <svg:g fill=”red” stroke=”black”>
    <svg:rect x=”1” y=”1” fill=”green” stroke=”blue” width=”10” height=”10”/>
    <svg:circle cx=”10” fill=”blue” cy=”20” r=”5”/>
    <svg:ellipse cx=”10” stroke=”yellow” cy=”30” rx=”5” ry=”10”/>
  </svg:g>
</svg:svg>
</iwb>
```

下面这个例子展示了<svg:g>元素的嵌套，使用黄色线条画一个圆形，使用黑色线条来画椭圆。

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
```

```
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
  <svg:g fill=”red”>
    <svg:g stroke=”yellow”>
      <svg:circle cx=”10” cy=”20” r=”5”/>
    </svg:g>
    <svg:ellipse cx=”10” cy=”30” rx=”5” ry=”10” stroke=”black”/>
  </svg:g>
</svg:svg>
</iwb>
```

<svg:g>标签只存在于文件的 SVG 部分。

6.2.2 分层

元素分层或 Z 轴次序是按照对象在 SVG 部分出现的次序隐含定义在文件中的。第一个元素在 Z 轴次序的低端（显示在所有对象之后），在 Z 轴次序中再往下的元素都比第一个元素的次序高。

如下示例，绿色矩形出现在红色矩形的上面。

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
  <svg:rect id=”rect1” fill=”red” x=”450” y=”450”
            width=”100” height=”100”/>
  <svg:rect id=”rect2” fill=”green” x=”470” y=”470”
            width=”60” height=”60”/>
</svg:svg>
</iwb>
```

7. 元素概要

7.1 元素类型

IWB 中包括三种不同类型的元素：

- 图形
- 文本
- 媒体

元素是出现在页面的对象。每个元素位置与视窗的 x，y 坐标相关，使用相同的用户坐标。任何在页面边缘之外的元素被认为是隐形元素，它们将不会呈现在页面的任何位置。

一个元素可能通过变换（如旋转、平移）而使得其所在位置超出当前页面最上面和最左

边的位置，从而所有的部分都不能在本页面中显示。

这些元素主要是基于 SVG 规范进行定义的，也有些拓展的属性通过<iwb:element>标签来定义的，这些拓展标签使用一个引用来与 SVG 元素匹配。任何有 IWB 属性的元素必须在文件 SVG 部分设置一个 id。Id 设置必须考虑整个 IWB 包内对象元素的唯一性机制。这些<iwb:element>属性放置在文件结尾方向 SVG 部分之后。

举例如下，这个文件有两个矩形，一个矩形的 id 为“rect1”，它在<iwb:element>标签中 ref=“rect1”，这个矩形被设置成锁定的。第二个矩形没有也不需要一个 id，它采用默认方式，锁定的初始值默认值为“false”（参见附录 A"锁定元素拓展编码"）。

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:rect id=”rect1” fill=”red” x=”450” y=”450”
      width=”100” height=”100”/>
<svg:rect fill=”green” x=”470” y=”470”
      width=”60” height=”60”/>
</svg:svg>
<iwb:element ref=”rect1” locked=”true”/>
</iwb>
```

本章节所描述的属性使用于任何 IWB 元素。

7.2 两类坐标系

在文档页面中，存在两种坐标系，一是页面级别的页面坐标系，另一个是元素级别的元素坐标系，这两个坐标系下的坐标可以相互转换。页面坐标系以页面左上角为坐标原点，向右、向下为坐标轴的正方向。元素坐标系一般也以左上角为坐标原点，在没有旋转的情况，也是以向右、向下为坐标轴的正方向。在存在旋转的情况下，旋转中心默认为元素的几何中心，元素旋转后，元素坐标系也会相应的进行旋转。

元素坐标系的坐标原点在页面坐标系中的坐标用元素属性 x，y 来标识，例如，

```
<svg:rect x=”500” y=”500” />
```

上面表明这个矩形的元素坐标系的坐标原点在页面坐标系的（500，500）

一般地，元素的交互操作（平移、旋转、缩放、镜像等）都是在页面坐标系中进行元素坐标系的变换。

7.3 交互操作

元素的交互操作是通过 SVG 属性或拓展的 IWB 属性实现的。

7.3.1 旋转

旋转是通过 SVG 的变换属性完成的，例如一个矩形可以以如下方式旋转：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
    version=”1.0”>
  <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:rect id=”rect1” transform=”rotate(-30)”
      fill=”red” x=”450” y=”450”
      width=”100” height=”100” />
  </svg:svg>
</iwb>
```

旋转默认的围绕当前坐标的 0 点位置，但是旋转点也可以以如下方式实现：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
    version=”1.0”>
  <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:rect id=”rect1”
      transform=” rotate(-30, 500, 500)”
      fill=”red” x=”-50” y=”-50”
      width=”100” height=”100”/>
  </svg:svg>
</iwb>
```

以上这个例子中，矩形围绕中心点 500,500 旋转-30° 。

7.3.2 平移

平移是通过设置元素 transform 属性来标识的，例如：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
  <svg:rect transform=”translate(500,500)” />
</svg:svg>
</iwb>
```

上面这个例子中，将矩形向右向下都平移 500 像素。

平移也可以在容器标签<svg:g>中进行定义，示例如下：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
     xmlns:svg="http://www.w3.org/2000/svg"
     version=”1.0”>
   <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
     <svg:g transform=”translate(500,500) rotate(-30)”>
       <svg:rect id=”rect1” fill=”red” x=”-50” y=”-50”
         width=”100” height=”100”/>
     </svg:g>
     </svg:svg>
</iwb>
```

在所有上述情况下，平移均发生在坐标系统中，并且发生在元素被放置在页面呈现之前。

7.3.3 缩放和镜像

缩放是通过设置元素 SVG 属性来标识的，例如：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
     xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
   <svg:rect transform=”scale(1.5,1)” />
</svg:svg>
</iwb>
```

上面这个例子将矩形在 X 方向拉伸 1.5 倍，Y 方向不变

镜像是一种特殊的缩放，比如将一个图片进行上下镜像操作：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
     xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
   <svg:rect transform=”scale(1, -1)” />
</svg:svg>
</iwb>
```

元素旋转、平移、缩放和镜像等交互操作都可以在<svg:g>中使用，示例如下：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
```

```
    xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
  <svg:g transform=”translate(500,500) rotate(-30)”>
    <svg:rect id=”rect1” fill=”red” x=”-50” y=”-50”
              width=”100” height=”100”/>
  </svg:g>
</svg:svg>
</iwb>
```

该示例同样使矩形围绕中心点 500,500 旋转-30° 。

7.3.4 锁定元素（拓展）

锁定元素意味着这个元素在阅读器中不能被移动或者选择，但任何设置在该元素上面的链接仍可以被访问。

锁定是 IWB 的一个属性，它被添加到<iwb:element> tag 标签中。

锁定元素拓展编码见附录 A。

如下是一个有关锁定矩形的示例：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
     xmlns:svg="http://www.w3.org/2000/svg"
     version=”1.0”>
     <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
       <svg:rect id=”rect1” fill=”red” x=”450” y=”450”
       width=”100” height=”100” />
     </svg:svg>
<iwb:element ref=”rect1” locked=”true” />
```

7.3.5 拖动复制（拓展）

元素可设置为可拖动复制的，当使用者拖动元素时，该元素的副本就被创建并移动，而原始元素仍然停留在原处。

复制是 IWB 的一个属性，被添加在<iwb:element>标签中。

复制元素拓展编码见附录 A。

如下是一个有关复制矩形元素的示例：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
     xmlns:svg="http://www.w3.org/2000/svg"
```

```
        version=”1.0”>
        <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
        <svg:rect id=”rect1” fill=”red” x=”450” y=”450”
           width=”100” height=”100” />
        </svg:svg>
    <iwb:element ref=”rect1” replicate=”true” />
</iwb>
```

8. 元素：图形

IWB 文件存储这些 SVG 元素：

- 矩形
- 圆形
- 椭圆
- 直线
- 折线（也代表手绘）
- 多边形
- 扩展的图形类型，包括三角形、平行四边形、菱形、梯形、箭头、角、弧等。

这些图像分为两类，由一个或多个线条（直线或折线）组成的元素以及可被填充的元素。

8.1 线条属性

8.1.1 通用线条属性

线条属性适用于所有图形，并且可以使用如下 SVG 属性进行设置。除非另有说明，否则这些属性定义遵循 SVG1.0 规范。具体编码描述参见附录 A “通用线条属性编码”。

8.1.2 拓展的线条属性

拓展的 IWB 线条属性，添加在<iwb:element>标签中。具体编码描述参见附录 A”**拓展的线条属性编码”**。

示例，一条末端带有箭头线条是这样定义的：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
    version=”1.0”>
  <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:line id=”line1” x1="200" y1="200" x2="300" y2="300"
      stroke="red" stroke-width="5" />
  </svg:svg>
  <iwb:element ref=”line1” stroke-lineshape-end=”arrow” />
</iwb>
```

8.2 填充属性

8.2.1 通用填充属性

通用填充属性用下面的 SVG 属性进行设置，除非另有说明，否则这些属性遵循 SVG1.0 规范。具体参见附录 A “通用填充属性编码”。

8.2.2 图形填充属性

拓展的 IWB 图形填充属性添加在<iwb:element>标签中。具体参见附录 A “拓展的图形填充属性编码”。

8.3 拓展的图形属性

拓展的 IWB 图形属性添加在<iwb:element>标签中。具体参见附录 A 拓展的图形属性编码。

8.4 图形标签

8.4.1 矩形

在 SVG 规范中矩形由 x，y 坐标，宽度、高度来进行界定。IWB 格式不能识别圆角，而在 SVG 规范中定义是可以的。

```
<svg:rect id=”rect1” x="400" y="100" width="400" height="200"
        fill="yellow" stroke="navy" stroke-width="10" />
```

矩形也可以设置 revealer 属性，参见“8.3 拓展的图形属性”。

8.4.2 圆形

在 SVG 规范中圆形以 x，y 坐标作为其圆心，以半径 r 来界定。

```
<svg:circle id=”circ1” cx="600" cy="200" r="100"
        fill="red" stroke="blue" stroke-width="10" />
```

8.4.3 椭圆

在 SVG 规范中椭圆以 x，y 坐标作为其中心还有两个半径（x 半径和 y 半径）。

```
<svg:ellipse cx=”500” cy=”500” rx="250" ry="100"
```

fill="red" stroke="navy" stroke-width="10" />

8.4.4 线条

在 SVG 规范中线条根据 x,y 坐标来确定。

<svg:line x1="100" y1="300" x2="300" y2="100"

stroke="blue" stroke-width="5" />

可以设置线条的拓展属性“stroke-lineshape-start”，“stroke-lineshape-end”。参见“8.1.2 拓展的线条属性”。

8.4.5 折线

在 SVG 规范中折线由一组 x,y 坐标确定，包括移动和绘图的命令。

<svg:polyline points="50,375 150,375 150,325 250,325 250,375”

stroke="blue" stroke-width="10" />

可以设置折线的拓展属性 stroke-lineshape-start”、“stroke-lineshape-end”、“freehand”和“highlight”。使用手绘功能时折线与只有大量点组成的线不同，更接近普通的折线。参见“8.1.2 拓展的线条属性”。

8.4.6 多边形

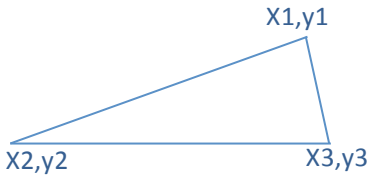
在 SVG 规范中多边形由一组 x,y 坐标确定，包括移动和绘图的命令。

<svg:polygon points="350,75 379,161 469,161 397,215"

fill="red" stroke="blue" stroke-width="10" />

8.4.7 三角形

通过扩展 SVG 多边形属性，在<iwb:element>标签定义 class 值为 triangle 的三角形，确定三角形三个顶点坐标顺序图示如下：

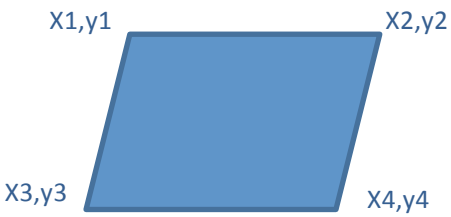


示例定义如下：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:polygon id="g1"
        points="340.00,207.00 229.00,400.00 562.00,400.00"
        fill="#96cd15" stroke="#000000" stroke-width="4.00"/>
</svg:svg>
<iwb:element ref=”g1”
        class=”triangle”
        bound="184,140 347,309"
        rotate-base="345.38,272.34" />
</iwb>
```

8.4.8 平行四边形

通过扩展 SVG 多边形属性，在<iwb:element>标签定义 class 值为 parallelogram 的平等四边形，确定平行四边行四个顶点坐标顺序图示如下：

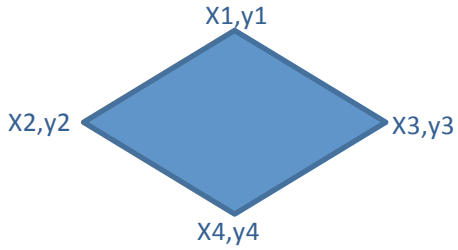


示例定义如下：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:polygon id="g1"
        points="249.00,205.00 391.00,205.00 533.00,358.00 391.00,358.00"
        fill="#96cd15" stroke="#000000" stroke-width="4.00"/>
</svg:svg>
<iwb:element ref=”g1”
        class=” parallelogram”
        bound="246,369 429,192"
        rotate-base="345.38,272.34" />
</iwb>
```

8.4.9 菱形

通过扩展 SVG 多边形属性，在<iwb:element>标签定义 class 值为 diamond 的菱形，确定菱行的四个顶点坐标顺序图示如下：

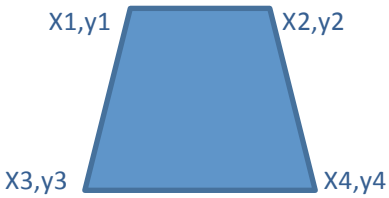


示例定义如下：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:polygon id="g1"
        points="540.00,256.00 637.00,377.00 540.00,498.00 443.00,377.00"
        fill="#96cd15" stroke="#000000" stroke-width="4.00"/>
</svg:svg>
<iwb:element ref=”g1”
        class=” diamond”
        bound="246,369 429,192"
        rotate-base="345.38,272.34" />
</iwb>
```

8.4.10 梯形

通过扩展 SVG 多边形属性，在<iwb:element>标签定义 class 值为 trapezia 的梯形，确定菱行的四个顶点坐标顺序图示如下：



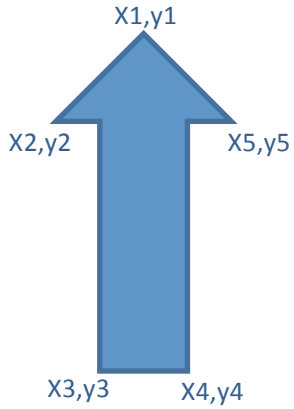
示例定义如下：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:polygon id="g1"
        points="321.00,309.00 429.00,309.00 538.00,513.00 321.00,513.00"
        fill="#96cd15" stroke="#000000" stroke-width="4.00"/>
</svg:svg>
```

```
<iwb:element ref=”g1”
        class=”trapezia”
        bound="246,369 429,192"
        rotate-base="345.38,272.34" />
</iwb>
```

8.4.11 箭头

通过扩展 SVG 多边形属性，在<iwb:element>标签定义 class 值为 Arrow 的箭头，确定箭头的五个顶点坐标顺序图示如下：

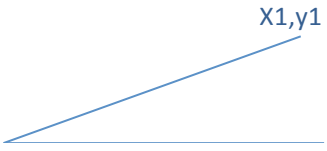


示例定义如下：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:polygon id="g1"
        points="413.00,399.00 646.00,399.00 646.00,613.00 413.00,613.00 413.00,399.00"
        fill="#96cd15" stroke="#000000" stroke-width="4.00"/>
</svg:svg>
<iwb:element ref=”g1”
        class=” upArrow”
        bound="246,369 429,192"
        rotate-base="345.38,272.34" />
</iwb>
```

8.4.12 角

通过扩展 SVG 多边形属性，在<iwb:element>标签定义 class 值为 angle 的角，确定角的三个顶点坐标如下图示：

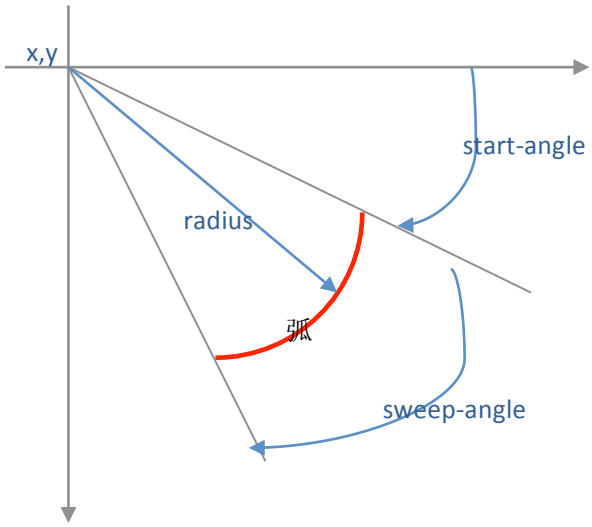


示例定义如下：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:polygon id="g1"
        points="397.51,470.00 334.00,580.00 625.00,642.00"
        fill="#96cd15" stroke="#000000" stroke-width="4.00"/>
</svg:svg>
<iwb:element ref=”g1”
        class=”angle”
        bound="246,369 429,192"
        rotate-base="345.38,272.34" />
</iwb>
```

8.4.13 弧

通过扩展 SVG 线条属性，在<iwb:element>标签定义 class 值为 arc 的弧，弧的大小通过左上角 x,y 和半径 radius 三个属性来确定，起始点通过 start-angle 和 sweep-angle 来确定，确定弧的各参数图示如下：



示例定义如下：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
xmlns:svg="http://www.w3.org/2000/svg"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
```

```
<svg:line id="g1"
        x1="571.00" y1="507.00"
        stroke="#000000" stroke-width="4.00"/>
</svg:svg>
<iwb:element ref=”g1”
        class=” arc”
        bound="465,419 211,193"
        rotate-base="570.50,515.50"
        radius=”100.00”
        start-angle=” 234.52”
        sweep-angle=” -235.07”/>
</iwb>
```

9．元素：文本

文本可表示为用 SVG <svg:text>标签元素表示的单行或是使用 SVGT1.2 <svg:textarea>标签元素标识的文本区域。<svg:textarea>标签元素用于标识需要自动换行的多行段落文本以及需要以某种方式合理布局的几行或段落文本。<svg:text>标签元素可用于需要更加精确的定位。

使用<svg:tspan>标签控制<svg:text>和<svg:textarea>内的元素以改变默认的文本格式。<svg:tspan>标签不属于 IWB 标签元素，它仅用于文本元素内部控制文本显示格式。

下面是一个将单个单词变成红色的示例：

```
<svg:text x=”10” y=”10” font-family="Verdana" font-size=”45”>
    This is some <svg:tspan fill=”red”>red</svg:tspan>text.
 </svg:text>
```

如下是一个设置了字体格式和大小的文本域（textarea），它封装了文本并且使用<svg:tbreak>标签强制换行。

```
<svg:textarea font-family="Verdana" font-size=”45” x=”10” y=”60” width=”50” height=”90”>
    This line will wrap into separate lines. This line breaks<svg:tbreak/> in two.
  </svg:textarea>
```

在<svg:text>或<svg:textarea>标签中的文本必须被正确地转义，这需要使用转义字符或者使用 XML CDATA 格式标识。如下给出了不正确的示例，“＆”和“<”字符应该进行转义：

```
  <svg:textarea x=”0” y=”0” width=”200” height=”100”>
    This is incorrect text with an ampersand “&” and
    this is incorrect text with an angle bracket “<”.
```

```
</svg:textarea>
```

下面使用两种方式来进行转义字符：

```
<svg:textarea x=”0” y=”0” width=”200” height=”100”>
    This is correct text with an ampersand “&amp;” and
    <![CDATA[this is also correct text with an angle bracket “<”.]]>
</svg:textarea>
```

9.1 通用的文本属性

使用下面 SVG 属性在<svg:text>, <svg:textarea>和<svg:tspan>内设置本文属性，除非另有说明，这些属性遵循 SVG1.0 建议。具体参见附录 A”通用的文本属性编码”。

使用<svg:a>标签能够在文本上添加链接，参见“13.1 链接”。

9.2 拓展的文本属性

在 IWB 中可使用如下的附加属性。这些附加属性有两种使用方式，<iwb:element>属性引用<svg:text>或<svg:textarea>标签，或者<iwb:tspan>引用<svg:tspan>。具体参见附录 A”**拓展的文本属性编码**”。

列表经常重新起一行，每一个新文本项目都在新的一行中开始。下面这个示例就是有三个项目的列表：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
    version=”1.0”>
  <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:textarea font-family="Verdana" font-size=”45”
      fill=”black” x=”10” y=”60”
      width=”100” height=”200”>
        This is a list:
      <svg:tspan id=”list1”>
        Item 1<svg:tbreak/>
        Item 2<svg:tbreak/>
        Item 3
      </svg:tspan>
```

```
    </svg:textarea>
  </svg:svg>
  <iwb:tspan ref=”list1” type=”list” list-style-type=”circle” />
</iwb>
```

9.3 文本标签

9.3.1 Text

文本标签所标识的字符采用的坐标方式如 SVG 基线方式，采用 x 和 y 坐标轴，但是与 SVG 不同，文本仅仅采用单坐标，只指定字符串中第一个字符的（x,y）位置，而不是多个字符的位置。并且这个坐标只能代表单一一行文本。

```
<svg:text x=”10” y=”10”>A line of text.</svg:text>
```

9.3.2 Textarea

在 SVG1.2 规范中，文本域使用用户坐标定义页面上起点坐标 x、y，以及特定宽度和高度的矩形。需要注意的是在不同阅读器中文本域文本可能不会呈现可选相同的显示效果，原因是部分文本因为自动换行而会超出窗口的高度。示例如下：

```
<svg:textarea x=”0” y=”0” width=”50” height=”90”>
    This line can wrap onto several lines.
  </svg:textarea>
```

9.3.3 Tbreak

如SVG规范中定义，该标签是用来定义一个新的行，并且只能用于文本域textarea标签中。示例如下：

```
<svg:textarea x=”0” y=”0” width=”50” height=”90”>
    This line is broken<svg:tbreak/> into two lines.
</svg:textarea>
```

9.3.4 Tspan

该标签必须在<svg:text>和<svg:textarea>这两个标签中使用。与 SVG 规范中定义不同的是，该标签在本标准中没有定位或者位置偏移量，旋转、x、y、dx 和 dy 这些属性都不能使

用。示例如下：

```
<svg:text x=”10” y=”10” font-family="Verdana" font-size=”45” fill=”black” >
   A <svg:tspan fill=”red”>line</svg:tspan> of <svg:tspan font-size=”60” font-
weight=”bold”>text</svg:tspan>.
</svg:text>
```

10. 元素：媒体

媒体包括图片、音频、视频和动画等，这些文件是嵌套在 ZIP 形式的 IWB 文件中，并放置在特定的文件夹。

图片、视频和动画本身就是元素，独立存在于文档中。音频文件不同，它不能单独存在，它必须链接到其他元素。

在同一个文档中，可以对同一个媒体文件有多个引用。

10.1 图片

图片被存放在名为“imedia/mages”的压缩文件内。嵌入的文件类型有：

- 联合图像专家小组（国际图像压缩标准），JPEG (*.jpeg; *.jpg)；
- 位图（*.bmp）；
- 动画和非动画图形交换格式，GIF (*.gif)；
- 窗口图元文件和增强型图元文件(*.wmf; *.emf)；
- 便携式网络图形，PNG (*.png)；
- 标签图像文件格式，TIFF (*.tif;*.tiff)。

图片元素放置在<svg:image>标签中，这些图片元素有一个 x，y 坐标位置以及高低和宽度值，图片会被伸展来填充到这个空间中。Xlink：href 表征了它代表的是那个图片，文件名中扩展命名必须保持完整，这样的文件格式容易被确认。

```
<svg:image xlink:href=”media/images/myimage.png” x=”10” y=”10”
    width=”100” height=”100”>
```

还可以增加 requiredExtensions 属性，扩展字符串类型如下：

```
requiredExtension=“http://www.becta.org.uk/iwb/wmf”
```

用三个字母的字符串来标识特定图片文件格式，之前示例有用“wmf”来定义其为“Windows Metafile”文件。如果<svg:switch>标签中不提供备选图片格式，那么就没有替代图片元素会被显示。参见“13.2 回退”。

拓展的图片属性编码参见附录 A。

10.2 视频

视频文件被存放在名为“media/video”的压缩文件内，嵌入的文件类型有：

- Mpeg (*.mpg;*.mpeg)

视频元素被放在 SVG1.2 标签<svg:video>中，这些视频元素有 x，y 坐标位置以及高度、宽度，视频会被拉伸来填充整个空间。Xlink：href 链接表征了它代表的那个视频。

<svg:video xlink:href=”media/video/myvideo.mpeg” x=”10” y=”10”

width=”100” height=”100”>

也可以增加 requiredExtensions 属性，扩展字符串类表征了视频类型：

requiredExtension=http://www.becta.org.uk/iwb/mpg

用三个字母的字符串来标识特定视频文件格式，之前示例有用“mpg”来定义为“MPEG video file”文件。如果<svg:switch>标签中不提供备选视频格式，那么就没有替代视频元素会被显示。参见“13.2 回退”。

10.3 动画

动画被存放在名为“media/flash”的压缩文件内。嵌入的文件类型有：

- Shockwave flash (*.swf)

动画元素被放在<svg:video>标签中，这些标签有 x，y 坐标位置以及高度、宽度，动画会被拉伸来填充整个空间。Xlink:ref 链接表征了它代表的那个动画。

<svg:video xlink:href=”flash/myflash.swf” x=”10” y=”10”

width=”100” height=”100”>

还可以在<svg:video> 标签中增加 requiredExtensions 属性，扩展字符串类表征了动画类型：

requiredExtension=http://www.becta.org.uk/iwb/swf

用三个字母的字符串来标识特定动画文件格式，之前示例用“swf”定义其为“Shockwave flash””文件。如果<svg:switch>标签不被用来提供备选视频格式，那么就没有替代动画元素会被显示。参见“13.2 回退”。

10.4 音频

音频被存放在名为“media\audio”的压缩文件内，嵌入的文件类型有：

- Mp3 (*.mp3)
- Wave (*.wav)

音频文件自身并非独立元素，需要链接到另一个元素。音频可以被链接到除<svg:video>之外的任何元素。参见“13.1 链接”。链接也可以参见“13.2 回退”。扩展字符串类表征了音频类型如下：

requiredExtension=http://www.becta.org.uk/iwb/mp3

用三个字母字符串来标识特定音频文件格式，之前示例有用“mp3”来定义其为“MP3 Audio file”文件。

11.元素：扩展对象

11.1 扩展对象元素类型描述

在交互式电子白板实践中，出现了各类复杂的交互对象，如学科对象，且这类对象伴随实践需求有一步拓展的需求。为支持这类多元化的、可发展的高级对象，本规范定义了高级对象，作为 BECTA IWB 文件格式中对象类型的拓展。拓展对象有统一文件结构，包含对象基本属性、对象特殊（扩展）属性、对象变化属性以及对象数据等内容。

对象基本属性包括显隐、锁定、无限克隆、链接等；

对象变换属性包括平移、缩放、旋转、错切等。

结构如下：

<iwbx:objectType id=""
type=""locked=""visible=""uclone=""width=""color=""opacity=""layer="">

<!—对象统一属性-->

<iwbx:element

</iwbx:element>

<!—对象特殊（扩展）属性-->

<iwbx:transform>

</iwbx:transform>

<!—对象变换属性-->

<iwbx:data>

</iwbx:data>

<!—对象数据-->

</iwbx:objectType >

11.2 常见扩展对象

11.2.1 画笔对象

<iwbx:stroke id="" type="curve"locked=""visible=""uclone=""width=""color="">

<iwb:element stroke-start="" stroke-end="" stroke-type=""> //线头、线尾、线型

</iwbx:element>

<iwbx:transform>

</iwbx:transform>

```
    <iwbx:data>
        <vertex points="x1,y1,x2,y2,……,xn,yn"/>
    </iwbx:data>
</iwbx:stroke>
```

表 6：画笔对象属性及取值说明

节点	子节点	二级子节点	属性	说明	数据类型	默认值	取值范围
Iwbx:stroke			id	对象标识	ulong		
			type	对象类型	string		硬笔，荧光笔，纹理笔，竹笔（排笔），软笔（毛笔）
			width	线宽	int	2	>=1
			color	线色	rgb	#ffffff	#000000-#ffffff
			opacity	透明度	float	1	[0,1]
			visible	显隐	bool	true	(true,false)
			locked	锁定	bool	false	(true,false)
			uclone	无限克隆	bool	false	(true,false)
			layer	所属层 id	int	0	
	Iwbx:element		stroke-type	线类型	string		
			stroke-start	线头	string		
			stroke-end	线尾	string		
	Iwbx:transform	matrix	tm	变换矩阵	matrix	(1 0 0 1 0 0)	
		translate	dx	x 平移分量	float	0	
			dy	y 平移分量	float	0	
		rotate	angle	旋转分量	float	0	[0,360]
		scale	scalex	x 缩放分量	float	1	
			scaley	y 缩放分量	float	1	
		shear	shearx	x 错切分量	float	0	
			sheary	y 错切分量	float	0	
	Iwbx:data	vertex	points	轨迹坐标	pointF		

线型、线头、线尾的取值范围如下：

```
Enum    stroke-type
{
    NoLine,                 //不显示线
```

```
    SolidLine,              //实线
    DashLine,               //虚线
    DottedLine,             //点线
    DashDotLine,            //点划线
    DashDotDotLine          //双点划线
};

Enum    stroke-start/ stroke-end
{
    NullDecorator = -1,         //无线头/尾属性
    NoDecorator,                //不显示线头/尾
    SquareDecorator,            //方形线头/尾
    CircleDecorator,            //圆形线头/尾
    DiamondDecorator,           //菱形线头/尾
    TriangleDecorator           //三角形线头/尾
};
```

11.2.2 表格对象

表格保存表格结构及单元格内容。表格的数据主要包括行、列、单元格信息。单元格内容可以为空，也可以是一个对象。

```
<iwbx:table id="" type="table" locked=""visible=""uclone="" layer="">
    <iwbx:element>
    </iwbx:element>
    <iwbx:transform>
    </iwbx:transform>
    <iwbx:data>
        <rect points="x1,y1,x2,y2"/>
        <!--网格信息-->
        <columns>
            <column width=""/>
            <!--width 列宽-->
        </columns>
        <rows>
            <row heigh="">
            <!--height 行高-->
                <cell rowSpan="" columnSpan="" horzMerge="" vertMerge="">
                <!--单元格属性 rowSpan y 方向上的合并的跨度 columnSpan x 方向
```

```
上合并的跨度, horzMerge 水平合并标签 vertMerge 竖直合并标签 -->
                <object id="">
                </object>
                <!--单元格内的对象-->
            </cell>
        </row>
        </rows>
    </iwbx:data>
</iwbx:table>
```

表 7：表格对象及属性取值说明

节点	子节点	二级子节点	属性	说明	数据类型	默认值	取值范围
Iwbx:table			id	对象标识	ulong		
			type	对象类型	string	table	
			visible	显隐	bool	true	(true,false)
			locked	锁定	bool	false	(true,false)
			uclone	无限克隆	bool	false	(true,false)
			layer	所属层 id	int	0	
	Iwbx:transform	matrix	tm	变换矩阵	matrix	(1 0 0 1 0 0)	
		translate	dx	x 平移分量	float		
			dy	y 平移分量	float		
		rotate	angle	旋转分量	float		
		scale	scalex	x 缩放分量	float		
			scaley	y 缩放分量	float		
		shear	shearx	x 错切分量	float		
			sheary	y 错切分量	float		
	Iwbx:data（见说明）						

Data 结点描述的是表格的数据，包括三部分内容：

表 7.1： 表格边界数据

节点	子节点	二级子节点	属性	说明	数据类型	默认值	取值范围
Rect			points	表格的边界矩形	string		

表 7.2：表格列数据

节点	子节点	二级子节点	属性	说明	数据类型	默认值	取值范围
columns							
	column		width	列宽	float	100	

表 7.3: 表格行数据，表格行数据的结点为 rows，它的子结点如下表

子节点	二级结点	三级节点	四级节点	属性	说明	数据类型	默认值	取值范围
Row	cell			height	行高	float	25	
				rowspan	行跨度	int	0	
				columnSpan	列跨度	int	0	
				horzmerge	水平合并标签	bool	false	(true,false)
				vertmerge	竖直合并标签	bool	false	(true,false)
		object		对象	对象的数据			保存对象的数据
		borders						
			left	width	线宽	int	1	
				dash	线型	int	1	
			top	width	线宽	int	1	
				dash	线型	int	1	
			right	width	线宽	int	1	
				dash	线型	int	1	
			bottom	width	线宽	int	1	
				dash	线型	int	1	

11.2.3 艺术字

```
<iwbx:artword uclone="" id="" layer="" visible="" type=" " locked="0">
    <iwbx:transform>
    </iwbx:transform>
    <iwbx:element>
      <artpath type=" "/>
      <artoutline color=" " type=" " width=""/>
      <artfill color="" type=" "/>
    </iwbx:element>
    <iwbx:data>
      <text uclone="" id="" layer="" visible="" type=" " locked="">
            <element modified="" layout="" type="" original="" level=""/>
            <transform>
            </transform>
            <data>
               <content text=" "/>
               <rect points=""/>
            </data>
      </text>
      <rect points=""/>
   </iwbx:data>
   </iwbx:artword>
```

表 8：艺术字及属性取值说明

节点	子节点	二级子节点	属性	说明	数据类型	默认值	取值范围
Iwbx:artword			id	对象标识	ulong		
			type	对象类型	string		直线，圆等 5 种类型
			width	线宽	int	2	>=1
			color	线色	rgb	#ffffff	#000000-#ffffff
			opacity	透明度	float	1	[0,1]
			visible	显隐	bool	true	(true,false)
			locked	锁定	bool	false	(true,false)
			uclone	无限克隆	bool	false	(true,false)
			layer	所属层 id	int	0	

	lwbx:element	art-path	type	艺术字类型	string		
		art- outline	type	轮廓填充类型	string		
			color		rgb	#ffffff	#000000-#ffffff
			width		int	1	
			href	图片路径	string		
		art-fill	type		string		
			color		rgb	#ffffff	#000000-#ffffff
			href	纹理路径	string		
	lwbx:transform	matrix	tm	变换矩阵	matrix	(1 0 0 1 0 0)	
		translate	dx	x 平移分量	float	0	
			dy	y 平移分量	float	0	
		rotate	angle	旋转分量	float	0	[0,360]
		scale	scalex	x 缩放分量	float	1	
			scaley	y 缩放分量	float	1	
		shear	shearx	x 错切分量	float	0	
			sheary	y 错切分量	float	0	
	lwbx:data	vertex	points	轨迹坐标	pointF		

艺术字类型属性值说明：

```
Enum      art-path
{
      Line,                  //直线
      Circle,                //圆
      Lean,                  //倾斜
      Cos,                   //余弦
      Sin,                   //正弦
};
```

艺术字轮廓及填充类型属性值说明：

```
Enum      art- outline/ art-fill      type
{
      SignalColor,           //单色
      Texture,               //纹理
      Image,                 //图片
};
```

12.颜色和背景

12.1 颜色

颜色和SVG协议表示形式一致，都基于CSS2格式定义（有关更多信息,请参见CSS2规范中4.3.6节的文件内容）。颜色用于填充和线条属性定义。有五种方式表示颜色。下面显示颜色的例子代表一个红色填充和绿色线条的矩形。颜色可以表示为一个名字,例如“name”。IWB标准使用SVG组命名的颜色。

```
<svg:rect x="0" y="0" width="10" height="10" fill="red" stroke="green" />
```

或者可以用一个“#”字符后跟3个表示颜色的十六进制字符,例如“# rgb”，示例如下：

```
<svg:rect x="0" y="0" width="10" height="10" fill="#f00" stroke="#0f0" />
```

或者可以用一个“#”字符后跟6个表示颜色的十六进制字符,例如“# rrggbb”，示例如下：

```
<svg:rect x="0" y="0" width="10" height="10" fill="#ff0000" stroke="#00ff00" />
```

或者可以用3个十进制值，该值在0到255内放，放在 “rgb()”括号内,即“rgb(红、绿、蓝)”，示例如下：

```
<svg:rect  x=" 0"  y=" 0"  width=" 10"  height=" 10"  fill=" rgb(255,0,0)"
stroke=" rgb(0,255,0)"  />
```

或者可以用三个百分比放在“rgb()”括号内,即“rgb(红%、绿%,蓝色%)”

```
<svg:rect x=" 0"  y=" 0"  width=" 10"  height=" 10"
fill=" rgb(100%,0%,0%)"  stroke=" rgb(0%,100%,0%)"  />
```

12.2 背景

可以设置一个背景颜色或者背景图片放置到交互式电子白板显示器上。

12.2.1 背景颜色

用<svg:rect>来设置背景颜色，其中必须包含视窗（viewBox）区域，它将利用属性background来标识背景。只有一个矩形可以作为背景,而且它必须首先被放置在页面元素列表中。下面的例子设置页面颜色为红色：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
xmlns:svg=”http://www.w3.org/2000/svg” version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:rect id=”rect1” fill=”red” x=”0” y=”0” width=”1000” height=”1000”/>
    <svg:rect fill=”white” x=”450” y=”450” width=”100” height=”100”/>
</svg:svg>
  <iwb:element ref=”rect1” background=”true”/>
</iwb>
```

拓展的背景颜色属性编码见附录A。

12.2.2 背景图片

用<svg:image>标签和<iwb:element>标签设置图片背景。当<svg:image>标签作为背景时，必须在每个页面元素列表之首出现,如果要出现矩形背景，则要紧随其后使用。

背景图片可以显示为一个图像或图像重复平铺至整个页面。

缺省情形下，背景图片< svg:image>标签标识的图片位置和大小属性相似于普通图像，然而，如果background-posture属性不再设值为“by-position”，那么图像将拉伸填补当前视窗(background-posture =“stretched-to-fill”)，缩放以适合在视窗以原始方式显示(background-posture = “scaled-to-fit”)或重复以覆盖整个页面(background-posture = “repeated”)。注意，除了重复的情形，图像位置属性应反映当前背景结构设置。

下面的例子展示了如何使用一个图像来填充视窗区域:

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg=”http://www.w3.org/2000/svg”
    xmlns:xlink="http://www.w3.org/1999/xlink"
    version=”1.0”>
    <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
    <svg:image id=”image1” xlink:href=”media/images/myimage.png”
        x=”0” y=”0” width=”1000” height=”1000” />
    </svg:svg>
    <iwb:element ref=”image1” background=”true” background-posture=”stretched-to-fill” />
</iwb>
```

如果一个图片被设为重复，图片位置会被忽略，大小被使用，图片将会重复在整个页面

而不仅仅在视窗区域。在这一情况下，x和y参数该被设为0。在如下的这个例子中，一个图片被重复呈现在背景中，图像将会在每100个单位用户横坐标和每100个单位用户纵坐标重复一次。

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
xmlns:svg=”http://www.w3.org/2000/svg”
xmlns:xlink="http://www.w3.org/1999/xlink"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
  <svg:image id=”image1” xlink:href=”medi/images/myimage.png”
  x=”0” y=”0” width=”100” height=”100” />
</svg:svg>
<iwb:element ref=”image1” background=”true” background-posture=”repeated” />
</iwb>
```

背景图像也可进行回退图像设置，查看章节“11.2 回退”。每个回退映像必须包含”背景”属性的设置。

拓展的背景图片属性编码见附录 A。

12.2.3 整合型背景

如下例子说明如何将背景颜色和背景图片放在一起使用。矩形必须在图片之前使用，以便于图片透明部分能显示设置的背景颜色。

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
xmlns:svg=http://www.w3.org/2000/svg
xmlns:xlink="http://www.w3.org/1999/xlink"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:rect id=”background-colour” fill=”red” x=”0” y=”0” width=”1000” height=”1000” />
<svg:image id=”background-image” xlink:href=”media/images/myimage.png”
x=”450” y=”450” width=”100” height=”100”/>
</svg:svg>
<iwb:element ref=”background-colour” background=”true”/>
<iwb:element ref=”background-image” background=”true”
```

background-posture=”repeated” />

</iwb>

背景元素必须不能包含在一个iwb:group列表中。

13. 扩展

13.1 链接

链接用<svg:a>标签和xlink:href属性实现，如果链接出现在文件中的某个位置，则命名空间xlink应该添加到文件：

```
<iwb xmlns:iwb="http://www.becta.org.uk/iwb"
     xmlns:svg="http://www.w3.org/2000/svg"
     xmlns:xlink="http://www.w3.org/1999/xlink"
     version=”1.0”>
</iwb>
```

链接可以设置在页面中的任何元素或一个元素的选定文本中。如果元素中包含链接，则元素将被< svg:a>标签包围。下面是矩形元素附加链接的例子：

```
<svg:a xlink:href=”http://www.becta.org.uk”>
     <svg:rect id=”rect1” x="4" y="1" width="40" height="20" />
</svg:a>
```

选定文本中的链接是通过<svg:a>标签包围标识的文本段实现的：

```
<svg:text x=”10” y=”10”>
  A <svg:a xlink:href=”http://www.becta.org.uk”>link</svg:a> in some text.
</svg:text>
```

多个元素可以在一个链接标识中：

```
<svg:a xlink:href=”http://www.becta.org.uk”>
<svg:rect x=”100” y=”100” width=”120” height=”40” />
<svg:text x="130" y="100">Open website</svg:text>
</svg:a>
```

一个链接的目标对象可以是内部对象，也可以是外部对象

内部对象包括:

- 当前页面的一个元素；
- 其他页面；
- 其他页面的一个元素；

- IWB 压缩文件中的媒体文件。

或外部对象包括:

- 一个外部文件；
- 一个网址。

13.1.1 内部对象链接

元素或页面的链接用对象ID加上“#”来指定。一个元素在同一页面上的链接导致包含该被链接元素所在页面部分在屏幕上可见；一个链接到另一个页面意味着该被链接页面的视窗（viewbox）在屏幕上是可见的；一个链接到另一个页面的某一个元素意味着包含该元素的页面的一部分在屏幕上是可见的。

举例，从一个长方形链接到另一个长方形的形式如下:

```
<svg:rect id=”rect1” x="0" y="0" width="40" height="20" />
<svg:a xlink:href=”#rect1”>
    <svg:rect id=”rect2” x="0" y="300" width="40" height="20" />
</svg:a>
```

13.1.2 内部文件链接

链接一个IWB压缩文件中的音频文件，需要将该音频的统一资源识别地址（URI）添加到xml文件中：

```
<svg:a xlink:href=”media/audio/mytune.wav”>
    <svg:text x="30" y="0">Play me<svg:text/>
</svg:a>
```

<svg:a>标签也可以赋予requiredExtensions属性。但如果< svg:switch>标签中没有标识替代的音频文件，则这一情形下没有声音播放。参见章节“13.2 回退”。

13.1.3 外部文件链接

存在于IWB压缩文件之外的文件可以链接到一个相对或绝对统一资源识别地址（URI）。为了区分内部元素中的外部文件链接，外部文件链接需要一个拓展属性，由< iwb:link>标签处理，该标签中包含 file=“internal”或 file=“external”设置。默认链接为内部。

如下示例，链接与IWB文件位于相同目录中的外部文件“help.txt”:

```
<svg:svg width=”800” height=”600” viewbox=”0 0 8000 6000”>
  <svg:text x="30" y="0">
    Open <svg:a id=”link1” xlink:href=”help.txt”>Help</svg:a>
```

```
    </svg:text>
  </svg:svg>
  <iwb:link ref=”link1” file=”external” />
```

拓展外部链接属性编码见附录 A。

13.1.4 网络链接

网址用如前面描述的相似方式链接：

```
<svg:a xlink:href=”http://www.becta.org.uk”>
    <svg:text x="30" y="0">The BECTA website</svg:text>
</svg:a>
```

13.2 回退

有些交互式电子白板阅读器可能无法显示所有类型的媒体，利用回退机制可以在该媒体位置使用一个替代者。这个将结合<svg:switch>标签和requiredExtensions属性来实现。Switch语句包围着可能需要替代的对象。被替换的对象必须包含一个requiredExtensions属性，该属性声明显示该对象所需要的扩展，如果该测试失败，则使用<svg:switch>标签之间的下一个元素，如果没有下一个对象则不显示。例如，一个文档可能包含一个窗口图元文件图片，但也包括适用于阅读器的便携式网络图形图片，而此时阅读器不能显示该窗口图元文件图片。该文件结构如下：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
    xmlns:svg="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    version=”1.0”>
    <svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
     <svg:switch>
       <svg:image x="0" y="0" width="10" height=”10”
         xlink:href=”media/images/myimage.wmf”xlink:type=”simple”
         requiredExtensions=”http://www.becta.org.uk/iwb/wmf”
       />
      <svg:image x="0" y="0" width="10" height=”10”
        xlink:href=”media/images/myimage.png” xlink:type=”simple” />
     </svg:switch>
```

```
    </svg:svg>
</iwb>
```

上面例子中，某阅读器可能不能显示PNG图像或者不支持显示某种格式的图像，在这种情况下，任何图形或文本元素可用于回退。对于在一个switch语句中添加的回退机制数量没有限制，当运行到一个阅读器能够显示的拥有requiredExtensions属性的元素时，switch语句将退出。下面的例子对于图元文件有两个可能的回退，如果不能显示图元文件，位图是可用的，如果位图也不能显示，则用一段文本代替图片显示。文件的结构如下：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
xmlns:svg="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:switch>
    <svg:image x="0" y="0" width="100" height=”100”
    xlink:href=”media/images/myimage.wmf”xlink:type=”simple”
    requiredExtensions=”http://www.becta.org.uk/iwb/wmf” />
    <svg:image x="0" y="0" width="10" height=”10”
      xlink:href=”media/images/myimage.bmp”xlink:type=”simple”
      requiredExtensions=”http://www.becta.org.uk/iwb/bmp” />
    <svg:text x=”10” y=”60” font-size=”20”>
      Sorry, unable to display image.
    </svg:text>
  </svg:switch>
</svg:svg>
</iwb>
```

<svg:g>标签在<svg:switch>标签中也很有用，它可以用来使多个元素替换另一个元素，在如下的例子中，如果不能显示图元文件则显示矩形和文本，如果没有<svg:g>标签，将只显示矩形：

```
<iwb xmlns:iwb=”http://www.becta.org.uk/iwb“
xmlns:svg="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" version=”1.0”>
<svg:svg width=”800” height=”600” viewbox=”0 0 1000 1000”>
<svg:switch>
```

```
        <svg:image x="0" y="0" width="50" height=”50”
        xlink:href=”media/image/myimage.wmf”xlink:type=”simple”
        requiredExtensions=”http://www.becta.org.uk/iwb/wmf”/>
    <svg:g>
        <svg:rect x=”0” y=”0” width=”50” height”50”/>
        <svg:text x=”10” y=”60” font-size=”20”/>
            Unable to display image.
        </svg:text>
    </svg:g>
</svg:switch>
</svg:svg>
</iwb>
```

附件 A：标准中的属性编码表

（1）锁定元素拓展编码

locked：一个元素是否被锁定；

值：　　[true|false]；

初始值：”false”；

引用：任何 SVG 元素。

（2）复制元素拓展编码

replicate：一个元素是否能够复制自身；

值：　　[true|false]；

初始值：”false”；

引用：任何 SVG 元素。

（3）通用线条属性编码：

stroke：图形轮廓的颜色，或者“none”表明没有线条轮廓。在本标准中与 SVG 规范不同，它没有“渲染服务器”；

stroke-opacity：线条的透明程度；

stroke-width：线条的宽度；

stroke-linecap：指定线段的末端如何绘制；

stroke-linejoin：说明如何绘制交点；

stroke-dasharray：用于创建虚线，逗号分隔破折号和间隙长度的列表。

（4）拓展的线条属性编码

stroke-lineshape-start：该属性设置在线条起始端的形状，如箭头形状；

值：　　[none | arrow | circle | line]；

初始值：　　”none”；

引用：　　svg:line 或者 svg:polyline 标签。

stroke-lineshape-end：该属性设置在线条末尾端的形状，如箭头形状；

值：　　[none | arrow | circle | line]；

初始值：　　”none”；

引用：　　svg:line 或者 svg:polyline 标签。

freehand：　　该属性这说明折线是手绘而不是单一的线条；

值：　　[true | false]；

初始值：　　”false”；

引用：　　svg:polyline 标签。

highlight：　该属性说明折线被用来高亮显示屏幕的部分内容。如果不做任何设置，该属性隐形设置了 stroke-opacity 属性的值；

值：　[true | false]；

初始值：　”true”；

引用：　svg:polyline 标签。

（5）通用填充属性编码

fill：该属性表示在图形内部填充的颜色，如果没有颜色则表示为空。和SVG不同，没有“渲染服务器”。

fill-opacity：该属性表示填充的透明程度。

要注意与 SVG 规范的不同，在 IWB 中没有填充规则的属性，所有填充使用 evenodd 算法进行计算。

（6）拓展的图形填充属性编码：

ColorDes：　该属性表示填充第二颜色，某些填充效果需要其与 fill 属性配合使用，不需要时会忽略此颜色设定特。

值：　[rgb 值]；

初始值：　”none”；

引用：　所有图标标签。

fillStyle：　该属性表示填充样式，模板需要此取值为 3。

值：　[none|1|2|3]；

1: 渐变填充

2: 纹理填充

3: 图像填充

初始值：　”none”；

引用：　所有图标标签。

fillStyleDetails：　该属性需要配合 fillStyle 使用，表示具体填充样式，如渐变样式下需要该属性说明是横向渐变还是竖向激变。

值：　[none|0|1|2|3|52]；

0: 横向渐变填充

1: 竖向渐变填充

2: 主对角渐变

3: 副对角渐变

52: 纹理填充

初始值： ”none”；

引用： 所有图标标签。

fillImagePath： 该属性表示图片路径，模板填充时忽略此属性。

值： [图片相对路径]；

初始值： ”none”；

引用： 所有图标标签。

fillImageDetails： 该属性表示图片填充具体样式，模板需要此取值为 2。

值： [none|0|1|2]；

0: 平铺

1: 居中

2: 拉伸

初始值： ”none”；

引用： 所有图标标签。

xlink:href： 该属性表示模板图片链接地址。

值： [模板图片链接地址]；

初始值： ”none”；

引用： svg:rect 标签。

（7）拓展的图形属性编码：

class： 该属性设定特定形状的多边形，包括三角形 triangle、平行四边行 parallelogram、菱形 diamond、梯形 trapezia、箭头 upArrow、角 angle、弧 arc。

值： [none | triangle | parallelogram | diamond | trapezia | upArrow | angle | arc]；

初始值： ”none”；

引用： svg: polygon 或 svg:line 标签。

bound： 该属性记录图形对象的边界大小，用于对象的局部刷新，单位像素；

值： [none | width hight]；

初始值： ”none”；

引用： 所有图形标签。

rotate-base： 该属性记录图形对象的中心点坐标，用于基于图形对象中心点的操作，单位像素；

值： [none | x y]；

初始值： ”false”；

引用： 所有图形标签。

radius： 该属性为弧的半径，单位像素；

值： [浮点数]；

初始值： ‘0px’；

引用： svg:line 标签。

start-angle： 该属性为弧的起始角度，从 x 轴到弧线的起始点沿顺时针方向度量的角度，单位度；

值： [浮点数]；

初始值： ‘0’；

引用： svg:line 标签。

sweep-angle： 该属性为弧的划过角度，从 startAngle 参数到弧线的结束点沿顺时针方向度量的角，单位度；

值： [浮点数]；

初始值： ‘0’；

引用： svg:line 标签。

（8）通用的文本属性编码：

fill：　　文本颜色，与 SVG 不同，没有“绘画服务器”；

font-family：　　字体使用的格式；

font-size:　　使用的字号。这在用户坐标中（在视窗中设置）定义的，这样在不同阅读器中字体显示的大小相同；

font-stretch:　　水平伸展文字；

font-style:　　字体样式，例如斜体；

font-weight:　　字体的粗细程度。

下面这些属性是 SVG Tiny1.2 中的属性：

text-align:　　文本如何对齐，例如居中或者左对齐；

值：　　[start | end | center | justify]；

初始值：　　” start”；

引用：　　svg:textarea 或者 svg:textarea 中的 svg:tspan 标签。

（9）拓展的文本属性编码：

background-fill：　字符背景颜色；

值：　　[<colour> | none]

初始值：　　”none”

引用：　　svg:text, svg:textarea 或者 svg:tspan 标签。

highlight-fill：　字符背景颜色专门用于突出显示文本；

值：　　[<colour> | none]；

初始值：　　”none”；

引用：　　svg:text, svg:textarea 或者 svg:tspan 标签。

type—tspan：　显示类型，其类型或者是普通或列表方式之一；

值：　　[normal | list]；

初始值：　　” normal”；

引用：　　svg:textarea 或者 svg:textarea 中的 svg:tspan 标签。

list-style-type：　列表标签的外观，如圆形或者低透明度；

值：　　CSS 列表“list-style-type”；

初始值：　　”circle”；

引用：　　svg:textarea 或者在 svg:textarea 中的 svg:tspan 标签。

list-style-type-fill： 列表标签的颜色。

值： [<colour>]

初始值： ”black”

引用： svg:textarea 或者 svg:textarea 中的 svg:tspan 标签。

editable： 文本是否可编辑（如果元素的锁定属性 locked=”true”，那么文本不能被编辑）

值： [true | false]

初始值： ”true”

引用： svg:text or svg:textarea 标签。

（10）拓展的图片属性编码：

IWB 定义了更多的属性，添加在<iwb:element>标签中。

flip: 原始图片的影像；

值： [none | horizontal | vertical | both]；

初始值： ” none”。

引用： svg:image 标签。

（11）拓展的背景颜色属性编码：

<iwb:element>标签能够承载附加的属性，

background： 声明该元素是否为背景；

值： [true|false]；

初始值： “false”；

引用： svg:rect或者svg:image标签。

（12）拓展的背景图片属性编码

<iwb:element>标签可以设置下面这些拓展的属性，

background-posture：声明图片如何呈现在视窗背景中，以调整比例到布满视窗；

属性值： [scaled-to-fit | stretched-to-fill | repeated | by-position]；

初始值： ”by-positon”；

引用： svg:image 标签。

（13）拓展外部链接属性编码

通过<iwb:link>标签进行外部链接属性定义，该属性定义如下，

file：　　声明链接是内部文件还是外部文件的属性名称；

属性值：　[internal | external]；

初始值：　“internal”；

引用：　　svg:a标签。

附件 B：交互式电子白板常见学科对象建议

在本标准草案中，学科对象没有作为基本内容考虑，作为可由本标准扩展机制作用的范围。

1. 常见的学科对象列表

(1) 数学

函数方程式、极坐标方程式、参数方程式；

线段、几何线段、箭头、角、弧、圆、椭圆、三边形、四边形、多边形、平面；

球体、球冠、圆锥体、立椎体、圆柱体、圆台体、长方体、正方体、棱台提、二面体、五棱锥体、五棱台体等。

(2) 物理

力学：螺旋桨、气缸、力、弹簧秤、钩码、滑轮、平面、斜面、传送带、小球、小车、木块、直尺、游标卡尺、圆槽、凹槽等；

电学：开关、按钮、电灯、安培表、伏特表、灵敏电流计、自定义表、滑动变阻器；

磁学：导线、线圈、铁芯、条形磁铁、U 型磁铁、磁场、电场、电荷、正负电子、小磁针；

光学：凸透镜、凹透镜、半凸透镜、半凹透镜、光学支架、蜡烛。

(3) 化学

符号：苯环功能、化学键、双线桥；

器皿：烧杯、试管、曲颈瓶、烧杯、蒸馏烧瓶、水槽、集气瓶、锥形瓶、容量瓶、试剂瓶、引流管；

器械：天平、砝码、酒精灯、火焰、温度计、量筒、铁架台、试管夹、三角架、滴定管夹、镊子、药匙、燃烧匙、木块、塞子、玻璃塞、玻璃棒、酸式滴定管、碱式滴定管、漏斗、长颈漏斗、分液漏斗、球形分液漏斗、启普发生器。

(4) 语文

笔画、拼音。

(5) 英语

音标。

2. 学科对象的封装

对学科对象采用拓展对象标签<iwbx:discipline></iwbx:discipline>进行标识，其语法结构按照第 11 章进行。如下是化学学科扩展对象的一个实例（烧杯）。其余学科对象定义与其类似。

```
<iwbx:discipline id ="0" type="chemistry:shaobei" visible="true" uclone="false" locked="false">
        <iwbx:element stroke-type="">
          <property name="显示液体" value="true"/>
          <property name="液体颜色" value="0,0,255"/>
          <property name="开口方向" value="Left"/>
        </iwbx:element>
        <iwbx:transform>
          <matrix  tm="1,0,1,0,0,1"/>
          <translate dx="" dy=""/>
          <rotated angle=""/>
          <scale scalex="" scaley=""/>
          <shear shearx="" sheary=""/>
        </iwbx:transform>
        <iwbx:data>
            <rect points="x1,y1,x2,y2" controlpoints="x1,y1,x2,y2" />
        </iwbx:data>
    </iwbx:discipline>
```

其中 iwbx:discipline 节点中的属性是与其他对象共有的属性，例如：显隐、锁定、无限克隆等

子节点 iwbx:element 中记录了学科对象的特有属性及属性值；

子节点 iwbx:transform 中记录了学科对象的变换属性；

子节点 iwbx:data 中记录了学科对象的位置信息。

附件 C：对 BECTA IWB v1.1.1 规范拓展的说明

1. 为提高系统对 IWB 文档的读、写速度，本草案中摈弃了 BECTA IWB 文档规范中依托 <svg:pageset> 和<svg:page>标签的文档布局结构，定义了新的文档结构；
2. 采用了经过实践验证的分页面存储结构，并且定义了系统化的文档布局，具体参见章节 6；
3. 对 BECTA IWB 图形标签进行了拓展，从该 BECTA IWB v1.1.1 版本的 6 个图形标签类型拓展到 13 个。新拓展的 7 个是在对国内交互式电子白板行业企业充分调查的基础上最终确定的；（2015.11 调查）
4. 在元素类型上，提出了拓展对象类型结构，以适应我国交互式电子白板行业现有实践中交互对象的复杂性、多元化的特征，定义了扩展对象格式；并依据该格式新扩展了 3 个扩展对象，包括画笔对象、表格对象、艺术字对象。新增的 3 个拓展对象是在对国内交互式电子白板行业企业充分调查的基础上最终确定的；（2015.11 调查）
5. 对交互式电子白板常见学科对象方面，结合对行业企业充分调查，本次标准提出了常见学科对象类型，给出了学科对象依据扩展对象格式的定义实例。方便标准的可拓展性和可维护性；
6. 其它一系类的拓展标签、属性描述。对不能够由 SVG 图形元素对象定义的复杂交互对象定义，采用扩展对象的格式进行定义，扩展对象格式定义以<<iwbx:objectType>为根节点标签进行定义的，具体参见章节 11。其它在 BECTA IWB v1.1.1 版本基础上的扩展属性参见章节 3。