



中华人民共和国国家标准

GB/T 28170.2—2021/ISO/IEC 19775-2:2015

信息技术 计算机图形和图像处理 可扩展三维组件(X3D) 第2部分:场景访问接口(SAI)

Information technology—Computer graphics and image processing—
Extensible 3D (X3D)—Part 2: Scene access interface(SAI)

(ISO/IEC 19775-2:2015 , IDT)

2021-03-09 发布

2021-10-01 实施

国家市场监督管理总局 发布
国家标准化管理委员会

目 次

前言	I
引言	II
1 范围	1
2 规范性引用文件	1
3 术语和定义	2
4 概念	3
4.1 介绍	3
4.2 综述	4
4.3 绑定和协议依赖性	5
4.4 接口构建	5
4.5 事件	8
4.6 标识符	9
4.7 相关的 URLs	9
4.8 执行模型	10
5 数据类型引用	16
5.1 介绍和主题	16
5.2 数据类型定义	18
5.3 错误类型	21
5.4 事件类型	23
6 服务引用	24
6.1 介绍和主题	24
6.2 创建连接	28
6.3 浏览器服务	29
6.4 执行上下文服务	40
6.5 场景服务	47
6.6 节点服务	49
6.7 域服务	50
6.8 路由服务	53
6.9 Proto 类型服务	54
6.10 配置服务	56
6.11 脚本内容提供的服务	58
6.12 矩阵服务	61
7 一致性和最低支持要求	62
7.1 介绍和主题	62
7.2 一致性	64
7.3 最低要求	64
附录 A (规范性附录) VRML 脚本向后兼容性	71

前 言

GB/T 28170《信息技术 计算机图形和图像处理 可扩展三维组件(X3D)》分为2个部分:

——第1部分:体系结构和基础组件;

——第2部分:场景访问接口(SAI)。

本部分为GB/T 28170的第2部分。

本部分按照GB/T 1.1—2009给出的规则起草。

本部分使用翻译法等同采用ISO/IEC 19775-2:2015《信息技术 计算机图形和图像处理 可扩展三维组件(X3D) 第2部分:场景访问接口(SAI)》。

与本部分中规范性引用的国际文件有一致性对应关系的我国文件如下:

——GB/T 13000—2010 信息技术 通用多八位编码字符集(UCS)(ISO/IEC 10646:2003, IDT)。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本部分由全国信息技术标准化技术委员会(SAC/TC 28)提出并归口。

本部分起草单位:中国电子技术标准化研究院、深圳赛西信息技术有限公司、合肥高维数据技术有限公司、浙江大学、北京理工大学、北京电影学院、南昌虚拟现实检测技术有限公司、中国移动通信有限公司研究院、湖南财政经济学院、深圳市掌网科技股份有限公司、福建网龙计算机网络信息技术有限公司、歌尔股份有限公司、北京邮电大学、深圳市商汤科技有限公司、三星电子(中国)研发中心。

本部分主要起草人:马珊珊、王聪、田辉、王涌天、焦廉洁、刘越、宋维涛、王锐、杨本植、赵辰羽、孙光、周航军、胡翔、孙其民、刘德建、林志宏、林琛、钟开华、严小天、乔秀全、商彦磊、蒋慧、张志海。

引 言

X3D 是一种用于描述交互式 3D 对象和场景的文件格式和相关的访问服务。X3D 旨在用于互联网、内联网和本地客户端系统。X3D 还旨在成为集成 3D 图形和多媒体的通用交换格式。X3D 可用于各种应用领域,例如工程和科学可视化、多媒体演示、娱乐和教育标题、网页和共享虚拟世界。

本部分定义了用于与 X3D 场景交互的场景访问接口,X3D 场景来自场景内或外部程序。

信息技术 计算机图形和图像处理

可扩展三维组件(X3D)

第2部分:场景访问接口(SAI)

1 范围

GB/T 28170 的本部分规定了可通过浏览器获得的一组标准的服务,以便用户可在场景图运行时访问,相关访问设计用来支持与场景图的交互和修改。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件,仅注日期的版本适用于本文件。凡是不注日期的引用文件,其最新版本(包括所有的修改单)适用于本文件。

GB/T 28170.1—2011 信息技术 计算机图形和图像处理 可扩展三维组件(X3D) 第1部分:体系结构和基础组件(ISO/IEC 19775-1:2004, IDT)

ISO/IEC 10646 信息技术 通用多八位编码字符集(UCS)[Information technology—Universal multiple—Octet coded character set (UCS)]

ISO/IEC 14496-1:2010 信息技术 音视频对象编码 第1部分:系统(Information technology—Coding of audio-visual objects—Part 1:Systems)

ISO/IEC 14772-1:1997 信息技术 计算机图形与图像处理 虚拟现实建模语言 第1部分:功能规范和 UTF-8 编码(Information technology—Computer graphics and image processing—The virtual reality modeling language—Part 1: Functional specification and UTF-8 encoding)

ISO/IEC 14772-2:2004 信息技术 计算机图形与图像处理 虚拟现实建模语言 第2部分:外部著作接口[Information technology—Computer graphics and image processing—The virtual reality modeling language (VRML)—Part 2:External authoring interface(EAI)]

ISO/IEC 19776-1:2015 信息技术 计算机图形、图像处理和环境数据表示 扩充 3D(X3D)编码 第1部分:扩充置标语言(XML)编码[Information technology—Computer graphics, image processing and environmental data representation—Extensible 3D (X3D) encodings—Part 1: Extensible markup language (XML) encoding]

ISO/IEC 19776-2:2015 信息技术 计算机图形、图像处理和环境数据表示 扩充 3D(X3D)编码 第2部分:典型 VRML(虚拟现实建模语言)编码[Information technology—Computer graphics, image processing and environmental data representation—Extensible 3D(X3D) encodings—Part 2: Classic VRML encoding]

ISO/IEC 19776-3:2015 信息技术 计算机图形、图像处理和环境数据表示 扩充 3D(X3D)编码 第3部分:压缩二进制编码[Information technology—Computer graphics, image processing and environmental data representation—Extensible 3D (X3D) encodings—Part 3:Compressed binary encoding]

ISO/IEC 19777-1 信息技术 计算机图形、图像处理和环境数据表示 扩充 3D(X3D)语言编码 第1部分:ECMAScript[Information technology—Computer graphics and image processing—Exten-

sible 3D (X3D) language bindings—Part 1:ECMAScript]

ISO/IEC 19777-2 信息技术 计算机图形、图像处理和环境数据表示 扩充 3D(X3D)语言编码

第 2 部分:Java[Information technology—Computer graphics and image processing—Extensible 3D (X3D) language bindings—Part 2:Java)

IETF RFC 4248 telnet URI 方案,互联网标准跟踪协议(The telnet URI Scheme,Internet standards track protocol)

W3C 文档对象模型(DOM) 2 级核心规范 1.0 版[W3C Document object model(DOM) level 2 core specification Version 1.0]

3 术语和定义

GB/T 28170.1—2011 界定的以及下列术语和定义适用于本文件。

3.1

错误 errors

服务不成功而终止的原因。

3.2

文件 file

存储在物理媒体上的或以数据流形式或以计算机程序内数据形式存在的相关数据的集合。

3.3

仅初始化字段 initializeOnly field

定义为节点定义的一部分的字段,其值仅在定义节点时指定。

3.4

可输入字段 input-capable field

仅输入字段或输入输出字段。

3.5

仅输入字段 inputOnly field

定义为节点定义的一部分的字段,该字段仅可接收事件。

3.6

输入输出字段 inputOutput field

定义为节点定义的一部分的字段,该字段既可接收事件也可发送事件。

3.7

此刻 now

用户所感知的当前时间。

3.8

可输出字段 output-capable field

输入输出字段或仅输出字段。

3.9

仅输出字段 outputOnly field

定义为节点定义的一部分的字段,该字段仅可发送事件。

3.10

参数 parameters

传递给服务的值。

3.11

公共接口 **public interface**

本部分中赋予节点类型的正式定义。

3.12

回复 **returns**

由服务调用返回的值。

3.13

运行时命名范围 **run-time name scope**

X3D 文件中定义的名称适用和可见的范围。

4 概念

4.1 介绍

本章描述了本部分中的主要概念,包括描述浏览器的各种组件及如何能实现与浏览器的交互。本章不定义单个交互。这些描述见第 6 章服务引用。

表 4.1 列出了本章的主题。

表 4.1 主题

<ul style="list-style-type: none">● 4.1 介绍● 4.2 综述<ul style="list-style-type: none">○ 4.2.1 概述○ 4.2.2 与 VRML 脚本的兼容性● 4.3 绑定和协议依赖性● 4.4 接口构建<ul style="list-style-type: none">○ 4.4.1 综述○ 4.4.2 用户代码○ 4.4.3 包含节点○ 4.4.4 应用程序○ 4.4.5 会话○ 4.4.6 浏览器○ 4.4.7 场景○ 4.4.8 节点和节点生命周期○ 4.4.9 字段○ 4.4.10 执行上下文● 4.5 事件<ul style="list-style-type: none">○ 4.5.1 概念○ 4.5.2 浏览器内部○ 4.5.3 浏览器到外部应用程序<ul style="list-style-type: none">■ 4.5.3.1 综述■ 4.5.3.2 初始化■ 4.5.3.3 关闭■ 4.5.3.4 无可用的 URL■ 4.5.3.5 连接中断

表 4.1 (续)

<ul style="list-style-type: none">● 4.6 标识符● 4.7 相关的 URLs● 4.8 执行模型<ul style="list-style-type: none">○ 4.8.1 交互类型综述○ 4.8.2 事件模型评估顺序○ 4.8.3 内部交互<ul style="list-style-type: none">■ 4.8.3.1 允许交互■ 4.8.3.2 浏览器交互■ 4.8.3.3 事件响应■ 4.8.3.4 更新场景图■ 4.8.3.5 异步活动■ 4.8.3.6 监控场景图中的变化■ 4.8.3.7 用户代码生命周期<ul style="list-style-type: none">■ 4.8.3.7.1 综述■ 4.8.3.7.2 设置■ 4.8.3.7.3 实现■ 4.8.3.7.4 清理■ 4.8.3.8 输入输出字段和包含节点■ 4.8.3.9 执行环境 and 安全性○ 4.8.4 外部交互<ul style="list-style-type: none">■ 4.8.4.1 允许交互■ 4.8.4.2 浏览器交互■ 4.8.4.3 更新场景图■ 4.8.4.4 监控场景图中的变化■ 4.8.4.5 同步多个应用程序○ 4.8.5 服务保证● 表 4.1 主题● 表 4.2 在节点生命周期期间允许的字段访问的能力● 表 4.3 允许的活动节点的字段交互

4.2 综述

4.2.1 概述

当用户希望通过使用客户代码与 X3D 场景图交互时,无论该用户是作为在 GB/T 28170.1—2011 中脚本组件 29 中定义的一个脚本节点,或者是来自于外部应用程序,均应使用本部分中定义的场景创作接口(SAI)。该接口是一个操作 X3D 场景图的协议,而不是场景图本身的部分。

本部分旨在对所有能由外部应用程序经该接口执行的动作提供一种语言中立的表示。对特定语言的绑定在 ISO/IEC 19777-1 和 ISO/IEC 19777-2 中进行定义。SAI 形成了一个通用的接口,可从外部应用程序或从场景图内部通过脚本节点来操作浏览器和场景图。但是,为外部应用程序编写的代码不能立即作为脚本使用。这两种环境对于场景图的访问和交互有着完全不同的要求和能力。本部分提供了一个单一的、统一的编程接口和取决于代码发现自身所处环境条件的约束。

从概念上来说,SAI 允许五种类型的 X3D 场景访问:

- 访问浏览器的功能；
- 接收浏览器操作的通知，如无法访问的网址、启动和关闭；
- 将事件发送到场景中可输入的节点字段；
- 读取场景中可输出节点字段发送的最后一个值；
- 当事件改变场景中节点字段的值时可获得通知。

4.2.2 与 ISO/IEC 14772 的兼容性

如果 X3D 浏览器希望符合 ISO/IEC 14772-1:1997 的要求，则浏览器应支持在附录 A 中“VRML 脚本向后兼容性”中定义的事件模型和语义，并支持 GB/T 28170.1—2011 所定义的功能。这种支持只应在处理符合 ISO/IEC 14772-1:1997 的文件时才会被使用。

如果 X3D 浏览器希望符合 ISO/IEC 14772-2:2004 的要求，则浏览器应使用下列规则来判断内容的有效性：

- a) 如果用户代码通过 EAI 访问浏览器，则只应加载 ISO/IEC 14772-2:2004 中定义的 VRML 文件。如果用户代码正在通过 EAI 访问浏览器，那么加载符合本部分要求的文件应是一个错误；
- b) 如果用户代码通过本部分中定义的外部交互所提供的的能力访问浏览器，则只应加载 GB/T 28170.1—2011 中定义的 X3D 文件。如果用户代码正在通过 SAI 访问浏览器，那么加载符合 ISO/IEC 14772-1:1997 的文件应是一个错误。

4.3 绑定和协议依赖性

实现依赖是根据语言绑定或本部分中定义的服务的协议编码进行定义的。如果某服务被定义为实现依赖的话，那么假如要实现时，每个绑定和编码都需要指定如何实现该服务。

对这些服务的绑定和编码可在该规范中定义它们自己的实现依赖部分。

4.4 接口构建

4.4.1 综述

在 X3D 浏览器中有四个可使用 SAI 服务进行访问的主要数据集合：浏览器、关于当前加载场景的元数据、场景图中的节点和节点中的字段。定义和规范是以服务的形式来构造的。X3D 浏览器公开了一组允许外部应用程序与之交互的服务。为了描述这些概念，定义了一些术语。

4.4.2 用户代码

任何使用本部分定义的服务的代码都被认为是用户代码。用户代码可存在于场景图中或浏览器外。只应使用本部分提供的服务，而非浏览器实现特定的服务。此外，这些服务既不是为浏览器编写本地节点扩展而设计的，也不是针对其使用的。浏览器可提供自己的专用程序接口来实现不属于本部分的本地扩展。如果代码使用专用扩展，就本部分而言，不应将其视为用户代码。

4.4.3 包含节点

包含节点是场景图中负责表示希望参与内部交互的用户代码的节点（见 4.8.3 内部交互）。用户代码的生命周期应完全由包含节点来控制。当包含节点处于激活状态时，用户代码也为激活状态。当包含节点被删除且不再被视为如 GB/T 28170.1—2011 中 4.4.2.5 对象生命周期定义的激活状态时，该节点所包含的用户代码应被终止。用户代码不能通过保持对其包含节点的引用来延长包含节点的生命周期。浏览器是包含节点何时不再处于激活状态的最终仲裁者。

不要求在包含节点及其用户代码之间一一对应。语言绑定可允许一个用户代码实例在包含节点的多个实例之间共享。

4.4.4 应用程序

应用程序是外部过程,不是 X3D 浏览器的隐式部分。该应用程序应浏览器的请求,以某种形式连接至 X3D 浏览器。应用程序既不属于 GB/T 28170.1—2011 中图 1 中定义的 X3D 浏览器的组成部分,也不属于 GB/T 28170.1—2011 中 4.4.8.3 执行模型定义的执行模型的组成部分。应用程序可处于另一台来自 X3D 浏览器的机器上。应用程序可负责创建一个嵌入在该应用程序中的,或者附着在已经运行的浏览器实例上的新浏览器实例(例如,网页上的小程序)。

4.4.5 会话

会话定义用户代码和 X3D 浏览器之间单个连接的生命周期。一个浏览器可同时服务多个会话(例如,一个场景中的多个脚本节点)。

一个应用程序可包含与多个浏览器的多个单独的会话,但一个脚本节点不应如此。外部应用程序和多个 X3D 浏览器之间同时有多个会话是允许的。然而,独立的实现可能会对此类多个同时会话设置一些限制。

会话不是本部分的可实现部分。它纯粹是一个概念机制,用户可通过它来请求服务。它可能在浏览器和外部应用程序之间建立任何连接之前就存在,或者与浏览器连接请求同时建立。

4.4.6 浏览器

浏览器是一个有效 X3D 场景图的基本封装机制(这是时间推进的而不是存储在磁盘上的文件形式的)。由于它包含了整个场景图,它还提供了一组最小的核心能力,用于在一个粗糙的级别上动态操纵场景图。此场景图可能包含最多一个活跃 LayerSet 节点,该节点应是此场景图的根节点(见 GB/T 28170.1—2011 中 4.3.2 根节点)。场景中包含的使用内联节点(或通过其他方式)导入的任何其他 LayerSet 节点都应忽略。

用户可能会在他们的机器上同时运行很多 X3D 浏览器。因此,每个浏览器应由会话中的唯一标识符来表示。对单个浏览器实例的多个请求,该标识符应完全相同。这是为了确保访问一个浏览器实例的两个应用程序可以一种明确的方式共享信息。

任何需要使用浏览器功能的操作都应使用浏览器标识符标识服务请求。

4.4.7 场景

场景表示单个 X3D 场景图和所有关于该场景图的信息。这个场景是一个 X3D 文件的程序化等价物。它可包含节点、路由、原型声明、导入和导出以及一个有效的 X3D 文件可能包含的所有信息。在任何给定的时间,浏览器可包含一个或多个场景。例如,一个场景使用 Inline 节点来包含另一个场景。

在浏览器中,场景不必是处于激活或者运行状态的。用户可构造一个未附加在浏览器实例上的新的场景,然后程序化地填充诸如节点和路由之类的信息。这个场景可随后被直接传递到一个通用程序,例如用于发布源文件的打印程序,或者用于替换浏览器中的当前场景。

4.4.8 节点和节点生命周期

节点是在场景图中与元素交互的最小单元。节点与 GB/T 28170.1—2011 中定义的 X3D 节点相同。一个节点可作为一个单元从场景图中删除、存储,然后在之后的某个时间重新插入到相同会话中的另一个位置,而不会产生不利影响。

每个节点都由唯一的标识符来定义,该标识符对于会话是唯一的。也就是说,单个浏览器可能同时

服务多个应用程序,因此所有节点标识符在该会话的周期里都是唯一且不变的。这允许两个外部应用程序可明确地在它们之间共享数据,并且仍然允许其他外部应用程序使用该共享数据向浏览器发出服务请求。

SAI 中的大部分操作都是从获取节点的引用开始的。获取节点的引用有很多种方法。可使用 DEF 构造来命名引用,并使用适当的服务来获取,或者可从某个任意父节点遍历场景图来获得。一旦获得了引用,就可访问该节点的所有字段,但是不一定要读取或写入,包括仅初始化字段。由于输入输出字段可能是隐式读取和写入的,因此可使用字段名或者 set_ 和 _changed 修饰符来访问这些字段。

节点引用伴随着节点的生命周期,在此期间具备不同的功能。生命周期可表示为:

- a) 创建:节点首先由浏览器内部实例化,所有字段值都设置为默认值;
- b) 设置:根据需要,字段值从默认值更改为定义的初始值;
- c) 实现:该节点参与场景图或/和脚本编写;
- d) 处置:该节点不再是场景图的一部分,并且在脚本中不再保留对它的引用。

读取和写入的字段访问取决于节点的状态。该状态和功能在表 4.2 中定义。

表 4.2 在节点生命周期期间允许的字段访问功能

字段类型	创建	设置	实现	处置
仅初始化字段	无	可读/可写	无	无
仅输入字段	无	无	可写	无
仅输出字段	无	无	可读	无
输入输出字段	无	可读/可写	可读/可写	无

从设置到实现状态的转换可以是隐式的或显式的。存在服务请求,以便用户可正式发布设置已经完成的通知,从而节点可完成所需的任何内部构造。由于用户的操作,转换也可能是隐式的。当用户对节点引用做了除设置字段值之外的其他任何操作,此节点将转换到实现状态。

示例:用户创建了一个 Box 节点,设置了字段大小,创建一个 Shape 节点,然后立即将 Box 添加到 Shape 节点。此操作将使 Box 节点的状态变为 Realized 状态,并在设置状态中保留 Shape 节点。

节点标识符也用来表示一个空节点。一个空的 SFNode 或 MFNode 字段值都用空值来表示。对于空的 MFNode 字段,可用字段值应为空。

4.4.9 字段

各个字段都在节点内定义。虽然无法直接操作节点,但字段是直接操作单个属性的方法,如表 4.2 所示。

不可能将节点属性当做独立于节点本身的实体直接加以操纵(即字段不存在于包含节点之外)。

字段类型和对于各个字段的访问是由 GB/T 28170.1—2011 定义的。每个字段指定一个字段标识符。但该标识符并非唯一的,并且需要一个节点标识符加上字段标识符来指定与之交互的特定字段。当访问一个字段时,用户应获得整个字段的标识符。通过服务定义,将所有字段隐式处理为可读和可写。使用标志指示该字段能否在该时间点上及时读取或写入(依赖于表 4.2 中所述的生命周期的节点状态)。此状态可能会随着节点生命周期的推进时间而改变。例如,一个处于非激活状态的节点的仅初始化字段可能是可写的,但是一旦该节点插入到场景图中,它就不再是可写的。这种规定有助于希望围绕第三方浏览器程序化构建场景的制作工具和用户。

在会话过程中任何时候,字段都是可读并可写的。用户代码可注册和注销,以接收当字段值发生变化时的通知。在注册过程中,用户代码可提供一个标记,该标记将随事件的数据值一起返回。此标记可

被用户代码用于在事件不是隐式唯一的情况下唯一标识此事件。标记不需要与服务请求一起传递,并且可留作应用程序接口内部实现的一部分保存。

被应用程序引用到的节点的所有可输出字段都能被读取。读取值是该字段发送的最后一个值,或者如果没有发送任何事件,则是该字段类型的默认值。所读取的数据特定于该字段的字段类型,并被格式化,以适合所用的语言或协议。

4.4.10 执行上下文

执行上下文是运行时语义,等价于在 GB/T 28170.1—2011 中 4.4.7 运行时命名范围所描述的名称范围。它提供了一种包含内部交互代码并为其加防火墙的方式,这种方式与命名空间在文件格式中提供的限制类似。例如,当一个 Proto 实例中的脚本添加一个 ROUTE 时,该路由被添加到该 proto 内,而不是添加到一般场景中。

场景是执行上下文的派生类型。当内部交互请求当前执行上下文时,返回一个场景对象。然后,用户代码可检查该执行上下文是否为完整场景的一个实例,并且通过将其转化为对应的派生类型(如果可用时)。

4.5 事件

4.5.1 概念

任何围绕 X3D 场景图的短暂数据使用都是通过事件来完成的。应用程序可注册从 X3D 场景图中接收事件,并可发起新的事件。事件被认为是短暂的,只有在特定的行为发生时才产生。事件不应被存储,也不应被交付给那些在事件发生时并未对该事件表示兴趣的各方。

示例: 一个在加载后才与浏览器连接的应用程序不应交付 Initialize 事件。

4.5.2 浏览器内部

应用程序可将值写入字段或从字段中读取值。在 X3D 浏览器内部表示并标记时间戳之前,此值不会成为事件。浏览器与应用程序的边界是事件停止的位置。事件不能从 X3D 浏览器外部存在;也就是说,应用程序无法插入事件级联中。可向应用程序通知事件,发起新事件,但是当通知事件时,在浏览器内阻止当前时间戳事件级联的处理时,不能处理和传递事件。允许记录事件以用于分析目的。

在创建级联之前不会生成事件。如果内部交互直接写入另一个节点的可输出字段,则不会生成任何事件,因此不会形成事件级联的一部分。如果内部交互写入包含节点的可输入字段,则如果该字段是来自其他地方的 ROUTE 的主题,则形成具有写入值的事件。

4.5.3 浏览器到外部应用程序

4.5.3.1 综述

浏览器可通过自己的一系列事件直接与外部应用程序通信。这些事件用于指示浏览器的状态或一些异步问题。可用的事件的数量和类型将依赖于实现。至少,遵循本部分的所有实现都应提供以下事件。

应保证从浏览器到外部应用程序的事件传递。

4.5.3.2 初始化

初始化事件用于指示浏览器已经在初始化过程中运行了一个场景(浏览器已经加载了世界,就在它即将发布其第一个与时间相关的事件之前)。此时,节点标识符将从场景的 getNode 服务中获得(见 6.4.7 getNode)。

初始化事件应立即在浏览器中生成并交付给应用程序。该事件被认为是异步的。也就是说,事件的交付(以及任何实现依赖确认方案)都不应延迟浏览器启动执行模型评估的时间。

4.5.3.3 关闭

关闭事件用于表示浏览器将停止运行当前场景。这可能发生在许多不同的条件下:

- a) 场景正在被替换(见 6.3.12 replaceWorld 和 6.3.14 loadURL);
- b) 浏览器本身正在退出;
- c) 客户端应用程序已经处理了它与浏览器的连接(见 6.3.25 dispose)。

关闭事件应立即在浏览器中生成并交付给应用程序。该事件被认为是异步的。也就是说,事件的交付(以及任何依赖实现的确认方案)都不会延迟浏览器停止执行模型评估和关闭浏览器资源,除非需要确保事件交付应用程序。

4.5.3.4 无可用的 URL

SAI_BROWSER_URL_ERROR 事件用于通知应用程序,浏览器无法在 loadURL 服务的一个异步调用中加载任何 URL/URNs(见 6.3.14 loadURL)。这表示无法从此调用中指定的任何 URL 加载或调用有效内容。其他调用可能涉及其他异步负载的调用,比如 replaceWorld(见 6.3.12 replaceWorld) 和 createX3DFromString 和 createX3DFromStream(见 6.3.16 createX3DFromString, 6.3.17 createX3DFromStream)也可使用此事件来指示 GB/T 28170.1—2011 中 9.3.2 X3DUrlObject 中指定的任何 X3DUrlNode 的加载问题,例如,内联,纹理和 EXTERNPROTO,尽管不是必需的。

4.5.3.5 连接中断

连接丢失的错误被用来通知应用程序,底层实现已经丢失了浏览器和应用程序之间的连接,从而导致服务请求不能被授予。例如,一个 TCP 网络连接超时或其他类似的问题。

如果实现执行了一些自动的重新连接尝试,那么实现可能会延迟发送连接丢失的事件。它只会在被认为不可能连接到浏览器的时候发送。在此事件生成后,不需要强制重新建立连接,也不需要尝试任何形式的自动重新连接功能。

4.6 标识符

标识符的构成依赖于实现。在某些情况下,将节点标识符表示为整个节点(包括所有字段信息)可能更有效。然后在本地节点上进行字段信息的请求。在其他实现中,标识符可能只是一个简单的整数。确保标识符唯一是浏览器的重要职责,这样应用程序可在合理的环境约束下共享数据。对该环境的约束可指定为在附加的附录中定义的独立实现的一部分。

使用不同服务实现的两个应用程序在浏览器环境之外交换数据是不合理的。

4.7 相关的 URLs

GB/T 28170.1—2011 中的 9.2.2 相对 URLs 指定了在浏览器环境中处理相对 URL 的规则。声明文件应被定义为浏览器中当前加载领域的基本 URL。当前加载的领域可通过 getWorldURL 服务的请求获得(见 6.4.6 getWorldURL)。在浏览器还没有加载 X3D 文件的情况下,应将基本文档目录作为浏览器的当前工作目录。当浏览器是 web 页面的一部分时,当前的工作目录应被视为嵌入 web 浏览器的页面的基本 URL。

当生成嵌套的相对 URLs(例如包含对脚本文件的引用的 EXTERNPROTO)时,根据 GB/T 28170.1—2011 解析顶层 RURL 基础。

4.8 执行模型

4.8.1 交互类型综述

因为 SAI 实现了外部应用程序和脚本的编程接口的角色,所以执行模型能在这两种情况下工作。尽管两种情况下的 API 调用都是相同的,对每个服务请求的运行时评估可能不同。

示例: 在脚本中服务字段更改通知将停止当前事件级联,但对于外部应用程序则不应如此。

该规范定义了服务可参与的两种交互类型:内部(即脚本)和外部(应用程序)。

4.8.2 事件模型评估顺序

脚本代码允许用户通过自定义行为修改场景图。为了达到一致的效果,在 GB/T 28170.1—2011 中 4.4.8.3 执行模型中定义的评估顺序被扩展到包含脚本所允许的服务交互。当提供内部交互代码时,应按以下顺序来评估事件模型的所有方面。

- 基于当前绑定视点的位置和方向来更新相机;
- 评估传感器输入;
- 从缓冲区收集外部输入并传递到节点;
- 调用场景中所有活动脚本节点的 prepareEvents 脚本服务;
- 评估路线;
- 在已收到 set_url 事件或正在从场景中删除的脚本上调用关闭服务;
- 为任何从场景中移除的传感器生成最终事件;
- 添加/删除调用 dynamicRouteHandling 服务请求所需的任何路由,如 6.4.17 dynamicRouteHandling 中所定义从步骤 6 中的任何脚本执行;
- 调用 eventsprocessing 脚本服务,用于发送在步骤 6 中生成的事件脚本;
- 为新加载的内部交互代码调用初始化服务;
- 如果从步骤 5 到 10 生成任何事件,请转到步骤 5 并继续直到完成当前事件级联。

如果内部交互向本部分定义的对象注册任何形式的回调或侦听功能,那么这些回调将在更改发生时进行。

示例: 脚本 A 中的用户代码在初始化服务处理期间在仅输出字段上发布事件,脚本 B 中的另一段用户代码则在 eventOut 上有侦听服务。在脚本 A 中的用户代码退出并将控制权返回给浏览器核心后,将立即触发脚本 B 中的侦听。

4.8.3 内部交互

4.8.3.1 允许交互

内部交互是发生于用户代码和包含节点构成 X3D 场景图的一部分时。这些节点服从并参与事件级联评估。内部交互可能发生在事件级联的中间,作为接收事件的直接结果,并可能产生一个或多个输出事件作为响应。这些事件将在当前级联中继续。当输出事件由异步脚本评估产生,或由其他与当前事件级联处理不直接相关的过程生成,如在 GB/T 28170.1—2011 中 29.2.4 EventsProcessed() 所定义的,一个全新的事件级联将会启动。

在 GB/T 28170.1—2011 中 29.3.1 X3DScriptNode 中指定的 X3DScriptNode 类型,其指定了一个包含节点,但将来也可定义其他节点类型。浏览器将只允许通过用户代码的内部交互,此用户代码是从 X3D 脚本节点或其他将来定义的包含节点类型中引用的。如果用户代码是从任何其他节点类型引用的,则应将代码视为外部交互并相应地采取行动。

内部交互还允许与其他节点的字段直接交互,或者在不参与事件级联的情况下进行一些浏览器操作。此操作应只允许依赖于包含 X3DScriptNode 节点的 directOutput 字段设置的值。关于什么时候

此行为是允许的定义,见 GB/T 28170.1—2011 中第 29 章脚本组件中的定义。

因为用户代码被认为是包含在包含节点内的,所以节点字段的含义会与其标准状态相反。一个仅输入型字段是可读不可写的;一个仅输出型字段是可写不可读的;仅初始化型字段和输入输出型字段都是可读又可写的。将此与非表 4.3 中包含节点的外部节点进行对比,任何其他内置或本机扩展节点可能拥有的访问类型与之类似。

表 4.3 允许的活动节点的字段交互

访问类型	包含节点	外部节点
仅初始化	可读/可写	无访问
仅输入	可读	可写
输入输出	可读/可写	可读/可写
仅输出	可写	可读

4.8.3.2 浏览器交互

浏览器允许内部交互。由于代码位于当前场景内,因此只允许有限的完整浏览器服务集。服务条款概述了哪些服务可用于内部操作以及哪些服务是不受限制的。

在内部操作的初始化阶段,应为脚本提供适合其交互的浏览器的引用。这应在脚本的整个生命周期内保持不变,而其包含的节点被认为是实时的。

4.8.3.3 事件响应

内部交互旨在响应事件,提供一些处理,并可选地也为包含节点生成新的输出。此外,它还可提供异步输出,它并不通过使用一个或多个线程对应任何输入。产生输出在 4.8.3.4 更新场景图中有被考虑。4.8.3.4 描述了由事件级联引起的输入的响应问题,该事件级联将事件发送到包含节点的一个或多个可输出字段。

为了响应事件,用户代码注册对包含节点的适当字段的兴趣。除了包含节点的仅输出型字段外,可对所有字段访问类型注册兴趣。一旦包含节点完成了它的初始化阶段,其中包含节点的某个字段接收到一个事件,用户代码就会通过特定于语言的绑定提供的通知机制通知它。浏览器可选择立即通知用户代码,或者将多个事件组合在一起,并提供一个延迟通知。在这两种情况下,浏览器应确保在该时间戳中交付所有事件,而不是在以后的某个时间。在决定是否推迟或立即通知时,浏览器还应遵循 GB/T 28170.1—2011 中 29.4.1 脚本规定的包含节点的 mustEvaluate 字段指示。

在通知之后,浏览器将不再处理包含节点的事件级联中的任何事件,直到处理已经从用户代码返回(尽管这允许其他事件级联继续同时处理)。在用户代码放弃控制之前,不应将写入其他节点字段和包含节点可输入字段的值传递给目标节点。

注意:如果存在并行事件级联被评估(例如在多 CPU 机器上运行的浏览器,并行事件级联可被评估,并导致两个级联将事件同时传递给包含的节点),那么这并不排除浏览器实现同时向用户代码交付多个事件。用户代码的作者应知道并对这种情况采取适当的预防措施。

当浏览器确定级联或级联完成时,浏览器可能会调用在 GB/T 28170.1—2011 中 29.2.4 EventProcessed()所定义的包含节点的 EventProcessed()方法。用户代码也被告知这种情况,此时用户代码可选择执行额外的评估并生成更多的输出。内部通知用户代码无法确定何时当前渲染帧完成及下一帧开始。

4.8.3.4 更新场景图

用户代码除了接收输入外,还可选择生成输出。对于内部交互,不要求用户代码生成响应输入的输出。在任何包含节点被认为是活动的时候,用户代码可异步地生成输出或直接写入到其他节点,在 4.8.3.3 事件响应和 4.8.2.5 异步活动中给出了一些限制。

用户代码有两种影响场景图的选项;它可写入包含节点的可输出字段,并使该值服从通常的事件级联,或者直接写入包含节点和用户代码已经具有引用的另一个节点的可输入字段。用户代码可随时根据表 4.2 中定义的访问规则写入包含节点的字段。与其他节点的内部交互应遵从包含节点的直接输出字段定义的规则,如 GB/T 28170.1—2011 中 29.2.6 带直接输出的脚本中表明的。

有两种特殊情况,用户代码不允许对包含节点的字段进行更改。X3DScriptNode 抽象类型 mustEvaluate 和 directOutput 的两个字段被认为是特殊的,用户代码在运行时不允许修改这些值。用户代码可读取这些值。如果包含的节点也来自 X3DUrlNode 抽象类型,它可能会选择更改自己的 URL 字段,从而用新的用户代码替换当前的用户代码。

脚本可自我修改,因此,如果包含节点也是从 X3DUrlNode 抽象类型(见 GB/T 28170.1—2011 中 9.3.2 X3DUrlObject)派生的,它可选择更改自己的 URL 字段,从而用新用户代码替换当前用户代码。

4.8.3.5 异步活动

一些语言中的用户代码可被允许使用异步执行线程进行操作。这允许用户代码直接运行而不需要通过浏览器的调用。这种情况的一个典型用途是监视网络连接,以便对场景图进行更改。这需要使用内部交互,该内部交互不是作为由用户代码接收到的字段更改的直接结果而创建的。

内部交互只允许在 4.8.3.7 用户代码生命周期指定的时间内进行。如果用户代码试图在任何其他时间内进行内部交互,那么浏览器将产生一个错误。prepareEvents 服务(见 6.11.4.1 prepareEvents),如果由用户代码定义,允许用户代码在已知的时间点对场景图进行完全异步的更改,而不需要使用 TimeSensor 或其他节点对脚本进行计时。这与 eventsProcessed 服务(见 6.11.4.2 eventsProcessed)形成对照,它只在包含节点必须处理字段更改之后才被调用。

此外,当用户代码注册为一个侦听器服务时,与此相关的回调被认为是异步操作。在此期间操作的用户代码,不对场景图进行修改。

4.8.3.6 监控场景图中的变化

字段的服务定义允许用户代码在其他节点的字段输出中注册兴趣。内部交互代码不允许在字段更改信息中注册兴趣。如果用户代码想要被字段告知更改信息,它将使用现有的路由机制和适当的场景服务,在感兴趣的字段和包含节点的可输入字段之间添加一条路由。

内部用户代码仅可在其他节点的可输出字段中添加注册兴趣,只有当包含节点的 directOutput 字段设置为 TRUE 时,才允许这样做。如果该值设置为 FALSE 而允许用户代码在输出中注册兴趣的话,这将是错误的。

4.8.3.7 用户代码生命周期

4.8.3.7.1 综述

用户代码的生命周期可能由于许多原因而更短,例如从远程站点获取代码的下载时间或其他用户代码更改包含节点的 URL 以替换当前用户代码。但是,用户代码的生命周期遵循包含节点的基本原则。它有相同的阶段,也经历过类似的转变。

在 GB/T 28170.1—2011 的 4.4.2.5 对象生命周期中,定义了包含节点的生命周期。

4.8.3.7.2 设置

假设在包含节点初始化和用户代码将通过初始化阶段时,会有一些延迟,但会很小。虽然包含节点可能已经完成了初始化阶段并处于运行阶段,但是用户代码可能还没有开始或者可能正在处理它的初始化。

在初始化阶段,内部交互代码将获得其生命周期余下阶段所需的所有信息。初始化阶段的第一步是用户代码的实例化。在实例化过程中,用户代码没有关于其包含环境或包含节点的信息。在此期间,用户代码可选择设置所需的任何资源,如线程、网络连接或包含环境的任何其他允许操作(见 4.8.3.9 执行环境 and 安全性)。

在实例化之后,用户代码会接收到在内部交互环境中运行所需的资源通知。给出包含浏览器的标识符,节点的字段列表(不包括任何特殊字段)和包含节点的标识符(需要用户代码可添加和删除到其包含节点的路由)。在此期间,用户代码除打印消息外,不应发出任何服务请求。如果是,浏览器就会产生错误。

作为初始化阶段的最后一步,用户代码应调用它的初始化服务(参见 6.11.3.3 initialize)。此时,用户代码可自由地使用内部交互可用的所有服务。

注:对于用户代码来说,这是一个很好的时机来注册包含节点字段的更改通知,或者执行一些外部任务,比如绑定一个特定的视图。

4.8.3.7.3 实现

在运行时阶段,用户代码服从此条款在接收、发送和监视事件以及 X3D 执行模型方面的要求。

用户代码在什么时间可对场景图进行修改是受到限制的。它应只在响应来自浏览器的提示时允许进行修改。许可时间应定义为:

- a) 在事件级联发生前处理服务请求-准备事件-期间(见 6.11.4.1 prepareEvents);
- b) 响应对包含节点的可输入字段的更改通知;和
- c) 在后事件处理服务请求期间经由 eventsProcessed 服务请求(参见 6.11.4.2 eventsProcessed)。

在不允许异步交互的情况下,用户代码在任何时候发出服务请求都是错误的。在第 6 章服务引用中指定了每个服务请求定义是否允许用户代码进行异步请求。

4.8.3.7.4 清理

当节点不再活动或其他用户代码的操作导致用户代码从包含节点中删除时,用户代码将进入关闭阶段(例如,更改脚本节点的 URL 以指向新的可执行内容)。

对关闭阶段的更改通知应通过用户代码中 shutdown 服务请求的调用(见 6.11.5.1 shutdown)。在此阶段,用户代码可将值设置在包含节点的可输出字段,或直接将最终值写入其他节点的可输入字段。在该阶段结束时,浏览器和包含字段的标识符将被视为无效。例如,如果用户代码包含一个在关闭阶段后继续运行的线程,则不允许对场景图进行修改。这样做将产生一个错误。

4.8.3.8 输入输出字段和包含节点

包含节点允许拥有输入输出访问类型的字段。因为输入输出字段表示同时具有输入和输出类型的字段,用户代码可能希望写入值,用户代码需遵从某些特殊条件,以便与核心规范保持一致。

为了定义允许的行为,包含的节点和用户代码被认为是解耦的、非相关的实体。字段值的更改通知是通知,没有更多。设置输入输出类型字段值被认为是一个瞬时的、原子动作。当字段值被设定时,输入和输出值都立即被设定。然后执行对用户代码的通知。由于输入输出字段的输出已经设置,所以在当前时间戳中任何进一步更改输入输出字段值的尝试都要遵从 GB/T 28170.1—2011 的 4.4.8.4 循环。

也就是说,如果用户代码收到它的包含节点的输入输出字段的事件通知,在相同的时间戳期间它不能写另一个值到那个相同的输入输出字段,因为输出事件已经发布,包含节点不允许在相同的时间戳内为同一字段发布另一个输出事件。

在接收事件中,脚本应只通过输入输出字段的包含节点接收到的第一个事件。

如果包含的节点在当前时间戳中尚未收到字段的更改,则允许用户代码为字段写入值。如果在用户代码修改后接收到字段的更改,则只处理该节点的只输入部分。通知发送到用户代码,但根据 GB/T 28170.1—2011 的 4.4.8.4“循环”中的循环打破规则,字段值不应更改。

4.8.3.9 执行环境 and 安全性

在特定内部交互环境中的所有用户代码的参与活动被认为是在单一执行空间中运行。此上下文中的代码取决于包含浏览器环境的安全设置以及语言的操作环境。这允许内部交互中的用户代码通过 X3D 执行环境外部的异步机制进行通信(例如,路由和事件评估)。有些语言绑定可能会受到比其他语言更严格的限制。这是独立实现的。

示例: 使用 Java 语言绑定的用户代码可在不允许网络连接到任何外部服务器的 Web 浏览器沙箱中操作,而使用 API 绑定的用户代码可被授予对整个底层操作系统的完全访问权。

出于安全考虑,浏览器可实施其认为必要的任何方案,以确保良好的安全性并防止内容进行恶意活动。此类活动可能包括用户计算机的病毒式修改或拒绝服务活动或当天被视为安全风险的任何其他活动。

4.8.4 外部交互

4.8.4.1 允许交互

从外部视角看,与浏览器交互的用户代码在整个生命周期不仅对场景图而且对包括浏览器有完全的控制。因此,外部交互具有对浏览器的全面控制。

因为应用程序被认为是浏览器外部的,所以它对内部状态及何时操作是安全或不安全的没有深刻的了解。因此,外部交互被定义为提议功能。外部交互请求浏览器做出更改,然后浏览器将准确地决定何时执行这些请求是安全的。浏览器应执行下面列出的各项服务指南范围内的所有请求。

外部应用程序也可能希望监视节点、字段甚至浏览器本身的变化。浏览器应将变化通知外部应用程序,但应以异步方式进行。也就是说,任何更新都被认为仅是通知,并且不应阻挡浏览器内部评估。结果是,通知从在浏览器内发生至到达外部应用程序会有一些延迟。

示例: 由于整个系统的传输滞后,坐在远程计算机上的外部应用程序与浏览器之间的数据传输可能会出现延迟。

4.8.4.2 浏览器交互

外部交互的浏览器交互包括提供给内部交互的所有基本服务。除此之外,还允许进行一些额外的交互。允许单个外部应用程序一次与多于一个浏览器交互。它还可指示多个浏览器像一个单一实体那样行动或独自工作。外部应用程序的生命周期独立于浏览器。

4.8.4.3 更新场景图

外部应用程序的一个特点是,它们会爆发性地对 X3D 浏览器制造大量变化。也有可能单个浏览器有许多应用程序连接到它,所有的这些应用程序都在请求浏览器。

事件可被批处理以有助于应用程序性能(见 6.3.19 updateControl)。这一机制提供了一个简单的

门机制来把持所有请求(BeginUpdate),以更新当前加载的世界,直到门被释放为止(EndUpdate)。

在调用 BeginUpdate 时,所有修改当前世界内容的请求都被缓存,而不是传递给浏览器。这种缓存作用会影响所有修改当前世界的请求,包括 loadURL 和 replaceWorld 的调用。一旦调用了 BeginUpdate,任何更多 BeginUpdate 请求都将被忽略,直到下一次调用 Endupdate 时,在那时 EndUpadte 释放当前缓存的浏览器要处理的所有更新。

如果对场景的修改服务请求是在 endUpdate 后和 beginUpdate 前,它应在浏览器的自由裁量权下立即带着时间戳传递给场景。

beginUpdate/endUpdate 请求应限制在单独的会话中。一个应用程序对 beginUpdate 的请求只会缓冲该应用程序所发出的请求,而不是其他任何可能连接到同一个浏览器实例的请求。

当 endUpdate 被调用时,应采用如下的请求执行顺序:

- a) 节点设置值;
- b) 事件级联评估,如在 GB/T 28170.1—2011 中 4.4.8.3 执行模型中所定义。

loadURL/replaceWorld 服务请求(见 6.3.14 loadURL 和 6.3.12 replaceWorld)不受更新控制过程的影响。一旦浏览器收到这些请求,它们的执行便开始。服务定义定义了这些请求的全部行为。

缓存的应用程序请求应在通过另一个缓存队列或单个请求来处理任何更多请求之前处理。

4.8.4.4 监控场景图中的变化

外部交互允许监视场景图中的任何更改。这些更改的通知应及时交付,并保持与浏览器内部生成相同的序列中。

4.8.4.5 同步多个应用程序

当多个应用程序请求浏览器时,请求将按到达浏览器的时间被服务。浏览器将决定到达时间。缓存的场景图更新,它们的到达时间应在 EndUpdate 被请求时确定。到达时间不一定与浏览器选择将事件发送到场景图的时间戳相同。事件被发送到场景图的时间戳应由浏览器决定,但不会早于 EndUpdate 被请求的时间。到达时间用于对来自多个应用程序的冲突请求进行排序,以确保以正确顺序应用事件得到一致的结果。

如果浏览器断定两个请求同时到达,其结果是依赖于实现的。

注意: 外部应用程序可同时向给定的仅输入字段发送新值。在这种情况下,浏览器应遵从 GB/T 28170.1—2011 中 4.4.8.5 扇入和扇出。作者可通过确保事件发生在单独的事件级联中来避免非反感行为。

如果浏览器收到 loadURL 或 replaceWorld 请求时当前正在处理类似的请求,旧的请求立即终止,新的请求开始。见 6.3.12 replaceWorld 和 6.3.14 loadURL 获取更多信息。

4.8.5 服务保证

所有服务请求在底层实现支持该服务的地方都应保证得到执行。一旦应用程序发出服务请求,该请求将被传送到浏览器,假设连接仍然可用。也就是说,所有通信都被假定是可靠的。如果浏览器和应用程序之间的连接被破坏(例如,TCP 连接失败),就不能保证交付。实现应定义一个错误条件,可通知用户,每个服务请求的连接都失败了。另外,浏览器接口可包含一个事件,它为失败的用户提供异步通知。

5 数据类型引用

5.1 介绍和主题

5.1.1 介绍

本章描述了第 6 章服务引用定义中使用的与语言无关的数据类型。

5.1.2 主题

表 5.1 指定了本章的主题。

表 5.1 主题

<ul style="list-style-type: none">● 5.1 介绍和主题<ul style="list-style-type: none">○ 5.1.1 介绍○ 5.1.2 主题○ 5.1.3 概念● 5.2 数据类型定义<ul style="list-style-type: none">○ 5.2.1 SAIAction○ 5.2.2 SAIBoolean○ 5.2.3 SAIBrowserApp○ 5.2.4 SAIBrowserName○ 5.2.5 SAIBrowserRef○ 5.2.6 SAIBrowserVersion○ 5.2.7 SAIComponentDeclaration○ 5.2.8 SAIComponent○ 5.2.9 SAIEncoding○ 5.2.10 SAIExecutionContext○ 5.2.11 SAIFieldAccess○ 5.2.12 SAIFieldDeclaration○ 5.2.13 SAIField○ 5.2.14 SAIFieldName○ 5.2.15 SAIFieldType○ 5.2.16 SAIFieldValue○ 5.2.17 SAIFrameRate○ 5.2.18 SAILayerID○ 5.2.19 SAILoadState○ 5.2.20 SAIMatrix○ 5.2.21 SAINavSpeed○ 5.2.22 SAINode○ 5.2.23 SAINodeType○ 5.2.24 SAIParameterList○ 5.2.25 SAIProfileDeclaration○ 5.2.26 SAIPropertyList○ 5.2.27 SAIProtoDeclaration○ 5.2.28 SAIR requester
--

表 5.1（续）

	<ul style="list-style-type: none">○ 5.2.29 SAIRoute○ 5.2.30 SAIScene○ 5.2.31 SAIScript○ 5.2.32 SAIScriptImplementation○ 5.2.33 SAISStream○ 5.2.34 SAISString○ 5.2.35 SAIUnitDeclaration○ 5.2.36 SAIURL○ 5.2.37 NULL
●	5.3 错误类型 <ul style="list-style-type: none">○ 5.3.1 SAIError○ 5.3.2 SAI_BROWSER_UNAVAILABLE○ 5.3.3 SAI_CONNECTION_ERROR○ 5.3.4 SAI_DISPOSED○ 5.3.5 SAI_IMPORTED_NODE○ 5.3.6 SAI_INSUFFICIENT_CAPABILITIES○ 5.3.7 SAI_INVALID_ACCESS_TYPE○ 5.3.8 SAI_INVALID_BROWSER○ 5.3.9 SAI_INVALID_DOCUMENT○ 5.3.10 SAI_INVALID_EXECUTION_CONTEXT○ 5.3.11 SAI_INVALID_FIELD○ 5.3.12 SAI_INVALID_NAME○ 5.3.13 SAI_INVALID_NODE○ 5.3.14 SAI_INVALID_OPERATION_TIMING○ 5.3.15 SAI_INVALID_URL○ 5.3.16 SAI_INVALID_X3D○ 5.3.17 SAI_NODE_IN_USE○ 5.3.18 SAI_NODE_NOT_AVAILABLE○ 5.3.19 SAI_NOT_SHARED○ 5.3.20 SAI_NOT_SUPPORTED○ 5.3.21 SAI_URL_UNAVAILABLE
●	5.4 事件类型 <ul style="list-style-type: none">○ 5.4.1 综述<ul style="list-style-type: none">■ 5.4.1.1 SAI_Browser_Connection_Error■ 5.4.1.2 SAI_Browser_Event■ 5.4.1.3 SAI_Browser_Initialized■ 5.4.1.4 SAI_Browser_Shutdown■ 5.4.1.5 SAI_Browser_URL_Error○ 5.4.2 SAIFieldEvent
●	表 5.1 主题

5.1.3 概念

本条款中的所有数据类型都是独立于语言绑定的。它们表示作为参数传递的单值信息,返回值或可通过外部创作界面生成的错误条件。每种语言绑定都应定义每种数据类型的实现。

这些数据类型代表每种类型的具体实现;也就是说,X3D 浏览器如何在内部表示它们。

5.2 数据类型定义

5.2.1 SAIAction

SAIAction 是一个单值字段,表示更一般服务类型的限定符。SAIAction 类型的每次使用应定义可接受值的范围。

5.2.2 SAIBoolean

SAIBoolean 表示一个真值或虚值。

5.2.3 SAIBrowserApp

SAIBrowserApp 是代表整个浏览器应用程序的数据类型。这与 5.2.5 SAIBrowserRef 不同,因为它定义了浏览器应用程序本身,而 SAIBrowserRef 定义了对浏览器功能的标准化接口的引用。该数据类型应包含一些获取 SAIBrowserRef 的方法。

5.2.4 SAIBrowserName

SAIBrowserName 定义浏览器名称的表示形式。如果浏览器的实现不支持此信息,则将空值视为此数据类型的合法表示形式。

5.2.5 SAIBrowserRef

TSaIBrowserRef 表示浏览器引用。这是每个浏览器实例的唯一标识符。单个语言绑定可能会对唯一性设置条件,允许其他方法检查对同一浏览器的等效引用。

在 4.4.6 浏览器中进一步定义了浏览器概念。

5.2.6 SAIBrowserVersion

SAIBrowserVersion 定义了浏览器版本的表示形式。如果浏览器实现不支持此信息,则将空值视为此数据类型的合法表示形式。

5.2.7 SAIComponentDeclaration

SAIComponentDeclaration 定义了有关组件及其声明的所有信息。它可用来表示文件中声明的组件信息和浏览器中的可用组件。

5.2.8 SAIComponent

SAIComponent 指定了在请求中使用的组件的标识符,组件由名称和级别组成,两者都封装在此标识符中。

5.2.9 SAIEncoding

SAIEncoding 为编码类型指定标识符。

5.2.10 SAISessionContext

SAISessionContext 是用于表示与作为运行时实体的当前名称空间相关的子信息的数据类型。

5.2.11 SAIFieldAccess

此数据类型定义允许对字段进行访问的类型。有效值只有初始化、输入、输出和输入输出。

5.2.12 SAIFieldDeclaration

SAIFieldDeclaration 表示节点字段的抽象声明。声明对于该节点的所有实例都是常量,并且不包括字段值。它可被认为是包含 SAIFieldAccess、SAIFieldName 和 SAIFieldType 数据类型的包装数据类型。

5.2.13 SAIField

SAIField 表示节点的特定字段的标识符。它保证在单个节点引用的范围内是唯一的。就生成的所有字段引用而言,它不一定是唯一的。要在场景图中唯一地定义字段,需要节点和字段标识符的组合。

字段概念在 4.4.9 字段中得到了进一步的定义。

5.2.14 SAIFieldName

SAIFieldName 表示字段的名称。

5.2.15 SAIFieldType

SAIFieldType 指定字段表示的数据类型。在某些情况下(字段类型表示 MFString 或 SFString),此字段类型可能对应于 SAISessionContext。有效的字段类型在 GB/T 28170.1—2011 中的 5 字段和事件引用中定义。

5.2.16 SAIFieldValue

SAIFieldValue 表示要以语言特定术语设置或返回 SAIFieldType 的值。可构造表,将表示的每个 SAIFieldType 映射到至少一个语言特定条目。此数据类型包含 SAIFieldType 类的项和由第一个项的值指定的字段类型的项。应支持 GB/T 28170.1—2011 中的 5 字段和事件参考中定义的所有字段类型。

5.2.17 SAIFrameRate

SAIFrameRate 表示浏览器当前正在以每秒帧数为单位的帧率。

5.2.18 SAISessionID

SAISessionID 是操作的目标层的标识符。层的排序是在 GB/T 28170.1—2011 中 35.4.2 LayerSet 中定义的 LayerSet 节点的层字段中的顺序位置。

5.2.19 SAISessionState

SAISessionState 表示节点或 EXTERNPROTO 实例的加载状态。该状态应为 NOT_STARTED、IN_PROGRESS、COMPLETE 或 FAILED 之一。

5.2.20 SAIMatrix

SAIMatrix 为 3×3 或 4×4 矩阵指定标识符。

5.2.21 SAINavSpeed

SAINavSpeed 表示用户的导航速度(以 m/s 为单位)。

5.2.22 SAINode

SAINode 指定节点的标识符。不同语言绑定可能对唯一性设置条件,允许其他方法检查对同一节点的等效引用。节点引用不需要是活动场景图的一部分。

空值是此数据类型的合法值。它用于指示不使用任何节点标识符。例如,对于字段服务 setValue (见 6.7.6 setValue),SFNode 字段类型上的值为 NULL 用于清除以前可能设置为值的节点。

节点概念在 4.4.8 节点中进一步定义。

5.2.23 SAINodeType

SAINodeType 表示节点的类型。

5.2.24 SAIParameList

SAIParameList 是用于表示可传递给服务请求的参数列表的抽象数据类型。每种语言绑定都应定义参数的确切列表及其与特定语言类型的映射。这可用于在绑定提供单个服务的多个重载实现时,表示多个参数列表。

5.2.25 SAIProfileDeclaration

SAIProfileDeclaration 指定有关配置文件及其声明的所有信息。它可用来表示文件中声明的配置文件信息和浏览器中可用的配置文件。

5.2.26 SAIPropertyList

SAIPropertyList 是定义一组用于提供属性的键/值对的抽象数据类型。

5.2.27 SAIProtoDeclaration

SAIProtoDeclaration 表示 PROTO 或 EXTERNPROTO 的声明。它不代表由声明创建的实例。该声明在运行时不能更改,只能由文件或字符串构成。这允许浏览器构建优化的内部存储机制,该内部存储机制可能不能使用正常场景图遍历机制进行遍历。每种语言绑定应定义其自己的表示形式和创建 PROTO 声明实例的方法。

5.2.28 SAIRequester

SAIRequester 表示请求要执行的服务的客户端应用程序或其部分的标识符。此数据类型的变量通常用于标识特定的客户端代码段,这些代码利用回调函数的机制来侦听场景图或浏览器状态中的某些信息的更改。

5.2.29 SAIRoute

SAIRoute 表示一个 ROUTE 构造。ROUTE 不是场景图中的节点,也不代表具体的结构(见 GB/T 28170.1—2011 中 4.4.8.2 ROUTEs)。

5.2.30 SAIScene

SAIScene 代表了一个完整的世界和定义它的所有信息,包括节点、路由、原型和导出。场景可能来自对文件或字符串的解析,也可能是通过这个 API 以编程方式构造的。

5.2.31 SAIScript

SAIScript 表示执行内部交互的用户代码的包含节点。它是作为 X3D 场景图的一部分存在的一个 X3DNode 对象。

5.2.32 SAIScriptImplementation

SAIScriptImplementation 是用户代码使用的标记数据类型,用于标记用户代码执行的入口点。它将提供浏览器在执行用户代码时调用的与生命周期有关的方法,如 initialize() 和 shutdown()。它不应用作任何服务定义的参数或返回任何服务定义的类型。

5.2.33 SAISStream

SAISStream 表示连续到达的 X3D 内容。

5.2.34 SAISString

SAISString 表示用 UTF-8 通用字符集格式化的字符串。(见 ISO/IEC 10646)。

5.2.35 SAIUnitDeclaration

SAIUnitDeclaration 定义了有关单位及其声明的所有信息。用于表示为文件声明的单元信息,无论是显式还是默认。

5.2.36 SAIURL

SAIURL 是引用单个 URL 的数据类型。URL 可以是任何有效的 URL 表示,但通常定义为符合 RFC 4248 的人类易读的字符串。

5.2.37 NULL

NULL 表示空值。它不包含任何数据类型或对任何数据类型的引用,作为从服务返回的有效值,当没有任何东西可返回时,也不会产生错误。

5.3 错误类型

5.3.1 SAIError

本条款定义响应服务请求可能生成的错误类型。错误作为服务请求的同步值生成,并作为 SAIError 类型的变量返回。这些错误类型出现在服务请求的错误定义中(见 6.1.3 使用的约定)。语言绑定应定义 SAIError 数据类型的表示形式,并为下面定义的每个错误赋值,但也可能定义其他的错误数据类型。

5.3.2 SAI_BROWSER_UNAVAILABLE

此错误表示获取 SAIBrowserApp 引用的请求失败。例如,网络连接中断,或者所需的引用类型不受特定于供应商的语言绑定实现的支持。

5.3.3 SAI_CONNECTION_ERROR

出现了一个错误,导致浏览器和外部应用程序之间的连接变得不起作用。因此,无法执行服务请求。这是一种与 SAI_BROWSER_UNAVAILABLE 不同的错误条件,因为它假设已经获得了有效的引用,并且错误发生在稍后的时间。

5.3.4 SAI_DISPOSED

当前 SAINode、SAIField 或 SAIBrowserRef 引用的请求是对在此服务请求之前已被释放的对象进行的。

5.3.5 SAI_IMPORTED_NODE

如 GB/T 28170.1—2011 中所定义的 4.4.6 Import/Export 语义不被允许时,尝试使用导入节点的操作。例如,将导入的节点作为子节点添加到当前场景中的另一个节点。

5.3.6 SAI_INSUFFICIENT_CAPABILITIES

用户试图向执行上下文中添加一个节点,该上下文大于该场景的配置文件和组件定义所定义的功能。

5.3.7 SAI_INVALID_ACCESS_TYPE

执行字段操作的尝试失败,因为该字段类型的操作无效。例如,试图读取仅输入字段的值将生成此错误。

5.3.8 SAI_INVALID_BROWSER

作为服务请求参数的一部分提供的 SAIBrowserRef 数据类型的实例在此请求之前已被处理。

5.3.9 SAI_INVALID_DOCUMENT

当用户试图将 World Wide Web Consortium Document Object Model(DOM)文档导入 X3D 场景时,文档不能完全解析为 X3D 场景图。在许多情况下,可能会产生此错误。

示例 1: 无效的文档结构或。

示例 2: 没有正确的根元素。

5.3.10 SAI_INVALID_EXECUTION_CONTEXT

作为此服务请求参数的一部分提供的 SAIExecutionContext 数据类型的实例在此请求之前已被处理。

5.3.11 SAI_INVALID_FIELD

作为此服务请求参数的一部分提供的 SAIField 数据类型实例在此请求之前已被处理。

5.3.12 SAI_INVALID_NAME

提供给服务请求的名称无效或无法在该对象的上下文中找到。

5.3.13 SAI_INVALID_NODE

作为此服务请求参数的一部分提供的 SAINodeID 数据类型的实例在此请求之前已被处理。

5.3.14 SAI_INVALID_OPERATION_TIMING

用户正在尝试发出在允许此类操作的上下文之外执行的服务请求(见 4.8.3.7 用户代码生命周期)。如果服务将此定义为可能的错误类型,则仅由内部交互引发。外部交互永远不会产生这个错误。

5.3.15 SAI_INVALID_URL

由于语法错误,作为此服务请求参数的一部分提供的 SAIURL 数据类型实例无效。由于请求的 URL 不可用而导致的错误将生成 SAI_URL_UNAVAILABLE 错误或通知此类问题的异步事件。

5.3.16 SAI_INVALID_X3D

传递给此服务请求的 SAIStream、SAIString 或 X3D 文件(例如,由于获取 URL 引用的结果)包含无效的语法,无法解析以生成用于其他服务请求的合法数据类型。

5.3.17 SAI_NODE_IN_USE

指示命名节点处理操作试图重用已在此当前场景中其他位置定义的名称。

示例: 用户试图将节点导入为已由 DEF 描述的名称。

此错误的另一种用法是指示该节点或其一个子节点,目前正在另一个场景中使用。同时在多个场景中共享单个节点实例是错误的。

5.3.18 SAI_NODE_NOT_AVAILABLE

用于 IMPORTed 节点的错误条件。用户已经描述了 IMPORT 语句所说的有效的节点,但尚未加载基础内联以验证它是否是正确的 EXPOSEd 节点。

5.3.19 SAI_NOT_SHARED

当浏览器当前没有参与一个共享场景图时,假设浏览器当前正在参与一个共享场景图的服务请求。

5.3.20 SAI_NOT_SUPPORTED

对浏览器实现中不可用的功能提出服务请求时的通用错误。例如,如果用户请求浏览器不支持的配置文件声明,则可能会生成此错误。

5.3.21 SAI_URL_UNAVAILABLE

由于未设置 URL,无法完成要求浏览器设置世界 URL 集的服务请求。此错误通常由 getWorldURL(6.4.6 getWorldURL)或 getNode(6.4.7 getNode)服务请求生成。

5.4 事件类型

5.4.1 综述

浏览器事件类型是针对浏览器实现状态的更改而生成的异步事件。下列事件类型应由每种语言绑定实现。可定义其他与实现相关的事件,以补充提供的事件类型。

5.4.1.1 SAI_Browser_Connection_Error

表示错误条件的数据类型发生在浏览器和外部应用程序之间的内部连接中(见 4.5.3.5 连接中断)。

5.4.1.2 SAI_Browser_Event

表示每个浏览器服务生成的事件的一般类的数据类型(见 4.5.3 浏览器到外部应用程序)。

5.4.1.3 SAI_Browser_Initialized

表示已完成加载 X3D 内容初始化过程的浏览器的数据类型(见 4.5.3.2 初始化)。

5.4.1.4 SAI_Browser_Shutdown

表示正在关闭的浏览器的数据类型。也就是说,执行模型不再运行或显示内容(见 4.5.3.3 关闭)。

5.4.1.5 SAI_Browser_URL_Error

数据类型表示没有 URL 可处理以形成有效的 X3D 内容时的错误条件(见 4.5.3.4 无可用的 URLs)。

5.4.2 SAIFieldEvent

用于表示外部应用程序注册感兴趣的字段值更改的通知的数据类型。

6 服务引用

6.1 介绍和主题

6.1.1 介绍

本章提供了浏览器应向本部分定义的外部应用程序提供的服务语义的详细定义。

6.1.2 主题

表 6.1 指定了本章的主题。

表 6.1 主题

<ul style="list-style-type: none">● 6.1 介绍和主题<ul style="list-style-type: none">○ 6.1.1 介绍○ 6.1.2 主题○ 6.1.3 使用的约定● 6.2 创建连接<ul style="list-style-type: none">○ 6.2.1 介绍○ 6.2.2 getBrowser○ 6.2.3 createBrowser● 6.3 浏览器服务<ul style="list-style-type: none">○ 6.3.1 介绍○ 6.3.2 getName○ 6.3.3 getVersion○ 6.3.4 getCurrentSpeed○ 6.3.5 getCurrentFrameRate○ 6.3.6 getSupportedProfiles
--

表 6.1 (续)

○	6.3.7	getProfile
○	6.3.8	getSupportedComponents
○	6.3.9	getComponent
○	6.3.10	getExecutionContext
○	6.3.11	createScene
○	6.3.12	replaceWorld
○	6.3.13	importDocument
○	6.3.14	loadURL
○	6.3.15	setDescription
○	6.3.16	createX3DFromString
○	6.3.17	createX3DFromStream
○	6.3.18	createX3DFromURL
○	6.3.19	updateControl
○	6.3.20	registerBrowserInterest
○	6.3.21	getRenderingProperties
○	6.3.22	getBrowserProperties
○	6.3.23	changeViewpoint
○	6.3.24	print/println
○	6.3.25	dispose
○	6.3.26	setBrowserOption
●	6.4	执行上下文服务
○	6.4.1	getSpecificationVersion
○	6.4.2	getEncoding
○	6.4.3	getProfile
○	6.4.4	getComponents
○	6.4.5	getUnits
○	6.4.6	getWorldURL
○	6.4.7	getNode
○	6.4.8	createNode
○	6.4.9	createProto
○	6.4.10	namedNodeHandling
○	6.4.11	getProtoDeclaration
○	6.4.12	protoDeclarationHandling
○	6.4.13	getExternProtoDeclaration
○	6.4.14	externprotoDeclarationHandling
○	6.4.15	getRootNodes
○	6.4.16	getRoutes
○	6.4.17	dynamicRouteHandling
○	6.4.18	dispose
●	6.5	场景服务
○	6.5.1	介绍
○	6.5.2	getMetaData
○	6.5.3	setMetaData
○	6.5.4	namedNodeHandling

表 6.1 (续)

○ 6.5.5	rootNodeHandling
● 6.6	节点服务
○ 6.6.1	介绍
○ 6.6.2	getTypeName
○ 6.6.3	getType
○ 6.6.4	getField
○ 6.6.5	getFieldDefinitions
○ 6.6.6	dispose
● 6.7	域服务
○ 6.7.1	介绍
○ 6.7.2	getAccessType
○ 6.7.3	getType
○ 6.7.4	getName
○ 6.7.5	getValue
○ 6.7.6	setValue
○ 6.7.7	registerFieldInterest
○ 6.7.8	dispose
● 6.8	路由服务
○ 6.8.1	getSourceNode
○ 6.8.2	getSourceField
○ 6.8.3	getDestinationNode
○ 6.8.4	getDestinationField
○ 6.8.5	dispose
● 6.9	Proto 类型服务
○ 6.9.1	isExternproto
○ 6.9.2	createInstance
○ 6.9.3	getFieldDefinitions
○ 6.9.4	checkLoadState
○ 6.9.5	requestImmediateLoad
● 6.10	配置服务
○ 6.10.1	介绍
○ 6.10.2	getComponentName
○ 6.10.3	getComponentLevel
○ 6.10.4	getProfileName
○ 6.10.5	getProfileComponents
○ 6.10.6	getProviderName
○ 6.10.7	getUnitCategory
○ 6.10.8	getUnitConversion
○ 6.10.9	getUnitName
● 6.11	脚本内容提供的服务
○ 6.11.1	介绍
○ 6.11.2	创建阶段
○ 6.11.3	安装阶段

表 6.1 (续)

■ 6.11.3.1	setBrowser
■ 6.11.3.2	setFields
■ 6.11.3.3	initialize
○ 6.11.4	实现阶段
■ 6.11.4.1	prepareEvents
■ 6.11.4.2	eventsProcessed
○ 6.11.5	处理阶段
■ 6.11.5.1	shutdown
● 6.12	矩阵服务
○ 6.12.1	介绍
○ 6.12.2	set
○ 6.12.3	get
○ 6.12.4	inverse
○ 6.12.5	transpose
○ 6.12.6	multiply
○ 6.12.7	multiplyWithVector
● 表 6.1	主题
● 表 6.2	SAI 请求约定
● 表 6.3	updateControl SAIAction 值
● 表 6.4	registerBrowserInterest SAIAction 值
● 表 6.5	标准呈现属性定义
● 表 6.6	扩展功能标准属性的描述
● 表 6.7	changeViewpoint SAIAction 值
● 表 6.8	getNode SAIAction 值
● 表 6.9	namedNodeHandling SAIAction 值
● 表 6.10	ProtoDeclarationHandling SAIAction 值
● 表 6.11	externprotoDeclaration SAIAction 值
● 表 6.12	dynamicRouteHandling SAIAction 值
● 表 6.13	rootNodeHandling SAIAction 值
● 表 6.14	registerFieldInterest SAIAction 值

6.1.3 使用的约定

本章中的每一个服务都定义了一个可通过 SAI 进行的特定请求。每个请求都由多个特征值定义。在表 6.2 中,第一列定义了每个特征值类型,第二列定义了该特性值的属性。

表 6.2 SAI 请求约定

参数:	第一个参数类型,第二个参数类型 [可选参数类型],[多选参数类型]s
返回:	返回值数据类型或预期范围的列表
错误:	数据类型错误列表

表 6.2 (续)

事件:	第一个事件 第二个事件
缓存:	是,否或者不适用
外部:	如果是的话,这只适用于外部接口, 如果不是的话,内部与外部接口均适用

参数按数据类型列出,并以逗号(,)和空格分隔。方括号[]中显示的参数表示在括号中指定的数据类型为单一可选值。[]s 符号,即方括号跟随“s”字符,表示允许该类型有多个可选参数。例如,[SAIURL]s 表示数据类型 SAIURL 可提供多个实例,而[SAIURL]则表示只能提供一个 SAIURL 实例。

每种服务的所有特征值都应针对支持的语言进行实现。在每个表的末尾,将会有一些解释性文本,其中包括与该服务的实现相关的额外信息。

6.2 创建连接

6.2.1 介绍

以下服务可用来建立会话并获取浏览器引用。单独的浏览器实现可能支持其中的一种或两种方法,但至少需要支持一种。

6.2.2 **getBrowser**

参数:	SAIParameterList
返回:	SAIBrowserRef
错误:	SAI_BROWSER_UNAVAILABLE
事件:	none
缓存:	N/A

getBrowser 服务返回 X3D 浏览器的实例的引用,通过该实例可处理其他服务请求。

这是一个阻塞调用。在生成 SAIBrowser 值(可能包括需要启动 X3D 浏览器的新实例)或出现错误条件之前,将不会处理来自这个外部应用程序的其他请求。

如果应用程序在同一会话中向相同的浏览器发送两次请求,则返回相同的浏览器标识符。

一个执行命令可用不同的参数类型定义多个服务变量。例如,访问嵌入在 HTML 页面中的浏览器时,可能有多种格式。而对同一语言绑定的另一台机器进行远程访问时,只有一种格式。

语言绑定可通过添加额外的错误类型来处理平台特定的问题。

6.2.3 **createBrowser**

参数:	SAIParameterList, SAIPropertyList
返回:	SAIBrowserApp
错误:	SAI_BROWSER_UNAVAILABLE
事件:	none
缓存:	N/A

createBrowser 服务创建一个浏览器应用程序的新实例。浏览器应从 URL 未设置开始(即为未活动的 X3D 场景图)。URL 可在稍后的时间使用 loadURL(见 6.3.14 loadURL)或 replaceWorld(见 6.3.12 replaceWorld) 服务请求来设置。

属性列表用于定义浏览器应用程序本身的属性。服务请求应使用与 URL 中定义的相同属性列表定义(见 6.3.14 loadURL)。不需要支持完全相同的功能,但是属性列表格式应是相同的,并且任何行为都是相同的。

这是一个阻塞请求。在生成 SAI 浏览器值(可能包括需要启动 X3D 浏览器的新实例)或出现错误条件之前,将不会处理来自这个外部应用程序的其他请求。

该服务的每个请求将根据提供的参数值生成一个新的浏览器应用程序实例。

一个执行命令可用不同的参数类型定义多个服务变量。例如,在远程机器上启动浏览器或在正在运行的应用程序中创建一个新窗口,都可能会有多种格式。

可通过单独的语言绑定添加额外的错误类型来处理平台特定的问题。

单独的语言绑定可能会向 SAI 浏览器应用程序添加额外的调用,以获得语言和特定于平台的低级句柄。

示例: 它可允许对脱机图像渲染器的原始图像像素数据进行访问,以便用户可使用特定于平台的调用来进行额外的绘图和合成操作。

6.3 浏览器服务

6.3.1 介绍

可从浏览器请求以下服务。虽然没有重复指定,但是如果应用程序和浏览器之间的会话失败,则每当发出请求时,所有服务都能抛出 SAI_CONNECTION_ERROR。

注意: 参数或返回值的数据表示没有指定。它可标识所有参数为二进制表示,或者同样有效地表示为字符串。

6.3.2 getName

参数:	SAIBrowserRef
返回:	SAIBrowserName
错误:	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getName 服务返回浏览器的名称。这个名称是依赖于实现的。如果该服务不支持,则返回 NULL 值。

6.3.3 getVersion

参数:	SAIBrowserRef
返回:	SAIBrowserVersion
错误:	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getVersion 服务返回浏览器应用程序的当前版本。浏览器的版本号是依赖于实现的。如果不支持

此服务,则返回 NULL 值。

6.3.4 getCurrentSpeed

参数:	SAIBrowserRef, SAILayerID
返回:	SAINavSpeed
错误:	SAI_DISPOSED SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	No
外部:	No

getCurrentSpeed 服务返回当前界面的导航速度。当前速度是在当前绑定视点的坐标系中活动层的当前绑定的 NavigationInfo 节点的平均导航速度。

6.3.5 getCurrentFrameRate

参数:	SAIBrowserRef
返回:	SAIFrameRate
错误:	SAI_DISPOSED SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	No
外部:	No

getCurrentFrameRate 服务以每秒帧数的形式返回浏览器的当前帧显示速率。如果不支持,则返回的值为零。

6.3.6 getSupportedProfiles

参数:	SAIBrowserRef
返回:	SAIProfileDeclaration [SAIProfileDeclaration]s
错误:	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getSupportedProfiles 服务返回该浏览器支持的所有配置文件的列表。所有浏览器应至少支持一个概要文件。如果浏览器返回一个不完全支持的配置文件的声明将会产生错误。

6.3.7 getProfile

参数:	SAIBrowserRef, SAIStrng
返回:	SAIProfileDeclaration
错误:	SAI_DISPOSED SAI_NOT_SUPPORTED
事件:	None

缓存: No
外部: No

getProfile 服务返回指定配置文件的声明。SAISting 参数的值是用于获取声明的概要文件的名称,并与 GB/T 28170.1—2011 中指定的名称一致。如果有错误的名称、拼写错误或不能精准匹配的名称,则会产生错误。如果浏览器支持命名配置文件,则只需要返回一个 SAIProfile 值。如果不支持命名配置文件,则生成 SAI_NOT_SUPPORTED。

6.3.8 getSupportedComponents

参数: SAIBrowserRef
返回: SAIComponentDeclaration[SAIComponentDeclaration]s
错误: SAI_DISPOSED
事件: None
缓存: No
外部: No

getSupportedComponents 服务返回该浏览器支持的所有组件的列表。所有浏览器都应根据需要支持配置文件,至少支持一个组件。

6.3.9 GetComponent

参数: SAIBrowserRef, SAIComponent
返回: SAIComponentDeclaration
错误: SAI_DISPOSED
SAI_NOT_SUPPORTED
事件: None
缓存: No
外部: No

GetComponent 服务返回指定组件的声明。SAIComponent 参数的值是从中获取声明的组件和级别的名称,并且应完全符合文件格式中使用的命名约定。如果用户提供错误案例,拼写错误或完全匹配以外的任何名称,则应是错误的。如果浏览器支持命名组件和请求的级别,则只需返回 SAIComponentDeclaration 值。如果它不支持所需级别的组件,则应生成 SAI_NOT_SUPPORTED。

6.3.10 getExecutionContext

参数: SAIBrowserRef
返回: SAIExecutionContext
错误: SAI_DISPOSED
SAI_INVALID_OPERATION_TIMING
事件: None
缓存: No
外部: No

getExecutionContext 服务返回当前的执行上下文。如果在内部交互中使用,则此服务返回包含节

点所在的执行上下文(见 4.4.3 Containing Node)。在外部交互中使用时,此服务返回当前的顶级场景。

执行上下文是场景的基本形式,但仅提供对由 GB/T 28170.1—2011 中 4.4.7 Run-time 名称范围中定义的 X3D 名称范围规则定义的本地节点,PROTOs 和路由的访问。根据场景图中的位置,返回的类型可以是 SAIScene 的实例,允许用户使用更强大的功能。

6.3.11 createScene

参数:	SAIBrowserRef,[SAIProfileDeclaration],[SAIComponentDeclaration]s
返回:	SAIScene
错误:	SAI_DISPOSED SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	No
外部:	No

createScene 服务创建一个符合给定配置文件和组件声明的新空场景。虽然规范不要求提供,但既不提供配置文件也不提供组件定义应是错误的。用户应为此请求提供至少一个有效的配置文件或组件标识符。

以这种方式创建的场景应始终将其规范版本设置为“3.0”“3.1”或“3.2”(视情况而定),并将编码设置为“脚本”。

6.3.12 replaceWorld

参数:	SAIBrowserRef, SAIScene
返回:	None
错误:	SAI_INVALID_SCENE SAI_DISPOSED SAI_INVALID_OPERATION_TIMING
事件:	SAI_Browser_Shutdown SAI_Browser_Initialized
缓存:	No
外部:	No

replaceWorld 服务用 SAIScene 参数指定的场景替换当前场景。如果在处理当前请求期间发出另一个 replaceWorld 或 loadURL(见 6.3.14 loadURL)请求,则终止当前请求并启动新请求。在这种情况下,不应生成额外的关闭事件。初始化事件应在世界准备运行时生成。场景不需要包含任何有效内容。设置 NULL 值将清除当前设置的场景并留下没有可渲染内容且没有当前场景的空白浏览器。

收到此服务请求后立即生成 SAI_Browser_Shutdown 事件。

SAI_Browser_Initialized 事件是在设置了新节点并且所有特定于浏览器的初始化已经发生但在第一次驱动事件级联启动之前生成的(事件级联可能是由于通过内部脚本的初始化过程而导致的)。

6.3.13 importDocument

参数:	SAIBrowserRef, DOMNode
返回:	SAIScene
错误:	SAI_INVALID_DOCUMENT

	SAI_DISPOSED
	SAI_INVALID_OPERATION_TIMING
	SAI_NOT_SUPPORTED
事件:	None
缓存:	No
外部:	No

importDocument 服务是一个实用程序请求,用于导入万维网联合会(W3C)文档对象模型(DOM)文档或文档片段并将其转换为 X3D 场景。输入允许由 2.[W3CDOM2]定义的任何形式的 DOM 节点。虽然所有派生类型都可用,但只需要在转换过程中支持 DOCUMENT,DOCUMENT_FRAGMENT 和 ELEMENT 类型。该方法仅执行转换过程,不显示结果场景。然后可将场景用作 replaceWorld(参见 6.3.12 replaceWorld)服务的参数。进行转换时,DOM 与生成的场景之间没有持久的连接。每个请求都应是单个转换尝试(如果 DOM 与 X3D 场景图结构不匹配,则转换可能不成功)。

对此方法的支持是可选的,并且取决于浏览器对 XML 编码的支持(见 ISO/IEC 19776-1:2015)。如果浏览器实现支持 XML 编码,则它应支持此服务。如果浏览器不支持 XML 编码,则实现可能支持此服务。用户代码可通过使用 getBrowserProperties 服务检查浏览器属性来检查是否支持此服务。如果浏览器实现不支持此服务,则应生成 SAI_NOT_SUPPORTED 错误。

6.3.14 loadURL

参数:	SAIBrowserRef, SAIURL [SAIURL]s, SAIPropertyList
返回:	None
错误:	SAI_INVALID_URL SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	SAI_Browser_Shutdown SAI_Browser_Initialize SAI_Browser_URL_Error
缓存:	No
外部:	No

loadURL 服务在 SAIPropertyList 实例的内容的控制下插入由当前世界中的 URL 标识的内容。如果参数列表使浏览器即将关闭(示例替换嵌入了浏览器的 HTML 帧),则在收到此服务请求后立即生成 SAI_Browser_Shutdown 事件。

SAI_Browser_Initialized 事件是在设置了新节点并且所有特定于浏览器的初始化已经发生但在第一次驱动事件级联启动之前生成的(事件级联可能是由于通过内部脚本的初始化过程而导致的)。

属性列表定义应包括至少一个属性,该属性定义提供的 SAIBrowserRef 中加载作为新场景提供的 URL。如果属性列表为空,则 URL 是合法的 X3D 文件,则用新场景替换当前浏览器的场景。

如果在处理当前请求期间发出另一个 replaceWorld(见 6.3.12 replaceWorld)或 loadURL 请求,则终止当前请求并启动新请求。在这种情况下,不应生成额外的关闭事件。如果调用 replaceWorld,则应在世界准备运行时生成 SAI_Browser_Initialized 事件。

6.3.15 setDescription

参数:	SAIBrowserRef, SAIStrString
-----	-----------------------------

返回:	None
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

如果浏览器支持描述标题,它将被设置为新的描述。通常,这将是窗口标题栏中的标题。在单个窗口中可能有多个浏览器的情况下,结果是依赖于实现的。

6.3.16 createX3DFromString

参数:	SAIBrowserRef, SAIStrng
返回:	SAIScene
错误:	SAI_INVALID_X3D SAI_INVALID_OPERATION_TIMING SAI_DISPOSED SAI_NOT_SUPPORTED
事件:	None
缓存:	No
外部:	No

createX3DFromString 服务从字符串创建节点。该字符串应包含有效的 X3D 语法;否则会产生错误。如果在此字符串中遇到任何相对 URL,则假定基地址为当前浏览器位置。该字符串不需要包含 X3D 文件头。如果存在,则应将其视为所含 X3D 版本的指标。如果不存在,则假定的默认版本应为 GB/T 28170.1—2011 中 7.2.4.2 标题声明中规定的版本。在制定 ISO/IEC 19775 之前,浏览器不需要支持上述何种版本。

如果字符串包含合法的 X3D 语句但不包含任何节点实例,则仍应返回不包含根节点但具有相应声明标识符的有效 SAIScene 值。例如,字符串可能包含 EXTERNPROTO 声明,但不包含任何节点的实例。如果 SAIStrng 以浏览器实现不支持的编码格式提供内容,则浏览器将生成 SAI_NOT_SUPPORTED 错误。

6.3.17 createX3DFromStream

参数:	SAIBrowserRef, SAIStrng
返回:	SAIScene
错误:	SAI_INVALID_X3D SAI_INVALID_OPERATION_TIMING SAI_DISPOSED SAI_NOT_SUPPORTED
事件:	None
缓存:	No
外部:	No

createX3DFromStream 服务从用户提供的任意输入数据流创建节点。流应包含第一个字符的有

效 X3D 语法;否则,会产生错误。如果在此字符串中遇到任何相对 URL,则假定基数为当前浏览器位置。根据格式的编码要求,流需要包含 X3D 文件头。

如果字符串包含合法的 X3D 语句但不包含任何节点实例,则仍应返回不包含根节点但具有相应声明标识符的有效 SAIScene 值。例如,字符串可能包含 EXTERNPROTO 声明,但不包含任何节点的实例。如果 SAISStream 标识的流以浏览器实现不支持的编码格式提供内容,则浏览器应生成 SAI_NOT_SUPPORTED 错误。

6.3.18 createX3DFromURL

参数:	SAIBrowserRef, SAIURL[SAIURL]s
返回:	SAIScene
错误:	SAI_INVALID_URL SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	SAI_B_URL_Error
缓存:	No
外部:	No

createX3DFromURL 服务根据 URL 表示的文件内容创建节点。URL 可以是相对 URL,其被认为是使用浏览器位置作为基地址。该 URL 描述的场景应由返回的 SAIScene 值标识。

6.3.19 updateControl

参数:	SAIBrowserRef, SAIAction
返回:	None
错误:	SAI_DISPOSED
事件:	None
缓存:	N/A
外部:	Yes

updateControl 用于处理缓冲更新的方式。
SAIAction 参数指定可对缓冲区应用的操作。可添加其他动作,例如,查询项目的数量或缓冲器的状态,并且是依赖于实现的。表 6.3 定义了本部分规定的操作。

表 6.3 updateControl SAIAction 值

服务	操作类型
updateControl	BeginUpdate
	EndUpdate

调用 EndUpdate 时生成的事件的时间戳取决于实现,但应与当前世界中的时间一致。也就是说,时间戳不能在与之同时生成的其他当前内部事件的相对“过去”。
BeginUpdate 和 EndUpdate 不是嵌套调用。一旦调用了 BeginUpdate,就可调用它多次,但只需要一次 EndUpdate 调用就可将缓冲的事件释放到场景图中。在没有先前匹配的 BeginUpdate 的情况下调用 EndUpdate 无效。

6.3.20 registerBrowserInterest

参数:	SAIBrowserRef, SAIAction, SAIRequester
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	接收所有的 SAIBrowserEvents
缓存:	No
外部:	Yes

registerBrowserInterest 服务指定请求者作为所有 SAIBrowserEvents 的接收者。提出此服务请求的行为本身并不意味着应生成任何事件。

SAIRequester 类型的参数可从输入源推断出来,可能不需要是参数的一部分,并且是依赖于实现的。对此服务的每个绑定都应指定此要求。

SAIAction 类型的参数指定这是添加对事件感兴趣的请求,还是删除对事件的兴趣。表 6.4 定义了本部分规定的操作。

表 6.4 registerBrowser Interest SAIAction 值

服务	操作类型
registerBrowserInterest	AddBrowserInterest
	RemoveBrowserInterest

对当前浏览器的任何更改都应发送给已注册对这些事件感兴趣的侦听器。这些事件通知应独立于与浏览器通信的方法。

符合条件的实现至少应将 4.5.3 浏览器中定义的事件提供给外部应用程序。

6.3.21 getRenderingProperties

参数:	SAIBrowserRef
返回:	SAIPropertyList
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getRenderingProperties 服务用于查询浏览器的呈现功能。这给出了浏览器的低级硬件功能列表,而不是支持的 X3D 组件。例如,它将使用户了解可处理多少个多纹理单元,并允许最终用户自定义要在 MultiTexture 节点中使用的节点数。属性列表中的键和值取决于实现,仅供参考。表 6.5 列出了此服务返回的属性。

表 6.5 标准呈现属性定义

属性名称	数据的值和类型	描述
Shading	String	使用的着色算法的类型。典型值为 Flat,Gouraud,Phong,Wireframe
MaxTextureSize	String	支持的最大纹理大小。格式应为 WIDTHxHEIGHT,描述每个方向的像素数(例如 1 024×1 024)
TextureUnits	Integer	执行多重纹理所支持的纹理单元数
AntiAliased	Boolean	如果渲染当前是否已使用抗锯齿,则为真或假
ColorDepth	Integer	屏幕支持的颜色深度位数。允许优化纹理选择,尤其是较低颜色深度的屏幕功能
TextureMemory	Float	在视频卡上可用于纹理放置的内存量(兆字节)

用户不能通过修改该服务返回的属性直接影响浏览器的呈现属性。

6.3.22 getBrowserProperties

参数:	SAIBrowserRef
返回:	SAIPropertyList
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getBrowserProperties 服务用于查询浏览器引用本身的功能。这给出了此浏览器引用能支持的扩展接口的列表。例如,它可用于查询本部分提供的已定义浏览器类或未来扩展的浏览器实现特定扩展的存在。

表 6.6 定义了 GB/T 28170 部分保留的一些标准属性名称。如果浏览器实现者选择添加其他功能,则这些附加属性的命名约定应遵循 GB/T 28170.1—2011 中 4.1.3 使用的约定中定义的准则。

表 6.6 扩展功能标准属性的描述

属性名称	数据的值和类型	描述
ABSTRACT_NODES	Boolean	除了支持 X3DNode 抽象类型的基本要求之外,浏览器实现还支持使用与 GB/T 28170.1—2011 中定义的抽象节点类型相对应的接口来描述每个节点类型的能力。这表明浏览器至少支持一致性级别 2
CONCRETE_NODES	Boolean	除了支持所有抽象类型的要求之外,浏览器实现还支持使用与 GB/T 28170.1—2011 中定义的具体节点类型相对应的接口来描述每个节点类型的能力。这表明浏览器至少支持一致性级别 3
EXTERNAL_INTERACTIONS	Boolean	此 SAIBrowserRef 支持外部接口所需的其它服务。在内部交互中提供给用户代码的 SAIBrowserRef 不应设置此属性

表 6.6 (续)

属性名称	数据的值和类型	描述
PROTOTYPE_CREATE	Boolean	浏览器实现支持通过服务请求动态创建 PROTO 和 EXTERNPROTO 表示的功能。基本服务功能仅允许创建从文件格式读取的预定义 PROTO 结构的实例
DOM_IMPORT	Boolean	浏览器实现支持 importDocument 服务请求
XML_ENCODING	Boolean	浏览器支持 XML 作为文件格式编码
CLASSIC_VRML_ENCODING	Boolean	浏览器支持 Classic VRML 编码
COMPRESSED_BINARY_ENCODING	Boolean	浏览器支持二进制文件格式编码

6.3.23 changeViewpoint

参数:	SAIBrowserRef, SAIAction, SAILayerID
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	Yes
外部:	No

changeViewpoint 服务将指定 Layer 上当前绑定的 X3DViewpointNode 实例更改为该操作定义的实例。有效的操作类型是 previous, next, first 和 last。如果未指定 Layer ID, 则使用当前的 activeLayer。当使用此服务请求更改视点时, 浏览器应首先取消绑定当前实例, 然后绑定新实例。也就是说, 可绑定堆栈可绑定节点上的项目数量不会因为发出此服务请求而增加。表 6.7 定义了本部分规定的操作。

表 6.7 changeViewpoint SAIAction 值

服务	操作类型
changeViewpoint	Next
	Previous
	First
	Last

此服务请求意味着 X3DViewpointNode 实例存在标准的, 众所周知的排序, 因此应遵循一致的视觉行为。排序应基于以下规则:

- a) 排序在最初解析的文件或流中声明, 包括 PROTO 实例的解析, 但不包括 EXTERNPROTO 或 X3DInlineNode 实例;
- b) 始终追加动态创建的节点实例;
- c) 位于 X3DInlineNode 实例和 EXTERNPROTO 实例中的实体应按照解析外部场景的顺序, 并

附加到列表中。包含这些外部实例还取决于浏览器属性 ListInlineViewpoints。

无效的 SAILayerID 将导致忽略该操作。SAILayerID 值小于零或大于或等于已定义层数的请求被视为无效,并将导致发出错误 SAI_INVALID_OPERATION_TIMING。

如果 world 仅包含默认的 X3DViewpointNode 实例,则此请求对可视输出无效。

6.3.24 print/println

参数:	SAIBrowserRef, SAIStrng
返回:	None
错误:	None
事件:	SAI_DISPOSED
缓存:	No
外部:	No

print 服务将消息打印到浏览器的控制台。特定于语言的绑定可能会提供此服务的重载变体,这些变体不采用 SAIStrng 值,而是采用其他数据类型。其他变体可能包括自动添加换行符/换行符的功能,而无需在 SAIStrng 值中显式声明它们。绑定应至少提供基本 SAIStrng 变体和附加换行符/换行符的变体。

除非浏览器引用已被处理,否则用户代码可随时调用此服务,不受任何限制。

6.3.25 dispose

参数:	SAIBrowserRef
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING
事件:	SAI_B_Shutdown
缓存:	No
外部:	Yes

dispose 服务指示客户端即将退出此会话,并且浏览器可自由处置此客户端可能已使用的任何资源。SAI_Browser_Shutdown 事件仅发送到此客户端,并且可由客户端计算机上的语言实现在内部生成(也就是说,不需要浏览器本身生成此事件,只需生成事件)。如果因为调用了 BeginUpdate 而保留了任何事件,则处理浏览器也应调用 EndUpdate 将这些事件发布到浏览器。

6.3.26 setBrowserOption

参数:	SAIBrowserRef, SAIStrng, SAIObjct
返回:	SAIBoolean
错误:	SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	No
外部:	Yes

setBrowserOption 服务允许在 GB/T 28170.1—2011 中的 9.2.5 浏览器选项中设置选项。名称字段应为 GB/T 28170.1—2011 中表 9.2 中定义的名称之一。此服务应返回 SAIBoolean 值,指示更改请求是否成功。浏览器不需要支持对任何选项的动态更改。如果不支持浏览器选项,则返回值 FALSE。

6.4 执行上下文服务

6.4.1 getSpecificationVersion

参数:	SAIExecutionContext
返回:	SAIString
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getSpecificationVersion 返回版本字符串,该字符串描述此场景所遵循的规范版本。此版本表示 ISO /IEC 14772-1:1997,GB/T 28170.1—2011 中定义的相应版本号,或者对于实现支持的 ISO/IEC 14772-1:1997 中规范之前的 VRML 版本具有值 1.0。

6.4.2 getEncoding

参数:	SAIExecutionContext
返回:	SAIEncoding
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getEncoding 服务返回用于生成由指定执行上下文表示的场景部分的编码类型。编码是固定集之一,但可能包括特定于浏览器实现的其他类型。所需的最小值集(但不一定受浏览器实现支持)应为:

- a) Scripted:对于完全通过 SAI 创建并且不是通过某个文件来源创建的场景;
- b) ASCII:用于 VRML 1.0 规范文件;
- c) VRML:用于 ISO/IEC 19776-2:2015 中规定的 VRML 和 X3D Classic VRML 编码;
- d) XML:用于 ISO/IEC 19776-1:2015 中规定的 X3D XML 编码文件;
- e) Binary:用于 ISO/IEC 19776-3:2015 中规定的 X3D 二进制编码文件;
- f) BIFS:用于 ISO/IEC 14496-1:2010 中规定的 MPEG-4 BIFS 编码格式。

6.4.3 getProfile

参数:	SAIExecutionContext
返回:	SAIProfileDeclaration
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getProfile 服务返回用于描述此场景的配置文件。如果规范版本是针对 X3D 之前的规范版本,则配置文件应为 VRML。如果未提供配置文件,则应返回 NULL。

6.4.4 getComponents

参数:	SAIExecutionContext
返回:	[SAIComponentDeclaration]s
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getComponents 服务返回用于描述场景的组件。返回的列表应仅表示显式组件声明,而不表示概要声明中的隐含组件。如果未设置组件定义,则应返回 NULL。

6.4.5 getUnits

参数:	SAIExecutionContext
返回:	[SAIUnitDeclaration]s
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	no
外部:	no

getUnits 服务返回用于描述场景的所有单位。返回的列表应代表所有显式单位声明和当前应用的默认单位。

6.4.6 getWorldURL

参数:	SAIExecutionContext
返回:	SAIURL
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	no
外部:	no

getWorldURL 服务返回此场景的完全限定 URL。这将返回整个 URL,包括可能与 CGI 调用或类似机制相关联的任何可能的参数。如果整个程序是以编程方式创建的,则 URL 应为 NULL。

6.4.7 getNode

参数:	SAIExecutionContext, SAIString, SAIAction
返回:	SAINode
错误:	SAI_INVALID_OPERATION_TIMING

	SAI_INVALID_NAME
	SAI_DISPOSED
	SAI_NODE_NOT_AVAILABLE
事件:	None
缓存:	No
外部:	No

getNode 服务根据指定的条件搜索节点,并返回节点的标识符。

SAIString 用于标识在当前加载的 X3D 场景中使用命名语句 DEF,IMPORT 或 EXPORT 之一标记的节点的名称,或之前添加的 namedNodeHandling 请求(见 6.4.10 namedNodeHandling)。

SAIAction 应指出应使用哪种命名类型来查找节点。例如,提供 ImportNode 的操作不应返回可能有效的名称,而是描述使用 DEF 语句命名的节点。表 6.8 定义了本部分规定的操作。

表 6.8 getNode SAIAction 值

服务	行为类型
getNode	DEFNode
	IMPORTNode
	EXPORTNode

访问权限仅适用于此场景中的名称。内联文件中的 DEF 不能按照 GB/T 28170.1—2011 中 4.4.3 DEF/USE 语义和 4.4.6 的导入/导出语义进行访问。

如果 SAIAction 是 IMPORTNode 并且名称有效但是源内联节点尚未提供节点定义,则应生成 SAI_NODE_NOT_AVAILABLE。

6.4.8 createNode

参数:	SAIExecutionContext, SAIString
返回:	SAINode
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED SAI_INVALID_NAME
事件:	None
缓存:	No
外部:	No

createNode 服务创建由包含 X3D 节点类型名称的 SAIString 值给定的节点的新默认实例。节点的可用性由包含场景的配置文件和组件声明定义。该名称仅应引用内置节点,不应用于创建 PROTO 或 EXTERNPROTO 的实例。如果该节点在当前指定的配置文件和组件中不可用,则浏览器应发出 SAI_INVALID_NAME 错误。

6.4.9 createProto

参数:	SAIExecutionContext, SAIString
返回:	SAINode

错误：SAI_INVALID_OPERATION_TIMING
SAI_DISPOSED
SAI_INVALID_NAME
事件：None
缓存：No
外部：No

createProto 服务创建名为 PROTO 的新默认实例。用于创建当前执行上下文在另一个 proto 中的 proto 实例的命名和作用域规则由 GB/T 28170.1—2011 中的 4.4.7 Run-time 名称范围定义。如果没有可用的 PROTO 声明与给定名称匹配,则浏览器应生成 SAI_INVALID_NAME 错误。

6.4.10 namedNodeHandling

参数：SAIExecutionContext, SAIAction, SAIAction, SAIStrng,
[SAINode | SAIStrng, [SAIStrng]]
返回：None
错误：SAI_INVALID_OPERATION_TIMING
SAI_DISPOSED
SAI_IMPORTED_NODE
SAI_NODE_IN_USE
SAI_INVALID_NAME
事件：None
缓存：Yes
外部：No

namedNodeHandling 服务是添加,删除或更新由 SAIStrng 值标识的节点的请求,其中该名称被视为使用 DEF 或 IMPORT 语义。添加/删除/更新应由第一个 SAIAction 值描述。如果名称已作为映射存在,则当前映射将替换为新映射。添加新的命名节点时,新的命名节点不需要是此场景的一部分。第二个 SAIAction 值描述了 DEF 或 IMPORT 命名工具中的哪一个是此服务请求的目标。这能确保正确的语义被使用。如果要添加操作并且已注册名称,则会生成 SAI_NODE_IN_USE。如果操作是要替换或更新,并且该节点尚未注册,则实现可将其视为添加请求。表 6.9 定义了本部分规定的操作。

表 6.9 namedNodeHandling SAIAction 值

服务	操作类型
namedNodeHandling	AddDEFNode/UpdateDEFNode
	RemoveDEFNode
	AddIMPORTNode/UpdateIMPORTNode
	RemoveIMPORTNode
	AddEXPORTNode/UpdateEXPORTNode
	RemoveEXPORTNode

第一个 SAIStrng 值标识一个节点的名称,它应被场景感知。这个名称不是节点的固有属性,而这只是一个映射函数。

第二个参数根据正在采取的行动提供一个选项。SAINode 值是对验证 DEF 名称添加可能需要的节点的引用。对于添加 IMPORT, 第二个字符串应是 DEF'd 内联中的导出节点名称, 可选的第三个字符串应是在此场景中存储它的名称。

6.4.11 getProtoDeclaration

参数:	SAIExecutionContext, SAIStrString
返回:	SAIPProtoDeclaration
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_NAME SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

对于 getProtoDeclaration 服务将从这个场景返回指定的 PROTO 声明表示。这只能用于请求 PROTO 声明。对 EXTERNPROTO 声明的请求应生成 SAI_INVALID_NAME。

6.4.12 protoDeclarationHandling

参数:	SAIExecutionContext, SAIStrString, SAINode, SAIAction
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	Yes
外部:	No

protoDeclarationHandling 服务是添加, 删除或更改 SAIStrString 值标识的 ProtoDeclaration 的请求。SAIAction 参数指定服务请求是添加还是删除声明节点。如果名称已作为映射存在, 则当前映射将替换为新映射。添加新声明时, 它可能来自另一个场景。表 6.10 定义了本部分规定的操作。

表 6.10 ProtoDeclarationHandling SAIAction 值

服务	操作类型
protoDeclarationHandling	AddProto/UpdateProto
	RemoveProto

6.4.13 getExternProtoDeclaration

参数:	SAIExecutionContext, SAIStrString
返回:	SAIPProtoDeclaration
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_NAME SAI_URL_UNAVAILABLE

SAI_DISPOSED

事件: None

缓存: No

外部: No

getExternProtoDeclaration 服务从场景返回指定的 EXTERNPROTO 声明表示。这将仅用于请求 EXTERNPROTO 声明。对 PROTO 声明的请求将生成 SAI_INVALID_NAME。

6.4.14 externprotoDeclarationHandling

参数: SAIExecutionContext, SAIStrng, SAINode, SAIAction

返回: None

错误: SAI_INVALID_OPERATION_TIMING
SAI_DISPOSED

事件: None

缓存: Yes

外部: No

externprotoDeclarationHandling 服务是添加,删除或更新 SAIStrng 值标识的 ExternProtoDeclaration 的请求。

SAIAction 参数用于指示服务请求是否是声明节点的添加或删除。如果名称已作为映射存在,则当前映射将替换为新映射。添加新声明时,它可能来自另一个场景。表 6.11 定义了本部分规定的操作。

表 6.11 externprotoDeclaration SAIAction 值

服务	操作类型
externprotoDeclarationHandling	AddExternProto/UpdateExternProto
	RemoveExternProto

6.4.15 getRootNodes

参数: SAIExecutionContext

返回: SAINodes

错误: SAI_INVALID_OPERATION_TIMING
SAI_INVALID_NAME
SAI_DISPOSED

事件: None

缓存: No

外部: No

getRootNodes 服务返回在执行上下文中返回当前根节点的列表。如果上下文是从文件中生成的,则根节点按照文件中声明的顺序排列。然后按照浏览器接收到的顺序将添加的节点附加到列表中。如果上下文是以编程方式生成的,则节点按照浏览器接收的顺序排列。

6.4.16 **getRoutes**

参数:	SAIExecutionContext
返回:	SAIRoutes
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getRoutes 服务在场景中获得搜索中当前路由的列表。返回的路由列表仅为顶级路由。包含在 Proto 定义或原型实例中的路由不应包含在此列表中。

6.4.17 **dynamicRouteHandling**

参数:	SAIExecutionContext, SAINode, SAIField, SAINode, SAIField, SAIAction
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_NODE SAI_INVALID_FIELD SAI_DISPOSED
事件:	None
缓存:	Yes
外部:	No

dynamicRouteHandling 服务是根据 SAIAction 值指定的操作处理路由的请求。
SAIAction 类型的参数指定是否应添加或删除此路由。可添加其他动作,例如查询指定路线的存在。表 6.12 中描述了本部分定义的操作。SAINode/SAIField 对参数被视为定义路由请求的源字段和目标字段。

表 6.12 **dynamicRouteHandling SAIAction 值**

服务	操作类型
dynamicRouteHandling	AddRoute
	DeleteRoute

路径修改请求应符合 GB/T 28170.1—2011 中 4.4.8 事件模型中定义的一般事件模型方案。事件级联的结束被认为是由应用程序将事件发送到 X3D 浏览器环境而启动的级联。由于处理初始事件而产生的任何新级联不应被视为用于确定事件级联。

如果操作是删除路由,并且先前已删除路由,则不应生成错误。

6.4.18 **dispose**

参数:	SAIExecutionContext
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING

事件:	SAI_Browser_Shutdown
缓存:	No
外部:	Yes

dispose 服务指定客户端对此执行内容所表示的资源不再感兴趣。浏览器现在可采取任何必要的操作来回收此执行上下文现在或将来任何时候使用的任何资源。如果该执行上下文已经被释放,则进一步的请求不会产生任何效果。

6.5 场景服务

6.5.1 介绍

场景是执行上下文服务的扩展,提供了附加服务。Scene 服务实现应包括 6.4 Execution context 服务中的所有服务,并包括以下附加服务。

6.5.2 getMetadata

参数:	SAIScene, SAIStrString
返回:	SAIStrString
错误:	SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	No
外部:	No

getMetadata 服务从使用在 GB/T 28170.1—2011 中的 7.2.5.5 META 语句中定义的 META 语句指定的场景中返回元数据项。META 语句中指定的元数据表示为 SAIStrString 键/值对。每个键对应于零或一个值。

可选地,浏览器可提供子服务以发现该场景的有效密钥作为该服务的一部分。
可使用 6.6 节点服务来操纵由 GB/T 28170.1—2011 中定义的元数据节点定义的元数据。

6.5.3 setMetadata

参数:	SAIScene, SAIStrString, SAIStrString
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	Yes
外部:	No

setMetadata 服务以 META 语句的形式在场景中插入元数据项,如 GB/T 28170.1—2011 中的 7.2.5.5 META 语句中所定义。元数据表示为 SAIStrString 键/值对。每个键对应于零或一个值。使用已存在的键设置项目将替换现有值。如果给定键的值为 NULL,则从该场景中删除与该键关联的 META 语句。

可使用 6.6 节点服务来操纵由 GB/T 28170.1—2011 中定义的元数据节点定义的元数据。

6.5.4 namedNodeHandling

除了 6.4.10 namedNodeHandling 中描述的功能外,场景服务还扩展了与导出节点一起使用的功能。定义扩展如下:

namedNodeHandling 服务是添加,删除或更改由 SAIString 值标识的节点的请求,其中该名称被视为使用 DEF 或 IMPORT 或 EXPORT 语义。添加/删除/更新应由第一个 SAIAction 参数值描述。如果名称已作为映射存在,则当前映射将替换为新映射。添加新的命名节点时,不需要将其作为此场景的一部分。

第二个 SAIAction 参数值描述了哪个 DEF,IMPORT 或 EXPORT 命名工具应是此服务请求的目标。这可确保应用正确的语义。如果要添加操作并且已注册名称,则会生成 SAI_NODE_IN_USE。如果操作是要替换或更新,并且该节点尚未注册,则实现可将其视为添加请求。

第一个 SAIString 值指定具有节点的名称,因为在此场景中应知道该节点。该名称不是节点的固有属性,仅用作映射函数。

第二个参数根据正在采取的行动提供一个选项。SAINode 值是对验证 EXPORT 或 DEF 名称添加可能需要的节点的引用。对于添加 IMPORT,第二个字符串应是 DEF 内联中的导出节点名称,可选的第三个字符串应是在此场景中存储它的名称(这与 IMPORT 的经典 VRML 语法 inlined_def.export_name [AS 相对应] IMPORT_NAME])。对于添加 EXPORT,第二个参数应为字符串,这是将节点导出为的可选名称。如果第一个 SAIString 没有描述标有 DEF 的节点,它将生成 SAI_INVALID_NAME 错误。

6.5.5 rootNodeHandling

参数:	SAIScene, SAINode, SAIAction
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_NODE SAI_DISPOSED SAI_IMPORTED_NODE SAI_NODE_IN_USE SAI_INSUFFICIENT_CAPABILITIES
事件:	None
缓存:	Yes
外部:	No

rootNodeHandling 服务是添加和删除此场景的根节点的请求。根节点可在其下面包含整个子“场景图”。

SAIAction 参数用于指示服务请求是添加还是删除节点。如果要删除操作并且节点不是已知的根节点,则应生成错误。如果操作是添加节点,则应将其附加到当前根节点列表。表 6.13 定义了本部分规定的操作。

表 6.13 rootNodeHandling SAIAction 值

服务	操作类型
rootNodeHandling	AddRootNode
	RemoveRootNode

节点受包含场景的功能约束。任何节点都不应具有比场景声明的配置文件和其他组件更强大的功能。如果操作是向场景添加节点并且该节点需要比场景允许的更多功能,则应生成 SAI_INSUFFI-

CIENT_CAPABILITIES。

如果操作是添加节点,并且节点或其任何子节点当前是另一个场景的一部分,则生成 SAI_NODE_IN_USE。

如果操作是删除节点并且该节点不是该字段的已知值,则应以静默方式忽略该请求。

6.6 节点服务

6.6.1 介绍

可请求单个节点的以下服务。每个服务都需要该节点的标识符。在单个节点请求处理其资源之后,对节点服务的任何进一步请求将产生已处置的错误。

虽然未指定,但是如果应用程序和浏览器之间的会话失败,则每当发出请求时,所有服务都能抛出 SAI_CONNECTION_ERROR。

6.6.2 getTypeName

参数:	SAINode
返回:	SAIString
错误:	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getTypeName 服务返回引用节点类型的名称。类型名称是 GB/T 28170.1—2011 中指定的名称,其中定义了节点类型(见 GB/T 28170.1—2011 中的 Node index 以便于访问节点定义)。如果节点表示 PROTO 节点实例,则返回的类型名称是 PROTO 声明的名称。

6.6.3 getType

参数:	SAINode
返回:	SAINodeType
错误:	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getType 服务返回引用节点的类型指示符。类型指示符是为 X3D 规范中的基本节点类型定义的类型,或者是 PROTO 类型名称(如果它是原型节点)。对于符合要求的实现,不需要支持这一点。

6.6.4 getField

参数:	SAINode, SAIFieldName
返回:	SAIField
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_NAME SAI_DISPOSED
事件:	None

缓存:	No
外部:	No

getField 服务返回字段标识符,以便对节点属性执行操作。如果请求的字段是 inputOutput 字段,则可使用字段名称或 set_和_changed 修饰符来根据需要访问节点的相应形式。对字段的访问取决于实现。

6.6.5 getFieldDefinitions

参数:	SAINodeType
返回:	SAIFields
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_NAME SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getFieldDefinitions 服务返回引用节点的所有字段定义的列表。这些定义提供了一种有限形式的 SAIField,除了能读取或写入特定节点实例的字段值之外,它具有所有相同的服务。此请求将 SAIField 值作为此节点的每个实例的通用响应而不是特定实例返回。

6.6.6 dispose

参数:	SAINode
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	Yes
外部:	No

dispose 节点服务指示客户端对该节点表示的资源没有进一步的兴趣。浏览器可采取任何必要的操作来回收此节点现在或将来任何时间消耗的任何资源。如果此节点已被处置,则进一步的请求不起作用。

处置节点不会从场景图中删除节点(如果它首先插入),而是删除每个客户端的任何本地信息。仅当没有其他应用程序或场景图结构包含对此节点的引用时,才会处理基础 X3D 节点表示,并且此操作的责任和时间是特定于浏览器实现的。

6.7 域服务

6.7.1 介绍

以下是可向节点的各个字段请求的服务。如果已经检索出一个字段的节点,则仍然可执行域服务操作,只要在处理节点之前已经获得了字段引用。如果在请求处理字段后对域服务进行调用,则会产生错误。

虽然未指定,但是如果应用程序和浏览器之间的会话失败,则每当发出请求时,所有服务都能抛出 SAI_CONNECTION_ERROR。

6.7.2 **getAccessType**

参数:	SAI Node, SAI Field
返回:	SAI Field Access
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getAccessType 服务返回的是引用节点中指定字段的访问类型。

6.7.3 **getType**

参数:	SAI Node, SAI Field
返回:	SAI Field Access
错误:	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getType 的域服务返回的是所引用节点中指定字段的类型。

6.7.4 **getName**

参数:	SAI Node, SAI Field
返回:	SAI Field Access
错误:	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

如果实现支持,getName 字段服务将返回从节点请求的字段名称。如果服务请求分组节点的 thist_children 字段,则返回“set_children”,但如果同一节点上的子节点有不同的请求,则应返回“children”。

6.7.5 **getValue**

参数:	SAI Node, SAI Field
返回:	SAI Field Access
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_ACCESS_TYPE

	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getValue 字段服务返回由指定字段表示的值,因为它存在于当前事件中。这表示请求时字段的当前值。如果请求是由具有通过 BeginUpdate 缓冲的 setValue 请求的字段组成的,则返回的值应为 setValue 请求之前的旧值。如果字段表示 MFNode 或 SFNode,则字段的值可以是节点。

所有字段类型都应支持从多值数组返回单个值的选项。

6.7.6 setValue

参数:	SAI Node, SAI Field, SAI Field Value
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_ACCESS_TYPE SAI_IMPORTED_NODE SAI_DISPOSED
事件:	No
缓存:	Yes
外部:	No

setValue 字段服务设置指定字段的值。设置请求应遵守为缓冲事件服务指定的要求。

如果字段表示 MFNode 或 SFNode,则字段的值可以是 SAINode 值。允许向节点或字段发送 null 以清除该字段中的值。例如,在 GB/T 28170(见 2.[28170-1])的第 1 部分中的 12 形状组件中指定的 Shape 节点的外观 inputOutput 字段中发送 null,将导致外观字段被清除并设置为默认值为 NULL。

如果 SAINode 值在此文件中注册为 IMPORTed 节点,则应生成 SAI_IMPORTED_NODE 错误。

所有现场设置服务实现都应包括设置单个值的能力。描述多值数组的字段还应包括从现有字段中追加和删除项目的功能。

6.7.7 registerFieldInterest

参数:	SAI Node, SAI Field, SAI Action, SAI Requester
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_ACCESS_TYPE SAI_INSUFFICIENT_CAPABILITIES SAI_NODE_IN_USE SAI_DISPOSED
事件:	SAIFieldInterest
缓存:	No
外部:	No

registerFieldInterest 服务指定请求者作为所有 SAIFieldEvents 的接收者。提出此服务请求的行为本身并不意味着应生成任何事件。表 6.14 定义了本部分规定的操作。

表 6.14 registerFieldInterest SAIAction 值

服务	操作类型
registerFieldInterest	AddInterest
	RemoveINterest

SAIRequester 类型的参数可从输入源推断出来,可能不需要是参数的一部分。
SAIAction 类型的参数指定这是添加事件兴趣还是删除对事件的兴趣的请求。
允许监听哪些功能取决于实现。例如,某些实现可能允许侦听 inputOnly 值和 outputOnly 值,而其他实现只允许侦听 outputOnly 值。

6.7.8 dispose

参数:	SAI Field
返回:	None
错误:	SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	Yes
外部:	No

dispose 字段服务指示客户端对该字段表示的资源没有兴趣。浏览器可采取任何必要的操作来回
收此字段消耗的任何资源,无论是现在还是将来的任何时间。如果此字段已被处理,则进一步的请求
无效。

6.8 路由服务

6.8.1 getSourceNode

参数:	SAI ROUTE
返回:	SAI Node
错误:	SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	Yes
外部:	No

getSourceNode 服务将返回指定路由的源节点。

6.8.2 getSourceField

参数:	SAI ROUTE
返回:	SAI String
错误:	SAI_INVALID_OPERATION_TIMING
事件:	None
缓存:	No

外部: No

getSourceField 服务将返回指定路由的源字段的名称。

6.8.3 getDestinationNode

参数: SAI ROUTE
返回: SAI Node
错误: SAI_INVALID_OPERATION_TIMING
事件: None
缓存: No
外部: No

getDestinationNode 服务将返回指定路由的目标节点。

6.8.4 getDestinationField

参数: SAI ROUTE
返回: SAI String
错误: SAI_INVALID_OPERATION_TIMING
事件: None
缓存: No
外部: No

getDestinationField 服务将返回指定路由的目标字段的名称。

6.8.5 dispose

参数: SAI ROUTE
返回: None
错误: SAI_INVALID_OPERATION_TIMING
事件: None
缓存: Yes
外部: No

dispose 路由服务表明客户端对此路由所表示的资源不再感兴趣。浏览器可采取任何必要的操作来回收此路由所消耗的任何资源,现在或将来的任何时间。如果该路由已经被释放,则进一步的请求不会产生任何效果。

处置路径不会从场景图中删除路径(如果它首先插入),而是删除每个客户端的任何本地信息。如果没有其他应用程序或场景图结构包含对此路由的引用,并且此操作的责任和时间是特定于浏览器实现,则仅处理基础 X3D 节点表示。

6.9 Proto 类型服务

6.9.1 isExternproto

参数: SAI Proto Declaration

返回:	SAI Boolean
错误:	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

isExternproto 服务检查原型声明是否代表 PROTO 或 EXTERNPROTO。如果是 EXTERNPROTO,则返回 TRUE 值,PROTO 声明返回 FALSE。

6.9.2 createInstance

参数:	SAIPProtoDeclaration
返回:	SAINode
错误:	SAI_INVALID_OPERATION_TIMING SAI_INVALID_NODE SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

createInstance 服务从 PROTO 或 EXTERNPROTO 的声明创建一个新实例。如果尚未加载定义,则 EXTERNPROTO 实例可能会因 SAI_INVALID_NODE 而失败。

6.9.3 getFieldDefinitions

参数:	SAI Proto Declaration
返回:	SAIField(S)
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getFieldDefinitions 服务返回 PROTO 或 EXTERNPROTO 声明的所有字段定义的列表。这些定义提供了有限形式的 SAIField 值,除了能读取或写入特定节点实例的字段值之外,它具有所有相同的服务。

6.9.4 checkLoadState

参数:	SAI ProtoDeclaration
返回:	SAILoadState
错误:	SAI_INVALID_NODE SAI_INVALID_URL SAI_DISPOSED
事件:	None
缓存:	No

外部: No

checkLoadState 服务检查 EXTERNPROTO 定义的当前加载状态。状态应为 NOT_STARTED, IN_PROGRESS, COMPLETE 或 FAILED 之一。如果在 PROTO 上调用它,则应生成 SAI_INVALID_NODE 错误。

6.9.5 requestImmediateLoad

参数: SAI ProtoDeclaration
返回: None
错误: SAI_INVALID_OPERATION_TIMING
SAI_DISPOSED
事件: None
缓存: No
外部: No

如果 SAIPprotoDeclaration 值表示 EXTERNPROTO,则 requestImmediateLoad 服务请求浏览器立即开始加载该定义。如果已加载定义或加载已在进行中,则此请求将被自动忽略。

6.10 配置服务

6.10.1 介绍

此处指定的服务允许应用程序识别当前世界的配置。
SAIComponentDeclaration 值指定一个组件声明,其中包含由 GB/T 28170.1—2011 中的 7.2.5.4 COMPONENT 语句定义的 COMPONENT 语句指定的信息。定义的服务是最低要求。实现可提供额外的仅提供信息的服务。
SAIProfileDeclaration 值指定一个概要文件声明,其中包含由 GB/T 28170.1—2011 中的 7.2.5.3 PROFILE 语句和 GB/T 28170.1—2011 附录中包含的概要定义的 PROFILE 语句指定的信息。定义的服务是最低要求。实现可提供额外的仅提供信息的服务。

6.10.2 GetComponentName

参数: SAI ComponentDeclaration
返回: SAI String
错误: None
事件: None
缓存: No
外部: No

GetComponentName 服务将返回指定组件的正式名称。

6.10.3 GetComponentLevel

参数: SAI Component Declaration
返回: SAI String
错误: None

事件:	None
缓存:	No
外部:	No

getComponentLevel 服务返回为组件指定的支持级别。当 SAIComponentDeclaration 来自浏览器服务时,它应代表浏览器支持的最高级别。当它来自场景服务时,该级别表示该场景的请求支持级别。

6.10.4 getProfileName

参数:	SAI Profile Declaration
返回:	SAI String
错误:	None
事件:	None
缓存:	No
外部:	No

getProfileName 服务将返回指定配置文件的正式名称。

6.10.5 getProfileComponents

参数:	SAIProfileDeclaration
返回:	SAIComponentDeclaration(s)
错误:	None
事件:	None
缓存:	No
外部:	No

getProfileComponents 服务将返回一个 SAI 组件声明实例的列表,该实例指定了包含配置文件的每个组件的允许支持。

6.10.6 getProviderName

参数:	SAIProfileDeclaration
返回:	SAIString
错误:	None
事件:	None
缓存:	No
外部:	No

getProviderName 服务是一个只提供信息的服务,它返回一个 SAIString 值,其中包含实现此配置文件的人员或公司的名称。

6.10.7 getUnitCategory

参数:	SAIUnitDeclaration
返回:	SAIString
错误:	SAI_INVALID_OPERATION_TIMING

	SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getUnitConversion 服务返回指定单元声明的正式类别。

6.10.8 getUnitConversion

参数:	SAIUnitDeclaration
返回:	SAIString
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getUnitConversion 服务返回指定单元声明的转换因子。

6.10.9 getUnitName

参数:	SAIUnitDeclaration
返回:	SAIString
错误:	SAI_INVALID_OPERATION_TIMING SAI_DISPOSED
事件:	None
缓存:	No
外部:	No

getUnitName 服务返回用户提供的指定单元声明的名称。

6.11 脚本内容提供的服务

6.11.1 介绍

当作者提供脚本的可执行内容时,应满足某些约定。这允许浏览器明确地传递状态信息,而不管内容语言的类型如何。本条款定义了需要由单个语言绑定定义的服务,以便以一致的、明确的方式通知脚本内容。无论编写内容的是何种语言,脚本内容都必须以相同的方式运行。与其他服务规范不同,浏览器应对用户代码提出这些服务请求,因此,用户代码应在必要时提供这些服务的实现。所有服务由用户自行定义,如果用户不定义服务实现,浏览器将继续运行。

6.11.2 创建阶段

在创建阶段,下载脚本内容并在适当的执行工程中创建内容的实例。一些内容可能需要单独的解释器,而另一些内容可能在与浏览器代码相同的地址和执行空间中创建(例如,与浏览器本身编写的相同语言创建的脚本)。除了与语言相关的实例化过程之外,浏览器不需要任何服务。

6.11.3 安装阶段

在安装阶段,浏览器向脚本提供它将能在系统中使用的所有运行时信息。

6.11.3.1 **setBrowser**

参数:	SAIBrowserRef
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	No

setBrowser 服务将要使用的 SAIBrowserRef 值传递给脚本实现代码。在脚本的生命周期中没有其他方法可获取 SAIBrowserRef,因此如果用户代码需要知道它,就需要在本服务中对它进行存储。该服务应在提出任何其他服务请求之前执行。浏览器可在创建阶段之间和初始化服务请求之前的任何时间调用此服务。浏览器不需要在初始化过程中按照 GB/T 28170.1—2011 中定义的 4.4.8 的事件模型请求它,尽管这是鼓励浏览器实施者请求这项服务的时候。

6.11.3.2 **setFields**

参数:	SAINode, SAIField(s)
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	No

setFields 脚本服务传入此脚本节点实例中声明的字段列表。它还传递包含脚本节点的外部视图,以便用户可直接添加和删除脚本的路径。SAIField 实例表示脚本的所有字段,包括预定义字段。(有关字段访问限制的详细信息,见 4.8.3.4 更新场景图)。该请求应在 setBrowser 和初始化服务请求之间执行。

6.11.3.3 **initialize**

参数:	None
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	No

initialize 脚本服务向脚本提供通知,表明所有基本初始化已完成,并且用户代码在场景图中处于活动状态。此时,用户代码可访问脚本字段值并更改脚本的状态。

6.11.4 实现阶段

6.11.4.1 prepareEvents

参数:	None
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	Yes

prepareEvents 服务提供浏览器即将开始事件级联处理的通知,根据 GB/T 28170.1—2011 中 4.4.8.3 的执行模型中所述的事件执行模型的步骤 4。在此调用期间更改的所有值都应具有当前时间戳,但在从用户代码返回时不应立即传播事件。无论包含节点是否接收到任何事件,都应每帧调用该服务请求。如果包含节点提供 directOutputs,则应立即将这些节点传递给底层节点。

6.11.4.2 eventsProcessed

参数:	SAIBrowserRef
返回:	None
错误:	None
事件:	SAI_Browser_Shutdown
缓存:	No
外部:	Yes

eventsProcessed 服务提供当前事件级联处理已完成的通知,并且现在允许包含节点对场景图进行更新。这对于希望更高效的用户代码非常有用,并且仅在收到字段更改集合后生成新事件。在给定的帧内,用户代码可能不止一次调用此服务。用户代码不能保证此时将收到对包含节点的所有更改,并应采取适当的预防措施。只有在包含节点在此时间戳中收到一个或多个事件后才能调用此服务请求。如果包含节点在当前时间戳中未收到任何事件,则浏览器请求此服务将是错误的。

6.11.5 处理阶段

6.11.5.1 shutdown

参数:	None
返回:	None
错误:	None
事件:	SAI_Browser_Shutdown
缓存:	No
外部:	Yes

shutdown 服务提供用户代码已由包含节点处理的通知。这可能是由于浏览器的完全关闭、加载的世界改变或包含将用户代码改变为另一实现的包含节点。完成此服务请求后,用户代码将不再正常运行或执行。

6.12 矩阵服务

6.12.1 介绍

矩阵对象表示使用双精度数和列主序的标准数学矩阵功能。此处的所有服务都应使用矩阵的标准数学定义进行解释。

实现应提供 3×3 和 4×4 的矩阵。它们可定义其他矩阵顺序。除了这个最小子集之外,实现还可定义其他便利服务;例如,单独访问矩阵元素的能力。实现可允许直接访问矩阵的各个行和列值。

在以下服务定义中,参数描述单精度输入。实现还应提供包含双精度输入的重载定义。

6.12.2 set

参数:	SAIMatrix, SFVec3f, SFRotation, SFVec3f, SFRotation, SFVec3f
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	No

set 矩阵服务将矩阵设置为从参数计算的新值。定义参数按顺序表示:平移,旋转,缩放,缩放方向和中心。如果未指定参数的值,则该参数的默认值应为在 GB/T 28170.1—2011 中定义的 10.4.4 Transform 中定义的 Transform 节点的等效字段的默认值。

6.12.3 Get

参数:	SAIMatrix, SFVec3f, SFRotation, SFVec3f, SFRotation, SFVec3f
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	No

get 服务计算并返回矩阵中的变换值。定义参数按顺序表示:平移,旋转,缩放,缩放方向和中心。

6.12.4 inverse

参数:	SAIMatrix
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	No

inverse 服务计算该矩阵的倒数。

6.12.5 **transpose**

参数:	SAIMatrix
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	No

transpose 服务计算该矩阵的转置。

6.12.6 **multiply**

参数:	SAIMatrix,SAIMatrix
返回:	None
错误:	None
事件:	None
缓存:	No
外部:	No

multiply 服务将第一矩阵乘以第二矩阵实例,将结果放入第一矩阵。实现应定义两个方向的乘法操作:左乘和右乘。

6.12.7 **multiplyWithVector**

参数:	SAIMatrix,SFVec3f
返回:	SFVec3f
错误:	None
事件:	None
缓存:	No
外部:	No

multiplyWithVector 服务将这个矩阵和一个向量一起相乘。实现应定义两个方向的乘法操作:左乘和右乘。

7 一致性和最低支持要求

7.1 介绍和主题

7.1.1 介绍

本章定义了符合本部分的语言绑定的最低要求支持。

7.1.2 主题

表 7.1 列出了本章的主题。

表 7.1 主题

<ul style="list-style-type: none">● 7.1 介绍和主题<ul style="list-style-type: none">○ 7.1.1 介绍○ 7.1.2 主题○ 7.1.3 目标○ 7.1.4 范围● 7.2 一致性<ul style="list-style-type: none">○ 7.2.1 符合 GB/T 28170 的本部分○ 7.2.2 语言绑定● 7.3 最低支持要求<ul style="list-style-type: none">○ 7.3.1 实现要求○ 7.3.2 支持内部和外部互动○ 7.3.3 级别 1○ 7.3.4 级别 2○ 7.3.5 级别 3● 表 7.1 主题● 表 7.2 SAI 数据类型实现规范● 表 7.3 SAI 浏览器实现规范● 表 7.4 SAI 一般服务规范● 表 7.5 SAI 脚本内容服务规范● 表 7.6 SAI 公用事业的规范

7.1.3 目标

本条款讨论提供创作界面(SAI)的 X3D 浏览器的一致性。
本条款中规范的主要目标是：

- a) 通过消除 GB/T 28170 的任意子集来促进互操作性；
- b) 在明确界定的环境中促进可扩展性；
- c) 促进一致性测试开发的一致性；
- d) 在 X3D 浏览器中促进一致的结果；
- e) 促进自动化测试生成。

7.1.4 范围

一致性是为符合本部分的语言绑定而定义的，因此 X3D 浏览器和应用程序使用了 ISO/IEC 19777-1 和 ISO/IEC 19777-2 中指定的场景创作接口语言特定绑定所提供的工具。

由于本部分的抽象性质，无法指定单个语言绑定到规范的一致性测试。在每种语言绑定中都应提供单独的一致性部分，以提供实现语言特定绑定一致性测试所需的信息。

定义了基本配置文件一致性的概念，以确保 X3D 应用程序和 X3D 浏览器的互操作性。基本配置文件一致性是基于一组限制和最低要求。基本配置文件一致性旨在为 X3D 语言绑定提供合理实用程序的功能级别，同时限制 X3D 浏览器的复杂性和资源需求。基本配置文件一致性可能不足以满足 SAI 的所有用途。

7.2 一致性

7.2.1 符合 GB/T 28170 的本部分

X3D 浏览器只有在符合 GB/T 28170.1—2011 中指定的当前配置文件的情况下才符合 GB/T 28170 的本部分。此外,还应满足下列条件:

- a) 浏览器服务的请求应完全符合本部分中规定的行为;
- b) 当浏览器需要读取和解析 X3D 内容时,它应能处理符合 GB/T 28170.1—2011 或这类配置文件的单独规范所支持的配置文件的任何 X3D 文件;
- c) 只有使用 DEF 构造命名的顶级世界中的节点(由 getWorldURL 服务定义的文件)才能对 getNodebrowser 服务请求可见;
- d) 与在 X3D 浏览器环境中生成的事件相比,外部服务请求所产生的事件的处理方式没有任何区别。也就是说,外部产生的事件级联不应有优惠待遇。

7.2.2 语言绑定

如果下列情况下,绑定到 GB/T 28170 本部分的语言是符合的:

- a) 它实现了本部分中定义的服务,包括返回值、错误条件和异步事件;
- b) 所实现的服务符合主题配置文件所需的功能;
- c) 它提供了概述该绑定实现的一致性和最低要求的部分。

7.3 最低要求

7.3.1 实现要求

所有支持创作组件的配置文件都应支持本条款定义的第 1 级功能。有两个级别的一致性,即语言绑定和浏览器实现。

语言绑定至少应实现支持配置文件所需的服务。它还可定义自己的可选的最低要求集,这些要求不低于本条款的要求。除了对主题概要文件的基本支持外,浏览器还应支持为主题概要文件定义的最低功能。通常,浏览器需求被指定为处理特定语言问题的一般需求。

当浏览器实现两种或两种以上具有不同级别最低要求的语言绑定时,浏览器应分别支持每种语言的最低要求。因此,如果一种语言有更高的要求,则不应使用另一种语言的较低要求。

7.3.2 支持内部和外部互动

尽管外部交互使用为内部交互定义的服务的超集,但是不需要对抽象规范的语言绑定来实现这两者。他们可选择其中一个或两个。(强烈提议绑定支持两者)

不需要符合标准的浏览器实现来提供给定语言绑定的内部和外部实现。浏览器应仅支持语言 A 的内部交互和仅支持语言 B 的外部交互。此外,浏览器可选择仅支持内部交互或仅支持外部交互。没有要求支持内部和外部交互。

7.3.3 级别 1

表 7.2~表 7.6 确定了基本概况的最低要求。第一列指定要为其定义一致性的项。这些引用了本部分定义的服务。第二列指定该项的语言绑定规范的要求。第三列指定该服务的浏览器实现的要求。SAI 实现应抛出 SAIError(例如,SAI_NOT_SUPPORTED)。

对于所有这些要求,应假定语言绑定应提供各个服务所需的所有参数的完整实现。数据类型的语言绑定可作为目标语言中的基本类型实现,而不是作为单独的数据类型实现。表 7.2~表 7.6 指出了允

许的位置。

符合此级别的浏览器应支持与 X3DNode 和 X3DMetadataObject 抽象表示的绑定以及所有字段类型。

表 7.2 SAI 数据类型实现规范

项目	绑定支持	最低浏览器支持
SAIAction	根据个人服务请求的要求提供充分支持	由语言定义
SAIBoolean	完全支持	完全支持
SAIBrowserApp	如果支持 createBrowser,则完全支持。 如果只支持 getBuffer,则不需要	可选的(基于 getBrowser/createBrowser 需求)
SAIBrowserName	基本类型	完全支持
SAIBrowserRef	完全支持	完全支持
SAIBrowserVersion	基本类型	完全支持
SAIComponentDeclaration	至少描述名称和级别的基本类型	完全支持
SAIComponent	基本类型	完全支持
SAIEncoding	指定编码类型	完全支持
SAIExecutionContext	提供对子场景的访问	完全支持
SAIFieldAccess	GB/T 28170.1—2011 定义的四种类型的 独立数据类型	完全支持
SAIFieldDeclaration	提供有关访问类型、数据类型和名称的信息	完全支持
SAIField	完全支持	完全支持
SAIFieldName	基本类型	完全支持
SAIFieldType	GB/T 28170.1—2011 的 5 字段类型参考中 定义的所有类型的独立数据类型	完全支持
SAIFieldValue	适合给定字段的基本类型。如果字段是 SF/MFNode,则为 SAINodeID	在 GB/T 28170.1—2011 中定义的适用配置 文件中所定义的用于设置和获取的值
SAIFrameRate	基本类型	完全支持
SAILayerID	基本类型	如果支持分层组件,则完全支持
SAILoadState	基本类型	完全支持
SAIMatrix	基本类型	完全支持
SAINavSpeed	基本类型	完全支持
SAINode	完全支持	完全支持
SAINodeType	基本类型	节点名称的 SAIStrng 表示
SAIParameterList	按照语言和服务定义的要求	依赖于语言和浏览器实现
SAIProfileDeclaration	配置文件中使用的至少名称和组件的说明	完全支持
SAIPropertyList	基本类型	5 对键值对。依赖于语言绑定的值
SAIPprotoDeclaration	完全支持	n/a

表 7.2 (续)

项目	绑定支持	最低浏览器支持
SAIRequester	完全支持	完全支持
SAIRoute	完全支持	完全支持
SAIScene	完全支持	完全支持
SAIScript	完全支持	完全支持
SAIScriptImplementation	完全支持	完全支持
SAIStream	完全支持	完全支持
SAIString	完全支持	完全支持
SAIUnitDeclaration	完全支持	单位类别的每种类型一个条目
SAIURL	支持 URL 和 URN	对于所有 URL 字段,在 GB/T 28170.1—2011 中定义的适用配置文件中指定
SAIError	可能发生的每个错误条件的单独类型,如 5.3 错误类型中定义的那样	适当地生成错误条件

表 7.3 SAI 浏览器实现规范

项目	绑定支持	最小内部支持	最小外部支持
建立连接			
getBrowser, create Browser	至少应提供一个 getBrowser 或 createBrowser 服务	N/A	应提供至少一种与浏览器连接的方法。不支持的连接方法将引发错误。忽略 SAIParameterList
浏览器服务			
getName	应提供	如果不支持返回 NULL	如果不支持返回 NULL
getVersion	应提供	如果不支持返回 NULL	如果不支持返回 NULL
getCurrentSpeed	应提供	如果不支持返回 0.0	如果不支持返回 0.0
getCurrentFrameRate	应提供	如果不支持返回 0.0	如果不支持返回 0.0
getSupportedProfiles	应提供	完全支持	完全支持
getProfile	应提供	完全支持	完全支持
getSupportedComponents	应提供	完全支持	完全支持
getExecutionContext	应提供	完全支持	完全支持
createScene	应提供	支持为数据编码使用的相同配置文件和组件创建场景	
replaceWorld	应提供	完全支持	完全支持
importDocument	应提供	如果不支持返回 NULL	如果不支持返回 NULL
loadURL	应提供	完全支持。忽略 SAIPropertyList 参数值	完全支持。忽略 SAIPropertyList 参数值

表 7.3 (续)

项目	绑定支持	最小内部支持	最小外部支持
浏览器服务			
setDescription	应提供	无限制	无限制
createX3DFromString	应提供	支持 GB/T 28170.1—2011 中定义的适用配置文件中指定的文件限制	
createX3DFromStream	依赖于语言功能来创建原始 I/O 流	支持 GB/T 28170.1—2011 中定义的适用配置文件中指定的文件限制	
createX3DFromURL	应提供	支持 GB/T 28170.1—2011 中定义的适用配置文件中指定的文件限制	
UpdateControl	启动缓冲和结束缓冲的 SAIActions	N/A	完全支持
registerBrowserInterest	添加和删除兴趣的 SAIActions		
getRenderingProperties	应提供	无限制	无限制
getBrowserProperties	应提供	无限制	无限制
setBrowserOptions	应提供	无限制	无限制
changeViewpoint	应提供	无限制	无限制
print/println	应提供	无限制	无限制
dispose	应提供	无限制	无限制

表 7.4 SAI 一般服务规范

项目	绑定支持	最低浏览器支持
执行上下文服务		
getSpecificationVersion	完全支持	完全支持
getEncoding	应提供	完全支持
getProfile	SAIProfileDeclaration	完全支持
getComponents	SAIComponentDeclarations	完全支持
getUnits	SAIUnitDeclarations	完全支持
getWorldURL	应提供	完全支持
getNode	完全支持	完全支持
createNode	完全支持	完全支持
createProto	完全支持	完全支持
namedNodeHandling	添加和删除节点和导入的 SAIActions	完全支持
getProtoDeclaration	应提供	完全支持

表 7.4 (续)

项目	绑定支持	最低浏览器支持
protoDeclarationHandling	添加和删除 Proto 的 SAIActions	完全支持
getExternProtoDeclaration	完全支持	完全支持
externprotoDeclarationHandling	添加和删除 EXTERNPROTO 的 SAIActions	完全支持
getRootNodes	应提供	完全支持
getRoutes	应提供	完全支持
dynamicRouteHandling	添加和删除路由的 SAIActions	完全支持
dispose	应提供	无限制
场景服务		
getMetaData	应提供	完全支持
setMetaData	应提供	完全支持
namedNodeHandling	添加和删除导出的 SAIActions	完全支持
rootNodeHandling	添加和删除节点的 SAIActions	完全支持
节点服务		
getTypeName	应提供	完全支持
getType	应提供	无限制
getField	完全支持	所有字段都应可访问,具体取决于内部和外部交互和节点生命周期的访问规则
getFieldDefinitions	完全支持	完全支持
dispose	应提供	无限制
字段服务		
getFieldDefinitions	完全支持	完全支持
dispose	应提供	无限制
字段服务		
dispose	完全支持	完全支持
getAccessType	应提供	完全支持
getType	见 SAIFieldType	完全支持
getName	完全支持	没有 set_or_changed 修饰符的字段名称
getValue	get1Value 不要求	完全支持
setValue	set1Value 不要求	完全支持。如果字段是 MF 字段,则支持在 GB/T 28170.1—2011 中定义的适用配置文件中指定的最小值

表 7.4 (续)

项目	绑定支持	最低浏览器支持
字段服务		
registerFieldInterest	添加和删除兴趣的 SAIActions 应支持 eventOut 和 exposedFields 的输出端	根据支持的语言绑定
路线服务		
dispose	完全支持	完全支持
getSourceNode	完全支持	完全支持
getSourceField	完全支持	完全支持
getDestinationNode	完全支持	完全支持
getDestinationField	完全支持	完全支持
原型服务		
isExternProto	完全支持	完全支持
createInstance	完全支持	完全支持
getFieldDefintions	完全支持	完全支持
checkLoadState	完全支持	完全支持
requestImmediate Load	完全支持	完全支持
配置服务		
getComponentName	完全支持	完全支持
getComponentLevel	完全支持	完全支持
getProfileName	完全支持	完全支持
getProfileComponents	完全支持	完全支持
getProviderName	完全支持	完全支持
getUnitCategory	完全支持	完全支持
getUnitConversion	完全支持	完全支持
getUnitName	完全支持	完全支持

表 7.5 SAI 脚本内容服务规范

项目	绑定支持	最低浏览器支持
setBrowser	完全支持	完全支持
setFields	完全支持	完全支持
initialize	完全支持	完全支持
prepareEvents	完全支持	完全支持
eventsProcessed	完全支持	完全支持
shutdown	完全支持	完全支持

表 7.6 SAI 公用事业的规范

项目	绑定支持	最低浏览器支持
Matrix	尺寸至少 3×3 和 4×4	完全支持

7.3.4 级别 2

符合级别 2 的浏览器应支持级别 1 中的所有内容,并应支持在浏览器中实现的组件从 X3DNode 派生的所有抽象节点类型。浏览器还应支持实现中可能引入的所有其他对象。

7.3.5 级别 3

符合级别 3 的浏览器应支持级别 2 的所有要求,并支持绑定每个具体节点表示的特定接口。

附 录 A
(规范性附录)
VRML 脚本向后兼容性

A.1 介绍和主题

A.1.1 介绍

本附录详细描述了符合 X3D 的应用程序如何为 ISO/IEC 14772-1:1997 虚拟现实建模语言 (VRML) 功能规范和 UTF-8 编码设计和实现的脚本代码的向后兼容性提供支持。它只供后向兼容的目的,如果浏览器不符合 ISO/IEC 14772 第 1 部分的条款,则不应要求它们实现该兼容。

VRML 事件和脚本模型存在许多缺陷和不兼容,甚至在语言之间也是如此。强烈提议不要使用除 GB/T 28170.1 之外不支持 ISO/IEC 14772-1:1997 的浏览器实现本附录。

A.1.2 主题

表 A.1 列出了本附录的主题。

表 A.1 主题

<ul style="list-style-type: none">● A.1 介绍和主题<ul style="list-style-type: none">○ A.1.1 介绍○ A.1.2 主题● A.2 概念<ul style="list-style-type: none">○ A.2.1 介绍<ul style="list-style-type: none">■ 4.2.1.1 向后兼容性的要求■ 4.2.1.2 VRML 和 X3D 之间的不同○ A.2.1 不支持脚本的行为○ A.2.1 遇到 exposedFields 的行为● A.3 ECMAScript 语言绑定<ul style="list-style-type: none">○ A.3.1 要求○ A.3.2 确定所需规范○ A.3.3 支持的脚本 URL<ul style="list-style-type: none">■ A.3.3.1 内联脚本定义■ A.3.3.2 MIME 类型● A.4 Java 语言绑定<ul style="list-style-type: none">○ A.4.1 要求○ A.4.2 确定所需规范● 表 A.1 主题

A.2 概念

A.2.1 介绍

A.2.1.1 向后兼容性的要求

本附录指定了希望符合本部分的浏览器的要求,并支持在 X3D 环境中运行 ISO/IEC 14772-1:1997 中定义的编程接口和事件模型语义编写的脚本。也就是说,在 GB/T 28170.1—2011 的 29.4.1 脚本中指定的脚本节点的 URL 定义了使用在本部分中定义的接口之一的内容。

如果浏览器只支持 ISO/IEC 14772-1:1997 中定义的脚本接口,则浏览器不符合本部分。符合本附录的浏览器还应支持 GB/T 28170 本部分的全部要求,包括所有所需的语言绑定。

A.2.1.2 VRML 与 X3D 的区别

ISO/IEC 14772-1:1997 与本部分的主要区别在于事件模型的定义。ISO/IEC 14772-1:1997 规范将许多决策留给浏览器实施者来决定,因此许多内容是不兼容的。主要问题是处理用户代码写入字段时场景图更改的传播方式。在 Java 语言附录中,需要立即传递值,但 ECMAScript 语言附录表示它们将被推迟,直到用户代码退出执行。这种不兼容意味着支持直接向后兼容是一个可选的特性。

A.2.2 不支持脚本的行为

支持脚本功能的确应遵循 GB/T 28170.1—2011 中 9 个网络组件中指定的规则。浏览器应尝试按所需顺序加载 URIs,以确定是否支持单个脚本文件。拒绝不受支持的脚本类型应基于 ISO/IEC 14772-1:1997 定义的附录中定义的规则。除了脚本节点定义的首选顺序之外,浏览器不应优先支持另一个规范。例如,如果用户内容在 GB/T 28170.1—2011 一致性脚本代码之前的 url 字段中定义了 ISO/IEC 14772-1:1997 一致性脚本代码,并且浏览器实现支持本附录定义的语言的向后兼容性,则应执行 ISO/IEC 14772-1:1997 脚本。

当用户内容定义了内部交互代码,其中包括符合 ISO/IEC 14772-1:1997 的脚本,而浏览器实现不支持向后兼容性,或者在该语言中不支持向后兼容性时,浏览器将忽略该脚本并移到 url 列表中的下一项。这与无法找到文件或代码不使用浏览器支持的语言是相同的行为。

A.2.3 遇到 exposedFields 的行为

ISO/IEC 14772-1:1997 不允许使用读和写字段(即 exposedFields)。本部分允许使用这些字段。尽管是非常不鼓励的做法,但可使用 exposedField 定义脚本节点并定义符合 ISO/IEC 14772-1:1997 的用户代码。

在这种情况下,浏览器应将 exposedField 视为一组独立的 eventIn, eventOut 和 field 对象。当从事件模型中写入 exposedField 时,应像通常的 eventIn 通知一样通知用户代码。在此之后,应遵循用于读取和写入 4.8.3.8 inputOutput 字段和包含节点中定义的 exposedField 语义的值的规则。

A.3 ECMAScript 语言绑定

A.3.1 要求

如果浏览器打算在脚本节点中支持 ECMAScript 语言的 ISO/IEC 14772-1:1997 向后兼容性,则它应符合 ISO/IEC 14772-1:1997 中的附录 C ECMAScript 语言绑定。

A.3.2 确定所需规范

浏览器应通过使用外部文件给出的协议定义或 MIME 类型来确定支持哪个版本的规范。不应使用其他指标。

A.3.3 支持的脚本 URL

A.3.3.1 内联脚本定义

支持 ECMAScript 向后兼容性的浏览器应通过使用定制协议定义 javascript 来支持内联脚本节点的使用。除了 ISO/IEC 14772-1:1997 中附录 A ECMAScript 脚本参考中指定的 ECMAScript 规范所需的其他协议之外,还需要 ISO/IEC 14772-1:1997 中 7 一致性和最低支持要求中规定的支持。

内联脚本定义的示例:

```
Script {
    url "javascript: function foo() { ... }"
}
```

url 字段可能包含多个 URL 引用,远程文件或者内联代码,如以下示例所示:

```
Script {
    url [
        "http://foo.com/myScript.js",
        "javascript: function foo( ) { ... }"
    ]
}
```

使用 JavaScript 协议将要求浏览器支持在附录 A 中定义的对象和语义。对于符合本部分的浏览器来说,支持具有使用本部分附录 A 定义的对象脚本内容的 JavaScript 协议是错误的。

使用 ecmascript: 协议应表明脚本符合本部分的附录 A 浏览器不应向使用 ecmascript: 协议的脚本提供 ISO/IEC 14772-1:1997 定义的对象。

A.3.3.2 MIME 类型

ISO/IEC 14772-1:1997 ECMAScript 源代码的 MIME 类型定义如下:

application/javascript

为了向后兼容旧 Web 服务,提议浏览器还支持以下 MIME 类型:

application/x-javascript

使用 application/ecmascript 或者 application/x-ecmascript 的 MIME 类型应表明脚本符合本部分附录 A。浏览器不应向使用这些 MIME 类型的脚本提供 VRML 定义的对象。

A.4 Java 语言绑定

A.4.1 要求

如果浏览器打算支持基于 Java 语言的脚本节点实现 ISO/IEC 14772-1:1997 向后兼容性,则必须符合 ISO/IEC 14772-1:1997 附录 B 的 Java 平台脚本引用。

A.4.2 确定所需规范

由于不可能预先确定 Java 类文件通过使用 URL 协议或 MIME 类型实现哪些类和接口,浏览器应

通过检查用户代码实现的基类或接口来确定规范的版本。可通过语言内省功能或使用 `instanceof` 运算符来确定这一点。

声称支持 ISO/IEC 14772-1:1997 的 Java 脚本应扩展基类 `vrml.node.Script`。如果脚本类文件扩展了 ISO/IEC 14772-1:1997 基类并实现了 `org.web3d.x3d.sai.X3DScriptImplementation` 接口,则浏览器应使用 X3D SAI 语义和执行,并且不需要支持 ISO/IEC 14772-1:1997 或符合 ISO/IEC 14772-1:1997 的对象。提议浏览器在检测到这种情况时发出警告。

中 华 人 民 共 和 国
国 家 标 准
信息技术 计算机图形和图像处理
可扩展三维组件(X3D)
第2部分:场景访问接口(SAI)

GB/T 28170.2—2021/ISO/IEC 19775-2:2015

*

中国标准出版社出版发行
北京市朝阳区和平里西街甲2号(100029)
北京市西城区三里河北街16号(100045)

网址:www.spc.org.cn

服务热线:400-168-0010

2021年3月第一版

*

书号:155066·1-64923

版权专有 侵权必究



GB/T 28170.2-2021