网络应用开发课程设计

基于X3DOM框架的Web3D操纵网站

**组员信息：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 姓名 | 学号 | 成员贡献 | 课程设计成绩 |
| 1 | 王晓雨 | 2022280075 | 关键技术，基础开发，高级渲染，中级进阶，实物模拟 |  |
| 2 | 李易婷 | 2022280079 | 功能设计，逻辑流程，动画，功能展示与评估 |  |

Github代码仓库URL：https://github.com/wxy2000/wxy1.git

# **引言**

JavaScript是创建Web网站最有用的编程语言之一，尤其是在WebGL库的支持下。WebGL是JavaScript API或基于OpenGL的库，它使Web能在浏览器中执行三维和二维图形，而无需其他插件。WebGL渲染到计算机的图形处理单元，以超快的速度呈现复杂的概念、动画和科学模拟。有了WebGL，可以很方便地使用HTML5 Canvas元素动态生成图形。因此，设计师和开发人员很容易创建流畅的2D和3D效果。

而其中X3DOM是最流行的WebGL框架。它用于生成可嵌入的3D Web图形。适用于网站和应用程序。运行时性能以声明方式，没有任何低级JavaScript或GLSL编码。

本开发项目的目标是基于X3DOM框架构建一个Web网站，实现Web3D建模，采用标准的浏览器技术，在HTML DOM中实现一个实时的X3D场景，使用结构化的文本表示形式来创建和显示交互式3D场景，使用户仅添加，删除或更改DOM元素就可以操纵3D内容，不需要特定的插件或插件接口，使3D内容就像文本、链接、图像或电影一样，成为网页内容的一部分。输入该3D物体的图形参数，颜色，经过3D建模，且通过JavaScript编程，可以实现利用鼠标对该模型进行的操纵，包括对该3D模型进行拖拽，旋转，缩放。本项目是一项独立的模块，而且全部内容自含，应用场景可用于网络购物平台商品的3D展示，从而给消费者带来更直观形象的购物体验。

# 功能与设计

## 功能介绍

本项目实现的核心功能包括：

无需插件即可显示X3DOM场景，基于标准的浏览器技术，例如HTML5和WebGL；

输入该3D物体的图形参数，颜色，经过3D建模，且通过JavaScript编程，可以实现利用鼠标对该模型进行的操纵，包括对该3D模型进行拖拽，旋转，缩放；

开发DOM操作引擎库，取名为wqMV.js，存放在wqMV文件夹下，其功能类似于jQueryAPI，主要是HTML文档的DOM遍历和操作，以及事件处理；

自己创建wqMV库对js编程进行了简化操作；

系统应支持的终端数取决于Web服务器所支持的最大终端数，一般为几百万台，应支持的并行操作的用户数为Web服务器所设定的并发用户数量，一般为几十万个；

应用场景可用于网络购物平台商品的3D展示，从而给消费者带来更直观形象的购物体验。

## 系统设计

### 2.2.1关键设计

本项目关键设计部分包括index.html主界面，index.js主控模块，wqMV库，X3DOM框架，css样式表。

|  |  |  |
| --- | --- | --- |
| 模块名称 | 功能 | 接口 |
| index.html主界面 | 呈现整个网页 | 无 |
| index.js主控模块 | 调用wqMV库和x3DOM框架，修改html文件 | 无 |
| wqMV库 | HTML 文档的 DOM 遍历和操作，以及事件处理，简化js的编写 | q，m |
| x3DOM框架 | 实现3d模型的呈现，缩放，旋转，渲染，实物模拟 | x3d |
| css样式表 | 设计网页样式 | 无 |

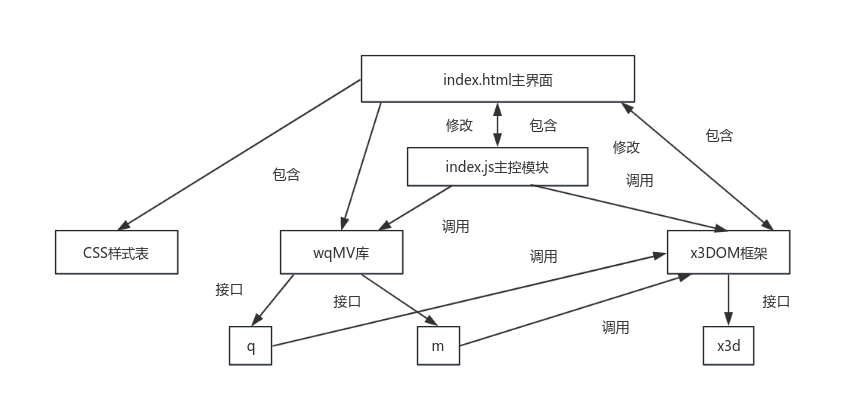
### 2.2.2系统架构

本网站整体架构如下所示：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Hello Web3D! | | | | 显示日期 | | | |
| 基础开发 | | | | | | | |
| 展示圆锥，正方形，圆球 | | | | | | | |
| 动画 | | | | | | | |
| 展示红色小球上下弹跳 | | | | | | | |
| 高级渲染 | | | | | | | |
| 展示图片，视频，声音渲染 | | | | | | | |
| 实物模拟 | | | | | | | |
| 展示小鹿模型 | | | | | | | |
|  | 点我 | 前面 | 右面 | 最佳视野 | 定向光 | 射灯 |  |

### 2.2.3逻辑流程

本项目的整体流程图如下所示：



其中定义了一些系统或产品中涉及的重要术语，为读者在阅读文档时提供必要的参考信息：

|  |  |  |
| --- | --- | --- |
| 序号 | 术语或缩略语 | 说明性定义 |
| 1 | q | Quick，基于DOM快速操作及事件处理 |
| 2 | m | Model，基于文档的排版及校验、日期等操作 |
| 3 | x3d | 引用X3DOM框架 |

### 2.2.4接口设计

1．wqMV库中q接口：Quick，基于DOM操作及事件处理。

q('idname')获取id为idname的网页元素节点；

q('idname').h获取或者设置id为idname网页元素的innerHTML；

q('.class')获取className为class的所有节点；

q('@LI')获取元素标签为li的所有节点；

2.wqMV库中m接口：Model，基于文档的排版及校验、日期等操作。

m.ut()获取当前日期；

3.x3d接口: 引用X3DOM框架

X3D基于XML格式开发，所以可以直接使用XMLDOM文档树、XMLSchema校验等技术和相关的XML编辑工具。目前X3D已经是通过ISO认证的国际标准。

可使用Maya、Blender、3DsMax或者WorldofWarcraft等软件来建立X3D类型的3D模型文件。

网站添加x3d格式MIME类型。加载x3d模型文件。

### 2.2.5系统说明

1.系统运行环境：简单服务器，Windows操作系统，VScode编程平台。

2.开发环境：常见的Web服务器，如apache和IIS等。

3.通信协议：http网络协议。

4.数据类型：

（1）常量：

常量对象wxy：所在文件为index.js文件，属于全局常量，功能为主控程序。

常量对象q：所在文件为wqMV.js文件，属于全局常量，功能为基于 DOM 操作及事件处理。

常量对象m：所在文件为wqMV.js文件，属于全局常量，功能为基于文档的排版及校验、日期等操作。

（2）变量：

wqMV.css文件中的root为变量。

deer.x3d文件为变量。

鼠标的点击，拖拽，滚轮输入为变量，3d模型的颜色，大小，方向的为变量。

日期的获取为变量。

# **功能实现**

## 3.1关键技术

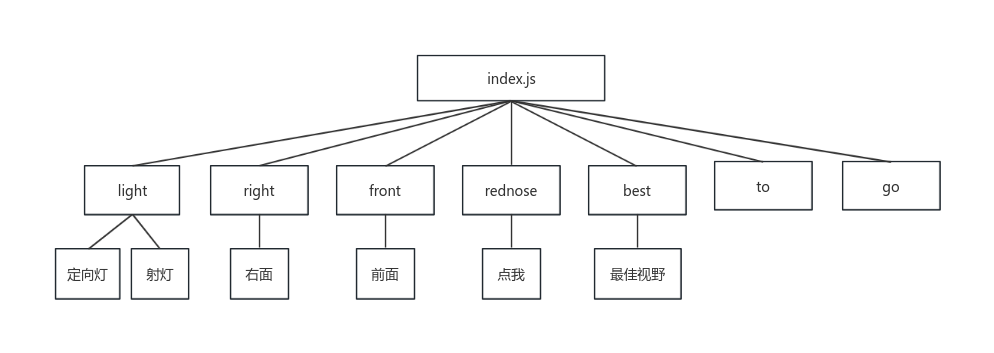
### 3.1.1 index.js主控模块

调用wqMV库和x3DOM框架，修改html文件，实现“点我”，“前面”，“右面”，“最佳视野”，“定向光”，“射灯”按钮的功能，通过鼠标左键点击产生3D模型颜色变化，大小变化和方向变化。

数据常量为一个名为wxy的对象。数据存储在index.js文件中。

首先创建一个wxy的常量对象，然后在该对象中定义了“light”，“best”，“right”，“front”，“rednose”，“to”六个元素，每个元素为一个函数。其中，当鼠标点击对应按钮时，对应的函数会调用q接口获得相应数据进行操作，从而改变实物模拟的deer模型的方向，颜色，大小。

其设计图如图所示：



### 3.1.2 wqMV库

其功能类似于jQuery API，主要是HTML文档的DOM遍历和操作，以及事件处理。其中q接口：quick，基于DOM操作及事件处理；m接口：Model，基于文档的排版及校验、日期等操作。

输入数据为程序中用到的节点id以及属性名称，字符串形式，输出数据为节点的属性的变化。数据存储为wqMV.js文件中。

其中q子模块为一个匿名自执行函数，其中包括q函数和对push还有join指令的简化操作。其中q函数实现了四个方法分别为

g:设置或读取属性值；

t:获取文本数据；

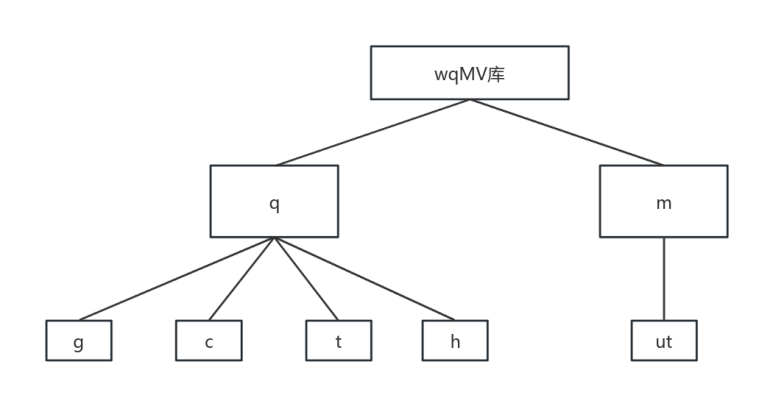
c:设置或读取classname；

h:设置或读取html数据。

并且对q函数的三个参数进行了矫正，另外运用判断语句对节点进行筛选，还实现了节点数组到数组的转换以方便对数组进行for循环操作。

m子模块也是一个匿名自执行函数，其中包括m常量对象，它的ut函数元素实现了年月日时分秒的获取，并对显示位数及中文汉字的显示进行了修正。

其设计图如图所示：



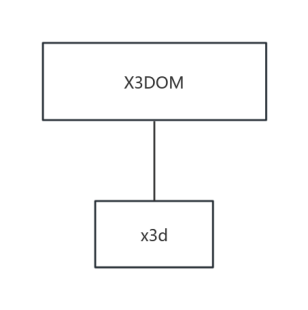
### 3.1.3 x3DOM框架

实现3d模型的呈现，缩放，旋转，渲染，实物模拟。

输入数据为鼠标输入设备或者x3d文件。有效性验证：x3d文件需符合XML标准。输出数据为3D模型的大小，方向，颜色的变化。

数据文件为js和css文件。保存的文件名为x3dom.css和x3dom.js。

其设计图如图所示：



### 3.1.4 css样式表

用于设计网页样式。

Root变量：

--ct: rgba(0, 0, 0, .8); /\* Color Text \*/

--cu: #08e; /\* Color blUe \*/

--gw: #fff; /\* backGround White \*/

--gh: azure; /\* backGround Hover \*/

--bdc: rgba(99, 99, 99, .382); /\* BorDer color \*/

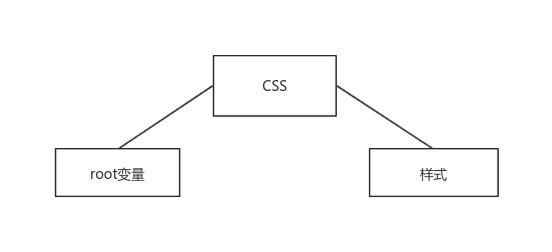
--bds: 1px solid var(--bdc); /\* BorDer Solid \*/

--bdd: 1px dotted var(--bdc); /\* BorDer Dotted \*/

--br: 3px /\* Border Radius \*/

输出网页的背景颜色，样式，按钮的样式以及划过时的变化。

其设计图如图所示：



### 3.1.5 index.html主界面

用于呈现整个网页。

输入数据为通过鼠标获取的数据，表现形式为滚轮，拖拽，以及html语言设计的网页文本数据。

输出浏览器网页形式。

## 3.2实现步骤

### 3.2.1 基础开发

1.创建开发环境，本例使用Microsoft IIS（已集成在Windows的任何最新版本中）作为Web服务器。

2.创建默认文档，index.html：

<!DOCTYPE html>

<html lang="zh-cmn-Hans">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">

<!-- 引入 X3DOM 框架 CSS 文件 -->

<link rel="stylesheet" type="text/css" href="x3dom/x3dom.css">

<title>Web 3D</title>

</head>

<body>

<h1>Hello, Web 3D !</h1>

<p>

This is my first html page with some 3d objects.

</p>

<!-- 引入 X3DOM 框架 JS 文件 -->

<script type="text/JavaScript" src="x3dom/x3dom.js"></script>

</body>

</html>

3.添加一个 3D 内容。首先，我们需要一个 X3DOM 元素，该元素描述了将在其中显示场景的 X3DOM 上下文。

在 <body> 标签内，添加以下代码：

<x3d width='800px' height='400px'>

<scene>

<shape>

<box></box>

</shape>

</scene>

</x3d>

4.更改材质颜色。现在有了一个充满白色的新区域，它是 X3DOM 的 3D 上下文。它的宽度和高度与 x3d 标签中定义的相应属性相匹配。

但是，添加的框不可见（或很难看到浅灰色边框）。这是由于没有向盒子分配任何材料，使用了默认的白色材料，由于页面的背景也是白色，因此该框不可见。

要更改材质颜色，需要在 shape 节点中添加一个 appearance 子节点。然后在这个子节点中再定义一个 material 节点。

它的 diffuseColor 属性，可定义简单的材料，并使用单一颜色表示形状的整个表面。

<shape>

<appearance>

<material diffuseColor='1 0 0'></material>

</appearance>

<box></box>

</shape>

5.打开或刷新 index.html 文件，查看效果。通过按住鼠标左键并四处移动鼠标从各个方面查看。按住鼠标右键或者通过鼠标滚轮来放大和缩小。

6.向场景中添加另外两个对象，一个蓝色球体和一个绿色圆锥体。除了sphere 和 cone 节点外，还需要将球体和圆锥体移到场景中相应的位置。和其他一些与图形相关的标准（例如 OpenGL ）中使用的坐标系一样，X3DOM 使用 transform 节点的 translation 属性来定义坐标。

<shape>

<appearance>

<material diffuseColor='1 0 0'></material>

</appearance>

<box></box>

</shape>

<transform translation='-3 0 0'>

<shape>

<appearance>

<material diffuseColor='0 1 0'></material>

</appearance>

<cone></cone>

</shape>

</transform>

<transform translation='3 0 0'>

<shape>

<appearance>

<material diffuseColor='0 0 1'></material>

</appearance>

<sphere></sphere>

</shape>

</transform>

### 3.2.2 动画

1.创建一个 timesensor ，它将开始并循环播放动画。属性 cycleInterval 以秒为单位指定动画的持续时间。循环属性的值设置为 true 。PositionInterpolator 用来定义关键帧。key 属性定义 keyFrames 的时间，而 keyValue 定义帧在坐标（x，y，z）中的位置。

在例中，需要两条路线。第一条路线将 positionInterpolator 绑定到 timeSensor，第二条路线将球体绑定到 positionInterpolator。

<timesensor id="wAnimate" DEF="time" cycleInterval="2" loop="true">

</timesensor>

<positioninterpolator DEF="move" key="0 0.3 1" keyValue="0 -3 0 0 3 0 0 -3 0">

</positioninterpolator>

<route fromNode="time" fromField="fraction\_changed" toNode="move" toField="set\_fraction">

</route>

<route fromNode="move" fromField="value\_changed" toNode="ball" toField="translation">

</route>

### 3.2.3 中级进阶

1.开发 DOM 操作引擎库，本例取名为：wqMV.js，存放在 wqMV 文件夹下。

其功能类似于 jQuery API ，主要是 HTML 文档的 DOM 遍历和操作，以及事件处理。

使用 ImageTexture 标签来渲染，例如，利用松鼠来渲染正方体。

<shape>

<appearance>

<ImageTexture url="img/squirrel.jpg"> </ImageTexture>

</appearance>

<box></box>

</shape>

（1）q 模块：quick ，基于 DOM 操作及事件处理。

q('idname') 获取 id 为 idname 的网页元素节点；

q('idname').h 获取或者设置 id 为 idname 网页元素的 innerHTML ；

q('.class') 获取 className 为 class 的所有节点；

q('@LI') 获取元素标签为 li 的所有节点；

（2）m 模块：Model ，基于文档的排版及校验、日期等操作。

m.ut() 获取当前日期；

2.创建页面的层叠样式表（ CSS ）文件，本例取名为：wqMV.css，存放在 wqMV 文件夹下。其功能是配合 wqMV 引擎，对网页中元素位置的排版进行精确控制，对网页对象和模型样式提供编辑。

3.创建页面主控程序文件，本例取名为：index.js，存放在 js 文件夹下。

4.修改 index.html 内容如下：

<!DOCTYPE html>

<html lang="zh-cmn-Hans">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">

<!-- 引入 X3DOM 框架 CSS 文件 -->

<link rel="stylesheet" type="text/css" href="x3dom/x3dom.css">

<!-- 引入 wqMV 框架 CSS 文件 -->

<link rel="stylesheet" type="text/css" href="wqMV/wqMV.css">

<title>Web 3D</title>

</head>

<body>

<section class="df dc d1 dy p5">

<div class="df p5 vc Mm">

<h1>Hello, Web 3D !</h1>

<p class="p5">This is my first html page with some 3d objects.</p>

</div>

<!-- 基础开发 -->

<div class="df dc m50 Mm">

<div class="bb p51"><span>基础开发</span></div>

<div class="d1 p51">

<x3d width="100%" height='400px'>

<!-- 这里是基础开发模块中的内容 -->

</x3d>

</div>

</div>

</section>

<!-- 引入 X3DOM 框架 JS 文件 -->

<script type="text/JavaScript" src="x3dom/x3dom.js"></script>

<!-- 引入 wqMV 框架 JS 文件 -->

<script type="text/JavaScript" src="wqMV/wqMV.js"></script>

<!-- 引入 主控 JS 文件 -->

<script type="text/JavaScript" src="js/index.js" defer></script>

</body>

</html>

### 3.2.4 高级渲染

1.使用图片渲染。

使用 PNG、JPG 对静态纹理数据进行编码。JPG 的内存配置文件较低，但压缩方式有损，并且不支持alpha通道。PNG压缩是无损的，可以处理Alpha。

使用 imagetexture 节点进行渲染，例如使用松鼠图片渲染一个正方体。

<shape>

<appearance>

<imagetexture url="img/squirrel.jpg"></imagetexture>

</appearance>

<box></box>

</shape>

2.使用视频渲染。

解决方案是将视频内容编码为 MP4 和 OGV 电影，并在 movietexture 节点中提供其来源。

由于目前流行浏览器禁止自动播放视频，所以可能视频无法播放。如果想看到自动播放效果，请使用谷歌浏览器 49 版本。

例如使用兔子影片渲染正方体。

<shape>

<appearance>

<movietexture url="img/bunny.ogv"></movietexture>

</appearance>

<box size="3,3,3">

</shape>

3.声音。

可以使用 WAV、MP3 和 OGG 作为声源。在 audioclip 节点中提供其来源。

由于目前流行浏览器禁止自动播放声音，所以可能听不见声音。如果想听到效果，请使用谷歌浏览器 49 版本。

例如加载声音片段。

<sound>

<audioclip loop="true" enabled="true" url="img/sound.mp3"></audioclip>

</sound>

### 3.2.5 实物模拟

1.x3d 格式。

X3D 是一种专为万维网而设计的三维图像标记语言。全称可扩展三维（语言），是由 Web3D 联盟设计的，是 VRML 标准的最新的升级版本。

X3D 基于 XML 格式开发，所以可以直接使用 XML DOM 文档树、XML Schema 校验等技术和相关的 XML 编辑工具。目前 X3D 已经是通过 ISO 认证的国际标准。

可使用 Maya、Blender、3DsMax 或者 World of Warcraft 等软件来建立 X3D 类型的 3D 模型文件。

2.网站添加 x3d 格式 MIME 类型。修改网站根目录下的 Web.config 文件，添加如下内容：

<system.webServer>

<staticContent>

<mimeMap fileExtension=".x3d" mimeType="model/x3d+xml" />

</staticContent>

</system.webServer>

3.加载 x3d 模型文件。

使用 inline 节点加载外部 x3d 文件。这些文件可能包含简单的对象，较复杂的对象甚至整个场景，而且仍然可以操纵其中的所有部分。

<inline url="x3d/deer.x3d" render="true"></inline>

4.调整视野。

一个场景可以包含任意数量的视点，但只能激活一个视点。要激活视点，只需将 set\_bind 参数设置为 true，摄像机将在视图之间自动动画。

<viewpoint id="wBest" position="-1.94639 1.79771 -2.89271"

orientation="0.03886 0.99185 0.12133 3.75685" description="camera" set\_bind="true">

</viewpoint>

5.光源渲染。

如果想让场景显得更明亮，可以设置定向灯，点光源和聚光灯。

白色的定向灯：

<directionallight id="wDirectional" direction="0 -1 0" on="true" intensity="2.0"

shadowintensity="0.0" color="1,1,1" shadowmapsize="1024" znear="-1" zfar="-1"

shadowcascades="1" shadowsplitfactor="1" shadowsplitoffset="0.1">

</directionallight>

蓝色的射灯：

<spotlight id="wSpot" on="true" beamwidth="0.9" color="0 0 1" cutoffangle="0.78"

location="0 0 12" radius="22" intensity="1" shadowmapsize="1024" znear="-1" zfar="-1"

direction="0,0,-1" attenuation="1,0,0" shadowcascades="1" shadowsplitfactor="1"

shadowsplitoffset="0.1"></spotlight>

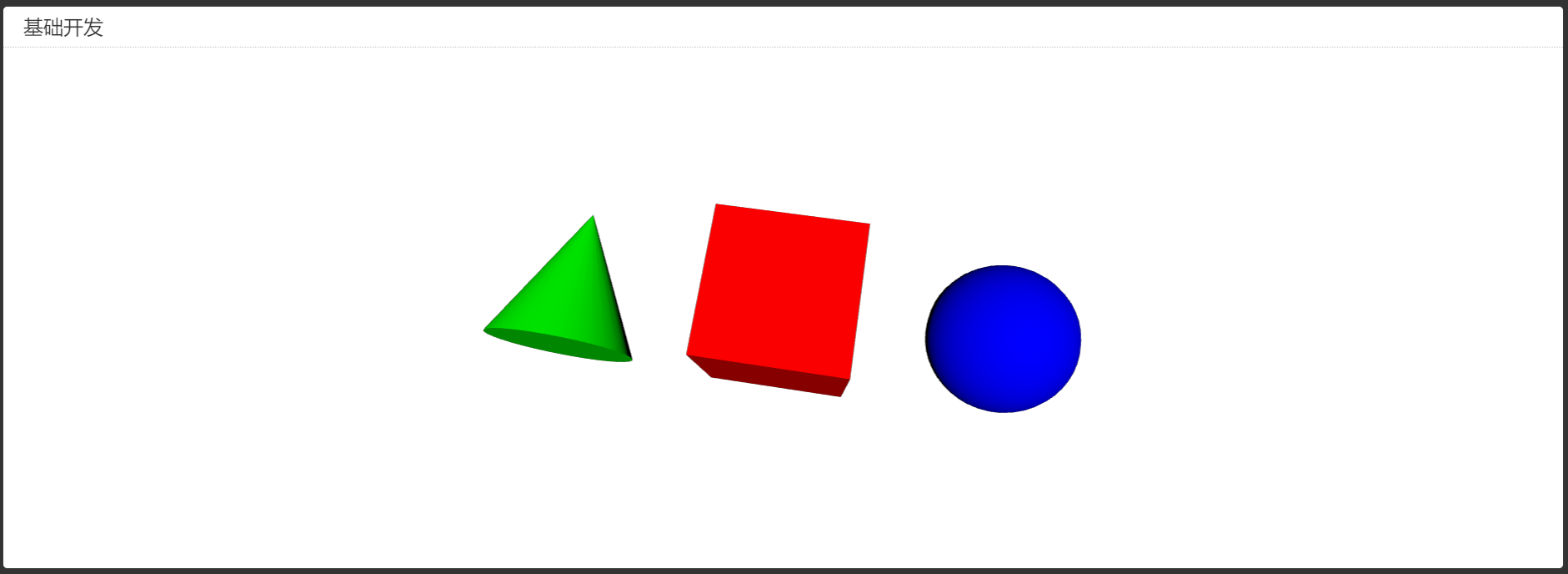
# **功能展示与评测**

## 4.1功能展示

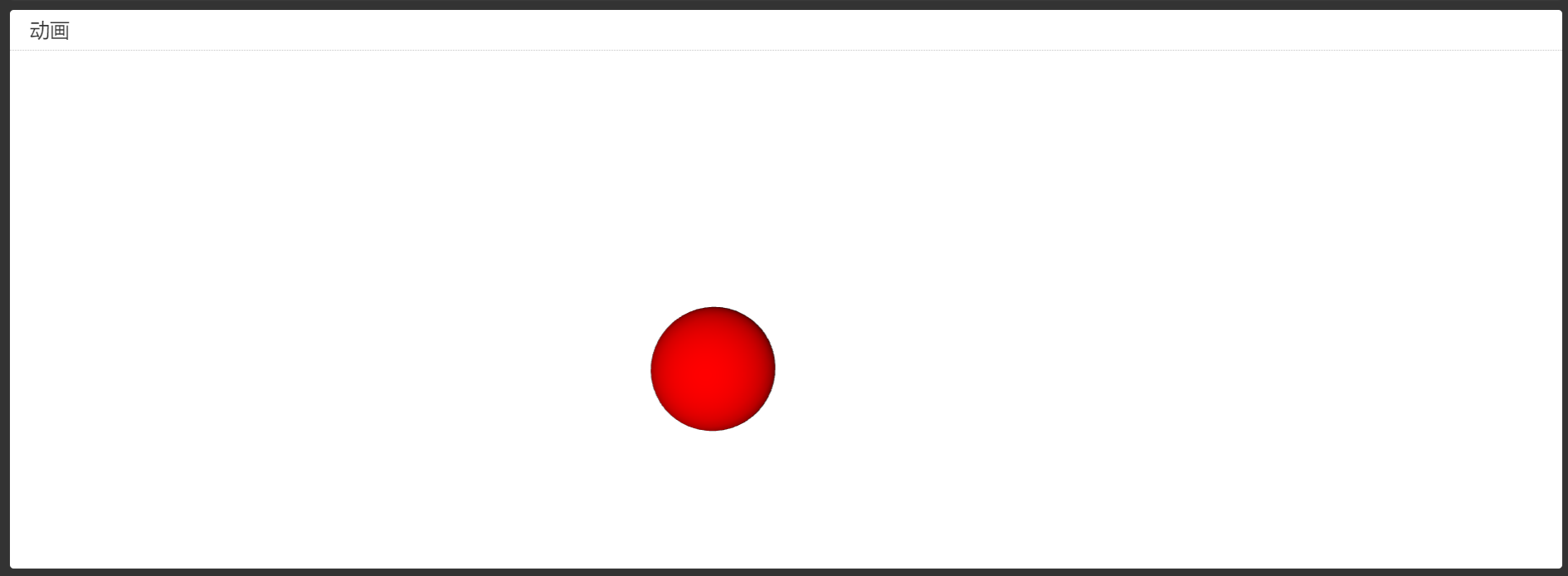
1.日期显示：



2.基础开发（可使用鼠标进行旋转缩放）：



3.动画（小球弹跳效果，可使用鼠标进行旋转缩放）：

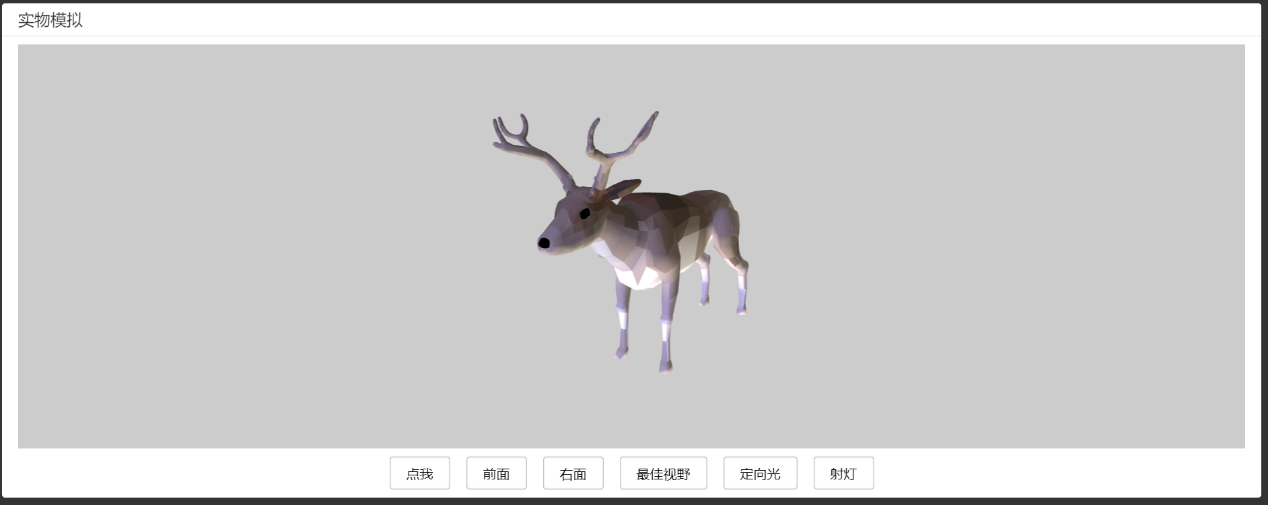


4.高级渲染（可使用鼠标进行旋转缩放）：

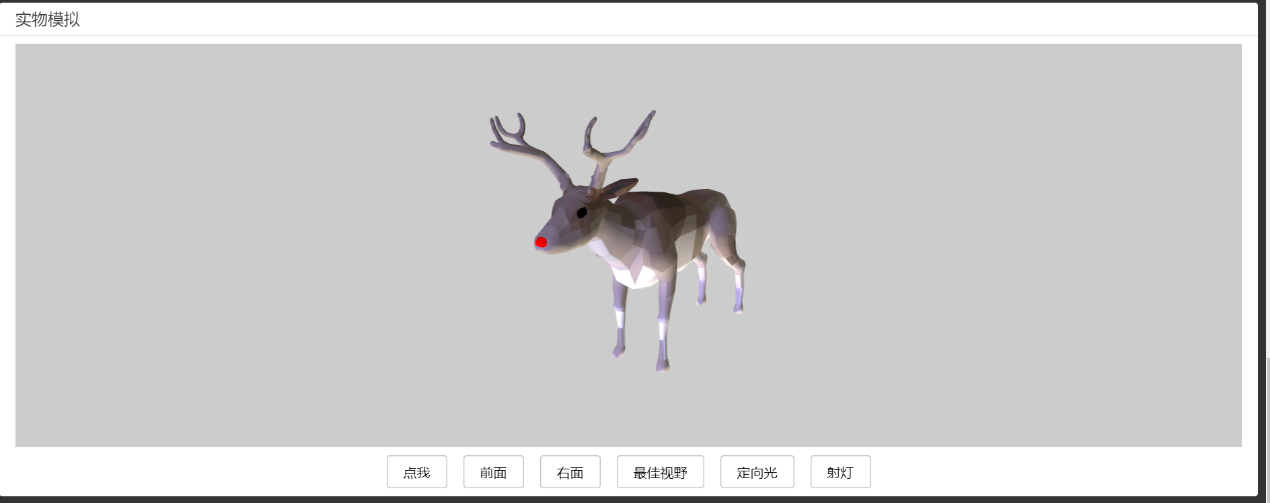


5．实物模拟（可使用鼠标进行旋转缩放）：

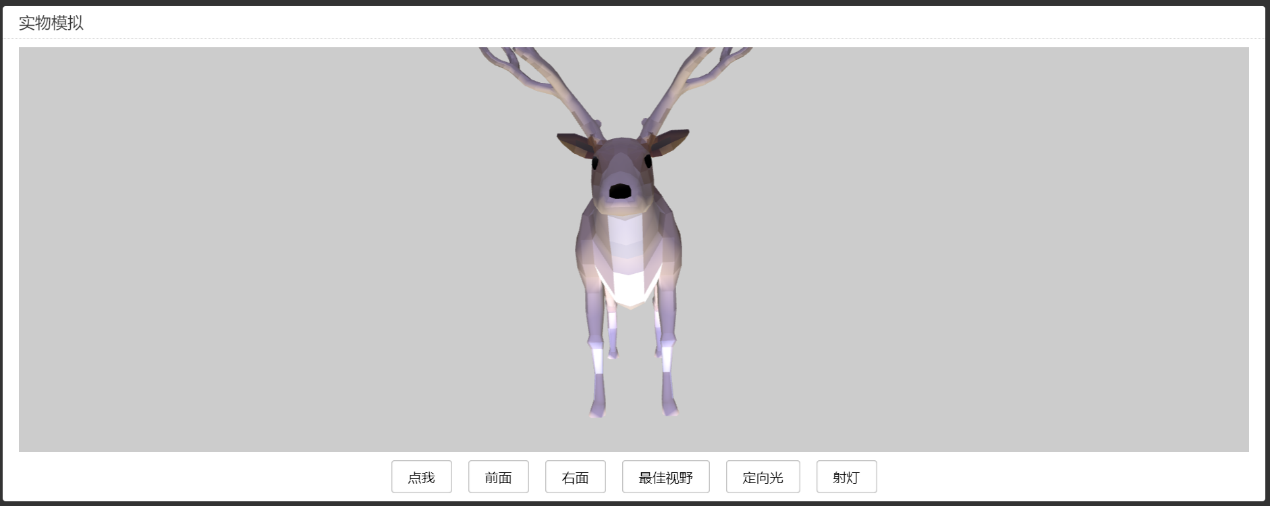
（1）默认展示最佳视野，定向光和射灯打开：



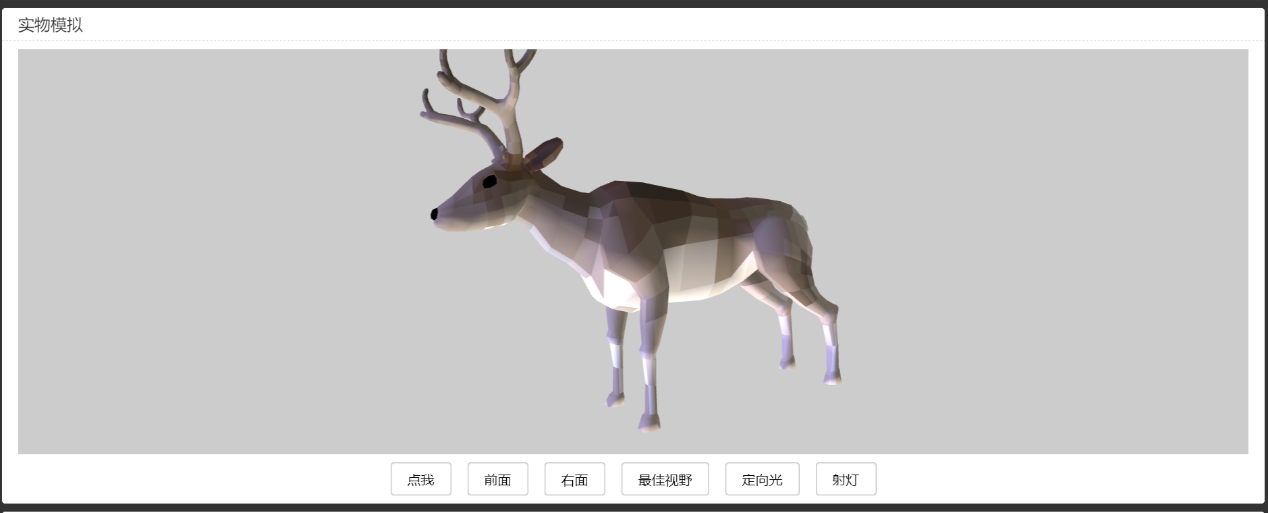
（2）点我（鼻子变红）：



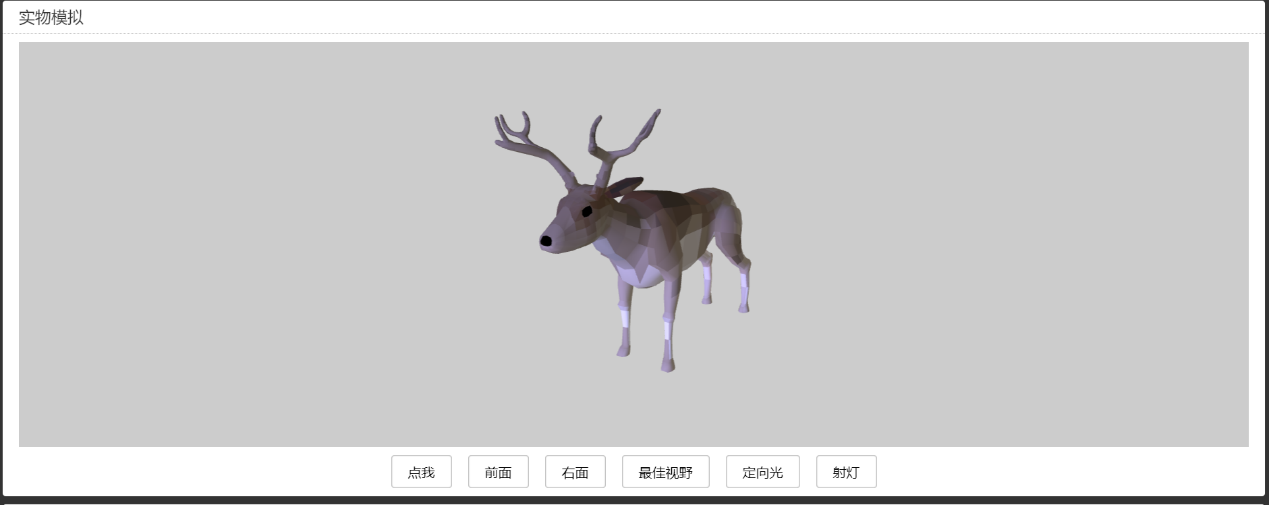
（3）前面：



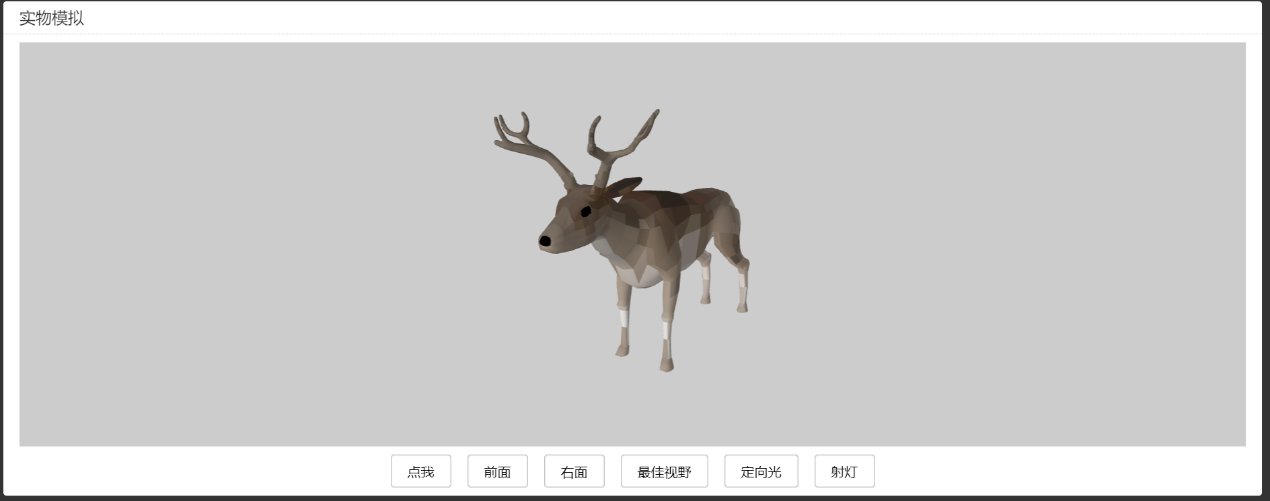
（4）右面：



（5）定向光关闭：



（6）射灯关闭：



## 4.2系统评估

1.系统功能完备，所设计的日期显示、基础开发、动画、高级渲染、实物模拟部分均已实现，系统可用性高，但由于目前流行浏览器禁止自动播放视频和声音，所以可能导致高级渲染部分无法正常显示，如果想看到自动播放效果，可以使用谷歌浏览器 49 版本。

2.出错信息：可能网络有延迟和卡顿。补救措施：

当html页面加载完成后，有一个127ms的缓冲时间，然后执行index.js文件。

to: () => {

const ut = m.ut();

q("wTime").h = ut.cn

},

go: () => { setTimeout(wxy.to, 127) }

# **总结**

本项目设计了一个基于X3DOM框架的Web网站，实现了Web3D建模，采用标准的浏览器技术，在HTML DOM中实现一个实时的X3D场景，使用结构化的文本表示形式来创建和显示交互式3D场景，使用户仅添加，删除或更改DOM元素就可以操纵3D内容，不需要特定的插件或插件接口，使3D内容就像文本、链接、图像或电影一样，成为网页内容的一部分。输入该3D物体的图形参数，颜色，经过3D建模，且通过JavaScript编程，可以实现利用鼠标对该模型进行的操纵，包括对该3D模型进行拖拽，旋转，缩放。本项目是一项独立的模块，而且全部内容自含，应用场景可用于网络购物平台商品的3D展示，从而给消费者带来更直观形象的购物体验。

本项目也有一些不足之处，如高级渲染部分由于浏览器版本限制可能无法正常显示，因此今后可进一步优化，例如将自动播放改为按钮播放等等，以此实现更好的效果。