高级 web 作业 2 文档
13302010007 王颖洲

项目简介：
基于 webgl 的 3d 技术和 websocket，实现的是一个多人聊天模型查看的功能。每个用户可以对模型进行查看和改变视角，通过“画布同步”按钮让其他成员共享自己的视角，同时，每个用户可以输入消息，由 websocket 向其他用户传达。

项目功能：
1 聊天室功能
    1 新用户登录时，对全体用户广播
    2 每个用户可以对其他用户发送消息
2 画布功能
    1 3d 模型绘制
    2 鼠标拖动旋转
    3 画布同步功能

技术简介：
1 webgl，canvas

WebGL 是一项使用 JavaScript 实现 3D 绘图的技术，浏览器无需插件支持，Web 开发者就能借助系统显卡（GPU）进行编写代码从而呈现 3D 场景和对象。

WebGL 基于 OpenGL ES 2.0，OpenGL ES 是 OpenGL 三维图形 API 的子集，针对手机、平板电脑和游戏主机等嵌入式设备而设计。浏览器内核通过对 OpenGL API 的封装，实现了通过 JavaScript 调用 3D 的能力。WebGL 内容作为 HTML5 中的 Canvas 标签的特殊上下文实现在浏览器中。

WebGL 标准由科纳斯组织（Khronos Group）开发和维护，Google、Mozilla、Opera 和 Apple 等浏览器厂商都是其中的成员，为这一标准做出了显著贡献。

WebGL 是一种 3D 绘图标准，这种绘图技术标准允许把 JavaScript 和 OpenGL ES 2.0 结合在一起，通过增加 OpenGL ES 2.0 的一个 JavaScript 绑定，WebGL 可以为 HTML5 Canvas 提供硬件 3D 加速渲染，这样 Web 开发人员就可以借助系统显卡来在浏览器里更流畅地展示 3D 场景和模型了，还能创建复杂的导航和数据视觉化。显然，WebGL 技术标准免去了开发网页专用渲染插件的麻烦，可被用于创建具有复杂 3D 结构的网站页面，甚至可以用来设计 3D 网页游戏等等。

WebGL 完美地解决了现有的 Web 交互式三维动画的两个问题：第一，它通过 HTML 脚本本身实现 Web 交互式三维动画的制作，无需任何浏览器插件支持;第二，它利用底层的图形硬件加速功能进行的图形渲染，是通过统一的、标准的、跨平台的 OpenGL 接口实现的。

Canvas 和 img 等标签一样，是一个可以自由制定大小的矩形区域。
通过 javascript 可以对矩形区域进行操作，可以自由的绘制图形，文字等。而且，可以添加影子，进行涂色，另外还可以对绘制的图形进行旋转等操作。而且，Web 上常用的 gif,jpg,png 等格式的图片，也可以直接进行绘制。但是，能够管理操作动画等处理的对象和方法是没有的。也就是说，为了制作动态的应用，通常会使用 javascript 进行循环处理。

利用 WebGL 可以模拟三维空间，但是最终必须输出显示在一个二维的显示器上。由深度决定的前后关系，根据远近进行放大和缩小，这些都必须提前进行运算得出结果。

使用程序来模拟三维空间的时候，最终的情报必须变换成二维数据。而且三维坐标，根据平台不同，Z 轴的处理是不一样的。WebGL 是 OpenGL 的处理系，使用的是右手坐标系。为了模拟三维空间，将三维空间的情报向二维的情报进行转换，需要三个坐标变换。分别是模型变换，视图变换和投影变换，将这些变换进行组合，最终决定描画的图形内容。

顶点中的属性是由程序员来自由添加的，需要的 VBO 的个数就是添加的属性个数。顶点属性的各个数据，使用纯粹的一维数组，当然，数组的元素个数要根据想要绘制的顶点个数来定义。

生成 VBO 的时候，首先要把缓存绑定到 WebGL，然后相应的数据，要转换为相应的类型，然后使用指定的常量来写入数据。而为了避免预想之外的错误发生，数据写入结束之后，要将 WebGL 中绑定的缓存无效化。这样，一连串的处理之后，模型数据就可以被顶点着色器利用了。

2 websocket

在浏览器中通过 http 仅能实现单向的通信,comet 可以一定程度上模拟双向通信,但效率较低,并需要服务器有较好的支持; flash 中的 socket 和 xmlsocket 可以实现真正的双向通信,通过 flex ajax bridge,可以在 javascript 中使用这两项功能. 可以预见,如果 websocket 一旦在浏览器中得到实现,将会替代上面两项技术,得到广泛的使用.面对这种状况，HTML5 定义了 WebSocket 协议，能更好的节省服务器资源和带宽并达到实时通讯。

在 JavaEE7 中也实现了 WebSocket 协议。

在实现 websocket 连线过程中，需要通过浏览器发出 websocket 连线请求，然后服务器发出回应，这个过程通常称为 “握手” (handshaking)。

现很多网站为了实现即时通讯，所用的技术都是轮询(polling)。轮询是在特定的的时间间隔（如每 1 秒），由浏览器对服务器发出 HTTP request，然后由服务器返回最新的数据给客户端的浏览器。这种传统的 HTTP request 的模式带来很明显的缺点 – 浏览器需要不断的向服务器发出请求，然而 HTTP request 的 header 是非常长的，里面包含的有用数据可能只是一个很小的值，这样会占用很多的带宽。而比较新的技术去做轮询的效果是 Comet – 用了 AJAX。但这种技术虽然可达到全双工通信，但依然需要发出请求。

在 WebSocket API，浏览器和服务器只需要做一个握手的动作，然后，浏览器和服务器之间就形成了一条快速通道。两者之间就直接可以数据互相传送。

环境配置：

1 j2ee eclipse mars

2 Tomcat7.0

3 tomcat-juli.jar，util，websocket-api.jar

4 在 eclipse 中将项目部署到本地 Tomcat 服务器上，启动即可运行

5 地址为 http://localhost:8080/chat/websocket/chat.html

实现过程：

1 chat.html，绘制 canvas,input,button,div

2 chat.css，每个组件的样式

3 chat.js，分为 webgl 代码和 websocket 代码。webgl 不断重绘 canvas，新建矩阵处理旋转；websocket 准备传输数据，准备接收数据，实现对数据的处理。

4 chatAnnotation.java，接收数据，并实现不同的方法，将数据广播至所有用户端

使用截图：
Guest0,guest1 为两个用户加入
Bfsync 和 aftsync 为两个用户同步画布
Chat 为两个用户发送信息
Newuser 为第三个用户加入，自动将最新的画布信息同步
Bfsync1 和 aftsync1 为三个用户同步画布
Userleft 为某一用户离开
Close 为 socket 关闭，同时移除 html 组件的事件监听