Design of Real Time Barrage Video System for Mobile Terminal

Zhao Jiantao  
School of Control and Computer Engineering  
North China Electric Power University  
Beijing, China  
zhaojiantao@eplian.com

Li Shuang  
School of Control and Computer Engineering  
North China Electric Power University  
Beijing, China  
lishuang@eplian.com

Abstract—Analyzed the current use and development of Barrage Comment Video Player on various video sites. According to the source of barrage video core technology, HTML5, CSS, JavaScript and so on [1], opposite to the mobile terminal and able to cross platform, we analyzed and designed a real time barrage video system. It is important and valuable for anyone who wants to add more fun and interactive elements into social and multi-media applications to attract more users. And we hope it would be more meaningful for practical use and gain larger scope in related areas.

Keywords—barrage comment video player; HTML5; CSS; JavaScript; mobile terminal

I. INTRODUCTION

Nowadays the major video sites have the corresponding comment function extended to video, the sliding subtitles way in real time video screen [2]. In this way, all the viewers will pay attention to, so as to realize the interaction between the viewers. Even with the expression of praise or criticism works, more fun to watch. With the popularity of mobile intelligent devices, the barrage video system with the major video sites in the mobile version of the client, enter the user's mobile device. Most of the mobile terminal barrage system at present is limited to special video client, cannot be embedded to the browser. This led to the mobile device according to the different types, different operating system. This requires the development of a corresponding one by one barrage play system, increase the development and maintenance work [3].

This paper introduces the design of a player in the mobile terminal real-time barrage playing system. Its function is similar to some of the video player based on FLASH. Can truly cross platform use. And can be easily integrated in any need to add functionality barrage video playback module. It reduces the coupling degree in the process of developing mobile applications [4]. And it is suitable as a component in a mobile middleware platform for rapid deployment.

II. KEY TECHNOLOGIES

A. Core of the barrage video player

The core of the barrage video player, called The CommentCoreLibrary, is a set of original barrage controller based on JavaScript, which is aimed to develop the barrage player in HTML5 [5]. At the same time, it is convenient for those who want to know about the operation principle of barrage player. And it can provide a simple but thorough introduction. Developers would be able to customize their own streaming media comments play mode according to the core component of the barrage player. Wherever you are in the development of JavaScript based Web services, and other services to the barrage play function. You can refer to the core of the implementation code of barrage player. This core construct the following functions in some simple ways. (1) time management. (2) the basic spatial planning. (3) the filter of the barrage. (4) senior barrage effect. (5) the basic format. (6) support code barrage. This core is derived from ABPlayerHTML5, and is used by it now. The design of this article is based on the open source core, CommentCoreLibrary, from GitHub.

The core of barrage video player mainly includes the following parts: the CommentManager, the CommentSpaceAllocator and the IComment.

The main function of the CommentManager is to control the barrage. The attribute “options”, is used set the default parameters. The attribute “global”, is the global attribute...
collection of the barrage. The “scroll” is the rolling barrage attribute set. The “opacity” is the transparency of the barrage. The “scale” is used to change the life time of the barrage. As follows:

```javascript
function CommentManager(stageObject){
    var __timer = 0;
    this._listeners = { }; 
    this.stage = stageObject;
    this.options = {
        global: {
            opacity:1,
            scale:1,
            className:"cmt"
        },
        scroll: {
            opacity:1,
            scale:1
        },
        limit: 0
    };
    .......
}
```

The CommentManager provides varies of basic methods for barrage controlling. The method “init ()”, is used to initialize The CommentManager and in the first time bind the scale of the stage. At the same time, initialize the filter, judge whether a barrage is shown or not according to the content of it. Also this function can control the operating way of a barrage. After the initialization, by calling the “start ()” method to start the process of barrage. After the call “send ()”, the barrage which have been sent to run list (runline) would begin to move. For just after the “init ()” and enter the state of play, or after the pause to restart the moving of barrage. It would be invalid when calling on state of playing. Another important method is time(currentTime:Number). It can notify the current time base. The manager will automatically deal with the case on forward or back of time. Including clearing the running barrage on the screen when needed. In here, “currentTime” means the real time referring to “stime” of a barrage. Time in milliseconds (MS). “time” will only put interrelated barrage into “runline” (run list). As for the barrage is on the move or not, according to the current state of the manager ——“isRunning”. “Load” (timeline:Array) method is used to load some abstract barrage object as the time axis. There is no need to sort these barrage objects. The manager will automatically sort them according to “stime”. Since `load` will clear the time axis before. It can’t be reload when playing. Because that would lead to incontinuous location pointer and strange bugs. We can do “time(0)” or “seek(0)” in advance to put the playing position back to 0. So here comes the question, how to change the barrage list dynamically. We designed two methods: insert () and remove (). “insert ()”method uses dichotomy to quickly find the position of the barrage object to be inserted. This makes the insert process dynamically. Method “send” (data:ICommentData) change an abstract data objects into a IComment and run into the list. When the data object does not conform to filter rules will not achieve send. The “send” method can send barrages which are out of time axis. This function is especially suitable for real-time barrage. Because of the broadcast real-time barrage, basically do not need to use the timeline. It would be more efficient to directly send out the just received barrages. Method “finish” (comment:IComment) is used to complete the barrage. Delete from the stage or space manager. A better way is to call the “finish ()” method from a barrage objects. In general the “finish” method of barrage class will call this after the destruction of related objects to release the resource manager. In addition, CommentManager is also designed with several common methods to manage events. Such as “addEventListener” (event:String, listener:Function), to add listener. It adds listener into corresponding event lists. “dispatchEvent” (event:String, data:Object) is used to dispatch events. Search the listener of specified event and pass the parameters to the methods of listener.

B. The Space Allocator of Comment

“CommentSpaceAllocator” is a space allocator mainly used to typeset the comment. Initialize the default width and height of 0. When setting the bound in the comment manager, method “setBounds()” locate width and height for comment in all kinds of locations. In methods of transmitting comments, according to the mode of each comment, the allocator adds it into specific list.

C. Abstract Comment Objects

The “IComment” can be expanded depending on the specific circumstances. So the properties and methods are very
The abstract comment object is easy to offer the support different romanced screens and complex trajectory. Under this abstract mode, a separate comment manager can easily support rendered to DOM, Canvas and WebGL levels. And it will be very easy to support new types of barrage comment. In order to increase the support of senior comment, after the parser, comments will be parsed as every independent object. These objects will have a series of their own properties. Here introduces the properties of comment object. Among the many attributes some are must, while others are optional. As shown in the following figure:

Figure 1. Some of the attributes

There must be attributes such as: text (refers to the content of comment), mode (refers to the type of comment, like rolling comment, top or bottom comment, backwards comment), stime (refers to the start time of comment relative to the video), size (the font size of comment) and color (the font color which is expressed by RGB).

The methods of abstract comment object are as follows: Method “constructor” (cm:CommentManager, data:Object), need the CommentManager to be the parent and “data” to fill the attribute. Method “init” (recycle:IComment = null) is used to initialize the comment. In the DOM version, the DOM object will be created only after initialization. Before the “init()”, position x and y cannot be read. Because it is uninitialized. You can give an old IComment to let the new one take over the original DOM of IComment (instead of constructing a new one). Of course, it will create a new rendering component if the attribute is not be provided. Method “time”(dt:Number), means forward dt milliseconds. This function will update the interface according to the situation. The comment controlled by CSS also need to be called [6]. Method “update()” can calculate the location of comments according to the “ttl” and “dur”, and render the comment. Different comments can override the update function to draw the canvas. Method “invalidate()” can invalidate the catch of comment space. Next time when it read width, height, x, y, right, bottom calculate will be forced to do again. Method “animate()” can animate the comment according to the “motion” object. Method “finish()” is used to notice that the comment is finished. Method “toString():String” returns debugging information.

III. DESIGN OF REAL TIME COMMENT SYSTEM

The real time comment system is made up by 3 parts. As follows:

- Sever: Listening for client connections and events of comment and response.
- Emit Client: Comments emitted by users.
- Screen Client: Receive and show comments.

Sequence diagram is as follows:

Figure 2. System sequence diagram

Real time barrage comment need the support of the backend server. And it will be more complicated than just emit comments. Real time comment can be implemented by using Polling or Push Notify. There are two kind of time mode, the absolute real time mode and the relative time axis mode [7].

Polling refers to the barrage player regularly access the server [8]. Ask if the sever has a new comment from the comment pool after a certain period of time. And then add it to the playlist or show. This is a kind of simple mode. The advantages are clear. It only based on HTTP, and can be implemented on all kinds of server and written by all kinds of programing languages. The disadvantages are as follows: First,
it needs to connect the server over and over. This seriously impact the efficiency of server. Here we recommend this mode in the system that real time is not strong.

Push Notify. When a client is connected to a port of a server and a new comment comes, the server actively send the message. Then after the message has been received, the client passively show or update list. The advantages are as follows: fast, highly efficient, low costs. Also the disadvantages are obvious. It need the Websockets or Flash to be the bridge. And the compatibility is not enough. Strongly recommend to use this mode into real time system such as live.

The difference are also in the mode of time axis.

Absolute real-time: whenever real-time comment is received then directly display. Nothing or a small amount of history will be saved. Not free to look back. A little memory taken.

Real-Time time axis: Regularly update the time axis. Insert the comment correctly in sequence. Keep the comment time axis fresh. Can be free to change the play time.

Here are two modes of some pseudo code:

```javascript
// Polling example code
var hasLastCheckReturned = true;
var lastCheckedTime = 0;
setTimeout(function(){
    if(!hasLastCheckReturned){
        return;
    }
    var xhr = new XMLHttpRequest();
    xhr.onreadystatechange = function(){
        if(xhr.readyState === 4){
            if(xhr.responseCode === 200){
                var danmakuList = yourFormatParser(xhr.responseText);
                for(var i = 0; i < danmakuList.length; i++){
                    CM.insert(danmakuList[i]);
                }
                lastCheckedTime = Date.now();
                hasLastCheckReturned = true;
            } else {
                hasLastCheckReturned = true;
            }
        }
    }
    xhr.open('GET', 'http://yoururl/somevideoid/?from=' + lastCheckedTime, true);
    xhr.send();
    hasLastCheckReturned = false;
}, 3000);
// Push notify example code
var socket = io();
socket.on('danmaku', function(data){
    var danmaku = yourFormatParser(data);
    CM.insert(danmaku);
});
```

IV. A INSTANCE OF THE PLAYER

According to the characteristics of the above, this paper implements a demo to show the function of the system directly.

This demo is based on Android platform [9]. Due to limitations of the experimental conditions, it can only be a simulation of real time environment. Here we use the multithreading mechanism of java to create many user processes on user client to act as there are a lot of users making comments. Each of them will send its own comments to the server. The server will handle the comments and their patterns. After packaging, caching and some other process, the server will send comments back to user’s device. Then the parser would be able to transform the packaged comments into the right form. After that the comments will be sent to the view to be shown on the client.

As the following figures, the default orientation is landscape. The demo has some basic functions such as rotation, hide or show comments, pause, resume, send a comment and send several comments.
When the button “hide” is pushed, the comments which are on the screen will be cleaned.

On the time that the button “send a comment” is pushed, the demo will send a default comment to the server. After a few milliseconds, the comment will appear on the screen. As the next figure, the one surrounded by a green rectangle is the comment just sent.

When pushing the “send several comments” button, the demo will send a lot of comments. Here we use the function “Timer()” of java to send a comment once every 20 milliseconds [10].

V. CONCLUSION

This paper proposes and designs a real-time barrage comment playing system. Through analyzing an open source and highly expandable core, analysis and summarizes the current domestic and foreign websites for their realization of the principle and process of the barrage comment system. As a result of the core is mainly composed of script language, it can be embedded into any browser. Truly realize the cross platform module. The system can be encapsulated into a video playing module, to expand its scope of application. For example, there is potential demand in some enterprise mobile information system for video playback.

REFERENCES

[1] Zhang Xuhong. Design and implementation of information system based on mobile platform [D]. Beijing Jiaotong University, 2012
[5] Huang Xiong. Research and design of video and audio transmission technology based on HTML5 [D]. Southwest Jiaotong University, 2014