



PLYmedia Inc.
JW FLV Player Integration Document
(Jeroen Wijering's FLV player)



Description

PLYmedia add-on is a transparent layer, loaded on top of a video, allowing you to create and display custom content synchronized with the stream, without altering the latter in any way.

Without the headache of video editing, you will be able to create and synchronize to your FLV stream dynamic content such as subtitles, images, SWF clip arts and more, all streaming from our servers in real-time.

This tutorial will accommodate the easiest way to integrate PLYmedia add-on with Jeroen Wijering's FLV player.

You can download the latest version of the player here:

http://www.jeroenwijering.com/upload/jw_flv_player.zip

Further information and FAQ can be found here:

<http://www.jeroenwijering.com/extras/readme.html>

PLYmedia add-on can be loaded from here:

<http://www.bubbleply.com/plyPlayer.swf>

Installation

PLYmedia add-on integration with JW player includes 4 main steps:

1. Defining a switch that will allow you to turn on/off the add-on at your convenience.
2. Defining an empty movie clip and loading the add-on.
3. Implementing a resize function when completing loading and going to full screen and back, to fit the video proportions at all time.
4. Sending the time parameter to the add-on in order synchronize it to the video.



You will need to make changes in three classes to implement those steps:

1. Open `MediaPlayer.as` in folder `com/jeroenwijering/players/`

In the parameters section you will find the definition of the `config` object.

Add a property to the object: `showPLYmedia:"true"`.

This will allow you to turn on/off the add-on at your convenience.

```
private var config:Object = {
    ...
    showPLYmedia:"true",
    ...
};
```

2. Open `FLVModel.as` in folder `com/jeroenwijering/players/`

In the parameters section, define parameter: `PLYObject:Object`.

```
public var PLYObject:Object;
```

In function `FLVModel` we'll define the movieclip that the add-on will be loaded into and call a function to handle the loading.

```
function FLVModel(vws:Array,ctr:AbstractController,cfg:Object,
    fed:Object,fcl:MovieClip) {
    ...
    videoClip.createEmptyMovieClip("snd",videoClip.getNextHighestDepth());
    soundObject = new Sound(videoClip.snd);
    if (config["showPLYmedia"]=="true") {

videoClip._parent.createEmptyMovieClip("PLYmedia",videoClip._parent.getNextHighestDepth());
        setPLYmedia(videoClip._parent["PLYmedia"]);
    }
};
```

Next we'll define a new function, `setPLYmedia`, that will handle the PLYmedia loading.

```
private function setPLYmedia(PLYloader_mc:MovieClip) {
    var mcLoader :MovieClipLoader = new MovieClipLoader();
```



```

var loadListener :Object          = new Object();
var ref = this;
loadListener.onLoadInit = function(target_mc:MovieClip):Void {
    target_mc.reSize(ref.config["displaywidth"],ref.config["displayheight"]);
    if (ref.currentURL!=undefined) target_mc.SitePath = ref.currentURL ;
}
mcLoader.addListener(loadListener);
mcLoader.loadClip("plyPlayer.swf",PLYloader_mc);
};

```

In function, *setStart*, we'll initiate the add-on by sending it the FLV URL parameter.

```

private function setStart(pos:Number) {
    ...
    if (flvType=="RTMP" && feeder.feed[currentItem]["id"] != currentURL) {
        connectObject.connect(feeder.feed[currentItem]["file"]);
        currentURL = feeder.feed[currentItem]["id"];
        setStreamObject(connectObject);
        streamObject.play(currentURL);
        if (config["showPLYmedia"]=="true") videoClip._parent.PLYmedia.SitePath =
currentURL ;
    } else if (flvType != "RTMP" && feeder.feed[currentItem]["file"] != currentURL) {
        connectObject.connect(null);
        currentURL = feeder.feed[currentItem]["file"];
        if (flvType == "HTTP" ) {
            setStreamObject(connectObject);
            if (config["streamscript"] == "lighttpd") {
                streamObject.play(currentURL);
            } else {
                streamObject.play(config["streamscript"] + "?file=" + currentURL);
            }
        } else {
            setStreamObject(connectObject);
            streamObject.play(currentURL);
        }
        if (config["showPLYmedia"]=="true") videoClip._parent.PLYmedia.SitePath =
currentURL ;
    } else {
        if (pos != undefined) { streamObject.seek(pos); }
        streamObject.pause(false);
    }
}

```



```

    }
    ...
};

```

In function, *updatePosition*, we'll send time updates to the add-on.

```

private function updatePosition() {
    ...
    if (pos != currentPosition) {
        currentPosition = pos;
        if (metaDuration < currentPosition) metaDuration = currentPosition;
        sendUpdate("time", currentPosition, metaDuration - currentPosition);
        if (config["showPLYmedia"] == "true")
            videoClip._parent.PLYmedia.updateTime(currentPosition);
    } else if (streamObject.bufferLength < mbl && stopFired == true) {
        currentState = 3;
        videoClip._visible = false;
        videoClip._parent.thumb._visible = true;
        sendUpdate("state", 3);
        sendCompleteEvent();
        stopFired = false;
    }
};

```

In function, *setPause*, we'll make an additional change in order to enable time update even when the player is in pause mode.

```

private function setPause(pos:Number) {
    ...
    if(pos != undefined) {
        currentPosition = pos;
        sendUpdate("time", currentPosition, Math.abs(metaDuration - currentPosition));
        if (config["showPLYmedia"] == "true")
            videoClip._parent.PLYmedia.updateTime(currentPosition);
        streamObject.seek(currentPosition);
    }
    ...
};

```



3. Open DisplayView.as in folder com/jeroenwijering/players/

In function, *setDimensions*, we'll handle the add-on's resize when full screen button is invoked.

```
private function setDimensions() {  
    ...  
    var tgt = config["clip"].display;  
    scaleClip(tgt.thumb,thumbSize[0],thumbSize[1]);  
    scaleClip(tgt.image,itemSize[0],itemSize[1]);  
    scaleClip(tgt.video,itemSize[0],itemSize[1]);  
    if (config["showPLYmedia"]=="true")  
    tgt.PLYmedia.resize(tgt.video._width,tgt.video._height);  
    ...  
};
```

Notes

- It's best to have JW player running at 24 Frames per Second. Any FPS under 12 may distort time accuracy and animation effects within PLYmedia add-on.
- Additionally, JW player must have this line of code embedded:
"System.security.allowDomain("http://www.bubbleply.com/")" in order to let the player communicate with the add-on.
- If the integration was successful you should be able to view the video with PLYmedia's menu.



Sample JW player with PLYmedia add-on integration:

The following link is to a sample JW player with the described PLYmedia integration:

http://www.plymedia.com/downloads/sample_jw_player.zip

PLYmedia information:

In order to add the ability to create, save & modify PLYs or professional subtitles to your site please contact our Technical department:

PLYmedia Inc.

support@plymedia.com