Creating searchable video with Adobe® Creative Suite® Production Premium

Turn spoken dialogue into keywords using Speech Search to make your video searchable

Table of contents

Powerful new Speech Search technology in Adobe Premiere® Pro CS4 and Adobe Soundbooth® CS4 transcribes spoken dialogue into text-based metadata that enables the video asset to be searchable on websites. This new capability turns any spoken word into a keyword that points precisely to the place in a clip where the word is spoken. This unleashes considerable power, for both post-production professionals and all of us who watch video online. During the editing process, use Speech Search to quickly find the relevant points in a particular clip or easily locate the right clip based on dialogue. Equally importantly, the time-accurate text that corresponds to spoken words is embedded in the output you render from Adobe Premiere Pro CS4 or Soundbooth CS4, so you can use ActionScript™ in Adobe Flash® CS4 Professional software to create video that is searchable in Adobe Flash Player.

The searchable video workflow

Search your assets to accelerate editing and find key sections instantly. When you render your video as an FLV or F4V, the keywords automatically travel with your clip. A customized SWF video player reads all this information, highlights keywords, and makes your video searchable—all you have to do is point the search term to your transcribed video.

This guide offers step-by-step instructions how to generate a searchable video online using Production Premium.

Generate a transcript in Adobe Premiere Pro

1. In Adobe Premiere Pro CS4, open the Metadata panel and import the media clips you wish to edit.
2. Select the clip you wish to transcribe first and choose Clip>Audio Options>Transcribe to Text from the drop-down menu.
3. In the dialog box, use the menus to select a language and quality setting. For the latter, select High (slower), and click OK to begin converting dialogue to text-based metadata.
4. Repeat the above steps for all your clips.

Adobe Media Encoder launches automatically, and your audio file appears in the source file column, while Speech Transcript appears as the target format. A progress bar across the bottom of the Adobe Media Encoder screen displays time remaining and other information related to its progress. When transcribing multiple clips, Adobe Media Encoder will batch process your files while you continue to edit in Adobe Premiere Pro. The transcription text is stored as timecode-

Adobe

accurate metadata. To edit the transcribed text in the Metadata panel, tab from word to word to correct, insert, combine, and delete words. This helps ensure that the timecode-accurate nature of the speech data remains intact.

Export using Adobe Media Encoder

Adobe Media Encoder, a separate software application, saves you time by automating the process of creating multiple encoded versions of your source files. You can set up multiple items for encoding, manage priorities, and control advanced settings for each item individually. Use any combination of sequences and clips as sources, and encode to a wide variety of video formats (see Figure 1).

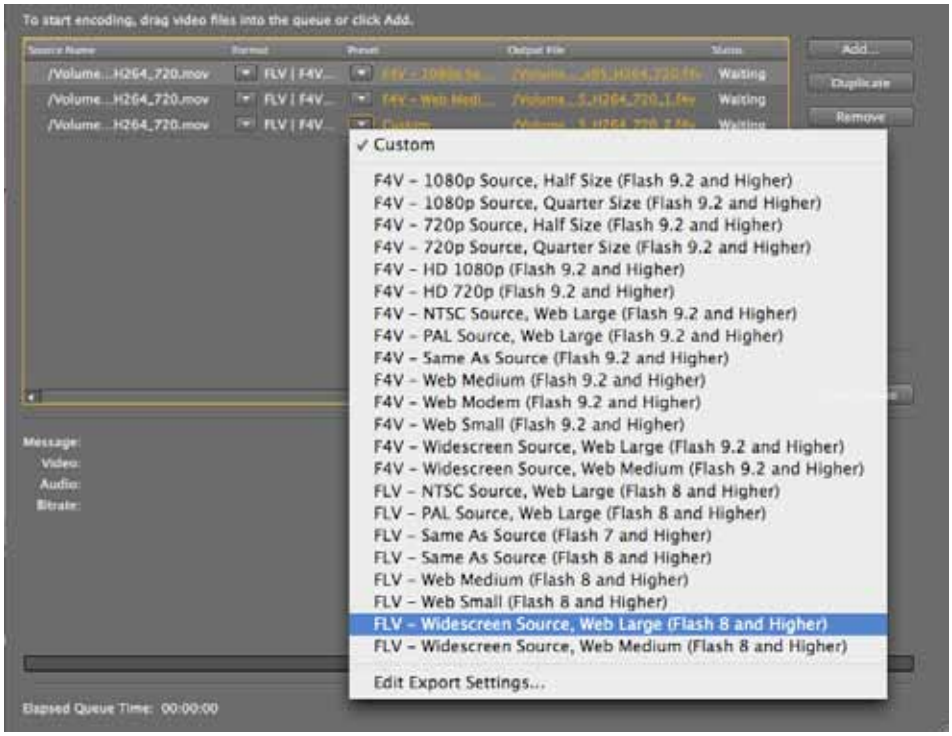

Figure 1. With Adobe Media Encoder, you can automate the process of creating multiple encoded versions of your source files and sequences and encode to a variety of video formats.

After you've edited your video in Adobe Premiere Pro, export the file to Adobe Media Encoder in FLV or F4V format.. The transcript/metadata travels with the video. Although you cannot see the transcript, it's inherent in the file. Place the video file in a Media folder on your desktop.

Export an .XML file in Soundbooth

After you have exported your Flash video, you want to import it into Soundbooth CS4 to create and eXtensible Markup Language (XML) file. An XML is a standardized language used by Web developers for marking up documents and data. The XML file you generate in Soundbooth enables you to create customized tags, such as cue points, that you could not create if you were working in HTML alone. In Soundbooth, open the transcribed video and choose File>Export> Speech Transcription. Save that file in the Media folder as well.

Create a customized video player

To create a customized video player, you need to have a fundamental knowledge of object-oriented programming, particularly ActionScript 3. How you organize the code in the following sections is really up to your architecture preference, but these are the code snippets you need to create the video player. These snippets are the fundamentals to extracting cue points and associating them to the video player seek bar. If you're comfortable with Ajax, CSS, and HTML, you can do much more with the external interface class and to interact with those technologies.

Loading and streaming/progressive video

In this example, we use the progressive approach—the NetConnection and NetStream classes public var videoStream:NetStream—to play the video

```
import flash.net.*;

// Before your Constructor, inside the class - declare your variable-
public var videoStream:NetStream;
private function loadProgressiveVideo(videoURL:String):void

{

 var nc:NetConnection = new NetConnection();
 nc.connect(null);

 videoStream = new NetStream(nc);
 videoStream.addEventListener(NetStatusEvent.NET _ STATUS, netStatusEv-
entHandler);
 videoStream.client = new StreamClient();
 videoStream.client.addEventListener(Event.COMPLETE, xmpHandler);
 videoStream.play(videoURL);
 videoStream.pause();

}
```

Loading and parsing the XML transcription

To load the XML file into Adobe Flash, use the URLRequest and URLLoader. Each transcripted cue point looks like this in XML from Soundbooth CS4:

```
<CuePoint>
 <Time>899</Time>
 <Type>event</Type>
 <Name>So</Name>
 <Parameters>
   <Parameter>
       <Name>source</Name>
       <Value>transcription</Value>
   </Parameter>
   <Parameter>
       <Name>duration</Name>
       <Value>500</Value>
   </Parameter>
   <Parameter>
       <Name>confidence</Name>
       <Value>-1</Value>
    </Parameter>
 </Parameters>
</CuePoint>
```

In ActionScript, create a new cue point object, so you can save this information as you're extracting it from XML:

```
public class CuePoint
{
 public var time:Number;
 public var text:String;
 public var duration:Number;
 public function CuePoint(time:Number, text:String,duration:Number):vo
```

```
id
 {
   time = _ _ time;
   text = _ _ text;
   duration = _ _ duration;
 }
}
```

Once you have the cue point object set up, you can begin extracting the parameters from the XML file:

```
// Declare a cuePoints array
public var cuePoints:Array = new Array();
private function loadXML(xmlURL:String):void
{
   var request:URLRequest = new URLRequest(xmlURL);
   var loader:URLLoader = new URLLoader();
   try
 {
                        loader.load(request);
               }
               catch ( _ _ error:SecurityError)
               {
                        trace("A SecurityError has occurred.");
               }
   loader.addEventListener(Event.COMPLETE,xmlloaderCompleteEventHandler);
}
private function xmlloaderCompleteEventHandler(event:Event):void
{
 var loader:URLLoader = URLLoader(event.target);
 try
 {
   var xml:XML = new XML(loader.data);
   for each (var node in xml.children()) {
       if (node.Parameters.Parameter[0].Value == "transcription")
              cuePoints.push(new CuePoint(node.Time, node.Name, node
Parameters.Parameter[1].Value)
 }
 catch (error:TypeError)
 {
   trace("Could not parse XML file");
 }
}
```

You want to loop through each of the transcription nodes and parse out the Node.Time, Node.Name, and Node.Values and anything from the Soundbooth CS4 XML export transcription that would be useful. A suggestion is to save theses objects into an Array (cue points—as indicated in the code), so that further access, such as searching will be easy.

Now that you have the cue points and times extracted, the next step is to write a function to search for the cue points with a keyword. To search the cue points array we created:

```
Public function search(keyword:String):Array
{
 var cuePointsFound:Array = new Array();
 for (var i:int = 0; I < cuePoints.length; i++) {
   if (cuePoints[i].text.toLowerCase() == keyword.toLowerCase()) {
```

```
 cuePointsFound.push(new CuePoint(cuePoints[i].time, cuePoints[i].text,
cuePoints[i].duration));
    }
 }
 return cuePointsFound;
}
```

User Interface

On the stage of the FLA file, have the video player play your video. On top of the seek bar area, add cueMarkers that indicate where your keyword exists on the video:

```
Public class CueMarker extends MovieClip {
 public var time:Number;
 public var text:String;
 public function CueMarker():void {
    addEventListener(MouseEvent.MOUSE _ OVER, mouseOverEventHandler);
    addEventListener(MouseEvent.MOUSE _ OUT, mouseOutEventHandler);
    addEventListener(MouseEvent.CLICK, mouseClickEventHandler);
 }
 private function mouseOverEventHandler(event:MouseEvent):void {
    dispatchEvent(new Event("CueMarkSeek"));
 }
 private function mouseOverEventHandler(event:MouseEvent):void {
    dispatchEvent(new Event("CueMarkerHover"));
 }
 private function mouseOutEventHandler(event:MouseEvent):void {
    dispatchEvent(new Event("CueMarkerNormal"));
 }
}
Public function AttachCueMarkers():void
{
 var marker:CueMarker;
 var markerContainer:MovieClip = new MovieClip();
 for (var i:int = 0; i< cuePointsFound.length; i++) {
    marker = new CueMarker();
    marker.time = cuePointsFound[i].time;
    marker.text = cuePointsFound[i].text;
    marker.x = Math.floor(Math.min(seekBar.width, seekBar.width *
((cuePointsFound[i].time/1000) / cuePointsFound[i].duration)));
    markerContainer.addChild(marker);
 }
 seekBar.addChild(markerContainer);
}
```

Summary

These instructions should help you generate a searchable video from Production Premium and create a player that will enable website visitors to seach the video on your website. Visit the Production Premium website for further information as we plan link to customer-created players when available.

For more information

For more details about Adobe Creative Suite 4 Production Premium,
visit *www.adobe.com/go/creativesuiteproduction.*

Adobe

Adobe Systems Incorporated
345 Park Avenue
San Jose, CA 95110-2704
USA
www.adobe.com