



Xenon Offline Encoder 4.1

Timed Text User Guide

June 29, 2007
Version 1.0

Vidiator Technology (US) Inc.
10900 NE Eighth Street, Suite 1486
Bellevue, WA 98004 U.S.A.
www.vidiator.com

Document Information

File Reference			
Xenon_Offline_Encoder_4.1_Timed_Text_User_Manual.doc			
Revision History			
Version		Date	
1.0		June 29, 2007	
Changes Since Last Version			
Initial release of this document.			
Related Documents			
Copyright Notice			
Copyright © 2007 Vidiator Technology (US) Inc. under license. All rights reserved. VIDIATOR and XENON are trademarks of Vidiator Technology (US) Inc. under license. All rights reserved. All other trademarks or registered trademarks are property of their respective companies. Specifications are subject to change without notice. CONFIDENTIAL INFORMATION. Do not disclose or distribute. Patent Notice: These products are protected under one or more of the following U.S. patents and/or their foreign equivalents: US Patent No.: 7,143,432 B1; 7,114,174; 6,981,045 B1; 6,948,131 B1; 7,069,573 B1; and 7,047,305 B1. Korean Patent No: 567157 Restricted Rights Legend Use, duplication or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause as DFARS 252.227-7013 for DOD agencies, and subparagraphs (c) (1) and (c) (2) of the Commercial Computer Software Restricted Rights clause at FAR 52.227-19 for other agencies. Vidiator Technology (US) Inc., 10900 NE 8th Street, Suite 1486, Bellevue, WA 98004 USA			

Table of Contents

1	INTRODUCTION	4
1.1	ABOUT THE XENON OFFLINE ENCODER	4
1.2	ABOUT TIMED TEXT PROFILES.....	4
2	SETTING THE TIMED TEXT FILE PATH	6
2.1	LOADING TIMED TEXT PROFILES FROM THE SOURCE FILE'S PATH	6
2.2	MANUALLY SELECTING THE TIMED TEXT PATH AND FILE NAME	7
3	RUNNING TIMED TEXT ENCODER JOBS.....	8
3.1	ADDING TIMED TEXT USING THE GUI	8
3.1.1	Select Source File and Timed Text Profile	8
3.1.2	Specify Preprocessor Settings	9
3.1.3	Select Encoder Settings	9
3.1.4	Set the Output Path and File Name	9
3.1.5	View Estimated File Size	9
3.1.6	Submit the Job.....	9
3.2	ADDING TIMED TEXT USING THE COMMAND LINE INTERFACE	10
3.2.1	Command Line Syntax.....	10
4	WRITING TIMED TEXT XML	11
4.1	SETTING THE REGION NODE.....	11
4.2	SETTING THE SAMPLE DESCRIPTION NODE	11
4.2.1	Text Align	12
4.2.2	Text Scroll	12
4.2.3	Background Color	12
4.2.4	Text Box.....	13
4.2.5	Default Style.....	13
4.2.6	Font Record	14
4.3	SETTING THE SAMPLE NODE	14
4.3.1	Sample Child Nodes.....	14
APPENDIX A.....		17
TIMED TEXT XML SCHEMA		17

1 Introduction

Xenon Offline Encoder supports 3GPP timed text. This feature allows text to be encoded into a video source file at a specified time in the file. Examples of timed text are the subtitles in a movie or the lyrics to a song in karaoke. This document describes timed text and how to add it to a source video file using Xenon Encoder.

1.1 About the Xenon Offline Encoder

Encoding timed text in a source video file with Xenon Offline Encoder requires the following components:

- **A source video file to encode.** A video file must be optimized for the before it can be streamed to a cell phone or other device.
- **The Xenon Encoder application.** The Xenon Encoder application is used to optimize the source file so it can be streamed to users. For timed text encoding jobs you can use the Xenon Encoder GUI or the command line interface. Both methods are described in this document.
- **A Xenon Encoder Profile.** A Xenon Encoder profile is an XML file (with extension .nep) and is created with the Xenon Encoder application. The encoder profile is an xml file that describes to the Xenon Encoder how to encode the file. Examples of information included in the profile are the codec to be used, video resolution, bit rate control, and file size.
- **A Timed Text Profile.** A timed text profile is an XML file (with extension .ntt) that describes to the Xenon Encoder the attributes of the text to display on the video. Examples of attributes in the .ntt file are
 - what text to display in the video
 - where on the screen (top, bottom, side)
 - when and for how long to display the text
 - the text color and size

This document describes the timed text feature of Xenon Offline Encoder and the included sample timed text xml files, how to modify a timed text xml file, and how to run an encoding job with timed text.

1.2 About Timed Text Profiles

Timed text profiles are XML files that conform to the 3GPP standards for timed text. Xenon Encoder includes sample timed text profiles. You can use these sample profiles or create your own. One way initially to create a timed text XML file is to modify one of the existing sample files included with the Xenon Encoder application. Chapter 4, “Writing Timed Text XML,” describes the parameters that comprise the timed text XML file.

If you install Xenon Offline Encoder in the default location then the sample timed-text xml files will be installed in:

C:\Program Files\Vidiator\Xenon Encoder Pro\Sample Timed Text Profile

Appendix A in this guide lists the XML schema for timed text.

2 Setting the Timed Text File Path

There are two ways to set the file path for timed text profiles that you encode in the source file. You configure Xenon Encoder so that timed text profiles are:

- by loading timed text profiles from the source file's path.
- by specifying the path and file name each time.

To set the timed text file path you will use the Timed Text Files Path on the Options dialog box (see Figure 1).

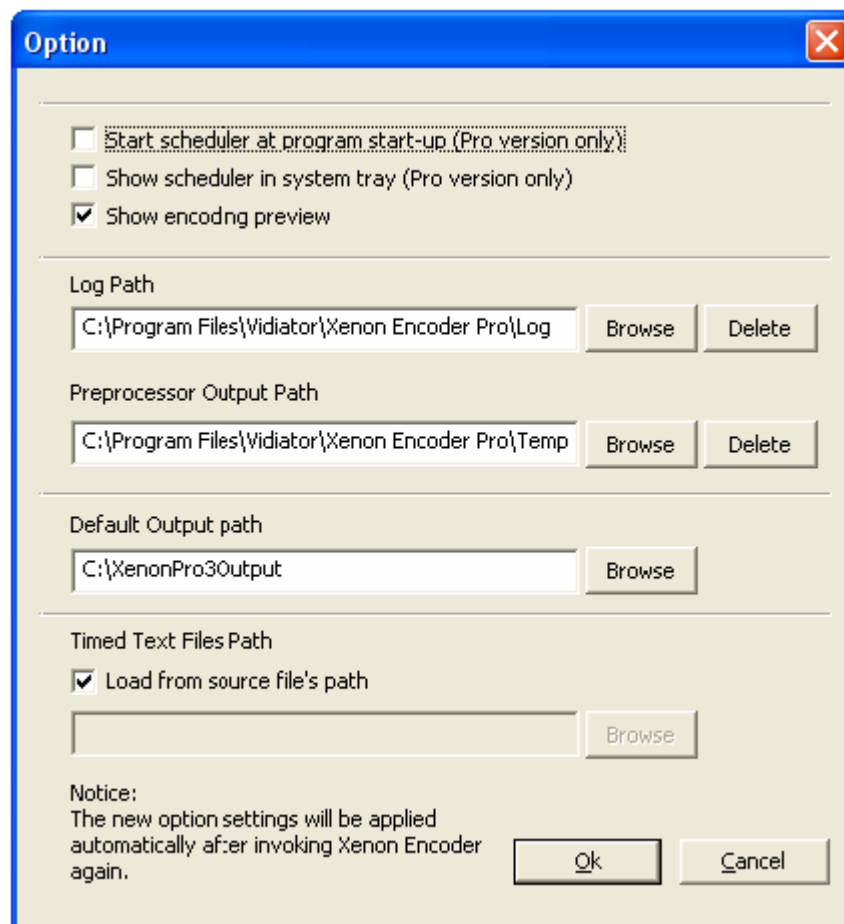


Figure 1 - The Options Dialog Box

2.1 Loading Timed Text Profiles from the Source File's Path

To configure Xenon Offline Encoder by loading the timed text profile from the same folder and path as the source file, select the **Load from source file's path** checkbox.

To encode a source file with timed text, you must do the following:

- Rename the timed text profile and the source file so that they have the same file name (except for the extension). For example: starwars.avi and starwars.ntt
- Put the timed text profile in the same folder as the source file.

2.2 Manually Selecting the Timed Text Path and File Name

To configure Xenon Offline Encoder by specifying a path and filename each time you encode a timed text job, you must perform the following steps:

- ensure the **Load from source file's** path box is not checked
- by using the **Browse** button, select a file path to load the profile from.

3 Running Timed Text Encoder Jobs

You can add timed text to a source file using Xenon Encoder's GUI or from a command line interface.

3.1 Adding Timed Text Using the GUI

This section describes how to add timed text to a source file using the GUI. For more information, you should refer to the online help, available by selecting Documentation from the Help menu.

The first time you use Xenon Encoder you will perform the following steps:

- Select source files
- Specify preprocessor settings
- Select encoder settings
- Set the output path and file name
- View estimated file size
- Submit the job

Each step is described in the sections that follow.

3.1.1 Select Source File and Timed Text Profile

To start setting up an encoding job, you must first select the source media file that you want to encode, as well as the timed text profile.

To select source files:

1. Open Xenon Offline Encoder. The main Xenon Offline Encoder window is displayed.
2. Click the **Select source file(s)** button. The **Select Source** dialog box is displayed.
3. Select one or *multiple source files to encode and then click **Open**.
4. Under **Timed Text**, click the **Use** checkbox.
 - If the **Load from source file's path** checkbox on the **Tools|Options** dialog box is selected, the timed text profile from same path and folder as the source file will be used.

NOTE: Note that the source file and timed text profile must be named the same (except for the file extension). For example: starwars.avi and starwars.ntt.

- If the Load from source file's path checkbox on the Tools|Options dialog box is **not** selected, you will be prompted to enter a path and filename for the timed text profile. Select the path and filename and click OK>

NOTE: Note: For more detailed information, see "Selecting a Source File" in the online help.

3.1.2 Specify Preprocessor Settings

Preprocessing can be applied if you want to make changes to the source file before encoding. A number of preprocessing options are available in Xenon Encoder, including changing the video resolution, cropping video, adding other files to the start or end of the source content, and adding a logo. For more detailed information on the different preprocessing options available, see "Specifying Preprocessor Settings" in the online help.

3.1.3 Select Encoder Settings

NOTE: For timed text encoding jobs, you **must** select the **MobileMP4** encode as the type of encoder to use for the encoding job. The other encoder types do not support timed text.

You can then specify the encoding settings that determine how the output file will be encoded, including the codec, resolution, number of bit rate tracks, and keyframe interval. For detailed information on all available encoder settings, see "Selecting Encoder Settings" in the online help.

To select encoder settings:

1. Click **Encoder** on the left side of the main Xenon Encoder window. A list of encoder profiles is displayed.
2. Click **MobileMP4 Encoder**.
3. Specify values for each encoding setting displayed in the right side of the window.

For detailed information on all available encoder settings, see "Selecting Encoder Settings" in the online help.

3.1.4 Set the Output Path and File Name

When you have set up preprocessing and encoder settings, you must specify the path and name of the output files that will be created by Xenon Encoder. For more detailed information, see "Set the Output Path and File Name" in the online help.

3.1.5 View Estimated File Size

After specifying encoding and preprocessor settings, you can view the estimated size of each encoded output file that will be generated by Xenon Encoder. For more detailed information, see "View the Estimated Output File Size" in the online help.

3.1.6 Submit the Job

When all the preparation is complete, click Submit Job to start the preprocessing and encoding process. The current processing status is displayed in the Job Status View window. For more detailed information see "Submitting an Encoding Job" in the online help.

3.2 Adding Timed Text Using the Command Line Interface

To add a timed text track to your encoded video files, you will specify the following components in the command line:

- A video file to be encoded
- A Xenon encoder profile (.nep) file
- A timed text xml (.ntt) file

3.2.1 Command Line Syntax

Use the following syntax to run an encoder job with timed text:

```
nxec full_path_for_input_filename full_path_encoding_nep_file  
full_path_for_output_filename /t:full_path_for_timed_text_ntt_file
```

For example:

```
nxec.exe input.avi profile.nep output.3gp /t:sample.ntt
```

Where *nxec.exe* is the Xenon Encoder executable file, *input.avi* is the name of the source input file, *profile.nep* is the name of the Xenon Encoder profile, *output.3gp* is the filename for the generated output, and */t:sample.ntt* is the filename for the timed text xml file.

4 Writing Timed Text XML

This chapter describes the timed text attributes supported by Xenon Encoder.

4.1 Setting the Region Node

The <region> node describes the area on the screen in which the timed text displays. The region node has four attributes.

<region> attributes

- "tx"
The left position from the screen, in pixels.
- "ty"
The top position from the screen, in pixels.
- "width"
The width of region, in pixels.
- "height"
The height of region, in pixels.

Figure 2 shows the region in relation to the entire screen. It also shows the other configurable attributes.

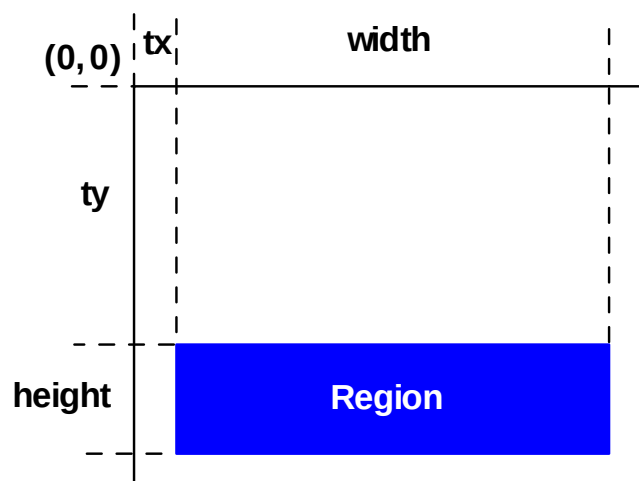


Figure 2 – Timed Text Region

4.2 Setting the Sample Description Node

The <sample_description> node sets the text align, scroll, background color, text box area, fonts and style.

4.2.1 Text Align

Describes the position of the text in the region.

<horizontal_justification>

Describes the text's horizontal alignment. It can have one of the following values: "left", "centered", "right".

<vertical_justification>

Describes the text's vertical alignment. It can have one of the following values: "top", "centered", "bottom".

4.2.2 Text Scroll

Text Scroll specifies information about the region and about how text is rendered.

<displayFlags>

- "fill_text_region":
Specifies whether the region is filled with background color that is mentioned at <background_color_rgba> node.

It can have two values: "true" and "false"
- "scroll_direction"
Represents scroll direction. It can have one of the following values: "up_to_down", "left_to_right", "right_to_left" and "down_to_up".
- "scroll_in"
Specifies if text is scrolled in. If this is "true", text will slide into center of text box.
- "scroll_out"
Specifies if text is scrolled out. If this is "true", text will come out of text box.
- "continuous_karaoke"
If this is "true", highlighted character will be kept.
- "write_text_vertically"
If this is "true", text will be displayed vertically.

4.2.3 Background Color

<background_color_rgba>

Specifies the background color of the region.

- "r" red color degree (0 ~ 255)
- "g" green color degree (0 ~ 255)

- “b” blue color degree (0 ~ 255)
- “a” alpha degree (0 ~ 255)

4.2.4 Text Box

<default_text_box> node

Defines a text box that displays within the region, as shown in Figure 3.

- “left”
left position from the region by pixel.
- “top”
top position from the region by pixel.
- “right”
right position from the region by pixel.
- “bottom”
bottom position from the region by pixel.

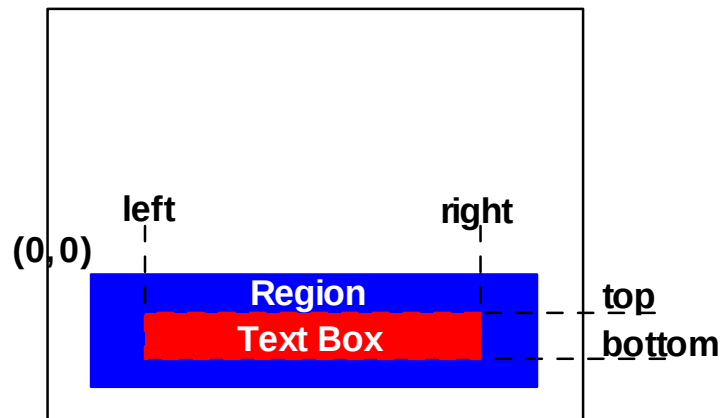


Figure 3 - Text Box

4.2.5 Default Style

<default_style> node

Sets a default font style. It has six child nodes. But only four nodes are used.

- <font_ID>
Specifies which font is used. This can have the values of those listed in <font_table>.
- <face_style_flags>

Has three attributes: “italic”, “bold” and “underline”. Each attribute has “true” or “false” as the value.

- `<text_color_rgba>`

Specifies the color of font. The attributes are same with `<background_color_rgba>`.

- `<font_table>`

Represents font collections. The “entry_count” attribute is the number of fonts.

4.2.6 Font Record

`<font_record>` node

Lists the available fonts. It has two attributes.

- “font_id”
Base index number of font.
- “font”
Font name by string.

4.3 Setting the Sample Node

The `<sample>` node has four attributes.

- “sample_description_id”
It has “id” value of `<sample_description>` attribute.
After “sample_description_id” is set. It affects all `<sample>` node until new “sample_description_id” is set.
- “timestamp”
Represents the starting time of this sample from the beginning, by msec.
- “duration” –
It represents duration time of this sample from the time is defined at “timestamp” by msec.
- “text”
Content string.

4.3.1 Sample Child Nodes

The `<sample>` node has nine child nodes, as described below:

- `<style>`
User can change font style partly.
“entry_count” attribute is the number of style that is applied on the text.
`<entry>` node is used to list style that is applied on this sample.

This node has same function with <default_style>. Refer to < default_style> for more information.

To make a range of style, use <startChar> and <endChar> nodes. Each node has an index of characters (index is ZERO base.).

The other nodes are same as <default_style>.

- <highlightcolor>
Sets the color for highlight.
“r” – red color degree (0 ~ 255)
“g” – green color degree (0 ~ 255)
“b” – blue color degree (0 ~ 255)
“a” – alpha degree (0 ~ 255)
- <highlight>
Sets the highlight block.
“startcharoffset” – The index for the beginning of block. (ZERO base).
“endcharoffset” – The index for the end of block. (ZERO base).
- <karaoke>
Has two attributes.
“entry_count” – The number of karaoke blocks.
“highlight_start_time” – The start time of the first karaoke block after this sample is started.
- <entry>
node represent the karaoke blocks. Each <entry> node is a karaoke block. This node has three attributes.
“startcharoffset” – The index for the beginning of block. (ZERO base).
“endcharoffset” – The index for the end of block. (ZERO base).
“highlight_end_time” – The end time of this block after this sample is started.
- <hypertext>
Makes the player invoke a URL.
“url” - User can assign URL through this attribute.
“alt” – This attribute may be used as tool-tip or other visual clue, as a substitute for the URL.
“startcharoffset” – The index for the beginning of block. (ZERO base).
“endcharoffset” – The index for the end of block. (ZERO base).

- `<blink>`
node make this block is blinked.

“startcharoffset” – The index for the beginning of block. (ZERO base).

“endcharoffset” – The index for the end of block. (ZERO base).
- `<textbox>`
Resets the position of current text box. This setting affects all subsequent samples. Refer to `<default_text_box>` node for attribute using.
- `<textwrap>`
May be used for indication of text wrapping.

“wrap_flag” – This attribute can have “no_wrap” or “soft_wrap” as a value. If “soft_wrap” is set, the player will wrap the text automatically.

Appendix A

Timed Text XML Schema

This appendix includes the XML Schema for 3GPP Timed Text Format. For more information see:

<http://www.w3.org/2001/XMLSchema>

```

    <?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation>
XML Schema for 3GPP Timed Text Format
based on 3GPP TS 26.245
-----
(C) Vidiator Korea

+ ver 0.6 at March 8, 2006
  Created.

+ ver 0.7 at March 10, 2006
  StyleRecordType has been changed to have attributes of boolean.

+ ver 0.8 at March 13, 2006
  'text' attribute was wrongly placed under 'samples' element.

+ ver 0.9 at March 15, 2006
  'entry_count' attribute was added to 'font_table' element.
  'duration' and 'time_scale' attributes were removed from 'timed_text_track' element.
  'timestamp' attribute was added to 'sample' element.
  'style' element has 'entry' element
  ...

+ ver 1.0 at March 20, 2006
  actual first version in use with the NTT converter after some changes for practical
  reasons.

+ ver 1.1 at May 24, 2006
  layer, timescale attributes of 'timed_text_track' element added.
  duration attribute of 'sample' element added.
  timestamp attribute of 'sample' element removed.
  start_timestamp attribute of 'samples' element added.

  Vidiator_NTT_Schema root element with a 'version' attribute was inserted.

  Currently, the timed text writer doesn't support the multiple sample descriptions.
  the others but the id '0' will be ignored.

    </xsd:documentation>
  </xsd:annotation>
<xsd:element name="vidiator_ntt_schema">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="timed_text_track">

```

```
<xsd:complexType>
  <xsd:sequence>
    <!-- region : position within display area -->
    <xsd:element name="region">
      <xsd:complexType>
        <xsd:attribute name="tx" type="xsd:unsignedInt" use="required"/>
        <xsd:attribute name="ty" type="xsd:unsignedInt" use="required"/>
        <xsd:attribute name="width" type="xsd:unsignedInt" use="required"/>
        <xsd:attribute name="height" type="xsd:unsignedInt" use="required"/>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="sample_descriptions">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="sample_description" maxOccurs="unbounded">
            <xsd:complexType>
              <xsd:all>
                <xsd:element name="horizontal_justification">
                  <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                      <xsd:enumeration value="left"/>
                      <xsd:enumeration value="centered"/>
                      <xsd:enumeration value="right"/>
                    </xsd:restriction>
                  </xsd:simpleType>
                </xsd:element>
                <xsd:element name="vertical_justification">
                  <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                      <xsd:enumeration value="top"/>
                      <xsd:enumeration value="centered"/>
                      <xsd:enumeration value="bottom"/>
                    </xsd:restriction>
                  </xsd:simpleType>
                </xsd:element>
                <xsd:element name="displayFlags">
                  <xsd:complexType>
                    <xsd:attribute name="scroll_in" type="xsd:boolean"/>
                    <xsd:attribute name="scroll_out" type="xsd:boolean"/>
                    <xsd:attribute name="scroll_direction">
                      <xsd:simpleType>
                        <xsd:restriction base="xsd:string">
                          <xsd:enumeration value="up_to_down"/>
                          <xsd:enumeration value="left_to_right"/>
                          <xsd:enumeration value="right_to_left"/>
                          <xsd:enumeration value="down_to_up"/>
                        </xsd:restriction>
                      </xsd:simpleType>
                    </xsd:attribute>
                  </xsd:complexType>
                </xsd:element>
              </xsd:all>
            </xsd:complexType>
          </xsd:element>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

```

        </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="continuous_karaoke"

type="xsd:boolean"/>

    <xsd:attribute name="write_text_vertically"

type="xsd:boolean"/>

    <xsd:attribute name="fill_text_region" type="xsd:boolean"/>
</xsd:complexType>
</xsd:element>
<xsd:element name="background_color_rgba">
    <xsd:complexType>
        <xsd:attributeGroup ref="ColorRGBAAttr"/>
    </xsd:complexType>
</xsd:element>
<xsd:element name="default_text_box">
    <xsd:complexType>
        <xsd:attributeGroup ref="BoxRecordAttr"/>
    </xsd:complexType>
</xsd:element>
<xsd:element name="default_style" type="StyleRecordType"/>
<xsd:element name="font_table">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="font_record" maxOccurs="unbounded">
                <xsd:complexType>
                    <xsd:attribute name="font_id" type="xsd:unsignedInt"

use="required"/>

                    <xsd:attribute name="font" type="xsd:string"

use="required"/>

                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="entry_count" use="required">
            <xsd:simpleType>
                <xsd:restriction base="xsd:unsignedByte">
                    <xsd:minInclusive value="1"/>
                    <xsd:maxInclusive value="20"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>
</xsd:all>
<xsd:attribute name="id" type="xsd:unsignedInt" use="required"/>
<!-- 'id' is zero-based. -->
</xsd:complexType>
</xsd:element>

```

```

        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="samples">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="sample" minOccurs="0" maxOccurs="unbounded">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="style" minOccurs="0">
                  <xsd:complexType>
                    <xsd:sequence>
                      <xsd:element name="entry" type="StyleRecordType"
maxOccurs="unbounded" />

                    </xsd:sequence>
                    <xsd:attribute name="entry_count" use="required">
                      <xsd:simpleType>
                        <xsd:restriction base="xsd:unsignedByte">
                          <xsd:minInclusive value="1" />
                          <xsd:maxInclusive value="100" />
                        </xsd:restriction>
                      </xsd:simpleType>
                    </xsd:attribute>
                  </xsd:complexType>
                </xsd:element>
                <xsd:element name="highlightcolor" minOccurs="0">
                  <xsd:complexType>
                    <xsd:attributeGroup ref="ColorRGBAAttr" />
                  </xsd:complexType>
                </xsd:element>
                <xsd:element name="highlight" minOccurs="0">
                  <xsd:complexType>
                    <xsd:attributeGroup ref="CharOffsetAttr" />
                  </xsd:complexType>
                </xsd:element>
                <xsd:element name="karaoke" minOccurs="0">
                  <xsd:complexType>
                    <xsd:sequence>
                      <xsd:element name="entry" maxOccurs="unbounded">
                        <xsd:complexType>
                          <xsd:attribute name="highlight_end_time"
type="xsd:integer" />

                          <xsd:attributeGroup ref="CharOffsetAttr" />
                        </xsd:complexType>
                      </xsd:element>
                    </xsd:sequence>
                    <xsd:attribute name="highlight_start_time"

```

```

type="xsd:integer" use="required"/>
    <xsd:attribute name="entry_count" use="required">
    <xsd:simpleType>
    <xsd:restriction base="xsd:unsignedByte">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="200"/>
    </xsd:restriction>
    </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>
</xsd:element>
<xsd:element name="hyper text" minOccurs="0">
    <xsd:complexType>
    <xsd:attributeGroup ref="CharOffsetAttr"/>
    <xsd:attribute name="alt" use="required">
    <xsd:simpleType>
    <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="url" type="xsd:string"
use="required"/>
    </xsd:complexType>
</xsd:element>
<xsd:element name="text box" minOccurs="0">
    <xsd:complexType>
    <xsd:attributeGroup ref="BoxRecordAttr"/>
    </xsd:complexType>
</xsd:element>
<xsd:element name="blink" minOccurs="0">
    <xsd:complexType>
    <xsd:attributeGroup ref="CharOffsetAttr"/>
    </xsd:complexType>
</xsd:element>
<xsd:element name="text wrap" minOccurs="0">
    <xsd:complexType>
    <xsd:attribute name="wrap_flag" use="required">
    <xsd:simpleType>
    <xsd:restriction base="xsd:string">
    <xsd:enumeration value="no_wrap"/>
    <xsd:enumeration value="soft_wrap"/>
    </xsd:restriction>
    </xsd:simpleType>
    </xsd:attribute>
    </xsd:complexType>
</xsd:element>
<xsd:element name="scroll delay" type="xsd:unsignedInt"
minOccurs="0"/>

```

```

        </xsd:sequence>
        <xsd:attribute name="timestamp" type="xsd:unsignedLong"
use="required"/>

        <xsd:attribute name="text" type="xsd:string" use="required"/>
        <xsd:attribute name="duration" use="required">
        <!--

duration :
the next sample timestamp = the current sample timestamp + the current sample duration.
If this condition is not met, the current sample duration is ignored and recalculated by the
next time stamp.
-->

        <xsd:simpleType>
        <xsd:restriction base="xsd:unsignedLong"/>
        </xsd:simpleType>
        </xsd:attribute>
        <xsd:attribute name="sample_description_id" use="optional"
default="0"/>

        </xsd:complexType>
        </xsd:element>
        </xsd:sequence>
        </xsd:complexType>
        </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="language" use="required">
        <xsd:simpleType>
        <xsd:restriction base="xsd:string">
        <xsd:minLength value="3"/>
        <xsd:maxLength value="3"/>
        </xsd:restriction>
        </xsd:simpleType>
        </xsd:attribute>
        <xsd:attribute name="layer" type="xsd:integer" use="optional" default="0"/>
        <xsd:attribute name="timescale" type="xsd:unsignedInt" use="optional"
default="1000"/>
        </xsd:complexType>
        </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="version" type="xsd:float" use="required"/>
        </xsd:complexType>
        </xsd:element>
        <!-- Attribute Groups -->
        <xsd:attributeGroup name="ColorRGBAAAttr">
        <xsd:attribute name="r" type="colorValType" use="required"/>
        <xsd:attribute name="g" type="colorValType" use="required"/>
        <xsd:attribute name="b" type="colorValType" use="required"/>
        <xsd:attribute name="a" type="colorValType" use="required"/>
        </xsd:attributeGroup>

```

```
<xsd:attributeGroup name="BoxRecordAttr">
  <xsd:attribute name="top" type="xsd:integer" use="required"/>
  <xsd:attribute name="left" type="xsd:integer" use="required"/>
  <xsd:attribute name="bottom" type="xsd:integer" use="required"/>
  <xsd:attribute name="right" type="xsd:integer" use="required"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="CharOffsetAttr">
  <xsd:attribute name="startCharOffset" type="xsd:unsignedShort" use="required"/>
  <xsd:attribute name="endCharOffset" type="xsd:unsignedShort" use="required"/>
</xsd:attributeGroup>
<!-- Type Groups -->
<xsd:simpleType name="colorValType">
  <xsd:restriction base="xsd:unsignedByte">
    <xsd:maxInclusive value="255"/>
    <xsd:minInclusive value="0"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="StyleRecordType">
  <xsd:all>
    <xsd:element name="startChar" type="xsd:unsignedInt"/>
    <xsd:element name="endChar" type="xsd:unsignedInt"/>
    <xsd:element name="font_ID" type="xsd:unsignedInt"/>
    <xsd:element name="face_style_flags">
      <xsd:complexType>
        <xsd:attribute name="italic" type="xsd:boolean"/>
        <xsd:attribute name="bold" type="xsd:boolean"/>
        <xsd:attribute name="underline" type="xsd:boolean"/>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="text_color_rgba">
      <xsd:complexType>
        <xsd:attributeGroup ref="ColorRGBAAttr"/>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="font_size" type="xsd:unsignedInt"/>
  </xsd:all>
</xsd:complexType>
</xsd:schema>
```