Creating Video DVDs under OS/2

By Alex Taylor

This document describes the process of DVD-video authoring under OS/2. Among the topics included:

- Converting video files to DVD format.
- Creating DVD menus, titlesets, titles, and chapters.
- Generating the DVD file system.
- Writing the DVD.

Requirements

You will need the following software:

- The *mjpegtools* package (available from *http://smedley.info/os2ports/*), in particular the programs:
  - `ppmtoy4m.exe`
  - `jpeg2yuv.exe`
  - `mpeg2enc.exe`
  - `mplex.exe`
- *DVDAuthor* (available from *http://smedley.info/os2ports/*), which includes the programs:
  - `dvdauthor.exe`
  - `spumux.exe`
- *FFmpeg* (available from *http://smedley.info/os2ports/*)
- The *cdrtools* package by nickk (available on Hobbes), in particular `mkisofs2.exe` (this version supports files over 2 GB, whereas the original port by Chris Wohlgemuth does not).
- Image editing software with the following capabilities:
  - Resizing and cropping of bitmap images.
  - Drawing of shapes and text.
  - Colour-depth conversion and palette editing.
  - Support for saving in PNG, JPEG, and (if possible) PPM format.

  I recommend using *PMView* in combination with an advanced image editor like the *GIMP*, *Pixel32*, or *Embellish*.

It is assumed that you have some basic familiarity with image files and editors.

Generating the Video Files

If you're planning to create a video DVD, presumably you have some video file(s) that you want to put on it. This may be something you've saved off your digital video camera, a recording from your PVR card, or a video file downloaded from the Internet. Whatever it is, I will assume that it's already on your hard disk (and if it isn't, yet, then getting it there is your responsibility).

The first thing you have to do is convert these video files into the correct format for a video DVD. The tool to do this is *FFmpeg*.

FFmpeg is a powerful video conversion program which supports a huge range of input and output formats (both video and audio). To convert a video file – of almost any type – into DVD video format, use the command:

```
ffmpeg -i <input video file> -target ntsc-dvd <output video file>
```

The above command will create a video file in the correct format for an NTSC DVD. If you will be creating a PAL (or SECAM) DVD, use this command instead:

```
ffmpeg -i <input video file> -target pal-dvd <output video file>
```

FFmpeg supports a great many command-line options for specifying input and output file formats. However, under normal circumstances, it is usually intelligent enough to figure out the correct options automatically.

Once you've got all your video files into the correct DVD video format, you can turn your attention to designing your DVD.

Creating a DVD Menu

Overview

A DVD menu consists of a background (which can be either a still image or a short movie clip) with one or more selectable menu items called *buttons*. The menu buttons can differ in appearance according to one of three button *states*:

"Normal" state
: This is the button as it normally appears on the menu.

"Highlighted" state
: This is what the button looks like when the user moves the selection cursor (using the DVD player's controls) over it.

"Selected" (or "active") state
: This is what the button looks like after the user has activated the button, in the brief instant before the associated action takes effect.

The menu itself is actually a small MPEG2 video clip, with the buttons implemented as DVD subtitles. You can create one from a still image (as described below), or you can use an actual video clip if you have one.

A DVD menu also requires an audio track, even if it has no actual sound. Unless you will be using an existing video clip with audio, you will need an audio clip of some kind to add to the menu. You can use a sound file of your choice; the instructions below assume an MP2 file, but any audio format supported by *FFmpeg* should work.

To create a menu with no sound, you will need an audio clip containing nothing but silence. If you don't have the means of creating one, you can obtain "`silence.mp2`" from the file `http://www.pcxperience.org/james/dvd/menus/static-menus-1.0.tar.gz` (this archive contains some other interesting sample files as well).

The procedure to create a DVD menu from a still background image is listed below. (If you already have a video clip in MPEG2 format that you plan to use, you should be able to adapt these instructions fairly easily.)

Instructions

1. Create the menu images

Description

Separate image files are created for:

- The background image of the menu.
- The button appearance in "normal" state.
- The button appearance in "highlighted" state.
- The button appearance in "selected" state.

In fact, you can incorporate the default button graphics into your background image, in which case you don't need to create the "normal" button image. But the instructions here will assume that you have separate image files for all three states.

Your final menu (once the DVD is created) will appear as your background image with the "normal" button image superimposed on top of it. The "highlighted" image will only be applied to the button currently under the cursor, and the "selected" image will only be applied to a button when it is activated.

So that you can see how these images will all look together, I advise you to create them using graphics software with a layer feature, such as the *GIMP* or *Pixel32* (or even *Photoshop*). The background image and each of the button images can then be implemented as separate layers within a single project. When it comes time to generate the individual image files, you can export each layer as a separate image by hiding all of the others.

All of your image files should be saved in the following dimensions (*width* × *height*, in pixels):

NTSC: 720x480 (North America, Japan & South Korea)
PAL: 720x576 (most of Europe & Australasia, as well as regions using SECAM)

When you convert these images to DVD video format (see step 3), they will each be resized to an aspect ratio of either 4:3 (fullscreen mode) or 16:9 (widescreen mode), depending on the options you pass to *mpeg2enc*. Effectively, this will result in a slight "squashing" of the image, either horizontally (4:3) or vertically (16:9), when the DVD is actually played.

Therefore, you may wish to create your images using a 4:3 or 16:9 aspect ratio initially, and then "stretch" it to the final, correct, dimensions.

For example:

NTSC: Create your image at 640x480 (for 4:3) or 640x360 (for 16:9), and then stretch it to 720x480.
PAL: Create your image at 720x576 (for 4:3) or 720x405 (for 16:9), and then stretch it to 720x576.

Designing the images

The background image contains everything that will be common to your menu's appearance in all three button states. In other words, any part of the menu graphic that **never** changes, regardless of what's happening to the buttons, should be part of the background image.

As I mentioned, you are certainly free to incorporate graphics representing the buttons themselves (or parts of them) into the background image; just remember that anything in the background image will remain constant, no matter what state each button is in.

Now for the button images. Unlike the background image, which can contain any number of colours, the button images are all limited to a single palette of no more than **three** (3) colours. This means that colourful graphics or anti-aliased text **cannot** appear in the button images.

The "normal" button image contains everything that is specific to your menu buttons as they will appear in their default state. For the remaining examples I will assume that this image is called

"menu_normal.png".

The "highlighted" button image contains everything that is specific to each menu button as it will appear when it has the input focus. I will assume that this image is called "menu_hilite.png".

The "selected" button image contains everything that is specific to each menu button as it will appear immediately after the user has activated (or "pressed") it, in the split second or so before the associated action begins. I will assume that this image is called "menu_select.png".

Saving the image files

The background image must be saved in either JPEG or PPM format. (Most online DVD tutorials seem to suggest using JPEG; personally, however, I prefer PPM, which is a "lossless" format). *PMView*, *Pixel32*, and the *GIMP* all support saving in PPM format.

For the remaining examples I will assume that the background image has been saved to a file named "menu_bg.ppm".

All three button images must be saved in colour-indexed (a.k.a. "palettized") PNG format. You are limited to a total of three (3) colours across all three images. The unused areas of these images (outside the button regions) should be transparent background. (If your image software isn't capable of saving transparent PNGs, you can use a solid background colour instead, but this will require some extra work when you create your menu XML file. The background does not count towards your three-colour limit.)

2. Create the menu background clip

Creating the video portion

Your menu background image must be converted into a special MPEG video clip. The first step is to create the video portion (without audio), which is referred to as an "M2V" file.

The necessary commands are slightly different depending on whether you are creating a PAL or NTSC DVD.

For an NTSC DVD, the commands to create the M2V file from a PPM image are:

```
ppmtoy4m -n 50 -F 30000:1001 -A10:11 -I p -r -S 420mpeg2 menu_bg.ppm > menu_bg.yuv
mpeg2enc menu_bg.yuv -n n -f8 -b5000 -o menu_bg.m2v
```

The corresponding commands for a PAL DVD are:

```
ppmtoy4m -n 50 -F 25:1 -A59:54 -I p -r -S 420mpeg2 menu_bg.ppm > menu_bg.yuv
mpeg2enc menu_bg.yuv -n p -f8 -b5000 -o menu_bg.m2v
```

You should also be able to do this in one step by piping the output of *ppmtoy4m* directly to *mpeg2enc*, i.e.:

```
ppmtoy4m -n 50 -F 30000:1001 -A 10:11 -I p -r -S 420mpeg2 menu_bg.ppm |
mpeg2enc -n n -f8 -b5000 -o menu_bg.m2v
```

Note: *mpeg2enc* will attempt to detect the correct aspect ratio from the YUV header. With these instructions, it generally seems to result in 4:3 (fullscreen) by default. Add the -a2 switch to *mpeg2enc* to specify 4:3 explicitly, or -a3 to force 16:9 (widescreen) instead. As noted previously, your image will be squashed (in one direction or the other) to fit the correct aspect ratio.

If you saved your background image as a JPEG instead of a PPM, replace the *ppmtoy4m* command in the syntax above with:

```
jpeg2yuv -n 50 -I p -f 50 -j menu_bg.jpg > menu_bg.yuv
```

(This command is the same for both PAL and NTSC.)

Creating the audio portion

Convert your selected menu audio clip into AC3 format using *FFmpeg*:

```
ffmpeg -i silence.mp2 -ab 224 -ar 48000 menu_audio.ac3
```

(If you are using your own audio file, substitute it for "`silence.mp2`" in the above command.)

Combining the audio and video

Now you have to multiplex your menu background's audio and video together using the *mplex* command:

```
mplex -f 8 -o menu_bg.mpg menu_bg.m2v menu_audio.ac3
```

You should end up with an MPEG video file which contains the background (audio and video) of your DVD menu. If the quoted examples were used, this file will be named "`menu_bg.mpg`".

The next step will be to combine this menu background with your button images.

3. Create the Final Menu

The menu buttons images must now be merged into the menu background, using the *spumux* utility (part of the *DVDAuthor* package).

This step requires you to create an *spumux* XML file (e.g. "`menu.xml`") which defines the menu layout. This is a specially-formatted ASCII text file that uses a syntax reminscent of HTML. A typical *spumux* XML file looks something like this:

```
<subpictures>
  <stream>
    <spu force="yes"
         start="00:00:00.0"
         image="menu_normal.png"
         highlight="menu_hilite.png"
         select="menu_select.png"
         transparent="000000"
         autooutline="infer"
```

```
          outlinewidth="5"
          autoorder="rows">
      </spu>
    </stream>
  </subpictures>
```

The general structure of the XML file is always the same; it is the parameters to the "<spu>" tag which define your specific menu layout:

`force="yes"`
`start="00:00:00.0"`
: These statements (with the values shown) are required.

`image="<filename>"`
`highlight="<filename>"`
`select="<filename>"`
: These three lines specify the three PNG files (created in step 2) that define your menu buttons. "image" is the name of the image containing the buttons in normal state, "highlight" is the image with the buttons in highlighted state, and "select" is the image with the buttons in selected state.

  If you don't have one (or more) of these button images, omit the corresponding parameter.

`transparent="<RRGGBB>"`
: This parameter is only required if your PNG files haven't been saved with a transparent background colour. It tells *spumux* to render the designated colour within all three button images as transparent (when overlaying them onto the background image). The value takes the form "RRGGBB", which specifies a colour value in hexadecimal RGB notation: pure black is "000000", and pure white is "FFFFFF".

  (Note that this parameter applies to all three images, so the value you use had better *be* the background colour in all three.)

`autooutline="infer"`
: Your DVD menu needs to know exactly which regions of the screen correspond to where the buttons are, so that the correct portions of the image can respond to selections during menu navigation. This parameter tells *spumux* to intelligently detect the boundaries of your menu buttons within the PNG images. It attempts to do this by determining what the images will look like when overlaid on top of one another, and then identifying enclosed regions of colour (other than the PNG background) within the composite image. This is usually sufficient, if your buttons are represented by clearly-delimited rectangular regions within at least one of the PNG files.

  If there are multiple regions of colour within a single button area (for instance, if the button images in each PNG file consist only of text), you may need to specify the "outlinewidth" parameter (below) as well.

`outlinewidth="<number>"`
: This parameter tells *spumux* how much transparent space (in pixels) must exist between two or more discrete regions of colour within the PNG files in order for them to be considered separate buttons. This is useful if your buttons have no coloured background, but consist only of multiple closely-spaced elements (such as letters in a text string) that should be treated as a

single element for the purpose of identifying a single button region.

`autoorder="<rows|columns >"`

This parameter specifies the axis along which the buttons are arranged, which *spumux* uses to determine their logical order. The button order will be used for identification purposes when the DVD layout is created using *DVDAuthor*; it also determines how the navigation buttons will behave when the DVD menu is played.

An value of "`rows`" indicates that the buttons lie along discrete rows, from top to bottom, with no vertical overlap. A value of "`columns`" indicates that they lie on discrete columns, from left to right, with no horizontal overlap.

This setting can only work if your buttons correspond to one of these arrangements. If all of your buttons do not lie on either non-overlapping rows or non-overlapping columns, you will have to specify the button regions manually (see below).

If the automated button-identification logic afforded by the "`autooutline`", "`outlinewidth`" and "`autoorder`" parameters is insufficient given the way your buttons are drawn or laid out, you can always specify the exact pixel boundaries of each button instead.

The following example defines a menu with four buttons:

```
<subpictures>
  <stream>
    <spu force="yes"
         start="00:00:00.00"
         image="menu_normal.png"
         highlight="menu_hilite.png"
         select="menu_select.png"
         autoorder="rows">
      <button name="ep1" x0="41"  y0="15"  x1="186" y1="42"  />
      <button name="ep2" x0="91"  y0="47"  x1="236" y1="74"  />
      <button name="ep3" x0="141" y0="80"  x1="286" y1="107" />
      <button name="ep4" x0="191" y0="112" x1="336" y1="139" />
    </spu>
  </stream>
</subpictures>
```

The "<button>" tags define the buttons explicitly (without relying on *spumux*'s autodetection logic). The attributes assign each button a name, and define the precise coordinates of each one.

`name="<name>"`

This attribute assigns each button a logical name which allows it to be referenced later – in particular, when you design the DVD layout for *DVDAuthor*.

Normally, *spumux* assigns each button a logical identifier based on its auto-detected button order (as described above, under the "`autoorder`" setting). When you design the DVD layout using a DVDAuthor XML file (described later), the buttons are automatically matched to these identifiers, based on their order in the XML file. However, if *spumux* is unable to automatically determine the button order, you will have to assign each button an explicit name, and use that to refer to it later.

`x0="<pixels>"`

`y0="<pixels>"`
: These attributes specify the horizontal and vertical coordinates, respectively, of the **top left-hand corner** of the button, as measured in pixels from the **upper left** of the image. Note that this is different from the way OS/2 normally measures coordinates.

`x1="<pixels>"`
`y1="<pixels>"`
: Similarly, these attributes specify the coordinates of the **bottom right-hand corner** of the button. Note that these are absolute coordinates – they are measured from the upper left of the image (and are not relative to the "`x0`" and "`y0`" values).

Once your XML file is finished, you can create the final menu. The following command assumes that the XML file is named "`menu.xml`" and the video file containing the menu background (created in step 5) is "`menu_bg.mpg`".

```
spumux menu.xml < menu_bg.mpg > menu_final.mpg
```

If successful, the file "`menu_final.mpg`" will be created. If *spumux* encounters an error, it should ideally output a message that describes the problem.

I say 'ideally' because the OS/2 port of *spumux* is not entirely perfect. If it encounters a problem during processing, it sometimes crashes with a "SIGABRT" signal instead of returning a meaningful error. This usually indicates one of the following problems:

- The PNG files containing the button images are malformed somehow. This often occurs when they may contain too many colours (remember, you are limited to a total of 3 colours).

- The design and/or layout of the button images is insufficient for *spumux* to be able to auto-detect the button boundaries. If this happens, you may need to adjust the "`autooutline`", "`outlinewidth`", and/or "`autoorder`" settings, or else explicitly specify the button boundaries in the XML file.

Once you have the final menu file, it can be added to the DVD as either a root menu (VMGM) or titleset menu (VTSM). See the next chapter for details.

Designing the DVD Layout

Before you can design your DVD's layout, some discussion of how video DVDs are structured is necessary.

Logical DVD Structure

The logical layout of a video DVD consists of a hierarchical structure which is capable of becoming quite complex.

Overview

A video DVD has the following overall structure:

- Zero or more top-level menus (VMGMs)
- One or more titlesets, each containing:
  - Zero or more titleset menus (VTSMs)
  - One or more titles, each containing:
    - One or more chapters

VMGMs

At the very top of the layout is a top-level (or root) menu, called the **VMGM** (Video Manager Menu). Normally, the VMGM is what will start playing first when you load the DVD into a player. I say "normally", because the VMGM is actually optional (or at least you can define it to have no content), in which case the DVD will start playing at the first titleset (see below) instead.

Conversely, you can have more than one VMGM, in which case the first one will be loaded when the DVD starts playing.

A VMGM has some restrictions in what it can do. Specifically, the buttons on a VMGM can only navigate (or "jump") to one of the following:

- Another VMGM
- The first VTSM within a titleset
- The first title within a titleset

This means that chapter-level menu navigation can only be implemented within a VTSM (see below), at the titleset level.

Titlesets & VTSMs

Titlesets are the next structural level below the VMGM. A video DVD always consists of one or more titlesets, which are essentially groups of separate but related titles (see below). Each titleset can have one or more menus of its own, called **VTSM**s (Virtual Titleset Menus). If a titleset has no VTSM, its first title will start playing automatically when that titleset is selected.

Everything within a single titleset must have the same audio, video and subtitle settings – this includes such things as aspect ratio, language, and resolution.

Titles and Chapters

At the next level down, each titleset consists of one or more **titles**. A title is basically a self-contained video (such as a complete movie), which can be played continuously from beginning to end.

Each title, in turn, consists of one or more **chapters**. Chapters represent the final (bottom-most) level of the DVD structure; they correspond to scenes (or some other kind of logical subdivision) within the title. Chapters are not physically separate: play continues seamlessly from one chapter to the next without interruption.

Chapter divisions are entirely arbitrary. You can place them at meaningful positions within the title (marking major scenes or episodes, for instance), or at fixed time intervals – it's up to you.

Choosing a Layout

The structure used by video DVDs is very flexible and powerful. Unfortunately, this also tends to make things quite complicated, not least because there are multiple different ways of accomplishing almost any task.

In general, it helps to conceptualize the DVD layout by planning upwards from the bottom-most level. Start by determining how you want your chapters and titles organized.

- Do you have multiple video files, or just one?
- If you have multiple videos, do you want to separate them into multiple titles, or just have one continuous video stream?
- Do you want play to continue from one video to the next, or should the DVD return to the menu after playing each one?
- Where do you want your chapter divisions to be? Do you want one chapter per video? Do you want to start new chapters at specific scenes? Do you just want them at fixed time intervals?
- Do all your videos use the same format (e.g. video resolution & aspect ratio, audio format, etc.)?

Whatever features you settle on, it's generally a good idea to implement them in the simplest possible way. Here are some points to keep in mind:

- If you want continuous play from one video to the next, it's usually easiest to make them all chapters within the same title.
- If all your videos have the same video and audio format (i.e. they were all created using the same *FFmpeg* parameters) then they can all go into the same titleset. Videos with different formats, on the other hand (for instance, 4:3 aspect ratio versus 16:9, or DTS audio versus AC3), then they must go into separate titlesets.
- If you only have one titleset, you shouldn't need any VMGM menus. It's easier to just implement a menu at the titleset (VTSM) level.

Creating the DVD Layout

Like the menus earlier, the DVD layout is defined using an XML script file. I'll assume that this file is

called "`dvd.xml`".

Creating the *DVDAuthor* XML file is usually the most difficult part of this whole procedure. This is partly because of its great flexibility: the syntax is unavoidably rather complex. The available documentation is also, regrettably, rather sparse.

The following is a summary of the complete XML file syntax, provided purely for reference.

```
<dvdauthor [dest="output-dir"] [jumppad="1|on|yes"] >
  <vmgm>
    <menus [lang="language-code"] >
      <video [format="ntsc|pal"] [aspect="4:3|16:9"] [resolution="XxY"]
             [caption="field1|field2"] [widescreen="nopanscan|noletterbox"] />
      <audio [format="mp2|ac3|dts|pcm"] [channels="numchannels"]
             [quant="16bps|20bps|24bps|drc"] [dolby="surround"]
             [samplerate="48khz|96khz"] [lang="language"] />
      [<audio ... />]
      <subpicture lang="language" />
      <pgc [entry="title"] [palette="yuvfile|rgbfile"] [pause="seconds|inf"]>
        <pre> commands; </pre>
        <vob file="file.mpg" [chapters="chapter-list"] [pause="seconds|inf"] />
        [<vob ... />]
        <button [name="buttonname"]> commands; </button>
        [<button ... />]
        <post> commands; </post>
      </pgc>
      [<pgc ... />]
    </menus>
  </vmgm>
  <titleset>
    <menus>
      [<video ... />]
      [<audio ... />]
      <pgc [entry="root|subtitle|audio|angle|ptt"]
           [palette="yuvfile|rgbfile"] [pause="seconds|inf"]>
         [...]
      </pgc>
      [<pgc ... />]
    </menus>
    <titles>
      [<video ... />]
      [<audio ... />]
      <pgc [palette="yuvfile|rgbfile"] [pause="seconds|inf"]>
        [...]
      </pgc>
      [<pgc ... />]
    </titles>
  </titleset>
  [<titleset ... />]
</dvdauthor>
```

I'm not going to describe every single one of these tags, parameters, and values. In order to keep things simple, I'll just cover the items we need for the task at hand.

Within the XML file, all instructions are nested inside the "`<dvdauthor>`" tag:

```
<dvdauthor [dest="output-dir"]> ... </dvdauthor>
```

The "dest" parameter specifies the output directory, which is where the DVD file and directory structure will be created by *DVDAuthor*.

The rest of the file is organized into sections according to the different parts of the DVD layout.

VMGM section

The first section defines the VMGM menu(s) using the "<vmgm>" tag (which takes no parameters):

```
<vmgm> ... </vmgm>
```

If you plan on having a VMGM menu, the corresponding menu section (see below) must be placed between these tags. If you don't want any VMGM menus, you can leave this section empty.

(Incidentally, if there's nothing between the opening and closing tags, XML allows you to leave out the closing tag by putting a "/" at the end of the opening tag. For instance, "<vmgm />" is equivalent to "<vmgm></vmgm>")

Titleset sections

After the VMGM section, you need to define at least one titleset section, using the "<titleset>" tag (which also takes no parameters):

```
<titleset> ... </titleset>
```

A titleset section can contain up to one menu section and one title section.

Menu and title sections

A menu section, indicated by the "<menus>" tag, defines one or menus. It can occur inside the VMGM section, in which case it describes the VMGM (root) menu(s) for the DVD; or inside a titleset section, in which case it describes the VTSM (titleset) menu(s) for that particular titleset.

```
<menus [lang="language-code"] > ... </menus>
```

A title section, indicated by the "<titles>" tag, defines all of the titles (one or more) for the current titleset.

```
<titles> ... </titles>
```

Both menu and title sections can contain up to one each of the optional "<video>" and "<audio>" tags, and any number of PGC sections (see below).

The <video> and <audio> tags

The "<video>" tag allows you to manually specify the video settings used in the current menu or titleset. A couple of the more common parameters are shown here; there are others.

```
<video [format="ntsc|pal"] [aspect="4:3|16:9"] [resolution="XxY"] ... />
```

In practice, it is almost never necessary to specify this tag at all, since *DVDAuthor* is intelligent enough to detect the correct settings from the video files.

Similarly, the "`<audio>`" tag allows you to manually specify the audio settings. These include things like format, number of channels, sample rate, and a few others not shown here.

```
<audio [format="mp2|ac3|dts|pcm"] [channels="num"] [samplerate="48khz|96khz"] ... />
```

Again, most of these can be determined automatically by DVDAuthor, so it is rarely necessary to include this tag at all.

PGC sections

A PGC section defines an individual title (when used within a title section) or menu (within a menu section). The name is, admittedly, somewhat unintuitive; to quote from the *DVDAuthor* documentation:

> A PGC is just a fancy term for either a menu or a title. It has a special meaning in the DVD spec so I have retained its use here.
> — DVDAUTHOR(1) man page, February 2004

The "`<pgc>`" tag supports several parameters (all optional), only two of which I've shown here.

```
<pgc [entry="title|root|subtitle|audio|angle|ptt"] [pause="seconds|inf"]> ... </pgc>
```

The "`entry`" parameter is only for use with menus. Most of its allowable values are undocumented. "`entry=title`" can only be used in VMGM menus (where it is the only permissible value); for VTSM menus, "`entry=root`" seems to be used most often.

The "`pause`" parameter is used to determine whether playback (of either the title or the menu) pauses after it finishes playing. This can be a number of seconds (from 1 to 254) or "`inf`", indicating infinity.

A PGC section can contain "`<pre>`", "`<post>`", "`<button>`", and "`<vob>`" tags.

The <pre> and <post> tags

These two tags are used to specify actions which take place, respectively, before and after the execution of the menu or title defined by the current PGC section.

```
<pre> commands; </pre>
<post> commands; </post>
```

Between the opening and closing tags is one or more *DVDAuthor* command statements (separated by semicolons) which define the action(s) to take. The available command statements are described below.

The <button> tag

This tag is used (in menu PGC sections) to define a menu button.

```
<button [name="buttonname"]> commands; </button>
```

The "`buttonname`" parameter is used to explicitly identify the button, if you manually assigned a name to it in your *spumux* XML file (described earlier).

Between the opening and closing tags is one or more *DVDAuthor* command statements (separated by semicolons) which define the action(s) that the button will execute (when selected by the user). The available command statements are described below.

The <vob> tag

Finally, the most important tag of all: the "`<vob>`" tag specifies the actual video file to be played.

```
<vob file="file" [chapters="chapter-list"] [pause="seconds|inf"] />
```

The "`file`" parameter must refer to an existing file in DVD video format (a particular kind of MPEG2 file called a VOB).

The optional "`chapters`" parameter determines how the chapter divisions will be placed within the video(s) that make up the current title. By default, *DVDAuthor* will create one chapter per video file. Specifying a list of comma-separated timestamps here, in the form "`[[HH:]MM:]SS`", will cause chapters to begin at those time offsets within the video instead. If this is done, the first chapter must always begin at timestamp 0 in the first video file. The last timestamp you specify will always mark the beginning of the last chapter, regardless of how many separate video files there are; this effectively allows you to combine multiple video files into a single chapter, by specifying a single timestamp for the first video and none for any of the others.

The "`pause`" parameter can be used to pause video playback for the specified number of seconds once the end of the specified video is reached.

Command statements

The two main commands are "`call`" and "`jump`".

```
call target;
jump target;
```

Both commands do basically the same thing – cause DVD playback to switch to the specified point – the difference is in where and how they can be used:

- Use "`call`" when switching from a title to a menu.
- Use "`jump`" in all other cases.

In both cases, the "`target`" parameter can be any of the following:

`[vmgm | titleset X] menu [Y]`
: Switches to the specified menu. If neither "`vmgm`" nor "`titleset`" is given, the current domain

is used. If the menu number is not given, the default menu is used.

`[vmgm | titleset X] title Y [chapter Z]`
Switches to the specified title. Titles and chapters are both numbered from 1. Note that if "`vmgm`" is given as the domain, all titles on the DVD are accessible. If neither "`vmgm`" nor "`titleset`" is given, the current domain is used.

`chapter Z`
Switches to the specified chapter within the current title.

There is also a "`resume`" command, which allows a menu to return to the title from which the last "`call`" command was issued. It takes no parameters.

Finally, the "`exit`" command can be used to stop DVD playback altogether. This will return you to the default screen of your DVD player.

Examples

1. A single video with no menus

This example XML script shows one of the simplest possible layouts: a single title created from a single source video file, no menus at all. The video consists of a single chapter; when it finishes playing, the DVD will stop.

```
<dvdauthor dest="./output/">
  <vmgm />
  <titleset>
    <titles>
      <pgc>
        <vob file="video.vob" />
        <post>exit;</post>
      </pgc>
    </titles>
  </titleset>
</dvdauthor>
```

2. One chapter from multiple source files

This example illustrates how to override the automatic assignment of chapters according to the source files. In this case, we have two source files which will end up as part of a single chapter. Otherwise, this is much the same as the first example: a single title with no menu.

```
<dvdauthor dest="./output/">
  <vmgm />
  <titleset>
    <titles>
      <pgc>
        <vob file="video1.vob" chapters="0" />
        <vob file="video2.vob" />
        <post>exit;</post>
      </pgc>
```

```
    </titles>
  </titleset>
</dvdauthor>
```

3. One titleset with four chapters and a menu

This example will create a DVD a single title created from four separate video files. The title has four chapters, one for each source file. There is a menu, which is implemented at the titleset (VTSM) level; the menu has four buttons, corresponding to the four chapters.

Note that, because all videos are part of the same title (as individual chapters), playback will automatically continue from one to the next. Once the last chapter has been played, the DVD will return to its menu.

```
<dvdauthor dest="./output/">
  <vmgm />
  <titleset>
    <menus>
      <pgc entry="root" pause="inf">
        <vob file="menu_final.mpg" />
        <button>jump title 1 chapter 1;</button>
        <button>jump title 1 chapter 2;</button>
        <button>jump title 1 chapter 3;</button>
        <button>jump title 1 chapter 4;</button>
      </pgc>
    </menus>
    <titles>
      <pgc>
        <vob file="episode1.vob" />
        <vob file="episode2.vob" />
        <vob file="episode3.vob" />
        <vob file="episode4.vob" />
        <post>call menu;</post>
      </pgc>
    </titles>
  </titleset>
</dvdauthor>
```

4. One titleset with four titles and a menu

This example illustrates a different way of laying out the same content shown in the previous example. Instead of having implementing each video as a chapter within the same title, we create a separate title for each one.

With separate titles, playback does not automatically continue from one to the next. Therefore, we put an explicit "`post`" call at the end of each title, instructing the DVD to start playing the next title right away.

As you can see, this XML file is rather more complicated than the one in the previous example (and in fact has little, if any, practical advantage over it).

```
<dvdauthor dest="./output/">
  <vmgm />
  <titleset>
    <menus>
      <pgc entry="title" pause="inf">
        <vob file="dvd_menu_final.mpg"></vob>
        <button>jump title 1 chapter 1;</button>
        <button>jump title 2 chapter 1;</button>
        <button>jump title 3 chapter 1;</button>
        <button>jump title 4 chapter 1;</button>
      </pgc>
    </menus>
    <titles>
      <pgc pause="2">
        <vob file="episode1.vob"></vob>
        <post>jump title 2 chapter 1;</post>
      </pgc>
      <pgc pause="2">
        <vob file="episode2.vob"></vob>
        <post>jump title 3 chapter 1;</post>
      </pgc>
      <pgc pause="2">
        <vob file="episode3.vob"></vob>
        <post>jump title 4 chapter 1;</post>
      </pgc>
      <pgc pause="2">
        <vob file="episode4.vob"></vob>
        <post>call vmgm menu;</post>
      </pgc>
    </titles>
  </titleset>
</dvdauthor>
```

Authoring the DVD

Writing the *DVDAuthor* XML script is the difficult part. Once your script is ready, you can author the DVD using the command:

```
dvdauthor -x dvd.xml
```

This assumes your XML script includes the "`dest`" parameter in the "`<dvdauthor>`" tag. If not, specify the output directory using the "`-o`" argument on the command line.

You should end up with a directory structure under the output directory that corresponds to the contents of the final video DVD. (This normally consists of two subdirectories: "`AUDIO_TS`" and "`VIDEO_TS`", and their contents.)

Creating the DVD Image

Use *mkisofs2* to create a DVD ISO image which can then be written to disc.

```
mkisofs2 -r -dvd-video -o dvd.iso <path>
```

where "`<path>`" is the name of the output directory from *DVDAuthor* (above).

Writing the DVD

Use *dvddao* (or your favourite DVD-writing software) to burn the DVD.

```
dvddao [options] dvd.iso
```

If you've never used *dvddao*, refer to the documentation for the options you need to specify according to your system configuration. At a minimum, you will probably need the "`--device`" parameter, and perhaps the "`--aspidriver`" parameter as well. I also recommend using the "`--lock`" parameter, to make sure nothing interferes with the writing process.

Standard *dvddao* options can be placed in the file "`%ETC%\dvddao.cfg`" instead of being specified on the command line every time.