May 2006
Vol 4 No. 03

Vegas + DVD Tips, Tricks, and Scripts

Learning and using Vegas and DVD Architect

Revealing a Map Path

By Edward Troxel

When creating a video illustrating moving between multiple points, it is often desirable to show the actual path being traveled. For example, suppose someone wanted to reveal a path across the United States showing a travel plan. This is an effect that can be very easily created in Vegas.

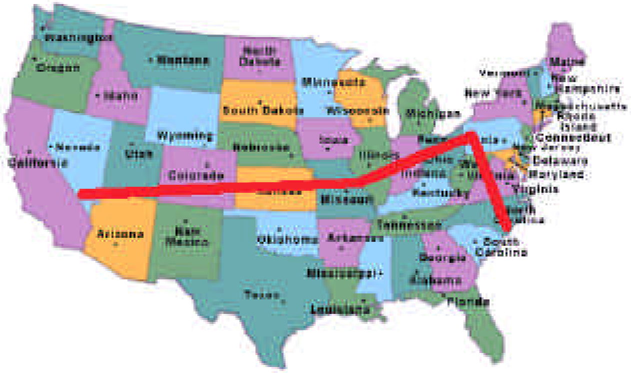
The first step is to create two versions of the map: An original map or background image and a version modified to properly indicate the path. Here are the two images which will be used in this example with the red line indicating the actual path to be revealed.

Now that the images have been created, add them both to the timeline and create a standard crossfade between the two images. This crossfade should be as long as you wish it to take for the line to appear. If you need the line to appear over two seconds, make a two second crossfade.

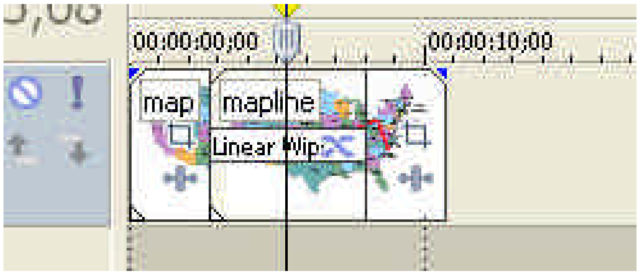
Next, the proper transition must be found for this dissolve. In this case, a linear wipe - left to right - transition seems appropriate. This will let the line slowly appear across the map from the left side toward the right side.

With this transition, it was easy to make the line appear. As long as the line is reasonably straight, this method will work perfectly. However, notice at the right side where the line takes a sudden dip down-

Washington
Oregon
Montana
North Dakota
Minnesota
Idaho
South Dakota
Wisconsin
Michigan
Vermont
New Hampshire
New York
Maine
Massachusetts
Rhode Island
Connecticut
Wyoming
Nevada
California
Utah
Nebraska
Iowa
Illinois
Indiana
Ohio
Pennsylvania
New Jersey
Delaware
Maryland
West Virginia
Virginia
Colorado
Kansas
Missouri
Kentucky
North Carolina
Tennessee
South Carolina
Arizona
New Mexico
Oklahoma
Arkansas
Georgia
Mississippi
Alabama
Texas
Louisiana
Florida

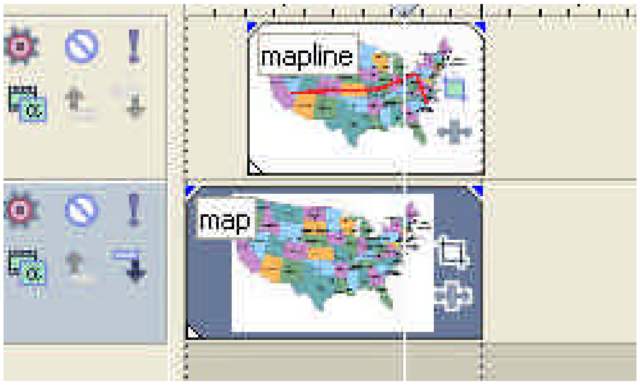


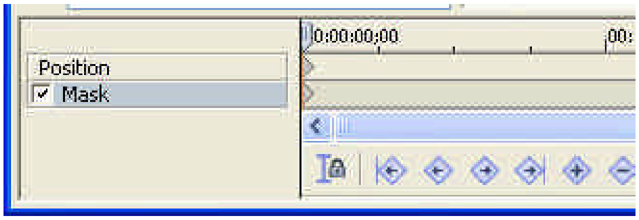
IN THIS ISSUE



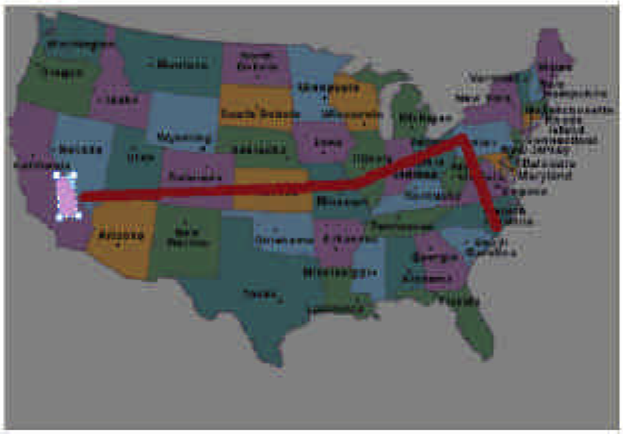
ward. In this case, that portion of the line will suddenly appear much faster. So let's look at an alternate method which will allow better control of how the line will appear.

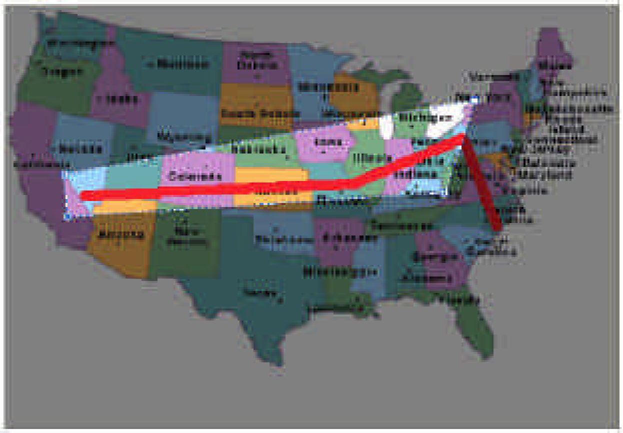
This time put the map with the line on track 1 and the map without the line on track 2. A mask will be used on track 1 to control how the line will appear. To generate the mask, open Pan/Crop for the map with the line and click on "Mask" to the left of the Pan/Crop timeline. This will activate the Bezier Mask feature in Vegas. For more complete details on using the Bezier Mask, look at Vol 3 #03 of this newsletter.



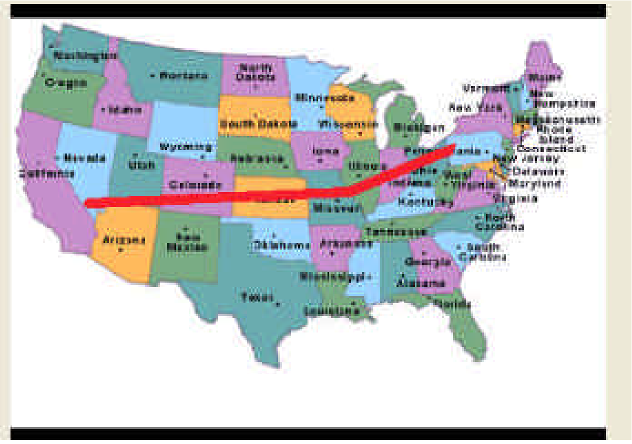


For the first keyframe, a small mask was made allowing none of the line to show through. Throughout the timeline, continue to add more keyframes and extend the Bezier Mask so that more of the line will appear. Notice how between the first and second keyframes that the major part of the line gets drawn. The next keyframe can then be created such that the downward segment of the line is drawn in a downward direction. This will give you full control over how, and in what direction, the line appears.





Notice the final result as the masked line is allowed to appear over the map without the line. Using masks, the line can be any shape. You just adjust the keyframes as needed to control the look and feel.

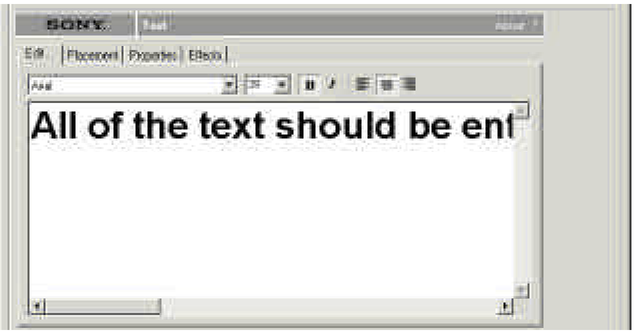


Beginner’s Corner - News Ticker

By Edward Troxel

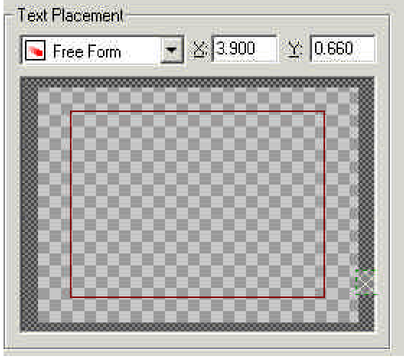
Last issue we saw how to create a credit roll using the standard text generated media. Expanding on this concept, we’ll see how to create a standard news ticker type scroll. A news ticker is typically a single line of text that will scroll across the bottom of the screen. This can be easily created using Vegas.

Start by adding a *Text Generated Media* event to the timeline. Then select all the text, change the font style and size as desired, and type all of the text onto a single text line. Remember that you can still combine multiple fonts, font sizes, and font properties (such as bold and italics) as needed.



All of the text should be ent

Once all of the text has been entered, You should determing how long the text scroll should take just like in the Credit Roll last issue. Then it is now time to position the text so go to the Positioning tab.

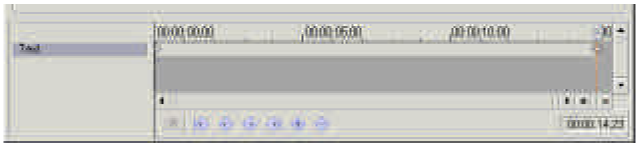

Text Placement

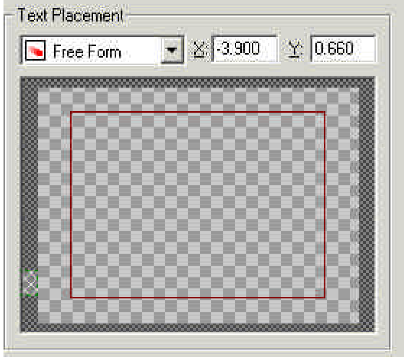
Free Form X: 3.900 Y: 0.660

First, set the position on the screen where the text will scroll. An easy way to do this is to click on the text (which, by default, is in the center of the screen) and then use the up/down arrows to move the text to the proper location. In this example, I moved the text to the bottom of the screen. However, there may be times when you wish the text to scroll across the top, or even the middle, of the screen.

Once the position has been established, the text needs to be moved off the right edge of the screen. Type increasingly larger values into the “X” field until the text is just off the right edge of the screen.

Now the scroll can be created. Add a second keyframe to the end of the timeline at the bottom of the dialog box. For this keyframe, adjust the “X” field so that the text is just off the left edge of the screen. Generally speaking, you can simply add a minus sign in front of the number but it may need to be adjusted slightly. Now the text will scroll from just off the right edge of the screen at the first keyframe to just off the left edge of the screen at the second keyframe.




Text Placement

Free Form X: -3.900 Y: 0.660

Upon playback, you will then have to determine whether the speed is correct. If it not, you can adjust the “Length” value - making sure you always move the last keyframe back to the end of the text dialog timeline (*move the keyframe first when making the time shorter*). Alternately, you could hold down the CTRL key and resize the event or apply a Velocity Envelope to the event.

A news ticker scroll does not have to be difficult. Use a couple of keyframes on a text event to quickly scroll text across the screen. You can even add a background behind it for a quick lower third.

Scripting in C#

By Edward Troxel

Starting with Vegas 4, scripting has allowed adding new functionality in Vegas. These scripts were written in JScript or VBScript. While compiled DLL files could be used, the process was complicated.

Moving to Vegas 6, scripting has become more flexible. First of all, Vegas 6 will allow directly executing DLL files. Beyond that, it can also run C# code without compiling to a DLL. This makes coding scripts in Vegas 6 more flexible and easier by allowing the full power of development environments such as Visual Studio 2003 to be used.

In its simplest form, a C# script is simply a text file saved with a .cs file extension. C# is more strict than JScript and needs a basic framework surrounding the actual code. Let’s start with a simple C# script that will select all events after the cursor.

Use the sample code shown here as your basic outline. The main functionality of the program will be placed in the “FromVegas” routine. This routine also passes the Vegas object to the script. In this example, I added the select events code directly to the FromVegas routine. Alternately, it could have been placed in a separate function called by the FromVegas routine.

In this case, the heart of the routine is the two loops. The first loop goes through all the tracks in the project (notice the brackets { } surround the actual code of the loop):

```
foreach(Track track in myVegas.Project.Tracks)
{
...
}
```

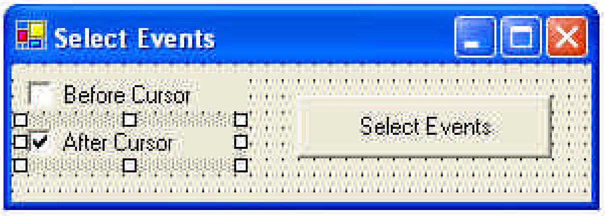
This loop sets a variable named “track” (lower case “t”) of type “Track” (upper case “T”) to each of the tracks in the project. The second loop then goes through every event on the track “track”.

```
foreach(TrackEvent evnt in track.Events)
{
...
}
```

This loop sets a variable named “evnt” of type “TrackEvent”. We can now examine each event on that track. If that event starts at the cursor location or after, we select that event. If the event starts before the cursor location, that event will be deselected. This check is done inside the event loop:

```
//Is it at or after the cursor?
if (evnt.Start >= myVegas.Cursor)
{
   //Yes
   evnt.Selected = true;
}
else
{
   //No
   evnt.Selected = false;
}
```

With these few lines of code, we have now written a useful script in the C# language which can be run directly in Vegas 6. However, suppose we really wanted to display a GUI that allowed picking whether we wanted all that started *after* the cursor, as this one does, or *before* the cursor as well. This can be done fairly easily using Visual Studio 2003 or a development environment such as Sharp Develop (free at: http://www.icsharpcode.net/OpenSource/SD/).



A lot of good info for creating compiled DLL files that will also run in Vegas 4 or 5 can be found on Randall Campbell’s Peachrock Production website: http://www.peachrock.com/software/tutorials/samples.html. However, with Vegas 6 we don’t need the workarounds listed as Vegas 6 can directly run the DLL file.

I typically start a new project of type “Windows

Basic C# script layout with Select Events After Cursor code
(Not all using statements are typically required.)

```
using System;
using System.IO;
using System.Text;
using System.Drawing;
using System.Reflection;
using System.Diagnostics;
using System.Collections;
using System.Windows.Forms;
using System.ComponentModel;
using System.Runtime.InteropServices;
using Microsoft.Win32;
using Sony.Vegas;

public class EntryPoint
{
      Vegas myVegas;

      //Do Script Function here
      public void FromVegas(Vegas vegas)
      {
            myVegas = vegas;
            //Select all events after the cursor location
            //Go through the list of tracks
            foreach(Track track in myVegas.Project.Tracks)
            {
                  //Go through each event on the track
                  foreach(TrackEvent evnt in track.Events)
                  {
                        //Is it at or after the cursor?
                        if (evnt.Start >= myVegas.Cursor)
                        {
                              //Yes
                              evnt.Selected = true;
                        }
                        else
                        {
                              //No
                              evnt.Selected = false;
                        } //if statment
                  } //evnt loop
            } //track loop
      }
}
```

Application" and then open the properties and change the output type to "Class Library" which tells it to compile into a DLL file. Using this method, I can easily draw the screen using the tools provided in Visual Studio.

You also need open the references and add the Vegas scripting dlls. Add references to Sony.Vegas.DLL and Interop.VegasCOM.DLL in the Vegas 6.0 folder.

By adding these, now the "code complete" feature in Visual Studio will work. For example, you type "Vegas." and a list of possible options will appear upon pressing the period.

As before, you need an entrypoint to the program. In the example on page 5, the entire program was simply placed in the entry point. For this version, the entry point simply sets up the form, calls the form, and allows the form to do all the work. Download the full c# example to see exactly how everything is setup and how it works.

C# compiled DLL EntryPoint
(Download the full Visual Studio script from http://www.jetdv.com/tts/sedll.zip)

```
public class EntryPoint
{
        private static SelectEvents.SelectEventsForm form;

        public void FromVegas(Vegas vegas)
        {
                form = new SelectEvents.SelectEventsForm(vegas);
                form.ShowDialog();
        }
}
```

Contact Information

Send your tips, tricks, article ideas, script ideas, questions, or articles to:

editdv@jetdv.com

To locate the index of newsletters and look through the archive, browse to:

www.jetdv.com

Excalibur updated to 5.0.1

An update for Excalibur has been released. Download version 5.0.1 and simply install it over version 5.0.0.

VIDEOGUYS
• FREE Tech Support • 30-Day Money-Back Guarantee • AND GREAT PRICES!