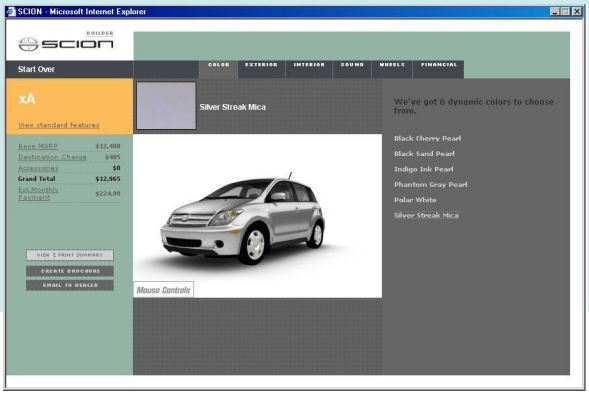
Viewpoint

SceneCapture Quick Reference Guide

Version 3.0.15
February 2004



Introduction

This document describes the features of Viewpoint SceneCapture, a new component of Viewpoint Media Player, and provides guidelines to content developers about using Viewpoint SceneCapture to capture and save images delivered by Viewpoint Media Player.

About Viewpoint SceneCapture

The newly released Viewpoint SceneCapture component introduces Viewpoint Media Player's first data capture functionality, making it possible for you to create and save scene images in a fast and easy manner without the hassle of using print screen functions and image manipulation programs to extract images.

Viewpoint SceneCapture technology can help you improve content authoring, verify how users interact with your online advertisements, and render exciting visual effects in your Viewpoint scenes.

Key Features

Viewpoint SceneCapture strengthens Viewpoint Media Player's position as the most versatile and effective graphics player on the market today, enabling you to:

- Capture scene images at high-resolutions
- Crop and size captured scene images to your specifications
- Capture scene images from multiple camera angles
- Save captured scene images as .jpg files to remote and local locations
- Render mirror effects within a Viewpoint scene

Required Software

- Viewpoint Media Player 3.0.8

> Note: The SceneCapture `Mirror` and `Camera` features require Viewpoint Media Player 3.0.14.

- Any XML text editor, such as XML Spy®

Viewpoint Media Player System Requirements

Windows

- Microsoft® Windows® 95, Windows 98, Windows 2000, Windows Millennium Edition, Windows NT® 4.x, or Windows XP
- Microsoft Internet Explorer® 5.x, Netscape Navigator® 7.x, or AOL® 7.0
- 64 MB RAM
- Pentium® II 300 MHz processor

Macintosh

- Apple® Macintosh® Jaguar™ 10.2
- Safari™ 1.1 or Microsoft® Internet Explorer 5.2
- 128 MB RAM
- PowerPC G3®

Using Viewpoint SceneCapture

Viewpoint SceneCapture enables you to create high-resolution .jpg files that can be cropped, sized, and saved according to your specifications while also allowing you to render planar or even curved mirror effects.

To utilize Viewpoint SceneCapture features in your content, familiarize yourself with the suite of Viewpoint XML functions and attributes that correspond to the Viewpoint `VMPExtremeShot` animation type. This animation type accesses SceneCapture features.

> Note: Find detailed information on `VMPExtremeShot` tags, properties, and functions in the Viewpoint XML Reference Guide.

Introducing VMPExtremeShot

Within your scene .mtx file, access Viewpoint SceneCapture features via the Viewpoint animation type, `VMPExtremeShot`. Like all Viewpoint animators, `VMPExtremeShot` pertains to the `MTSTimeElem` element and is declared in MTX code in the same manner as all `MTSTimeElem` animator types. For example:

```
<MTSTimeElem Type="VMPExtremeShot"
Name="MyImage" />
```

The `VMPExtremeShot` type mainly exports files. Specifically, the SceneCapture component creates and exports .jpg files from a Viewpoint scene. Viewpoint SceneCapture saves all captured scenes as .jpg files.

Compatibility Issues

Though introduced in Viewpoint Media Player 3.0.14, The VMPExtremeShot type is compatible with previous player versions, extending back to Viewpoint Media Player 3.0.8, with the exception of two SceneCapture features, Mirror and Camera, which require Viewpoint Media Player 3.0.14.

About VMPExtremeShot Functions and Attributes

The SceneCapture component includes a robust suite of VMPExtremeShot functions and attributes that you declare separately in your scene .mtx file:

- **Functions** – VMPExtremeShot special functions invoke commands to the SceneCapture component of Viewpoint Media Player. You declare these functions within the MTSInteractor element.
- **Attributes** – VMPExtremeShot attributes are typical Viewpoint XML attributes. You declare VMPExtremeShot attributes within the MTSTimeElem element. Alternatively, you can animate them via the SetProperty JavaScript function.

Using VMPExtremeShot Functions

The scene capture and save features of the SceneCapture component are accessed via these VMPExtremeShot special functions:

- VMPCapture — Captures an image from a Viewpoint scene.
- VMPLocalPersist — Saves captured data from a Viewpoint scene onto the user's local drive.

> Note: When saving captured images locally, a **File Save** dialog box automatically displays, requiring the user to save the file to a local location.

- VMPRemotePersist — Saves captured data from a Viewpoint scene to a remote host.

> Note: To save a captured image to a remote host, you need a special broadcast key and the CGI script posted in the sample below.

Declaring VMPExtremeShot Functions in MTX Code

The particular syntax you use to declare these functions in your scene .mtx file can vary:

- **Recommended syntax** – Valid for Viewpoint Media Player versions 3.0.11 and higher, the VETDispatchCall command calls the VMPExtremeShot special functions. For example:

```
<MTSHandle Action="VETDispatchCall"
 Function="ShotIt::VMPCapture()"
  Event="MouseRightClick" />
```

> Note: For more information on VETDispatchCall and Viewpoint special functions in general, see the Viewpoint XML Reference Guide.

- **Alternative syntax** – Valid for Viewpoint Media Player versions 3.0.8 and higher, the MTSSetProperty tag calls the VMPExtremeShot special functions.
  For example:

```
<MTSSetProperty
Target="capture::DISP"
Value="VMPCapture"
Event=MouseLeftClick" />
```

Using VMPExtremeShot Attributes

The SceneCapture component enables you to specify how a scene image is captured or mirrored via these principal attributes of the VMPExtremeShot animation type:

> Note: For detailed information and examples of VMPExtremeShot features, including those not listed, see the Viewpoint XML Reference Guide.

- AntiAliasingPassesCount — Specifies how many progressive anti-alias passes the SceneCapture component makes on an image being captured.
- Quality — Specifies the quality of the .jpg file that is created from captured content.
  The higher the value assigned to this attribute, the higher the quality AND size of the image.
- Size — Specifies the size of the image that is captured (or mirrored) from Viewpoint Media Player.
- Rect — Crops the image that is created from captured (or mirrored) Viewpoint Media Player content.
- Camera — Specifies an alternative camera to the scene's default camera by which an image is captured (or mirrored).

> Note: The alternativee camera, VETCamera for example, can produce image captures that are different to the camera angle used by the scene when the content contains several cameras.

- Mirror — Creates and displays a mirror image effect of a scene's content.

> Note: This attribute defines the instance that is used as a mirror, which should be a flat plane surface, depending on desired visual effects.

Viewpoint SceneCapture Sample MTX Code

The following MTX code samples illustrates two simple scenes using Viewpoint SceneCapture technology.

Capturing to a Local Host

In the following example, two functions are called in the scene .mtx file, the first, `VMPCapture`, captures the current scene, named Image01, when the user left mouse clicks; and the second, `VMPLocalPersist`, saves the scene capture to the user's local drive when the user right mouse clicks.

Within the `MTSTimeElem` opening and closing tags, the `VMPExtremeShot` type is introduced, specifying the same name of the scene capture as declared in the function above, along with the captured image's resolution, compression, and its cropping parameters.

```
<MTSInteractor>
  <MTSHandle Action="VETDispatchCall"
  Function="Image01::VMPCapture()"
  Event="MouseLeftClick" />
<MTSHandle Action="VETDispatchCall"
Function="Image01::VMPLocalPersist()"
  Event="MouseRightClick" />
</MTSInteractor>

<MTSTimeElem Type="VMPExtremeShot"
Name="Image01" Quality="60"
AntiAliasingPassesCount=”15” Rect=”12
12 80 80” />
```

Capturing to a Remote Host

In the following example, two functions are called in the scene .mtx file, the first, `VMPCapture`, captures the current scene, named Imge02, when the user left mouse clicks; and the second, `VMPRemotePersist`, saves the scene capture to the a remote host when the user right mouse clicks.

Within the `MTSTimeElem` opening and closing tags, the `VMPExtremeShot` type is introduced, specifying the same name of the scene capture as declared in the function above, along with the captured image's resolution, compression, and its cropping parameters. Importantly, here you must specify the path to the CGI script where the folder directory for the images is contained.

```
<MTSInteractor>
  <MTSHandle Action="VETDispatchCall"
  Function=" Imge02::VMPCapture()"
  Event="MouseLeftClick" />
<MTSHandle Action="VETDispatchCall"
Function="Imge02::VMPRemotePersist()"
  Event="MouseRightClick" />
</MTSInteractor>

<MTSTimeElem Type="VMPExtremeShot"
Name=" Imge02" Rect=”12 12 80 80”
Path=””http://mysite.com/cgi-
bin/putdata.pl?myimage2” Quality="60"
AntiAliasingPassesCount=”15” />
```

In addition to a special broadcast key, which can be obtained at sales@viewpoint.com, the following CGI script is required for the remote save to work properly.

Here, the script is contained in a Perl .pl file, but other server-side scripting formats, such as PHP or ASP, also can be used. The script handles the request from the scene .mtx or .html files to save the captured image to the remote host.

In most cases CGI scripts are placed in the /cgi-bin directory of the remote host and set to executable (chmod 755). Based in this information, the path to this script must be referenced either in the `VMPExtremeShot` declaration or via a JavaScript call.

For Perl users, copy this code a .pl file and modify the value of the `$imageDir` function to reflect the directory where you want to save the captured .jpg image.

```
#!/usr/local/bin/perl

$imageDir =
"/web/disk3/virtual/3w3d.com/htdocs/v
et/CaptureMirrorRemote/images/";
$imageExt = ".jpg";

if ($ENV{'REQUEST_METHOD'} eq 'POST')
{
   $fileName = $ENV{'QUERY_STRING'};
   read(STDIN, $buffer,
$ENV{'CONTENT_LENGTH'});
} else {
   &err;
}

$output =
$imageDir.$fileName.$imageExt;
open ( SAVOUT, ">$output" );
binmode ( SAVOUT );
print SAVOUT $buffer;
close ( SAVOUT );

sub err { # print error
   print STDERR "error";
   exit;
}
```