Manual: FDO91 Manual
Chapter 35: Video (VID) Protocol defines the atoms that manage and transmit video and still-camera images between members.
Last updated: April 1997

CHAPTER 35
Video (VID) Protocol

The Video (VID) protocol (protocol ID 53) consists of atoms that execute image capture and transmission tasks to enhance communications methods such as instant messages, chat sessions, and e-mail with slow-motion video and still shots. The protocol supports cameras such as Connectix QuickCam, which provides real-time video pictures, and TWAIN (standard scanner interface) devices for still images, such as the Kodak or Snappy color digital cameras. The protocol also captures .bmp, .gif, .jpg and .art files downloaded from the service or from other local directory paths. For instant messages, the protocol supports a receive view to display incoming images and a preview view to display images before they are sent.

Note that at start-up of the AOL application, the **video** tool searches for compatible video devices. If none are found, it switches to the file mode, which lets members retrieve, attach, and place image files from their local directory.

Atom streams using this protocol must set context to the form display object view or edit_view, which display the video and image files. For more information about form display objects, see Chapter 3, Attribute Manager Protocol.

Video Protocol Atoms

The Video (VID) protocol functions and their associated atoms follow:

Function	**Atoms**
Setting up the **video** tool	atom$vid_is_available atom$vid_setup_popup atom$vid_flag_setup atom$vid_setup
Managing the video cache or scrolling received images (frames)	atom$vid_prev_count atom$vid_next_count atom$vid_show_prev atom$vid_show_next atom$vid_show_old atom$vid_show_new
Capturing images or video frames	atom$vid_can_preview atom$vid_activate_capture atom$vid_capture atom$vid_preview atom$vid_open
Sending images or video frames	atom$vid_manage atom$vid_repeat_send atom$vid_set_token atom$vid_send
Receiving images or video frames	atom$vid_manage atom$vid_receive_enable atom$vid_receive atom$vid_save_as

The VID protocol atoms are described in alphabetical order in the rest of this chapter.

atom$vid_activate_capture
9 ($09)

Description

atom$vid_activate_capture activates the preview view and defines the Capture button and status text field objects of the form.

Syntax

```
atom$vid_activate_capture <raw>
```

`<raw>`	Specifies the relative IDs of both the Capture button and the status text field. The first byte defines the Capture button and the second byte defines the status text field.

Return Value

None.

Example

The following example defines the relative IDs of both the Capture button and the status text field in the capture form:

```
atom$uni_start_stream
atom$man_start_sibling <edit_view>
atom$mat_size <A0x 78x>
atom$mat_relative_tag <101>
atom$vid_manage
➪ atom$vid_activate_capture <79x 06x>
atom$vid_receive <68x 67x 06x 6Cx>
atom$vid_set_token <56x 4Ex 00x 00x 00x 63x>
.
.
.
atom$uni_end_stream
```

In this example, the object relative IDs have been set as follows:

127 (79x) = Capture button

6 (06x) = Send button

atom$vid_can_preview
10 ($0a)

Description

atom$vid_can_preview detects the type and functionality of the capture device. This atom also detects the capture speed and whether the device has an auto-capture feature. This information determines whether to enable the auto-send function as opposed to a manual snapshot trigger.

Syntax

```
atom$vid_can_preview
```

Return Value

Boolean value specifying true or false to the detection of an auto-send capture device. For example, it determines whether the device sends data automatically (such as a QuickCam device) or requires manual intervention to send data (such as a Kodak digital camera).

Example

The following example detects whether there is an auto-capture device or a manually triggered device connected to the client computer and will set the frame refresh rate of any auto-capture device to 250 milliseconds:

```
atom$uni_start_stream
.
.
.
atom$vid_can_preview
atom$if_last_return_true_then <1>
atom$vid_preview <250>
atom$uni_sync_skip <1>
.
.
.
atom$uni_end_stream
```

atom$vid_capture
0 ($00)

Description

atom$vid_capture triggers the **video** tool to capture data from an external video capture device (external camera).

Syntax

```
atom$vid_capture
```

Return Value

None.

Example

The following example captures the current video image displayed in the preview view:

```
  atom$uni_start_stream
  atom$man_set_context_relative <101>
➪ atom$vid_capture
  atom$vid_preview 0
  atom$man_end_context
  atom$uni_end_stream
```


atom$vid_flag_setup
20 ($14)

Description

atom$vid_flag_setup sets a setup flag associated with the view object in context signifying that this object needs new setup data (such as resolution size). This atom works in conjunction with **atom$vid_setup**, which identifies which view has the correct setup data.

Syntax

```
atom$vid_flag_setup
```

Return Value

None.

Example

The following example sets the setup flag to signify that the view object in context needs new setup data:

```
atom$uni_start_stream
.
.
.
atom$man_set_context_relative <104>
➪ atom$vid_flag_setup
atom$vid_is_available
atom$if_last_return_true_then <1, 2>
atom$man_change_context_relative <110>
atom$mat_bool_disabled <no>
atom$uni_sync_skip <1>
atom$man_change_context_relative <110>
atom$mat_bool_disabled <yes>
atom$uni_sync_skip <2>
atom$uni_end_stream
```

atom$vid_is_available
13 ($0D)

Description

atom$vid_is_available returns a true if a video or image capture device is connected to the client computer.

Syntax

```
atom$vid_is_available
```

Return Value

True or false. When a capture device is connected, the value is true.

Example

The following example detects whether a video device is installed on the client computer and will disable a video selection object with a relative ID of 110 if none are found:

```
  atom$uni_start_stream
  atom$man_set_context_relative <104>
➪ atom$vid_is_available
  atom$if_last_return_true_then <1, 2>
  atom$man_change_context_relative <110>
  atom$mat_bool_disabled <no>
  atom$uni_sync_skip <1>
  atom$man_change_context_relative <110>
  atom$mat_bool_disabled <yes>
  atom$uni_sync_skip <2>
  .
  .
  .
  atom$uni_end_stream
```


atom$vid_manage
2 ($02)

Description

atom$vid_manage marks the beginning of a video definition and associates the **video** tool ID as the object in context.

Syntax

```
atom$vid_manage
```

Return Value

None.

Example

The following example establishes the **video** tool ID as the view object in context:

```
atom$man_start_object <edit_view>
atom$mat_size <A0x, 78x>
atom$mat_relative_tag <104>
atom$vid_manage
atom$man_start_sibling <org_group>
```

atom$vid_next_count
15 ($0F)

Description

atom$vid_next_count returns the number of images in cache after the current image on display. The use of this atom is comparable to the scrolling feature within a document or a list box except it supports the viewing of the received images stored in cache.

Syntax

```
atom$vid_next_count
```

Return Value

A numeric value indicating the number of images in cache following the current image.

Example

The following example determines whether there are any images in cache more recent than the one on display. If there are more recent images in cache, a trigger object (button or arrow) with a relative ID of 113 is enabled. If there are no images in cache more recent than the current image on display, the same trigger object is disabled.

```
atom$uni_start_stream
atom$man_set_context_relative 104
atom$vid_prev_count
atom$if_last_return_true_then <1, 2>
atom$man_change_context_relative <110>
atom$mat_bool_disabled <no>
atom$uni_sync_skip <1>
atom$man_change_context_relative <110>
atom$mat_bool_disabled <yes>
atom$uni_sync_skip <2>
atom$man_change_context_relative <104>
➪ atom$vid_next_count
atom$if_last_return_true_then <3, 4>
atom$man_change_context_relative <113>
atom$mat_bool_disabled <no>
atom$uni_sync_skip <3>
atom$man_change_context_relative <113>
atom$mat_bool_disabled <yes>
atom$uni_sync_skip <4>
atom$man_end_context
```



```
.
.
.
atom$uni_end_stream
```

atom$vid_open
22 ($16)

Description

atom$vid_open switches the **video** tool to file mode and opens a file to display an image. The atom retrieves image files (.bmp, .gif, or .jpg) stored in a member's local directory.

Syntax

```
atom$vid_open
```

Return Value

Boolean value. If the file opened successfully, a 1 is returned. If the file did not open or the member canceled the operation, a 0 is returned.

Example

The following example opens an image file stored in a member's local directory and displays it in the preview view:

```
  atom$uni_start_stream
➪ atom$vid_open
  atom$uni_end_stream
```


atom$vid_prev_count
14 ($0E)

Description

atom$vid_prev_count returns the number of images in cache before the current image on display. The use of this atom is comparable to the scrolling features within a document or a list box except it supports the viewing of received images stored in cache.

Syntax

```
atom$vid_prev_count
```

Return Value

A numeric value indicating the number of images in cache before the current image.

Example

The following example determines whether there are any images in cache older than the current one on display. If there are any older images in cache, a trigger object (button or arrow) with a relative ID of 110 is enabled. If there are no images in cache older than the current image on display, the same trigger object is disabled.

```
  atom$uni_start_stream
  atom$man_set_context_relative 104
➪ atom$vid_prev_count
  atom$if_last_return_true_then <1, 2>
  atom$man_change_context_relative <110>
  atom$mat_bool_disabled <no>
  atom$uni_sync_skip <1>
  atom$man_change_context_relative <110>
  atom$mat_bool_disabled <yes>
  atom$uni_sync_skip <2>
  .
  .
  .
  atom$uni_end_stream
```

atom$vid_preview
03 ($03)

Description

atom$vid_preview updates the video capture image in the preview view. It is used only for real-time video cameras.

Syntax

```
atom$vid_preview <dword>
```

`<dword>`	A numeric value specifying the desired frame (refresh) rate in milliseconds. Values are 0 or greater. For example, an entry of 250 (milliseconds) would equal 4 frames per second. A value of 0 turns the preview view off.

Return Value

None.

Example

The following example sets the frame (refresh) rate of the video images for 250 milliseconds:

```
atom$uni_start_stream
.
.
.
atom$vid_can_preview
atom$if_last_return_true_then <1>
➪ atom$vid_preview <250>
atom$uni_sync_skip <1>
atom$vid_capture
.
.
.
atom$uni_end_stream
```


atom$vid_receive
8 ($08)

Description

atom$vid_receive defines the objects on the form, which are otherwise hidden in non-video message mode, for the receiver **video** tool.

Syntax

```
atom$vid_receive <dword>
```

`<dword>` Specifies the relative IDs of the hidden objects of the form. The 4-byte argument is segmented into 4 fields to accept 4 separate values as follows:

Byte 1	Defines the relative ID of the receive view.
Byte 2	Defines the relative ID of the send button.
Byte 3	Defines the relative ID of the status text field.
Byte 4	Defines the relative ID of the auto-send button.

Return Value

None.

Example

The following example assigns relative IDs to the hidden objects of the video form:

```
atom$uni_start_stream
.
.
.
atom$man_start_sibling <edit_view>
atom$mat_size <A0x 78x>
atom$mat_relative_tag <101>
```

```
  atom$vid_manage
  atom$vid_activate_capture <79x 06x>
➪ atom$vid_receive <68x 67x 06x 6Cx>
  atom$vid_set_token <56x 4Ex 00x 00x 00x 63x>
  .
  .
  .
  atom$uni_end_stream
```

In this example, the object relative IDs have been set as follows:

104 (68x) = Receive view

107 (67x) = Send button

6 (06x) = Status text field

108 (6Cx) = Auto-send button

atom$vid_receive_enable
12 ($0C)

Description

atom$vid_receive_enable sets a member's video messaging preference for allowing incoming message forms to display images in the receive view. This atom enables the receive view by assigning a maximum resolution size to this window.

Syntax

```
atom$vid_receive_enable <dword>
```

`<dword>`	Enables, disables or pauses incoming video messages. Values are as follows:	
	00x	Disables incoming video messages.
	01x or greater	Enables incoming video messages, and specifies the limit or maximum resolution size allowed in the receive view.
	FFx FFx FFx FFx	Indicates the pause action is set on the receive view, which temporarily blocks the reception of images from the sender.

Return Value

None.

Example

The following example enables the reception of video messages for a member's account and sets the maximum resolution to a 320 by 240 video window:

```
atom$uni_start_stream
atom$vid_receive_enable <01x 08x 00x F0x>
atom$uni_end_stream
```

atom$vid_repeat_send
4 ($04)

Description

atom$vid_repeat_send controls the refresh rate for the picture(s) being automatically sent to another member. Note that the maximum send rate is limited to the technology of the send and receive connections and devices.

Syntax

```
atom$vid_repeat_send <dword>
```

`<dword>`	Specifies the maximum allowable frame rate in milliseconds for the pictures you want to send. For example, 1000 milliseconds would allow transmissions of up to 1 frame per second. Numeric values are 1 or greater.

Return Value

None.

Example

The following example specifies a maximum frame rate for the pictures being sent to another member. In this case, the maximum frame rate is set to 1000 milliseconds:

```
atom$vid_set_token
atom$vid_activate_capture <07x 06x>
atom$act_replace_select_action
atom$uni_start_stream
atom$man_set_context_relative <108>
atom$man_get_attribute <prot$mat, atom$mat_value>
atom$if_last_return_false_then <1, 2>
atom$mat_value <1>
atom$man_change_context_relative <101>
➪ atom$vid_repeat_send <1000>
atom$uni_sync_skip <1>
atom$mat_value <0>
atom$man_change_context_relative <101>
atom$vid_repeat_send <0>
atom$uni_sync_skip <2>
atom$uni_end_stream
```


atom$vid_save_as
23 ($17)

Description

atom$vid_save_as saves to disk the image on display in the receive view.

Syntax

```
atom$vid_save_as
```

Return Value

None.

Example

The following example saves to disk the image in the receive view:

```
atom$uni_start_stream
.
.
.
atom$man_change_context_relative <104>
atom$vid_save_as
atom$man_end_context
atom$uni_end_stream
```

atom$vid_send
1 ($01)

Description

atom$vid_send sends the captured image data (from **atom$vid_capture**) to the host (to forward to the target receiver). It uses parameters that were set with **atom$vid_set_token**.

Syntax

```
atom$vid_send
```

Return Value

None.

Example

The following example sends captured image data to the host (to forward to the target receiver):

```
  atom$uni_start_stream
  atom$man_set_context_relative <101>
➪ atom$vid_send
  atom$man_end_context
  atom$uni_end_stream
```


atom$vid_set_token
5 ($05)

Description

atom$vid_set_token sends a token and a token argument to the host. This atom establishes a known connection between the form and a host server process.

Syntax

```
atom$vid_set_token <token n>
```

<token>	Specifies the token (2 bytes) to send to the host. Values are a code of 2 ASCII characters. VN (56x 4Ex) is the default token.
<n>	Specifies the numeric token argument (4 bytes) to send to the host server. The default token argument is 00x 00x 00x 63x.

Return Value

None.

Example

The following example sends a VN token with an argument value of 63 hex to establish a connection with the host. In this example, 56x = V and 4Ex = N.

```
atom$uni_start_stream
.
.
.
atom$man_start_sibling <edit_view>
atom$mat_size <A0x 78x>
atom$mat_relative_tag <101>
atom$vid_manage
atom$vid_activate_capture <79x 06x>
atom$vid_receive <68x 67x 06x 6Cx>
➪ atom$vid_set_token <56x 4Ex 00x 00x 00x 63x>
.
.
.
atom$uni_end_stream
```

atom$vid_setup
21 ($15)

Description

atom$vid_setup indicates that the setup data of the view in context should be applied to update any view objects with setup flags. Note that **atom$vid_flag_setup** marks the other views that need this view's setup data.

Syntax

```
atom$vid_setup
```

Return Value

None.

Example

The following example updates view objects flagged for setup data:

```
atom$uni_start_stream
.
.
.
atom$man_set_context_relative <104>
atom$vid_flag_setup
➪ atom$vid_setup
atom$vid_is_available
atom$if_last_return_true_then <1, 2>
atom$man_change_context_relative <110>
atom$mat_bool_disabled <no>
atom$uni_sync_skip <1>
atom$man_change_context_relative <110>
atom$mat_bool_disabled <yes>
atom$uni_sync_skip <2>
atom$man_end_context
atom$uni_end_stream
```


atom$vid_setup_popup
24 ($18)

Description

atom$vid_setup_popup sets the action of a pull-down menu selection or a right mouse button click. It displays the preference setup form.

Syntax

```
atom$vid_setup_popup <rid>
```

`<rid>`	Specifies the relative ID of the object that triggers the pull-down menu.

Return Value

None.

Example

The following example displays the setup form that can be pulled down from a trigger object with a relative ID = 21:

```
atom$uni_start_stream
.
.
.
atom$man_change_context_relative <104>
atom$vid_setup_popup <21>
atom$man_end_context
atom$uni_end_stream
```

atom$vid_show_new
19 ($13)

Description

atom$vid_show_new displays the newest image stored in cache. This atom sets the action of the scroll forward object on the receive view.

Syntax

```
atom$vid_show_new
```

Return Value

None.

Example

The following example displays the newest video image stored in cache:

```
atom$uni_start_stream
.
.
.
atom$man_change_context_relative <104>
atom$vid_show_new
atom$man_end_context
atom$uni_end_stream
```


atom$vid_show_next
17 ($11)

Description

atom$vid_show_next displays the next image stored in cache. This atom sets the action of the scroll forward object on the receive view.

Syntax

```
atom$vid_show_next
```

Return Value

None.

Example

The following example displays the next video image stored in cache:

```
atom$uni_start_stream
.
.
.
atom$man_change_context_relative <104>
atom$vid_show_next
atom$man_end_context
atom$uni_end_stream
```

atom$vid_show_old
18 ($12)

Description

atom$vid_show_old displays the oldest image stored in cache. This atom sets the action of the scroll backward object on the receive view.

Syntax

```
atom$vid_show_old
```

Return Value

None.

Example

The following example displays the oldest video image stored in cache:

```
atom$uni_start_stream
.
.
.
atom$man_change_context_relative <104>
➪ atom$vid_show_old
atom$man_end_context
atom$uni_end_stream
```


atom$vid_show_prev
16 ($10)

Description

atom$vid_show_prev displays the previous image stored in cache. This atom sets the action of the scroll backward object on the receive view.

Syntax

```
atom$vid_show_prev
```

Return Value

None.

Example

The following example displays the previous video image stored in cache:

```
atom$uni_start_stream
.
.
.
atom$man_change_context_relative <104>
atom$vid_show_prev
atom$man_end_context
atom$uni_end_stream
```