Manual: FDO91 Manual
Chapter 20: Host Forms Server Protocol defines host forms server protocol atoms and provides host forms information.
Last updated: February 1998

CHAPTER 20
Host Forms Server (HFS) Protocol

The Host Forms Server (HFS) protocol (protocol ID 51) consists of atoms that handle the communications between the Host Forms Server (HFS) and the **VP Designer** tool running on a client PC.

HFS stores and installs forms to the online service. The HFS protocol contains two groups: attribute atoms and command atoms.

The attribute atoms are sent by **VP Designer** to provide required information (such as field mapping) for HFS to store and install a form. Note that the attribute atoms used by HFS are stripped from the form stream before the form is installed.

The command atoms support a communication protocol between HFS and **VP Designer** that synchronizes the form opening, saving, and installation operations. For example, these commands are used to open a form stored on the host.

Host Forms Server Atoms

The Host Forms Server (HFS) protocol functions and their associated atoms follow:

Function	**Atoms**
Commands:	
Granting access rights	atom$hfs_cmd_access_rights
Opening a form with **ho** token to HFS	atom$hfs_cmd_form_gid atom$hfs_cmd_form_name atom$hfs_cmd_template_name atom$hfs_cmd_tag
Sending the opened form to **VP Designer**	atom$hfs_cmd_start_form_data atom$hfs_cmd_form_gid atom$hfs_cmd_template_name atom$hfs_cmd_read_only_form atom$hfs_cmd_fdo (*with form stream*) atom$hfs_cmd_end_form_data
Saving a form to HFS with **hc** token:	
VP Designer requests HFS to save a form	atom$hfs_cmd_form_name atom$hfs_cmd_template_name atom$hfs_cmd_save_as
HFS requests to receive a form	atom$hfs_cmd_start_get_fdo atom$hfs_cmd_tag atom$hfs_cmd_form_gid atom$hfs_cmd_form_name atom$hfs_cmd_response_id atom$hfs_cmd_reference_id atom$hfs_cmd_end_get_fdo
VP Designer sends a form stream with **he** token to HFS	atom$hfs_cmd_form_name atom$hfs_cmd_gid atom$hfs_cmd_template_name atom$hfs_cmd_tag

HFS replies with the save results	atom$hfs_cmd_response_id atom$hfs_cmd_reference_id atom$hfs_cmd_fdo (*with form stream*) atom$hfs_cmd_start_save_result atom$hfs_cmd_tag atom$hfs_cmd_form_gid atom$hfs_cmd_template_name atom$hfs_cmd_form_name atom$hfs_cmd_result_code atom$hfs_cmd_end_save_result
VP Designer sends request with **hj** token to install form	atom$hfs_cmd_form_name
Attributes:	
Mapping form objects	atom$hfs_attr_checkbox_mapping atom$hfs_attr_field_mapping atom$hfs_attr_variable_mapping
Marking stream areas	atom$hfs_attr_end_in_stream atom$hfs_attr_end_post_stream atom$hfs_attr_end_pre_stream atom$hfs_attr_start_in_stream atom$hfs_attr_start_post_stream atom$hfs_attr_start_pre_stream
Setting flags	atom$hfs_attr_flags atom$hfs_attr_object_flags
Assigning a database	atom$hfs_attr_database_type
Assigning a canned-form server	atom$hfs_attr_server_name
Defining window size and positioning styles	atom$hfs_attr_style_id

Defining a preset global ID	atom$hfs_attr_preset_global_id
Defining a plus group	atom$hfs_attr_plus_group_number atom$hfs_attr_plus_group_type
Naming and commenting objects	atom$hfs_attr_object_comment atom$hfs_attr_layer_name atom$hfs_attr_object_name

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

atom$hfs_attr_checkbox_mapping
30 ($1E)

Description

atom$hfs_attr_checkbox_mapping maps the checkbox or radio button object relative ID to a known text string.

Syntax

```
atom$hfs_attr_checkbox_mapping <dword1, string>
```

`<dword1>`	Specifies the relative ID of the object to be mapped. The 4-byte value is 1 or greater.
`<string>`	Specifies the associated text.

Return Value

None.

Example

The following **VP Designer** example maps object 5 to the text Remote enabled:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
➪ atom$hfs_attr_checkbox_mapping <5, “Remote enabled”>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
```

```
        atom$hfs_attr_object_comment <"The quick brown fox jumps
          over the lazy dog.">
        atom$hfs_attr_object_name <"Julius Caesar">
        atom$hfs_attr_end_object
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```


atom$hfs_attr_database_type
31 ($1F)

Description

atom$hfs_attr_database_type specifies the database where the form will reside.

Syntax

```
atom$hfs_attr_database_type <dword>
```

`<dword>`	Specifies the database where the form will reside. Possible values are:
`0`	Client-based form
`1`	Host-based form

Return Value

None.

Example

The following **VP Designer** example specifies the host database as the storage target for the form:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
➪ atom$hfs_attr_database_type <1>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
```

```
        atom$hfs_attr_start_in_stream
        atom$uni_void
        atom$hfs_attr_end_in_stream
        atom$hfs_attr_object_comment <"The quick brown fox jumps
          over the lazy dog.">
        atom$hfs_attr_object_name <"Julius Caesar">
        atom$hfs_attr_end_object
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```


atom$hfs_attr_end_in_stream
15 ($0F)

Description

atom$hfs_attr_end_in_stream marks the end of the in-stream segment of a form definition. Note that **atom$hfs_attr_start_in_stream** marks the start of the in-stream.

Syntax

```
atom$hfs_attr_end_in_stream
```

Return Value

None.

Example

The following **VP Designer** example marks the end of the in-stream section of the code:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
      .
      .
      .
➪   atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
```

```
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```


atom$hfs_attr_end_post_stream
17 ($11)

Description

atom$hfs_attr_end_post_stream marks the end of the post-stream segment of a form definition. Note that **atom$hfs_attr_start_post_stream** marks the start of the post-stream.

Syntax

```
atom$hfs_attr_end_post_stream
```

Return Value

None.

Example

The following **VP Designer** example marks the end of the post-stream section of the code:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
    atom$hfs_attr_start_post_stream
      .
```

```
          .
          .
➪      atom$hfs_attr_end_post_stream
  atom$uni_end_stream
```


atom$hfs_attr_end_pre_stream
13 ($0D)

Description

atom$hfs_attr_end_pre_stream marks the end of the pre-stream segment of a form definition. Note that **atom$hfs_attr_start_pre_stream** marks the start of the pre-stream.

Syntax

```
atom$hfs_attr_end_pre_stream
```

Return Value

None.

Example

The following **VP Designer** example marks the end of the pre-stream section of the code:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  .
  .
  .
➪ atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
```

```
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```


atom$hfs_attr_field_mapping
0 ($00)

Description

atom$hfs_attr_field_mapping maps a specified object to a field on the form for the host server.

Syntax

```
atom$hfs_attr_field_mapping <dword1, dword2>
```

`<dword1>`	Specifies the relative ID of the object to be mapped. The 4-byte value is 1 or greater.
`<dword2>`	Specifies the field mapping data. The 4-byte value is 1 or greater.

Return Value

None.

Example

The following **VP Designer** example maps object 1 to field 16:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
```

```
          over the lazy dog.">
        atom$hfs_attr_object_name <"Julius Caesar">
        atom$hfs_attr_end_object
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
atom$uni_end_stream
```


atom$hfs_attr_flags
6 ($06)

Description

atom$hfs_attr_flags sets flags to define various attributes on a form.

Syntax

```
atom$hfs_attr_flags <dword>
```

`<dword>` Specifies an attribute flag to be set. Flag values are:

- `2` `insert_global_id` — Specifies that a form has a preset global ID.
- `128` `direct_accessible` — Specifies that a form has a Favorite Places heart (icon).

Return Value

None.

Example

The following **VP Designer** example specifies that the form has a preset global ID:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
➪ atom$hfs_attr_flags <2>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_plus_group_number <34>
  atom$hfs_attr_plus_group_type <2>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
```

atom$hfs_attr_layer_name
39 ($27)

Description

atom$hfs_attr_layer_name defines the current object as a layer-defined object within **VP Designer**. This atom also specifies the object name.

Syntax

```
atom$hfs_attr_layer_name <string>
```

`<string>` Specifies the layered object name.

Return Value

None.

Example

The following **VP Designer** example defines the current object as a layered object called Foobar:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <org_group, "">
➪   atom$hfs_attr_layered_name <”Foobar”>
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$man_end_object
```



```
        atom$hfs_attr_end_object
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```

atom$hfs_attr_object_comment
34 ($22)

Description

atom$hfs_attr_object_comment specifies the comment text that describes the current object in the stream.

Syntax

```
atom$hfs_attr_object_comment <string>
```

`<string>` Specifies the text of the comment.

Return Value

None.

Example

The following **VP Designer** example specifies the comment text that describes the current object:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
➪   atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
    atom$hfs_attr_start_post_stream
```



```
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```

atom$hfs_attr_object_flags
44 ($2C)

Description

atom$hfs_attr_object_flags sets flags to define various attributes on a form.

Syntax

```
atom$hfs_attr_object_flags <dword>
```

`<dword>` Specifies an object flag to be set. Flag value is:

`1` `slideshow` — Specifies that the object is a graphic view.

Return Value

None.

Example

The following **VP Designer** example specifies that the current object is a graphic-view object:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$man_start_object <view, "">
➪     atom$hfs_attr_object_flags <1>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
```



```
        atom$hfs_attr_object_name <"Julius Caesar">
        atom$hfs_attr_end_object
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```

atom$hfs_attr_object_name
35 ($23)

Description

atom$hfs_attr_object_name specifies the name of the current object in the form.

Syntax

```
atom$hfs_attr_object_name <string>
```

`<string>` Specifies the name of the current object.

Return Value

None.

Example

The following **VP Designer** example specifies the name (Julius Caesar) to the current object:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
➪   atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
    atom$man_end_object
```



```
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```

atom$hfs_attr_plus_group_number
37 ($25)

Description

atom$hfs_attr_plus_group_number defines a plus group number for the form.

Syntax

```
atom$hfs_attr_plus_group_number <dword>
```

`<dword>`	Specifies the plus group number. Values are 1 or greater.

Return Value

None.

Example

The following **VP Designer** example specifies the plus group number 34 for the form:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
➪   atom$hfs_attr_plus_group_number <34>
    atom$hfs_attr_plus_group_type <2>
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
```



```
        atom$hfs_attr_object_name <"Julius Caesar">
        atom$hfs_attr_end_object
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```

atom$hfs_attr_plus_group_type
38 ($26)

Description

atom$hfs_attr_plus_group_type defines a plus group type for the form.

Syntax

```
atom$hfs_attr_plus_group_type <dword>
```

`<dword>`	Specifies the plus group type. Values are:	
	0	Same
	1	Pay area
	2	Free area

Return Value

None.

Example

The following **VP Designer** example specifies the plus group type (free) for the form:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$hfs_attr_plus_group_number <34>
➪   atom$hfs_attr_plus_group_type <2>
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
```



```
        atom$hfs_attr_start_in_stream
        atom$uni_void
        atom$hfs_attr_end_in_stream
        atom$hfs_attr_object_comment <"The quick brown fox jumps
          over the lazy dog.">
        atom$hfs_attr_object_name <"Julius Caesar">
        atom$hfs_attr_end_object
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```

atom$hfs_attr_preset_global_id
4 ($04)

Description

atom$hfs_attr_preset_global_id requests that the host assign a preset global ID (GID) for the current form.

Syntax

```
atom$hfs_attr_preset_global_id <dword>
```

`<dword>` Specifies the GID of the current form.

Return Value

None.

Example

The following **VP Designer** example requests a preset GID (0-43-4535) for the current form:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_preset_global_id <0-43-4535>
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
```


atom$hfs_attr_server_name
41 ($29)

Description

atom$hfs_attr_server_name specifies the name of the canned form server.

Syntax

```
atom$hfs_attr_server_name <string>
```

`<string>` Specifies the server name holding the canned forms.

Return Value

None.

Example

The following **VP Designer** example specifies the name of the server (Rainman) that holds the canned forms:

```
atom$uni_start_stream_wait_on
➪   atom$hfs_attr_server_name <”Rainman”>
    atom$hfs_attr_field_mapping <1, 16>
    atom$hfs_attr_variable_mapping <4, 20>
    atom$hfs_attr_style_id <0-43-1234>
    atom$hfs_attr_server_name <"Rainman">
    atom$hfs_attr_start_pre_stream
    atom$uni_void
    atom$hfs_attr_end_pre_stream
    atom$man_start_object <ind_group, "Form">
      atom$hfs_attr_layered_object <”Foobar”>
      atom$mat_orientation <vff>
      atom$mat_precise_width <500>
      atom$mat_precise_height <300>
      atom$mat_bool_precise <yes>
      atom$mat_bool_resize_vertical <no>
      atom$mat_bool_resize_horizontal <no>
      atom$mat_relative_tag <1>
      atom$mat_dirty_query <yes>
      atom$hfs_attr_start_in_stream
      atom$uni_void
      atom$hfs_attr_end_in_stream
      atom$hfs_attr_object_comment <"The quick brown fox jumps
        over the lazy dog.">
      atom$hfs_attr_object_name <"Julius Caesar">
      atom$hfs_attr_end_object
```

```
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```


atom$hfs_attr_start_in_stream
14 ($0E)

Description

atom$hfs_attr_start_in_stream marks the start of the in-stream segment of a form definition. Note that **atom$hfs_attr_end_in_stream** marks the end of the in-stream.

Syntax

```
atom$hfs_attr_start_in_stream
```

Return Value

None.

Example

The following **VP Designer** example marks the start of the in-stream section of the code:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
➪   atom$hfs_attr_start_in_stream
      .
      .
      .
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
```

```
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```


atom$hfs_attr_start_post_stream
16 ($10)

Description

atom$hfs_attr_start_post_stream marks the start of the post-stream segment of a form definition. Note that **atom$hfs_attr_end_post_stream** marks the end of the post-stream.

Syntax

```
atom$hfs_attr_start_post_stream
```

Return Value

None.

Example

The following **VP Designer** example marks the start of the post-stream section of the code:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
➪   atom$hfs_attr_start_post_stream
      .
```

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


atom$hfs_attr_start_pre_stream
12 ($0C)

Description

atom$hfs_attr_start_pre_stream marks the start of the pre-stream segment of a form definition. Note that **atom$hfs_attr_end_pre_stream** marks the end of the pre-stream.

Syntax

```
atom$hfs_attr_start_pre_stream
```

Return Value

None.

Example

The following **VP Designer** example marks the start of the pre-stream section of the code:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
➪ atom$hfs_attr_start_pre_stream
  .
  .
  .
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
```

```
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```


atom$hfs_attr_style_id
4 ($04)

Description

atom$hfs_attr_style_id defines the global ID (GID) of an existing form to apply as the style ID for the window size and positioning of the new form.

Syntax

```
atom$hfs_attr_style_id <dword>
```

`<dword>` Specifies the GID to apply as the form style.

Return Value

None.

Example

The following **VP Designer** example specifies GID 0-43-1234 as the style of the new form:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
  atom$hfs_attr_variable_mapping <4, 20>
➪ atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
      over the lazy dog.">
    atom$hfs_attr_object_name <"Julius Caesar">
    atom$hfs_attr_end_object
    atom$hfs_attr_start_post_stream
```

```
        atom$uni_void
        atom$hfs_attr_end_post_stream
    atom$uni_end_stream
```


atom$hfs_attr_variable_mapping
28 ($1C)

Description

atom$hfs_attr_variable_mapping maps a specified variable object to a data value on the form for the host variables.

Syntax

```
atom$hfs_attr_variable_mapping <dword1, dword2>
```

`<dword1>`	Specifies the variable object ID of the object to be mapped. The 4-byte value is 1 or greater.
`<dword2>`	Specifies the variable value. The 4-byte value is 1 or greater.

Return Value

None.

Example

The following **VP Designer** example maps variable object 4 to a value of 20:

```
atom$uni_start_stream_wait_on
  atom$hfs_attr_field_mapping <1, 16>
➪ atom$hfs_attr_variable_mapping <4, 20>
  atom$hfs_attr_style_id <0-43-1234>
  atom$hfs_attr_server_name <"Rainman">
  atom$hfs_attr_start_pre_stream
  atom$uni_void
  atom$hfs_attr_end_pre_stream
  atom$man_start_object <ind_group, "Form">
    atom$mat_orientation <vff>
    atom$mat_precise_width <500>
    atom$mat_precise_height <300>
    atom$mat_bool_precise <yes>
    atom$mat_bool_resize_vertical <no>
    atom$mat_bool_resize_horizontal <no>
    atom$mat_relative_tag <1>
    atom$mat_dirty_query <yes>
    atom$hfs_attr_start_in_stream
    atom$uni_void
    atom$hfs_attr_end_in_stream
    atom$hfs_attr_object_comment <"The quick brown fox jumps
```

```
          over the lazy dog.">
        atom$hfs_attr_object_name <"Julius Caesar">
        atom$hfs_attr_end_object
        atom$hfs_attr_start_post_stream
        atom$uni_void
        atom$hfs_attr_end_post_stream
atom$uni_end_stream
```


atom$hfs_cmd_access_rights
42 ($2A)

Description

atom$hfs_cmd_access_rights is sent by HFS to the client so that **VP Designer** can be run.

Note: The access rights to protected areas of the service, such as **VP Designer**, are granted by a Host Security System (HSS) server. For more information about HSS, see the *Host Subsystems* manual.

Syntax

```
atom$hfs_cmd_access_rights <dword1 dword2>
```

`<dword1 dword2>`	The first 4 bytes (dword1) specifies a security rights number. The second 4 bytes (dword2) specifies a rights value.

Return Value

None.

Example

The following HFS example grants the client access to **VP Designer**:

```
atom$uni_start_stream
  atom$hfs_cmd_access_rights <8 bytes of encrypted values>
atom$uni_end_stream
```

atom$hfs_cmd_end_form_data
19 ($13)

Description

atom$hfs_cmd_end_form_data is sent by HFS to indicate the end of a load form sequence. A load form sequence from HFS is a response to a **VP Designer** request to load a form from the host. The sequence contains a series of **hfs_cmd** atoms and object definitions that send the form from the host to the client. Note that an **atom$hfs_cmd_start_form_data** is required at the start of the load form atom sequence.

Syntax

```
atom$hfs_cmd_end_form_data
```

Return Value

None.

Example

The following HFS example marks the beginning of a load form sequence:

```
atom$uni_start_stream
atom$hfs_cmd_start_form_data
atom$hfs_cmd_form_gid <43-5931>
atom$hfs_cmd_form_name <tim_black2>
atom$hfs_cmd_template_name <”1”>
atom$hfs_cmd_read_only_form <no>
atom$hfs_fdo
  <
  atom$uni_start_stream
    atom$man_start_object <ind_group, “Inky Blackness”>
      atom$mat_orientation <vff>
      atom$mat_precise_width <500>
      atom$mat_precise_height <300>
      atom$mat_bool_precise <yes>
      atom$mat_bool_resize_vertical <no>
      atom$mat_bool_resize_horizontal <no>
      atom$mat_color_face <1, 0, 0>
      atom$mat_bool_background_flood <yes>
      atom$mat_start_object <ornament, “The color Black”>
        atom$mat_precise_x <228>
        atom$mat_precise_y <130>
        atom$mat_orientation <hlf>
        atom$mat_font_sis <arial, 9, normal>
        atom$mat_color_text <225, 255, 255>
```



```
          atom$man_end_object
        atom$man_update_display
      atom$man_update_woff_end_stream
      >
➪  atom$hfs_cmd_end_form_data
 atom$uni_end_stream
```

atom$hfs_cmd_end_get_fdo

23 ($17)

Description

atom$hfs_cmd_end_get_fdo is sent by HFS to mark the end of a form definition request sequence. Note that **atom$hfs_cmd_start_get_fdo** is required at the start of the request sequence.

Syntax

```
atom$hfs_cmd_end_get_fdo
```

Return Value

None.

Example

The following HFS example marks the end of a form display object definition request sequence:

```
atom$uni_start_stream
  atom$hfs_cmd_start_get_fdo
  atom$hfs_cmd_tag <138366636>
  atom$hfs_cmd_form_gid <0>
  atom$hfs_cmd_form_name <tim_black2>
  atom$hfs_cmd_template_name <"default">
  atom$hfs_cmd_response_id <28>
  atom$hfs_cmd_reference_id <2>
➪ atom$hfs_cmd_end_get_fdo
atom$uni_end_stream
```


atom$hfs_cmd_end_save_result
21 ($15)

Description

atom$hfs_cmd_end_save_result is sent by HFS to indicate the end of a save form sequence. A save form sequence from HFS is a response to a **VP Designer** token (**he**) request, which contains a series of **hfs_cmd** atoms that save a form. Note that **atom$hfs_cmd_start_save_result** is required at the beginning of the save form sequence.

Syntax

```
atom$hfs_cmd_end_save_result
```

Return Value

None.

Example

The following HFS to **VP Designer** example marks the end of a save form sequence:

```
atom$uni_start_stream
   atom$hfs_cmd_start_save_result
   atom$hfs_cmd_tag <138366636>
   atom$hfs_cmd_form_gid <0>
   atom$hfs_cmd_form_name <tim_black2>
   atom$hfs_cmd_template_name <"default">
   atom$hfs_cmd_result_code <0>
➪  atom$hfs_cmd_end_save_result
atom$uni_end_stream
```

atom$hfs_cmd_fdo
7 ($07)

Description

atom$hfs_cmd_fdo is sent by HFS to send a specific atom stream that defines the form.

Syntax

```
atom$hfs_cmd_fdo <stream>
```

`<stream>` Contains an atom stream that defines the form.

Return Value

None.

Example

The following HFS to **VP Designer** example contains the stream that defines the form:

```
atom$uni_start_stream
atom$hfs_cmd_start_form_data
atom$hfs_cmd_form_gid <43-5931>
atom$hfs_cmd_form_name <tim_black2>
atom$hfs_cmd_template_name <”1”>
atom$hfs_cmd_read_only_form <no>
➪ atom$hfs_cmd_fdo
    <
    atom$uni_start_stream
      atom$man_start_object <ind_group, “Inky Blackness”>
        atom$mat_orientation <vff>
        atom$mat_precise_width <500>
        atom$mat_precise_height <300>
        atom$mat_bool_precise <yes>
        atom$mat_bool_resize_vertical <no>
        atom$mat_bool_resize_horizontal <no>
        atom$mat_color_face <1, 0, 0>
        atom$mat_bool_background_flood <yes>
        atom$mat_start_object <ornament, “The color Black”>
          atom$mat_precise_x <228>
          atom$mat_precise_y <130>
          atom$mat_orientation <hlf>
          atom$mat_font_sis <arial, 9, normal>
          atom$mat_color_text <225, 255, 255>
        atom$man_end_object
```



```
        atom$man_update_display
      atom$man_update_woff_end_stream
      >
    atom$hfs_cmd_end_form_data
atom$uni_end_stream
```

atom$hfs_cmd_form_gid
1 ($01)

Description

atom$hfs_cmd_form_gid is sent by **VP Designer** to specify a global ID (GID) of the form to open.

Syntax

```
atom$hfs_cmd_form_gid <dword>
```

`<dword>` Specifies the GID of the form.

Return Value

None.

Example

The following **VP Designer** to HFS example specifies GID 0-43-4535 of a form to open:

```
atom$uni_start_stream
  atom$hfs_cmd_form_gid <0-43-4535>
  atom$hfs_cmd_form_name <tim_black2>
  atom$hfs_cmd_template_name <"default">
  atom$hfs_cmd_tag <9688552>
atom$uni_end_stream
```


atom$hfs_cmd_form_name
2 ($02)

Description

atom$hfs_cmd_form_name is sent by **VP Designer** to assign a name to a form, or it is sent by HFS to define the name of the form as the target context of other command streams.

Syntax

```
atom$hfs_cmd_form_name <string>
```

`<string>` Specifies the name of the form.

Return Value

None.

Example

The following **VP Designer** to HFS example assigns the name tim_black2 to a form:

```
atom$uni_start_stream
   atom$hfs_cmd_form_gid <0-0-0>
➪  atom$hfs_cmd_form_name <tim_black2>
   atom$hfs_cmd_template_name <"default">
   atom$hfs_cmd_tag <9688552>
atom$uni_end_stream
```

atom$hfs_cmd_read_only_form
40 ($28)

Description

atom$hfs_cmd_read_only_form determines whether the form is opened in a read-only state.

Syntax

```
atom$hfs_cmd_read_only_form <boolean>
```

`<boolean>`	Specifies whether the form is opened in a read-only state. Values are:	
	`Yes`	The form is read-only.
	`No`	The form is editable. (Default)

Return Value

None.

Example

The following **VP Designer** example specifies that the form is read-only:

```
atom$uni_start_stream
atom$hfs_cmd_start_form_data
atom$hfs_cmd_read_only_form <yes>
atom$hfs_cmd_form_gid <43-5931>
atom$hfs_cmd_form_name <tim_black2>
atom$hfs_cmd_template_name <”1”>
atom$hfs_cmd_read_only_form <no>
atom$hfs_fdo
  <
  atom$uni_start_stream
    atom$man_start_object <ind_group, “Inky Blackness”>
      atom$mat_orientation <vff>
      atom$mat_precise_width <500>
      atom$mat_precise_height <300>
      atom$mat_bool_precise <yes>
      atom$mat_bool_resize_vertical <no>
      atom$mat_bool_resize_horizontal <no>
      atom$mat_color_face <1, 0, 0>
      atom$mat_bool_background_flood <yes>
      atom$mat_start_object <ornament, “The color Black”>
```



```
            atom$mat_precise_x <228>
            atom$mat_precise_y <130>
            atom$mat_orientation <hlf>
            atom$mat_font_sis <arial, 9, normal>
            atom$mat_color_text <225, 255, 255>
          atom$man_end_object
        atom$man_update_display
      atom$man_update_woff_end_stream
      >
  atom$hfs_cmd_end_form_data
atom$uni_end_stream
```

atom$hfs_cmd_reference_id

32 ($20)

Description

atom$hfs_cmd_reference_id is sent by HFS and **VP Designer** to track commands in the form-building sequences.

Syntax

```
atom$hfs_cmd_reference_id <dword>
```

`<dword>`	Specifies the reference ID in the form-building sequence. Integer values are 0 or greater.

Return Value

None.

Example

The following HFS example identifies the reference ID (2) in an object definition request to **VP Designer**:

```
atom$uni_start_stream
  atom$hfs_cmd_start_get_fdo
  atom$hfs_cmd_tag <138366636>
  atom$hfs_cmd_form_gid <0>
  atom$hfs_cmd_form_name <tim_black2>
  atom$hfs_cmd_template_name <"default">
  atom$hfs_cmd_response_id <28>
➪ atom$hfs_cmd_reference_id <2>
  atom$hfs_cmd_end_get_fdo
atom$uni_end_stream
```


atom$hfs_cmd_response_id
27 ($1B)

Description

atom$hfs_cmd_response_id is sent by HFS and **VP Designer** to track commands in the form-building sequences.

Syntax

```
atom$hfs_cmd_response_id <dword>
```

`<dword>`	Specifies the response ID in the form-building sequence. Integer values are 0 or greater.

Return Value

None.

Example

The following HFS example identifies the response ID (28) in an object definition request to **VP Designer**:

```
atom$uni_start_stream
   atom$hfs_cmd_start_get_fdo
   atom$hfs_cmd_tag <138366636>
   atom$hfs_cmd_form_gid <0>
   atom$hfs_cmd_form_name <tim_black2>
   atom$hfs_cmd_template_name <"default">
➪  atom$hfs_cmd_response_id <28>
   atom$hfs_cmd_reference_id <2>
   atom$hfs_cmd_end_get_fdo
atom$uni_end_stream
```

atom$hfs_cmd_result_code
10 ($0A)

Description

atom$hfs_cmd_result_code is sent by HFS to reply to **VP Designer** with the results of the operation.

Syntax

```
atom$hfs_cmd_result_code <dword>
```

`<dword>`	Specifies the result code. A zero indicates a successful operation. Values 1 or greater are error codes.

Return Value

None.

Example

The following HFS example replies to **VP Designer** with a zero indicating a successful save operation:

```
atom$uni_start_stream
  atom$hfs_cmd_start_save_result
  atom$hfs_cmd_tag <9688552>
  atom$hfs_cmd_form_gid <0-0-0>
  atom$hfs_cmd_template_name <”default”>
  atom$hfs_cmd_form_name <tim_black2>
➪ atom$hfs_cmd_result_code <0>
  atom$hfs_cmd_end_save_result
atom$uni_end_stream
```


atom$hfs_cmd_save_as
11 ($0B)

Description

atom$hfs_cmd_save_as is sent by **VP Designer** to request a save operation. Note that the host responds to this request with a prompt for a form name.

Syntax

```
atom$hfs_cmd_save_as
```

Return Value

None.

Example

The following **VP Designer** example requests HFS to save a form:

```
atom$uni_start_stream
   atom$hfs_cmd_form_gid <0-0-0>
   atom$hfs_cmd_form_name <tim_black2>
   atom$hfs_cmd_template_name <"default">
   atom$hfs_cmd_tag <9688552>
➪  atom$hfs_cmd_save_as
atom$uni_end_stream
```

atom$hfs_cmd_start_form_data
18 ($12)

Description

atom$hfs_cmd_start_form_data is sent by HFS to indicate the start of a load form sequence. A load form sequence from HFS is a response to a **VP Designer** request to load a form from the host. The sequence contains a series of **hfs_cmd** atoms and object definitions that send the form from the host to the client. Note that **atom$hfs_cmd_end_form_data** is required at the end of the load form atom sequence.

Syntax

```
atom$hfs_cmd_start_form_data
```

Return Value

None.

Example

The following HFS to **VP Designer** example marks the beginning of a load form sequence:

```
  atom$uni_start_stream
➪ atom$hfs_cmd_start_form_data
  atom$hfs_cmd_form_gid <43-5931>
  atom$hfs_cmd_form_name <tim_black2>
  atom$hfs_cmd_template_name <”1”>
  atom$hfs_cmd_read_only_form <no>
  atom$hfs_fdo
    <
    atom$uni_start_stream
      atom$man_start_object <ind_group, “Inky Blackness”>
        atom$mat_orientation <vff>
        atom$mat_precise_width <500>
        atom$mat_precise_height <300>
        atom$mat_bool_precise <yes>
        atom$mat_bool_resize_vertical <no>
        atom$mat_bool_resize_horizontal <no>
        atom$mat_color_face <1, 0, 0>
        atom$mat_bool_background_flood <yes>
        atom$mat_start_object <ornament, “The color Black”>
          atom$mat_precise_x <228>
          atom$mat_precise_y <130>
          atom$mat_orientation <hlf>
```



```
            atom$mat_font_sis <arial, 9, normal>
            atom$mat_color_text <225, 255, 255>
          atom$man_end_object
        atom$man_update_display
      atom$man_update_woff_end_stream
      >
    atom$hfs_cmd_end_form_data
atom$uni_end_stream
```

atom$hfs_cmd_start_get_fdo
22 ($16)

Description

atom$hfs_cmd_start_get_fdo is sent by HFS to mark the beginning of the request for the form definitions. Note that **atom$hfs_cmd_end_get_fdo** is required at the end of the request sequence.

Syntax

```
atom$hfs_cmd_start_get_fdo
```

Return Value

None.

Example

The following HFS to **VP Designer** example marks the beginning of a form display object definition request sequence:

```
atom$uni_start_stream
➪   atom$hfs_cmd_start_get_fdo
    atom$hfs_cmd_tag <138366636>
    atom$hfs_cmd_form_gid <0>
    atom$hfs_cmd_form_name <tim_black2>
    atom$hfs_cmd_template_name <"default">
    atom$hfs_cmd_response_id <28>
    atom$hfs_cmd_reference_id <2>
    atom$hfs_cmd_end_get_fdo
atom$uni_end_stream
```


atom$hfs_cmd_start_save_result
20 ($14)

Description

atom$hfs_cmd_start_save_result is sent by HFS to indicate the start of a save form sequence. A save sequence from HFS is a response to a **VP Designer** token (**he**) request, which contains a series of **hfs_cmd** atoms that save a form. Note that **atom$hfs_cmd_end_save_result** is required at the end of the save form sequence.

Syntax

```
atom$hfs_cmd_start_save_result
```

Return Value

None.

Example

The following HFS to **VP Designer** example marks the beginning of a save form sequence:

```
atom$uni_start_stream
➪   atom$hfs_cmd_start_save_result
    atom$hfs_cmd_tag <138366636>
    atom$hfs_cmd_form_gid <0>
    atom$hfs_cmd_form_name <tim_black2>
    atom$hfs_cmd_template_name <"default">
    atom$hfs_cmd_result_code <0>
    atom$hfs_cmd_end_save_result
atom$uni_end_stream
```

atom$hfs_cmd_tag
8 ($08)

Description

atom$hfs_cmd_tag is sent by **VP Designer** to assign an ID to track commands being sent.

Syntax

```
atom$hfs_cmd_tag <dword>
```

`<dword>` Specifies the ID of the command set.

Return Value

None.

Example

The following **VP Designer** to HFS example assigns an ID to the command set:

```
atom$uni_start_stream
   atom$hfs_cmd_form_gid <0-0-0>
   atom$hfs_cmd_form_name <tim_black2>
   atom$hfs_cmd_template_name <”default”>
➪  atom$hfs_cmd_tag <9688552>
atom$uni_end_stream
```


atom$hfs_cmd_template_name
2 ($02)

Description

atom$hfs_cmd_template_name is sent by **VP Designer** to assign a template name to a form, or it is sent by HFS to define the template name of the form as the target context of other command streams.

Syntax

```
atom$hfs_cmd_template_name <string>
```

`<string>` Specifies the template name of the form.

Return Value

None.

Example

The following **VP Designer** to HFS example assigns a template name (default) to the form:

```
atom$uni_start_stream
   atom$hfs_cmd_form_gid <0-0-0>
   atom$hfs_cmd_form_name <tim_black2>
➪  atom$hfs_cmd_template_name <”default”>
   atom$hfs_cmd_tag <9688552>
atom$uni_end_stream
```