Chapter 4.
Functions

Topic: Ignore

The function interface is a set of Application Procedural Interface (API) and Direct Interface (DI) functions that an application can invoke to interact with ACIS. API functions, which combine modeler functionality with application support features such as argument error checking and roll back, are the main interface between applications and ACIS. The DI functions provide access to modeler functionality, but do not provide the additional application support features, and, unlike APIs, are not guaranteed to remain consistent from release to release. Refer to the *3D ACIS Online Help User's Guide* for a description of the fields in the reference template.

add_rbase_app_cb

Function: Rendering Control, Callbacks

Action: Creates a register of the callback.

Prototype:
```
void add_rbase_app_cb (
    rbase_app_callback* cb  // callback to register
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "rnd_husk/intrface/rbapp_cb.hxx"
```

Description: Refer to action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rbapp_cb.hxx

Effect: System routine

api_initialize_rendering

Function: Modeler Control, Rendering Control

Action: Initializes the rendering library.

Prototype: `outcome api_initialize_rendering ();`

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_convert_image_end

Function: Image Output

Action: Terminates conversion of an image.

Prototype: `outcome api_rh_convert_image_end ();`

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: Call this function to end the conversion of an image. It must match a corresponding call to LwConvertImageStart. Call this function after the last scanline of the image has been converted using LwConvertRGBFloatScanline or LwConvertRGBScanline.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_convert_image_start

Function: Image Output

Action: Starts conversion of an image.

Prototype:
```
outcome api_rh_convert_image_start (
    LwInt32 width,          // image width
    LwInt32 height          // image height
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_types.h"
```

Description: This function initializes the image format process for an image of the given size. The image to be converted has dimensions of width pixels horizontally and height pixels vertically.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_convert_rgb_float_scanline

Function: Image Output

Action: Converts a scanline of LwFloat values.

Prototype:
```
outcome api_rh_convert_rgb_float_scanline (
    LwFloat* input,         // array of pixel colors
    LwInt32* output         // returns array of color
                            // map indexes
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_types.h"
```

Description: This function converts an input array of pixel colors into a corresponding array of color map indexes, using the image conversion method previously set by LwSetConversionMethod. Call this function from the application–supplied function, rh_image_scanline.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_convert_rgb_scanline

Function: Image Output

Action: Converts a scanline of LwNat8 values.

Prototype:
```
outcome api_rh_convert_rgb_scanline (
    LwNat8* input,          // array of pixel colors
    LwInt32* output         // returns array of color
                            // map indexes
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_types.h"
```

Description: This function converts an input array of pixel colors into a corresponding array of color map indexes, using the image conversion method previously set with LwSetConversionMethod. Call this function from the application–supplied function, rh_image_scanline.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_copy_background

Function: Backgrounds and Foregrounds

Action: Creates a copy of a background.

Prototype:
```
outcome api_rh_copy_background (
    RH_BACKGROUND* background,      // old background
    RH_BACKGROUND*& new_background  // returns new
                                    // background
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API creates a new RH_BACKGROUND with the same properties as the old background.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_copy_foreground

Function: Backgrounds and Foregrounds

Action: Creates a copy of a foreground.

Prototype:
```
outcome api_rh_copy_foreground (
    RH_FOREGROUND* foreground,      // old foreground
    RH_FOREGROUND*& new_foreground  // returns new
                                    // foreground
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API creates a new RH_FOREGROUND with the same properties as the old foreground.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_copy_light

Function: Lights and Shadows

Action: Creates a copy of a light.

Prototype:
```
outcome api_rh_copy_light (
    RH_LIGHT* light,         // old light
    RH_LIGHT*& new_light     // new light returned
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API creates a new RH_LIGHT with the same properties as the old light.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_copy_material

Function: Materials

Action: Creates a copy of a material.

Prototype:
```
outcome api_rh_copy_material (
    RH_MATERIAL* material,      // old material
    RH_MATERIAL*& new_material  // new material
                                // returned
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API creates a new RH_MATERIAL with the same properties as the old material.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_copy_texture_space

Function: Texture Spaces

Action: Copies a texture space.

Prototype:
```
outcome api_rh_copy_texture_space (
    RH_TEXTURE_SPACE*        // original texture
        texture_space,       // space
    RH_TEXTURE_SPACE*&       // new texture space
        new_texture_space    // returned
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API creates a new RH_TEXTURE_SPACE with the same properties as the old texture_space.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_create_background

Function: Backgrounds and Foregrounds

Action: Creates a background.

Prototype:
```
outcome api_rh_create_background (
    const char* name,           // name of background
    RH_BACKGROUND*& background  // new background
                                // returned
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/include/rh_enty.hxx”
```

Description: This API creates a new RH_BACKGROUND. Once the background has been created, set the its properties using api_rh_set_background_arg. Set the current background using api_rh_set_background. Obtain a list of available background types by calling api_rh_get_background_types.

Errors: The name of the background is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_create_cube_environment

Function: Environment Maps

Action: Creates an environment map from six raster images.

Prototype:
```
outcome api_rh_create_cube_environment (
    unsigned char*,             // array of six
                                // raster image
                                // pointers
    int channels,               // number of channels
                                // in image data 1
                                // or 3
    int width[],                // width of images
    int height[],               // height of images
    Image_Access_Function afunc,// image access
                                // function
    RH_ENVIRONMENT_MAP*& env_map// new environment
                                // map returned
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API supports external reflections by creating a cubical environment map from six raster images supplied by the application. If the image access function is NULL, the default treats the image data as a raster image stored in row major order; otherwise, it is up to the application–supplied image access function to interpret the data.

Errors: None

Limitations: Replay is not supported.

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_create_foreground

Function: Backgrounds and Foregrounds

Action: Creates a foreground.

Prototype:
```
outcome api_rh_create_foreground (
    const char* name,           // name of foreground
    RH_FOREGROUND*& foreground  // new foreground
                                // returned
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API creates a new RH_FOREGROUND. Once the it is created, set the foreground properties using api_rh_set_foreground_arg. Set the current foreground using api_rh_set_foreground. Obtain a list of available foreground types by calling api_rh_get_foreground_types.

Errors: The name of the foreground is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_create_light

Function: Lights and Shadows, Viewing

Action: Creates a light.

Prototype:
```
outcome api_rh_create_light (
    const char* name,       // type of light
    RH_LIGHT*& light        // new light returned
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API creates a new RH_LIGHT. Once it is created, set the light properties using api_rh_set_light_arg. Obtain a list of supported light types by calling api_rh_get_light_types.

Errors: The name of the light is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_create_light_shadow

Function: Lights and Shadows

Action: Creates a shadow mask for a light from a list of entities.

Prototype:
```
outcome api_rh_create_light_shadow (
    RH_LIGHT* light,            // light
    ENTITY_LIST const& entities // light entity list
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API creates a shadow mask for a light from a list of entities. A shadow mask is a depth buffered image computed from the view point of a light source. Shadow masks help render shadows. The resolution of the shadow image depends upon the shadow resolution parameter associated with the light source shader. Set the resolution using api_rh_set_light_arg.

Errors: None

Limitations: Recompute a shadow map if certain parameters of the light, such as location, are changed or if the geometry of the objects in a scene has altered.

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_create_material

Function: Materials

Action: Creates a material.

Prototype:
```
outcome api_rh_create_material (
    RH_MATERIAL*& material  // new material returned
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API creates a new RH_MATERIAL. Each material contains four shaders, color, displacement, reflectance, and transparency, and it is initialized to the default shaders specified using the api_rh_def_* APIs. Once a material is created, attach the material to entities using api_rh_set_material. Set a material's properties by manipulating its four component shaders.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_create_texture_space

Function: Texture Spaces

Action: Creates a texture space.

Prototype:
```
outcome api_rh_create_texture_space (
    const char* name,       // name of texture space
    RH_TEXTURE_SPACE*&      // new textures space
        texture_space       // returned
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API creates an RH_TEXTURE_SPACE. The texture space name must correspond to that of an existing texture space shader prototype with that name. Set texture space properties using api_rh_set_texture_space_arg. Texture spaces modify the image based on the coordinates of each pixel. Texture spaces can make an object appear to be carved out of a solid material or wrapped in a sheet of material.

Errors: The name of the texture space is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_def_color_comp

Function: Color Patterns, Colors, Materials

Action: Sets a material's default color component.

Prototype:
```
outcome api_rh_def_color_comp (
    const char* name        // name of color shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API sets the default color shader to one of the known color shaders, obtained using api_rh_get_color_comp_list.

An RH_MATERIAL is defined by four component shaders: a color shader, a displacement shader, a reflectance shader, and a transparency shader. When an instance of an RH_MATERIAL is created, the type of shader components that it is initially created with depends on the default values set using this and other default shader component functions.

Errors: The name of the color shader is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_def_displace_comp

Function: Displacement, Materials

Action: Sets a material's default displacement component.

Prototype:
```
outcome api_rh_def_displace_comp (
    const char* name        // name of displacement
                            // shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API sets the default displacement shader to one of the known displacement shader names, obtained using api_rh_get_displace_comp_list.

An RH_MATERIAL is defined by four component shaders: a color shader, a displacement shader, a reflectance shader, and a transparency shader. When an instance of an RH_MATERIAL is created, the type of shader components that it is initially created with depends on the default values set using this and other default shader component functions.

Errors: The name of displacement shader is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_def_reflect_comp

Function: Displacement, Materials

Action: Sets a material's default displacement component.

Prototype:
```
outcome api_rh_def_reflect_comp (
    const char* name         // name of displacement
                             // shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API sets the default reflectance shader to one of the known reflectance shader names, obtained using api_rh_get_reflect_comp_list.

An RH_MATERIAL is defined by four component shaders: a color shader, a displacement shader, a reflectance shader, and a transparency shader. When an instance of an RH_MATERIAL is created, the type of shader components that it is initially created with depends on the default values set using this and other default shader component functions.

Errors: The name of reflectance shader is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_def_transp_comp

Function: Reflectance, Materials

Action: Sets a material's default transparency component.

Prototype:
```
outcome api_rh_def_transp_comp (
    const char* name        // name of transparency
                            // shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API sets the default transparency shader to one of the known transparency shaders, obtained using api_rh_get_transp_comp_list.

An RH_MATERIAL is defined by four component shaders: a color shader, a displacement shader, a reflectance shader, and a transparency shader. When an instance of an RH_MATERIAL is created, the type of shader components that it is initially created with depends on the default values set using this and other default shader component functions.

Errors: Name of transparency shader is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_delete_background

Function: Backgrounds and Foregrounds

Action: Deletes a background.

Prototype:
```
outcome api_rh_delete_background (
    RH_BACKGROUND* background   // background
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API explicitly deletes a background. When a background is deleted, do not use its handle in subsequent calls of api_rh_set_background. If the background currently in use by the rendering library has been deleted, reset it by calling api_rh_set_background.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_delete_environment_map

Function: Environment Maps

Action: Deletes an environment map.

Prototype:
```
outcome api_rh_delete_environment_map (
    RH_ENVIRONMENT_MAP*      // environment map
        environment_map      // to be deleted
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API explicitly deletes the environment map. When an environment map is deleted, do not use its handle in subsequent calls of api_rh_set_environment_map. If the environment map currently in use by the rendering library is deleted, reset the current environment by calling api_rh_set_environment_map.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_delete_foreground

Function: Backgrounds and Foregrounds

Action: Deletes a foreground.

Prototype:
```
outcome api_rh_delete_foreground (
    RH_FOREGROUND* foreground   // foreground
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API explicitly deletes the foreground. When a foreground is deleted, do not use its handle in subsequent calls of api_rh_set_foreground. If the foreground currently in use by the rendering library has been deleted, reset it by calling api_rh_set_foreground.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_delete_light

Function: Lights and Shadows, Viewing

Action: Deletes a light.

Prototype:
```
outcome api_rh_delete_light (
    RH_LIGHT* light         // light
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API explicitly deletes the light. When a light is deleted, do not use its pointer in subsequent calls of api_rh_set_light_list. If a light is deleted from the rendering library's current light list, reset the light list by calling api_rh_set_light_list.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_delete_light_shadow

Function: Lights and Shadows

Action: Deletes a light's shadow.

Prototype:
```
outcome api_rh_delete_light_shadow (
    RH_LIGHT* light          // light
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API deletes the shadow map associated with the light instance whose handle is given by light. Obtain the light handle by previously calling api_rh_create_light or api_rh_copy_light. To create a shadow map for a light, call api_rh_create_light_shadow.

If this API is called with an invalid light or if the light is not currently maintaining a shadow map, the call is ignored.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_delete_material

Function: Materials

Action: Deletes a material.

Prototype:
```
outcome api_rh_delete_material (
    RH_MATERIAL* material    // material
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API deletes a material. Delete all materials that have been created before terminating the renderer.

Errors: Errors may occur if a material attached to one or more entities is deleted.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_delete_texture_space

Function: Texture Spaces

Action: Deletes a texture space.

Prototype:
```
outcome api_rh_delete_texture_space (
    RH_TEXTURE_SPACE* texture_space // texture space
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API deletes a texture space. Texture spaces can be attached to other entities as properties. Delete all texture spaces that have been created before terminating the renderer. Do not delete a texture space that may be referenced by some other entity.

Errors: Errors may occur if a texture space attached to one or more entities is deleted.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_display_image

Function: Viewing

Action: Sets the display image parameters.

Prototype:
```
outcome api_rh_display_image (
    LwInt32 width,          // image width
    LwInt32 height,         // image height
    LwDisplayMethod method, // render mode
    LwInt32 n_colours,      // number of colors
    void(* input_scanline)  // number of scan
        (LwInt32 y,         // lines from the
        LwNat8** scan),     // input display
    void(* output_scanline) // number of scan
        (LwInt32 y,         // lines used for
        LwInt32* scan),     // the output
    void(* set_colour_map)  // set the color map
        (LwNat8* colours)   // for the output
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/include/rh_types.h”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_get_background

Function: Backgrounds and Foregrounds

Action: Gets the current background.

Prototype:
```
outcome api_rh_get_background (
    RH_BACKGROUND*& background  // returns background
        // to be queried
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API gets the current background shader used by the Rendering Base Component to color pixels that are not covered by an object. If the current background is NULL, the background is black.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_background_args

Function: Backgrounds and Foregrounds

Action: Gets the arguments of a background.

Prototype:
```
outcome api_rh_get_background_args (
    RH_BACKGROUND* background,  // background to be
                                // queried
    const char*& name,          // returns name of
                                // background type
    int& no_of_args,            // returns number of
                                // arguments
    const char** arg_names&,    // returns array of
                                // argument names
    Render_Arg*& arg_values     // returns array of
                                // arguments
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: Each background has a list of named arguments. The number and names of those arguments depend on the type of the background. Access the value of each argument via the Render_Arg class.

This API returns the name of the background type with two lists containing the names and values of those arguments. All returned pointer values point to internal data structures.

For efficiency, these values are not copied when the API is called. The application should not free the memory pointed to by those pointers. In the case of Render_Arg, the internal array is re–used by other inquiry functions. It is up to the application to copy those values, if required, before any subsequent inquiry functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_background_types

Function: Backgrounds and Foregrounds

Action: Gets a list of valid background type names.

Prototype:
```
outcome api_rh_get_background_types (
    int& n_backgrounds,      // length of list of
                             // names
    const char**& names      // list of names
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API points to a list of valid RH_BACKGROUND type names. The list should not be freed by the application.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_clipping

Function: Faceting, Viewing

Action: Gets the depth clipping parameters that affect view–dependent faceting.

Prototype:
```
outcome api_rh_get_clipping (
    double& near_clipping_plane,// returns near
                                // clipping plane
    double& far_clipping_plane  // returns far
                                // clipping plane
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: This API gets the depth clipping parameters that affect view–dependent faceting.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_color_comp

Function: Color Patterns, Colors, Materials

Action: Gets the arguments of a material’s color component.

Prototype:
```
outcome api_rh_get_color_comp (
    RH_MATERIAL* material,  // material to be queried
    const char*& name,      // name of color
                            // component type
    int& no_of_args,        // number of arguments
    const char** arg_names&,// array of argument
                            // names
    Render_Arg*& arg_values // returns array of
                            // arguments
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/include/rh_args.hxx”
#include ”rnd_husk/include/rh_enty.hxx”
```

Description: Each material color component has a list of named arguments. The number and names of those arguments depend on the type of color component. Access the value of each argument via the Render_Arg class.

This API returns the name of the color component type with two lists containing the names and values of those arguments. All returned pointer values point to internal data structures.

For efficiency, these values are not copied when the API is called. The application should not free the memory pointed to by those pointers. In the case of Render_Arg, the internal array is re–used by other inquiry functions. It is up to the application to copy those values, if required, before any subsequent inquiry functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_color_comp_list

Function: Color Patterns, Colors, Materials

Action: Gets a list of valid color component names.

Prototype:
```
outcome api_rh_get_color_comp_list (
    int& n_colors,          // returns length of list
                            // of names
    const char**& names     // returns list of names
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: This API points to a list of valid color component names. The list should not be freed by the application.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_control_variable

Function: Rendering Control, Viewing

Action: Gets a render control variable.

Prototype:
```
outcome api_rh_get_control_variable (
    Render_Control_Var var, // control variable
    Render_Arg& variable    // returns control
                            // variable's value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
```

Description: This API gets the value of a render control variable. For a list of control variable names, refer to the chapter on Control Variables.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_default_rgb

Function: Color Patterns, Colors

Action: Gets the default RGB color for newly-created entities.

Prototype:
```
outcome api_rh_get_default_rgb (
    rgb_color& color        // returned color
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/rgbcolor.hxx"
```

Description: This API gets the default RGB color for newly-created entities.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read-only

api_rh_get_displace_comp

Function: Displacement, Materials

Action: Gets the arguments of a material's displacement component.

Prototype:
```
outcome api_rh_get_displace_comp (
    RH_MATERIAL* material,   // material to be queried
    const char*& name,       // returns name of
                             // displacement component
                             // type
    int& no_of_args,         // returns number of
                             // arguments
    const char** arg_names&, // returns array of
                             // argument names
    Render_Arg*& arg_values  // returns array of
                             // arguments
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: Each material displacement component has a list of named arguments. The number and names of those arguments depend on the type of displacement component. Access the value of each argument via the Render_Arg class.

This API returns the name of the displacement component type with two lists containing the names and values of those arguments. All returned pointer values point to internal data structures.

For efficiency, these values are not copied when the API is called. The application should not free the memory pointed to by those pointers. In the case of Render_Arg, the internal array is re–used by other inquiry functions. It is up to the application to copy those values, if required, before any subsequent inquiry functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_displace_comp_list

Function: Displacement, Materials

Action: Gets a list of valid displacement component names.

Prototype:
```
outcome api_rh_get_displace_comp_list (
    int& n_displaces,       // returns length of list
                            // of names
    const char**& names     // returns list of names
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API points to a list of valid RH_MATERIAL displacement component names. The list should not be freed by the application.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_displace_status

Function: Displacement, Materials

Action: Gets the status of a material's displacement component.

Prototype:
```
outcome api_rh_get_displace_status (
    RH_MATERIAL* material,  // material to be queried
    logical& status         // returns displacement
                            // status
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
#include ″baseutil/logical.h″
```

Description: This API obtains the status of a material's displacement shader. If the status of a displacement shader is TRUE, the entity renders with displacement for those rendering modes that support displacement.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_entity_rgb

Function: Color Patterns, Colors

Action: Gets the RGB color of an entity.

Prototype:
```
outcome api_rh_get_entity_rgb (
    ENTITY* ent,            // entity to change
    rgb_color& color,       // color returned
    logical inherit,        // inherited color
    logical& found          // Flag indicating that
                            // the color was
                            // explicitly set
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/rgbcolor.hxx″
```

Description: This API gets the RGB color of a specified entity (ent). If the entity has an ATTRIB_RGB or ATTRIB_COL, the API returns the recorded color; otherwise, it returns the current default color. If color is –1, it means that a color is not specified for the entity.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_environment_map

Function: Environment Maps

Action: Gets the current environment map.

Prototype:
```
outcome api_rh_get_environment_map (
    RH_ENVIRONMENT_MAP*& emap   // returned
                                // environment map
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API gets the current environment map; it returns NULL if the map is not set.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_foreground

Function: Backgrounds and Foregrounds

Action: Gets the current foreground.

Prototype:
```
outcome api_rh_get_foreground (
    RH_FOREGROUND*& foreground  // returns foreground
                                // to be queried
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API gets the current foreground shader used by the Rendering Base Component as a post processing step in the shader pipeline. The current foreground may be NULL, in which case, no post processing is performed.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_foreground_args

Function: Backgrounds and Foregrounds

Action: Gets the arguments of a foreground.

Prototype:
```
outcome api_rh_get_foreground_args (
    RH_FOREGROUND* foreground,  // foreground to be
                                // queried
    const char*& name,          // returns name of
                                // foreground type
    int& no_of_args,            // returns number of
                                // arguments
    const char** arg_names&,    // returns array of
                                // argument names
    Render_Arg*& arg_values     // returns array of
                                // arguments
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_args.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: Each foreground has a list of named arguments. The number and names of those arguments depend on the type of foreground. Access the value of each argument via the Render_Arg class.

This API returns the name of the foreground type with two lists containing the names and values of those arguments. All returned pointer values point to internal data structures.

For efficiency, these values are not copied when the function is called. The application should not free the memory pointed to by those pointers. In the case of Render_Arg, the internal array is re–used by other inquiry functions. It is up to the application to copy those values, if required, before any subsequent inquiry functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_foreground_types

Function: Backgrounds and Foregrounds

Action: Gets a list of valid foreground type names.

Prototype:
```
outcome api_rh_get_foreground_types (
    int& n_foregrounds,      // returns number of
                             // foregrounds
    const char**& names      // returns list of names
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: This API points to a list of valid RH_FOREGROUND type names. The list should not be freed by the application.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_highlight_rgb

Function: Color Patterns, Colors

Action: Gets the RGB color used to highlight entities.

Prototype:
```
outcome api_rh_get_highlight_rgb (
    rgb_color& color        // highlight_rgb
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/rgbcolor.hxx″
```

Description: This API returns the RGB color used to highlight entities.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read-only

api_rh_get_light_args

Function: Lights and Shadows

Action: Gets the arguments of a light.

Prototype:
```
outcome api_rh_get_light_args (
    RH_LIGHT* light,        // light to be queried
    const char*& name,      // returns name of light
                            // type
    int& no_of_args,        // returns number of
                            // arguments
    const char** arg_names&,// returns array of
                            // argument names
    Render_Arg*& arg_values // returns array of
                            // arguments
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_args.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: Each light has a list of named arguments. The number and names of those arguments depend on the type of light. Access the value of each argument via the Render_Arg class.

This API returns the name of the light type with two lists containing the names and values of those arguments. All returned pointer values point to internal data structures.

For efficiency, these values are not copied when the API is called. The application should not free the memory pointed to by those pointers. In the case of Render_Arg, the internal array is re–used by other inquiry functions. It is up to the application to copy those values, if required, before any subsequent inquiry functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_light_list

Function: Lights and Shadows

Action: Gets the contents of the active light list.

Prototype:
```
outcome api_rh_get_light_list (
    ENTITY_LIST& lights     // returns list of light
                            // entities, expected to
                            // be empty when
                            // specified to function
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/lists/lists.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: The Rendering Base Component maintains an internal list of active lights that are used as light sources when an image is rendered. This API gets a copy of the elements of that list. At least one RH_LIGHT must reside in the active light list; otherwise, shaded entities are not visible. The API adds entries to the ENTITY_LIST provided by the application developer. It is up to the application to clean out the list.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_light_state

Function: Lights and Shadows

Action: Gets the on/off state of light.

Prototype:
```
outcome api_rh_get_light_state (
    RH_LIGHT* light,         // light to be queried
    logical* on_off          // returns on/off
                             // state of light
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/include/rh_enty.hxx”
#include ”baseutil/logical.h”
```

Description: This API gets the active state of the RH_LIGHT entity.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_light_types

Function: Lights and Shadows

Action: Gets a list of valid light type names.

Prototype:
```
outcome api_rh_get_light_types (
    int& n_types,            // returns number of
                             // types of lights
    const char**& names      // returns list of names
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API points to a list of valid RH_LIGHT type names. The list should not be freed by the application.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_material

Function: Materials

Action: Gets the material attached to an entity.

Prototype:
```
outcome api_rh_get_material (
    ENTITY* entity,          // entity to be queried
    RH_MATERIAL*& material   // returns material
                             // attached to entity
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/data/entity.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API gets the material attached to an entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_material_color

Function: Materials, Color Patterns, Colors

Action: Gets the material color associated with a geometric entity.

Prototype:
```
outcome api_rh_get_material_color (
    ENTITY* entity,          // entity
    double& red,             // returns red color
                             // value
    double& green,           // returns green color
                             // value
    double& blue             // returns blue color
                             // value
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/data/entity.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: This API returns the material color attached to an entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_material_reflection

Function: Materials, Reflectance

Action: Gets the material reflection properties associated with a geometric entity.

Prototype:
```
outcome api_rh_get_material_reflection (
    ENTITY* entity,          // entity
    double& ambient,         // returns ambient
                             // reflection
    double& diffuse,         // returns diffuse
                             // reflection
    double& specular,        // returns specular
                             // reflection
    double& exponent         // returns exponent value
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This API gets the material reflection properties (if any) associated with a geometric entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_material_texture

Function: Materials, Textures

Action: Gets the material texture associated with a geometric entity.

Prototype:
```
outcome api_rh_get_material_texture (
    ENTITY* entity,          // entity
    const char*& tex_name    // returns type of
                             // texture
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This API gets the material texture associated with a geometric entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_material_transp

Function: Materials, Transparency

Action: Gets the material transparency associated with a geometric entity.

Prototype:
```
outcome api_rh_get_material_transp (
    ENTITY* entity_list,     // list of entities
    double& transp           // returns transparency
                             // value
                             // (0 is transparent;
                             // 1 is opaque)
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This API gets the material transparency associated with a geometric entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_output_mode

Function: Rendering Control, Image Output, Viewing

Action: Gets the current output mode.

Prototype:
```
outcome api_rh_get_output_mode (
    Output_Mode& output_mode // returns output mode
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_args.hxx″
```

Description: This API gets the current output mode. Refer to the chapter on Output Modes for a list of applicable modes.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_reflect_comp

Function: Reflectance, Materials

Action: Gets the arguments of a material's reflectance component.

Prototype:
```
outcome api_rh_get_reflect_comp (
    RH_MATERIAL* material,  // material to be queried
    const char*& name,      // returns name of
                            // reflectance component
                            // type
    int& no_of_args,        // returns number of
                            // arguments
    const char** arg_names&,// returns array of
                            // argument names
    Render_Arg*& arg_values // returns array of
                            // arguments
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: Each material reflectance component has a list of named arguments. The number and names of those arguments depend on the type of reflectance component. Access the value of each argument via the Render_Arg class.

This API returns the name of the reflectance component type with two lists containing the names and values of those arguments. All returned pointer values point to internal data structures.

For efficiency, these values are not copied when the API is called. The application should not free the memory pointed to by those pointers. In the case of Render_Arg, the internal array is re–used by other inquiry functions. It is up to the application to copy those values, if required, before any subsequent inquiry functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_reflect_comp_list

Function: Reflectance, Materials

Action: Gets a list of valid reflectance component names.

Prototype:
```
outcome api_rh_get_reflect_comp_list (
    int& n_reflects,         // returns number of
                             // names
    const char**& names      // returns list of names
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: This API points to a list of valid RH_MATERIAL reflectance component names. The list should not be freed by the application.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_reflect_status

Function: Rendering Control, Viewing

Action: Gets the status of a material’s reflection component.

Prototype:
```
outcome api_rh_get_reflect_status (
    RH_MATERIAL* material,   // shader to query
    logical& status          // returns on/off
                             // shader status
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API obtains the status of a material's reflection shader. If the status of a reflection shader is on (TRUE), the RH_MATERIAL renders with reflections for those rendering modes that support reflection.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read-only

api_rh_get_render_mode

Function: Rendering Control, Viewing

Action: Gets the current render mode.

Prototype:
```
outcome api_rh_get_render_mode (
    Render_Mode& render_mode// returns render mode
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_args.hxx″
```

Description: This API gets the current render mode. Refer to the chapter on Render Modes for a list of applicable modes.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_resolution

Function: Faceting, Viewing

Action: Gets the default screen faceting resolution controls.

Prototype:
```
outcome api_rh_get_resolution (
    int& width,                 // returns image
                                // space width
    int& height,                // returns image
                                // space height
    double& pixel_aspect_ratio, // returns aspect
                                // ratio in pixels
    double& image_scale         // returns scale of
                                // the image
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: This API gets the default screen faceting resolution controls.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_sidedness

Function: Rendering Control

Action: Gets the sidedness of an entity.

Prototype:
```
outcome api_rh_get_sidedness (
    ENTITY* entity,         // entity to be queried
    int& sides              // returns sidedness
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/data/entity.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: Geometric entities must be of the type BODY, LUMP, SHELL, or FACE. If sides is 2, the entity is treated as double–sided for rendering. If sides is 1, the entity is treated as single–sided for rendering. If sides is 0, the entity has no sidedness set, and it inherits sidedness from its parent.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_sub_image

Function: Faceting, Viewing

Action: Gets the pixel coordinates of the region of interest on a screen.

Prototype:
```
outcome api_rh_get_sub_image (
    int& left,              // returns left side
    int& right,             // returns right side
    int& top,               // returns top
    int& bottom             // returns bottom
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API gets the pixel coordinates of the region of interest on a screen.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_texture_space

Function: Texture Spaces

Action: Gets the texture space attached to an entity.

Prototype:
```
outcome api_rh_get_texture_space (
    ENTITY* entity,          // entity to be queried
    RH_TEXTURE_SPACE*&       // returns the new
        texture_space        // texture space
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API gets the texture space attached to an entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_texture_space_args

Function: Texture Spaces

Action: Gets the arguments of a texture space.

Prototype:
```
outcome api_rh_get_texture_space_args (
    RH_TEXTURE_SPACE*        // texture space to
        texture_space,       // be queried
    const char*& name,       // returns name of
                             // texture space type
    int& no_of_args,         // returns number of
                             // arguments
    const char** arg_names&, // returns array of
                             // argument names
    Render_Arg*& arg_values  // returns array of
                             // arguments
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_args.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: Each texture space has a list of named arguments. The number and names of those arguments depend on the type of texture_space. Access the value of each argument via the Render_Arg class.

This API returns the name of the texture_space type with two lists containing the names and values of those arguments. All returned pointer values point to internal data structures.

For efficiency, these values are not copied when the function is called. The application should not free the memory pointed to by those pointers. In the case of Render_Arg, the internal array is re–used by other inquiry functions. It is up to the application to copy those values, if required, before any subsequent inquiry functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_texture_space_types

Function: Texture Spaces

Action: Gets a list of valid texture space type names.

Prototype:
```
outcome api_rh_get_texture_space_types (
    int& n_types,           // returns number of
                            // names
    const char**& names     // returns list of names
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API points to a list of valid RH_TEXTURE_SPACE type names. The list should not be freed by the application.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_transp_comp

Function: Transparency, Materials

Action: Gets the arguments of a material's transparency component.

Prototype:
```
outcome api_rh_get_transp_comp (
    RH_MATERIAL* material,   // material to be queried
    const char*& name,       // returns name of
                             // transparency component
                             // type
    int& no_of_args,         // returns number of
                             // arguments
    const char** arg_names&, // returns array of
                             // argument names
    Render_Arg*& arg_values  // returns array of
                             // arguments
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: Each material transparency component has a list of named arguments. The number and names of those arguments depend on the type of transparency component. Access the value of each argument via the Render_Arg class.

This API returns the name of the transparency component type with two lists containing the names and values of those arguments. All returned pointer values point to internal data structures.

For efficiency, these values are not copied when the API is called. The application should not free the memory pointed to by those pointers. In the case of Render_Arg, the internal array is re–used by other inquiry functions. It is up to the application to copy those values, if required, before any subsequent inquiry functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_transp_comp_list

Function: Transparency, Materials

Action: Gets a list of valid transparency component names.

Prototype:
```
outcome api_rh_get_transp_comp_list (
    int& n_transparencies,  // returns number of
                            // transparencies
    const char**& names     // returns list of names
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: This API points to a list of valid RH_MATERIAL transparency component names. The list should not be freed by the application.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_transp_status

Function: Transparency, Materials

Action: Gets the status of a material’s transparency component.

Prototype:
```
outcome api_rh_get_transp_status (
    RH_MATERIAL* material,  // shader to be queried
    logical& status         // returns on/off status
                            // of shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
#include "baseutil/logical.h"
```

Description: This API obtains the status of a material's transparency shader. If the status of a transparency shader is on (TRUE), the RH_MATERIAL renders with transparency for those rendering modes that support transparency.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_get_view

Function: Faceting, Viewing

Action: Gets the view parameters that affect view–dependent faceting.

Prototype:
```
outcome api_rh_get_view (
    SPAposition& from_point,// returns from point
    SPAposition& to_point,  // returns to point
    SPAvector& view_up_vector,// returns normal to
                              // view
    Projection_Type&          // returns type of
        projection,           // projection
    double& field_of_view     // returns field of view
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/vector.hxx"
```

Description: This API gets the view parameters that affect view–dependent faceting.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read–only

api_rh_initialise

Function: Rendering Control, Modeler Control, Viewing

Action: Initializes the renderer.

Prototype:
```
outcome api_rh_initialise ();
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This API initializes the internal data structures required by the renderer. Call this API before any other Rendering Base Component functions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_initialise_image_utilities

Function: Image Output, Modeler Control

Action: Initializes the Image Utilities Library.

Prototype:
```
outcome api_rh_initialise_image_utilities ();
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This function initializes the use of the Image Format Utility Library. Call this function before any others in the Image Format Utility Library. Call LwTerminateImageUtilities to terminate the use of the library. Call this function when the rendering session is initiated with api_rh_initialise.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_initialise_supl_shaders

Function: Rendering Control, Modeler Control, Viewing

Action: Initializes the supplemental shaders.

Prototype:
```
outcome api_rh_initialise_supl_shaders ();
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: When the Rendering Base Component is initialized, the shaders supplied with the Core Shader Library are automatically initialized. To initialize the larger set of shaders located in the Supplementary Shader Library, call api_rh_initialise_supl_shaders.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_read_lightworks_image

Function: Image Output

Action: Gets the LightWorks image for display.

Prototype:
```
outcome api_rh_read_lightworks_image (
    FILE* fp,               // filename
    LwInt32 channels,       // color channel
    LwInt32 width,          // image width
    LwInt32 height,         // image height
    LwNat8* data            // image data
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/include/rh_types.h”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_read_lightworks_image_size

Function: Image Output

Action: Gets the LightWorks image size.

Prototype:
```
outcome api_rh_read_lightworks_image_size (
    FILE* fp,                // filename
    LwInt32* width,          // image width
    LwInt32* height          // image height
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/include/rh_types.h”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_render_cube_environment

Function: Environment Maps

Action: Creates an environment map from a list of entities.

Prototype:
```
outcome api_rh_render_cube_environment (
    ENTITY_LIST const& entities,// list of entities
                                // from which the
                                // environment map is
                                // to be computed
    int resolution,             // resolution
    SPAposition const& view_point,  // view point
    RH_ENVIRONMENT_MAP*& env_map// returns new
                                // environment map
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
#include "baseutil/vector/position.hxx"
```

Description: Environment maps simulate reflections on objects caused by other objects in the scene and by an external environment. This API supports inter–object reflections by computing an environment map from a list of entities. The environment map is represented internally by six square images of a given resolution. Each image represents a perspective view as seen from the given view_point in each of the six orthogonal directions.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_render_entities

Function: Rendering Control, Viewing

Action: Renders a list of entities.

Prototype:
```
outcome api_rh_render_entities (
    ENTITY_LIST const& entities,// entities to be
                                // rendered
    logical clear_screen        // clear image
        = TRUE                  // before rendering
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "baseutil/logical.h"
```

Description: This API renders one or more geometric entities of the type BODY, LUMP, SHELL, or FACE. Facet the entities before rendering.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_set_background

Function: Backgrounds and Foregrounds

Action: Sets the current background.

Prototype:
```
outcome api_rh_set_background (
    RH_BACKGROUND* background   // background to set
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets the current background shader used by the Rendering Base Component to color pixels that are not covered by the surface of an object. If the current background shader is NULL, the background is black.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_background_arg

Function: Backgrounds and Foregrounds

Action: Sets an argument for a background.

Prototype:
```
outcome api_rh_set_background_arg (
    RH_BACKGROUND* background,  // background
    const char* arg_name,       // name of argument
    const Render_Arg& value     // argument value
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/include/rh_args.hxx”
#include ”rnd_husk/include/rh_enty.hxx”
```

Description: This API sets arguments controlling the background specified by name. name must correspond to one of the supported arguments for the type of background associated with RH_BACKGROUND. The underlying argument type depends on the argument set with the Render_Arg class.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_clipping

Function: Faceting, Viewing

Action: Sets the sets the near and far clipping planes for faceting.

Prototype:
```
outcome api_rh_set_clipping (
    double near_clipping_plane, // near clipping
                                // plane
    double far_clipping_plane   // far clipping plane
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
```

Description: This API sets the near and far clipping planes for faceting.

When view–dependent refinements are used, objects beyond the far planes or in front of the near planes are not faceted. Objects that cross the clipping planes facet correctly only in the region between the planes.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_color_arg

Function: Color Patterns, Colors, Materials

Action: Sets an argument of a material's color component.

Prototype:
```
outcome api_rh_set_color_arg (
    RH_MATERIAL* material,  // material
    const char* arg_name,   // name of argument
    const Render_Arg& value // argument value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets arguments controlling the color component specified by name. name must correspond to one of the supported arguments for the type of color component associated with the RH_MATERIAL. The underlying argument type depends on the argument set with the Render_Arg class.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_color_comp

Function: Color Patterns, Colors, Materials

Action: Sets the color component of a material.

Prototype:
```
outcome api_rh_set_color_comp (
    RH_MATERIAL* material,          // material
    const char* color_shader_name   // name of color
                                    // shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: An RH_MATERIAL comprises four component shaders: a color shader, a displacement shader, a reflectance shader, and a transparency shader. When an instance of an RH_MATERIAL is created, it is created with a default set of component shaders. This API changes the type of the color shader to name.

Errors: The name of color component is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_control_variable

Function: Rendering Control, Viewing

Action: Sets a render control variable.

Prototype:
```
outcome api_rh_set_control_variable (
    Render_Control_Var var, // control variable
    const Render_Arg& value // control value
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_args.hxx″
```

Description: This API sets a render control parameter. The value given depends on the control variable being affected. Refer to the chapter on Control Variables for a list of all control variable names.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_set_conversion_colour_map

Function: Image Output

Action: Sets the color map for conversion.

Prototype:
```
outcome api_rh_set_conversion_colour_map (
    LwInt32 r_max,          // red color map index
    LwInt32 g_max,          // green color map index
    LwInt32 b_max,          // blue color map index
    LwInt32 r_mult,         // red color multiplier
    LwInt32 g_mult,         // green color multiplier
    LwInt32 b_mult,         // blue color multiplier
    LwInt32 base_index      // base index
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_types.h″
```

Description: This function describes the color map of the output device, and it takes the form of an RGB cube. An RGB cube is a 3D array containing an ordered spectrum of colors. The subscripts of an RGB cube are values representing the respective amounts of each red, green, and blue color component of a particular color.

Images are displayed on a display device using a color–map containing the spectrum of colors described by the RGB cube.

Display devices' color–maps are normally 1D arrays, with the subscript being a color–map index. r_mult, g_mult, and b_mult are the multipliers for the respective RGB cube subscripts used to build a color–map index. The color–map index of a color within the RGB cube, with RGB cube subscript levels of r, g, and b are base_index + r x r_mult + g x g_mult + b x b_mult.

When LwConvertRGBScanline or LwConvertRGBFloatScanline is called, the pixels' colors are converted into color–map indexes within the RGB cube.

This function's parameters describe the layout of a 3D array representing the RGB cube. Set r_max, g_max, and b_max to the maximum number of different shades of each primary color in the RGB cube. Set r_mult, g_mult, and b_mult to the multipliers for the respective components within the RGB cube when converting RGB levels into color–map indexes. Set base_index to the start index of the RGB cube within the color–map. For a standard 8–bit display device, the values are typically: r_max = 5, g_max = 5, b_max = 5, r_mult = 36, g_mult = 6, b_mult = 1, describing a 216–color cube with six levels of each color component. Set base_index to the index of the first entry of the cube in the map.

For the grey–scale conversion method only r_max, r_mult, and base_index are significant.

For the monochrome conversion method, only base_index is significant.

Set the conversion color–map for an image before calling LwConvertImageStart. This function is ignored if it is called between calls to LwConvertImageStart and LwConvertImageEnd.

The default conversion color–map set by LwInitialiseImageUtilities is a 6 x 6 x 6 RGB cube with parameters of r_max = 5, g_max = 5, b_max = 5, r_mult = 36, g_mult = 6, b_mult = 1, and base_index = 0.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_set_conversion_method

Function: Image Output

Action: Sets the method for conversion.

Prototype:

```
outcome api_rh_set_conversion_method (
    LwConversionMethod method   // conversion method
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_types.h"
```

Description: This function sets the image conversion method used by LwConvertRGBScanline and LwConvertRGBFloatScanline for subsequent images. method must be one of the following conversion method types:

LW_CONVERSION_TC_DITHER adds random dither (noise) separately to each pixel color component in the image data. The amplitude of the dither is half a quantization level. Use the true–color dither image conversion on true–color devices.

LW_CONVERSION_FS_RGB distributes the color errors between the image color of a pixel and the nearest color to it in the RGB cube across the image to minimize any color artifacts. The process produces good color reproduction, and it is the recommended image conversion method for the higher–quality rendering modes on pseudo–color displays. This conversion method is based on the Floyd–Steinberg algorithm. This is the default conversion method.

LW_CONVERSION_O_DITHER_RGB screens the pixel colors by a color dithering matrix. This increases the color resolution of the device at the expense of reducing the spatial resolution. The ordered dither image conversion method is faster than the Floyd–Steinberg method, but it tends to introduce a noticeable pattern across the image. Use it as an alternative conversion method to LW_CONVERSION_FS_RGB with simpler rendering modes, such as flat and Gouraud, where speed is more important than image quality.

LW_CONVERSION_GREYSCALE converts the color of a pixel into a grey level using the standard NTSC color to luminance conversion. This luminance value is the basis for the index assigned to the pixel.

```
luminance = 0.3r + 0.59g + 0.11b
```

This method assumes that the grey–scale ramp is defined by the red channel of the color map defined by calling LwSetConversionColourMap. This conversion method applies to grey–scale display devices.

LW_CONVERSION_FS_MONOCHROME distributes errors in the colors between the image color of a pixel and the black or white pixel value area across the image to minimize any artifacts. The conversion is based on the Floyd–Steinberg algorithm. This image conversion method provides good image reproduction on 1–bit or bit–mapped display devices.

LW_CONVERSION_FS_DITHER_RGB augments a color Floyd–Steinberg error distribution with the addition of random noise to the input colors. For some images, the standard color Floyd–Steinberg method creates artifacts in the image, and the addition of noise helps to minimize them. Generally, the improvements to the output image are not enough to justify the extra time spent by this method.

LW_CONVERSION_DITHER_RGB adds a random noise to the image before quantization into the color map. The dither amplitude is equal to one quantization level. This improves average image quality at the expense of some localized noise.

LW_CONVERSION_NEAREST_RGB quantizes pixel colors are quantized into the color map by selecting the nearest color in the map. For pseudo–color displays, this usually causes color banding in the image.

Call this function before calling LwConvertImageStart. Do not call this function during the conversion of an image, which is initiated by LwConvertImageStart and terminated by LwConvertImageEnd. The default method is set to LW_CONVERSION_FS_RGB when LwInitialiseImageUtilities is called.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_set_default_rgb

Function: Color Patterns, Colors

Action: Modifies the default RGB color for newly-created entities.

Prototype:

```
outcome api_rh_set_default_rgb (
    rgb_color color         // color to set
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/rgbcolor.hxx"
```

Description: This API changes the default RGB color used when creating new entities. Use this color to display entities with no color attribute. It is also used when color is –1 in api_rh_set_entity_color.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Read-only

api_rh_set_displace_arg

Function: Displacement, Materials

Action: Sets an argument of a material's displacement component.

Prototype:
```
outcome api_rh_set_displace_arg (
    RH_MATERIAL* material,  // material
    const char* arg_name,   // name of argument
    const Render_Arg& value // argument value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets arguments controlling the displacement component specified by name. name must correspond to one of the supported arguments for the type of displacement component associated with the RH_MATERIAL. The underlying argument type depends on the argument set with the Render_Arg class.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_displace_comp

Function: Displacement, Materials

Action: Sets a material's displacement component.

Prototype:
```
outcome api_rh_set_displace_comp (
    RH_MATERIAL* material,   // material
    const char* name         // name of displacement
                             // shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: An RH_MATERIAL comprises four component shaders: a color shader, a displacement shader, a reflectance shader, and a transparency shader. When an instance of an RH_MATERIAL is created, it is created with a default set of component shaders. This API changes the type of the displacement shader to name.

Errors: The name of the displacement component is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_displace_status

Function: Displacement, Materials

Action: Sets the status of a material's displacement component.

Prototype:
```
outcome api_rh_set_displace_status (
    RH_MATERIAL* material,  // material to set
    logical status          // TRUE (displacement
                            // on) or FALSE
                            // (displacement off)
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
#include ″baseutil/logical.h″
```

Description: This API enables or disables the defined displacement component of a material. The API uses the displacement component if its status is TRUE; otherwise, it is ignored.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_entity_rgb

Function: Color Patterns, Colors

Action: Modifies the RGB color index of an ENTITY.

Prototype:
```
outcome api_rh_set_entity_rgb (
    ENTITY* ent,            // entity to modify
    rgb_color color         // new color index
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/rgbcolor.hxx″
```

Description: This API changes the color index of an entity (ent). Change the entity color by adding or changing an ATTRIB_RGB. Also, if the entity is in the display list, its color is changes. Any ATTRIB_COL is removed from the entity.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_environment_map

Function: Environment Maps

Action: Sets the current environment map.

Prototype:
```
outcome api_rh_set_environment_map (
    RH_ENVIRONMENT_MAP* emap// environment map to set
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: This API sets the renderer to use a particular cube environment map during rendering. If the current mode of rendering and current shaders use reflectance, the environment map generates a single level of reflections. Environment maps are generated by reading six raster images supplied by the application using api_rh_create_cube_environment or generated by rendering six images from the entity list using api_rh_render_cube_environment.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_foreground

Function: Backgrounds and Foregrounds

Action: Sets the current foreground.

Prototype:
```
outcome api_rh_set_foreground (
    RH_FOREGROUND* foreground   // foreground to set
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets the current foreground shader used by the Rendering Base Component to color pixels that are not covered by the surface of an object. If the current foreground shader is NULL, the foreground is black.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_foreground_arg

Function: Backgrounds and Foregrounds

Action: Sets an argument of a foreground.

Prototype:
```
outcome api_rh_set_foreground_arg (
    RH_FOREGROUND* foreground,  // foreground
    const char* arg_name,       // name of argument
    const Render_Arg& value     // argument value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets arguments controlling the foreground specified by name. name must correspond to one of the supported arguments for the type of foreground associated with RH_FOREGROUND. The underlying argument type depends on the argument set with the Render_Arg class.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_highlight_rgb

Function: Color Patterns, Colors

Action: Modifies the RGB color used to highlight entities.

Prototype:
```
outcome api_rh_set_highlight_rgb (
    rgb_color color          // new color index
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/rgbcolor.hxx”
```

Description: This API changes the color index used to display highlighted entities.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_light_arg

Function: Lights and Shadows

Action: Sets an argument for a light.

Prototype:
```
outcome api_rh_set_light_arg (
    RH_LIGHT* light,         // light
    const char* arg_name,    // name of argument
    const Render_Arg& value  // argument value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets sets arguments controlling the light specified by name. name must correspond to one of the supported arguments for the type of light associated with RH_LIGHT. The underlying argument type depends on the argument set with the Render_Arg class.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_light_list

Function: Lights and Shadows

Action: Sets the active light list.

Prototype:
```
outcome api_rh_set_light_list (
    ENTITY_LIST const& lights   // list of lights
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: The Rendering Base Component maintains an internal list of active lights that are used as light sources when an image is rendered. This API sets the elements of that list. At least one RH_LIGHT must reside in the active light list; otherwise, shaded entities are not visible.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_set_light_state

Function: Lights and Shadows

Action: Sets the on/off state for a light.

Prototype:
```
outcome api_rh_set_light_state (
    RH_LIGHT* light,         // light
    logical on_off           // on/off state
                             // for the light
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
#include "baseutil/logical.h"
```

Description: This API sets the active state of the RH_LIGHT entity.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_material

Function: Materials

Action: Attaches a material to a list of entities.

Prototype:
```
outcome api_rh_set_material (
    ENTITY_LIST const&       // entities to which
        entities,            // to attach material
    RH_MATERIAL* material    // material to attach
        =(RH_MATERIAL*)NULL
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "rnd_husk/include/rh_enty.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: Geometric ENTITYs must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_material_color

Function: Materials, Color Patterns, Colors

Action: Sets the color of a material associated with a geometric entity.

Prototype:
```
outcome api_rh_set_material_color (
    ENTITY_LIST const& entities,// list of entities
    double red,                 // red color value
    double green,               // green color value
    double blue                 // blue color value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API sets the color of a material associated with a geometric entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_material_reflection

Function: Materials, Reflectance

Action: Sets the reflection properties in the material associated with a geometric entity.

Prototype:
```
outcome api_rh_set_material_reflection (
    ENTITY_LIST const&      // list of entities
        entity_list,        // to be set
    double ambient,         // ambient light value
    double diffuse,         // diffuse light value
    double specular,        // specular light value
    double exponent         // exponent value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API sets the reflection properties in the material associated with a geometric entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_material_texture

Function: Materials, Textures

Action: Sets the texture file name in the material associated with a geometric entity.

Prototype:
```
outcome api_rh_set_material_texture (
    ENTITY_LIST const&      // list of entities
        entity_list,        // to be set
    const char* tex_name    // texture file name
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/lists/lists.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This API sets the texture file name in the material associated with a geometric entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_material_transp

Function: Materials, Transparency

Action: Sets the transparency properties in the material associated with a geometric entity.

Prototype:
```
outcome api_rh_set_material_transp (
    ENTITY_LIST const&      // list of entities
        entity_list,        // to be set
    double transp           // transparency value
                            // (0 is transparent;
                            // 1 is opaque)
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/lists/lists.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This API sets the transparency properties in the material associated with a geometric entity. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_output_mode

Function: Rendering Control, Viewing

Action: Sets the current rendering output mode.

Prototype:
```
outcome api_rh_set_output_mode (
    Output_Mode output_mode // output mode to set
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”rnd_husk/api/rnd_api.hxx”
#include ”rnd_husk/include/rh_args.hxx”
```

Description: This API sets the current output mode. Any subsequent rendering operations output scan lines in the format governed by that mode. Refer to the chapter on Output Modes for a list of applicable output modes.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_set_reflect_arg

Function: Reflectance, Materials

Action: Sets an argument of a material’s reflectance component.

Prototype:
```
outcome api_rh_set_reflect_arg (
    RH_MATERIAL* material,  // material
    const char* arg_name,   // name of argument
    const Render_Arg& value // argument value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets arguments controlling the reflectance component specified by name. name must correspond to one of the supported arguments for the type of reflectance component associated with the RH_MATERIAL. The underlying argument type depends on the argument set with the Render_Arg class.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_reflect_comp

Function: Reflectance, Materials

Action: Sets a material's reflectance component.

Prototype:
```
outcome api_rh_set_reflect_comp (
    RH_MATERIAL* material,      // material
    const char*                 // name of
        reflectance_shader_name // reflectance shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: An RH_MATERIAL comprises four component shaders; a color shader, a displacement shader, a reflectance shader, and a transparency shader. When an instance of an RH_MATERIAL is created, it is created with a default set of component shaders. This API changes the type of the reflectance shader to the specified name.

Errors: The name of the reflectance component is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_reflect_status

Function: Reflectance, Materials

Action: Sets a material's reflection status.

Prototype:
```
outcome api_rh_set_reflect_status (
    RH_MATERIAL* material,  // material
    logical status          // toggle on/off
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API enables or disables a material's reflection shader. The reflection component is used if its status is TRUE; otherwise, it is ignored.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_render_mode

Function: Rendering Control, Viewing

Action: Sets the current render mode.

Prototype:
```
outcome api_rh_set_render_mode (
    Render_Mode render_mode // render mode to set
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
```

Description: This API sets the current render mode. Any subsequent rendering operations use the specified mode. Refer to the chapter on Render Modes for a list of applicable modes.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_set_resolution

Function: Faceting, Viewing

Action: Sets the default screen faceting resolution controls.

Prototype:
```
outcome api_rh_set_resolution (
    int width,              // image space width
    int height,             // image space height
    double                  // aspect ratio
        pixel_aspect_ratio, // in pixels
    double image_scale      // scale of the image
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API sets the default screen faceting resolution controls.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_set_sidedness

Function: Rendering Control

Action: Sets the sidedness of faces of entities.

Prototype:
```
outcome api_rh_set_sidedness (
    ENTITY_LIST const&      // entities to have
        entities,           // sidedness set
    int sides               // 0 (sidedness
                            // undefined),
                            // 1 (single–sided),
                            // or 2 (double–sided)
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API defines the surface of an entity to be single–sided or double–sided for rendering purposes. This may be used to help visualize faces of nonmanifold objects where it is desirable to view both sides of a face. Geometric entities must be of the type BODY, LUMP, SHELL, or FACE. A two–sided face is always treated as double–sided.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_sub_image

Function: Faceting, Viewing

Action: Sets the region of image importance within a screen image.

Prototype:
```
outcome api_rh_set_sub_image (
    int left,               // left side
    int right,              // right side
    int top,                // top
    int bottom              // bottom
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This API sets the region of image importance within a screen image.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_texture_space

Function: Texture Spaces

Action: Attaches a texture space to a list of entities.

Prototype:
```
outcome api_rh_set_texture_space (
    ENTITY_LIST const&      // entities to
        entities,           // have texture space
                            // attached
    RH_TEXTURE_SPACE*       // given
        texture_space       // texture space
        =(RH_TEXTURE_SPACE*)NULL
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/lists/lists.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
#include ″rnd_husk/api/rnd_api.hxx″
```

Description: This API sets the texture space associated with an entity list. Each ENTITY must be of the type BODY, LUMP, SHELL, or FACE.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_texture_space_arg

Function: Texture Spaces

Action: Sets an argument for a texture space.

Prototype:
```
outcome api_rh_set_texture_space_arg (
    RH_TEXTURE_SPACE*        // given
        texture_space,       // texture space
    const char* arg_name,    // name of argument
    const Render_Arg& value  // argument value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets arguments controlling the texture space specified by name. name must correspond to one of the supported arguments for the type of texture space associated with RH_TEXTURE_SPACE. The underlying argument type is depends on the argument set with the Render_Arg class.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_transp_arg

Function: Transparency, Materials

Action: Sets an argument of a material's transparency component.

Prototype:
```
outcome api_rh_set_transp_arg (
    RH_MATERIAL* material,   // material
    const char* arg_name,    // name of argument
    const Render_Arg& value  // argument value
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_args.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: This API sets arguments controlling the transparency component specified by name. name must correspond to one of the supported arguments for the type of transparency component associated with RH_MATERIAL. The underlying argument type depends on the argument set with the Render_Arg class.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_transp_comp

Function: Transparency, Materials

Action: Sets a material's transparency component.

Prototype:
```
outcome api_rh_set_transp_comp (
    RH_MATERIAL*                    // material
        material,                   // material
    const char*                     // name of
        transparency_shader_name // transparency
                                    // shader
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
```

Description: An RH_MATERIAL comprises four component shaders: a color shader, a displacement shader, a reflectance shader, and a transparency shader. When an instance of an RH_MATERIAL is created, it is created with a default set of component shaders. This API changes the type of the transparency shader to the specified name.

Errors: The name of the transparency component is not one of the supported types.

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_transp_status

Function: Transparency, Materials

Action: Sets a material's transparency status.

Prototype:
```
outcome api_rh_set_transp_status (
    RH_MATERIAL* material,   // material
    logical status           // TRUE (transparency
                             // on) or FALSE
                             // (transparency off)
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "rnd_husk/include/rh_enty.hxx"
#include "baseutil/logical.h"
```

Description: This API enables or disables a material's transparency shader. The transparency component is used if its status is TRUE; otherwise, it is ignored.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_set_view

Function: Faceting, Viewing

Action: Sets the view parameters that affect view–dependent faceting.

Prototype:
```
outcome api_rh_set_view (
    SPAposition from_point, // from point
    SPAposition to_point,   // to point
    SPAvector view_up_vector,   // normal to view
    Projection_Type         // given
        projection,         // type of projection
    double field_of_view    // field of view
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/vector.hxx"
```

Description: This API sets the view parameters that affect view–dependent faceting.

In view–dependent faceting, objects outside the field of view are left unfaceted, which is similar to treatment of objects outside of the clipping planes.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: Changes model

api_rh_terminate

Function: Rendering Control, Rendering Control, Viewing

Action: Terminates the renderer.

Prototype:
```
outcome api_rh_terminate ();
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This API destroys the internal data structures required by the renderer. It must be the final Rendering Base Component function call.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_rh_terminate_image_utilities

Function: Image Output

Action: Terminates the use of the Image Format Utility Library.

Prototype:
```
outcome api_rh_terminate_image_utilities ();
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: This function terminates the use of the Image Format Utility Library, and it must match a corresponding call to LwInitialiseImageUtilities. Any data structures that have been allocated by the library are de–allocated. Call this function when terminating a rendering session with api_rh_terminate.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

api_terminate_rendering

Function: Rendering Control, Modeler Control

Action: Terminates the rendering library.

Prototype:
```
outcome api_terminate_rendering ();
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "rnd_husk/api/rnd_api.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/api/rnd_api.hxx

Effect: System routine

is_ATTRIB_COL

Function: Colors

Action: Determines if the entity is an ATTRIB_COL.

Prototype:
```
logical is_ATTRIB_COL (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/attribs/col_attr.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/attribs/col_attr.hxx

Effect: Read–only

is_ATTRIB_RGB

Function: Colors

Action: Determines if the entity is an ATTRIB_RGB.

Prototype:
```
logical is_ATTRIB_RGB (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/attribs/rgb_attr.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/attribs/rgb_attr.hxx

Effect: Read–only

is_Background_Type

Function: Backgrounds and Foregrounds

Action: Determines if a character string represents a legal background type.

Prototype:
```
logical is_Background_Type (
    const char* type        // entity
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″rnd_husk/intrface/rh_util.hxx″
#include ″baseutil/logical.h″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rh_util.hxx

Effect: Read–only

is_Color_Type

Function: Colors, Color Patterns

Action: Determines if a character string represents a legal color type.

Prototype:
```
logical is_Color_Type (
    const char* type        // string
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "rnd_husk/intrface/rh_util.hxx"
#include "baseutil/logical.h"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rh_util.hxx

Effect: Read–only

is_Displace_Type

Function: Displacement

Action: Determines if a character string represents a legal displacement type.

Prototype:
```
logical is_Displace_Type (
    const char* type        // string
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "rnd_husk/intrface/rh_util.hxx"
#include "baseutil/logical.h"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rh_util.hxx

Effect: Read–only

is_Foreground_Type

Function: Backgrounds and Foregrounds

Action: Determines if a character string represents a legal foreground type.

Prototype:
```
logical is_Foreground_Type (
    const char* type        // string
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″rnd_husk/intrface/rh_util.hxx″
#include ″baseutil/logical.h″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rh_util.hxx

Effect: Read–only

is_Light_Type

Function: Lights and Shadows

Action: Determines if a character string represents a legal light type.

Prototype:
```
logical is_Light_Type (
    const char* type        // string
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″rnd_husk/intrface/rh_util.hxx″
#include ″baseutil/logical.h″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rh_util.hxx

Effect: Read–only

is_NORENDER_ATTRIB

Function: Rendering Control

Action: Determines if the entity is a NORENDER_ATTRIB.

Prototype:
```
logical is_NORENDER_ATTRIB (
    const ENTITY* e         // entity to test
    );
```

```
int is_NORENDER_ATTRIB (
    const ENTITY*           //
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/kerndata/data/entity.hxx”
#include ”rnd_husk/attribs/no_rend.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/attribs/no_rend.cxx
rbase/rnd_husk/attribs/no_rend.hxx

Effect: Read–only

is_Reflect_Type

Function: Reflectance

Action: Determines if a character string represents a legal reflection type.

Prototype:
```
logical is_Reflect_Type (
    const char* type        // string
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”rnd_husk/intrface/rh_util.hxx”
#include ”baseutil/logical.h”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rh_util.hxx

Effect: Read–only

is_RH_BACKGROUND

Function: Backgrounds and Foregrounds

Action: Determines if an ENTITY is an RH_BACKGROUND.

Prototype:
```
logical is_RH_BACKGROUND (
    const ENTITY* e          // entity
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kerndata/data/entity.hxx”
#include ”rnd_husk/include/rh_enty.hxx”
#include ”baseutil/logical.h”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/include/rh_enty.hxx

Effect: Read–only

is_RH_ENTITY

Function: Rendering Control

Action: Determines if an ENTITY is a RH_ENTITY.

Prototype:
```
logical is_RH_ENTITY (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/kerndata/data/entity.hxx”
#include ”rnd_husk/include/rh_enty.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/include/rh_enty.hxx

Effect: Read–only

is_RH_ENVIRONMENT_MAP

Function: Environment Maps

Action: Determines if the entity is a RH_ENVIRONMENT_MAP.

Prototype:
```
logical is_RH_ENVIRONMENT_MAP (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/include/rh_enty.hxx

Effect: Read–only

is_RH_FOREGROUND

Function: Backgrounds and Foregrounds

Action: Determines if an ENTITY is an RH_FOREGROUND.

Prototype:
```
logical is_RH_FOREGROUND (
    const ENTITY* e          // entity
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
#include ″baseutil/logical.h″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/include/rh_enty.hxx

Effect: Read–only

is_RH_LIGHT

Function: Lights and Shadows

Action: Determines if an ENTITY is an RH_LIGHT.

Prototype:
```
logical is_RH_LIGHT (
    const ENTITY* e          // entity
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kerndata/data/entity.hxx″
#include ″rnd_husk/include/rh_enty.hxx″
#include ″baseutil/logical.h″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/include/rh_enty.hxx

Effect: Read–only

is_RH_MATERIAL

Function: Materials

Action: Determines if an ENTITY is an RH_MATERIAL.

Prototype:
```
logical is_RH_MATERIAL (
    const ENTITY* e          // entity
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kerndata/data/entity.hxx”
#include ”rnd_husk/include/rh_enty.hxx”
#include ”baseutil/logical.h”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/include/rh_enty.hxx

Effect: Read–only

is_RH_TEXTURE_SPACE

Function: Texture Spaces

Action: Determines if an ENTITY is an RH_TEXTURE_SPACE.

Prototype:
```
logical is_RH_TEXTURE_SPACE (
    const ENTITY* e          // entity
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kerndata/data/entity.hxx”
#include ”rnd_husk/include/rh_enty.hxx”
#include ”baseutil/logical.h”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/include/rh_enty.hxx

Effect: Read–only

is_Texture_Space_Type

Function: Texture Spaces

Action: Determines if a character string represents a legal texture space type.

Prototype:
```
logical is_Texture_Space_Type (
    const char* type        // string
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "rnd_husk/intrface/rh_util.hxx"
#include "baseutil/logical.h"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rh_util.hxx

Effect: Read–only

is_Transp_Type

Function: Transparency

Action: Determines if a character string represents a legal transparency type.

Prototype:
```
logical is_Transp_Type (
    const char* type        // entity
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "rnd_husk/intrface/rh_util.hxx"
#include "baseutil/logical.h"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rh_util.hxx

Effect: Read–only

remove_rbase_app_cb

Function: Rendering Control, Callbacks

Action: Removes the callback.

Prototype:
```
void remove_rbase_app_cb (
    rbase_app_callback* cb  // given callback
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″rnd_husk/intrface/rbapp_cb.hxx″
```

Description: Refer to action.

Errors: None

Limitations: None

Library: rnd_husk

Filename: rbase/rnd_husk/intrface/rbapp_cb.hxx

Effect: System routine