

Transformation Rules

Transformation rules are the key concept of the GUI Version. Once the customer application is started, the server converts every character-mode screen to a dialog according to the defined transformation rules. All rules, except the built-in rules, are defined by the developer.

If the application for which you plan to define transformation rules does not follow any standards, it will be a time-consuming task to define the rules. Therefore, before defining the transformation rules, it may be helpful to apply minor modifications to your legacy application in order to give it a homogeneous look. This step can include the use of similar application headers (titles, application names, dates), similar use of function keys and message lines, etc.

If your Natural application uses the technique of dynamically changing the fields of a map, you should turn the Natural output optimization off. Otherwise, Entire Screen Builder does not get consistent screen input. If Natural output optimization is turned on, the screens are sent in a compressed form. In this case, the field attributes may also be compressed. This makes recognizing and defining the rules for such a screen very complicated.

Use the Natural parameter `DCS=OFF` to turn the Natural output optimization off when you start Natural. You can also use the Natural statement `%ROOFF`. Note that `%ROOFF` only works correctly in the mainframe version of Natural 3.1 when ZAP NA46014 has been applied.

Entire Screen Builder distinguishes the following types of transformation rules, offering different levels of "GUIfication":

- Built-in Rules
- Basic Rules
- Extended Rules

The transformation rules are structured in scopes, where each scope can have its own set of transformation rules. A scope is entered when the screen detection defined for this scope is successful. If scope detection is not successful for a screen, global scope is used (if not defined otherwise with the **Last Rules Used** command). Scope detection is performed each time a new screen is received from the host. For further information, see *Using Different Scopes* in the documentation *Defining the Rules Using the SDK*.

Built-in Rules

The built-in rules cannot be changed by a developer. They guarantee that an application is executed by means of Entire Screen Builder as it is normally executed using the character interface.

The built-in rules are:

- Each output field in the character screen is converted to a static text control.
- Each input field in the character screen is converted to an edit box control.
- Double-clicking the left mouse button is equivalent to pressing the ENTER key.
- When the server application is running on a UNIX system, a special cursor is provided for fields with online help. Help can then be displayed by pressing the right mouse button.

Basic Rules

Basic rules immediately improve the user interface of an application with very little development effort.

Depending on the scope in which the basic rules have been defined, the rules are applied to all screens (global scope), only to screens of a specific application (application scope) or only to a specific screen (map scope).

Examples for basic rules are:

- Display the name of the application in the title bar of the application window.
- Display the fields sent by the application with a specific font or color.
- Provide command (push) buttons for all available host keys.
- Convert popup windows of the character screens to child windows.

For an overview of all basic rules, see the *Basic Rules Reference*.

Extended Rules

Extended rules provide the highest level of transformation. A character screen application can be transformed in such a way that it almost looks like a native Windows application. However, this requires quite some development effort.

If you want to create extended rules, you have to create a DLL using a resource editor such as Microsoft Visual Studio. This DLL must contain a dialog for each character screen on which the extended rules are to be applied. A dialog may contain controls such as option (radio) buttons, combo boxes and command (push) buttons. You then have to load the DLL into Entire Screen Builder's SDK in order to apply the extended rules. This includes the definition of the corresponding fields in the character screens.

Examples for extended rules are:

- Convert input fields consisting of several lines to edit boxes.
- Convert input fields in which one option from a number of mutually exclusive alternatives has to be entered to option (radio) buttons.
- Display list view controls for character screens with several rows where each row has an input field.
- Display tab controls for character screens that can be divided into several logical units.

For an overview of all extended rules, see the *Extended Rules Reference*.