1 Differences to the WinCC Basic System

1.1 Script execution

Unlike the WinCC Basic system, the event-triggered scripts (OnClick etc.) are executed in the same process space as the picture display, that is to say, in the context of the Internet Explorer.

This is to be taken into consideration when working with delays in scripts (e.g. Sleep(3000)). In this case, the picture displayed naturally cannot be used for precisely this period of time.

1.2 Picture selection

In the Web Navigator, a WinCC picture is downloaded asynchronously via the Internet. That is why the picture change is not synchronous, but asynchronous compared with the Basic system. You must consider this fact when using scripts that trigger a picture change.

The following holds:

If a picture change to a picture window is triggered in a script, then objects of the new picture cannot be accessed within the same script, because the picture is not yet loaded at that time. A delay (e.g. Sleep(2000)) cannot help here either, because the script and thus also the Internet Explorer are waiting.

A better solution is to execute the subsequent script parts of the OpenPicture event that is triggered when the picture is loaded.
1.3 Scripting

Preprocessor definition:

There is a preprocessor definition especially for the Web Navigator in the scripting.

RUN_ON_WEBNAVIGATOR

Code that comes under this preprocessor definition is executed only in the Web Navigator. Thus, it is possible to write scripts that behave differently in Web Navigator than in the Basic system.

Example:

```c
void OnOpenPicture (char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName)
{
  #ifdef RUN_ON_WEBNAVIGATOR
    // here you write the code that will be executed only in the Web Navigator
  }
  #else
    // here you write the code that will be executed only in the Basic system
  #endif
}
```
1.4 Addressing of objects in other pictures

If you want to access objects of another picture (e.g. objects in the parent picture or objects in a picture window) by script from a picture, you can only do this with relative addressing, as opposed to the Basic system.

All C script functions that access objects of a picture demand the picture name as the first parameter.

Example:  `SetPropChar ("Start picture", "Text1", "Text", "Hallo here is a text");`

When accessing objects that are not in the same picture, there are specially defined picture names in the Web Navigator, which specify the addressing of the pictures relative to the current picture.

Addressing an object in the basic picture

If there is uniqueness, you simply have to specify the picture name as in the Basic system.

`SetPropChar ("Start picture", "Text1", "Text", "Hallo here is a text");`

Otherwise, independent of the picture name, the addressing can be made either as an empty string or as `("/")`.

Example:

`SetPropChar ("", "Text1", "Text", "Hallo here is a text");`

or

`SetPropChar ("/", "Text1", "Text", "Hallo here is a text");`

Addressing an object in the parent picture

If there is uniqueness, you simply have to specify the picture name of the parent picture as in the Basic system.

`SetPropChar ("Picture1", "Text1", "Text", "Hallo here is a text");`

Otherwise, independent of the picture name, the addressing can be made either as `("../")`.

Example:

`SetPropChar ("../", "Text1", "Text", "Hallo here is a text");`
Picture in a picture window

If the object to be addressed is in a picture of a picture window "Picture window 1", then this can be addressed as follows relative to the current picture.

SetPropChar ("./Picture window 1", "Text1", "Text", "Hallo here is a text");

or, if the picture window 1 is in the parent picture,

SetPropChar ("../Picture window 1", "Text1", "Text", "Hallo here is a text");

or, if the picture window 1 is in the basic picture,

SetPropChar ("/Picture window 1", "Text1", "Text", "Hallo here is a text");

1.5 Synchronous script functions

You must be careful with synchronous script functions like SetTagCharStateWait etc., which write a tag synchronously and then wait for the event. Since these calls go synchronously to the server, frequent calling of these script functions can have a great negative influence on the performance of the client. Via an Internet connection, such a call can even take seconds.

Script functions that manipulate the objects in the picture displayed are safe, because there is no round trip to the server.
1.6 Define local picture script variables

You should avoid data manager tags in the Web Navigator if you want to mark any picture-specific data in scripts. If they are read or written in scripts, these always cause a round trip to the server, which is expensive in the Internet.

There are several ways of obtaining picture-specific data.

Configure "invisible" graphical objects

By creating invisible graphical objects, you can use the properties of these invisible objects as picture-specific tags.

In addition, you have the advantage of being able to perform actions by changing these stored values.

Example:

Creating an invisible text field.

Here, you can conveniently save any texts in the Text property, which can then be read or written from any script handler.

In the Color properties of the text field, you can then save colors that can be read or written via scripts and thus influence the state of the picture without a round trip to the server being necessary.