Table of Contents

[Change history 2](#_Toc106798346)

[General hints 3](#_Toc106798347)

[Apps 3](#_Toc106798348)

[Libraries 3](#_Toc106798349)

[Startup 4](#_Toc106798350)

[Known bugs 5](#_Toc106798351)

[Motion App 5](#_Toc106798352)

[HMI 5](#_Toc106798353)

[PLC 5](#_Toc106798354)

[Restrictions 5](#_Toc106798355)

[Features 6](#_Toc106798356)

[Step mode for debugging 11](#_Toc106798357)

[Code generation for inputs/outputs 6](#_Toc106798358)

[Copy & Paste python script to editor and save it 6](#_Toc106798359)

[Definition of the names of the inputs/outputs 7](#_Toc106798360)

[Export the names to a csv file 7](#_Toc106798361)

[Start the Python script 7](#_Toc106798362)

[Alarm Management 8](#_Toc106798363)

[Adapt used axes and kinematic limits 8](#_Toc106798364)

[Adapt axis configuration 9](#_Toc106798365)

[3D viewer 10](#_Toc106798366)

[Python script 11](#_Toc106798367)

[WebIQ 12](#_Toc106798368)

[Docu 12](#_Toc106798369)

[Adapt design 12](#_Toc106798370)

# Change history

V0.4 April 14, 2022

* PLC
  + Improvements in state machine
  + Code optimization for stability and better maintainability
  + Update of internal visualization
  + Additional checks in code generator for Input/Output force mechanism
  + There is still a bug in KinInterface library, that some errors could not be deleted.

V0.3 March 18, 2022

* PLC
  + Bugfix in state machine. Now all errors can be cleared
  + Code optimization for stability and better maintainability
  + Step mode in PLC automatic program for debugging
  + Configuration of motion data like max. Jog velocity, Kinematic limits
  + Usage of safe areas in ctrlX CORE
  + Code generator for Input/Outputs force mechanism
  + EuroMap variables are moved to variables ix: ST\_INPUT and qx : ST\_OUTPUT;
* HMI
  + Rework of GUI design
  + Update of changed OpcUa variable names
  + Display hint, when WebIQ screen is no editable, because of wrong op mode or user.

V1.0 June 21, 2022

* PLC
  + Implement State machine in ST, this allows customer specific code extensions
  + Workaround for Start/Pause in mode Automatic
  + Code optimization for stability and better maintainability
* HMI
  + Rework of GUI design

# General hints

This template is based on an example of a customer’s project. Therefor keep in mind the following things:

* Code is customer specific and need to be adapted to your requests.
* Be careful with changes, to ensure that everything is still working correct.
* Code uses specific axes, kinematic names. We continuously working on making the code more general.
* Implementation is on some places “quick and dirty”
  + due to constantly changing requirements
  + lack of time
  + still looking for best/optimized solution
* Missing verification for successful execution of some action.
* The development for a template project based on this example is in progress

To understand this short documentation, you must be familiar with the basic handling of ctrlX AUTOMATION and WebIQ.

The template is created with ctrlX CORE version 1.12.7 (RM21.11.6). Use this version for first steps with it.

# Apps

The following apps are necessary to run all features of the provided example.

Mandatory

* EtherCAT Master (Get access to I/Os & drives via EtherCAT fieldbus)
* Motion (Set up and commission axes and kinematics)
* OpcUa server (for communication to WebIQ)
* PLC (develop and run PLC applications)
* WebIQ Server App (To execute HMI app you need to install WebIQ designer on your PC)

Optional

* 3D Viewer (Display 3D model of moving kinematics)
* IDE (Integrated develop environment for python with file manager)
* Python Runtime (to execute python scripts)

You need the licenses for all apps. For WebIQ Server you need a license with 500 items or more.

# Libraries

* The libraries App\_CommonData and App\_Tools are not shipped with the ctrlX PLC Engineering installation.
* In the open library CXA\_MotionInterfaceUser variables were added

These 3 libraries are included in the project archive.

# Startup

ctrlX CORE:

* Install apps and licenses (see chapter Apps)
* Create user *opcua* with password *boschrexroth*
* Alternatively, you can adapt the user in WebIQ used for OpcUa communication to user *boschrexroth*with the matching password.

Motion:

* Create Axis **X**,**Y** & **Z** you can keep the default settings
* Create kinematic **Robot** assign **X**,**Y** & **Z** to robot

PLC:

* Open project archive **PLC-Template.projectarchive** in ctrlX PLC Engineering
* “Update device” to ctrlX CORE or ctrlX COREvirtual   
  (**restriction for ctrlX COREvirtual**: WebIQ must run with WebIQ PC server. Currently there is no WebIQ app with a corresponding license for the ctrlX COREvirtual.)
* Adapt IP address. Download & start program
* Adapt “Task Configuration”. Used settings:
  + ctrlX COREvirtual: Cycle time: 40ms, Watchdog 40ms, Sensitivity 10
  + ctrlX CORE: Cycle time: 20ms, Watchdog 20ms, Sensitivity 10

**Hint**: ctrlX COREvirtual has no real time operating system, therefore the cycle time and watchdog must be big, to avoid watchdogs. Or disable the watchdog.

* Dependent on your drive hardware, set PersistentVars.iPowerVars to 1-3 to select a power activation method mPower1()-mPower3().
* Without real axes, ignore the value of iPowerVars
* So far, no performance tests and analyses have been carried out.
* Therefore, no experience about shorter and longer cycle times is available.

WebIQ Designer:

* Username: *admin*
* Password: *boschrexroth* (=default password. Must be changed at first login)
* Upload **HMI-Template.zip**
* Start program
* Open program in Web browser
* Define point on Home/Teachpoints as shown below (Mode Manual)

# Known bugs

## Motion App

* Blend distances are constant. Commanded value not used right now. **fixed in 1.12.1**
* For some smaller distances the robot velocity is lower than commanded value: **fixed in 1.12.1**
* Multiple changes of the override can cause unclearable error. **fixed 1.12.0**
* Commanded distances smaller than 0.000001 can cause an interpolation error: **fixed in 1.12.0**
* Sometimes the jog movement is not stopped in Manual jog mode, when entering in a Safe Area. Workaround implemented. Use high acceleration & small cycle time for early stops. **fixed in 1.12.4**
* Multiple changes between START and PAUSE on HMI (~10 x in 3 seconds), can result in stopping the kinematic movement. To start movement again. Press STOP and Start. Workaround implemented
* Moving slow in Auto mode, generates sometimes an error
* For further information see release notes of ctrlX CORE

## HMI

* Visual Programming are still empty
* Graphical programming page (=Tiger from IDE app) not working correctly => disabled
* Nested lock of editable items sometimes not works. This means e.g. an item is editable although it is disabled.
* Unicode characters (e.g. special Turkish chars) can’t be exchanged via OpcUa

## PLC

* Library CXA\_MOTIONINTERFACE 1.12.1.0 must be used. With later versions 1.12.2.x the motion interface can get stuck when switching a kinematic to mode Ab. Workaround implemented
* Some errors can only be cleared by reset of PLC program
* Sometimes activating an OP fails and must be selected a second time
* “Axis in reference” is not checked in the program, because info is not available
* Sometimes override value is not used correct (slow movement). Change override to fix.

## Restrictions

* In mode Automatic and Manual, the Safe Areas are used
* In mode Setup the Safe Areas are not used.

# Features

* Jog axis in mode **Setup** (X, Y, Z) (HMI menu item: Home/Machine Overview)
* Jog kinematic in mode **Manual** (Robot) (Home/Machine Overview)
* Table to edit teach points (Home/Teachpoints)
* PLC example code moving between 3 different points (Pos-Home, Pos-Mold-Up, Pos-User-10)

See *FB\_Machine\_Customer.mAutoPlc()*

* Recipe handling with Open, Save, SaveAs and Delete in mode **Manual** & **Setup**(Home/Machine Overview) (PLC recipe structure GVL\_GatCore.stRecipe)
* Last loaded recipe is automatically loaded at ctrlX CORE startup
* Alarm handling (Diagnostics Alarms Online/Alarms History)
* Predefined HMI languages: German, English and Turkish (Settings/General Settings)
* HMI Usermanager which allows to add, edit, delete… users (Settings/Usermanager)
* 3D viewer showing the moving kinematics (Hint: URL must be adapted in WebIQ)
* PLC example code to manage Python interpreter (create instance, start script, …)  
  See *FB\_Machine\_Customer.mAutoPython()*
* Axis/Kin configuration checking at startup of PLC program. Problems are shown in HMI.
* Communication check: HMI is locked, when not connected to PLC

# Code generation for inputs/outputs

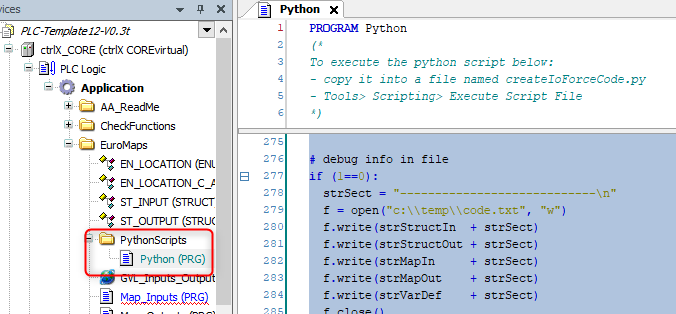
The template includes a mechanism that allows forcing of inputs and outputs via the HMI. The code required for this feature is generated by a Python script. No additional work, like adding OpcUa variables or Hmi screens, is necessary. So, you save a lot of implementation time.

The following work steps are necessary for this:

* Copy & Paste python script to a text editor (e.g. notepad) and save it (only once)
* Definition of the names of the inputs/outputs
* Export the names to a csv file
* Start the Python script

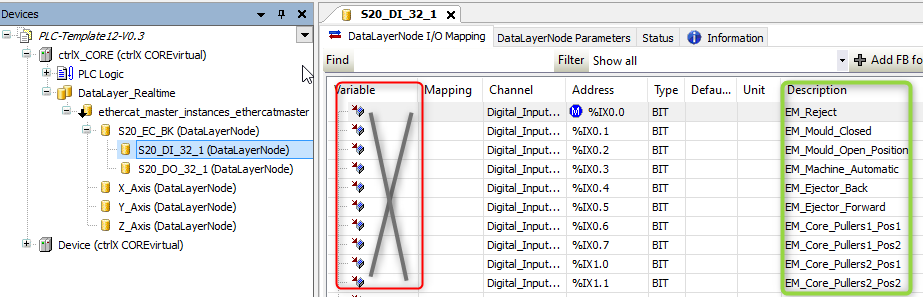
## Copy & Paste python script to editor and save it

Creating script must be done only once. The second time the script can be used directly.

* Open the python script in PLC engineering   
  
* Mark the complete python program code and copy it to a text editor

## Definition of the names of the inputs/outputs

Open the configuration mapping page. Write the variable name of inputs/outputs into the description field of the input/output page. Keep in mind to create valid PLC variable names.

The column "Variables" must not be filled in. This prevents writing to the IOs via 2 different paths. 

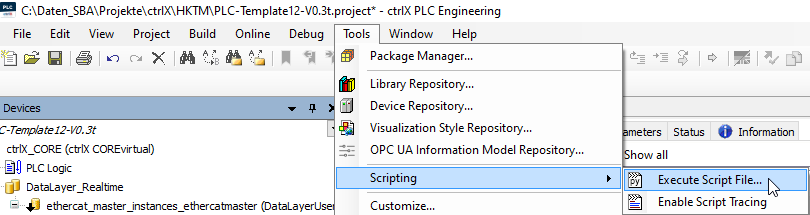
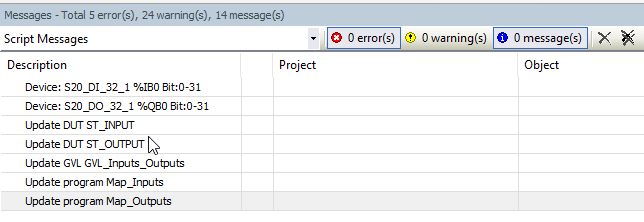
## Export the names to a csv file

Right click the bus coupler, select “Export mappings to CSV…” and provide a valid file name.

|  |  |
| --- | --- |
|  | Hint: Do not export the Ethercat master node,   to avoid script failure. |

## Start the Python script

In PLC Engineering

* Click on Tools/Scripting/Execute Script File  
  
* Start the python script, by selecting it within the file browser
* Select your exported csv file, confirm with “Open”
* Check Message window, if code generation was successful and which DUT, GVL and POUs are updated.  
  
* Compile the program. When compile errors are thrown, fix the invalid variable names. Start creation process again.

# Alarm Management

Alarm IDs starts at 1000. This ensures a correct numeric sort in the alphabetically sorted “Localization Manager” list of WebIQ. This section shows the steps how to create a new error.

1. PLC: In EN\_APP\_ALARM add a new error enumeration item  
   e.g APE\_APP\_SECURITY\_DOOR\_OPEN := 1015
2. PLC: To raise the error call method FB\_Machine\_Base.Tools.mAlarmSet().  
   Beside the error enum you can provide 3 optional strings, which are displayed in HMI.  
   IF bDoorOpen THEN  
    mAlarmSet(enAppError:= APE\_APP\_SECURITY\_DOOR\_OPEN, '', '', '');

END\_IF

1. WebIQ: In “Localization Manager” create variable *alm1015* if not exists and enter the alarm text for all languages.

We have implemented a 30-channel multiplexing for alarm handling with the following features.

* The number of different alarms is unlimited
* Up to 30 alarms can be displayed at the simultaneously time
* With increasing number of alarms, there is no need to add further communication variables
* Only the translation text must be defined in WebIQ.

When you do not like this behavior, you can use the default implementation of WebIQ where a single variable for every alarm item is used.

# Adapt used axes and kinematic limits

In **FB\_Machine\_Customer/00\_Init/mUserCfg** some settings for SetupMode can be adapted to application needs. E.g. Set max. jog velocity for each axis is set to 10% of the max. axis velocity.

arAxisStatus\_gb[uiAxNo].Lim.lrVelJogMax := MIN(arAxisStatus\_gb[uiAxNo].Lim.lrVelPos,arAxisStatus\_gb[uiAxNo].Lim.lrVelNeg) \* 0.1;

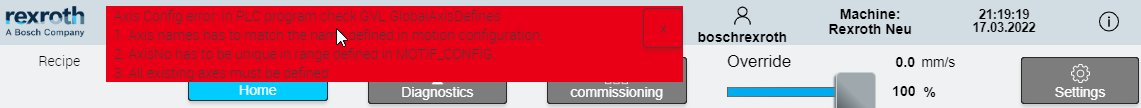
**Hint**: The structures TE\_AXIS\_STATUS\_TYPE01 and TE\_KINEMATICS\_STATUS\_TYPE01 of the library CXA\_MotionInterfaceUser have been extended by application specific variables.

# Adapt axis configuration

For the first steps you should take the axis settings. But later you might have to add axes or change their names. To do this adapt the axis and kinematic variables in Application.MotionInterface.GlobalAxisDefines & Application.MotionInterface.GlobalKinematicsDefines.

**Hint**: When you change the axis names, you must change them in your 3D viewer model, too.

In the init procedure of the program this data is verified. If a configuration error is detected it is shown in the HMI. The error must be fixed to run the program. E.g. You define a axis name which not exists:

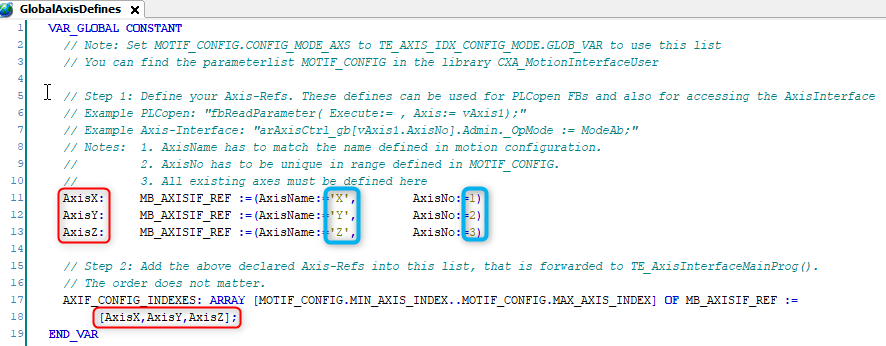


When a config error happens, check the following:

|  |  |
| --- | --- |
| TE\_AxisInterfaceMainProg | TE\_KinInterfaceMainProg |
| 1. AxisName has to match the name defined in motion configuration.  2. AxisNo has to be unique in range defined in MOTIF\_CONFIG.  3. All existing axes must be defined here | 1. KinName has to match the name defined in motion configuration.  2. GroupNo has to be unique in range defined in MOTIF\_CONFIG.  3. Axes assigned to kinematic must exist |

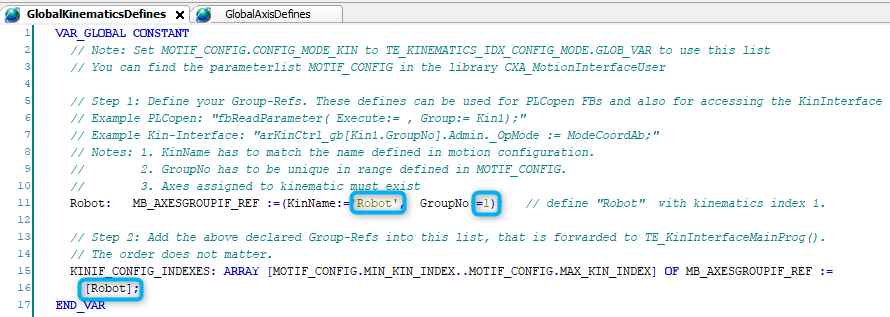
Config Error: TE\_AxisInterfaceMainProg

Definition see: MotionInterface/cfg\_axes-kinematic/GlobalAxisDefines/



Config Error: TE\_KinInterfaceMainProg

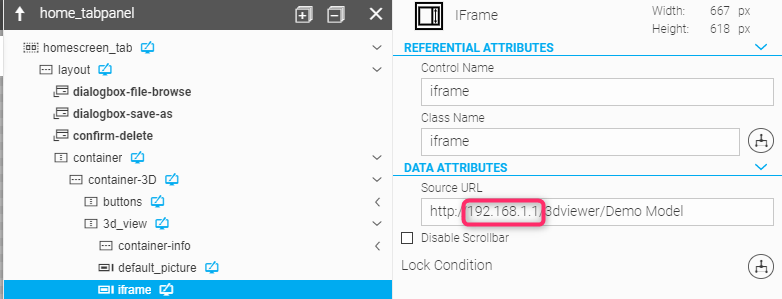
Definition see: MotionInterface/cfg\_axes-kinematic/GlobalKinematicsDefines/



# 3D viewer

To run the 3D viewer, you must adapt the IP-address of the iframe link to the IP of your ctrlX CORE. Alternatively you can set the IP of your ctrlX CORE to 192.168.1.1 . The axes name must be X,Y and Z to show movement, because this are the axes names used by the 3D model.

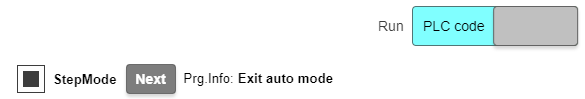
**Hint**: Only Firefox displays the 3D viewer within WebIQ. Chrome automatically changes to https protocol, which is currently not provided by the 3D viewer. Therefore, nothing is displayed. You can change this default behavior of Chrome. Please check internet how to do it.



# Debug screen

For demonstration purpose the PLC template can either run the PLC method or any python script. Select mode Manual, to enable the execution of “PLC code” or “python script”.

The plc state machine contains a step mode for testing the program. It can be switched on/off while the program is running.



# Python script

Python scripts can move axes and kinematics and read/write data layer variables. Even the teach points of this PLC template. An example how to do this, will be shipped in a later version.

The example script only reads/writes variables from the data layer, which are shown in WebIQ HMI:

To use demo HMI page for python script, do the following steps:

|  |  |
| --- | --- |
| * Install IDE & python app on ctrlX CORE * Start IDE / Textual Coding   IDE “Explorer” select path “script”   * Save embedded file **write\_2DataLayer.py**. Remove extension .txt which is necessary, because python scripts are blocked by Word. * Alternative to IDE you can upload the script file e.g. via Backup/Restore or WebDav protocol. |  |

import time

sNode\_bTest = '/plc/app/Application/sym/GVL\_OPCUA/bTest'

sNode\_sTest = '/plc/app/Application/sym/GVL\_OPCUA/strTest'

sNode\_iStopPython = '/plc/app/Application/sym/GVL\_OPCUA/iStopPython'

sStopMsg = '\*\*\* STOP signal received by python script => exit program \*\*\*'

i = 10

bTest = True

sTest = ""

iStop = 0

while (i > 0):

i -= 1

sTest = "Count down {}".format(i)

if bTest:

bTest = False

else:

bTest = True

datalayer.write(sNode\_bTest, bTest)

datalayer.write(sNode\_sTest, sTest)

time.sleep(1)

iStop = datalayer.read(sNode\_iStopPython)

if (iStop != 0):

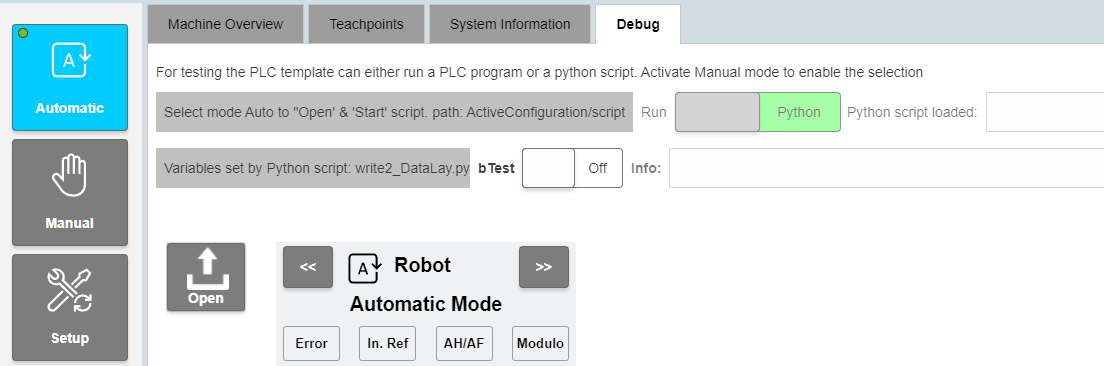
datalayer.write(sNode\_sTest, sStopMsg)

exit(0)

datalayer.write(sNode\_sTest, "\*\*\* Python script finished \*\*\*")

In WebIQ HMI:

* Select mode “Manual”
* On page Home/Visual Programming select Run: Python
* Select mode “Automatic” press button “Open” and select new created file
* “Start” the script which changes the fields “bTest” and “Info”



# WebIQ

## Docu

WebIQ is a HTML5 based visualization. It works totally different than WinStudio. When you do not have any HTML5 know how you should read view the videos and documentation provided by WebIQ.

**Hint**: Especially the selection of tabulator panels (=sub screens) takes some time to get familiar. At least check this video and the forum in the ctrlX AUTOMATION Community.

ctrlX AUTOMATION Community

<https://developer.community.boschrexroth.com/t5/ctrlX-AUTOMATION-Community/ct-p/dcdev_community>

WebIQ Forum (questions & answers)

<https://developer.community.boschrexroth.com/t5/Smart-HMI-WebIQ-Designer-and/bd-p/dcdev_community-dcae-smarthmi>

WebIQ Dokumentation

<https://docs.webiq.de/docs/webiq-designer-manual/>

WebIQ Videos

<https://www.smart-hmi.de/dokumentation/>

## Adapt design

Right now, there is no Theme manger, where you can easily define global design setting. Therefore, we started to implement some basic CSS stylesheets. This enables you to adapt some **basic** design settings of the HMI.

The CSS stylesheets are located at: Code Manager/Stylesheets.

* One predefined color scheme.
* Modify the colors in the scheme
* The color variables are used in the other css stylesheets. You can modify them, too.

|  |
| --- |
|  |

Modify CSS stylesheet

The names of the stylesheets describes where they are used. Just change the color variable, and reload the screen, then you can see the changed color.

The example below shows the stylesheet bg-content, where:

* background-color property uses the variable –col-bg-content.
* An example how to use gradient color is shown
* Padding for all sides is set to 4px
* Some border settings are defined.

|  |
| --- |
| /\* css/custom/bg-content.css \*/  /\* control base element \*/  .iq-container.bg-content {  padding-top: 4px;  padding-right: 4px;  padding-bottom: 4px;  padding-left: 4px;  background-color: var(--col-bg-content);  /\* example for gradient color  background-image: linear-gradient(to bottom right, var(--col-bg-content), white;  \*/  border-style: solid;  border-color: black;  border-width: 1px;  border-radius: 0px;  }  /\* element that contains child controls \*/  .iq-container.bg-content .margin-compensator {  } |

When you want to use this design feature, you always must use:

* the stylesheets for content areas.
* the button composite widgets

Please check the WebIQ application to find out how the stylesheets are used.