

# SAP Toolbar Control (BC-CI)



HELP.BCCITTOOLBAR

**Release 4.6C**



## Copyright

© Copyright 2001 SAP AG. Alle Rechte vorbehalten.

Weitergabe und Vervielfältigung dieser Publikation oder von Teilen daraus sind, zu welchem Zweck und in welcher Form auch immer, ohne die ausdrückliche schriftliche Genehmigung durch SAP AG nicht gestattet. In dieser Publikation enthaltene Informationen können ohne vorherige Ankündigung geändert werden.

Die von SAP AG oder deren Vertriebsfirmen angebotenen Software-Produkte können Software-Komponenten auch anderer Software-Hersteller enthalten.

Microsoft<sup>®</sup>, WINDOWS<sup>®</sup>, NT<sup>®</sup>, EXCEL<sup>®</sup>, Word<sup>®</sup>, PowerPoint<sup>®</sup> und SQL Server<sup>®</sup> sind eingetragene Marken der Microsoft Corporation.

IBM<sup>®</sup>, DB2<sup>®</sup>, OS/2<sup>®</sup>, DB2/6000<sup>®</sup>, Parallel Sysplex<sup>®</sup>, MVS/ESA<sup>®</sup>, RS/6000<sup>®</sup>, AIX<sup>®</sup>, S/390<sup>®</sup>, AS/400<sup>®</sup>, OS/390<sup>®</sup> und OS/400<sup>®</sup> sind eingetragene Marken der IBM Corporation.

ORACLE<sup>®</sup> ist eine eingetragene Marke der ORACLE Corporation.

INFORMIX<sup>®</sup>-OnLine for SAP und Informix<sup>®</sup> Dynamic Server<sup>™</sup> sind eingetragene Marken der Informix Software Incorporated.

UNIX<sup>®</sup>, X/Open<sup>®</sup>, OSF/1<sup>®</sup> und Motif<sup>®</sup> sind eingetragene Marken der Open Group.

HTML, DHTML, XML, XHTML sind Marken oder eingetragene Marken des W3C<sup>®</sup>, World Wide Web Consortium, Massachusetts Institute of Technology.

JAVA<sup>®</sup> ist eine eingetragene Marke der Sun Microsystems, Inc.

JAVASCRIPT<sup>®</sup> ist eine eingetragene Marke der Sun Microsystems, Inc., verwendet unter der Lizenz der von Netscape entwickelten und implementierten Technologie.

SAP, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo und mySAP.com sind Marken oder eingetragene Marken der SAP AG in Deutschland und vielen anderen Ländern weltweit. Alle anderen Produkte sind Marken oder eingetragene Marken der jeweiligen Firmen.

## Symbole

Symbol	Bedeutung
	Achtung
	Beispiel
	Empfehlung
	Hinweis
	Syntax
	Tip

## Inhalt

<b>SAP Toolbar Control (BC-CI)</b> .....	<b>5</b>
Instanz für das SAP Toolbar Control .....	6
Anlegen eines Controls am Beispiel des SAP Picture .....	7
Arbeiten mit dem SAP Toolbar Control .....	9
Registrieren und Verarbeiten von Ereignissen .....	11
Ereignisse des SAP Toolbar Controls .....	13
Verwendung von Controls im WAN .....	15
Spezielle Hinweise für das SAP Toolbar Control .....	17
<b>Methoden der Klasse CL_GUI_TOOLBAR</b> .....	<b>18</b>
constructor .....	19
add_button .....	21
add_button_group .....	23
fill_buttons_data_table .....	24
delete_button .....	26
delete_all_buttons .....	27
set_button_state .....	28
set_button_info .....	29
set_static_ctxmenu .....	30
assign_static_ctxmenu_table .....	32
track_context_menu .....	34
<b>Methoden des OO Control Frameworks</b> .....	<b>35</b>
<b>Methoden der Klasse CL_GUI_CFW</b> .....	<b>36</b>
dispatch .....	37
flush .....	38
get_living_dynpro_controls .....	39
set_new_ok_code .....	40
update_view .....	41
<b>Methoden der Klasse CL_GUI_OBJECT</b> .....	<b>42</b>
is_valid .....	43
free .....	44
<b>Methoden der Klasse CL_GUI_CONTROL</b> .....	<b>45</b>
finalize .....	46
set_registered_events .....	47
get_registered_events .....	48
is_alive .....	49
set_alignment .....	50
set_position .....	51
set_visible .....	52
get_focus .....	53
set_focus .....	54
get_height .....	55
get_width .....	56

## SAP Toolbar Control (BC-CI)

### Einsatzmöglichkeiten

Mit dem SAP Toolbar Control können Sie zusätzlich zur normalen Drucktastenleiste weitere Drucktastenleisten definieren:





### Integration

Das SAP Toolbar Control wird in einem beliebigen SAP Control Container eingebettet.

### Funktionsumfang

Mit dem SAP Toolbar Control können Sie zusätzliche Drucktastenleisten definieren. Auf der Drucktastenleiste können Sie folgende Objekte platzieren:

- Drucktasten
- Drucktaste mit Dropdown-Menü : Wählt der Benutzer die Drucktaste, wird eine für dieses Objekt voreingestellte Aktion ausgelöst. Klickt der Benutzer dagegen auf den Pfeil, erhält er ein Menü.
- Menü : Der Benutzer erhält auf Mausklick ein Menü.
- Trennlinien/-zeichen
- Drucktastengruppen (analog zu Auswahlfeldern)
- Toggle-Buttons (analog zu Ankreuzfeldern)

Betätigt der Benutzer eine dieser Drucktasten, erhält das Anwendungsprogramm über die Ereignissteuerung des Control Frameworks die Kontrolle. Als Ereignisparameter wird dabei der Funktionscode der ausgewählten Aktion an das Anwendungsprogramm übergeben.

### Einschränkungen

Das SAP Toolbar Control setzt das Microsoft Common Control voraus. Daher wird die Installation des MS Internet Explorers Version 4.0 vorausgesetzt.

---

Instanz für das SAP Toolbar Control

## Instanz für das SAP Toolbar Control

### Definition

Diese Instanz wird mit Referenz auf die Klasse `c1_gui_toolbar` definiert:

```
data toolbar type ref to c1_gui_toolbar.
```

### Verwendung

Eine Instanz für das SAP Toolbar Control verwaltet alle Informationen bezüglich einer zusätzlichen Werkzeugleiste in Ihrem Dynpro. Auf diese Instanz können Sie Methoden aufrufen, mit denen Sie die Eigenschaften des Toolbar Controls definieren und ändern können.

### Integration

Die Klasse `c1_gui_toolbar` beinhaltet sowohl [control-spezifische \[Seite 18\]](#) Methoden als auch Methoden [des OO Control Frameworks \[Seite 35\]](#).

## Anlegen eines Controls am Beispiel des SAP Picture

### Voraussetzungen

Der folgende Ablauf ist für alle von SAP ausgelieferten Custom Controls anwendbar. In den Code-Beispielen wird immer auf das SAP Picture Control eingegangen. Für andere Controls ist aber im Prinzip nur die Klasse des Controls auszutauschen.

Weiterhin geht das Beispiel davon aus, daß das Custom Control in einem Custom Container eingebaut wird. Andere Szenarios werden in der Dokumentation zu den Control Containern beschrieben.

### Ablauf

#### Instanz anlegen

1. Definieren Sie eine Referenzvariable für den Custom Container, in dem das Custom Control angezeigt werden soll ([Siehe SAP Container \[Extern\]](#))

```
DATA container TYPE REF TO cl_gui_custom_container.
```

2. Definieren Sie eine Referenzvariable für das Picture Control:

```
DATA picture TYPE REF TO cl_gui_picture.
```

3. Erzeugen Sie den Custom Container. Den Bereich 'CUSTOM' für den Custom Container müssen Sie vorher im Screen Painter angelegt haben. Beim Erzeugen des Containers legen Sie auch seine [Lebensdauer \[Extern\]](#) fest ([siehe constructor \[Extern\]](#)).

```
CREATE OBJECT container  
  EXPORTING container_name = 'CUSTOM'  
           lifetime       = lifetime.
```

4. Erzeugen Sie das Picture Control. Für dieses können Sie auch eine Lebensdauer festlegen. Allerdings darf die Lebensdauer nicht größer sein als die seines Containers.

```
CREATE OBJECT picture  
  EXPORTING parent = container  
           lifetime = lifetime.
```

#### Ereignis anmelden

5. Das Anmelden von Ereignissen gliedert sich in das Registrieren des Ereignisses am Control Framework, das Definieren einer Behandlungsmethode und das Anmelden der Behandlungsmethode. Diese Schritte finden Sie in [Registrieren und Verarbeiten von Ereignissen \[Seite 11\]](#).

#### Arbeiten mit dem Control

6. Diese Schritte sind control-spezifisch und werden hier nicht beschrieben.

#### Abbau der Controls

In der Regel übernimmt das [Lifetime Management \[Extern\]](#) den Abbau der verwendeten Controls. Wenn Sie die Controls aber in Ihrem Programm selbst abbauen wollen, führen Sie folgende Schritte aus:

**Anlegen eines Controls am Beispiel des SAP Picture**

7. Bauen Sie das Custom Control mit der Methode [free \[Seite 44\]](#) am Frontend ab. Sofern Sie den Control Container nicht mehr benötigen, bauen Sie auch diesen ab:

```
CALL METHOD picture->free
  EXCEPTIONS cntl_error      = 1
             cntl_system_error = 2.
CALL METHOD container->free
  EXCEPTIONS cntl_error      = 1
             cntl_system_error = 2.
```



Beachten Sie die Reihenfolge, in der Sie die Controls am Frontend abbauen. Wenn Sie einen Container abbauen, werden nämlich automatisch alle in dem Container eingebundenen Controls auch abgebaut. Der Versuch, ein bereits abgebautes Control erneut abzubauen, führt zu einem Fehler. Mit der Methode [is\\_alive \[Seite 49\]](#) kann nachgeprüft werden, ob das Control schon abgebaut wurde.

8. Löschen Sie die Referenzvariablen des Custom Controls und des Control Containers:

```
FREE PICTURE.
FREE CONTAINER.
```

## Arbeiten mit dem SAP Toolbar Control

In diesem Abschnitt werden die für das SAP Toolbar Control spezifischen Funktionen aufgeführt.

### Voraussetzungen

Der in diesem Abschnitt beschriebene Prozeß stellt nur eine control-spezifische Ergänzung [des allgemeinen Prozesses zur Control-Einbindung \[Seite 7\]](#) dar und ist isoliert nicht lauffähig.

### Ablauf



Die Code-Abschnitte sind Beispiele, die nicht immer den vollen Funktionsumfang ausnutzen. Genauere Informationen finden Sie immer im Referenzteil dieser Dokumentation.

Das Beispielprogramm `SAPTOOLBAR_DEMO1` finden Sie im System.

### Instanziieren

1. Definieren Sie eine Referenzvariable für das SAP Toolbar Control:

```
DATA toolbar TYPE REF TO cl_gui_toolbar.
```

2. [Erzeugen Sie eine Instanz \[Extern\]](#) des SAP Toolbar Controls:

```
CREATE OBJECT toolbar
    EXPORTING parent = container.
```

3. Fügen Sie [einzelne Drucktasten \[Seite 21\]](#) oder [Drucktastengruppen \[Seite 23\]](#) in das Toolbar Control ein.

```
CALL METHOD toolbar->add_button
    EXPORTING fcode      = 'FUNC_1'
             icon       = '@03@'
             butn_type  = cntb_btype_dropdown
             text       = 'My Function'
    EXCEPTIONS cntl_error = 1.
```

### Ereignis anmelden

4. Melden Sie [Ereignisse für das SAP Toolbar Control \[Seite 13\]](#) an. Folgende Ereignisse unterstützt das Control:

Ereignisname	Bedeutung
FUNCTION_SELECTED	Drucktaste wurde gedrückt
DROPDOWN_CLICKED	Anforderung eines Kontextmenüs zu einer Drucktaste vom Typ <code>cntb_btype_dropdown</code> und <code>cntb_btype_menu</code> .

## Arbeiten mit dem SAP Toolbar Control

### Veränderung von Controleigenschaften zur Laufzeit

5. Fügen Sie weitere Drucktasten oder Drucktastengruppen hinzu.
6. Verändern Sie den [Status \[Seite 28\]](#) einzelner Drucktasten:

```
CALL METHOD toolbar->set_button_state
    EXPORTING  enabled = 'X'
              fcode   = 'FUNC_1'
    EXCEPTIONS cntl_error = 1.
```

7. Löschen Sie [eine \[Seite 26\]](#) oder [alle \[Seite 27\]](#) Drucktasten:

```
CALL METHOD toolbar->DELETE_BUTTON
    exporting fcode      = 'FUNC_1'
    exceptions CNTL_ERROR = 1.
```

8. Interpretieren Sie die vom Benutzer ausgewählten Funktionen des Toolbar Controls in der Ereignisbehandlungsmethode.

### Abbau der Controls

In der Regel übernimmt das [Lifetime Management \[Extern\]](#) den Abbau der verwendeten Controls. Wenn Sie die Controls aber in Ihrem Programm selbst abbauen wollen, führen Sie folgende Schritte aus:

9. Bauen Sie das Custom Control am Frontend ab. Sofern Sie den Control Container nicht mehr benötigen, bauen Sie auch diesen ab:

```
CALL METHOD toolbar->free.
```

10. Löschen Sie die Referenzvariablen des Custom Controls und des Control Containers:

```
FREE toolbar.
```



Als Beispiel für die Integration eines Toolbar Controls in eine Anwendung können Sie den Report `SAPTOOLBAR_DEMO1` heranziehen.

## Registrieren und Verarbeiten von Ereignissen

### Einsatzmöglichkeiten

Über den Ereignismechanismus des Control Frameworks können Sie in Ihrem Programm in Behandlermethoden auf Ereignisse reagieren, die auf dem Control ausgelöst wurden (z.B. Doppelklick).

### Voraussetzungen

Der Ablauf wurde allgemeingültig für alle Custom Controls gehalten. Control-spezifische Informationen finden Sie in der jeweiligen Dokumentation zu der Control-Verschaltung.

### Ablauf

1. Es wird davon ausgegangen, daß Sie mit einem Custom Control mit der Verschaltung `cl_gui_xyz` arbeiten:

```
DATA my_control TYPE REF TO cl_gui_xyz.
```

### Anmelden von Ereignissen beim Control Framework

2. Definieren Sie eine interne Tabelle (Typ `cntl_simple_events`) und einen dazugehörenden Arbeitsbereich (Typ `cntl_simple_event`).

```
DATA events TYPE cntl_simple_events.  
DATA wa_events TYPE cntl_simple_event.
```

3. Füllen Sie nun die Ereignistabelle mit den relevanten Ereignissen. Dazu benötigen Sie die Identifikation des Ereignisses (`event_id`). Diese Information finden Sie wiederum in der Klassenbibliothek in den Attributen der Klasse `cl_gui_xyz`. Weiterhin müssen Sie sich entscheiden, ob das Ereignis als Systemereignis (`appl_event = ' '`) oder als Applikationsereignis (`appl_event = 'X'`) definiert werden soll.

```
wa_events-eventid = event_id.  
wa_events-appl_event = appl_event.  
APPEND wa_events TO events.
```

4. Im nächsten Schritt muß die Ereignistabelle an das Frontend geschickt werden, damit die relevanten Ereignisse vom Frontend an das Backend weitergeleitet werden.

```
CALL METHOD my_control->set_registered_events  
events = events.
```

Um auf ein Ereignis Ihres Custom Controls eingehen zu können, müssen Sie nun noch eine Behandlermethode angeben. Die Behandlermethode kann eine Instanzmethode oder eine statische Methode sein:

### Behandlung des Ereignisses als Instanzmethode

5. Definieren Sie die (lokale) Klassendefinition für den Ereignisbehandler. Dabei legen Sie den Namen der Ereignisbehandlungsmethode fest (`Event_Handler`). Als Information müssen Sie sich in der Klassenbibliothek für die Klasse des Custom Controls `cl_gui_xyz` den Namen des Ereignisses (`event_name`) und die zu diesem Ereignis gehörenden Ereignisparameter (`event_parameter`) besorgen. Der Ereignisparameter `sender` wird bei jedem Ereignis zur

## Registrieren und Verarbeiten von Ereignissen

Verfügung gestellt. Der Parameter enthält die Referenz des Controls, das dieses Ereignis ausgelöst hat.

```
CLASS lcl_event_receiver DEFINITION.
PUBLIC SECTION.
METHODS Event_Handler
  FOR EVENT event_name OF cl_gui_xyz
  IMPORTING event_parameter
           sender.
ENDCLASS.
```

- Melden Sie jetzt noch für die registrierten Ereignisse Behandlungsmethoden beim OO Control Framework an.

```
DATA event_receiver TYPE REF TO lcl_event_receiver.
CREATE OBJECT event_receiver.
SET HANDLER event_receiver->Event_Handler
  FOR my_control.
```

## Registrieren als statische Methode

- Definieren Sie die (lokale) Klassendefinition für den Ereignisbehandler. Dabei legen Sie den Namen der statischen Ereignisbehandlungsmethode fest (**Event\_Handler**). Als Information müssen Sie sich in der Klassenbibliothek für die Klasse des Custom Controls **cl\_gui\_xyz** den Namen des Ereignisses (**event\_name**) und die zu diesem Ereignis gehörenden Ereignisparameter (**event\_parameter**) besorgen.

```
CLASS lcl_event_receiver DEFINITION.
PUBLIC SECTION.
CLASS-METHODS Event_Handler
  FOR EVENT event_name OF cl_gui_xyz
  IMPORTING event_parameter
           sender.
ENDCLASS.
```

- Melden Sie jetzt noch für die registrierten Ereignisse Behandlungsmethoden beim OO Control Framework an.

```
SET HANDLER lcl_event_receiver=>Event_Handler
  FOR my_control.
```

## Verarbeiten von Control-Ereignissen

- Wie Sie auf ein Ereignis reagieren wollen, legen Sie im Implementierungsteil der Behandlungsmethode fest.

```
CLASS lcl_event_receiver IMPLEMENTATION.
METHOD Event_Handler.
* Abarbeitung des Ereignisses
ENDMETHOD.
ENDCLASS.
```

- Je nach Art des Ereignisses (System- oder Applikationsereignis) müssen Sie jetzt noch die Methode **CL\_GUI\_CFW=>DISPATCH** aufrufen. Lesen Sie dazu [Ereignisbehandlung \[Extern\]](#).

## Ereignisse des SAP Toolbar Controls

### Verwendung

Durch Auswahl von Drucktasten des SAP Toolbar Controls werden je nach Drucktastenart Ereignisse ausgelöst:

Ereignis	Ereignis-ID cl_gui_toolbar=>	Bedeutung
FUNCTION_SELECTE D	M_ID_FUNCTION_SELECTE D	Drucktaste wurde gedrückt Menü zu einer Drucktaste vom Typ <code>cntb_btype_dropdown</code> und <code>cntb_btype_menu</code> wurde ausgewählt.
DROPDOWN_CLICKE D	M_ID_DROPDOWN_CLICKE D	Anforderung eines Kontextmenüs zu einer Drucktaste vom Typ <code>cntb_btype_dropdown</code> und <code>cntb_btype_menu</code> .

Zu dem Ereignis erhalten Sie folgende Ereignisparameter:

Ereignis	Parameter	Bedeutung
FUNCTION_SELECTED	fcode	Funktionscode der Taste
DROPDOWN_CLICKED	fcode	Funktionscode der Taste
	posx posy	Position der Drucktaste, für die das Menü angezeigt werden soll

### Integration

Wenn Sie auf Ereignisse in Ihrem ABAP-Programm reagieren möchten, müssen Sie sich auf diese Ereignisse registrieren. Dazu verwenden Sie die Methode [set\\_registered\\_events \[Seite 47\]](#). Ereignisse, auf die Sie sich nicht registrieren, werden schon am Frontend gefiltert und gelangen nicht zum Backend. [Siehe Ereignisbehandlung \[Extern\]](#).

### Funktionsumfang

#### Das Ereignis FUNCTION\_SELECTED

Dieses Ereignis wird immer dann ausgelöst, wenn eine Drucktaste gedrückt wurde oder ein Menüeintrag einer Drucktaste vom Typ `cntb_btype_dropdown` und `cntb_btype_menu` ausgewählt wurde.

Drucktasten vom Typ `cntb_btype_dropdown` bestehen aus zwei Teilen. Auf der linken Seite befindet sich die eigentliche Drucktaste, mit der eine vordefinierte Funktion verknüpft wurde (analog zu einer normalen Drucktaste). Wird diese Taste gedrückt, wird das Ereignis `FUNCTION_SELECTED` ausgelöst. Auf der rechten Seite ist eine Drucktaste mit einem Pfeil zu sehen. Wird diese angeklickt, wird das Ereignis `DROPDOWN_CLICKED` ausgelöst.

## Ereignisse des SAP Toolbar Controls

Bei Drucktasten vom Typ `cntb_btype_menu` wird immer das Ereignis `DROPDOWN_CLICKED` ausgelöst.

Als Ereignisparameter des Ereignisses `FUNCTION_SELECTED` erhalten Sie den Funktionscode der Taste. Über diesen Funktionscode können Sie die Taste identifizieren und entsprechend reagieren.

### Das Ereignis `DROPDOWN_CLICKED`

Dieses Ereignis wird immer dann ausgelöst, wenn zu einer Drucktaste ein Menü angezeigt werden kann. Dies ist nur bei Drucktasten vom Typ `cntb_btype_dropdown` und `cntb_btype_menu` möglich.

Als Ereignisparameter erhalten Sie den Funktionscode der Taste und die Position der Drucktaste. Über den Funktionscode können Sie die Taste identifizieren. Die Position können Sie dazu verwenden, das Kontextmenü an der entsprechenden Stelle zu platzieren.

Das Kontextmenü bauen Sie mit den Methoden der Klasse `CL_CTMENU` auf. Zur Anzeige bringen Sie es mit der Methode [track\\_context\\_menu \[Seite 34\]](#).

## Aktivitäten

Lesen Sie den allgemeinen [Prozeß \[Seite 11\]](#) für das Arbeiten mit Ereignissen des Control Frameworks.

## Verwendung von Controls im WAN

Die Verwendung von Controls zur Oberflächengestaltung führt in der Regel zu einer Belastung des Kommunikationskanals zwischen Frontend und Backend. Dies kann schon im LAN-, aber insbesondere im WAN-Umfeld ein performance-kritischer Aspekt sein.

Puffermechanismen helfen, diese Problematik zu entschärfen (siehe auch [Automation Queue \[Extern\]](#)). Die aufgeführten Punkte sollen als Richtlinien bei der Verwendung von Controls im WAN dienen.

Control-spezifische Hinweise zur Verwendung im WAN finden Sie in der Dokumentation zu dem jeweiligen Control.

### Verwendung von CL\_GUI\_CFW=>FLUSH

Der Aufruf [CL\\_GUI\\_CFW=>FLUSH \[Seite 38\]](#) dient zum Synchronisieren der Automation Queue und der in der Queue enthaltenen ABAP-Variablen. Dieser Aufruf erzeugt in vielen Fällen einen synchronen RFC vom Applikationsserver zum Frontend. Um optimale Performance zu erreichen, sollten die Aufrufe dieser Methode minimiert werden.

Es ist in vielen Fällen zu empfehlen, alle Eigenschaften von allen Controls zentral an einer Stelle (z.B. am Anfang des PAI) in einer Automation Queue zu lesen und dann über einen einmalige Synchronisation zu besorgen. Diese Variante ist auch dann zu bevorzugen, wenn dabei Eigenschaften gelesen werden, die nicht immer für den Ablauf des Ereignisbehandlers bzw. des PAI – PBO Zyklus notwendig sind.

Es ist nicht notwendig, einen „Sicherheits-Flush“ am Ende von PBO zu codieren, damit Methodenaufrufe der Controls garantiert an das Frontend transportiert werden. Diese Funktionalität wird systemseitig garantiert, wenn das nächste Dynpro gesendet wird. Damit ist es auch nicht möglich, eine Automation Queue über mehrere Bildwechsel hinweg aufzubauen.

**Es ist nicht garantiert, daß eine Automation Queue durch den Aufruf CL\_GUI\_CFW=>FLUSH gesendet wird. Die Queue erkennt, ob Returnwerte enthalten sind. Ist dies nicht der Fall, wird das Senden unterdrückt!**

Für alle Fälle, in denen auch bei einer Queue ohne Returnwert gewünscht wird, daß die Automation Queue synchron versendet wird, gibt es im Control Framework die Methode [CL\\_GUI\\_CFW=>UPDATE\\_VIEW \[Seite 41\]](#). Diese Methode darf nur dann verwendet werden, wenn es zwingend notwendig ist, ein Update des GUI zu erreichen. Beispiele hierfür sind sehr lange laufende Anwendungen, die in regelmäßigen Abständen dem Benutzer ein Feedback über den Fortschritt der Aktion anzeigen möchten.

Nach dem Lesen von Eigenschaften ist der Inhalt der entsprechenden ABAP-Variablen erst nach dem nächsten FLUSH garantiert. Solange dieser Aufruf nicht erfolgt ist, ist der Inhalt der entsprechenden ABAP-Variablen nicht definiert. In Zukunft wird es Fälle gegeben, in denen dieser FLUSH unnötig sein wird. Diese Fälle werden von der Automation Queue erkannt; der entsprechende FLUSH-Call wird dann ignoriert.

### Erzeugen von Controls, Datenversorgung

Das Erzeugen eines Controls und die Datenversorgung ist in den meisten Fällen ein einmaliger Vorgang und im Vergleich zu Dynproelementen sehr teuer. Deshalb sollten Controls nicht unnötig erzeugt bzw. nicht unnötig mit Daten versorgt werden.

Ein typisches Beispiel hierfür sind TabStrips mit mehreren Seiten. Wenn diese Seiten Controls tragen, ist immer abzuwägen, ob man auf lokale Seiten verzichtet und die Controls erst dann

## Verwendung von Controls im WAN

erzeugt, wenn der Benutzer die Seite aktiviert. Das gleiche trifft für die Datenversorgung dieser Controls auf TabStrip-Seiten zu.

Muß bei der Datenversorgung eine Unterscheidung zwischen einer WAN- und einer LAN-Anmeldung vorgenommen werden, steht der Funktionsbaustein `SAPGUI_GET_WANFLAG` zur Verfügung. In manchen Fällen kann es notwendig werden, daß eine Anwendung andere Datenmengen oder ganze Fallbacks für die WAN-Anmeldung zur Verfügung stellen sollte. Ein Beispiel, wann die WAN- bzw. LAN- Anmeldung einen Einfluß haben kann, ist die Anzahl von Geschwistern in einem Tree Control, die ohne künstliche Zwischenebenen übertragen werden können.

Im Gegensatz zu Dynproelementen werden die Controls nur einmalig erzeugt und mit Daten versorgt. Controls werden unter Performance-Aspekten dann immer preiswerter, je länger diese leben. In Anwendungen, die ständig neu aufgerufen und damit neu initialisiert werden, kann dies zu einem erheblichen Performance-Nachteil werden; in Anwendungen, die sehr lange auf den gleichen Bildern arbeiten, kann daraus sogar ein Performance-Vorteil entstehen.

Im Einzelfall kann über entsprechende [Performance-Werkzeuge \[Extern\]](#) überprüft werden, welche Nachteile oder Vorteile die Verwendung eines Controls unter dem Aspekt der Netzwerkauslastung bringt.

## Ablegen von Dokumenten, Bildern etc.

Zum Release 4.6A wird ein Frontend-Cache für Zugriffe auf Dokumente aus dem BDS (Business Dokument Service) realisiert. Es wird dringend empfohlen, Office-Dokumente, Bilder etc. im BDS und nicht in der R/3-Datenbank abzulegen. Dokumente aus dem BDS können danach im Frontend-Cache abgelegt werden. Sie müssen nur einmalig über das Netz geladen werden.

## Spezielle Hinweise für das SAP Toolbar Control

Für das SAP Toolbar Control bestehen keine Einschränkungen beim Einsatz im WAN.

---

**Methoden der Klasse CL\_GUI\_TOOLBAR****Methoden der Klasse CL\_GUI\_TOOLBAR**

Diese Klasse beinhaltet sowohl control-spezifische Methoden als auch vom Control Framework geerbte Methoden. In diesem Abschnitt werden nur die control-spezifischen Methoden beschrieben. Die relevanten Methoden des Control Frameworks werden in [Methoden des OO Control Frameworks \[Seite 35\]](#) beschrieben.

## constructor

Diese Methode wird für die Instanzierung eines SAP Toolbar Controls verwendet.

CREATE OBJECT toolbar

```
EXPORTING parent      = parent
          shellstyle  = shellstyle
          lifetime    = lifetime
          display_mode = display_mode
EXCEPTIONS cntl_install_error = 1
           cntl_error         = 2.
```

Parameter	Bedeutung
lifetime	<p>Parameter für das <a href="#">Lifetime Management [Extern]</a>. Folgende Werte sind möglich:</p> <p><b>toolbar-&gt;lifetime_imode:</b> Das Control lebt, solange der interne Modus nicht abgebaut wird (d.h. durch eine der folgenden Anweisungen beendet wird: <code>leave program.</code> <code>leave to transaction.</code> <code>set screen 0,</code> <code>leave screen.</code>). Danach wird die Methode <a href="#">finalize [Seite 46]</a> aufgerufen.</p> <p><b>toolbar-&gt;lifetime_dynpro:</b> Das Control lebt, solange die Instanz des Dynpros existiert, d.h. sich noch im Dynprostapel befindet. Danach wird die Methode <a href="#">free [Seite 44]</a> aufgerufen. Die Benutzung dieses Modus regelt automatisch die Sichtbarkeit von Controls. Controls werden immer nur dann eingeblendet, wenn das Dynpro aktiv ist, auf dem sie erzeugt wurden. Ist ein anderes Dynpro aktiv, werden sie automatisch unsichtbar geschaltet.</p> <p><b>toolbar-&gt;lifetime_default:</b> Wird das Control in einen Container eingebaut, übernimmt es die Lebensdauer des Containers. Wird es nicht in einen Container eingebaut (z.B. es ist selbst ein Container), dann wird die Lebensdauer auf <code>toolbar-&gt;lifetime_imode</code> gesetzt.</p>
shellstyle	<p>Steuerung des Erscheinungsbilds und des Verhaltens des Controls</p> <p>Konstanten aus dem ABAP-Include <code>&lt;CTLDEF&gt;</code>, die mit <code>WS</code> beginnen, können Sie übergeben. Kombinationen von mehreren Styles können Sie durch Addieren der Konstanten erreichen. Der Vorschlagswert führt intern zum Setzen einer ausreichenden Kombination von Style-Konstanten.</p>
parent	<p>Container, in dem das SAP Toolbar Control angezeigt werden kann (<a href="#">siehe SAP Container [Extern]</a>)</p>
display_mode	<p>Ausrichtung der Drucktastenleiste:</p> <ul style="list-style-type: none"> <li>• <code>cl_gui_toolbar=&gt;m_mode_horizontal</code>: Horizontal</li> <li>• <code>cl_gui_toolbar=&gt;m_mode_vertical</code>: Vertikal</li> </ul>

---

constructor

## add\_button

Mit dieser Methode fügen Sie eine neue Drucktaste zur Drucktastenleiste hinzu.

```
CALL METHOD toolbar->add_button
    EXPORTING fcode      = fcode
             icon       = iconid
             is_disabled = is_disabled
             butn_type  = butn_type
             text       = text
             quickinfo  = quickinfo
             is_checked = is_checked
    EXCEPTIONS cntl_error = 1.
```

Parameter	Bedeutung
fcode	Funktionscode, der als Ereignisparameter beim Auslösen der hinzuzufügenden Drucktaste an Ihr Applikationsprogramm übergeben wird
icon	Ikone, die auf der Drucktaste angezeigt werden soll
is_disabled	'x': Drucktaste ist inaktiv ' ': Drucktaste ist aktiv
butn_type	<b>cntb_btype_button</b> : Drucktaste <b>cntb_btype_dropdown</b> : Drucktaste mit Menü <b>cntb_btype_menu</b> : Menü <b>cntb_btype_sep</b> : Trennlinien/-zeichen <b>cntb_btype_group</b> : Drucktastengruppe <b>cntb_btype_check</b> : Toggle-Button  <b>Hinweis</b> : Wenn immer möglich, sollten Sie <b>statische Menüs</b> verwenden, da dadurch weitläufige Roundtrips vermieden werden. Dies ist besonders im SAP GUI for HTML wichtig. Weitere Informationen finden Sie unter <a href="#">set_static_ctxmenu [Seite 30]</a> oder <a href="#">assign_static_ctxmenu_table [Seite 32]</a> .
text	Text, der auf der Drucktaste angezeigt werden soll
quickinfo	Quick-Info für die Drucktaste
is_checked	Nur für Drucktasten vom Typ <b>cntb_btype_group</b> und <b>cntb_btype_check</b> : 'x': Taste ist ausgewählt ' ': Taste ist nicht ausgewählt



Die Ikone kann über ihren Namen (z.B. **ICON\_ANNOTATION**) angesprochen werden. Dazu muß aber in Ihrem Programm das Include **<ICON>** eingebunden werden. Eine Übersicht über alle verfügbaren Ikonen liefert der Report **SHOWICON**.

**add\_button**

Ansonsten sprechen Sie die Ikonen über @xy@ an, wobei xy die entsprechende Buchstabenkombination darstellt.

## add\_button\_group

Mit dieser Methode können Sie eine Liste von Drucktasten zu Ihrer Drucktastenleiste hinzufügen.

```
CALL METHOD toolbar->add_button_group
    EXPORTING data_table = data_table
    EXCEPTIONS dp_error = 1.
```

Parameter	Bedeutung
data_table	Tabelle aller Drucktasten, die zu der Drucktastenleiste hinzugefügt werden sollen. Die Tabelle wird mit Bezug auf den Typ <code>TTB_BUTTON</code> angelegt. Sie können die Tabelle mit der Methode <a href="#">fill_buttons_data_table [Seite 24]</a> füllen.



Wenn immer möglich, sollten Sie **statische Menüs** verwenden, da dadurch weitläufige Roundtrips vermieden werden. Dies ist besonders im SAP GUI for HTML wichtig. Weitere Informationen finden Sie unter [set\\_static\\_ctxmenu \[Seite 30\]](#) oder [assign\\_static\\_ctxmenu\\_table \[Seite 32\]](#).

## fill\_buttons\_data\_table

## fill\_buttons\_data\_table

Diese Methode stellt eine Hilfsfunktion zum Füllen einer Tabelle dar, die Sie dann der Methode [add\\_button\\_group \[Seite 23\]](#) zum Anlegen mehrerer Drucktasten übergeben.

```
CALL METHOD toolbar->fill_buttons_data_table
    EXPORTING fcode      = fcode
             icon       = iconid
             disabled   = disabled
             butn_type  = butn_type
             text       = text
             quickinfo  = quickinfo
             checked    = checked
    CHANGING data_table = data_table.
```

Parameter	Bedeutung
fcode	Funktionscode, der als Ereignisparameter beim Auslösen der hinzuzufügenden Drucktaste an Ihr Applikationsprogramm übergeben wird
icon	Ikone, die auf der Drucktaste angezeigt werden soll
disabled	'X': Drucktaste ist inaktiv ' ': Drucktaste ist aktiv
butn_type	<b>cntb_btype_button</b> : Drucktaste <b>cntb_btype_dropdown</b> : Drucktaste mit Menü <b>cntb_btype_menu</b> : Menü <b>cntb_btype_sep</b> : Trennlinien/-zeichen <b>cntb_btype_group</b> : Drucktastengruppe <b>cntb_btype_check</b> : Toggle-Button  <b>Hinweis</b> : Wenn immer möglich, sollten Sie <b>statische Menüs</b> verwenden, da dadurch weitläufige Roundtrips vermieden werden. Dies ist besonders im SAP GUI for HTML wichtig. Weitere Informationen finden Sie unter <a href="#">set_static_ctxmenu [Seite 30]</a> oder <a href="#">assign_static_ctxmenu_table [Seite 32]</a> .
text	Text, der auf der Drucktaste angezeigt werden soll
quickinfo	Quick-Info für die Drucktaste
checked	Nur für Drucktasten vom Typ <b>cntb_btype_group</b> und <b>cntb_btype_check</b> : 'X': Taste ist ausgewählt ' ': Taste ist nicht ausgewählt
data_table	Tabelle aller Drucktasten, die zu der Drucktastenleiste hinzugefügt werden sollen. Die Tabelle wird mit Bezug auf den Typ <b>TTB_BUTTON</b> angelegt. Diese Tabelle übergeben Sie dann der Methode <a href="#">add_button_group [Seite 23]</a> .



**fill\_buttons\_data\_table**

Die Ikone kann über ihren Namen (z.B. `ICON_ANNOTATION`) angesprochen werden. Dazu muß aber in Ihrem Programm das Include `<ICON>` eingebunden werden. Eine Übersicht über alle verfügbaren Ikonen liefert der Report `SHOWICON`.

Ansonsten sprechen Sie die Ikonen über `@xy@` an, wobei `xy` die entsprechende Buchstabenkombination darstellt.

delete\_button

## delete\_button

Mit dieser Methode löschen Sie eine Drucktaste aus der Druckstastenleiste.

```
CALL METHOD toolbar->DELETE_BUTTON  
    exporting fcode      = fcode  
    exceptions CNTL_ERROR = 1.
```

Parameter	Bedeutung
fcode	Funktionscode, der als Ereignisparameter beim Auslösen der zu löschenden Drucktaste an Ihr Applikationsprogramm übergeben wird



Wurden mehrere Drucktasten mit dem gleichen Funktionscode angelegt, wird nur die letzte Taste aus der Druckstastenleiste entfernt.

## delete\_all\_buttons

Mit dieser Methode löschen Sie alle Drucktasten aus der Drucktastenleiste.

```
CALL METHOD toolbar->DELETE_ALL_BUTTONS  
    exceptions CNTL_ERROR = 1.
```

**set\_button\_state****set\_button\_state**

Mit dieser Methode können Sie den Status einzelner Drucktasten verändern:

```
CALL METHOD toolbar->set_button_state
    EXPORTING  enabled = enabled
              checked = checked
              fcode   = fcode
    EXCEPTIONS cntl_error = 1.
```

Parameter	Bedeutung
<b>enabled</b>	<ul style="list-style-type: none"> <li>' ': Drucktaste ist inaktiv</li> <li>'x': Drucktaste ist aktiv</li> </ul>
<b>is_checked</b>	Nur für Drucktasten vom Typ <code>cntb_btype_group</code> und <code>cntb_btype_check</code> : <ul style="list-style-type: none"> <li>'x': Taste ist ausgewählt</li> <li>' ': Taste ist nicht ausgewählt</li> </ul>

## set\_button\_info

Mit dieser Methode ändern Sie den Text, die Ikone oder die Quick-Info für eine Drucktaste.

```
CALL METHOD toolbar->set_button_info
    EXPORTING fcode = fcode
             icon  = icon
             text  = text
             quickinfo = quickinfo.
```

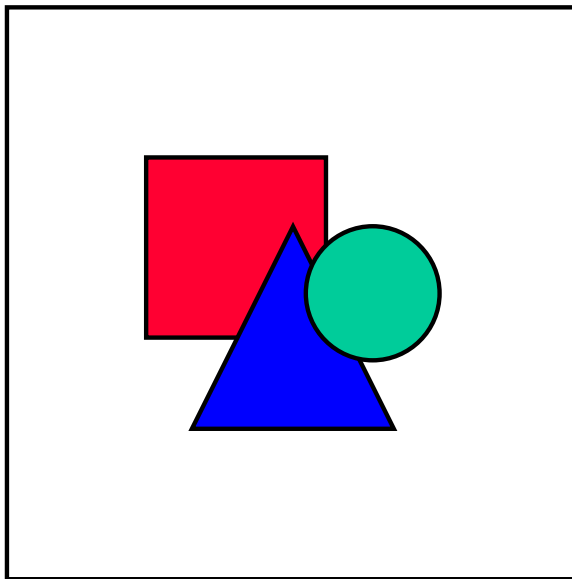
Parameter und Typ	Optional	Bedeutung
fcode TYPE UI_FUNC		Funktionscode der Drucktaste (zur Identifizierung der Drucktaste - darf nicht geändert werden)
icon TYPE ICOPNNAME	X	Neue Ikone der Form '@XY@'
text TYPE TEXT40	X	Neuer Text für die Drucktaste
quickinfo TYPE ICONQUICK	X	Neuer Quick-Info-Text für die Drucktaste

`set_static_ctxmenu`

## set\_static\_ctxmenu

Mit dieser Methode ordnen Sie einer Drucktaste für die gesamte Lebensdauer des Controls ein Kontextmenü zu. Wenn Sie ein Kontextmenü statisch zuordnen, wird es in das Frontend-Control geladen und bleibt dort auch, wenn es vom Benutzer geschlossen wurde. Normalerweise wird das Ereignis `DROPDOWN_CLICKED` ausgelöst, wenn der Benutzer eine Drucktaste mit Dropdown-Menü anklickt. In Ihrer Anwendung würden Sie dann der Drucktaste das Kontextmenü zuordnen, aber nur für diesen Fall. Wenn Sie das Kontextmenü statisch zuordnen, bleibt es auf dem Frontend und steht dann immer zur Verfügung, wenn der Benutzer die entsprechende Drucktaste anklickt. Sie müssen es also nicht jedes Mal erneut zuordnen.

**Sie sollten statische Kontextmenüs außer in besonders kontextabhängigen Fällen immer verwenden.**



Alle Änderungen, die am Kontextmenü während der Lebensdauer des Controls vorgenommen werden, werden automatisch am Frontend nachgezogen.

```
CALL METHOD toolbar->set_static_ctxmenu
      EXPORTING   fcode = fcode
                ctxmenu = ctxmenu
```

Parameter und Typ	Bedeutung
fcode TYPE UI_FUNC	Funktionscode der Drucktaste, der Sie das Kontextmenü zuordnen möchten  Hinweis: Die Drucktaste muß vom Typ <code>cntb_btype_dropdown</code> oder <code>cntb_btype_menu</code> sein.
ctxmenu TYPE REF TO CL_CTMENU	Referenzvariable, die auf die Kontextmenü-Instanz zeigt, die Sie zuordnen möchten (siehe <a href="#">Kontextmenüs [Extern]</a> )

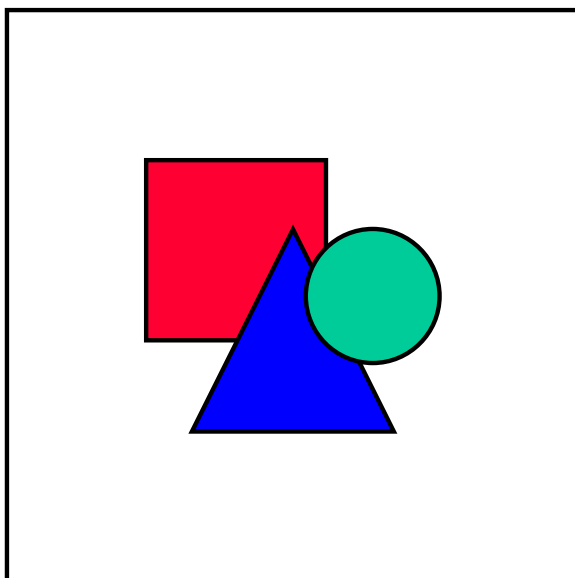


## assign\_static\_ctxmenu\_table

## assign\_static\_ctxmenu\_table

Mit dieser Methode ordnen Sie einer Drucktastengruppe für die gesamte Lebensdauer des Controls Kontextmenüs zu. Wenn Sie ein Kontextmenü statisch zuordnen, wird es in das Frontend-Control geladen und bleibt dort auch, wenn es vom Benutzer geschlossen wurde. Normalerweise wird das Ereignis DROPDOWN\_CLICKED ausgelöst, wenn der Benutzer eine Drucktaste mit Dropdown-Menü anklickt. In Ihrer Anwendung würden Sie dann der Drucktaste das Kontextmenü zuordnen, aber nur für diesen Fall. Wenn Sie die Kontextmenüs statisch zuordnen, bleiben sie auf dem Frontend und stehen dann immer zur Verfügung, wenn der Benutzer die entsprechende Drucktaste anklickt. Sie müssen sie also nicht jedes Mal erneut zuordnen.

**Sie sollten statische Kontextmenüs außer in besonders kontextabhängigen Fällen immer verwenden.**



Alle Änderungen, die am Kontextmenü während der Lebensdauer des Controls vorgenommen werden, werden automatisch am Frontend nachgezogen.

```
CALL METHOD toolbar->assign_static_ctxmenu_table
      EXPORTING table_ctxmenu = table_ctxmenu.
```

Parameter und Typ	Bedeutung
table_ctxmenu TYPE TTB_BTNMNU	Interne Tabelle mit den Zuordnungen der Kontextmenüs zu den Drucktasten in der Toolbar-Instanz. Sie hat den Zeilentyp STB_BTNMNU (siehe unten).

### Struktur STB\_BTNMNU

Komponente und Typ	Bedeutung
Function TYPE UI_FUNC	Funktionscode der Drucktaste in der Drucktastenleiste, der Sie das Kontextmenü zuordnen möchten

**assign\_static\_ctxmenu\_table**

ctmenu TYPE REF TO CL_CTMENU	Referenzvariable, die auf das Kontextmenü zeigt, das Sie der Drucktaste zuordnen möchten (siehe <a href="#">Kontextmenüs [Extern]</a> )
------------------------------------	---

`track_context_menu`

## track\_context\_menu

Mit dieser Methode können Sie ein Kontextmenü anzeigen. Diese Methode macht vor allem in Verbindung mit Ereignissen der Drucktasten vom Typ `cntb_btype_dropdown` und `cntb_btype_menu` Sinn.

```
CALL METHOD toolbar->track_context_menu
```

```
    EXPORTING context_menu = menu
```

```
             posx         = posx
```

```
             posy         = posy
```

```
    EXCEPTIONS ctmenu_error = 1.
```

Parameter	Bedeutung
<code>context_menu</code>	Name des Kontextmenüs. Das Menü wird mit Referenz auf die Klasse <code>CL_CTMENU</code> angelegt. Sie müssen das Menü mit Methoden dieser Klasse füllen.
<code>posx</code>	Horizontale Position des angezeigten Menüs
<code>posy</code>	Vertikale Position des angezeigten Menüs

## Methoden des OO Control Frameworks

In diesem Abschnitt werden relevante Methoden des OO Control Frameworks beschrieben, die Sie bei der Implementierung des SAP Toolbar Controls benötigen.

---

**Methoden der Klasse CL\_GUI\_CFW****Methoden der Klasse CL\_GUI\_CFW**

Die Klasse `CL_GUI_CFW` beinhaltet statische Methoden, die beim Aufruf auf **alle** instanziierten Custom Controls wirken.

## dispatch

Die Methode `dispatch` verteilt Applikationsereignisse ([siehe Ereignisbehandlung \[Extern\]](#)) an die für das Ereignis angemeldeten Ereignisbehandler. Wenn diese Methode nicht im Applikationsprogramm innerhalb von PAI aufgerufen wird, dann wird sie automatisch vom System nach dem Abarbeiten von PAI aufgerufen. Die Methode liefert einen Returncode zurück, über den der Erfolg des Aufrufs abzulesen ist.

```
CALL METHOD cl_gui_cfw=>dispatch
      IMPORTING return_code = return_code.
```

Parameter	Bedeutung
<code>return_code</code>	<p><code>cl_gui_cfw=&gt;rc_found</code>: Das Ereignis konnte erfolgreich an eine Behandlungsmethode übergeben werden.</p> <p><code>cl_gui_cfw=&gt;rc_unknown</code>: Das Ereignis wurde nicht in der Ereignisliste registriert.</p> <p><code>cl_gui_cfw=&gt;rc_noevent</code>: Es wurde kein Ereignis auf einem Control ausgelöst. Der <code>OK_CODE</code> war daher ein normaler <code>OK_CODE</code> (z.B. von einem Menüeintrag).</p> <p><code>cl_gui_cfw=&gt;rc_nodispatch</code>: Dem Ereignis konnte keine Behandlungsmethode zugeordnet werden.</p>



Das Ereignis kann nur einmalig verteilt werden. Danach ist das Ereignis verbraucht. Daher wird ein zweiter Aufruf der Methode nicht nochmals zu einem Sprung in den Ereignisbehandler führen.

flush

## flush

Mit dieser Methode synchronisieren Sie explizit die [Automation Queue \[Extern\]](#). Die gepufferten Operationen werden dann zum Frontend per GUI-RFC geschickt. Dort wird die Automation Queue in der Reihenfolge abgearbeitet, wie Sie sie gefüllt haben.

Im Fehlerfall wird eine Ausnahme ausgelöst, die Sie auf jeden Fall abfragen und behandeln sollten. Da eine Zuordnung des Fehlers in der Regel nicht mehr möglich ist, stehen Ihnen sowohl im Debugger als auch im SAP GUI Werkzeuge zur Verfügung, um den Fehler zu lokalisieren:

**Debugger:** Markieren Sie in den Einstellungen das *Ankreuzfeld Automation Controller: Aufträge immer synchron verarbeiten*. Dies führt dazu, daß nach jeder Methode, die den Automation Controller ruft, die Methode `c1_gui_cfw=>flush` automatisch aufgerufen wird.

**SAP GUI:** In den Einstellungen zum SAP GUI können Sie auf der Karteikarte *Trace* das Ankreuzfeld *Automation* wählen. Dadurch wird die Kommunikation zwischen Applikationsserver und Automation Controller in einer Trace-Datei mitgeschrieben. Diese kann dann ausgewertet werden.

```
CALL METHOD c1_gui_cfw=>flush
          EXCEPTIONS CNTL_SYSTEM_ERROR = 1
                   CNTL_ERROR = 2.
```



Führen Sie nur so viele Synchronisationspunkte in Ihr Programm ein, wie wirklich nötig sind. Bei jeder Synchronisation wird nämlich eine RFC-Verbindung zum SAP GUI geöffnet.

## get\_living\_dynpro\_controls

Mit dieser Methode können Sie sich eine Liste von Referenzvariablen zu allen noch aktiven Custom Controls besorgen.

```
CALL METHOD cl_gui_cfw=>get_living_dynpro_controls  
IMPORTING control_list = control_list.
```

Parameter	Bedeutung
control_list	Liste der Referenzvariablen zu aktiven Custom Controls. Liste ist vom Typ CNTO_CONTROL_LIST (in der Klasse CL_GUI_CFW definiert)

`set_new_ok_code`

## set\_new\_ok\_code

Diese Methode darf nur in Behandlermethoden zu Systemereignissen eingesetzt werden. Sie setzt einen `OK_CODE`, der ein Ausführen von `PAI` nach sich zieht. Dadurch können Sie nach dem Feldtransport nochmals die Kontrolle in Ihren `PAI`-Modulen bekommen.

```
CALL METHOD cl_gui_cfw=>set_new_ok_code
      EXPORTING new_code = new_code
      IMPORTING   rc = rc.
```

Parameter	Bedeutung
<code>new_code</code>	Funktionscode, der in das <code>OK_CODE</code> -Feld ( <code>SY-UCOMM</code> ) gestellt werden soll.
<code>return_code</code>	<p><code>cl_gui_cfw=&gt;rc_posted</code>: Der <code>OK_CODE</code> wurde mit Erfolg gesetzt, und die Verarbeitung wird nach Abschluß der Behandlermethode mit <code>PAI</code> fortgesetzt (vorher wird noch die automatische Feldprüfung des Dynpros durchgeführt).</p> <p><code>cl_gui_cfw=&gt;rc_wrong_state</code>: Die Methode wurde nicht bei einem Systemereignis aufgerufen.</p> <p><code>cl_gui_cfw=&gt;rc_invalid</code>: Der gesetzte <code>OK_CODE</code> ist kein erlaubter <code>OK_CODE</code>.</p>

## update\_view

Die Automation Queue wird durch den Aufruf der Methode [flush \[Seite 38\]](#) nur dann synchronisiert, wenn in der Automation Queue Returnwerte enthalten sind.

Für alle Fälle, in denen auch im Fall einer Returnwert-freien Queue gewünscht wird, daß die Automation Queue synchron versendet wird, gibt es im Control Framework die Methode `CL_GUI_CFW=>UPDATE_VIEW`. Diese Methode darf nur dann verwendet werden, wenn es zwingend notwendig ist, ein Update des SAP GUI zu erreichen. Beispiele hierfür sind sehr lange laufende Anwendungen, die in regelmäßigen Abständen dem Benutzer ein Feedback über den Fortschritt der Aktion anzeigen möchten.

```
CALL METHOD cl_gui_cfw=>update_view
      EXCEPTIONS CNTL_SYSTEM_ERROR = 1
                CNTL_ERROR         = 2.
```

---

**Methoden der Klasse CL\_GUI\_OBJECT****Methoden der Klasse CL\_GUI\_OBJECT**

Die Klasse `CL_GUI_OBJECT` beinhaltet wichtige Methoden zum Verschalen von Custom Controls. Für Anwendungsprogramme ist einzig die Methode [is\\_valid \[Seite 43\]](#) relevant.

## is\_valid

Diese Methode liefert als Ergebnis, ob ein Custom Control zu einer Objektreferenz noch am Frontend vorhanden ist.

```
CALL METHOD my_control->is_valid  
IMPORTING result = result.
```

Parameter	Bedeutung
result	0: Custom Control ist nicht mehr am Frontend aktiv 1: Custom Control ist noch aktiv

free

## free

Diese Methode baut ein Custom Control am Frontend ab. Nach Aufruf dieser Methode sollten Sie auch die Objektreferenz initialisieren (**FREE my\_control**).

```
CALL METHOD my_control->free
      EXCEPTIONS cntl_error      = 1
                 cntl_system_error = 2.
```

## Methoden der Klasse CL\_GUI\_CONTROL

Die Klasse `CL_GUI_CONTROL` beinhaltet Methoden, die zum Setzen von Control-Eigenschaften (z.B. Visualisieren des Controls), Registrieren von Ereignissen und zum Abbau des Controls dienen.

---

finalize

## finalize

Diese Methode wird von der verwendeten Controlverschaltung überdefiniert. In ihr werden control-spezifische Funktionen zum Abbau des Controls aufgerufen. Diese Methode wird automatisch von der Methode [free \[Seite 44\]](#) aufgerufen, bevor das Control am Frontend abgebaut wird.

```
CALL METHOD my_control->finalize.
```

## set\_registered\_events

Mit dieser Methode registrieren Sie sich auf Ereignisse des Controls ([siehe auch: Ereignisbehandlung \[Extern\]](#)).

```
CALL METHOD my_control->set_registered_events
    EXPORTING events          = events
    EXCEPTIONS cntl_error    = 1
               cntl_system_error = 2
               illegal_event_combination = 3.
```

Parameter	Bedeutung
events	Tabelle der zu registrierenden Ereignisse für das Custom Control <code>my_control</code>

Die Tabelle `events` ist eine Liste mit Ereignissen, auf die Sie sich registrieren wollen. Die Tabelle wird mit Bezug auf den Tabellentyp `CNTL_SIMPLE_EVENTS` definiert. Dem Tabellentyp liegt die Struktur `CNTL_SIMPLE_EVENT` zugrunde. Dieser besteht aus folgenden Feldern:

Feld	Bedeutung
EVENTID	Name des Ereignisses
APPL_EVENT	Unterscheidung, ob es sich um ein Systemereignis (initial) oder ein Applikationsereignis (x) handeln soll.

Die Werte, die dem Feld `EVENTID` zuzuordnen sind, sind control-spezifisch und werden daher bei den entsprechenden Controls beschrieben.

**get\_registered\_events****get\_registered\_events**

Diese Methode liefert eine Liste aller für das Custom Control `my_control` registrierten Ereignisse zurück.

```
CALL METHOD my_control->get_registered_events
  IMPORTING events = events
  EXCEPTIONS cntl_error = 1.
```

Parameter	Bedeutung
<code>events</code>	Tabelle der zu registrierenden Ereignisse für das Custom Control <code>my_control</code>

Die Tabelle `events` ist eine Liste mit Ereignissen, auf die Sie sich registriert haben. Die Tabelle wird mit Bezug auf den Tabellentyp `CNTL_SIMPLE_EVENTS` definiert. Dem Tabellentyp liegt die Struktur `CNTL_SIMPLE_EVENT` zugrunde. Dieser besteht aus folgenden Feldern:

Feld	Bedeutung
<code>EVENTID</code>	Name des Ereignisses
<code>APPL_EVENT</code>	Unterscheidung, ob es sich um ein Systemereignis (initial) oder ein Applikationsereignis (x) handeln soll.

Die Werte, die dem Feld `EVENTID` zuzuordnen sind, sind control-spezifisch und werden daher bei den entsprechenden Controls beschrieben.



Allgemeine Informationen zur Ereignisbehandlung finden Sie in der Dokumentation des SAP Control Frameworks unter [Ereignisbehandlung \[Extern\]](#).

## is\_alive

Diese Methode liefert als Ergebnis, ob ein Custom Control zu einer Objektreferenz noch am Frontend vorhanden ist.

```
CALL METHOD my_control->is_alive  
RETURNING state = state.
```

Parameter	Bedeutung
state	<code>my_control-&gt;state_dead</code> : Custom Control ist nicht mehr am Frontend aktiv <code>my_control-&gt;state_alive</code> : Custom Control ist auf aktuellem Dynpro aktiv <code>my_control-&gt;state_alive_on_other_dynpro</code> : Custom Control ist auf dem aktuellen Dynpro nicht aktiv, aber am Frontend noch aktiv (d.h. unsichtbar)

**set\_alignment****set\_alignment**

Diese Methode richtet das Custom Control innerhalb seines Containers aus:

```
CALL METHOD my_control->set_alignment
    EXPORTING alignment      = alignment
    EXCEPTIONS cntl_error    = 1
               cntl_system_error = 2.
```

Parameter	Bedeutung
alignment	Ausrichtung des Controls

Der Parameter **alignment** kann aus Kombinationen folgender Ausrichtungen bestehen:

Name	Bedeutung
my_control->align_at_left	Ausrichtung am linken Rand
my_control->align_at_right	Ausrichtung am rechten Rand
my_control->align_at_top	Ausrichtung am oberen Rand
my_control->align_at_bottom	Ausrichtung am unteren Rand

Kombinationen erhält man durch Aufaddieren der Komponenten:

alignment = my\_control->align\_at\_left + my\_control->align\_at\_top.

## set\_position

Diese Methode plaziert das Control an eine bestimmte Stelle des Dynpros.



In der Regel wird die Position des Controls über seinen Container geregelt.

```
CALL METHOD my_control->set_position
  EXPORTING height      = height
            left        = left
            top         = top
            width       = width
  EXCEPTIONS cntl_error      = 1
            cntl_system_error = 2.
```

Parameter	Bedeutung
height	Höhe des Controls
left	Linker Rand des Controls
top	Oberer Rand des Controls
width	Breite des Controls

**set\_visible****set\_visible**

Mit dieser Methode können Sie die Sichtbarkeit eines Custom Controls verändern.

```
CALL METHOD my_control->set_visible
  EXPORTING visible          = visible
  EXCEPTIONS cntl_error     = 1
             cntl_system_error = 2.
```

Parameter	Bedeutung
visible	x: Custom Control ist sichtbar : Custom Control ist nicht sichtbar

## get\_focus

Diese statische Methode liefert die Objektreferenz des Custom Controls zurück, welches den Fokus hat.

```
CALL METHOD cl_gui_control=>get_focus
IMPORTING control          = control
EXCEPTIONS cntl_error     = 1
            cntl_system_error = 2.
```

Parameter	Bedeutung
control	Objektreferenz (TYPE REF TO cl_gui_control) auf das Control, das den Fokus hat

**set\_focus****set\_focus**

Mit dieser statischen Methode können Sie den Fokus auf ein Custom Control setzen.

```
CALL METHOD cl_gui_control=>set_focus
    EXPORTING control          = control
    EXCEPTIONS cntl_error     = 1
               cntl_system_error = 2.
```

Parameter	Bedeutung
control	Objektreferenz (TYPE REF TO cl_gui_control) auf das Control, das den Fokus bekommen soll

## get\_height

Diese Methode liefert die Höhe des Controls zurück.

```
CALL METHOD control->get_height
  IMPORTING height          = height
  EXCEPTIONS cntl_error    = 1.
```

Parameter	Bedeutung
height	Aktuelle Höhe des Controls

get\_width

## get\_width

Diese Methode liefert die Breite des Controls zurück.

```
CALL METHOD control->get_width
    IMPORTING width          = width
    EXCEPTIONS cntl_error   = 1.
```

Parameter	Bedeutung
width	Aktuelle Breite des Controls