

# SAP HTML Viewer (BC-CI)



**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 HTML Viewer (BC-CI)</b> .....               | <b>6</b> |
| Instanz für den SAP HTML Viewer .....              | 9        |
| Arbeiten mit dem SAP HTML Viewer .....             | 10       |
| Registrieren und Verarbeiten von Ereignissen ..... | 12       |
| navigate_complete .....                            | 14       |
| sapevent .....                                     | 15       |
| Verwendung von Controls im WAN .....               | 17       |
| Spezielle Hinweise für den SAP HTML Viewer .....   | 19       |
| Methoden der Klasse CL_GUI_HTML_VIEWER .....       | 20       |
| constructor .....                                  | 21       |
| show_url .....                                     | 23       |
| stop .....   | 24       |
| go_back .....                                      | 25       |
| go_forward .....                                   | 26       |
| go_home .....                                      | 27       |
| do_refresh .....                                   | 28       |
| get_current_url .....                              | 29       |
| load_html_document.....                            | 30       |
| load_mime_object .....                             | 31       |
| load_data .....                                    | 32       |
| show_data .....                                    | 33       |
| show_url_in_browser.....                           | 34       |
| Methoden des OO Control Frameworks.....            | 35       |
| Methoden der Klasse CL_GUI_CFW.....                | 36       |
| dispatch .....                                     | 37       |
| flush .....  | 38       |
| get_living_dynpro_controls.....                    | 39       |
| set_new_ok_code.....                               | 40       |
| update_view .....                                  | 41       |
| Methoden der Klasse CL_GUI_OBJECT.....             | 42       |
| is_valid.....                                      | 43       |
| is_alive.....                                      | 44       |
| Methoden der Klasse CL_GUI_CONTROL .....           | 45       |
| free.....  | 46       |
| finalize.....                                      | 47       |
| set_registered_events .....                        | 48       |
| get_registered_events .....                        | 49       |
| set_alignment .....                                | 50       |
| set_position .....                                 | 51       |
| set_visible.....                                   | 52       |
| get_focus .....                                    | 53       |
| set_focus .....                                    | 54       |
| get_height.....                                    | 55       |
| get_width .....                                    | 56       |

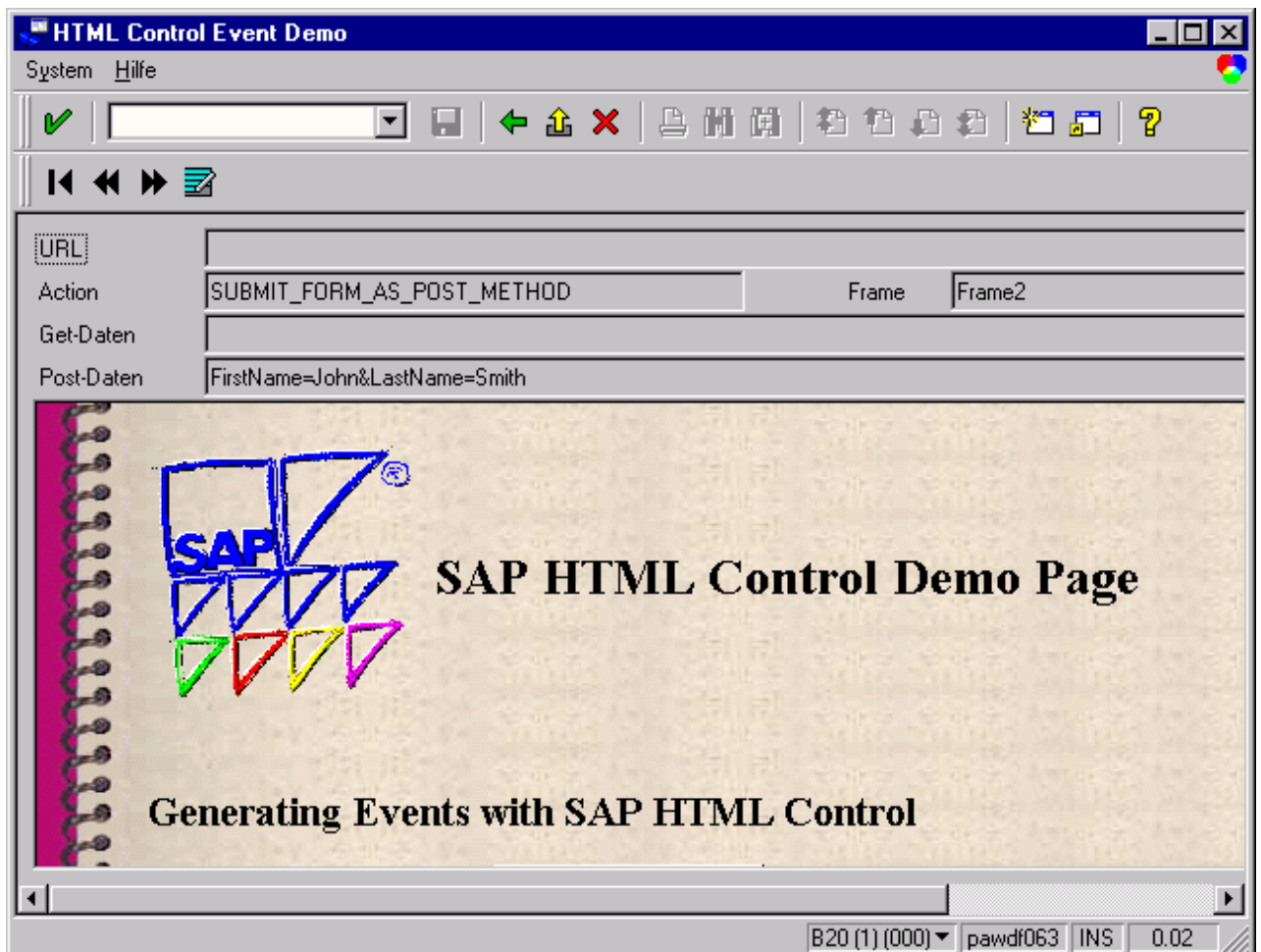


## SAP HTML Viewer (BC-CI)

### Einsatzmöglichkeiten

Der SAP HTML Viewer ist ein von SAP entwickeltes Control für den Einsatz innerhalb des SAP GUI. SAP hat keinen eigenen Browser für das World Wide Web entwickelt. Statt dessen benutzt der SAP HTML Viewer das Microsoft Internet Explorer Web Browser Control. Unterstützt wird der MS Internet Explorer ab Version 4.0.

Die folgende Abbildung zeigt den SAP HTML Viewer in einem R/3-Bildschirmbild:



Das Bild ist das Ergebnis des Beispielprogramms SAPHTML\_EVENTS\_DEMO.

### Einführungshinweise

Zwar ist die Schnittstelle des SAP HTML Viewers für alle Plattformen identisch, doch sind die verfügbaren Funktionen vom verwendeten HTML-Browser abhängig. Im SAP GUI for Windows wird der Internet Explorer 4.0 von Microsoft als Fremd-HTML-Browser verwendet.

## Integration

Der SAP HTML Viewer kann zusammen mit anderen Controls, z.B. Tasten und Listboxen, auf einem Bild einer R/3-Transaktion verwendet werden. Mit diesem Control können ABAP-Entwickler in ihren Transaktionen in der Datenbank gespeicherte oder zur Laufzeit erzeugte HTML-Seiten, -Graphiken oder Bilder darstellen.

## Funktionsumfang

Der SAP HTML Viewer kann nicht nur von einer R/3-Transaktion bereitgestellte Daten darstellen, sondern auch auf Eingaben eines Anwenders reagieren, indem Ereignisse ausgelöst werden. Auf diese können Sie dann innerhalb des Anwendungsprogramms reagieren (siehe [Registrieren und Verarbeiten von Ereignissen \[Seite 12\]](#)). Die Ereignisse werden ausgelöst, wenn der Anwender eine URL wählt oder ein HTML-Dokument absendet. Die gewählte URL und die Dokumenteingabedaten stehen dann für die Verarbeitung mit dem ABAP-Code zur Verfügung.

Der SAP HTML Viewer unterstützt außerdem die Navigation durch HTML-Seiten und MIME-Dokumente aus Datenquellen außerhalb der laufenden R/3-Transaktion, z.B. aus einem Desktop-Dateisystem oder einem HTTP-Server, die eine enge Einbindung eines SAP GUI 4.6 in die Internet-/Intranet-Umgebung eines Kunden ermöglichen.

## Einschränkungen

### SAP GUI for HTML

Der SAP HTML Viewer ist nur auf Frontend-Plattformen einsetzbar, auf denen Microsoft Internet Explorer mit der Version 4.0 oder höher bereits installiert ist.

Die Installation des IE wird zur Zeit **nicht** von der SAP-GUI-Installation durchgeführt. Sie benötigen eine SAP-GUI-Installation 4.6 oder höher.

Einige Funktionen des SAP HTML Viewers sind im SAP GUI for HTML **nicht verfügbar** (oder zeigen ein anderes Verhalten). Die folgenden Komponenten sind davon betroffen:

| Methoden                                   | Ereignisse                                   |
|--|--|
| <a href="#">go_back [Seite 25]</a>         | <a href="#">sapevent [Seite 15]</a>          |
| <a href="#">go_forward [Seite 26]</a>      | <a href="#">navigate_complete [Seite 14]</a> |
| <a href="#">go_home [Seite 27]</a>         |  |
| <a href="#">do_refresh [Seite 28]</a>      |  |
| <a href="#">stop [Seite 24]</a>            |  |
| <a href="#">get_current_url [Seite 29]</a> |  |

### SAP GUI for Java

Einige Funktionen dieses Controls können unter Umständen im SAP GUI for Java nicht ausgeführt werden. Andere können nur mit Einschränkungen benutzt werden. Weitere Informationen hierzu finden Sie in der mit dem SAP GUI for Java ausgelieferten Dokumentation.



## Instanz für den SAP HTML Viewer

### Definition

Diese Instanz wird mit Referenz auf die Klasse `c1_gui_html_viewer` definiert:

```
data html_viewer type ref to c1_gui_html_viewer.
```

### Verwendung

Eine Instanz für den SAP HTML Viewer verwaltet alle Informationen bezüglich eines HTML-Browsers in Ihrem Dynpro. Auf diese Instanz können Sie Methoden aufrufen, mit denen Sie die Eigenschaften des SAP HTML Viewers definieren und ändern können.

### Integration

Die Klasse `c1_gui_html_viewer` beinhaltet sowohl [control-spezifische Methoden \[Seite 20\]](#) als auch [Methoden des OO Control Frameworks \[Extern\]](#).

## Arbeiten mit dem SAP HTML Viewer

In diesem Abschnitt werden die für den SAP HTML Viewer spezifischen Funktionen aufgeführt.

### Voraussetzungen

Der in diesem Abschnitt beschriebene Prozeß stellt nur eine control-spezifische Ergänzung des [allgemeinen Prozesses zur Control-Einbindung \[Extern\]](#) 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.

### Instanzieren

1. Definieren Sie eine Referenzvariable für den SAP HTML Viewer:

```
DATA html_viewer TYPE REF TO cl_gui_html_viewer.
```

2. [Erzeugen Sie eine Instanz \[Seite 21\]](#) des SAP HTML Viewers:

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

### Ereignis anmelden

3. Melden Sie Ereignisse für den SAP HTML Viewer an. Folgende Ereignisse unterstützt der SAP HTML Viewer:

| Ereignisname      | Bedeutung  |
|-------------------|--|
| SAPEVENT          | Klick auf einen speziellen HTML-Link, der ein SAP-Ereignis auslöst |
| NAVIGATE_COMPLETE | HTML-Seite wurde komplett geladen                                  |

### Arbeiten mit dem Control

4. Laden Sie [HTML Seiten \[Seite 30\]](#) oder [Teilobjekte \[Seite 31\]](#) von HTML Seiten:
5. Bringen Sie die geladenen Objekte zur [Anzeige \[Seite 23\]](#).
6. Navigieren Sie aus Ihrem Anwendungsprogramm im Navigationsstapel des HTML-Browsers [vor \[Seite 26\]](#) und [zurück \[Seite 25\]](#), oder verzweigen Sie zur [Startseite \[Seite 27\]](#).
7. [Aktualisieren \[Seite 28\]](#) Sie aus Ihrem Anwendungsprogramm die angezeigte HTML-Seite.
8. Fragen Sie die [Adresse \[Seite 29\]](#) der aktuell angezeigten Seite ab.

### 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:

**Arbeiten mit dem SAP HTML Viewer**

9. [Bauen \[Seite 46\]](#) Sie den SAP HTML Viewer am Frontend ab. Sofern Sie den Control Container nicht mehr benötigen, bauen Sie auch diesen ab:

```
CALL METHOD html_viewer->free.
```

10. Löschen Sie die Referenzvariablen des SAP HTML Viewers und eventuell des Control Containers:

```
FREE html_viewer.
```



Als Beispiel für das Einbinden eines HTML Viewers können Sie den Report `SAPHTML_EVENTS_DEMO` heranziehen.

## Registrieren und Verarbeiten von Ereignissen

# 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 Ereignisbehandlermethode 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.
```

6. 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

7. 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.
```

8. 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

9. 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.
```

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

navigate\_complete

## navigate\_complete

### Verwendung



Aufgrund der Control-Framework-Architektur wird dieses Ereignis unter Umständen nicht immer vom Frontend zurück an den Anwendungsserver übertragen. **Daher sollten Sie die Ablauflogik Ihres Programms nie vom Empfang dieses Ereignisses abhängig machen.**

Das Ereignis `navigate_complete` wird gesendet, wenn das Laden einer HTML-Seite abgeschlossen ist.

| Ereignis          | Ereignis-ID            | Bedeutung                         |
|-------------------|------------------------|-----------------------------------|
|                   | html_viewer->          |                                   |
| navigate_complete | m_id_navigate_complete | HTML-Seite wurde komplett geladen |

### Funktionsumfang

Das Ereignis liefert dabei folgende Parameter mit:

| Schnittstellenparameter | Bedeutung                                  |
|-------------------------|--|
| url                     | URL-Adresse der geladenen Seite im Browser |

### Aktivitäten

Lesen Sie den allgemeinen [Prozeß \[Seite 12\]](#) für das Arbeiten mit Ereignissen des Control Frameworks.

## sapevent

### Verwendung

Dieses Ereignis wird gesendet, wenn auf der HTML-Seite Ereignisse vom Typ **SAPEVENT** definiert wurden und diese vom Benutzer ausgelöst wurden.

| Ereignis | Ereignis-ID   | Bedeutung  |
|----------|---------------|--|
|          | html_viewer-> |  |
| sapevent | m_id_sapevent | Der Benutzer hat einen HTML-Link vom Typ <b>SAPEVENT</b> angeklickt. |

### Voraussetzungen

Bei der Definition der anzuzeigenden HTML-Seite müssen Sie spezielle HTML-Ereignisse (**SAPEVENT**) definieren. Ein HTML-Ereignis, das dieses Control-Ereignis auslösen soll, muß immer folgendes Format haben:

**SAPEVENT:<action>?data.**

### Verwendung von SAPEVENT im SAP GUI for HTML

- Damit dieses Ereignis im HTML-GUI korrekt ausgelöst wird, müssen Sie es in einer der folgenden Formen programmieren:
  - <FORM name=myform action="SAPEVENT:xxx">
  - <A HREF=SAPEVENT:xxx>

Wenn Sie das Ereignis auf andere Art programmieren (z.B. innerhalb eines Skripts), **wird es nicht erkannt** und daher **nicht an den Anwendungsserver zurückgegeben**.
- Verwenden Sie SAPEVENT außerdem **niemals** in einem OnLoad-Ereignis, da dies zu einer Endlosschleife führt. Nach einem Ereignis lädt der Internet Transaction Server die entsprechende Seite neu (wobei ein neues OnLoad-Ereignis ausgelöst wird). Daher würde jedes SAPEVENT implizit zu einem neuen OnLoad führen, der wiederum ein weiteres SAPEVENT auslöst.

### Funktionsumfang

Das Ereignis liefert dabei folgende Parameter mit:

| Parameter   | Bedeutung   |
|-------------|---|
| action      | Aktionsparameter des <b>SAPEVENT</b>                    |
| frame       | Name des HTML-Frames, auf dem das Event ausgelöst wurde |
| getdata     | Daten des Ereignisstrings                               |
| postdata    | Datentabelle der Post-Daten                             |
| query_table | Datentabelle in der Form 'Name   Wert'                  |



## sapevent

HTML-Ereignisse können zusätzlich zu einer Aktion auch Daten verschicken. Wenn Sie die 'POST'-Methode im HTML-Quelltext verwenden, werden die Daten nicht mit dem Ereignisstring, sondern in einem parallelen Datenstrom mitgeschickt. Diese Daten stehen in der Tabelle `postdata` zur Verfügung. Die Tabelle `query_table` stellt die Umwandlung der Tabelle `postdata` in Wertepaare dar. Die Daten müssen folgendes Format haben: **Param1=Value1&Param2=Value2**.

## Aktivitäten

Lesen Sie den allgemeinen [Prozeß \[Seite 12\]](#) 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 den SAP HTML Viewer

Folgende Aspekte sind beim Einsatz des SAP HTML Viewers zu beachten :

1. Hintergrundbilder und Grafiken, die nur zur Verschönerung der Seite dienen, sollten im WAN nur dann verwendet werden, wenn diese als GIF oder JPEG vorhanden sind.
2. Unnötige Ereignisse (**SAPEVENT**) sollten vermieden werden.
3. Grafiken, die zur Informationsbeschaffung dienen und unverzichtbar sind, sollten über ein Ereignis (**SAPEVENT**) bei Bedarf geladen werden, wenn der Benutzer dieses Ereignis auslöst.
4. Das Speichern der HTML Dokumente sollte nach Möglichkeit auf einen HTTP-Server ausgelagert werden, damit die Daten zur Laufzeit unabhängig vom Applikationsserver asynchron geladen werden können und damit auch den Cache des Internet Explorers nutzen.

---

**Methoden der Klasse CL\_GUI\_HTML\_VIEWER****Methoden der Klasse CL\_GUI\_HTML\_VIEWER**

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 vererbten Methoden des Control Frameworks werden in [Methoden des OO Control Frameworks \[Seite 35\]](#) beschrieben.

## constructor

Diese Methode wird für die Instanzierung des SAP HTML Viewers verwendet.

```
CREATE OBJECT html_viewer
  EXPORTING shellstyle      = shellstyle
           parent          = parent
           saphtmlp        = saphtmlp
           lifetime         = lifetime
  EXCEPTIONS cntl_error    = 1
             cntl_install_error = 2
             dp_install_error  = 3
             dp_error         = 4.
```

| Parameter  | Bedeutung   |
|------------|---|
| lifetime   | <p>Parameter für das <a href="#">Lifetime Management [Extern]</a>. Folgende Werte sind möglich:</p> <p><b>html_viewer-&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 47]</a> aufgerufen.</p> <p><b>html_viewer-&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 46]</a> aufgerufen.</p> <p>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>html_viewer-&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. weil es selbst ein Container ist), dann wird die Lebensdauer auf <code>html_viewer-&gt;lifetime_imode</code> gesetzt.</p> |
| saphtmlp   | <p>'X': Aufbereitung der HTML-Seiten für Scriptlets und Applets</p> <p>': Normale Aufbereitung der HTML-Seiten</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 der SAP HTML Viewer angezeigt werden kann (<a href="#">siehe SAP Container [Extern]</a>)</p>   |

---

constructor

## show\_url

Mit dieser Methode zeigen Sie Daten im SAP HTML Viewer an.



Die Unterscheidung zwischen dieser Methode und der Methode `show_data` ist weggefallen. Ab Release 4.6C sollten Sie **immer diese Methode verwenden**, um Daten im HTML Viewer anzuzeigen, unabhängig davon, aus welcher Quelle sie stammen.

```
CALL METHOD html_viewer->show_url
  EXPORTING url          = url
           frame        = frame
           in_place     = in_place
  EXCEPTIONS cntl_error = 1.
```

| Parameter | Opt. | Bedeutung  |
|-----------|------|--|
| url       |      | URL-Adresse der anzuzeigenden Seite  |
| frame     | X    | Name des HTML-Frames, in dem die Seite angezeigt werden soll   |
| in_place  |      | Gibt an, wo die Seite angezeigt werden soll. Mögliche Werte: <ul style="list-style-type: none"> <li>'x' (Standard): Seite wird im SAP-GUI-Fenster angezeigt.</li> <li>' ': Seite wird in einem eigenen Browser-Fenster angezeigt.</li> </ul> |

stop

## stop

Mit dieser Methode beenden Sie das Laden einer HTML-Seite.

```
CALL METHOD html_viewer->stop  
EXCEPTIONS cntl_error = 1.
```



Sämtliche Verwendungen dieser Methode in Programmen, die unter dem SAP GUI for HTML laufen, werden **ignoriert**, da die Methode in dieser Umgebung nicht sauber ausgeführt werden kann.

## go\_back

Mit dieser Methode können Sie im Navigationsstapel des HTML-Browsers eine Seite zurückgehen.

```
CALL METHOD html_viewer->go_back
```

```
EXCEPTIONS cntl_error = 1.
```



Sämtliche Verwendungen dieser Methode in Programmen, die unter dem SAP GUI for HTML laufen, werden **ignoriert**, da die Methode in dieser Umgebung nicht sauber ausgeführt werden kann.

go\_forward

## go\_forward

Mit dieser Methode können Sie im Navigationsstapel des HTML-Browsers eine Seite weitergehen.

```
CALL METHOD html_viewer->go_forward
```

```
EXCEPTIONS cntl_error = 1.
```



Sämtliche Verwendungen dieser Methode in Programmen, die unter dem SAP GUI for HTML laufen, werden **ignoriert**, da die Methode in dieser Umgebung nicht sauber ausgeführt werden kann.

## go\_home

Nach Aufruf dieser Methode wird die Homepage des installierten HTML-Browsers angezeigt.

```
CALL METHOD html_viewer->go_home
```

```
EXCEPTIONS cntl_error = 1.
```



Sämtliche Verwendungen dieser Methode in Programmen, die unter dem SAP GUI for HTML laufen, werden **ignoriert**, da die Methode in dieser Umgebung nicht sauber ausgeführt werden kann.

do\_refresh

## do\_refresh

Mit dieser Methode können Sie aus dem ABAP-Programm heraus die aktuell angezeigte HTML-Seite erneut laden.

```
CALL METHOD html_viewer->do_refresh  
EXCEPTIONS cntl_error = 1.
```



Sämtliche Verwendungen dieser Methode in Programmen, die unter dem SAP GUI for HTML laufen, werden **ignoriert**, da die Methode in dieser Umgebung nicht sauber ausgeführt werden kann.

## get\_current\_url

Mit dieser Methode können Sie die URL-Adresse des aktuell angezeigten HTML-Dokuments ermitteln.

```
CALL METHOD html_viewer->get_current_url  
    IMPORTING url = url  
    EXCEPTIONS cntl_error = 1.
```

| Parameter | Bedeutung                                    |
|-----------|--|
| url       | URL-Adresse der angezeigten Seite im Browser |



Sämtliche Verwendungen dieser Methode in Programmen, die unter dem SAP GUI for HTML laufen, werden **ignoriert**, da die Methode in dieser Umgebung nicht sauber ausgeführt werden kann.

## load\_html\_document

## load\_html\_document

Diese Methode lädt ein HTML-Dokument aus dem SAP-Web-Repository. Das SAP-Web-Repository wird mit der Transaktion SMW0 gepflegt.

```
CALL METHOD html_viewer->load_html_document
  EXPORTING document_id      = document_id
            document_textpool = document_textpool
            document_url     = document_url
            as_compressed_data =
  IMPORTING assigned_url    = assigned_url
  CHANGING  merge_table    = merge_table
  EXCEPTIONS document_not_found = 1
            dp_error_general   = 2
            dp_invalid_parameter = 3.
```

| Parameter          | Bedeutung   |
|--------------------|---|
| document_id        | ID des Dokuments im SAP-Web-Repository  |
| document_textpool  | Name des Programms, in dem der Textpool zum Abmischen mit dem Dokument abgelegt ist. Textelemente des HTML-Dokuments (Format: <code>&lt;!text-xxx!&gt;</code> , wobei <code>xxx</code> die ID des Textes im Textpool ist) werden automatisch durch Texte des Textpools ersetzt.   |
| document_url       | URL, die dem Dokument zugewiesen wird. Sofern keine URL angegeben wird, wird eine eindeutige URL vom System vergeben.   |
| assigned_url       | Wenn im Parameter <code>document_url</code> keine URL zugewiesen wurde, steht in diesem Parameter eine vom System vergebene URL. Ansonsten entspricht die URL der vergebenen URL im Parameter <code>document_url</code> .   |
| merge_table        | Tabelle mit Inhalten, die in das aktuelle HTML-Dokument eingemischt werden sollen.  |
| as_compressed_data | Der HTML Viewer komprimiert die Daten standardmäßig, bevor sie an das Frontend-Control übertragen werden. In einigen Fällen (z.B. nach einer Umwandlung von SAPscript- in HTML-Format) kann dies zu Problemen führen. Sie können diese Funktion daher an- und ausschalten: <ul style="list-style-type: none"> <li>• 'x': Daten komprimieren</li> <li>• ' ': Daten nicht komprimieren</li> </ul> |

## load\_mime\_object

Diese Methode lädt ein Objekt beliebigen Typs aus dem SAP-Web-Repository und schickt es an den SAP HTML Viewer. Das SAP-Web-Repository wird mit der Transaktion SMW0 gepflegt.



Alle SAP-Ikonen sind bereits in der Transaktion **SMW0** gepflegt. Sie finden diese, indem Sie nach binären Daten in der Entwicklungsklasse **SWWW** suchen.

Einen Überblick über alle SAP-Ikonen liefert der Report **SHOWICON**.

```
CALL METHOD html_viewer->load_mime_object
EXPORTING object_id = object_id
          object_url = object_url
IMPORTING assigned_url = assigned_url
EXCEPTIONS object_not_found = 1
            dp_error_general = 2
            dp_invalid_parameter = 3.
```

| Parameter    | Bedeutung   |
|--------------|---|
| object_id    | ID des Dokuments im SAP-Web-Repository  |
| object_url   | URL, die dem Dokument zugewiesen wird. Sofern keine URL angegeben wird, wird eine eindeutige URL vom System vergeben.   |
| assigned_url | Wenn im Parameter <code>document_url</code> keine URL zugewiesen wurde, steht in diesem Parameter eine vom System vergebene URL. Ansonsten entspricht die URL der vergebenen URL im Parameter <code>document_url</code> . |

## load\_data

## load\_data

Diese Methode sendet Daten aus dem ABAP-Programm an den Präsentationsrechner. Die Daten sind danach unter der angegebenen URL-Adresse verfügbar.

```
CALL METHOD html_viewer->load_data
  EXPORTING url          = url
           type          = type
           subtype       = subtype
           size          = size
  IMPORTING assigned_url = assigned_url
  CHANGING data_table   = data_table
  EXCEPTIONS dp_invalid_parameter = 1
             dp_error_general    = 2.
```

| Parameter    | Bedeutung   |
|--------------|---|
| url          | URL, mit der auf die Daten zugegriffen werden soll  |
| type         | Typ der Daten in Form eines MIME-Typs (z.B. text)   |
| subtype      | Subtyp der Daten in Form eines MIME-Typs (z.B. html)  |
| size         | Größe der Daten in Byte   |
| assigned_url | Wenn im Parameter <code>url</code> keine URL zugewiesen wurde, steht in diesem Parameter eine vom System vergebene URL. Ansonsten entspricht die URL der vergebenen URL im Parameter <code>url</code> . |
| data_table   | Tabelle mit den Daten   |

## show\_data



Diese Methode ist ab Release 4.6C veraltet. Sie sollten jetzt die Methode [show\\_url \[Seite 23\]](#) verwenden, um Daten im HTML Viewer anzuzeigen, unabhängig davon, aus welcher Quelle sie stammen. Verwendungen von `show_data` in Programmen aus früheren Releases sind dennoch weiterhin gültig und müssen nicht geändert werden.

Die Methode zeigt im Control die Daten an, die vorher mit den Methoden [load\\_mime\\_object \[Seite 31\]](#), [load\\_data \[Seite 32\]](#) und [load\\_html\\_document \[Seite 30\]](#) an das Frontend gesendet wurden. Sofern Sie die Daten nicht mit diesen Methoden an das Frontend gesendet haben, benutzen Sie die Methode [show\\_url \[Seite 23\]](#) zur Anzeige.

```
CALL METHOD html_viewer->show_data
  EXPORTING url    = url
           frame  = frame
  EXCEPTIONS cntl_error = 1.
```

| Parameter | Bedeutung  |
|-----------|--|
| url       | URL-Adresse des Dokuments                                    |
| frame     | Name des HTML-Frames, in dem die Seite angezeigt werden soll |

**show\_url\_in\_browser**

## show\_url\_in\_browser

Mit dieser Methode zeigen Sie eine URL in einem eigenen Browser-Fenster an.



Diese Methode ist ab Release 4.6C veraltet. Sie sollten jetzt die Methode [show\\_url](#) [\[Seite 23\]](#) verwenden, um Daten im HTML Viewer anzuzeigen, unabhängig davon, aus welcher Quelle sie stammen. Verwendungen von **show\_url\_in\_browser** in Programmen aus früheren Releases sind dennoch weiterhin gültig und müssen nicht geändert werden.

```
CALL METHOD html_viewer->show_url_in_browser  
EXPORTING URL = URL.
```

| Parameter und Typ | Opt. | Bedeutung  |
|-------------------|------|--|
| URL<br>TYPE C     |      | URL-Adresse der Seite, die Sie im Browser anzeigen möchten |

## Methoden des OO Control Frameworks

In diesem Abschnitt werden relevante Methoden des OO Control Frameworks beschrieben, die Sie bei der Implementierung des SAP HTML Viewers 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.<br>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<br>1: Custom Control ist noch aktiv |

is\_alive

## 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     | <p><code>my_control-&gt;state_dead</code>: Custom Control ist nicht mehr am Frontend aktiv</p> <p><code>my_control-&gt;state_alive</code>: Custom Control ist auf aktuellem Dynpro aktiv</p> <p><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)</p> |

## 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.

---

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.
```

## 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 \[Seite 46\]](#) aufgerufen, bevor das Control am Frontend abgebaut wird.

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

**set\_registered\_events****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  |
|---------------------|--|
| <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 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  |
|-------------------------|--|
| <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.

## 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 \[Extern\]](#).

**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<br>· : 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 |