



"Best Practices", Visualisierung

Version: 1.0
Template: templ_tecdoc_de_V2.0.docx
Dateiname: Best Practices Visualization DE.docx

INHALT

	Seite
1 Ratschläge	3
1.1 Statische Elemente im Hintergrund halten	3
1.2 Häufig geänderte Elemente so klein wie möglich entwerfen	3
1.3 Selektion verdeckter Elemente	3
1.4 Viele unsichtbare Elemente vermeiden	3
1.5 Visualisierung mit Titelzeile und Menü	3
1.6 Visualisierungsobjekte zu Funktionsbausteinen – Strukturierung der Hauptseiten in Unterbereiche	4
1.7 Visualisierungszustände vs. Instanzzustände	4
1.8 Visualisierungsstile	4
2 Typische Anwendungsfälle	5
2.1 Dynamische Bildumschaltung	5
2.2 Dynamische Textumschaltung	5
2.3 Modus-Schalter mit verschiedenen Stufen	6
2.4 Dynamisches Wechseln zu verschiedenen Visualisierungen aus einer Hauptseite heraus	6
2.5 Anpassung der Visualisierungsdialoge	6
Änderungshistorie	7

1 Ratschläge

1.1 Statische Elemente im Hintergrund halten

Die Visualisierung kann Elemente ohne Verknüpfung mit Variablen am besten optimieren, wenn diese in den Visualisierungen hinter den Elementen mit Variablenverknüpfungen liegen (wo möglich).

Die Optimierungen resultieren in besserer Performance und geringerem Speicherbedarf der Visualisierung.

1.2 Häufig geänderte Elemente so klein wie möglich entwerfen

Ein großes Rechteckelement, in dem sich häufig ein kleiner Text ändert, führt zu einer schlechteren Performance als ein von der Größe her besser angepasstes Rechteck.

1.3 Selektion verdeckter Elemente

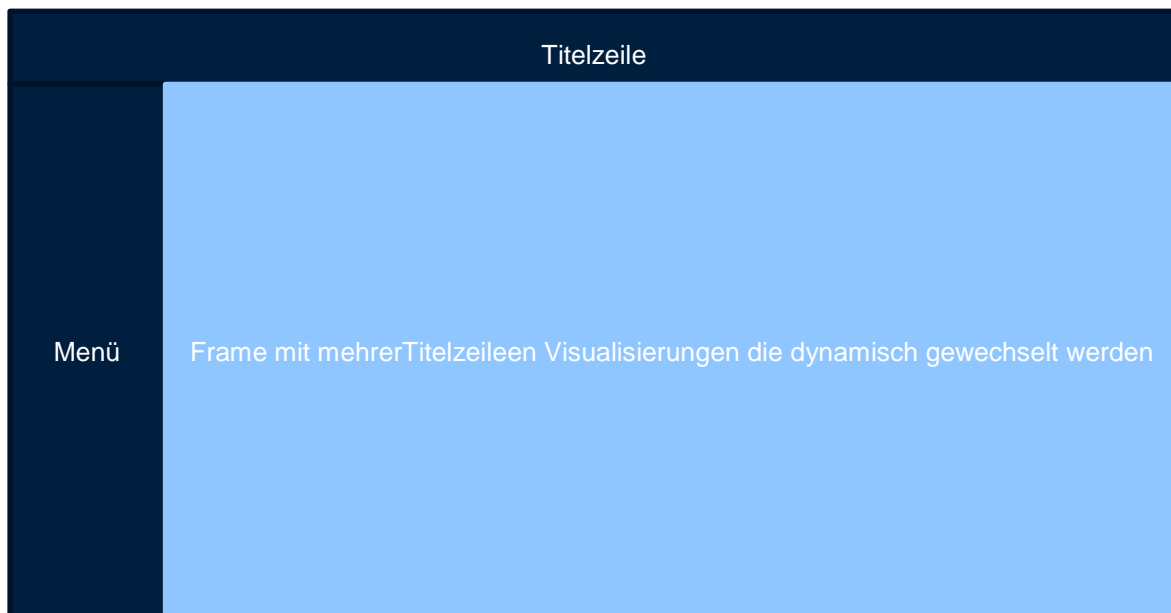
Mittels <Strg>+<Shift>+Klick lassen sich reihum alle Elemente an einer Mausposition selektieren.

1.4 Viele unsichtbare Elemente vermeiden

Die Performance der Visualisierung ist besser, wenn eine Gruppe bzw. ein Frame als unsichtbar konfiguriert ist, als wenn die Einzelelemente unsichtbar gesetzt sind. Wenn viele einzelne unsichtbare Elemente vorliegen, kann also durch eine sinnvolle Gruppierung die Performance möglicherweise verbessert werden.

1.5 Visualisierung mit Titelzeile und Menü

Für die Performance ist es besser, wenn nur ein Teil des gesamten Anzeigebereichs neu gezeichnet werden muss. Es ist besser, die Inhalte in einem Frame zu wechseln, als die ganze Visualisierung zu wechseln. Also nicht mit Copy/Paste Titelzeile und Menü in viele Hauptvisualisierungen bringen, sondern eine Einstiegsseite (mit Titelzeile und Menü) mit Unterseiten versehen, die über den Frame umgeschaltet werden.



1.6 Visualisierungsobjekte zu Funktionsbausteinen – Strukturierung der Hauptseiten in Unterbereiche

Eine Visualisierung bietet einen Schnittstelleneditor. Dieser dient dazu, an eine Visualisierung Parameter zu übergeben. Dadurch lassen sich Visualisierungsobjekte erstellen, die mehrfach über das Frame-Element verwendet und in andere Visualisierungen eingebunden werden können.

Für die Schnittstellenparameter gilt:

- Es ist besser, einen Parameter (z.B. eine Funktionsbausteininstanz) zu übergeben, als mehrere skalare Parameter.
- Hierbei sollte VAR_IN_OUT verwendet werden, da dadurch die Instanz nicht kopiert wird.

Eine elegante Form der Eingabeverarbeitung ist es, in der Eingabekonfiguration in einem Event (z.B. OnMouseClicked) „ST-Code ausführen“ zu verwenden, und im ST-Code eine Methode oder eine Eigenschaft der übergebenen Funktionsbausteininstanz aufzurufen. Das erleichtert das Debuggen und verbessert die Struktur.

1.7 Visualisierungszustände vs. Instanzzustände

Falls die Visualisierung wie empfohlen strukturiert ist (Zuordnung FB->Visu über Interface/VAR_IN_OUT, siehe Kapitel 1.6), sollten Variablen, die den Zustand des Objekts (z.B. Betriebsmode operational/starting-up/in error) enthalten, im FB angelegt sein.

Nur Variablen, die rein den Zustand der Visualisierung beinhalten, sollten mit VAR in der Visualisierung selbst deklariert werden.

Beispiel: Es soll die Selektion in einer selbst gezeichneten Liste umgesetzt werden. Annahme: Die Liste enthält zehn Zellen mit Rechteck-Elementen und einen Text, der aus dem Baustein pou kommt.

```
VAR_IN_OUT
    pouIn : POUList;
END_VAR

VAR
    arrSelection : ARRAY[VISU_MIN_NUMBER_OF_CLIENTS..VISU_MAX_NUMBER_OF_CLIENTS] OF
    INT;
END_VAR
```

Die Konstanten VISU_MIN_NUMBER_OF_CLIENT/VISU_MAX_NUMBER_OF_CLIENTS stehen ab CODESYS Version V3.5 SP10 zur Verfügung.

Die Zellen sollten nun wie folgt konfiguriert sein, wobei der INDEX durch den jeweiligen Index der Zelle ersetzt werden muss.

Texte – Text:%s

Textvariablen – Textvariable: pouIn.arr[INDEX] → Die Zeichenfolge, die in der Zelle angezeigt werden soll.

Farbvariablen – Farbumschlag: arrSelection[CurrentClientId] = INDEX

Eingabekonfiguration – OnMouseEnter – ST-Code ausführen: arrSelection[CurrentClientId] := INDEX;

Hinweis: Zusätzlich muss noch die Bibliothek VisuGlobalClientManager eingebunden werden, damit man mit der Variablen „CurrentClientId“ arbeiten kann.

In diesem Beispiel wird gezeigt, wie man mit clientabhängigen Daten direkt in der Visualisierung umgehen kann.

1.8 Visualisierungsstile

Der Visualisierungsstil, mit dem das Projekt gestaltet wird, sollte am besten zu Beginn festgelegt werden. Wenn sich die Visualisierungselemente bei einer Stiländerung entsprechend verändern sollen, müssen die Elementeigenschaften, soweit möglich, mit Stilwerten konfiguriert sein. Fix konfigurierte Farbwerte oder Fonts ändern sich bei einer Stiländerung nicht.

Typische Stileinträge sind:

- Liste von Farben
- Liste von Schriftarten
- Bilder für Visualisierungselemente
- Auswahl des Basisstils für die grundlegende Stilistik der Visualisierungselemente

```
<AdditionalStyles>
  <!-- Styledefinition Style1-->
  <Value name="Style" type="string">STYLE1_XP</Value>
    ○ STYLE1_XP
    ○ STYLE2_W7
    ○ STYLE3_GRADIENT1_LINEAR1
    ○ STYLE4_GRADIENT2_LINEAR2
    ○ STYLE5_GRADIENT3_AXIAL1
    ○ STYLE6_GRADIENT4_AXIAL2
    ○ STYLE7_GRADIENT5_DOUBLELINEAR1
    ○ STYLE8_GRADIENT6_DOUBLELINEAR2
    ○ STYLE9_FLAT
    ○ STYLE_WHITE
```

2 Typische Anwendungsfälle

Führt man im CODESYS Store in der Kategorie „Visualization“ ein „Suchen“ durch, erhält man viele Beispiele aus dem Bereich Visualisierung.

Beispiele:

- [Global Client Manager](#)
- [MultiTouch Example](#)
- [Recipe Management](#)
- [Visu Event Handler](#)
- [Sound Demo](#)
- [Selection Manager](#)
- [Responsive Design Example](#)
- ...

2.1 Dynamische Bildumschaltung

Anforderung:

In einer Visualisierung sollen verschiedene Bilder (Pixel- oder Vektorgrafiken) dynamisch umgeschaltet werden.

Realisierung:

Bild-Element einfügen. In den Eigenschaften des Elements unter „Bild-ID-Variable“ eine String-Variable für die „Bild-ID“ angeben. Diese String-Variable kann von einem IEC 61131-3 Baustein mit der entsprechenden Zeichenfolge kommen, der dem Namen des Bilds aus dem Imagepool entspricht, z. B. „ImagePoolStates.Run“.

Es gibt eine Bildersammlung mit dem Namen „ImagePoolStates“. Sie enthält folgende Einträge in der ID-Spalte: Run, Stop,..., sowie in der Spalte Dateiname die entsprechenden Dateinamen für die Bilder.

2.2 Dynamische Textumschaltung

Anforderung:

In einer Visualisierung soll in einem Rechteck der Text des aktuellen Fehlers angezeigt werden. Der Fehler ist in einer DWORD-Variablen enthalten.

Realisierung:

Einfügen eines Rechteck-Elements. In den Eigenschaften des Elements unter „Dynamische Texte“ ist der Name der Textliste z.B. „Fehlertexte“, und der Textindex anzugeben. Der Textindex ist eine String-Variable, die von einem IEC 61131-3 Baustein kommen kann. Steht die Fehlernummer in einer DWORD-Variablen, muss beispielsweise „DWORD_TO_STRING(fb.dwError)“ konfiguriert werden.

Es gibt eine Textliste mit dem Namen „Fehlertexte“, die zu jedem Fehlerwert einen Fehlertext bereitstellt. Die Fehlertexte können mehrsprachig zur Verfügung gestellt werden.

2.3 Modus-Schalter mit verschiedenen Stufen

Anforderung:

In der Visualisierung sollen verschiedene Betriebsmodi geschaltet werden können.

Realisierung:

Einfügen eines Potentiometer-Elements und entsprechende Konfiguration in den Eigenschaften:

- Ändern der Anzahl der auswählbaren Modi: Im Bereich „Scale“ die Werte für „Main Scale“ und „Sub Scale“ auf „1“ setzen. „Scale Start“ und „Scale End“ so einstellen, dass die Anzahl der Werte den gewünschten Betriebsmodi entspricht.
- Eingabe der zu schreibenden Variable unter „Variable“.
- Ggf. Standard-Eintrag „Scale Format“ auf „%s“ umstellen (nur noch ganzzahlige Anzeige) und fixen Text davor stellen: z. B. „Modus %s“ → „Modus 1“, „Modus 2“ ...
- Ggf. die Schaltpositionen im Bereich „Arrow“ durch neue Werte für „Arrow Start“ und „Arrow End“ anpassen.

2.4 Dynamisches Wechseln zu verschiedenen Visualisierungen aus einer Hauptseite heraus

Anforderung:

Aus einer Hauptseite heraus soll in „Unterseiten“ gesprungen werden. Am linken Rand soll ein Menü für die Umschaltung erstellt werden (siehe Projektaufbau aus Kapitel 1.6 oder Beispielprojekt „VisuDemoFactory“ im Installationsverzeichnis \CODESYS\Projects\Visu\Examples)

Realisierung:

Als erstes sind die Visualisierungen für die „Unterseiten“ zu erstellen. Danach kann man in der Hauptseite ein Frame-Element einfügen und über das Kontextmenü in „Frame-Auswahl“ die Unterseiten als Liste in das Frame-Element aufnehmen.

Das Menü kann man mit Schaltfläche-Elementen erstellen. In einer solchen Schaltfläche ist folgende Konfiguration vorzunehmen:

Eingabekonfiguration – OnMouseDown – Framevisualisierung umschalten.

Es sollte nun das Frame-Element aus der Hauptseite erscheinen.

Nun kann man diejenige Visualisierung aus dem Frame auswählen, zu welcher man mit dieser Schaltfläche wechseln möchte. Danach wählt man noch „Auswahl zuweisen“ aus.

2.5 Anpassung der Visualisierungsdialoge

Anforderung:

Das Design der Visualisierungsdialoge Numpad/Keypad oder auch der Dialoge Login/ChangePassword für die Benutzerverwaltung der Visualisierung soll an das Corporate Design des Unternehmens angepasst werden.

Realisierung:

Im Installationsverzeichnis \CODESYS\Projects\Visu\Dialogs sind die Source-Bibliotheken der Visualisierungsdialoge zu finden. Diese Bibliotheken kann man öffnen und die darin enthaltenen Visualisierungen anpassen. Danach unbedingt auch die Projektinformation der Bibliothek anpassen (eigener Firmenname, eigene Version, ...).

Hinweis: Alle Dialoge sind Visualisierungen, welche in den Visualisierungsobjekt-Eigenschaften, Kategorie „Visualisierung“ mittels Auswahl „Dialog“ oder „Nummernfeld/Tastatur/Dialog für Eingabekonfiguration“ speziell gekennzeichnet wurden.

Im Projekt kann man nun die neu erstellte firmenspezifische Bibliothek in den Bibliotheksverwalter aufnehmen. Danach sind die Dialoge dieser Bibliothek und danach im Visualisierungsmanager unter „Einstellungen“/„Einstellungen für Standardtexteingabe“ oder „Einstellungen für Benutzerverwaltungsdialoge“ auswählbar.

Zur Laufzeit werden dann diese spezifischen Dialoge verwendet.

Änderungshistorie

Version	Beschreibung	Datum
0.1	Erstellt	07.12.2016
0.2	Formaler Review und Überarbeitung	05.01.2017
1.0	Formelle Freigabe	10.01.2017