Subject: Re: Schleife in Event Case zu Ende laufen lassen
Posted by Franz Josef Ahlers on Sat, 02 Jul 2005 15:52:07 GMT
View Forum Message <> Reply to Message

Vincent Peikert wrote on Sat, 02 July 2005 15:31Die Abbruchbedingung soll so aussehen, wie das Beispiel "Event Contitional Stop.vi" skizziert. Ich hätte also erst suchen sollen Mein Problem lag bei der Formulierung darin, dass ich den Kern der Events in Labview nicht richtig verstehe. Nehmen wir das oben genannte Beispiel: Ich verstehe Events so, dass sobald das Event eintrifft der zugehörige Case ausführt werden sollte. Anscheinend läuft aber der derzeitige Case immer noch bis zum Ende. Erst danach wird ein eventueller neuer Case ausgeführt, wenn das entspr. Event eingetreten ist. Dabei habe ich aber noch folgende Verständnisprobleme:
- Ist es richtig, dass das gewollte Timeout im "Timeout" Case dafür sorgt, dass der Case überhaupt gewechselt wird ? Wieso ist denn das Timeout hier notwendig ?
- Liegt der Witz der Events also lediglich darin, andere Überwacher zur Verfügung zu stellen, die nicht schon in Form von Kontollen (Beispielsweise Buttons) vorliegen ?
Der Witz der Events liegt darin, dass kein 'polling' von Controls mehr nötig ist, um auf user Eingaben zu reagieren. Es wird also, wenn keine User Eingabe erfolgt, auch keine Prozessorzeit verbraucht (außer natürlich, wenn das VI noch andere Dinge tut, als auf Nutzereingaben zu warten). Das Beispiel 'Event Conditional Stop.vi' ist insofern kein gutes Beispiel, als dort doch regelmäßig im timeout event buttons abgefragt werden. Das passiert aber nur, weil in dem simplen Beispiel ein Messvorgang simuliert wird. Das Beispeil dient wohl vor allem dazu, die 'Filter'Events (abfangen des Close-Befehls) zu erläutern.

Um die Events besser zu verstehen, war es für mich sehr hilfreich, die beiden Beispiel VIs 'Old Event Handler.vi' und 'New Event Handler.vi' zu vergleichen. Der timeout case in New Event Handler ist praktisch überflüssig, denn das timeout terminal ist gar nicht angeschlossen, der timeout Wert steht also auf 'unendlich'.

Die einzelnen cases des timeouts werden in der Tat vollständig abgearbeitet, sobald sie angesprungen werden. Das bedeutet insbesondere, dass die Ausführungszeit jedes timeout cases so kurz sein sollte, dass für den Nutzer keine merkbare Verzögerung eintritt. Die eigentliche Reaktion des Programms auf Nutzereingaben sollte am besten in einer asynchronen Schleife erfolgen, die Event-Loop darf sich nur um die Nutzereingaben selbst kümmern. Diese Eingaben werden per Queue an die asynchron laufende Arbeitsschleife übergeben.

Wenn Du per 'File-->New...' aus dem design-pattern-template 'Producer/Consumer Design Pattern (Events)' ein neues VI erzeugt, siehst Du wie so was im Prinzip aussieht.

-Franz