



# IDE Changes and Features In Allegro CL 8.2

David Margolies ([dm@franz.com](mailto:dm@franz.com))

Questions and comments to  
[support@franz.com](mailto:support@franz.com)



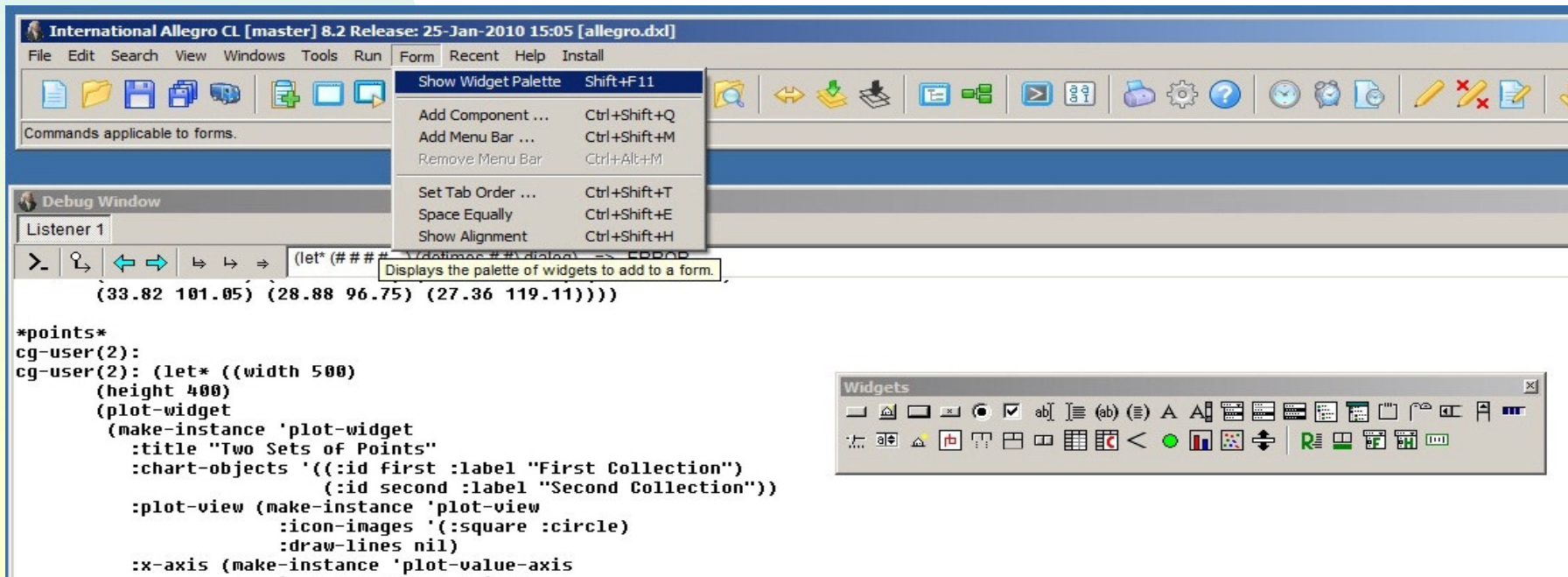
## The IDE in 8.2: Topics

- New look and feel
- Works on the Mac
- New split-bar widget
- Chart and Plot widgets: discussion and examples



## New Look and Feel

- New icons
- Widgets moved to a separate palette (display with Form menu item or clicking on background of form window).



```
International Allegro CL [master] 8.2 Release: 25-Jan-2010 15:05 [allegro.dxl]
File Edit Search View Windows Tools Run Form Recent Help Install
Show Widget Palette Shift+F11
Add Component ... Ctrl+Shift+Q
Add Menu Bar ... Ctrl+Shift+M
Remove Menu Bar Ctrl+Alt+M
Set Tab Order ... Ctrl+Shift+T
Space Equally Ctrl+Shift+E
Show Alignment Ctrl+Shift+H
Commands applicable to forms.
Debug Window
Listener 1
> (let* (###) (dotimes # #) (dialog) => FPPPP
(33.82 101.05) (28.88 96.75) (27.36 119.11)))
*points*
cg-user(2):
cg-user(2): (let* ((width 500)
(height 400)
(plot-widget
(make-instance 'plot-widget
:title "Two Sets of Points"
:chart-objects '((:id first :label "First Collection")
(:id second :label "Second Collection")))
:plot-view (make-instance 'plot-view
:icon-images '(:square :circle)
:draw-lines nil)
:x-axis (make-instance 'plot-value-axis
```



## IDE works on the Mac

- Now supported on Windows, Linux, and the Mac.
- Programs developed on any platform (with minor tweaks) will work on the others.
- (Minor tweaks: GTK and Windows not fully compatible, some Windows-only features.)

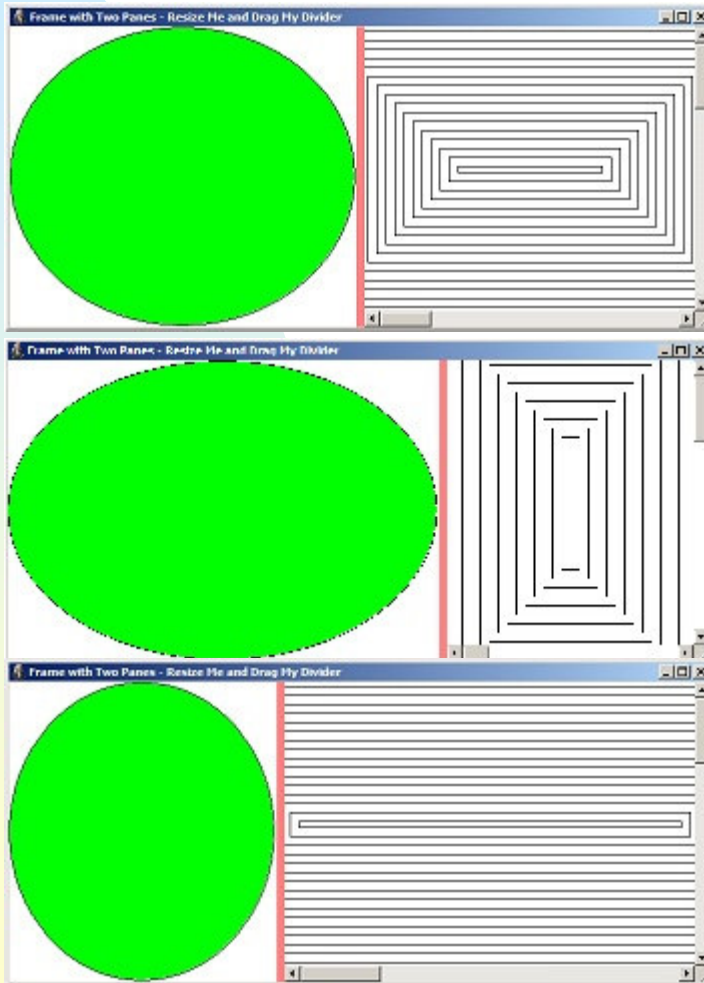


## Split-bar widget

- The splitbar widget divides a window into two pieces and can be moved, moving or resizing areas on either side.
- There was an example of writing a split-bar from scratch in 8.1 (see 8.1 Navigator Example “Two Panes with a Draggable Pane Divider”).
- The similar example in 8.2 is “Two Window Panes with a Draggable Split-bar Widget”



# Here is the Navigator Example of the Splitbar





## Here is another example:

On a dialog, we place these widgets from top to bottom:

- multi-line-editable-text
- button
- split-bar
- single-item-list

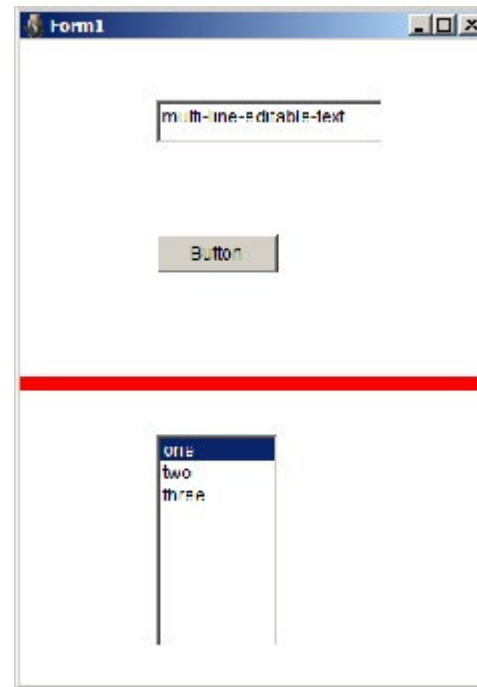
Add the names of the multi-line-editable-text and the single-item-list to the widgets-to-resize list of the split-bar and the name of the button to the widgets-to-move list.

Changes the dialog background color to white, and the split-bar colors to red, blue under mouse, yellow when dragging.



## Here is resulting dialog:

As first displayed on left, after split-bar is moved up on right (resized to fit on slide so a bit fuzzy):







## More split-bar notes

- They can also be used with window panes.
- With panes you typically you'd have no space between the split-bar and two child window panes.
- With widgets you typically have some additional space between the widgets and insert the split-bar about midway between the widgets.
- You specify colors with `color-when-idle`, `color-under-mouse`, and `color-when-dragging`. In the default, not visible except when dragging (cursor shows when over split-bar)



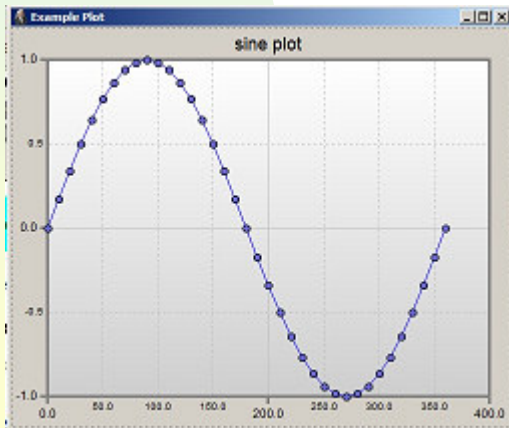
## Plot- and chart-widgets

- Have been around since 8.1 (plot widgets added as a patch).
- Very powerful, which means lots of options.
- Lots of options may make initial use daunting!
- But easy to get started, and then refine result to exactly what you want.



## A really simple plot

- `*sinevals*` is list of pairs, x from 0 to 360 by 10s, y is  $(\sin (/ (* x \text{pi}) 180))$
- We define the plot widget as  
(make-instance 'plot-widget :title "Sine Plot" :width 500 :height 400))
- We draw points with a loop doing  
(set-plot-value plot-widget :x (first (nth i \*sinevals\*)) :y (second (nth i \*sinevals\*)))





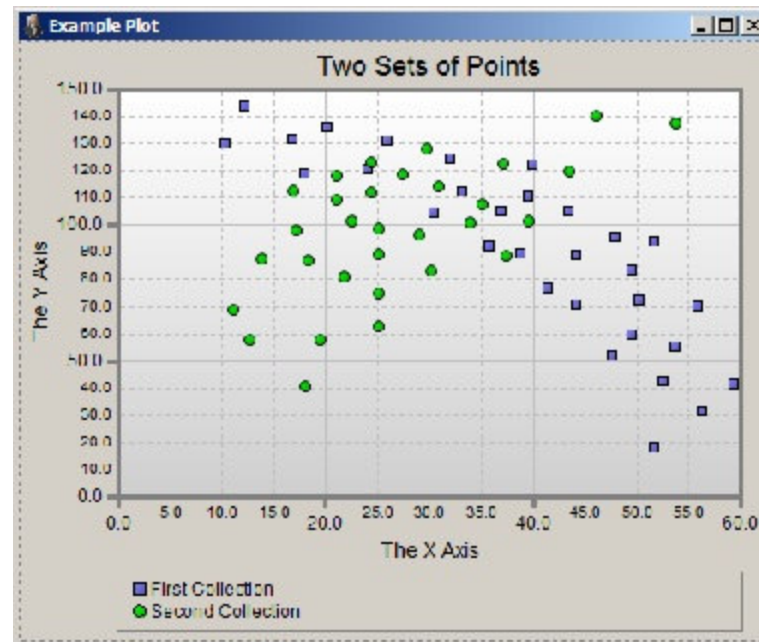
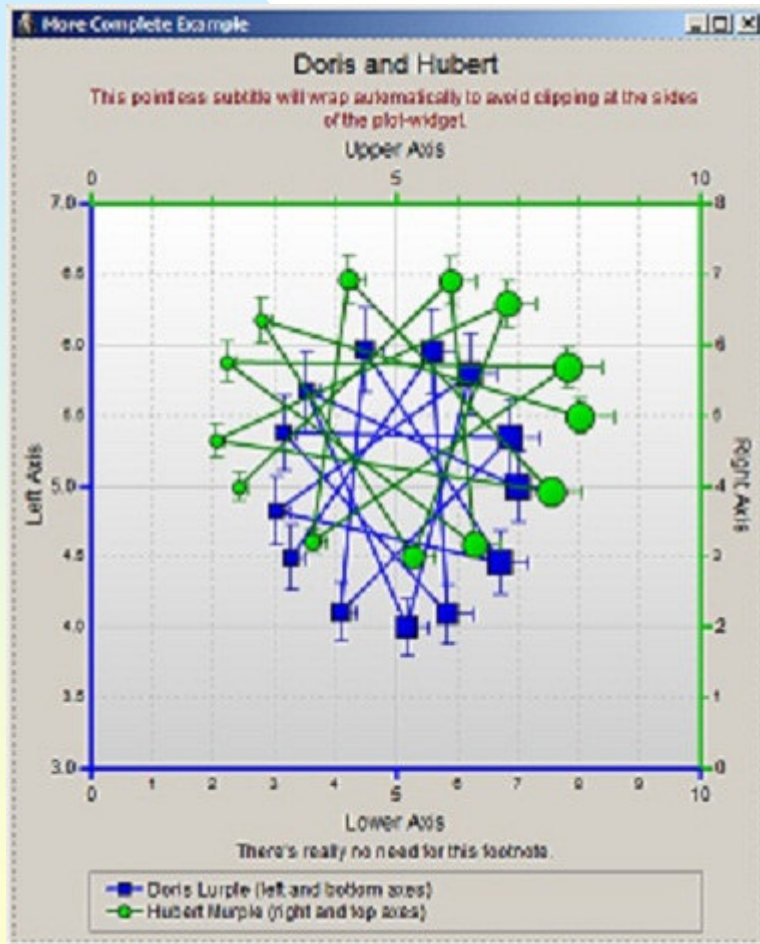
## Argument defaults are reasonable

- Simply providing the data (with set-plot-value) is essentially all that is required for getting a plot.
- The defaults provide two levels of tick marks, points marked with connecting lines (you have to specify if you want points only).
- If you want things to look different, find examples that look like you want and imitate them.



## More plot examples

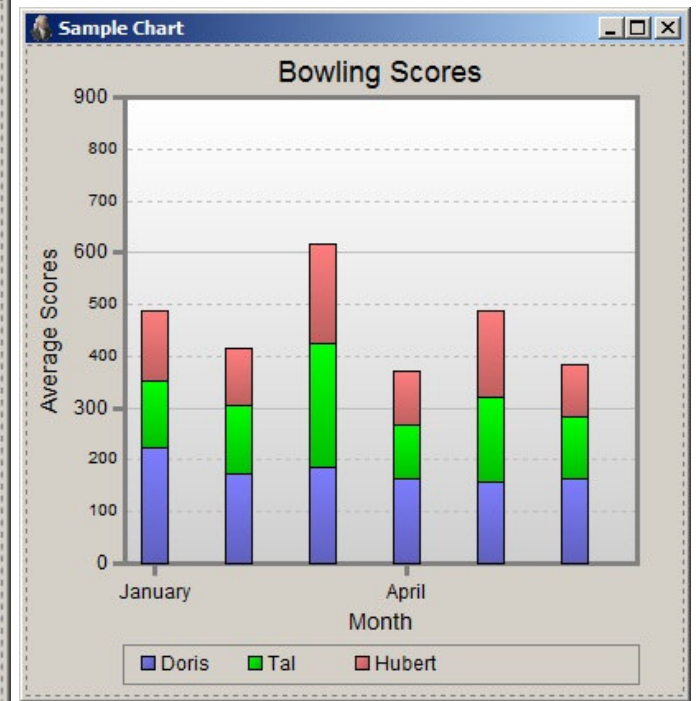
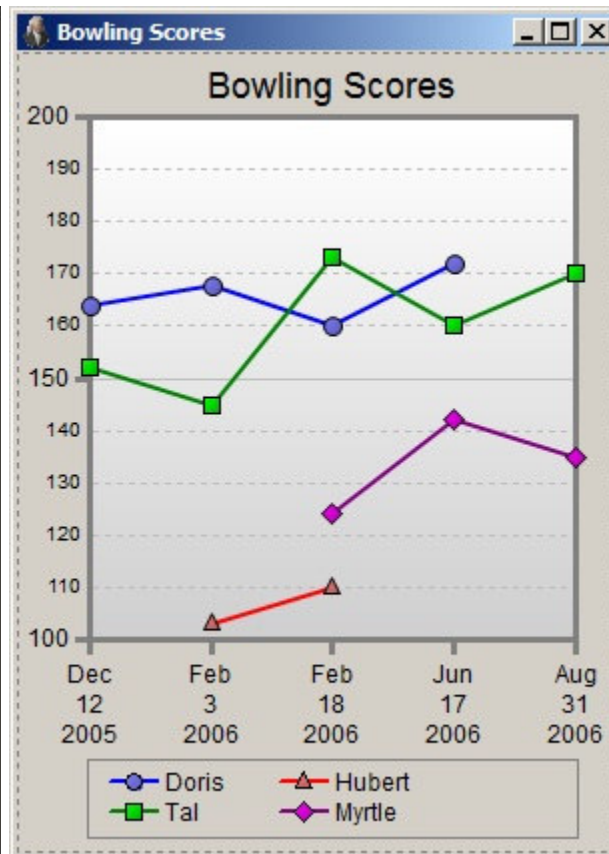
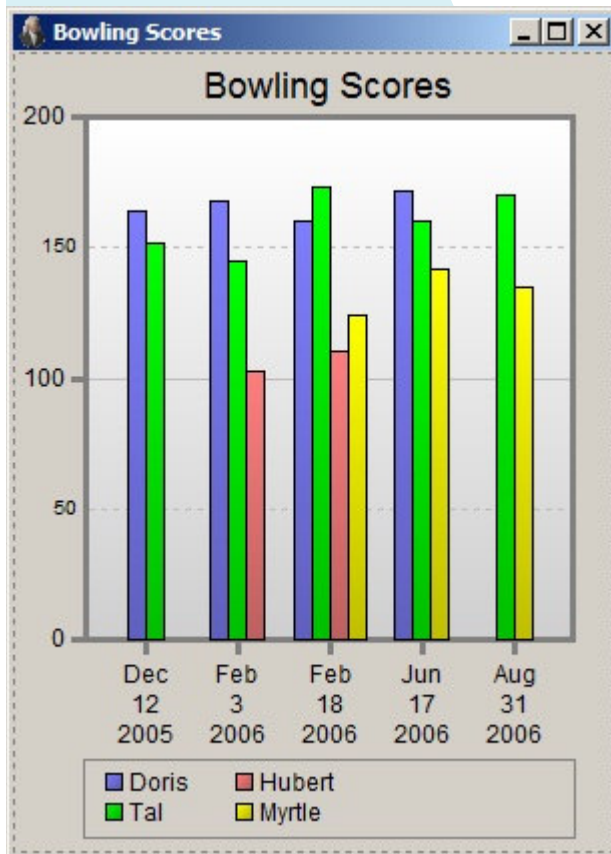
These examples  
are on the plot-  
widget  
documentation  
page





# The chart widget

- The chart-widget shares many properties (particularly those associated with labels and axes) with the plot widget





## The chart-widget versus the plot-widget

- The plot-widget is similar to the chart-widget, but designed for arbitrary (x, y) data. (The chart-widget evenly spaces items on the X axis so can do graphs but only when the X data is evenly spaced.)
- The plot-widget axes are both numeric. The chart-widget X axis is often categorical (objects like people, countries, months or years, etc.)



## Some chart-widget (and sometimes also plot-widget) features

- set-chart-value vs. chart-value-returned (also plot-in place of chart-)
- Views: chart-view, bar-chart-view (also plot-view)
- Axis tics: major and minor.
- Labels
- Documentation





## set-chart-value vs. chart-value-returner (also plot- in place of chart-)

- You can specify values in a data structure and then have the set-chart-value (or set-plot-value) extract them.
- You can define a chart-value-returner (or plot-value-returner) function which will be called on each item.
- If using a returner function, you must somehow specify the number of items.

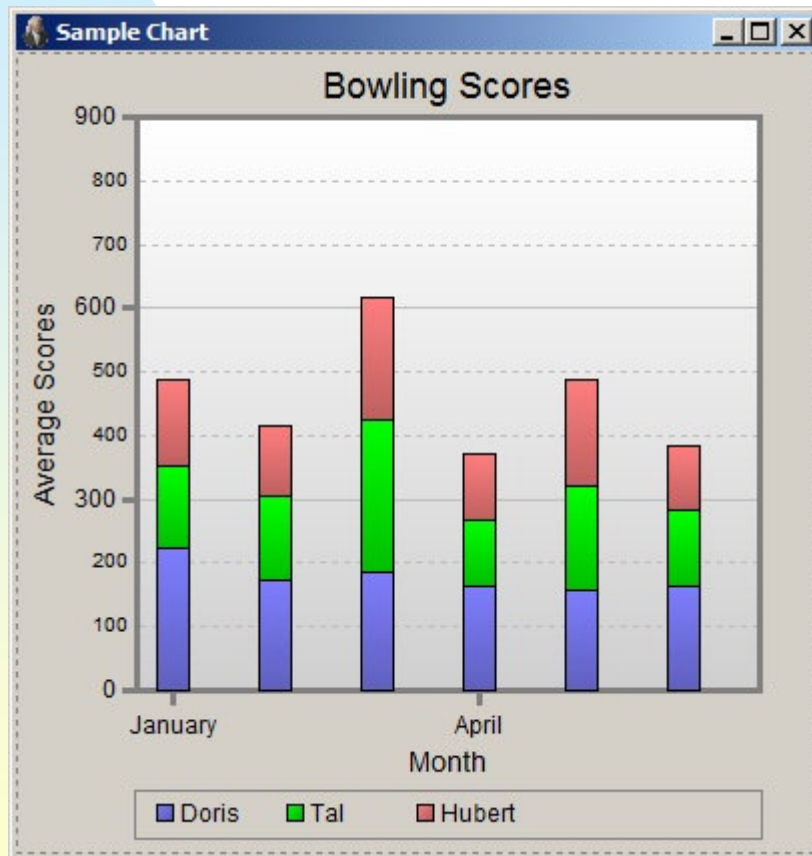


## Views: chart-view, bar-chart-view (also plot-view)

- Views give a broad description of how the data should be displayed.
- For charts: :bar and :line, or a chart-view object. Bar-chart-view can have values-are-stacked t.
- For plots: plot-view object with :draw-lines t or nil.



## Axis tics: major and minor



Note the Y axis has major and minor tics, while the X axis only labels some bars.



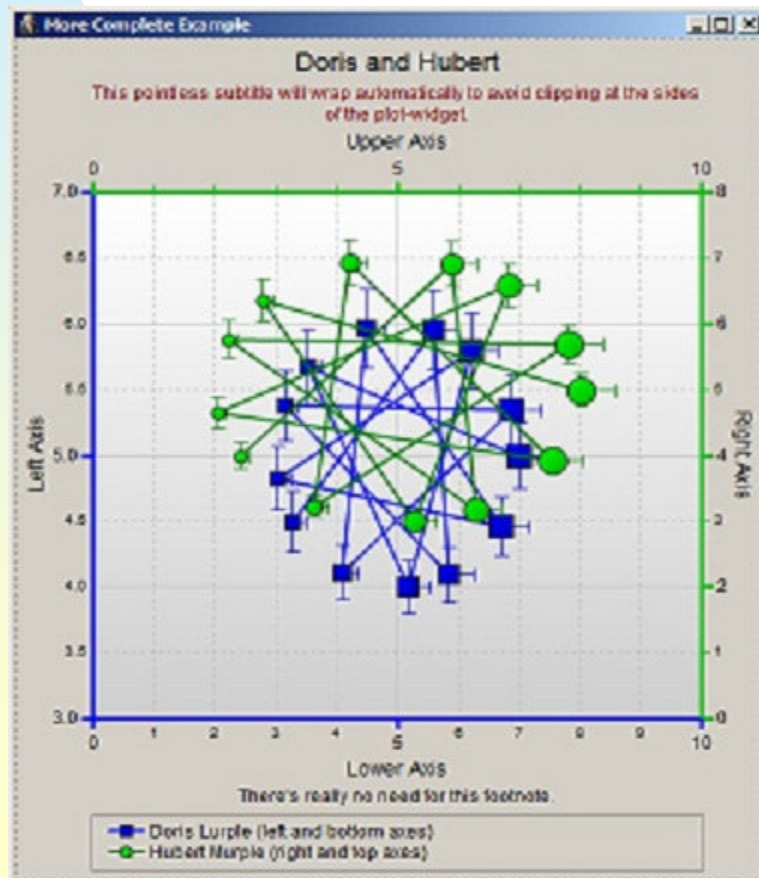
## More on axis tics and which bars are labeled

```
:item-axis (make-instance 'item-axis
                          :axis-label "Month"
                          :minor-tics-per-major-tic 3
                          :draw-minor-labels nil
                          :on-print-major-label
                          (lambda (month-symbol)
                            (format nil "~:(~a~)" month-symbol)))
:value-axis (make-instance 'value-axis
                          :axis-label "Average Scores"
                          :margin-inside-axis-label 8
                          :range-bottom 0
                          :range-top 900
                          :major-tic-increment 300)
```



## Labels

- There are axis labels, a title, and a place where the things being graphed/charted is shown





## The properties which display the labels

```
:title "Doris and Hubert"  
:subtitle #.(format nil "This pointless subtitle will wrap ~  
                automatically to avoid clipping at  
                the sides ~  
                of the plot-widget.")  
:subtitle-color dark-red  
:footnote "There's really no need for this footnote."
```

The axis labels and the multiple axes described on previous slide. See the last example on the [doc/classes/cg/plot-widget.htm](http://doc/classes/cg/plot-widget.htm) page



## Documentation

- chart-widget and plot-widget pages  
(doc/classes/cg/chart-widget.htm,  
doc/classes/cg/plot-widget.htm)
- Pages for properties and also for related  
classes
- doc/cg/cg-chart-widget.htm



## IDE Changes and Features Allegro CL 8.2

- David Margolies ([dm@franz.com](mailto:dm@franz.com))
- Questions and comments to  
[support@franz.com](mailto:support@franz.com)