Subject: [solved] ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Sun, 10 Nov 2019 00:13:46 GMT
View Forum Message <> Reply to Message

Dear gentlemen,

I wasn't really sure if this would better be asked in the newbie corner, but anyway, other than being a noob, hopefully I am not off topic here... :)

I keep playing with the ScatterCtrl a bit, discovering its endless capabilities - I haven't found any hard limits yet, and I keep coming across amazing things, such as the inherent "save image" context menu option (a true marvel).

And the question today is: I've been wondering for a while, how to nail particular dimensioning/limits per X/Y axis. I would be fine with a fixed setup, I just don't know *how* for the life of me :) More specifically, I've been fumbling through the ScatterCtrl demo, I'm slightly disoriented in the source code... I liked the RangePlot example, but I guess that the whole dimensioning in the source code is handled by a single call to ZoomToFit(), which is something I'd prefer not to use... But, at runtime, I have discovered another inherent menu of the ScatterCtrl object: the "properties" dialog, containing a "Measures" tab. See the screenshot attached. Is there any chance to access those attributes from my own program? I'd love to set them even before I hand over my dataset to the control.

I've noticed a recent topic on "Grid lines at round dates or values." That's not exactly my problem. I can choose a nice round per-axis "major unit" (= grid line spacing, if I understand correctly) and configure the respective per-axis min/max such that they're integer multiples of the "unit". It's no problem to come up with those numbers, I just don't know *how* to do it in my program.

Looking at the configurable attributes of the control, I suspect a slightly different problem just around the corner: my data is a "spectrum", the output of FFT. (Several FFT outputs combined, in fact.) In the data plane it's pretty similar to time series, but let me be frank and speak the actual axis units that I have. I can smell a problem in that the data set has a single explicit dimension: the amplitude per a frequency "bucket" or quantum = the height of each "spectral line". The data set is a series of evenly spaced "discrete frequency bands". Say one kHz bandwidth per frequency line. The ScatterCtrl tends to label the X axis e.g. 0 to 2048 = just counting the "data points". I'd love to tell ScatterCtrl that the data set really starts at 540 MHz and ends at 542.048 MHz - and have a grid line per 1 MHz or per 10 MHz or some such nice step. I can align the min and max on the X axis to integer multiples thereof. But I'm not sure if this is gonna fly... would I perhaps have to provide the data as two-dimensional series, having the X axis as a second dimension? Or is there an easier way to scale+shift+decorate the X axis?

So far, I've been passing a plain C array of Double to the ScatterCtrl... is there perhaps some container object that would allow me to set the Y-axis limits and to describe the scaling of the implicit X dimension, for ScatterCtrl to understand it?

Thanks for your time and patience with me :)

Frank

File Attachments
1) ScatterTest_demo_RangePlot.png, downloaded 347 times

Subject: Re: ScatterCtrl: program access to the "Measures" ?
Posted by koldo on Sun, 10 Nov 2019 09:23:20 GMT
View Forum Message <> Reply to Message

Sorry Frank

I am lost in your explanation. Do you just want to define by code the plot area to show?

Subject: [solved] Re: ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Sun, 10 Nov 2019 10:34:41 GMT
View Forum Message <> Reply to Message

Hello Koldo,

yes I do... and, oops, I'm probably home already :) I must've been way sleepy yesterday.

I've just spotted:

the "x0" and "delta X" attribute to ScatterCtrl::AddSeries() :) which sure does help with the scaling along the X axis ScatterCtrl::SetMinRange(x,y) and ScatterCtrl::SetMaxRange(x,y) ScatterCtrl::SetMinXmin(x), ScatterCtrl::SetMinYmin(y), ScatterCtrl::SetMaxXmax(x), ScatterCtrl::SetMaxYmax(y), ScatterCtrl::SetMajorUnits(x,y) == meaning major unit/tick/grid along axis x and y ScatterCtrl::SetMinUnits(x,y) == meaning minor unit/tick/grid?

I guess I have all that I need :)
Apologies for the stupid question.

Frank

Subject: Re: [solved] Re: ScatterCtrl: program access to the "Measures" ?
Posted by koldo on Sun, 10 Nov 2019 11:38:06 GMT
View Forum Message <> Reply to Message

Quote:ScatterCtrl::SetMinUnits(x,y) == meaning minor unit/tick/grid?Yes, that's it.

No worries :)

In addition, if you go to Process/Frequency, you could try the FFT options included.

Subject: Re: [solved] Re: ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Sun, 10 Nov 2019 17:23:25 GMT
View Forum Message <> Reply to Message

Thanks for mentioning the built-in FFT again... so far I'm pretty happy with kissFFT and some trivial pre- and post-processing around it. And just about now I'm eyeing I+Q imbalance correction, which I don't think the ScatterCtrl would allow me to do... Still you guys have my respect for including FFT in ScatterCtrl :)

Subject: Re: [solved] Re: ScatterCtrl: program access to the "Measures" ?
Posted by koldo on Sun, 10 Nov 2019 19:20:18 GMT
View Forum Message <> Reply to Message

Hi Frank

You know more than me about signal processing.
Anyway, ScatterDraw calls Eigen, that includes kissFFT.
So in summary, ScatterDraw includes kissFFT :)

Subject: Re: [still wondering] ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Tue, 12 Nov 2019 21:58:30 GMT
View Forum Message <> Reply to Message

Hello Koldo and everybody,

I've moved a little bit further with ScatterCtrl, but it seems like I'm stuck again.
My code contains the following snippet:

```
Chart1.AddSeries(buffer, buf_size,
  /*x0*/ from_freq_rounded, // Hz
  /*delta x*/ freq_sample_width // HZ
  );
//Chart1.SetMinXmin(from_freq_rounded);
//Chart1.SetMaxXmax(to_freq_rounded);
//Chart1.SetMinYmin(-90);
//Chart1.SetMaxYmax(0);
Chart1.SetLabels("Hz","dBFS");
Chart1.NoMark(0);
Chart1.Stroke(0,1,Blue());
Chart1.ZoomToFit(true,true,0); // X, Y, fill factor (0 = fill the whole control)
//Chart1.Refresh();
```

This way, it does display a trace, the labels do get modified, and the ZoomToFit works too.
But, if I remove (comment out) the ZoomToFit, and instead use the

SetMinXmin/Xmax/Ymin/Ymax, those functions don't seem to work. The range just stays put at 0..100 at both axes. See the attached screenshot.

I'm wondering if I'm missing some other member method or flag to tweak, to "unpeg" those range setting functions. For instance, in the runtime "Properties" tabbed dialog of the ScatterCtrl, there's a check mark called "Attach" per axis (one for X, one for Y) - if that's ticked, the min and max input boxes are grey (deactivated). But I haven't found any access to those "Attach" check marks from my program code... ZoomToFit seems to work regardless of those "Attach" check marks. And if at runtime I first remove the "Attach" check marks manually, and then run the piece of code outlined above, the minXmin(), maxXmax() et al still do not work...

So this is what it behaves like in the 12610 nightly. I've tried upgrading to 13655, but I'm stuck in another problem (problem to include the kissfft plugin - see the thread in newbie corner if interested.) So at the moment I cannot test if the upgrade solves my problem with ScatterCtrl range setting.

So if at this point there are some suggestions, I'd love to hear them :)

File Attachments

1) skyline.png, downloaded 335 times

Subject: Re: [still wondering] ScatterCtrl: program access to the "Measures" ?
Posted by koldo on Wed, 13 Nov 2019 07:32:15 GMT
View Forum Message <> Reply to Message

Hello Frank

To clarify it I have changed ScatterDraw_demo:
scatter.SetXYMin(10, 0).SetRange(50, 100).SetMinUnits(20, 0).SetMajorUnits(10, 20);
//scatter.ZoomToFit();The result is:

File Attachments

1) Scatter.png, downloaded 691 times

Subject: Re: [still wondering] ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Wed, 13 Nov 2019 22:05:52 GMT
View Forum Message <> Reply to Message

Hello Koldo,

thanks a lot for that snippet of code, exactly what I was looking for.
Your sketch demonstrates to me something that I guess I've grasped before: the offset and spacing, given via AddSeries(), allows you to submit several time series that are not precisely aligned along axis X (= in time, typically).

And, I now finally understand what the "minor" units are.
The Major unit is clear, that is the dotted grid spacing.
The "minor" unit, as given by SetMinUnits(), is the offset of the dotted grid from the axis minima, as given by SetXYMin().
(Counter-intuitively to me, this is not a second smaller "axis tick mark" or anything like that.)

For the record, for people who may come later and may be just as lost as I was, I'll attach two screenshots of my app, with comments inscribed. They correspond to the following two snippets of UPP code:

```
Chart1.SetXYMin(min_freq, -90).SetRange(max_freq - min_freq,
90).SetMinUnits(0,0).SetMajorUnits(1000000, 10);
```

```
Chart1.SetXYMin(min_freq, -90).SetRange(max_freq - min_freq,
90).SetMinUnits(100000,5).SetMajorUnits(1000000, 10);
```

And, if I divide the x0 and DeltaX (= parameters to AddSeries()) by a million, the DeltaX gets fractional, which is no problem, and I get a display in 3-digit MHz, rather than 9-digit Hz :)
Good...

Frank

Subject: Re: [still wondering] ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Wed, 13 Nov 2019 22:08:00 GMT
View Forum Message <> Reply to Message

Screenshot #1:
Chart1.SetXYMin(min_freq, -90).SetRange(max_freq - min_freq,
90).SetMinUnits(0,0).SetMajorUnits(1000000, 10);

File Attachments
1) skyline_minor_0-0.png, downloaded 350 times

Subject: Re: [still wondering] ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Wed, 13 Nov 2019 22:11:13 GMT
View Forum Message <> Reply to Message

Screenshot #2:
Chart1.SetXYMin(min_freq, -90).SetRange(max_freq - min_freq,
90).SetMinUnits(100000,5).SetMajorUnits(1000000, 10);

File Attachments
1) skyline_minor_100k_5.png, downloaded 322 times

Subject: Re: [still wondering] ScatterCtrl: program access to the "Measures" ?
Posted by koldo on Thu, 14 Nov 2019 07:07:20 GMT
View Forum Message <> Reply to Message

Yes, you are right, the explanation is unclear.

I have changed the doc, adding that, in SetMinUnits(double ux, double uy), if either ux or uy is not visible, it ensures that values obtained adding or substracting SetMajorUnits() values will be visible.

This way you can change SetXYMin(), and SetMinUnits() will still be valid.

Subject: Re: [solved] ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Thu, 14 Nov 2019 11:51:55 GMT
View Forum Message <> Reply to Message

Dear Koldo and everyone,

i have one last snippet that I'd like to append to this topic, again for people coming after me:

The ScatterCtrl allows me to update the contents of the "data series buffer" after I have passed the buffer to the ScatterCtrl instance. The fixed-length buffer is passed by a pointer, and the documentation contains enough of a warning that the storage for the buffer must remain allocated until either the ScatterCtrl object is destroyed or the particular "data series" is withdrawn from the ScatterCtrl object using the RemoveSeries() or RemoveAllSeries() method. So: apparently ScatterCtrl merely "refers" to the external buffer, owned by me - and if I update some data in the buffer, I can tell the ScatterControl object to update its appearance using the Refresh() method.

In my overview spectrum scanner toy app, I'm using this to keep the user entertained: to show some data regularly as I obtain it from the radio. I scan the band in "chunks", corresponding to the "channel bandwidth" available from the radio (about 2 MSps = 2 MHz, centered on the tuned frequency). I split the broader band into 2MHz channels that I scan in sequence. And I can display each channel as soon as the sampling is done and the FFT gets calculated. To the user, the band-scan starts growing from the left to the right "in real time" - displayed by the ScatterCtrl object.

The following is a piece of pseudo-code, showing how I go about it:

```
// at the click of a button = in a "button event callback":
allocate global_buffer[buf_len]; // the aggregate buffer spanning all channels
zero_fill(global_buffer); // the chart starts out showing a flat line at y=0
Chart1.AddSeries(global_buffer, buf_len, starting_freq, channel_width);
Chart1.adjust_axis_units_and_view_dimensions();
Ctrl::ProcessEvents();

for (each channel)
{
```

```
 tune_into(channel);
 sample_chunk_from_channel();
 calc_FFT_for_the_channel();
 // project per channel FFT results into the overview buffer
 memcpy(global_buffer+chnl_offset, channel_FFT_output, chnl_data_len);
 Chart1.Refresh();    // tell the ScatterCtrl to update its output
 Ctrl::ProcessEvents();  // allow the GUI to actually carry out the updates pending :-)
}
```

And the key point why I'm writing this blurb is: Ctrl::ProcessEvents() :)
This turns out to be a key ingredient.

Why:

The GUI runs in a thread, handling Windows messages (those translate into U++ Events).
The message-driven programming model mostly ticks away swiftly, as the events are swiftly handled.
But: my band-scan activity is different. It hogs a long period of the single GUI thread's time.
Without Ctrl::ProcessEvents(), my app behaves this way:
While my lengthy "button click service routine" is running, taking its loops in scanning the channels,
the whole GUI is stuck, does not respond to events - that including the events I would like to incite by calling Chart1.Refresh().
The whole band scan can take a couple dozen seconds - during which time, nothing on the screen moves. Even my "gradual" updates of the data series in the chart (ScatterCtrl) all appear to rush in only at the very end of the band-scanning function, rather than gradually - in spite of me calling Chart1.Refresh() after every channel is processed.

Again the sprinkle of secret sauce here is called Ctrl::ProcessEvents(). If I call it after every Refresh(), the chart actually updates, the "go" button returns to its "unclicked" position, generally the GUI becomes more "alive" :) during the band-scan.

Still it needs to be said that the GUI does not actually get fully alive, as it's still not asynchronous to my band-scan function. To let the GUI respond to events in the usual way, while the band-scan is running, I'd have to run the scan in a background thread. Which has been my next goal anyway. Being conversant in Posix Threads and locking primitives, I can't wait to learn how much of those goodies has U++ possibly managed to include in its multi-platform API :) Last night I've discovered a nice U++ forum thread on the topic.

I hope this helps someone...

Frank

P.S.: For people who remember Delphi, ProcessMessages() might ring a bell :)

Subject: Re: [solved] ScatterCtrl: program access to the "Measures" ?
Posted by Oblivion on Thu, 14 Nov 2019 12:56:38 GMT
View Forum Message <> Reply to Message

Hello Frank,

Quote:
Being conversant in Posix Threads and locking primitives, I can't wait to learn how much of those goodies has U++ possibly managed to include in its multi-platform API Smile Last night I've discovered a nice U++ forum thread on the top

As for U++ threads, and MT primitives, I'd suggest you looking at CoWork (Worker thread pool), and AsyncWork (a worker thread impl. similar to promise/future pattern, with result gathering, based on CoWork). Or if you dont require pools, buy simply asynchronicity, and somewhat lower overhead, there is also a dedicated/worker thread hybrid implementation called "Job" (This, again,is a future/promise pattern impl. with result gathering, error management and, cancelletion. It is basically a stripped down version of AsyncWork. And it can be found on my git repo.)

Best regards,
Oblivion

Subject: Re: [solved] ScatterCtrl: program access to the "Measures" ?
Posted by koldo on Thu, 14 Nov 2019 13:30:17 GMT
View Forum Message <> Reply to Message

Hello Frank

In addition, ScatterDraw has some functions for faster response:
- SetMode() lets you choosing between default good quality MD_ANTIALIASED and faster MD_DRAW
- SetFastViewX() just paints one value per x pixel
- SetSequentialX() considers that all points are ordered in X axis

Subject: Re: [solved] ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Thu, 14 Nov 2019 13:50:57 GMT
View Forum Message <> Reply to Message

@Oblivion: thanks for the pointers to your pieces :o) I'll take a look, although I suspect in advance that I'll probably just try to implement a single thread to do stuff in the background. I don't really need a "farm of grunts", because I have just a single radio anyway. Handing over the results to a superior thread (pass objects by reference into a queue atomically, or some such) might come in handy though - it will save me some learning of the basic locking primitives, if I want to be lazy :)

Subject: Re: [solved] ScatterCtrl: program access to the "Measures" ?
Posted by xrysf03 on Thu, 14 Nov 2019 14:06:10 GMT
View Forum Message <> Reply to Message

@Koldo: thanks for the suggestions. It feels pretty fast the way it is, but I really haven't done any profiling yet, so maybe those tweaks will come in handy.

Antialiased view isn't really necessary, although I have to say it looks sexy :)

One pixel per value doesn't feel necessary or beneficial - especially when looking at a high-res display from further away, the connecting lines are better visible than single dots, and even some "dense fuzz" (multiple values per horizontal pixel) may be useful for that same reason. Has to do with "ergonomy in field use" :)

Not sure about SetSequentialX()... this is probably automatically true in my case, as the data series has the X coordinate taken "implicitly" from the order in the buffer?

Anyway U++ feels like "locked by mistake in a toy store overnight". I have plenty of toys to go through...