**Programming Plan – Spine Labels / Library Labels Sotware for Zebra Printer**

**Wednesday March 19, 2014 Log Hill Mesa**Better late than never. I wrote the SpineLabels program 2 or 3 years ago but have not located any original documentation.

SpineLabels is a print application targeting the Zebra Thermal Transfer printer. Basically the printer can print text according to internal or stored fonts, or it can print a raster graphic. We use a mix of both methods.

The labels we buy and print on are really two labels side by side. the smaller label is called the spine and the larger one to the right is the pocket. These are historical library terms but they describe pretty well that the spine of a book needs like a 1” x 1.5” label and that almost anywhere else you would put a label it can be more like 1” x 3”

At our library, WPL, we don’t really use the pocket label for much of anything, which is kind of a waste. CMU Steamboat library is using this program, and they do use the pocket label on their materials.

In order to get a beautiful ‘wysisyg’ spine label, I use the raster graphic method. The user composes the label in a standard windows textbox control which happens to be an exact pixel x pixel match to the printable area of our label. Then we send the entire image as a raster to the printer.

Because WPL uses the pocket label simply to identify which book the SpineLabels goes on, I use the simpler text print method with the default font to specify text on that label.

The program also has a listbox built into it and some primitive file capabilities. So we can create a list in Sierra/Millennium of items needing spinelabels, and if this list is properly exported it can then be opened by SpineLabels and the listbox will be populated. Each row of the listbox is a call number, title, and barcode.

The user can highlight and print each row in order. Alternatively if we have loaded a very large file, the user can search for the item they have in hand and print just that spine label.

all in all a very useful program when the zebra cooperates.

Time for an Upgrade

Besides for applying all my new C# knowledge to an old and crudely written program, we have specific improvements we want to make:

* treat the pocket label like the spine, i.e. create a full scale textbox to contain the data and raster that down to the printer.
* liberalize the allowed format of the file data and its representation in the listbox. The idea is that any number of fields may be in the list rather than exactly 3. The first item will go to the spine label and the rest to the pocket label.
* if the first field is labeled ‘Call Number’ then we perform the stacking operation.
* all subsequent fields go to the pocket label. we give them each a newline if needed, and they are free to wrap around onto second lines as needed.
* be smart about the list delimiters. recognize the common CSV schemes
* we have a setup page that allows the user to micro adjust the raster image right or left. Include similar capability for the pocket label box and see if we really need a separate window of if it could all live on the main form.
* paste directly into the textboxes. we have a special paste that recognizes and parses a call number onto multiple lines. this is called stacking. I don’t think we need a special button for this, it should be the default paste.
* a proper settings dialog box. better update mechanism.
* I want to share the font buttons between the two textboxes. So I guess I will put them in the middle between the two. So I need to be clear about which is the active textbox…maybe some colored border that could light up? Just watching for where the cursor is seems dodgy.

**Thursday March 20, 2014 Log Hill Mesa**

Ok got a bit of work on this done yesterday, although the textbox focus thing got me way hung up. I did track down that annoying thing where the textbox has a string sent to it and the whole damn thing displays blue. For that I copied some dude’s miracle ‘NonSeletectingTextBox’ from the internet. Only problem I have is that I went into the designer code and replaced all the plain TextBox declarations with the NonSelectingTB, and designer keeps wanting to ‘adjust’ these settings. I suppose I will end up pulling out all the code that references this and putting it in a separate module. Until then I want the freedom to drag and tweak the text boxes.

It looks like the way to add custom controls to the designer is to create a .dll and then import it. OK, well is that so hard? I suppose I will want to use this textbox on the next project…

meanwhile I guess I am just coping as best I can.

**Saturday March 22, 2014 Log Hill Mesa**

program is starting to take shape. I am just dragging the routines over one at a time and cleaning them up.

I remember a discussion from the first go around. That’s the notion of doing the file in series. It goes back to the fact that we think there are 2 major ways to use the program. We could call them the Oak way and the Amy way. Oak’s way is to load 10,000 lines of DVD and CD data into the program. He then scans in an item as it comes across his desk, hits print and relabels the item. Oak’s way was called Random Access in the old days. Amy’s way is to load a file with at most a few dozen lines, and print each one in turn. She hands the resulting strip of labels to the assistant.

So the current program serves Oak a little better than it does Amy. Oak just has to scan the DVD, and every effort is made to have focus on the correct textbox all of the time. Amy has to manually mouse over to the list and click the next item in the list. We have always felt that there should be a ‘Next’ button or even an auto delete.

How about a new column, added to the file data, called ‘printed’. Perhaps there is an option to include a checkbox on our listview control so that the program can checkoff the items that have been printed.

so what about a checkbox in our PrintingOptions menu called AutoIncrementAfterPrinting? We can modify the display text on our big button to say ‘Print++’ and even include a set of next/back buttons.

**Sunday March 23, 2014 Log Hill**

So far this has been an easy rewrite. Our core routines to print zebra graphic commands were sound, now we are just creating an alternate gui.

2 major areas left to attend to. One is a gui thing, how to supply a page setup screen. the second one is deeper thorn, it’s the recurring problem of parsing exported files.

I have attempted to get to the bottom of this on several occasions. I think ultimately we decided that we needed the literal names of nearly every header token that could be produced. So then that makes it pretty easy to find out what the separator character is. I think I got pretty close one time with lots of regex and complicated logic.

I can seem to find it on this hard drive or figure out how to make Kaseya work.

So for now I think I need to add the extra column just by having maybe like a couple acceptable header patterns. Next week I will try to find the code I remember building.

just out of curiosity, what would this thing look like? For starters a big old list of fields. So “BARCODE”, “PCODE2”, “TITLE” “NAME” “ADDRESS”

what our program was attempting to do was also to determine if we had a set of Patron Data, Order Records, etc. So there was a lot of note taking to perform. I think we did a bunch for item and bib records. The pertinent question here is whether there is a list that we can get directly from our administrators in G.J.

Keith was able to pretty easily get me location codes. These must be different in each consortium. but the terms “call number” or “author”? They may well be hard coded into the program. I know these folks play to a global market…I wonder how many librarians have to learn English in order to use Sierra.

well I just spent a while looking on the iug listserv to no avail. I guess I would go through the channels and ask Mark or Keith.

the idea though, is that we have a list of known names with various attributes. at least some indication if they are fixed or variable fields – meaning they might be multivalued – and whether they are of patron, item, order type etc. I recall that the marcfields are an interesting subset and so are the record numbers which frequently need the checksum tweaked (that I think we will find on the listserv) The other info in this table was also a regex pattern to recognize the data fields. so barcodes would be \d{13}, record numbers might be \.[i|b|o|p]\d{9}, that kind of thing.

I just read a post where one librarian called in normalizing the data when you take a row that has a multi-valued field (like address) and dupe out the rows one per variant. I wrote a version of this for the patron analysis project as I recall.

what about our spinelabels project? Coming from Millennium, how often would I have multiple call numbers? barcodes? titles? I think I could tweak some records and show you dupes in the two item fields. Don’t know about the title, which is some kind of extraction from the marc record. it may only ever be single valued…as if it were a fixed field, even though the 245 marc fields are necessarily multivalued.

Okay, and I think where I got a bit hung up with this project was parsing the subfields. are ‘;’ all that easy to detect? Probably not. I wonder if there is some gratuitous column of data we can always add that abounds in multiple values. Patron barcodes would be a good example. The barcodes themselves are easy to spot, and a healthy percentage of patrons have multiple barcodes. Not true typically of item records. Nice if we could get the locations string from the bib record.

Now…we think our poor little spine labels doesn’t need too much fanciness. I suppose I need to think about the level of processing each of these export files needs and have a tiered system. I don’t think I need to normalize the data in the sense referred to above.

Here’s a thought though. Our usual end product is an 2D array of string values that we can then use to create our table in Word or a ListView control, whatever. So we usually do a bunch of splitting via regex. Okay…is there any advantage in using regex to directly transform our input into a standard output? Like my standard seems to have evolved to (‘|’ separator no quotes ;) Tabs as delimiter must also be a popular choice, as is the iii default (‘,’ quotes ;) which fails miserably!

so just saying…can we regex directly to AndyStandard? or is it in fact smart to create the split? Well for one thing…we do want to check that after we parse our header all of our rows split to the same n columns.

**Monday March 24, 2014 Log Hill Mesa**

so I may be onto a better or at any rate different way to calculate whether text fits. A RichTextBox can be set so that it only displays a vertical scrollbar when one is needed. So a multiline box with WordWrap set true and scrollbars set to Vertical has that property. It seems to work visibly, now I am looking at some winapi to figure if the scrollbars are indeed visible.

Weird thing about WordWrap is that in our spine label box the algorithm will break a word wherever. like not just spaces between the words but wherever it possibly can. I guess that’s up to the user to figure out…or I suppose I can check it myself.

we had a technique that was also looking good with a plain text box, use the getCharIndexFromScreenCoordinate or whatever exactly it’s called. It had some weird behavior but I think would also work.

But as we know a rtb is a nice thing in and of itself…so let’s give it a try.

okay, that was pretty easy. and I may not even need the api, apparently the client size decreases as soon as the scroll bar appears. Hey I guess the api is a whole lot easier.

so…what about this breaking in the middle of a word? I suppose at some point the number of physical lines exceeds the number of logical lines…but the phenomenon I am trying to detect can happen lots sooner.

so…what happens when we wrap a line normally at a space? does the space stick around at the end of the line? detectable in the lines? no they don’t.

so how do I tell the difference between a wrapped line and a line with a crlf? do I need to figure that out? Do I have to parse the two datasets in parallel?

okay, we break up each line into space separated tokens. Can all of these then be found in the .text complete? that’s one way I guess. basically we are going to tokenize both formats and see if they are equal. equal in count might be all that we really need..

okay…suppose we use regex to count the delimiters in the original text…and then the delimiters in the individual lines plus the line count…would that do it?

.Lines is always the logical array without any indication of wrapping. when we use GetLineFromCharIndex we get a physical line. So if we scan the text char by char we might see the physical line jump while we are in the middle of a word…that would tell us something.

okay, that wasn’t so bad in the end. we troll for physical lines and each one we find we look up its predecessor in the .Text. When that char is a space we continue, but when it’s a printable we know our long word got broken.

Okay, another issue to solve. I hooked up this routine to the text changed and the font changed. it seems like a font change actually triggers a text changed, so perhaps we just need the latter?

what about margins? Rtb doesn’t have a left margin but it seems to have an indent feature which can be applied to then entire contents if you use the selection. It also has a horizontal alignment feature that allows each line to be individually centered. I don’t think that’s what I want, we desire left justification, I just don’t want the users constantly adjusting the print alignment in order to move the spine label over 1/16”.

I believe that our existing bit map routines can tell us our enclosing rectangle and with a little modification tell us how many pixels off center that rectangle is.

The spine labels probably should center (as a block) and the pocket label might just want a little extra width to look less crowded.

so how would this play out for the user? we could have 2 options for each box. The spine label would most often be set to BlockCenter which means of course left justify but use the graphic to specify an indent amount and put that in. we might do this after every text changed.

the pocket label would be a natural for BlockLeftIndent mode where the user can specify an exact setting using a trackbar, or there is some simple formula we can apply. it might be like center but more of 25%/75% split or something like that.

tasks are of course to see if our graphics can provide that information.

oh shit. I am looking at our graphics commands. when we wrote this for a plain textbox we used something called drawToBitmap in order to get the pixels into an array we could traverse. This is not available for a rtb.

so I might have just found some api code that will do the trick. but why did we switch to the rtb? Basically I got tired of going into the designer code every 2 minutes to edit the assignments of the NonSelectingTextBox objects.

since that time I also have used the scrollbar feature of the rtb to help with fitting and the red background. and now I am talking about indents. Are these features available in the plain jane tb?

**March 28, 2014 Log Hill**

So I think on the home stretch with spine labels. if I can finish, test it, make a brochure and post it on the IUG website then Amy Sieving will be in Detroit at the IUG conference to talk about it. Wish I was going, frankly. maybe next year…

Okay so one thing is I decided to junk my ‘updates’ code and figure out how to use the standard deployment options. There is something called one-click that sounds like all I would need. Otherwise just tell folks to put the program in the right directory.

I wrote a little printer alignment thing that works pretty good, but I think I want vertical axis alignment as well. it seems like we have always been sending our image with the vertical axis at 0 (top of the page) when I guess by measurement we would want something more like 5 or 10 pixels down.

Struggling a bit with the autoAdvance and the two modes of operation (Amy vs Oak) I don’t want to lock anybody into one mode or the other, I just want the controls to all play nice together. but if you scan a barcode or type in a title and hit search wouldn’t that suggest that the following print maybe should not invoke the autoAdvance? I suppose that if we have a good way of showing which rows have been printed then that is not such a big deal.

which leads us to how do we show which labels are printed? There are checkboxes available but maybe I can do something with the row color. Is there some way of using the checked state of the checkboxes without displaying them? the checkboxes seem designed around the selection set thing. we just want to select one row at a time, but checkboxes allow a complex line by line set construction. like go through pick the ones you want and make your purchase, that kind of thing. I am not ready to get complicated that way with select 6 random rows and ‘print selected rows’ Fact is this program is all about being Leonardo with each and every label, doing the review and pressing print.

so…what about changing the font or back color of individual items?

okay, what about a step back. there is a control called the datagridview that might be a contender as well. we load our csv data into a data Table, and then bind that as a data Source to a datagridview control.

what are the display options with a datagridview? does it have some kind of click and select a row? does clicking just select the individual cell that you are on? I guess an event handler would extend that selection to the whole line and fill the boxes. Since each cell it appears can itself be a text box or a button or a checkbox or a listbox, that boggles the mind. Imagine if every row was actually a printable size pocket and spine label..interesting. or just two columns, call number and multiline data. Okay, but I don’t think that’s what I want. the user should see the familiar half dozen rows and maybe even disable a row if they decide they don’t want to include that in the pocket label. okay, that’s a new idea that is probably also do-able with listview. Is it something a user would want? sure let’s say they have 4 rows but normally don’t need to print one item.

data grid view

[AllowUserToOrderColumns](http://msdn.microsoft.com/en-us/library/system.windows.forms.datagridview.allowusertoordercolumns(v=vs.100).aspx) this is as easy as setting the property true and the user can drag the columns around. Now of course we might want to inhibit that for the first column, which we think has to be the call number…right? so is there an event? ColumnOrderChanged?

looks like columns can in fact be hidden, but there is no built in button to do that. CodeProject has a neat little popup selector tool that some dude wrote. I wonder if right clicking on the header cell and using a context menu is the more logical way to go.

how to change the color of individual rows: <http://www.codeproject.com/Tips/51128/Change-individual-DataGridView-row-colors-based-on>

so we might have a hidden column a Boolean called alreadyPrinted. This would not display but we could refer to it. In fact…can we just change the color after a print job and that color will stay put? no need for a column of data? Looks like it’s time to put some data into the gridview and see what happens.

so..is there some sense in wanting to declare some kind of abstract dataset, which has associated file methods? Surely we think that our data is coming from disk. How else would it get produced?

databindings is a property of controls and central to the ADO (means ah don no). A dataSource is specific to the datagridviewer. a datatable seems like a pretty complicated thing. I think we can assign an array as a datasource. so I guess I need to create that array before reading in the file. What can I do with the designer?

**Sunday March 30, 2014 Log Hill**

Okay I got the dataviewer working. I just went with direct load, I don’t have a bound data source like an array even though that might probably be the way to go.

I realize that I dropped something when we made this change: In the old version I recognized title and removed anything after the semi colon.

which leads me to a real pain in the ass…my regex fails when there is a text qualifier and there are subfields. here is what it might look like: “fic smith”, “gone fishing”;”fishing for fun”, “123000”

you see the problem? it’s a bit tricky to see which is the delimiter and which is the subfield delimiter.

fortunately we have the header row, which never contains subfields. so we should be able to extract the field delimiter, and then figure out the subfield delimiter.

better add subfield delimiter to the settings and controls…that was easy. what about the regex?

well, since one of our formatting options for the pocket label is about to be FirstSubFieldOnly, we are going to want to figure that one out. But of course there is a disconnect. the regex parsing we are talking about is between the file and the dataGridViewer. So in Excel we would typically have that problem of normalization…do we duplicate rows in order to have single valued cells? or just lump it all together with the ; separators? The dataGridViewer does allow us to use a multiline textbox in each cell, or even I think a list box.

Let’s keep this simple. Our old parser threw away the extra values during the file to listView load, and I think that is probably the way to go. And keep that separate from anything that really is pocket label formatting, like InsertBlankLineToLookGood. We are talking about an option in the Data Menu.

btw, this project is kind of a test of the menu system. I might have chosen to create a little popup form but it’s cool to see if we can just do our options this way…particularly since there is a decision tree we are exploring…

okay, what about subfield discovery? is it true that without a text qualifier, the subfield is ‘undiscoverable’? That is an interesting thought. My brain and eyes would discover the subfields by looking at a field we know is a variable field and identifying the subfields directly. So I know a title when I see one? I certainly know a barcode when I see one. Titles and Author names too often have commas and semicolons embedded. So I guess our discovery mode which we call SpecifyFirstColumn will still need a hint about the subfield indicator. or I can just assume ; but why not let the user specify?

okay, now got a million or so controls and flags…time to integrate them with the actual regex.

so…let’s say the user specifies the DQS (delimiter, qualifier, subfield delimiter) directly and they are wrong. Do we bug out? or try our best the other way? I guess a message box giving them that choice. The message box could even update the control settings to match the file at hand. wow!

okay, lets figure out the regex. we examine the header row. we fork on whether there is a qualifier or not. No qualifier is simple and I think our code is OK. all we have to do is add the defaults instead of just assuming ;

but if we are using quotes, then we need to replace our token, which was simply (?<datum>[^”]\*) with something a bit more sophisticated. We don’t know yet what the subfield delimiter is. I guess all we know is that it’s not the same as the main delimiter. fuck I hope not. So first of all, I think we want to explicitly recognize the subfields. Yep, capture them in their own groups!

actually, two ways to go. we can capture the fields and the subfields in groups, and then use the algorithm from Marquis to arrange the contents of those groups into some kind of nested structure based on the their (index, length) properties. alternatively we can recognize but not capture the subfields on the first pass, but generate an additional simple regex that will extract the subfields from single field substrings.

string qualifiedToken = @"({0}(?<datum>[^{1}]\*){0})"

.WithArgs(Regex.Escape(textQualifier), textQualifier);

string pattern = @"^{0}({1}{0})\*$"

.WithArgs(qualifiedToken, Regex.Escape(delimiter));

you see our qualified token is a bit too simple. I guess it will do as a qualified subtoken

string qualifiedSubField = @"({0}(?<subfield>[^{1}]\*){0})"

.WithArgs(Regex.Escape(textQualifier), textQualifier);

string qualifiedField = @"^{0}([^{1}]{0})\*$" // poop grab the SFD

.WithArgs(qualifiedSubField, delimiter);

string pattern = @"^{0}({1}{0})\*$"

.WithArgs(qualifiedField, Regex.Escape(delimiter));

**Monday March 31, 2014 Log Hill Mesa**

making progress. I think I have the essentials of the more complicated regex test. by the way, these are not subfields that are causing the problem, the correct term is repeated field. okay whatever.

so my code tests the user settings and there are 2 constructors for our parser. one takes the 3 needed strings,: delimiter, qualifier, RFD. So that’s pretty easy. There is another constructor for the parser that takes the first column name and a default RFD. That seems to work pretty good too. Now of course our fields are showing up with quotes and shit, we need to implement the filtering stage. This is where we will clean out the quotes and figure what to do with our various repeated fields.

Meanwhile though, I notice that our routine can only load a file once. After that it seems to trip out. Is that because our dataGridView has some kind of problem? I got a bit hung up trying to clear it out or replace it, but that isn’t helping, its hurting.

Also…maybe it’s time to create a simple data structure and bind that to the dGV instead of populating it by hand. Why? It might be useful for the filtering stage, but more importantly I think I am going to like have to take multiple runs at parsing the file.

Suppose the user chooses the fixed DQR mode. She supplies the delimiter, qualifier and repeat delimiter. Very good but then she tries to load a file that her co-worker prepared and he uses a different signature. The file wont parse. Should my program then try the alternate method? attempt to discover the correct signature? If we want to play around with any of that its helpful to work in a simple array structure or list without having to mess with the dGV until we are done.

Okay…well what are we going to do with the repeat fields? we think we are going to swipe a routine from the marquis that takes a set of regex groups, and assembles their captures into a linear list of named entities. So we would take 2 capture groups [repeatfields] and [fields] and sort them into a list of named fields. This allows us to then populate a data structure:

List<goobers> myData;

class goobers  
{  
 List<List<string>> row

}

well some kind of thing that will hold a list of row data. Each row data is a list of fields, each field is a list of repeated fields. Also it seems like what Sierra pulls out as a title is a mishmash of stuff? or is it just the 245 field verbatim? this shit is so confusing.

well it comes down to this. repeat fields exist so we have to deal with them. As for filtering, I’ll find something simple that works for wpl…other users can let me know.

hey it looks like I defined something in marquis called a superRegex. it’s a decorated version that contains a ToNamedCaptures() method that is just what I am looking for.

so I guess my Regex in the CVS parser should be a superRegex..OK

**Tuesday April 1, 2014 Log Hill Mesa**

I think the regex is better now. I want to change my horrible menu tangle and put a tabPages control in the main upper space. 3 tabs: the main page (preview?), alignment, and data options.

right now trying to get a special rule in place for the marc tags that show up in the header as “245|a” and can confuse the column count when Andy uses bar as a separator.

April 9, 2014 Log hill mesa

so I have a frustrating problem with the user settings. I did something last week with renaming some of the user settings. I’m not sure how, but this lead to my having an obscure little file called Settings.designer.cs replaced with Settings1.designer.cs. This doesn’t sound like a big deal but I think it’s the cause of me now having to have the application config file always present next to the .exe file. Since I have been touting the easy of “single file Xcopy” installation, this frankly is a pain in the ass.

There does not appear to be anyway to undo the change.

**November 20, 2015 Telluride Colorado**

I've just tweaked the layout to fit nicely on our older laptop screens 1366 x 768  
And then I've gone and moved the print/search buttons to the left side.

I think I want a set of controls that set the Center vs Left Justify feature. right now that is a right click menu command, but I think I want it more like the font control, it stays in effect. Now this could be a single control and get switched about

like the font controls in the middle. 3 Radio buttons? at the heart of things its rtb.SelectionAlignment = HorizontalAlignment.Center;

so I'm sure there is Vertical Alignment too. What we want is like the Word 2010 control which is a grid of 9 little icons! well…maybe. RichTextBox.TextAlign takes on HorizontalAlignment values: Center, Left, Right

okay so I'm just adding it to the right click menu. but it seems like paste and special paste reset it to left align. So I think i want the users choice to persist. And since it’s not in a control, it’s a thing I must extract from the textbox itself..OK

Hey deleting the final character in the text box also sets it to left align. do I need to trap this?  
okay maintaining this is getting complicated because we have lots of ways to assign text and then to make it really hard we have the portrait/landscape issue to deal with as well. If I change orientation of the spine label do I want my choice alignment

choice to copy over? or maintain its own identity? What’s easy?

**December 1st, 2015 Telluride CO**

Oak notice an annoying behavior with the program. We had a big file loaded with all titles and Pam the volunteer was working her way through the stacks. Everything worked fine as long as she had focus in the search box. but as soon as she put focus somewhere else, the auto advance feature kept walking her senselessly through the list.

* duplicate the bug, understand it completely
* how many places can there be for focus to reside? Is there a no focus state?
* I see right away that there is a good display tweak as we jump between the two rtb edit windows...but when I click on the
* listbox the last editbox stays highlighted...that’s part of the problem.
* when I click the autoadvance button, focus shifts to it or perhaps the groupbox it inhabits.
* so can a container and a control within it both have focus?
* if every control has a onfocus and focusgone event, managing that will be a nightmare. We could write a central function
* that would at least report which control was invovled...
* what about tab order? How do we edit this?
* the program should distinguish between and empty and full list
* hey and we've got a ton of different tabs, each of which interacts with the print button and the spreadsheet in a different way!

Almost wonder if the tabs should have contained the other controls rather than sharing space...that’s a big rewrite though! okay so with the raw data tab up and running, we have a textbox to view raw data, the spreadsheet and the search box/button. It seems that these should work analogously with the preview print page. so if we put focus back to the search box in one, it stands to reason we would do the same in the other. Now users will spend 99% of their time in previewprint, so that logic should take precedence. we've used text editors that change focus after every CTRL-F and sometimes that is obnoxious.

data settings page should disable the print, the spreadsheet and the search. We should think of the search as being in the same group as the spread sheet. Is that something we want to actually do?  
so we are not going to move controls around, and actually the spread sheet is disabled in data settings, it doesn’t necessarily show that way however. so maybe that is a tweak that we do around the spreadsheet? okay when the groupbox's enable state changes, we capture that in one handler and set the datagridview's color properties. seems to be OK.

so if all we had was the raw data view with the textbox, the datagrid and the searchbox, we could expect one of the following:

- enter text in the search box and hit enter. if found we could either stay in the textbox, enter the dataview, or enter the textbox

and what if it isn’t found? well it should stay in the searchbox, and deselect both the textbox and the dataview. So what would be the point of entering the dataview? only to scroll around or right click...doesn’t make sense to me we can enter the textbox.

if you think about how excel works, they bring up a floater box and it keeps focus, so that Enter presses the FindNext Button and plain text goes into the searchtextbox. I'm not ready for the floater, but I think the principle still stands. Of course the nice thing about the floater is that goes away, so you know your just clicking in the worksheet...

so I think that means that when the user enters the search box they stay there until they leave. Highlighting the found text should be all we need to do. The user can leave and go to the datagrid or the textboxes.

so there is where we go, what things look like. Enabled and focused are not at all the same thing of course...also Print button wants to return the focus to some other place...needs a memory

so do I need a pseudo variable which takes on a few values  
ListMode = EmptyOff, AutoAdvance, RandomAccess

so when the user hits Search we need obviously to respond if not found. But if it is found, we stay focused where we were, in the search box.

its Print. Can we always return focus to where it was?  
CurrentFocus = TabControl, GridView, SearchBox

okay am i making this harder than it needs to be? The print button itself does not take focus, so focus is either in our textboxes, the searchbox or the datagrid. Can we just leave the focus there? How bad would that be? and then just figure out the list mode. So if the list is empty we should not have the search box enabled.

okay how often do I need to check and update that? I guess every time the datagrid changes.so I think part of the problem is this. if we are focused on the datagrid, well currently we can delete rows but we cannot change data or add new rows. So typing text makes no sense, it’s just ignored. but the arrow keys work, so we do have focus. But this is what happened to Pam, her text was going nowhere but the enter key was doing something. Enter key does a down arrow. is this some kind of default action? Sure and it’s what Excel would do in the same situation.

**for tomorrow**

* perhaps we should store what the user types into the list for future use.
* we don't need a special setting for the first file to load, that is accomplished with a shortcut and command line arguments
* command line argument for autoadvance mode? Or can we just detect how the user got to each line in the listbox? So if I click
* in the listbox then autoadvance is on, if I search for an item then autoadvance is off? hmmm
* get an icon for this program, sick of seeing the generic windows form!
* resize the form to just have the left hand column

**Wednesday December 2, 2015 Telluride**

So watching Pam this morning our setup really isn’t so bad. if you have focus on the searchbox, scan a book, then it finds the item and brings it up in the edit boxes. I'm not really sure where the focus is. If you then mouse around with the font buttons and then mouse over to the print, the program puts focus on the search box and highlights the text. Really it’s pretty cool. It would be nice to know where focus goes when we search.

The real problem is that Oak has her going into the spine label boxes and adding blank lines and spaces. We really need some way to let the user quickly select a top and left margin. If we can put that into our Janus controls so that they can be used with spine or pocket, that would be awesome. Some kind of 4 way arrow button would be nice. okay well the problem is that the rtb doesn’t have native padding or margin properties as implemented in C#. The actual windows control underlying the C# implementation does have those features, so there are some advanced techniques for processing. But for us there is no dice!

well fuck, can I automate the padding with chars that Pam and Oak are doing by hand? Its cheesy but if I can add some very small fixed font spaces and new lines, perhaps that would work. Of course that’s going to cause a nightmare when we go to resize text...or at least require a procedure to scan the text line by line, select the text that follows the padding, change the font, etc.

okay well a better kludge would be the standard fix: place the rtb inside of a panel with padding. Set the rtb to Dock.Fill and then play with the Panel's padding properties. Most users make the background colors the same, we might enjoy having different colors to make the margin explicit. okay well we already have rather specialized stack up of panels and overlapping rtbs...don’t we?  
Panel Spine Rotation is AutoSize=False, Padding=0000, size=285x285  
RTB Landscape is Dock=None, Location=00, Size=285x190  
RTB Portrait is similar, Size=190x285

So cool...I don’t think we need to add anymore panels, just use the one we have. Of course we want separate values depending on which orientation we are using.

Now of course we will have to update our SendToZebra() functions...but that shouldn't be too hard...it’s part of the arithmetic of scanning through the pixels...can't be too hard. Nice if we could read the panel directly? Well how does it work now? Well it’s called AddTextBoxGraphicCommand() and it takes an offset of some kind already our printer offset function. So that’s good news and bad. Good cause at least one margin is already programmed in, bad news because it means I have to get off my ass and test the print driver horizontal offset functions to see if they work.

The other thing it does of course is rotate the image and so that is always a bit confusing to say the least. this breaks down into 2 parts:  
map = bitmapExtraction(RTB, rotate?) and then separate  
add\_graf\_command(offset, map)

and the offset is already a true xy point, so that really is looking pretty easy.

where does the offset come from currently? it’s our printer alignment control which is simply a pair of panels inside another panel called paperSpace. so really...this is all pretty well configured. I mean we might keep the printer alignment and add that in when we print, so that we can adjust for hardware...and then have padding in the editboxes, and add that in as well when we print.

And you know it looks like the Zebra doesn’t really know quite where our Pocket labels start, it’s my program's job to add that information. so maybe the down low stuff really is ok.

so a test of this would be to tweak the graphic commands to add in the Panel padding. Then we need a hack to actually tweak the padding in a way we can recognize. Note that currently our portrait spine box is over on the right side of the panel

Pam also requested a CTRL-P function. I might implement this as ALT-P since I think I can just name the button with an asterisk or & sign or something like that.

okay well yes and no. I think we will be adding more panels...we want our construction to still clearly show the orientation of the label. That works well when the container in the Panel Rotation has nice bold border like Fixed3D and is white on the panel's gray background. so what I've done now is just pushing the label around in that gray space...might work arithmetic wise but give the wrong visual.

so we already have a lot of variables. List em again:  
panelSpineRotation  
richTextBoxLandscape  
richTextBoxPortrait  
SpineBoxVisible  
SpineBoxHidden  
richTextBoxPocket

okay so the pocket might be the place to start. It doesn’t have a panel, it’s just itself. so we would create a panel the exact same size, give it the border stuff. let’s add 'panelPocket' and set its size, location and border from the rtbPocket

Note that the margin property of a control refers to minimum space around its outside relative to other controls in the same container.

okay that wasn't so hard. the background of the panel and the rtb change as focus moves back to the spine label, so that was some tweaking. we should be able to duplicate this with our 2 spine rtbs, put panels behind them. we still will need a control for this, so far it’s a cheesy button.

**Friday December 4th, 2015 Telluride CO**

So I had this whacky stack up of panels and separate textboxes to handle the spine label rotation. I've simplified this so that we have a panel that holds the rtb, just like the pocket label. We are not using the padding, but directly setting the rtb's Location and Size properties.

For the rotation I just have a button that resizes the panel and rtb. And actually it resizes the panel but not the rtb, so it’s kind of not quite right! It probably would be if we used the padding and fill approach...but I don't think I was happy with that. so I guess I need to explicitly resize the rtb to fit. And keep the margin data as Settings and probably a separate set for each orientation.

Or...what about automatic margins? what I mean by this is that we would measure the size of the text by looking at the textbox pixel by pixel, hopefully using the same routines we've already written. Then we would split the difference with some simple arithmetic and then set the margins.

so that seems like a bit of catch-22. we need the text in the rtb in order for it get rendered and see if even fits. But if we've got margins set in the rtb then that effects how much room there is. Do I need a hidden rtb to dump the text into and measure it there? Or some kind of graphics field? I imagine I might. Let’s look at how the existing routines work.

**Sunday December 6, 2015 Log Hill Mesa**

So I want to understand what happens when we print, it seems round about. And what happens during a Paint event because perhaps we want to handle that event ourselves. And I think I need to render the text into the rtb starting tight to the upper left corner, and then just tweak the location and size of the rtb within the panel. But this is sort of a catch 22 since the size of the rtb is what determines if it sets the scroll bar or wraps around…so I kind of need to put it into a full size textbox, compute the margin, and then offset and resize the box. Seems like a lot of screen flashing and such.

Print Events:

* we invoke this command twice AddTextBoxGraphicCommand()
* AddTextBoxGraphicCommand() does ‘bitmap extraction’ and then ‘add\_graf\_command’
* bitmap extraction creates a homebrew ‘mapp’ object from the control. Its complicated
* add\_graf\_command is pretty simple, it puts the mapp object into a Zebra friendly stream and sends it off.
* bitmap extraction calls my controToBitmap() routine. It then locks the data in the Bitmap object and creates a fullsize single byte per pixel image\_bytes array. Then it creates a really compact bit-packed mapp object of just the actual text rectangle.
* controlToBitmap() looks like something I copied off the internet. It seems to be working with floating point pixel information which is frankly bizarre…but I guess it works.

Okay well I guess it makes sense. It seems weird that there is so much conversion to get from the control to the bitmap, but from there I understand why we do so much processing: first to eliminate the 4 bytes of color information to a full size black and white, and from there compressing the data into bit-packed 70’s style Zebra data. I am curious, is the minimal bit-packed mapp object required to correspond to byte boundaries in the zebra? I.e. can we send the mapp to the zebra with an arbitrary pixel offset? or just those that fall on {0, 8, 16…}

so I know I can create a handler for onPaint even though its not listed in the IDE. I’ve done it before in Truckin to paint the tabPages tabs a special color. And I know that the handler can fall through to allow the normal OnPaint to take place. It may even require an explicit action to prevent that. But what worries me is having to duplicate some infinitely complicated procedure. And actually I’m not sure of any of this. The RTB is like a windows thing with a wrapper around it, so unlike the tab control I don’t think we can modify the paint calls.

we do think that TextChanged and FontChanged all sooner or later result in a paint call. So perhaps that has been the place all along to have our CrimsonFit() software running. And it does seem like that is pretty close to where our new code should go…right?

Can we recall the order of events when the user changes the text or font in the RichTextBox? Or is it really just the TextChanged() routine that matters? The MS documentation gives the example of examining the text in a balance sheet and changing the font color to red if the amount is negative. So we think this occurs before the paint call, and our current use is probably correct. Recall our textBoxOverflow() called from TextChanged\_Handler(). Essentially it returns the results of a call to .VerticalScrollBarsVisible() on the RTB itself. Does this work in one go? or are we cascading through multiple calls to the handler? In other words, when the user types a key and the Event is fired, our handler is invoked. At this time is the internal state of the control such that the .VerticalScrollbarsVisible() will return valid information? Or by chance does it take a cycle or two to update and our system works only because our Color change is itself trigger another event that our handler sees?

Maybe its time to log the events just to see.

But supposing all of this to be cherry. How does our automatic margin work? Does it get performed in our existing FontChanged handler? Or must the text be displayed with the existing margin and then recomputed? This really does seem to be a dilemma, since we must either A) keep our rtb squished in the panel in order to detect overflow and B) let it display in a full spine label sized RTB in order that we can measure it. Of course this timing is consistent with how the print button works…way after all textchanged events have been handled.

Okay I guess it could be a separate button but that seems like a drag.

So if the textchanged handler is called and that is where we measure the fit simply by observing the scrollbars and other highlevel information, what can we do there about centering? We can of course extract the bitmap and measure it. And perhaps we will need to move the measuring routine up a few levels, I think it now happens at the middle level. Fine. And then we can use that information to compute a new Location and Size for the textbox within the panel. But if the original image had been cut off, our recentering will likely be s partial step in the correct direction but not actually perfectly centered. That’s because the portion of the image that was not displayed now will be.

So maybe every TFChanged event starts by moving the RTB back to the top left corner of the panel. This will of course fire off the sizeChanged event. And they recommend instead see the Layout Event as being more general. What about it? well first off it allows you to suspend these events, while you reposition child controls and then Resume Layout. That sounds like a tool we may need. <https://msdn.microsoft.com/en-us/library/system.windows.forms.control.layout(v=vs.110).aspx>

So lets say that we handled the TFChanged by resizeing and relocating the rtb. At some point that triggers the layout event and we do what? If the image has already been rendered we can measure it. And compute the margin. And put that into effect. And probably have a beenMargined flag because the Layout event will have been triggered now a second time and we don’t want to keep computing the same margin over and over again!

so it looks like the layout event occurs after the text or font changed event. when we alter the font it looks like the event fires twice. Wonder why that is?

deleting a char only fires the text changed, not layout.

so when I click the increase font button we do get text,font,layout events in that order. why? well there’s nothing odd going on, we pull the font, increase it and assign a new font to the rtb. No hanky panky with the text like reassigning it. Okay well I guess the textchanged event just fires a lot. So probably I could get rid of the font changed…have we ever tried this? well it seems to work just fine. so maybe that is a minor victory

so I have 2 major I dunnos going on. 1) is our crimsonFit routine actually working properly on the first go around? or does it sorta work because we events multiple times? 2) do we need a real rtb and its conversion to a graphic object to determine the fit of our text? Or can we just fit in in a graphics object and be done? it seems like centering text shouldn’t really be that hard.

BTW, an RTB does have a zoom property, which we use in Marquis to good use. But is that also why the controlToBitmap looks to use floating point pixels? just wondering

so I found an autoscaling rtb on the internet that is kind of interesting. What we want is an autocentering rtb as a complement to it.

now doesn’t this have all the elements we need?

private void AutoSizeControl(Control control, int textPadding)

{

// Create a Graphics object for the Control.

Graphics g = control.CreateGraphics();

// Get the Size needed to accommodate the formatted Text.

Size preferredSize = g.MeasureString(

control.Text, control.Font).ToSize();

// Pad the text and resize the control.

control.ClientSize = new Size(

preferredSize.Width + (textPadding \* 2),

preferredSize.Height+(textPadding \* 2) );

// Clean up the Graphics object.

g.Dispose();

}

So the preferred size above, computed with a graphics object, font and text, returns to us the rectangle we need. So cant I just use this to set the Location and Size of the rtb inside the panel? And get the fitment information at the same time?

Monday 12/7/15 Telluride, CO 81435

So I think we have gone back and forth with TextBox vs RichTextBox a couple of times. I wonder if its time to go back to a plain old TextBox?

* Centering text works with both
* No Zoom feature in TextBox
* Control.DrawToBitmap() available with textbox
* One Color, one Font only in TextBox
* What about a right click context menu? OK
* Cut copy paste seem to be part of the textboxbase, so that’s cool

Well what do we think we would gain by this? An easier or faster or more efficient bitmap extraction.

Size stringSize = graphics.MeasureString(this.Text, this.SelectionFont).ToSize();

It seems like this is the critical line that we want to get working. Find a graphics thing we can use, measure the size, compute the margin and the error, and then set the Location and Size of the (rich)TextBox that we are using.

Is this so very difficult? Will it work with the RTB? Lets find out! Hey it seems to work fine. Seriously this does rather kill two birds with one stone. Overflow and automargins.

Lets try adding the automargins.

Marge = label wide – text wide / 2. Non zero not allowed.

Hey it looks like a rectangle structure contains a Location and Size, so that is a natural way to pass back the data. If location is zero, that almost always means there is overflow…or close enough?

Hey we should be able to get rid of scroll bars

Hey it sort of works if we drop one char in. but if I just start typing the box gets smaller and smaller!

**Thursday 12/10/15 Telluride Library**

Hey I just noticed that right clicking one of our editboxes doesn’t actually switch the focus over and move the buttons, etc.

And I got pretty far with it on Monday, enough to demo it to Oak and find that the manual control is a nice thing to keep but that the autoMargins will likely fill most needs. I googled that our dropdown items have an autoclose property that we can likely override. Lets take a look.

**Tuesday 12/15/15 Telluride Library**

I’m trying to get all these events to play nice together.

When we first click into the spine rtb that fires the enter event. There we call the setActive which really sets

Active and inactive boxes. This doesn’t appear to trigger any more events, but I’m not sure.

Okay that’s more sorted. Next thing is in manual margin mode, I indent the goober over and it never registers red.

So check the log and see if the update fit is being called? Yes, and textbox overflw is reporting correctly. So we just need to do the display c all

**Wednesday 12/16/15 Telluride Library**

Well I did get a bunch of stuff cleaned up yesterday, all the events talking nicely to each other. So I have the manual margin and auto margin controls nice enough. Think I was still having an issue with the font display below each textbox.

The font buttons are a bit funny, can we figure out a way to right click and let the user select a font and a button.text for that button? Ie, configurable and saved in our settings? That would be nice, even if our team here don’t really need it. Also I am thinking that for our pocket label it would be nice if the fillboxes routine would recognize “First line\nSecond Line” ie the handful of escape chars. We could get wild and have an .rtf feature. It would be easy to try, can I create my beautiful labels in Word, save as .rtf, copy and paste that text into a csv file? Maybe.

I had a thought abuot the autofollow checkbox. Toss it out, and make it the next button. So when Amy needs to print 9 labels she has to hit next, but that’s not a big deal, its easier than manipulating the rows of the spreadsheet directly.

And we could think about whther that next button might act like it does in a search box…where if there are multiple items found next takes you through the set of matching items. So our convention would be that a null search field (string.empty) actually matches all fields. Well matches all rows to be precise, we always want Next to take us to a new row if it can.

And maybe we need more attention given to search string not found.

Also Pam wants a keyboard shortcut. Which reminds me, if we make a shortcut for an item in a contextmenustrip, will my wutcontrol() method still work?

* How does the Enter key work? It needs to be a bit smarter about how it works. The form has a property called AcceptKey that directs the ENTER to the button we choose
* Something is broken in my new Log tabpage because ENTER always goes right to the form. Okay that the AcceptsTab feature of the textbox.
* Track down groupboxsearch.enable and groupboxdata.enable OK
* Select printer needs a wait cursor. Also we now display printer name in Print Button OK
* Restore Default Fonts seems unnecessary, at least until we have a right click configure option for each of our buttons.
* Right click programmable font buttons. **Right Click events detected, need a little editor form, settings.**
* Make sure actual print job has all the right arithmetic: margins and the printer alignment offsets
* Are we saving and restoring the autoMargin flag to disk? yes
* In portrait I am seeing MAHONEY on the screen but its not printing. And do we save and restore this setting?
* Okay, that is still happening. The author name is sometimes displayed, sometimes not.
* Flow of focus. After print, where should focus go? To the search box under certain circumstances.

**Thursday 12/17/15 Telluride Library**

So How should focus work? Its actually fine that we are not normally focused on the 2 label edit boxes, Amy and Pam will rarely want to actually type in to them.

The only place we really need keyboard text is via the scanner and that is in Pam mode. So as long as there is data loaded, is this where focus returns to?

**Tuesday 12/22/15 Teluride Library**

Oh my gosh, what a nightmare this turned into!

I got way into the pixels and the rotation and the border around the panel and kept getting very confused about client area and a host of miserable details…

But I think I am through the worst of it. I got the textbox finally resizing itself nicely in the panel. The trick was to compute the minimimalist rectangle that would fit the text and using that to set margins but not actually making the textbox that small. We make sure the textbox is always at least as large as the panel.

The low level routines did need carving up a bit in order to impose the panel as a limiting region from which to get pixels to send to the printer. That was the real problem because the underlying textbox would get massive and bleed onto the other labels. So now all the pixel routines are actually part of the custom control, and while I had some super tight routines I’ve now broken them all out and it’s a bit easier to mix and match.

Okay so focus was something I was working with. Now our DataGridViewer takes the focus but there is nothing you can do with the keyboard. So lots of times where you can type and nothing happens – not even a beep so you know what is happening.

**Thursday 12/31/15 Telluride CO**

So progress on two fronts. First I started mucking around seeing what it would take to introduce an emulator screen and it involves some low level graphics and bitmap programming but should be worth the effort.

Then I just now dove back into the Margins vs the Center/Left justify controls and I think I have been making some progress. I did just now do an experiment with the zoom factor and the mouse scroll wheel. This may be something we want to explore, as it could be a promising way to quickly resize the text, but for now I’ve commented it out and moved on.

So when we have centered text it no longer makes sense to adjust the left margin, either manually or in autoMargin mode. So the Center and Left commands take that into account. Lets make sure that switching the automargin checkbox also updates properly.

It does seem very flickery and when I look at the log it seems like there is just a ton of calls to the update fit routine.

I do wonder if we are getting events sparking events sparking events, etc. So one way I’ve seen is that the first event to fire turns off the other events.

**Monday 1/4/16 Log Hill Mesa**

Well I finally feel like I have remembered enough about classes, events, c#, winforms to get the program behaiving properly. I estimate I have rewritten every gui routine 10 times. So this weekend moved controls into and out of the custom class vs form level but I really think its getting pretty good.

I partly implemented the print emulator and I still want more control over formatting the column contents and having a 3rd textbox for display only – and allow useful labels to be printed. Also I want to play more with what happens when I paste complex formatted text in from MS Word – although for now I have disabled that. We played briefly with Zoom instead of Font Size for making bigger images, and that might be the way to go in the future, as it would allow the complex formatting.

Also I still think I need a background worker and progress bar to help with reading in large files

**Friday 1/8/16 Log Hill Mesa**

Getting back into the emulator. I think I need to figure out my zebraski/mapp classes a bit better

Create a class called CompactBitmap that is our byte[,] format. Each pixel is a byte, 0 or FF. we need a constructor taking a real windows bitmap, and some methods like clip and rotate, and ToPackedBitmap.

PackedBitmap{} is basically our existing Mapp class. It’s a 2D array of bytes where each byte holds 8 pixels. It may describe a larger region than the CompactMap it came from. It has a ToBytes() method that just collapses the stream in the way we need for Zebra. Constructor that takes CompactBitmap is all we need I think.

Okay I’ve a nice pair of classes, CompactBitmap and PackedBitmap. Constructors to go back and forth between them and to derive from windows Bitmap.

Next is our Zebraski, which we implemented as a singleton pattern. Any good reason for this? I think it should be instantiated each time we print so that in theory at least we could keep it around for emulation. Maybe this is not such a big deal, I don’t know. We don’t need any of its static methods, do we? At any rate what I have in mind is that it contains the intermediate data that it was passed, so the arguments to the graphics command which of course we think are just these PackedBitmaps. I’m saying each instance of the Zebraski would have a list of these. Certainly it can generate in theory this list from the byte[] zebraCommands, right? so keeping the originals is a way to test our software as we go?

Lets think about this differently. I was thinking I would save the compact version that panelBoxes uses and start by emulating that. make sure it looks good. Then emulate the packed version. only then try to extract packed from the commands and work with it.

lets think about something else. PackedBitmap has X and Y fields, CompactBitMap does not, so I wonder if it should? how often does it drag around x,y information with it?

So I upgraded the compact bitmap to contain a locaton point field. seems like a good idea.

okay so I am confused about the rotation issue. if we are in portrait mode there is no rotate, but in landscape there is.

is that correct? well here is the panelBox.RotatePrint method does

public bool RotatePrint { get { return alignmentPanel.Size.IsWide() ^ OrientLandscape; } }

not quite what I excpected but I guess I remember doing this.

so spinealignment is not wide, pocket alignment is wide.

spine landscape rotate print = true

spine portrait rotate print false

pocket landscape rotate false

well sure I guess that works. and I guess that’s pretty smart. I am not sure that if we filled in the 4th case our rotate would be correctly I am thinking it might be in the wrong direction!

Okay but it seems like library labels needs a bitmap from the panelBox and it gives us one already nicely rotated, that’s what the DrawToZebra() method does. And it comes with an xy offset relative to the upper left hand corner. So why do we need to do more calculation about the location in librarylabels except add in the alignment panel location? (forget the fudge factor)

Shouldn’t panelBox just give us a located bitmap that we can use? seems like it to me! I see that panelBox encapsulates the alignment panels, that makes it easy to do all the calculations in one place, even if that place is outside the control where we could easily reference it ourselves, but since we have many panelBoxes its nice to keep it organized.

here is what is weird. DrawToZebra first computes a location. Naturally this starts with the panelalignment location

but then it adds in the value from pb.PrintableArea\_PanelClientCoords, and maybe rotates this point!

the sum of these two points is then what we stuff into the compactbitmap before sending it to the graphics command.

what does this method PA\_PCC() return? Basically it’s the textbox.Bounds trimmed to fit the panel, since the textbox can extend underneath the panel we don’t want that.

Now there is also a tweak for the border. I know I worked hard on the border but I don’t actually remember what I did or how its supposed to work. As I recall the border comes out of the panels size. so a 100x100 panel has a useable area that differs depending on whether the borders are on or not.

Seems like I would have arranged things so that the text on the screen wouldn’t change when we turn on the border, perhaps by expanding and contracting the border I dunno.

what is the textbox usable area? does this change when I turn on the border? I think it must.

So why did we do this? I think I could create a panel that had no official border but OnPaint could draw a rectangle around the outside. Or resize the panel so that its client area stayed the same.

so I guess this begs the question, are the textbox coordinates relative to the panel or the panel’s client area? well our panel doesn’t change size, but its client area does and we see the textbox visibly move toward and away from the topleft corner. and the textbox location doesn’t change.

I think if my textbox were dockfill we would see its size alter.yes we do. So what the fuck did I spend a whole weekend doing? I think its clear that we want the textbox to be located in a clientrectangle that is the same size whether there is a border or not. So the textbox bounds are really just fine I think if we can resize the panel. And honestly I thought that this is what I had done. Okay and it was broken (ever tested?) by the use of the Bounds.Inflate() method. the fact is that Bounds is a property not a field, and the thing it returns, Rectangle, is a value object, not a reference object. and when you use a method like .Inflate, the copy is properly inflated, but the original value in the conrol is not effected. This is subtle and not flagged by the compiler. makes we wonder if I have more of these lovelies hidden in the code. So now the panel adjusts bigger and the client area remains the same.

I still wonder what I need the PA\_PCC() method before. Cant we take textBox.Bounds and limit them to the PrintableArea in PanelCords? sure, the question is whether we need to check the border active state, I think ti should just work all the time now. actually, I don’t understand papcc at all. we have the textbox compacted and clipped with xy relative to the textboxes original clientrectange, which is now always the proper size. so just use this? really I don’t think we need the papcc at all?

sure it seems simpler without it. fixed the borders bug and now that is good. Panelbox is doing it all now, including adding in the alignment offset, since it encapsulates the alignment panel control. got rid of the silly fudge factor. I think the double dose of offset may have been causing problems that the fudge factor was offsetting, lets fix the bug (done) get rid of fudge (done) and see what comes out on Monday when we have a real printer.

Now I do think it would be nice if we could emulate not just the minimal text but also the outline of the panel’s client area. So how hard is that? we already have the panelAlignment and likely that is all we really need to paint.

So can we just queue up a compactBitmap from panelBox.DrawAlignment(e.graphics) or something simple like that?

Even easier to just FillRectangle directly onto the graphics. Now in fact I wonder if I could just project the images onto the TabAlignment paperspace and be done with it.

Okay, somethings not quite right. My automargins doesn’t seem to be printing correctly. Lets go back to manual margins, there really should not be a difference. Hmm…I thought it was working a second ago, did I do anything to fuck it up? all I did was add the panel rectangles, and they look fine. so okay it looks to me like we are maybe creating our first compactBitmap – the one from the fullsize – without supplying the textBox.Location?

well yeah and I was pretty insistant on doing it this way – that the bitmap would be noloco. But I think that has to change

**Sunday 1/10/16 Log Hill Mesa**

Okay so I have an emulator tabpage, and I just added an programmer only emulation mode. There are times I need to be in the mode when I am working on the program, so I will change the switch in VS and recompile.

But I think I also want to be able to test where the bitmaps are going when other people are using it. So I think it might be fun to have a button in the alignment tab that lets me see the emulator.

So maybe what I really want is just PrinterDisable as a switch, and a permanat seeEmulation solution that can be done n the existing alignment tab.

So I think that means I need the emulation panel to be a child of the alignment tabpage. either the panel paper space and it can be swapped in and out as child controls, or hopefully just be made visible and invisible.

Friday 2/26/16 Telluride, CO

Small change for today. I just wrote the DNA query that can create ‘TLW ALL LABELS.TXT’ file. Only one small issue, which is that this approach renders the call numbers with the subfields in place. This is actually a pretty good thing, and my Library Labels program can deal with this in its special paste routine, now I hope its easy enough to put it into the loadboxes routine as well. Lets see!

Okay well it turns out I already had made that one line change, and it was working fine. I do have a problem with the query, it has so many joins that it takes forever to run and timed out on me!