**pbt / Design Notes**

This file ([**repo ver**](https://trivedienterprisesinc.github.io/ui/2024/form/gridx_Styling/tasks.html)) created Mar 6 '25 to test + work on pbt-related updates *Last updated Apr 04th '25*

* pbt 4 wins
* Expr parser: excl Fields we can compl rest of the exprs (curr tested w/ 'x > y')
* meethoo updates [Appearance(icon pic+btn + IconLbl editable) | FldSetup | ImportData (CSV) | WebClient (see under Table|CSV|Settings ]
* Convert tasks to ToC (generator)
* Update pbt article add vdos
* We also nd to get snap of DzDv
* 4 internalFlds; instd of '\_' use ("brj\_" + fNm)
* 4 internalCounters (tbl006, fld003) store in Def (intnl, init 0) & increment on member .getNxt() Also in the same place store stateVars like dskChanged (triggers send2svr + snapshot)
* Mar24@TBD: No dzDoc copy/paste in v1? (cpy indiv els within docx)
* Mar26@TBD: Is edHist mandatory w/core? Sep mod?
* related: isDel: is this a sep mode w/editHist? Any pluses?
* @rsch: SubForm notes/tks

Mar 13th Control, we have **redundancy**

(about 20%, inspired by [this](#redundancy))

* Basically all webCli mods don't need dsk equiv.s (Form, Dv)
* By now all devs have a browser win open
* If Ozzie was blding now, wld he have included these? Poss not
* (they were legacy pre late 90s before the move to web; when all users used dsktopClis)
* So unless we launch user desktop clients there is no nd for the dzCli having these mods:
* the webCli has full Functionality
* Poss. eventual utility in releasing user dskClis (say for security purp.s);
* we can evaluate then.
* This reduces considerable effort incl windows valid8n.

**NEW approach to state**

Desktop + Others:   
Instd of CE(wld), ea el's win getsFromBackend + keeps its own state   
Therefore ea indiv state is auto-disposed on winClose   
openWin fetches state as part of handler

Bld API coverage tables 4 All Modules

**dskCli: Desktop**

Mar 6:   
 Move all pbt stuff to sep plk   
 Merge code diffs 4 openWins   
- OpenDb/Dz checks (4 now) ONLY Menu Model updates   
- For actual stubs (members) the model becomes:   
let (\_, str li) = m   
  Lookup api for mdi.getTopWin().Title()   
Mar 7:   
  Dsk Ty: Add refcell 4 StatusMsgs + member showStatus(t) which shows txt on bar + appends to ref; barDblClick >> txtDlg() [for relev loc srch "Ready..."]   
Mar 10:   
Also, devState needs nextTableID; this belongs 2 OrgDoc/AdminTbl   
Something like (5 demo dbs) CustID\_Table006 ...   
Add stubs 2 AddIcon() to show selDlg with tbls 4 curr org + editBox below lbl: "Create New" titleText: "Please select an existing tbl or enter the title for the new Table (spaces allowed)"  
Plus checkBox 4 "Create new Copy"  
@TBD In Prod, shd we relabel AddIcon() to AddTable()?

Win SubMenu

- MDI: Find dox/links; impl barebones funct.   
- Determine if we give em min/max options   
- Curr MDI child gets tick mark on MenuItm   
- 1st choice is alw Desktop   
- Impl/chk switch mechanics   
  (Already exists) OpenDb gets DbTitle (lbl); OpenDz gets ("Design Mode: " + DbTitle)   
?? @TBD:   
For OpenDb + OpenDzVw tests, create a form w/a snapshot for the tests   
- Add stubs 4 OpenDb | OpenDz | switchWin

Login/Init()

@rsch: Loookup l8st ProgBar snippets   
- State provides prior Dsk snapshot (or default initial)   
- DlgBox "Login successful, now loading settings..."

async {

do! Dsk snapshot FadeIn @ 50% -> 100%

do! Show progBar

do! Cycle thru loading msgs }

 Cre8 loading msgs (last: "Restoring Open Windows...")   
["Downloading Desktop settings..."; "Downloading Desktop icons..."; "Downloading Table settings..."; "Setting up Developer Environment..."; "Configuring Developer access to backend..."; "Downloading Table marked records..."; "Restoring Desktop state..."; "Restoring Desktop windows..."]

Logout()

- Trigger: File|Exit   
- Takes a snapshot of curr dsk (if necc)   
- Saves State (openWins) + snapshot locally 4 nxt login   
snapShot initially set 2 default; we'll nd: ref cell DskChanged = false; (Mar20: nix, user may keep cli open forever. So trigger on ea change, poss BOTH local/2svr.

**dskCli: TableDef**

 Loc8 l8st Meethoo comb thru logs/notes 4 "DFldString" et al.   
- The model will be a tblDef w/valid fldTys (gen |> from ty)   
- Incomplete fn.ality in tblSetup: choose Mods; Default dv/frm...   
- Incomplete fn.ality in fldSetup: schema rollbacks/fldTy change propag8n   
ex. Change int 2 currency (or str2int) >> Dlg "XX recs fail. Gen qry + update before we can continue..." BOTH broadening & narrowing   
- Move all Tbl-level Settings here incl   
CSV Import btn   
Btn 2 import custom css @TBD Scope for err: instd allow colorOpts etc. via btns?   
+ve for customCss is it allows devs more control; -ve is support calls if they don't know/try to mess with things they're not supposed to.   
Btn 2 gen webCli code   
- Tabbed Vw (as is) 1st tab Appearance (w/Icon pic + btn / IconLbl editable + DarkScheme + General Settings (locationCode/Lang/etc since it cld be diff for diff tbls.)   
SO: .NET 9 (preview) has introduced dark mode for the entire application see [this](https://learn.microsoft.com/en-us/dotnet/desktop/winforms/whats-new/net90?view=netdesktop-9.0#toolstrip)  
This req.s cli restart, see this:

// Get the current executable path

string exePath = Assembly.GetExecutingAssembly().Location;

// Start a new instance of the application

Process.Start(exePath);

// Optionally, you can perform some clean-up operations here before exiting

Environment.Exit(0); // Exit the current instance of the app

then tabs for **| Appearance | FldSetup | Import Data | WebCli |**

**CSV**

- Expand MusicTbl 2 incl all fldTys (incl dateOnly/currency) Loc8, use gpt 2 gen ~500 recs   
- Test is [Float -> Currency -> Float] == same model   
- Impl csv func. (modify existing tkImp)   
Chk csv spec (link exists) 4 usr provided params (fld/row delims, fld/row seps)   
In test embed incorr rows, shd behave as exp.   
- v1 gets NO custom delims; just standard + Dlg 4 errLog (selectable)   
- 4 test bld li w erroneous data; feed 2 fn; Model is <model, errLi>   
Mar 8th:   
v2: Parsec @rsch if exists curr.   
wobblyGen/Dat already converts the other way (fldTy -> fmt) We nd to repurpose that; split to ro |> lim (fun r -> split "," |> tryParse |> ToResult)   
@ToDo: For this we nd to pullout BrijTyCommon   
Mar 24th:  
mod csv\_Test

* cr8 csvErrLi = [(roTxt, errTxt); ...]
* varying # of errs (1,2,3); pick 0-8 items >> insert into spl
* ie, split2Li >> insert@rnd >> join2str
* InitialVal is the regular spl (no errs)
* Model is errOutput; Abstract blt from errLi; Act from fn
* Test members incl csvParse(); wHeaders(); woHeaders()
* Rebuild Mock tbl with actual names/images (Claud?) + patch rdm cols
* Mar 26
* Curr regex won't work for non-strings, nd 2 tinker
* Does DateTime use a comma?
* Does xl escape both commas + qts?
* Create sample MusicDoc + export via xl.

**dskCli: DvDef**

**dskCli: DzDv**

- The (apparently l8st) snapshot lists TblDefDoc + FldDefDoc; which we've combined; so nd 2 remove   
- We nd to add a categ/doc 4 CardVwDef below DataVwDef

**dskCli: FormDef**

Notes:

* the two above have been merged ('23-34 mainRepo lockd)
* Some work also compl. on selectedPnl (red bkgd instd o' handles)

Notes:

* Prior vers of this mod exist, chk past commits (bld\_v1 interleaves BlankRows)
* DnDTgt spans Row (see PostPitch Notes)
* this curr impl autoCasts to BMdzCell\_v2Struc
* as decided, changes via UI updates def & autoUpdates UI
* @Add: bld\_v1 interleaves dropCells(see output); nd to manually do that

Mar '25:

* The curr sys on drop will relayout subseq rows/sections; this is \_not\_ desired: it may spoil completed layouts. Instd: if drop causes overflo/wrap, do an li.inserAt & leave rest of struct unchanged (this means the tbl ros subseq have 2 be incr.d)
* Also beauc. deltas for FsChk modelling purps.

Apr03\_25:

* The curr BMAux uses fldDefs. We nd a fn (fldDefs -> BMFldDef) to gen the default cellStruct.
* Ren above BMfld to this + modify BMAux to use this \_not\_ the fldDefs (usrFlds above)
* Modify tbl struct to allow BlankRows + InfoBoxes (this exists somewhere) (poss ty changes nd.ed?)
* cld use existing BMdzCell w/txt as "" / info
* BM already contains \_colN\_; roN shd be autoCalc (add member) via lilen
* For \_Undo\_ we nd refCell 'StateChanged' + member 2 chk/set if necc.
* Add this call to the 1st line of every handler 4 changes
* + last line of handler saves state 2 BM list last (unless cancelld)

Note; the BM ref'll now store li not single, + pointer.  
For FormDnD Model chk [this](https://github.com/TrivediEnterprisesInc/TrivediEnterprisesInc.github.io/blob/837151d3c68f6b452917f5e9895de62ba799d885/winFrms.fs#L3433)  
There's also a DnD Monad; tested; [here](https://github.com/TrivediEnterprisesInc/TrivediEnterprisesInc.github.io/blob/837151d3c68f6b452917f5e9895de62ba799d885/winFrms.fs#L3675)

dskCli: AgentDef

**dskCli: Auth/State/Other**

**Lookup Masalo**

Mar24

* From (dz) will have LookupFldNm [ ]; Values (Pls enter ea on a newLine) + btns 2 sort asc/dec + Chkbox "Allow itms not in list"
* OnSave creates Masalo
* @TBD: Didn't ln offer bar-seperated Disp/Save vals? (CA|California) Any util?

**State**

- CustID   
To facilitate multiple devs/oneCorpID, CustID contains OrgID^CustID   
So to fetch all tblNms 4 2nd dev in one org we qry based on OrgID...   
- User.db: The webClient has only DV\_state   
for ea dv: TopRec|PgSize|ColWidths|MarkedRecs (Color)   
- Dev.db: Contains more   
DesktopIconDefs|AppearanceColorScheme|AppearanceFontSz|...   
- All State stored in tbls as:   
<CustID\_TableID\_ElID, SettingName, Value>   
- All State downloaded on login but ONLY for openWins; subseq newOpens get only for that win.   
(new Handler for userStateDnload: type 'Other')   
- Each State change saved/applied \_locally\_   
- @TBD:   
Mar12 auto added 2 payload   
OnSave (e.g. DvColWidth) >> StateChanged = true; run:   
stateHandler dzEl tblTy user winH   
this updates Dev/UsrTbl; pushes new state 2 winH   
new members: dsk.getWinState/put~

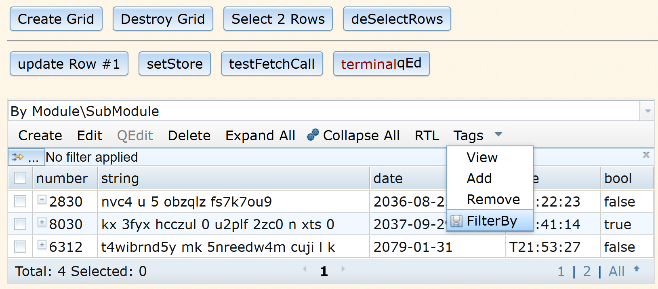
**dskCli: Embedded Tbl/Embedded DV (in Frm)**

* Mar25Loc8 dz notes for subforms, impl embedded dv
* Def will have to be separate; therefore new dzDocTy
* Allow summary rows (see oria)
* Mar26Just like PVs but no Qry (EdHist), SubDoc. DzVw allows ColResize/Choice
* Won't appear except on Frm (ins adds 100% wid cell; dev sets ht in px)
* Tbar allows adding mult ros; edit in-place, no switching
* Only one embTbl (like tbl) but allowed: multi-frm embeds (some cols here/there)
* In FrmDz allow embedding

**dskCli: PropBox**

Much work on PropBox complete, chk [this](https://github.com/TrivediEnterprisesInc/TrivediEnterprisesInc.github.io/blob/837151d3c68f6b452917f5e9895de62ba799d885/winFrms.fs#L3340)

**webCli: DvDef**



Note: We still nd the **CardVw** (Some work completed in Film TagCloud, Loc8)

Also needed: Condensed Vw (+ logic)

**webCli: Form**

**- Expr bldr/valid8n:**We don't nd (expr, js) but (\_, str) which can be manually inserted into the JS code.   
So we can just straight-up move the Filter code branches which we use to bld strings and return a tpl   
@rsch: AND/OR   
@rsch: How to test 4 valid js exprs from within FsChk?

**JS validity**

PS + Node JSHint, JSLint, ESLint all either req node or a custom setup; instd you can invoke node from PS directly as shown below:   
Instd of PS call frm fs via process; grab console outpt ...   
  
Certainly! You can use Node.js to evaluate JavaScript expressions from PowerShell by following these steps:   
Install Node.js: from nodejs.org.   
Create a JavaScript File: Create a JavaScript file (e.g., evaluate.js) that will evaluate the expressions.

// evaluate.js

const vm = require('vm');

const expression = process.argv[2];

try {

const result = vm.runInNewContext(expression);

console.log(result);

} catch (error) {

console.error('Error evaluating expression:', error.message);

}

Run the JavaScript File from PowerShell: Use PowerShell to run the Node.js script with the expression you want to evaluate.

# Example expression

$expression = "2 + 2"

# Run the Node.js script with the expression   
node evaluate.js $expression   
Example Usage   
Save the above JavaScript code in a file named evaluate.js.   
Open PowerShell and navigate to the directory containing evaluate.js.   
Run the following command in PowerShell:

node evaluate.js "5 \* (3 + 2)"

This will output:   
25   
Feel free to adapt this approach to suit your specific needs! If you have any more questions or need further assistance, I'm here to help.