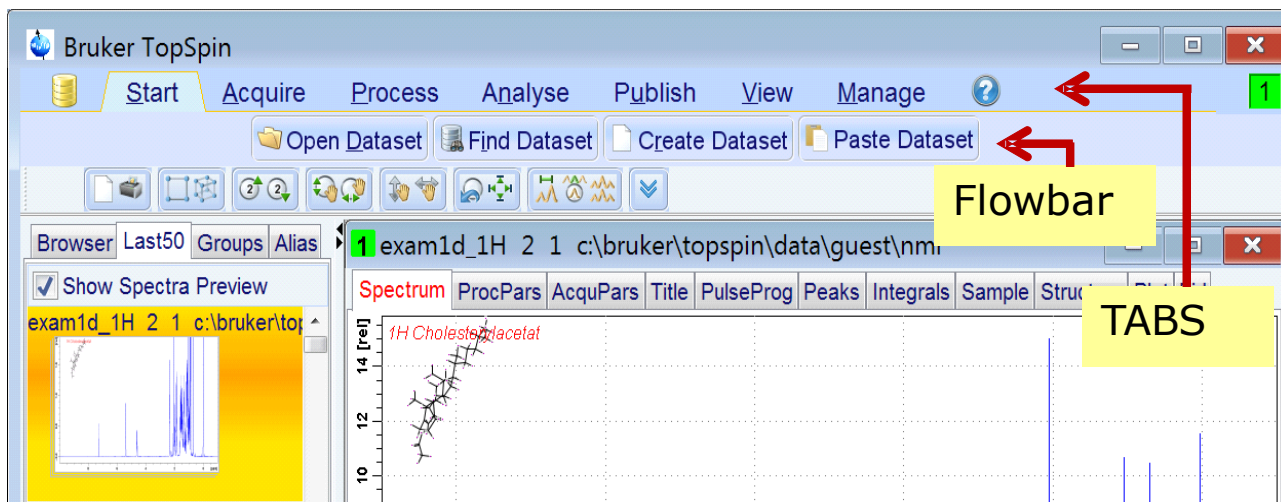


Creating User-Defined Flowbars And Tabs

(Version 2009-11-02)

From version 3.0 on TopSpin is offering the *flow user interface*, replacing the traditional menu bar with its pull down menus. The flow interface consists of TAB panels, each panel containing a *flowbar* consisting of a series of action buttons which may describe a *workflow*.



The Command *fbar*

Flowbars are represented as text files of a special format and can be setup without any “actual” programming. The command *fbar* is provided to display a flowbar.

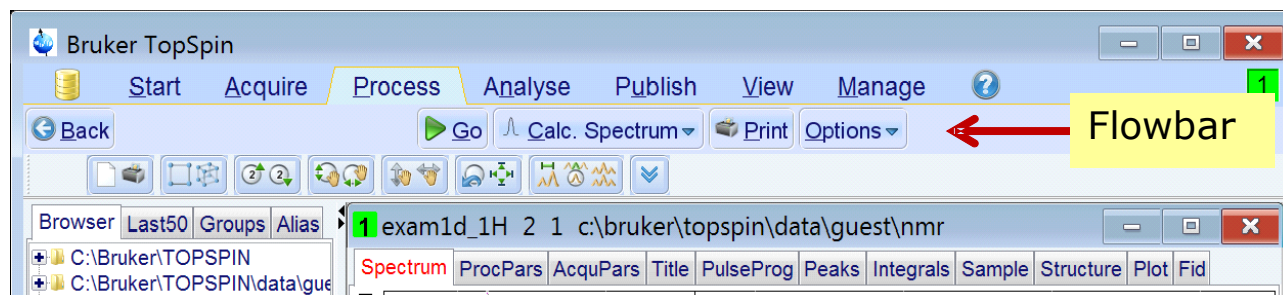
<p><code>fbar <filename>.prop <flowbarID> <tabkey></code> Possible values for <tabkey>: UI2_TAB_START, UI2_TAB_ACQUIRE, UI2_TAB_PROCESS, UI2_TAB_ANALYSE, UI2_TAB_PUBLISH, UI2_TAB_VIEW, UI2_TAB_MANAGE (or you own tabkey, since you may also add own tabs)</p>	<p>Displays the flowbar with the specified ID under the specified TAB. The flowbar must be defined in the text file given by <i><filename>.prop</i>. The file must be stored in the following TopSpin directory: <code><topspin home>/classes/prop/flowbars</code>. The 3 arguments of <i>fbar</i> are separated by space characters.</p>
<p><code>fbar <file path>.prop <flowbarID> <tabkey></code></p>	<p>Displays the flowbar with the specified ID under the specified TAB. The flowbar must be defined in the text file given by <i><filename>.prop</i>. The file can be stored anywhere, requiring that it be specified with its full path. The 3 arguments of <i>fbar</i> are separated by ' ' (vertical bar) characters because the file path may contain spaces). <i>Note:</i> Each flowbar definition file <i><filename>.prop</i> requires an associated file <i><filename>.txt.prop</i> containing the textual button descriptions (see further below for details).</p>
<p><code>fbar <tabkey></code></p>	<p>Displays the standard Bruker flowbar under the specified TAB.</p>

You can utilize *fbar* in the following way: Type it into the command line with all its arguments. Or embed it into a TopSpin macro (*edmac*), a Python program (*edpy*), an AU program (*edau*, using *sendgui*), or assign it to a user-defined tool button (*right-click into a free area of the tool bar to define new tool buttons*).

TopSpin provides an example flowbar which can be used as a basis for your own development. In order to display it, please perform as follows:

- 1) Click on the *Process Tab* of TopSpin
- 2) Enter the command *fbar exam1 EXAMI_FLOWBAR UI2_TAB_PROCESS*

This will display the following flowbar under the *Process Tab* and replace the standard flowbar.



The new flowbar has 5 buttons, *Back, Go, Calc. Spectrum, Print, Options*.

Clicking on *Back* will close this flowbar and restore the standard Processing flowbar. The other buttons are assigned to respective TopSpin command. A command description (*tooltip*) will be displayed when the cursor enters a button area.

Flowbar Definition

The command *fbar exam1 EXAMI_FLOWBAR UI2_TAB_PROCESS* looks up the file *exam1.prop* in the directory *<topspin home>/classes/prop/flowbars*. The associated button text definition file *exam1_txt.prop* is looked up in *<topspin home>/classes/prop/English/flowbars*. The command instructs TopSpin to display the flowbar called *EXAMI_FLOWBAR* under the *Process Tab*, referred to as *UI2_TAB_PROCESS*. The files *exam1.prop/ exam1_txt.prop* are part of a TopSpin installation and has the following contents (Table 1).

```
# flowbar definition
EXAMI_FLOWBAR=\
  NM=nav_left_blue.png, MC2=Y, TXT=EXAMI_BACK, CMD=fbar UI2_TAB_PROCESS,\
    TIP=EXAMI_BACK_TOOLTIP, END=,\
  NM=media_play_green.png, TXT=EXAMI_ZG, CMD=zg,\
    TIP=EXAMI_ZG_TOOLTIP, END=,\
  NM=ez_pk_40.gif, EM=1, TXT=EXAMI_EFP, CMD=lb 0.3;ef;apk, CMD2=_pop,\
    EXAMI_EFP_POPUP,\
    TIP=EXAMI_EFP_TOOLTIP, END=,\
  NM=printer3.png, TXT=EXAMI_PRINT, CMD=prnt,\
    TIP=EXAMI_PRINT_TOOLTIP, END=,\
  NM=, EM=1, TXT=EXAMI_OPTIONS, CMD=_pop EXAMI_OPTIONS_POPUP,\
    CMD2=_pop EXAMI_OPTIONS_POPUP,\
    TIP=EXAMI_OPTIONS_TOOLTIP, END=

# popup menu definitions
EXAMI_EFP_POPUP=\
  NM=exam1_manual_phasing, MC2=Y, CMD=.ph, END=

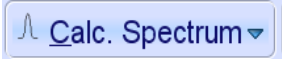

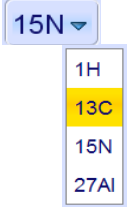

EXAMI_OPTIONS_POPUP=\
  NM=exam1_multdisp, CMD=.md, END=,\
  NM=-, END=,\
  NM=exam1_calib, CMD=.cal, END=
```

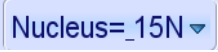



Table 1: flowbar definition file *exam1.prop* for the command *fbar exam1 EXAMI_FLOWBAR UI2_TAB_PROCESS*

The flowbar *EXAMI_FLOWBAR* is described in the file by a single line. *Line* means a sequence of characters terminated by a *new line* character (as generated by pressing the *Enter* button of the keyboard). Since the line describing a flowbar can be become rather long, it can be split into convenient pieces by the character sequence *backslash, immediately followed by new line*, as it was done in the flowbar description above. Beware: Backslash followed by a space character followed by new line is illegal and will lead to errors!

A flowbar consists of a sequence of buttons. Each button description starts with *NM=*, and ends with *END=*. Between these two tags, the properties of the flowbar are defined according to the following Table 2.

Button Tag	Description
<i>NM=<icon file name></i>	<p>Starts a new button. The button may contain an icon, followed by text. <i><icon file name></i> is the name of an icon file in containing a <i>png</i> or <i>gif</i> image. A full file path may be specified. If only a name is given, but not a full file path, it must be one of the TopSpin standard icons. They are located in the files <i><tophome>/classes/prop/gif_icons.jar</i> and <i>png_icons.jar</i>. This files are in zip format. <i><icon file name></i> may just consist of a comma. In this case no icon is displayed in the button.</p> <p>Examples: <i>NM=printer3.png</i> (uses a TopSpin standard icon with this name) <i>NM=,</i> (no icon displayed) <i>NM=c:/users/guest/myicon.png</i> (a full icon file path)</p> <p>The example flowbar in Table 1 uses 3 TopSpin standard icons: <i>nav_left_blue.png</i> - left arrow <i>media_play_green.png</i> - right arrow <i>printer3.png</i> - printer symbol</p>
<i>TXT=<button text key></i> <i>this key is a name to be defined by the flowbar designer</i>	<p>Defines the text to be displayed in the button. <i>Note:</i> This is not the text itself, but a key to the text. The text itself must be stored in a different file. In our example, the flowbar is described in the file <i>exam1.prop</i>. All button text, as well as other texts such as tooltips, must then be stored in the file <i>exam1_txt.prop</i>. This rule is generally valid: Texts are looked up in files with <i>_txt</i> appended to the flowbar file name.</p> <p>Example: <i>TXT=EXAM_ZG</i>. If the file <i>exam1_txt.prop</i> contained the line <i>EXAM_ZG=zg</i>, then the button would show the text <i>zg</i>. The flowbar text file must be contained in the same directory as the flowbar description file.</p> <p>The contents of <i>exam1_txt.prop</i> are displayed in Table 3 below.</p>
<i>CMD=<command></i>	<p>A command, macro, AU or Python program will be executed when clicking on the button.</p> <p>Examples: <i>CMD=zg</i> (execute <i>zg</i>) <i>CMD=lb 0.3;ef;apk</i> (execute <i>lb 0.3+em + ft + apk</i>, this example shows queued command using “;” as a separator) <i>CMD=xau proc_1d</i> (execute this AU program) <i>CMD=.ph</i> (enter manual phase correction)</p>

<p><i>TIP=<tooltip text key></i> (optional)</p>	<p>Defines the tool tip text to be displayed when the mouse enters the button area. <i>Note:</i> Similar to the <i>TXT= tag</i>, this is not the text itself, but a key to the text. The text itself must be stored in the flowbar text file as described under <i>TXT=</i>. Example: <i>TIP=EXAM1_EFP_TOOLTIP</i>. Then the file <i>exam1_txt.prop</i> must contain an entry <i>EXAM1_EFP_TOOLTIP</i>, e.g. <i>EXAM1_EFP_TOOLTIP=Compute Spectrum
\</i> <i>Executes command <i>lb 0.3;ef;apk</i></i>. The example shows that the tool tip text is html capable for nice text formatting: The html tags <i></i> (bold), <i><i></i> (italic), <i>
</i> (break=new line) are being used.</p>
<p><i>EM=1</i> <i>CMD2=<command></i> <i>used to display a popup menu from a button</i></p> 	<p>If the description of a button contains the tag <i>EM=1</i>, then the button area will have two different active field: The area containing the button, and the area containing a down arrow. In fact, <i>EM=1</i> causes the down arrow to be displayed right of the button text, inside the button area. Now the following happens: When you click on the button text, the command described by <i>CMD=<command></i> is executed. When you click on the down arrow, the command described by <i>CMD2=<command></i> is executed. As the down arrow indicates, its associated command <i>CMD2</i> is normally pops up a menu, although in principle any command could be assigned. Example: <i>CMD2=_pop EXAM_EFP_POPUP</i> This will popup the menu whose description is provided by the popup identifier <i>EXAM_OPTIONS_POPUP</i>. The structure of popup identifiers is explained below in the section <i>Popup Menu Definition</i>.</p>
<p><i>TIP2=<tooltip text key></i> (optional)</p>	<p><i>CMD2</i> may have its own tool tip text.</p>
<p><i>MC2=Y</i> (optional)</p> 	<p>This tag will cause the button to be displayed at the left most position of the TopSpin window. The purpose of the tag is to separate the <i>Back</i> button (if existing) of a flowbar from the normal function buttons. Example: The <i>Back</i> button of our <i>exam1</i>.</p>
<p><i>ME=</i> <i>menu selection</i></p>  <p><i>after menu selection</i></p>  <p>or <i>ME=<fixed text key></i></p>	<p>This tag, in conjunction with <i>EM=1</i> and the <i>CMD2=<command></i>, will make the button a <i>combo-box-like</i> button. This means: You can click on the arrow to open an associated menu. As soon as you click on a menu item, the menu command is executed (e.g. <i>13C</i>, see left), and in addition, the button text is replaced by the text of the clicked menu item. The advantage is that now you can click on the button (<i>13C</i>) to execute the command immediately, with having to open the menu. If <i><fixed text key></i> is specified after <i>ME=</i>, the respective text will be displayed first (see left). Please note: <i>TXT=</i> must be identical to the first menu entry, and <i>CMD=</i> must be the command of the first menu entry!</p>

<i>menu selection</i>  <i>after menu selection</i> 	
<i>BG=<R G B></i> <i>(optional)</i> 	<p>You can change the default background color of the button by specifying the color's red/green/blue component.</p> <p>Example: <i>BG=0 255 255</i> (button background color is yellow)</p>
<i>FG=<R G B></i> <i>(optional)</i> 	<p>You can change the default text color of the button by specifying the color's red/green/blue component.</p> <p>Example: <i>FG=255 0 0</i> (button text color is red)</p>
<i>ME2=keep</i> <i>(optional)</i>	<p>Normally, when the user loads a new dataset, the currently visible user-defined flowbar is replaced by the standard flowbar of the active TAB. In order to prevent that and have the user-defined flowbar not removed, this option must be set.</p>
<i>END=,</i>	<p>Terminates a button definition</p>
<i>END=</i>	<p>Terminates a button definition and also terminates the definition of the flowbar.</p>
Table2: Flowbar definition tags	

Flowbar Text Definition

Table 3 below shows the text definition file for our example. A text definition is defined by a line in the format: `<text key>=<text>`.

The *text key* is used in the flowbar file (e.g. *exam1.prop*) to reference the actual *text* to be displayed. The key definitions are defined in a file derived from the flowbar file by appending *_txt*, i.e. *exam1_txt.prop* in our example. As described under *TXT=* and *TIP=* in Table 2, *html-formatted* text can be used. Since a text definition line `<text key>=<text>` can be become rather long, it can be split into convenient pieces by the character sequence *backslash, immediately followed by new line*. Beware: Backslash followed by a space character followed by new line is illegal and will lead to errors!

The line `EXAM1_BACK=Back `b` in Table 3 shows a special feature: The characters ``b` (back-apostrophe, followed by b) define that you can activate the *Back* button by typing ALT B on the keyboard, in addition to clicking into the button. The B of Back will be underlined in the button display (Back). Please note: The characters f, s, a, p, n, u, v, n, h cannot be used as accelerator keys because they are already consumed by the TABs (File, Acquire, Process, ...).

```
# button texts
EXAM1_BACK=Back `b
```

```

EXAM1_ZG=Go `g
EXAM1_EFP=Calc. Spectrum `c
EXAM1_PRINT=Print `r
EXAM1_OPTIONS=Options `o

# popup texts
exam1_manual_phasing=Manual Phase Correction
exam1_multdisp=Multiple Spectra Display
exam1_calib=Calibrate Spectrum Manually

# tooltip texts
EXAM1_BACK_TOOLTIP=<b>Close example 1 flow bar</b><br>\
Display standard <i>Process Tab</i> flowbar.

EXAM1_ZG_TOOLTIP=<b>Start Acquisition</b><br>\
Executed <i>zg</i> command with <i>eda</i> parameters.

EXAM1_EFP_TOOLTIP=<b>Compute Spectrum</b><br>\
Executes command <i>lb 0.3;ef;apk</i>

EXAM1_PRINT_TOOLTIP=<b>Print active window</b><br>\
Spectrum, integrals, peaks etc. are printed<br>\
as shown on the display. You may adjust printing colors<br>\
and fonts using the drop-down menu of this button.<br>\
In the View Tab, you may define which spectra components<br>\
(integrals, ...) are displayed.<br><br>\
<em>For formatted plots, use plot layouts.</em>

EXAM1_OPTIONS_TOOLTIP=<b>Command Options</b><br>\
Some frequently used commands.

```

Table 3: flowbar text definition file *exam1_txt.prop* for the command *fbar exam1 EXAM1_FLOWBAR UI2_TAB_PROCESS*

Popup Menu Definition

The descriptions of the flowbar tags *EM=1*, *CMD2=* in Table 2 display how a button may get a down arrow right of the button text, normally used to open a popup menu. The respective command is *_pop <popup menu key>*, e.g. *_pop EXAM_OPTIONS_POPUP*. The menu key describes the popup properties and must be defined in the flowbar definition file (Table 1). A popup menu consists of a sequence menu entries. Each menu entry description starts with *NM=*, and ends with *END=*. Between these two tags, the properties of the menu entry are defined according to the following Table 4.

Popup Menu Tag	Description
<i>NM=<menu text></i>	Defines the text to be displayed in the menu entry. <i>Note:</i> This is not the text itself, but a key to the text. The text itself must be stored in the flowbar text definition file (in our example <i>exam1_txt.prop</i>). Please note: <i>NM=-</i> represents a separator line in the menu. Example: <i>exam1_manual_phasing=Manual Phase Correction</i>
<i>NM2=<icon file name></i> <i>(optional)</i>	Displays an icon in front of the text defined by <i>NM=</i> . See more about icon in <i>Table 2</i> under the tag <i>NM=</i> .
<i>CMD=<command></i>	A command, macro, AU or Python program will

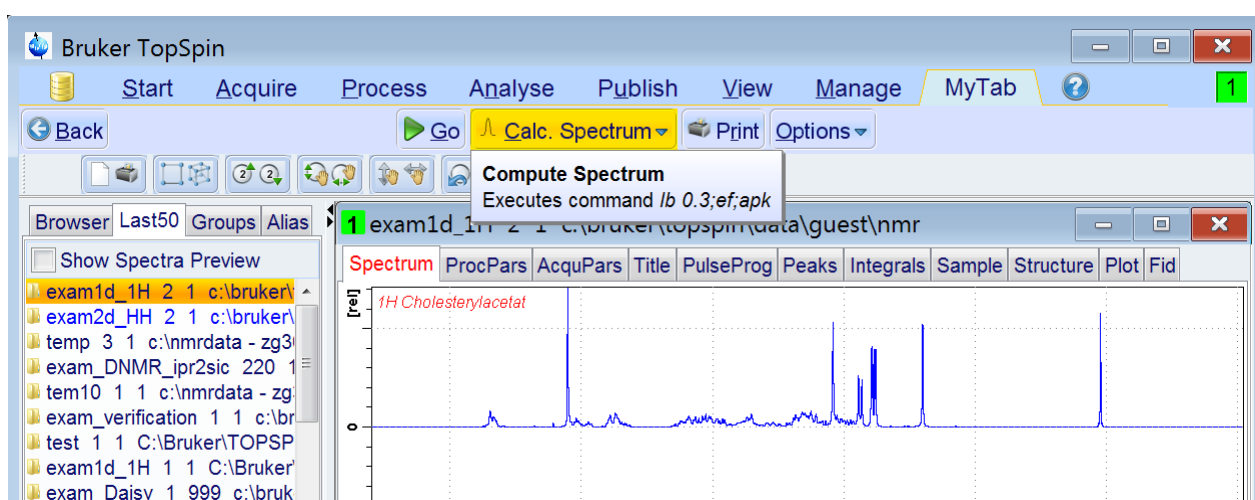
	be executed when clicking on the menu entry. See <i>Table 2</i> , tag <i>CMD=</i> for details.
<i>END=,</i>	Terminates a menu entry definition
<i>END=</i>	Terminates a menu entry definition and also terminates the definition of the popup menu.
Table 4: Popup menu definition tags	

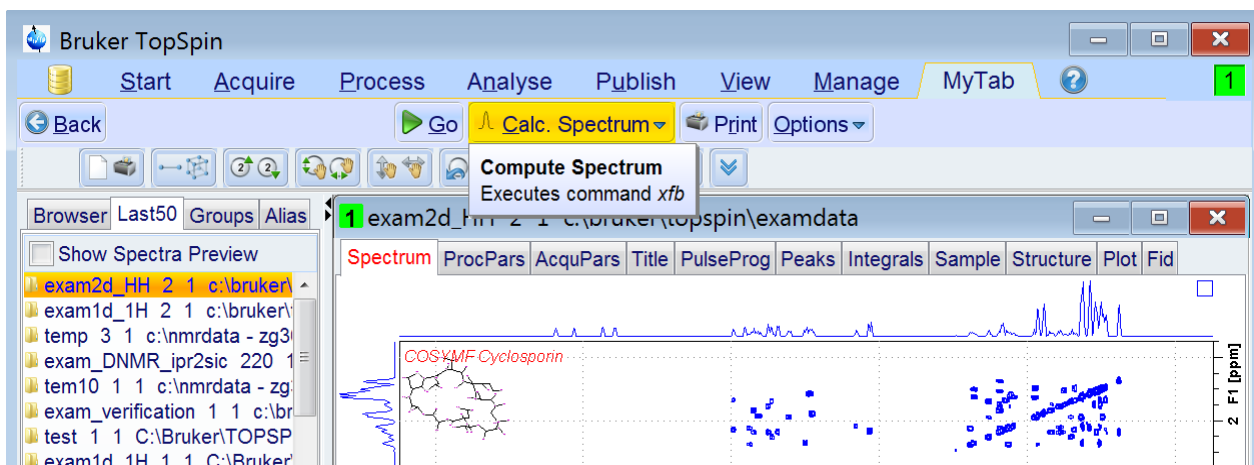
Flowbar Summary

- 1) Flowbars are defined in a flowbar definition file *x.prop* (x being an arbitrary name).
- 2) x is either an absolute file path or just a file name. In the latter case it must be located in *<topspin home>/classes/prop/flowbars*.
- 3) If the flowbar definition file is *x.prop*, the associated flowbar texts must be defined in *x_txt.prop*.
If x is an absolute file path, *x_txt.prop* must be stored in the same directory as *x.prop*, otherwise in *<topspin home>/classes/prop/English/flowbars*.
- 4) A flowbar definition file may contain several flowbar definitions (differing in their flowbar IDs)
- 5) A flowbar is displayed (or closed) using the command *fbar* described above. *fbar* may be called from a user-defined tool button, a user-defined command, a Macro, an AU program or a Python program.
- 6) A flowbar may also be displayed from a user-defined TAB. This is described below.

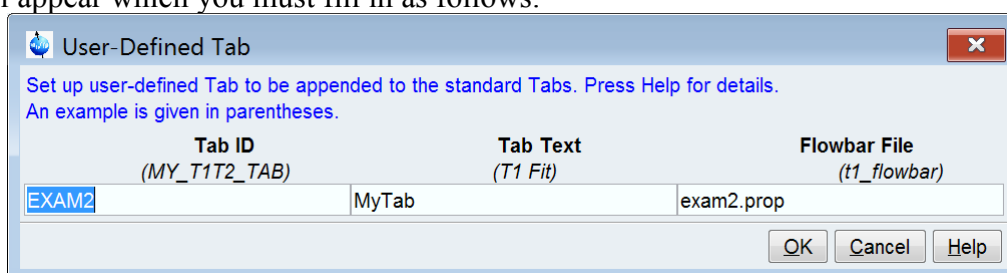
User-Defined TABs And Associated Flowbars

TopSpin allows you add your own TABs to the standard TABs: See the next two pictures containing the user-defined *MyTab*.





To set up this TAB, right-click onto an existing TAB and select *Edit Tab/Add New Tab*. A table type dialog will appear which you must fill in as follows:



Press OK, and the new TAB will appear on the screen. How does this work?

EXAM2 is our chosen Tab ID and *MyTab* is what gets displayed in TopSpin inside the new TAB. Now, when you click on *MyTab*, TopSpin will try to display a flowbar associated to the TAB. The flowbar ID is *USERTAB_EXAM2_ID*: It is constructed from the Tab ID by prepending *USERTAB_* and by appending *_ID*. The flowbar file entry of the table above finally defines where the flowbar ID is looked up. *exam2.prop* is part of your TopSpin installation, this is why the example is working immediately.

The nice thing about flowbars associated to TABs is that the displayed flowbar can be data dimension dependent. In the two TopSpin screenshots above *MyTab* displays the same sequence of buttons for 1D and 2D data. However, different commands and tool tips are associated to the *Calc. Spectrum* button. If desired, even the entire flowbar layout could be made dimension dependent. In order to force TopSpin to display a flowbar different from *USERTAB_EXAM2_ID* for 2D data, the flowbar definition file *exam2.prop* must contain the definition of a second flowbar called *USERTAB_EXAM2_2D* (the same naming convention applies to data of even more dimensions). The respective *exam2.prop* is displayed in Table 5.

```
# flowbar definitions
USERTAB_EXAM2_1D=\
  NM=nav_left_blue.png, MC2=Y, TXT=EXAM2_BACK, CMD=fbar UI2_TAB_PROCESS,\
  TIP=EXAM2_BACK_TOOLTIP, END=,\
  NM=media_play_green.png, TXT=EXAM2_ZG, CMD=zg,\
  TIP=EXAM2_ZG_TOOLTIP, END=,\
  NM=e_z_pk_40.gif, EM=1, TXT=EXAM2_EFP, CMD=1b 0.3;ef;apk, CMD2=_pop
EXAM2_EFP_POPUP,\
  TIP=EXAM2_EFP_TOOLTIP, END=,\
  NM=printer3.png, TXT=EXAM2_PRINT, CMD=prnt,\
  TIP=EXAM2_PRINT_TOOLTIP, END=,\
  NM=, EM=1, TXT=EXAM2_OPTIONS, CMD=_pop EXAM2_OPTIONS_POPUP, CMD2=_pop
EXAM2_OPTIONS_POPUP,\
  TIP=EXAM2_OPTIONS_TOOLTIP, END=
```



```

USERTAB_EXAM2_2D=\
  NM=nav_left_blue.png, MC2=Y, TXT=EXAM2_BACK, CMD=fbar UI2_TAB_PROCESS,\
  TIP=EXAM2_BACK_TOOLTIP, END=,\
  NM=media_play_green.png, TXT=EXAM2_ZG, CMD=zg,\
  TIP=EXAM2_ZG_TOOLTIP, END=,\
  NM=ez_pk_40.gif, EM=1, TXT=EXAM2_EFP, CMD=xfb, CMD2=_pop
EXAM2_EFP_POPUP,\
  TIP=EXAM2_EFP_TOOLTIP_2D, END=,\
  NM=printer3.png, TXT=EXAM2_PRINT, CMD=prnt,\
  TIP=EXAM2_PRINT_TOOLTIP, END=,\
  NM=, EM=1, TXT=EXAM2_OPTIONS, CMD=_pop EXAM2_OPTIONS_POPUP, CMD2=_pop
EXAM2_OPTIONS_POPUP,\
  TIP=EXAM2_OPTIONS_TOOLTIP, END=

# popup definitions
EXAM2_EFP_POPUP=\
  NM=EXAM2_manual_phasing, MC2=Y, CMD=.ph, END=

EXAM2_OPTIONS_POPUP=\
  NM=EXAM2_multdisp, CMD=.md, END=,\
  NM=-, END=,\
  NM=EXAM2_calib, CMD=.cal, END=

```

Table 5: flowbar definition file *exam2.prop*

Summary for flowbars associated with user-defined TABs

- 1) If Tab ID = x, then the associated flowbar ID must be USERTAB_x_1D. The Tab ID is specified during Tab definition. Also, the file containing the flowbar definition(s) is specified in this phase.
- 2) If the flowbar definition file contains additional flowbar definitions with IDs called USERTAB_x_2D or USERTAB_x_3D, then these flowbars get displayed when 2D or 3D data are on the screen, respectively.
- 3) If only USERTAB_x_1D is present in the flowbar definition file, it will be displayed for data of any dimension.