Icetips
XP-Taskpanel

10/14/2018

Taskpanel for Clarion

Published: October 2018

Publisher

Icetips Creative, Inc.

Managing Editor

Arnor Baldvinsson

Table of Contents

Part I

Chapter 1 - Introduction

1 Introduction

1.1 Welcome

Taskpanel is a native control for Clarion. It emulates the taskpanes that are found in Microsoft's Windows XP. Taskpanels provide a flexible way of presenting a group of related tasks to your users. Taskpanel provides you with a graphically rich, modern looking and visually appealing way of presenting logically grouped sets of choices.

Features:

- "Easy-to-use" Control Template
- Icons in headers and tasks
- Blue and white styled headers
- Multiple controls per window
- Mimic buttons
- Dynamically create and delete headers and tasks
- Hide/Unhide/Enable/Disable headers and tasks
- Change header and task properties runtime
- Task tooltips
- Wizard-mode
- Automatic scrollbar
- Mousewheel scrolling
- Real, resizable gradients
- Customizable colors
- Multi-DLL support.
- Clarion 5.5, Clarion 6 and Clarion 6.1
- ABC and Legacy
- Full Clarion source code (no black-box DLL's)
- Windows 95/98/ME/NT/2000/XP

Users upgrading from version 1.4 or earlier should read Upgrade issues.

Note that Icetips Alta LLC took this product over from PowerOffice in Norway in December 2008. It is possible that there are some references to PowerOffice in this text or in the source codes. If you find references to PowerOffice, please let us know.

1.2 Quickstart

Follow the steps below to quicly implement a Taskpanel in your application:

Add the global template

- Open your application (or create a new one)
- Press the "Global"-options button ()
- Click the "Extensions"-button
- Click the "Insert"-button
- Choose the "XPTaskpanelGlobal - Icetips XP Taskpanel Global"-template

- Press "Ok" two times to get back to the application
- In the Application frame properties check the "Immediate" option.

Populate the control

- Create a Window
- Populate "XPTaskPanel" control template by clicking the

- Choose the "XPTaskPanel - Icetips XP Taskpanel Control"-template

Legacy-NOTE: Select XPTaskPanelCW instead..

- Place the control somewhere on you window and adjust the size.

Add 2 global variables (you can do that in the dictionary if you want to) Here we are using:

```
MainThreadID    LONG
ToolboxMenuID   LONG
```

In the Appframe procedure, add this code in OpenWindow event handling embed, after generated code:

```
MainThreadID = THREAD()
START(TaskPanelProcedure, 25000)
```

In the Sized window event handling embed, after generated code, add:

```
pix# = 0{PROP:Pixels}
0{PROP:Pixels} = TRUE
h# = 0{PROP:ClientHeight} - 4
NOTIFY(h#, ToolboxMenuID)
0{PROP:Pixels} = pix#
```

On the Taskpanel procedure add a local variable:

```
nHeight          UNSIGNED
```

Then, in ThisWindow.Init method, right at the beginning:

```
ToolboxMenuID = THREAD()
```

In ThisWindow.TakeWindowEvent method, priority 2501:

```
OF EVENT:Notify
  IF NOTIFICATION (nHeight,,)
      pix# = 0{PROP:Pixels}
      0{PROP:Pixels} = TRUE
      0{PROP:Height} = nHeight
      ?XPTaskpanel{PROP:Height} = nHeight
      0{PROP:Pixels} = pix#
      DISPLAY
  END
```

And finally in the CloseWindow event handling embed, before generated code put:

```
ToolboxMenuID = 0
```

That should be all the code you need to get the Taskpanel up and running.

Template settings

- Right-click on the XPTaskpanel control and choose "Actions".
- Now, press the "Insert"-button to bring up the "Header Editor".

(Header Settings)

  - In the "Title"-field, enter "Header 1" (without the quotes)
  - If you'd like an icon in the header, you can select the "Style"-tab and press the elipsis-button (...) to right of the icon-field, and choose an .ico-file.

- Click the "Tasks"-tab
- Now, click "Insert" to add a task.

(Task Settings)

- Enter "Task 1" in the "Title"-field

- Click "Ok" to add the Task

- Click "Insert" to add another task

- Enter "Quit" in the title-field

- Choose the "Action"-tab

- Action: Post Event
- Event: EVENT:CloseWindow
- Click "Ok" to add the Task

- Click "Ok" three times to get back to the window formatter.
- Save the window

- Now, go to the embed-point "Local Objects -> XPTaskpanel1 -> Header1 -> Header1:Task1 -> Task Clicked -> Before Generated Code", and enter the following code:

```
Message('You clicked ' & Self.GetTaskTitle(HeaderID, TaskID))
```

- Save, compile and run your application.

The result should look something like this:

When you click on "Task 1" a message saying "You clicked Task 1" should appear.

You'll find this application under "Clarion6\3rdParty\Examples\XPTaskpanel\quickstart_c6.app"

1.3 Contact us

Upgrades

The latest upgrades can be downloaded from www.icetips.com Note that you must have a valid Gold Subscription login to download.

Bug reports and suggestions

If you want to contact us, please send us emails to support@icetips.com

If your app is a multi-dll app, please try to recompile all your apps before reporting possible bugs.

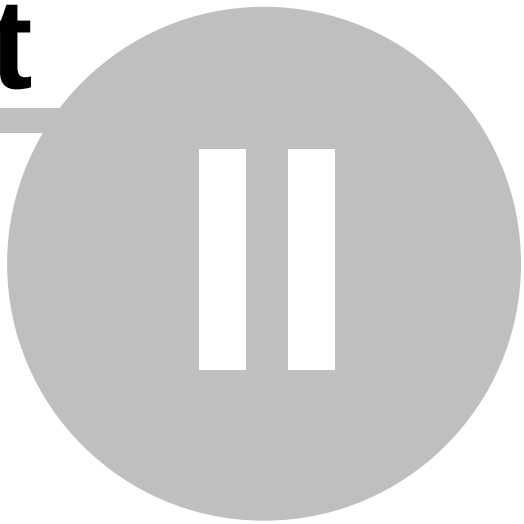
When reporting bugs, please let us know your version of Clarion and Icetips Taskpanel, and if possible, provide a screen shot. An example app or steps to reproduce in the demo app is highly appreciated.

Please report bugs to our bug tracking system at http://icetips.fogbugz.com Make sure that you select the proper project (XPTakspanel) and the correct area (Class/Documentation/Install/Misc/Templates) and give us good, detailed description of what the problem is. You can attach files to your bug reports.

Other Clarion Products

You'll find more of our Clarion products at www.icetips.com

Part



Chapter 2 - Templates

2 Templates

2.1 About the templates

Global Template
The global template is responsible for including the needed files in your application. This template must be present in all apps using the control-template. It should also be added to the data-dll application of multi-dll apps.

The control-template will not be visible in the control template-list until the global template has been added to the application.

Control Template
The control template generates the actual taskpanel-control.

Have a look at Adding the templates for more info on how to add the templates.
For a detailed description of the various settings, read Control Template, Header Settings and Task Settings.

2.2 Adding the templates

How to add the global template

- Open your application (or create a new one)
- Press the "Global"-options button ()
- Click the "Extensions"-button
- Click the "Insert"-button
- Choose the "XPTaskpanelGlobal - XP Taskpanel Global"-template

- Press "Ok" two times to get back to the application

Populating a Taskpanel control

- Create a Window
- Populate "XPTaskPanel" control template by clicking the

- Choose the "XPTaskPanel - XP Taskpanel Control"-template

Legacy NOTE: Select XPTaskPanelCW instead..

2.3 Template settings

2.3.1 Control Template Template settings

Open the template settings:
- Right-click the taskpanel
- Select "Actions"

Task panel
Insert
Add new header. This will open the Header Settings

Properties
Change the properties for the selected header.
This will open the Header Settings

Delete
Delete header

Options
Only allow one expanded header
Only one header can be expanded at any time. When a user expands a header, all other headers will become contracted.

Enable Wizard Mode
If the taskpanel is in "Wizard mode" the last clicked task is bolded, to indicate the current step in the wizard.

Disable word wrap
If this option is checked, all tasks will be draw in single-line modus (appending ellipsis (...) at the end of the task title, if it's to long to fit)

Tooltips Mode
Set the mode of tooltips. Disabled, Normal or Balloon.

Customize Colors
Brings up the color settings for this taskpanel

Set margins
Change the margins used when drawing headers and tasks

Font
The font used for this taskpanel

Advanced
Object Name
This is the object name used when generating code.

Object Class
The class to derive when generating code

Init code priority
Set the init-code priority. This can be set to 9001 if you want the taskpanel to be initialized at the end of the %WindowManagerMethodCodeSection.

If you are using 0-Hazzle, then you must set init-code priority to 9001.

2.3.2 Header Settings — Template settings

Header settings

Title
This is the text that will be shown onscreen. If you would likt to use a variable, just prefix the variable name with '!' (without the quotes)

ID
Each header need a unique identifier. This ID will be used by the template when generating embed-points. If you change the ID after inserting embedded code, this will become orphaned. A LONG-variable in the class will be generated. This LONG will be automatically set runtime, and can be used for calls to class-methods.

Expanded
Sets if this header is initially expanded upon window open

Always Expanded
The header will not be possible to contract (and the contract/expand button will be hidden).

Visible
Is this header initially visible upon window open

Tasks

Insert
Add a new task to this header. This will open Task Settings.

Properties
Change the properties of the selected task. This will open Task Settings.

Delete

Delete the selected task

Style

Icon

Name of the icon. You can use a variable by prefixing the variablename with '!' (wihtout the quotes)

Style

Normal - Standard blue header
Highlightet - White header

Task Icon Size

The icon size of the tasks under this header. The height of each task is based on the icon size, however no task be less than 20 pixels (even though task icons is 16x16.

Boxed

When a header is boxed, it will be drawn as a single line. This gives you a box with task in it.

Word-wrap

Uncheck this option to disable wordwrapping for this header.

Disable wizard mode for this header

This header will not be included in the wizard mode, and tasks will be drawn as usual.

2.3.3 Task Settings Template settings

Task settings

Title

This is the text that will be shown onscreen. If you would likt to use a variable, just prefix the variable name with '!' (without the quotes)

ID

Each task need a unique identifier. This ID will be used by the template when generating embed-points. If you change the ID after inserting embedded code, this will become orphaned. A LONG-variable in the class will be generated. This LONG will be automatically set runtime, and can be used for calls to class-methods.

Tooltip

The tooltip for this task. Tooltip text accepts linefeeds (<13,10>).

Visible

Controls the visibillity upon window open

Enabled
If unchecked, the text will not be clickable.

Action
Action

Specifies the type of action for this task. Each type brings up different options:

No Special Action
Do nothing. You should use one (or several) of the embed-points to hand code some action.

Call Procedure
Call a prodecure directly. The settings are just as the standard Clarion "Call Procedure" .

Mimic button
Hides the task when the button specified in "Control" is disabled. If the button is enabled, then the task is visible. If the PROP:Text of the button is changed, this will also be reflected in the taskpanel. Use "mimic button" to add browse-control in your taskpanel (refer the example-application).

When "Mimic Text" is checked, the tasks title will be extracted from the button (using PROP:Text). If "Mimic Text" is unchecked, the title specified for the task will be used.

If the "Mimic icon" option is checked, the icon-name is extracted from the button (STD-icons are not supported). Otherwise the icon specified for the task is used. If "Mimic icon" is checked and the mimiced button got no compiletime iconname, Button{PROP:Icon} is used instead.

Post Event
Posts the specified event. If you specify "Control" the event will be posted to this event. You can also specify which thread should reveice the event.

Select Tab
Activate a Tab-control. Just choose the Tab-control you want to activate (bring to front). When this action is executed, EVENT:TabChanging will be posted to the parent Sheet.

Set Field Value
Assign a value to a variable. The Value field could be any valid Clarion expression/assignment.

Style
Icon

Name of the icon. You can use a variable by prefixing

the variablename with '!' (wihtout the quotes)

Justification
Text justification

Bold
If checked, the task's text will be draw with bold font

2.4 Code templates

The Taskpanel has two code templates that were added on July 16, 2009.

Set Header Text
Set Task Text

2.4.1 Set Header Text Code templates

This template makes it easy to set the text of a header.

Select the header from the "Header to set" drop down and then enter the new text into the "Text expression" field. You can use a string or you can use variables or you can use combination. Use the lookup button ([...] button) to insert variable names or fields from datafiles.

See also:
Set Task Text

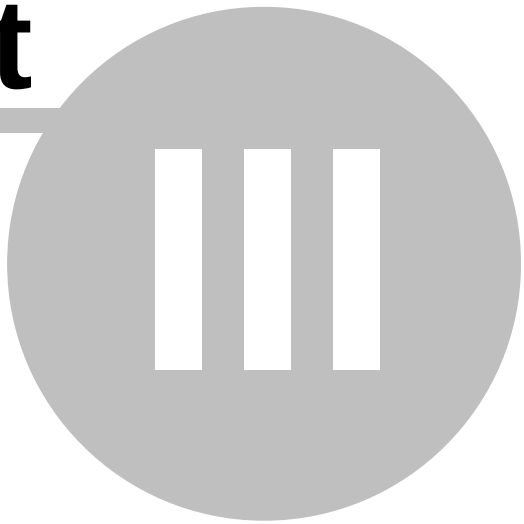
2.4.2 Set Task Text Code templates

This template makes it easy to set the text of a task.

Select the header from the "Header to set" drop down and then enter the text. The "Task to set" drop down will then show the tasks related to the header. Enter the new text into the "Text expression" field. You can use a string or you can use variables or you can use combination. Use the lookup button ([...] button) to insert variable names or fields from datafiles.

See also:
Set Header Text

Part



Chapter 3 - Misc

3 Misc

3.1 Multi DLL Applications

In your data-dll you must add the global template "**XPTaskPanelGlobal - XP Taskpanel Global**".

Filenames NOTE!
DLL's with a Taskpanel control must not be renamed after compilation. If the dll is not found by its compile-time name, in the current dir or path, Taskpanel will be unable to load the icons from that DLL. Instead it will try to load them from the .exe file (which in most cases will lead to "invisible" icons)...

If you MUST rename your DLL. The programmer is responisble for setting the correct filename, at runtime, with the method SetAppName.

3.2 msimg32.dll

The API calls for gradients resideds in msimg32.dll.
If this dll is not present on the client system, no gradients will be drawn. Plain colors will be used instead.

msimg32.dll is included by default in Windows 98/2000 and later, and is subject of copyright by Microsoft Corp.

3.3 Installed Files

Below is a list of files installed with the Icetips Taskpanel. Please note that the file size may NOT be exactly match what you have on your drive.

Install Files

ID	File	Destination Folder / Source Folder	Last Modified	Size (bytes)
00001	c6_vista_fix.exe	%PROGRAMFILESDIR%\Icetips Creative\Clarion6Fix C:\Products	2007/08/14 12:40:50	149,864
00002	XPTaskPanel.pdf	%CLA_DOCS%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Docs\XPTaskpanel	2008/12/30 11:21:59	382,065
00003	itutil.exe	%CLA_UTILS% C:\Products\XPTaskpanel\Latest\3rdParty\Tools\ITInstall	2009/01/02 10:36:45	1,340,312
00004	XPTaskpanel.itc	%CLA_UTILS% C:\Products\XPTaskpanel\Latest\3rdParty\Tools\ITInstall	2008/12/27 14:38:42	334
00005	XPTaskpanel.chm	%CLA_DOCS%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Docs\XPTaskpanel	2008/12/30 11:22:04	258,191
00006	earth.ico	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2003/10/06 22:32:34	26,694
00007	help2.ico	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2003/10/06 22:37:40	26,694
00008	key1.ico	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2003/10/06 22:33:06	26,694
00009	quickstart_C7.app	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2008/12/15 12:39:15	45,680
00010	quickstart_C7.cwproj	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2008/12/15 12:39:07	1,796
00011	quickstart_C7.sln	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2008/12/15 12:37:38	1,066

00012	xptaskpaneldemo_C7.app	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2008/12/15 12:43:34	417,752
00013	xptaskpaneldemo_C7.cwproj	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2008/12/15 12:42:32	2,564
00014	xptaskpaneldemo_C7.sln	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\CSIDL_COMMON_DOCUMENTS\SoftVelocity\Icetips\XPTaskpanel	2008/12/15 12:39:48	1,086
00015	document_text.ico	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Examples\XPTaskpanel	2003/10/06 22:36:48	26,694
00016	earth.ico	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Examples\XPTaskpanel	2003/10/06 22:32:34	26,694
00017	help2.ico	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Examples\XPTaskpanel	2003/10/06 22:37:40	26,694
00018	key1.ico	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Examples\XPTaskpanel	2003/10/06 22:33:06	26,694
00019	quickstart_c55.app	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Examples\XPTaskpanel	2004/08/10 19:49:18	17,920
00020	taskpanel_xptheme_c6.app	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Examples\XPTaskpanel	2008/12/15 12:43:02	270,080
00021	xptaskpaneldemo_c55.app	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Examples\XPTaskpanel	2004/08/05 21:26:10	193,280
00022	xptaskpaneldemo_c6.app	%CLA_EXAMPLES%\XPTaskpanel C:\Products\XPTaskpanel\Latest\3rdParty\Examples\XPTaskpanel	2008/12/15 12:43:57	264,704
00023	ITRun32.dll	%CLA_BIN% C:\Products_Binaries	2008/12/27 19:13:37	59,904
00024	document_text.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2003/10/06 22:36:48	26,694

00025	earth.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2003/10/06 22:32:34	26,694
00026	help2.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2003/10/06 22:37:40	26,694
00027	ITLogo.gif	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2007/01/03 22:31:57	2,847
00028	key1.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2003/10/06 22:33:06	26,694
00029	taskxpc.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2008/12/15 11:44:51	1,758
00030	taskxpcs.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2008/12/15 11:44:51	1,758
00031	taskxpcsw.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2008/12/15 11:44:51	1,758
00032	taskxpcw.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2008/12/15 11:44:52	1,758
00033	taskxpe.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2008/12/15 11:44:52	1,758
00034	taskxpes.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2008/12/15 11:44:52	1,758
00035	taskxpesw.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2008/12/15 11:44:52	1,758
00036	taskxpew.ico	%CLA_IMAGES% C:\Products\XPTaskpanel\Latest\3rdParty\Images	2008/12/15 11:44:52	1,758
00037	potaskpanel.clw	%CLA_LIBSRC% C:\Products\XPTaskpanel\Latest\3rdParty\Libsrc	2008/12/30 11:22:39	96,095

00038	potaskpanel.inc	%CLA_LIBSRC% C:\Products\XPTaskpanel\Latest\3rdParty\Libsrc	2008/12/30 11:22:39	13,416
00039	PoXPtask.tpl	%CLA_TEMPLATE% C:\Products\XPTaskpanel\Latest\3rdParty\Template	2008/12/30 11:22:39	3,416
00040	PoXPtaskCW.tpl	%CLA_TEMPLATE% C:\Products\XPTaskpanel\Latest\3rdParty\Template	2008/12/30 11:22:39	3,030
00041	PoXPtaskGrp.tpw	%CLA_TEMPLATE% C:\Products\XPTaskpanel\Latest\3rdParty\Template	2008/12/30 11:22:40	49,666
00042	XPTaskP.hlp	%CLA_BIN% C:\Products\XPTaskpanel\Latest\CSIDL_PROGRAMFILES_COMMON\Icetips Creative\Clarion\Bin	2008/12/30 11:22:15	206,518
00043	bg.jpg	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/15 07:53:23	2,851
00044	body-left.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/15 10:32:33	844
00045	body-right.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/15 10:32:25	844
00046	down.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/08/05 15:10:46	825
00047	footer.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/14 22:07:31	27,226
00048	hdg-left.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/14 22:06:05	26,066
00049	hdg-right.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/14 22:06:19	27,485
00050	menu-bg.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/14 22:06:45	839

00051	menu-left.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/15 10:29:38	1,091
00052	menu-right.gif	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support\images C:\Products_support\images	2008/12/15 10:29:20	1,091
00053	favicon.ico	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support C:\Products_support	2008/12/15 09:20:05	1,150
00054	IcetipsDirectSupport.htm	%_CSIDL_COMMON_APPDATA%\Icetips Creative\Support C:\Products_support	2008/12/29 11:19:16	29,296
00055	ITRun32.dll	%_CSIDL_PROGRAM_FILES_COMMON%\Icetips Creative\Clarion\Bin C:\Products_Binaries	2008/12/27 19:13:37	59,904

Results

Build log was saved at:
C:\Products\XPTaskpanel\Latest\XPTaskpanel\XPTaskpanel_install.htm
0 error(s), 0 warning(s)

3.4 Upgrade issues

Upgrading from version 1.4 or earlier

Some groups has been renamed to be consistent with the prefixing added to API prototypes. If you have handcode using one of the following groups, this code has to be changed to use the new names:

Old name	New name
POColorGrp	PTP:ColorGrp
POTaskInfo	PTP:TaskInfo

Upgrading from version 1.3.1 or earlier

Insert the Global template "XPTaskPanelGlobal - XP Taskpanel Global" to all apps using the control-template.
You should also add this template to your data-dll, and remove the deprecated multi-dll template.

From version 1.4 Taskpanel is full sourcecode. Thiis means that no extra dll's needs to be deployed with your application.

Upgrading from version 1.22 or earlier

1.
To improve the template, some changes where necessary. The most significant change is the ID-system. This is the only change that requiers you to modify your code (a little bit).

In your embed-code, all ID's should be prefixed with the name of the Taskpanel object (eg. Taskpanel1).

Old code:

```
Case HeaderID
Of HID:MyFirstHeader
    Case TaskID
    Of TID:Task1
        SomeCode()
   End
End
```

New code:

```
Case HeaderID
Of Taskpanel1.HID:MyFirstHeader
    Case TaskID
    Of Taskpanel1.TID:Task1
        SomeCode()
   End
End
```

Now you should delete all those local variables previously used to store the ID.

The template will also generate one embed point for each task, for you to put your code directly into, and forget about the Case-statements and ID's.

2.
The action "Execute control" has been removed. If you use this action in your template, you should change it to "Post Event".

Upgrading from version 1.11 or earlier

When uppgrading from version 1.11 to 1.2 you might have to do some changes to your application.

The embed-point for handcoded Task-actions (Control Events|?XPtaskPanel|Execute Task|Before Generated Code) has changed. If you use theese embeds, you will notice that they've all become "Orphaned".

The correct action to fix this is to cut'n'paste the code from the orphaned embed to the appropriate embedpoint.

For instance, you got one control named ?XPTaskpanel. Cut the code from the orphaned embedpoint and copy it to the embedpoint named:
"Control Events -> ?XPTaskpanel -> Execute Task -> Before Generated Code".

Recompile your application, and everything should work as it used to.

3.5 FAQ

I get the following error-message when trying to compile "Unknown template symbol %DontGenerateTaskpanelCode".
You should add the global template to your application. If you're upgrading from a previous version, please read the Upgrade issues section.

Can I use XPTaskpanel to control a browse in a different window?
Yes you can. But you have to handcode the actions in one of the embed-points. By using the Clarion functions Notify() or Post() you can activate the update buttons in another window.

Is Taskpanel compatible with ClarioNet?
No.

3.6 License Agreement

Icetips "Taskpanel" End-User License Agreement

Important - read carefully!

By installing this software you have agreed to be bound by the following End-User Licence Agreement.

ICETIPS ALTA LLC ("ICETIPS") IS WILLING TO LICENSE THE SOFTWARE ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS SOFTWARE LICENSE AGREEMENT. PLEASE READ THE TERMS CAREFULLY. BY CLICKING ON "YES, ACCEPT" OR BY INSTALLING THE SOFTWARE, YOU WILL INDICATE YOUR AGREEMENT WITH THEM. IF YOU ARE ENTERING INTO THIS AGREEMENT ON BEHALF OF A COMPANY OR OTHER LEGAL ENTITY, YOUR ACCEPTANCE REPRESENTS THAT YOU HAVE THE AUTHORITY TO BIND SUCH ENTITY TO THESE TERMS, IN WHICH CASE "YOU" OR "YOUR" SHALL REFER TO YOUR ENTITY. IF YOU DO NOT AGREE WITH THESE TERMS, OR IF YOU DO NOT HAVE THE AUTHORITY TO BIND YOUR ENTITY, THEN ICETIPS IS UNWILLING TO LICENSE THE SOFTWARE, AND YOU SHOULD SELECT THE "NO, DECLINE" BUTTON AND THE DOWNLOAD OR INSTALL WILL NOT CONTINUE.

SOFTWARE LICENSE AGREEMENT

1. Parties. The parties to this Agreement are you, the licensee ("You") and Icetips. If You are not acting on behalf of Yourself as an individual, then "You" means Your company or organization.

2. The Software. The Software licensed under this Agreement consists of computer programs only in compiled, object code form, data compilation(s), and documentation referred to as Icetips subscription product (the "Software").

3. Subscription Term For Registered User Version. The term of the license granted herein for the registered version of the Software shall be on a subscription basis with an initial term of one (1) year, and optional recurring renewal terms of one (1) year each, unless prior to renewal this license is terminated by written notice by You for convenience or terminated by either party for material breach. Renewal procedures are described in the accompanying documentation, and unless such procedures are strictly satisfied, including the payment of any required license fee, Your updates to the Software is not authorized, but use of Your existing Software is authorized. No updates or upgrades to the Software can be authorized unless the license fee is paid.

4. Registered Version License Grant for Single Copies (Non-Network Use). If You are a registered user of the Software, You are granted non-exclusive rights to install and use the Software by a single person who uses the Software only on one or more computers or workstations. You may copy the

Software for archival purposes, provided that any copy must contain the original Software's proprietary notices in unaltered form.

5. Registered Version License Grant For Network Use. If You are a registered user of the Software, You are granted non-exclusive rights to install and use the Software and/or transmit the Software over an internal computer network, provided You acquire and dedicate a licensed copy of the Software for each user who may access the Software concurrently with any other user. You may copy the Software for archival purposes, provided that any copy must contain the original Software's proprietary notices in unaltered form.

6. Restrictions. You may not: (i) permit others to use the Software, except as expressly provided above for authorized network use; (ii) modify or translate the Software; (iii) reverse engineer, decompile, or disassemble the Software, except to the extent this restriction is expressly prohibited by applicable law; (iv) create derivative works based on the Software; (v) merge the Software with another product; (vi) copy the Software, except as expressly provided above; or (vii) remove or obscure any proprietary rights notices or labels on the Software.

7. Purchase of Additional Licenses. Registered users of the Software may purchase license rights for additional authorized use of the Software in accordance with Icetips's then-current volume pricing schedule. Such additional licenses shall be governed by the terms and conditions hereof. You agree that, absent Icetips's express written acceptance thereof, the terms and conditions contained in any purchase order or other document issued by You to Icetips for the purchase of additional licenses, shall not be binding on Icetips to the extent that such terms and conditions are additional to or inconsistent with those contained in this Agreement.

8. Transfers. You may make a one-time permanent transfer of all of your license rights to the Software to another party, provided that all of the following conditions are satisfied: (a) you notify us in writing of your intent to transfer your license rights and identify the party receiving the Software with complete contact information; (b) the transfer must include all of the Software, including all its component parts, original media, printed materials and this License Agreement; (c) you do not retain any copies of any version of the Software, full or partial, including copies stored on a computer or other storage device; and (d) the party receiving the Software reads and agrees to accept the terms and conditions of this License Agreement. Notwithstanding the foregoing, we reserve the right to require the transfer of possession of all physical copies of the Software to us for purposes of re-issue of replacement copies to the party receiving the Software.

9. Ownership. Icetips and its suppliers own the Software, all physical copies thereof, and all intellectual property rights embodied therein, including copyrights and valuable trade secrets embodied in the Software's design and coding methodology. The Software is protected by United States copyright laws and international treaty provisions. This Agreement provides You only a limited use license, and no ownership of any intellectual property. We reserve the right to require

you to transfer possession of all physical copies of the Software to us for purposes of re-issue of replacement copies.

10. Warranty Disclaimer; Limitation of Liability. ICETIPS PROVIDES THE SOFTWARE "AS-IS" AND PROVIDED WITH ALL FAULTS. NEITHER ICETIPS NOR ANY OF ITS SUPPLIERS OR RESELLERS MAKES ANY WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. ICETIPS AND ITS SUPPLIERS SPECIFICALLY DISCLAIM THE IMPLIED WARRANTIES OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, SYSTEM INTEGRATION, AND DATA ACCURACY. THERE IS NO WARRANTY OR GUARANTEE THAT THE OPERATION OF THE SOFTWARE WILL BE UNINTERRUPTED, ERROR-FREE, OR VIRUS-FREE, OR THAT THE SOFTWARE WILL MEET ANY PARTICULAR CRITERIA OF PERFORMANCE, QUALITY, ACCURACY, PURPOSE, OR NEED. YOU ASSUME THE ENTIRE RISK OF SELECTION, INSTALLATION, AND USE OF THE SOFTWARE. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS AGREEMENT. NO USE OF THE SOFTWARE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

11. Local Law. If implied warranties may not be disclaimed under applicable law, then ANY IMPLIED WARRANTIES ARE LIMITED IN DURATION TO THE PERIOD REQUIRED BY APPLICABLE LAW. Some jurisdictions do not allow limitations on how long an implied warranty may last, so the above limitations may not apply to You. This warranty gives you specific rights, and You may have other rights which vary from jurisdiction to jurisdiction.

12. Limitation of Liability. INDEPENDENT OF THE FORGOING PROVISIONS, IN NO EVENT AND UNDER NO LEGAL THEORY, INCLUDING WITHOUT LIMITATION, TORT, CONTRACT, OR STRICT PRODUCTS LIABILITY, SHALL ICETIPS OR ANY OF ITS SUPPLIERS BE LIABLE TO YOU OR ANY OTHER PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND, INCLUDING WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER MALFUNCTION, OR ANY OTHER KIND OF COMMERCIAL DAMAGE, EVEN IF ICETIPS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY TO THE EXTENT PROHIBITED BY APPLICABLE LAW. IN NO EVENT SHALL ICETIPS'S LIABILITY FOR DAMAGES FOR ANY CAUSE WHATSOEVER, AND REGARDLESS OF THE FORM OF ACTION, EXCEED IN THE AGGREGATE THE AMOUNT OF THE PURCHASE PRICE PAID FOR THE SOFTWARE LICENSE.

13. Export Controls. You agree to comply with all export laws and restrictions and regulations of the United States or foreign agencies or authorities, and not to export or re-export the Software or any direct product thereof in violation of any such restrictions, laws or regulations, or without all necessary approvals. As applicable, each party shall obtain and bear all expenses relating to any necessary licenses and/or exemptions with respect to its own export of the Software from the U.S. Neither the Software nor the underlying information or technology may be electronically transmitted or otherwise exported or re-exported (i) into Cuba, Iran, Iraq, Libya, North Korea, Sudan,

Syria or any other country subject to U.S. trade sanctions covering the Software, to individuals or entities controlled by such countries, or to nationals or residents of such countries other than nationals who are lawfully admitted permanent residents of countries not subject to such sanctions; or (ii) to anyone on the U.S. Treasury Department's list of Specially Designated Nationals and Blocked Persons or the U.S. Commerce Department's Table of Denial Orders. By downloading or using the Software, Licensee agrees to the foregoing and represents and warrants that it complies with these conditions.

14. U.S. Government End-Users. The Software is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" and "commercial computer software documentation," as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire the Software with only those rights as are granted to all other end users pursuant to the terms and conditions herein. Unpublished rights are reserved under the copyright laws of the United States.

15. Licensee Outside The U.S. If You are located outside the U.S., then the following provisions shall apply: (i) Les parties aux presentes confirment leur volonte que cette convention de meme que tous les documents y compris tout avis qui siy rattache, soient rediges en langue anglaise (translation: "The parties confirm that this Agreement and all related documentation is and will be in the English language."); and (ii) You are responsible for complying with any local laws in your jurisdiction which might impact your right to import, export or use the Software, and You represent that You have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

16. Severability. If any provision of this Agreement is declared invalid or unenforceable, such provision shall be deemed modified to the extent necessary and possible to render it valid and enforceable. In any event, the unenforceability or invalidity of any provision shall not affect any other provision of this Agreement, and this Agreement shall continue in full force and effect, and be construed and enforced, as if such provision had not been included, or had been modified as above provided, as the case may be.

17. Arbitration. Except for actions to protect intellectual property rights and to enforce an arbitrator ’s decision hereunder, all disputes, controversies, or claims arising out of or relating to this Agreement or a breach thereof shall be submitted to and finally resolved by arbitration under the rules of the American Arbitration Association ("AAA") then in effect. There shall be one arbitrator, and such arbitrator shall be chosen by mutual agreement of the parties in accordance with AAA rules. The arbitration shall take place in Port Angeles, Washington, USA, and may be conducted by telephone or online. The arbitrator shall apply the laws of the State of Washington, USA to all issues in dispute. The controversy or claim shall be arbitrated on an individual basis, and shall not be

consolidated in any arbitration with any claim or controversy of any other party. The findings of the arbitrator shall be final and binding on the parties, and may be entered in any court of competent jurisdiction for enforcement. Enforcements of any award or judgment shall be governed by the United Nations Convention on the Recognition and Enforcement of Foreign Arbitral Awards. Should either party file an action contrary to this provision, the other party may recover attorney's fees and costs up to $1000.00.

18. Jurisdiction And Venue. The courts of Clallam County in the State of Washington, USA and the nearest U.S. District Court shall be the exclusive jurisdiction and venue for all legal proceedings that are not arbitrated under this Agreement.

19. Force Majeure. Neither party shall be liable for damages for any delay or failure of delivery arising out of causes beyond their reasonable control and without their fault or negligence, including, but not limited to, Acts of God, acts of civil or military authority, fires, riots, wars, embargoes, Internet disruptions, hacker attacks, or communications failures. Notwithstanding anything to the contrary contained herein, if either party is unable to perform hereunder for a period of thirty (30) consecutive days, then the other party may terminate this Agreement immediately without liability by ten (10) days written notice to the other.

20. Miscellaneous. This Agreement constitutes the entire understanding of the parties with respect to the subject matter of this Agreement and merges all prior communications, representations, and agreements. This Agreement may be modified only by a written agreement signed by the parties. If any provision of this Agreement is held to be unenforceable for any reason, such provision shall be reformed only to the extent necessary to make it enforceable. This Agreement shall be construed under the laws of the State of Washington, USA, excluding rules regarding conflicts of law. The application the United Nations Convention of Contracts for the International Sale of Goods is expressly excluded. The parties agree that the Uniform Computer Transactions Act or any version thereof, adopted by any state, in any form ("UCITA"), shall not apply to this Agreement, and to the extent that UCITA may be applicable, the parties agree to opt out of the applicability of UCITA pursuant to the opt-out provision(s) contained therein.

Icetips Alta LLC
3430 East Highway 101, Ste. #28
Port Angeles WA 98362
EMail: support@icetips.com
http://www.icetips.com

3.7 Version history

Version 2018.10.175 [October 14, 2018]

November 17, 2015:

- Classes — Equates all moved to the .inc file. Helps when deriving the Taskpanel class.

June 18, 2015:

- Template — "Use XP Theme colors (requires PowerXP-Theme)" would get turned OFF causing the colors to go off kilter. Fixed.

Version 2.0.163 February 24, 2015]

January 25, 2015:

- Template — Name for XPTheme global class would clash with XPTheme when used with PowerToolbar. Fixed.

Version2.0.161 [December 8, 2014]

December 8, 2014:

- Classes — Wrong icon was drawn for the right hand icon on header. Fixed.

July 22, 2014:

- Classes — Call to PTP:ShowScrollbar caused "Calling function as procedure" warning. Fixed.

Version 2.0.157 [July 3, 2014]

July 2, 2014:

- Classes — Horizontal scrollbar could show up at the bottom of Taskpanel controls in Clarion 9.1. build 11014 or newer. Fixed.

Version 2.0.155 [January 29, 2014]

January 29, 2014

- Installer: — Installer is now compatible with Clarion 9.1
- Installer: — XPTheme files are now included in the Taskpanel Install.

January 10, 2014

- Classes: Using ULONG for SetColorGrp and FixColor would fail if Taskpanel used XPTheme. Fixed.
- Classes: GetVisible would fail when called with a task ID. Fixed.
- Classes: SetColorGrp and FixColor where using LONG for the color parameters and this caused the color to turn black. Changed to ULONG. Fixed.
- Templates: Updated export section to reflect changes in SetColorGrp and FixColor prototypes.

September 27, 2013

- Classes: Default Y position for header icons was set incorrectly causing overlap of the header and icon. Fixed.

Version 2.0.148 [August 1, 2013]

June 6, 2013

- Install: Install built for Clarion 9.0 beta

June 3, 2013

- Classes: Added multiple methods to handle icons in headers. Note that all size values are in pixels.

```
!! AB 2013-05-04:  Added properties for Header Icons.
Using Short rather than Byte, since some values can be
negative
HeaderHeight            Short

!! Left side header icon
HeaderLeftIconTMargin   Short
HeaderLeftIconLMargin   Short
HeaderLeftIconSize      Short
HeaderRightIconTMargin  Short
HeaderRightIconSize     Short

!! Right side header icon
HeaderRIconNameNormal   String(256)  !! Collapsed
HeaderRIconNameExpand   String(256)  !! Expanded
HeaderRIconNameHover    String(256)  !! Collapsed +
hover
HeaderRIconNameExpHover String(256)  !! Expanded +
hover
HeaderRIIconUseFont     Byte(0)  !! Set to True if the
SetRightIconFont is called.  Defaults to 1.  Set to
False if SetRightIcon is called
HeaderRIconFontName     String(60)
HeaderRIconFontSize     Byte
HeaderRIFontColNormal   Long
HeaderRIconColExpand    Long
HeaderRIconColHover     Long
```

```
!left icon top margin      left icon left m.   left
icon size      right icon top m.   right icon size
SetHeaderSizes          Procedure(<Short hHeight>,
<Short hLITmargin>, <Short hLILMargin>, <Short
hLIconSize>, <Short hRITMargin>, <Short hRIconSize>)
SetheaderHeight         Procedure(Short hHeight)
SetLeftIconMargins      Procedure(<Short hLITmargin>,
<Short hLILMargin>)
SetLeftIconSize         Procedure(Short hLIconSize)
SetRightIcon            Procedure(String hIconName,
String hIconNameExpanded, String hIconNameHovering,
String hIconNameExpandedHovering, Short hIconSize)
SetRightIconFont        Procedure(String hIconFontName,
<Byte hIconFontSize>, <Long hIconFontCol>, <Long
hIconFontColExp>, <Long hIconFontColHov>)
SetRightIconSizes       Procedure(<Short hRITMargin>,
<Short hRIconSize>)
FormatIconPath          Procedure(String
pIconName),String  !! AB 2013-05-07:  Modified
```

Those methods are not yet documented.

Version 2.0.145 [May 4, 2011]

May 4, 2011

- Install: Install built for Clarion 8.0.

December 7, 2010

- Install: Install built for Clarion 7.3. Tested with Clarion 7.3.7852 and no problems were detected.

November 23, 2010

- Template: SetTaskMimicTip and GetTaskMimicTip were not included in the list of exports. Fixed.
- Template: Some class method exports were not correctly formed. Fixed.
- Classes: Ampersands were not dealt with correctly in enabled and disabled items. Fixed.
  Note: Currently ampersands are shown exactly like they are except in mimic controls, where they are removed. I.e. "Name & Address" will show up exactly like that. "Name && Address" will show up as "Name && Address".

Version 2.0.137 [November 10, 2010]

November 11, 2010

- Class: When the SetVisible method is used on a headers and hid and unhide other headers, it could leave tooltip artifacts below the visible tasks from the previous visible header. Fixed.

November 10, 2010

- Class: When the SetVisible method is used on a headers and unhide other headers, it could leave the task tooltips out of sync. Fixed.

November 9, 2010

- Classes: Class source adapted to Icetips standard, i.e. two lines between methods and double line above and single line below method declaration for readability.

August 30, 2010

- Help: Typo corrected "Taskpanle" to "Taskpanel"

August 2, 2010

- Product: Name changed from XP-Taskpanel to just Taskpanel.

July 27, 2010

- Template: Added option to Mimic Buttons option to mimic tooltips. Defaults to true so tooltips are now automatically added to tasks that mimic buttons.

July 21, 2010

- Class: Added ClearActiveWizardTask method. Also added to export list in template. This method clears the visual wizard settings from the active wizard task, returning the wizard to "normal" settings. This includes removing font styles and active icon if it is specified.

July 6, 2010

- Template: Legacy (Clarion chain) did not support %EnableRunTimeTranslator. Fixed.

Version 2.0.135 [June 6, 2010]

February 7, 2010

- Template: XPTaskpanel template did not support runtime translation. Fixed.

July 16, 2009

- Template: Added two new code templates to set the Header Text and the Task Text.

June 12, 2009

- Classes: Added SetFontSize and GetFontSize methods to the class. Also modified the SetFont method by adding an omittable size parameter. This allows setting both the fontname and font size.
- Template: Modified export list to reflect changes in classes.

Version 2.0.124 [May 13, 2009]

■ Classes	When using XP Taskpanel along with XP-Themes, disabled task items were drawn in the same color as normal task items when the disable color was overridden. Added option to override the XP Theme colors for disabled items, which will draw the disabled tasks in the specified color for disabled tasks. Fixed.

Version 2.0.122 [April 15, 2009]

■ Classes	Disabled task items were still drawn with a hotkey underline where an ampersand was in the text. For example "Name & Address" would be drawn as "Name _Address". Fixed.
■ Documentation	The SetHeaderIcon and SetItemIcon methods were using the wrong method name in the example code. Fixed.

Version 2.0.112 [January 2, 2009]

■ Install	The xptaskpane.chm file was missing from the install. Fixed.

Version 2.0.111 [December 29, 2008]

- All code and documentation modified from PowerOffice AS to Icetips Creative, Inc.
- Bug tracking set up at http://icetips.fogbugz.com
- Added link to bug tracking to Global template
- Updated link to website to point to www.icetips.com
- Added ITRun32.dll to install
- Version information added to global template
- Previous beta changes are all implemented in public release - see below:
- Added documentation for new methods, such as ContractAll, DeleteAllTasks, ExpandAll, GetHeaderTitle, SetHeaderFontWeight, SetHeaderTitle and SetTaskSpacing

1.6 BETA 9 - not released publicly before

- New: Support for Windows Theme colors (requires PowerXP-Theme)
- New methods: GetHeaderTitle, SetHeaderTitle
- Change: SetColorGrp prototype changed from ULONG to LONG to allow COLOR:None
- Fixed: GPF in reports
- Fixed: Listboxes doesn't refresh content
- Fixed: Bug in id-generator

1.6 BETA 8 - not released publicly before

- New methods: SetHeaderIcon, GetHeaderIcon
- Change: SetActiveWizardTask accepts 0,0 as argument to deselect current wizard task

1.6 BETA 7 - not released publicly before

- Added methods: Disable, Enable and IsDisabled
- Added: Separator option for tasks

1.6 BETA 6 - not released publicly before

- Fixed a bug that caused GPF in C55 build A-F
- Fixed a bug that could cause non-dropping DropLists
- Removed the 'no win' message

1.6 BETA 5 - not released publicly before

- New methods: DeleteAllTasks, SetHeaderFontWeight
- Template now shows header and task ID's in lists
- Option to turn off bold for headers

1.6 BETA 4 - not released publicly before

- Mimiced buttons wasn't unsubclassed when calling DeleteTask
- Added a call to Update just before calling embed-code

1.6 BETA 3 - not released publicly before

- Added flags to avoid WM_LBUTTONUP to fire when closing another window by double-clicking

1.6 BETA 2 - not released publicly before

- New method: SetTaskSpacing
- Added template prompt for task-spacing

1.6 BETA 1 - not released publicly before

- Added option to use icons to indicate selected wizard task
- New method: SetWizardStyle
- Added support for fallback colors when the desktop uses 256 or less colors
- New methods: SetFallbackColors, SetFallbackMode
- New method: SetDefaultTaskMargin
- Added support for right mouse button

Version 1.5 [August 10, 2004]

- Added "Mimic Text" and changed "Don't mimic icon" to "Mimic Icon" for "Mimic Button"-action
- Added "Reset to default colors" button
- Added "Wizard mode"
- Added Balloon-tooltip mode
- Added Boxed-property for headers
- Added new "Tooltip" argument to AddTask-protype
- Added new task action: Set Field Value
- Added new template prompt: Only allow one expanded header
- Added optional argument to skip a header when calling ContractAll and ExpandAll
- Added prefix to all API prototypes and some groups to avoid conflicts with other templates
- Added support for non-linked icons
- Added Tooltip support
- Added word-wrapping for tasks
- Changed the ID-generator to limit ID's to 127 characters
- Changed one of the AddTask-prototypes to avoid confusing the compiler
- Changed the window subclassing to use a safer way of storing class references
- Fixed a bug causing scrollbar buttons to disappear when compiled in C6.1
- Fixed a bug causing the cursor to disappear on Windows NT4
- Fixed a bug in the template ID-generator
- Fixed a bug that could cause a blank taskpanel

- Fixed a bug that could cause bad filename for locally linked mimiced icons
- Fixed bug causing the variable to be added to the project if iconname is variable
- Fixed bug that could cause underlining of the last selected task when the mouse is moved off the control
- Fixed bug with embed-points causing compile-time GPF
- Fixed mousecursor flicker
- Init code is moved into the new virtually derived Init-method
- Moved header-text a few pixels to align with tasks
- New embed: Before and After Init-code
- New embed: Init->Before Parent Call
- New embed: Before and after execute-action
- New embed: Before and after task logic
- New methods: GetTaskTitle, GetExpanded, GetVisible, GetDisabled
- New methods: GetTaskIcon and SetTaskIcon
- New methods: SetHeaderBoxed, GetHeaderBoxed
- New methods: SetTaskMimicText, GetTaskMimicText, SetTaskMimicIcon, GetTaskMimicIcon
- New methods: SetTaskTooltip, GetTaskTooltip, SetTaskUserData and GetTaskUserData
- New methods: SetTooltipMaxWidth, GetTooltipMaxWidth, SetTooltipMode, GetTooltipMode
- New methods: SetWizardMode, GetWizardMode, SetDisableWizard, GetDisableWizard, SetActiveWizardTask
- New methods: SetWordWrap, GetWordWrap, SetHeaderWordWrap, GetHeaderWordWrap
- Removed many unused equates
- Removed the use of TransparentBlt as it has a memory-leak in Win95/98 which was causing problems
- Reworked the template-GUI
- Rewrote mouse-move and mouse-click handlers
- Template generated class methods are now generated as DERIVED instead of VIRTUAL
- When "Only one expanded header" is checked, all but the first expanded header will be contracted
- When a task mimics a button, and no compiletime icon-name is available, PROP:Icon is used instead

Version 1.4 [April 28, 2004]

- Added UNIQUE-check when generating ID's in the template
- Added refreshing on a few more events
- Added embed-point for header clicks
- Added the IsExpanded member to the POTaskInfo-group
- Added text justification options for tasks
- Added BottomMargin setting
- Added ExpandAll/ContractAll methods
- Added "Select Tab" action for tasks
- Added "Don't Generate Template Code" option
- Added a optional "Thread" option to "Post Event"-action
- Added dynamic headers and tasks
- Added the possibillity to disable a task
- Added the possibillity to draw a task with bold text
- Added optional argument for SetColorGrp to set disabled text color
- Added option to not mimic icon on "Mimic Button" action
- Added a check to see if the mouse is over the control when scrolling with the mousewheel
- Added SetTaskTitle method
- Added option to set a header to be "always expanded"
- Added Global template
- Added template option for init-code priority
- Added a dynamic linkloader-class
- Changed to be sourcecode only (no more blackbox DLL's)
- Changed the "Control" in the "Post Event"-action to be optional
- Changed default font to MS Sans Serif

- Changed the internal storage of a static class reference (removed the need fore some static data)
- Removed the call to Kill() from the template (it's now called in the destructor)
- Removed the multi-dll template (use the global template instead)
- Removed some ABC-specific code in the template
- Moved the code for expanding/contracting headers fromt the class to the template.
- Fixed bug causing a glitch between the header and tasks, when the first task is hidden
- Fixed issue with the taskpanel not showing when opening maximized MDI-children
- Fixed issue with icons not in app-dir or Clarion6\images
- Fixed issues with mimic-tasks when opening a form
- Fixed issue with tasks not showing on window reopening

Version 1.3.1 [December 7, 2003]

- Removed a memory leak in the C55 version

Version 1.3 [December 5, 2003]

- Redesigned the ID-system
- Improved icon-loading
- Rewrote the Multi-DLL template
- Added embed-point for each task
- Added "Call a procedure"-action
- Added "Post Event"-action
- Added SetExpanded-method
- Added SetAppName-method (for Multi-DLL apps)
- Added support for STD-colors (eg. COLOR:BtnFace)
- Removed "Execute Control"-action (use "Post Event" instead)
- Fixed the "invalid ordinal" C55 bug.
- Updated the documentation

Version 1.22 [December 3, 2003]

- Moved some template generated code to avoid clashing with other templates
- Fixed icon cache bug

Version 1.21 [December 15, 2003]

- Added possibillity to set margins

Version 1.2 [November 11, 2003]

- Fixed a bug with the embed-point (new ABC-classes could cause the embed-point to become Orphaned)
- Transparency bug in Win98SE

Part

Chapter 4 - Reference

4 Reference

4.1 Embed points

The Taskpanel control template generates several embed points. With theese embed-points you can control actions and the behaviour of the control with your own customized sourcecode.

Have a look at the Class reference for a list of available functions.

All embed points is generated under "Local Objects-><object name>". For instance "Local Objects->XPTaskpanel1".

Header embeds

Embed points
```
Local Objects->XPTaskpanel1->Header1->Header Clicked->Before Generated Code
Local Objects->XPTaskpanel1->Header1->Header Clicked->After Generated Code
```

Available properites
`HeaderID` - The unique ID of the header beeing clicked
`MouseBtn` - Mouse button that activated the header

Theese embed points are executed everytime a header is clicked. If you want to avoid the header beeing contracted or expaned, you can execute a `return` in the "Before Generated Code"-embed.

Task embeds

Embed points
```
Local Objects->XPTaskpanel1->Header1->Task_1->Task Clicked->Before Generated Code
Local Objects->XPTaskpanel1->Header1->Task_1->Task Clicked->After Generated Code
```

Available properites
`HeaderID` - The unique ID of the parent header of the task beeing clicked
`TaskID` - The unique ID of the task beeing clicked
`MouseBtn` - Mouse button that activated the header

Embeds are executed whenever a task is clicked. You can avoid any action beeing executed by doing a `return` in the "Before Generated Code"-embed.

Init embeds

Embed points:
```
Local Objects->XPTaskpanel1->Init->Before Generated Code
Local Objects->XPTaskpanel1->Init->Before Parent Call
Local Objects->XPTaskpanel1->Init->After Generated Code
```

Available properties:
`nControl` - The XPTaskpanel control's Feq.

Embededed code is execute when the taskpanel is beeing initialized (right after window opening)

Execute Action embeds

Embed points:
```
Local Objects->XPTaskpanel1->Execute Action->Before Generated Code
Local Objects->XPTaskpanel1->Execute Action->After Generated Code
```

Availale properties:
`HeaderID` - The unique ID of the parent header of the task beeing clicked
`TaskID` - The unique ID of the task beeing clicked
`MouseBtn` - Mouse button that activated the header

Theese embeds are executed every time a task or header is clicked.

Task logic embeds

Embed points:
```
Local Objects->XPTaskpanel1->Header1->Task Logic->Before Generated Code
Local Objects->XPTaskpanel1->Header1->Task Logic->After Generated Code
```

Availale properties:
`HeaderID` - The unique ID of the parent header of the task beeing clicked
`TaskID` - The unique ID of the task beeing clicked
`MouseBtn` - Mouse button that activated the header

Embedpoints are called just before and after a task in this header has been clicked.

4.2 Methods by category

POTaskpanelClass contains the following public methods:

Click here for a alphabetized list of functions.

Header functions
- AddHeader
- DeleteHeader
- GetExpanded
- SetExpanded
- SetAlwaysExpanded
- GetVisible
- SetVisible
- GetHeaderWordWrap
- SetHeaderWordWrap
- SetHeaderBoxed
- GetHeaderBoxed

Task functions
- AddTask
- DeleteTask
- GetDisabled
- SetDisabled
- GetTaskIcon
- SetTaskIcon
- GetTaskInfo
- SetTaskInfo
- GetVisible
- SetVisible
- SetTaskUserData
- GetTaskUserData
- SetTaskTitle
- GetTaskTitle
- GetTaskMimicIcon
- GetTaskMimicText
- SetTaskMimicIcon
- SetTaskMimicText
- SetTaskFontWeight
- SetJustification

Tooltip functions
- SetTooltipMode
- GetTooltipMode
- SetTaskToolTip
- GetTaskToolTip
- SetTooltipMaxWidth
- GetTooltipMaxWidth

Wizard functions
- GetWizardMode
- SetWizardMode
- SetActiveWizardTask
- SetDisableWizard
- GetDisableWizard
- SetWizardStyle

Style functions
- SetColorGrp
- SetFont
- SetMargins
- SetDefaultTaskMargin
- SetFallbackColors
- SetFallbackMode
- SetAlwaysExpanded

Word-wrapping functions
- GetWordWrap
- SetWordWrap
- GetHeaderWordWrap
- SetHeaderWordWrap

Other functions
- Init
- Refresh
- RGB
- SetAppName
- ExecuteAction
- Disable
- Enable
- IsDisabled

Also, have a look at the Embed points section for appropriate embed points.

4.3 Methods by alphabet

POTaskpanelClass contains the following public methods:

Click here for a categorized list of functions.

- AddHeader
- AddTask
- DeleteHeader
- DeleteTask
- Disable
- Enable
- ExecuteAction
- GetDisabled
- GetExpanded
- GetHeaderBoxed
- GetHeaderWordWrap
- GetTaskIcon
- GetTaskInfo
- GetTaskMimicIcon
- GetTaskMimicText
- GetTaskTitle
- GetTaskToolTip
- GetTaskUserData
- GetTooltipMaxWidth
- GetTooltipMode
- GetVisible
- GetWizardMode
- GetWordWrap
- Init
- IsDisabled
- Refresh
- RGB
- SetActiveWizardTask
- SetAlwaysExpanded
- SetAppName
- SetColorGrp
- SetDefaultTaskMargin
- SetDisabled
- SetDisableWizard
- GetDisableWizard
- SetExpanded
- SetFallbackColors
- SetFallbackMode
- SetFont
- SetHeaderBoxed
- SetHeaderWordWrap
- SetJustification
- SetMargins
- SetTaskFontWeight
- SetTaskIcon
- SetTaskInfo
- SetTaskMimicIcon
- SetTaskMimicText
- SetTaskTitle
- SetTaskToolTip
- SetTaskUserData

- SetTooltipMaxWidth
- SetTooltipMode
- SetVisible
- SetWizardMode
- SetWizardStyle
- SetWordWrap

Also, have a look at the Embed points section for appropriate embed points.

4.4 Methods

4.4.1 AddHeader Methods

This function adds a new header to the taskpanel. The header will be visible at the next redraw.

Prototype:
AddHeader(STRING Title, BYTE nStyle, <BYTE IsExpanded>, <STRING IconName>, <BYTE IconSizes>),LONG,PROC

Arguments:

Title	This is the text that the header will display
nStyle	Visual style type. HSTYLE_NORMAL - Standard blue header HSTYLE_HIGHLIGHTED - White header
IsExpanded	Optional. If false, the header will be initially contracted (defaults to true)
IconName	Optional. The name of the icon that should be shown in the header. If ommited, no icon is shown. NOTE: The icon MUST be linked into your application.
IconSizes	Optional, The icon size used for tasks in this header. This size will affect the height of each task. If icon size is less then 20 the task-height will be set to 20.

Return value:
This function returns a LONG, representing an unique ID for this header.

See also:
AddTask

Example:
```
  HID:XpTaskpanel = TaskPanel5.AddHeader('XP taskpanel', HSTYLE_NORMAL, True,
'xpprosjekt.ico', 24)
```

4.4.2 AddTask Methods

Adds a task to a header. The task will be visible on the next redraw.

Prototypes:
AddTask(LONG HeaderID, STRING Title, LONG ActionFeq, <BYTE MimicButton>, <STRING IconName>, <BYTE DontMimicIcon>, <BYTE IsSeparator>),LONG,PROC
AddTask(STRING Title, LONG ActionFeq, BYTE MimicButton, STRING IconName, BYTE DontMimicIcon, STRING Tooltip, <BYTE IsSeparator>),LONG,PROC

Arguments:

HeaderID	A previously obtained ID for parent header the task should be added to. If you ommit this value (i.e. use the first prototype) the task will be added to the last added header.
Title	This is the screentext for this task
ActionFeq	The FEQ (Field Equate Label) for the control that should receive an EVENT:Accepted, when this task is clicked. Set this to zero if you don't want the task to activate a control.

MimicButton	Optional. If a button was specifed as the ActionFeq, you could set MimicButton to True. The task will then be hidden whenever the button is disabled. When the button is enabled, the task will be visible to the user. (Defaults to false)
IconName	Optional. The name of the icon that should be shown to the left of the task. If ommited, no icon is shown. NOTE: The icon MUST be linked into your application.
DontMimicIcon	Optional, If set to True and MimicButton is specified, the icon set in the template settings will be used instead of the icon of the mimiced button.
Tooltip	Tooltip for this task.
IsSeparator	Optional. If you set this to True, the task will become a separator

Return value:
This function returns a LONG, representing an ID for this task. The ID is only uniqe within its header. Tasks in different headers could have the same ID.

See also:
AddHeader

Example:
```
  TaskPanel5.AddTask('Introduction', ?Button1{PROP:Feq}, False, 'user.ico',, 'This is a tooltip...')
```

4.4.3 ClearActiveWizardTask Methods

Clears the wizard style from the wizard styled item. This includes changing the font style and removing the active icon if it is set.

Prototype:
ClearActiveWizardTask()

See also:
SetWizardMode, SetDisableWizard, GetDisableWizard, SetActiveWizardTask, SetWizardStyle

4.4.4 ContractAll Methods

Contracts all headers with an optional exception of one header.

Prototype:
ContractAll(<LONG ExceptHeaderID>)

Arguments:

ExceptHeaderID	The unique ID of the header to exclude from the contraction.

See also:
ExpandAll

4.4.5 DeleteAllTasks Methods

Delete all tasks for the specified header. The method calls the DeleteTask method to delete each task in the header.

Prototype:
DeleteAllTasks(LONG HeaderID)

Arguments:

HeaderID	The unique ID of the header in wich the task resides.

See also:
DeleteTask

4.4.6 DeleteHeader Methods

This function removes a header from the taskpanel.

Prototype:
DeleteHeader(LONG HeaderID)

Arguments:

HeaderID	An unique header ID.

See also:
DeleteTask

Example:

```
TaskPanel.DeleteHeader(Taskpanel1.Header_1)
```

4.4.7 DeleteTask Methods

This function removes a task from a header.

Prototype:
DeleteTask(LONG HeaderID, LONG TaskID)

Arguments:

HeaderID	An unique header ID.
TaskID	The ID of the task that should be deleted

See also:
DeleteHeader

Example:

```
TaskPanel.DeleteTask(Taskpanel1.Header_1, Taskpanel1.Header_1:Task_1)
```

4.4.8 Disable Methods

Disables the taskpanel. Mouse events will not be monitored while the taskpanel is disabled.

Prototype:
Disable()

4.4.9 Enable Methods

Enables the taskpanel.

Prototype:
Enable()

4.4.10 ExecuteAction Methods

Every class based on the POTaskPanelClass must implement this method.
Whenever a task is clicked, this method will be called, and you use this method to implement actions for your tasks.

Prototype:
ExecuteAction(Long HeaderID, Long TaskID, Long MouseBtn, Long Feq),VIRTUAL

Arguments:

HeaderID	This is the uniqe ID of the header in which the clicked task resides.
TaskID	The ID of the task. This value is only unique within it's header
MouseBtn	Which mousebutten was used to activate the task (currently only MouseLeft is supported)
Feq	The "Field Equate Lable" of the control that should be executed.

Example:

```
TaskPanel5      Class(POTaskPanelClass)
ExecuteAction       Procedure(Long HeaderID, Long TaskID, Long MouseBtn, Long Feq),VIRTUAL
              End

!! Handle Mouse-clicks on tasks
TaskPanel5.ExecuteAction   Procedure(Long HeaderID, Long TaskID, Long MouseBtn, Long Feq)
   Code
   If MouseBtn = MouseLeft
      Case HeaderID
      Of HID:XpTaskpanel
         Select(?Sheet1, TaskID)
         Post(EVENT:NewSelection, ?Sheet1)
         Post(EVENT:Accepted, ?Sheet1)

      Of HID:Demos
         Select(?Sheet1, 2+TaskID)
         Post(EVENT:NewSelection, ?Sheet1)
         Post(EVENT:Accepted, ?Sheet1)
```

```
    Of HID:HideUnhide
       Message('You selected Task ' & TaskID,'Hide/Unhide Demo',ICON:Asterisk,'Ok',1)
    End
  End

  If Feq > 0 Then Post(EVENT:Accepted, Feq) .
```

4.4.11 ExpandAll Methods

Expands all headers with an optional exception of one header.

Prototype:
ExpandAll(<LONG ExceptHeaderID>)

Arguments:

ExceptHeaderID	The unique ID of the header to exclude from the expansion.

See also:
ContractAll

4.4.12 Fix Color Methods

Returns the proper API color for Clarion colors and Clarion system colors.

Prototype:
FixColor Procedure(ULONG cIn)

Arguments:

cIn	Clarion color to check

See also:
SetColorGrp

4.4.13 GetDisabled Methods

Is the task disabled.

Prototype:
GetDisabled(LONG HeaderID, LONG TaskID),BYTE

Arguments:

HeaderID	The unique ID of the header.
TaskID	The unique ID of the task.

Returns
Returns True if the task is disabled, False otherwise.

See also:
SetDisabled

4.4.14 GetDisableWizard Methods

Returns if the wizard-mode has been disabled for this header .

Prototypes:
GetDisableWizard(LONG HeaderID),BYTE

Arguments:
HeaderID A previously obtained header-ID.

Returns:
True if wizard-mode has been disabled for this header, False otherwise

See also:
ClearActiveWizardTask, SetWizardMode, GetWizardMode, SetDisableWizard, SetActiveWizardTask, SetWizardStyle

4.4.15 GetExpanded Methods

Retrieves the expanded state of a header.

Prototype:
GetExpanded(LONG HeaderID),BYTE

Arguments:
HeaderID The unique ID of the task's parent header

Returns
Returns True if the header is expanded, False otherwise.

See also:
SetExpanded

4.4.16 GetHeaderBoxed Methods

This method returns whether the header is boxed or not.

When a header is boxed, it will be drawn as a single line. This results in a box with task in it.

Prototype:
GetHeaderBoxed(LONG HeaderID),BYTE

Arguments:
HeaderID Unique header ID

Returns
Returns True if the the "Boxed"-property is set for the header, False otherwise.

See also:
SetHeaderBoxed

4.4.17 GetHeaderIcon Methods

Retrieves the iconname of a header.

Prototype:
GetHeaderIcon(LONG HeaderID),STRING

Arguments:
HeaderID The unique ID of the header

Returns
Returns a STRING containing the header iconname. If the header was not found, an empty string is returned.

4.4.18 GetHeaderTitle Methods

Retrive the header title string

Prototype:
GetHeaderTitle(LONG HeaderID),STRING

Arguments:
HeaderID The unique ID of the header.

Returns
If the HeaderID is valid the function returns the header string. Otherwise it returns an empty string.

See also:
SetHeaderTitle

4.4.19 GetHeaderWordWrap Methods

Use this method to find out if wordwrapping is enabled or disabled for task in a header.

Prototype:
GetHeaderWordWrap(LONG HeaderID),BYTE

Arguments:

HeaderID	Unique header ID

Returns
Returns True if the wordwrapping is enabled for this header, False otherwise.

See also:
SetHeaderWordWrap, SetWordWrap, GetWordWrap

4.4.20 GetTaskIcon Methods

Retrieves the iconname of a task.

Prototype:
GetTaskIcon(LONG HeaderID, LONG TaskID),STRING

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change

Returns
Returns a STRING containing the task's iconname. If the task was not found, an empty string is returned.

See also:
SetTaskIcon

4.4.21 GetTaskInfo Methods

Retrive info about a task.

Prototype:
GetTaskInfo(LONG HeaderID, LONG TaskID),PTP:TaskInfo

Arguments:

HeaderID	The unique ID of the header in wich the task resides.
TaskID	The unique ID of the task you want info from

Returns

Upon succesfull return, this function will return a PTP:TaskInfo-group.

See also:
SetTaskInfo

Example:
```
HeaderInfo      GROUP(PTP:TaskInfo),PRE(HIG) .
TaskInfo        GROUP(PTP:TaskInfo),PRE(TIG) .

 Code
 HeaderInfo = Self.GetTaskInfo(HeaderID, 0)
 If HeaderInfo = '' Then Return .

 TaskInfo = Self.GetTaskInfo(HeaderID, TaskID)
 If TaskInfo = '' Then Return .

 Message('Header: ' & HeaderInfo.Title & '<13,10><13,10>' & |
       'Title: ' & TaskInfo.Title & '<13,10>' & |
       'Icon: ' & TaskInfo.IconName & '<13,10>' & |
       'Visible: ' & TaskInfo.IsVisible, 'You clicked a task',ICON:Asterisk,'Ok',1)
```

4.4.22 GetTaskMimicIcon Methods

Use this method if you want to know if the task mimics the icon (PROP:Icon) of the mimiced button

Prototype:
GetTaskMimicIcon(LONG HeaderID, LONG TaskID),BYTE

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change

Returns
Returns True if this task is set to mimic a buttons icon, or False if not.

See also:
SetTaskMimicIcon, SetTaskMimicText, GetTaskMimicText

4.4.23 GetTaskMimicText Methods

Use this method if you want to know if the task mimics the text (PROP:Text) of the mimiced button

Prototype:
GetTaskMimicIcon(LONG HeaderID, LONG TaskID),BYTE

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change

Returns
Returns True if this task is set to mimic the text of the mimiced button, or False if not.

See also:
SetTaskMimicText, SetTaskMimicIcon, GetTaskMimicIcon

4.4.24 GetTaskTitle Methods

Retrieves the title-text of a task.

Prototype:
GetTaskTitle(LONG HeaderID, LONG TaskID),STRING

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change

Returns
Returns a STRING containing the task's title. If the task was not found, an empty string is returned.

See also:
SetTaskTitle

4.4.25 GetTaskToolTip Methods

Use this function to get the tooltip of a task.

Prototype:
GetTaskToolTip(LONG HeaderID, LONG TaskID),STRING

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you get the tooltip from

Returns
Returns a STRING containing the task's Tooltip. If the task was not found, an empty string is returned.

See also:
SetTaskToolTip, SetTooltipMode, GetTooltipMode, SetTooltipMaxWidth, GetTooltipMaxWidth

4.4.26 GetTaskUserData Methods

Retrieves the userdata of a task. Userdata is a ULONG variable stored with each task. In this variable you can store whatever value you'd like. For instace an ID of which procedure that should be started (in case of a enduser-defiend dynamic menu), or maybe a reference address of an object or group.

Prototype:
GetTaskUserData(LONG HeaderID, LONG TaskID),ULONG

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change

Returns
Returns an ULONG containing the task's userdata. If the task was not found, this function returns 0.

See also:
SetTaskUserData

4.4.27 GetTooltipMaxWidth Methods

Get the maximum width of a tooltip.

Prototype:
GetTooltipMaxWidth(),LONG

Returns:
The maximum width a tooltip can be. Returnvalue is number of pixels.

See Also:
SetTooltipMaxWidth, SetTooltipMode, GetTooltipMode, SetTaskToolTip, GetTaskTooltip

4.4.28 GetTooltipMode Methods

Get the current tooltip-mode.

Prototype:
GetTooltipMode(),BYTE

Returns:
The current tooltip-mode.

0	Disabled
1	Normal
2	Balloon

See Also:
SetTooltipMode, SetTaskToolTip, GetTaskTooltip, SetTooltipMaxWidth, GetTooltipMaxWidth

4.4.29 GetVisible Methods

Returns header or task visibillity.

Prototype:
GetVisible(LONG HeaderID),BYTE
GetVisible(LONG HeaderID, LONG TaskID),BYTE

Arguments:

HeaderID	The unique ID of the header.
TaskID	The unique ID of the task.

Returns
Returns True if the header or task is visible, False otherwise.

See also:
SetVisible

4.4.30 GetWizardMode Methods

Returns if wizard-mode is enabled or not.

Prototype:
GetWizardMode(),BYTE

Returns:
Returns True if "Wizard mode" is enabled, False otherwise.

See also:
ClearActiveWizardTask, SetWizardMode, SetDisableWizard, GetDisableWizard, SetActiveWizardTask, SetWizardStyle

4.4.31 GetWordWrap Methods

Use this function to return if wordwrapping is enabled for this taskpanel.

Prototype:
GetWordWrap(),BYTE

Returns:
Returns True if wordwrapping is enabled, False otherwise.

See also:
SetWordWrap, SetHeaderWordWrap, GetHeaderWordWrap

4.4.32 Init Methods

Call this method to initialize the control. The control will not be draw until this function has been called. NOTE: Never call this function before the window has been opened (i.e. "Local Objects|Abc Objects|Init|Open Window"-embed is a nice place).

Prototype:
Init(LONG nControl),VIRTUAL

Arguments:
nControl The control to be used.

Example:
```
TaskPanel5.Init(?XPTaskPanel)
```

4.4.33 IsDisabled Methods

Returns whether or not the taskpanel is disabled.

Prototype:
IsDisabled(),BYTE

Return value:
Returns True if the taskpanel is disabled, False otherwise.

4.4.34 Refresh Methods

Redraws the control

Prototype:
Refresh()

4.4.35 RGB Methods

Creates a ULONG color value based on the R,G,B-values

Prototype:
RGB(Byte R, Byte G, Byte B),ULONG

Arguments:
R - The value for red (0-255)
G - The green value

B - The blue value

Return value:
This function returns a color value that can be used as an argument for the color-functions.

See also:
SetColorGrp

4.4.36 SetActiveWizardTask Methods

Calling this method will result in the task being drawn bold, and the last selected wizard-task (if any) is set to normal.

Prototypes:
SetActiveWizardTask(LONG HeaderID, LONG TaskID)

Arguments:

HeaderID	A previously obtained header-ID
TaskID	A previously obtained task-ID

See also:
ClearActiveWizardTask, SetWizardMode, GetWizardMode, SetDisableWizard, GetDisableWizard, SetWizardStyle

4.4.37 SetAlwaysExpanded Methods

Use this method to set a header to be expanded at all times. The contract/expand icon is not visible, and clicking the header will not contract it.

Prototype:
SetAlwaysExpanded(<LONG HeaderID>, <BYTE AlwaysExpanded>)

Arguments:

HeaderID	The unique header ID of the header beeing modified (if omitted, the last added header will be used)
AlwaysExpanded	If true the header will be always be expanded (if omitted, AlwaysExpanded is set to true)

4.4.38 SetAppName Methods

This method is only used with Multi-DLL apps. If you want to load icons from one of your DLL's, the DLL's name must be set with this method. Failing to do so will result in the Taskpanel trying to load icons from the .exe file.

Prototype:
SetAppName(STRING szAppName)

Arguments:
szAppName The name (casesensitive) of the DLL, including .dll

Example:
```
  TaskPanel5.SetAppName('MyMultiApp.dll')       !Icons is linked into
MyMultiApp.dll
```

4.4.39 SetColorGrp Methods

Change the colors of the taskpanel.

Prototype:
SetColorGrp(LONG GrpNum, ULONG cBackground, ULONG cGradientFrom, ULONG cGradientTo, ULONG cText, ULONG cHoveredText, <ULONG cDisabled>)

Arguments:

GrpNum	Specifies the colorgroup you want to change. COLORGRP_PANEL - Panel background colors COLORGRP_HEADERS - Colors for normal headers COLORGRP_HEADERS2 - Colors for highlighted headers COLORGRP_TASKS - Colors for tasks COLORGRP_SEPARATORS - Tasks with separator attribute on
cBackground	The background color (used on Win95/98/NT)
cGradientFrom	Top-left gradient color
cGradientTo	Bottom-right gradient color
cText	Textcolor
cHoveredText	Textcolor when the mouse is hovering over the text.
cDisabled	Textcolor for disabled tasks

See also:
RGB

Note:
The color values can be obtained with the RGB-method. Clarion's COLOR:xxx-equates are also accepted.
If you'd like to specify colors as hex-values, the format is 0<bb><gg><rr>H (eg. 000FFFFH equals yellow).

Example:
```
TaskPanel5.SetColorGrp(COLORGRP_PANEL, 15114362,15114362,14054755,0,0)
TaskPanel5.SetColorGrp(COLORGRP_HEADERS,
11685127,11685127,13065250,16777215,16748098)
TaskPanel5.SetColorGrp(COLORGRP_HEADERS2,
16777215,16777215,16373191,13065250,16748098)
TaskPanel5.SetColorGrp(COLORGRP_TASKS, 16377824,0,0,12333369,0,0)
```

4.4.40 SetDefaultTaskMargin Methods

Use this to set your own left margin for tasks that are set to "Default" justification (see Task settings)

Prototype:
SetDefaultTaskMargin(LONG nLeftMargin)

Arguments:
nLeftMargin The new task margin (in pixels). Set this to 0 to use the the default task margin.

See also:
SetMargins

Example:
```
TaskPanel3.SetDefaultTaskMargin(5)
```

4.4.41 SetDisabled Methods

Use this method to disable or enable a task.

Prototypes:
SetDisabled(<LONG HeaderID>, <LONG TaskID>, BYTE bIsDisabled)

Arguments:

HeaderID	A previously obtained ID for parent header the task should be added to.
TaskID	The unique ID of the task you want to set justification for.
bIsDisabled	If this value is True, the task will be disabled. If False, the task will be enabled.

See also:
GetDisabled

Example:
```
TaskPanel5.SetDisabled(Taskpanel5.Header1, Taskpanel5.Header1_Task1, TRUE)
```

4.4.42 SetDisableWizard Methods

Turn off wizard-mode for a header.

Prototypes:
SetDisableWizard(LONG HeaderID, BYTE DisableWizard)

Arguments:

HeaderID	A header ID.
DisableWizard	If this value is True, tasks in this header will not part of the wizard-mode.

See also:
ClearActiveWizardTask, SetWizardMode, GetWizardMode, GetDisableWizard, SetActiveWizardTask, SetWizardStyle

4.4.43 SetExpanded Methods

Use this method to expand or contract a header.

Prototype:
SetExpanded(LONG HeaderID, <BYTE IsExpanded>)

Arguments:

HeaderID	The unique header ID of the header beeing modified
IsExpanded	(Optional) If true the header will be expanded, false will contract the header. If this argument is omitted the header will be expanded.

Example:
```
TaskPanel5.SetExpanded(Self.MyFirstHeader, False)  !Contracts MyFirstHeader
```

See also:
GetExpanded

4.4.44 SetFallbackColors Methods

When fallback colors are enabled, different colors will be used when Windows only allows 256 colors. This could be when running Windows NT or through a Terminal Server login. Use this method to change the default colors (which is a grey color scheme).

Prototype:
SetFallbackColors Procedure(ULONG cBkg, ULONG cNormHdr, ULONG cNormHdrTxt, ULONG cHighHdr, ULONG cHighHdrTxt, ULONG cTasksBkg, ULONG cTasks)

Arguments:

cBkg	Panel background color
cNormHdr	The color for headers with style set to "Normal"
cNormHdrTxt	Text color for "Normal" headers
cHighHdr	The color for headers with style set to "Highlighted"
cHighHdrTxt	Text color for "Highlighted" headers
cTaskBkg	Background color for tasks
cTasks	Textcolor for tasks

See also:
SetFallbackMode, RGB, SetColorGrp

Note:
The color values can be obtained with the RGB-method. Clarion's COLOR:xxx-equates are also accepted.
If you'd like to specify colors as hex-values, the format is 0<bb><gg><rr>H (eg. 000FFFFH equals yellow).

4.4.45 SetFallbackMode Methods

When fallback colors are enabled, different colors will be used when Windows only allows 256 colors. This could be when running Windows NT or through a Terminal Server login. Use this method to enable or disable the use of fallback colors.

Prototype:
SetFallbackMode(BYTE FallbackOn)

Arguments:

FallbackOn	Set to False turn off the use of fallback colors (Default on).

See Also:
SetFallbackColors

4.4.46 SetFont Methods

Change the current font for the taskpanel. The default font is "MS Sans Serif"

Prototype:
SetFont(STRING fName)

Arguments:

fName	The name of the new font.

4.4.47 SetHeaderBoxed Methods

Set the header to be "boxed"

When a header is boxed, it will be drawn as a single line. This results in a box with task in it.

Prototype:
SetHeaderBoxed(LONG HeaderID, BYTE IsBoxed),BYTE

Arguments:

HeaderID	Unique header ID
IsBoxed	Set to True to draw the header boxed. False to draw the header as normal.

See also:
GetHeaderBoxed

4.4.48 SetHeaderFontWeight Methods

Sets or removes the BOLD attribute for a specified or all headings.

Prototype:
SetHeaderFontWeight(<LONG HeaderID>, BYTE bIsBold)

Arguments:

HeaderID	Optional unique ID of the header for which to set the font weight. If omitted, the setting is applied to all headers.
bIsBold	True to turn headers bold, false to turn them to normal

See also:

4.4.49 SetHeaderIcon Methods

Change the iconname of a header.

Prototype:
SetHeaderIcon(LONG HeaderID, STRING IconName)

Arguments:

HeaderID	Unique ID of the header
IconName	The new iconname

Remarks
Usually the class tries to load the icon as a resource from the exe or dll's in the application. However, if the icon is not found there, the taskpanel tries to load it from disk.

4.4.50 SetHeaderTitle Methods

Sets the header title of the specified header.

Prototype:
SetHeaderTitle (LONG HeaderID, STRING Title)

Arguments:

HeaderID	The unique ID of the header.
Title	The title string to set.

See also:
GetHeaderTitle

4.4.51 SetHeaderWordWrap Methods

Use this enable or disable wordwrapping for tasks in a header.

Prototype:
SetHeaderWordWrap(LONG HeaderID, BYTE WordWrap),BYTE

Arguments:

HeaderID	Unique header ID
WordWrap	Set to True to enable wordwrapping, False to disable.

See also:
GetHeaderWordWrap, SetWordWrap, GetWordWrap

4.4.52 SetJustification Methods

Set the justification for a tasks text.

Prototypes:
SetJustification(<LONG HeaderID>, <LONG TaskID>, BYTE bAlignment)

Arguments:

HeaderID	A previously obtained ID for parent header the task should be added to. If you ommit this value the justification will be set for the last added task .
TaskID	The unique ID of the task you want to set justification for. If you ommit this value the justification will be set for the last added task .
bAlignment	Specify one of the following equates: *TALIGN_DEFAULT* - Make space for icon (even if there's no icon) *TALIGN_LEFT* - Left aligned text *TALIGN_CENTER* - Center the text *TALIGN_RIGHT* - Right align text

Example:

```
  TaskPanel5.SetJustification(Taskpanel5.Header1, Taskpanel5.Header1_Task1,
TALIGN_RIGHT)
```

4.4.53 SetMargins Methods

This method is used to set the top,left,right and bottom-margins. All margins are in pixels.

Prototype:
SetMargins(LONG nLeftMargin, LONG nTopMargin, <LONG nRightMargin>, <LONG nBottomMargin>)

Arguments:

nLeftMargin	Left marign
nTopMargin	Top margin
nRightMargin	Right margin
nBottomMargin	Spacing between headers

Example:
```
TaskPanel5.SetMargins(0,0,0,0) !No margins
```

4.4.54 SetTaskFontWeight Methods

With this method you can set a task to be draw with bold text.

Prototypes:
SetTaskFontWeight(<LONG HeaderID>, <LONG TaskID>, BYTE bIsBold)

Arguments:

HeaderID	A previously obtained ID for parent header the task should be added to. If you ommit this value the justification will be set for the last added task .
TaskID	The unique ID of the task you want to set justification for. If you ommit this value the justification will be set for the last added task .
bIsBold	If this value is True, the task will be drawn with bold text. If False, the task will be drawn as normal.

Example:
```
TaskPanel5.SetTaskFontWeight(Taskpanel5.Header1, Taskpanel5.Header1_Task1, TRUE)
```

4.4.55 SetTaskIcon Methods

Change the iconname of a task.

Prototype:
SetTaskIcon(LONG HeaderID, LONG TaskID, STRING IconName)

Arguments:

HeaderID	Unique ID of the task's parent header
TaskID	Unique ID of the task you want change
IconName	The new iconname

Remarks
Usually the class tries to load the icon as a resource from the exe or dll's in the application. However, if the icon is not found there, the taskpanel tries to load it from disk.

See also:
GetTaskIcon

4.4.56 SetTaskInfo Methods

Change the properties of a task at runtime

Prototype:
SetTaskInfo(LONG HeaderID, LONG TaskID, PTP.askInfo TaskInfo)

Arguments:

HeaderID	The unique ID of the header in wich the task resides
TaskID	The unique ID of the task you want info from
TaskInfo	A PTP:TaskInfo group with the task-properties

See also:
GetTaskInfo

4.4.57 SetTaskMimicIcon Methods

This function can be used if you want the task to use it's icon instead of the icon of the mimiced button

Prototype:
SetTaskMimicIcon(LONG HeaderID, LONG TaskID, BYTE MimicIcon)

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change
MimicIcon	Set to True if the task should mimic a buttons icon. Set to False if you want to mimic the button, but not the icon.

See also:
GetTaskMimicIcon, SetTaskMimicText, GetTaskMimicText

4.4.58 SetTaskMimicText Methods

This function can be used if you want the task to use it's title instead of the text of the mimiced button

Prototype:
SetTaskMimicText(LONG HeaderID, LONG TaskID, BYTE MimicText)

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change
MimicText	Set to True if the task should mimic a buttons text. Set to False if you want to mimic the button, but not the text.

See also:
GetTaskMimicText, SetTaskMimicIcon, GetTaskMimicIcon

4.4.59 SetTaskSpacing Methods

Sets the vertical spacing between tasks in the tasklist.

Prototype:
SetTaskSpacing(LONG nTaskSpacing)

Arguments:

nTaskSpacing	Vertical spacing between tasks in pixels

See also:
SetTaskFontWeight

4.4.60 SetTaskTitle Methods

Change the title-text of a task.

Prototype:
SetTaskTitle(LONG HeaderID, LONG TaskID, STRING Title)

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change
Title	New task title

See also:
GetTaskTitle

4.4.61 SetTaskToolTip Methods

Change the tooltip of a task.

Prototype:
SetTaskToolTip(LONG HeaderID, LONG TaskID, STRING ToolTip)

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change
ToolTip	New task tooltip

See also:
GetTaskToolTip, SetTooltipMode, GetTooltipMode, SetTooltipMaxWidth, GetTooltipMaxWidth

4.4.62 SetTaskUserData Methods

Sets the userdata of a task. Userdata is a ULONG variable stored with each task. In this variable you can store whatever value you'd like. For instace an ID of which procedure that should be started (in case of a enduser-defiend dynamic menu), or maybe a reference address of an object or group.

Prototype:
SetTaskUserData(LONG HeaderID, LONG TaskID, ULONG UserData)

Arguments:

HeaderID	The unique ID of the task's parent header
TaskID	The unique ID of the task you want change
UserData	Any ulong value

See also:
GetTaskUserData

4.4.63 SetTooltipMaxWidth Methods

Set the maximum width of a tooltip. This is the point where tooltip text will wordwrap.

Prototype:
SetTooltipMaxWidth(LONG TooltipMaxWidth)

Arguments:

TooltipMaxWidth	Maximum width of a tooltip (in pixels). Tooltips will be wordwrapped at this point.

See Also:
GetTooltipMaxWidth, SetTooltipMode, GetTooltipMode, SetTaskToolTip, GetTaskTooltip,

4.4.64 SetTooltipMode Methods

Set the tooltip-mode (Disabled, Normal, Balloon)

Prototype:
SetTooltipMode(BYTE TooltipMode)

Arguments:

TooltipMode	0 - Tooltips is disabled 1 - Normal tooltips 2 - Balloon tooltips (Win98/2000 and later)

See Also:
GetTooltipMode, SetTaskToolTip, GetTaskTooltip, SetTooltipMaxWidth, GetTooltipMaxWidth

4.4.65 SetVisible Methods

Changes the visibillity of a header or task.

Prototype:
SetVisible(BYTE IsVisible)
SetVisible(LONG HeaderID, BYTE IsVisible)
SetVisible(LONG HeaderID, LONG TaskID, BYTE IsVisible)

Arguments:

HeaderID	The unique ID of the header you want to change
TaskID	The ID of the task you want to change
IsVisible	If you set this value to true the item will become visible. If false, the item is hidden.

Note:
If you use the first prototype, the last added header or task will be affected.

If you uses the second prototype to hide or unhide a header, the tooltips for the tasks in the header will be refreshed.

See also:
GetVisible

4.4.66 SetWizardMode Methods

Switches Wizard-mode on or off.

If Wizard-mode is enabled, the last clicked task is bolded, to indicate the current step in the wizard. You can exlude a header from the wizard mode using the SetDisableWizard

Prototype:
SetWizardMode(BYTE WizardMode)

Arguments:

WizardMode	If this is set to True, wizard-mode is enabled, otherwhise wizard-mode is turned off.

See also:
ClearActiveWizardTask, GetWizardMode, SetDisableWizard, GetDisableWizard, SetActiveWizardTask, SetWizardStyle

4.4.67 SetWizardStyle Methods

Use this method to change how active and inactive tasks look like when using wizard mode.

You can set an icon to be used for the active and inactive tasks. You can also turn on or off bold text.

Prototype:
SetWizardStyle(STRING WizardActiveIcon, STRING WizardInactiveIcon, BYTE WizardBold)

Arguments:

WizardActiveIcon	Name of the icon to use for the active wizard task. Empty string is

	allowed to specify no icon.
WizardInactiveIcon	Name of the icon to use for the inactive wizard task. Empty string is allowed to specify no icon.
WizardBold	Set to false to turn of bold text on the active wizard task.

See also:
ClearActiveWizardTask, GetWizardMode, SetDisableWizard, GetDisableWizard, SetActiveWizardTask

4.4.68 SetWordWrap Methods

Use this function to return if wordwrapping is enabled for this taskpanel.

Prototype:
SetWordWrap(BYTE WordWrap),BYTE

Arguments:

WordWrap	Set to True to enable wordwrapping, False to disable.

See also:
GetWordWrap, SetHeaderWordWrap, GetHeaderWordWrap

4.5 Structures

- POTaskInfo

4.5.1 PTP:TaskInfo Structures

```
PTP:TaskInfo  GROUP,TYPE
Id                LONG
Title             CSTRING(255)
IsVisible         BYTE
IconName          CSTRING(255)
LeftActionFeq     LONG
IsExpanded        BYTE
Justification     BYTE
ToolTip           CSTRING(512)
              END
```

Index

- 0 -

- A -

- B -

- C -

- D -

- E -

- F -

- G -

- H -

- I -

- L -

- M -

- N -

- P -

- Q -

- R -

- S -

- T -

- U -

- V -

- W -