
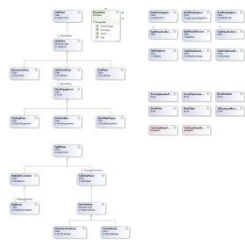


RadDock vNEXT

Contents

| | |
|--|---|
| RadDockingManager vs RadDock Comparison | 2 |
| New features and enhancements..... | 5 |
| Fig.1 (RadDockingManager Architecture)..... | 6 |
| Fig.2 (RadDock Architecture) | 7 |
| Fig. 3 RadDock..... | 8 |
| Fig.4 RadDock Visual and Logical Tree - Final Design | 8 |

RadDockingManager vs RadDock Comparison

| Current feature set | | |
|---|---|---|
| Architecture/ Functionality | RadDockingManager (current version, pre-Q1 2009) | RadDock (new version, after Q1 2009) |
| Architecture | <p>Complex architecture, based on too many interfaces, which were supposed to give the developer the ability to freely change the behavior of the docking control. It turned out that the architecture and API is too convoluted, thus not used by the users, because it unnecessarily complicates the basic functionality and requires constant casting of objects:</p>  <p>Fig.1 (also available at the end of the document)</p> | <p>Simplified and completely transparent architecture based on a single interface – IDockPane – which will offer direct access to all logical objects.</p>  <p>Fig.2 (also available at the end of the document)</p> |
| Adding docking to external controls and forms | <p><i>Case 1: by using IDockable and related interfaces.</i></p> <p>The IDockable and all related interfaces need to be implemented directly, which makes the implementation extremely difficult, and almost impossible to provide the same functionality as if it was a built-in one. This case needed extensive support by the Telerik staff. (fig. 1)</p> | <p>Case1:</p> <p>The new version will not implement IDockable or similar interfaces, as in the old version. Rather, it will offer a much simplified IDockPane interface, which will be backed up by an internal HostDockPane which will guarantee the correct behavior of the hosted and hosting control(s). (fig. 2)</p> <p>IDockable and all related interfaces will become obsolete.</p> |
| | <p><i>Case 2: by using UserDockForm and UserDockControl</i></p> <p>A simpler solution than Case1, which is also easier to implement, and as such the preferred [by our customers] way to do things. The problem with this case is that it causes (or leads to) variations in the way objects are treated, depending on whether they are DockPane, DocumentPane, UserDockControl, or UserDockForm, which confused our clients.</p> | <p>Case2: RadDock will offer automated way of hosting and treating Control and/or Form objects, including a simplified and unified API for handling each case.</p> <p>The old and custom UserDockControl, UserDockForm will become obsolete.</p> |
| Drag&Drop functionality | Window drag-and-drop is based on a number of interfaces, which work with snapshots of | The drag-and-drop functionality is embedded directly in RadDock, rather than depending on |

| | | |
|---------------------------------|---|--|
| | <p>the content in the docking panels to allow for showing the content while dragging. One of the issues here is that, in the case of Floating panels the snapshots were not shown when the content was substantial and complex (the dragged window did not show its contents, rather it showed just a placeholder).</p> <p>Another issue is that the drag-and-drop functionality uses window messages and a number of smaller forms (one for each of the Docking guides) in order to work, leading to flickering and/or deactivation of the main dock form while ToolWindows are dragged.</p> | <p>window messages, which means that it will work properly with Floating panels by showing their real content (just like in Visual Studio). It will also use a single form for all the DockingGuides, solving the flickering and deactivation problems of the main form.</p> |
| Active document/ tool window | <p>Controlling the active document/tool window is complicated and does not always return the correct result.</p> | <p>The new version provides real control over the active document (tool window, component, and/or panel) and will return correct information on which one is active. Implements IContainerControl of the Base Class Library (BCL).</p> |
| Layouts | <p>The current (soon to be old) version is one of the few remaining Telerik WinForms components to still use the older TPF (Telerik Presentation Framework) layouts.</p> | <p>The upcoming version will use the latest TPF layouts, which will bring better drawing performance, better theme management, and better positioning of items.</p> |
| Themes | <p>The DockPresenterControl, DocumentPresenterControl, and DockingGuides have to be themed separately in the VSB (Visual Style Builder), making the theming process cumbersome and time consuming. The AutoHide tabs were not themeable.</p> | <p>All docking elements have been merged in a single design form. Theming in the VSB will be done for this form only, thus considerably simplifying the process. If the application already uses themed TPF elements, which are also used in RadDock, the theme will automatically be reused.</p> |
| Design-time | <p>The RadDockingManager uses XML-serialized layout for the docking components in design-time.</p> | <p>The new version will use the conventional serialized WinForms code for adding docking components, which gives better design-time performance and faster loading.</p> |
| Run-time | <p>The Load/Save layout is based on a GUID for determining the content, which makes the XML code complicated and hard to read.</p> <p>Example:</p> <pre><?xml version="1.0" encoding="utf-8" ?> <DockingTree xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <Sites Type="Telerik.WinControls.Docking.AutoHideContent" Assembly="Telerik.WinControls.Docking, Version=8.2.0.0, Culture=neutral, PublicKeyToken=5bb2a467cbec794e"> <Sites Orientation="Vertical" Size="855;595" HidePosition="0"> <Dockables /> <Left Orientation="Horizontal" Size="855;595" HidePosition="0"> <Dockables /> <Left Orientation="Vertical" Size="276;595" HidePosition="0"> <Dockables /> <Left Orientation="Horizontal" Size="276;285" HidePosition="Left"> <Dockables></pre> | <p>The Load/Save layout will be based on Name for determining the content. Similar to XAML, the XML code will correspond to the logical objects in RadDock. Example:</p> <pre><?xml version="1.0" encoding="utf-16" ?> <RadSplitContainer Orientation="Horizontal" Size="855;595"> <RadSplitContainer Orientation="Vertical" Size="276;595"> <ToolTabStrip Size="276;286" CaptionVisible="True" TabStripVisible="False"> <DockPane Name="ToolWindow1" Text="Tool Window 3" AutoHideSize="316;100" FloatSize="150;300" /> <DockPane Name="ToolWindow2" Text="Tool Window 4" /> </ToolTabStrip> <ToolTabStrip> <ToolTabStrip Size="276;305"> <DockPane Name="ToolWindow3" Text="Tool Window 2" DockSize="316;268" FloatSize="150;300" /> </ToolTabStrip> </RadSplitContainer> </DocumentTabStrip Size="575;595"> <DocumentPane Name="Document1" Text="DocumentWindow</pre> |

| | |
|---|--|
| <pre> <XmlDockable Text="Tool Window 3" DockState="Docked" DockPosition="Left" PreferredDockSize="316;100" PreferredFloatSize="150;300" Guid="94763a12-139e- 4968-af69-72e36282d54e" Type="Telerik.WinControls.Docking.DockPanel" Assembly="Telerik.WinControls.Docking, Version=8.2.0.0, Culture=neutral, PublicKeyToken=5bb2a467cbec794e" CaptionVisible="True" TabStripVisible="True" CloseButtonVisible="True" HideButtonVisible="True" DropDownButtonVisible="True" /> <XmlDockable Text="Tool Window 4" DockState="Docked" DockPosition="Fill" PreferredDockSize="100;100" PreferredFloatSize="150;300" Guid="b4945fa9-b317-4a6c- bc7c-adae84d8b49c" Type="Telerik.WinControls.Docking.DockPanel" Assembly="Telerik.WinControls.Docking, Version=8.2.0.0, Culture=neutral, PublicKeyToken=5bb2a467cbec794e" CaptionVisible="True" TabStripVisible="True" CloseButtonVisible="True" HideButtonVisible="True" DropDownButtonVisible="True" /> </Dockables> </Left> <Righth Orientation="Horizontal" Size="276;306" HidePosition="Left"> <Dockables> <XmlDockable Text="Tool Window 2" DockState="Docked" DockPosition="Bottom" PreferredDockSize="316;268" PreferredFloatSize="150;300" Guid="6d215d7c-2bd3-4ffe- 8977-9153e4df4567" Type="Telerik.WinControls.Docking.DockPanel" Assembly="Telerik.WinControls.Docking, Version=8.2.0.0, Culture=neutral, PublicKeyToken=5bb2a467cbec794e" CaptionVisible="True" TabStripVisible="True" CloseButtonVisible="True" HideButtonVisible="True" DropDownButtonVisible="True" /> </Dockables> </Righth> </Left> <Righth Orientation="Horizontal" Size="575;595" HidePosition="0"> <Dockables> <XmlDockable Text="Tool Window 1" DockState="TabbedDocument" DockPosition="Fill" PreferredDockSize="316;268" PreferredFloatSize="150;300" Guid="14b7df8b-c9e0-4cf6-813e-226a95824886" Type="Telerik.WinControls.Docking.DockPanel" Assembly="Telerik.WinControls.Docking, Version=8.2.0.0, Culture=neutral, PublicKeyToken=5bb2a467cbec794e" CaptionVisible="True" TabStripVisible="True" CloseButtonVisible="True" HideButtonVisible="True" DropDownButtonVisible="True" /> </Dockables> </Righth> </Left> </Sites> </Sites> <Sites Type="Telerik.WinControls.Docking.DockSite" Assembly="Telerik.WinControls.Docking, Version=8.2.0.0, Culture=neutral, PublicKeyToken=5bb2a467cbec794e" FloatingLocation="100;100" Visible="True"> <Sites Orientation="Horizontal" Size="183;411" HidePosition="0"> <Dockables /> <Left Orientation="Horizontal" Size="183;411" HidePosition="Left"> <Dockables> <XmlDockable Text="Tool Window 5" DockState="Floating" DockPosition="Left" PreferredDockSize="100;100" PreferredFloatSize="183;411" Guid="190f426d-281a-4a75- a92a-347a2697596b" Type="Telerik.WinControls.Docking.DockPanel" Assembly="Telerik.WinControls.Docking, Version=8.2.0.0, Culture=neutral, PublicKeyToken=5bb2a467cbec794e" CaptionVisible="True" TabStripVisible="True" CloseButtonVisible="True" HideButtonVisible="True" DropDownButtonVisible="True" /> </Dockables> </Left> </Sites> </Sites> </DockingTree> </pre> | <pre> 1" /> </DocumentTabStrip> <ToolTabStrip Size="276;595"> <ToolPane Name="ToolWindow4" Text="Tool Window 5" IsFloating="True" /> </ToolTabStrip> RadSplitContainer> </pre> |
|---|--|

New features and enhancements

- A much requested feature is to allow for objects different from IDockPane (such as TabPanels и TabStripPanel) to be added to the RadDock layout, i.e. to be hosted in RadDock, but with disabled docking operations, such as docking other objects in them;
- New CloseAction and AutoDispose properties will allow for controlling the Close behavior, as well as the memory management of the SplitPanel and DockPane objects (currently the Dispose method has to be called explicitly);
- The new version will collect and use the information about the state of each DockPane— FloatingSize, FloatingLocation, AutoHideSize, Previous position, AutoHidePostion, etc. dynamically. The old version offered partial support for this;
- New control is taking shape from the Dock re-design – RadSplitContainer

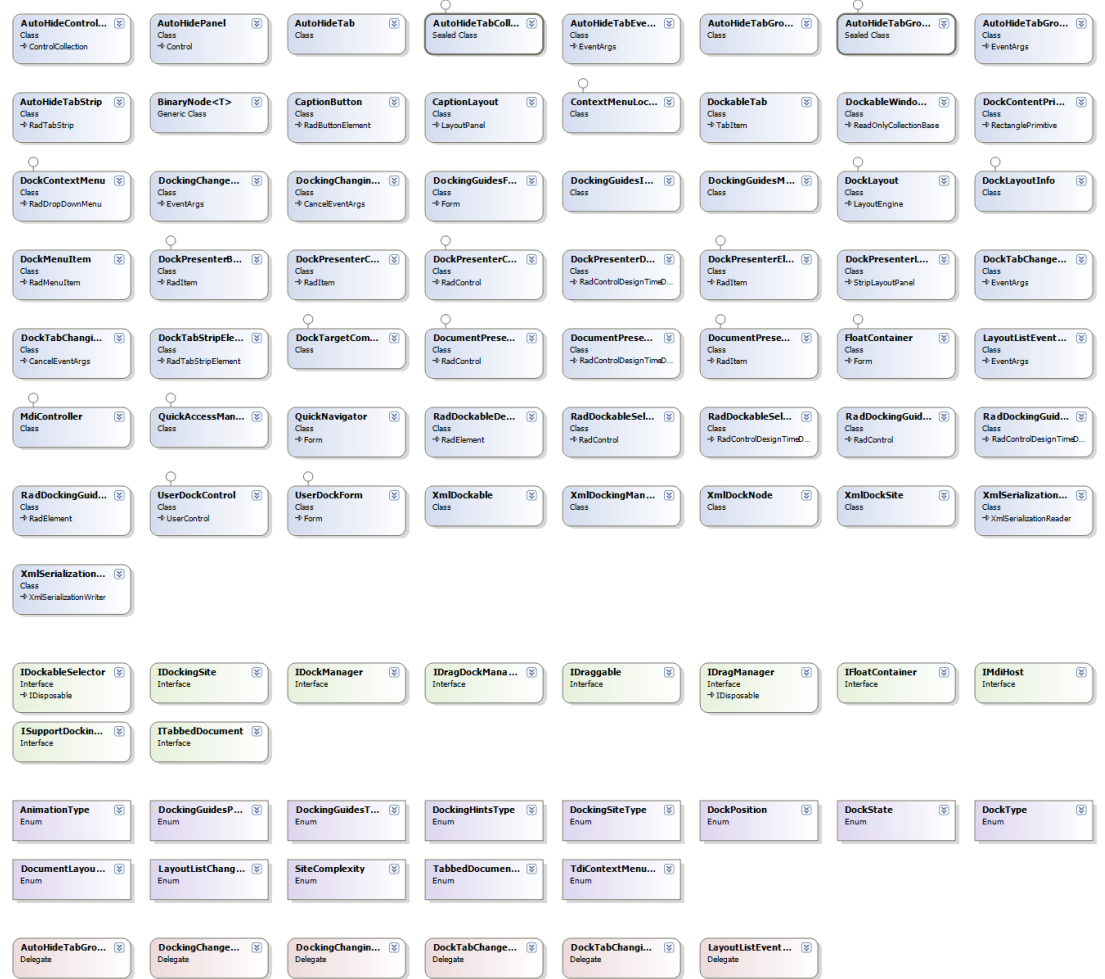


Fig.2 (RadDock Architecture)

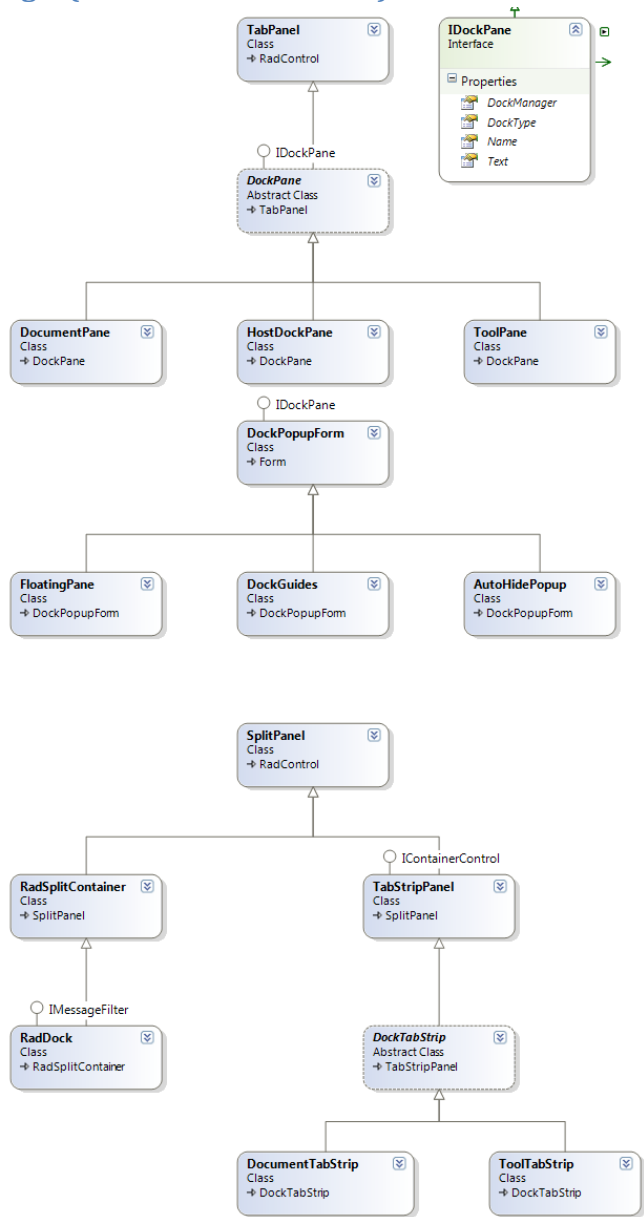


Fig. 3 RadDock

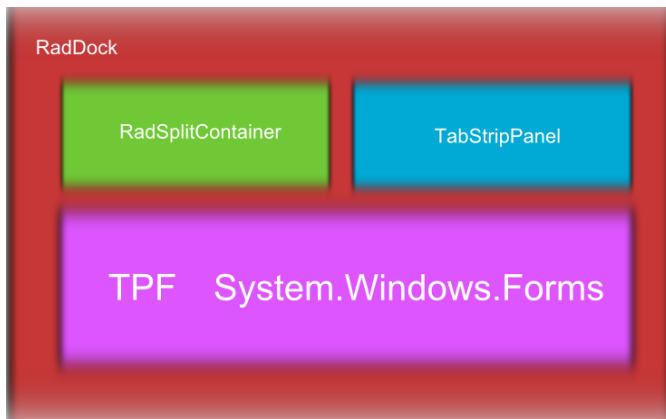


Fig.4 RadDock Visual and Logical Tree - Final Design

