

# DockingFrames 1.0.5 - Transition

Benjamin Sigg

April 20, 2008

## Contents

<b>1</b>	<b>Version 1.0.3</b>	<b>3</b>
1.1	Incompatibilities	3
1.1.1	DefaultKeyboardController	3
1.1.2	DefaultDockable/DefaultCDockable	3
1.1.3	CDockableListener	3
1.1.4	FlapDockStation	4
1.1.5	XML	4
1.1.6	DockTheme	4
1.1.7	DockFactory	4
1.2	Features	4
1.2.1	SplitDockStation	4
1.2.2	SplitLayoutManager	5
1.2.3	CDockable resize lock	5
1.2.4	FlapLayoutManager	5
1.2.5	ColorManager/ColorScheme	5
1.2.6	ColorMap	5
1.2.7	LookAndFeel	5
1.2.8	CDockable resize request	5
<b>2</b>	<b>Version 1.0.4</b>	<b>6</b>
2.1	Incompatibilities	6
2.1.1	Binary file format	6
2.1.2	DockableListener	6
2.1.3	Title visibility on CDockables	6
2.1.4	BasicDropDownButtonHandler	6
2.1.5	CDockable.getClose	6
2.1.6	CLocation	7
2.1.7	working area	7
2.2	Features	7
2.2.1	Border around BubbleDisplayer	7
2.2.2	Backup factories (core)	7
2.2.3	Backup factories (common)	7
2.2.4	Unregister factories from DockFrontend	7
2.2.5	Action support keyboard	8
2.2.6	FocusTraversalPolicies	8
2.2.7	override predefined actions	8

2.2.8	CBlank	8
2.2.9	CStation	8
2.3	Bugfixes	8
2.3.1	BubbleDisplayer.getDockableInsets	8
2.3.2	IndexOutOfBoundsException from ButtonPanel	8
2.3.3	Mode change of CDockable	9
2.3.4	Opening maximized CDockable	9
2.3.5	Unbind of DockAction called to often	9
<b>3</b>	<b>Version 1.0.5</b>	<b>9</b>
3.1	Incompatibilities	9
3.1.1	DockStationListener	9
3.1.2	DockableFocusListener	10
3.1.3	DockTheme.getDockableSelection	10
3.1.4	tap-strip no longer painted by TapPainter	10
3.1.5	KeyboardController does fire less events	10
3.1.6	ComponentHierarchyObserver	10
3.2	API and Layout	11
3.2.1	KeyStroke for closing Dockable	11
3.2.2	New listeners	11
3.2.3	ComponentHierarchObserver	11
3.2.4	Root window for DockController	11
3.2.5	FocusTraversalPolicies	11
3.2.6	Dialog to select focused Dockable	11
3.2.7	Extracting colors from LookAndFeel	12
3.2.8	EclipseTheme	12
3.2.9	SplitDockStation	12
3.3	Bugfixes	12
3.3.1	Missing colors for BasicTheme	12
3.3.2	Cutting bounds of children of SplitDockStation	12
3.3.3	NullPointerException when changing focus	12
3.3.4	Undecorated dialogs not undecorated	12
3.3.5	RexTabbedComponent not adding/removing children	13
3.3.6	Focusing a hidden CDockable	13
3.3.7	Missing events when changing state of CDockable	13

## Abstract

This document describes the most important changes between versions, and how developers should change their application in order to use new features. This document does not make any distinction between the core-library and the common-project. Not all changes are listed up in this document, only those enhancements which might be interesting for the majority of developers.

# 1 Version 1.0.3

Version 1.0.3 emphasizes on background enhancements. The API remains unchanged for most parts.

## 1.1 Incompatibilities

These changes break with the API from 1.0.2, clients must change their interfaces in order to work properly.

### 1.1.1 DefaultKeyboardController

**Short** The class `DefaultkeyBoardController` has been renamed to `DefaultKeyboardController`

**Reason** The new name looks better

**Clients** Replace any occurrence of `DefaultkeyBoardController` to `DefaultKeyboardController`

### 1.1.2 DefaultDockable/DefaultCDockable

**Short** `DefaultDockable` and `DefaultCDockable` now have `BorderLayout` set as default `LayoutManager`

**Reason** `BorderLayout` is the most often used `LayoutManager`.

**Clients** If another `LayoutManager` than `BorderLayout` is needed, set it up.

### 1.1.3 CDockableListener

**Short** `CDockableListener` divided into `CDockableStateListener` and `CDockablePropertyListener`

**Reason** `CDockableListener` was too big. Most clients either need information about the state, or about the properties of a `CDockable`. The case that both informations are needed is seldom.

**Clients** Need to decide which listener they implement. Note that `CDockableAdapter` implements both listeners, but not all methods get invoked when the adapter is registered only as one kind of listener.

#### 1.1.4 FlapDockStation

**Short** FlapDockStations layout is stored in a new format. The xml format will do the transition automatically, but the `DataInput/OutputStream` will not work properly.

**Reason** the old format did not carry enough information

**Clients** Store the layout in xml-format and load it again to do the transition.

#### 1.1.5 XML

**Short** `XElement` now extends `XContainer`, and no longer `XAttribute`. `XAttribute` extends `XContainer` as well.

**Reason** An element of a xml file is not an attribute, that is now reflected in the class structure

**Clients** May need to replace some occurrences of `XAttribute` by `XContainer`

#### 1.1.6 DockTheme

**Short** The common-project uses its own set of `DockThemes`. Each theme `XTheme` gets replaced by `CXTheme`

**Reason** The new themes make use of the new `ColorMap`

**Clients** Should use the new themes when possible. The old themes will work, but the user will see less features.

#### 1.1.7 DockFactory

**Short** `DockFactories` can now create any `Object` they want, and are no longer required to create `DockLayouts`. `DockLayout` has been converted into a class that wraps the `Object` that was created by a `DockFactory`

**Reason** All `DockLayouts` need to do the same things, hence clients would need to write the same code over and over again. Clients have now more freedom in how to implement `DockFactory`

**Clients** Should remove all occurrences of `implements DockLayout` and the methods `set/getFactoryId` that were defined in `DockLayout`

### 1.2 Features

This is the set of new features.

#### 1.2.1 SplitDockStation

**Short** The tree of elements of a `SplitDockStation` is now accessible from outside and can be modified directly

**Reason** It is more intuitive to work directly with the tree, some new algorithms work on the tree and are easier to implement that way.

### 1.2.2 SplitLayoutManager

**Short** New `SplitLayoutManager` calculates where to drop, and how to divide, elements of a `SplitDockStation`

**Reason** New features, like the locked size of `CDockable`, were only possible if the behavior of a `SplitDockStation` can be changed on runtime.

### 1.2.3 CDockable resize lock

**Short** The size of a `CDockable` can be locked during resize of its parent. See `setResizeLocked`, a method of `AbstractCDockable`.

**Reason** This was a request from a user

### 1.2.4 FlapLayoutManager

**Short** `FlapDockStation` now uses `FlapLayoutManager` to arrange its children

**Reason** Exchangeable behavior was a requirement for new features in the common-project.

### 1.2.5 ColorManager/ColorScheme

**Short** Many graphical elements now use `ColorManager` and `ColorSchemes`

**Reason** Colors can now be exchanged by clients. The control goes deep, even the color of a single element can be exchanged without affecting other elements of the same kind.

### 1.2.6 ColorMap

**Short** `CDockable` uses a `ColorMap` to define special colors for tabs and titles that are related to the `CDockable`

**Reason** This was a request from a user

### 1.2.7 LookAndFeel

**Short** Changes of `LookAndFeel` noted by `DockController` and forwarded to all `UIListeners`.

**Reason** Because the `ColorManager` would not be informed of the new `LookAndFeel` otherwise

### 1.2.8 CDockable resize request

**Short** `CDockables` can now request a size they would like to have, and in most environments they will get this size. See the method `setResizeRequest` of `AbstractCDockable`.

**Reason** This was a request from a user

## 2 Version 1.0.4

Version 1.0.4 introduces a few new features that add customizability

### 2.1 Incompatibilities

These changes break with the API from 1.0.3, clients must change their interfaces in order to work properly.

#### 2.1.1 Binary file format

**Short** The binary file format has been changed

**Reason** The format now includes version numbers so that backwards compatibility should be possible in the next versions

**Clients** Need to delete all binary files. They might try to write their properties with the old version in xml, and then load the xml file with the new version. This should convert the files.

#### 2.1.2 DockableListener

**Short** Has an additional method `titleExchanged`

**Reason** Allows to exchange a `DockTitle` while the `Dockable` is visible

**Clients** Need to update any class that implements `DockableListener`.

#### 2.1.3 Title visibility on CDockables

**Short** Any `CDockable` can now hide its titles at any time

**Reason** user request

**Clients** Need to update any class implementing `CDockablePropertyListener` since that listener has an additional method `titleShownChanged`.

#### 2.1.4 BasicDropDownButtonHandler

**Short** Requests now a `BasicDropDownButtonTrigger` instead of a `BasicTrigger`

**Reason** to allow steering any drop down action with the keyboard.

**Clients** unlikely to have an effect on any client

#### 2.1.5 CDockable.getClose

**Short** Method has been moved into `CommonDockable`

**Reason** The action can now be replaced through `CDockable.getAction`. There is no need for any client to replace the action by replacing the whole `DockActionSource`

**Clients** should use `putAction`, a method of `AbstractCDockable` to exchange the close-action. No fix for clients which added additional elements to the close-source.

### 2.1.6 CLocation

**Short** Additional CLocations, some methods have been moved

**Reason** To allow the new CStation more flexible CLocations were needed.

**Clients** No general solution available, clients should recompile their project and check all compiler errors.

### 2.1.7 working area

**Short** Every CStation can now be a working area

**Reason** To allow more flexibility in grouping CDockables

**Clients** That should not be visible for any client using version 1.0.3

## 2.2 Features

This is the set of new features.

### 2.2.1 Border around BubbleDisplayer

**Short** BubbleDisplayer now shows a border if the title is not null, or if the dockable is not a station

**Reason** Looks better

### 2.2.2 Backup factories (core)

**Short** DockFrontend and PredefinedDockSituation can now use backup factories. These factories are used to load elements which should be in the cache, but are missing. In case of DockFrontend they are automatically added to the frontend.

**Reason** Removes the need to add all Dockables to a DockFrontend before loading a layout from a file.

### 2.2.3 Backup factories (common)

**Short** CControl now supports lazy initialisation of SingleCDockables through the SingleCDockableBackupFactory.

**Reason** saves memory

### 2.2.4 Unregister factories from DockFrontend

**Short** DockFactorys can now be unregistered from DockFrontend

**Reason** Was missing

### 2.2.5 Action support keyboard

**Short** DockActions are triggered by pressing SPACE on the focused button, DropDownActions pop up when the DOWN (non numpad) key is pressed

**Reason** Ongoing work to allow navigating in DF without the mouse.

### 2.2.6 FocusTraversalPolicies

**Short** New FocusTraversalPolicies allow to navigate within all elements of a DockableDisplayer (including title).

**Reason** Ongoing work to allow navigating in DF without the mouse.

### 2.2.7 override predefined actions

**Short** CDockable has an additional method `getAction` which is used by various modules to override their default actions.

**Reason** Answer to a user request

### 2.2.8 CBlank

**Short** New action CBlank, which does not show anything.

**Reason** As value for `CDockable.getAction` when a predefined action should be hidden

### 2.2.9 CStation

**Short** Additional interface CStation in common. Two new stations: CMinimizeArea and CGridArea.

**Reason** Allows clients to add their own DockStations to CControl, allows to create other layouts than the "one center, four minimize areas"-layout.

## 2.3 Bugfixes

These are the bugs that were fixed/

### 2.3.1 BubbleDisplayer.getDockableInsets

**Short** The method did not calculate its result correctly.

**Reason** A flaw in the design of BasicDockableDisplayer

### 2.3.2 IndexOutOfBoundsException from ButtonPanel

**Short** The exception was thrown when an invisible action was on the panel

**Reason** invisible actions were not considered when writing ButtonPanel



### 2.3.3 Mode change of CDockable

**Short** CDockable did not go into normalized-mode when externalized and never normalized before

**Reason** Properties were missing and could not be created automatically

### 2.3.4 Opening maximized CDockable

**Short** CDockable could not be opened maximized.

**Reason** framework got confused because CDockable did not have a parent.

### 2.3.5 Unbind of DockAction called to often

**Short** A DockAction could throw an exception "unbind called to often"

**Reason** When a DockAction was a child of a MenuMenuHandler, its unbind method was called even if the action was not displayed. However the bind action was called only if the action was displayed, so the internal counter was no longer correct. Every time a menu with such an action was shown, the counter was decremented by one. When it reached a value below 0, an exception was thrown. Since an action could be bound by many elements, the exception occurred at random places.

## 3 Version 1.0.5

Version 1.0.5 brings the possibility to navigate around only by hitting some keys on the keyboard. When clicking the `ctrl+shift+e` combination, a dialog opens on which a Dockable can be selected.

DockActions in button form can be activated with `space`, and the dropdown actions menu can be opened with the `arrow down` key.

This release contains some tricky incompatibilities which need to be handled very carefully.

### 3.1 Incompatibilities

The changes that need special care.

#### 3.1.1 DockStationListener

**Short** The method `dockableSelected` of `DockStationListener` has an additional parameter that indicates which element was selected before the change.

**Reason** No need for listeners to store the old values.

**Clients** Must carefully update all classes and interfaces that implement `DockStationListener`. Be especially careful not to mix up the new arguments with the old ones.

### 3.1.2 DockableFocusListener

**Short** The `DockableFocusListener` has been divided into two interfaces: `DockableFocusListener` and `DockableSelectionListener`. The remaining method in `DockableFocusListener` now takes a `DockableFocusEvent` and no longer directly the involved elements. The class `DockableFocusAdapter` has been deleted.

**Reason** Events allow further changes of the system without change of the `DockableFocusListener` itself. Since every client needs to update its methods anyway, `DockableFocusAdapter` could be deleted.

**Clients** Should use `DockableFocusListener` instead of `DockableFocusAdapter`.

### 3.1.3 DockTheme.getDockableSelection

**Short** `DockTheme` has an additional method `getDockableSelection`.

**Reason** A `DockableSelection` is needed to change the focused `Dockable` using only the keyboard. Since `DockableSelection` is a graphical element, it has to be handled by the `DockTheme`.

**Clients** Should implement the missing method in their `DockThemes`. Using `DefaultDockableSelection` is an easy solution.

### 3.1.4 tap-strip no longer painted by TapPainter

**Short** `TabPainter` does no longer paint the tab-strip directly. It now creates a `TabStripPainter` that paints the strip.

**Reason** The new object can work with the color map.

**Clients** Have to provide a `TabStripPainter` as well.

### 3.1.5 KeyboardController does fire less events

**Short** The `KeyboardController` does no longer fire events when it could not find the source-`Dockable` of the event. As a result the `KeyboardListener` does no longer receive `null` as argument of any of its methods.

**Reason** Events were fired which had nothing to do with the framework at all.

**Clients** If they need all key events, then they can add a global `KeyListener` to `KeyboardController` using the method `addGlobalListener`.

### 3.1.6 ComponentHierarchyObserver

**Short** The `ComponentHierarchyObserver` includes more `Components` in its search. The `ComponentHierarchyObserverListener` now works with an event and does no longer receive all the elements as arguments.

**Reason** Allows more features to work correctly in restricted environments.

**Clients** Need to be aware that not every `Component` that is found by the observer is a child of a `Dockable`.

## 3.2 API and Layout

A list of new API elements and changes that affect the layout.

### 3.2.1 KeyStroke for closing Dockable

**Short** The `KeyStroke` for closing a `CDockable` or `Dockable` has been changed from `ctrl+c` to `ctrl+F4`.

**Reason** Andrew pointed out, that `ctrl+c` is already used by many applications...

### 3.2.2 New listeners

**Short** There are new listeners, `CFocusListener`, `CKeyListener` and `CDoubleClickListener`, which can be added to `CDockable` or to `CControl` if all `CDockables` should be monitored.

**Reason** Might be helpful for some applications

### 3.2.3 ComponentHierarchObserver

**Short** Clients can now add and remove `Components` from the `ComponentHierarchyObserver`. The observer also includes `DockTitles` in its search for `Components`.

**Reason** Might become necessary for complex applications that run in a restricted environment.

### 3.2.4 Root window for DockController

**Short** The `DockController` can now find the root window of the application. The window can also be set directly using `setRootWindow`. If so, then the root window is added to the `ComponentHierarchyObserver`.

**Reason** Necessary to show small dialogs like the new `DockableSelector`

### 3.2.5 FocusTraversalPolicies

**Short** All `DockThemes` now support `FocusTraversalPolicies`. Now each `DockAction` and all `Components` of a `Dockable` can be reached by using only the keyboard.

**Reason** A nice feature for people which do not like the mouse

### 3.2.6 Dialog to select focused Dockable

**Short** The `DockableSelector` and `DockableSelection` allow users to select the focused `Dockable` using only the keyboard. The feature is activated as soon as `ctrl+shift+e` is pressed.

**Reason** A nice feature for people which do not like the mouse

### 3.2.7 Extracting colors from LookAndFeel

**Short** The mechanism to read colors from `LookAndFeel`s has been upgraded. Each `LookAndFeel` can now have its own specialized `LookAndFeelColors` that reads the colors.

**Reason** Allows to be more flexible with colors, allows the correct use of Nimbus and Windows.

### 3.2.8 EclipseTheme

**Short** `EclipseTheme` uses more colors from the `LookAndFeel`

**Reason** looks better

### 3.2.9 SplitDockStation

**Short** When dropping an element onto a `SplitDockStation`, the elements that are put aside receive at least a quarter of their original size.

**Reason** Sometimes the old elements shrunk too much.

## 3.3 Bugfixes

### 3.3.1 Missing colors for BasicTheme

**Short** `BasicTheme` did not update colors for the keys `paint.line`, `paint.divider` and `paint.division`. As a result some painting was not as in the older versions.

### 3.3.2 Cutting bounds of children of SplitDockStation

**Short** The bounds of children of `SplitDockStation` are now cut such that they are always within the stations boundaries.

**Reason** Rounding errors sometimes lead to little failures that made a single line of pixels invisible.

### 3.3.3 NullPointerException when changing focus

**Short** A `NullPointerException` could be thrown when the focus changed.

### 3.3.4 Undecorated dialogs not undecorated

**Short** When using `LookAndFeel`s that can draw window decorations on their own (like `JTattoo`), then `FlapWindow`, `ScreenDockDialog` and others could have decorations.

**Reason** The flag that advices the `LookAndFeel` not to paint a decoration was not set in the `JRootPanels` of these windows.

### **3.3.5 RexTabbedComponent not adding/removing children**

**Short** `RexTabbedComponent` does no longer add and remove its children to change their visibility, it now uses a `CardLayout`.

**Reason** Some `Components` did miss the change of the `LookAndFeel` when they were a child of `RexTabbedComponent`.

### **3.3.6 Focusing a hidden CDockable**

**Short** When focusing a normalized `CDockable` that was hidden behind a maximized `CDockable`, then the focused dockable did not become visible.

**Reason** An old security system prevents change of the maximized element by the focus system.

### **3.3.7 Missing events when changing state of CDockable**

**Short** When the `ExtendedMode` of a `CDockable` did not change because of a call of a special method, no state-change-events were fired.

**Reason** It was not intended that one action could change the state of many `CDockables`.