

# Migration from MFC to Qt

Qt World Summit 2019, Berlin



Nicolas Arnaud-Cormos  
nicolas.arnaud-cormos@kdab.com

# About me

- Senior Software Engineer, KDAB
- C++/Qt developer since ~2002
- Living in Limoges, France



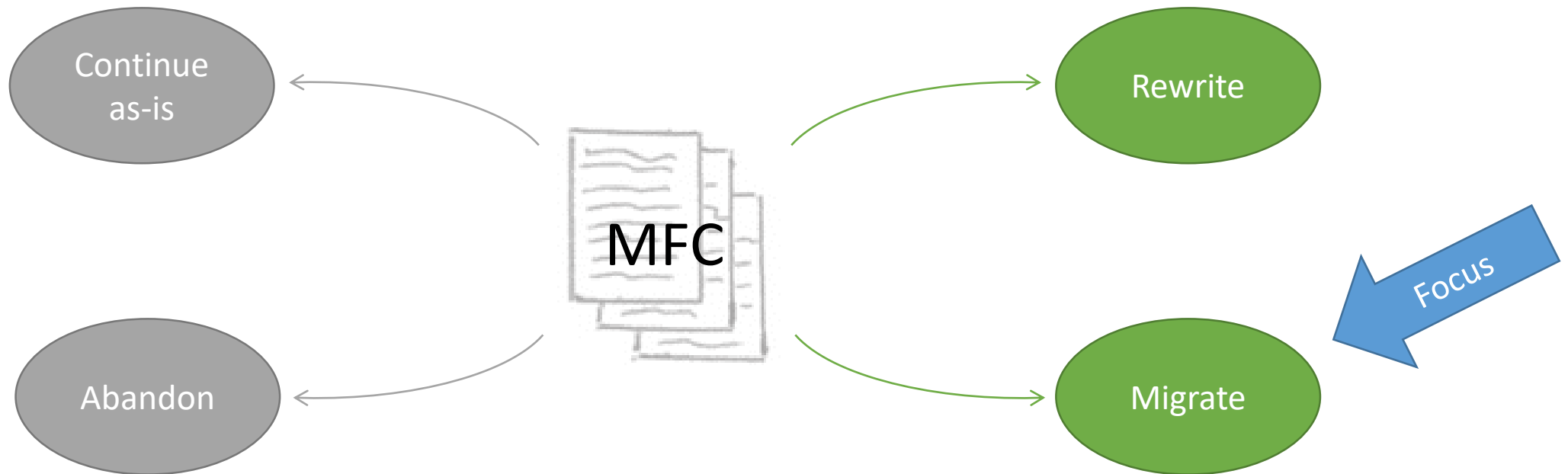
Who doesn't mind working on Windows?



# Modernizing legacy MFC code

# You got a big pile of MFC code...

... and now what?



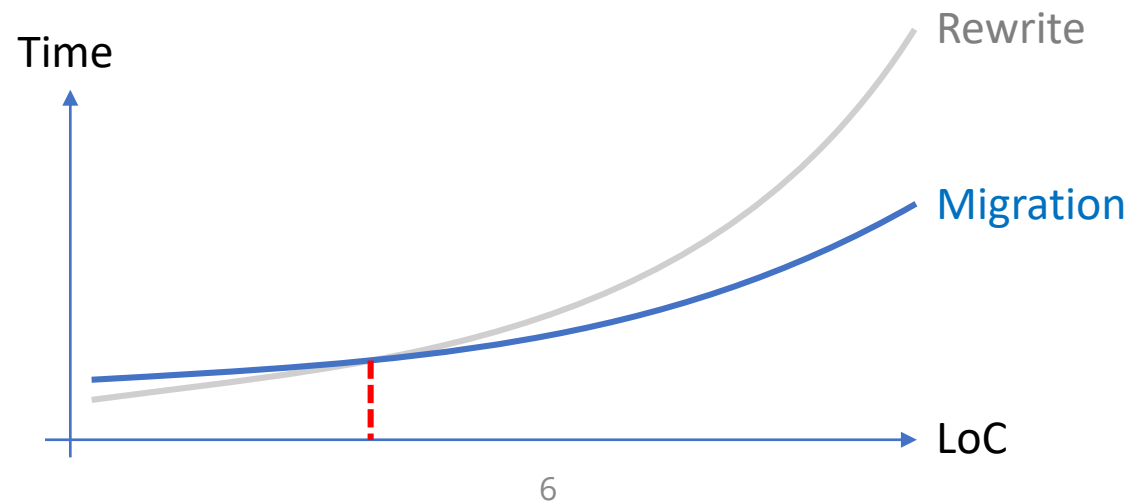
# Reasons to move to Qt

- The MFC framework is on life support
  - Qt is supported, is improving, and has a lively community
- The UI feels very out-dated
- The application needs to run on multiple platform
- The application needs to follow new UI paradims (touch-screens...)
- It's getting harder to add new features, fix bugs (technical debt)
- It's getting harder to find good MFC developers
- ...

**“It’s never the right time to  
move off MFC.”  
Make it happen!**

# Rewrite vs Migration

- + Start from clean state
  - + Not bound by technical decisions made decades ago
  - + Revisit and clean up requirements
  - Long time to market
  - Lost years of knowledge, corner-cases
- + Keep the good, replace the bad
  - + Migration can be done incrementally
  - + Smaller time to market
  - Developers must **know both MFC and Qt**
  - Developers must **know migration techniques**



# Refactoring

- **Avoid mixing migration and refactoring:**
  - Increase development time
  - Prevent comparing old / new code
  - Make it harder to find bugs
  
- **Advices for refactoring:**
  - Refactor once the migration is done (or once the module is migrated)
  - Rewrite if the module is isolated from the rest of the code (partial rewrite)



# What to migrate?

- **User interface**
  - Moving dialogs and UI elements to Qt equivalents.
- **Build system**
  - Moving from Microsoft-specific builds to a platform-independent build system
- **Non-UX code**
  - Moving the core MFC classes (like strings, files, containers...) to pure C++ or C++ with Qt as desired
- **Windows APIs**
  - Removing any Windows-specific APIs and replacing them with OS-generic equivalents





# MFC Migration Techniques

# Clean up your codebase

- Clean up warnings
- Document and simplify development setup
  - Make sure to have a reproducible setup
  - Make sure 3<sup>rd</sup>-party libraries are available
- Clean up warnings
- Remove dead-code
  - Use static analysis tools
- Clean up warnings
- (Optional) Beautify your code, update coding style
  - Automate the work with uncrustify, clang-format...
- Clean up warnings

There is a subliminal message hidden.  
**Can you find it?**

# Integrate Qt into the build system

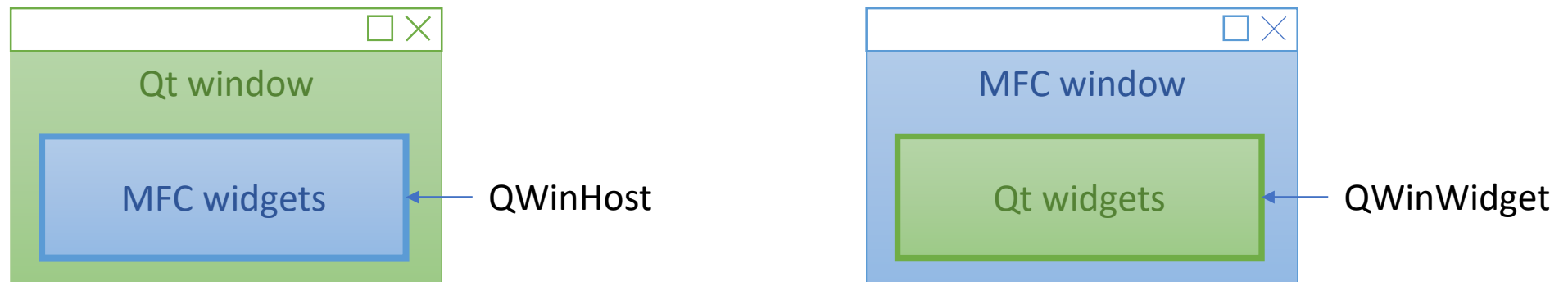
- Download and install Qt Visual add-in
  - <https://download.qt.io/archive/vsaddin/>
- Add support of Qt add-in in your project
  - Edit *project.vcxproj*
  - Add *QtVS\_v301* keyword
  - Add *qt\_defaults* property sheet
  - Add *qt.props* property sheet
  - Add *qt.targets*
  - Add *QtSettings*
- Decide how to handle PCH
  - **moc** supports adding an include
  - **rcc does not!**

```
diff --git a/MFC_UpdateGUI.vcxproj b/MFC_UpdateGUI.vcxproj
index 7eee3fa..48b3c67 100644
--- a/MFC_UpdateGUI.vcxproj
+++ b/MFC_UpdateGUI.vcxproj
@@ -14,6 +14,7 @@
<ProjectGuid>{78446EED-3469-467F-999B-AC5C2E7BD249}</ProjectGuid>
<RootNamespace>MFC_UpdateGUI</RootNamespace>
<Keyword>MFCProj</Keyword>
+ <Keyword>QtVS_v301</Keyword>
<WindowsTargetPlatformVersion>10.0.17763.0</WindowsTargetPlatformVersion>
</PropertyGroup>
<Import Project="$(VCTargetsPath)\Microsoft.Cpp.Default.props" />
@@ -30,6 +31,10 @@
<PlatformToolset>v141</PlatformToolset>
</PropertyGroup>
<Import Project="$(VCTargetsPath)\Microsoft.Cpp.props" />
+ <PropertyGroup Condition="'$(QtMsBuild)'!='' or !Exists('$(QtMsBuild)\qt.targets')">
+ <QtMsBuild>$(MSBuildProjectDirectory)\QtMsBuild</QtMsBuild>
+ </PropertyGroup>
+ <Import Project="$(VCTargetsPath)\Microsoft.Cpp.props" />
<ImportGroup Label="ExtensionSettings">
</ImportGroup>
<ImportGroup Condition="'$(Configuration)|$(Platform)'=='Debug|x64'">
@@ -41,6 +46,20 @@
<Import Project="$(VCTargetsPath)\Microsoft.Cpp.targets" />
</ImportGroup>
<PropertyGroup>
+ <Import Project="$(VCTargetsPath)\Qt\QtVS\PropertySheets\qt_defaults.props" />
+ <Import Project="$(VCTargetsPath)\Qt\QtVS\PropertySheets\qt.props" />
+ </ImportGroup>
+ <PropertyGroup Label="QtSettings" Condition="'$(Configuration)|$(Platform)'=='Release|x64'">
+ <QtInstall>5.12.5</QtInstall>
+ <QtModules>core;gui;widgets</QtModules>
+ </PropertyGroup>
+ <ImportGroup Condition="Exists('$(QtMsBuild)\qt.props')">
+ <Import Project="$(QtMsBuild)\qt.props" />
+ </ImportGroup>
<PropertyGroup>
<ProjectFileVersion>10.0.30319.1</ProjectFileVersion>
<LinkIncremental Condition="'$(Configuration)|$(Platform)'=='Debug|x64'">true</LinkIncremental>
@@ -117,6 +136,9 @@
<ResourceCompile Include="MFC_UpdateGUI.rc" />
</ItemGroup>
<Import Project="$(VCTargetsPath)\Microsoft.Cpp.targets" />
+ <ImportGroup Condition="Exists('$(QtMsBuild)\qt.targets')">
+ <Import Project="$(QtMsBuild)\qt.targets" />
+ </ImportGroup>
<ImportGroup Label="ExtensionTargets">
</ImportGroup>
</Project>
\ No newline at end of file
```

• Create an empty Qt project  
• Copy all XML tags related to Qt

# Add QtWinMigrate to your project

- Qt/MFC migration framework
  - <https://github.com/qtproject/qt-solutions/tree/master/qtwinmigrate>
  - BSD license
- Contains 3 classes
  - **QMfcApp** – merge the Qt and MFC event loops
  - **QWinHost** – integrate native Win32 widgets/windows into Qt ones
  - **QWinWidget** – integrate Qt widgets into native Win32 widgets



# Note on mixed MFC/Qt application

- QtWinMigrate allows to create mixed MFC/Qt application
  - The application is almost 100% feature complete during the port
  - Two issues with mixed MFC/Qt application:
    - Drag'n'drop from MFC to Qt (or vice-versa)
    - Focus handling
- 2 different approaches to migration



Avoid nesting too many levels of `QWinHost` / `QWinWidget`

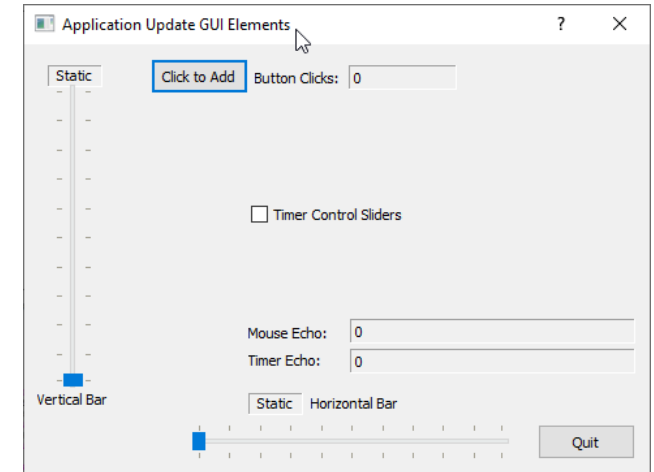
# Migrate dialogs

- UI descriptions are in the resource file (.rc)
  - Declarative description
  - “Easy” to parse
  - Different sections for:
    - Dialogs
    - Assets
    - Strings
    - Menus
    - Toolbars
    - ...

```

IDD_UPDATEGUI_DIALOG DIALOGEX 0, 0, 320, 200
STYLE DS_SEIFONT | DS_MODALFRAME | DS_FIXEDSYS | WS_POPUP | WS_VISIBLE |
WS_CAPTION
EXSTYLE WS_EX_APPWINDOW
CAPTION "Application Update GUI Elements"
FONT 8, "MS Shell Dlg", 0, 0, 0x1
BEGIN
  DEFPUSHBUTTON "Click to Add",ID_BTN_ADD,65,7,50,16
  PUSHBUTTON "Quit",IDCANCEL,263,177,50,16
  LTEXT "Button Clicks:",IDC_STATIC,118,11,50,8
  LTEXT "0",IDC_ECHO_AREA,166,10,55,11,0,WS_EX_CLIENTEDGE
  CONTROL "",IDC_V_SLIDER_BAR,"msctls_trackbar32",TBS_VERT |
TBS_BOTH | TBS_NOTICKS | WS_TABSTOP,17,19,17,141
  "",IDC_H_SLIDER_BAR,"msctls_trackbar32",TBS_BOTH |
TBS_NOTICKS | WS_TABSTOP,86,178,162,15
  LTEXT "Vertical Bar",IDC_STATIC,7,161,39,8
  LTEXT "Horizontal Bar",IDC_STATIC,147,163,47,8
  CTEXT "Static",IDC_V_SLIDER_ECHO,12,10,28,10,0,
WS_EX_CLIENTEDGE
  CTEXT "Static",IDC_H_SLIDER_ECHO,115,163,28,10,0,
WS_EX_CLIENTEDGE
  "0",IDC_TIMERECHO,167,142,146,11,0,WS_EX_CLIENTEDGE
  "0",IDC_MOUSEECHO,167,128,146,11,0,WS_EX_CLIENTEDGE
  LTEXT "Timer Echo:",IDC_STATIC,115,143,45,8
  LTEXT "Mouse Echo:",IDC_STATIC,115,130,45,8
  CONTROL "Timer Control Sliders",IDC_TIMER_CONTROL_SLIDERS,"Button",
BS_AUTOCHECKBOX | WS_TABSTOP,116,74,86,10
END

```



**Automate ui  
files creation**

# Migrate dialogs – cont.

- Migrate the MFC message map into Qt paradigms
  - Qt event handlers
  - Qt slots (or normal functions)

```
BEGIN_MESSAGE_MAP(CTutorialDlg, CDialog)
```

```
ON_WM_PAINT()  
ON_WM_TIMER()  
ON_WM_LBUTTONDOWN()  
ON_WM_MOUSEMOVE()  
ON_WM_RBUTTONDOWN()
```

```
ON_WM_HSCROLL()  
ON_WM_VSCROLL()  
ON_BN_CLICKED(ID_BTN_ADD, OnBnClickedBtnAdd)  
ON_BN_CLICKED(IDC_TIMER_CONTROL_SLIDERS, ...)
```

```
END_MESSAGE_MAP()
```

## Qt event handlers

```
void paintEvent(QPaintEvent *event);  
void timerEvent(QTimerEvent *event);  
void mousePressEvent(QMouseEvent *event);  
void mouseMoveEvent(QMouseEvent *event);
```

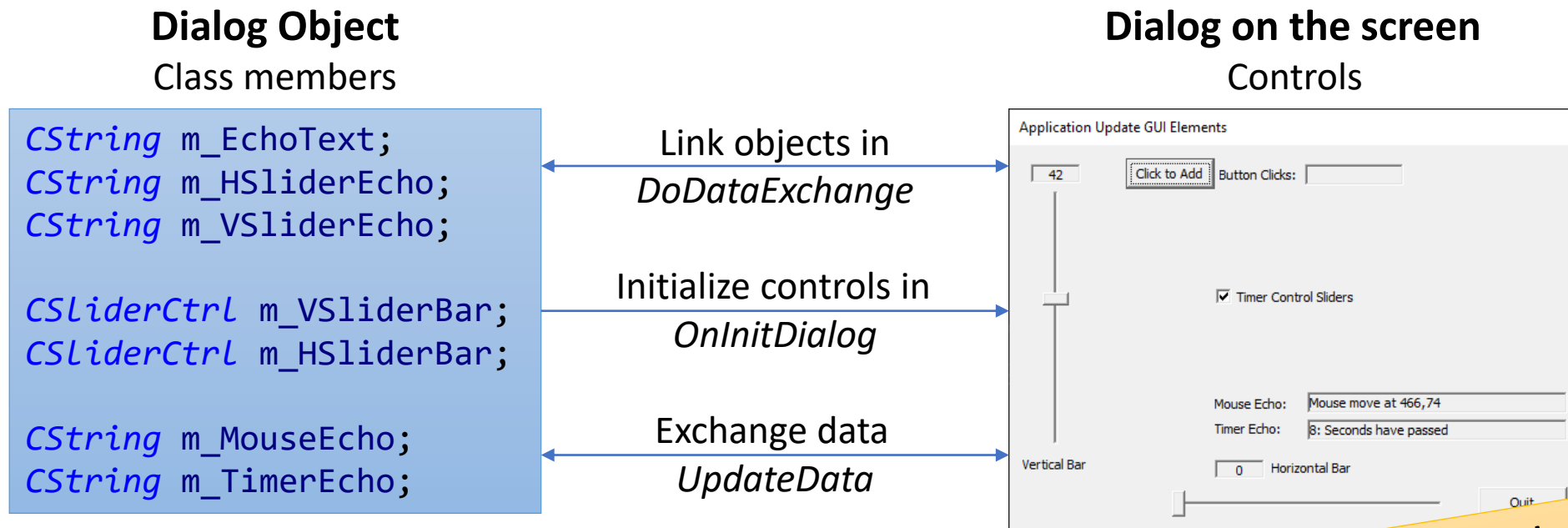
## Qt slots

```
void OnHScroll(int value);  
void OnVScroll(int value);  
void OnBnClickedBtnAdd();  
void ...();
```

**Automate message  
map migration**

# Migrate dialogs – cont.

- MFC dialog data exchange mechanism



- Qt does not need this complex mechanism
  - Use the widget pointers for the ui directly
  - Use Qt API for widget calls
  - Defer initialization using *showEvent*

**Automate** data exchange migration and API migration



# Migrate other resources

- The resource files (.rc) contain:
  - Menus definitions
  - Toolbars definitions
  - Shortcuts (accelerators) definitions
  - Assets list
  - Strings translations
  - Languages, versions...
- During the migration, it's important:
  - To automate the migration of resources
  - To be able to refactor the migrated code



**Automate any  
resource migration**

# Migrate controls

- Simple controls (checkbox, buttons...):
  - Use simple Qt widgets directly
  - Avoid any behavior changes (even if done in MFC)
- Complex controls (property grid, dock system...):
  - Usually from existing MFC libraries (ex: [Toolkit Pro](#))
  - Create a new Qt widget with:
    - Same behaviour as MFC controls
    - Same (or close) API as MFC controls
  - **Minimize changes in code during migration**
- Use existing Qt libraries:
  - Advancing docking system: [KDDockWidgets](#)

Image from CodeJock website

Drag a column header here to group by that column.					
	From	Subject	Received	Size	
!	postmaster@mail.codejock...	Undeliverable Mail	Wed 06/01/2005 ...	7	<input type="checkbox"/>
	<b>Peter Parker</b>	<b>RE: Hi Mary Jane</b>	<b>Tue 05/31/200...</b>	<b>14</b>	<input type="checkbox"/>
	<b>James Howlett</b>	<b>RE:</b>	<b>Thu 06/02/200...</b>	<b>24</b>	<input checked="" type="checkbox"/>
↓	Bruce Banner [bbanner@c...	Re: error C2039: 'Serialize' : is not a m...	Tue 06/28/2005 0...	14	<input checked="" type="checkbox"/>
!	<b>QueryReply</b>	<b>Re: Download GDI+ version 1.1</b>	<b>Sat 06/18/2005...</b>	<b>13</b>	<input type="checkbox"/>
↓	<b>James Howlett</b>	<b>I don't understand:</b>	<b>Mon 06/06/200...</b>	<b>24</b>	<input checked="" type="checkbox"/>
	Matthew Murdock	Re: WIN32_FIND_DATA help	Sun 07/10/2005 0...	14	<input type="checkbox"/>
	Louis Lane	Re: Licensing	Fri 06/10/2005 03...	13	<input checked="" type="checkbox"/>
	postmaster@mail.codejock...	Undeliverable Mail	Thu 06/23/2005 1...	7	<input type="checkbox"/>
	CodeToolBox	[CodeToolBox] Newsletter (10 May 2004)	Thu 06/23/2005 0...	7	<input type="checkbox"/>
	<b>Codejock Exchange</b>	<b>Good Answer! Capture message s...</b>	<b>Mon 06/13/200...</b>	<b>8</b>	<input type="checkbox"/>
	<b>CodeToolBox.com Team</b>	<b>CodeToolBox.com Sitewide updat...</b>	<b>Sun 06/26/200...</b>	<b>46</b>	<input type="checkbox"/>
	Codejock Exchange	Comment Added: returning a <m...	Mon 06/13/200...	5	<input type="checkbox"/>
	Bruce Wayne	Comment Added: memory leak in ...	Sun 05/29/200...	5	<input type="checkbox"/>
	Codejock Exchange	Good Answer: stl containers	Thu 06/16/200...	4	<input type="checkbox"/>
	Codejock Exchange	Codejock Exchange: Lost Member...	Thu 07/07/200...	7	<input type="checkbox"/>
	CodejockDirect	New volume/issues available on C...	Sun 07/03/200...	9	<input type="checkbox"/>
	Kyle Rayner	info	Sat 05/28/2005...	67	<input type="checkbox"/>

Find the right level of abstraction

Automate migration of project-specific controls

# We have barely scratched the surface

- Document / View architecture
  - CDocument, CView...
  - SDI, MDI...
- Printing and print previewing
- Threads
  - CWinThread...
- Component Object Model
  - COM, OLE, ActiveX

# Conclusion

## To sum up

- **Choose** your migration strategy wisely
  - Be prepared to defend it internally
  - Don't mix migration and refactoring
- **Clean up** the code before anything
- **Use** the Qt/MFC migration framework
  - Don't lose sight of the big picture: removal of MFC
- **Find** the right level of abstraction
  - Use existing libraries if they exist
- **Automate** all things:
  - resource migration
  - MFC code migration
  - project-specific code migration
- **Limit** code differences

# Thanks!

[nicolas.arnaud-cormos@kdab.com](mailto:nicolas.arnaud-cormos@kdab.com)