



delphiedintorni.it
 *delphi italian community*

Automating IDE Components



PAOLO ROSSI
WINTech ITALIA **CTO**

SENCHA & EMB



BLOG



blog.paolorossi.net

GITHUB PROJECTS



github.com/paolo-rossi





GITHUB PROJECTS



Delphi JWT

JSON Web Token Library for REST



Linux Daemon

Library to build real Linux daemons



Delphi Neon

JSON Serialization Library for REST



OpenAPI-Delphi

OpenAPI 3.0 library for Delphi



WiRL Project

JAX-RS Like REST Library for Delphi



delphiedintorni.it
 *delphi italian community*

Automating IDE Components

AGENDA

- The problem
- Source code, libraries and components
- Version Control System
- Installing (IDE) components
- File batch or MSBuild compile
- Automating component installation

THE PROBLEM (1)

Several component sets

+

New Delphi release

=

Time consuming

THE PROBLEM (2)

Several component sets

+

Several Delphi releases

=

Nightmare

* Yes I know, there is a technology called virtualization

THE PROBLEM (3)

Different component versions

+

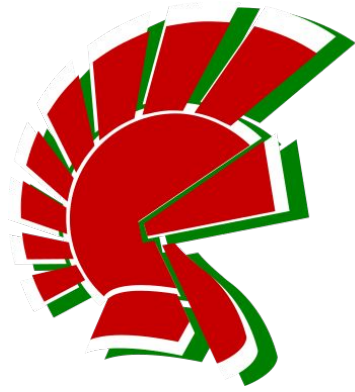
Same Delphi release

=

Impossibile (is it?)

* Yes I know, there is a technology called virtualization

demo time



Example:

The goal: automatic build!

SOURCE CODE CATEGORIES

- Source code
 - ◆ Your own code
- Libraries
 - ◆ External libraries
 - ◆ Your own code
- Design-Time Components
 - ◆ Installed in the Delphi IDE
 - ◆ Used at Design-time

SOURCE CODE CATEGORIES

- Source code
 - ◆ Your own code
- Libraries
 - ◆ External libraries
 - ◆ Your own code
- Design-Time Components
 - ◆ Installed in the Delphi IDE
 - ◆ Used at Design-time

CODE STRUCTURE IS ALL

→ Project's structure evolution

- ◆ 1 directory, all the files (dpr, pas, dfm, res, dcu, exe)
- ◆ I've seen things you people wouldn't believe...

CODE STRUCTURE IS ALL

→ Project's structure evolution

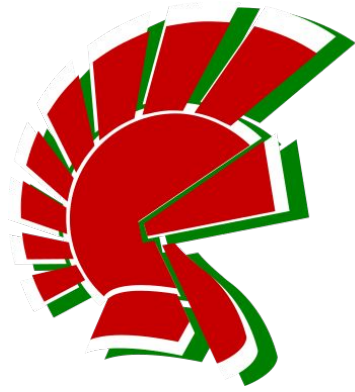
- ◆ Different directories from different file types
 - source (pas, dfm, inc), lib (dcu), bin (exe, dll, etc...)
 - projects (dpr, dproj, res, etc...)

CODE STRUCTURE IS ALL

→ Project's structure evolution

- ◆ Different directories for sub-projects
 - source/configurator, source/client, source/server
 - External libraries (svn ext, git sub-modules)
 - Components ->

demo time



Example:

Complex project structure

VERSION CONTROL

→ Centralized (cvs, svn)

- ◆ Central copy of your repository
- ◆ Commit (only) to the remote repo
- ◆ Simpler workflow

→ Distributed (git, hg)

- ◆ Every developer “clones” the full repository locally
 - Has the full history of the project
- ◆ Commit to your repo and the push to the remote
- ◆ More complex workflow

IDE COMPONENTS

- Design-time components management knowledge is virtually non-existent
 - ◆ Even in skilled Delphi Teams
- Who manages them?
 - ◆ (Usually) the most experienced developer
 - ◆ He writes some documentation about installing them

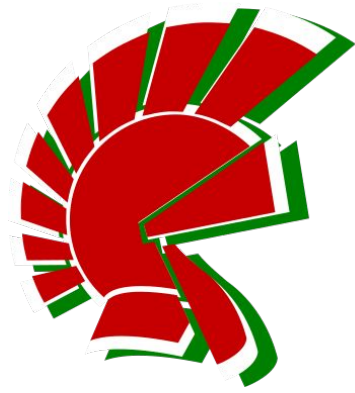
WHAT THEY ARE

- File .dpk (Delphi Package)
- Runtime packages / Designtime packages
- You have to build (runtime) and build+install (designtime)

SOURCE OR NOT ?

- Always purchase the source of your components
 - ◆ I know! This is the most expensive option
 - ◆ But in the long run it's more affordable!
- With the source you can move your actual Delphi components for the next Delphi version

demo time



Example:

Component library structure

INSTALLATION

- Windows Installer (*.exe, *.msi)
 - ◆ I hate them (for component installations)
- Make file / Batch file
- Manual install (*.dpk)
 - ◆ Runtime packages
 - ◆ Designtime packages

WINDOWS INSTALLERS

- No thanks!
- Binary installers (exe, msi, etc...) for components are (usually) bad!
- Because (some) vendors do not follow the rules
- Preferable a batch file or a make file

WINDOWS INSTALLERS

- What if I have only the installer option?
- It's not that hard to DIY
 - ◆ Install a copy of the components in a VM
 - ◆ Copy the source (.pas, .dpk)
 - ◆ Install the .dpk your way

AUTOMATIC INSTALL

- Automatic package installer
 - ◆ Compile a .dpk file
 - ◆ Copy the .bpl and the .dcp in the right directory
 - ◆ Add the .bpl to the registry (Known Packages)
- Several (OS) projects
- Best of all: DIY

THE .DPK (DPROJ)

→ Meet the LIBSUFFIX

◆ Project Options → Description → LIB su^ux

→ LIBSUFFIX is Delphi version dependant

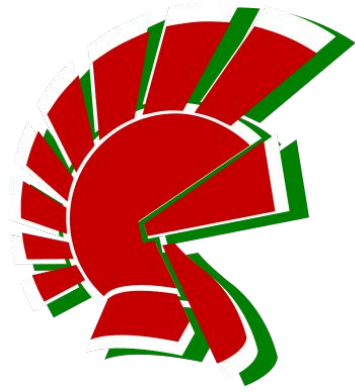
◆ Adds the su^ux to the BPL

● You must refer to the BPL without the su^ux

◆ You have to change it for every Delphi version

◆ Unfortunately there is no “AUTO” setting (JIRA issue)

demo time



Example:
dpk options (Delphi)

BUILDING THE DPK

→ dcc32.exe

- ◆ A looooot of options!
- ◆ Parameter passing on the command line

→ MSBuild

- ◆ MSBuild uses the dproj file
- ◆ A few parameters to pass
- ◆ Easiest way to build Delphi projects
 - Not only packages

NEXT DELPHI VERSION

- Copy the Dxxx components root
- For each component
 - ◆ Update the dpk & dproj
 - LIBSUFFIX & DLLSUFFIX
 - ◆ Update component's IFDEF
 - {\$IFDEF VERxxx}
 - Use {\$IF CompilerVersion > xx}

demo time



Example:

Simulate the D10.2 to D10.3 porting

BOTTOM LINE

Automate the boring stuff !

**Take control of the components
installation process**

**The “Next Delphi” full
automated install could be
a few clicks away!**



GRAZIE PER
LA PARTECIPAZIONE

DOMANDE?



Automating IDE Components



**NEED HELP
WITH YOUR
COMPONENTS?**

CONTACT ME

paolo.rossi@wintech-italia.com



Automating IDE Components

HAI UN ARGOMENTO INTERESSANTE?



Contattami a questo indirizzo

paolo.rossi@wintech-italia.com