

Re: Delphi version control

Source: <http://coding.derkeiler.com/Archive/Delphi/comp.lang.pascal.delphi.misc/2004-12/0356.html>

From: Martin Harvey (Demon account) (*martin_at_nospam_pergolesi.demon.co.uk*)

Date: 12/17/04

Date: Fri, 17 Dec 2004 20:01:20 GMT

On Wed, 15 Dec 2004 10:35:11 GMT, Jeremy Collins
<jd.collins@ntlworld-not.com> wrote:

>Finally, anyone out there who is not using some sort of revision
>control system, **START TODAY!** There's no excuse not to, even if
>you're a single developer. I like CS-RCS (it's free, Google it),
>which IIRC comes with a nice beginners tutorial.

Just a few comments, having used CS-RCS, RCS, CVS, PVCS, SourceSafe,
and a quick look at SubVersion.

First off, I agree – EVERYONE should use some sort of revision control
system. If you're ever thinking of releasing anything to the public,
you **MUST** have a way of keeping track of which versions contain which
source code. Even if you're not, it an excellent way of keeping track
of changes, formalising what's in the "program proper" and what's
currently being worked on, and god forbid, if you ever go down the
wrong route, getting back to where you were before.

Second off, there are some files that the compiler generates that you
should archive. I **STRONGLY** recommend getting Delphi to generate a
detailed map file, and archiving that as well.

Third off: Proper build procedures. You need to be sure that you can
recreate any version precisely from source control. This means, that
when you do a release, You should do at least the following:

- Update the version info in resources.
- Save and check in as appropriate.
- Get a completely fresh copy from source control. Most people do
release builds in a separate part of the directory tree, and gives you
confidence that there's no caching or out of date information
involved. Also gives you confidence that you'll be able to rebuild
that version later if need be.
- Do a full rebuild of everything, and make sure it generates a map
file.
- Check in binaries, map files and other symbolic information as
required (e.g. .pdb files).

- Label the whole lot (source and binaries) with the same label.
- Relax in the knowledge that when the customer phones with: "I've got access violation at address XXX", you actually stand a hope in hell of finding where that is.

At my previous company, the build procedure ran to about 50 points, and several pages – and giving anything to a customer that hadn't been properly released was darn close to a disciplinary offence. On the one occasion when I went away on holiday, and someone else released a copy of my app without labelling it properly, they got shouted at very loudly when I got back!

Comments on revision control systems.

I'm currently using CS–RCS at home, and for single users it's great. If you want to have multiple users and start doing branching . merging, it gets more tricky.

CVS is fine, but it has a slightly different way of thinking from (for example) Source–Safe, and RCS. However, it's trivial to import archives from RCS to CVS.

PVCS – probably not much used nowadays, and considering you have to pay for it, there are much better alternatives.

Source–Safe. Okay, but it has two downsides:

- Branching is a pain.
- The version I used at least (5?): It has a magical ability to corrupt its own databases, and they're not easily rebuildable. Only

MH.