Subject: Re: The end (?) of ubuntu packages
Posted by seasoned_geek on Fri, 10 Aug 2018 16:23:04 GMT
View Forum Message <> Reply to Message

Alboni wrote on Fri, 10 August 2018 06:36Compiling on a pi is indeed is bit painful. Takes a whole night to compile theide & umk with the makefile in the tarball. Of course using umk it could go faster but I can't really imagine someone to be interested in nightly binaries of theide and umk on a pi or embedded system and the source packages can be used on all platforms. So I can imagine doing something like this:

Stable repository:
-stable binary ubuntu xenial 32+64 bit
-stable binary ubuntu bionic 32+64 bit
-stable binary ubuntu cosmic 32+64 bit (when it becomes available)
-stable binary for current version of raspian
-stable source package for all platforms

nightly repository
-nightly binary ubuntu xenial 64 bit
-nightly binary ubuntu bionic 64 bit
-nightly binary ubuntu cosmic 64 bit (when it becomes available)
-stable binary ubuntu xenial 32 bit
-stable binary ubuntu bionic 32 bit
-stable binary ubuntu cosmic 32 bit (when it becomes available)
-stable binary for current version of raspian
-nightly source package all platforms

(trusty's compiler only has experimental support for C++11 so maybe an oldstable version also if there is a need)

& the occasional odd request

(but I still need to check out launchpad)

Honestly, I wouldn't worry about nightly builds. Anyone doing serious development for a Raspberry is using an LTS YABU and most likely has locked it down from updates.

I did have some stuff for cross compiling Qt to Raspberry Pi on my geek blog:

http://www.logikalsolutions.com/wordpress/raspberry-pi/raspb erry-qt-part-1/

When you select a category you will find one for Raspberry Pi with about 30 posts. No, I didn't get all the way to the end of the series as paying work came up and development OS changed significantly. We all had to come up with our own because the wiki instructions were horribly broken for years.

The boys and girls at Qt swear the new wiki instructions work for 64-bit host.
https://wiki.qt.io/Raspberry_Pi_Beginners_Guide

You should be able to leave out the Qt parts.

I stumbled into just how broken the original instructions were around here

http://www.logikalsolutions.com/wordpress/information-techno logy/how-far-weve-come-pt-18/

When it comes to most of the business client situations I encounter in the Qt world (and therefore where U++ should focus IMO) it is a full screen app without a desktop. Just mouse and touchscreen support. Versions 4.x had a command line switch you could use which would load a minimalist windowing engine (with more than a few quirks), other places had other solutions.

Most places want to develop a completely custom UI for their embedded system. I wrote roughly half of the Qt based source code for the UI on this product.

https://www.welchallyn.com/en/products/categories/patient-mo
nitoring/vital-signs-devices/connex-spot-monitor.html

We had no GPU so had to preload and blit our images onto the screen. There were times when stepping in front of an Amtrak train seemed a better option than staying until the end of the project, but that UI finally came out sexy. That main screen dynamically reconfigures itself based on whatever mode a clinician is in. The blocks reshape and reposition themselves. Some place on that link is a link to pull down the user manual for the device. It contains some really sexy looking screen shots of various other screens.

My initial interest in U++ is simply in getting it working on my desktop to create some of the initial test applications I always write. A lottery tracker, that serial keypad thing and a few others. Then I wanted to see how it would be for embedded system type applications with a completely custom screen, message queues and a bit of network communications. I'm kicking the tires to see just how viable it would be in the world where I now live given it doesn't have dynamically changing license requirements.

I liked some of the screen shots I saw which appeared to indicate the package has dynamically selectable styles. I'm a bit interested in just how difficult it is to create a custom style (a totally dark art in the Qt world)

Sorry for the long winded reply. Not trying to brag, just trying to provide a frame of reference since people on here don't know me from Adam.