Subject: U++ versioning
Posted by johnevans77 on Sat, 21 Jul 2007 16:43:21 GMT
View Forum Message <> Reply to Message

Deal all,

How ultimate++ deal with version numbering? Are 505, 605, 606 development version and 2007.1 is stable version? Or bigger version always better and stable?

And, i think in Status & Roadmap, we should put release date:)

Please advise.
JE

Subject: Re: U++ versioning
Posted by fudadmin on Sat, 21 Jul 2007 16:52:29 GMT
View Forum Message <> Reply to Message

As I understand, from the year 2007 => 2007.x , where x is month number, was decided to mark those as stable versions.

701, 702 etc.
^ the same year - but dev(elopement) versions.
605 - from year 2006 - 5 month (May?)

Subject: Re: U++ versioning
Posted by johnevans77 on Sat, 21 Jul 2007 17:01:31 GMT
View Forum Message <> Reply to Message

Thanks for the information

Subject: Re: U++ versioning
Posted by mirek on Sat, 21 Jul 2007 19:11:45 GMT
View Forum Message <> Reply to Message

Note that there were two changes in versioning on the way...

First it was regular 0.1, 0.2 etc...

Then we changed to 510, 605 etc... but later started 605dev1 for development snapshots... 10 and 05 are months.

Starting this year, it is 2007.1, where first number is year, second is the release number in the

year. Dev version keep 610dev1 "method".

Subject: Re: U++ versioning
Posted by johnevans77 on Sun, 22 Jul 2007 04:18:49 GMT
View Forum Message <> Reply to Message

Big thanks for explanation. The latter one should be clearer for some people like me

JE

Subject: Re: U++ versioning
Posted by amrein on Wed, 27 Aug 2008 14:22:23 GMT
View Forum Message <> Reply to Message

Hi

I have an issue with those version names. If I want to make a dynamic .dll for Windows or an dynamic .so for Linux/Unix, I will be in trouble.

Could U++ use something like upp.version.release.bugfix instead of year.bugfix ?

Explanations

Version: incremented if dynamically linked software won't work any more (api changed and old class/functions removed)

release: incremented if new class/functions/variables are added

bugfix: incremented if only bug fix have bean added to previous bugfix-1 release

Subject: Re: U++ versioning
Posted by mirek on Wed, 27 Aug 2008 14:32:22 GMT
View Forum Message <> Reply to Message

amrein wrote on Wed, 27 August 2008 10:22Hi

I have an issue with those version names. If I want to make a dynamic .dll for Windows or an dynamic .so for Linux/Unix, I will be in trouble.

I think if you are going to do .so, you can perhaps adopt your own scheme.

Quote:
Version: incremented if dynamically linked software won't work any more (api changed and old

class/functions removed)

release: incremented if new class/functions/variables are added

bugfix: incremented if only bug fix have bean added to previous bugfix-1 release

Well, unfortunately, in U++ all these things usually happen at the same time... and in reality, nobody tracks them.

Sorry, DLL hell is something we deliberately decided to completely avoid and forget about...

The problem is that in C++, it is way too easy to break binary compatibility. You either have to screw your C++ style and adhere to some "limited C++", or forget about it. We decided, long time ago, to forget about it.

Mirek

---

Subject: Re: U++ versioning
Posted by amrein on Wed, 27 Aug 2008 19:08:46 GMT
View Forum Message <> Reply to Message

2008 for the year. 01 for the release. From now, all new U++ release with have a name like 20xy.z.

But, for bug fix release, will it be 2008.2 or 2009.1 if the new release come in 2009?

Could U++ use 2009.0.0 instead of 2009.1 for next major release? That way, I will be able to use 9.0.0.

---

Subject: Re: U++ versioning
Posted by mirek on Wed, 27 Aug 2008 21:49:56 GMT
View Forum Message <> Reply to Message

If there is bugfix release, it would be 2008.1.1 or perhaps 2008.1a...

But all of this is not carved in the stone. We can adapt...

Mirek

---

Subject: Re: U++ versioning
Posted by Mindtraveller on Wed, 27 Aug 2008 22:52:38 GMT
View Forum Message <> Reply to Message

May be plain solution?
YYYY.MM.DD-vv

Subject: Re: U++ versioning
Posted by amrein on Thu, 28 Aug 2008 00:25:32 GMT
View Forum Message <> Reply to Message

Well there is an issue with y.m.d.v: year month and day are to far from version.release.fix.

It's not interesting to name a library 2008.8.25 and its next release 2009.1.5. Version should not change if you just add a few class/func without breaking compatibility. This is how Linux library naming works. It's because when program X want to link to library 2008.x.x, the dynamic loader won't use the new library 2009.x.x. The dynamic loader will only find and load the library with name 2008.zz.zz

The only solution is from luzr message:

"I think if you are going to do .so, you can perhaps adopt your own scheme."

Subject: Re: U++ versioning
Posted by amrein on Wed, 03 Sep 2008 17:08:57 GMT
View Forum Message <> Reply to Message

No problem with 20xy.z.

For libraries, I will be able to use 0.xy.z.

Note: The zero releases can break API as they want.

Subject: Re: U++ versioning
Posted by bytefield on Wed, 03 Sep 2008 20:24:25 GMT
View Forum Message <> Reply to Message

Hi amrein. Even if your suggestions are not all applied to U++ them are useful, so keep posting your ideas. For example i will know which version scheme to apply to my applications from now on  (version.release.bugfix).