Subject: U++ and Anboto join forces
Posted by Klugier on Mon, 25 Jul 2022 20:39:10 GMT
View Forum Message <> Reply to Message

Hello U++ community,

I would like to inform you that we came into agreement with Inaki. You might know Inaki as Koldo ;) It means that right now UppHub is using packages directly from his repositories that live in Anboto organization. Previously we used forked version of this packages within Libraries4U repository. This is the great news to all U++ community, because after separation, we are together again.

What does it mean for regular users? Regular UppHub users will have access to better maintain packages which are developed on an ongoing basis. From U++ developers point of view, we do not need to maintain forks, which was costly. We could use this time to develop new functionalities or fix bugs.

For Libraries4U contributors. If you found that your changes are not in Anboto's versions, you need to create PR for Anboto and discuss with Inaki. He will make final decision whenever your PR should be integrated or not. I know that some time ago our forum user busiek made several PR to our forks. Here is an example.

If you found that something doesn't work within UppHub. Just uninstall all Anboto's packages and install again. The links to Anboto in UppHub have been replaced in this commit.

Klugier

Subject: Re: U++ and Anboto join forces
Posted by jjacksonRIAB on Thu, 28 Jul 2022 07:44:25 GMT
View Forum Message <> Reply to Message

Cool, I kind of saw the schism developing and was wondering what was going on but didn't want to wade into it. Koldo's a solid contributor with a lot of interesting ideas and his work has helped fill in gaps in my own projects. It's good everything's coming back together.

Subject: Re: U++ and Anboto join forces
Posted by koldo on Mon, 01 Aug 2022 14:34:19 GMT
View Forum Message <> Reply to Message

With the invaluable help of Klugier, it was very easy.

I advise all of you who have libraries to contribute them to UppHub, which is easy to use and very powerful. This way we will all be rowing in the same direction.

Maybe we won't make U++ stop being one of the best kept secrets in efficient programming, but

at least we will make it more useful for the high-level developers who use it, without making the main core of U++ too big.