Subject: [Proposition] Simply source package manager for Upp
Posted by Klugier on Sat, 13 Jun 2020 20:18:34 GMT
View Forum Message <> Reply to Message

Hello,

I would like to start the discussion about adding simply source package manager for Upp. The general idea behind it is to add easier way of consuming 3-rd party code. For example, I would like to use Oblivion's MessageCtrl or TerminalCtrl. Right now I need to copy that source code from GitHub to one of my assemblies and consumes it there. Of course such approach is problematic due to the need of manual work like coping file, also updating such package requires additional manipulations. My idea assumes that this work can be simplify by specifying the GitHub repo link with the version (latest by default). Here is my proposition to .upp file modification:

```
description "Rudimentary HelloWorld application with menu and status bar\377";

uses
    CtrlLib;

uses_remotes
    https://github.com/ismail-yilmaz/message-ctrl.git@master;
    https://github.com/ismail-yilmaz/terminal-ctrl.git@1.40.1;


file
    hello.cpp;

mainconfig
    "" = "GUI";
```

In the source code, all you need to do is to include this packages like the regular ones

```
#include <MessageCtrl/MessageCtrl.h>
#include <TerminalCtrl/TerminalCtrl.h>

// ...
```

It will require one repository per package. Repositories that will aggregate several packages will be prohibited.

The general idea is exactly the same as we can see in golang. It is source only package manager without binaries. Precompile binaries support can be added in the future, however it is not critical right now.

Adding simply package manager will encourage developers to create their own packages. In the

long term it will mage Upp much stronger with bigger community engagement. Moreover, it will be huge opportunity to clean bazzar.

I would like to here what do you think about such improvement.

Sincerely,
Klugier

---

Subject: Re: [Proposition] Simply source package manager for Upp
Posted by koldo on Sat, 13 Jun 2020 21:13:32 GMT
View Forum Message <> Reply to Message

I am not able to understand this obsession against Bazaar, as it gives consistency to U++ supporters effort, that if not, it could be dispersed in different external repositories. I would ask Mirek not to lose this cohesion.

However, if it is decided that GitHub is going to be the main repository for U++, Klugiers's proposition, or something similar, would have to be sooner or later implemented.

In fact, I have felt myself in this situation today. If you expect to deploy your project in GitHub, like BEMRosetta in my case, today it is a problem to deploy your sources with U++ in a coordinated manner so user can get and compile updated sources properly.

---

Subject: Re: [Proposition] Simply source package manager for Upp
Posted by Klugier on Sat, 13 Jun 2020 22:35:11 GMT
View Forum Message <> Reply to Message

Hello Koldo,

Quote:I am not able to understand this obsession against Bazaar, as it gives consistency to U++ supporters effort, that if not, it could be dispersed in different external repositories. I would ask Mirek not to lose this cohesion.
I would like to thank you for your feedback.I think there is no obsession against Bazaar. It is good project, however we can handle it better and in more decentralize manner. Moreover, all packages can be developed independently. Even if we remove Bazaar from upp distribution, it doesn't mean that we will stop promoting Bazzar packages. Their packages should be present on website and there should be the possibility to download Bazzar examples from TheIDE.

Quote:However, if it is decided that GitHub is going to be the main repository for U++, Klugiers's proposition, or something similar, would have to be sooner or later implemented.
It is just a proposition, however as we want to grown. We should focus on making upp a platform that is easy to expand.

Quote:In fact, I have felt myself in this situation today. If you expect to deploy your project in

GitHub, like BEMRosetta in my case, today it is a problem to deploy your sources with U++ in a coordinated manner so user can get and compile updated sources properly.
Optimally with the package manager we should be able to overcome this issue. You should have single script/makefile/cmakelist that will download umk, package manager and target upp platform. So, .upp file should support remote assemblies. Here is some conceptual proposition:

```
description "Rudimentary HelloWorld application with menu and status bar\377";

uses
    CtrlLib from remote uppsrc; // <- Remote assemblies support...
    MessageCtrl from remote;
    TerminalCtrl from remote;

remotes
    https://github.com/ultimatepp/uppsrc.git@2020.1;
    https://github.com/ismail-yilmaz/message-ctrl.git; // <- If version not specified use latest master version
    https://ultimatepp.org/ismail-yilmaz/terminal-ctrl.git@1.40.1; // <- You should be able to use any git provider not only GitHub

file
    hello.cpp;

mainconfig
    "" = "GUI";
```

The version compatibility and dependencies should be designed with careful. Please notice that some other remote packages can have dependencies, too. So, we should be extremely careful. We do not want to create dependency hell

Sincerely,
Klugier

---

Subject: Re: [Proposition] Simply source package manager for Upp
Posted by Oblivion on Sun, 14 Jun 2020 17:46:21 GMT
View Forum Message <> Reply to Message

Hello,

As a long time U++ user, allow me to state my initial stance (before we delve into any discussion)

1) I get the impression that the proposed approach has a potential to break things in TheIDE, umk, and/or makefiles.
Because it seems to require heavy patching and has a tendency to get overly complex. (I may be wrong, of course.)

2) Bazaar is a nice way to keep extras in one place. So I agree with Koldo here.

2) OTOH, I do think that at least TheIDE can do better. At least there should be an easy way to download, set up, update, check, remove GIT/SVN/etc. repos as assemblies, nests, or packages, in a guided manner ("next -> next -> ok", I mean.). This can be done in the main package selection screen of TheIDE and later via a file menu item. In short, IMHO the main package selection screen should allow users to manage their local/remote assemblies/packages/nests as transparently as possible.

I believe this general approach can provide a smoother way to achieve something similar to "remotes" integration. In this way, say a remote git repo, could be direcly cloned as an assembly/nest/package and can be worked on locally. But we'll get the chance to update the local copy periodically, push changes if we have a write access, etc. Since the latter already exists in TheIDe in some form, however rudimentary, all that has to be done is to polish the existing svn/git integration and then write a good repo setup dialog (next->next->ok) which will present the user with options to set up the repo as assembly, nest or package, and which can serve to both new and experienced U++ users.

P.s. I am not very happy about the overall file structure of upp-components. I am going to overhaul and polish it with -hopefully- Upp's 2020.2 release (and sync with it thereafter, using only a dev branch to add new stuff in the meanwhile), Still I think agregating packages in a repo is better for me than having a repo for each package (given that I am going to publish new packages and some other tools/apps in the near future)

Best regards.
Oblivion

---

Subject: Re: [Proposition] Simply source package manager for Upp
Posted by mirek on Tue, 16 Jun 2020 15:27:21 GMT
View Forum Message <> Reply to Message

I agree with general concept, not so sure about details... I think it should be a bit more like Oblivion's suggestion. That is, keep basic structure unchanged and add another level of metainformation to download dependencies.

Also, being there, it would be nice to have some sort of "distributed protocol" to have a list of downloadable "remotes" in theide.

Either way, I am adding this to a long list of things I am thinking about

Mirek

---

Subject: Re: [Proposition] Simply source package manager for Upp
Posted by mirek on Wed, 19 Aug 2020 10:38:39 GMT
View Forum Message <> Reply to Message

Some thoughts:

- the "remote unit" should be nest. It is more general, often the code will comprise of more packages anyway.
- Package names should be unique in "uppiverse". Here I was considering that maybe we should introduce some UUID to resolve name clashes, in the end I have found that impractical
- I think the whole system should work by defining a special file, e.g. "remote" or "uppiverse" in the nest root which would contain references to other repositories. When loading main package, if package is missing, it would by automatically resolved by following these links. Repo would be downloaded into some "uppiverse" folder of nests and all nests of "uppiverse" would be automatically added at the end of assembly.
- Also, this establishes "distributed" network of packages, as crawling through these files should identify all nests/packages available, so some sort of "package manager" service would be possible (probably something like "add uppiverse package").
- One problem I see is that git does not provide functions to read individual files / folder from remote repo without cloning it. I think that this could be solved by limiting repo to github, where there is AFAIK API to get this info. We need this to list info about library and to get list of packages...

Mirek