hi,

I read the solution for using the images multiple cpp files ..

it says,

using in header:

#define IMAGECLASS NAME
#define IMAGEFILE <pathtofile/images.iml>
#include <Draw/iml_header.h>

and in sources:

#define IMAGECLASS NAME
#define IMAGEFILE <pathtofile/images.iml>
#include <Draw/iml_source.h>

I created one source for my images which I want to use in multiple header and sources. So I put the code for header and for the sources in files, which I include in my other files

#include "images-header.h"

in code1.h and code2.h

#include "images-sources.h"

in code1.cpp and code2.cpp

There were no problems with the compiler, but the linker stopped with errors ... "class NAME" redefined ...

is there a solution to use it in multiple files? It would help to keep the files smaller.

regards

reinhard

Subject: Re: ImageFile
Posted by mirek on Sun, 02 Nov 2008 15:44:35 GMT

View Forum Message <> Reply to Message

sapiency wrote on Sun, 02 November 2008 08:55hi,

I read the solution for using the images multiple cpp files ..

it says,

using in header:

#define IMAGECLASS NAME
#define IMAGEFILE <pathtofile/images.iml>
#include <Draw/iml_header.h>

and in sources:

#define IMAGECLASS NAME
#define IMAGEFILE <pathtofile/images.iml>
#include <Draw/iml_source.h>

I created one source for my images which I want to use in multiple header and sources. So I put
the code for header and for the sources in files, which I include in my other files

#include "images-header.h"

in code1.h and code2.h

#include "images-sources.h"

in code1.cpp and code2.cpp

There were no problems with the compiler, but the linker stopped with errors ... "class NAME"
redefined ...

is there a solution to use it in multiple files? It would help to keep the files smaller.

regards

reinhard

Oh, not that complicated. Just put the

#define IMAGECLASS NAME
#define IMAGEFILE <pathtofile/images.iml>
#include <Draw/iml_header.h>

into some header that you include in all files that are using images (this is same as declaring function or class in header) and then put

#define IMAGECLASS NAME
#define IMAGEFILE <pathtofile/images.iml>
#include <Draw/iml_source.h>

into the *ANY* .cpp, but only ONE, that includes the header of the previous step. This is equivalent of defining function.

If you put it to more than single file, it is like defining function twice - that does not work either

Mirek

Subject: Re: ImageFile
Posted by sapiency on Sun, 02 Nov 2008 19:14:47 GMT
View Forum Message <> Reply to Message

Hi Mirek,

thanks for you answer, I'm still a little bit confused , but I think I solved the problem ...

I have following files:
image-header.h (with #ifndef)
image-source.h (with #ifndef)
code_1.h (with #ifndef)
code_2.h (with #ifndef)
code_1.cpp
code_2.cpp

this works:
code_1.h includes image-header.h
code_1.cpp includes code_1.h, image-source.h
code_1.cpp use images

code_2.h includes code_1.h

code_2.cpp includes code_2.h
code_2.cpp use images

whenever I include image-source.h too in code_2.cpp, I get following message from the linker:

C:/upp/out/path/MINGW.Debug.Debug_full.Gui.Main.Mt\devicetree.o: In function
`_ZN11OL_Controls3ImlEv':
C:/upp/uppsrc/Draw/iml_source.h:33: multiple definition of `OL_Controls::Iml()'
C:/upp/out/path/MINGW.Debug.Debug_full.Gui.Main.Mt\$blitz.o:C:/upp/uppsrc/Draw/iml_source.h:
33: first defined here

object-file: devicetree -> code_2

regards

reinhard

Subject: Re: ImageFile
Posted by mirek on Sun, 02 Nov 2008 20:46:17 GMT
View Forum Message <> Reply to Message

sapiency wrote on Sun, 02 November 2008 14:14Hi Mirek,

thanks for you answer, I'm still a little bit confused , but I think I solved the problem ...

I have following files:
image-header.h (with #ifndef)
image-source.h (with #ifndef)
code_1.h (with #ifndef)
code_2.h (with #ifndef)
code_1.cpp
code_2.cpp

this works:
code_1.h includes image-header.h
code_1.cpp includes code_1.h, image-source.h
code_1.cpp use images

code_2.h includes code_1.h
code_2.cpp includes code_2.h
code_2.cpp use images

whenever I include image-source.h too in code_2.cpp, I get following message from the linker:

C:/upp/out/path/MINGW.Debug.Debug_full.Gui.Main.Mt\devicetree.o: In function

`_ZN11OL_Controls3ImlEv':
C:/upp/uppsrc/Draw/iml_source.h:33: multiple definition of `OL_Controls::Iml()'
C:/upp/out/path/MINGW.Debug.Debug_full.Gui.Main.Mt\$blitz.o:C:/upp/uppsrc/Draw/iml_source.h:
33: first defined here

object-file: devicetree -> code_2

regards

reinhard

Ehm, I am totally confused by number of your headers In any case, the error is because there is
more that one .cpp file that directly or indirectly includes

#define IMAGECLASS NAME
#define IMAGEFILE <pathtofile/images.iml>
#include <Draw/iml_source.h>

Usually, and I would recommend that, when developing U++ app, we are using single "main"
header per package and all C++ files include this one.

In that case, put those .iml lines with "iml_header.h" to this header and "iml_source.h" to any
single .cpp.

Anyway, what you have done works too, of course. All you need to remember is that
"iml_source.h" include is equivalent of defining all iml elements. And there is "one definition rule"
in C++ - no function or variable can be defined twice (or linker error).

Just one more note - if you are doing things wrongly, it sometimes might lead to weird problems
when you compile in debug mode (it compiles sometimes, sometimes not). This is caused by
BLITZ compile accelerator, which sometimes hides wrong arrangement of headers.

Mirek

Mirek

Subject: Re: ImageFile
Posted by sapiency on Sun, 02 Nov 2008 22:29:11 GMT
View Forum Message <> Reply to Message

luzr wrote on Sun, 02 November 2008 21:46
Usually, and I would recommend that, when developing U++ app, we are using single "main"

---

header per package and all C++ files include this one.

I believe you but the project grows and it takes really long (on my machine) to compile everytime the hole file - and I don't like large files .

luzr wrote on Sun, 02 November 2008 21:46
In that case, put those .iml lines with "iml_header.h" to this header and "iml_source.h" to any single .cpp.

Anyway, what you have done works too, of course. All you need to remember is that "iml_source.h" include is equivalent of defining all iml elements. And there is "one definition rule" in C++ - no function or variable can be defined twice (or linker error).

Ok, now I understand the description in the docu

luzr wrote on Sun, 02 November 2008 21:46

Just one more note - if you are doing things wrongly, it sometimes might lead to weird problems when you compile in debug mode (it compiles sometimes, sometimes not). This is caused by BLITZ compile accelerator, which sometimes hides wrong arrangement of headers.

Mirek

Thanks.

regards

reinhard

Subject: Re: ImageFile
Posted by mirek on Mon, 03 Nov 2008 07:47:24 GMT
View Forum Message <> Reply to Message

sapiency wrote on Sun, 02 November 2008 17:29luzr wrote on Sun, 02 November 2008 21:46 Usually, and I would recommend that, when developing U++ app, we are using single "main" header per package and all C++ files include this one.

I believe you but the project grows and it takes really long (on my machine) to compile everytime the hole file - and I don't like large files .

Well, with BLITZ, the compile time influence of single big header is almost irrelevant.

Second argument (sometimes the file is too large to one's good taste) is valid - in that case, I tend to split to many smaller header files, but to combine them in the main header anyway, then

include just this single header in .cpp. See e.g. CtrlLib/CtrlLib.h.

Mirek

Subject: Re: ImageFile
Posted by sapiency on Mon, 03 Nov 2008 19:30:49 GMT
View Forum Message <> Reply to Message

thanks