Notes from trying to build the Revaluator XPCOM Component

Note:
Apparently all code compiled to use XPCOM headers/functionality needs to include xpcom-config.h (according to https://developer.mozilla.org/en/xpcom_glue)
We were not doing this for Revaluator, but I have added it to RXPCOM.h

Note:
*** I think it may be a problem that we're using -DMOZILLA_INTERNAL_API, but currently we can't make Revaluator.o without it
"
Mozilla internal code defines MOZILLA_INTERNAL_API while compiling and links against xpcom.lib and xpcom_core.lib. In almost all cases embedders should *not* use internal linkage. Components using internal linkage will have shared-library dependencies against non-frozen symbols in the XPCOM libraries, and will not work with any other versions of XPCOM other than the one it was compiled against.

Internal linkage will be unavailable to extension authors in XULRunner 1.9 (Firefox 3) because the nonfrozen symbols will not be exported from libxul. Extension and application authors currently using internal linkage should read the guide on Migrating from Internal Linkage to Frozen Linkage."

Note:
From the above, it looks like if we need to use MOZILLA_INTERNAL_API then we should be linking against libxpcom.so and something called xpcom_core.lib, NOT libxpcomglue_s

This is an issue because libxpcom_core.a is NOT in the sdk!

Note:
After making this change, it still isn't linking. In fact, it still claims that NS_GetServiceManager is undefined, as well as other symbols (which I don't understand because NS_GetServiceManager IS defined in libxpcom.so).

At least some of the symbols are in libxul, so I'll try linking against that.

This did not fix the issue. Going to go back and try something simpler.

Note:
***So in Revaluator.cpp, we actually undefine then reprototype NS_GetServiceManager.m
Could this be causing the problems with it being undefined during the

linking phase?

Removing the undef/reprototype code from Revaluator.cpp causes the linker to stop complaining about NS_GetServiceManager being undefined.

However, now the problem is that NS_GetServiceManager_P is undefined, but for some reason is included in Revaluator.o as an undefined symbol.

NS_GetServiceManager_P seems to be the Internal (non-frozen) API version of the frozen method NS_GetServiceManager.

Because we are using MOZILLA_INTERNAL_API, all the frozen symbols get redefined to their non-frozen internal counterparts, eg:

```
  #define NS_GetServiceManager NS_GetServiceManager_P
```

The only place in xulrunner-sdk/lib that a symbol called NS_GetServiceManager_P is actually defined is in libxul.so

This is because they stopped exporting the non-frozen symbols from libxul to libxpcom after 1.9 (see XPCOM GLUE page).

I'm not sure how Duncan convinced this to compile before but to do it the Right Way (and possibly just to do it at all) we need to choose between the XPCOM GLUE route (strongly recommended by MDC) _OR_ the MOZILLA_INTERNAL_API route. Currently we are trying to mix them and I think this is causing a lot of our problems (or even if not, contributing significantly to the confusion that IS causing them)

Note:
****
I got it to sucessfully link by linking against ONLY the .o files we generate and libxul.so

This is dangerous and definitely not best practice, but it is somewhere to move forward from. I think our code will need to be linked against the exact version of libxul that Firefox is using to get it to load properly (or possibly at all).

Note:
Currently the test pages don't work. The component is not getting registered properly. This is not that surprising, though, because I had to link against libxul, which I don't think extensions are supposed to do.

Note:
I was able to build everything in inst/components using the non -fPIC'ed xulrunner build.

I think it was the Installation of the R Package that gives us trouble without -fPIC.

Nope, the R Package will install fine using the non -fPICed sdk. Whatever was causing the problems we were seeing was a symptom of something else.

I'm going to recompile xulrunner without -fPIC again just to make sure the buildI THINK didn't have -fPIC actually didn't have -fPIC

Yep, it compiles, installs, and loads fine. Its not clear that my earlier attempts to add -fPIC to the CPPFLAGS for the mozilla build process actually did anything.

Note:
following the suggestions for linux at:
http://developer.mozilla.org/en/Troubleshooting_XPCOM_components_registration

I ran :

```
./run-mozilla.sh `which ldd` -r
/home/gmbecker/gabe/checkedout/RFirefox/inst/components/Revaluator.so
```

and found that even though it links properly, some of the symbols are undefined eg:

```
undefined symbol:
_ZN13nsCOMPtr_base18assign_from_helperERK15nsCOMPtr_helperRK4nsID

(/home/gmbecker/gabe/checkedout/RFirefox/inst/components/Revaluator.so)
undefined symbol: _ZN13nsCOMPtr_base16begin_assignmentEv

(/home/gmbecker/gabe/checkedout/RFirefox/inst/components/Revaluator.so)
```

libxul.so in the copy of the xulrunner 1.9.2 sdk that I built myself contains this symbol with a lower case t next to it.

I'm going to change back to linking against this copy.

Note:
When I add the -Wl,-z,defs flag to the linking command, the linking fails, complaining about undefined references.

Note:
There are multiple references all around the web, eg:

http://forums.mozillazine.org/viewtopic.php?f=19&t=958465
http://benjamin.smedbergs.us/blog/2008-09-22/when-linking-the-order-of-your-command-line-can-be-important/

https://developer.mozilla.org/en/xpcom_glue

Which reference the exact order of terms in the linker call being very important. One of them even references the exact problem we're seeing as being able to be resolved with correct linker call order (though they are in the xpcom glue case, not MOZILLA_INTERNAL_API). I have fiddled with the linker call a bit but either I haven't hit on exactly the right order yet, or we are having some other, related problem.

Note:
I think I'm going to try to get it to link properly in 1.9.1 because that was before they stopped exporting unfrozen symbols from libxul. Because of this I should be able to just follow the linking directions on the xpcom_glue page for Internal API linking

Note:
Version 191 seems to exhibit the same behavior, complaining about kPStaticModules and kStaticModuleCount being undefined if libxul is not linked against, but complianing about undefined references.

I looked at the xpcom_glue page on MDC again and it turns out 191 also doesn't export the symbols from libxul (and so has the same problem). The change was between versions 1.8.* and 1.9.*

Note:
it seems like the way to go is to figure out how the cpp code for the python support (pyXPCOM and pyXPCOM Extension, possibly also pyDOM?) is built and linked, as that is internal code.

Note:
this seems like it may be helpful:
https://developer.mozilla.org/en/Adding_XPCOM_components_to_Mozilla_build_system

Note:
I think we should probably be modeling our build sequence after the xpcom components that ship within the mozilla system, since we are using the internal API.

I'm going to go dig around a bit in the log file I created during the build process (its not very new but that shouldn't matter).

Note:
I noticed in makefile.in for the autocomplete xpcom component they defined LIB_XUL=1, I think its possible we need to do that since we are having to link against libxul

Note:

For some reason version 1.9.1 has the python extension ship with it, but the 1.9.2 version does not. I'm building 1.9.1 and shoving the output into a log file so I can look at what python is doing.

Note:
I am having trouble finding where in the log the python extension is built from cpp code. The obvious things such as searching for python and pyGBase do not work.

It looks like it doesn't actually get built by default when you build mozilla/firefox. We need to add --enable-extensions=python\xpcom

BUT the extension doesn't compile properly, instead erroring out of the build process!!

```
../../../../_javagen/default/org/mozilla/interfaces/nsIPythonTestInte
rface.java:21: integer number too large: 4294967295
  int BiggerLong = 4294967295;
```

Note:
So I finally got the python XPCOM extension that shipped with the source code for 1.9.1 (but NOT 1.9.2) to build properly during the mozilla build process by disabling javaXPCOM (the error that I referenced last time was actually in java code, NOT cpp code so adding
ac_add_option --disable-javaxpcom to my .mozconfig file prevented it from attempting to compile the broken code

I have saved the relevant parts of the build cycle to pythonbuild.log

Note:
Some quick observations about what I see before I try to replicate it:
They do NOT define MOZILLA_INTERNAL_API, even though they do use variants ...

Actually they only use nsIVariant.h, NOT nsVariant.h. They then go on to use nsIVariants without having to define MOZILLA_INTERNAL_API. Can we do this???

Note:
So when I changed from including nsVariant.h to nsIVariant.h, I suddenly can't build my .o files because malloc is undefined. There is /a/ version of malloc in stdlib.h/cstdlib.h, though I don't know if its the right one. I will try including that.

Note:
when I included stdlib.h to define malloc it then complains about the following error for RVariants.cpp (the other two build into .o

without complaining):

```
In file included from /home/gmbecker/gabe/checkedout/FirefoxSource192/obj-
xulrunner/dist/xulrunner-sdk/include/nsStringGlue.h:52,
                 from /home/gmbecker/gabe/checkedout/FirefoxSource192/obj-
xulrunner/dist/xulrunner-sdk/include/nsIAtom.h:17,
                 from /home/gmbecker/gabe/checkedout/FirefoxSource192/obj-
xulrunner/dist/xulrunner-sdk/include/nsINodeInfo.h:59,
                 from /home/gmbecker/gabe/checkedout/FirefoxSource192/obj-
xulrunner/dist/xulrunner-sdk/include/nsINode.h:45,
                 from /home/gmbecker/gabe/checkedout/FirefoxSource192/obj-
xulrunner/dist/xulrunner-sdk/include/nsIDocument.h:40,
                 from RVariants.cpp:5:
/home/gmbecker/gabe/checkedout/FirefoxSource192/obj-xulrunner/dist/xulrunner-
sdk/include/nsStringAPI.h:1058: error: size of array 'arg' is negative
```

Note:
According to
http://markmail.org/message/hgwyda3oruvorubl#query:related
%3Ahgwyda3oruvorubl+page:1+mid:q4oqq22riczunmbv+state:results

A solution to this is to add -fshort-wchar to the compile line. I will try this.

Note:
Adding -fshort-wchar to CPPFLAGS causes all 3 .o files to build, though RVariants.o has multiple warnings about visibility:
eg:
warning: 'nsCOMArray<nsINode>' declared with greater visibility than its base 'nsCOMArray_base'

Note:
Everything builds properly now following the instructions for dependent glue!!!

Now I just need to see if it will actually load into firefox.

Note:
It loads into firefox and the xpcom component is registered, but firefox routinely crashes to desktop on the following javascript code from within the go function in testR.js

```
        var args = ["R"];
        obj.init(args, 1);
```

The crash is of the most unhelpful kind, no uncaught exceptions, no core dump, it firefox just disappears.