User Interface - Feature #4179

create a standalone native launcher (Windows EXE and Linux/UNIX version) which can be registered as a browser's application helper for the "application/fwd-local-launcher" mime type

08/08/2019 02:15 PM - Greg Shah

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Roger Borrello	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	

Description

History

#1 - 08/08/2019 02:22 PM - Greg Shah

We've encountered multiple client projects in which there is a need to open a specific document/resource or otherwise launch a local program. In the web client this is executed on the server where the FWD client JVM runs. Often, this requirement is needed on the system on which the user's browser is executing. To solve this will require some native code.

This task is to create a standalone native launcher (Windows EXE and Linux/UNIX version) which can be registered as a browser's application helper for the "application/fwd-local-launcher" mime type. The Windows version would use ShellExecuteA.

Then the 4GL code can invoke OPEN-MIME-RESOURCE with "application/fwd-local-launcher" as the mime type and the resource will be a filename or command line to execute. The result is the execution of a local program with local data access.

This approach may be required to meet application needs in the converted environment. **The technique is dangerous because anyone that is able to modify the resource name/command line will be able to access/run programs on the user's system.** It is critical that customers know this before they use this feature. Try to avoid its usage unless there is no alternative.

#2 - 02/10/2020 04:14 PM - Roger Borrello

What is the development environment to use on Windows?

#3 - 02/10/2020 04:23 PM - Greg Shah

There are no conversion aspects to this task.

The WIN32 development environment would be the same one we use for all FWD WIN32 development.

#4 - 02/18/2020 02:21 PM - Roger Borrello

Greg Shah wrote:

> There are no conversion aspects to this task.
>
> The WIN32 development environment would be the same one we use for all FWD WIN32 development.

On what hardware/virtual environment?

#5 - 02/18/2020 02:23 PM - Greg Shah

For Windows, we only support Intel architecture 32-bit and 64-bit.

But: you don't need to work on this right now.

#7 - 02/02/2021 09:46 AM - Greg Shah

- Assignee set to Roger Borrello

- Start date deleted (08/08/2019)

#8 - 02/19/2021 07:38 AM - Greg Shah

Your testcase to run this can be as simple as this:

```
def var launch-url as char format "x(128)".
```

```
update launch-url.
```

```
open-mime-resource "application/fwd-local-launcher" launch-url.  /* not sure if NOT-EMBEDDED is needed here */
```

#9 - 02/20/2021 11:44 AM - Roger Borrello

How should the path to the local resource be formatted? Is it up to the user to enter **file:**/// or should any valid path be acceptable? For example, when I enter /home/rfb/Pictures/Faceshot.png, when UiUtils.resolveResourceUrl gets to:

```
         URL url = new URL(urlString);
```

A MalformedURLException is thrown. If the use case uses a file browser, we'd not be able to rely upon that to add the prefix.

#10 - 02/21/2021 07:09 PM - Greg Shah

> UiUtils.resolveResourceUrl

As of now, file:// resources are local to the FWD client. We will have to format a URL as needed to be passed to te client. The launcher will primarily need to open local files on the browser system. On Windows this will be whatever format is needed by ShellExecuteA. Format a protocol specifier that works. We can always change OPEN-MIME-RESOURCE as needed, but note that we must maintain the current functionality of loading files from the FWD client.

#11 - 02/22/2021 06:52 PM - Roger Borrello

So the tc.openMimeResource should make a special case of mimeType=application/fwd-local-launcher and force the given urlString to include the FILE_PROTOCOL_PREFIX if it isn't already there?

Are there already specs for whether or not parameters are to be passed to the launcher? Or if there is just a resource to open, and it would handle the rest?

#12 - 02/23/2021 07:33 AM - Greg Shah

> So the tc.openMimeResource should make a special case of mimeType=application/fwd-local-launcher

Yes, if needed. It is a custom mime type so we can do whatever is needed.

> and force the given urlString to include the FILE_PROTOCOL_PREFIX if it isn't already there?

If that is needed. But we would not want to try to open the local FWD client file system resource in this case, right?

> Are there already specs for whether or not parameters are to be passed to the launcher?

URIs can include encoded query parameters. This is part of the RFCs/W3C standards and/or best practices. We could use that approach if the resource being specified requires something like that.

You can look at the existing customer WIN32 code to see how ShellExecuteA is used.

#13 - 02/23/2021 08:56 AM - Roger Borrello

A question about implementation. Would the launcher only function on web client? What if you run OPEN-MIME-RESOURCE of this custom mime-type in ChUI or Swing clients? Should they reject?

#14 - 02/23/2021 09:23 AM - Greg Shah

> What if you run OPEN-MIME-RESOURCE of this custom mime-type in ChUI or Swing clients? Should they reject?

OPEN-MIME-RESOURCE is a GUI feature, supported in both web (GuiWebDriver.openMimeResource()) and Swing (SwingGuiDriver.openMimeResource() and SwingGuiDriver.openDocument).

ChUI is and can be ignored.

> Would the launcher only function on web client?

Yes and no. Technically, we only need a separate launcher executable on the web client, so the launcher itself is not strictly needed elsewhere. On the other hand, it may make sense for consistency to use the launcher there as well.

Since we support OPEN-MIME-RESOURCE on Swing, we need to ensure that we do something safe, as least. We need to decide if we implement the backing support for this application/fwd-local-launcher mime type or not. And if we support it, do we use the separate launcher or do we just include the equivalent functionality in libp2j. I lean toward YES and LIBP2J.

#15 - 02/24/2021 06:21 PM - Roger Borrello

- % Done changed from 0 to 20

First pass committed in 3821c_12057. There is an update to the native makefile to use the new makefile.launcher which builds src/native/fwd_local_launcher.c. This currently runs in Windows and Linux. The Linux implementation does not attempt to use an associated program. It just tries to run the program.

This doesn't really work if we are trying to utilize the default application. For example, if you run a plain text file, you'd want the default editor to open the file, like **Kate** or **xed** or **gedit**. Instead, it tries to run the file like a script. Some discussion needs to take place about this part of the design.

Also, in terms of parameters being passed, at the present time, whatever argv parameters are passed after the first one, which is the file to open, are simply passed on to that program. This would put the onus on the Java side to format the string properly.

#16 - 02/25/2021 09:08 AM - Roger Borrello

I removed the launcher target to prevent the continuation of the failures found in #5034-317, which was created by my incomplete modification of the makefile.launcher, created as a copy of makefile.spawn. It calls postbuild.sh, which positions the spawner in to the correct location using the srv-certs.store file. For now I'll just remove the use of that script until I understand what the deployment of fwd_local_launcher should be.

#17 - 02/25/2021 06:50 PM - Roger Borrello

I made updates in revs 12060/12061 to allow the build to proceed. How are the native parts deployed if you are simply using a web client? We just place the fwd_local_launcher.exe somewhere in the PATH?

#18 - 02/26/2021 06:21 AM - Greg Shah

In #4179-1 is this requirement: **which can be registered as a browser's application helper for the "application/fwd-local-launcher" mime type**.

This will be browser-specific. Please document the process for Chrome and Firefox.

#19 - 03/01/2021 07:52 PM - Roger Borrello

Regarding Linux, I came across xdg-open, which is part of xdg-utils:

https://wiki.archlinux.org/index.php/Xdg-utils
https://linux.die.net/man/1/xdg-open
http://manpages.ubuntu.com/manpages/hirsute/en/man1/xdg-open.1.html

#20 - 03/01/2021 08:08 PM - Roger Borrello

Greg Shah wrote:

> In #4179-1 is this requirement: **which can be registered as a browser's application helper for the "application/fwd-local-launcher" mime type**.
>
> This will be browser-specific. Please document the process for Chrome and Firefox.

Firefox

1. Quit Firefox completely.
2. BACKUP mimeTypes.rdf in your Firefox's **profile** folder
3. Open mimeTypes.rdf in your Firefox's profile folder by text editor
4. Find line of

```
<RDF:Seq RDF:about="urn:mimetypes:root">
```

5. Insert the following code AFTER line which is found in step 4.

```
<RDF:li RDF:resource="urn:mimetype:application/fwd-local-launcher"/>
```

6. Find line of

```
<RDF:Seq RDF:about="urn:mimetypes:root">
```

7. Insert the following code BEFORE line which is found in step 6.

```
  <RDF:Description RDF:about="urn:mimetype:application/fwd-local-launcher"
                   NC:fileExtensions=""
                   NC:description="rar Archive"
                   NC:value="application/fwd-local-launcher"
                   NC:editable="true">
    <NC:handlerProp RDF:resource="urn:mimetype:handler:application/fwd-local-launcher"/>
  </RDF:Description>
  <RDF:Description RDF:about="urn:mimetype:handler:application/fwd-local-launcher"
                   NC:alwaysAsk="true">
  </RDF:Description>
```

8. Save mimeTypes.rdf (should not change character encoding)
9. Restart Firefox
Ref: http://kb.mozillazine.org/MimeTypes.rdf

#21 - 03/05/2021 01:54 AM - Roger Borrello

I did see information indicating that Chrome uses whatever the desktop's mime database contains. I don't know if there is a common mime database across all the Linux environments we want to support. The below 2 references seem to describe 2 different methods. I see the first link's information on my system, but after using xfce4-mime-settings, it is unclear how to add our mime type. The 2nd shows how to update the mime types, but it doesn't seem to be how my system is configured.

https://unix.stackexchange.com/questions/507943/config-files-for-mime-settings
https://www.freedesktop.org/wiki/Specifications/shared-mime-info-spec/

#22 - 03/06/2021 10:56 AM - Roger Borrello

In trying to understand how to register an application helper for our mime-type helper, I was confused by whether or not we need to register a custom protocol handler for each browser type, or if the file:// prefix would do all we need. Is that part of development handled by other tasks, while this task is purely a wrapper for ShellExecuteA and xdg-open? I don't want to move too much into this, if not necessary.

Right now the fwd_local_launcher and fwd_local_launcher.exe will receive commands to execute and open them in the default application configured in the OS will handle them from there. For example, fwd_local_launcher /home/rfb/Pictures/Faceshot.png will launch Xviewer with the image in it on my system.

Ref:
https://developer.mozilla.org/en-US/docs/Web/API/Navigator/registerProtocolHandler
https://developers.google.com/web/updates/2011/06/Registering-a-custom-protocol-handler

#23 - 03/06/2021 11:30 AM - Roger Borrello

Committed fwd_local_launcher.c in 3821c_12090 which utilizes xdg-open.

While the code can accept parameters, the use of parameters is a little contrary to the concept of launching a file, and allowing the configured program to handle opening it. If I've missed the mark, please let me know, because there is a conceptual difference between running a program and opening a file using the default program type.

#24 - 03/06/2021 11:33 AM - Greg Shah

I'm not sure what you mean by "mime-type helper". Are you referring to the launcher that will be registered as an application helper? Or are you talking about OPEN-MIME-RESOURCE which is 4GL enhanced syntax for loading a URI into the browser?

> I was confused by whether or not we need to register a custom protocol handler for each browser type, or if the file:// prefix would do all we need.

It is part of this task to answer this question. You need to tell us the answer. Can a customer do what they need to do with just the file:// protocol or do we need something custom like Sergey noted in #4608-75 (e.g. add a launcher:// protocol). As I noted above (#4179-10):

> We can always change OPEN-MIME-RESOURCE as needed, but note that we must maintain the current functionality of loading files from the FWD client.

Even using file:// there will be more changes needed.

> Is that part of development handled by other tasks, while this task is purely a wrapper for ShellExecuteA and xdg-open? I don't want to move too much into this, if not necessary.

Without these other changes, this launcher is useless. So it really is part of this task.

#25 - 03/06/2021 12:54 PM - Greg Shah

Code Review Task Branch 3821c Revision 12090

It is a good start.

1. As mentioned in #4179-14, we need to assume that the core launching capability is in libp2j.so/p2j.dll. This will give us the ability to support this launching seamlessly even in Swing. I don't want the launcher to require the library to be installed. Rather, I'm thinking the same module linked into the library AND statically linked into the launcher.

- This will be a new module called "desktop" whose source code is in desktop*.[ch].
- Implement this in the same pattern as the rest of our native code.
  - The language independent JNI API is in desktop.c. Do not have any JNI functionality outside of this file. This will be the file used for the native methods in FWD.
  - The desktop.h will have the language-indepenent C function definitions of the helper functions.
  - The core launching will be in platform-specific files.
    - The desktop_linux.c will implement the Linux versions of the helper functions.
    - The desktop_win.c will implement the Windows versions of the helper functions.
    - Move the associated logic from the fwd_local_launcher.c. fwd_local_launcher.c should only be generic C code, no platform specific code should be in it.
  - Rework the build to statically link the correct module into fwd_local_launcher.c and call the platform independent C API (as exposed by desktop.h).
- So that the code can be used by the FWD client, you will need to add a Java class with the native methods.
- Please see src/native/shell*.[ch] and src/com/goldencode/terminal/NativePty.java for an example.

2. Do not use stdio for any error messages. Any errors should be reported as a return code. Please create our own platform-independent list of return codes (and constants for them in desktop.h). For local testing, it is OK to create a helper to print the decoded return code to stdio. This could be enabled by a command line option.

3. Please clean up the formatting to match our code standards. For example, you are using tabs everywhere which is evil. :)

#26 - 03/09/2021 03:46 PM - Roger Borrello

Right now the JNI is in com.goldencode.p2j.util and the method is Java_com_goldencode_p2j_util_shellOpenFile. Is that appropriate?

#27 - 03/09/2021 04:27 PM - Greg Shah

I think this code is very GUI specific. Please put it in com.goldencode.p2j.ui.client.gui and change the shellOpenFile() to desktopOpen().

We already have a JNI module for shell (e.g. bash or cmd.exe) and I don't want this to be confused. Also, I'm not sure that it will only be about opening file:/// resources so, the open should be more general.

#28 - 03/09/2021 05:59 PM - Roger Borrello

Greg Shah wrote:

> I think this code is very GUI specific. Please put it in com.goldencode.p2j.ui.client.gui and change the shellOpenFile() to desktopOpen().

Will the interface be src/com/goldencode/p2j/ui/client/gui/Desktop.java?

> We already have a JNI module for shell (e.g. bash or cmd.exe) and I don't want this to be confused. Also, I'm not sure that it will only be about opening file:/// resources so, the open should be more general.

Just for some background, in the customer's 4GL, the procedure determines if the if the URL/URI/path is a directory, and if so opens the explorer.exe against it. If it's not a directory, and does **not** contain "http" or "www" and doesn't have an extension from a know list of web site domains (.com, .biz, .mobi, etc), it is passed to ShellExecuteA with the "open" option. No parameters are utilized. The net results is whatever the OS has configured for whatever is passed to it will handle the execution. If it's a ".doc", ".xls", ".ppt", or whichever, the application assigned will take over.

In this task, we are being more generic. We want to allow OPEN-MIME-RESOURCE to take a URI, and break it down into parts to be "launched". If the protocol portion of the URI is launcher: or similar, we break up the rest into program and parameters. If the protocol portion is file:// we are going to "open" it. Do we wait for completion?

#29 - 03/09/2021 08:24 PM - Greg Shah

> and break it down into parts to be "launched".

I don't think this is correct. OPEN-MIME-RESOURCE just sends the URI to the browser. It does not parse the URI itself. If we are using a launcher:// protocol, then we don't even need any changes in OPEN-MIME-RESOURCE because it can still load file:/// from the FWD client's file system and send that file down to the browser instead of the URI.

> launcher: or similar, we break up the rest into program and parameters

Not sure about "or similar", but I don't think OPEN-MIME-RESOURCE needs to parse this at all. The browser is supposed to handle the URI.

Consider this: OPEN-MIME-RESOURCE is not a requirement to use the launcher. The URI can be sent to the browser in any way possible, including in static HTML. The only reason OPEN-MIME-RESOURCE is being discussed here is so that we make sure that it works properly since many customers will in fact make 4GL code changes to use it. The 4GL doesn't have such a feature since they have no web client.

> If the protocol portion is file:// we are going to "open" it.

Forget about this case. As noted above, we can leave that as is, which means it is a file from the FWD client's system.

> Do we wait for completion?

This is not something you have to consider. Leave OPEN-MIME-RESOURCE alone from this perspective.

#30 - 03/09/2021 10:18 PM - Roger Borrello

- Status changed from New to WIP

- % Done changed from 20 to 30

#31 - 03/11/2021 06:21 AM - Roger Borrello

With respect to the JNI and the makefile, the JNIHEADER macro includes generated headers. How are those generated, and how do you include a net-new module in the makefile, if you haven't built yet (since they are generated, not checked in)?

In other words, $(SRCDIR)/com_goldencode_p2j_ui_client_gui_Desktop.h can't be added until it is created.

#32 - 03/11/2021 07:23 AM - Greg Shah

Search for javah in build.xml.

#33 - 03/11/2021 09:35 AM - Roger Borrello

- % Done changed from 30 to 40

Please review rev 12111. Right now the fwd_local_launcher.c is a little bit "hokey", but until I get the connective tissue in place, I'm not going to spend too much time on it.

#34 - 03/11/2021 10:35 AM - Greg Shah

Code Review Task Branch 3821c Revision 12111

This is much better.

1. I don't think that desktop.h should have #include <jni.h>. That pulls JNI into our "generic" API. JNI should be limited to desktop.c.

2. There are still hard tabs in desktop_linux.c and fwd_local_launcher.c.

3. In Desktop.java and desktop.c there is this description:

> Start an OS Shell session. On Linux/Unix xdg-utils are used to open the default associated program. On Windows, ShellExecuteA API is used.

Please change this to:

> Open the given resource or program. If the target is a file/resource then open it using the native graphical desktop API for "file associations". If the target is a program, launch it using traditional operating system program execution APIs.

Notice how this version is:

- avoiding use of the word "shell" which might be confused with other native services
- avoiding operating system specifics in the description of a generic API

4. In fwd_local_launcher.c, the following is very duplicative (in the switch statement):

sprintf(msg, "Error opening %s (%s)", argv[1], <string_literal>);

Please assign the <string_literal> to a local variable and then just have one instance of the sprintf() at the end of the switch.

#35 - 03/11/2021 06:58 PM - Roger Borrello

See 12113 for updates.

Also, it must have been some time since ant-native was built on 32-bit, because there were warnings/errors in memory.c and library.c:

```
ant-native:
     [echo]
     [exec] gcc -c C:\projects\fwd\3821c/src/native/memory.c -o memory.o -DWIN_TARGET -DWORD_SIZE_32 -IC:\jdk1
.8.0_281\jre/../include -IC:\jdk1.8.0_281\jre/../i
nclude/win32 -IC:\projects\fwd\3821c/src/native -Wall
     [exec] gcc -c C:\projects\fwd\3821c/src/native/terminal.c -o terminal.o -DWIN_TARGET -DWORD_SIZE_32 -IC:\
jdk1.8.0_281\jre/../include -IC:\jdk1.8.0_281\jre/
../include/win32 -IC:\projects\fwd\3821c/src/native -Wall
     [exec] C:\projects\fwd\3821c/src/native/memory.c: In function 'Java_com_goldencode_p2j_util_MemoryManager
_allocate':
     [exec] C:\projects\fwd\3821c/src/native/memory.c:154:28: warning: cast from pointer to integer of differe
nt size [-Wpointer-to-int-cast]
     [exec]   154 |    jlong result = (jlong) ((int32_t) addr);
     [exec]       |                            ^
     [exec] C:\projects\fwd\3821c/src/native/memory.c: In function 'Java_com_goldencode_p2j_util_MemoryManager
_deallocate':
     [exec] C:\projects\fwd\3821c/src/native/memory.c:179:9: warning: cast to pointer from integer of differen
t size [-Wint-to-pointer-cast]
     [exec]   179 |    free((void*) ((int32_t) addr));
     [exec]       |         ^
     [exec] C:\projects\fwd\3821c/src/native/memory.c: In function 'Java_com_goldencode_p2j_util_MemoryManager
_findNextNull':
     [exec] C:\projects\fwd\3821c/src/native/memory.c:216:18: warning: cast to pointer from integer of differe
nt size [-Wint-to-pointer-cast]
     [exec]   216 |    char* value = (char*) ((int32_t) addr);
     [exec]       |                  ^
     [exec] C:\projects\fwd\3821c/src/native/memory.c: In function 'Java_com_goldencode_p2j_util_MemoryManager
_read':
     [exec] C:\projects\fwd\3821c/src/native/memory.c:267:61: warning: cast to pointer from integer of differe
nt size [-Wint-to-pointer-cast]
     [exec]   267 |       (*env)->SetByteArrayRegion(env, data, 0, (jsize) len, (jbyte*) ptr);
     [exec]       |                                                             ^
     [exec] C:\projects\fwd\3821c/src/native/memory.c: In function 'Java_com_goldencode_p2j_util_MemoryManager
_write':
     [exec] C:\projects\fwd\3821c/src/native/memory.c:307:50: warning: cast to pointer from integer of differe
nt size [-Wint-to-pointer-cast]
     [exec]   307 |    (*env)->GetByteArrayRegion(env, data, 0, len, (jbyte*) ptr);
     [exec]       |                                                  ^
     [exec] C:\projects\fwd\3821c/src/native/memory.c: In function 'Java_com_goldencode_p2j_util_MemoryManager
_copy':
     [exec] C:\projects\fwd\3821c/src/native/memory.c:343:17: warning: cast to pointer from integer of differe
nt size [-Wint-to-pointer-cast]
     [exec]   343 |    jbyte* src = (jbyte*) ((int32_t) source);
     [exec]       |                 ^
     [exec] C:\projects\fwd\3821c/src/native/memory.c:344:17: warning: cast to pointer from integer of differe
nt size [-Wint-to-pointer-cast]
     [exec]   344 |    jbyte* dst = (jbyte*) ((int32_t) (dest + offset));
     [exec]       |                 ^
     [exec] gcc -c C:\projects\fwd\3821c/src/native/library.c -o library.o -DWIN_TARGET -DWORD_SIZE_32 -IC:\jd
k1.8.0_281\jre/../include -IC:\jdk1.8.0_281\jre/..
/include/win32 -IC:\projects\fwd\3821c/src/native -Wall
     [exec] C:\projects\fwd\3821c/src/native/library.c: In function 'calcCallingConvention':
     [exec] C:\projects\fwd\3821c/src/native/library.c:362:16: error: 'FFI_STDCALL' undeclared (first use in t
his function)
     [exec]   362 |          abi = FFI_STDCALL;
     [exec]       |                ^~~~~~~~~~~
     [exec] C:\projects\fwd\3821c/src/native/library.c:362:16: note: each undeclared identifier is reported on
ly once for each function it appears in
     [exec] C:\projects\fwd\3821c/src/native/library.c:366:16: error: 'FFI_MS_CDECL' undeclared (first use in 
this function)
     [exec]   366 |          abi = FFI_MS_CDECL;  // or should this be FFI_SYSV?
     [exec]       |                ^~~~~~~~~~~~
     [exec] C:\projects\fwd\3821c/src/native/library.c: In function 'jlongToPointer':
     [exec] C:\projects\fwd\3821c/src/native/library.c:440:16: warning: cast to pointer from integer of differ
ent size [-Wint-to-pointer-cast]
     [exec]   440 |    void* ptr = (void*) ((int32_t) addr);
     [exec]       |                ^
     [exec] make: *** [library.o] Error 1

BUILD FAILED
C:\projects\fwd\3821c\build.xml:1300: exec returned: 2
```

```
Total time: 19 seconds
```

#36 - 03/12/2021 07:44 AM - Greg Shah

Code Review Task Branch 3821c Revision 12113

The changes are good.

#37 - 03/12/2021 07:44 AM - Greg Shah

> because there were warnings/errors in memory.c and library.c

I assume you are fixing them.

#38 - 03/12/2021 07:49 AM - Greg Shah

At this point, we are going with our own launcher:// protocol for specifying the resource to load?

If so, then I think no changes are needed in OPEN-MIME-RESOURCE but we do need changes to parse that protocol before launch. This needs to be in the common code since it needs to be used for both the Swing client and for the launcher.

I guess it may be the last item needed to make this work. Do you know of anything else?

#39 - 03/12/2021 08:49 AM - Roger Borrello

Greg Shah wrote:

> I assume you are fixing them.

With respect to the Java header, which is what triggered a rebuild of everything, since each of them are listed as a dependency for the compile rule... I now have a file in the bzr tree that isn't checked in:

```
rfb@rfb:~/projects/fwd/p2jdev$ bzr status
unknown:
  src/native/com_goldencode_p2j_ui_client_gui_Desktop.h
```

I don't see any of the other src/native/com_goldencode_* files get listed in my bzr status, even though they are there. I don't see them when I issue bzr log -v:

```
rfb@rfb:~/projects/fwd/p2jdev$ bzr log -v src/native/com_goldencode_terminal_NativePty.h
bzr: ERROR: Path unknown at end or start of revision range: src/native/com_goldencode_terminal_NativePty.h
```

How does that happen? I don't want to accidentally commit it.

#40 - 03/12/2021 08:56 AM - Greg Shah

See .bzrignore.

#41 - 03/12/2021 11:52 AM - Roger Borrello

Greg Shah wrote:

> See .bzrignore.

Thanks.

With respect to library.c and the non-declaration, I'll have to double check the development environment on Windows, since I see some oddities. Even though I installed 64-bit mingw64, the uname -s command yields MINGW32_NT-6.2. The C:\mingw64 directory has x86_64-w64-mingw32 folder in it. It smells like part of my environment thinks this is 32-bit.

I guess something happens with the OS := $(shell uname -s) within make, since we do handle rules surrounded by the ifeq "$(OS)" "Windows". From the command line, uname -s shows MINGW32_NT-6.2.

Some of the directives follow the 64-bit paths. For example, the fftarget.h file in the listing output has these enumerations for ffi_abi:

```
# 63 "c:\\mingw64\\x86_64-w64-mingw32\\include\\ffitarget.h" 3
typedef unsigned long long ffi_arg;
typedef long long ffi_sarg;
# 78 "c:\\mingw64\\x86_64-w64-mingw32\\include\\ffitarget.h" 3
typedef enum ffi_abi {
  FFI_FIRST_ABI = 0,
# 98 "c:\\mingw64\\x86_64-w64-mingw32\\include\\ffitarget.h" 3
  FFI_WIN64,
  FFI_LAST_ABI,
  FFI_DEFAULT_ABI = FFI_WIN64
# 118 "c:\\mingw64\\x86_64-w64-mingw32\\include\\ffitarget.h" 3
} ffi_abi;
# 68 "c:\\mingw64\\x86_64-w64-mingw32\\include\\ffi.h" 2 3
```

That would only happen when X86_WIN64 is defined.

But the output for library.c contains:

```
# 346 "C:\\projects\\fwd\\3821c/src/native/library.c"
ffi_abi calcCallingConvention(jint conv)
{

   ffi_abi abi = FFI_DEFAULT_ABI;

      if (conv == 1L ||
          conv == 2L ||
          conv == 4L)
      {

         abi = FFI_STDCALL;
      }
      else if (conv == 3L)
      {
         abi = FFI_MS_CDECL;
      }

   return abi;
}
```

This would only happen if WIN_TARGET and WORD_SIZE_32 are defined. In makefile, we define here:

```
# final compile/link variable definitions
```

```
ifeq "$(ARCH)" "64bit"
   override COMPILECMD+=-DWORD_SIZE_64
else
   override COMPILECMD+=-DWORD_SIZE_32
endif
```

So something is wrong.

#42 - 03/12/2021 12:31 PM - Roger Borrello

It was msys. I'm not sure where that comes into play, but when I don't include c:\msys\bin in the path, everything builds.

#43 - 03/12/2021 03:49 PM - Roger Borrello

Greg Shah wrote:

> At this point, we are going with our own launcher:// protocol for specifying the resource to load?

Yes, although I am not sure how to get parameters to the program. If it's a file, that's pretty straightforward. It would be very similar to file://.

> If so, then I think no changes are needed in OPEN-MIME-RESOURCE but we do need changes to parse that protocol before launch. This needs to be in the common code since it needs to be used for both the Swing client and for the launcher.

I would think that UiUtils.resolveLocalFileResource() would be the first place. It throws a Malformed URLException when I pass in launcher:///home/rfb/Pictures/Faceshot.png when it creates a URL from the string. Should that be creating a URI?

> I guess it may be the last item needed to make this work. Do you know of anything else?

I'll find something I missed. Where does the protocol handler get added to the configuration? In Windows, there are registry updates.

#44 - 03/12/2021 05:07 PM - Roger Borrello

With respect to the OPEN-MIME-TYPE testcase and Swing, it also works "out of the box" using the file:// protocol, as long as you don't embed (include NOT-EMBEDDED). At that point, you have a web page that opens in the browser, and looks at you quietly:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8"/>
<title>/home/rfb/projects/VirtualBox-VMs/shared/projects/testcases/deploy/client/./c8f58507-3182-4852-950a-f2e
2fe2e0c98.html</title>
<style type="text/css">
.view {
position: relative;
width: 100%;
height: 100%;
margin: 0px;
background: white;
}
</style>
</head>
<body class="view">
<embed id="embeddedDocument" type="application/fwd-local-launcher"/>
</body>
<script type="text/javascript">
function onLoad()
{
   var config = {'isStreamed' : false };
   var el = document.getElementById("embeddedDocument");
   if (config.isStreamed)
   {
      el.src = "https://" + window.location.host + "${webRoot}/${documentHandler}/file:/tmp/fwd/copiedResource
1612920352337214616.png";
   }
   else
   {
      el.src = "file:/tmp/fwd/copiedResource1612920352337214616.png";
   }
   el.width  = window.screen.availWidth;
   el.height = window.screen.availHeight;
}

onLoad();
</script>
</html>
```

When you are not embedded, SwingGuiDriver.openDocument checks if (Desktop.isDesktopSupported()). At that point, the open method to java.awt.Desktop opens the file with the associated viewer, as defined in the OS.

So a testcase for the launcher that doesn't include open-mime-type would be just typing in launcher:///home/rfb/Pictures/Faceshot.png in the URL bar?

#45 - 03/12/2021 06:28 PM - Greg Shah

Roger Borrello wrote:

> Greg Shah wrote:
>
> > At this point, we are going with our own launcher:// protocol for specifying the resource to load?
>
> Yes, although I am not sure how to get parameters to the program. If it's a file, that's pretty straightforward. It would be very similar to file://.

The format is:

https://en.wikipedia.org/wiki/Query_string

The parameters would be parsed out from that format.

> > If so, then I think no changes are needed in OPEN-MIME-RESOURCE but we do need changes to parse that protocol before launch. This needs to be in the common code since it needs to be used for both the Swing client and for the launcher.
>
> I would think that UiUtils.resolveLocalFileResource() would be the first place. It throws a Malformed URLException when I pass in launcher:///home/rfb/Pictures/Faceshot.png when it creates a URL from the string. Should that be creating a URI?

Didn't Sergey already make that change?

#46 - 03/12/2021 06:50 PM - Greg Shah

> With respect to the OPEN-MIME-TYPE testcase and Swing, it also works "out of the box" using the file:// protocol

We don't want to use file:/// protocol in the launcher because this is always something local to the FWD client. The point to the launcher is to open something on the browser machine's file system. And we will want the Swing client to be able to load a URI with launcher:// protocol. We want the same 4GL code to work for both Swing and the web client.

But Swing is not the place to start your testing. Use the FWD web client please.

> So a testcase for the launcher that doesn't include open-mime-type would be just typing in launcher:///home/rfb/Pictures/Faceshot.png in the URL bar?

Yes, this will work too.

#47 - 03/14/2021 05:26 AM - Sergey Ivanovskiy

Roger Borrello wrote:

> Greg Shah wrote:
>
> > At this point, we are going with our own launcher:// protocol for specifying the resource to load?
>
> Yes, although I am not sure how to get parameters to the program. If it's a file, that's pretty straightforward. It would be very similar to file://.
>
> > If so, then I think no changes are needed in OPEN-MIME-RESOURCE but we do need changes to parse that protocol before launch. This needs to be in the common code since it needs to be used for both the Swing client and for the launcher.
>
> I would think that UiUtils.resolveLocalFileResource(String uriString) would be the first place. It throws a Malformed URLException when I pass in launcher:///home/rfb/Pictures/Faceshot.png when it creates a URL from the string. Should that be creating a URI?

No, this helper function is needed for local file resource encoded by file//: schema, and also it is safe to invoke this function for launcher://, because it doesn't throw malformed URL exception and it returns almost the same string for this case except that each appearance of the window path separator is replaced by the unix path separator.

#48 - 03/14/2021 06:16 AM - Sergey Ivanovskiy

Sergey Ivanovskiy wrote:

> UiUtils.resolveLocalFileResource(String uriString) doesn't throw malformed URL exception and it returns almost the same string for this case except that each appearance of the window path separator is replaced by the unix path separator.

I committed minor changes rev. 12120 (3821c) for UiUtils.resolveLocalFileResource(String uriString) that it doesn't change the provided uri string and fixed javadoc for WebBrowserManager.

#49 - 03/15/2021 08:18 AM - Roger Borrello

Sergey, take a look at this update I propose for UiUtils:

```
=== modified file 'src/com/goldencode/p2j/ui/client/UiUtils.java'
--- src/com/goldencode/p2j/ui/client/UiUtils.java    2021-03-14 10:06:12 +0000
+++ src/com/goldencode/p2j/ui/client/UiUtils.java    2021-03-15 12:14:58 +0000
@@ -87,6 +87,8 @@
 **     HC  20200726 Initial implementation of SPREADSHEET widget and related changes.
 ** 097 SBI 20210314 Renamed resolveResourceUrl to indicate its specific logic to resolve local files
 **                  and moved path separator replacement out of this method.
+**     RFB 20210312 Removed extra backslash from FILE_PROTOCOL_PREFIX and modified checks for "8" to
+**                  use the .length.
 */

 /*
@@ -167,7 +169,7 @@
 public class UiUtils
 {
    /** file protocol prefix to construct file related URL. */
-   public static final String FILE_PROTOCOL_PREFIX = "file:///";
+   public static final String FILE_PROTOCOL_PREFIX = "file://";

    /** Logger. */
    protected static final Logger LOG = LogHelper.getLogger(UiUtils.class.getName());
@@ -192,7 +194,8 @@
          // replace current or parent directory chars with absolute path to the working directory
          // but if the input URL string has something more than just prefix and has either
          // current or parent directory chars at least once
-         if (urlString.startsWith(FILE_PROTOCOL_PREFIX) && urlString.length() > 8 &&
+         if (urlString.startsWith(FILE_PROTOCOL_PREFIX) &&
+             urlString.length() > FILE_PROTOCOL_PREFIX.length() &&
              urlString.indexOf("./") != -1)
          {
             // get a candidate for canonicalize
@@ -232,7 +235,7 @@

             if (urlString.startsWith(FILE_PROTOCOL_PREFIX))
             {
-               path = urlString.substring(8);
+               path = urlString.substring(FILE_PROTOCOL_PREFIX.length());
             }
             else
             {
```

#50 - 03/15/2021 09:01 AM - Sergey Ivanovskiy

These changes are not correct because file:/// is important for local files. Please look at https://en.wikipedia.org/wiki/File_URI_scheme.

#51 - 03/15/2021 09:10 AM - Sergey Ivanovskiy

For an example, this url throws exception new URL("file://" + "C:\\tmp\\name.txt"); but this one is correct

```
new URL("file:///" + "C:\\tmp\\name.txt");
```

#52 - 03/15/2021 09:38 AM - Roger Borrello

Sergey Ivanovskiy wrote:

> For an example, this url throws exception new URL("file://" + "C:\\tmp\\name.txt"); but this one is correct
> [...]

From how I read RFC3986 (https://tools.ietf.org/html/rfc3986#section-3.1), the slashes are related to whether or not the scheme indicates a hierarchy or not. Our launcher: scheme won't be a hierarchy, so after the colon starts the actual path to what is launched, no authority is needed.

In the case of "file:", this scheme is definitely for a hierarchy. The prefix is file://, and the next part is the authority to access the hierarchy. This can be host, but can be excluded if the authority is a local resource. This implies localhost but can be left out. Hence the file:/// can be broken up to hierarchical scheme (file://) + no authority + start of path (/...).

That's how I interpret the RFC. I know section 3.3 is really confusing. Did you try your example including localhost as the authority?

```
new URL("file://localhost/" + "C:\\tmp\\name.txt");
```

#53 - 03/15/2021 09:40 AM - Roger Borrello

As to what that means to our code, if we agree on the "file://" prefix... we have to further decide if there is or isn't an authority included by validating the next character is a /, which would imply no authority is included, and the resource is local.

#54 - 03/15/2021 09:43 AM - Roger Borrello

To close out my thoughts (I have a doctor appointment shortly), if we do find file:///, we can proceed as if the resource is local. So perhaps the issue I have is with the name of the prefix:

```
   /** file protocol prefix to construct a local file related URL. */
   public static final String FILE_PROTOCOL_LOCAL_PREFIX = "file:///";
```

#55 - 03/15/2021 09:46 AM - Greg Shah

Why are we messing with file:// in this task? The plan was to **always** use a launcher:// protocol for the launcher. This allows file:/// to remain as the tool for pushing FWD client files to the browser. That is not needed for the launcher, whose entire concept is about opening files that already exist on the browser's system.

#56 - 03/15/2021 09:51 AM - Sergey Ivanovskiy

Roger Borrello wrote:

> To close out my thoughts (I have a doctor appointment shortly), if we do find file:///, we can proceed as if the resource is local. So perhaps the issue I have is with the name of the prefix:
> [...]

Correct.

#57 - 03/15/2021 02:06 PM - Greg Shah

I've just spoken with Roger and we've agreed on the following approach.

1. OPEN-MIME-RESOURCE will be enhanced to be aware of the local launcher.

The 4GL code will use OPEN-MIME-RESOURCE "application/fwd-local-launcher" "<browser_system_file_name_or_command_line>". where the URI is the exact command line to launch on the browser system. Then OPEN-MIME-RESOURCE will handle this as follows:

- GUI Web Client
  - Create a temporary file.
  - Write the <browser_system_file_name_or_command_line> into the file as a single line.
  - Close the temporary file.
  - Stream that file to the browser with the "application/fwd-local-launcher" MIME type.
  - Delete the temporary file.
  - I hope that we don't need a MIME-type-specific filename extension for the temp-file but if needed we will do so.
- Swing GUI Client
  - Call the new Desktop.desktopOpen() with the <browser_system_file_name_or_command_line>.

2. The launcher will be installed as a helper application for "application/fwd-local-launcher" as documented for Firefox in #4179-20. We need to document the registration for Chrome and Edge too.

3. The launcher must be changed:

- To accept the filename of this input file.
- To read the line of text from the file and call desktopOpen() to do the launch on that input.
- Close the temp-file.
- If the browser doesn't delete the temp-file, we should do so.

I think this avoids any need for a custom protocol and it should work within the bounds of the browser MIME type processing.

#58 - 03/15/2021 02:13 PM - Greg Shah

For security reasons, we will prompt the user before calling desktopOpen(). The idea is to display the command line that is going to be invoked and allow the user to confirm or cancel. This extra confirmation dialog must be done in the launcher.

I mentioned this security exposure in #4179-1. Once this is installed in a user's browser, any malicious website can use this MIME type to invoke anything like format.com c: with predicably bad results. I see no alternative except to force the user to decide if it is safe.

We still strongly recommend that you do not use this launcher. It is too dangerous. You've been warned.

#59 - 03/15/2021 06:06 PM - Roger Borrello

Some change questions...

I want to add GuiWebDriver.openLauncherResource that will mimic openMimeResource but will create a temp wrapper file, write the uriString into it, create a resource ID for the wrapper file, and call websock.openMimeResource on that wrapper.

Does this require a TC.openLauncherResource?

#60 - 03/15/2021 07:42 PM - Greg Shah

> I want to add GuiWebDriver.openLauncherResource that will mimic openMimeResource but will create a temp wrapper file, write the uriString into it, create a resource ID for the wrapper file, and call websock.openMimeResource on that wrapper.
>
> Does this require a TC.openLauncherResource?

No. I don't want this feature to be callable from the server. Let's leave the external interface as OPEN-MIME-RESOURCE which is generic to all GUI clients. I like having a helper GuiWebDriver.openLauncherResource which can isolate this code BUT it is purely a specific path will only ever be called from the GUI Web Client. This means it would be could just be called from GuiWebDriver.openMimeResource and can just be a private method.

#61 - 03/16/2021 09:28 AM - Roger Borrello

- File 4179_launcher_take1.mkv added

- File GuiWebDriver.patch added

The wrapping end is coded and attached as a patch. I would appreciate some review. I am working on the native code, which will unwrap and open.

The current behavior on my Firefox is a warning that a popup was attempted, and when I allow popups from localhost, the file is downloaded to ~/Downloads and Firefox prompts me for the helper to use. Upon choosing FWD Local Launcher, the actual file is launched (since I haven't updated the native code yet). Video attached.

#62 - 03/16/2021 10:13 AM - Greg Shah

Code Review #4179-61 Patch

1. The changes to GuiWebDriver.openMimeResource() are:

- should not be unconditional
    - The launcher-related code should be inside a if (LAUNCHER_MIME_TYPE.equalsIgnoreCase(mimeType)) conditional block.
    - I don't see the advantage to having the schema checking (LAUNCHER_SCHEME.equalsIgnoreCase(scheme)) be optional (it uses || in the if() condition). Either it is required or we drop it. Having it as optional just creates confusion.
- can be simplified
    - It is not clear what the catch block for URISyntaxException is meant to fix. If the format is not right, shouldn't we just error out?
    - I think this is the case for the IOException as well. Trying to keep going at that point seems to be a bad idea.
- I think the use of UiUtils.resolveLocalFileResource() is wrong. We are not trying to load a local resource. Even if the filename happens to exist on the FWD client system, it could just be by chance that the filename is the same as the one to be opened on the browser side.

2. Let's rename openLauncherResource to writeLauncherWrapper. The helper does not really open anything in the browser so it the current name is confusing.

3. In the javadoc, the use of "presentation" is confusing. I think it should be more clear that the uriString is the content that will be written into the wrapper file.

#63 - 03/16/2021 11:21 AM - Roger Borrello

Greg Shah wrote:

> Code Review #4179-61 Patch
>
> 1. The changes to GuiWebDriver.openMimeResource() are:
>
> - should not be unconditional
>   - The launcher-related code should be inside a if (LAUNCHER_MIME_TYPE.equalsIgnoreCase(mimeType)) conditional block.
>   - I don't see the advantage to having the schema checking (LAUNCHER_SCHEME.equalsIgnoreCase(scheme)) be optional (it uses || in the if() condition). Either it is required or we drop it. Having it as optional just creates confusion.
> - can be simplified
>   - It is not clear what the catch block for URISyntaxException is meant to fix. If the format is not right, shouldn't we just error out?

At this point we don't know whether we have a URI for launcher://file_to_launch or simply a path to a file. If a non-URI like /home/rfb/Pictures/Faceshot.png or C:\Users\rfb\Pictures\Faceshot.png are passed in, we would throw a URISyntaxException. It helps us parse the launchFilePath will be wrapped up. Do we need launcher:// or not?

> - I think this is the case for the IOException as well. Trying to keep going at that point seems to be a bad idea.

The (now) writeLauncherWrapper can thrown an IOException which was uncaught, and the compiler didn't like that.

#64 - 03/16/2021 11:29 AM - Greg Shah

> At this point we don't know whether we have a URI for launcher://file_to_launch or simply a path to a file.

We define what is valid and not valid, so we in fact can know this in advance.

> Do we need launcher:// or not?

You tell me. Does it serve a useful function?

> > I think this is the case for the IOException as well. Trying to keep going at that point seems to be a bad idea.
>
> The (now) writeLauncherWrapper can thrown an IOException which was uncaught, and the compiler didn't like that.

I understand. I'd like to hide the specifics of this launcher processing in the helper as much as possible. Can you deal with the exception there?

#65 - 03/16/2021 11:56 AM - Roger Borrello

- % Done changed from 40 to 60

#66 - 03/16/2021 01:13 PM - Sergey Ivanovskiy

Roger Borrello wrote:

> Greg Shah wrote:
>
> > Code Review [#4179-61](#4179-61) Patch
> >
> > 1. The changes to GuiWebDriver.openMimeResource() are:
> >
> > - should not be unconditional
> >   - The launcher-related code should be inside a if (LAUNCHER_MIME_TYPE.equalsIgnoreCase(mimeType)) conditional block.
> >   - I don't see the advantage to having the schema checking (LAUNCHER_SCHEME.equalsIgnoreCase(scheme)) be optional (it uses || in the if() condition). Either it is required or we drop it. Having it as optional just creates confusion.

- can be simplified
    - It is not clear what the catch block for URISyntaxException is meant to fix. If the format is not right, shouldn't we just error out?

> At this point we don't know whether we have a URI for launcher://file_to_launch or simply a path to a file. If a non-URI like /home/rfb/Pictures/Faceshot.png or C:\Users\rfb\Pictures\Faceshot.png are passed in, we would throw a URISyntaxException. It helps us parse the launchFilePath will be wrapped up. Do we need launcher:// or not?

It seems you can encode these file paths and build URI correctly.

#67 - 03/16/2021 05:54 PM - Roger Borrello

A minor problem with the Swing that I can move on from. I was just trying to complete the code. But the JNI cannot be found. Is there a particular location the libp2j.so need to be installed to?

```
Caused by: java.lang.UnsatisfiedLinkError: com.goldencode.p2j.ui.client.gui.Desktop.desktopOpenUnwrapped(Ljava/lang/String;)I
        at com.goldencode.p2j.ui.client.gui.Desktop.desktopOpenUnwrapped(Native Method)
```

#68 - 03/16/2021 06:59 PM - Greg Shah

It just needs to live wherever the p2j.jar exists. It looks like you changed the name of the native method. Did you do a clean rebuild of everything including re-generating the header file and all native code? If you did, did you deploy.prepare to push your changes into the project?

#69 - 03/16/2021 10:49 PM - Roger Borrello

Greg Shah wrote:

> It just needs to live wherever the p2j.jar exists. It looks like you changed the name of the native method. Did you do a clean rebuild of everything including re-generating the header file and all native code? If you did, did you deploy.prepare to push your changes into the project?

Yes to all of those.

I created a function that will and won't unwrap the launch command. This allows SwingGuiDriver.openMimeResource to simply call the com.goldencode.p2j.ui.client.gui.Desktop.desktopOpenUnwrapped method (since I left the desktopOpen code the same as the fwd_local_launcher native code which unwraps. (There's another Desktop class that creates ambiguity).

```
Caused by: java.lang.UnsatisfiedLinkError: com.goldencode.p2j.ui.client.gui.Desktop.desktopOpenUnwrapped(Ljava/lang/String;)I
        at com.goldencode.p2j.ui.client.gui.Desktop.desktopOpenUnwrapped(Native Method)
        at com.goldencode.p2j.ui.client.gui.driver.swing.SwingGuiDriver.openMimeResource(SwingGuiDriver.java:860)
        at com.goldencode.p2j.ui.chui.ThinClient.openMimeResource(ThinClient.java:23694)
```

I committed my changes in Rev 12127 for review.

#70 - 03/17/2021 08:08 AM - Greg Shah

Code Review Task Branch 3821c Revision 12127

1. LAUNCHER_MIME_TYPE should only be defined in one place. Why not put it in Desktop.java and use it in both SwingGuiDriver and GuiWebDriver?

2. In SwingGuiDriver it is OK to import com.goldencode.p2j.ui.client.gui.* and then use the unqualified Desktop class name. We don't really ever use fully-qualified package names.

3. In SwingGuiDriver.openMimeResource(), if you just return after the use of desktopOpenUnwrapped(), then you don't have to nest the rest of the method inside the else block. The advantage is that there is one less nesting level and the code is easier to read.

4. In GuiWebDriver.openMimeResource(), you can delete anything associated with the scheme local variable, it is useless.

5. I don't understand the need for the extra complexity in GuiWebDriver.openMimeResource(). Can't this code:

```
      if (LAUNCHER_MIME_TYPE.equalsIgnoreCase(mimeType))
      {
          launchFilePath = uriString;
          try
          {
             uri = new URI(uriString);
             scheme = uri.getScheme();
             launchFilePath = uri.getPath();
          }
          catch(URISyntaxException e)
          {
              // ignore
          }

          finalDelete = true;
          finalEmbedded = false;
          try
          {
             finalUri = writeLauncherWrapper(UiUtils.resolveLocalFileResource(launchFilePath));
          }
          catch (IOException e)
          {
              // Do nothing
          }
      }
```

be written like this:

```
if (LAUNCHER_MIME_TYPE.equalsIgnoreCase(mimeType))
{
         uri           = writeLauncherWrapper(uriString);
         deleteContent = true;
         embedded      = false;
      }
```

Anything needed for creating the wrapper can be hidden inside writeLauncherWrapper() including the exception processing. openMimeResource() is easy to read instead of getting cluttered with code for one special case. Also, note that we don't need the extra intermediate variables, which add code and changes without any real advantage.

If we needed to implement error processing, we could always return null from writeLauncherWrapper() and process off that.

6. GuiWebDriver.LAUNCHER_SCHEME is dead code.

7. Using unwrapLauncherFile() from the desktop.c **JNI code** desktopOpen() makes no sense to me. The only code that ever needs unwrapLauncherFile() is the launcher itself. It definitely should not be in JNI code since the launcher would never be calling JNI. I think the desktop.c changes are not needed at all. There does not need to be a desktopOpenUnwrapped(). Just make a desktopOpen() and call that from the SwingGuiDriver. You can move the unwrapLauncherFile() function into the launcher and avoid the need to link to desktop.o. unwrapLauncherFile() is not common code because it is only needed in one place.

8. What is the benefit of removing the argument handling in the native code? I think we want to keep that as a feature.

9. Let's change the name of open_file to desktop_open_worker. It is more clear that way and won't conflict with future "file-systemy" usage.

#71 - 03/17/2021 09:26 AM - Roger Borrello

Greg Shah wrote:

> 1. LAUNCHER_MIME_TYPE should only be defined in one place. Why not put it in Desktop.java and use it in both SwingGuiDriver and GuiWebDriver?

Thanks for this review... I'm working on them. I found that in using Desktop.java, we have some conflicts with awt.Desktop. Should I name that something else?

#72 - 03/17/2021 10:25 AM - Greg Shah

> I found that in using Desktop.java, we have some conflicts with awt.Desktop. Should I name that something else?

I see. Yes, change it to DesktopHelpers.

#73 - 03/17/2021 05:40 PM - Roger Borrello

- % Done changed from 60 to 70

Changes committed in 12140.

Regarding the parameters, I didn't want to hold up the task for this. There's some complexity related to whether there are space-delineated parameters to pass to argv array of strings method of API and how to handle embedded spaces within parameters that I didn't want to spend too much time on, given the ability to handle "batch" files for this.

#74 - 03/17/2021 06:06 PM - Eugenie Lyzenko

Roger Borrello wrote:

> Changes committed in 12140.
>
> Regarding the parameters, I didn't want to hold up the task for this. There's some complexity related to whether there are space-delineated parameters to pass to argv array of strings method of API and how to handle embedded spaces within parameters that I didn't want to spend too much time on, given the ability to handle "batch" files for this.

Roger,

With update 12140 it is not possible to build FWD anymore with ./gradlew all:

```
...
:ant-native (Thread[Execution worker for ':',5,main]) started.
[ant:exec] make: *** No rule to make target 'process.o', needed by 'libp2j.so'.  Stop.

> Task :ant-native FAILED
Caching disabled for task ':ant-native' because:
  Build cache is disabled
Task ':ant-native' is not up-to-date because:
  Task has not declared any outputs despite executing actions.
[ant:echo] 
:ant-native (Thread[Execution worker for ':',5,main]) completed. Took 0.008 secs.

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':ant-native'.
> exec returned: 2

* Try:
Run with --stacktrace option to get the stack trace. Run with --debug option to get more log output. Run with 
--scan to get full insights.

* Get more help at https://help.gradle.org

Deprecated Gradle features were used in this build, making it incompatible with Gradle 6.0.
Use '--warning-mode all' to show the individual deprecation warnings.
See https://docs.gradle.org/5.6.4/userguide/command_line_interface.html#sec:command_line_warnings

BUILD FAILED in 1m 44s
...
```

Can you please take a look.

#75 - 03/17/2021 06:23 PM - Eugenie Lyzenko

Eugenie Lyzenko wrote:

> Roger Borrello wrote:
>
> > Changes committed in 12140.
> >
> > Regarding the parameters, I didn't want to hold up the task for this. There's some complexity related to whether there are space-delineated parameters to pass to argv array of strings method of API and how to handle embedded spaces within parameters that I didn't want to spend too much time on, given the ability to handle "batch" files for this.
>
> Roger,
>
> With update 12140 it is not possible to build FWD anymore with ./gradlew all:
>
> [...]
>
> Can you please take a look.

I have a fix for this. Will commit soon.

#76 - 03/17/2021 06:27 PM - Eugenie Lyzenko

Eugenie Lyzenko wrote:

> Eugenie Lyzenko wrote:
>
> > Roger Borrello wrote:
> >
> > > Changes committed in 12140.
> > >
> > > Regarding the parameters, I didn't want to hold up the task for this. There's some complexity related to whether there are space-delineated parameters to pass to argv array of strings method of API and how to handle embedded spaces within parameters that I didn't want to spend too much time on, given the ability to handle "batch" files for this.
> >
> > Roger,
> >
> > With update 12140 it is not possible to build FWD anymore with ./gradlew all:
> >
> > [...]

> > Can you please take a look.
>
> I have a fix for this. Will commit soon.

Never mind. I see this is already fixed in 12141. Thanks.

#77 - 03/17/2021 06:33 PM - Roger Borrello

Eugenie Lyzenko wrote:

> Never mind. I see this is already fixed in 12141. Thanks.

My apologies. The makefile commit was missed.

#78 - 03/17/2021 07:31 PM - Roger Borrello

I was missing the DesktopHelpers classname in the method declaration, leading to the issue in #4179-67. Fixed in 12142.

#79 - 03/17/2021 08:52 PM - Greg Shah

Code Review Task 3821c Revisions 12140 through 12142

The changes are good.

The GuiWebDriver.writeLauncherWrapper() should return null to signal failure. Returning a non-null string is confusing. The return value isn't going to be used in the error case anyway so returning null will work.

#80 - 03/17/2021 08:52 PM - Greg Shah

Is it fully working at this point?

#81 - 03/17/2021 10:55 PM - Roger Borrello

Greg Shah wrote:

> Code Review Task 3821c Revisions 12140 through 12142
>
> The changes are good.
>
> The GuiWebDriver.writeLauncherWrapper() should return null to signal failure. Returning a non-null string is confusing. The return value isn't going to be used in the error case anyway so returning null will work.

The thought was that the original string could be passed back for the original processing, should it not be able to be wrapped. The parameters for deleteContent and embedded would only be modified if wrapping occurred.

#82 - 03/17/2021 11:25 PM - Roger Borrello

Greg Shah wrote:

> Is it fully working at this point?

My testcase works on Firefox. I am trying to get the same results on Chrome.

When I try to debug the customer application, I somehow have occupied my debug port. Whenever I try to connect with the web client, I get his in server.log:

```
ERROR: transport error 202: bind failed: Address already in use
ERROR: JDWP Transport dt_socket failed to initialize, TRANSPORT_INIT(510)
JDWP exit error AGENT_ERROR_TRANSPORT_INIT(197): No transports initialized [debugInit.c:750]
```

I have a conversion running, so I can't reboot.

#83 - 03/17/2021 11:34 PM - Roger Borrello

Good article: https://stackoverflow.com/questions/6267546/how-do-i-associate-a-custom-mime-type-to-my-local-application-in-the-major-brows

#84 - 03/18/2021 07:19 AM - Greg Shah

> The thought was that the original string could be passed back for the original processing, should it not be able to be wrapped. The parameters for deleteContent and embedded would only be modified if wrapping occurred.

The caller already has the original string, so it doesn't need to be passed back. Comparing against null to find an error is much cleaner, much faster and much less ambiguous that detecting the difference in the returned string. As a general rule, null is commonly used to signal a failure whereas passing back non-null text can be easily confused with a successful run.

#85 - 03/18/2021 10:27 AM - Roger Borrello

Using a Chrome on Windows, with the client/server in my Ubuntu VM, I see the wrapper file:

```
fwd@rfbvm:~$ sudo cat /tmp/launcher_1566999833568740664.fwl 
c:/debug.txt
```

I didn't not receive the file at the web client, which has Downloads set to go to C:\Users\Administrator\Downloads.

Unfortunately I don't have Eclipse setup on the VM, and I was unable to setup my laptop as the client/server due to the firewall. Is that easier to reconfigure, or should I try to get Eclipse going? My memory is very low with 2 VMs running.

#86 - 03/18/2021 11:05 AM - Greg Shah

> I didn't not receive the file at the web client

The double negative makes me question: are you saying "I did I receive the file at the web client"?

> Unfortunately I don't have Eclipse setup on the VM, and I was unable to setup my laptop as the client/server due to the firewall. Is that easier to reconfigure, or should I try to get Eclipse going? My memory is very low with 2 VMs running.

The easiest thing to do is just to add logging to the launcher.

#87 - 03/18/2021 12:04 PM - Roger Borrello

I won't even quote that... I meant to say, I didn't receive the file at the web client.

I was able to get firewall configured so I could do real testing. The file **is** being returned to the Windows webclient. So far I was able to validate: c:\debug.txt

```
c:\>type c:\users\Administrator\Downloads\launcher_4251185209128260650.fwl
c:/debug.txt
```

C:\<path with spaces>\file.jpg

```
c:\>type c:\users\Administrator\Downloads\launcher_3734747507685649092.fwl
c:/Images/3D/Frame-Vue Tray 3D.JPG
```

I tried a UNC, and it is returned incorrectly in the file:

```
c:\>type c:\Users\Administrator\Downloads\launcher_6025407207394570286.fwl
//vboxsrv/projects/fwd_600_wide_600_high.png
c:\>fwd_local_launcher.exe c:\Users\Administrator\Downloads\launcher_6025407207394570286.fwl
Error opening //vboxsrv/projects/fwd_600_wide_600_high.png (DESKTOP_ERR_FORK): No error
```

In the web client, I can put whatever I want in. But at the TC.openMimeResource, it converts:

```
      // first get rid of Windows style separators as preparation step
      uriString = uriString.replace('\\', '/');

      uriString = UiUtils.resolveLocalFileResource(uriString);
```

Perhaps this should be aware of the mime-type being the local-launcher, and avoid touching the given URL?

#88 - 03/18/2021 12:19 PM - Sergey Ivanovskiy

You are free to do any required changes for your task taking into account the existing functionality of open-mime-resource.

#89 - 03/18/2021 12:56 PM - Greg Shah

> In the web client, I can put whatever I want in. But at the TC.openMimeResource, it converts:

Yes, in TC.openMimeResource you can bypass the modification for launcher mime types.

#90 - 03/18/2021 03:59 PM - Roger Borrello

There was also a need to handle the file:/// prefix. Updates in 12149.

#91 - 03/18/2021 04:20 PM - Greg Shah

Code Review Task Branch 3821c Revision 12149

I'm OK with the changes.

#92 - 03/18/2021 04:21 PM - Greg Shah

What is left to do? Is this really just 70% done?

#93 - 03/18/2021 05:08 PM - Roger Borrello

- % Done changed from 70 to 90

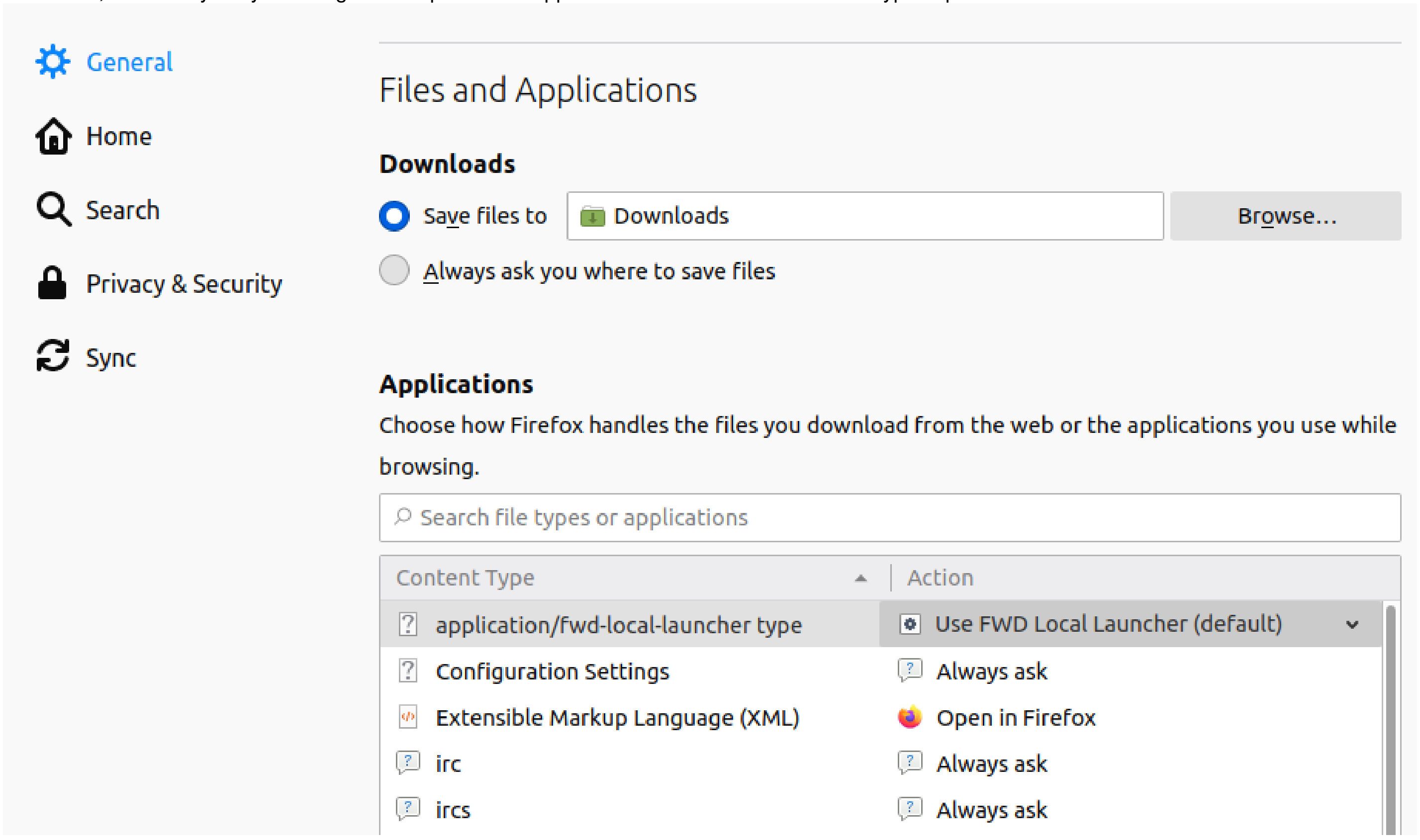
- File 4719_firefox_preferences1.png added

Greg Shah wrote:

> What is left to do? Is this really just 70% done?

More like 90%.

The integration into the browser is very clumsy and un-precise. I cannot find the mimeType.rdf file for the Firefox configuration mentioned in #4179-20 . However, it was very easy to configure a helper for the application/fwd-local-launcher mime-type in preferences:



Files and Applications

Downloads

Save files to: Downloads [Browse...]

Always ask you where to save files

Applications

Choose how Firefox handles the files you download from the web or the applications you use while browsing.

Search file types or applications

Content Type	Action
application/fwd-local-launcher type	Use FWD Local Launcher (default)
Configuration Settings	Always ask
Extensible Markup Language (XML)	Open in Firefox
irc	Always ask
ircs	Always ask

When you launch, the wrapper file is downloaded, and you can load the file with the launcher helper.

In Chrome, you have to configure the OS to handle .fwl file types. It was easier under Windows.

#94 - 03/18/2021 05:11 PM - Greg Shah

> The integration into the browser is very clumsy and un-precise. I cannot find the mimeType.rdf file for the Firefox configuration mentioned in #4179-20. However, it was very easy to configure a helper for the application/fwd-local-launcher mime-type in preferences:

If it is easy for a user to do, then we are good to go. I don't really care about mimeType.rdf.

It sounds like this is done. What is left to do?

#95 - 03/18/2021 05:48 PM - Roger Borrello

- Status changed from WIP to Review

- % Done changed from 90 to 100

OK... I concur, so I believe it is done. I've moved on to the customer's implementation of OPEN-MIME-TYPE.

#96 - 03/18/2021 06:32 PM - Greg Shah

- Status changed from Review to Closed

Good work!

#97 - 03/19/2021 11:06 AM - Roger Borrello

- vendor_id deleted (GCD)

- Assignee changed from Roger Borrello to Eric Faulhaber

- Priority changed from Normal to Low

- billable changed from No to Yes

Greg Shah wrote:

> Good work!

Thanks... I do need to see if the way we are spawning under Linux properly returns control to the invoking process. I do notice a "hang" in the customer application on Linux that I will try to see if I can replicate in a testcase.

#98 - 03/19/2021 11:07 AM - Roger Borrello

- billable changed from Yes to No

- Assignee changed from Eric Faulhaber to Roger Borrello

- Priority changed from Low to Normal

Not sure how all that changed in the task header. Changing back.

#99 - 03/19/2021 12:15 PM - Roger Borrello

I wanted to note that when the application hangs, the web client can never seem to re-connect. In server.log:

```
ERROR: transport error 202: bind failed: Address already in use
ERROR: JDWP Transport dt_socket failed to initialize, TRANSPORT_INIT(510)
JDWP exit error AGENT_ERROR_TRANSPORT_INIT(197): No transports initialized [debugInit.c:750]
```

I've stopped all running programs, restarted postgresql, everything. And retried to no avail. Rebooting corrected this, but that's not a good state to leave things in.

```
rfb@rfb:~/project$ sudo netstat -lp
[sudo] password for rfb:
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 localhost:2025          0.0.0.0:*               LISTEN      2794/ssh
tcp        0      0 localhost:2222          0.0.0.0:*               LISTEN      2794/ssh
tcp        0      0 localhost:2224          0.0.0.0:*               LISTEN      2794/ssh
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN      11198/systemd-resol
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN      1932/sshd
tcp        0      0 localhost:ipp           0.0.0.0:*               LISTEN      10599/cupsd
tcp        0      0 localhost:25432         0.0.0.0:*               LISTEN      2799/ssh
tcp        0      0 localhost:5433          0.0.0.0:*               LISTEN      13446/postgres
tcp        0      0 localhost:2143          0.0.0.0:*               LISTEN      2794/ssh
udp        0      0 0.0.0.0:53222           0.0.0.0:*                           1686/avahi-daemon:
udp        0      0 0.0.0.0:mdns            0.0.0.0:*                           1686/avahi-daemon:
udp        0      0 localhost:domain        0.0.0.0:*                           11198/systemd-resol
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*                           1547/dhclient
udp        0      0 0.0.0.0:ipp             0.0.0.0:*                           10600/cups-browsed
Active UNIX domain sockets (only servers)
Proto RefCnt Flags       Type       State         I-Node   PID/Program name     Path
unix  2      [ ACC ]     SEQPACKET  LISTENING     19712    1/systemd            /run/udev/control
unix  2      [ ACC ]     STREAM     LISTENING     41482    2136/systemd         /run/user/1000/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     41486    2136/systemd         /run/user/1000/gnupg/S.gpg-age
nt.extra
unix  2      [ ACC ]     STREAM     LISTENING     41487    2136/systemd         /run/user/1000/gnupg/S.gpg-age
nt.ssh
unix  2      [ ACC ]     STREAM     LISTENING     41488    2136/systemd         /run/user/1000/gnupg/S.gpg-age
nt
unix  2      [ ACC ]     STREAM     LISTENING     41489    2136/systemd         /run/user/1000/gnupg/S.gpg-age
nt.browser
unix  2      [ ACC ]     STREAM     LISTENING     41490    2136/systemd         /run/user/1000/gnupg/S.dirmngr
unix  2      [ ACC ]     STREAM     LISTENING     41491    2136/systemd         /run/user/1000/bus
unix  2      [ ACC ]     STREAM     LISTENING     33498    2153/gnome-keyring-  /run/user/1000/keyring/control
unix  2      [ ACC ]     STREAM     LISTENING     39359    2153/gnome-keyring-  /run/user/1000/keyring/pkcs11
unix  2      [ ACC ]     STREAM     LISTENING     36815    2156/cinnamon-sessi  @/tmp/.ICE-unix/2156
unix  2      [ ACC ]     STREAM     LISTENING     34959    1958/Xorg            /tmp/.X11-unix/X0
unix  2      [ ACC ]     STREAM     LISTENING     44045    2281/pulseaudio      /run/user/1000/pulse/native
unix  2      [ ACC ]     STREAM     LISTENING     36768    2222/ssh-agent       /tmp/ssh-LWnzgOi7LleM/agent.21
56
unix  2      [ ACC ]     STREAM     LISTENING     36816    2156/cinnamon-sessi  /tmp/.ICE-unix/2156
unix  2      [ ACC ]     STREAM     LISTENING     34958    1958/Xorg            @/tmp/.X11-unix/X0
unix  2      [ ACC ]     STREAM     LISTENING     3214016  13446/postgres       /var/run/postgresql/.s.PGSQL.5
433
unix  2      [ ACC ]     STREAM     LISTENING     21319    1249/irqbalance      @irqbalance1249.sock
unix  2      [ ACC ]     STREAM     LISTENING     655154   1/systemd            /run/cups/cups.sock
unix  2      [ ACC ]     STREAM     LISTENING     136502   1/systemd            /run/systemd/private
unix  2      [ ACC ]     SEQPACKET  LISTENING     19666    1/systemd            /run/systemd/coredump
unix  2      [ ACC ]     STREAM     LISTENING     19674    1/systemd            /run/systemd/fsck.progress
unix  2      [ ACC ]     STREAM     LISTENING     19678    1/systemd            /run/systemd/journal/stdout
unix  2      [ ACC ]     STREAM     LISTENING     19702    1/systemd            /run/lvm/lvmpolld.socket
unix  2      [ ACC ]     STREAM     LISTENING     19704    1/systemd            /run/lvm/lvmetad.socket
unix  2      [ ACC ]     STREAM     LISTENING     23997    1/systemd            /run/uuidd/request
unix  2      [ ACC ]     STREAM     LISTENING     17964    1/systemd            /run/avahi-daemon/socket
unix  2      [ ACC ]     STREAM     LISTENING     17966    1/systemd            /run/acpid.socket
unix  2      [ ACC ]     STREAM     LISTENING     17968    1/systemd            /var/run/dbus/system_bus_socke
t
unix  2      [ ACC ]     STREAM     LISTENING     36185    1333/NetworkManager  /run/NetworkManager/private-dh
cp
unix  2      [ ACC ]     STREAM     LISTENING     37521    2239/dbus-daemon     @/tmp/dbus-j5cjhoKLQK
unix  2      [ ACC ]     STREAM     LISTENING     24185    1385/nvidia-persist  /var/run/nvidia-persistenced/s
ocket
```

#100 - 03/19/2021 12:22 PM - Greg Shah

The error you posted looks like something associated with the Java debugger support. Disable that in the directory before you test this.

#101 - 03/19/2021 12:59 PM - Roger Borrello

Greg Shah wrote:

> The error you posted looks like something associated with the Java debugger support. Disable that in the directory before you test this.

Yes, that's what I determined, as well. It does hamper my ability to determine what is hanging upon use of the launcher.

#102 - 03/19/2021 01:20 PM - Greg Shah

Logging is a girl's best friend.

#103 - 03/19/2021 01:34 PM - Roger Borrello

Greg Shah wrote:

> Logging is a girl's best friend.

The launcher is completely gone. The web client application is not responding to anything. The process list shows there's a "Tooltip Worker" under the spawner still running.

#104 - 03/19/2021 02:00 PM - Greg Shah

> The process list shows there's a "Tooltip Worker" under the spawner still running.

Are you saying the JVM has a single thread remaining? Get a thread dump.

#105 - 03/19/2021 03:54 PM - Roger Borrello

It is sporadic, because I was able to open links repetitively without issue on Firefox. Then it happened on Chrome, and again, not all the time. I'm adding some logging, chica.

#106 - 03/19/2021 06:07 PM - Roger Borrello

I did find an issue related to execv which I fixed. Also:

- postbuild processing was added
- some more specific error messages were added

These are in 3821c-12153

This version does not have the enhancement to warn the user about upcoming launch. Should I leave that as a future enhancement? I would think this might work in GuiWebDriver.openMimeResource at the time we find mimeType=LAUNCHER_MIME_TYPE before wrapping the launch string. If we wait until native code, we'll to deal with Win32 and X. We could also have a JNI we call into.

OPEN-MIME-RESOURCE needs to be enhance to receive NO-CONFIRM and enable bypassing (or for backward compatibility, make it CONFIRM and default to no-confirm).

In Linux, the desktop integration when freedesktop is installed is very easy.

Create /usr/share/mime/packages/fwd-local-launcher.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<mime-info xmlns="http://www.freedesktop.org/standards/shared-mime-info">
    <mime-type type="application/fwd-local-launcher">
        <sub-class-of type="text/plain"/>
        <comment>FWD Local Launcher</comment>
        <glob pattern="*.fwl"/>
    </mime-type>
</mime-info>
```

Then run:

```
sudo update-mime-database /usr/share/mime
sudo chmod o+r -R /usr/share/mime/*
```

Create /usr/share/applications/fwd_local_launcher.desktop

```
[Desktop Entry]
Name=FWD Local Launcher
Comment=Launch Local Files
Exec=/usr/bms/bin/fwd_local_launcher %U
GenericName=FWD Local Launcher
Terminal=false
Type=Application
MimeType=application/fwd-local-launcher;
NoDisplay=true
```

Then run:

```
sudo update-desktop-database /usr/share/applications
sudo chmod o+r -R /usr/share/applications/*
```

Alternatively, these can be performed in the user directories after placing the files above in $HOME/.local/share/mime/application/ and $HOME/.local/share/applications respectively:

```
update-mime-database $HOME/.local/share/mime
update-desktop-database $HOME/.local/share/applications
```

At that point, FWD Local Launcher is available to open the launcher files.

#107 - 03/19/2021 06:26 PM - Greg Shah

- Status changed from Closed to WIP

- % Done changed from 100 to 90

> This version does not have the enhancement to warn the user about upcoming launch. Should I leave that as a future enhancement?

Good point. We need to handle this now.

> I would think this might work in GuiWebDriver.openMimeResource at the time we find mimeType=LAUNCHER_MIME_TYPE before wrapping the launch string.

No can do. This would only protect customers from launching that is driven by the converted 4GL code. The real danger here doesn't come from converted apps. The danger comes from any HTML (on random sites, whatever) that loads a malicious file using "our" mime type. Once it is installed in a browser, that user is exposed from anywhere. We MUST implement this as a safety value in the native code.

> If we wait until native code, we'll to deal with Win32 and X.

Yes, this is what we must do. I think we can just use something like WIN32 MessageBoxA right?

> We could also have a JNI we call into.

There is no guarantee that a JVM is installed on the browser system. We must use native WIN32 and X.

#108 - 03/21/2021 10:34 PM - Roger Borrello

Is language support a requirement? We need to have:

1. A warning message about the danger of launching a resource
2. An error when the resource is not found. Currently it exits silently

#109 - 03/22/2021 08:13 AM - Greg Shah

> Is language support a requirement?

Yes, but it can be deferred until another customer needs it.

> A warning message about the danger of launching a resource

Yes, but make sure that the user has the option to CANCEL. Also, make the CANCEL the default button so that a "trigger happy" user must actively engage to open a linked document. I know it is a pain, but this launcher is REALLY dangerous.

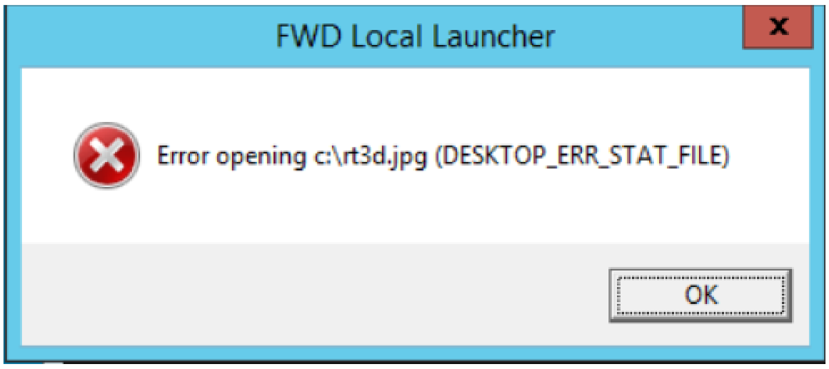
> An error when the resource is not found. Currently it exits silently

Yes, this makes sense.

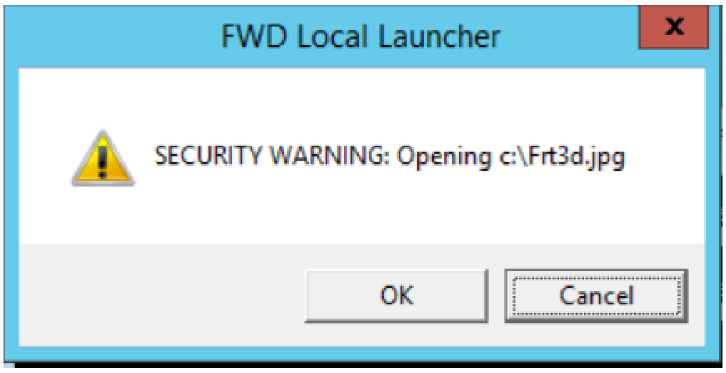
#110 - 03/22/2021 12:05 PM - Roger Borrello

- *File 4179_demo_of_message_boxes_on_windows.mkv added*
- *File 4179_error_message.png added*
- *File 4179_warning_message.png added*
- *Status changed from WIP to Review*
- *% Done changed from 90 to 100*
- *billable changed from No to Yes*

Take a look at the demo video and below messages:


FWD Local Launcher

Error opening c:\rt3d.jpg (DESKTOP_ERR_STAT_FILE)


FWD Local Launcher

SECURITY WARNING: Opening c:\Frt3d.jpg

#111 - 03/22/2021 12:07 PM - Roger Borrello

- billable changed from Yes to No

#112 - 03/22/2021 12:14 PM - Greg Shah

The error message is good.

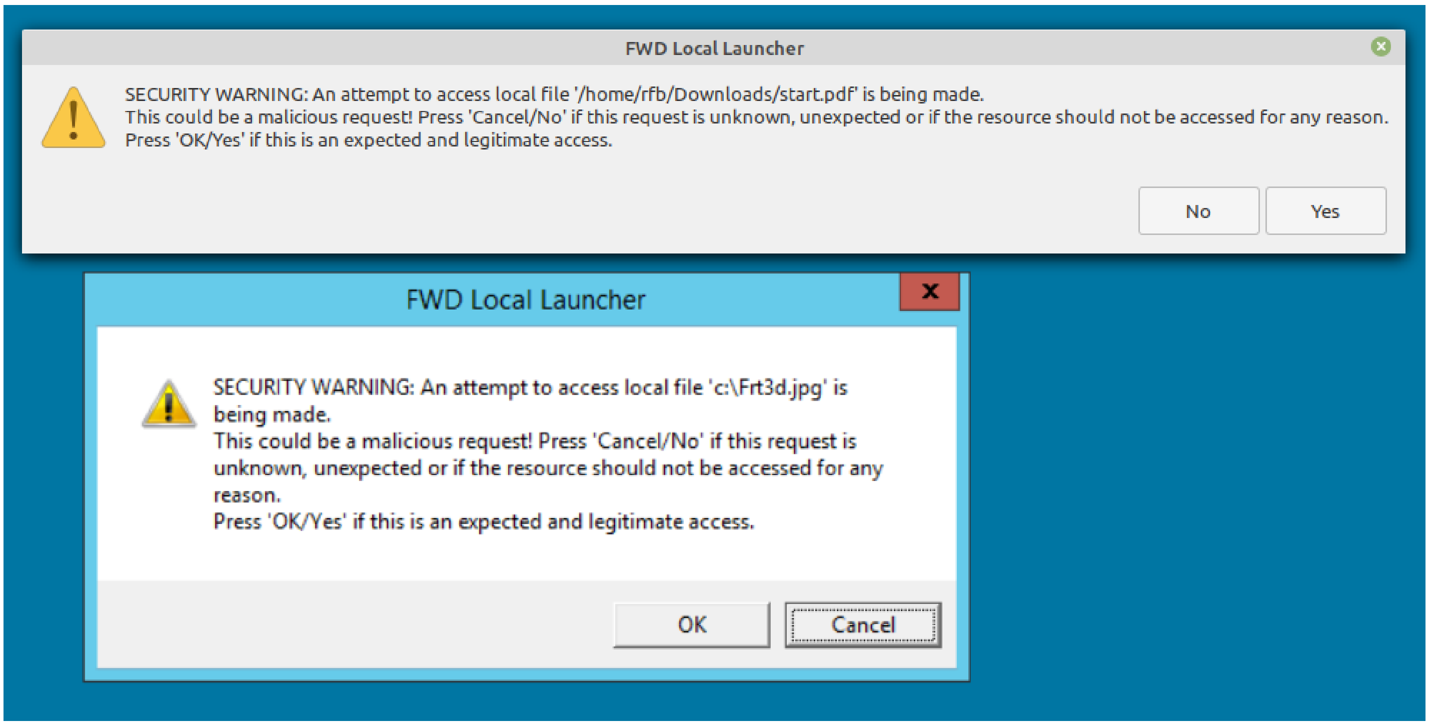
In regard to the warning:

- The yellow warning icon is good.
- Please change the text of the warning to this: "SECURITY WARNING: A page in your web browser is trying to access the local file '%s'. This could be a malicious request! Press 'Cancel' if this request is unknown, unexpected or if the resource should not be accessed for any reason. Press 'OK' if this is an expected and legitimate access."

#113 - 03/22/2021 01:00 PM - Roger Borrello

- File 4179_both_os_warning_messages.png added

Here are the 2 (Linux/Windows). I had to merge some text, since I don't have as much control over the zenity tool:


FWD Local Launcher

SECURITY WARNING: An attempt to access local file '/home/rfb/Downloads/start.pdf' is being made.
This could be a malicious request! Press 'Cancel/No' if this request is unknown, unexpected or if the resource should not be accessed for any reason.
Press 'OK/Yes' if this is an expected and legitimate access.

FWD Local Launcher

SECURITY WARNING: An attempt to access local file 'c:\Frt3d.jpg' is being made.
This could be a malicious request! Press 'Cancel/No' if this request is unknown, unexpected or if the resource should not be accessed for any reason.
Press 'OK/Yes' if this is an expected and legitimate access.

#114 - 03/22/2021 01:37 PM - Greg Shah

It works for me.

#115 - 03/22/2021 02:26 PM - Roger Borrello

Updates in 3821c_12165

#116 - 03/22/2021 03:31 PM - Greg Shah

Code Review Task Branch 3821c Revision 12165

1. I'm not a fan of this zenity dependency, but I think writing the X11 code directly is way more code than we would want to write at this time. This is especially true because it is unlikely this launcher will be used on Linux. So leave this alone for now.

2. I don't think we should put the message functions and messages in FWD's native layer since I never expect to call that code from FWD directly. Consider that the Swing client is by nature a fat client and is expected to potentially access the local disk. So there is no need for this messaging from the Swing client. And the web client doesn't call this code directly either. On the other hand, making the interface to that code to be platform neutral is a good thing. Please move this message code/messages into the launcher or into some platform specific files that are only linked into the launcher. Doing that will reduce the unnecessary code from libp2j which is better from a security and readability perspective.

#117 - 03/22/2021 06:42 PM - Roger Borrello

Updates in 12167. Created launcher_xxx modules to support the launcher, removed JNI code from desktop_xxx modules.

#118 - 03/23/2021 02:40 PM - Greg Shah

Code Review Task Branch 3821c Revisions 12167 and 12169

I'm good with the changes.

#119 - 03/23/2021 02:40 PM - Greg Shah

- Status changed from Review to Closed

Files

GuiWebDriver.patch	4.08 KB	03/16/2021	Roger Borrello
4179_launcher_take1.mkv	572 KB	03/16/2021	Roger Borrello
4719_firefox_preferences1.png	59.1 KB	03/18/2021	Roger Borrello
4179_demo_of_message_boxes_on_windows.mkv	328 KB	03/22/2021	Roger Borrello
4179_error_message.png	18.5 KB	03/22/2021	Roger Borrello
4179_warning_message.png	17.2 KB	03/22/2021	Roger Borrello
4179_both_os_warning_messages.png	69.3 KB	03/22/2021	Roger Borrello