**BugReport ID="2"**

**Title: "(449596) Firefox - remove the browser.sessionstore.enabledpref "**

While this has worked somehow for Tab Mix Plus, we've had several issues with people ending up both Session Restore and Tab Mix Plus disabled (see bug 435055 and its duplicates).

Furthermore, there are several code points which will also break when Session Restore has been disabled (such as the list of recently closed tabs). Instead of adding try-catch-blocks wherever we use Session Restore. This would also make the lives of those other extension authors simpler who so far can't be too sure that the Session Store component actually works (through whatever implementation). Note that privacy concerned users will still be able to disable writing to sessionstore.js through the browser.sessionstore.resume\_from\_crash pref. Created an attachment (id=332726) [details]. Remove the pref (note: bug 448725 should be wont fixed if this is fixed). Created an attachment (id=333820) [details]. Remove (buggy) note from API comments. (From update of attachment 332726 [details]) , a problem with this patch is that the session data is still stored in memory while the app is running, and by removing this pref, there's no way to disable that. some users do not want the recently-closed-tabs menu, and others don't want any session tracks stored in memory at all. &gt; some users do not want the recently-closed-tabs menu,I'd rather introduce a different pref or different means to cater the privacy sensitive users than have this half-baked cut-it-all pref which AFAICT so far has produced more issues than it's solved. &gt; Then again, we don't save any data that wouldn't be in memory, anyway, or do we? Fair point. I'm not sure. &gt; I'd rather introduce a different pref or different means to cater the privacy sensitive users than have this half-baked cut-it-all pref which AFAICT so far has produced more issues than it's solved. Yes, agreed the pref is not ideal for this purpose.One men: This patch significantly affects Tab Mix Plus: Instead of disabling SessionStore, you'll now have to replace it by shipping a component which implements the same API. You should be able to keep that component minimal, though, and just call your own code whenever the API is used. This will make the lives of people using the SessionStore API (such as Session Manager) somewhat simpler, as they can just use the API without having to worry too much about the implementation behind it. If you want to offer the option of switching between your and our implementation, see e.g. the ignore-history component of the Torbutton extension for how to overwrite a component with the original still being available internally (so that you can either pass API calls forward or handle them yourself). Sounds like this has impact for extension authors and so should be documented on MDC no problem. currentTabmix dev-build already not disable SessionStore. I currently have only one problem , how to disable the restore after restart. can you add a pref for this, or some other way to do it?

**BugReport ID="3"**

**Title :"(491925) Firefox - Disable multitouch \"rotate\" gesture for cycling tabs"**

The basic reasoning is that it's highly disruptive to be switched to another tab when you're not expecting it. When it happens, you don't know what just happened until you notice that you're on some entirely different page, that's randomly to the left or right (1 or more tabs) from the page you thought you were on. I'm don't think the rotate gesture for switching tabs is nearly as useful, discoverable, or a good fit as the other gestures are.Just play around with browser. gesture.twist.\* in about:config as a temporary workaround. Justin, the odd thing here is that I find I trigger zoom more frequently than tab rotate ... far far more frequently.I think I've never accidentally switched tab because of rotate gesture. Actually I'd probably decrease the default threshold for tab switching. I've triggered zoom repeatedly too; usually I fix it right away but a quick browse though content-prefs.sqlite shows ~20 sites with zoom settings that I didn't know were set.Around that point, my fingers want to spread or join, and that ends up read as a pinch. I guess that's part of why I'm not really fond of this gesture, and would just as soon turn it off. I'll try running with a threshold of 40, but I suspect there's an unavoidable conflict between making it easy to use on purpose, yet hard to invoke accidentally.I'm right handed and I position my index finger at 7oclock and ring finger at 1oclock. To rotate right I end up at 8/2oclocks.

The first time I came into contact with tab switching via gesture was accidentally.I wasn't sure what happened, I just appeared on another tab and couldn't figure out why. The tab-switching gesture right now isn't obvious and feels unexpected when I hit it. Mardak: the way you've written this code, disabling the gesture is a pref-flip, right? Blocking here for decision and mulling, will probably take it though. Yup, should be able to just unset the default value for the twist prefs: -pref(\"browser.gesture.twist.right\", \"Browser:NextTab\"); -pref(\"browser.gesture.twist.left\", \"Browser:PrevTab\"); Would setting the value to empty-string (\"\") work, or something similar?But Rob says such hardware/drivers doesn't exist today, so it shouldn't be a common problem. We should keep on eye out for problems with issues on touchscreens, andt from Rob demoing it on his touchscreen it would appear harder to trigger.This bug is just rotate, but do we want to consider pinching too? If we take those out, we'll have swipe up/down as jump to top/bottom and swipe left/right as back/forward by default &gt; This bug is just rotate, but do we want to consider pinching too? Maybe; it certainly has been noted that people are accidentally hitting that gesture too. Though I think it's a separate decision: rotate is an unnatural gesture for switching tabs (nothing else uses it this way), with a high penalty for accidentally triggering.So, I'd strongly lean towards only removing the rotate.It's also nearly impossible to trigger without a definitive pinch with two fingers on the screen. If you want to consider disabling pinch, please file a separate bug, do not conflate. FWIW, I'm not nearly as keen on disabling that one, and am fairly-to-surely likely to WONTFIX such a bug.But go ahead ... try me ; Created an attachment (id=376850) Don't do anything on os x when twisting. Alternatively, have people tried upping the threshold for twist? It would make it less likely to accidentally trigger. But I just noticed something interesting in os x's gesture detection. - If I use 2 hands (1 finger each) and place one at the top and one at the bottom, moving the top finger registers as a normal mouse move while moving the bottom finger is registered as a twist.I suppose the heuristic there is to assume the top finger is used for scrolling.- Also, if your fingers are too close to the edge, it won't treat it as a rotate. But you can still move the pointer from the edge.

**BugReport ID="7"**

**Title :"(437797) Firefox - about:mozilla page is hardcoded to LTR in RC2"**

i also think the quote source should be on the left side. perhaps the page should take CSS from intl.css, where localisers can override some of the classes or add a class to set direction. Actually I would like not to add yet another entry in intl.css but to fix it in aboutMozilla.xhtml. Pike, can we think about it?We can, actually, support RTL using something like &lt;bodydir=&locale.dir;&gt;. "Created an attachment (id=324147) [details] patch: Here is my patch.Tested on Hebrew locale. Please let me know if you prefer 'body{direction:&locale.dir;}' instead of the current '&lt;bodydir=\"&locale.dir;\"&gt;'. the W3C prefers markup and not CSS: <http://www.w3.org/International/questions/qa-bidi-css-markup>.**(**In reply to comment #6) &gt; the W3C prefers markup and not CSS:Actually I'm not sure how we can do it using CSS. I have no idea if something like 'body{direction:&locale.dir;;}' is a valid.Sounds fine to me, I guess.do we really need to worry about the \"RTL locale that's falling back to en-US\" though? I deally we could just take this patch and not worry about having localizers update various different attributes based on localization state, and just rely on them completing all the localization work to make it look decent. In other words, isn't bug 427029 an edge case given the nature of about:robots and it's \"optional\" status for localizers? Seems like we could just live with it not looking perfect for RTL locales that haven't localized it rather than adding complexity. I'm adding another screenshot to bug 427029. Axel, Gavin - You are welcome to comment on that issue right there. We are not far from 1.9.0.1 deadline. Please approve1.9.0.1.Thank you. Please get this landed on mozilla-central before getting approval for 1.9.0.x **(**From update of attachment 324147 [details]) Please re-request approval after getting this landed on mozilla-central. I'm not really sold on taking this given that it's more \"cosmetic\" than anything else...

**BugReport ID="9"**

**Title :"(328600) Firefox - Option for mailto: links to go to a web based e-mail client (webmail service)"**

The idea is that if one prefers to use a web based e-mail account, they should have the option to configure firefox to log into that web based e-mail account and open a compose new message window. I understand this may require some colaboration with the popular web based e-mail services such as gmail and aim mail, but I think it would be worth the benifit. Reproducible: Always Something like the WebmailCompose extension? It's sad that mailto: links don't work for most webmail users, especially since webmail is so popular. Not going to block on this, would be good to have this generically for feeds as well, but this may have to wait for Gecko 1.9.

**BugReport ID="32"**

**Title: "(238215) Firefox - Add a history window (like in Seamonkey)"**

It's more convenient to have a single history window that doesn't take up screen real estate than having to open the history in whichever window I'm currently browsing. I really hate sidebars and would want to be able to disable them in place of these windows before I would migrate to Firefox. Since the roadmap is now to focus on the standalone apps with \"a conservative,sustaining engineering fashion\" type of support for Mozilla/SeaMonkey, I think a \"SeaMonkey Parity\" of some form should be established. Thanks. while i'm brainstorming, you could probably even add a search shortcut ability so you could type something like about:history? Plus, I'd like to be able to open this easily, such as with /-\\H. Please add the Mozilla history window in Firefox! I think this is just one of the things that's missing in inferior browsers like Microsoft Internet Explorer. The Mozilla history window is just perfect! (In reply to comment #3), &gt; Not all of this is needed but at least the first four are helpful. Firefox sure doesn't seem like a total product with a decent history. EHM, Enhanced History Manager, is an extension that does it nicely. <http://forums.mozillazine.org/viewtopic.php?p=1749502#1749502>. I agree that the firefox history is limited at best. I'm amazed that there is no way of knowing what the url or date of a history entry is. Mozilla history was excellent since you could re-trace your browsing using the timestamp and find collections of pages you'd visited. History (in life and browsing) is more useful when there is a context. Shouldn't there be an option to see when and what order a page was visited?

**BugReport ID="1"**

**Title: "(495584) Firefox - search suggestions passes wrong previous result to form history"**

This bug wasn't visible before 469443, because nsFormFillController::StartSearch tries to QI the provided result to a nsIAutoCompleteSimpleResult. The search-suggestion result is only implements nsIAutoCompletResult (no \"Simple\"), so the QI fails, historyResult nee previousResult becomes null, and thus Satchel was doing a new search every time. form history finds 1 entry (\"blah\"), search-suggestions finds \"baaa\", \"bloop\", \"bzzz\", the autocompete menu shows these in order with a divider between \"blah\" and \"baaa\". Created an attachment (id=380567) [details] Patch v.1 (WIP) This fixes the problem, but isn't quite correct... If you type \"a&lt;backspace&gt;b\", satchel trying to use the results from the \"a\" search for the \"b\" search, and so nothing is found. That seems to be handled somewhere else for normal form field entries, oddly enough. Created an attachment (id=383211) [details] Patch v.2 Ah. Otherwise it's trying to use a old form history result that no longer applies for the search string. (From update of attachment 383211 [details]). Perhaps we should rename one of them to \_fhResult just to reduce confusion? &gt; (From update of attachment 383211 [details] [details]) &gt; Perhaps we should rename one of them to \_fhResult just to reduce confusion? Good point. I renamed the one in the wrapper to \_formHistResult. fhResult seemed maybe a bit too short.Pushed <http://hg.mozilla.org/mozilla-central/rev/097598383614>

**BugReport ID="10"**

**Title :"(156905) GIMP - GimpAspectPreview doesn\'t respect the layer offset and can crash"**

Open a new image,create a layer much bigger than the image, and with a different aspect ratio.

This is due to the fact that the selection mask is taken with gimp\_drawable\_get\_thumbnail\_data () in the preview code. So if the layer and the image have different size, they are streched differently. On a big image, use crop tool on right, bottom or right bottom, click on resize. The active picture is then little than the layer. Using any tool with preview give a one pixel (or less ;) ) preview instead of the good one. Compute intersection between layer and active picture could be the solution. As far as I can see, there are two reasonable options, and either of them will require a substantial amount of fixing in the aspect preview code and the plugins that use it.

Fixing this might make API changes necessary. We need to look into this as soon as possible. It would help a lot if someone could attach an example XCF file to this bug report that shows the problem. Created an attachment (id=89354) [details] bug156905.xcf.bz2 - a layer bigger than image. Following assertion failed using Channel mixer on the layer.

The example image has dimensions 377\*233, the selection starts at x=125, y=97 and extends to the lower right corner of the image. If there are no objections, I could commit the patch.

Please commit this. It doesn't make sense to patch 2 [details].4 at this point. So please do this in trunk only.

2008-09-02 Aurimas JuÅ¡ka &lt;aurisj@svn.gnome.org&gt;

&gt;-------\* libgimp/gimpzoompreview.c (gimp\_zoom\_preview\_draw\_buffer),

&gt;-------(gimp\_zoom\_preview\_set\_drawable): fixed GimpZoomPreview to handle

&gt;-------layers with offsets and display intersection of layer and selection

&gt;-------in the preview. (bug #156905)

**BugReport ID="11"**

**Title : "(164995) GIMP - Remember last scale method setting"**

It would be handy that the scale dialog just remembered the last setting and always present that one. I didn't know about the preferences setting until I saw this bug report.

We would need an entry whether to save the last used setting, for the users who don't want this. Michael, I don't agree with having an extra entry whether to save the setting yes or no. This would clutter the interface even more. Furthermore I think it's not really a problem if you have to change the setting now and then (In your case you also have to click once more to not save the setting). And most of the times you will not change this setting. I agree that it is convenient to be able to set a default value from the place it is used (e.g., the Image-&gt;Scale dialog) instead of having to open the Preferences and find the corresponding option. This is very useful for those who are not familiar with all options available in the Preferences. But from time to time, I scale up a bitmap image without interpolation. I don't want this to become the new default and give me bad results when I scale up another image two days later. This could be a button like \"set default\" or a checkbox \"remember this setting\" next to the \"Interpolation type\" dropdown list. &gt;&gt;Jaap is right. LOL, me too. I think the suggestion for a \"set as defaults\" button here is a good all-round solution that satisfies both camps. See this bug for request to deal with scaling units (percent/pixel) in the same way. <http://bugzilla.gnome.org/show_bug.cgi?id=357424>. I think a \"set as defaults\" button would clutter the UI. This change remember the settings across dialog invocations. Closing as FIXED. (Please reopen if you strongly disagree, but please not without a non-ugly suggestion for a GUI) 2006-10-14 Michael Natterer &lt;mitch@gimp.org&gt; Remember the settings in the scale and resize dialogs for images and layers. Fixes bug #164995, bug #316392, and bug #357424.

**BugReport ID="12"**

**Title : "(170801) GIMP - Converting image from grayscale to black&white is painfully slow"**

For an example image of \"2460\*3483\", it takes 34 seconds on my computer (while applying the Threshold tool, which basically does the same, only needs maybe 2 seconds). This slow speed is not acceptable for interactive image processing, and this slowness is not necessary at all. Indeed it is very slow, so I will confirm this bug report. There is, however, a workaround. I found with a little experimentation that if you use the \"Optimal Palette\" option for step 4.1, then the conversion occurs in a flash, literally a fraction of a second for me. This happens, I think, because if you use this option, by the time the algorithm reaches the stage of actually converting the image, it has built a color histogram and \"knows\" that there are exactly two colors in the image, and can assign colors directly, without having to test whether black or white is the best match for each pixel. Because of the existence of this workaround, I am going to lower the severity, but this \*is\* something that should be looked into. Created an attachment (id=38911) [details]. Because this is a different issue, I am going to open a different bug report for it -- using your image as example, if you don't mind. . Still just as slow for RGB images, though. Committed to both branches: 2005-03-18 Bill Skaggs &lt;weskaggs@primate.ucdavis.edu&gt; \* app/core/gimpimage-convert.c: when converting grayscale to mono, treat the palette as gray rather than rgb, giving more than tenfold speedup. Fixes bug #170801. The new problem is described in bug #170825. Thank you for fixing this bug so fast, so I just need to wait for the next release. (Building for Windows would be rather messy, so I try to be patient...) . I'm interested in seeing the patch.

). I expect it to yield poorer results though, to some extent (probably most obviously darker, or lighter, in mid-greys, that is a damaged gamma), because the greyscale colour-matching and error-quantifying is in a comparatively non-perceptually-uniform space (that's half the reason why it's faster, though it's not as big a quality difference in a greyscale source as an RGB source). Were side-by-side comparisons done? I hope so, though I could do some here if anyone cares. I don't think anyone has done side-by-side comparisons. differences, but not so strong that I'm really going to lose any sleep. Note that the analysis in comment #1 is correct; the 'optimal' palette finder does a first pass to see if there are already a suitable number of colours, and then knows that it can skip both the expensive quantization and dithering stages. The 'mono' palette option doesn't even bother to start this pre-pass because it could only possibly pay off the extra effort if the entire image is pure black and pure white, which is expected to be a comparatively rare occurance. &gt; I don't think that this operation is so rare, because for getting an \"optimal distribution between black and white\" for grayscale images (for example: you want to generate faximilies the receiver actually is able to read), you use the; \"threshold\" operation, and then a \"convert to 1bit\" operation to actually adjust the internal memory requirements. So at least for me, this chain of operations is rather frequent. &gt; and then a \"convert to 1bit\" operation to actually adjust the internal memory requirements. . For having 10 or 20 fax pages in one OOo document, it quite makes a difference wether they are 8bit or 1bit.

**BugReport ID="13"**

**Title : "(364852) GIMP - screenshot plug-in incomplete for Windows platform"**

There is no option to \"Select a Region to Grab\" when the user chooses \"Screen Shot.\" However, this option exists in the dialogue box that appears after the user selects :Screenshot.\" However, the functionality does not work, even after selecting a long delay period. Instead, the whole screen is captured each time. I was expecting the mouse cursor onscreen to change to \"crosshairs\" or something similar to allow me to select the region I required. Steps to reproduce:

Does this happen every time? Yes. Other information: (I presume the two entries for screenshots under the submenu are there in order to test old and new code, by the way but I just noted it in case there was an issue.); D. Le Sage; APF; Hobart, Tasmania. What's this bug-report about? Do you want to contribute the missing bits or do you want us to disable the plug-in for the Windows platform? We are waiting for a volunteer to implement the missing functionality for more than a year now. Perhaps it's about time to drop support for the Win32 platform completely. The bug report was to alert you in case you did not know it was broken. You are obviously aware it is incomplete. Perhaps you should disable it to prevent more questions from users. I am not a programmer so I cannot implement the functionality. I am trying to do my part by reporting issues for you. You have complained before about how short of volunteers you are. Of course there are incomplete and even broken features in there. If you are just a user, you should be using the stable release. . Fixes bug #364852. \* plug-ins/common/mkgen.pl: reverted addition of 'extralibs' parameter that was introduced for the screenshot plug-in only.