Awesometab

A proposal to up the "awesome" level of Instantbird as part of GSoC 2013.

Personal Details

- Name: Nihanth Subramanya
- Email: nhnt11@gmail.com
- Telephone: +919561130163
- Other contact methods: IRC: nhnt11@irc.mozilla.org | GTalk: nhnt11
- Country of residence: India
- Timezone: Indian Standard Time (UTC+05:30)
- Primary language: English

Project Proposal

I propose to develop a "smart" user interface, in the form of a user-openable tab, that allows the user to quickly and efficiently open new chats - based on who they chat with most often, and on what accounts they do so. The tab will auto-complete a simple form which specifies the other party and the account through which the chat will be started. It will also have a textfield which will allow the user to start typing a contact's details (name, account, etc.), auto-completing it and allowing the user to instantly start the chat. This will replace the current "join chat" dialog.

The user should be able to go from actively chatting in one conversation to doing so in a new one, losing no time in finding the contact to start the conversation with, refocusing to the correct UI element, or opening extra windows.

Also if time permits, I propose to develop a system for saving and restoring chat sessions - that is, Instantbird, on opening, should restore all chats that were in session when it last quit. This may be made optional to the user.

Schedule of Deliverables

So far, I have built Instantbird from source and worked on a few bugs (my Bugzilla contributions). I have also been teaching myself a bit of XUL and going through the codebase to find out where and how different functionality has been implemented.

Before the coding period begins, I intend to further work on my XUL and Javascript skills, and further familiarise myself with the codebase and other parts of the Instantbird project. I intend to do this by working on more bugs, and perhaps develop a UI prototype for the project as an add-on (this could also save time later). Another thing I intend to do is research more about the various data that different servers provide so that as many fields as possible can be auto-completed. I will also be looking into Firefox's Awesomebar implementation to see if I can use it as a reference for any part of the project.

The following is a tentative to-do list that I think will be suitable for the project timeframe. Of course, one must keep in mind that things can come up during the project that may cause delays (or early completions).

1. Development of the UI

    a. The UI must be intuitive. The aim is to have a way for the user to jump into a chat without unnecessary delay.

    b. The UI at this stage will not be populated with any data, just the outline/design with the required UI elements.

    c. A textfield (similar in functionality to Firefox's Awesomebar) will be focused upon opening Awesometab. When the user starts typing a contact name, a channel, etc., it will be auto-completed. When a contact or chatroom is finally selected, any other required fields (for e.g. channel password on IRC) will be displayed. Pressing Enter will start the conversation.

    d. **The textfield is only convenient when the user knows what he is looking for.** In addition to it, a UI similar in functionality to the current Join Chat dialog will be displayed, through which the user can manually select an account, etc. and start a conversation. This will be displayed while the textfield is blank.

    e. This is estimated to take a week's time, though the UI may need to be continually tweaked during the project.

2. Storage of statistical data

    a. Storing/reading the data must be convenient (for example as JSON objects)

    b. Data stored must be sorted based on user activity. A ranking system must be introduced to decide the order in which auto-complete data is populated. This system will assign a score to each contact, which will be based on a variety of parameters, including:

i. how often he/she is contacted,

ii. how recent the previous chat with him/her was, and

iii. total number of messages exchanged.

iv. Apart from these factors, a "Favourites" tag may be introduced, as a way of giving the user some control over the system. Auto-joined chatrooms will also be given priority.

c. The data handler should do all the work for storing and reading history. The Awesome***tab*** itself must have as little code as possible related to statistical data - it should be able to obtain a pre-sorted list from the handler and populate.

d. On opening a chat, the statistics for that contact/account must be updated on the fly (and not on quit) to prevent loss of data in case of a crash.

e. In addition to user statistics, data for auto-completable fields will also be obtained from the server. For example, IRC channels and public chatroom lists. This data will be similarly stored by the handler, and will be included in the list it returns when asked.

g. Any duplicate data will be purged. For example, a frequently visited IRC channel may appear once as a ranked item, and again (redundantly) in the channel list.

h. This is estimated to take 2 to 3 weeks, keeping in mind that the ranking system will need to be perfected.

3. Integration into Instantbird

a. Data must be saved on the fly as mentioned

b. Awesometab should obtain data and populate. Also, a mechanism must be added such that the populated data updates itself as the user types, to best match what he/she is typing as well as preserve the ranking system to the maximum extent.

c. Since the main goal is to provide a smooth, fluid transition from one chat to the next, keyboard shortcuts must be introduced. Currently I have in mind the following:

i. Press ctrl+t to open awesometab. The auto-completing

textfield will be focused.

ii. Type first few letters of contact if required, till the desired contact is selected.

iii. Press enter to open the new conversation, with focus ready at the message input textfield.

d. Estimated to take 1 to 2 weeks.

4. Bugfixing and unit tests

a. Real world testing and bug-fixing will be done. A few laymen may be asked to help with testing. Any improvements/ modifications required (to the ranking system in particular) will happen during this time.

b. Wherever required (perhaps in data storage/ranking/ autocompletion), unit tests will be written to exhaustively test as many cases as possible to ensure all bugs are caught.

c. Estimated time: 2 weeks

5. Session restore

a. If the previous objectives are completed faster than expected, a session restore feature will be implemented as earlier described. If time does not permit this, I will be happy to do it outside of GSoC later on.

b. Along with session restore, any other features/improvements which improve user experience that I can think of - particularly pertaining to navigation within the application, which is the main goal of the project - will be implemented in the remaining time.

6. Finalising

a. In the last week or so, all code will be reviewed to ensure compatibility in function, code formatting, building, and all other aspects so that the project can be successfully merged into the Instantbird tree, for all users to enjoy.

I have no other commitments during GSoC that I currently know of, and will be able to concentrate fully on the project.

Open Source Development Experience

I was an active contributor to CyanogenMod - a community distribution of the Android operating system. I did quite a bit of work on bringing native resolution graphics to low pixel density (LDPI) devices, and added a few UI improvements. I also added quite a few modifications to help improve theme support, and fixed a couple of bugs. Recently, I've been involved with OpenSEMC - an effort to bring the latest version of Android (through CyanogenMod 10.1) to the Sony Xperia S, which I own.

Apart from this, I usually upload the source code of any small project I do on my Github, mainly just to show my support for Open Source.

Though my previous open source experience has been nearly exclusively in the Android world, I'm very eager to try something new. I've already been fixing a few bugs for the Instantbird team, and can't wait to do something bigger.

My profile on XDA-Developers

My CyanogenMod contributions

My Bugzilla contributions

My Github (many of the projects here are old, from when I was a beginner)

Work/Internship Experience

I've never been professionally employed or taken any internships.

Academic Experience

I'm studying Electrical and Electronics Engineering (EEE) at BITS Pilani - KK Birla Goa Campus. BITS Pilani is a highly acclaimed institute in India (acceptance rate less than 1.47%). I opted for Computer Science as my first choice, but unfortunately my score in the entrance exam was a few marks short and I did not qualify. My second choice was EEE because I'd done electronics in high school - it interests me and it has ties with CS. I've only finished one semester, but in that semester my grade point average was above the 85th percentile of our year.

Why Me

I think user experience is a fundamental part of any personal computer software and love working on things to improve it. Awesometab immediately caught my eye as something that was almost obvious to have - I myself have instinctively have pressed Ctrl+T (expecting a new tab) many times while using Instantbird.

I learn fast and adapt easily to new situations (languages, styles of programming, and so on). I started programming a few years ago with HTML and JavaScript, quickly moving on to Java. Over time I've done bits of other languages as well, depending on what the project required. Recently I've been using JavaScript again, fixing a couple of bugs for Instantbird.

In addition, I've always been very interested in instant messaging in general, as well as protocols. One of the first apps I ever wrote was a simple server-client chat application that used its own simple protocol (though at the time, I'm not sure if I even knew what protocol meant!). More recently, for fun, I wrote an IRC bot which lets you remotely control your PC by sending it specific commands - learning quite a bit about IRC protocol along the way. Both projects are on my Github (NChat and biIRC). I acknowledge that this is not extremely relevant to the awesometab project specifically, but I want to emphasise that I have some idea of how instant messaging works.

Most of all, I think it's very important to contribute to projects that you actually use yourself, because that gives you all the more motivation and inspiration to work on them. I personally use Instantbird and want this feature as much as anyone else, and this makes me itch to start coding away.

Why Mozilla

I've always respected Mozilla for its openness. I think it's a great organisation with lots of brilliant people, bringing out excellent software that I personally use and enjoy contributing to.