

Writing Ayttn Plugins

Philip S Tellis
Yahoo! Inc

Foss.in/2007 • December 4-8, 2007 • Bangalore, India

Chatters in the crowd, raise your hands!

What's the best part about chatting with friends?

What can you do to enhance the experience?

Two parts to Instant Messaging

- Chatting - one on one or in groups
- Everything else

Everything else must be pretty huge

- File transfers
- Encryption
- Language translation
- Smiley themes
- Audio/Video?

Everything else must be pretty huge

- File transfers
- Encryption
- Language translation
- Smiley themes
- Audio/Video?

Everything else must be pretty huge

- File transfers
- Encryption with GPG
- Language translation
- Smiley themes
- Audio/Video?

Everything else must be pretty huge

- File transfers
- Encryption with GPG
- Language translation
- Smiley themes
- Audio/Video?

Everything else must be pretty huge

- File transfers
- Encryption with GPG
- 14n9u493 7r4n5l47i0n
- Smiley themes
- Audio/Video?

Everything else must be pretty huge

- File transfers
- Encryption with GPG
- 14n9u493 7r4n5l47i0n
- Smiley themes
- Audio/Video?

Everything else must be pretty huge

- File transfers
- Encryption with GPG
- 14n9u493 7r4n5l47i0n
- Smiley themes
- Audio/Video?

Everything else must be pretty huge

- File transfers
- Encryption with GPG
- 14n9u493 7r4n5l47i0n
- Smiley themes
- Audio/Video - Perhaps not.

But simple chat is no small matter either

- How many service providers?
- Different features in each
- Some features are clearly beyond simple chatting
- What about chatting over non-chat channels?

But simple chat is no small matter either

- How many service providers?
- Different features in each
- Some features are clearly beyond simple chatting
- What about chatting over non-chat channels?

But simple chat is no small matter either

- How many service providers?
- Different features in each
- Some features are clearly beyond simple chatting
- What about chatting over non-chat channels?

But simple chat is no small matter either

- How many service providers?
- Different features in each
- Some features are clearly beyond simple chatting
- What about chatting over non-chat channels?

But simple chat is no small matter either

- How many service providers?
- Different features in each
- Some features are clearly beyond simple chatting
- What about chatting over non-chat channels?

But simple chat is no small matter either

- How many service providers?
- Different features in each
- Some features are clearly beyond simple chatting
- What about chatting over non-chat channels - SMTP/SMS/Blog posting

Enter plugins

- The primary development team shouldn't have to worry about too much

- Build the core, and the most popular modules
- Make it possible for others to extend this

Ayttm's plugin mechanism

- Introduced in 2001
- Based on libtool and dynamically loaded modules at runtime
- 3 types of modules
- Must implement a predefined API
- Must export certain symbols

Let's get down and dirty and build one

- [Linkify Bugzilla links](#)
- [Weather module](#)
- [Image converter](#)

1. Linkify Bugzilla links

1 Linkify Bugzilla links

- Change Bug XXXXX into `Bug XXXXX`
- Do this before sending a message and before printing a message to screen
- Do not double convert

1a The module skeleton

```
#include "plugin_api.h"

#define plugin_info bugzilla_LTX_plugin_info
#define module_version bugzilla_LTX_module_version

static int ref_count = 0;

static int plugin_init();
static int plugin_finish();

PLUGIN_INFO plugin_info = {
    ...
};

unsigned int module_version() { return CORE_VERSION; }

...
```

1b Module signature

```
PLUGIN_INFO plugin_info = {
```

```

    PLUGIN_FILTER,
    "Bugzilla linker",
    "Linkify bugzilla links",
    "$Revision: $",
    "$Date: $",
    &ref_count,
    plugin_init,
    plugin_finish,
    reload_prefs,
    NULL
};

```

1cOnload handler

```

static int plugin_init()
{
    input_list *il = calloc(1, sizeof(input_list));
    plugin_info.prefs = il;

    il->widget.checkbox.value = &enable_bugzilla_links;
    il->name = "enable_bugzilla_links";
    il->label = _("Enable automatic Bugzilla Linking");
    il->type = EB_INPUT_CHECKBOX;

    il->next = calloc(1, sizeof(input_list));
    il = il->next;
    il->widget.entry.value = &bugzilla_url;
    il->name = "bugzilla_url";
    il->label = _("Base Bugzilla URL:");
    il->type = EB_INPUT_ENTRY;

    eb_debug(DBG_MOD, "Bugzilla initialised\n");

    outgoing_message_filters = l_list_prepend(outgoing_message_filters
        &bugzilla_linkify);
    incoming_message_filters = l_list_append(incoming_message_filters,
        &bugzilla_linkify);

    return 0;
}

```

1dOnUnload handler

```

static int plugin_finish()
{
    eb_debug(DBG_MOD, "Auto-bugzilla shutting down\n");
    outgoing_message_filters = l_list_remove(outgoing_message_filters,
        &bugzilla_linkify);
    incoming_message_filters = l_list_remove(incoming_message_filters,
        &bugzilla_linkify);

    while(plugin_info.prefs) {
        input_list *il = plugin_info.prefs->next;
        free(plugin_info.prefs);
        plugin_info.prefs = il;
    }
}

```

```

    return 0;
}

```

1eFilter signatures

```

static char *bugzilla_linkify(const eb_local_account * local,
                              const eb_account * remote,
                              const struct contact *contact,
                              const char * s);

```

1fAttaching filters

```

outgoing_message_filters = l_list_prepend(outgoing_message_filters,
                                           &pre_out_filter_function);

outgoing_message_filters = l_list_append(outgoing_message_filters,
                                          &post_out_filter_function);

incoming_message_filters = l_list_append(incoming_message_filters,
                                          &in_filter_function);

```

1gDoing the do

```

static char *bugzilla_linkify(const eb_local_account * local,
                              const eb_account * remote,
                              const struct contact *contact,
                              const char * s)
{
    char *p;
    if (!enable_bugzilla_links) {
        return strdup(s);
    }

    /* Now we need to do something like this: */
    /* s/\[(bug:? )(\d+)\]/[<a href="$bugzilla_url/$2">$1$2</a>]/g */

    return p;
}

```

2. Weather module

2Weather module

- Search messages for keyword
 - Fetch Weather for specified city
 - Display weather to local and remote users
-

Use the source! `utility/weather.c`

3. Image converter

3Image converter

- Provide an API to convert images from one format to another
 - We'll use * <-> JPG2000 conversion
-

Use the source! `image_filter/img2jpc.c`

Getting your code back into ayttm

- You may want to have your code distributed through ayttm's repository
- You can still manage your own packages and releases separate from ayttm

It's fairly easy to get commit access

1. Submit 3 or 4 good patches, OR
2. Submit a good module of general usefulness
3. Stick to ayttm's coding standards
4. Stick to ayttm's release guidelines
5. If you still don't get it, ask for commit access

Pointers or References?

- [Ayttn](#)
- ayttm-users@lists.sourceforge.net

Greetz

Philip S Tellis

philip@bluesmoon.info

<http://bluesmoon.info/>

Made with Eric A Meyer's S5