

MessageHelper

```
FS.MessageHelper = [];
```

FS.MessageHelper

Item

```
{
  executable : function(msg){},
  action : function(msg){}
}
```

executabletruefalseaction

MessageHelper

```
/**
 *
 */
public class MessageHelper {

    private static Map<String, MessageDataAccessObject> map = new ConcurrentHashMap<String,
MessageDataAccessObject>();

    public static int getMessageDataAccessObjectCount() {
        return map.size();
    }

    /**
     *
     * @param mark
     * @param dao
     */
    public static void registerMessageDataAccessObject(String mark, MessageDataAccessObject dao) {
        map.put(mark, dao);
    }

    public static MessageDataAccessObject getMessageDataAccessObject(String mark) {
        return map.get(mark);
    }

    /**
     *
     * @param mark
     * @param message
     * @return id
     * @throws Exception
     */
    public static long save(String mark, Message message) throws Exception {
        return getMessageDataAccessObject(mark).save(message);
    }
}
```

```

*
* @param mark
* @param id id
* @throws Exception
*/
public static void updateToasted(String mark, long id) throws Exception {
    getMessageDataAccessObject(mark).updateToasted(id);
}

/**
* id
* @param mark
* @param id id
* @return
* @throws Exception
*/
public static Message findById(String mark, long id) throws Exception {
    return getMessageDataAccessObject(mark).findById(id);
}

/**
* id
* @param mark
* @param id id
* @return
* @throws Exception
*/
public static boolean deleteByID(String mark, long id) throws Exception {
    return getMessageDataAccessObject(mark).deleteByID(id);
}

/**
*
* @param userID ID
* @throws Exception
*/
public static void deleteAll(long userID) throws Exception {
    for (String category : map.keySet()) {
        MessageDataAccessObject messageDataAccessObject = getMessageDataAccessObject(category);
        if (messageDataAccessObject != null) {
            messageDataAccessObject.deleteByID(userID);
        }
    }
}

/**
*
*
* @param username
* @return
* @throws Exception
*/
public static List<Message> getMessages(String username) throws Exception {
    List<Message> messages = new ArrayList<Message>();
    for (String category : map.keySet()) {
        MessageDataAccessObject messageDataAccessObject = getMessageDataAccessObject(category);
        if (messageDataAccessObject != null) {
            messages.addAll(messageDataAccessObject.getMessages(username));
        }
    }
    return messages;
}

/**
*
*
* @param username
* @return
* @throws Exception
*/
public static List<Message> getFreshMessages(String username) throws Exception {

```

```

        List<Message> messages = new ArrayList<Message>();
        for (String category : map.keySet()) {
            MessageDataAccessObject messageDataAccessObject = getMessageDataAccessObject(category);
            if (messageDataAccessObject != null) {
                messages.addAll(messageDataAccessObject.getFreshMessages(username));
            }
        }
        return messages;
    }
}

/**
 * JSON
 * @param username
 * @return
 * @throws Exception JSON
 */
public static JSONObject getMessageJSONObject(String username) throws Exception {
    List<Message> messages = MessageHelper.getMessages(username);
    List<Message> freshMessages = MessageHelper.getFreshMessages(username);
    JSONObject jo = new JSONObject();
    JSONArray toasted = new JSONArray();
    for (Message message : messages) {
        toasted.put(message.createJSONObject());
    }
    jo.put("toasted", toasted);

    JSONArray fresh = new JSONArray();
    for (Message message : freshMessages) {
        fresh.put(message.createJSONObject());
    }
    jo.put("fresh", fresh);
    return jo;
}
}

```

FSPlateMessageHelper

```

(function(){
    var openMessage = function(msg) {
        FS.tabPane.addItem({
            title : "" + msg.id,
            src : msg.url
        });
    };
    var executable = function(msg) {
        return msg.username != null;
    };
    FS.MessageHelper.push({
        executable : executable,
        action : openMessage
    });
})(jQuery);

```

<https://git.oschina.net/fanruan/plugins-free/tree/master/plugin-message>