Guide to Submitting Bugs and Feature Requests

Last Updated: 1/4/2023 8:03 AM

Contents

BUGS

Identifying and Submitting a Bug 2

When you find a Bug with an open Status (e.g., Status "Verifying" through "Ready for Production", and ***not*** "Closed") 2

When you find a Bug with a Closed Status 2

How to write a New Bug 3

FEATURE REQUESTS

Customer Directly Requests a New Feature 5

Technician's Assesment of an Issue or Bug report leads to a Feature Request 5

How to write a new Feature Request 6

BUGS

When we discover an issue with MedInformatix software, we attempt to resolve it using our collective knowledge and capabilities via MI Support, Product, and Development.

Identifying and Submitting a Bug

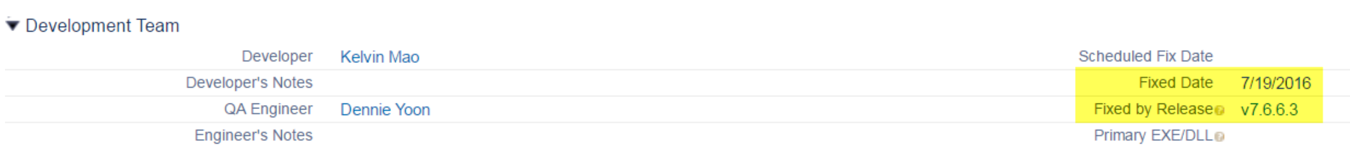
1. Obtain detailed information from the customer about how to replicate the problem.
    - You may record the session with the customer if it has many steps to document. This will help you replicate the problem.
    - Debug logs captured while the problem occurs are often helpful.
2. Attempt to replicate the problem using your internal *release* servers.
3. If you can replicate this error:
    - Search MIForce for any Bugs with an open status that are similar or the same as the issue you are investigating.
    - Review the different lists of Bugs in the Bugs View

When you find a Bug with an open Status (e.g., Status "Verifying" through "Ready for Production", and ***not*** "Closed")

1. If you are sure this is the same issue the customer is reporting, associate the Bug number with your own internal support ticketing system, if you have one.
2. Monitor releases for notification of the Bug being fixed.

When you find a Bug with a Closed Status

1. Attach the Bug to the Case.
2. Review the Bug Close Description section. This field will let you know why the Bug was closed and which version/build contains the fix for this Bug if it's "Closed – Fixed".



3. If the Fixed Release version is greater than the current customer MI Version, then you can proceed with the update process following normal update protocols.
4. If the Bug was closed as "Closed - Not A Bug", review the Bug Close Reason and Chatter for details.

BUGS (cont'd)

How to write a New Bug

If you don't find a Bug with an open or closed status, please create a **new Bug**. Complete the following fields:

Focus Area	Problem	Priority
Focus Sub Area	Problem Description	Severity
MI Release	Steps Taken to Generate Bug	

1. **Focus Area and Focus Sub Area:** Select the most appropriate choices based on the issue you are investigating.
2. **MI Release:** Select the version the customer has when reporting the issue. This does not indicate the version the issue appeared in.
3. **Problem:** A short and meaningful description of the issue.
4. **Problem Description:** The description should be concise and to the point. Avoid "Telling A Story". Use a numbered bullet list to describe the elements of the issue, when appropriate. Include the following sections (see samples below):
   a. **BACKGROUND:**
   {describe what users expect and/or what lead to this being reported as a Bug}
   b. ISSUE:
   {describe the issue in detail}
   c. CAUSE:
   {optional. As best you can, describe the cause. Use observation and debug logs to assist}
   d. PROPOSED SOLUTION:
   (optional. Use only when you've clearly identified the issue and have a proposed solution. You do not need to include this if you don't know or must guess.)
5. **Steps Taken to Generate Bug:** Always include the steps you used to generate the Bug, using numbers to indicate the steps.
6. **Priority:** Since it is a required field, select a choice; however, this will ultimately be set by Product.
7. **Severity:** The Severity level should the severity, as it pertains to the customer.
8. Be sure you attach supportive documentation in Chatter such as:
   a. Screenshots
   b. Video
   c. Debug File(s)
9. "Follow" the Bug to be notified of changes.
10. If you need a status on a Bug, please post to the Chatter Feed in the Bug to the Product team (@MI Product). The Product Manager will reply.

BUGS (cont'd)

EXAMPLE

Bug Detail

Edit | Delete | Clone

Bug #	BUG-002163	Owner	Chris Bruns [Change]
1 Focus Area	Main Application	Status	Closed - Fixed
Focus Sub Area	Collections	6 Priority	P4 - Low
2 MI Release	v7.6.6.3	7 Severity	Minor

▼ Bug Description

3 Problem: Tickler View: Unable to Delete Ticklers

4 Problem Description:

BACKGROUND: The MedInformatix client includes the abilty to manage Ticklers, including deleting them.

ISSUE 1:
The Workgroup Profile setting **Delete Tickler** is not read correctly in the Tickler View, and thus users can't delete ticklers from these screen, regardless of the setting being enabled or disabled.

CAUSE:
From a debug file, the workgroup is defined incorrectly when reading the Delete Tickler setting
07/26/16 07:49:23 GetProfileFromWorkgroup(workgroup:, section:Security, entry:Delete Tickler, default:0, value:0, length:2)

ISSUE 2:
Ticklers can be deleted from Collection Notes -> Ticklers regardless of the setting.

CAUSE:
Undetermined. Debug file shows it reading the setting, but deletion is still allowed.

ISSUE 3 (LEGACY, DISCOVERED BY QA): In both the Tickler View and #TICKLER screen, deleting ticklers does not take the TCODE into account, which can cause multiple ticklers due on the same day for the same day to be deleted. This is a legacy issue predating v7.6. The likelihood of this occurring is generally low, and thus it has never been identified "in the wild".

5 Steps Taken to Generate Bug:

1. Enable deletion of ticklers in the Collection section of the Workgroup Profile.
2. Open the Tickler View.
3. Attempt to delete a tickler.
4. Access is denied (due to the incorrect reading of the setting as noted above)

FEATURE REQUESTS

Customers will suggest enhancements to the MedInformatix EHR software and the rest of the product suite. Enhancements are considered Feature Requests. There are two scenarios for creating a Feature Request:

1. A customer may call or e-mail, directly requesting a new feature.
2. A support technician may determine that a Case is not a Bug, but rather a Feature Request.

Customer Directly Requests a New Feature

a. Determine if the request is valid.
   - Is there an alternative solution to the customer's request? If you are not certain, always ask a question internally or to the MICentral VAR Group first.
   - Does the request makes sense and is it feasible? You will need to do your best and use your MedInformatix knowledge on the module(s) you support to evaluate the workflow and impact this may have on the software across the entire spectrum of users.

b. Before adding the Feature Request, explain to the customer what doing so means.
   i. A Feature Request will be submitted to the MedInformatix Product team for evaluation.
      - If it is approved, it will be released in the future version.
      - The Product team will Deny the Feature Request if it is determined it will not be implemented.
      - Feature Requests will be reviewed by the Product team on a periodic basis, depending on current priorities across the breadth of sources for Feature Requests.

   ii. It is important the customer understands that we do not guarantee the Feature Request will be approved, and that the unsolicited direct customer request is treated as a suggestion.

Technician's Assesment of an Issue or Bug report leads to a Feature Request

a. There may be scenarios where a customer calls with a problem, calling it a ″Bug″, when it is actually a workflow issue or lack of specific functionality.

   - Ask yourself is this is a Bug or a Feature Request. Developers target a specific workflow when writing coding and when a feature request is placed, they have to evaluate how this request can affect the application.
   - For example, if the customer describes a workflow to post payments without having to ever use the mouse, which does not exist in MedInformatix, and they call it a "Bug", as a support technician you are to evaluate this carefully and take the opportunity to let them know this will be considered a Feature Request.

FEATURE REQUESTS (cont'd)

How to write a new Feature Request

Before submitting a new Feature Request, search for any existing Feature Requests that may be the same or similar.

- If you find an open Feature Request, attach it to the Case and close the Case. If you don't find one, proceed to create a new Feature Request. You must complete the following fields:

  1. **Feature Request:** A brief and meaningful description of the request.
  2. **Focus Area and Sub Focus Area:** Select the most appropriate areas.
  3. **Feature Descripton:** The description should be concise and to the point. Avoid "Telling A Story". Use a numbered bullet list to describe the elements of the Feature Request, when appropriate. Include the following sections (see samples below):

     a. **BACKGROUND:**
     {describe what users expect and/or what lead to this Feature Request}
     b. **REQUEST:**
     {describe the desired functionality in detail}
     c. **USER STORY:**
     {as best you can, write the need from the user's perspective}
        i. Format: As a {type of user}, I can {describe the action}, so that {describe the benefit}
        ii. You can write multiple User Stories if there is more than one perspective
     d. **ACCEPTANCE CRITERIA:**
     {as best you can, describe the expected behavior, using numerical bullet points for multiple criteria}
     e. **NOTES:**
     {optional – include any relevant notes about the request}

- Upon completion, associate the Feature Request with your Support Case, as applicable
- Close the ticket.
  a. Why? The Feature Request may or may not be implemented. The MedInformatix Product team will process the requests and will update the status as needed.
  b. Be sure to follow the FR to stay abreast of changes.
  c. If the request is approved, the Product team will release it in a future version. The Release Notes will inform you of this.

FEATURE REQUESTS (cont'd)

EXAMPLE

Request #	FR-001719	Owner	Chris Bruns [Change]
1 Feature Request Name	Differentiate flagged and moved appointments	Request Origin	MI-CAB
2 Focus Area	Main Application	Request Type	Enhancement to Existing Feature
Focus Sub Area	Scheduling	Priority	
Included in MI Release	v7.7.0.0	Target Release Date	
Status	In QA Final	Release Date	

▼ Feature Request Information

Bug		Account	
QA Engineer	Dennie Yoon	Contact	
Developer	Kelvin Mao	Related Feature Request	
Developer's Notes			

▼ Testing Steps

3 Feature Description

BACKGROUND:
Flagging appointments and moving appointments both have the same Flag of (R)escheduled.

REQUEST:
Differentiate the flag status between moving an appointment and flagging/rescheduling.

USER STORY:
As a Supervisor, I want to know how an appointment is rescheduled so that I can tell the difference between moved and flagged appointments. Appointments may be flagged multiple times and moved multiple times so when patients themselves or referring providers question what has transpired, I can tell them exactly how and when each change occurred.

ACCEPTANCE CRITERIA:

1. If the coverage type changes, I can verify a user intentionally modified or input incorrect data that cleared during flagging.
2. Flagging and Last Flagged sometimes deletes or inserts incorrectly (e.g., referring provider). User error is often assumed but I can better troubleshoot if a pattern emerges between Moved and Flagged appointments.
3. Move vs Flag clues me into user behavior since Moving only occurs in Books View