Resubmitting:

All the figures highlighted in orange are newly added to this document. References to the changes are as indicated below:

[1] Better explains the code with this sentence.

[2] Explains the sequence diagram, and points out how BibEntry and EntryTypeView is connected to insertEntry.

[3] We moved this part from the supporting material to this document.

[4] The previous sentence was confusing and hence modified.

Feature: Add a new article

Add a new article (a single entry) using the “+” button.

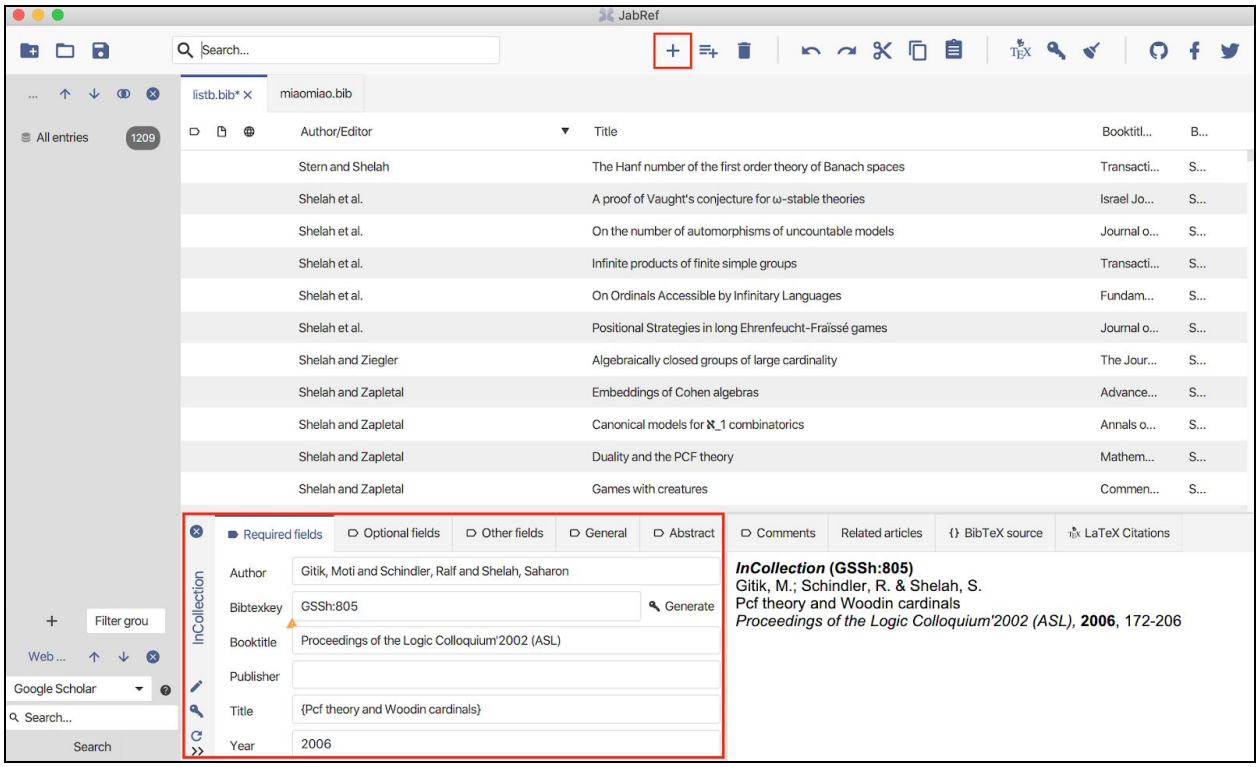

Figure (1) : Entry Editor feature

Reason: As a citation and reference management tool, JabRef helps users collect and organize resources in research. Adding a new entry with the type of article is very important as it keeps track of different research articles which the users are utilizing.

Components:

1. **JabRefFrame.java** `package org.jabref.gui;`

This class initializes the main window of the JabRef application. Inside the createToolbar() method, the tool bar which includes the plus sign button gets created. Specifically, the createIconButton() method of ActionFactory.java takes in this NEW_ARTCILE action and corresponding command, which in this case is a NewEntryAction.

```
HBox rightSide = new HBox(
       factory.createIconButton(StandardActions.NEW_ARTICLE, new
NewEntryAction(this, StandardEntryType.Article, dialogService,
Globals.prefs, stateManager)), ....)
```

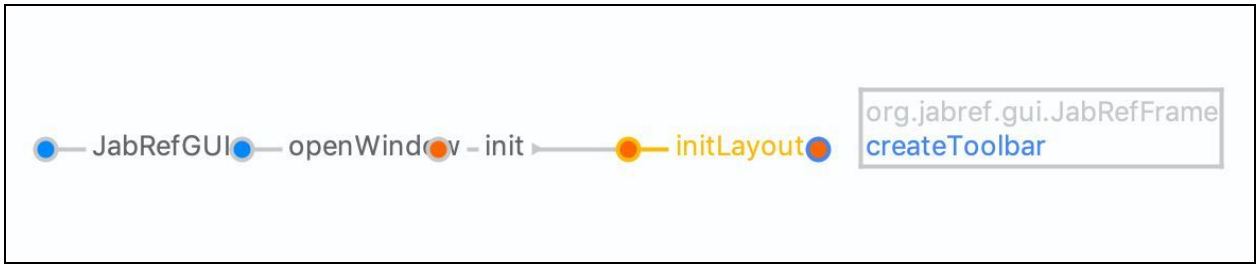

Figure (2) : createToolBar (Upstream)

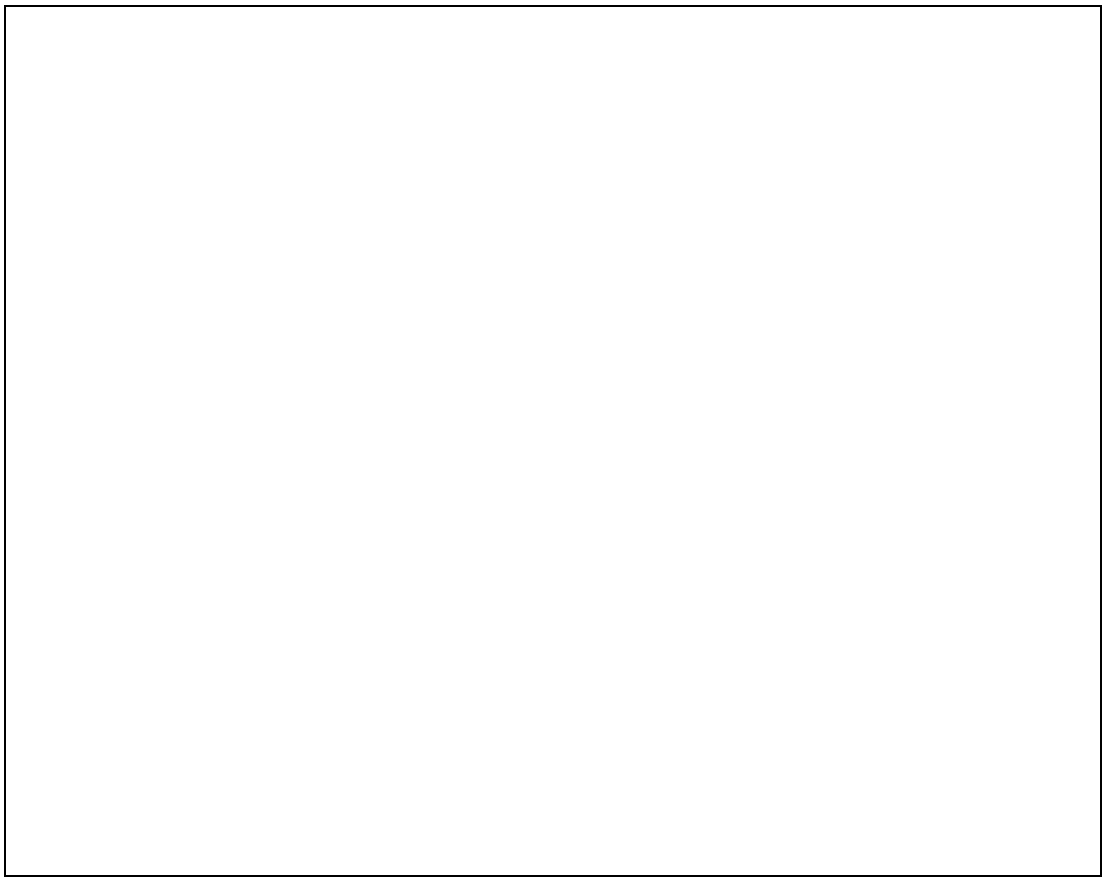

Figure (3) : createToolBar sequence diagram

On further exploring the downstream call graph for the createToolbar, one can locate the
createIconButton which is necessary to create the plus icon.

2. **NewEntryAction.java** `(package org.jabref.gui.importer)`
   When users click the plus sign, the execute() method of NewEntryAction.java gets called subsequently. There can be many types of entries such as articles, books, journals, etc. With the present “article” type, the current JabRefFrame gets the current BasePanel, and executes the function of inserting new bib entry. The core function in execute() is insertEntry(), which calls specific functions for creating a new entry.[1]

```
@Override
public void execute() {
   if (jabRefFrame.getBasePanelCount() <= 0) {
       LOGGER.error("Action 'New entry' must be disabled when no
database is open.");
       return;
   }
   if (type.isPresent()) {
      jabRefFrame.getCurrentBasePanel().insertEntry(new
BibEntry(type.get()));
   } else {
       EntryTypeView typeChoiceDialog = new
EntryTypeView(jabRefFrame.getCurrentBasePanel(), dialogService,
preferences);
       EntryType selectedType =
typeChoiceDialog.showAndWait().orElse(null);
       if (selectedType == null) {
           return;
       }
       trackNewEntry(selectedType);
      jabRefFrame.getCurrentBasePanel().insertEntry(new
BibEntry(selectedType));
   }
}
```

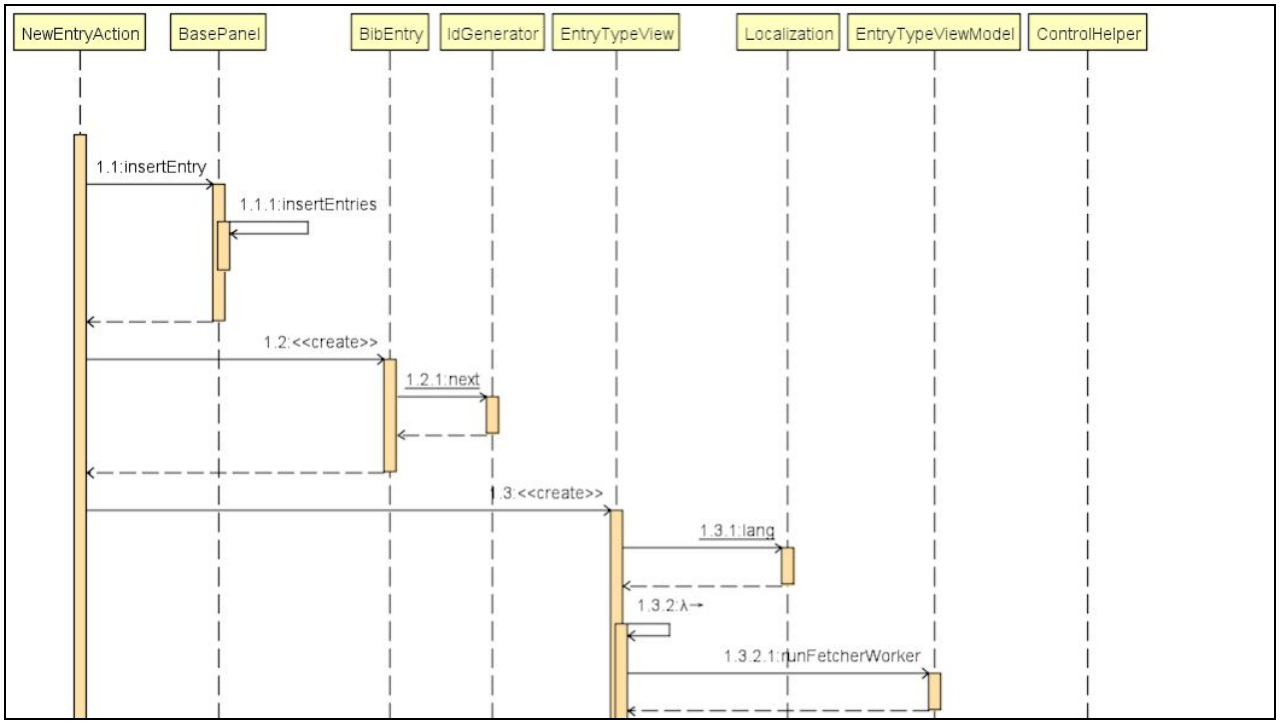

Figure (4) : sequence diagram of insertEntry()

As figure (4) shows, insertEntry will call functions in BibEntry and EntryTypeView classes, which are related with database, to insert a new entry.[2]

3. **BasePanel.java** (`package org.jabref.gui`)
   The following methods get called immediately.

```
public void insertEntry(final BibEntry bibEntry) {
   if (bibEntry != null) {
       insertEntries(Collections.singletonList(bibEntry));
   }
}
```

```
public void insertEntries(final List<BibEntry> entries) {
   if (!entries.isEmpty()) {
       try {
           bibDatabaseContext.getDatabase().insertEntries(entries);

           // Set owner and timestamp
           for (BibEntry entry : entries) {
               UpdateField.setAutomaticFields(entry, true, true,
Globals.prefs.getUpdateFieldPreferences());
```

```
            }
            // Create an UndoableInsertEntries object.
            getUndoManager().addEdit(new
UndoableInsertEntries(bibDatabaseContext.getDatabase(), entries));

            markBaseChanged(); // The database just changed.
            if
(Globals.prefs.getBoolean(JabRefPreferences.AUTO_OPEN_FORM)) {
                showAndEdit(entries.get(0));
            }
            clearAndSelect(entries.get(0));
        } catch (KeyCollisionException ex) {
            LOGGER.info("Collision for bibtex key" + ex.getId(), ex);
        }
    }
}
```

insertEntries() is one of the downstream calls for execute(), which as the name suggests is what we are looking for. When we analyzed this, we noticed that paste functions usually lead to similar insertEntries calls from various parts of the program. Figure (5) shows the details of those calls.[3]

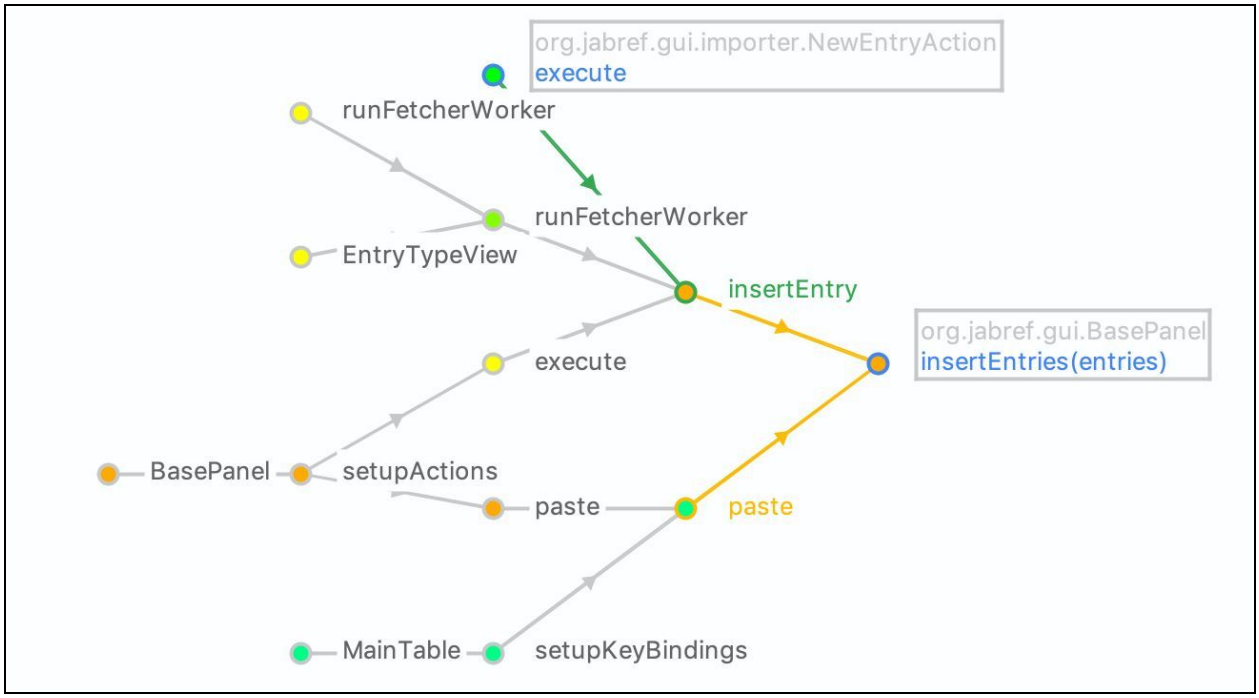

Figure (5) : insertEntries (Upstream)

4. **BibDatabaseContext.java** (`package org.jabref.model.database`)
   On the database side, this new entry is also inserted into the current database with the help of BibDatabaseContext, which gets the database object and has it inserting the new entry.

5. **BibDatabase.java** (`package org.jabref.model.database`)
   It is noticed that in this code block, eventbus's post method gets fired and posts a new event of EntriesAddedEvent. MainTable, on the other side, subscribes to the eventbus. Every time we add a new entry into the bib database, MainTable's listen method will be invoked.

```
public synchronized void insertEntries(List<BibEntry> newEntries,
EntriesEventSource eventSource) throws KeyCollisionException {
   Objects.requireNonNull(newEntries);
   for (BibEntry entry : newEntries) {
       String id = entry.getId();
       if (containsEntryWithId(id)) {
           throw new KeyCollisionException("ID is already in use,
please choose another", id);
       }

       internalIDs.add(id);
       entry.registerListener(this);
   }
   if (newEntries.isEmpty()) {
       eventBus.post(new EntriesAddedEvent(newEntries, eventSource));
   } else {
       eventBus.post(new EntriesAddedEvent(newEntries,
newEntries.get(0), eventSource));
   }
   entries.addAll(newEntries);
}
```

6. **Maintable.java** (`package org.jabref.gui.maintable`)

```
@Subscribe
public void listen(EntriesAddedEvent event) {
```

```
   DefaultTaskExecutor.runInJavaFXThread(() ->
clearAndSelect(event.getFirstEntry()));
}
```

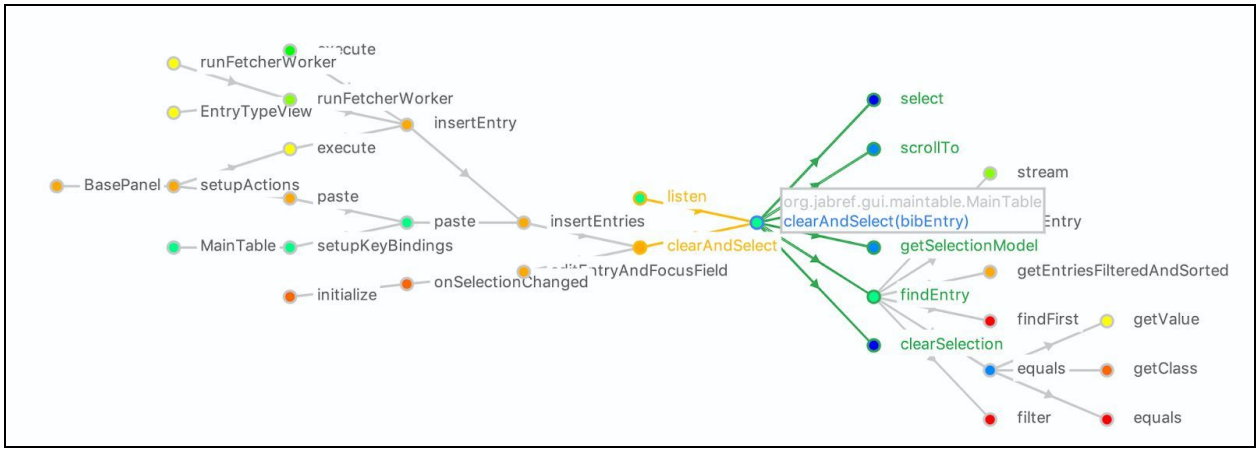

Figure (6) : clearAndSelect

This clears the highlight of the currently selected row and highlights the currently editing row for the new entry.

```
public void clearAndSelect(BibEntry bibEntry) {
   findEntry(bibEntry).ifPresent(entry -> {
       getSelectionModel().clearSelection();
       getSelectionModel().select(entry);
       scrollTo(entry);
   });
}
```

6. **EntryEditorTab.java** (`package org.jabref.gui.entryeditor`)
When a new entry is created, the entry editor tab becomes focused.[4]

```
/**
* The tab just got the focus. Override this method if you want to
perform a special action on focus (like selecting
* the first field in the editor)
*/
protected void handleFocus() {
   // Do nothing by default
}
```

HandleFocus can be invoked in a lot of ways, as noticed there are no functions called through this method, one can modify this method to get focus and accomplish any task.

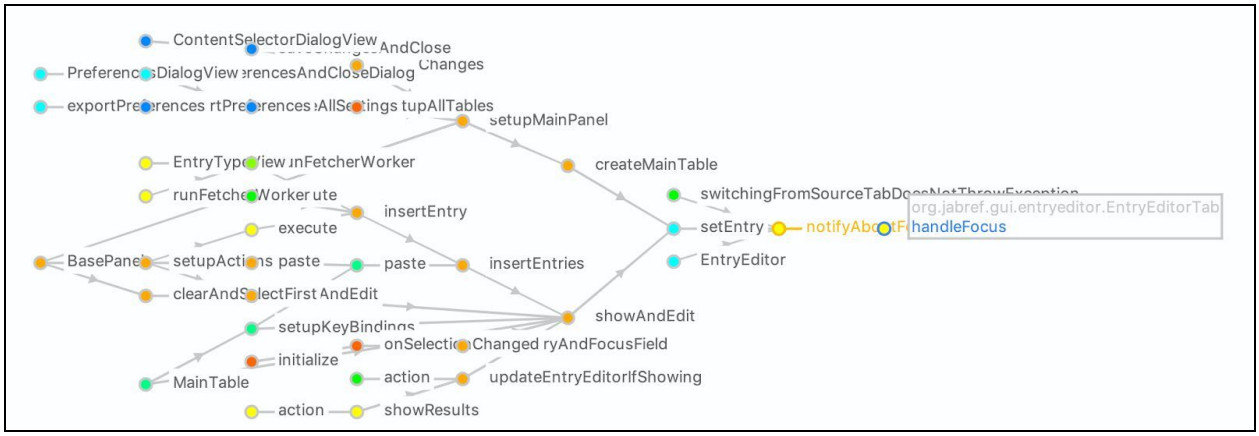

Figure (7) : handleFocus