TicketManagementSystem 5000®

USER and TECHNICAL GUIDE

I. Using TicketManagementSystem 5000®

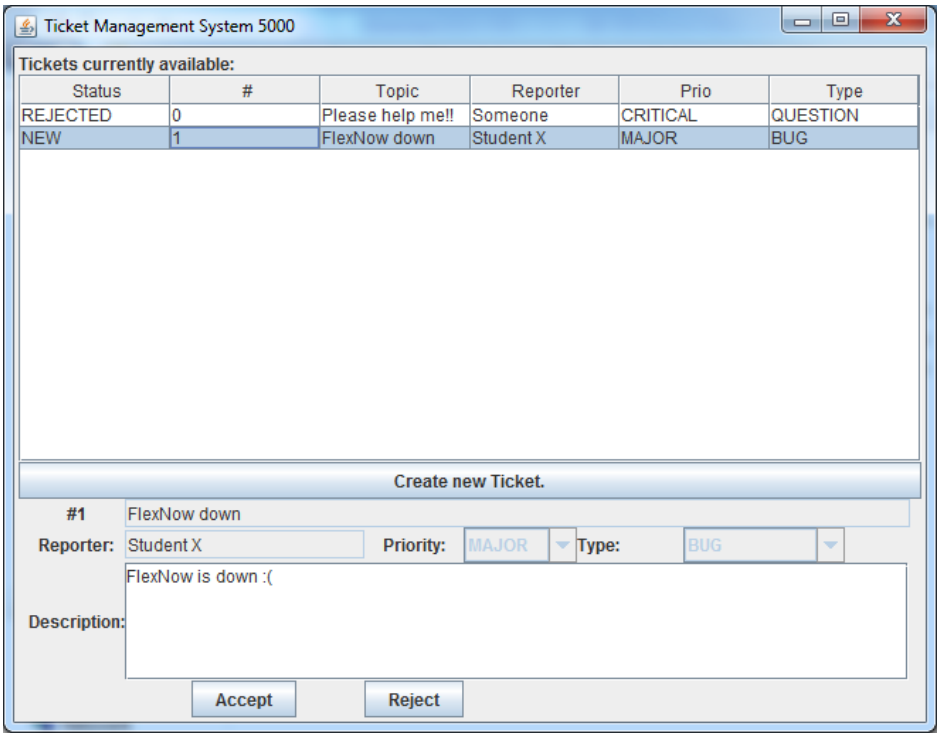
A. Starting the Tool

Run `TicketManagementSystem5000.jar` – either by double-click or by calling

```
java –jar TicketManagementSystem5000.jar
```

B. Using the GUI

As TicketManagementSystem 5000® is developed by professionals in cooperation with respected HCI experts the tool is self-explanatory:



Just hit the Buttons to Create, Save, Accept and Reject some tickets!

II. Extension Guide

A. Tool Architecture

Figure 1 presents the relevant overall basic architecture of the tool as a class diagram.

The two most important parts are the class `GlobalAppController` and the interface `TicketManagement`.

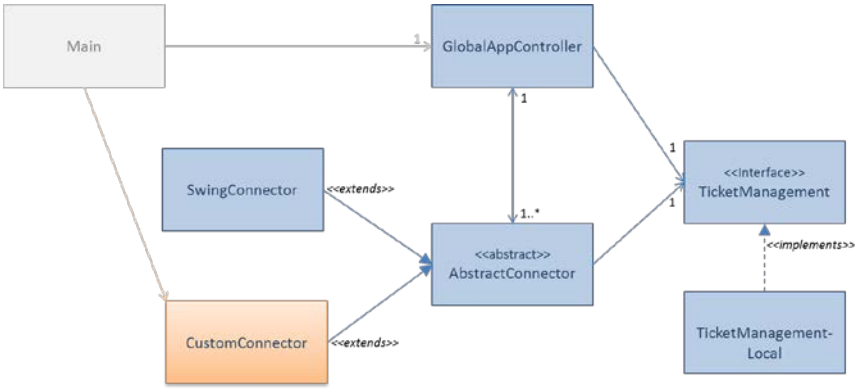


Fig. 1 Basic archicture of TicketManagementSystem 5000®

The `TicketManagement` interface defines all allowed methods to create and modify Tickets in the System. It is implemented by the class `TicketManagementLocal` which stores Tickets in-memory.

An overview of the functionality of `TicketManagement` gives the Javadoc listing below:

Method Summary

All Methods | Instance Methods | Abstract Methods

Modifier and Type	Method and Description
`Ticket`	**`acceptTicket`**`(int id)` Method to accept a Ticket, i.e., changing the **`Status`** to `Status.ACCEPTED` Throws an exception if this status change is not possible (i.e., the current status is not `Status.NEW`) or if the `id` refers to a **`Ticket`** that does not exist.
`Ticket`	**`closeTicket`**`(int id)` Method to close a Ticket, i.e., changing the **`Status`** to `Status.CLOSED` Throws an exception if this status change is not possible (i.e., the current status is not `Status.ACCEPTED`) or if the `id` refers to a **`Ticket`** that does not exist.
`Ticket`	**`createNewTicket`**`(Ticket newTicket)` Method to create a new Ticket - Data stored in the `newTicket` will be used to create a new Ticket which will be stored internally.
`java.util.List<Ticket>`	**`getAllTickets`**`()` Returns a list of **`Ticket`**s currently available in the system.
`Ticket`	**`rejectTicket`**`(int id)` Method to reject a Ticket, i.e., changing the **`Status`** to `Status.REJECTED` Throws an exception if this status change is not possible (i.e., the current status is not `Status.NEW`) or if the `id` refers to a **`Ticket`** that does not exist.

The `GlobalAppController` manages the creation of the `TicketManagement` implementation and provides the functionality to extend the application by adding "Connectors" which can use the `TicketManagement` to create and modify Tickets.

The default Connector is the `SwingConnector` which is created and started automatically when an instance of `GlobalAppController` is created. `SwingConnector` is the connection between the default Swing GUI and the application internals.

B. Extending TicketManagementSystem 5000®

If you want extend TicketManagementSystem 5000 you can add `CustomConnectors`.

The following steps must be fulfilled to extend the system:

- Write your own `CustomConnector` which "extends" the abstract Java class `AbstractConnector`.
- You can use the reference to the `TicketManagement` inherited from `AbstractConnector` to access the internal implementation.
- Implement all abstract methods (`run()` and `shutdownConnector()`)
- Register your Connector in the system: Create a new `main()`-Method which creates an instance of `GlobalAppController` and register an instance of your `CustomConnector` by calling the method `addConnector(AbstractConnector newConn)`.

For more technical details check the API documentation provided!