

GUI II: Events, Dialogs and MVC

Making buttons work, reporting errors, and separating Model, View and Controller

Lecture + Lab 6

Branch: week9-gui-events-mvc

Learning outcome 4

What you will be able to do after today

- Attach **listeners** so buttons run your code.
- Show messages and errors with `JOptionPane`.
- Explain **MVC** and map it onto the LMS packages.
- Keep the model free of `System.out` and `javax.swing`.

1. Event-driven programming

A console program decides what happens next. A GUI program **waits**. The user clicks, types or selects, and Swing calls *your* code. Three ingredients:

Ingredient	In the LMS
Event source	the "Borrow" <code>JButton</code>
Listener	an <code>ActionListener</code> : code to run when clicked
Registration	<code>borrowButton.addActionListener(...)</code>

```
borrowButton.addActionListener(e -> onBorrow()); // lambda: "when clicked, run onBorrow()"
```

`ActionListener` is an **interface** with one method (Week 4 again!). A lambda is a short way to implement it.

2. A complete handler

```
private void onBorrow() {
    String itemId = selectedItemId(); // from the selected table row
    if (itemId == null) return;
    String memberId = ((String) memberBox.getSelectedItem()).split(" - ")[0];
    try {
        controller.borrow(itemId, memberId);
        setStatus(itemId + " borrowed by " + memberId);
        refreshTable();
    } catch (ItemNotAvailableException ex) { // checked: Week 5 pays off
        showError("Cannot borrow", ex.getMessage());
    } catch (MemberNotFoundException ex) {
        showError("Unknown member", ex.getMessage());
    }
}

private void showError(String title, String message) {
    JOptionPane.showMessageDialog(this, message, title, JOptionPane.WARNING_MESSAGE);
}
```

The exceptions we designed in Week 5 become **friendly dialogs**. This is why we threw them instead of printing.

Dialog	Call
Information / warning / error	<code>JOptionPane.showMessageDialog(parent, msg, title, type)</code>
OK / Cancel	<code>JOptionPane.showConfirmDialog(...)</code>
Ask for text	<code>JOptionPane.showInputDialog(...)</code>

Library Management System

New item

Type: Book

ID:

Title:

Detail:

Add Item

ID	Title	Type	Detail	Status
B001	Clean Code	Book	Robert C. Mart...	On loan: Nade...
B002	Refactoring	Book	Martin Fowler	Available
D001	Introduction to Algorithms	DVD	120 min	Available
G001	Software Weekly	Magazine	Issue 42	Available

B001 borrowed by M001

Member: M001 - Nadeesha Perera Borrow Return

Library Management System

New item

Type: Book

ID:

Title:

Detail:

Add Item

ID	Title	Type	Detail	Status
B001	Clean Code	Book	Robert C. Mart...	On loan: Nade...
B002	Refactoring	Book	Martin Fowler	Available
D001	Introduction to Algorithms	DVD	120 min	Available
G001	Software Weekly	Magazine	Issue 42	Available



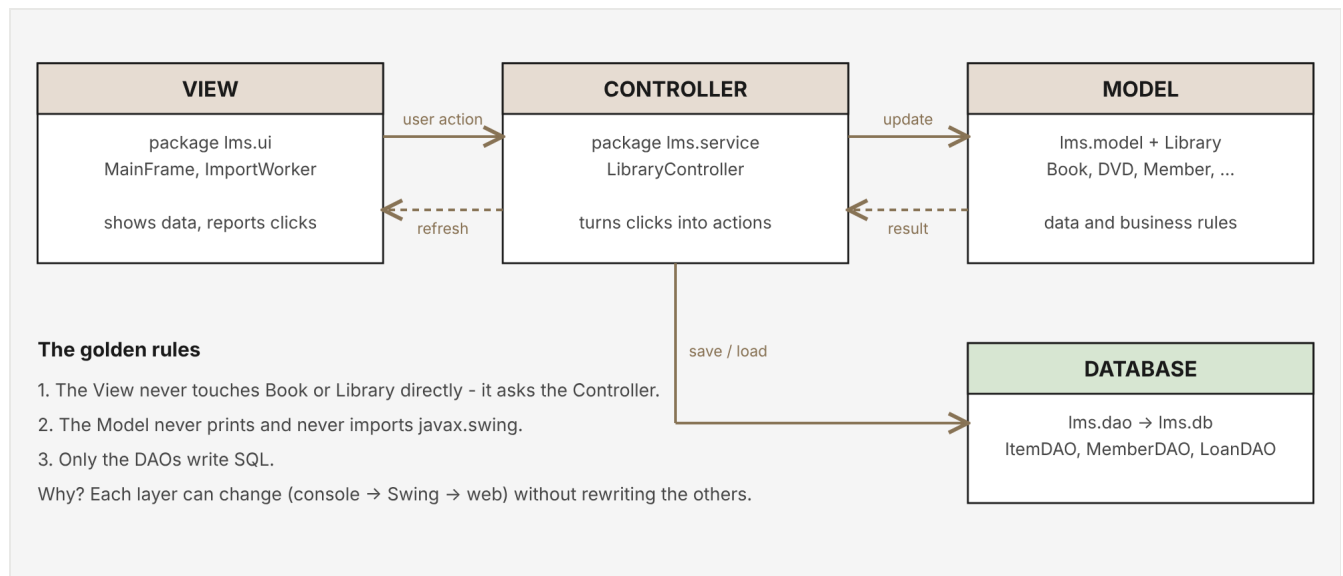
Clean Code is already on loan.

OK

B001 borrowed by M001

Member: M002 - Mr. Kasun Silva Borrow Return

3. MVC: three jobs, three places



Part	Job	LMS package	Must not
Model	Data and business rules	<code>lms.model</code> , <code>Library</code>	print, or import Swing
View	Show things, forward clicks	<code>lms.ui</code>	contain rules or touch <code>Book</code> logic
Controller	Turn user actions into model calls, convert raw text	<code>lms.service.LibraryController</code>	create widgets

The controller converts raw text into objects and rejects bad input:

```
public void addItem(String type, String id, String title, String detail) {
    if (library.hasItem(id))
        throw new IllegalArgumentException("An item with id '" + id + "' already exists");
    switch (type) {
        case "Book"      -> library.addItem(new Book(id, title, detail));
        case "DVD"       -> library.addItem(new DVD(id, title, parseWholeNumber(detail,
"Duration"))));
        case "Magazine" -> library.addItem(new Magazine(id, title, parseWholeNumber(detail, "Issue
number"))));
        default -> throw new IllegalArgumentException("Unknown item type: " + type);
    }
}
```

Typing letters into the duration box now gives a clear dialog instead of a crash:

Library Management System

New item

Type: DVD

ID: D010

Title: Operating Systems

Detail: abc

Add Item

ID	Title	Type	Detail	Status
B001	Clean Code	Book	Robert C. Mart...	On loan: Nade...
B002	Refactoring	Book	Martin Fowler	Available
D001	Introduction to Algorithms	DVD	120 min	Available
G001	Software Weekly	Magazine	Issue 42	Available

Duration must be a whole number

OK

Cannot borrow: Clean Code is already on loan.

Member: M002 - Mr. Kasun Silva

Borrow
Return

Why bother? Next week the controller gets a search feature. In Week 11 the controller starts talking to a database. **The view and the model do not change.** That is what a good separation buys you.

4. Refactoring step: remove printing from the model

In Weeks 4 and 5, `LibraryItem.loan()` printed "loaned to ...". A model that prints cannot be reused in a window. This week, delete those `System.out.println` calls. The **view** reports success in its status bar.

5. Common problems

Symptom	Cause
Nothing happens on click	Listener not registered, or registered on a different button object
<code>NullPointerException</code> in a handler	Reading a field before it is created, or no table row selected
Dialog text is cryptic	You showed <code>e.toString()</code> ; show <code>e.getMessage()</code>
View calls <code>new Book(...)</code>	Move that into the controller

6. Lab 6 preview (Week 9)

Add listeners for Add, Borrow and Return; show every failure with a dialog; create `LibraryController` and make `MainFrame` use **only** the controller; delete printing from the model. Take **three screenshots** (success, an error, bad input) for the PR. Branch `week9-gui-events-mvc`.

7. Self-check

1. Name the event source, listener and registration in `refreshButton.addActionListener(e -> reload())`.

2. Which MVC part should parse the text "120" into an `int` ?
3. Why is `System.out.println` inside `LibraryItem` a problem for a GUI?
4. A teammate imports `javax.swing.*` into `Book`. What do you tell them?