Software Engineering
The University of Auckland

Quality Assurance
User Interface Modeling

Part II - Lecture 4

Interviewing Methods of the FBI

Cognitive interview: method to enhance accurate recall

1. **Context**: reinstate the context of the event
2. **Detail**: don't hold back any details
3. **Order**: recall the event in a different sequence
4. **Perspective**: look at the event from different perspectives

Major cause of project failure: **wrong requirements**

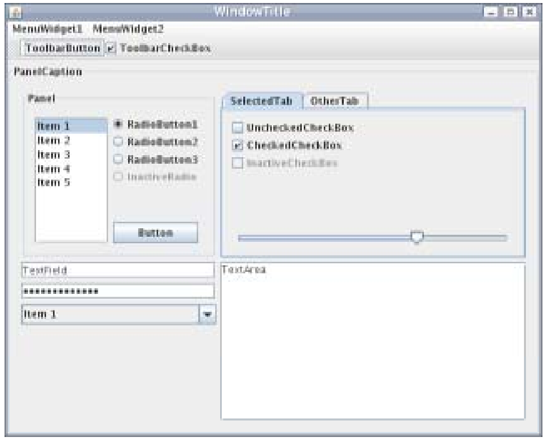
Reference:

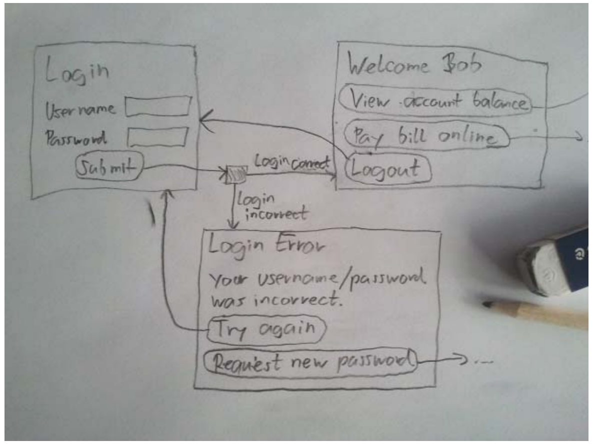
Geiselman et al. Enhancement of Eyewitness Memory with the Cognitive Interview. *American Journal of Psychology* , 99(3), 1986. http://www.jstor.org/stable/10.2307/1422492

Today's Outline

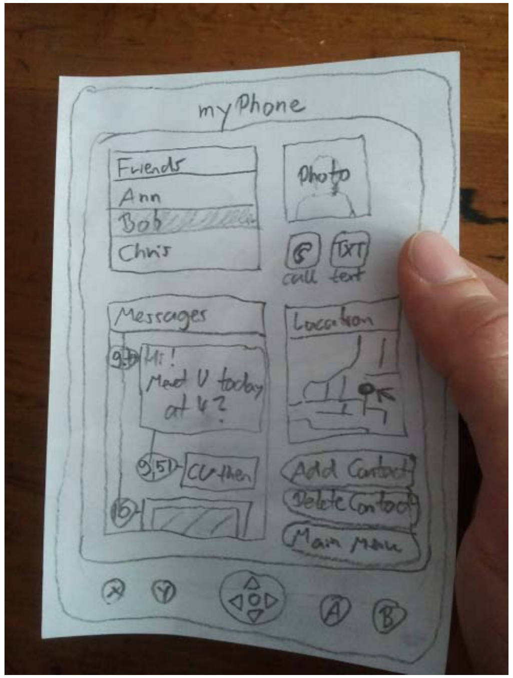
- User Interface Modeling
- GUI Builders: WindowBuilder for Eclipse
- User Interface Prototypes ("Click Dummies")

User Interface Modeling





Introduction to Modeling

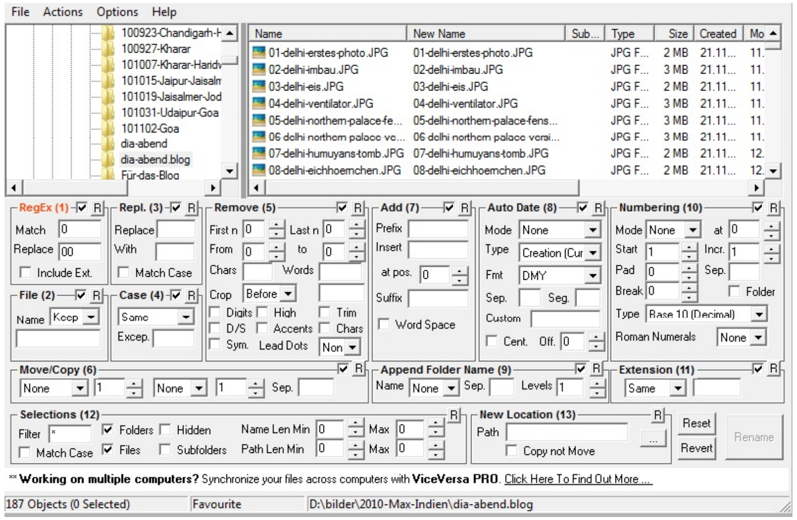


- Software is complex;
  how can we deal with it?
- Common solution:
  try to use a **good representation**
- **Model**: represents certain properties of an object in a different context
  - Abstraction: reduce complexity by taking away unnecessary details
  - Clarity: make interesting properties more visible
  - Facilitate application of a methodology
  - Usability (e.g. easy to create, change, understand…)
- Usually many different models conceivable; different models for different purposes

User Interfaces (UIs)

SE Software Engineering
The University of Auckland

- User interfaces are the interfaces between humans and computers
  - **Input**: "How does the user talk to the system?"
  - **Output**: "How does the system talk to the user?"
  - **Interaction:** input and output between human and computer over time (HCI=Human-Computer Interaction)
- The UI is a **crucial part** of a system
  - Functionality is useless if users don't know how to use it
  - Users won't use it if usage is cumbersome



100923-Chandigarh-H
100927-Kharar
101007-Kharar-Haridv
101015-Jaipur-Jaisalm
101019-Jaisalmer-Jod
101031-Udaipur-Goa
101102-Goa
dia-abend
dia-abend.blog
Für-das-Blog

Name	New Name	Sub...	Type	Size	Created	Mo
01-delhi-erstes-photo.JPG	01-delhi-erstes-photo.JPG		JPG F...	2 MB	21.11...	11.
02-delhi-imbau.JPG	02-delhi-imbau.JPG		JPG F...	3 MB	21.11...	11.
03-delhi-eis.JPG	03-delhi-eis.JPG		JPG F...	2 MB	21.11...	11.
04-delhi-ventilator.JPG	04-delhi-ventilator.JPG		JPG F...	3 MB	21.11...	11.
05-delhi-northern-palace-fe...	05-delhi-northern-palace-fens...		JPG F...	3 MB	21.11...	11.
06-delhi-northern-palace-ve...	06-delhi-northern-palace-versi...		JPG F...	3 MB	21.11...	11.
07-delhi-humuyans-tomb.JPG	07-delhi-humuyans-tomb.JPG		JPG F...	2 MB	21.11...	12.
08-delhi-eichhoernchen.JPG	08-delhi-eichhoernchen.JPG		JPG F...	2 MB	21.11...	12.

RegEx (1) ☑ R — Match 0; Replace 00; ☐ Include Ext.
Repl. (3) ☑ R — Replace; With; ☐ Match Case
Remove (5) ☑ R — First n 0; Last n 0; From 0 to 0; Chars; Words; Crop Before; ☐ Digits ☐ High ☐ Trim; ☐ D/S ☐ Accents ☐ Chars; ☐ Sym. Lead Dots Non
Add (7) ☑ R — Prefix; Insert; at pos. 0; Suffix; ☐ Word Space
Auto Date (8) ☑ R — Mode None; Type Creation (Cur; Fmt DMY; Sep.; Seg.; Custom; ☐ Cent. Off. 0
Numbering (10) ☑ R — Mode None at 0; Start 1 Incr. 1; Pad 0 Sep.; Break 0 ☐ Folder; Type Base 10 (Decimal); Roman Numerals None
File (2) ☑ R — Name Keep
Case (4) ☑ R — Same; Excep.
Move/Copy (6) ☑ R — None 1 None 1 Sep.
Append Folder Name (9) ☑ R — Name None Sep. Levels 1
Extension (11) ☑ R — Same
Selections (12) R — Filter *; ☑ Folders ☐ Hidden; Name Len Min 0 Max 0; ☐ Match Case ☑ Files ☐ Subfolders; Path Len Min 0 Max 0
New Location (13) R — Path ...; ☐ Copy not Move
Reset
Revert
Rename

** **Working on multiple computers?** Synchronize your files across computers with **ViceVersa PRO**. Click Here To Find Out More ...

187 Objects (0 Selected) | Favourite | D:\bilder\2010-Max-Indien\dia-abend.blog

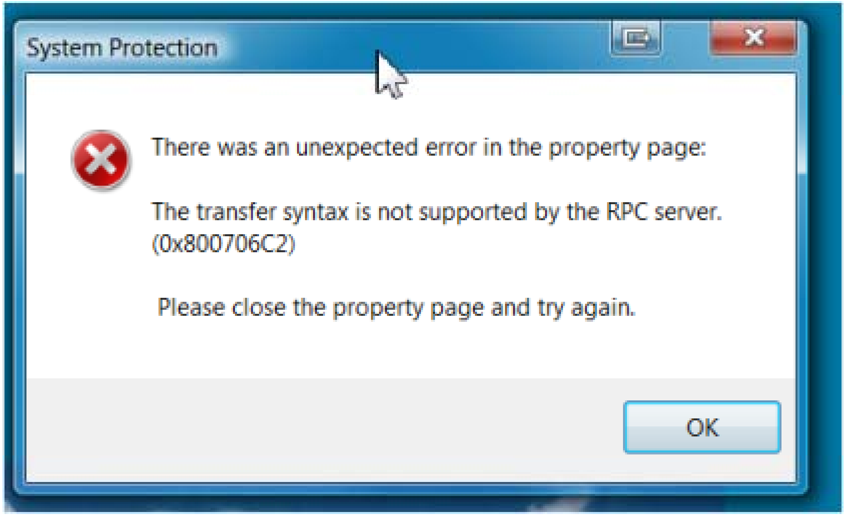
Usability

ISO 9241 definition:

"The **effectiveness**, **efficiency**, and **satisfaction** with which specified *users* achieve specified *goals* in particular *environments*."

Some usability heuristics
(by Jacob Nielsen http://www.useit.com/papers/heuristic)

- Visibility of system status
- Match between system and the real world
- User control and freedom
- Consistency and standards
- Error prevention
- Help users recognize, diagnose, and recover from errors



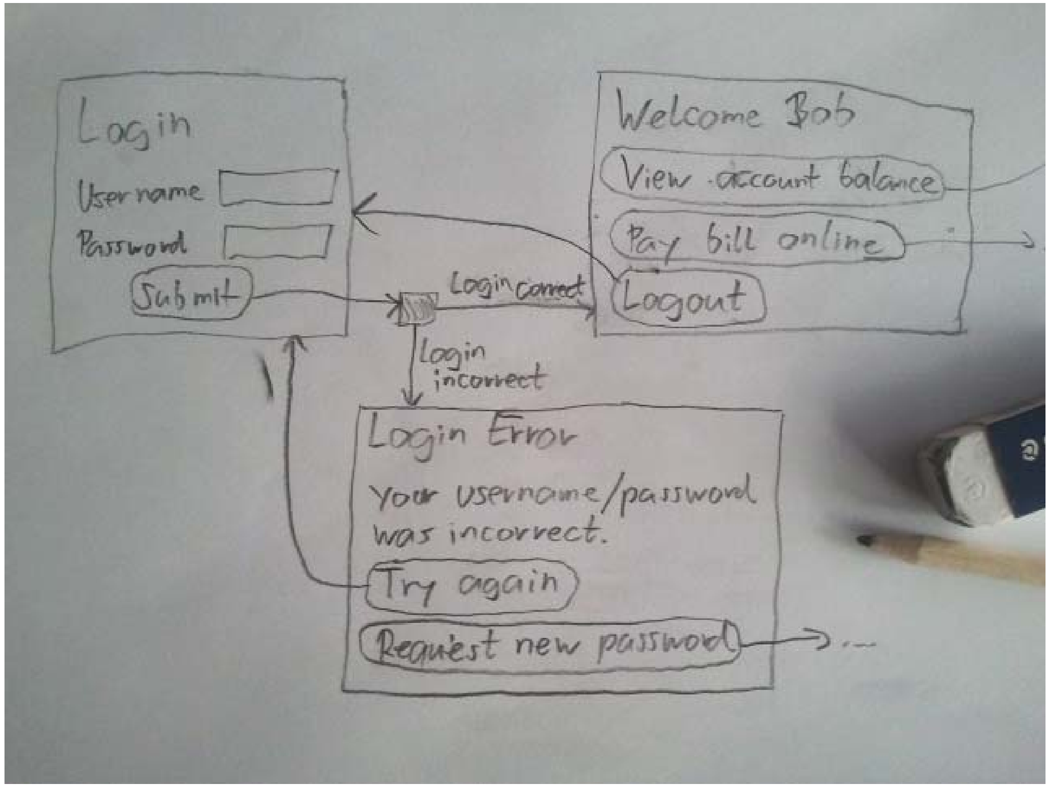
There was an unexpected error in the property page:

The transfer syntax is not supported by the RPC server. (0x800706C2)

Please close the property page and try again.

Screen Diagrams

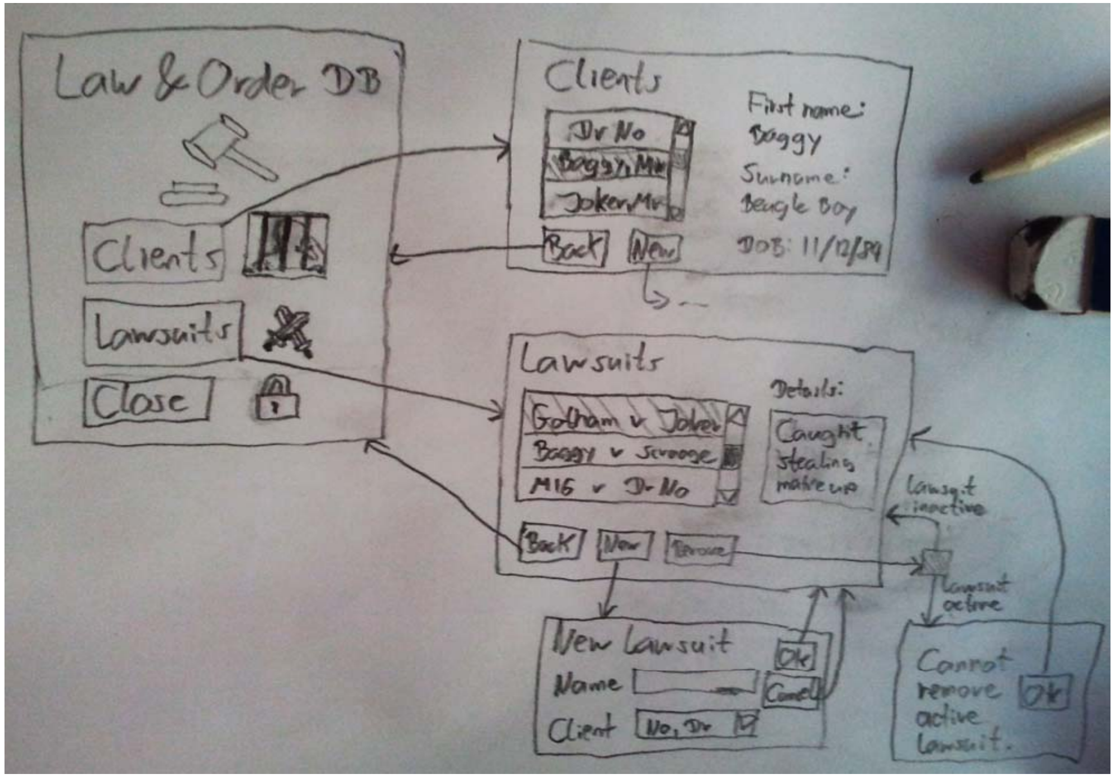
- **Idea**: get UI right through early user feed-back
- Use models to discuss UI with users
- Screen diagrams are a simple informal model for UIs
  1. Draw **prototypical screens** of a system which look like real screens, with real data (graphical details not important)
  2. Draw **arrows** from the controls of a screen to the screens that follow when the control is used (e.g. button click)
  3. If multiple screens are connected to same control, insert black square signifying **conditional branch**



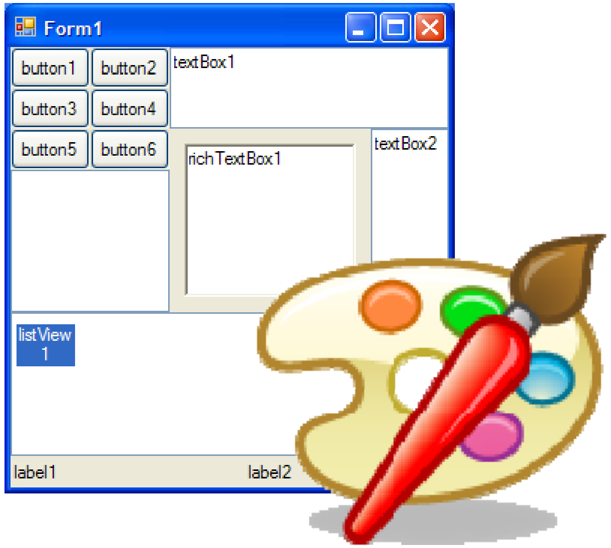
Screen Diagram Example

Create a click dummy for the following system:

A lawyer needs a program to manage clients and lawsuits. When she opens the program, she wants to see a menu with functions for listing all clients, listing all lawsuits, and closing the program. The screen that lists all the clients has a function for showing the details of a client and a function for going back to the main menu. Similarly, the screen that lists the lawsuits has a function for showing the details of a lawsuit and a function for going back.

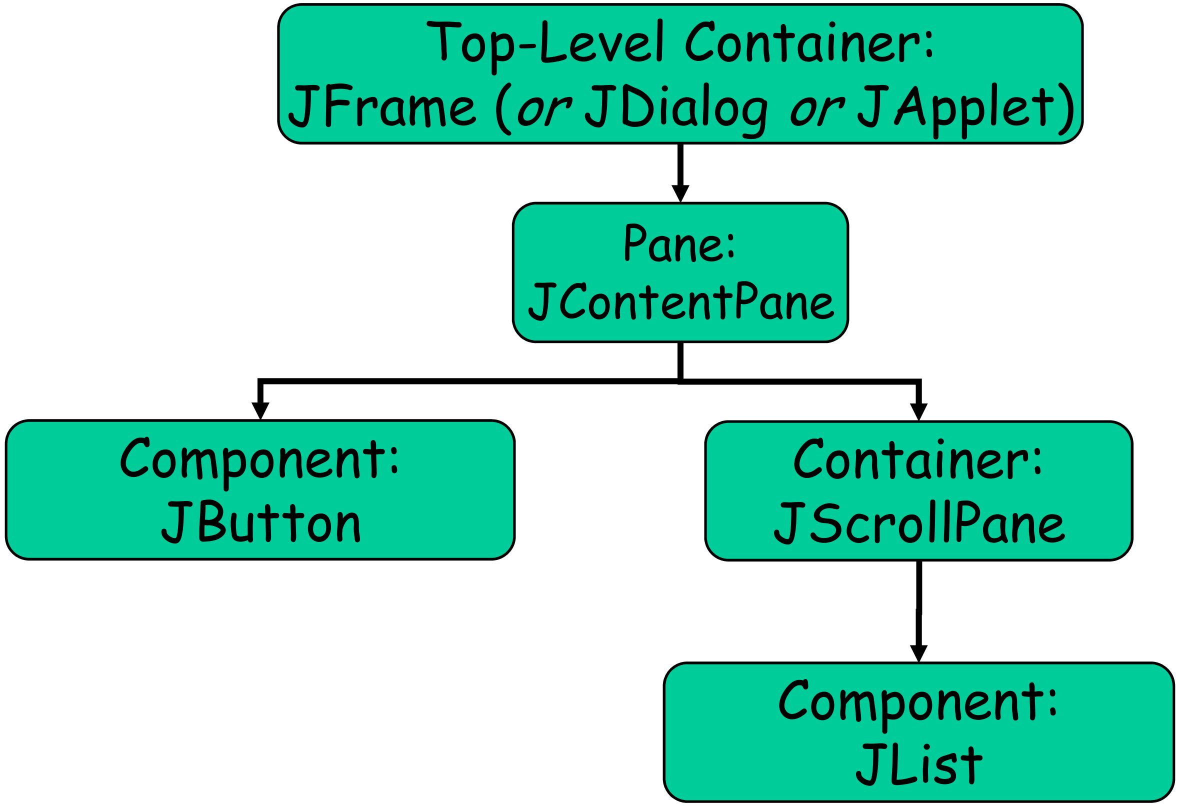


GUI Builders: WindowBuilder for Eclipse



Containment Hierarchy

- Most UI are created by nesting controls (aka widgets/ UI elements) into other controls (containers)
- **Containment hierarchy**: the way the controls of a UI are nested
- Not all controls visible; often invisible internal containers
- E.g. for Java Swing:

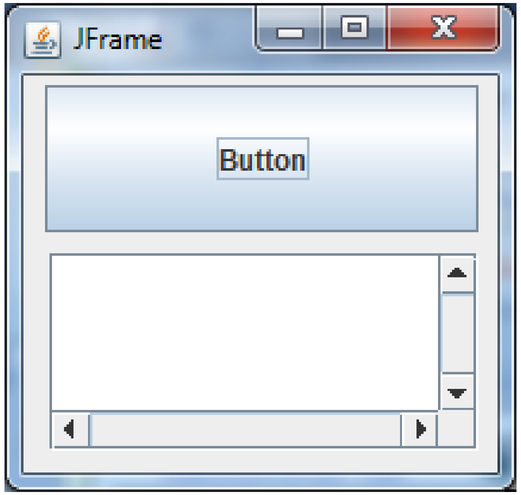

Top-Level Container: JFrame (*or* JDialog *or* JApplet)

Pane: JContentPane

Component: JButton

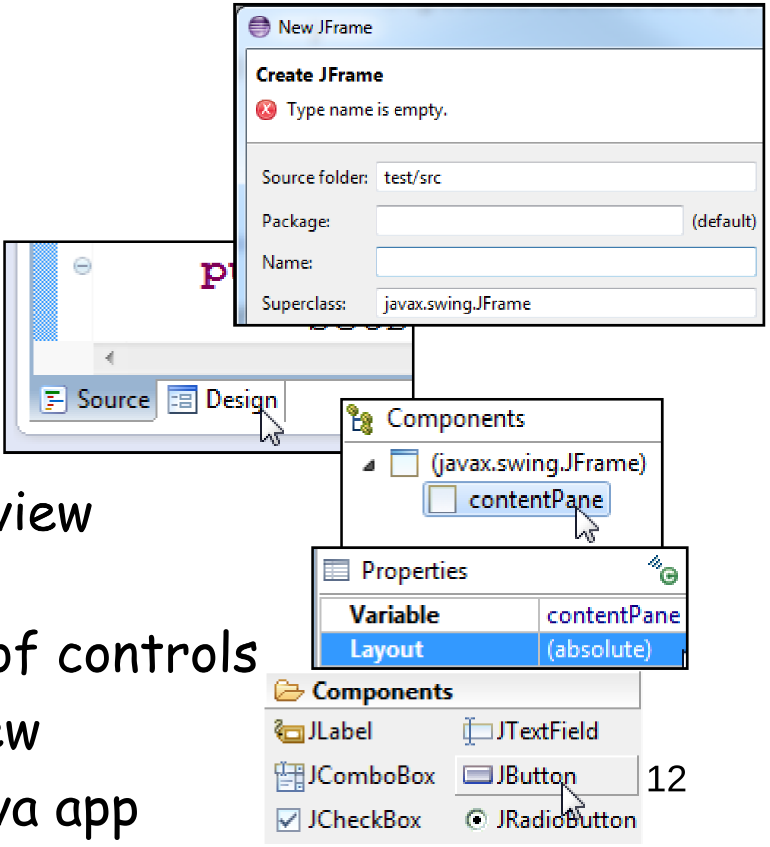
Container: JScrollPane

Component: JList



Creating a JFrame

1. Install WindowBuilder with Help -> "Install New Software" using the "update site" link for your Eclipse version from here: http://www.eclipse.org/windowbuilder/download.php (you need all of "Swing Designer" and all "Core" components of "WindowBuilder Engine")
2. Add a "JFrame" to your project using New -> Other -> WindowBuilder -> Swing Designer -> JFrame
3. Choose a package and class name
4. Switch between code and UI using the tabs at the bottom of the editor
5. Select "contentPane" in Components view
6. In Properties view: set the Layout to "(absolute)" to allow free placement of controls
7. Add components from the Palette view
8. Run the application using "Run" as Java app



New JFrame

Create JFrame

Type name is empty.

Source folder: test/src

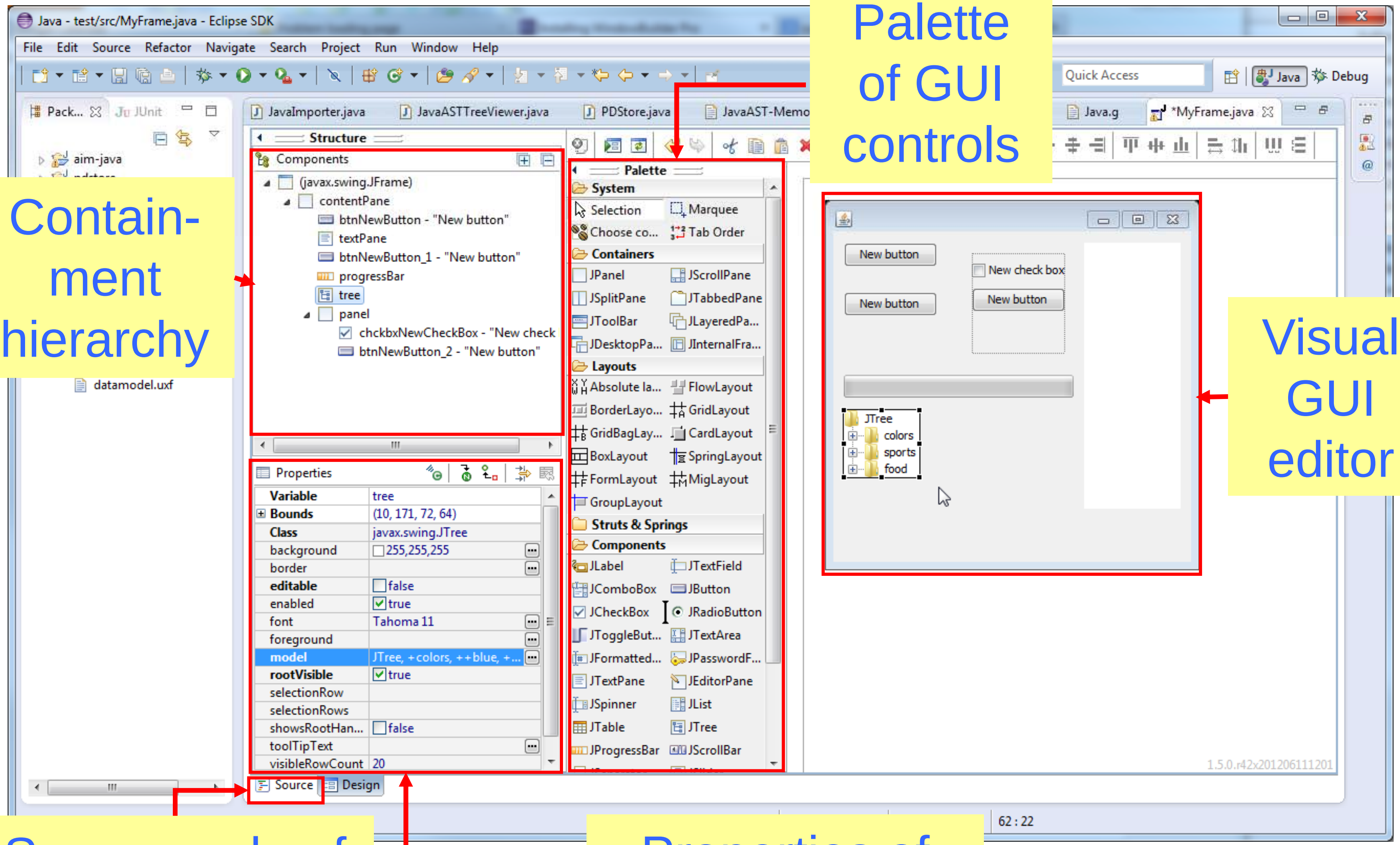
Package: (default)

Name:

Superclass: javax.swing.JFrame

Variable	contentPane
Layout	(absolute)

WindowBuilder User Interface



Variable	tree
Bounds	(10, 171, 72, 64)
Class	javax.swing.JTree
background	□255,255,255
border	
editable	□false
enabled	☑true
font	Tahoma 11
foreground	
model	JTree, +colors, ++blue, +...
rootVisible	☑true
selectionRow	
selectionRows	
showsRootHan...	□false
toolTipText	
visibleRowCount	20

Palette of GUI controls

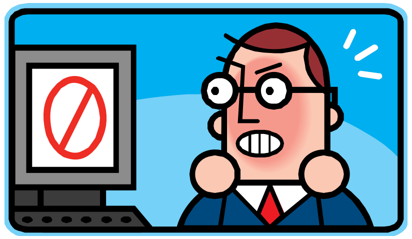
Containment hierarchy

Visual GUI editor

Source code of the GUI

Properties of selected control

User Interface Prototypes ("Click Dummies")

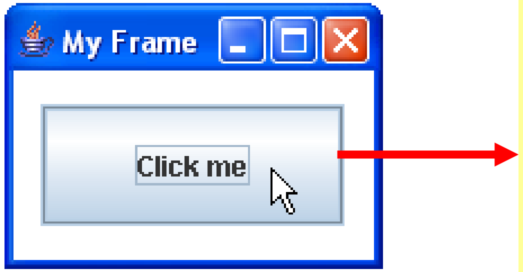


Click Dummies

- Bring screen diagrams to life:
  UI mockups / UI prototypes / “click dummies”
  - The user can **nevigate** between the screens
  - The user can see how **input & output** is done by the system
  - The user can imagine the real system
- Very restricted but very easy to create
  - No functionality implemented
  - All data is just hypothetical
- Very good for early user testing & feedback!!!

Event Handlers

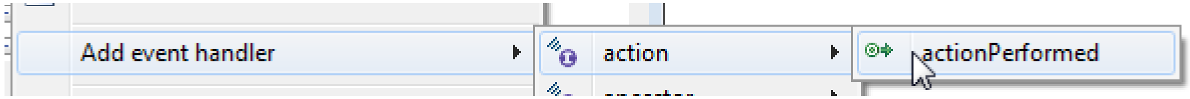
- When the user does something (click, mouse move, key press, etc) an **event object** is created
- **Event handler**:
  method that is called when a particular event occurs
- In Java:
  - Event handler is method in "**event listener**" object
  - For each type of event a **particular method** of a particular event listener
  - Set event handlers with dedicated setter methods in control



```
jButton.addActionListener(
  new java.awt.event.ActionListener() {
    public void actionPerformed(
        java.awt.event.ActionEvent e)
    {
        System.out.println("You clicked me!!!");
    }
});
```

Opening and Closing Frames on Button Click

Add event handler by right-clicking on component and using menu



Code for opening new frame ("Frame2"):

```
Frame2 f2 = new Frame2();
f2.setVisible(true);
```

Code for closing the current frame:

```
// get event source
// and dispose its JFrame
((JFrame)
  ((java.awt.Component)e.getSource()) // get button
      .getParent()    // get contentPane
      .getParent()    // get JLayeredPane
      .getParent()    // get JRootPane
      .getParent())   // get JFrame
  .dispose();
```

Today's Summary

- **Models** try to represent interesting aspects of a system in a clear and manageable way
- **Screen diagrams** illustrate the UI of a system
- **GUI builders** help to create UIs quickly
- **UI prototypes** ("click dummies") can be used for early user feed-back

Friday: lab & lecture
Assignment 3 coming out this week

Quiz

1. Describe a situation where using a model would be useful, and explain why (give 3 reasons).

2. Create a screen diagram for a simple online music store.

3. What is a click dummy? What is it used for?