



ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ  
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ  
ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ «ΠΛΗΡΟΦΟΡΙΚΗ»  
ΘΕΜΑΤΙΚΗ ΕΝΟΤΗΤΑ ΠΛΗ24 «ΣΧΕΔΙΑΣΜΟΣ ΛΟΓΙΣΜΙΚΟΥ»

# Netbeans Binding

Α. ΘΑΝΟΣ



# Περιεχόμενα Διάλεξης

- Τι είναι το Binding
- Βασικά components
- Παραδείγματα Binding
- Παράδειγμα GUI

# Τι είναι το Binding

A series of horizontal lines in teal and light blue colors, extending across the width of the slide below the title.



# Τι είναι το Binding

- Είναι ο τρόπος σύνδεσης και συγχρονισμού ενός Entity Bean ή μιας λίστας από Entity Beans με ένα GUI Component.



# Τι είναι το Binding

- Επιτυγχάνεται με χρήση wizards (γίνεται και χειροκίνητα αλλά δεν ενδείκνυται)
- Εξασφαλίζει διαφανή συγχρονισμό μεταξύ GUI και Entity Bean:
  - όποια αλλαγή γίνεται στο bean παρουσιάζεται αυτόματα στο GUI
  - όποια αλλαγή γίνεται στο GUI αυτόματα αποτυπώνεται στα properties του bean
- Επιτυγχάνεται με την χρήση της κλάσης PropertyChangeSupport (γίνεται αυτόματα κατά τη δήλωση του binding στο wizard, χωρίς τη συγγραφή κώδικα από το χρήστη) και πυροδότηση ειδικών event κατά την κλήση των setter μεθόδων στα entity beans.

# Βασικά Components

A series of horizontal lines in teal and light blue colors, some solid and some dashed, extending across the width of the slide below the title.



# Βασικά Components

- Persistence tools
  - Entity Manager
  - Query
  - Result (List)
- Entity Beans
- Swing Components



# Entities και components

- Για να μπορούν να χρησιμοποιηθούν τα entities ή τα components στα wizards θα πρέπει να γίνει εισαγωγή τους στο design περιβάλλον με drag 'n drop

The screenshot illustrates the process of adding a Java entity to a design form in NetBeans. On the left, the 'Navigator' pane shows a project structure with a 'model' package containing 'Customer.java', which is circled in red. A red arrow points from this file to the design form in the center. The design form contains several text input fields (Κωδ. Πελάτη, Όνομα, Επώνυμο, ΑΔΤ, ΑΦΜ) and a table with columns: Card Id, Category Id, Colour, Factory Name, Model Type, and Reg Number. Below the table are buttons for 'Νέο', 'Διαγραφή', and 'Επεξεργασία'. Another red arrow points from the 'Customer.java' file to the 'Entity Manager' option in the 'Beans' palette on the right. The 'Entity Manager' option is also circled in red. Below the palette, the 'Properties' window for 'Customer.java' is visible, showing details like Name, Extension, File Size, and Classpaths.

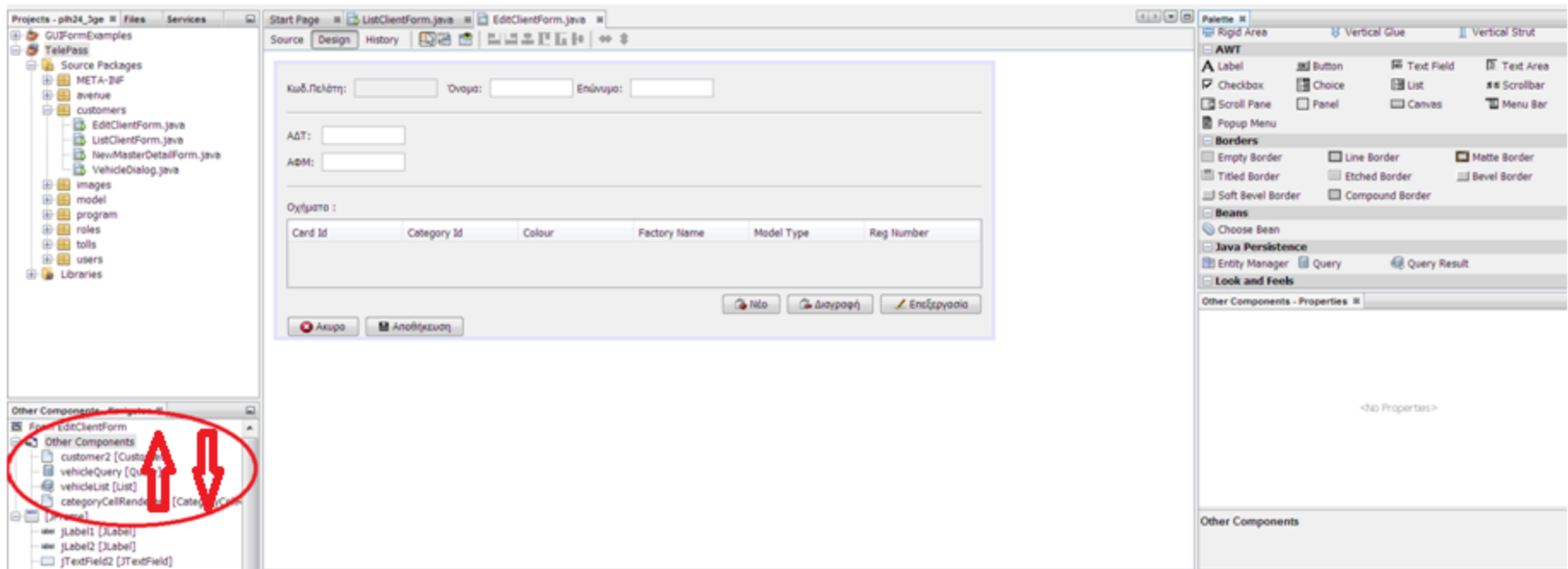
drag 'n drop





# Design Navigator

- Σε design view όλα τα components που μπορούν να χρησιμοποιηθούν σε wizards παρουσιάζονται στο Design Navigator.
- Μπορούμε να αλλάξουμε τη σειρά που εμφανίζονται στον navigator, επηρεάζοντας τη σειρά που εκτελούνται στο πρόγραμμα





# Query Component

- Μπορούμε να χρησιμοποιήσουμε δικό μας κώδικα, αντί να δημιουργήσει το wizard δικό του κώδικα

The screenshot shows the NetBeans IDE interface for configuring the Query Component. The main window displays a form with the following fields:

- Κωδ.Πελάτη:
- Όνομα:
- Επώνυμο:
- ΑΔΤ:
- ΑΦΜ:

Below these fields is a table for 'Οχήματα' (Vehicles) with the following columns:

Card Id	Category Id	Colour	Factory Name	Model Type	Reg Number
---------	-------------	--------	--------------	------------	------------

At the bottom of the form are buttons for 'Νέο' (New), 'Διαγραφή' (Delete), and 'Επεξεργασία' (Edit).

The 'vehicleQuery [Query] - Properties' window is open, showing the 'Code' tab. The custom creation code is:

```
em.createQuery("select v from Ve...
```

The 'vehicleQuery [Query] - Navigator' window is also open, showing the 'vehicleQuery [Query]' component selected.



# Query Component (Custom Code)

- Στο προηγούμενο παράδειγμα χρησιμοποιήσαμε custom code για το query component. Αυτό σημαίνει ότι στο πρόγραμμα θα γραφεί ακριβώς ο κώδικας που θα δώσουμε, χωρίς να χαθεί η δυνατότητα να μπορούμε να χρησιμοποιούμε το component στα wizards

```
49  * regenerated by the Form Editor.
50  */
51  @SuppressWarnings("unchecked")
52  // <editor-fold defaultstate="collapsed" desc="Generated Code">
53  private void initComponents() {
54      bindingGroup = new org.jdesktop.beansbinding.BindingGroup();
55
56      customer2 = customer1;
57      vehicleQuery = em.createQuery("select v from Vehicle v where v.custId=:customer").setParameter("customer",customer2);
58      vehicleList = java.beans.Beans.isDesignTime() ? java.util.Collections.emptyList() : org.jdesktop.observablecollections.ObservableCollections.unmodifiableList(vehicleQuery.getResultList());
59      categoryCellRenderer1 = new avenue.CategoryCellRenderer();
60      jLabel1 = new javax.swing.JLabel();
61      jLabel2 = new javax.swing.JLabel();
62      jTextField2 = new javax.swing.JTextField();
63      jTextField3 = new javax.swing.JTextField();
64      jLabel3 = new javax.swing.JLabel();
65      jButton1 = new javax.swing.JButton();
66      jButton2 = new javax.swing.JButton();
67      jButton3 = new javax.swing.JButton();
68      jLabel4 = new javax.swing.JLabel();
69      jTextField4 = new javax.swing.JTextField();
70      jSeparator1 = new javax.swing.JSeparator();
71      jLabel5 = new javax.swing.JLabel();
72      jTextField5 = new javax.swing.JTextField();
```



# Query Component (2)

- Μπορούμε να χρησιμοποιήσουμε τα properties για να δηλώσουμε το query και τον entity manager
- Σε περίπτωση που ο entity manager δεν είναι component στον design navigator (και συνεπώς δεν μπορούμε να τον χρησιμοποιήσουμε στο wizard) τότε χρησιμοποιούμε custom code

The screenshot illustrates the configuration of the Query Component in the NetBeans IDE. The main window shows a form with fields for customer information (FName, LName, Address, Adt, Afrm, City, Email, Id, Phone, Zip Code). A dialog box titled "query1 [Query] - entityManager" is open, showing the "Set query1's entityManager property using:" dropdown set to "Custom code". The "Property Code" field contains "em" and "java.lang.Object". The "query1 [Query] - Properties" window on the right shows the "entityManager" property set to "<User Code>". The "query1 [Query] - Navigator" window at the bottom left shows the "query1 [Query]" component selected.



# List Component

- Ορίζουμε το query component από το οποίο θα λάβει αποτελέσματα, καθώς και τον τύπο της λίστας
- Ορίζουμε επίσης ότι η λίστα θα είναι observable, που σημαίνει ότι θα είναι συγχρονισμένη με το GUI με το οποίο θα τη συνδέσουμε (bind)

The screenshot illustrates the configuration of a List component in the NetBeans IDE. The main window displays a form with input fields for 'Κωδ.Πελάτη', 'Όνομα', and 'Επώνυμο', and a table for 'Οχήματα' with columns: Card Id, Category Id, Colour, Factory Name, Model Type, and Reg Number. The 'vehicleList [List] - Properties' window is open, showing the 'Code' tab with the following properties: Bean Class (interface java.util.List), Variable Name (vehicleList), Variable Modifiers (private), Type Parameters (<model.Vehicle>), Use Local Variable (checked), Custom Creation Code, Pre-Creation Code, Post-Creation Code, Pre-Init Code, Post-Init Code, Post-Listeners Code, and observable (checked). The 'vehicleList [List] - Navigator' window shows the component hierarchy, with 'vehicleList [List]' selected. The 'Properties' window also shows the 'query' property set to 'vehicleQuery'.



# Entity Component

- Εάν δεν θέλουμε να γίνει δημιουργία νέου αντικειμένου (new) αλλά να δείχνει σε ένα άλλο entity, τότε χρησιμοποιούμε custom code. Την τεχνική αυτή χρησιμοποιούμε όταν θέλουμε να δείχνει σε μεταβλητή που περνάμε στη φόρμα. (αν δεν το κάνουμε αυτό δεν μπορούμε να έχουμε ένα entity που να το περνάμε σαν μεταβλητή και να το χρησιμοποιούμε στα wizards).

The screenshot shows the NetBeans IDE with the following components:

- Projects:** ph24\_3ge, Files, Services.
- Source Packages:** GUIFormExamples, TelePass, Source Packages, META-INF, avenue, customers, EditClientForm.java, ListClientForm.java, NewMasterDetailForm.java, VehicleDialog.java, images, model, program, roles, tolls, users, Libraries.
- Form Design:** The form has fields for "Κωδ. Πελάτη:", "Όνομα:", "Επώνυμο:", "ΑΔΤ:", "ΑΦΜ:", and a table "Οχήματα:" with columns: Card Id, Category Id, Colour, Factory Name, Model Type, Reg Number. Buttons at the bottom: "Νέο", "Διαγραφή", "Επεξεργασία", "Ακύρο", "Αποθήκευση".
- customer2 [Customer] - Navigator:** Shows the "customer2 [Customer]" entity selected under "User Components".
- customer2 [Customer] - Properties:** The "Code" tab is active, showing the "Custom Creation Code" field with the value "customer1".



- ```
public EditClientForm(Customer c, boolean readOnly) {
    em = DBManager.em;
    customer1=c;
    this.readOnly = readOnly;
    if ( !(em.getTransaction().isActive()) ){
        em.getTransaction().begin();
    }
    initComponents(); // εδώ θα γίνει το πέραςμα της
                      // customer1 στην customer2
}
```

# Παραδείγματα Binding

A series of horizontal lines in teal and light blue colors, some solid and some dashed, extending across the width of the slide below the title.



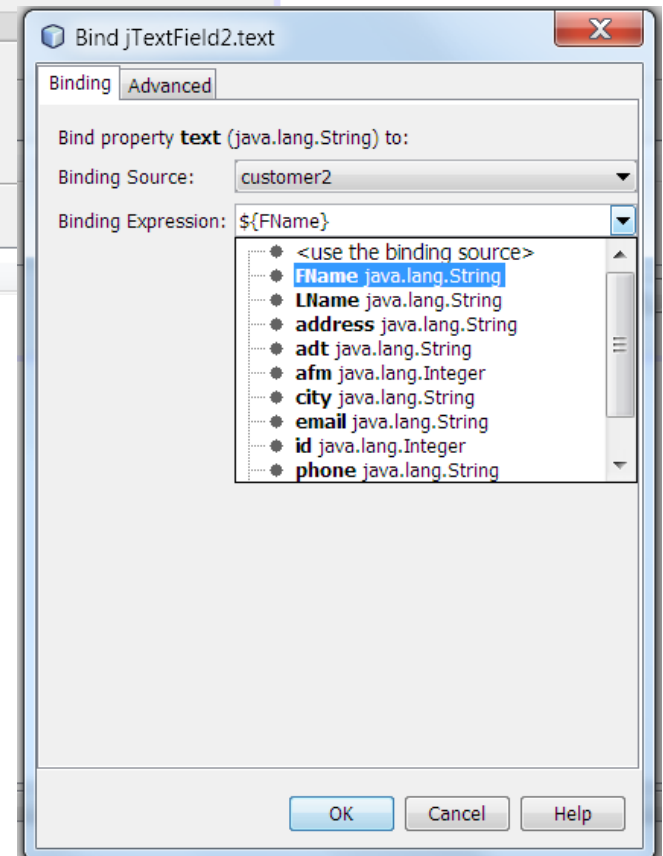
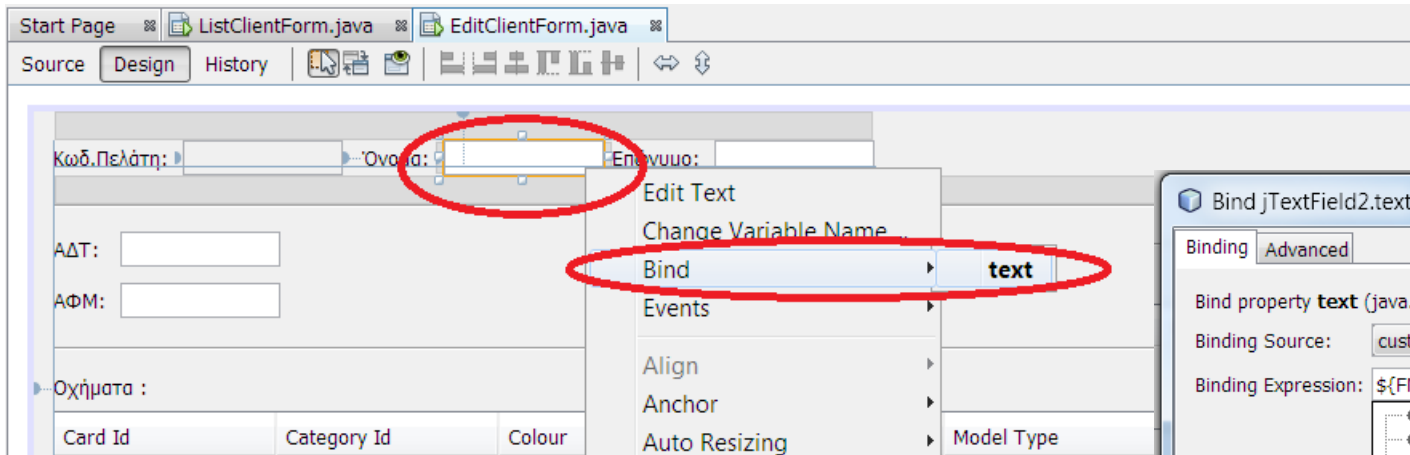


# Παραδείγματα Binding

- Text field
- Table
- Combo Box

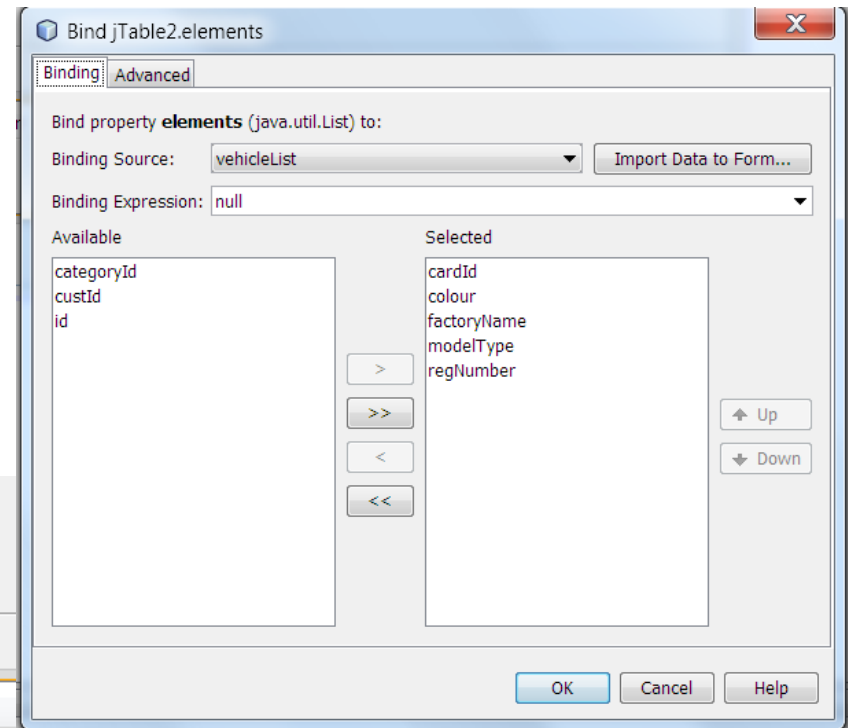
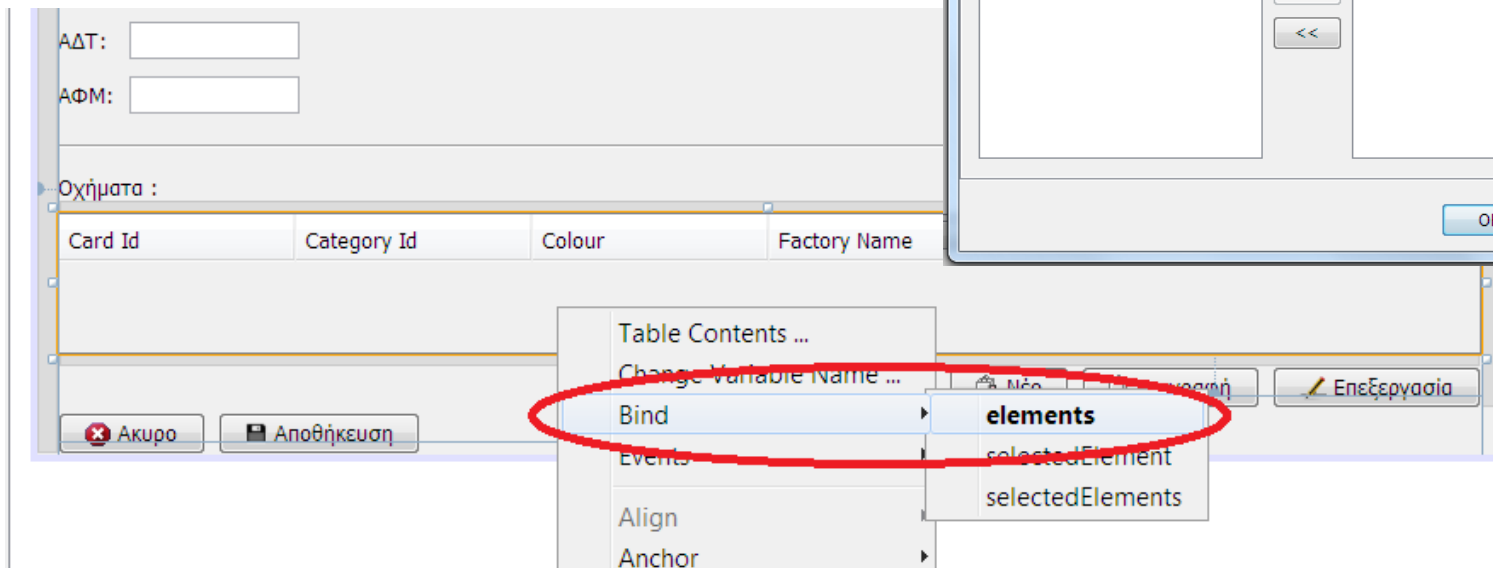


# Text Field Binding



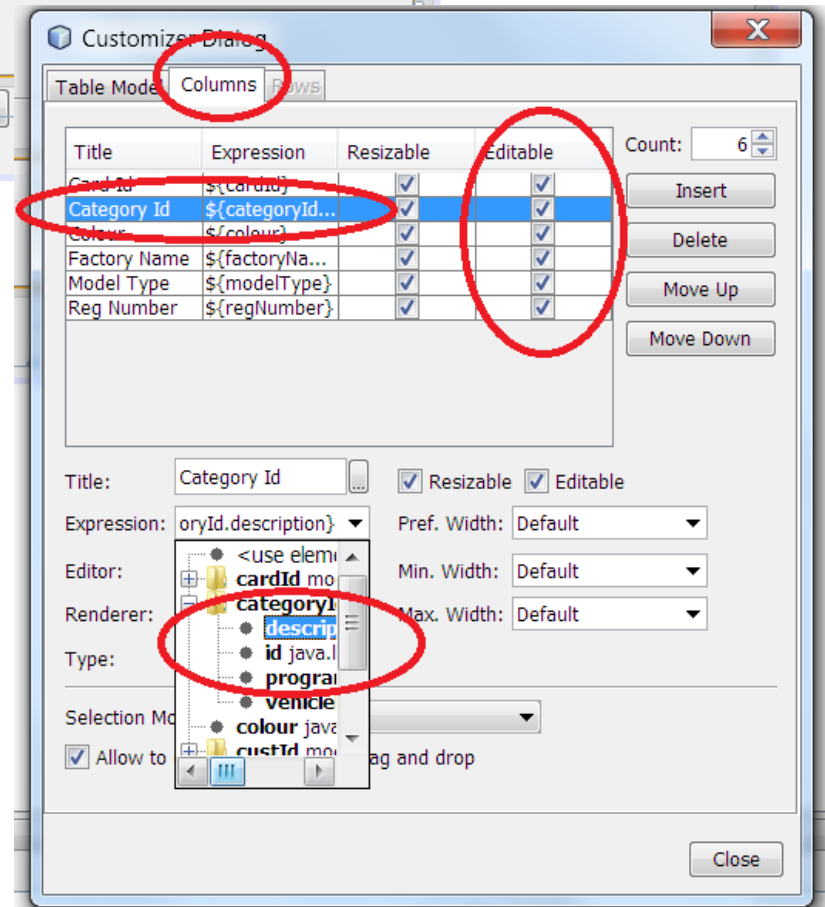
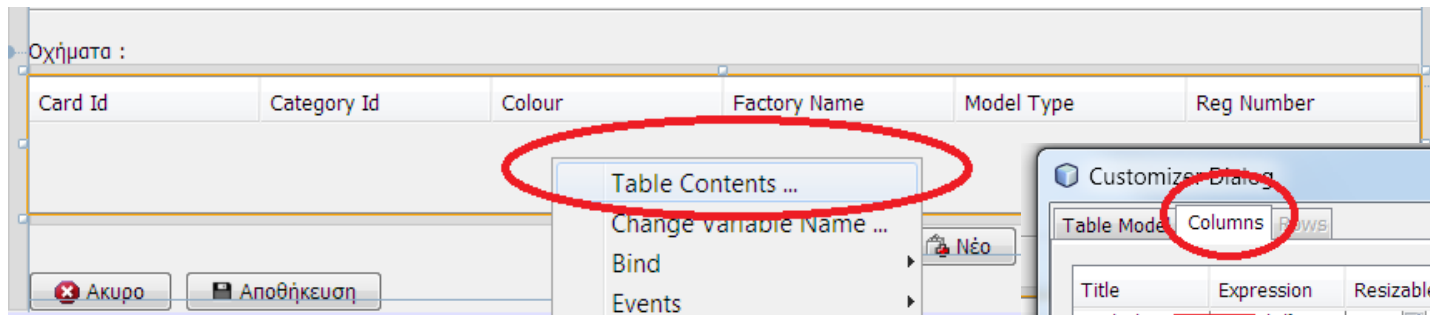


# Table binding (1)





# Table Binding (2)





# ComboBox Binding

- Το `elements` δείχνει από ποια λίστα θα γεμίσει τις επιλογές
- Το `selectedItem` δείχνει σε ποιο component θα συνδεθεί η επιλογή που θα κάνουμε στη πτυσσόμενη λίστα

The screenshot illustrates the configuration of a ComboBox in a Java IDE. A context menu is open over the ComboBox, showing options: "Change Variable Name ...", "Bind", "Events", "elements", and "selectedItem". The "elements" and "selectedItem" options are highlighted with a red circle.

Two "Bind" dialog boxes are shown:

- Bind jComboBox1.elements**:
  - Binding Source: `list1`
  - Binding Expression: `null`
- Bind jComboBox1.selectedItem**:
  - Binding Source: `appUser2`
  - Binding Expression: `${roleId}`



# ComboBox Binding Renderer

- Για να δείξουμε ένα συγκεκριμένο πεδίο από τα αντικείμενα της λίστας θα πρέπει να δημιουργήσουμε έναν Renderer (μια κλάση που κληρονομεί την DefaultListCellRenderer). Αρκεί να αντικαταστήσουμε στον παρακάτω κώδικα τον τύπο των αντικειμένων της λίστας και το πεδίο που θέλουμε να δείξουμε.

```
/**
 *
 * @author Aggelos
 */
public class PeripheryListRenderer extends DefaultListCellRenderer{

    @Override
    public Component getListCellRendererComponent(
        JList list, Object value, int index, boolean isSelected, boolean cellHasFocus)
    {
        super.getListCellRendererComponent(list, value, index, isSelected, cellHasFocus);
        if (value instanceof TblElectoralPeriphery) {
            TblElectoralPeriphery mec = (TblElectoralPeriphery) value;
            setText(mec.getFldName());
        }
        return this;
    }
}
```

Όνομα του Renderer

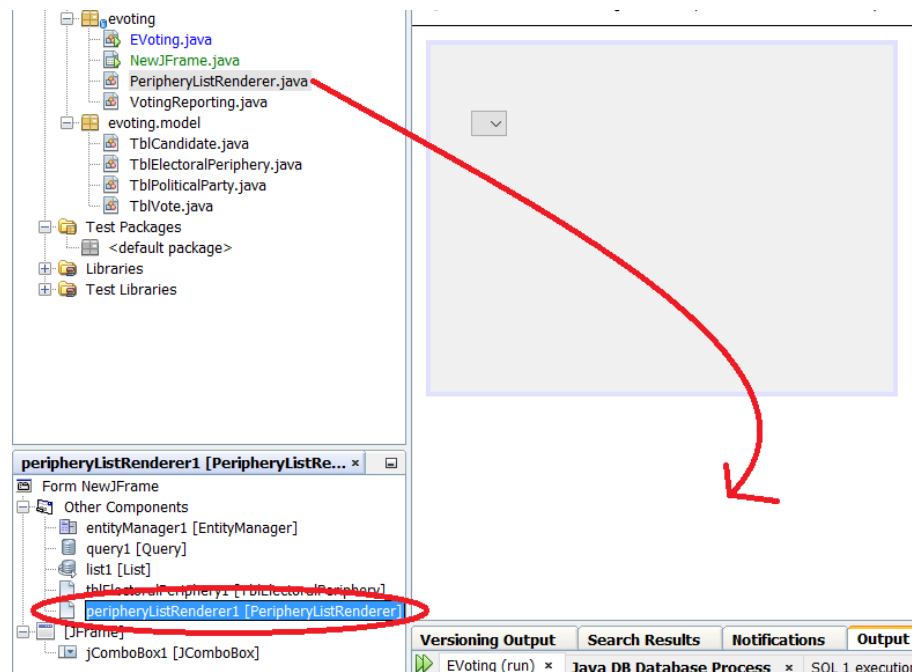
Τύπος των αντικειμένων της λίστας που είναι συνδεδεμένο το ComboBox

Πεδίο του αντικειμένου που θέλουμε να φαίνεται στο ComboBox



# ComboBox set Renderer (1/2)

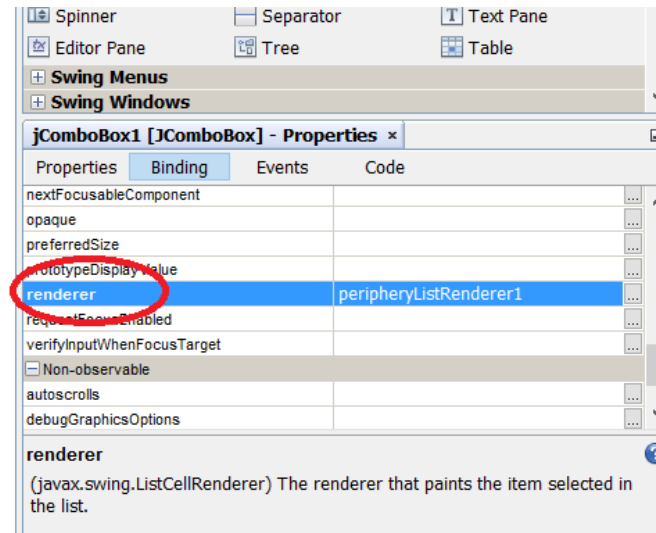
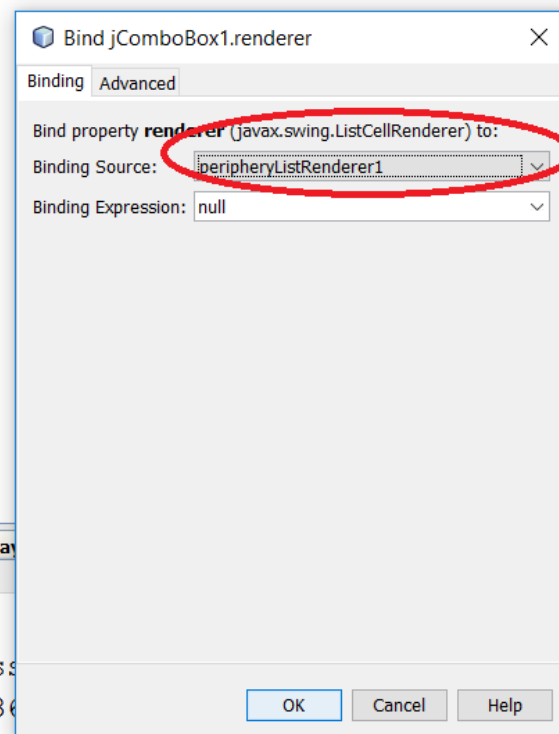
- Κάνουμε drag'n'drop τον renderer στη φόρμα





# ComboBox set Renderer (2/2)

- Ορίζουμε ότι το ComboBox χρησιμοποιεί renderer

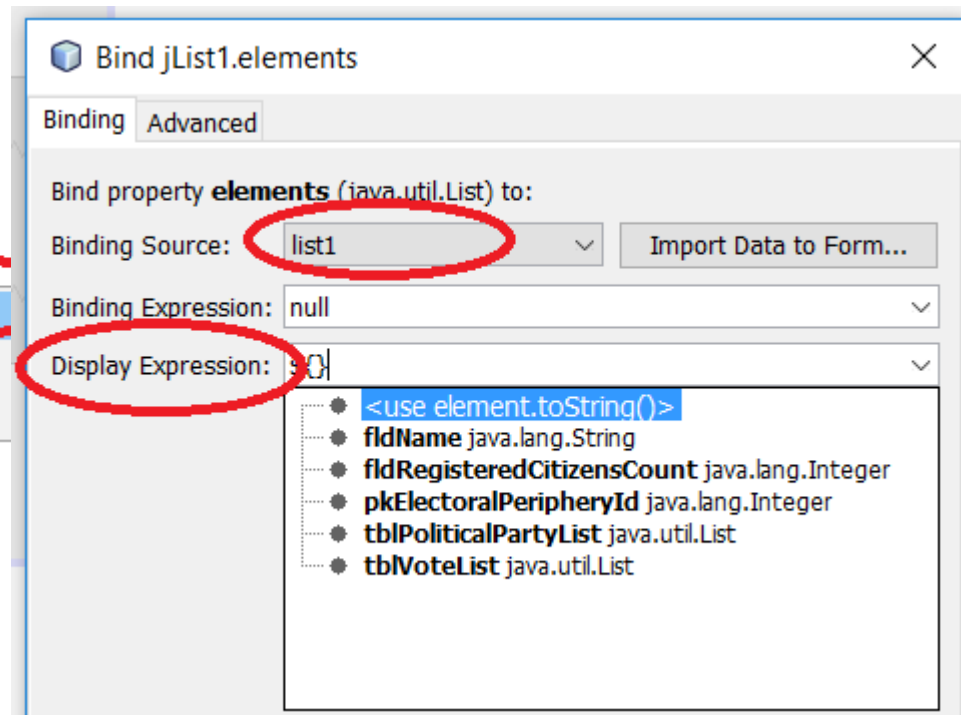
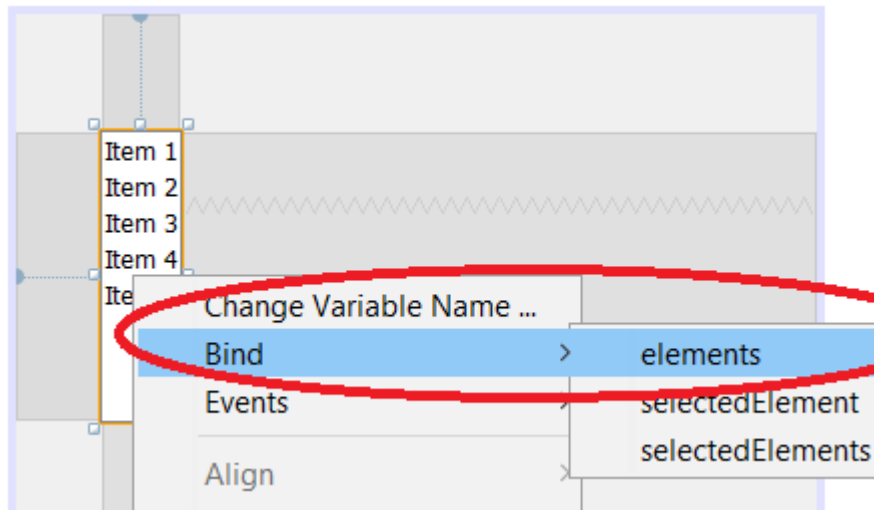






# JList Binding 1/2

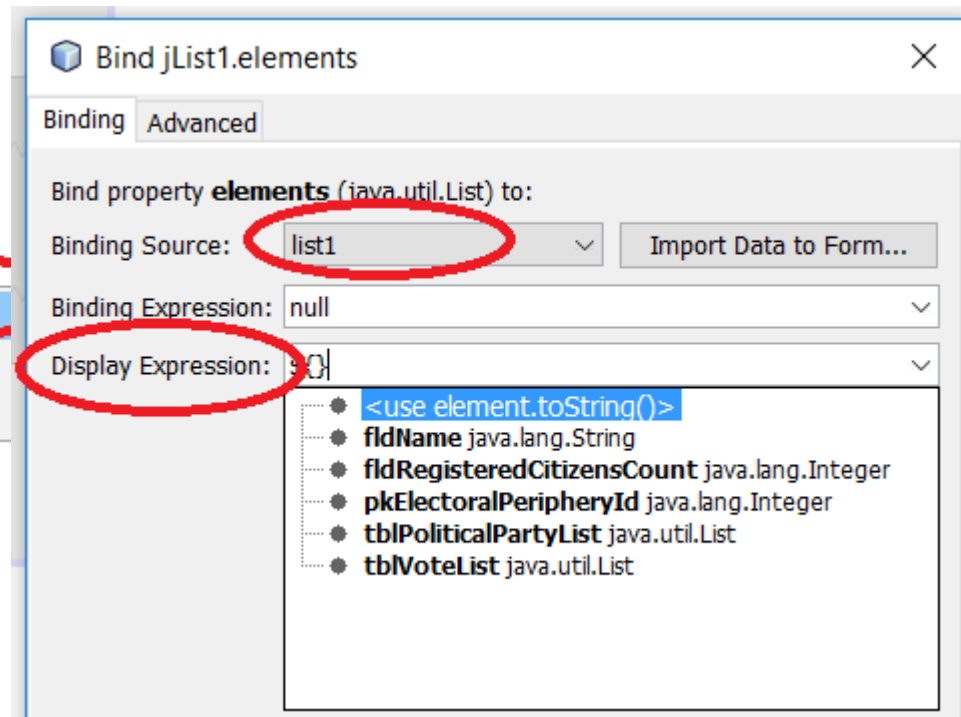
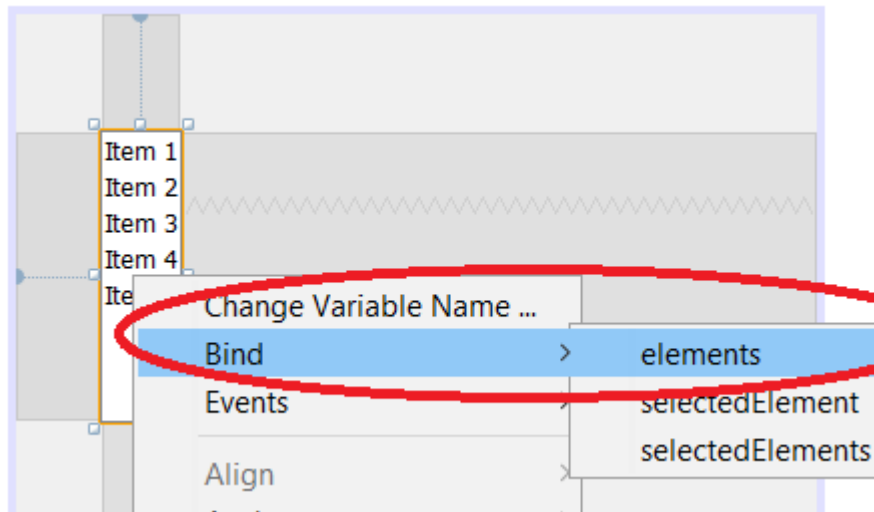
- Το `elements` δείχνει από ποια λίστα θα γεμίσει τις επιλογές
- Το `selectedItem` δείχνει σε ποιο component θα συνδεθεί η επιλογή που θα κάνουμε από τη λίστα
- Το `selectedItems` δείχνει σε ποιο component θα συνδεθεί η λίστα των πολλαπλών επιλογών που μπορεί να κάνουμε από λίστα





# JList Binding 2/2

- Το display Expression επιλέγει ποιο πεδίο από τα αντικείμενα της λίστας θα εμφανίζεται (Δεν χρειάζεται να ορίσουμε renderer δηλαδή, μας δίνει τη δυνατότητα ο wizard να ορίσουμε το πεδίο)





# Jlist/ComboBox Selected Item

- Για να ορίσουμε σε ποιο αντικείμενο θα καταχωρηθεί η εκάστοτε επιλογή μας από μια πτυσσόμενη (JComboBox) ή απλή λίστα (JList), θα πρέπει να το έχουμε κάνει drag'n'drop στη φόρμα

The screenshot illustrates the process of binding a JList's selected item to a specific object in the project. The top-left pane shows the project structure with files like EVoting.java, NewJFrame.java, and various model classes. The bottom-left pane, titled 'jList1 [JList] - Navigator', shows the components of the JList, with 'tblElectoralPeriphery1 [TblElectoralPeriphery]' circled in red. The top-right pane shows a visual representation of the JList with items 'Item 1' through 'Item 5'. The bottom-right pane is a dialog titled 'Bind jList1.selectedElement', which is used to configure the binding. It shows the 'Binding Source' as '<not bound>' and the 'Binding Expression' as a list of objects. The object 'tblElectoralPeriphery1' is selected and circled in red, indicating it is the target of the binding.