Języki i metody programowania Java
Lab6
Budowa interfejsu graficznego użytkownika (GUI – Graphical User Interfaces) z wykorzystaniem pakietu Swing

Strona do pobrania tutoriala:
http://www.oracle.com/technetwork/java/javase/java-tutorial-downloads-2005894.html
Bezpośredni dostęp do strony tutoriala
https://docs.oracle.com/javase/tutorial/

Zofia Kruczkiewicz

Dostępne źródła informacji dotyczące budowy GUI

1. Po ściągnięciu i rozpakowaniu, tutorial można używać lokalnie, po uruchomieniu strony index.html w katalogu: …\tutorial.

1.1. Creating a GUI with Swing — A comprehensive introduction to GUI creation on the Java platform:

file:///C:/Studia/labjavapwr/tutorial/uiswing/index.html

1.2. How to use Menu (from Using Swing Components):

file:///C:/Studia/labjavapwr/tutorial/uiswing/components/menu.html

1.3. How to use CardLayout (from Laying Out Components Within a Container):

file:///C:/Studia/labjavapwr/tutorial/uiswing/layout/card.html

lub

2. Informacje o tworzeniu interfejsu graficznego użytkownika są dostępne ze strony tutoriala

2.1. Creating a GUI with Swing — A comprehensive introduction to GUI creation on the Java platform:

https://docs.oracle.com/javase/tutorial/uiswing/index.html

2.2. How to use Menu (from Using Swing Components):

https://docs.oracle.com/javase/tutorial/uiswing/components/menu.html

2.3. How to use CardLayout (from Laying Out Components Within a Container):

https://docs.oracle.com/javase/tutorial/uiswing/layout/card.html

1. Utworzenie GUI z wykorzystaniem pakietu Swing -zastosowanie klasy Produkt1 oraz klasy Fasada_warstwy_biznesowej w wersji SE (na podstawie projektu Sklep_6 z lab6)

1.1. Należy wykonać projekt typu **Java Class Library:** należy wybrać kolejno **File/New Project…/Java/Java Class Library** i nacisnąć klawisz **Next**. Na kolejnym formularzu w polu **Project Location** należy wybrać katalog projektu, a w polu **Project Name** należy wpisać nazwę projektu (np. **Sklep6_SE**) i następnie nacisnąć klawisz **Finish**.

1.2. Należy otworzyć projekt **Sklep_6 (lab6 z TINT)** i skopiować następujące pakiety:

warstwa_biznesowa- w pakiecie powinna być klasa **Fasada_warstwy_biznesowej**

warstwa_biznesowa.entity - w pakiecie powinna być klasa **Produkt1**-

pomoc – w tym pakiecie usunąć klasę Zmiana_danych (Safely Delete), a pozostawić klasę **Uslugi**.

Kopiowanie: należy kliknąć prawym klawiszem myszy na kopiowany pakiet i wybrać z listy **Copy.** Następnie, w nowym projekcie typu Java ClassLibrary **(Sklep6_SE)** należy kliknąć prawym klawiszem myszy na pozycję **Source Packages** i kliknąć na pozycję **Paste**.

1.3. Należy usunąć błędy składni w nowym projekcie typu Java **ClassLibrary (Sklep6_SE)**:

- Aby usunąć błędy składni w klasie **Produkt1**, należy kliknąć prawym klawiszem na folder **Libraries** i wybrać pozycję **Add Library…**, i następnie wybrać bibliotekę **EclipseLink (JPA 2.1)** z listy bibliotek

- aby usunąć błędy składni z klasy **Fasada_warstwy_biznesowej,** należy usunąć adnotację **@Stateless** i kliknąć prawym klawiszem na okno edytora tej klasy i wybrać pozycję **Fix Imports** w celu usunięcia niepotrzebnych importów.

1. Utworzenie GUI z wykorzystaniem pakietu Swing -zastosowanie klasy Produkt1 oraz klasy Fasada_warstwy_biznesowej w wersji SE (cd)

1.4. Należy wykonać projekt typu **Java Application**: należy wybrać kolejno **File/New Project…/Java/Java Application** i nacisnąć klawisz **Next**. Na kolejnym formularzu w polu **Project Location** należy wybrać katalog projektu (ten sam, w którym utworzono projekt **Sklep6_SE**), w polu Project Name należy wpisać nazwę projektu (np. **Sklep6_GUI**), a w zaznaczonym polu **Create Main Class** za pomocą klawisza CheckButton wpisać **sklep_gui.GUI_main** i następnie nacisnąć klawisz **Finish**. W projekcie zdefiniowano w ten sposób pakiet s**klep_gui** oraz klasę **GUI_main** w tym pakiecie.

1.5. W celu powiązania projektu **Sklep6_SE** z projektem **Sklep6_GUI** należy kliknąć prawym klawiszem na folder **Libraries** projektu **Sklep6_GUI** i wybrać pozycję **Add Project…**, i następnie w formularzu **Add Project** wybrać projekt **Sklep6_SE** z listy projektów i zatwierdzić wybór za pomocą przycisku **AddProject JAR Files**.

1.6. Należy w pakiecie **sklep_gui** utworzyć następujące klasy Java: **Produkt_form**, **Produkty_form** oraz **Pusty_form.** Na kolejnych slajdach pokazano kod źródłowy tych klas.

1. 7. Utworzenie GUI z wykorzystaniem pakietu Swing

-zastosowanie klasy Produkt1 oraz klasy Fasada_warstwy_biznesowej w wersji SE – klasa GUI_main

```
package sklep_gui;

import java.awt.BorderLayout;
import java.awt.CardLayout;
import java.awt.Container;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import javax.swing.JFrame;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JPanel;
import javax.swing.KeyStroke;
import warstwa_biznesowa.Fasada_warstwy_biznesowej;

/**
 * @author Zofia
 */

public class GUI_main implements ActionListener {
    static   JPanel cards;          //panel, który posiada obiekt typu CardLayout, ktory przechowuje panele poszczegolnych
                                    //formularzy: Produkt_form do wprowadzania danych  produktu
                                    //oraz Produkty_form do wyswietlania danych o produktach

   static CardLayout cl;            //kolekcja paneli typu JPanel
```

1. 7. Utworzenie GUI z wykorzystaniem pakietu Swing

-zastosowanie klasy Produkt1 oraz klasy Fasada_warstwy_biznesowej w wersji SE – klasa GUI_main

```
//utworzenie 4 paneli  reprezentujących formularze aplikacji
 static Pusty_form card0 = new Pusty_form();                    //pusty formularz
 static Produkt_form card1 = new Produkt_form();                //panel formularza do wstawiania danych produktu
 static Produkty_form card2 = new Produkty_form();

 final static String PUSTY = "Pusty";
 final static String PRODUKT = "Produkt form";
 final static String PRODUKTY = "Produkty form";

 static Fasada_warstwy_biznesowej facade = new Fasada_warstwy_biznesowej();      //obiekt z warstwy biznesowej

 static public Fasada_warstwy_biznesowej getFacade() {
    return facade;
  }
```

1.7 cd

```
public JMenuBar createMenuBar() {
        JMenuBar menuBar;
        JMenu menu, submenu;
        JMenuItem menuItem;

         menuBar = new JMenuBar();                                          //tworzenie belki z glownymi .

        menu = new JMenu("A Menu");
        menu.setMnemonic(KeyEvent.VK_A);                                    //mozliwosc wyboru pozycji menu za pomocą klawiszy Alt-A
        menuBar.add(menu);                                                  //dodanie pozycji menu do obiektu typu JMenuBar

        menuItem = new JMenuItem(PRODUKT, KeyEvent.VK_P);                   //możliwosc wyboru opcji za pomocą klawiszy Alt-P
        menuItem.setAccelerator(KeyStroke.getKeyStroke(
                                        KeyEvent.VK_1, ActionEvent.ALT_MASK)); //możliwosc wyboru opcji za pomocą klawiszy Alt-1
        menuItem.addActionListener(this);                                   // dodanie obsługi zdarzenia kliknięcia na pozycję JMenuItem
        menu.add(menuItem);                                                 //dodanie pozycji menu do obiektu typu JMenu

        menuItem = new JMenuItem(PRODUKTY);
        menuItem.setMnemonic(KeyEvent.VK_R);    //możliwość wyboru opcji za pomocą klawiszy Alt-R – zamiast w konstruktorze JMenuItem
        menuItem.addActionListener(this);                                   // dodanie obsługi zdarzenia kliknięcia na pozycję JMenuItem
        menu.add(menuItem);                                                 //dodanie pozycji menu do obiektu typu JMenu
```

1. 7. cd

```
    menuItem = new JMenuItem(PUSTY);
    menuItem.setMnemonic(KeyEvent.VK_U);          //mozliwosc wyboru opcji za pomocą klawiszy Alt-U
    menuItem.addActionListener(this);             // dodanie obsługi zdarzenia kliknięcia na pozycję JMenuItem
    menu.add(menuItem);                           //dodanie pozycji menu do obiektu typu JMenu
    menu.addSeparator();

    submenu = new JMenu("A submenu");             //wykonanie do podrzędnego obiektu submenu typu JMenuJMenu
    submenu.setMnemonic(KeyEvent.VK_A);           //mozliwosc wyboru opcji za pomocą klawiszy Alt-A

    menuItem = new JMenuItem(PUSTY);
    menuItem.setAccelerator(KeyStroke.getKeyStroke(
          KeyEvent.VK_2, ActionEvent.ALT_MASK));  //mozliwosc wyboru opcji za pomocą klawiszy Alt-2
    menuItem.addActionListener(this);             // dodanie obsługi zdarzenia kliknięcia na pozycję JMenuItem
    submenu.add(menuItem);                        //dodanie pozycji menu do obiektu typu JMenu

    menuItem = new JMenuItem(PUSTY);
    menuItem.setMnemonic(KeyEvent.VK_S);          //mozliwosc wyboru opcji za pomocą klawiszy Alt-S
    menuItem.addActionListener(this);             // dodanie obsługi zdarzenia kliknięcia na pozycję JMenuItem
    submenu.add(menuItem);                        //dodanie pozycji menu do obiektu typu JMenu

    menu.add(submenu);          //Dodanie do nadrzędnego obiektu typu JMenu (menu) podrzędnego obiektu submenu typu JMenu

    menu = new JMenu("Inne Menu");                //dodanie nowego menu typu JMenu w obiekcie typu JMenuBar
    menu.setMnemonic(KeyEvent.VK_I);              //mozliwosc wyboru opcji za pomocą klawiszy Alt-I
    menuBar.add(menu);                            //dodanie pozycji menu do obiektu typu JMenuBar

    return menuBar;                               //zwrocenie wykonanego komponentu typu JMenuBar
}
```

1. 7 cd

```
    public Container createContentPane() {

                card1.init();
                card2.init();
    //wykonanie panelu cards do przechowania paneli typu Produkt_form, Produkty_form oraz Pusty_form
                cards = new JPanel(new CardLayout());
                cards.add(card0, PUSTY);                    //dodanie panelu typu Pusty_form
                cards.add(card1, PRODUKT);                  // dodanie panelu typu Produkt_form
                cards.add(card2, PRODUKTY);                 // dodanie panelu typu Produkty_form

                JPanel p1 = new JPanel();                   //utworzenie glownego panela aplikacji
                p1.add(cards, BorderLayout.CENTER);         //dodanie do glownego panelu zbioru cards innych paneli
                return p1;                                  // zwrócenie glownego panela aplikacji zawierającego kolekcję paneli
  }

public static void updateProdukty_form()
  {
     card2.table_content();                 //wywołanie metody table_content panelu typu Produkty_form
                                            //do pobrania aktualnych danych o produktach z klasy Fasada_warstwy_biznesowej
     cl.show(cards, PRODUKTY);              //wyswietlenie panelu do wyswietlenia danych produktow

  }
```

1.7. cd

```
@Override
   public void actionPerformed(ActionEvent e) {

      JMenuItem source = (JMenuItem) (e.getSource());       //obsluga klikania na pozycje menu
     cl = (CardLayout) (cards.getLayout());
      switch (source.getText()) {
         case PRODUKT:
            cl.show(cards, PRODUKT);                        //wyswietlenie panelu do wprowadzania danych produktu
            break;
         case PRODUKTY:
            updateProdukty_form();                          //wyswietlenie panelu do  prezentacji danych produktow
            break;
         case PUSTY:
            cl.show(cards, PUSTY);                          //wyswietlenie pustego panelu
            break;
         default:
            break;
      }
   }
```

1.7 cd

```
private static void createAndShowGUI() {                    //metoda tworzaca okno typu JFrame i dodanie do niego
                                                              // obiekt  typu JMenuBar utworzony w metodzie createMenuBar oraz zbior
                                                             // paneli w metodzie createContentPane
    JFrame frame = new JFrame("MenuDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(900, 400);
    GUI_main demo = new GUI_main();
    frame.setJMenuBar(demo.createMenuBar());          //dodanie komponentu typu MenuBar do okna typu JFrame
    frame.setContentPane(demo.createContentPane());
     frame.setVisible(true);                                  //wyswietlenie okna głównego typu JFrame

  }

  public static void main(String[] args) {
          //utworzenie wątku zarządzajcego zdarzeniemi utworzonego GUI ze zbiorem paneli reprezentujących formularze aplikacji

   java.awt.EventQueue.invokeLater(new Runnable() {
        @Override
            public void run() {
                createAndShowGUI();
            }
    });
  }
}
```

1. 8. Utworzenie GUI z wykorzystaniem pakietu Swing
-zastosowanie klasy Produkt1 oraz klasy Fasada_warstwy_biznesowej w wersji SE – klasa typu Produkt_form do wprowadzanie danych produktu

```
package sklep_gui;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.Date;
import java.util.GregorianCalendar;
import java.util.regex.PatternSyntaxException;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;
/**
 * @author Zofia
 */
public class Produkt_form extends JPanel implements ActionListener {

    JLabel lnazwa = new JLabel("Nazwa");                    //utworzenie etykiety pola do wprowadzania nazwy produktu
    JTextField nazwa = new JTextField(15);                  // utworzenie pola wejsciowego do wprowadzania nazwy produktu
    JLabel lcena = new JLabel("Cena");                      //utworzenie etykiety pola do wprowadzania ceny produktu
    JTextField cena = new JTextField(15);                   // utworzenie pola wejsciowego do wprowadzania ceny produktu
    JLabel lpromocja = new JLabel("Promocja");              //utworzenie etykiety pola do wprowadzania promocji produktu
    JTextField promocja = new JTextField(15);               // utworzenie pola wejsciowego do wprowadzania promocji produktu
    JLabel ldata = new JLabel("Data");                      //utworzenie etykiety pola do wprowadzania daty produkcji produktu
    JTextField data = new JTextField(15);                   // utworzenie pola wejsciowego do wprowadzania daty produkcji produktu
    JButton dodaj_produkt = new JButton("Dodaj produkt"); //utworzenie przycisku do wywołania akcji dodania produktu w aplikacji
```

1. 8. cd

```
public void init() {
      setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));      //dodanie sposobu rozmieszczania elementów formularza
      add(lnazwa);                                           //dodanie etykiety nazwy do obiektu typu JPanel
      add(nazwa);                                            //dodanie pola do wprowadzania nazwy do obiektu typu JPanel
      add(lcena);                                             //dodanie etykiety ceny do obiektu typu JPanel
      add(cena);                                              //dodanie pola do wprowadzania cenu do obiektu typu JPanel
      add(lpromocja);                                         //dodanie etykiety promocji do obiektu typu JPanel
      add(promocja);                                          //dodanie pola do wprowadzania promocji do obiektu typu JPanel
      add(ldata);                                             //dodanie etykiety daty do obiektu typu JPanel
      add(data);                                              //dodanie pola do wprowadzania daty do obiektu typu JPanel
      dodaj_produkt.addActionListener(this);                 //przycisk uruchamiający zdarzenie po kliknieciu
                                                             //obslugiwany przez obiekt typu Produkt_form za pomoca metody actionPerformed
      add(dodaj_produkt);                                     //dodanie przycisku do obiektu typu JPanel
   }
```

1. 8. cd

```
public String content_validate(JTextField val, int typ) {//walidacja danych
    String s = val.getText();
    if (s.equals("") || s.length() > 15) {                    //sprawdzenie, czy lancuch jest pusty lub dluzszy niż 15 znakow
       JOptionPane.showMessageDialog(this, "Lancuch danych jest dluzszy niz 15 lub jest pusty");
       return null;
    } else {
       s = s.replaceAll(" ", "_");                            //wyeliminowanie spacji z lancucha
       val.setText(s);
    }
    if (typ == 1) {                                           // jesli sa dane liczbowe, sprawdzenie, czy można dokonać parsowania na liczbe
       try {
          Float.parseFloat(s);
       } catch (NumberFormatException e) {
                JOptionPane.showMessageDialog(this, "Blad formatu danych liczbowych");
                return null;
       }
    }
    return s;
  }
```

1. 8. cd

```
public String[] form_produkt() {
    if (content_validate(nazwa, 0) == null)                          //walidacja danych nazwy jako lancucha
       return null;
    if (content_validate(cena, 1) == null)                           //walidacja danych ceny jako liczby
       return null;
    if (content_validate(promocja, 1) == null)                       //walidacja danych ceny jako liczby
       return null;
    String dane[] = {(String) nazwa.getText(), (String) cena.getText(), (String) promocja.getText()};          //tablica z danymi produktu
    return dane;
  }

  public Date data() {
    if (content_validate(data, 0) == null)                           //walidacja danych daty jako lancucha
       return null;
    int rok, miesiac, dzien;
    String data1 = data.getText();
    try {
       String[] data2 = data1.split("-");          //podzial lancucha daty np 12-12-2016 na tablicę lancuchow, reprezentujacych elementy daty
       rok = Integer.parseInt(data2[2]);
       miesiac = Integer.parseInt(data2[1]);
       dzien = Integer.parseInt(data2[0]);
    } catch (PatternSyntaxException | NumberFormatException | ArrayIndexOutOfBoundsException e) {       //kontrola poprawności
                JOptionPane.showMessageDialog(this, "Blad formatu daty");                                  //formatu daty
                return null;
    }
    GregorianCalendar gc = new GregorianCalendar();                  //pomocnicza klasa do utworzenia daty
    gc.set(rok, miesiac - 1, dzien);                                 //tworzenie daty
    return gc.getTime();                                             //pobranie daty jako obiektu typu Date
  }
```

1. 8. cd

```
 @Override
   public void actionPerformed(ActionEvent evt) {              //obsluga zdarzenia kliknięcia na przycisk "Dodaj_produkt"
      String[] dane = form_produkt();                          //utworzenie tablicy z danymi produktu: nazwa, cena, promocja
      if (dane == null) {
         return;
      }
      Date data_ = data();                                     //utworzenie daty
      if (data_ == null) {
         return;
      }
      GUI_main.getFacade().utworz_produkt(dane, data_);   // wywołanie metody logiki biznesowej tworzacej obiekt typu Produkt1
   }
}
```

1. 9. Utworzenie GUI z wykorzystaniem pakietu Swing
-zastosowanie klasy Produkt1 i klasy Fasada_warstwy_biznesowej w wersji SE – klasa typu Produkty_form do wyświetlania danych produktów w tabeli

```
package sklep_gui;

import java.awt.Dimension;
import java.awt.Graphics;
import java.util.ArrayList;
import java.util.Iterator;
import javax.swing.BoxLayout;
import javax.swing.JComboBox;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import javax.swing.table.AbstractTableModel;
/**
 *
 * @author Zofia
 */
public class Produkty_form extends JPanel {

    private JTable tabela_produktow;                    //komponent typu tabela do wyświetlania danych produktów
    MyTableModel model;                                 //model widoku
    JComboBox lista_produktow;                          //lista wyswietlajaca dane produktów
```

1. 9. cd

```
public void init() {
      setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));
      model = new MyTableModel();                                   //tworzenie modelu danych tabeli
      tabela_produktow = new JTable(model);                         // utworzenie tabeli i przekazanie jej modelu z danymi produktow
      table_content();                                              //wypelnienie danymi produktow tabeli
      tabela_produktow.setPreferredScrollableViewportSize(new Dimension(800, 100));
      tabela_produktow.setFillsViewportHeight(true);
      tabela_produktow.getSelectionModel().addListSelectionListener(new RowListener());     //dodanie słuchacza zdarzen do obslugi
                                                                                            //zmiany wyboru wiersza
      add(new JScrollPane(tabela_produktow));                       //dodanie panelu przewijania tabdli
      JLabel lprodukty = new JLabel("Produkty");
      add(lprodukty);
      lista_produktow = new JComboBox();
      add(lista_produktow);
   }

 void table_content() {                                             //wypelnienie tablicy typu JTable danymi produktow
      ArrayList<ArrayList<String>> produkty = GUI_main.getFacade().items();
      model.setData(produkty);
      tabela_produktow.repaint();
   }
```

1. 9. cd

```
private void list_content(ArrayList<ArrayList<String>> col, JComboBox list) {
    ArrayList<String> s;                                          //wypelnienie listy typu JComboBox danymi produktow
    list.removeAllItems();
    Iterator<ArrayList<String>> iterator = col.iterator();
    while (iterator.hasNext()) {
       s = iterator.next();
       list.addItem(s);
    }
  }

  void print_produkty() {                    //metoda wypelniajaca listę typu JComboBox danymi produktow pobranymi metodą items.
    ArrayList<ArrayList<String>> help3 = GUI_main.getFacade().items();          // pobranie danych produktow metoda items
    if (help3 != null) {
       list_content(help3, lista_produktow);                                    //wypelnianie listy typu JComboBox danymi produktow
    }
  }

  private class RowListener implements ListSelectionListener {          //klasa wewnetrzna do obslugi zdarzen zmiany wyboru wiersza tabeli

    @Override
    public void valueChanged(ListSelectionEvent event) {                //metoda do obsługi zdarzenia zmiany  wybranego wiersza tabeli
       if (event.getValueIsAdjusting()) {                               //za pomocą kliknięcia myszy na wybrany rowek
          return;
       }
       print_produkty();                    // po zmianie wiersza wykonanie metody wypelniajacej listę typu JComboBox danymi produktow
    }
  }
```

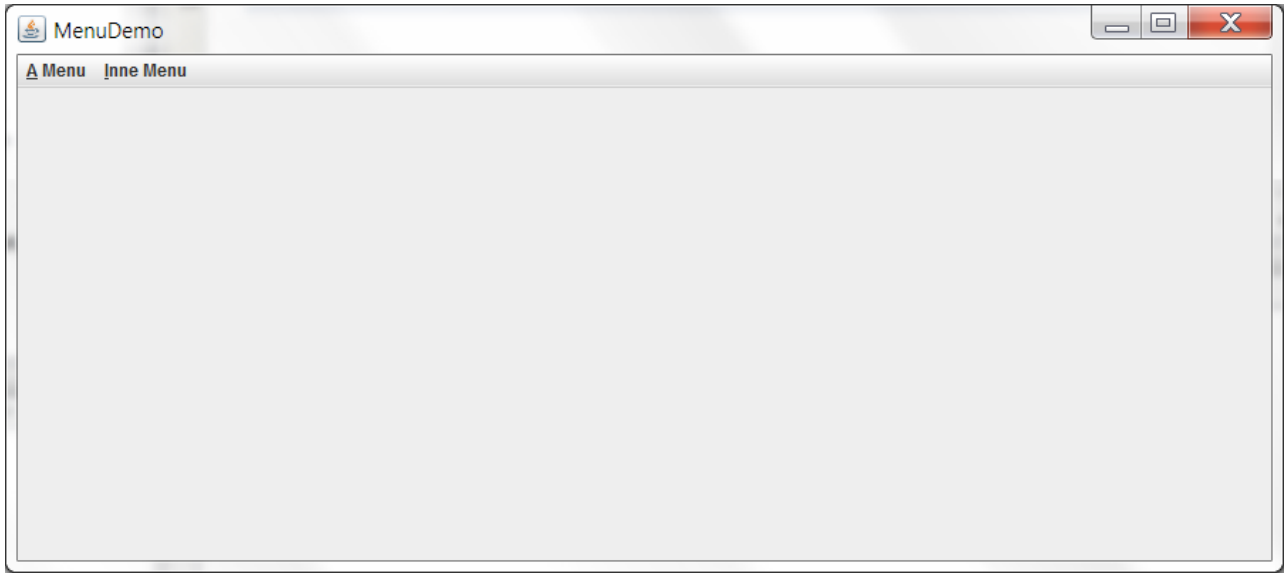
1. 9. cd

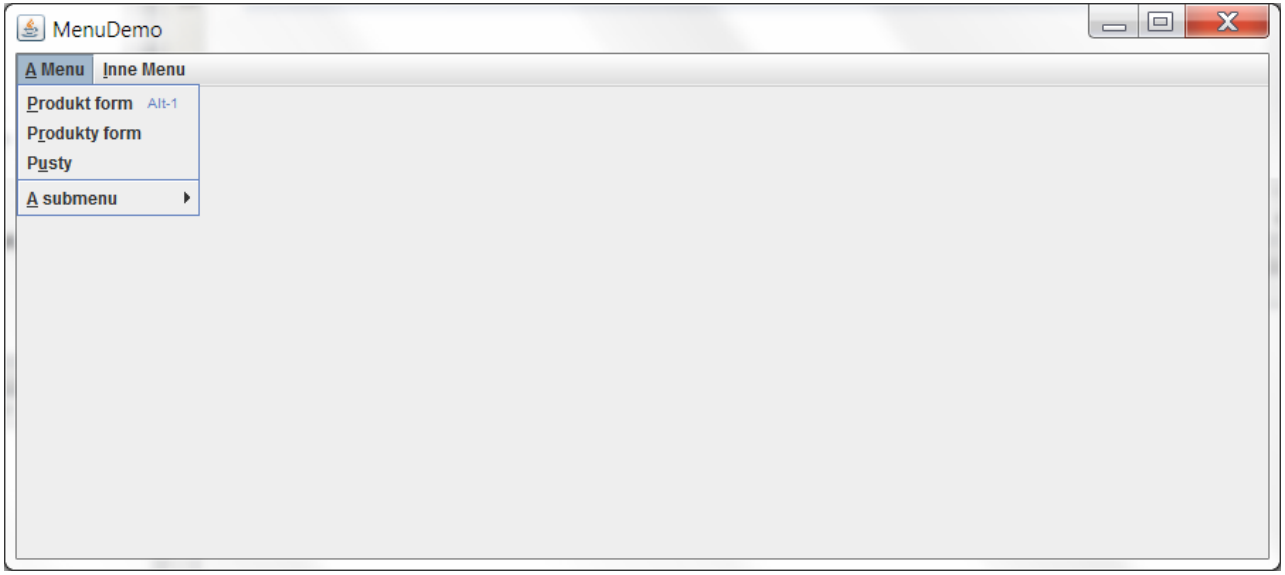
```
class MyTableModel extends AbstractTableModel {          //klasa wewnetrzna reprezentujaca model danych obiektu typu JTable

    private final String[] columnNames = {"Id produktu", "Nazwa","Cena",          //nazwy kolumn tabeli
                                          "Promocja", "Data", "Cena brutto"};
    private ArrayList<ArrayList<String>> data;          //dane tabeli-kazdy element zawiera elementy wiersza, jako kolekcja  lancuchow

    public void setData(ArrayList<ArrayList<String>> val) {          //wstawienie danych modelu
       data = val;
    }
    @Override
    public int getColumnCount() {
       return columnNames.length;          //liczba kolumn
    }
    @Override
    public int getRowCount() {
       return data.size();          //liczba rowkow
    }
    @Override
    public Object getValueAt(int row, int col) {
       return data.get(row).get(col);          //pobrane elementu z podanej komorki tabeli w wieszu row i kolumnie col
    }
  @Override                                    //metoda umozliwia przypisanie nazw z tablicy columnNames
    public String getColumnName(int col) {     //do nazw kolumn wyświetlanej tabeli
       return columnNames[col];
    }
  }
}
```

1. 10. Utworzenie GUI z wykorzystaniem pakietu Swing
-zastosowanie klasy Produkt1 i klasy Fasada_warstwy_biznesowej w wersji SE – klasa typu Pusty_form
cd

```
package sklep_gui;

import javax.swing.JPanel;

/**
 *
 * @author Zofia
 */
public class Pusty_form extends JPanel {

    public Pusty_form() {
    }

}
```

1. 11. Prezentacja działania aplikacji - widok po uruchomieniu aplikacji i prezentacja listy rozwijanej, wybranej z pozycji **A Menu**, z pozycjami umożliwiającymi wybór jednego z formularzy

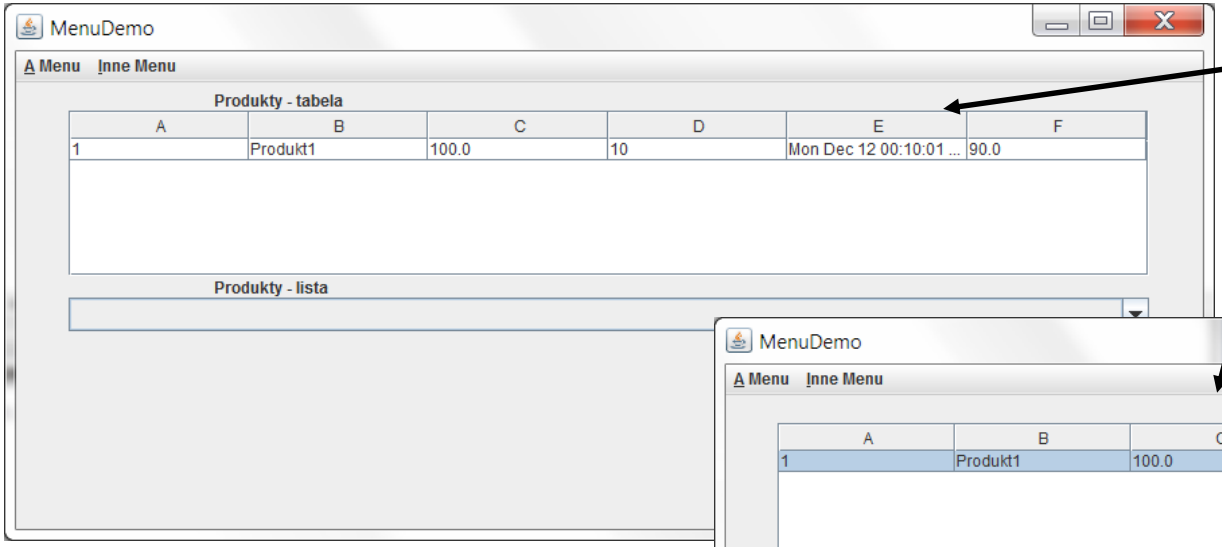




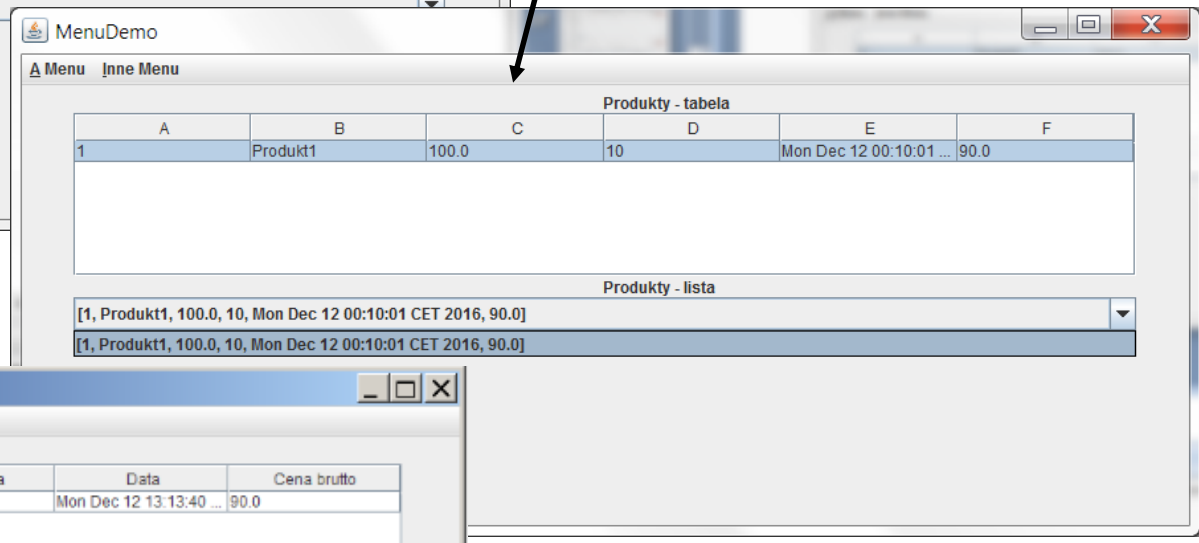
Prezentacja formularza do wprowadzania danych produktu po wybraniu pozycji **Produkt_form**

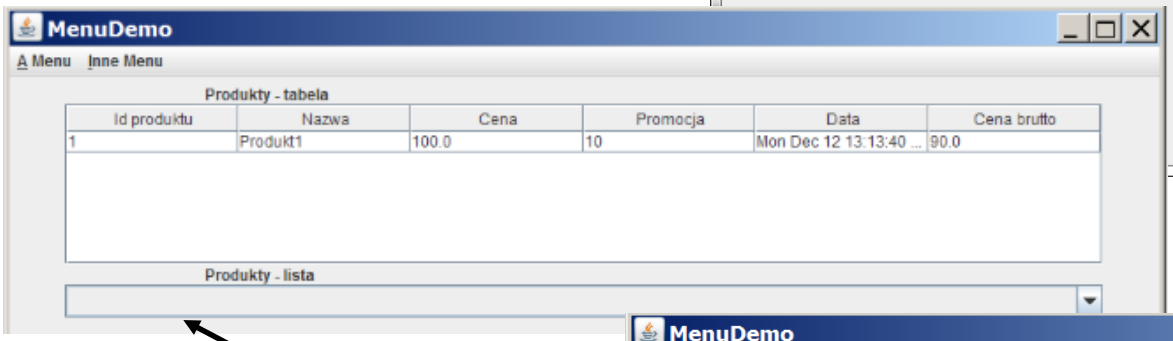
Prezentacja wyboru z listy rozwijanej pozycji **Produkty_form** po kliknięciu na przycisk **Dodaj_produkt** na formularzu **Produkt_form**

Widok formularza **Produkty_form**

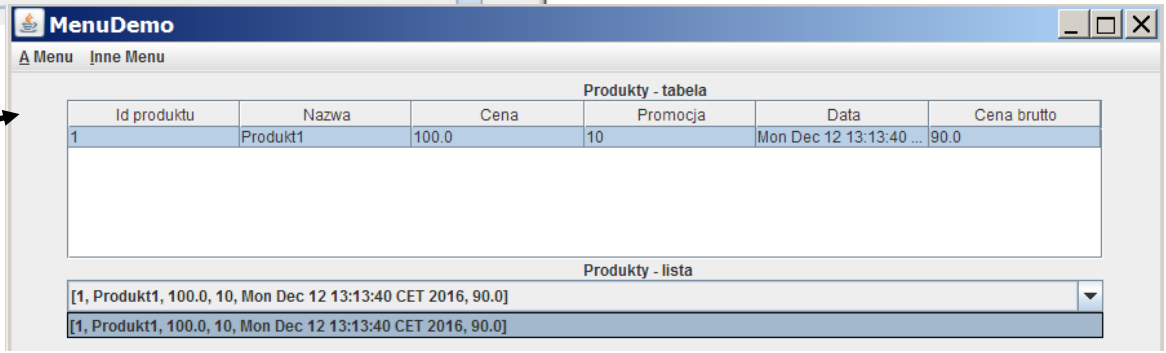


Prezentacja wypełnienia widoku **Produkty-lista** po kliknięciu na rowek tabeli z danymi wprowadzonego produktu - przed dodaniem metody geColumnName w klasie MyTableModel (str. 22)

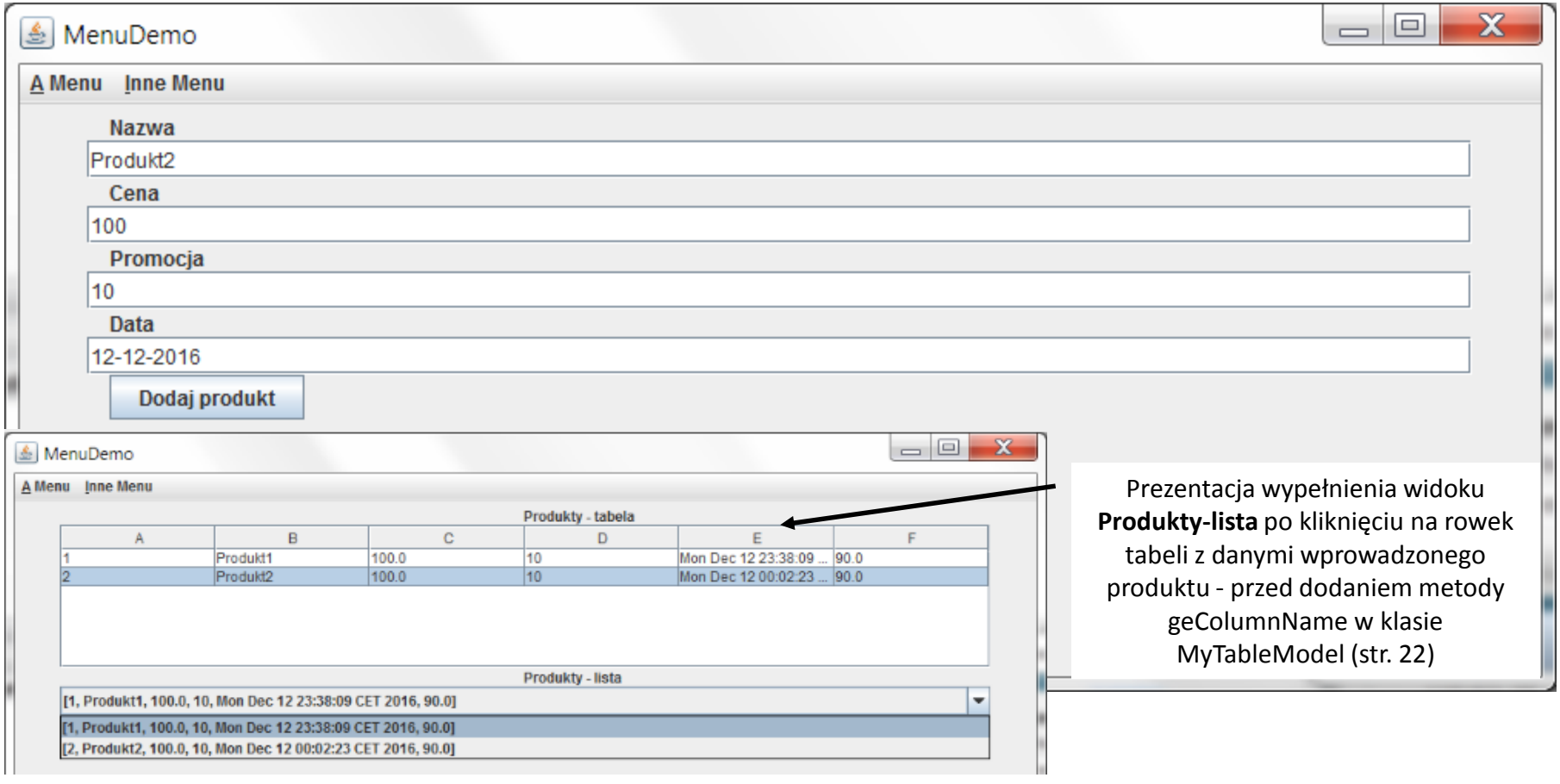




Prezentacja wypełnienia widoku **Produkty-lista** po kliknięciu na rowek tabeli z danymi wprowadzonego produktu – po dodaniu metody geColumnName w klasie MyTableModel (str. 22)

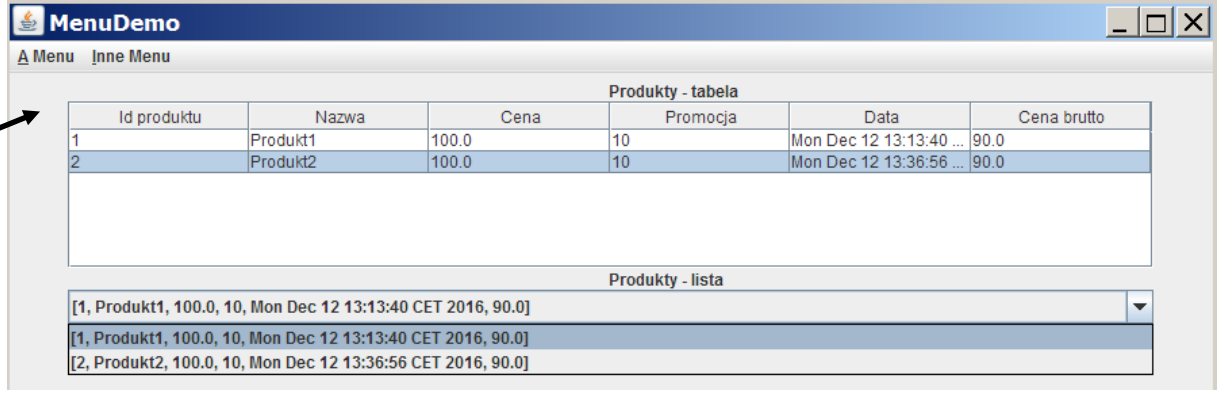


Widok formularza **Produkty_form** po wprowadzeniu nowych danych i kliknięciu na nowy rowek w tabeli

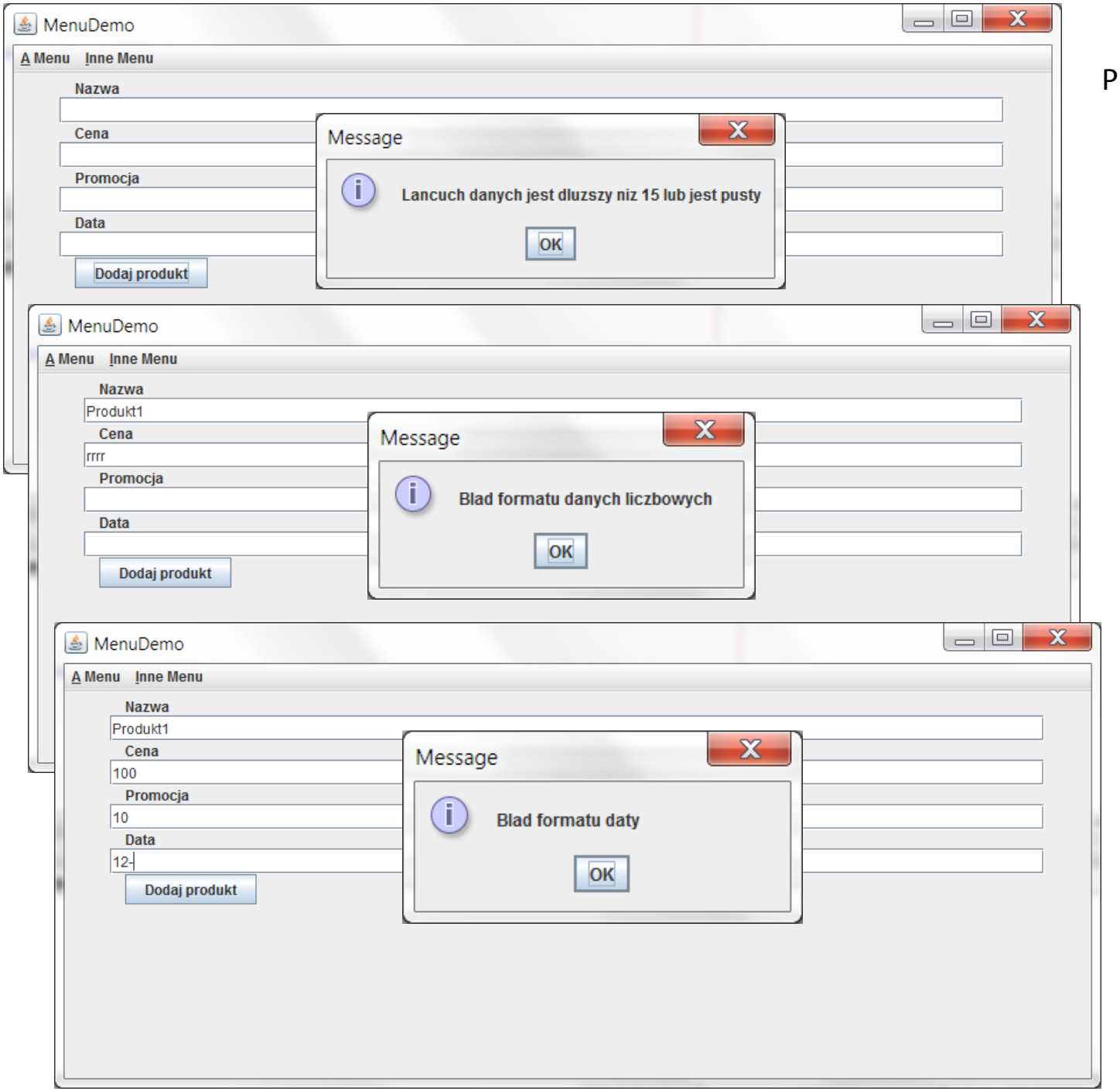


Prezentacja wypełnienia widoku **Produkty-lista** po kliknięciu na rowek tabeli z danymi wprowadzonego produktu - przed dodaniem metody geColumnName w klasie MyTableModel (str. 22)

Prezentacja wypełnienia widoku **Produkty-lista** po kliknięciu na rowek tabeli z danymi wprowadzonego produktu - po dodaniu metody geColumnName w klasie MyTableModel (str. 22)



Prezentacja obsługi błędu formatu danych



Zadanie 2 do wykonania

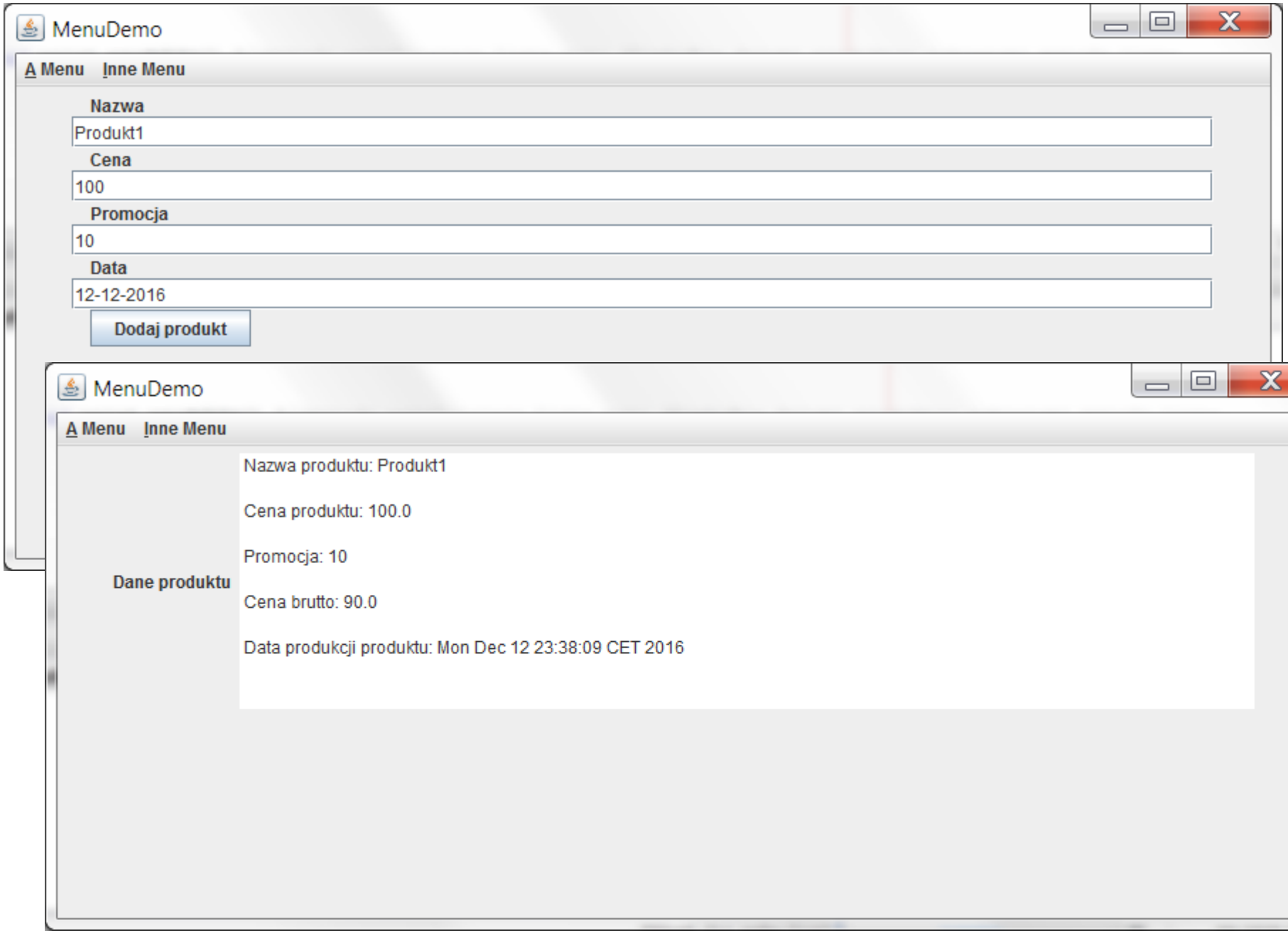
1. Należy dodać w programie brakujące atrybuty w klasie **Produkt1**, zdefiniowane podczas zajęć z TINT
2. Należy uzupełnić formularz **Produkt_form** w celu wprowadzenia dodanych atrybutów
3. Należy uzupełnić formularz **Produkty_form** w celu wyświetlenia dodanych atrybutów w tabeli
4. Należy dodać i wywołać dodatkowy formularz **Rezultat2_form,** zawierający podobną zawartość jak strona **Rezultat2.xhtml,** podczas kliknięcia na przycisk „Dodaj_produkt" w formularzu **Produkt_form** (czyli w metodzie **actionPerfomed**) np. wywołując zdefiniowaną metodę:

   GUI_main.updateRezultat2_form();

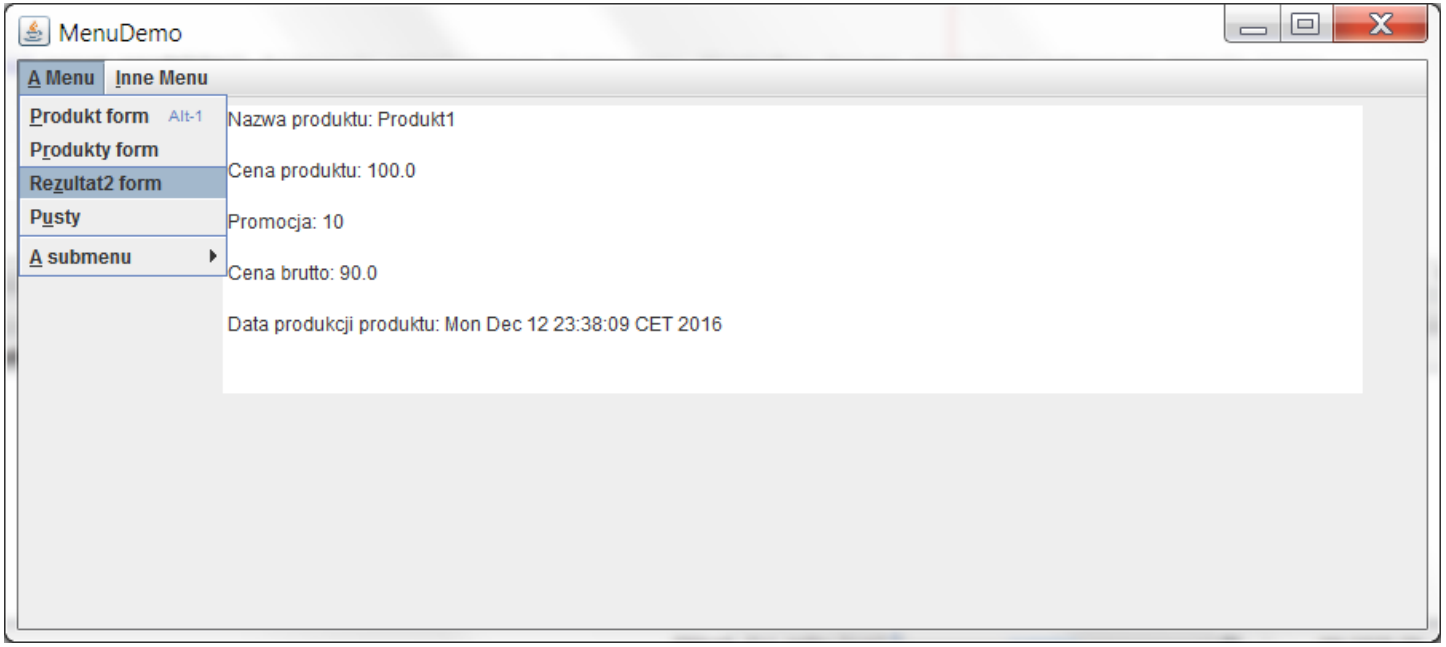
   z klasy **GUI_main**. Metoda **updateRezultat2_form** powinna zawierać wywołanie metody z formularza **Rezultat2_form** aktualizującej jego widok. Ta metoda z kolei, w celu aktualizacji danych widoku, wywołuje metodę **dane_produktu()** z obiektu klasy **Fasada_warstwy_biznesowej** i tworzy tekst wyświetlany w komponencie typu **JTextArea** formularza **Rezultat2_form**. Ta sama metoda powinna być wykorzystana podczas wyświetlania danych wprowadzonego produktu w formularzu **Rezultat2_form** podczas wywołania z listy rozwijanej **A Menu** (punkt 5).
5. Należy dodać wywołanie tego dodatkowego formularza, **Rezultat2_form** również z listy **A Menu,** wyświetlanego podobnie jak pozostałe formularze**.**

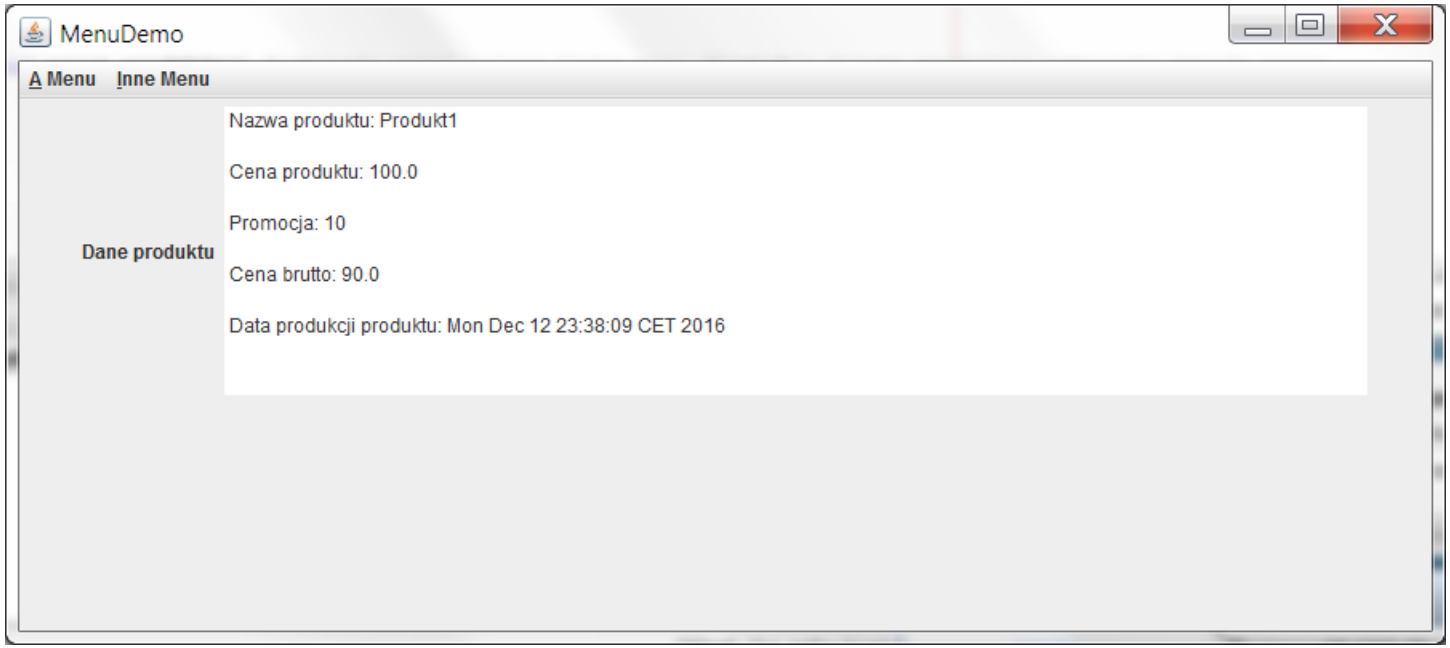
Dalej pokazano przykładowe działanie aplikacji po dodaniu nowego formularza.

Prezentacja wyniku działania aplikacji po dodaniu nowego formularza



Prezentacja wyniku działania aplikacji po dodaniu nowego formularza (cd)





Zadanie 3 do wykonania na ocenę 5.5.

- Należy dodać do zadania 2 formularz rysujący wykres (grafika 2D lub 3D), przedstawiający, ile wprowadzono produktów w zadaniach przedziałach cen lub promocji. Do prezentacji należy dodać nowy formularz np. Wykres_form wywoływany podobnie jak pozostałe formularze.

lub

- Należy dodać do zadania **3 lub 4 z lab5** formularz rysujący wykres (grafika 2D lub 3D), przedstawiający, ile różnych typów figur narysowano; typu **Punkt**, typu **Kwadrat** i typu **Prostokat**.