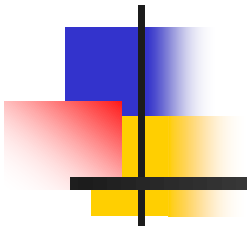
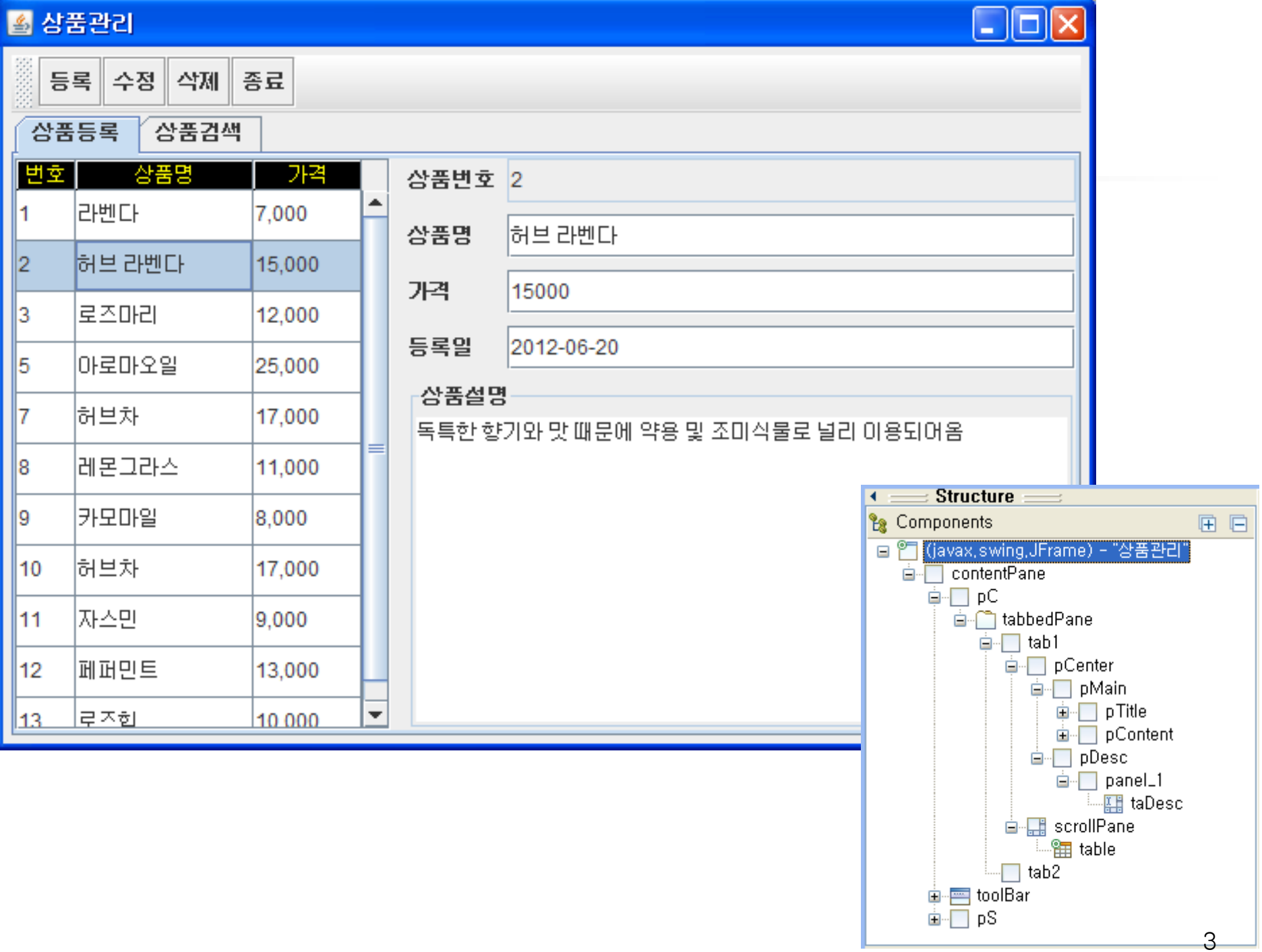


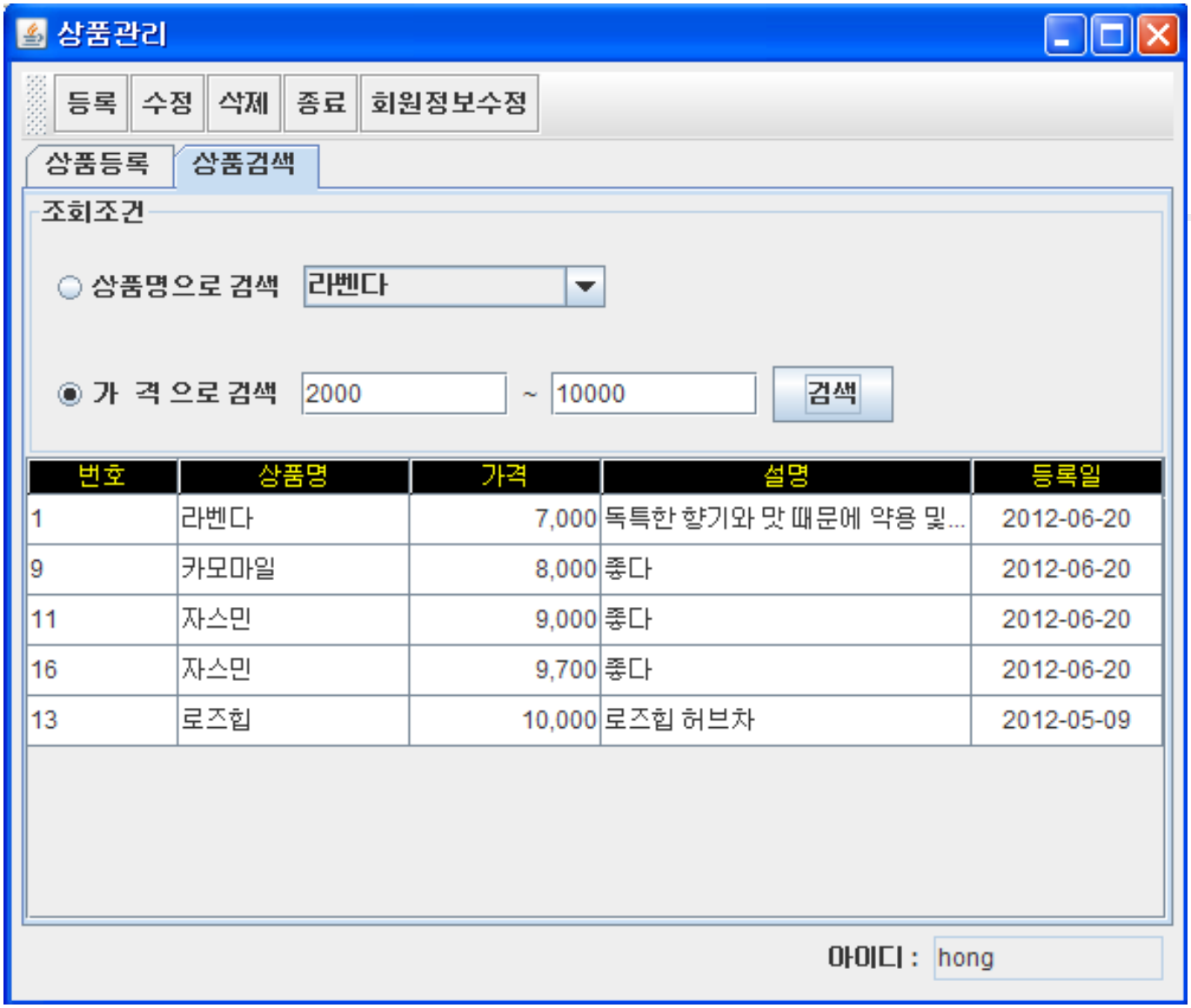
java 19강-Swing2

양 명 숙
[now4ever7@gmail.com]

상품 테이블

```
--상품테이블

create table product
(
    no              number  primary key,        --일련번호
    productName     varchar2(50) Not Null,      --상품이름
    price           number,                     --소비자가
    --company       varchar2(50),               --제조회사
    description     varchar2(255),              --설명
    regDate         date        default  sysdate --등록일
);

create sequence product_seq
start with 1
increment by 1
nocache;
```

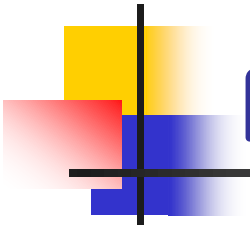

상품관리
등록
수정
삭제
종료
상품등록
상품검색
번호
상품명
가격
1
라벤다
7,000
2
허브 라벤다
15,000
3
로즈마리
12,000
5
아로마오일
25,000
7
허브차
17,000
8
레몬그라스
11,000
9
카모마일
8,000
10
허브차
17,000
11
자스민
9,000
12
페퍼민트
13,000
13
로즈힙
10,000
상품번호
2
상품명
허브 라벤다
가격
15000
등록일
2012-06-20
상품설명
독특한 향기와 맛 때문에 약용 및 조미식물로 널리 이용되어옴
Structure
Components
(javax.swing.JFrame) - "상품관리"
contentPane
pC
tabbedPane
tab1
pCenter
pMain
pTitle
pContent
pDesc
panel_1
taDesc
scrollPane
table
tab2
toolBar
pS


상품관리
등록
수정
삭제
종료
회원정보수정
상품등록
상품검색
조회조건
상품명으로 검색
라벤다
가 격 으로 검색
2000
~
10000
검색
번호
상품명
가격
설명
등록일
1
라벤다
7,000
독특한 향기와 맛 때문에 약용 및...
2012-06-20
9
카모마일
8,000
좋다
2012-06-20
11
자스민
9,000
좋다
2012-06-20
16
자스민
9,700
좋다
2012-06-20
13
로즈힙
10,000
로즈힙 허브차
2012-05-09
아이디 :
hong

DBUtil

```
public class DBUtil {
    //static 초기화블럭
    static{
        //1. 드라이버 로딩 - 한번만 처리하면 됨
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            System.out.println("드라이버 로딩 성공!!");
        } catch (ClassNotFoundException e) {
            System.out.println("드라이버 로딩 실패!!");
            e.printStackTrace();
        }
    }
    public static Connection getConnection() throws SQLException{
        //2. db서버에 연결할 Connection 객체 생성
        String url = "jdbc:oracle:thin:@userpc:1521:xe";
        String uid = "javauser", upwd = "javauser123";
        Connection conn = getConnection(url, uid, upwd);
        return conn;
    }
    public static Connection getConnection(String url, String uid, String upwd) throws SQLException{
        //2. db서버에 연결할 Connection 객체 생성
        Connection conn = DriverManager.getConnection(url, uid, upwd);
        System.out.println("db연결 결과 : "+conn);

        return conn;
    }
```

```
    public static Connection getConnection(String uid, String upwd) throws SQLException{
        //2. db서버에 연결할 Connection 객체 생성
        String url = "jdbc:oracle:thin:@localhost:1521:orcl11";
        Connection conn = getConnection(url, uid, upwd);
        return conn;
    }
    public static void dbClose(PreparedStatement ps, Connection conn) throws SQLException{
        if (ps!=null) ps.close();
        if (conn!=null) conn.close();
    }
    public static void dbClose(ResultSet rs, PreparedStatement ps, Connection conn)
                    throws      SQLException{
        if (rs!=null) rs.close();
        if (ps!=null) ps.close();
        if (conn!=null) conn.close();
    }
}
```



PdDTO

```
public class PdDTO {
    private int no;
    private String productName;
    private int price;
    private String description;
    private Timestamp regdate;

    …
```

PdDAO

```
public class PdDAO {
    public ArrayList<PdDTO> selectAll() throws SQLException{ //상품 전체조회
        Connection conn = null;
        PreparedStatement ps = null;
        ResultSet rs = null;
        //여러개의 레코드(여러개의 DTO)를 담을 컬렉션
        ArrayList<PdDTO> alist = new ArrayList<PdDTO>();
        try{        //1. 드라이버 로딩 2. db에 연결할 Connection객체 생성
                    conn = DBUtil.getConnection();
                    //3. sql문장을 처리할 PreparedStatement 객체 생성
                    String sql = "select * from product order by no desc";
                    ps = conn.prepareStatement(sql);
                    //4. 실행
                    rs = ps.executeQuery();
                    while(rs.next()){
                                int no = rs.getInt("no");
                                String productName = rs.getString("productName");
                                int price = rs.getInt("price");
                                String description = rs.getString("description");
                                Timestamp regdate = rs.getTimestamp("regdate");

                                PdDTO dto = new PdDTO(no, productName, price, description, regdate);
                                alist.add(dto);
                    }
                    System.out.println("상품 전체 조회 결과 : alist.size() = "+alist.size());
        }finally{   //5. 자원반납, 해제
                    DBUtil.dbClose(rs, ps, conn);
        }
        return alist;
    }
```

```
public ArrayList<String> selectPdName() throws SQLException{
	//combo에서 보여줄 상품명 조회
	Connection conn = null;
	PreparedStatement ps = null;
	ResultSet rs = null;
	ArrayList<String> vec = new ArrayList<String>();
	try{
		//1,2. conn
		conn = DBUtil.getConnection();
		//3. ps
		String sql = "select distinct productname from product";
		ps = conn.prepareStatement(sql);

		//4. exec
		rs = ps.executeQuery();
		while(rs.next()){
			String pdName = rs.getString(1);
			vec.add(pdName);
		}
		System.out.println("상품명 조회 결과 vec.size() = "+vec.size());

	}finally{
		//5. 자원반납
		DBUtil.dbClose(rs, ps, conn);
	}
	return vec;
}
```

ProductGUI

```
public class ProductGUI extends JFrame implements ActionListener{
	private JToolBar toolBar;
	private JTabbedPane tabbedPane;
	//tab1
	private JPanel pDesc, pTab1, pTab2, ptab1Center, pTab1C, pTab1S, pW, pC;
	private JTextField tfNo, tfPdName, tfPrice, tfRegdate;
	private JTextArea taDescription;
	private JLabel lb1, lb2, lb3, lb4;

	private JButton btAdd, btEdit, btDel, btExit;
	private JScrollPane scrollPane;
	private JTable table;
	private DefaultTableModel model;
	private String[] colNames = {"번호","상품명","가격"};
	private String[][] dataArr;
	//tab2
	private JPanel pTab2Main, pTab2N, pTab2C, pTab2N1, pTab2N2;
	private JTextField tfPrice1, tfPrice2;
	private JLabel lbPrice;
	private JRadioButton rdPdName, rdPrice;
	private JComboBox<String> cbPdName;
	private JButton btSearch;
	private ButtonGroup btGroup; //라디오버튼을 하나의 그룹으로 묶어줌
	private JScrollPane scrollPane2;
	private JTable table2;
	private DefaultTableModel model2;
	private String[] colNames2 = {"번호","상품명","가격","설명","등록일"};
	private String[][] dataArr2;
	PdDAO dao;
	private DecimalFormat df = new DecimalFormat("#,###"); //1,000단위 구분
```

```
    public ProductGUI(){
//          super("상품관리");
            this.setTitle("상품관리");
            dao = new PdDAO();
            init();
            addEvent();
            showAllPd(); //전체 상품 조회
    }//생성자
    private void showAllPd() {
            //1. 사용자로부터 입력받기
            try {
                    //2. db작업 - select
                    ArrayList<PdDTO> alist = dao.selectAll();

                    //3. 결과처리
                    //데이터가 없는 경우
                    if(alist == null || alist.isEmpty()){
                            System.out.println("데이터가 존재하지 않습니다");
                            return;
                    }
                    //모델에 담을 2차원 배열 - 결과 데이터가 들어감
                    dataArr = new String[alist.size()][colNames.length];

                    for(int i=0;i<alist.size();i++){
                            PdDTO dto = alist.get(i);
                            int price = dto.getPrice();
                            String sPrice = df.format(price);

                            dataArr[i][0] = dto.getNo()+"";                 //0컬럼 => 0번째 열
                            dataArr[i][1] = dto.getProductName();           //1컬럼 => 1번째 열
                            dataArr[i][2] = sPrice;                         //2컬럼 => 2번째 열 1,000단위 구분
```

```
                    }//for
                    //모델에 데이터 넣기
                    model.setDataVector(dataArr, colNames);

                    //table에 모델 연결
                    table.setModel(model);

                    //각 컬럼 사이즈 조절하기
                    table.getColumnModel().getColumn(0).setPreferredWidth(40);
                    table.getColumnModel().getColumn(1).setPreferredWidth(120);
//                  table.getColumnModel().getColumn(2).setPreferredWidth(60);

                    //가격 오른쪽 정렬하기
                    DefaultTableCellRenderer dtcr = new DefaultTableCellRenderer();
                    dtcr.setHorizontalAlignment(SwingConstants.RIGHT);
                    table.getColumnModel().getColumn(2).setCellRenderer(dtcr);
            } catch (SQLException e) {              e.printStackTrace();        }
    }
    private void init() {
            //1.ToolBar에 버튼추가
            toolBar = new JToolBar();
            btAdd = new JButton("등록");
            btEdit = new JButton("수정");
            btDel = new JButton("삭제");
            btExit = new JButton("종료");

            toolBar.add(btAdd);
            toolBar.add(btEdit);
            toolBar.add(btDel);
            toolBar.add(btExit);
```

```
this.add(toolBar,"North"); //프레임 북쪽에 toolbar 추가

//2. tabbedPane에 tab(Panel)2개 추가
tabbedPane = new JTabbedPane();

pTab1 = new JPanel();
pTab1.setLayout(new BorderLayout(10, 10));

tabbedPane.addTab("상품등록", pTab1);

pTab2 = new JPanel();
pTab2.setLayout(new BorderLayout());

tabbedPane.addTab("상품검색", pTab2);
this.add(tabbedPane, "Center"); //프레임 중앙에 tabbedPane 추가

//3. 상품등록 tab에 table(West), panel(Center) 추가하기
//1) 왼쪽에 table 추가
table = new JTable();
scrollPane = new JScrollPane();
model = new DefaultTableModel();
model.addColumn("번호");
model.addColumn("상품명");
model.addColumn("가격");
table.setModel(model);
//table header 디자인하기
table.getTableHeader().setBackground(Color.black);
table.getTableHeader().setForeground(Color.yellow);
table.getTableHeader().setReorderingAllowed(false);
table.setRowHeight(20);
```

```
table.getTableHeader().setPreferredSize(
    new Dimension(scrollPane.getWidth(), 35));
```

```
//scrollPane에 table 추가
scrollPane.setViewportView(table);
//scrollPane 크기조절
scrollPane.setPreferredSize(new Dimension(300, 300));

//tab1 서쪽에 table이 담긴 scrollPane 추가
pTab1.add(scrollPane,"West");

//2) 가운데에 panel(textField, textArea) 추가
ptab1Center = new JPanel(new BorderLayout(10, 10));
pTab1.add(ptab1Center, "Center");
pTab1C = new JPanel(new BorderLayout()); //textField들를 담을 패널
pTab1S = new JPanel(new BorderLayout()); //textArea를 담을 패널
pC = new JPanel(new GridLayout(4, 1, 5, 5));
pW = new JPanel(new GridLayout(4, 1, 5, 5));
//pW에 label 추가
lb1 = new JLabel("상품번호");
lb2 = new JLabel("상품명");
lb3 = new JLabel("가격");
lb4 = new JLabel("등록일");
pW.add(lb1);
pW.add(lb2);
pW.add(lb3);
pW.add(lb4);
//pC에 textField 추가
tfNo = new JTextField();
tfPdName = new JTextField();
tfPrice = new JTextField();
tfRegdate = new JTextField();
tfNo.setEditable(false);
tfRegdate.setEditable(false);
```

```
pC.add(tfNo);
pC.add(tfPdName);
pC.add(tfPrice);
pC.add(tfRegdate);

//pTab1S에 textArea추가
taDescription = new JTextArea();
taDescription.setLineWrap(true); //문장이 길 시 자동으로 줄바꿈
pDesc = new JPanel(new BorderLayout());
pDesc.setBorder(new TitledBorder("상품설명"));
pDesc.add(taDescription);

pTab1S.add(pDesc);
pTab1S.setPreferredSize(new Dimension(400, 320));
pTab1C.add(pC, "Center");
pTab1C.add(pW, "West");
ptab1Center.add(pTab1C,"Center");
ptab1Center.add(pTab1S,"South");

//4. 상품검색 tab의 North에 검색어 추가, Center에 table추가
pTab2Main = new JPanel(new BorderLayout());
pTab2Main.setBorder(new TitledBorder("조회조건"));
pTab2.add(pTab2Main);

//1) North에 pTab2N 패널 추가
pTab2N = new JPanel(new GridLayout(2, 1, 5, 5));
pTab2Main.add(pTab2N, "North");

//pTab2N의 1행에는 pTab2N1 패널 추가 - 상품명으로 조회
pTab2N1 = new JPanel(new FlowLayout(FlowLayout.LEFT));
pTab2N.add(pTab2N1);
```

```
//1행에 radio, combo 추가
rdPdName = new JRadioButton("상품명으로 검색",true);
cbPdName = new JComboBox<String>();
pTab2N1.add(rdPdName);
pTab2N1.add(cbPdName);

//2행에는 pTab2N2 패널 추가 - 가격으로 조회
pTab2N2 = new JPanel(new FlowLayout(FlowLayout.LEFT));
pTab2N.add(pTab2N2);

//2행에 radio, textfield, button 추가
rdPrice = new JRadioButton();
rdPrice.setText("가격으로 검색");

//라디오버튼을 버튼 그룹으로 묶어서 하나만 선택되도록 한다
btGroup = new ButtonGroup();
btGroup.add(rdPdName);
btGroup.add(rdPrice);

tfPrice1 = new JTextField(10);
tfPrice2 = new JTextField(10);
lbPrice = new JLabel(" ~ ");
btSearch = new JButton("검색");

pTab2N2.add(rdPrice);
pTab2N2.add(tfPrice1);
pTab2N2.add(lbPrice);
pTab2N2.add(tfPrice2);
pTab2N2.add(btSearch);
//2) Center에 pTab2C 패널 추가
pTab2C = new JPanel(new BorderLayout());
```

```
        pTab2Main.add(pTab2C, "Center");

        //pTab2C패널에 (table이 담긴)scrollPane 추가
        scrollPane2 = new JScrollPane();
        table2 = new JTable();

        //table header 디자인
        model2 = new DefaultTableModel();
        model2.addColumn("번호");
        model2.addColumn("상품명");
        model2.addColumn("가격");
        model2.addColumn("설명");
        model2.addColumn("등록일");
        table2.setModel(model2);

        table2.getTableHeader().setBackground(Color.black);
        table2.getTableHeader().setForeground(Color.yellow);
        table2.getTableHeader().setReorderingAllowed(false);
        table2.setRowHeight(20);

        scrollPane2.setViewportView(table2);

        pTab2C.add(scrollPane2);

        this.setSize(700, 600);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
private void addEvent() {
        table.addMouseListener(new EventHandler());
        btAdd.addActionListener(this);
        btEdit.addActionListener(this);
```

```
btDel.addActionListener(this);
btExit.addActionListener(this);
btSearch.addActionListener(this);

//tab을 마우스로 클릭한 경우
tabbedPane.addMouseListener(new MouseAdapter() {
        public void mouseClicked(MouseEvent e) {
                int index = ((JTabbedPane)e.getSource()).getSelectedIndex();
                if(index==1){ //상품검색 tab을 클릭한 경우
                        showPdName();
                }
                if(index==0){
                        clear_tf();
                }else if(index==1){
                        clear_tf2();
                }
        }
});
//콤보에서 항목을 선택하면 상품명으로 검색 라디오 버튼이 체크되도록 하기
cbPdName.addItemListener(new ItemListener() {
        public void itemStateChanged(ItemEvent e) {
                if(e.getSource()==cbPdName){
                        if(e.getStateChange()==ItemEvent.SELECTED){
                                rdPdName.setSelected(true);
                        }
                }
        }
});
//가격2 텍스트박스에 값이 입력되면 가격1 텍스트박스에도 값이 있는지 확인해서 있으면 가격으로
//검색 라디오 버튼을 체크한다
tfPrice2.addKeyListener(new EventHandler());
```

```
        tfPrice1.addKeyListener(new EventHandler());
}
protected void clear_tf2() {
        tfPrice1.setText("");
        tfPrice2.setText("");
        rdPdName.setSelected(true);
}
protected void showPdName() {
        try {
                Vector<String> vec = dao.selectPdName();
                DefaultComboBoxModel<String> cModel = new DefaultComboBoxModel<String>(vec);
                cbPdName.setModel(cModel);
        } catch (SQLException e) {              e.printStackTrace();        }
}
public static void main(String[] args) {
        ProductGUI f = new ProductGUI();
        f.setVisible(true);
}
class EventHandler extends MouseAdapter implements KeyListener{
        public void mousePressed(MouseEvent e) {
                //table에서 특정 행을 마우스로 누르면 이벤트 발생
                int row = table.getSelectedRow();

                Object objVal = table.getValueAt(row, 0);
                int no = Integer.parseInt((String)objVal); //Object - String 은 부모자식관계이므로 가능!!

                //해당 행의 0번째 컬럼(번호)의 값을 읽어서
                //db에서 no(번호)에 해당하는 상품을 조회해서 출력한다
                showByNo(no);
        }
```

cbPdName.setSelectedIndex(0);

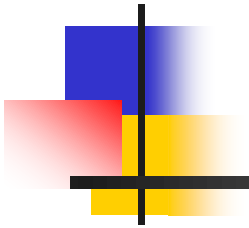
```
        public void keyReleased(KeyEvent e) {
                if(e.getSource()==tfPrice2 || e.getSource()==tfPrice1){
                        int len1 = tfPrice1.getText().length();
                        int len2 = tfPrice2.getText().length();
                        if(len2>0 && len1>0)  rdPrice.setSelected(true);
                }
        }
}
public void actionPerformed(ActionEvent e) {
        if(e.getSource()==btAdd){
                addPd();
                clear_tf(); //화면 지우기
                showAllPd(); //전체 상품 최신정보 조회
        }else if(e.getSource()==btEdit){
                editPd();
                clear_tf(); //화면 지우기
                showAllPd(); //전체 상품 최신정보 조회
        }else if(e.getSource()==btDel){
                int res = JOptionPane.showConfirmDialog(this, "삭제하시겠습니까?",
                                "데이터 삭제", JOptionPane.YES_NO_OPTION);
                if(res==JOptionPane.YES_OPTION){
                        delPd();
                        clear_tf(); //화면 지우기
                        showAllPd(); //전체 상품 최신정보 조회
                }
        }else if(e.getSource()==btExit){
                System.exit(0);
        }else if(e.getSource()==btSearch){
                search();
        }
}
```

```
private void search() {
	if(rdPdName.isSelected()){
		String pdName = (String) cbPdName.getSelectedItem();
		System.out.println("콤보에서 선택한 상품 : "+pdName);
		//2. db작업 - select
		//3. 결과처리
	}else if(rdPrice.isSelected()){
		//가격으로 검색
		String sPrice1 = tfPrice1.getText();
		String sPrice2 = tfPrice2.getText();
		System.out.println("가격1 = "+sPrice1 + ", 가격2 = "+sPrice2);
	}
}
private void showByNo(int no) {
	PdDTO dto = null;
	try {
		dto = dao.selectByNo(no);
	} catch (SQLException e) {		e.printStackTrace();			}
	if(dto==null){
		System.out.println(no+"에 해당하는 데이터가 없습니다");
		return;
	}
	tfNo.setText(dto.getNo()+"");
	tfPdName.setText(dto.getProductName());
	tfPrice.setText(dto.getPrice()+"");
	tfRegdate.setText(dto.getRegdate()+"");
	taDescription.setText(dto.getDescription());
}
private void clear_tf() {
	tfNo.setText("");	tfPdName.setText("");	tfPrice.setText("");	tfRegdate.setText("");
	taDescription.setText("");		}
```

```
private void addPd() {
	String pdName = tfPdName.getText();
	String sPrice = tfPrice.getText();
	String desc = taDescription.getText();

	//필수항목을 입력하지 않은 경우 에러처리
	if(pdName == null || pdName.trim().isEmpty()){ //trim : 앞뒤 공백제거
		JOptionPane.showMessageDialog(this, "상품명은 반드시 입력해야 합니다");
		return;
	}
	if(sPrice == null || sPrice.trim().isEmpty()){ //trim : 앞뒤 공백제거
		JOptionPane.showMessageDialog(this, "가격은 반드시 입력해야 합니다");
		return;
	}
	PdDTO dto = new PdDTO();
	dto.setProductName(pdName);
	dto.setPrice(Integer.parseInt(sPrice));
	dto.setDescription(desc);

	int cnt=0;
	try {
		cnt = dao.insertPd(dto);
	} catch (SQLException e) { e.printStackTrace();	}
	String msg = "";
	if(cnt>0){
		msg = "상품 등록 성공";
	}else{
		msg = "상품 등록 실패";
	}
	JOptionPane.showMessageDialog(this, msg);
}
```

```
private void editPd() {
	//상품정보 수정하기
	String sNo = tfNo.getText();
	String pdName = tfPdName.getText();
	String sPrice = tfPrice.getText();
	String desc = taDescription.getText();
	if(pdName == null || pdName.trim().isEmpty()){ //trim : 앞뒤 공백제거
		JOptionPane.showMessageDialog(this, "상품명은 반드시 입력해야 합니다");
		return;
	}
	if(sPrice == null || sPrice.trim().isEmpty()){ //trim : 앞뒤 공백제거
		JOptionPane.showMessageDialog(this, "가격은 반드시 입력해야 합니다");
		return;
	}
	PdDTO dto = new PdDTO();
	dto.setNo(Integer.parseInt(sNo));
	dto.setProductName(pdName);
	dto.setPrice(Integer.parseInt(sPrice));
	dto.setDescription(desc);
	int cnt = 0;
	try {
		cnt = dao.editProduct(dto);
	} catch (SQLException e) {		e.printStackTrace();		}
	String msg = "";
	if(cnt>0){
		msg = "상품 수정 성공";
	}else{
		msg = "상품 수정 실패";
	}
	JOptionPane.showMessageDialog(this, msg);
}
```

```
    private void delPd() {
        //상품정보 삭제하기
        //1. 사용자가 입력한 값 읽어오기
        int no = Integer.parseInt(tfNo.getText());
        int cnt = 0;
        //2. db작업 - delete
        try {
            cnt = dao.deleteProduct(no);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        //3. 결과 처리
        String msg = "";
        if(cnt>0){
            msg = "상품 삭제 성공";
        }else{
            msg = "상품 삭제 실패";
        }
        JOptionPane.showMessageDialog(this, msg);
    }
}
```



Swing Component

JTable

```
table.getTableHeader().setPreferredSize(
    new Dimension(scrollpane.getWidth(), 40));
```

- 각 컬럼 사이즈 조절

```
//table.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
table.getColumnModel().getColumn(0).setPreferredWidth(40);
table.getColumnModel().getColumn(1).setPreferredWidth(120);
table.getColumnModel().getColumn(2).setPreferredWidth(67);
```

- 가격 오른쪽 정렬

```
DefaultTableCellRenderer dtcr = new DefaultTableCellRenderer();
        //DefaultTableCellRenderer  생성
dtcr.setHorizontalAlignment(SwingConstants.RIGHT); // Renderer의 가로정렬 지정
table.getColumnModel().getColumn(2).setCellRenderer(dtcr);
```

- JScrollPane 크기 지정

```
JScrollPane scrollPane = new JScrollPane();
tab1.add(scrollPane, BorderLayout.WEST);
scrollPane.setPreferredSize(new Dimension(250, 360));
scrollPane.setViewportView(table);
```

Format

- 가격 – 천단위 구분 기호

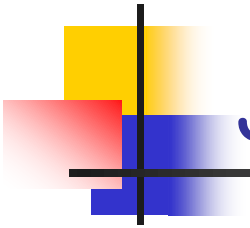
```
for(int i=0;i<dtoArr.length;i++){
        ProductDTO dto = dtoArr[i];
        data[i][0]=new Integer(dto.getNo()).toString();
        data[i][1]=dto.getPName();

        String price = NumberFormat.getInstance().format(dto.getPrice());
        data[i][2]=price;
}//for
```

- 또는 포맷형태 지정하여 사용하기

```
DecimalFormat df = new DecimalFormat("#,###.##");
String y = df.format(123456.124);
System.out.println(y); // 결과 123,456.12
```

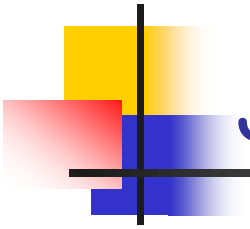
JToolBar, JTabbedPane

```
//1. North - 툴바
JToolBar toolBar = new JToolBar();
contentPane.add(toolBar, BorderLayout.NORTH);

btAdd = new JButton("등록");
toolBar.add(btAdd);
btEdit = new JButton("수정");
toolBar.add(btEdit);

//2. Center-JTabbedPane
JPanel pC = new JPanel();
contentPane.add(pC, BorderLayout.CENTER);
pC.setLayout(new BorderLayout(0, 0));

JTabbedPane tabbedPane = new JTabbedPane(JTabbedPane.TOP);
pC.add(tabbedPane);

//[1] tab1 상품등록
JPanel tab1 = new JPanel();
tabbedPane.addTab("상품등록", null, tab1, null);
tab1.setLayout(new BorderLayout(10, 10));
```



JTextArea

```
taDesc = new JTextArea();
taDesc.setRows(10);
taDesc.setColumns(27);
taDesc.setLineWrap(true); //자동 줄바꿈

JScrollPane scrollPane_1 = new JScrollPane(taDesc);
panel_1.add(scrollPane_1, BorderLayout.CENTER);
```



JRadioButton

```
rdPName = new JRadioButton("상품명으로 검색", true);
panel.add(rdPName);

cbPname = new JComboBox();
cbPname.setPreferredSize(new Dimension(150, 21));
panel.add(cbPname);

JPanel panel_2 = new JPanel();
pQuery.add(panel_2);
FlowLayout flowLayout = (FlowLayout) panel_2.getLayout();
flowLayout.setHgap(7);
flowLayout.setVgap(12);
flowLayout.setAlignment(FlowLayout.LEFT);

rdPrice = new JRadioButton("가    격 으로 검색");
panel_2.add(rdPrice);

ButtonGroup bg = new ButtonGroup();
bg.add(rdPName);
bg.add(rdPrice);
```

JRadioButton

```
tfPrice1 = new JTextField();
tfPrice1.addKeyListener(new KeyAdapter() {
        @Override
        public void keyReleased(KeyEvent e) {
                if(tfPrice1.getText().trim().length()>0)
                        rdPrice.setSelected(true); //라디오버튼이 선택되도록 지
정
        }
});
```

tabbedPane 선택시 index값

```
tabbedPane.addMouseListener(new MouseAdapter() {
	public void mouseClicked(MouseEvent e) {
		int index = ((JTabbedPane)e.getSource()).getSelectedIndex();
		if(index == 1){
			//상품검색
			showProductName();
		}
	}
});

cbPname.addItemListener(new ItemListener() {
	public void itemStateChanged(ItemEvent e) {
		//상품명으로 검색 - 콤보상자 선택시
		if(e.getStateChange() == ItemEvent.SELECTED){
			rdPName.setSelected(true);
			findData(PRODUCTNAME);
		}
	}
});

pack();
```

다른 폼 띄우기

```
class EventHandler extends MouseAdapter implements ActionListener{
        public void actionPerformed(ActionEvent e) {
                if(e.getSource()==btExit)
                {
                        //종료
                        exit();
                }else if(e.getSource()==btMemberEdit){
                        //회원정보 수정
                        MemberGUI m = new MemberGUI(MemberGUI.EDIT);
                        m.tfId.setText(tfId.getText());
                        m.viewMember();
                        m.setVisible(true);
                }else if(e.getSource()==btSearch){
                        //검색 버튼 클릭시
                        if(rdPName.isSelected())  //라디오버튼 선택여부 확인
                                findData(PRODUCTNAME);
                        else if(rdPrice.isSelected())
                                findData(PRICE);
                        }
        }//
```

콤보박스에 데이터 추가하는 3가지 방법

```
//[1] addItem() 이용하는 방법
cbPdName.addItem("사과");
cbPdName.addItem("배");
cbPdName.addItem("귤");

//[2] 배열 이용
String[] sArr = {"하나","둘","셋","넷"};
DefaultComboBoxModel<String> cModel = new DefaultComboBoxModel<String>(sArr);
cbPdName.setModel(cModel);

//[3] Vector 이용
Vector<String> vec = new Vector<String>();
vec.add("java");
vec.add("jsp");
vec.add("oracle");

DefaultComboBoxModel<String> cModel2 = new DefaultComboBoxModel<String>(vec);
cbPdName.setModel(cModel2);
```

상품명 검색하여 콤보상자에 보여주기

```
//상품명 검색하여 콤보상자에 보여주기
    public void showProductName(){
        Vector<String> vector = dao.selectAllPName();
        DefaultComboBoxModel cbModel = new DefaultComboBoxModel(vector);
        cbPname.setModel(cbModel);
    }

    //검색
    public void findData(int flag){
        Vector<ProductDTO> vector = new Vector<ProductDTO>();
        if(flag==PRODUCTNAME){
                vector = dao.selectByPName(cbPname.getSelectedItem().toString());
        }else if(flag==PRICE){
                String price1 = tfPrice1.getText();
                String price2 = tfPrice2.getText();
                if(price1 ==null || price1.isEmpty() ||
                        price2 ==null || price2.isEmpty()){
                        JOptionPane.showMessageDialog(this, "가격을 입력하세요!");
                        return;
                }

                int iPrice1 = Integer.parseInt(price1);
                int iPrice2 = Integer.parseInt(price2);
                vector = dao.selectByPrice(iPrice1, iPrice2);
        }
```

검색

```
        if(vector == null){
                JOptionPane.showMessageDialog(this, "데이터가 없습니다");
                return;
        }

        String[] colName={"번호", "상품명", "가격", "설명", "등록일"};
        String[][] data = new String[vector.size()][5];

        for(int i=0;i<vector.size();i++){
                ProductDTO dto = vector.get(i);
                data[i][0]=new Integer(dto.getNo()).toString();
                data[i][1]=dto.getPName();

                String price = NumberFormat.getInstance().format(dto.getPrice());
                data[i][2]=price;
                data[i][3]=dto.getDescription();
                data[i][4]=dto.getRegDate();
        }//for

        model2.setDataVector(data, colName);
        table2.setModel(model2);

        //각 컬럼 사이즈 조절
        table2.getColumnModel().getColumn(0).setPreferredWidth(40);
        table2.getColumnModel().getColumn(1).setPreferredWidth(80);
        table2.getColumnModel().getColumn(2).setPreferredWidth(60);
        table2.getColumnModel().getColumn(3).setPreferredWidth(150);

        //가격 오른쪽 정렬
        DefaultTableCellRenderer dtcr = new DefaultTableCellRenderer();
        dtcr.setHorizontalAlignment(SwingConstants.RIGHT);
        table2.getColumnModel().getColumn(2).setCellRenderer(dtcr);
}//findData()
```

성별 처리-radio버튼 선택시

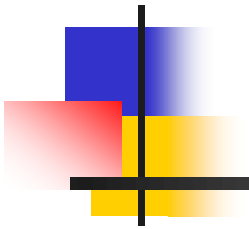
```
rdM = new JRadioButton("남");
rdM.setActionCommand("Male");
rdM.addItemListener(new ItemListener() {
        public void itemStateChanged(ItemEvent e) {
                JRadioButton rb1 = (JRadioButton)e.getSource();
                String gender="";
                //선택된 값 읽어오는 방법1
                if(rb1.isSelected()){
                        gender = rb1.getText();
                }
                System.out.println("성별 : "+ gender);

                //방법2
//              if(e.getStateChange() == ItemEvent.SELECTED){
//                      gender = rb1.getText();
//              }
//              System.out.println("성별 : "+ gender);
        }
});
```

이메일 처리

```
String[] email = {"gmail.com", "nate.com", "daum.net", "hotmail.com","직접입력"};
cbEmail = new JComboBox(email);
cbEmail.addItemListener(new ItemListener() {
            public void itemStateChanged(ItemEvent e) {
                        if(e.getStateChange()==ItemEvent.SELECTED){

                                    if(e.getItem().equals("직접입력")){
                                                System.out.println(e.getItem());
                                                tfEtc.setEditable(true);
                                    }else{
                                                tfEtc.setEditable(false);
                                                tfEtc.setText("");
                                    }
                        }
            }
});
pEmail.add(cbEmail, BorderLayout.CENTER);

tfEtc = new JTextField();
tfEtc.setEditable(false);
pEmail.add(tfEtc, BorderLayout.EAST);
tfEtc.setColumns(10);
```

휴대폰 처리-JComboBox

```
//          JComboBox cbHp1 = new JComboBox();
//          cbHp1.addItem("010");
//          cbHp1.addItem("011");

            Vector<String> v = new Vector<String>();
            v.addElement("010");v.addElement("011");v.addElement("016");
            v.addElement("017");v.addElement("018");v.addElement("019");
            DefaultComboBoxModel cbModel = new DefaultComboBoxModel(v);
            cbHp1 = new JComboBox();
            cbHp1.setModel(cbModel);

            cbHp1.addItemListener(new ItemListener() {
                        public void itemStateChanged(ItemEvent e) {
                                    if(e.getStateChange() == ItemEvent.SELECTED){
                                                //System.out.println("itemStateChanged=> hp : " +
    e.getItem());
                                                //System.out.println("selectedItem hp : " +
    cbHp1.getSelectedItem());
                                                //System.out.println("selectedIndex hp : " +
    cbHp1.getSelectedIndex()+"₩n");
                                    }
                        }
            });
```



상품 관리 – netbeans 이용

상품 테이블

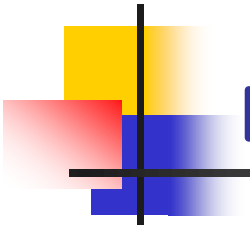
```
--상품테이블

create table product
(
    no              number  primary key,        --일련번호
    productName     varchar2(50) Not Null,      --상품이름
    price           number,                     --소비자가
    --company       varchar2(50),               --제조회사
    description     varchar2(255),              --설명
    regDate         date        default  sysdate --등록일
);

create sequence product_seq
start with 1
increment by 1
nocache;
```

회원 테이블

```
--회원테이블

create table member
(
    no              number  primary key,            --일련번호
    userid          varchar(20)         not null,   --아이디
    name            varchar(20)         not null,   --이름
    pwd             varchar(15)         not null,   --비밀번호
    email           varchar(50)         null,       --이메일
    hp              varchar(20)         null,       --휴대폰
    zipcode         char(7)             null,       --우편번호
    address1        varchar(50)         null,       --주소1
    address2        varchar(50)         null,       --주소2(상세주소)
    regDate         date                default  sysdate  null  --등록일
);

create sequence member_seq
start with 1
increment by 1
nocache;
```

우편번호 테이블

```
--우편번호 테이블
create table zipcode
(
 zipcode varchar2(5),
 sido varchar2(100),
 gugun varchar2(100),
 dong varchar2(200),
 startbunji varchar2(100),
 endbunji varchar2(100),
 seq number primary key
);

create sequence zipcode_seq
start with 1
increment by 1
nocache;
```



netbeans

- 다운로드
  - https://netbeans.apache.org/download/index.html

  - netbeans-8.2-javase-windows.exe
  - jdk-8u121-nb-8_2-windows-x64.exe

netbeans 단축키

- 1. psvm => tab 키 => main() 생성
  - sout => tab 키 => sysout
- 2. 주석처리 한번에 하기
  - 주석하고 싶은 줄에 커서를 가져다 놓거나 여러 줄을 드래그하여 선택 한 후 Ctrl+Shift+C
  - 주석을 해제하고 싶으면 다시 Ctrl+Shift+C 키
- 3. 변수명 한번에 바꾸기
  - 변경하고 싶은 변수명 위에 커서를 올리고 Ctrl+R 키를 누르면 해당 변수명이 지워지면서 수정 가능한 박스 상태로 변경하게 됨.
  - 그 상태에서 변수명을 수정하면 해당 변수를 사용하고 있던 모든 곳의 변수명이 같이 변경됨
- 4. 라인 자동 정렬 기능
  - 외부에서 복사한 소스나 자신이 만든 코드일지라도 들여쓰기가 제대로 이루어지지 않은 경우 라인 및 들여쓰기를 자동으로 정렬 하려면 Alt+Shift+F 키를 누르면 한번에 자동정렬이 이루어짐
- 5. 현재 작업중인 문서의 위치 찾기
  - 현재 수정중인 코드 및 문서가 프로젝트 내의 어느 패키지에 있는지 찾기 위해 프로젝트 네비게이션을 스크롤해가며 찾아봤던 경험이 있을 것임
  - NetBeans에서는 간단히 Ctrl+Shift+1 을 누르면 바로 현재 작업중인 문서의 패키지가 펼쳐지며 해당 문서에 하이라이트가 되어 찾을 수 있게 됨

- 6. 펼쳐져 있는 모든 문서를 닫기
  - Ctrl+Shift+w 또는 Ctrl+Shift+F4 키를 누르면 현재 열려있던 모든 창들이 닫히며 수정 후 저장되지 않았던 문서가 있다면 저장 후 닫을 것인지에 대해서도 물어봄

- 7. 자동으로 import 생성
  - 소스만 복사해 와서 붙인 경우 import 문이 생략된 경우가 많아 소스에 빨간 줄이 많이 쳐지는데 이럴 때 Ctrl+Shift+i 를 눌러 자동으로 import문을 생성

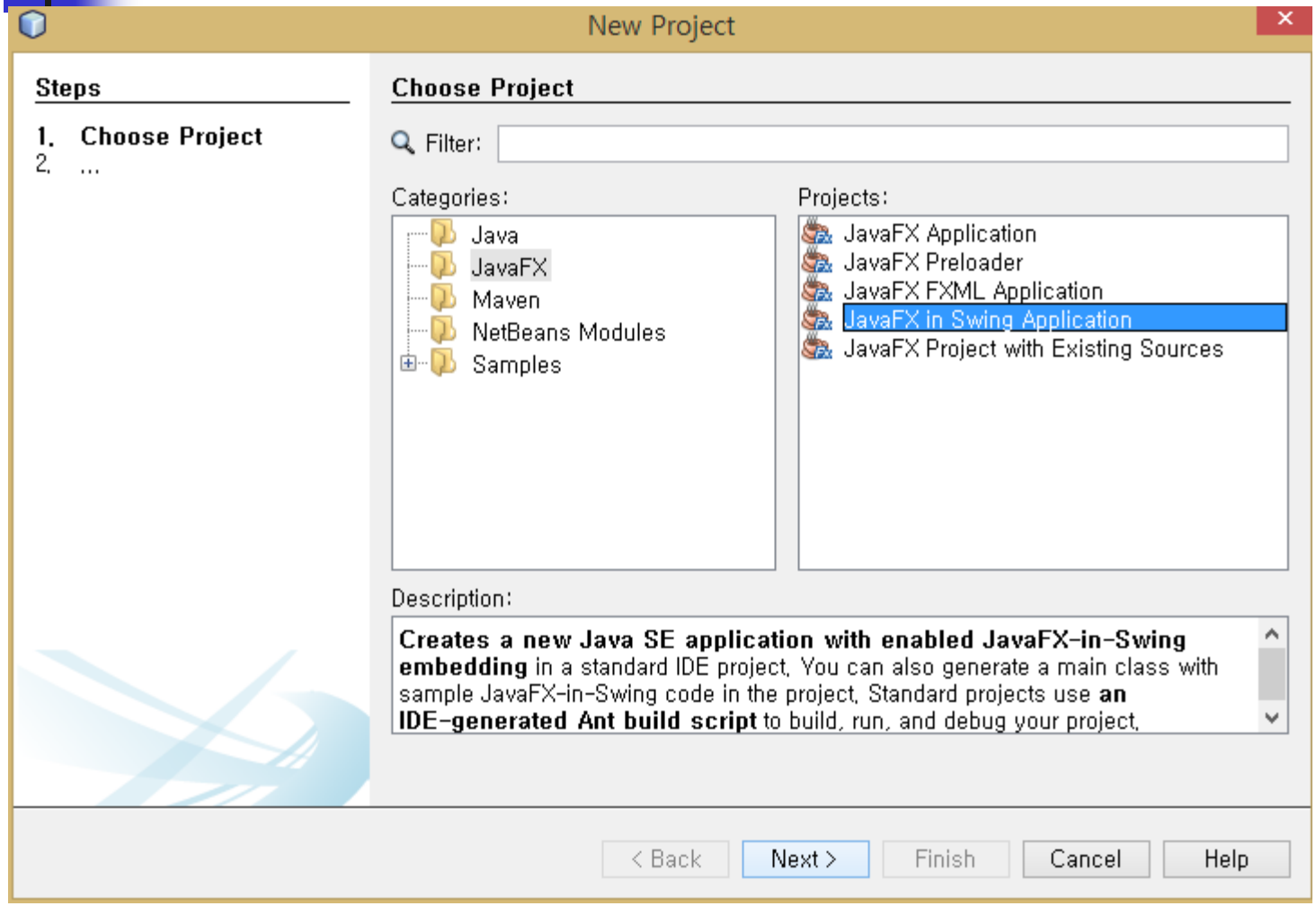
- 8. 자동으로 관련 Method를 찾는 기능 및 해설
  - 해당 메소드의 내용 및 해설을 보고 싶을 때에는 Ctrl+Space 키를 누르면 Java-doc까지 자세한 설명과 관련 메소드들을 NetBeans에서 보여주며 autocomplete기능까지 제공

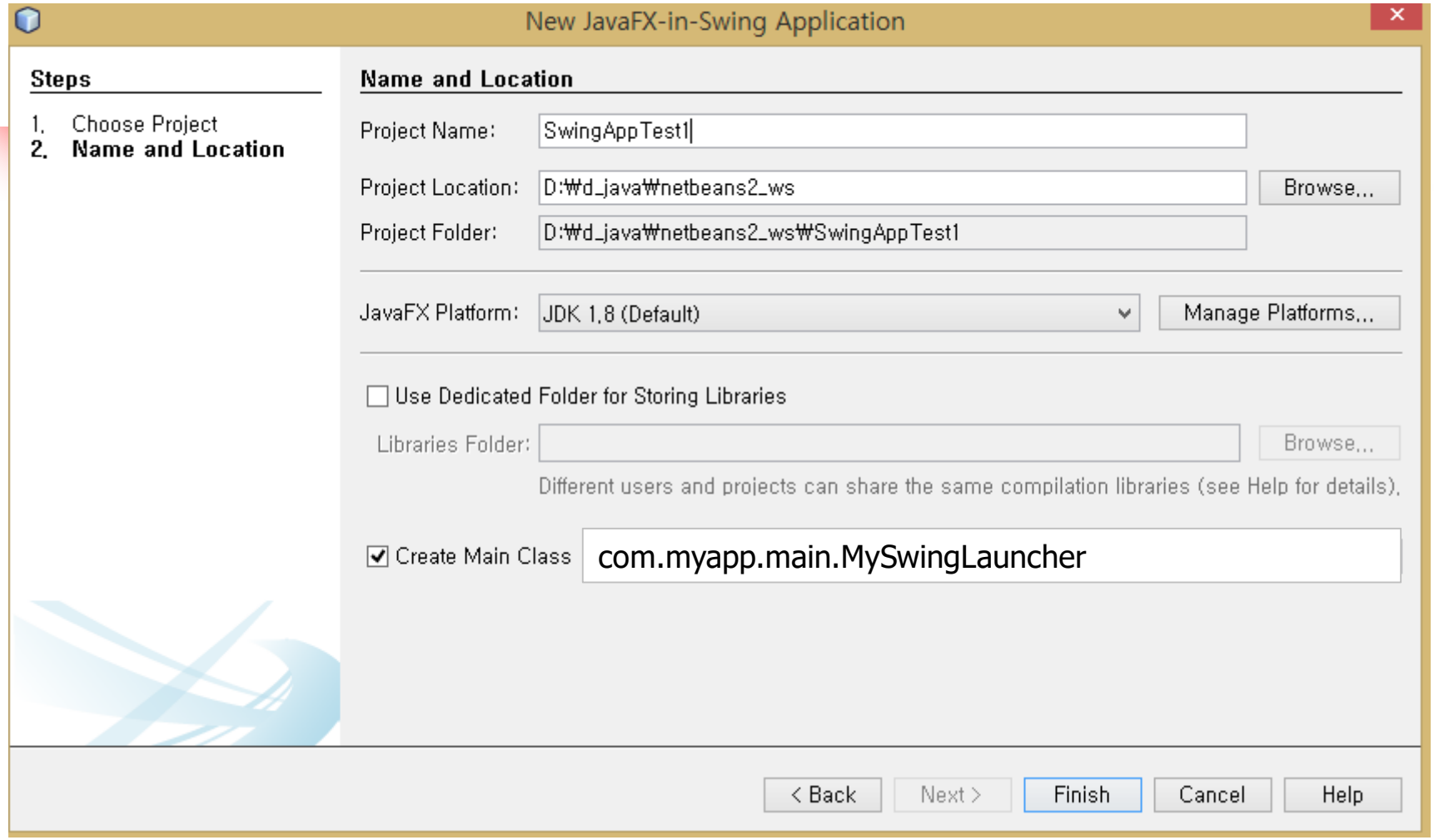
- 9. getter, setter 메소드 자동 생성
  - 클래스에 멤버변수(클래스 변수)들만 선언해주고 Alt+Insert 키를 누르면 Generate 창이 뜨는데, 해당 창에서 getter and setter를 선택하면 변수 별로 get, set 메소드를 자동으로 생성

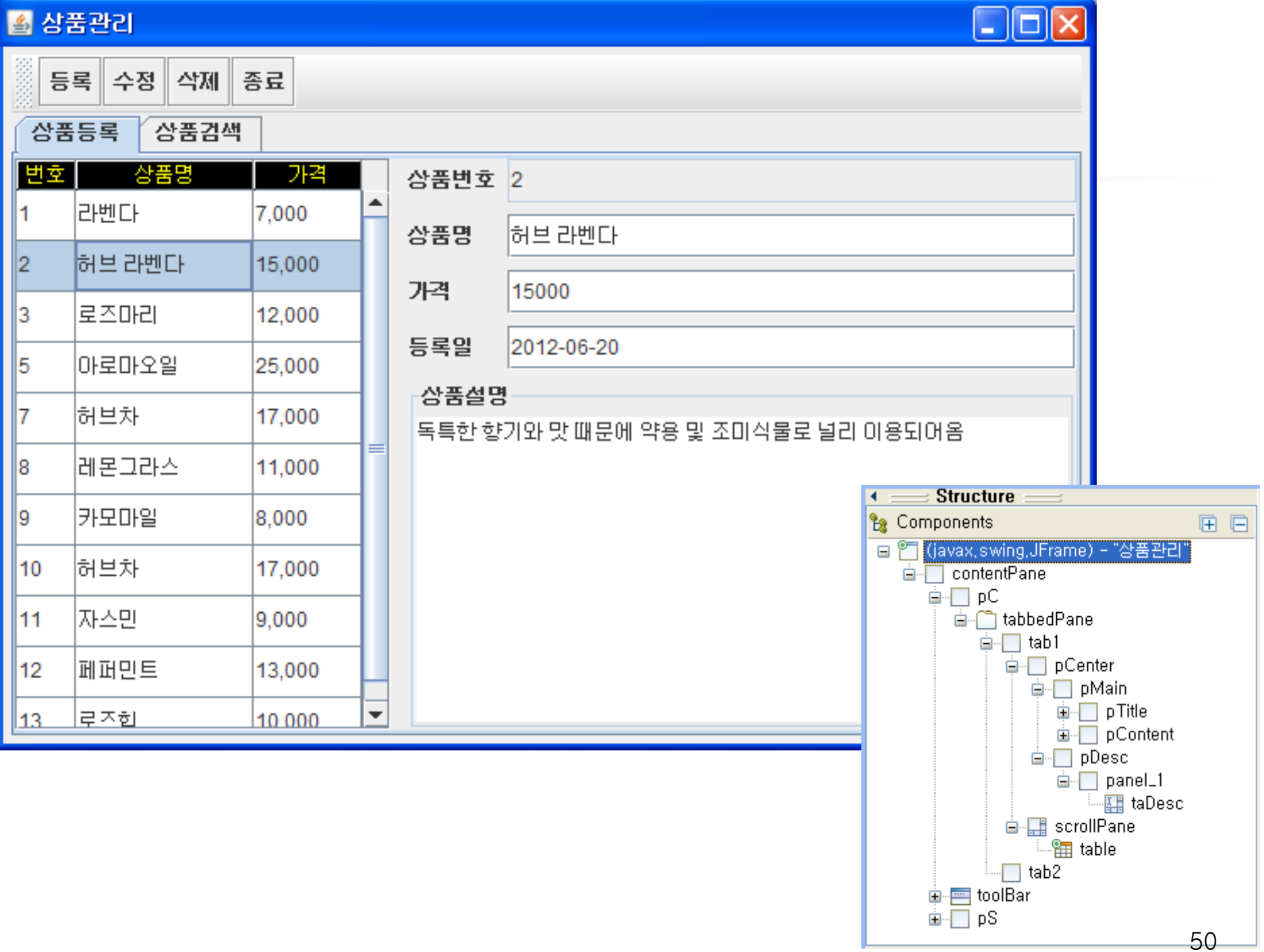
- 10. 파일끼리 비교하여 무엇이 변했는지 한눈에 볼 수 는 비교 자동결과 출력
  - 상단 메뉴에서 Tools - Diff 를 눌러 비교할 파일을 선택하면 현재 열려 있는 문서와 비교하여 무엇이 변했고 추가/삭제 된 부분이 어느 곳인지 한눈에 알 수 있게 됨

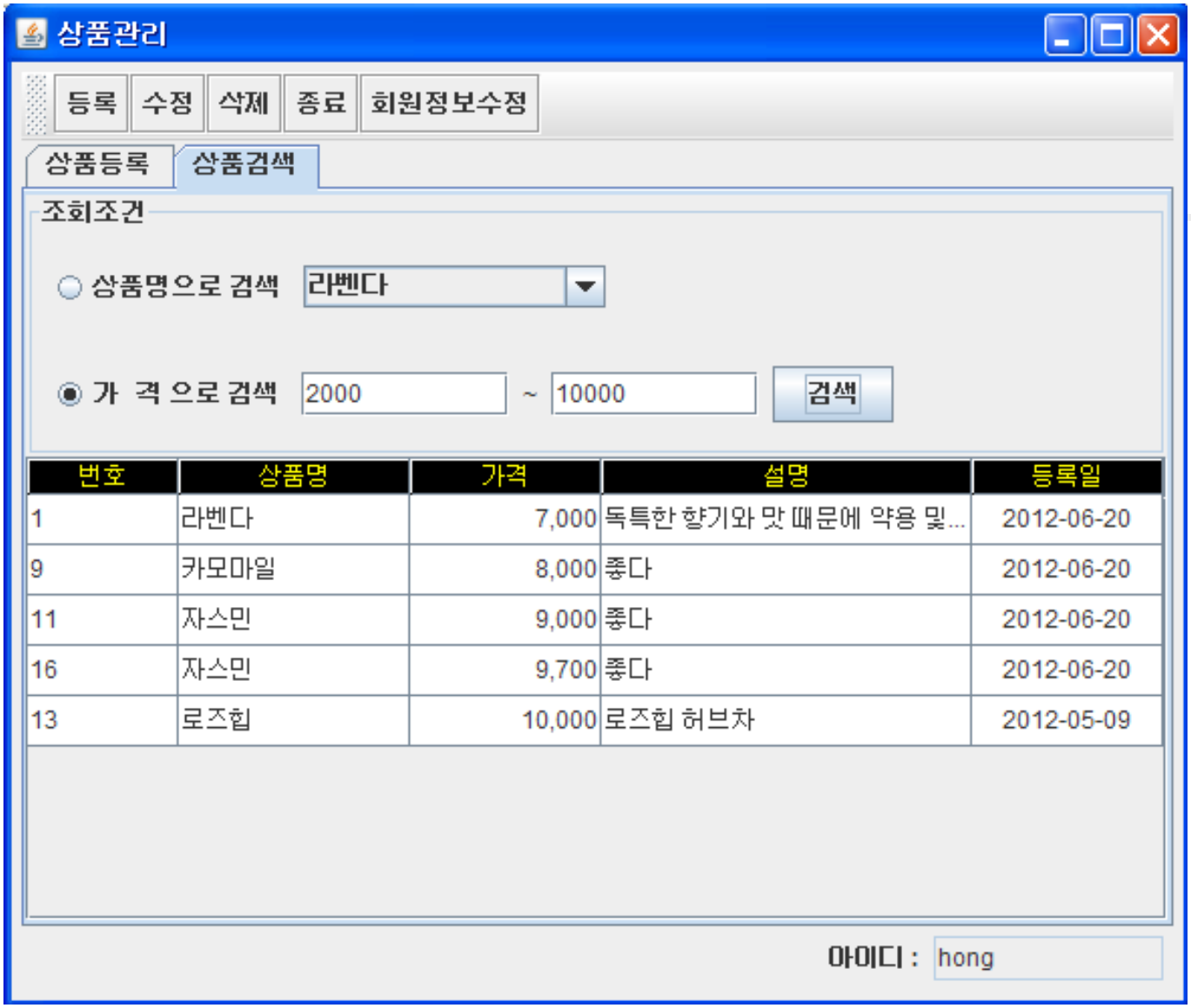
- 11. ctrl + 마우스 올리면 메서드 설명이 나옴
- 12. alt + 위, 아래 화살표 => 해당 메서드나 변수를 찾아줌
- 13. alt + shift + 위, 아래 화살표 => 이동
- 14. ctrl + shift + 위, 아래 화살표 => copy

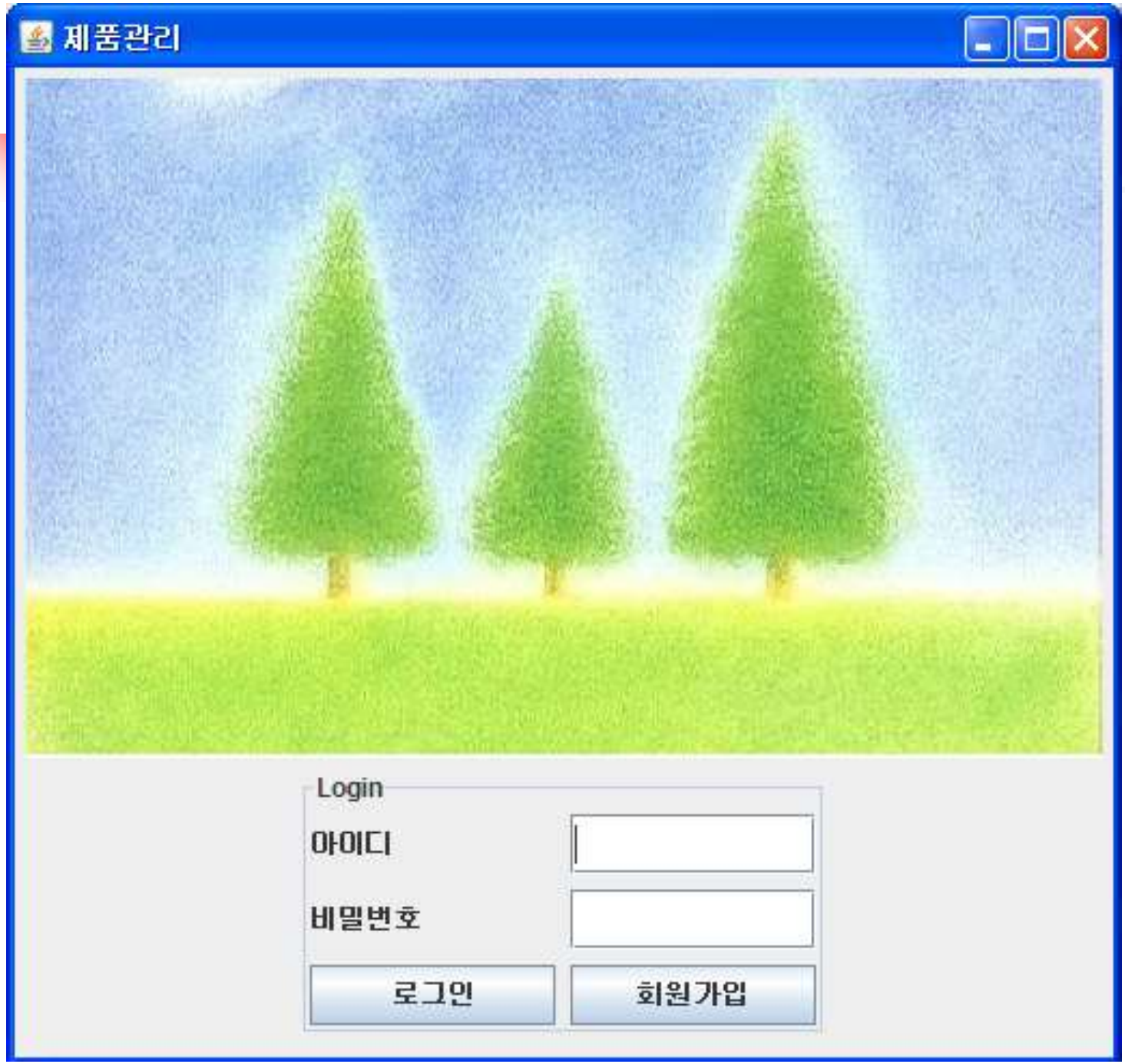
netbeans 이용한 프로젝트 생성



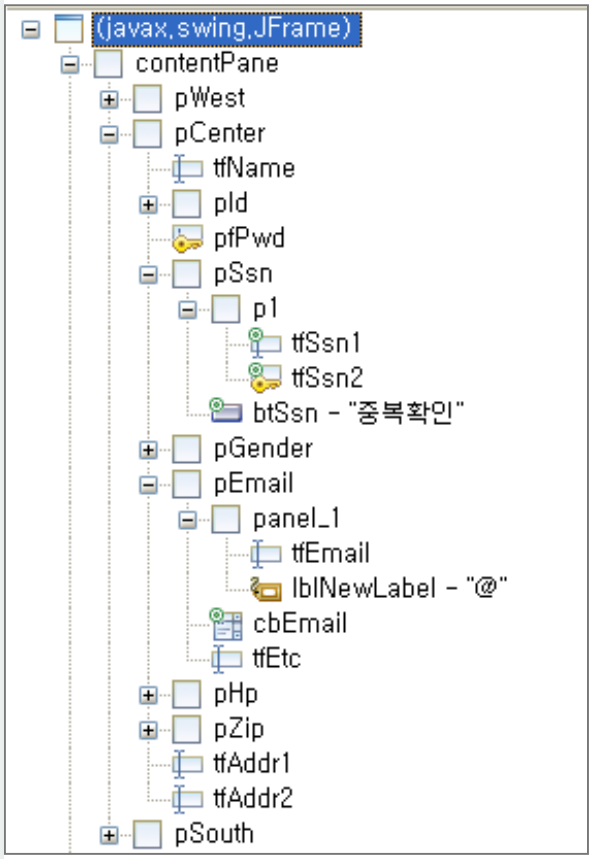

New JavaFX-in-Swing Application
Steps
1. Choose Project
2. Name and Location
Name and Location
Project Name: SwingAppTest1
Project Location: D:₩d_java₩netbeans2_ws
Browse...
Project Folder: D:₩d_java₩netbeans2_ws₩SwingAppTest1
JavaFX Platform: JDK 1.8 (Default)
Manage Platforms...
Use Dedicated Folder for Storing Libraries
Libraries Folder:
Browse...
Different users and projects can share the same compilation libraries (see Help for details).
Create Main Class
com.myapp.main.MySwingLauncher
< Back
Next >
Finish
Cancel
Help

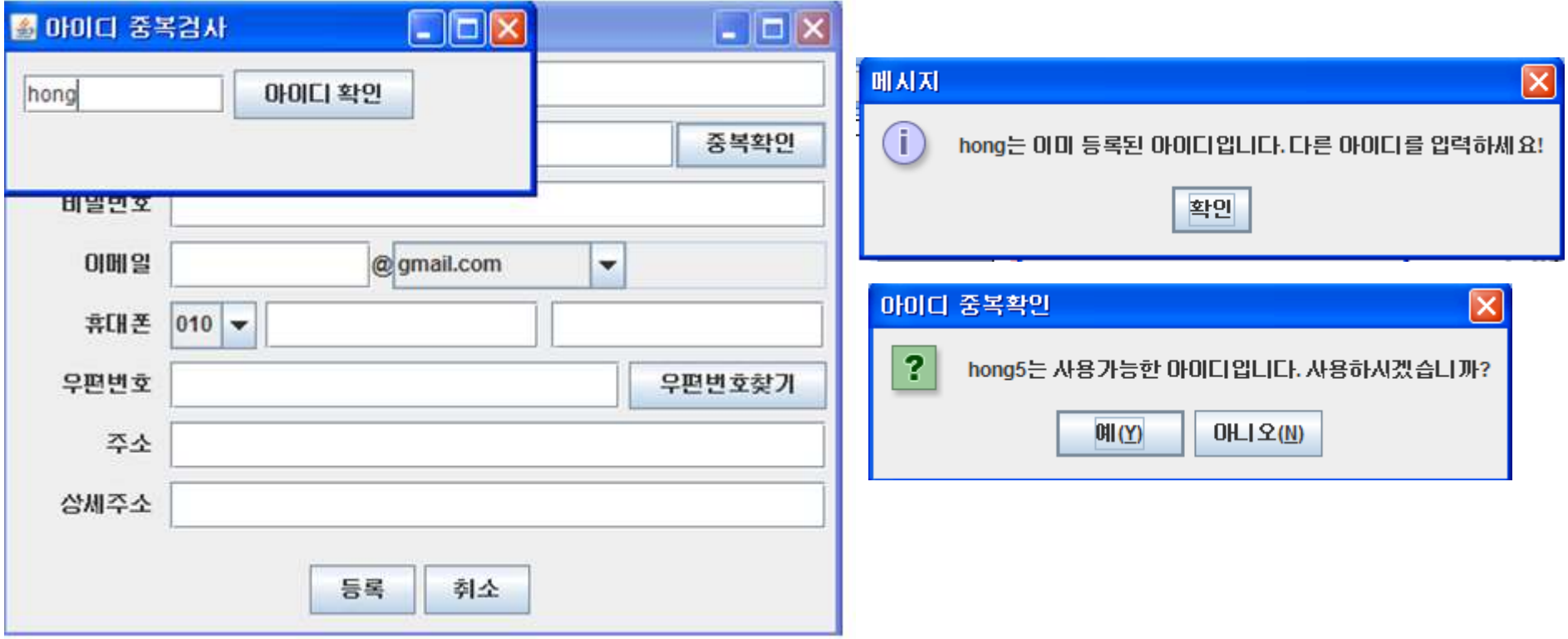

상품관리
등록
수정
삭제
종료
상품등록
상품검색
번호
상품명
가격
1
라벤다
7,000
2
허브 라벤다
15,000
3
로즈마리
12,000
5
아로마오일
25,000
7
허브차
17,000
8
레몬그라스
11,000
9
카모마일
8,000
10
허브차
17,000
11
자스민
9,000
12
페퍼민트
13,000
13
로즈힙
10,000
상품번호
2
상품명
허브 라벤다
가격
15000
등록일
2012-06-20
상품설명
독특한 향기와 맛 때문에 약용 및 조미식물로 널리 이용되어옴
Structure
Components
(javax.swing.JFrame) - "상품관리"
contentPane
pC
tabbedPane
tab1
pCenter
pMain
pTitle
pContent
pDesc
panel_1
taDesc
scrollPane
table
tab2
toolBar
pS


상품관리
등록
수정
삭제
종료
회원정보수정
상품등록
상품검색
조회조건
상품명으로 검색
라벤다
가 격 으로 검색
2000
~
10000
검색
번호
상품명
가격
설명
등록일
1
라벤다
7,000
독특한 향기와 맛 때문에 약용 및...
2012-06-20
9
카모마일
8,000
좋다
2012-06-20
11
자스민
9,000
좋다
2012-06-20
16
자스민
9,700
좋다
2012-06-20
13
로즈힙
10,000
로즈힙 허브차
2012-05-09
아이디 :
hong


제품관리
Login
아이디
비밀번호
로그인
회원가입

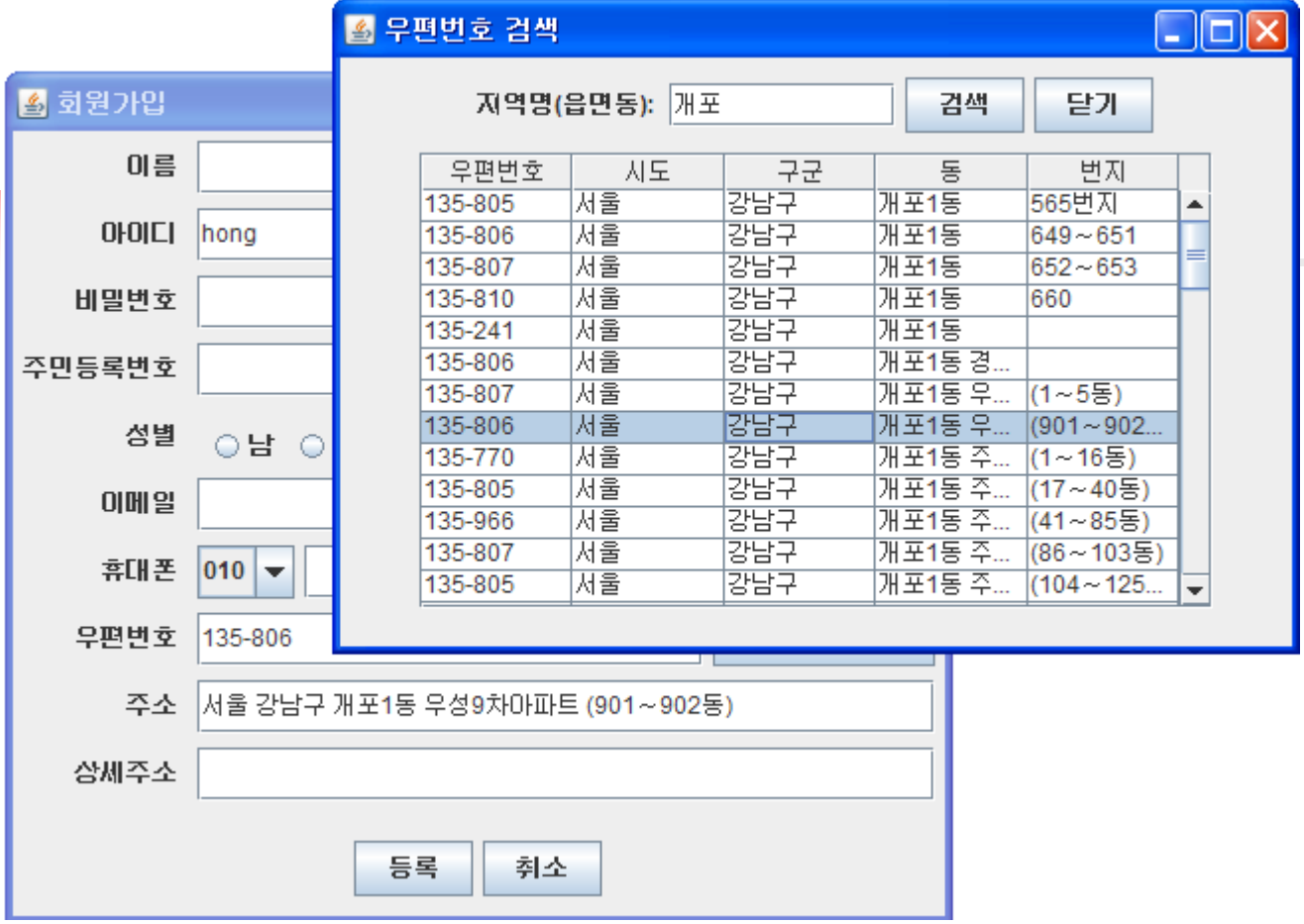

회원가입
이름
아이디
중복확인
비밀번호
이메일
@
naver.com
전화번호
010
우편번호
우편번호 찾기
주소
상세주소
회원가입
취소
비밀번호
로그인
회원가입

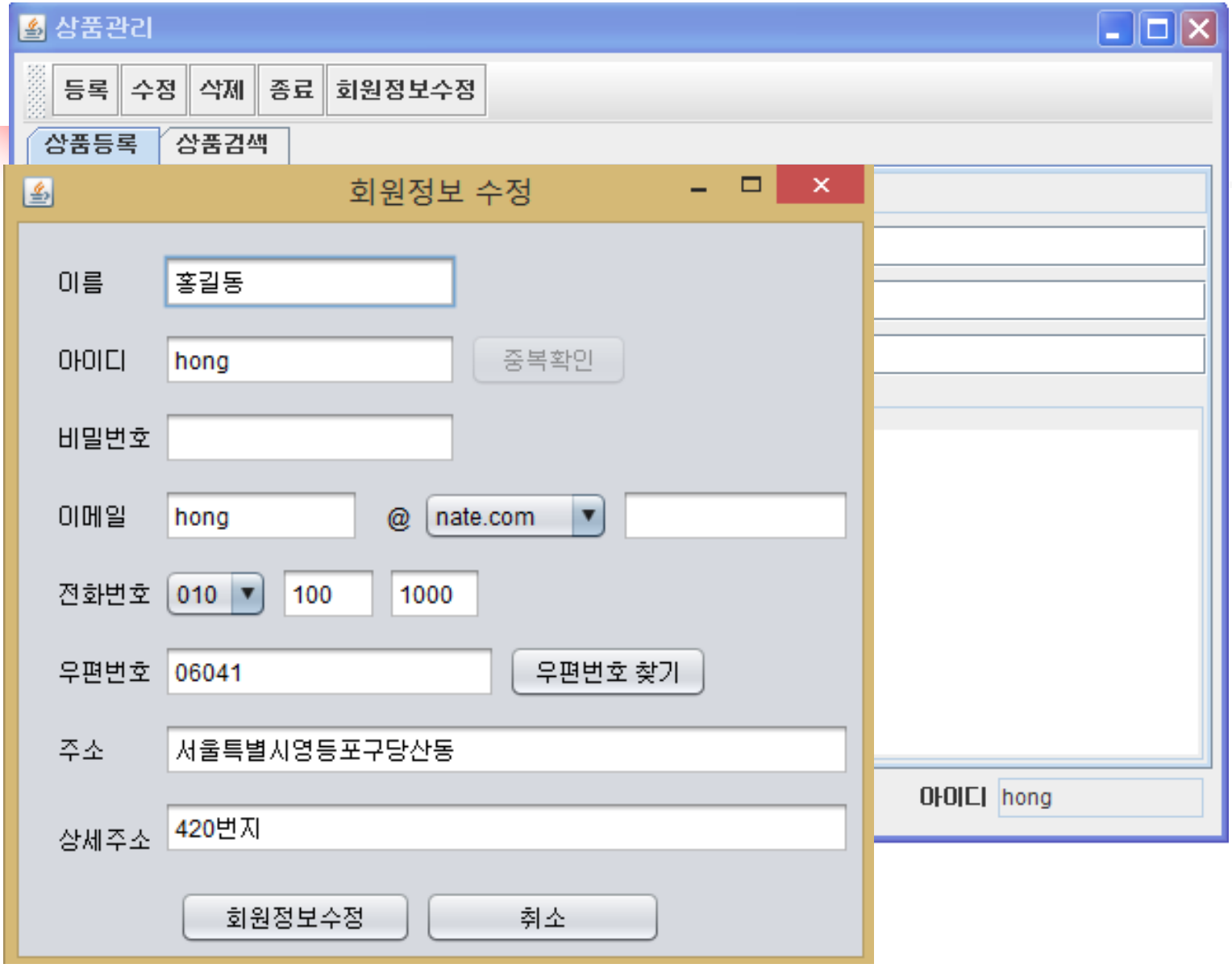

(javax.swing.JFrame)
contentPane
pWest
pCenter
tfName
pId
pfPwd
pSsn
p1
tfSsn1
tfSsn2
btSsn - "중복확인"
pGender
pEmail
panel_1
tfEmail
lblNewLabel - "@"
cbEmail
tfEtc
pHp
pZip
tfAddr1
tfAddr2
pSouth


아이디 중복검사
hong
아이디 확인
중복확인
이메일
@gmail.com
휴대폰
010
우편번호
우편번호찾기
주소
상세주소
등록
취소

메시지
hong는 이미 등록된 아이디입니다. 다른 아이디를 입력하세요!
확인

아이디 중복확인
hong5는 사용가능한 아이디입니다. 사용하시겠습니까?
예(Y)
아니오(N)


우편번호 검색
지역명(읍면동): 개포
검색
닫기
우편번호 시도 구군 동 번지
135-805 서울 강남구 개포1동 565번지
135-806 서울 강남구 개포1동 649~651
135-807 서울 강남구 개포1동 652~653
135-810 서울 강남구 개포1동 660
135-241 서울 강남구 개포1동
135-806 서울 강남구 개포1동 경...
135-807 서울 강남구 개포1동 우... (1~5동)
135-806 서울 강남구 개포1동 우... (901~902...
135-770 서울 강남구 개포1동 주... (1~16동)
135-805 서울 강남구 개포1동 주... (17~40동)
135-966 서울 강남구 개포1동 주... (41~85동)
135-807 서울 강남구 개포1동 주... (86~103동)
135-805 서울 강남구 개포1동 주... (104~125...
회원가입
이름
아이디 hong
비밀번호
주민등록번호
성별 남
이메일
휴대폰 010
우편번호 135-806
주소 서울 강남구 개포1동 우성9차아파트 (901~902동)
상세주소
등록
취소


상품관리
등록
수정
삭제
종료
회원정보수정
상품등록
상품검색
회원정보 수정
이름
홍길동
아이디
hong
중복확인
비밀번호
이메일
hong
@
nate.com
전화번호
010
100
1000
우편번호
06041
우편번호 찾기
주소
서울특별시영등포구당산동
상세주소
420번지
회원정보수정
취소
아이디
hong

SwingApp1

```
public class SwingApp1 extends JApplet {
    private static final int JFXPANEL_WIDTH_INT = 300;
    private static final int JFXPANEL_HEIGHT_INT = 250;
    private static JFXPanel fxContainer;

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                try {
                    UIManager.setLookAndFeel("com.sun.java.swing.plaf.nimbus.NimbusLookAndFeel");
                } catch (Exception e) {
                }

                LoginGUI f = new LoginGUI();
                f.show();
            }
        });
    }
}
```

MemberDAO

```
public class MemberDAO {
    //로그인 체크 관련 상수
    public static final int LOGIN_OK=1;  //로그인 성공
    public static final int PWD_DISAGREE=2; //비밀번호 불일치
    public static final int USERID_NONE=3; //아이디 없음
    //아이디 중복확인체크 관련 상수
    public static final int USABLE_ID=1;  //사용가능한 아이디
    public static final int NONE_USABLE_ID=2; //사용불가 아이디
    public int loginCheck(String userid, String pwd) throws SQLException{
        //로그인 처리
        Connection conn=null;
        PreparedStatement ps=null;
        ResultSet rs=null;
        int result=0;
        try{
            conn=DBUtil.getConnection();
            String sql="select pwd from member where userid=?";
            ps=conn.prepareStatement(sql);
            ps.setString(1, userid);
            rs=ps.executeQuery();
            if(rs.next()){
                String dbPwd = rs.getString(1);
                if(dbPwd.equals(pwd)){        //비밀번호도 일치하는 경우-로그인 성공
                    result=LOGIN_OK;
                }else{ //비밀번호가 맞지 않는 경우
                    result=PWD_DISAGREE;
                }
            }else{        //해당 아이디가 없는 경우
                result=USERID_NONE;
            }//if
            System.out.println("로그인 체크 결과 result="+result    +", 입력값 userid="+userid +", pwd="+pwd);
```

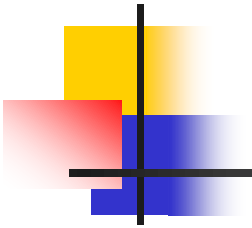
MemberDAO

```
    }finally{
        DBUtil.dbClose(rs, ps, conn);
    }
    return result;
}

public int insertMember(MemberDTO dto) throws SQLException{
    //회원가입
    ......
        String sql="insert into member(no, userid, name, pwd, email, hp,"
                + " zipcode, address1, address2) values(member_seq.nextval, ?,?,?,?,?,?,?,?)";
        ps=conn.prepareStatement(sql);
        ps.setString(1, dto.getUserid());
        ps.setString(2, dto.getName());
        ps.setString(3, dto.getPwd());
        ps.setString(4, dto.getEmail());
        ps.setString(5, dto.getHp());
        ps.setString(6, dto.getZipcode());
        ps.setString(7, dto.getAddress1());
        ps.setString(8, dto.getAddress2());

        //4.exec
        cnt = ps.executeUpdate();
        System.out.println("회원가입 결과 cnt="+cnt+", 입력값 dto="+dto);
    }finally{
        DBUtil.dbClose(ps, conn);
    }
    return cnt;
}
```

MemberDAO

```
public int checkUserid(String userid) throws SQLException{
    //해당 아이디가 사용가능한지 확인
    Connection conn=null;
    PreparedStatement ps=null;
    ResultSet rs=null;
    int result=0;
    try{
       conn=DBUtil.getConnection();

       String sql="select count(*) from member  where userid=?";
       ps=conn.prepareStatement(sql);
       ps.setString(1, userid);

       rs=ps.executeQuery();
       if(rs.next()){
         int count = rs.getInt(1);
         if(count>0){              //해당 아이디가 이미 존재하는 경우-사용 불가
           result=NONE_USABLE_ID;
         }else{              //해당 아이디가 없는 경우-사용 가능
           result=USABLE_ID;
         }
       }//if
       System.out.println("아이디 중복확인 체크 결과 result="  + result +", 입력값 userid="+userid);
    }finally{
       DBUtil.dbClose(rs, ps, conn);
    }
    return result;
  }
```

```
public int updateMember(MemberDTO dto) throws SQLException{
      //회원정보 수정 - member 테이블 update
      Connection conn=null;
      PreparedStatement ps=null;
      int cnt=0;
      try{
         conn=DBUtil.getConnection();
         String sql="update member set email=?,hp=?,zipcode=?, address1=?," +
            " address2=?  where userid=? and pwd=?";
         ps=conn.prepareStatement(sql);

         ps.setString(1, dto.getEmail());
         ps.setString(2, dto.getHp());
         ps.setString(3, dto.getZipcode());
         ps.setString(4, dto.getAddress1());
         ps.setString(5, dto.getAddress2());
         ps.setString(6, dto.getUserid());
         ps.setString(7, dto.getPwd());

         cnt = ps.executeUpdate();
         System.out.println("회원정보 수정 결과 cnt="+cnt  +", 입력값 dto="+dto);
      }finally{
         DBUtil.dbClose(ps, conn);
      }
      return cnt;
   }
}//class
```



MemberService

```
public class MemberService {
    private static String userid;

    public static String getUserid() {
        return userid;
    }

    public static void setUserid(String p_userid) {
        userid = p_userid;
    }

}
```

LoginGUI

```
public class LoginGUI extends javax.swing.JFrame {
    public LoginGUI() {
        initComponents();
    }
    private void loginProc(){
        try {
            //로그인 처리
            String userid = tfId.getText();
            String pwd = tfPwd.getText();

            if(userid==null || userid.isEmpty()){
                JOptionPane.showMessageDialog(this, "아이디를 입력하세요");
                tfId.requestFocus();
                return;
            }else if(pwd==null || pwd.isEmpty()){
                JOptionPane.showMessageDialog(this, "비밀번호를 입력하세요");
                tfPwd.requestFocus();
                return;
            }
            MemberDAO dao = new MemberDAO();
            int result = dao.loginCheck(userid, pwd);
            if(result==MemberDAO.LOGIN_OK){              //로그인 성공
                JOptionPane.showMessageDialog(this,   userid+"로 로그인되었습니다");
                //MemberService.setUserid(userid);
                ProductMain f = new ProductMain(userid);
                f.setVisible(true);
               this.dispose();  //로그인 화면은 보여주지 않는다
            }else if(result==MemberDAO.PWD_DISAGREE){
                JOptionPane.showMessageDialog(this,        "비밀번호가 일치하지 않습니다");
            }else if(result==MemberDAO.USERID_NONE){
                JOptionPane.showMessageDialog(this,    "해당 아이디가 존재하지 않습니다");
```

```
            }else{
                JOptionPane.showMessageDialog(this, "로그인 처리 실패");
            }//if
        } catch (SQLException ex) { }
    }
    private void btLoginActionPerformed(java.awt.event.ActionEvent evt) {
        //로그인 처리
        loginProc();
    }
    private void btRegisterActionPerformed(java.awt.event.ActionEvent evt) {
        //회원가입 폼 보여주기
        MemberFrame memFrame  = new MemberFrame(MemberFrame.MEMBER_REGISER, this);
        memFrame.setVisible(true);
    }
     btLogin.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btLoginActionPerformed(evt);
            }
        });
        btRegister.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btRegisterActionPerformed(evt);
            }
        });
public static void main(String args[]) {
      java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new LoginGUI().setVisible(true);
            }
        });
```

ProductMain

```
public class ProductMain extends javax.swing.JFrame implements ActionListener{
    private String userid;

    public ProductMain() {
        initComponents();
        init();
    }
    public ProductMain(String userid)   {
        this();
        this.userid=userid;

        //userid 셋팅
        //로그인 화면에서 보낸 아이디를 메인 화면에 출력한다
        tfUserid.setText(userid);
        //tfUserid.setText(MemberService.getUserid());
   }
    private void init() {        //이벤트 연결
        btMemEdit.addActionListener(this);
    }
    public void actionPerformed(ActionEvent ae) {
        if(ae.getSource()==btMemEdit){            //회원정보 수정 화면 보여주기
            MemberFrame f = new MemberFrame(MemberFrame.MEMBER_EDIT);
            f.setVisible(true);
        }else if(ae.getSource()==btExit){
            System.exit(0);
        }
    }
}
```

MemberFrame

```
public class MemberFrame extends javax.swing.JFrame {
    //회원가입, 회원정보 수정 구분 플래그
    private int memberFlag;
    //관련 상수
    public static final int MEMBER_REGISER=1; //회원가입
    public static final int MEMBER_EDIT=2; //회원정보수정

    private LoginGUI loginGui;
    private MemberDAO dao;
    private boolean isDuplicateChecked;
    //=> 서브에서 중복확인을 정상적으로 처리했으면 true로 셋팅

    private String[] emailList = {"naver.com","nate.com","daum.net",   "gmail.com", "직접입력"};

    public boolean getIsDuplicateChecked() {
        return isDuplicateChecked;
    }
    public void setIsDuplicateChecked(boolean isDuplicateChecked) {
        this.isDuplicateChecked = isDuplicateChecked;
    }
    public MemberFrame() {
        initComponents();
        init();
    }
    public MemberFrame(int flag){
        this();
        this.memberFlag=flag;
        memberProc();
    }
```

MemberFrame

```
public MemberFrame(int flag, LoginGUI loginGui){
    this();

    this.memberFlag=flag;
    this.loginGui=loginGui;

    memberProc();
}

private void init(){
    dao=new MemberDAO();

    this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

    //email 콤보박스에 항목 추가
    DefaultComboBoxModel<String> comboModel
            = new DefaultComboBoxModel<String>(emailList);
    cbEmail2.setModel(comboModel);

    //직접입력시 입력하는 텍스트박스는 처음에 편집불가
   tfEmail3.setEditable(false);
}
```

```
private void memberAdd(){
        //회원가입 처리
        //1. 사용자 입력값 읽어오기
        String address1 = tfAddress1.getText();
        String address2 = tfAddress2.getText();
        String email1 = tfEmail1.getText();
        String email3 = tfEmail3.getText();
        String hp2 = tfHp2.getText();
        String hp3 = tfHp3.getText();
        String name = tfName.getText();
        String pwd = tfPwd.getText();
        String userid = tfUserid.getText();
        String zipcode = tfZipcode.getText();

        String email2 = (String) cbEmail2.getSelectedItem();
        String hp1 = (String) cbHp1.getSelectedItem();

        //유효성 검사
        if(name==null || name.isEmpty()){
            JOptionPane.showMessageDialog(this, "이름을 입력하세요");
            tfName.requestFocus();
            return;
        }else if(userid==null || userid.isEmpty()){
            JOptionPane.showMessageDialog(this, "아이디를 입력하세요");
            tfUserid.requestFocus();
            return;
        }else if(pwd==null || pwd.isEmpty()){
            JOptionPane.showMessageDialog(this, "비밀번호를 입력하세요");
            tfPwd.requestFocus();
            return;
```

```
}else if(!isDuplicateChecked){
    //아이디 중복확인을 하지 않은 경우
    JOptionPane.showMessageDialog(this,              "아이디 중복확인을 하세요");
    btDup.requestFocus();
    return;
}
//2. db작업 - insert
try {
    MemberDTO dto = new MemberDTO();
    dto.setAddress1(address1);
    dto.setAddress2(address2);
    dto.setName(name);
    dto.setPwd(pwd);
    dto.setUserid(userid);
    dto.setZipcode(zipcode);
    //휴대폰 처리
    String hp="";
    if(hp2!=null && !hp2.isEmpty()        && hp3!=null && !hp3.isEmpty()){
        hp=hp1+"-"+hp2+"-"+hp3;
    }
    //이메일 처리
    String email="";
    if(email1 !=null && !email1.isEmpty()){
        if(email2.equals("직접입력")){
            email=email1+"@"+email3;
        }else{
            email=email1+"@"+email2;
        }
    }
    dto.setEmail(email);
    dto.setHp(hp);
```

```
      int cnt = dao.insertMember(dto);

      String msg="";
        if(cnt>0){
            msg="회원가입되었습니다";
            this.dispose();

            //로그인 화면의 id에 회원가입한 아이디를 보여줌
            loginGui.tfId.setText(tfUserid.getText());
        }else{
            msg="회원가입 실패";
        }
        JOptionPane.showMessageDialog(this, msg);
    } catch (SQLException ex) { }
}

private void btAddActionPerformed(java.awt.event.ActionEvent evt) {
    if(memberFlag==MEMBER_REGISER){             //회원가입
        memberAdd();
    }else if(memberFlag==MEMBER_EDIT){            //회원정보 수정
        memberEdit();
    }//if
}
```

```
private void cbEmail2ItemStateChanged(java.awt.event.ItemEvent evt) {
    if(evt.getStateChange()==ItemEvent.SELECTED){
        System.out.println("1.선택된 항목:"          + evt.getItem());
        System.out.println("2.선택된 항목:" + cbEmail2.getSelectedItem()  + ", 선택된 index:"
            + cbEmail2.getSelectedIndex()+"₩n");
        if(evt.getItem().equals("직접입력")){
            tfEmail3.setEditable(true);
            tfEmail3.requestFocus();
        }else{
            tfEmail3.setEditable(false);
            tfEmail3.setText("");
        }
    }
}

private void btDupActionPerformed(java.awt.event.ActionEvent evt) {
   //중복확인 페이지 보여주기
    SubUserId f = new SubUserId(this);

    f.setVisible(true);
}
```

```
private void memberProc() {
        //회원가입, 회원정보 수정에 따라 화면 처리하기
        if(memberFlag==MEMBER_REGISER){            //회원가입인 경우
            setTitle("회원가입");
            btAdd.setText("회원가입");
        }else if(memberFlag==MEMBER_EDIT){          //회원정보 수정인 경우
            setTitle("회원정보 수정");
            btAdd.setText("회원정보수정");
            setEditing();
            showByUserid(); //회원정보 조회
        }
    }
    private void setEditing(){
        //회원정보 수정화면에서는 이름, 아이디, 중복확인버튼을 사용할 수 없도록 셋팅
        tfName.setEditable(false);
        tfUserid.setEditable(false);
        btDup.setEnabled(false);
    }
    private void showByUserid() {
        //회원정보 조회해서 화면에 출력하기
       String userid = MemberService.getUserid();
        try {
            MemberDTO dto = dao.selectByUserid(userid);
            tfAddress1.setText(dto.getAddress1());
            tfAddress2.setText(dto.getAddress2());
            tfName.setText(dto.getName());
            tfUserid.setText(dto.getUserid());
            tfZipcode.setText(dto.getZipcode());

            //휴대폰 번호
            String hp = dto.getHp();
```

```
if(hp!=null && !hp.isEmpty()){
    //010-100-2000 =>  - 를 구분자로 split
    String[] hpArr =hp.split("-");
    cbHp1.setSelectedItem(hpArr[0]); //010
    tfHp2.setText(hpArr[1]);  //100
    tfHp3.setText(hpArr[2]);  //2000
 }
 //이메일
 String email = dto.getEmail();
 if(email!=null && !email.isEmpty()){
    //hong@nate.com
    //kim@yahoo.com => 직접입력한 경우
    String[] emailArr = email.split("@");
    String email1 = emailArr[0]; //hong, kim
    String email2 = emailArr[1];  //nate.com, yahoo.com
    tfEmail1.setText(email1);

    //email2가 콤보박스의 이메일 항목중에 있는지 체크=>있으면 콤보에 넣고, 없으면 tfEmail3에 넣는다
    boolean existEmail=false;
    for(String s : emailList){
       if(email2.equals(s)){
          existEmail=true;
          break;
       }
    }//for
    if(existEmail){                //이메일항목 중에 있는 경우
       cbEmail2.setSelectedItem(email2);
    }else{                 //직접 입력한 경우
       tfEmail3.setText(email2);
       cbEmail2.setSelectedItem("직접입력");
    }//if
```

```
        } //if
    } catch (SQLException ex) {        }
}
 private void memberEdit() {
    //회원정보 수정 처리
    String address1= tfAddress1.getText();
    String address2= tfAddress2.getText();
    String pwd= tfPwd.getText();
    String zipcode= tfZipcode.getText();
    String email1= tfEmail1.getText();
    String email3= tfEmail3.getText();
    String hp2= tfHp2.getText();
    String hp3= tfHp3.getText();
    String hp1= (String) cbHp1.getSelectedItem();
    String email2= (String) cbEmail2.getSelectedItem();
    String userid=tfUserid.getText();
    //휴대폰 정보를 입력한 경우에만 3자리 번호를 연결
    String hp="";
    if(hp2!=null && !hp2.isEmpty()     && hp3!=null && !hp3.isEmpty()){
       hp=hp1+"-"+hp2+"-"+hp3;
    }
    //이메일
    String email="";
    if(email1!=null && !email1.isEmpty()){
       if(email2.equals("직접입력")){
          email=email1+"@"+email3;
       }else{
          //직접입력이 아닌 경우 콤보상자의 값과 연결
          email=email1+"@"+email2;
       }
```

```
    }
    if(pwd==null || pwd.isEmpty()){
        JOptionPane.showMessageDialog(this,              "비밀번호를 입력하세요");
        tfPwd.requestFocus();
        return;
    }
    MemberDTO dto = new MemberDTO();
    dto.setAddress1(address1);
    dto.setAddress2(address2);
    dto.setEmail(email);
    dto.setHp(hp);
    dto.setPwd(pwd);
    dto.setZipcode(zipcode);
    dto.setUserid(userid);
    try {
        int cnt = dao.updateMember(dto);
        String msg="";
        if(cnt>0){
            msg="회원정보를 수정하였습니다";
        }else{
            msg="회원정보 수정 실패!";
        }
        JOptionPane.showMessageDialog(this, msg);
    } catch (SQLException ex) {        }
}
public static void main(String args[]) {
   java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new MemberFrame().setVisible(true);
        }
    });    }
```

SubUserId

```
public class SubUserId extends javax.swing.JFrame {
    private MemberFrame memberFrame;
    public SubUserId() {
        initComponents();
        init();
    }
    public SubUserId(MemberFrame memFrame) {
        this();
        this.memberFrame=memFrame;
        this.tfUserid.setText(memFrame.tfUserid.getText());
    }
     private void btOkActionPerformed(java.awt.event.ActionEvent evt) {
        //아이디 중복확인 처리
        checkIdDuplicate();
    }
```

```
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new SubUserId().setVisible(true);
        }
    });
}
private void init() {
    setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);  //자신의 창만 닫히도록
}
private void checkIdDuplicate() {
    //해당 아이디를 사용 가능한지 체크
    //1. 사용자 입력값 읽어오기
    String userid = tfUserid.getText();

    //유효성 검사
    if(userid==null || userid.isEmpty()){
        JOptionPane.showMessageDialog(this, "아이디를 입력하세요");
        return;
    }

    //2. db작업 - select
    MemberDAO dao=new MemberDAO();
    try {
        int result = dao.checkUserid(userid);
```

```
        //3. 결과 처리
         if(result==MemberDAO.USABLE_ID){                    //아이디 사용가능
           int type = JOptionPane.showConfirmDialog(this, "사용가능한 아이디 입니다."
                 + "사용 하시겠습니까?", "아이디확인",JOptionPane.YES_NO_OPTION);

           if (type == JOptionPane.YES_OPTION) {
                  //아이디 사용에 따른 처리
                 //아이디를 회원가입화면에 출력해주고 창을 닫는다.
                 memberFrame.tfUserid.setText(this.tfUserid.getText());
                 memberFrame.setIsDuplicateChecked(true); //아이디 중복 확인을 했다.
                  this.dispose(); //창닫기
           }
         }else if(result==MemberDAO.NONE_USABLE_ID){                  //아이디 사용불가
           JOptionPane.showMessageDialog(this, "해당 아이디가 이미 존재합니다."
                 + "다른 아이디를 입력하세요");
         }else{
            JOptionPane.showMessageDialog(this,    "아이디 중복확인 실패");
         }
      } catch (SQLException ex) {}
   }

}//class
```

Subzipcode

```
public class Subzipcode extends javax.swing.JFrame {
    MemberFrame memFrame;
    ZipcodeDAO dao = new ZipcodeDAO();

    private String[] colNames={"우편번호","시도","구군","동","번지"};
    private String[][] data;

    public Subzipcode() {
        super("우편번호 찾기");
        initComponents();
        myinit();
    }
    public Subzipcode(MemberFrame memFrame) {
        this();
        this.memFrame = memFrame;
    }
    private void myinit(){
        setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
    }
    private void btSearchActionPerformed(java.awt.event.ActionEvent evt) {
        searchZipcode();
    }
    private void jTable1MousePressed(java.awt.event.MouseEvent evt) {
        //jtable에서 row를 선택하면 부모창에 우편번호와 주소 셋팅하기
        int row = jTable1.getSelectedRow();
        String address="";
        String zipcode = jTable1.getValueAt(row, 0).toString();
        for(int i = 1; i<=3; i++){
                String str = jTable1.getValueAt(row, i).toString();
                address += str + "";
      }//for
```

Subzipcode

```
    //부모 창의 우편번호와 주소필드에 셋팅
    memFrame.tfZipcode.setText(zipcode);
    memFrame.tfAddress1.setText(address);
}
private void btSelectActionPerformed(java.awt.event.ActionEvent evt) {
    this.dispose(); //자신의 창 닫기
}
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new Subzipcode().setVisible(true);
        }
    });
}
private void searchZipcode(){        //우편번호찾기
    String dong = tfDong.getText();
     if (dong==null || dong.isEmpty()) {
        JOptionPane.showMessageDialog(this, "읍면동을 입력하세요");
        tfDong.requestFocus();
        return;
    }
   ArrayList<ZipcodeDTO> alist =  dao.selectByDong(dong);
   data = new String[alist.size()][colNames.length];
   for(int i = 0; i < alist.size() ; i++){
      ZipcodeDTO dto = alist.get(i);
      data[i][0] = dto.getZipcode();
      data[i][1] = dto.getSido();
      data[i][2] = dto.getGugun();
```

Subzipcode

```
    data[i][3] = dto.getDong();
     //endBunji가 있을 때만 startBunji와 EndBunji를 연결해서 보여준다.
     String startBunji = dto.getStartbunji();
     String Endbunji = dto.getEndbunji();
     String bunji = startBunji;
     if (Endbunji!=null && !Endbunji.isEmpty()) {
        bunji =  startBunji + "~" + Endbunji;
     }
     data[i][4] = bunji;
}

DefaultTableModel model=new DefaultTableModel();
model.setDataVector(data, colNames);
jTable1.setModel(model);

 //jTable의 컬럼 width 조절하기 - 각 컬럼 사이즈 조절
 //table.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
 jTable1.getColumnModel().getColumn(0).setPreferredWidth(70);
 jTable1.getColumnModel().getColumn(1).setPreferredWidth(110);
 jTable1.getColumnModel().getColumn(2).setPreferredWidth(120);
 jTable1.getColumnModel().getColumn(3).setPreferredWidth(120);
 //jTable1.getColumnModel().getColumn(4).setPreferredWidth(67);
}
```

```
public  class ZipcodeDAO {
   public ArrayList<ZipcodeDTO> selectByDong(String subdong){
      //동이 포함된 주소 조회하기
      // select *  from zipcode where dong like '%개포%';
      ArrayList<ZipcodeDTO> alist = new ArrayList<ZipcodeDTO>();
      Connection conn = null;
      PreparedStatement ps = null;
      ResultSet rs = null;
      try {
         conn = DBConn.getConnection();
         String sql = "select * from zipcode where dong like '%'||?||'%'";
         ps = conn.prepareStatement(sql);
         ps.setString(1, subdong);
         rs = ps.executeQuery();
         while(rs.next()){
            String zipcode = rs.getString("zipcode");
            String sido = rs.getString("sido");
            String gugun = rs.getString("gugun");
            String dong = rs.getString("dong");
            String startbunji = rs.getString("startbunji");
            String endbunji = rs.getString("endbunji");
            int seq = rs.getInt("seq");
            ZipcodeDTO dto = new ZipcodeDTO(zipcode,sido,gugun,dong, startbunji,endbunji,seq);
            alist.add(dto);
         }//while
         System.out.println("우편번호검색결과 : " + alist.size());
      } catch (Exception e) {        System.out.println("sql 예외!!");         e.printStackTrace();
      }finally{  DBConn.dbClose(rs, ps, conn);  }
      return alist;
   }
}
```