


BÀI 1: BILLJDLIALOG

Xây dựng cửa sổ chức năng hiển thị chi tiết phiếu bán hàng và cho phép cập nhật thông tin của phiếu:


Phiếu bán hàng
×

Mã phiếu

10028

Thẻ số

13

Thời điểm đặt hàng

16:31:56 13-04-2025

Nhân viên

user1@gmail.com

Trạng thái

Servicing

Thời điểm thanh toán

	Mã phi...	Đồ uống	Đơn giá	Giảm giá	Số lượng	Thành tiền
<input type="checkbox"/>	100023	Nước trái cây #2x21	\$408.50	17%	6	\$2034.33
<input type="checkbox"/>	100024	Sinh tố #3x17	\$585.80	5%	3	\$1669.53

Xóa đồ uống

Thêm đồ uống

Thanh toán

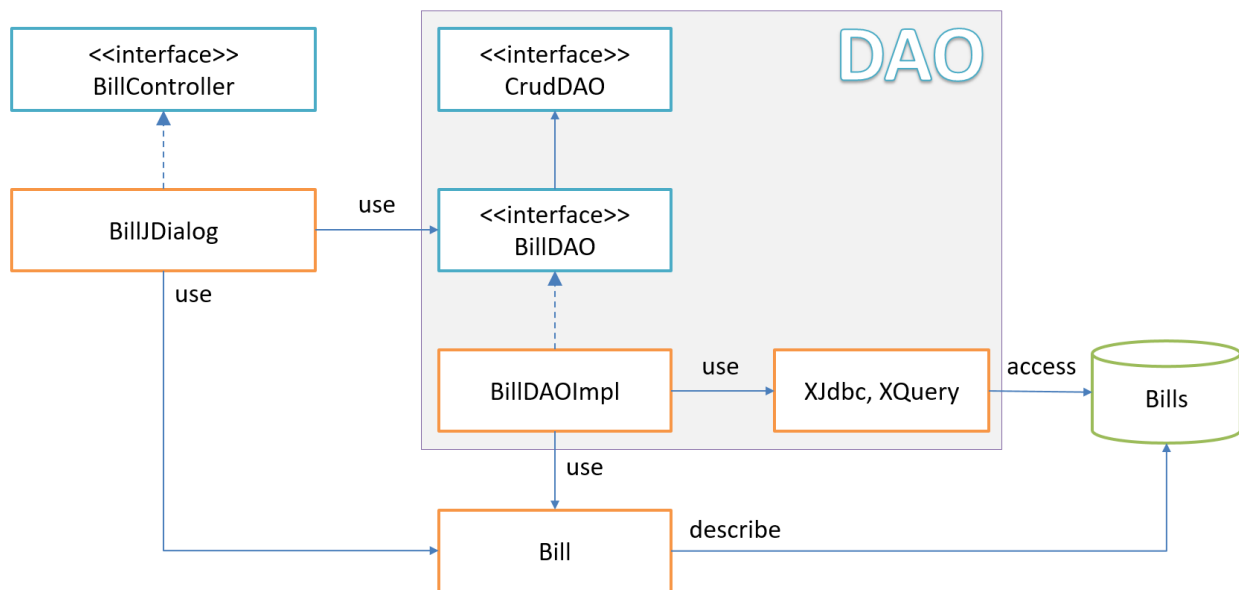
Hủy phiếu

Tương tác	Xử lý tương tác
Mở cửa sổ (BillJDialog)	Hiển thị bill lên form Tải và hiển thị chi tiết bill lên bảng chi tiết phiếu
Click đúp vào đồ uống	Lấy số lượng mới từ người dùng Cập nhật số lượng đồ uống
Click nút Xóa đồ uống (btnRemove)	Xóa đồ uống được chọn khỏi chi tiết bill
Click nút Thêm đồ uống (btnAdd)	Mở cửa sổ chức năng chọn và thêm đồ uống vào chi tiết phiếu
Click nút Thanh toán (btnCheckout)	Cập nhật thời gian và trạng thái Checkout
Click nút Hủy phiếu (btnCancel)	Cập nhật trạng thái Cancel hoặc xóa phiếu nếu phiếu không có đồ uống
Đóng cửa sổ	Xóa phiếu nếu phiếu không có đồ uống

CÁC THÀNH PHẦN CẦN THIẾT

- GUI: BillJDialog
- Controller: BillController
- Tables: Bills, BillDetails
- Entities: Bill, BillDetail
- DAO:
 - CrudDAO, BillDAO, BillDAOImpl, BillDetailDAO, BillDetailDAO
 - XJdbc, XQuery

SƠ ĐỒ TỔ CHỨC



THIẾT KẾ GIAO DIỆN

- Thiết kế giao diện như hình trên
- Đặt tên cho các thành phần giao diện
 - tblBillDetails: bảng chứa chi tiết phiếu
 - btnRemove: Nút xóa đồ uống khỏi phiếu
 - btnAdd: Mở hộp thoại thêm đồ uống vào phiếu
 - btnCheckout: Thanh toán
 - btnCancel: Hủy hóa đơn

CONTROLLER

Tạo interface BillController và cài đặt mã để điều khiển các tương tác

```

public interface BillController {
    void setBill(Bill bill); // truyền bill vào cửa sổ để hiển thị
    void open(); // Hiển thị bill
    void close(); // Xóa bill nếu ko chứa đồ uống nào
    void showDrinkJDialog(); // Hiển thị cửa sổ bổ sung đồ uống vào bill
    void removeDrinks(); // Xóa đồ uống khỏi bill
}
    
```

```

void updateQuantity(); // Thay đổi số lượng đồ uống
void checkout(); // Thanh toán
void cancel(); // Hủy bill
}

public class BillJDialog extends JDialog implements BillController {
    ...
    @Override
    public void removeDrinks() { // xóa đồ uống được tích chọn
        for (int i = 0; i < tblBillDetails.getRowCount(); i++) {
            Boolean checked = (Boolean) tblBillDetails.getValueAt(i, 0);
            if(checked)
                billDetailDao.deleteByld(billDetails.get(i).getld());
        }
        this.fillBillDetails();
    }
    @Override
    public void showDrinkJDialog() { // hiển thị cửa sổ chọn và bổ sung đồ uống
        DrinkJDialog dialog = new DrinkJDialog((Frame) this.getOwner(), true);
        dialog.setBill(bill); // Khai báo vào DrinkJDialog @Setter Bill bill
        dialog.setVisible(true);
        dialog.addWindowListener(new java.awt.event.WindowAdapter() {
            @Override
            public void windowClosed(java.awt.event.WindowEvent e) {
                BillJDialog.this.fillBillDetails();
            }
        });
    }
    @Override
    public void updateQuantity() { // thay đổi số lượng đồ uống
        if (bill.getStatus() == 0) { // chưa thanh toán hoặc chưa bị canceled
            String input = XDialog.prompt("Số lượng mới?");
            if (input != null && input.length() > 0) {
                BillDetail detail = billDetails.get(tblBillDetails.getSelectedRow());
                detail.setQuantity(Integer.parseInt(input));
                billDetailDao.update(detail);
                this.fillBillDetails();
            }
        }
    }
    @Override
    public void checkout() {
        if (XDialog.confirm("Bạn muốn thanh toán phiếu bán hàng?")) {
            bill.setStatus(Bill.Status.Completed.ordinal());
            bill.setCheckout(new Date());
        }
    }
}

```

```

        billDao.update(bill);
        this.setForm(bill);
    }
}
@Override
public void cancel() {
    if (billDetails.isEmpty()) {
        billDao.deleteById(bill.getId());
        this.dispose();
    } else if (XDialog.confirm("Bạn muốn hủy phiếu bán hàng?")) {
        bill.setStatus(Bill.Status.Canceled.ordinal());
        billDao.update(bill);
        this.setForm(bill);
    }
}
void setForm(Bill bill) { // hiển thị bill lên form
    txtId.setText(String.valueOf(bill.getId()));
    txtCardId.setText("Card #" + bill.getCardId());
    txtCheckin.setText(XDate.format(bill.getCheckin(), "HH:mm:ss dd-MM-yyyy"));
    txtUsername.setText(bill.getUsername());
    String[] statuses = {"Servicing", "Completed", "Canceled"};
    txtStatus.setText(statuses[bill.getStatus()]);
    if (bill.getCheckout() != null) {
        txtCheckout.setText(XDate.format(bill.getCheckout(), "HH:mm:ss dd-MM-yyyy"));
    }
    boolean editable = (bill.getStatus() == 0);
    btnAdd.setEnabled(editable);
    btnCancel.setEnabled(editable);
    btnCheckout.setEnabled(editable);
    btnRemove.setEnabled(editable);
}
}

```

KẾT NỐI CONTROLLER VỚI EVENT HANDLER

```

private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.showDrinkJDialog();
}

private void btnRemoveActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.removeDrinks();
}

```

```
private void btnCheckoutActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.checkout();
}

private void btnCancelActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.cancel();
}

private void tblBillDetailsMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    if (evt.getClickCount() == 2) {
        this.updateQuantity();
    }
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    // TODO add your handling code here:
    this.open();
}

private void formWindowClosed(java.awt.event.WindowEvent evt) {
    // TODO add your handling code here:
    this.close();
}
```

BÀI 2: DRINKJIALOG

Xây dựng cửa sổ chức năng cho phép lọc đồ uống theo loại và chọn đồ uống thêm vào phiếu bán hàng được minh họa như hình sau:

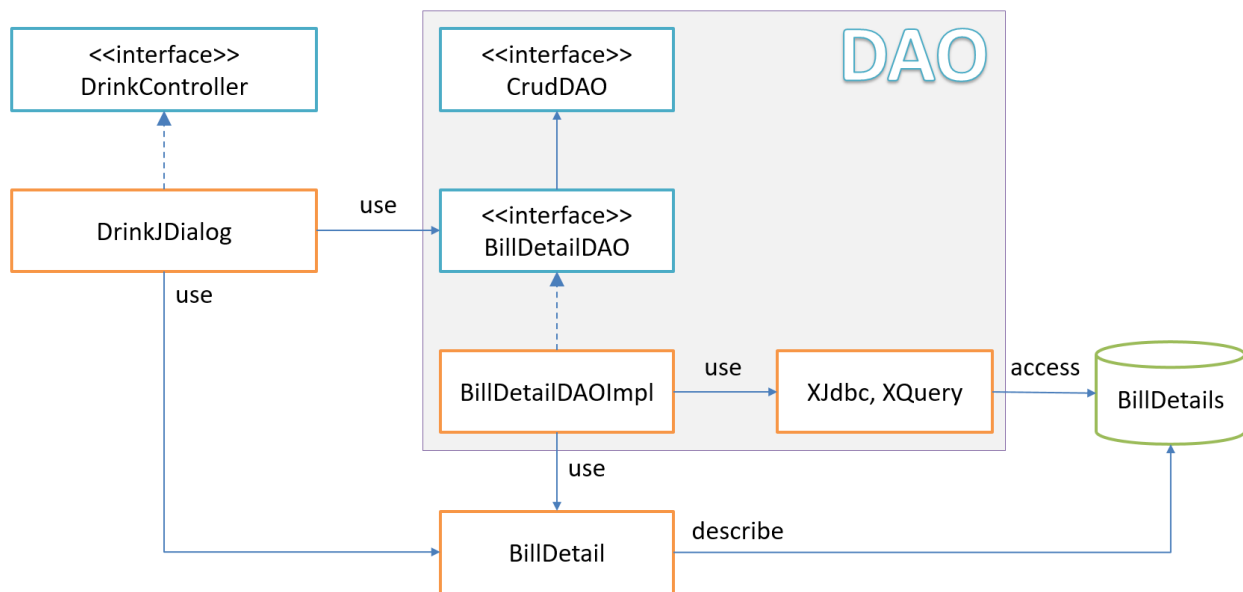


TƯƠNG TÁC	XỬ LÝ TƯƠNG TÁC
Mở cửa sổ (DrinkJDialog)	Tải và hiển thị loại đồ uống, Chọn loại đầu tiên Tải và hiển thị đồ uống của loại được chọn
Click loại đồ uống	Tải và hiển thị lại đồ uống của loại được chọn
Click đúp vào đồ uống	Nhập số lượng và Thêm đồ uống vào bill

CÁC THÀNH PHẦN CẦN THIẾT

- GUI: DrinkJDialog
- Controller: DrinkController
- Tables: Categories, Drinks, Bills, BillDetails
- Entities: Category, Drink, Bill, BillDetail
- DAO:
 - CrudDAO
 - CategoryDAO, CategoryDAOImpl
 - DrinkDAO, DrinkDAOImpl
 - BillDAO, BillDAOImpl
 - BillDetailDAO, BillDetailDAO
 - XJdbc/XQuery

SƠ ĐỒ TỔ CHỨC



THIẾT KẾ GIAO DIỆN

- Thiết kế giao diện như hình trên
- Đặt tên các thành phần
 - tblCategories: Bảng chứa loại đồ uống
 - tblDrinks: Bảng chứa đồ uống theo loại

CONTROLLER

Tạo interface DrinkController và cài đặt mã để điều khiển các tương tác

```

public interface DrinkController {
    void setBill(Bill bill); // nhận bill từ BillJDialog
    void open(); // hiển thị loại và đồ uống
    void fillCategories(); // tải và hiển thị loại đồ uống
    void fillDrinks(); // tải và hiển thị đồ uống
    void addDrinkToBill(); // thêm đồ uống vào bill
}

public class BillJDialog extends JDialog implements BillController {
    ...
    @Setter
    Bill bill;

    CategoryDAO categoryDao = new CategoryDAOImpl();
    List<Category> categories = List.of();
    DrinkDAO drinkDao = new DrinkDAOImpl();
    List<Drink> drinks = List.of();

    @Override
    
```

```

public void open() {
    this.setLocationRelativeTo(null);
    this.fillCategories();
    this.fillDrinks();
}
@Override
public void fillCategories() {
    categories = categoryDao.findAll();
    DefaultTableModel model = (DefaultTableModel) tblCategories.getModel();
    model.setRowCount(0);
    categories.forEach(d -> model.addRow(new Object[] {d.getName()}));
    tblCategories.setRowSelectionInterval(0, 0);
}
@Override
public void fillDrinks() {
    Category category = categories.get(tblCategories.getSelectedRow());
    drinks = drinkDao.findById(category.getId());
    DefaultTableModel model = (DefaultTableModel) tblDrinks.getModel();
    model.setRowCount(0);
    drinks.forEach(d -> {
        Object[] row = {
            d.getId(),
            d.getName(),
            String.format("$%.1f", d.getUnitPrice()),
            String.format("%.0f%%", d.getDiscount()*100)
        };
        model.addRow(row);
    });
}
@Override
public void addDrinkToBill() {
    String quantity = XDialog.prompt("Số lượng?");
    if(quantity != null && quantity.length() > 0){
        Drink drink = drinks.get(tblDrinks.getSelectedRow());
        BillDetail detail = new BillDetail();
        detail.setBillId(bill.getId());
        detail.setDiscount(drink.getDiscount());
        detail.setDrinkId(drink.getId());
        detail.setQuantity(Integer.parseInt(quantity));
        detail.setUnitPrice(drink.getUnitPrice());
        new BillDetailDAOImpl().create(detail);
    }
}
}

```


KẾT NỐI CONTROLLER VỚI EVENT HANDLER

```
private void tblDrinksMouseClicked(java.awt.event.MouseEvent evt) {  
    // TODO add your handling code here:  
    if(evt.getClickCount() == 2){  
        this.addDrinkToBill();  
    }  
}  
  
private void formWindowOpened(java.awt.event.WindowEvent evt) {  
    // TODO add your handling code here:  
    this.open();  
}  
  
private void tblCategoriesMouseClicked(java.awt.event.MouseEvent evt) {  
    // TODO add your handling code here:  
    this.fillDrinks();  
}
```