1. Refactor lần 1

Thiết kế ban đầu có 3 lớp liên quan đến địa chỉ: DeliveryInfo và 2 lớp con của nó là NormalDeliveryInfo và RushDeliveryInfo

Sửa: Chỉ giữ lại một lớp DeliveryInfo duy nhất, cho Order thêm 2 thuộc tính: rushDeliveryInfo, và normalDeliveryInfo

1. Refactor method trong lớp ShippingScreenHandler

Tách lệnh để hiển thị màn hình ra thành một phương thức để tránh trường hợp phương thức bị quá dài

public class ShippingScreenHandler extends BaseScreenHandler implements Initializable {  
  
 @FXML  
 private Label screenTitle;  
  
 @FXML  
 private TextField name;  
  
 @FXML  
 private TextField phone;  
  
 @FXML  
 private TextField address;  
  
 @FXML  
 private TextField instructions;  
  
 @FXML  
 private ComboBox<String> province;  
  
 @FXML  
 private ToggleGroup deliveryTypeGroup;  
  
 @FXML  
 private RadioButton normalDeliveryRadioBtn;  
  
 @FXML  
 private RadioButton rushDeliveryRadioBtn;  
  
  
 private Order order;  
  
 public ShippingScreenHandler(Stage stage, String screenPath, Order order) throws IOException {  
 super(stage, screenPath);  
 this.order = order;  
 }  
  
 @Override  
 public void initialize(URL arg0, ResourceBundle arg1) {  
 final BooleanProperty firstTime = new SimpleBooleanProperty(true); // Variable to store the focus on stage load  
 name.focusedProperty().addListener((observable, oldValue, newValue) -> {  
 if(newValue && firstTime.get()){  
 content.requestFocus(); // Delegate the focus to container  
 firstTime.setValue(false); // Variable value changed for future references  
 }  
 });  
 this.province.getItems().addAll(Configs.*PROVINCES*);  
 normalDeliveryRadioBtn.setSelected(true);  
 }  
  
 @FXML  
 void submitDeliveryInfo(MouseEvent event) throws IOException, InterruptedException, SQLException {  
 // add info to messages  
 DeliveryInfo messages = new DeliveryInfo(  
 name.getText(),  
 phone.getText(),  
 province.getValue(),  
 address.getText(),  
 instructions.getText());  
  
 RadioButton selectedBtn = (RadioButton) deliveryTypeGroup.getSelectedToggle();  
 order.setNormalDeliveryInfo(messages);  
 if(selectedBtn.getText().equals(new String("Normal"))){  
 handleNormalSubmit(messages);  
 }  
 else{  
 handleRushSubmit(messages);  
 }  
 }  
  
 private void handleRushSubmit(DeliveryInfo messages) throws IOException {  
 // create rush delivery screen  
 try {  
 PlaceRushOrderController.*checkRushOrderCondition*(order);  
 messages.setType(DeliveryInfo.*RUSH\_TYPE*);  
 List<OrderMedia> rushMedias = PlaceRushOrderController.*getSupportRushMedias*(order);  
 if(rushMedias.size() == order.getlstOrderMedia().size()){  
 order.setType(Order.*RUSH*);  
 }  
 else{  
 order.setType(Order.*RUSH\_AND\_NORMAL*);  
 }  
 Invoice invoice = getBController().createInvoice(order);  
 showRushDeliveryForm(invoice);  
 }  
 catch (NotSatisfiedRushConditionException e){  
 showDialog(e.getMessage());  
 }  
 }  
  
 private void handleNormalSubmit(DeliveryInfo messages) throws InterruptedException, IOException {  
 try{  
 PlaceOrderController.*validateDeliveryInfo*(messages);  
 messages.setType(DeliveryInfo.*NORMAL\_TYPE*);  
 order.setType(Order.*NORMAL*);  
 // calculate shipping fees  
 int shippingFees = getBController().*calculateShippingFee*(order);  
 order.setShippingFees(shippingFees);  
 // create invoice screen  
 Invoice invoice = getBController().createInvoice(order);  
 showInvoiceScreen(invoice);  
 }catch (InvalidDeliveryInfoException e){  
 showDialog(e.getMessage());  
 }  
 }  
  
 private void showRushDeliveryForm(Invoice invoice) throws IOException {  
 BaseScreenHandler RushDeliveryFormHandler = new RushDeliveryFormHandler(this.stage, Configs.*RUSH\_DELIVERY\_FORM\_PATH*, invoice);  
 RushDeliveryFormHandler.setPreviousScreen(this);  
 RushDeliveryFormHandler.setHomeScreenHandler(homeScreenHandler);  
 RushDeliveryFormHandler.setScreenTitle("Rush Delivery Screen");  
 RushDeliveryFormHandler.setBController(new PlaceRushOrderController());  
 RushDeliveryFormHandler.show();  
 }  
  
 private void showInvoiceScreen(Invoice invoice) throws IOException {  
 BaseScreenHandler InvoiceScreenHandler = new InvoiceScreenHandler(this.stage, Configs.*INVOICE\_SCREEN\_PATH*, invoice);  
 InvoiceScreenHandler.setPreviousScreen(this);  
 InvoiceScreenHandler.setHomeScreenHandler(homeScreenHandler);  
 InvoiceScreenHandler.setScreenTitle("Invoice Screen");  
 InvoiceScreenHandler.setBController(getBController());  
 InvoiceScreenHandler.show();  
 }  
  
 public PlaceOrderController getBController(){  
 return (PlaceOrderController) super.getBController();  
 }

1. Tách phương thức validate thông tin form thành nhiều phương thức con (PlaceRushOrderController)

*/\*\*  
 \* The {****@code*** *PlaceRushOrderController} performs the internal services  
 \* relating to place rush order process  
 \*  
 \** ***@author*** *thamnt  
 \*/*public class PlaceRushOrderController extends BaseController {  
  
 */\*\*  
 \* get all items that support rush delivery in the order being processed  
 \** ***@returns*** *List\<OrderMedia\>  
 \*/* public static List<OrderMedia> getSupportRushMedias(Order order) {  
 List<OrderMedia> rushMedias = new ArrayList<>();  
 for(OrderMedia orderMedia : order.getlstOrderMedia()){  
 if(orderMedia.getMedia().doSupportRushDelivery()){  
 rushMedias.add(orderMedia);  
 }  
 }  
 return rushMedias;  
 }  
  
 */\*\*  
 \* validate if user enters all required field in form and their inputs are valid  
 \*  
 \** ***@param*** *info - the user input got from form  
 \** ***@returns*** *1 if all fields are filled with valid input  
 \** ***@throws*** *InvalidFormInputException if at least one field in form has invalid input  
 \*/* public void validateRushDeliveryForm(DeliveryInfo info){  
 String address = info.getAddress();  
 String dateInput = info.getExpectedDeliveryDate();  
 if(!PlaceOrderController.*validateAddress*(address)){  
 throw new InvalidFormInputException("Please enter valid address");  
 }  
 if(!validateDateInput(dateInput)){  
 throw new InvalidFormInputException("Please enter valid date");  
 }  
 }  
  
 */\*\*  
 \* {****@code*** *PlaceRushOrderController.validateDateInput} only returns true  
 \* if date string follow the format of JavaFx DatePicker value ("uuuu-MM-dd")  
 \* and the input date must be after the date when user places order  
 \** ***@param*** *dateInput - user input got from form  
 \** ***@return*** *\*/* public boolean validateDateInput(String dateInput){  
 if(dateInput == null) return false;  
 DateTimeFormatter dateFormater = DateTimeFormatter.*ofPattern*("uuuu-MM-dd");  
 LocalDate date;  
 try{  
 date = LocalDate.*parse*(dateInput, dateFormater);  
 }  
 catch (DateTimeParseException e){  
 return false;  
 }  
 if(date.isBefore(LocalDate.*now*())) return false;  
  
 return true;  
 }  
  
 */\*\*  
 \* check if the order being processed satisfies rush condition  
 \** ***@param*** *order - the order being processed  
 \** ***@returns*** *boolean - true if order satisfies and false for the other case  
 \** ***@param*** *order  
 \** ***@throw*** *NotSatisfiedRushConditionException  
 \*/* public static void checkRushOrderCondition(Order order) {  
 DeliveryInfo initialDeliveryInfo = order.getNormalDeliveryInfo();  
 if(!*checkRushDeliveryInfoCondition*(initialDeliveryInfo)){  
 throw new NotSatisfiedRushConditionException("We do not support rush delivery to this delivery address");  
 }  
 if(!*doesContainRushMedia*(order)){  
 throw new NotSatisfiedRushConditionException("Your order does not contain any item supporting rush delivery");  
 }  
 }  
  
 */\*\*  
 \* check if order has any item that supports rush delivery  
 \** ***@param*** *order  
 \** ***@return*** *\*/* public static boolean doesContainRushMedia(Order order){  
 List<OrderMedia> orderMediaList = order.getlstOrderMedia();  
 for(OrderMedia o : orderMediaList){  
 if(o.getMedia().doSupportRushDelivery()) return true;  
 }  
 return false;  
 }  
  
 */\*\*  
 \* Check if delivery information satisfies rush condition, i.e province is "Hà Nội"  
 \** ***@param*** *info  
 \** ***@return*** *\*/* public static boolean checkRushDeliveryInfoCondition(DeliveryInfo info) {  
 if(info.getProvince().equals("Hà Nội")) return true;  
 return false;  
 }  
}