CdL: Ingegneria dell'innovazione per le imprese digitali
Corso: PROGETTAZIONE DEL SOFTWARE
Docente: Valeria Seidita
A.A. 2020/21

ODD: Object Design Document

Progetto:

Tesina a cura di:
Sciacchitano Antonio Pio - 0684157
Scroppo Federico - 0667708
Seminara Giuseppe - 0667494
Tortomasi Gianvito - 0679722

Object Design Document

Indice

1. Introduzione

1.1. Scelte nella progettazione degli oggetti

Per la progettazione e realizzazione del sistema il gruppo ha deciso di utilizzare il linguaggio di programmazione Java. Il linguaggio Java, essendo orientato agli oggetti, permette di realizzare una buona modellazione e progettazione di tutte quelle entità che interagiscono in un contesto di noleggio di autoveicoli. Per la realizzazione delle interfacce grafiche si è scelto di utilizzare il framework Swing, il quale include una vasta palette di componenti grafici.

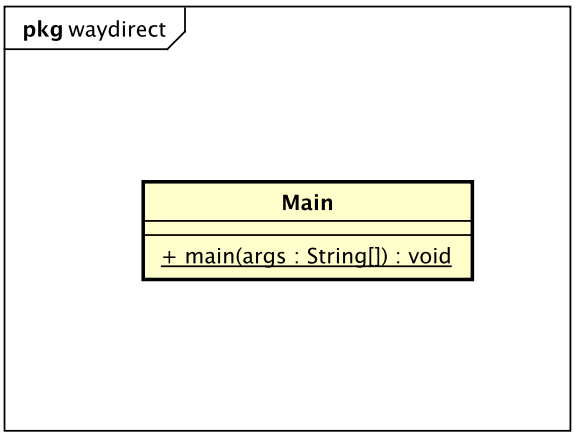
1.2. Librerie di terze parti

Durante la realizzazione del codice sorgente è emersa più volte la necessità di includere librerie esterne per facilitare alcune 'routine' quali la connessione al DBMS oppure la validazione di alcuni dati.
Segue l'elenco di tutte le librerie utilizzate e le relative descrizioni e motivazioni di utilizzo.

- **mysql-connector-java:** driver che permette l'accesso al database, l'esecuzione di query e l'ottenimento del risultato delle stesse;
- **JCalendar:** libreria che aggiunge componenti di controllo Swing quali JDateChooser, utilizzato ad esempio per la selezione della data di nascita in fase di Registrazione;
- **commons-validator:** libreria Java realizzata da Apache che mette a disposizione alcuni metodi che permettono di validare la correttezza di e-mail, numeri di carte di credito e altro;
- **commons-codec:** libreria Java realizzata da Apache utilizzata per generare l'hash in SHA256 delle password;

2. Organizzazione degli oggetti in package

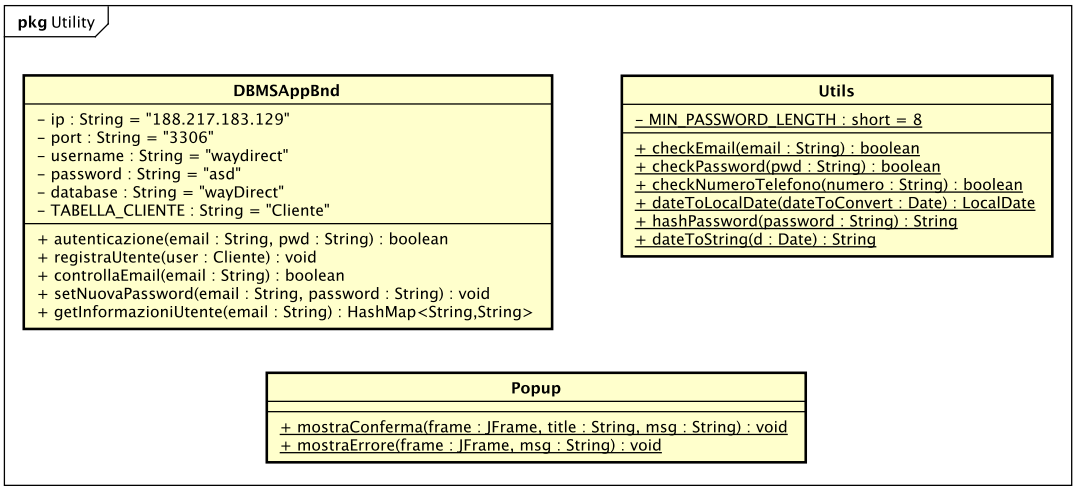
2.1. Package Principale



In questo package è contenuta la classe principale.

Classe	Descrizione
Main	Classe che contiene il metodo main necessario per l'esecuzione del programma.

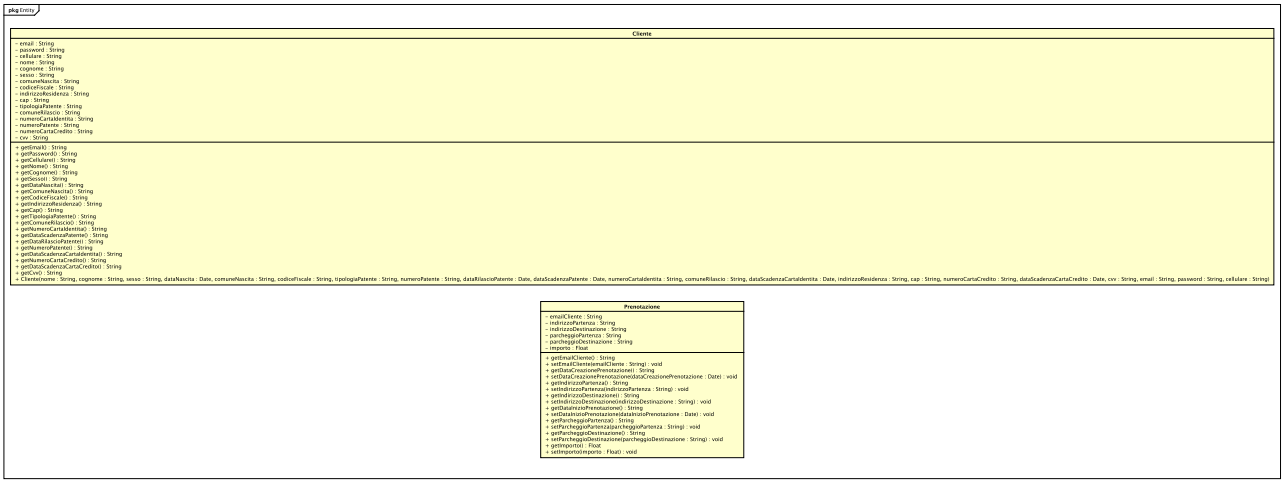
2.2. Package Utility



In questo package sono contenute tutte quelle classi di utilizzo frequente in tutta la struttura del programma.

Classe	Descrizione
DBMSAppBnd	Classe che contiene quei metodi che permettono alle Control di interagire con il database.
Utils	Classe con metodi statici come ad esempio hashPassword() o dateToLocalDate(). Tali metodi sono stati racchiusi in questa classe per favorirne il riuso.
Popup	Classe che permette la creazione e la visualizzazione a video di popup di errore o di conferma.

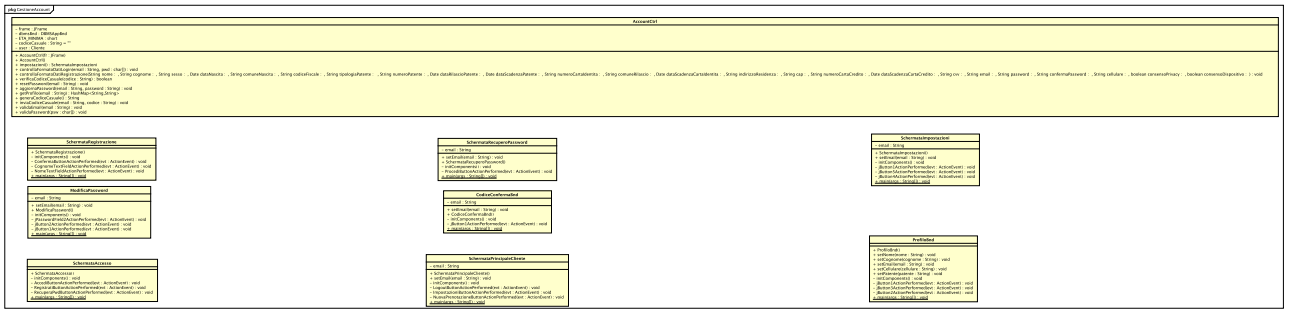
2.3. Package Entity



In questo package sono contenute le classi che modellano le Entity individuate in fase di progettazione.

Classe	Descrizione
Cliente	Classe che contiene le informazioni relative al Cliente.
Prenotazione	Classe che contiene le informazioni relative alle Prenotazioni.

2.4. Package Gestione Account

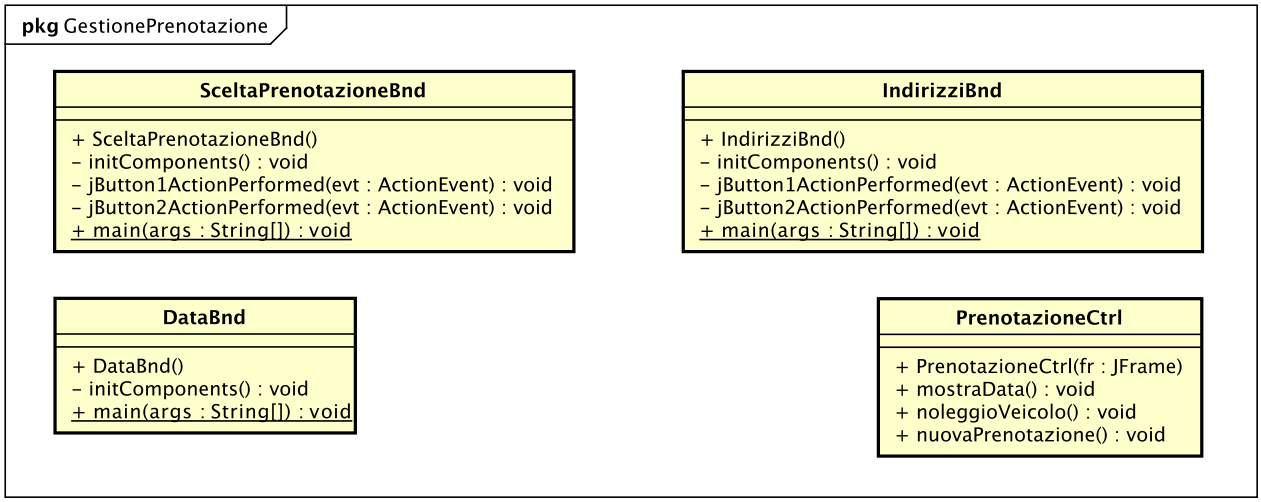


Nel package Gestione Account sono contenute le classi che gestiscono l'autenticazione nel software e la gestione dell'account dell'Utente attraverso il menù impostazioni.

Classe	Descrizione
AccountCtrl	Control che si occupa di verificare il formato dei dati inseriti in fase di registrazione/login e di inviare al DBMS le richieste inerenti alla gestione dell'account Utente.
SchermataAccesso	Schermata che permette all'Utente di inserire le credenziali per effettuare il login, o di accedere alla registrazione o al form di recupero password.
SchermataPrincipaleCliente	Schermata che permette al Cliente creare nuove prenotazioni, gestire le prenotazioni e riconsegnare il veicolo. Il cliente inoltre può accedere al menù di impostazioni ed effettuare il logout dal software.
SchermataImpostazioni	Tramite questa schermata il Cliente può accedere alle varie funzionalità per gestire il proprio account.

SchermataRegistrazione	Schermata che permette al Cliente di creare un nuovo account.
CodiceConfermaBnd	Schermata visualizzata in fase di registrazione che permette al Cliente di inserire il codice di conferma ricevuto alla propria e-mail.
SchermataRecuperoPassword	Schermata che permette al Cliente di ottenere una nuova password all'indirizzo email associato al suo account.
ProfiloBnd	Schermata che mostra al Cliente i dati associati al suo account.

2.5. Package Gestione Prenotazione



Nel package Gestione Account sono contenute le classi che gestiscono la creazione, eliminazione e modifica della prenotazione.

Classe	Descrizione
PrenotazioneCtrl	Control che si occupa di verificare il formato dei dati inseriti in fase di prenotazione e di inviare al DBMS le richieste per la gestione delle prenotazioni.
SceltaPrenotazioneBnd	Schermata che permette al Cliente di scegliere il tipo di prenotazione da creare.
DataBnd	Schermata che permette al Cliente di inserire le date di inizio e fine prenotazione.
IndirizziBnd	Schermata che permette al Cliente di inserire l'indirizzo di partenza e di destinazione per cui vuole effettuare la prenotazione.

3. Interfacce delle classi

Package Principale

- **Main.java**

Metodo	Comportamento
public static void main (String [] args)	Il seguente metodo viene invocato all'avvio del programma e si occupa di istanziare la schermata di accesso.

Package Utility

- **DBMSAppBnd.java**

Metodo	Comportamento
public boolean autenticazione (String email, String pwd)	Il seguente metodo si occupa di tentare l'autenticazione dell'Utente comunicando con il DBMS. Restituisce true se l'autenticazione va a buon fine, altrimenti false.
public void registraUtente (Cliente user)	Il seguente metodo si occupa di registrare nel DBMS un nuovo account. Se l'e-mail dell'Utente che tenta di registrarsi è già presente nel DBMS restituisce false, altrimenti se la registrazione va a buon fine restituisce true.
public boolean controllaEmail (String email)	Il seguente metodo si occupa di controllare se una data e-mail è già presente nel DBMS. Se già presente restituisce true, altrimenti false.
public HashMap<String,String> getInformazioniUtente (String email)	Il seguente metodo si occupa di ottenere le informazioni dell'Utente al fine di visualizzare nella schermata "Visualizza profilo".

- **Popup.java**

Metodo	Comportamento
public static void mostraConferma (JFrame frame, String title, String msg)	Il seguente metodo si occupa di mostrare un messaggio conferma per il quale è possibile specificare il titolo della finestra e il messaggio di conferma.
public static void mostraErrore (JFrame frame, String msg)	Il seguente metodo si occupa di mostrare un messaggio di conferma. È possibile specificare il messaggio di errore come parametro.

- **Utils.java**

Metodo	Comportamento
public static boolean checkEmail(String email)	Il seguente metodo si occupa di validare il formato dell'e-mail ricevuta come parametro. Restituisce true se valido, altrimenti false.
public static boolean checkPassword (String pwd)	Il seguente metodo si occupa di validare il formato della password ricevuta come parametro. Restituisce true se valido, altrimenti false.
public static boolean checkNumeroTelefono (String numero)	Il seguente metodo si occupa di validare il formato del numero di telefono ricevuto come parametro. Restituisce true se valido, altrimenti false.
public static LocalDate dateToLocalDate (Date dateToConvert)	Il seguente metodo si occupa di convertire un oggetto Date in LocalDate.
public static String hashPassword (String password)	Il seguente metodo si occupa di generare l'hash SHA 256 di una password e restituirlo.
public static String dateToString (Date d)	Il seguente metodo si occupa di convertire un oggetto Date in String, rispettando il formato YYYY-MM-DD richiesto dal DBMS.

Package Gestione Account

- **AccountCtrl.java**

Metodo	Comportamento
public SchermataImpostazioni impostazioni ()	Il seguente metodo si occupa di istanziare e mostrare la schermata di impostazioni.
public void controllaFormatoDatiLogin (String email, char [] pwd)	Il seguente metodo si occupa di controllare il formato dei dati inseriti nelle JTextField del login. Se i dati sono validi avvia la procedura di autenticazione e restituisce true, altrimenti restituisce false.
public Boolean controllaFormatoDatiRegistrazione	Il seguente metodo si occupa di controllare il formato dei dati inseriti nei campi di controllo della schermata di registrazione. Se i dati sono validi avvia la procedura di registrazione e restituisce true, altrimenti restituisce false.
public boolean verificaCodiceCasuale (String codice)	Il seguente metodo si occupa di verificare che il codice di conferma inserito in fase di registrazione dal Cliente coincida con quello generato dal sistema.
public void resetPassword (String email)	Il seguente metodo si occupa di generare una password casuale e di modificarla comunicando con il DBMS. Viene richiamato in fase di Recupero Password.
public void aggiornaPassword (String email, String password)	Il seguente metodo si occupa di aggiornare la password la password dell'Utente in fase di Modifica Password.
public HashMap<String,String> getProfilo(String email)	Il seguente metodo si occupa di ottenere tramite il DBMS i dati del profilo dell'Utente e di restituirli sotto forma di HashMap.
private String generaCodiceCasuale ()	Il seguente metodo si occupa di generare e restituire un codice casuale da inviare all'email dell'Utente in fase di Registrazione.
private void inviaCodiceCasuale(String email, String codice)	Il seguente metodo si occupa di inviare all'e-mail del Cliente il codice casuale precedentemente generato.
private boolean validaEmail(String email)	Il seguente metodo si occupa di validare l'e-mail e restituire true in caso di e-mail valida, altrimenti false.
private boolean validaPassword(char[] psw)	Il seguente metodo si occupa di validare la password e restituire true in caso di password valida, altrimenti false.

- **CodiceConfermaBnd.java**

Metodo	Comportamento
public void setEmail(String email)	Il seguente metodo si occupa di memorizzare l'e-mail all'interno della boundary.

- **ModificaPassword.java**

Metodo	Comportamento
public void setEmail(String email)	Il seguente metodo si occupa di memorizzare l'e-mail all'interno della boundary.

- **ProfiloBnd.java**

Metodo	Comportamento
public void setNome(String nome)	Il seguente metodo si occupa di memorizzare il nome all'interno della boundary.
public void setCognome(String cognome)	Il seguente metodo si occupa di memorizzare il cognome all'interno della boundary.
public void setEmail(String email)	Il seguente metodo si occupa di memorizzare l'e-mail all'interno della boundary.
public void setCellulare(String cellulare)	Il seguente metodo si occupa di memorizzare il numero di cellulare all'interno della boundary.
public void setPatente(String patente)	Il seguente metodo si occupa di memorizzare la tipologia di patente posseduta all'interno della boundary.

- **SchermataImpostazioni.java**

Metodo	Comportamento
public void setEmail(String email)	Il seguente metodo si occupa di memorizzare l'e-mail all'interno della boundary.

- **SchermataPrincipaleCliente.java**

Metodo	Comportamento
public void setEmail(String email)	Il seguente metodo si occupa di memorizzare l'e-mail all'interno della boundary.

- **SchermataRecuperoPassword.java**

Metodo	Comportamento
public void setEmail(String email)	Il seguente metodo si occupa di memorizzare l'e-mail all'interno della boundary.

Package Gestione Prenotazione

- **PrenotazioneCtrl.java**

Metodo	Comportamento
public void mostraData()	Il seguente metodo si occupa di istanziare e mostrare la schermata di scelta della data di prenotazione.
public void noleggioVeicolo()	Il seguente metodo si occupa di istanziare e mostrare la schermata di scelta degli indirizzi di prenotazione.

public void nuovaPrenotazione()	Il seguente metodo si occupa di istanziare e mostrare la schermata di scelta del tipo di noleggio.