

WinkelApplicatie Aanpassen

Je hebt nu een werkende WinkelApplicatie binnen NetBeans. Om je wegwijs te maken in de applicatie, gaan we de volgende aanpassingen doorvoeren:

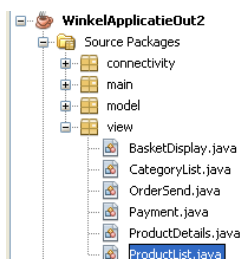
- De tekst euro vervangen door het €-teken
- Klant toevoegen aan de WinkelApplicatie

De tekst euro vervangen door het €-teken

Tijdens het uitvoeren van de applicatie zien we dat alle bedragen worden geschreven in de vorm euro0.0. Het zou mooier zijn om hier €0.0 van de maken.



Het bovenstaande scherm is een van de plaatsen waar we euro0.0 terug kunnen vinden. Dit scherm nemen we als startpunt om te kijken waar we euro moeten aanpassen.



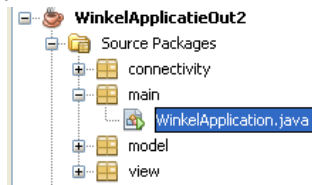
1. Open de java klasse view.ProductList, dit is de klasse waarmee de lijst met producten wordt weergegeven.

```

59
60 | JLabel lblPrice = new JLabel(WinkelApplication.CURRENCY + product.getPrice());
61 | lblPrice.setBounds(380, verticalPosition + i * offset + 15, 80, 20);
62 | lblPrice.setFont(WinkelApplication.FONT_12_BOLD);
63 | add(lblPrice);
64

```

- Op regel 60 van ProductList.java, wordt het label voor de prijs gemaakt. Bij de constructor aanroep kunnen we zien dat een prijs wordt opgebouwd uit CURRENCY en product.getPrice(). Hierin is CURRENCY waarschijnlijk de String "euro" en geeft product.getPrice() de prijs van het product.



- Open het bestand WinkelApplication.java. Hierin moeten we de variabele CURRENCY kunnen vinden.

```

23 | public static final String NAME = "Winkelapplicatie";
24 | public static final String CURRENCY = "euro";

```

- Op regel 24 vinden we de String CURRENCY. In de hele applicatie, waar deze string gebruikt wordt, zal euro komen te staan. Nu we weten waar de tekst euro vandaan komt, kunnen we hem gaan aanpassen naar het €-teken.

```

23 | public static final String NAME = "Winkelapplicatie";
24 | public static final String CURRENCY = "\u20AC";

```

- Verander de tekst **euro** naar **\u20AC**. De string \u20AC staat voor het unicode karakter €.



- Als alles goed is gegaan, heb je nu overal waar nodig het €-teken staan.

Een klant toevoegen aan de WinkelApplicatie.

Als er een bestelling wordt geplaatst, dan wordt er altijd gevraagd om de verzendgegevens. Het zou makkelijker zijn om dit voor een klant op te slaan en weer te geven bij een bestelling. Zo kan er sneller een bestelling geplaatst worden, omdat de gegevens niet nogmaals ingevoerd moeten worden.

Het toevoegen van een klant aan de WinkelApplicatie doen we in twee stappen:

- Klant toevoegen aan de database.
- Klant toevoegen aan de applicatie
- Het betalingsscherm aanpassen

Producten	Aantal	Prijs
Dames onderbroek	1	€3.5
Cars	1	€4.25
Dames spijkerbroek	1	€8.9
Totaal:		€16.65

Verzendgegevens

Naam: Postcode:

Adres: Woonplaats:

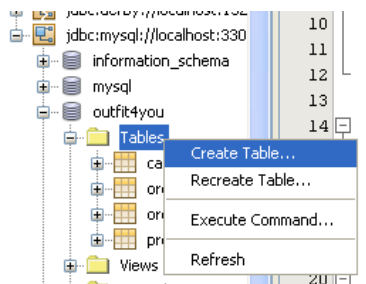
Betaalmethode:

Opmerking:

Klant toevoegen aan de database

Een klant in de WinkelApplicatie krijgt de volgende velden:

- Id
- Naam
- Adres
- Postcode
- Woonplaats



1. Open in de services tab, de database outfit4you. Klik met de rechtermuisknop op Tables en kies voor Create Table...

Create Table

Table name: klant

Key	Index	Null	Unique	Column name	Data type	Size
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	klant_id	INT	0
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	naam	VARCHAR	50
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	adres	VARCHAR	50
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	postcode	VARCHAR	50
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	woonplaats	VARCHAR	50

Buttons: Add column, Remove, OK, Cancel

2. Vul de gegevens in zoals hierboven weergegeven.

SQL Command 1


Connection: jd...

```
1 select * from outfit4you.klant
```

select * from outfit4you.klant

Page Size: 20 | Total Rows: 0 | Matching Rows:

#	klant_id	naam	adres

3. Klik met de rechtermuisknop om de tabel klant en kies voor View Data... Je krijgt nu bovenstaand scherm te zien. Onderaan staan alle regels die in de tabel klant staan. Omdat we deze tabel net hebben gemaakt, is deze nog leeg. Klik op  om een nieuwe klant toe te voegen.

Insert Record(s)

Press CTRL+Tab to exit data entry mode from the table. Press CTRL+0 to set NULL value and CTRL+1 to set DEFAULT value for a given column.

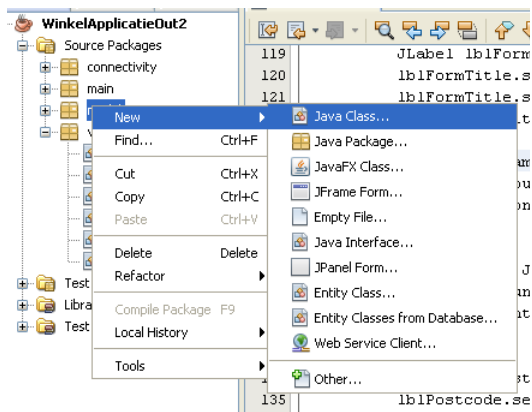
#	klant_id	naam	adres	postcode	woonplaats
1	1	Ralph Benjamin Ruijs	Groetstraat 52	1024TX	Amsterdam

Buttons: Show SQL, Add Row, Remove, OK, Cancel

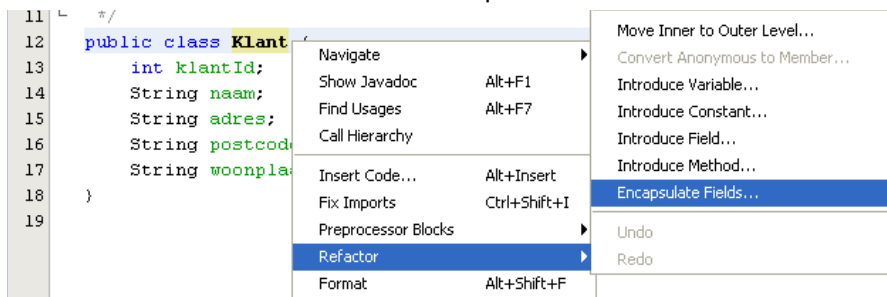
4. Vul de gegevens in van een klant. Het scherm met de inhoud van de database zal meteen je nieuw ingevoerde klant laten zien.

Klant toevoegen aan de applicatie

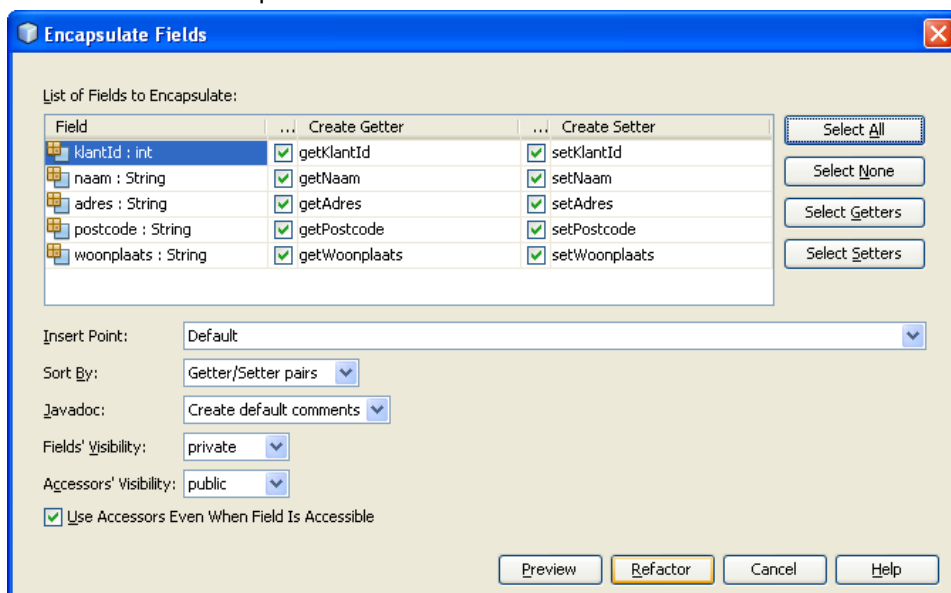
Nu er een klant in de database staat, gaan we deze toevoegen aan de applicatie. We beginnen met het maken van een klant model.



1. Klik met de rechtermuisknop op de package model en kies voor New -> Java Class...
2. Geef deze klasse de naam Klant en klik op Finish.



3. Voeg de nodige velden toe aan de klasse klant. Klik met de rechtermuisknop op Klant en kies voor Refactor -> Encapsulate Fields...



4. Klik op de knop Select All en daarna op Refactor

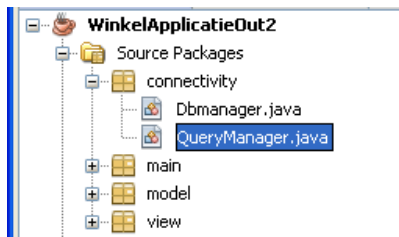
```

19 public Klant() {
20 }
21
22 public Klant(int klantId, String naam, String adres,
23             String postcode, String woonplaats) {
24     this.klantId = klantId;
25     this.naam = naam;
26     this.adres = adres;
27     this.postcode = postcode;
28     this.woonplaats = woonplaats;
29 }

```

5. Maak in de klasse klant twee constructors, een default constructor en een constructor waarin alle velden worden geïnitialiseerd.

Er bestaat nu een klant in de database en we hebben een klasse om deze klant in op te slaan. Nu kunnen we de klasse QueryManager aanpassen, zodat we een klant kunnen ophalen uit de database.



1. Open de java klasse QueryManager.

```

49 public Product getProduct(int productId) {
50     Product product = new Product();
51     try {
52         String sql = "SELECT * FROM product WHERE product_id='" + productId;
53         ResultSet result = dbmanager.doQuery(sql);
54         if (result.next()) {
55             product = new Product(result.getInt("product_id"),
56                                   result.getInt("categorie_id"),
57                                   result.getString("naam"),
58                                   result.getString("omschrijving"),
59                                   result.getDouble("prijs"));
60         }
61     } catch (SQLException e) {
62         System.out.println(Dbmanager.SQL_EXCEPTION + e.getMessage());
63     }
64     return product;
65 }

```

2. Net als de methode getProduct op regel 49, gaan we een methode getKlant maken.

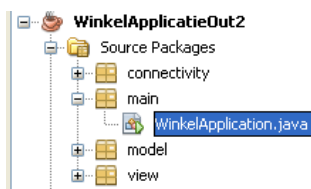
```

68 public Klant getKlant(int klantId) {
69     Klant klant = new Klant();
70     try {
71         String sql = "SELECT * FROM klant " +
72             "WHERE klant_id=" + klantId + "'";
73         ResultSet result = dbmanager.doQuery(sql);
74         if (result.next()) {
75             klant = new Klant(result.getInt("klant_id"),
76                 result.getString("naam"),
77                 result.getString("adres"),
78                 result.getString("postcode"),
79                 result.getString("woonplaats"));
80         }
81     } catch (SQLException e) {
82         System.out.println(Dbmanager.SQL_EXCEPTION + e.getMessage());
83     }
84     return klant;
85 }

```

- Schrijf de methode `getKlant` zoals hierboven weergegeven.

Voordat we het betalingsscherm kunnen aanpassen, moet er een klant zijn aangemeld bij onze applicatie. De klasse `WinkelApplication` zorgt voor globale toegang van de database en het winkelmandje. Hier zullen we de klant aan toe voegen.



- Open het bestand `WinkelApplication.java`.

```

33 /** models used in the application */
34 private model.Basket basket;
35 /** klant */
36 private model.Klant klant;
37 /** the main instance */

```

- Voeg bij de lijst van velden de klant toe.

```

117 /**
118  * @return de klant
119  */
120 public static model.Klant getKlant() {
121     return getInstance().klant;
122 }

```

- Voeg de methode `getKlant` toe. Deze methode willen we overal kunnen gebruiken in de applicatie. Let daarom op het keyword `static` en het gebruik van de methode `getInstance`.

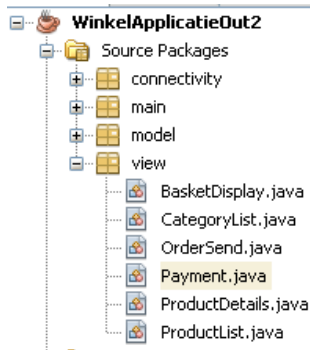
```

60 public void startup() {
61     klant = queryManager.getKlant(1);
62
63     mainWindow = new JFrame(NAME);

```

- Er is nog geen functionaliteit om je aan te melden bij de applicatie, daarom zal bij het starten van de applicatie klant #1 aangemeld worden. Pas de methode `startup` hiervoor aan.

Het betalingsscherm aanpassen



1. Open het bestand Payment.java uit de package view.

```
124      JLabel lblNaam = new JLabel("Naam:");
125      lblNaam.setBounds(20, verticalPosition + products.s
126      lblNaam.setFont(WinkelApplication.FONT_10_BOLD);
127      add(lblNaam);
128
129      tfNaam = new JTextField();
130      tfNaam.setBounds(120, verticalPosition + products.s
131      tfNaam.setFont(WinkelApplication.FONT_10_BOLD);
132      add(tfNaam);
133
134      JLabel lblPostcode = new JLabel("Postcode:");
135      lblPostcode.setBounds(320, verticalPosition + produ
136      lblPostcode.setFont(WinkelApplication.FONT_10_BOLD)
137      add(lblPostcode);
138
139      tfPostcode = new JTextField();
140      tfPostcode.setBounds(420, verticalPosition + produc
141      tfPostcode.setFont(WinkelApplication.FONT_10_BOLD);
142      add(tfPostcode);
```

2. Vanaf regel 124 vinden we de velden voor de verzend gegevens.

```
132      tfNaam.setText(WinkelApplication.getKlant().getNaam());
143      tfPostcode.setText(WinkelApplication.getKlant().getPostcode());
154      tfAddress.setText(WinkelApplication.getKlant().getAdres());
165      tfWoonplaats.setText(WinkelApplication.getKlant().getWoonplaats());
```

3. Verander de tekst in de invoervelden door bovenstaande regels op de juiste plek toe te voegen

Verzendgegevens			
Naam:	<input type="text" value="Ralph Benjamin Ruijs"/>	Postcode:	<input type="text" value="1024TX"/>
Adres:	<input type="text" value="Groetstraat 52"/>	Woonplaats:	<input type="text" value="Amsterdam"/>
Betaalmethode:	<input type="text" value="Vooraf per bank"/>		
Opmerking:	<input type="text"/>		
<input type="button" value="Verzend bestelling"/>			

4. Als alles goed is gegaan, dan worden de gegevens van klant automatisch ingevuld.