

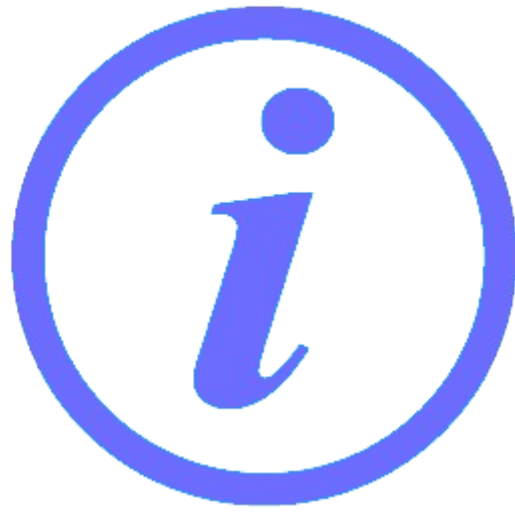


Cheat Sheet

Brent Courtois

Inhoud

2	Inhoud	
	Info	
4	Globaal	
		JDK JAR, WAR & EAR Primitieve datatypes
5	Eclipse	
	Voorbeelden	



Info

Globaal

JDK

JDK (Java Development kit)

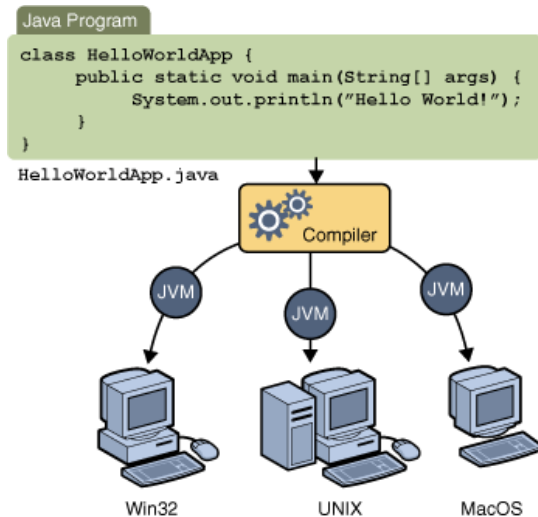
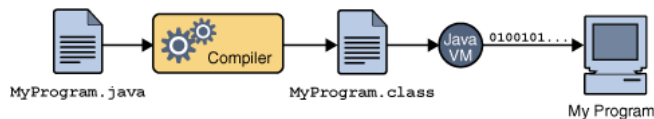
JRE + Development tools

JRE (Java Runtime Environment)

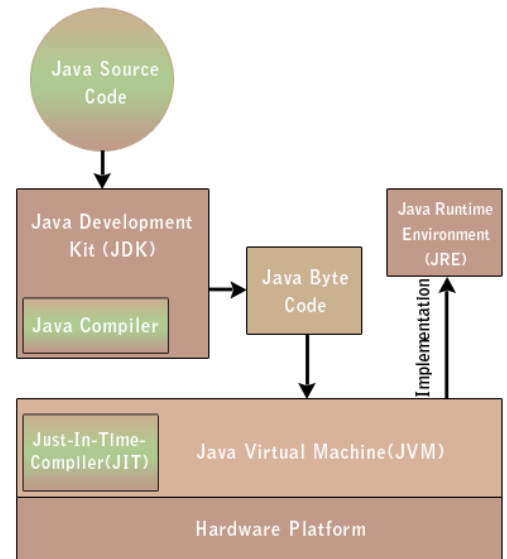
JVM + libraries en andere bestanden dat de JVM gebruikt gedurende runtime

JVM (Java Virtual Machine)

Machine die Java bytecode kan uitvoeren



De gehele JDK is steeds platform afhankelijk, maar:
De JVM van eender welk platform kan classfiles omzetten naar bytecode voor die machine



JAR, WAR & EAR

JAR (Java Archive)

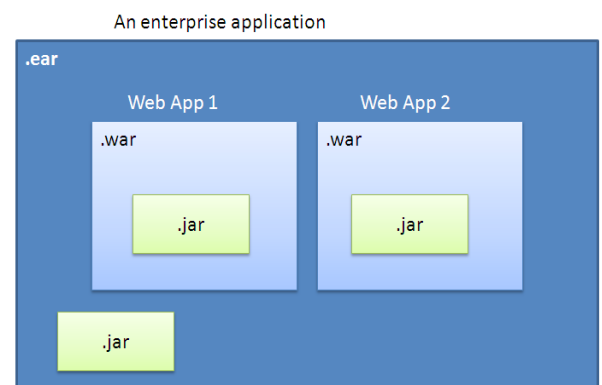
.class files + eventueel MANIFEST.MF

WAR (Web Archive)

jars + web.xml

EAR (Enterprise Archive)

+ deployment descriptor
kan wars en jars bevatten



Primitieve datatypes

1 byte = 8bits

Datatype	Langte in bytes	Bereik
byte	1	-128 ... 127
short	2	-32 768 ... 32767
int	4	-2 147 483 648 ... 2 147 483 647
long	8	-9 223 372 036 854 775 808 ... 9 223 372 036 854 775 807
float	4	$2^{-128} \dots 2^{127}$
double	8	$2^{-1024} \dots 2^{1023}$

Een Java programma compileren/uitvoeren

Installatie jdk 8

- download Java SE Development Kit
(<http://www.oracle.com/technetwork/java/javase/downloads/index.html>)
- voer de exe uit en laat de standaardinstellingen staan
- Java wordt voor u geïnstalleerd in [C:\Program Files\Java](#)



System/environment variable

- Properties computer
- Advanced system settings
- Environment variables
- onder System variables: PATH
- voeg het path van uw JDK vanachter toe, variabelen worden gescheiden door ;
bv: [C:\Program Files\Java\jdk1.8.0_51\bin](#)



- Deze stap maakt compileren via de commandprompt een stuk gemakkelijker



Schrijf programma

- Maak ergens een directory
- Schrijf een programma in dit directory

`package` directory;

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

EXAMPLE



- Zorg ervoor dat het package de directory is die je hebt aangemaakt

- Geef het bestand de extensie .java

Uitvoeren

- Open je command prompt (windows key+r → cmd)
- Ga naar je directory (gebruik cd en cd.)
- compileer:
javac HelloWorld.java
- ga een directory hoger (cd.)
- voer uit:
java directory.HelloWorld

Eclipse

Installatie

- Download Eclipse IDE voor Java EE Developers:
(<http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/lunasr2>)
- pak het zip-bestand uit. Bv in C:\



Eclipse naar uw JDK verwijzen

- Window
- Preferences
- Java
- Installed JREs
- add...
- Standard VM
- Next>
- Directory...
- Ga naar waar je JDK hebt geïnstalleerd en klik OK
(bv: C:\Program Files\Java\jdk1.8.0_51)



- Alt+Shift+O zorgt voor alle nodige imports
- Window → Preferences → Java → Editor → Templates





Standard Edition

Gloabaal

Versie verschillen

Java 1.0	1996
	initiële release
Java 1.1	1997
	inner classes jdbc reflection
Java 1.2	1998
	collections swing
Java 1.3	2000
Java 1.4	2002
Java 1.5	2004
	generics annotations enum for-each
Java 1.6	2006
	@Override
Java 1.7	2011
	switch try with resources
Java 1.8	2014
	lambda streams default method

Terminologie

Association

Has a-relatie. (bv: Een bedrijf heeft een werknemer)

→ Aggregation

Uses a-relatie (Een cursus heeft een leerkracht, als de cursus ophoudt blijft de leerkracht bestaan)

→ Composition

Owns a-relatie (bv: Een school heeft een cursus, als de school ophoudt verdwijnt die cursus ook)

Inheritance

Is a-relatie (bv: Een leerkracht is een persoon)

Overloading

→ Method overloading

Methodes met zelfde naam, maar verschillende parameterlijsten

→ Constructor overloading

Constructors met verschillende parameterlijsten

Overriding

Een inherited/implemented methode overschrijven

Encapsulation

Velden en methodes worden samengehouden in een klasse

Information hiding

Velden enkel aanpasbaar via methodes van die klasse (private)

Polymorphism

Een object kan als verschillende dingen gebruikt worden (superklasse/ interface)



- Indien je een constructor binnen een constructor aanroept moet dit het eerste statement zijn
- In een static method mag je geen non-static veld gebruiken (main is een speciaal geval)
- Object a = Object b; → Adres van a wordt nu b's adres. Wijzigingen op a worden ook op b toegepast en vice versa

- Gebruik voor berekeningen met grote getallen of kommagetallen steeds BigDecimal
- Als je volgende zaken kan toepassen, doe dat dan:
 - inheritance
 - static
 - private
 - final (meestal ook static)
 - enum
 - generics
- Om tijd tussen datums te berekenen:
 - `int[] array = {1,2,3,4,5};` → is geldige syntax
 - `(int)(Math.random()*x)+y` geeft een random int terug tussen y en x+y
 - Indien je een String vaak wijzigt gebruik je beter een StringBuilder



Visibility (Access Levels)

Modifier	Class	Package	Subclass	World
public	Y	Y	Y	Y
protected	Y	Y	Y	N
none(package)	Y	Y	N	N
private	Y	N	N	N

- Gebruik steeds de laagste visibility mogelijk waarmee je model nog steeds werkt. Dit bevordert information hiding

TIP

Alle voorbeelden vind je in:

<https://github.com/BRNTZN/VOORBEELDEN>

Invoer

System

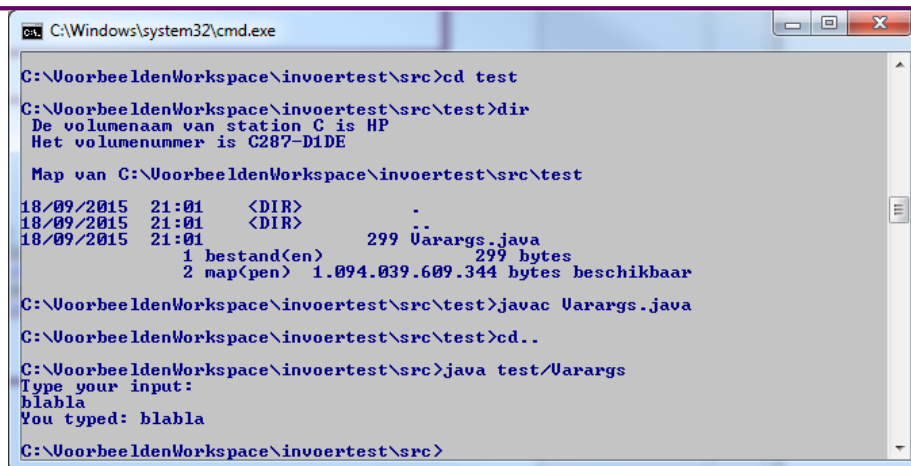
<https://github.com/BRNTZN/VOORBEELDEN/tree/master/invoertest>

```
package test;

import java.util.Scanner; ⚠

public class Varargs {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Type your input:");
        String invoer = sc.nextLine();
        System.out.println("You typed: "+invoer);
        sc.close();
    }
}
```



```
C:\Windows\system32\cmd.exe

C:\VoorbeeldenWorkspace\invoertest\src>cd test
C:\VoorbeeldenWorkspace\invoertest\src\test>dir
De volumenaam van station C is HP
Het volumenummer is C287-D1DE


Map van C:\VoorbeeldenWorkspace\invoertest\src\test

18/09/2015  21:01    <DIR>          .
18/09/2015  21:01    <DIR>          ..
18/09/2015  21:01                299 Varargs.java
               1 bestand(en)          299 bytes
               2 map(pen)  1.094.039.609.344 bytes beschikbaar

C:\VoorbeeldenWorkspace\invoertest\src\test>javac Varargs.java
C:\VoorbeeldenWorkspace\invoertest\src\test>cd..
C:\VoorbeeldenWorkspace\invoertest\src>java test/Varargs
Type your input:
blabla
You typed: blabla
C:\VoorbeeldenWorkspace\invoertest\src>
```

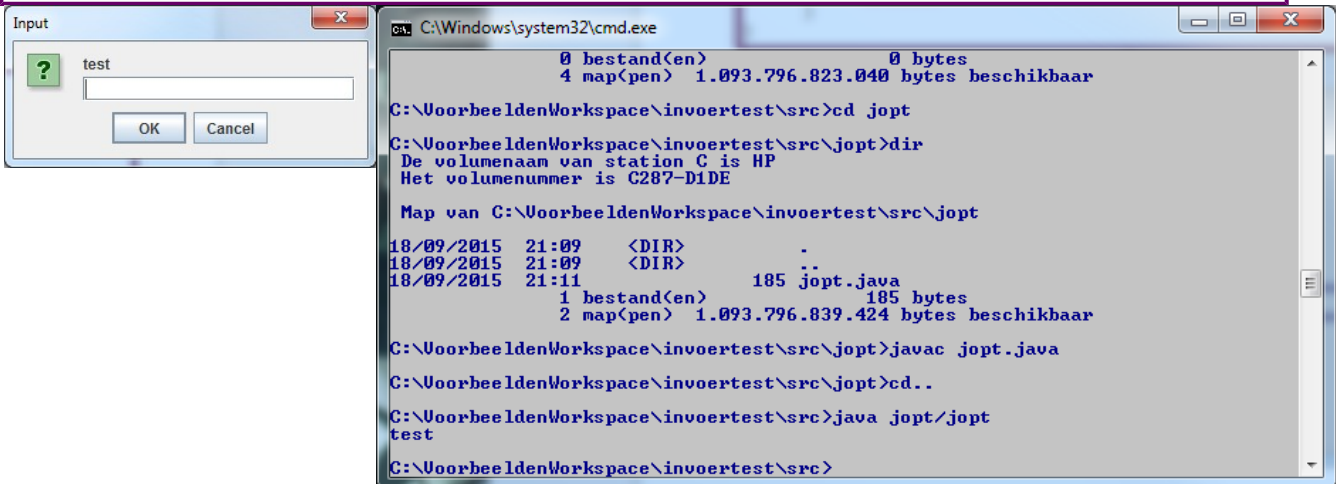
Jopt

```
package jopt;

import javax.swing.JOptionPane; 

public class jopt {

    public static void main(String[] args) {
        System.out.println(JOptionPane.showInputDialog("test"));
    }
}
```

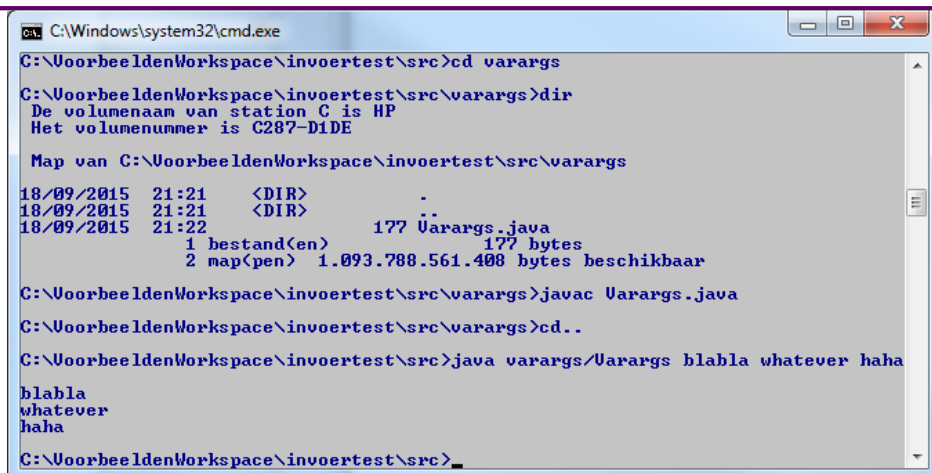


Using varargs

```
package varargs;

public class Varargs {

    public static void main(String[] args) {
        for (int i = 0; i < args.length; i++) {
            System.out.println(args[i]);
        }
    }
}
```



OBP en OOP

int & double

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/primitiveVsReference>

Loops

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/loops>

Switch

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/switchh>

Toegelaten datatypes in een switch:

- Byte, short, char, int, enum, String, Character, Byte, Short and Integer
- Strings kan pas sinds Java 7

Primitive vs Reference

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/primitiveVsReference>

- Twee objecten vergelijken met == vergelijkt adressen, niet de inhoud. Bij primitieve types wordt de inhoud vergeleken
- Primitieve types zijn call by value, anderen zijn call by reference
- Om de inhoud van objecten te vergelijken gebruik je best equals(), overschrijf deze indien nodig

Multiple classes in one file

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/multipleClassesInOneFile>

- Ieder Java bestand moet één public klasse bevatten
- Een publieke klasse MOET in zijn eigen bestand zitten (public class Test in Test.java)
- Ook binnen hetzelfde bestand kunnen klassen niet aan elkaars private velden

Inner class (1/2)

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/innerClass>

- De inner klasse kan aan alle velden en methodes van de outerklasse
- Om van de outer klasse aan de velden van de inner klasse te kunnen moeten we iets extra doen

Inner class (2/2)

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/innerClass2>

- Om via de outer klasse aan de inner klasse te geraken moet de compiler er zeker van zijn dat er ook een instantie is van deze inner klasse

? operator

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/questionmark>

Inheritance

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/inheritance>

- Het aanroepen van een constructor roept steeds de constructors van de superklassen op
- De constructor van de hoogste superklasse wordt eerst aangeroepen, dan de tweede hoogste etc.

Abstract class

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/abstractClass>

Interface

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/interfaces>

- Velden in interfaces worden per definitie static final
- Methodes in een interface mogen geen body bevatten (tenzij default methode)
- Default methodes mogen overschreven worden
- Iedere methode in een interface moet door de implementatie overschreven worden (behalve default methodes)

Anonymous class

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/anonymousClass>

Lambda + functionele interface (1/2)

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/lambdaInterface>

- Functionele interfaces zijn interfaces met slechts één methode
- Functionele interfaces mogen nog steeds constanten bevatten
- @FunctionalInterface is optioneel en wordt gebruikt om compilation errors te vermijden

- Dit kan alleen met functionele interfaces
- Dit kan pas sinds Java 8

- Gebruik () om lambda zonder params te gebruiken

Lambda + functionele interface (2/2)

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/lambdaInterface2>

- Meerdere parameters bij een lambda moet je verzamelen binnen ()
- Bij een enkele parameter is dit niet nodig

Method reference (verkorte lambda)

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/methodReference>

- Sinds Java 8
- De referenced method moet static zijn

Enum

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/enums>

- Enums verbruiken binnen een object het geheugen van een int
- Bij het vergelijken van twee enumerations wordt de index binnen de enum vergeleken

(Un)Boxing

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/boxing>

- Sinds Java 5

Generics

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/generics>

- Sinds Java 5

Builder

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/builder>

Strings

Stringbuilder

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/strings>

- Sinds Java 5

- StringBuffer en StringBuilder zijn mutable
- StringBuffer is synchronized, StringBuilder niet

intern()

<https://github.com/BRNTZN/VOORBEELDEN/tree/master/StandardEdition/src/strings>

```
package strings;

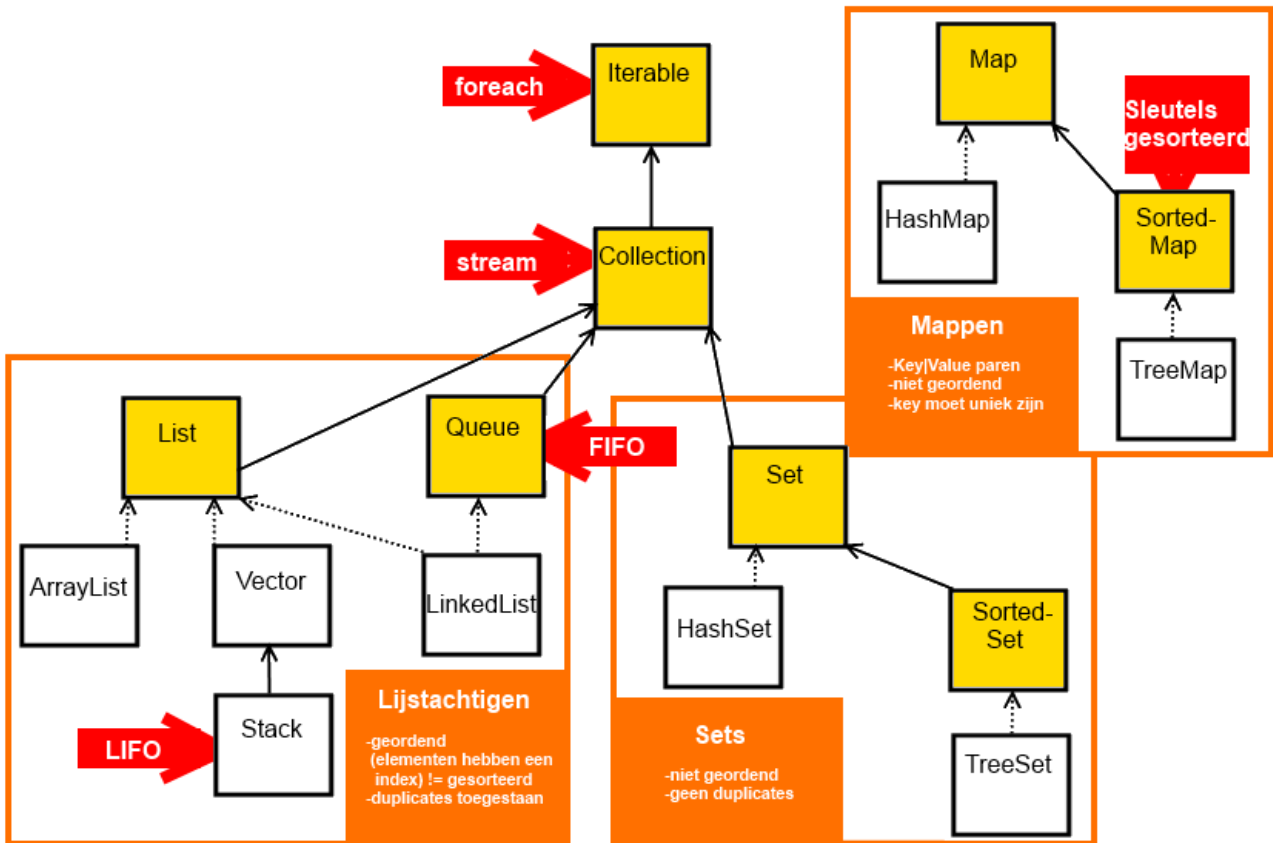
public class Regex {

    public static void main(String[] args) {
        String message = "check";
        if (message.matches(".hec.")) {
            System.out.println("match");
        }
    }
}
```

Regex (regular expressions)

Char	Description
\	Marks the next character as a special character, a literal, a backreference, or an octal escape. For example, 'n' matches the character "n". '\n' matches a newline character. The sequence '\\' matches "\" and \"(\" matches "(".
^	Matches the position at the beginning of the input string. If the RegExp object's Multiline property is set, ^ also matches the position following '\n' or '\r'.
\$	Matches the position at the end of the input string. If the RegExp object's Multiline property is set, \$ also matches the position preceding '\n' or '\r'.
*	Matches the preceding character or subexpression zero or more times. For example, zo* matches "z" and "zoo". * is equivalent to {0,}.
+	Matches the preceding character or subexpression one or more times. For example, 'zo+' matches "zo" and "zoo", but not "z". + is equivalent to {1,}.
?	Matches the preceding character or subexpression zero or one time. For example, "do(es)?" matches the "do" in "do" or "does". ? is equivalent to {0,1}
{n}	<i>n</i> is a nonnegative integer. Matches exactly <i>n</i> times. For example, 'o{2}' does not match the 'o' in "Bob," but matches the two o's in "food".
{n,}	<i>n</i> is a nonnegative integer. Matches at least <i>n</i> times. For example, 'o{2,}' does not match the "o" in "Bob" and matches all the o's in "foooooo". 'o{1,}' is equivalent to 'o+'. 'o{0,}' is equivalent to 'o*'.
{n,m}	<i>M</i> and <i>n</i> are nonnegative integers, where $n \leq m$. Matches at least <i>n</i> and at most <i>m</i> times. For example, "o{1,3}" matches the first three o's in "foooooo". 'o{0,1}' is equivalent to 'o?'. Note that you cannot put a space between the comma and the numbers.
?	When this character immediately follows any of the other quantifiers (*, +, ?, {n}, {n,}, {n,m}), the matching pattern is non-greedy. A non-greedy pattern matches as little of the searched string as possible, whereas the default greedy pattern matches as much of the searched string as possible. For example, in the string "oooo", 'o+?' matches a single "o", while 'o+' matches all 'o's.
.	Matches any single character except "\n". To match any character including the '\n', use a pattern such as '[\s\S]'.
(pattern)	A subexpression that matches <i>pattern</i> and captures the match. The captured match can be retrieved from the resulting Matches collection using the \$0...\$9 properties. To match parentheses characters (), use '\(' or '\)'. To match the backslash character, use '\\'.
(?:pattern)	A subexpression that matches <i>pattern</i> but does not capture the match, that is, it is a non-capturing match that is not stored for possible later use. This is useful for combining parts of a pattern with the "or" character (). For example, 'industr(?:y ies)' is a more economical expression than 'industry industries'.
(?=pattern)	A subexpression that performs a positive lookahead search, which matches the string at any point where a string matching <i>pattern</i> begins. This is a non-capturing match, that is, the match is not captured for possible later use. For example 'Windows (?!95 98 NT 2000)' matches "Windows" in "Windows 2000" but not "Windows" in "Windows 3.1". Lookaheads do not consume characters, that is, after a match occurs, the search for the next match begins immediately following the last match, not after the characters that comprised the lookahead.
(?!pattern)	A subexpression that performs a negative lookahead search, which matches the search string at any point where a string not matching <i>pattern</i> begins. This is a non-capturing match, that is, the match is not captured for possible later use. For example 'Windows (?!95 98 NT 2000)' matches "Windows" in "Windows 3.1" but does not match "Windows" in "Windows 2000". Lookaheads do not consume characters, that is, after a match occurs, the search for the next match begins immediately following the last match, not after the characters that comprised the lookahead.
x y	Matches either x or y. For example, 'z food' matches "z" or "food". '(z f)ood' matches "zood" or "food".
[xyz]	A character set. Matches any one of the enclosed characters. For example, '[abc]' matches the 'a' in "plain".

Collections



<https://github.com/BRNTZN/VOORBEELDEN/tree/master/collections>

- Voor sets wordt de equals() en hashCode() gebruikt om duplicates te vermijden
- Voor sortering wordt de compareTo() gebruikt
- Om een klasse in de enhanced for te gebruiken moet deze iterable<> implementeren, en de elementen moeten comparable implementeren
- Foreach gebruikt kopies, aanpassingen hebben geen effect op het origineel

Exceptions

Unchecked exceptions

- subclasses van RuntimeException
- moeten niet opgevangen worden

Checked exceptions

- subclasses van Exception
- moeten opgevangen worden

Result	<pre> SELECT DISTINCT * [kolom ,kolom] FROM [tabel <i>alias</i> ,tabel] WHERE voorwaarde GROUP BY [kolom ,kolom] HAVING voorwaarde ORDER BY [kolom(nr) {ASC DESC} ,kolom(nr) {ASC DESC}] </pre>
Update	<pre> INSERT INTO <i>tabel</i> (<i>kolom1</i>,...,<i>kolomN</i>) VALUES (<i>waarde1</i>,...,<i>waardenN</i>) </pre>
	Of:
	<pre> INSERT INTO <i>tabel</i> (<i>kolom1</i>,...,<i>kolomN</i>) selectStatement </pre>
	<pre> UPDATE <i>tabel</i> SET kolom1 = expr1,..., kolomN = exprN WHERE voorwaarde </pre>
	<pre> DELETE FROM <i>tabel</i> WHERE voorwaarde </pre>
	<pre> DROP TABLE <i>tabel</i> </pre>
	<pre> CREATE VIEW <i>viewnaam</i> AS selectStatement </pre>

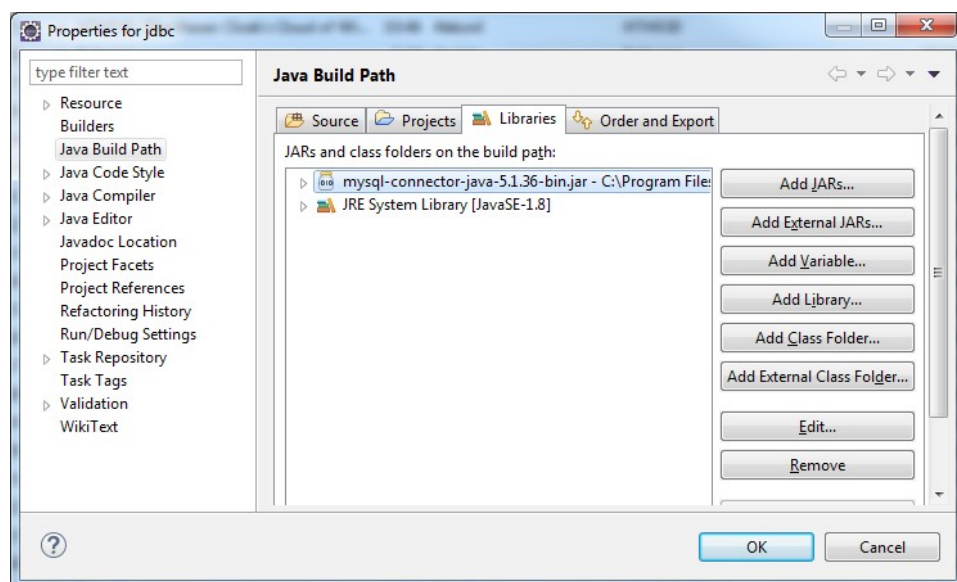
DBMS: database management systeem (zoals MySQL, Oracle Database, Microsoft Access, Microsoft SQL Server, PostgreSQL, ...)

RDBMS: relationeel database management systeem. Iedere rdbms is een DBMS maar niet omgekeerd.

```
1 • create database databank;
2
3 • use databank;
4
5 • create table tabel (id int auto_increment, naam varchar(25), voornaam varchar(25), primary key (id));
6
7 • insert into tabel (naam, voornaam) values ('Courtois', 'Brent');
8
9 • select * from tabel;
```

MySQL

- Download MySQL Installer
(<http://dev.mysql.com/downloads/windows/installer/>)
Je hoeft niet te registreren. Onderaan is een link "No thanks, just start my download"
- Voer afgehaald bestand (bv [mysql-installer-web-community-5.6.25.0.msi](#)) uit
- Vink aan dat je de licentievoorwaarden aanvaardt en laat standaardinstellingen staan.
- Belangrijkste componenten: MySQL Server, MySQL Workbench en Connector/J



Eclipse en MySQL

- In het build path van uw JDBC project moet je de mysql-connector-java jar zetten. Deze heb je net gedownload.

Stored Procedure

Stored procedure: deze zijn sql instructies om op te slaan in je databank. Deze kan je gebruiken voor opdrachten die je niet telkens opnieuw wil uitvoeren. Door deze procedures op de databank te storen verdeel je performantie aan je databank.

Normaal:

```
1 • use databank;
2
3   delimiter //
4 • create procedure procedure_all()
5   begin
6     select * from tabel;
7   end //
8   delimiter ;
```

We gebruiken een tijdelijk nieuwe delimiter (// ipv ;) anders zou de instructie stoppen na "tabel" en zou de instructie dus niet opgeslagen worden.

```
1 • use databank;
2
3 • call procedure_all;
```

Dit roept de procedure op.

Met input:

```
1 • use databank;
2
3 delimiter //
4 • create procedure procedure_in(
5   in inputnaam varchar(50))
6   begin
7     select * from tabel where naam = inputnaam;
8   end//
9 delimiter ;
```

```
1 • use databank;
2
3 • call procedure_in('Courtois');
```

Met output:

```
1 • use databank;
2
3 delimiter //
4 • create procedure procedure_out(
5   out totaal int)
6   begin
7     select count(*) into totaal from tabel;
8   end//
9 delimiter ;
```

```
1 • use databank;
2
3 • call procedure_out(@getal);
4
5 • select @getal;
```

Met input én output:

```
1 • use databank;
2
3 delimiter //
4 • create procedure procedure_inout(
5   in inputnaam varchar(50),
6   out totaal int)
7   begin
8     select count(*) into totaal from tabel where naam = inputnaam;
9   end//
10 delimiter ;
```

```
1 • use databank;
2
3 • call procedure_inout('Turneer', @getal);
4
5 • select @getal;
```

Glassfish

Glassfish

Installatie GlassFish 4.1

- download GlassFish 4.1 (Java EE Full platform)
(<https://glassfish.java.net/download.html>)
- pak glassfish-4.1.zip uit, bijvoorbeeld in map C:\



Installatie GlassFish tools for Eclipse luna

- (in eclipse) Kies Help > Eclipse Marketplace... uit het menu
- tik glassfish in als zoekterm en klik op de Go-knop
- klik op de install-knop van Glassfish Tools for Luna en volg de installatieprocedure

GlassFish 4.1 registreren in eclipse

- (in eclipse) Kies Window > Preferences uit het menu
- in linkerluik: klap Server > Runtime Environments open
- klik op de add-knop en kies GlassFish 4
- in het tekstvak server root vul je de subfolder glassfish van de folder waar GlassFish geïnstalleerd is
(bv [C:\glassfish-4.1\glassfish](#))

Enterprise

JPA (hibernate) + jdbc(MySQL, glassfish) + jsf

Set up

Hibernate



- 1) Download hibernate
→ <http://hibernate.org/orm/downloads/>
- 2) Pack zip uit
- 3) Ga naar:
→ <C:\\--waar je zip hebt uitgepakt--\\hibernate-release-x.x.x\\lib\\required>
- 4) Kopieer al deze jars naar:
→ <C:\\--waar glassfish staat--\\glassfish-x.x\\glassfish\\lib>
- 5) Kopieer `jboss-logging-3.1.x.GA.jar` daarnaast ook nog eens naar:
→ <C:\\--waar glassfish staat--\\glassfish-x.x\\glassfish\\lib\\endorsed>
(dit is voor hybernate-logging)

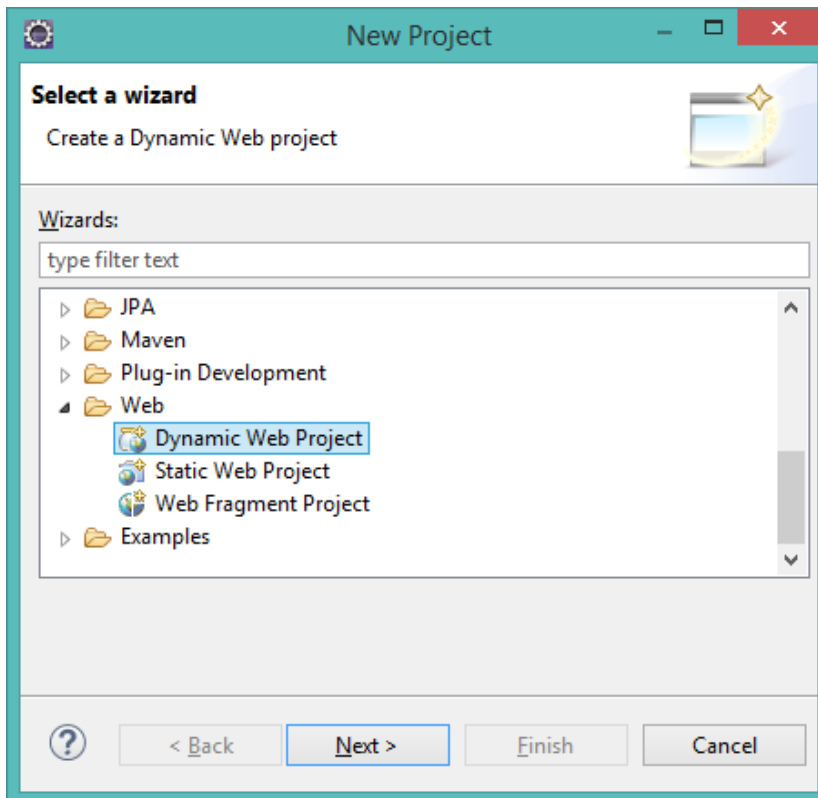
Glassfish



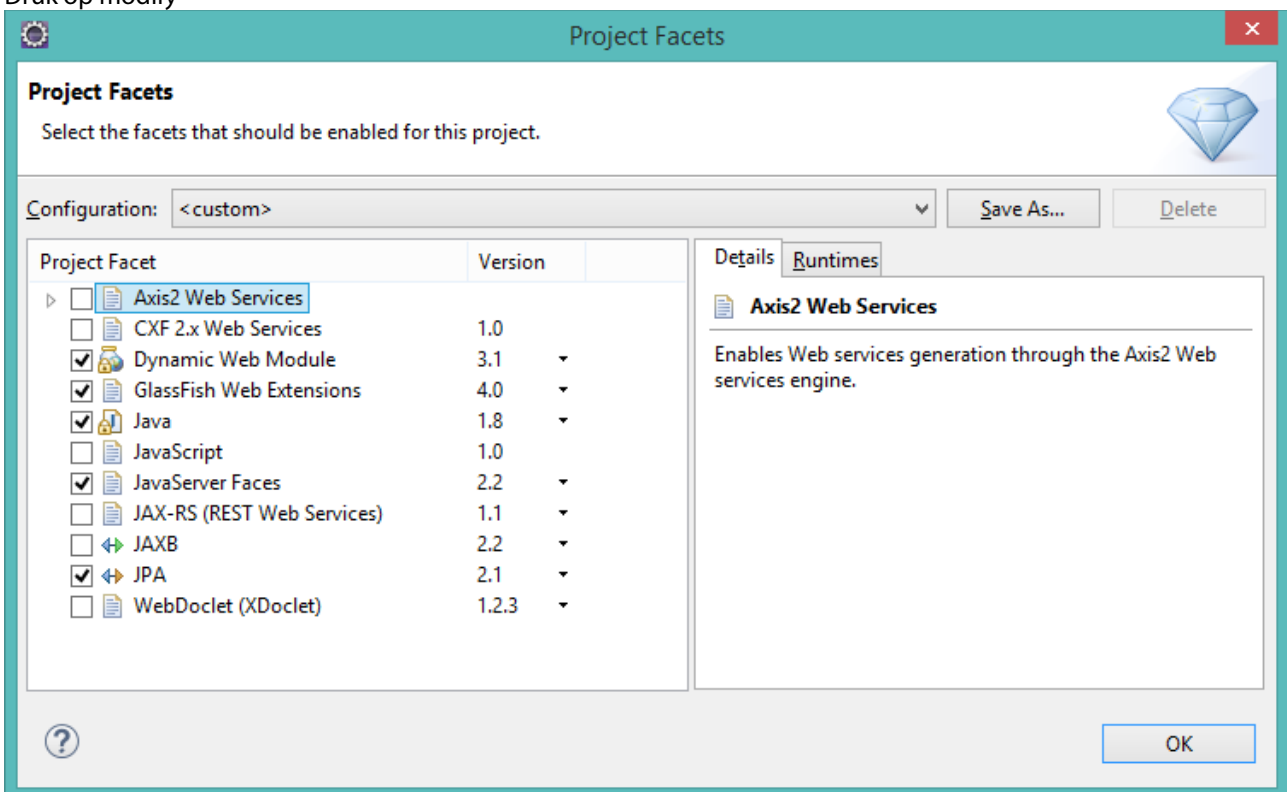
- 6) Ga naar je glassfish admin console (start glassfish eerst wel op):
in browser
→ <http://localhost:4848/common/index.jsf>
in eclipse
→ zorg dat je in java EE perspectief zit
→ klik op servers
→ rechtermuisklik op glassfish
→ klik dan op glassfish (in het menuutje)
→ klik op view admin console
- 7) Links in admin console: resources
→ jdbc
→ connection pools
- 8) New
- 9) Kies een naam (bv: `pool`)
- 10) Resource type: `javax.sql.ConnectionPoolDataSource`
- 11) Database vendor: MySQL
- 12) Next
- 13) Volgende velden minstens:
DatabaseName: (bv: `datablabla`)
Password: (bv: `root`)
URL: `jdbc:mysql://localhost:3306/datablabla`
Url: `jdbc:mysql://localhost:3306/datablabla`
Servername: `localhost`
User: (bv: `root`)
- 14) Finish
- 15) Klik op de connection pool die je net gecreëerd hebt. Je wordt dan naar de 'Edit Connection Pool' pagina gebracht.
- 16) Druk op de knop 'Ping' (niet het aankruisvak). Indien 'Ping Succeeded'
- 17) Ga naar 'JDBC Resources'
- 18) New
- 19) Volgende velden minstens:
JNDI-Name: (bv: `jdbc/_project`)
Pool-Name: kies de pool die je gemaakt hebt (bv: `pool`)
Status Enabled: aangekruisd
- 20) Ok
- 21) Herstart glassfish

Creatie project

Maak een new dynamic web project.



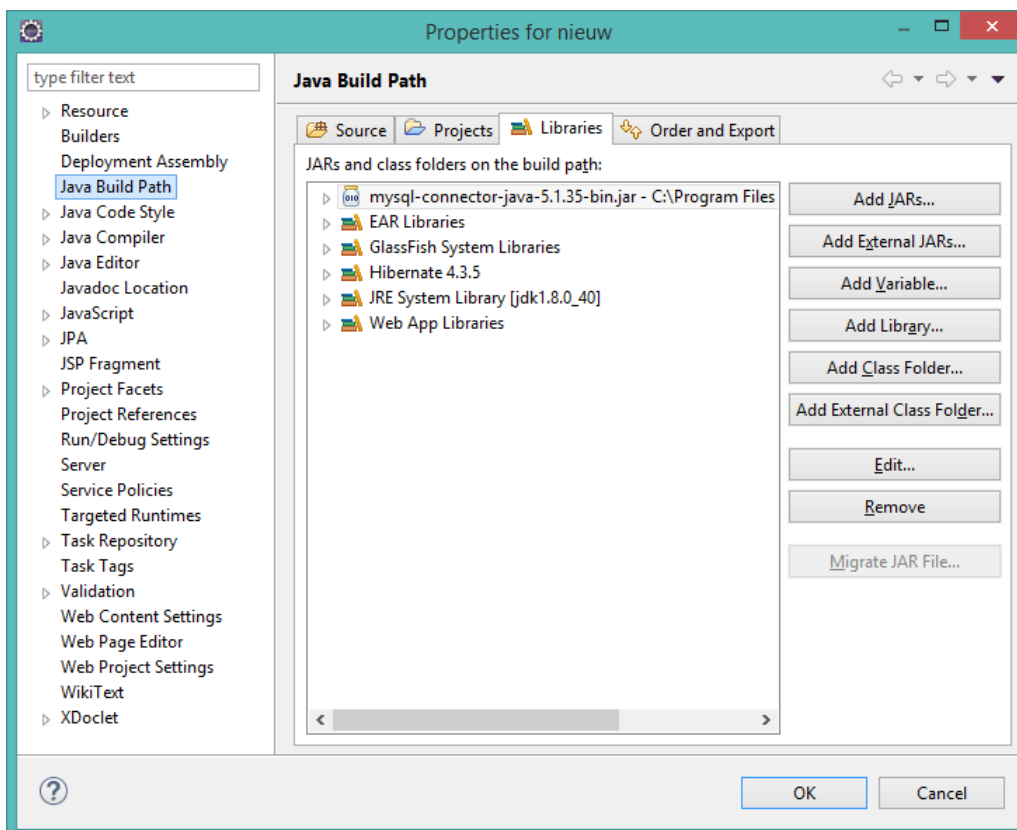
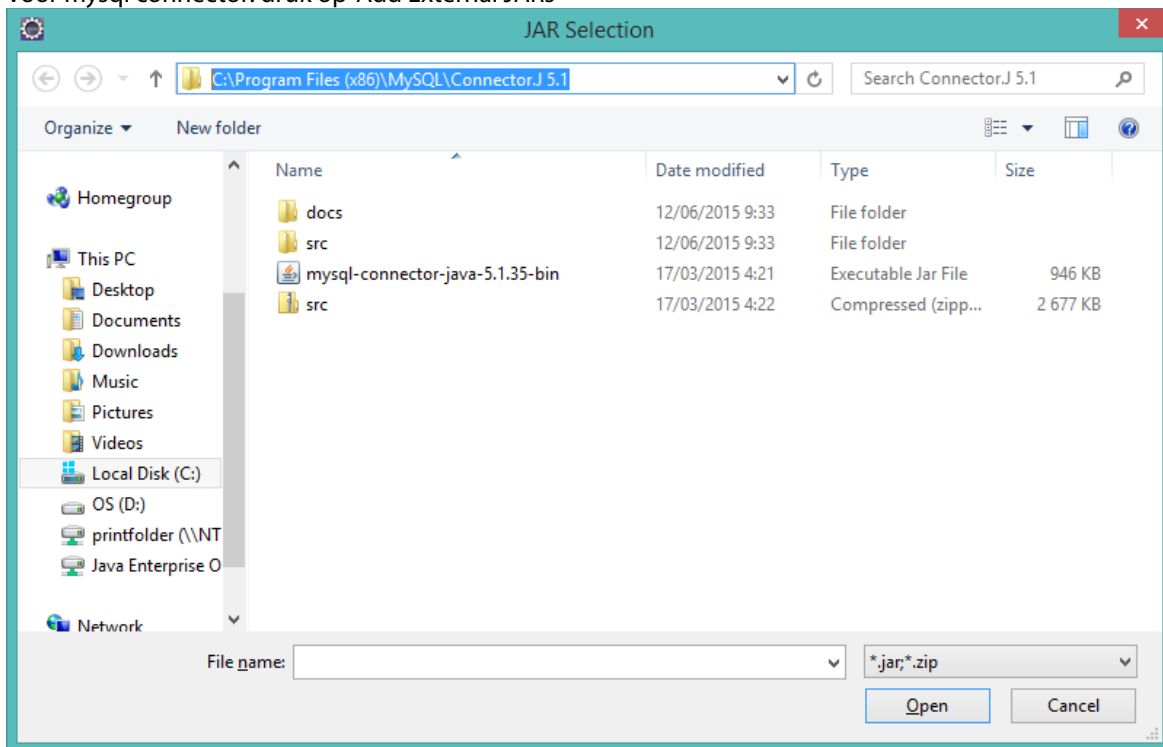
Druk op modify



finish

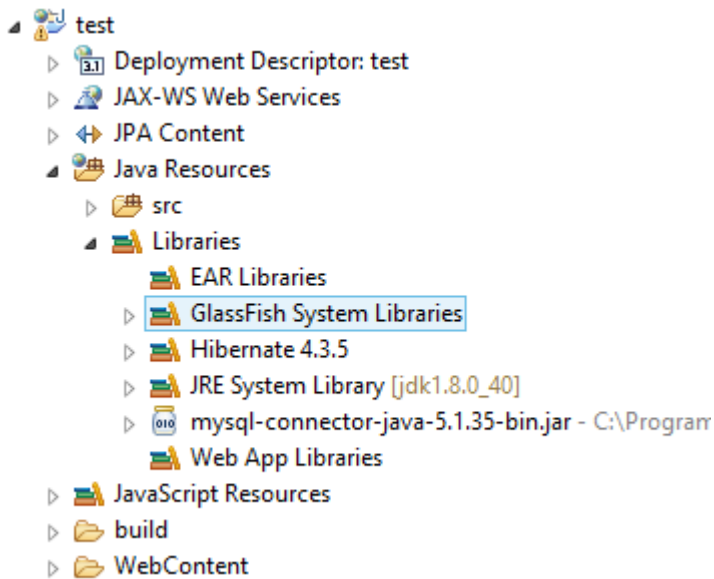
Build path

Voor mysql connector: druk op 'Add External JARs'



Voor Hibernate versie x.x.x:

- druk op 'Add Library'
- user library
- selecteer hibernate versie



persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1"
  xmlns="http://xmlns.jcp.org/xml/ns/persistence"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
  <persistence-unit name="smurf" transaction-type="JTA">
    <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>
    <jta-data-source>jdbc/_project</jta-data-source>
    <class>package.class</class>
    <!-- andere klassen -->
    <properties>
      <property name="hibernate.transaction.jta.platform"
        value="org.hibernate.service.jta.platform.internal.SunOneJtaPlatform" />
      <!-- andere properties hier definiëren -->
      <!-- geen properties definiëren voor password, user, driver etc;
        want we gebruiken datasource -->
      <property name="hibernate.hbm2ddl.auto" value="create" />
      <property name="hibernate.connection.autocommit" value="true" />
    </properties>
  </persistence-unit>
</persistence>
```

view.xhtml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets">
<h:head>
  <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
  <title>paginatitel</title>
</h:head>
<h:body>
<h:form>
  <h:dataTable value="#{backingBean.lijst}" var="tmp">
    <f:facet name="caption">tabeltitel</f:facet>
    <h:column>
      <h:commandButton value="verwijder"
        action="#{backingBean.verwijder(tmp.id)}"></h:commandButton>
      <h:commandButton value="update"
        action="#{backingBean.goUpdate(tmp.id)}"></h:commandButton>
    </h:column>
    <h:column>
      <f:facet name="header">kolomtitel: id</f:facet>
      #{tmp.id}
    </h:column>
    <h:column>
      <f:facet name="header">Naam</f:facet>
      #{tmp.naam}
      <f:facet name="footer">Aantal klanten:
        #{backingBean.lijst.size()}</f:facet>
    </h:column>
  </h:dataTable>
  <h:commandButton value="terug" action="paginanaamzonderextensie">
    </h:commandButton>
</h:form>
<br/>
<ul>
  <ui:repeat var="tmp" value="#{backingBean.lijst}">
    <li>#{tmp.id} #{tmp.naam}</li>
  </ui:repeat>
</ul>
</h:body>
</html>
```

BackingBean

```
@Named
@SessionScoped
public class BackingBean implements Serializable {

    /**
     *
     */
    private static final long serialVersionUID = 1L;

    @PersistenceContext(unitName = "smurf")
    private EntityManager em;
    private List<Object> lijst = new ArrayList<>();

    @Transactional
    public void makeObject() {
        Object tmp = new Object();
        em.persist(tmp);
    }

    @Transactional
    public String verwijder(int id) {
        em.remove(em.find(Object.class, id));
        return null;
    }

    public String overzicht() {
        TypedQuery<Object> q = em.createQuery(
            "SELECT r FROM Object r", Object.class);
        setLijst(q.getResultList());
        return "paginanaam";
    }
}
```

Enterprise Javabeans (EJB) 3.1

titel

blabla

//TODO

generic method
final classes
abstract class
maven
hybernate
spring