# Java Programming Cheat Sheet

by sally sung via cheatography.com/216190/cs/47965/

## Inheritance

```
class Animal {
void eat() {
System.out.println("eat");
}
}
class Dog extends Animal {
void bark() {
System.out.println("bark");
}
}
```

## Interface

```
interface Animal {
void sound();
}
class Dog implements Animal {
public void sound() {
System.out.println("bark");
}
}
```

## Abstract

```
abstract class Animal {
abstract void sound();
void eat() {
System.out.println("eat");
}
}
class Dog extends Animal {
void sound() {
System.out.println("bark");
}
}
```

## Exception

```
public class UsedCarException extends
Exception {
private static final long serialVersionUID =
1L;
public UsedCarException(String message) {
super(message);
}
}
```

## Working with Files

```
public class UsedCarApp {
public static void writeToFile(ArrayList<Us-
edCar> cars, String fileName) {
try {
PrintWriter writer = new PrintWriter(new
FileWriter(fileName));
for (UsedCar car : cars) {
writer.println(car.toString());
}
writer.close();
System.out.println("\nSuccessfully saved
valid cars to file.");
} catch (IOException e) {
System.out.println("Error writing to file: " +
e.getMessage());
}
}
public static void readFromFile(String
fileName) {
try {
Scanner fileInput = new Scanner(new File(f-
ileName));
System.out.println("\nVIN and Price from
file:");
while (fileInput.hasNextLine()) {
String line = fileInput.nextLine();
String[] parts = line.split(",");
if (parts.length == 4) {
String vin = parts[0];
String price = parts[3];
System.out.println("VIN: " + vin + " | Price: "
+ price);
}
}
fileInput.close();
} catch (FileNotFoundException e) {
System.out.println("Error reading file: " +
e.getMessage());
}
}
public static void main(String[] args) {
Scanner input = new Scanner(System.in);
ArrayList<UsedCar> validCars = new
ArrayList<>();
```

## Working with Files (cont)

```
final int NUMBER_OF_CARS = 3;
for (int i = 0; i < NUMBER_OF_CARS; i++) {
try {
System.out.println("\nEnter information for
car #" + (i + 1));
System.out.print("Enter VIN (4 digits): ");
String vin = input.nextLine();
System.out.print("Enter make: ");
String make = input.nextLine();
System.out.print("Enter year: ");
int year = Integer.parseInt(input.nextLine());
System.out.print("Enter price: ");
double price = Double.parseDouble(in-
put.nextLine());
UsedCar car = new UsedCar(vin, make,
year, price);
validCars.add(car);
System.out.println("Car created successfu-
lly.");
} catch (UsedCarException e) {
System.out.println("UsedCarException: " +
e.getMessage());
} catch (Exception e) {
System.out.println("Exception: Invalid input.
" + e.getMessage());
}
}
String fileName = "UsedCars.txt";
writeToFile(validCars, fileName);
readFromFile(fileName);
input.close();
}
}
```

## JDBC

```
module-info.java >
module FinalTermPractice {
requires java.sql;
}
```


By **sally sung**

cheatography.com/sally-sung/



Published 16th April, 2026.

Last updated 16th April, 2026.





Sponsored by **ApolloPad.com**

Everyone has a novel in them. Finish Yours!

https://apollopad.com

## JDBC

```
import java.sql.*;
public class EmployeeDBApp {
private static final String DB_FILE = "Emp-
loyeeDB.accdb";
private static final String DB_URL = "jdbc:u-
canaccess://" + DB_FILE;
public static Connection getConnection()
throws SQLException, ClassNotFoundEx-
ception {
Class.forName("net.ucanaccess.jdbc.Uc-
anaccessDriver");
return DriverManager.getConnection(DB-
_URL);
}
public static void insertEmployee(Scanner
input) {
Connection conn = null;
Statement statement = null;
try {
conn = getConnection();
statement = conn.createStatement();
System.out.print("Enter employee name: ");
String name = input.nextLine();
System.out.print("Enter employee salary: ");
double salary = Double.parseDouble(in-
put.nextLine());
String insertQ = "INSERT INTO Employee
(Name, Salary) VALUES ('"
+ name + "', " + salary + ")";
statement.executeUpdate(insertQ);
System.out.println("Employee inserted
successfully.");
} catch (ClassNotFoundException ex) {
System.out.println("Driver error: " + ex.get-
Message());
} catch (SQLException ex) {
System.out.println("Database error: " +
ex.getMessage());
} catch (Exception ex) {
System.out.println("Input error: " + ex.get-
Message());
} finally {
try {
if (statement != null)
```

## JDBC (cont)

```
statement.close();
if (conn != null)
conn.close();
} catch (SQLException ex) {
System.out.println(ex.getMessage());
}
}
}
public static void updateEmployeeSalary(-
Scanner input) {
Connection conn = null;
Statement statement = null;
try {
conn = getConnection();
statement = conn.createStatement();
System.out.print("Enter employee name to
update: ");
String name = input.nextLine();
System.out.print("Enter new salary: ");
double salary = Double.parseDouble(in-
put.nextLine());
String updateQ = "UPDATE Employee SET
Salary = " + salary
+ " WHERE Name = '" + name + "'";
int rows = statement.executeUpdate(upd-
ateQ);
if (rows > 0) {
System.out.println("Salary updated succes-
sfully.");
} else {
System.out.println("Employee not found.");
}
} catch (ClassNotFoundException ex) {
System.out.println("Driver error: " + ex.get-
Message());
} catch (SQLException ex) {
System.out.println("Database error: " +
ex.getMessage());
} catch (Exception ex) {
System.out.println("Input error: " + ex.get-
Message());
} finally {
```

## JDBC (cont)

```
try {
if (statement != null)
statement.close();
if (conn != null)
conn.close();
} catch (SQLException ex) {
System.out.println(ex.getMessage());
}
}
}
public static void deleteEmployee(Scanner
input) {
Connection conn = null;
Statement statement = null;
try {
conn = getConnection();
statement = conn.createStatement();
System.out.print("Enter employee name to
delete: ");
String name = input.nextLine();
String deleteQ = "DELETE FROM
Employee WHERE Name = '" + name + "'";
int rows = statement.executeUpdate(del-
eteQ);
if (rows > 0) {
System.out.println("Employee deleted
successfully.");
} else {
System.out.println("Employee not found.");
}
} catch (ClassNotFoundException ex) {
System.out.println("Driver error: " + ex.get-
Message());
} catch (SQLException ex) {
System.out.println("Database error: " +
ex.getMessage());
} catch (Exception ex) {
System.out.println("Input error: " + ex.get-
Message());
} finally {
try {
if (statement != null)
statement.close();
```



Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com

## JDBC (cont)

```
if (conn != null)
conn.close();
} catch (SQLException ex) {
System.out.println(ex.getMessage());
}
}
}
public static void displayAllEmployees() {
Connection conn = null;
Statement statement = null;
ResultSet rs = null;
ArrayList<Employee> employees = new
ArrayList<Employee>();
try {
conn = getConnection();
statement = conn.createStatement();
String query = "SELECT * FROM Employ-
ee";
rs = statement.executeQuery(query);
while (rs.next()) {
int id = rs.getInt("ID");
String name = rs.getString("Name");
double salary = rs.getDouble("Salary");
Employee emp = new Employee(id, name,
salary);
employees.add(emp);
}
if (employees.size() == 0) {
System.out.println("No employee records
found.");
} else {
System.out.println("\n--- All Employees ---");
for (Employee emp : employees) {
System.out.println("ID: " + emp.getId()
+ ", Name: " + emp.getName()
+ ", Salary: " + emp.getSalary());
}
}
} catch (ClassNotFoundException ex) {
System.out.println("Driver error: " + ex.get-
Message());
} catch (SQLException ex) {
System.out.println("Database error: " +
ex.getMessage());
} finally {
```

## JDBC (cont)

```
try {
if (rs != null)
rs.close();
if (statement != null)
statement.close();
if (conn != null)
conn.close();
} catch (SQLException ex) {
System.out.println(ex.getMessage());
}
}
}
public static void showMenu() {
System.out.println("\n===== Employee
Database Menu =====");
System.out.println("1. Insert Employee");
System.out.println("2. Update Employee
Salary");
System.out.println("3. Delete Employee");
System.out.println("4. Display All Employ-
ees");
System.out.println("5. Exit");
System.out.print("Enter your choice: ");
}
public static void main(String[] args) {
Scanner input = new Scanner(System.in);
int choice = 0;
do {
showMenu();
try {
choice = Integer.parseInt(input.nextLine());
switch (choice) {
case 1:
insertEmployee(input);
break;
case 2:
updateEmployeeSalary(input);
break;
case 3:
deleteEmployee(input);
break;
case 4:
displayAllEmployees();
```

## JDBC (cont)

```
break;
case 5:
System.out.println("Program terminated.");
break;
default:
System.out.println("Invalid choice. Please
enter a number from 1 to 5.");
}
} catch (Exception ex) {
System.out.println("Invalid input. Please
enter a valid number.");
}
} while (choice != 5);
input.close();
}
}
```



Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com