UNSW
THE UNIVERSITY OF NEW SOUTH WALES

Component Programming

Never Stand Still | Faculty of Engineering | CSE

Weisi Chen

chenw@cse.unsw.edu.au

Components are like black boxes

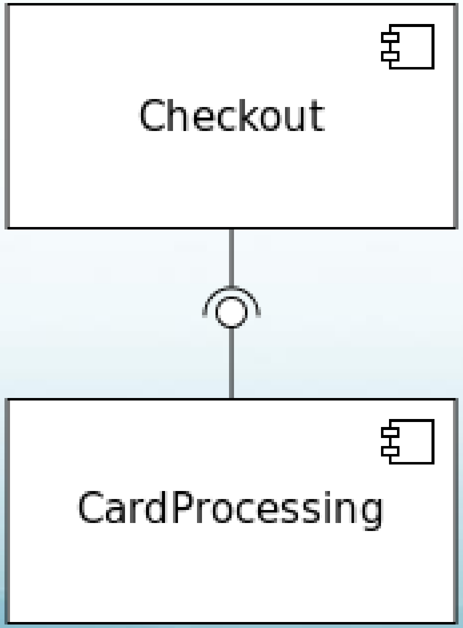
- The programmer ***knows***:
  - how the outside looks like
  - what the component can provide
- The programmer ***does NOT know***:
  - how it works internally

Characteristics

- Self-contained
- Explicit dependencies
- Well-defined **interface**

Examples

- A small interest calculator plug-in
- An interface to a database manager
- A web service
- Paypal:



Ebay checkout app

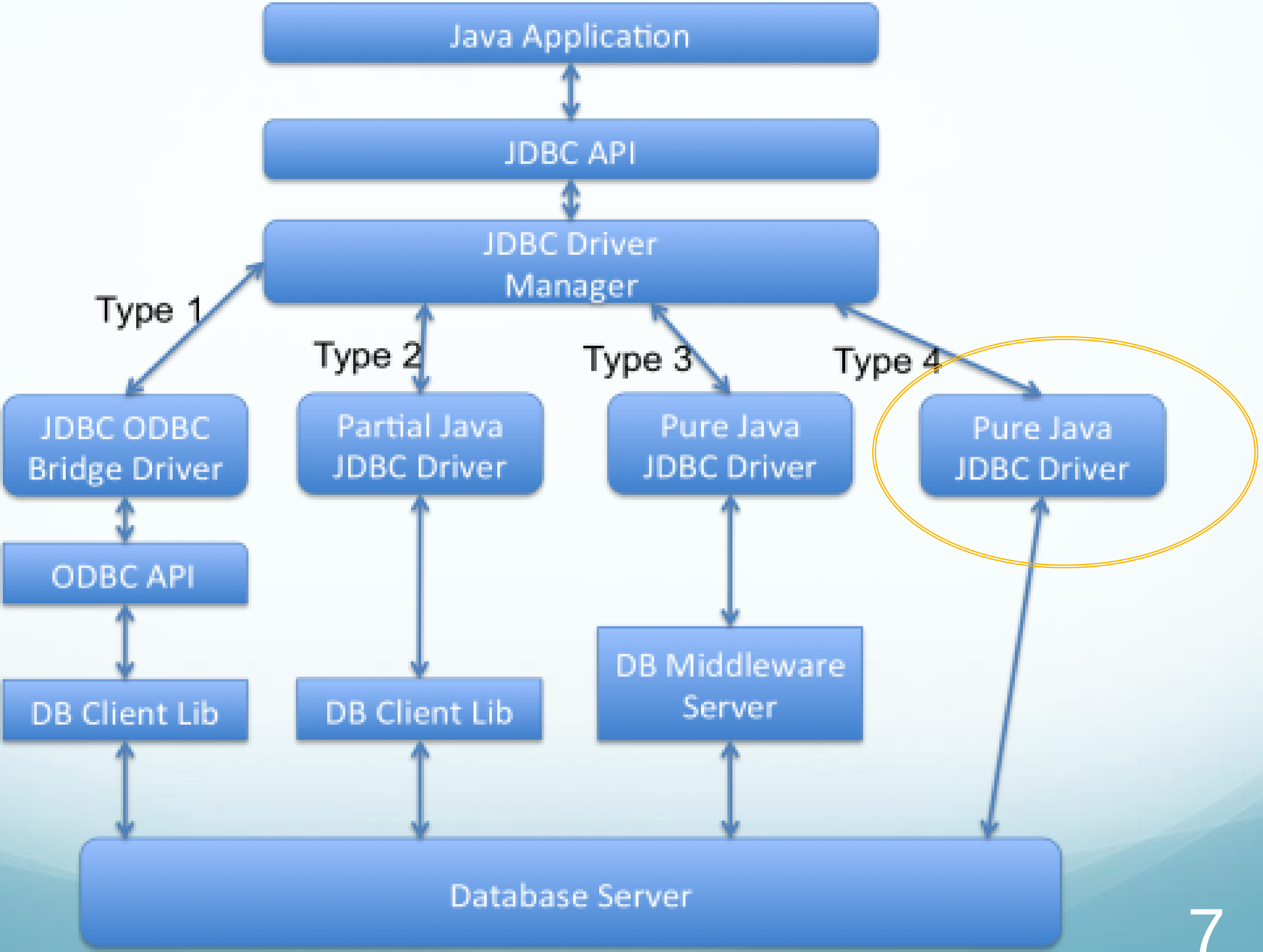
Paypal service

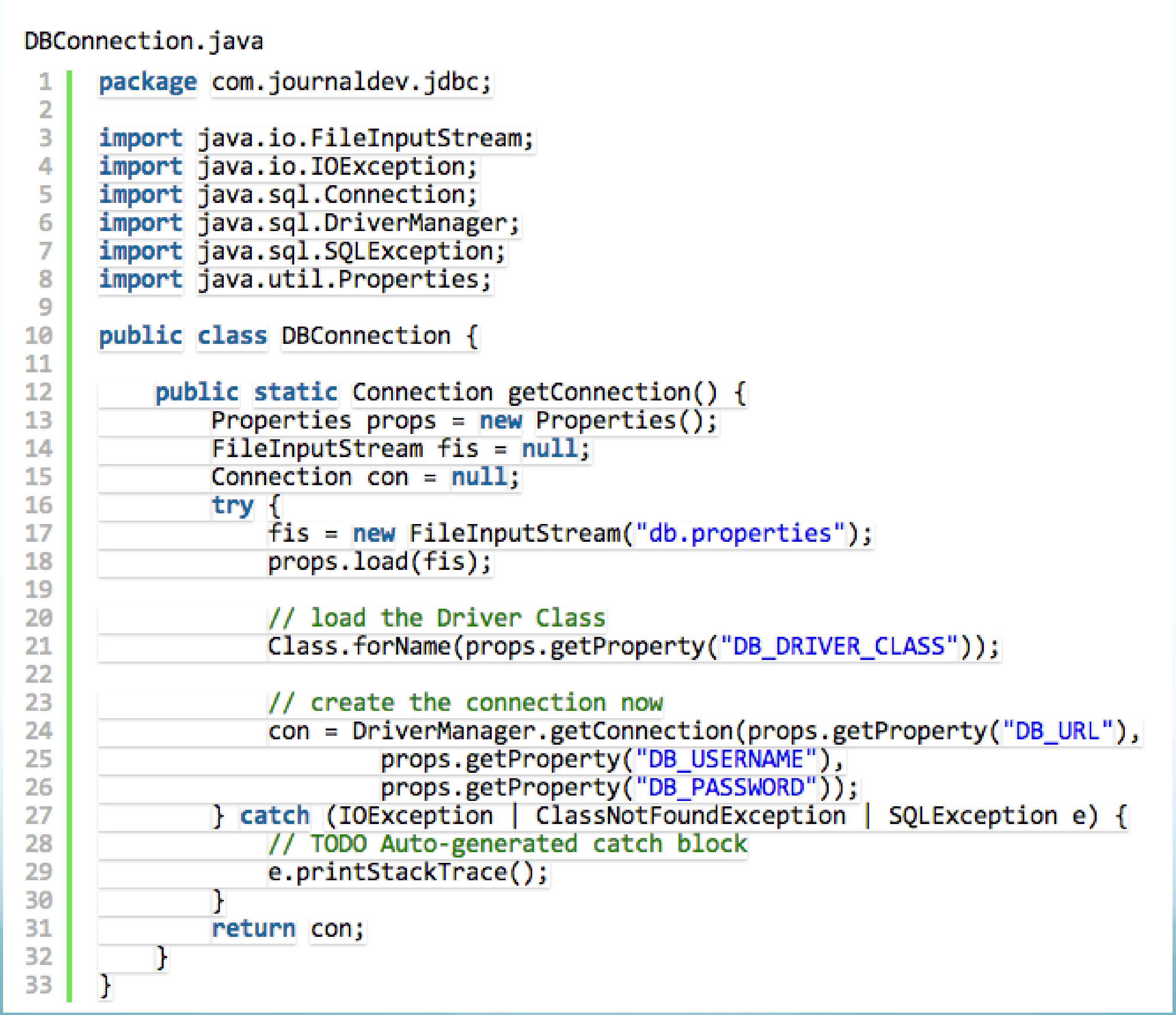
Benefits?

- For application builder
  - Able to replace the component by another one (with the same **interface** and functionality)
  - No need to understand the inner working, but only the **interface** of the component
- For component provider
  - Able to change the implementation of the component as long as the **interface** is still satisfied
  - Can be reused by various app builder
- Potentially – prevent fraud ☺

Example – JDBC Database Driver

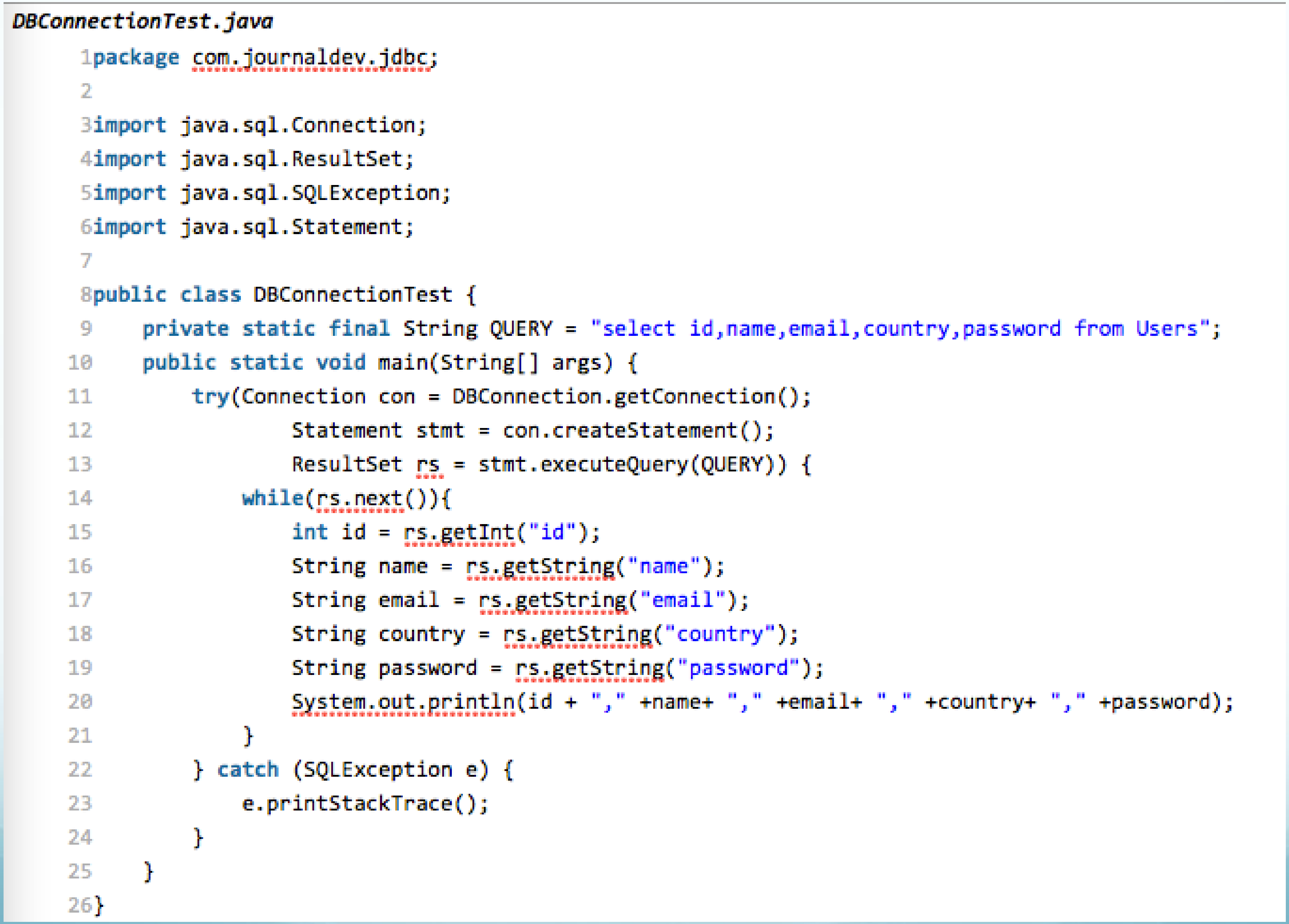
- JDBC drivers implement the defined interfaces in the JDBC API for interacting with your database server.
- JDBC drivers handle internally:
  - which database driver you code against
  - the details of the database communications


Java Application
JDBC API
JDBC Driver Manager
Type 1
Type 2
Type 3
Type 4
JDBC ODBC Bridge Driver
Partial Java JDBC Driver
Pure Java JDBC Driver
Pure Java JDBC Driver
ODBC API
DB Middleware Server
DB Client Lib
DB Client Lib
Database Server



DBConnection.java

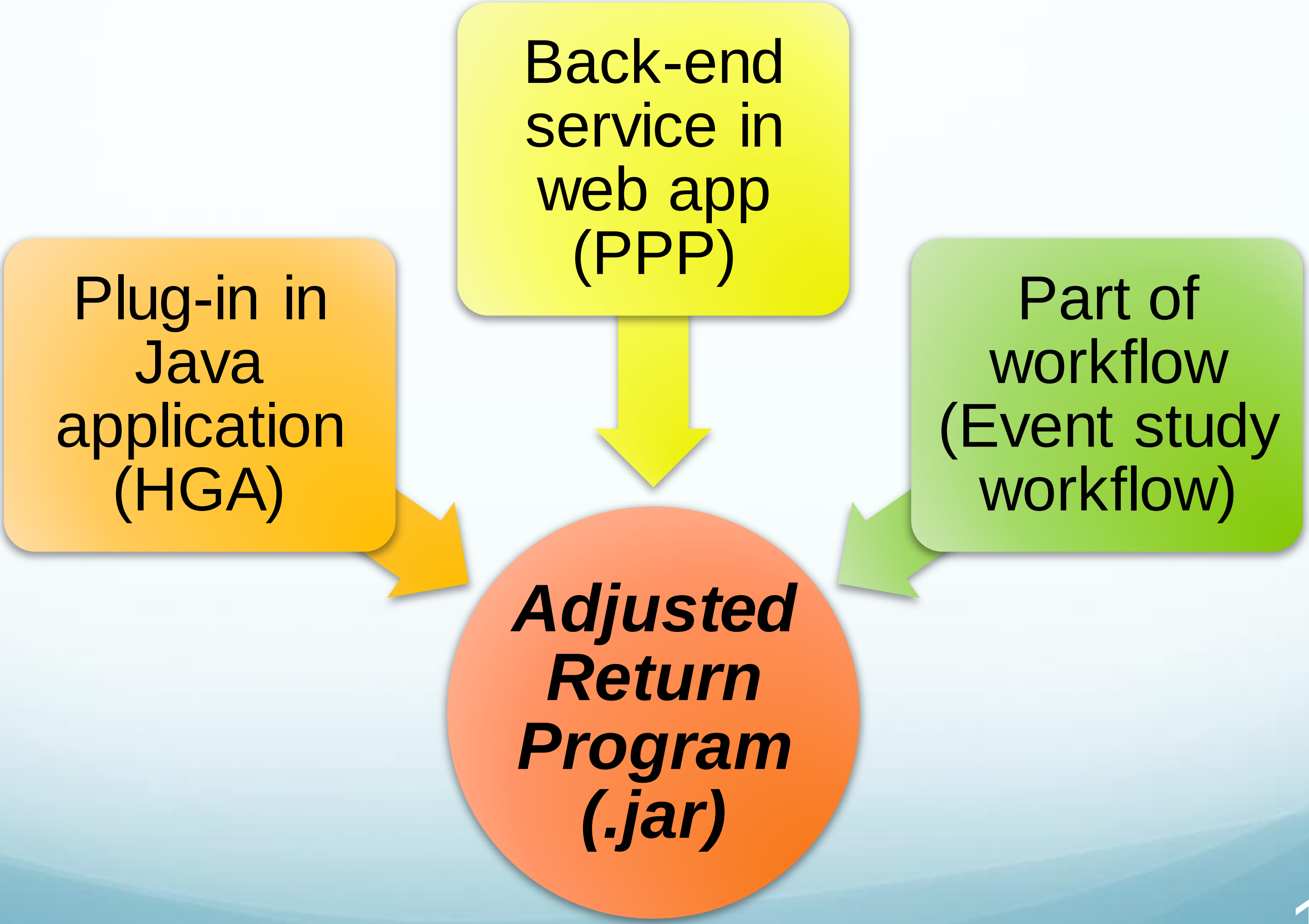
```java
package com.journaldev.jdbc;

import java.io.FileInputStream;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

public class DBConnection {

    public static Connection getConnection() {
        Properties props = new Properties();
        FileInputStream fis = null;
        Connection con = null;
        try {
            fis = new FileInputStream("db.properties");
            props.load(fis);

            // load the Driver Class
            Class.forName(props.getProperty("DB_DRIVER_CLASS"));

            // create the connection now
            con = DriverManager.getConnection(props.getProperty("DB_URL"),
                    props.getProperty("DB_USERNAME"),
                    props.getProperty("DB_PASSWORD"));
        } catch (IOException | ClassNotFoundException | SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return con;
    }
}
```



DBConnectionTest.java

```java
package com.journaldev.jdbc;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class DBConnectionTest {
    private static final String QUERY = "select id,name,email,country,password from Users";
    public static void main(String[] args) {
        try(Connection con = DBConnection.getConnection();
                Statement stmt = con.createStatement();
                ResultSet rs = stmt.executeQuery(QUERY)) {
            while(rs.next()){
                int id = rs.getInt("id");
                String name = rs.getString("name");
                String email = rs.getString("email");
                String country = rs.getString("country");
                String password = rs.getString("password");
                System.out.println(id + "," +name+ "," +email+ "," +country+ "," +password);
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

Java component



Back-end service in web app (PPP)

Plug-in in Java application (HGA)

Part of workflow (Event study workflow)

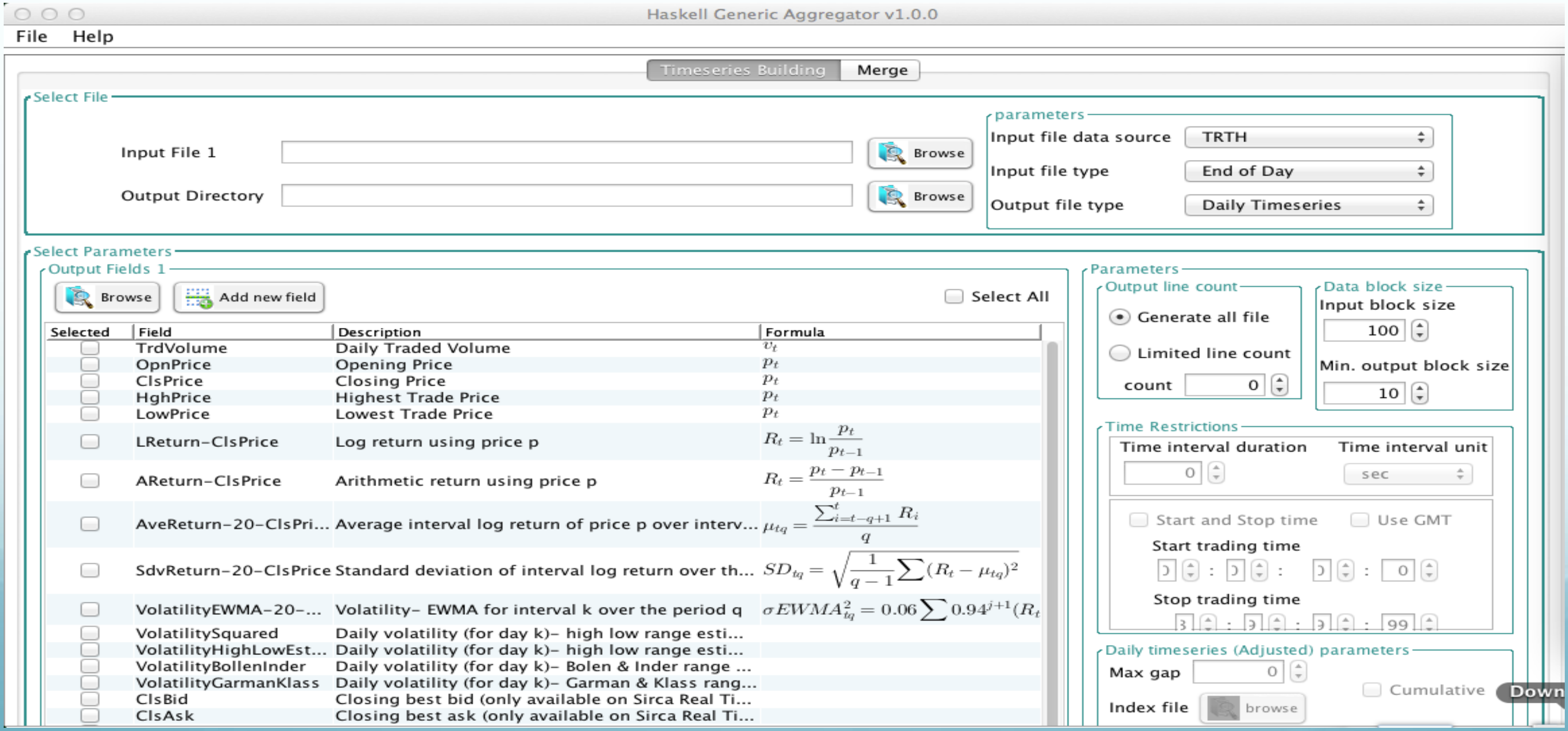
Adjusted Return Program (.jar)

How to Generate **.jar** File

- Export from IDE (e.g. Eclipse)
- Use command line:
  - jar cf jar-file input-file(s)
- Use popular build tools:
  - Maven
  - Ant
  - Buildr
  - … …

1. Java Application (HGA)

- Include the .jar into the build path
- Input: SysVar; Output: results + log + error message



Select File

Input File 1

Output Directory

parameters

Input file data source: TRTH

Input file type: End of Day

Output file type: Daily Timeseries

Select Parameters

Output Fields 1

□ Select All

Selected	Field	Description	Formula
□	TrdVolume	Daily Traded Volume	v_t
□	OpnPrice	Opening Price	p_t
□	ClsPrice	Closing Price	p_t
□	HghPrice	Highest Trade Price	p_t
□	LowPrice	Lowest Trade Price	p_t
□	LReturn-ClsPrice	Log return using price p	$R_t = \ln \frac{p_t}{p_{t-1}}$
□	AReturn-ClsPrice	Arithmetic return using price p	$R_t = \frac{p_t - p_{t-1}}{p_{t-1}}$
□	AveReturn-20-ClsPri...	Average interval log return of price p over interv...	$\mu_{tq} = \frac{\sum_{i=t-q+1}^{t} R_i}{q}$
□	SdvReturn-20-ClsPrice	Standard deviation of interval log return over th...	$SD_{tq} = \sqrt{\frac{1}{q-1}\sum(R_t - \mu_{tq})^2}$
□	VolatilityEWMA-20-...	Volatility- EWMA for interval k over the period q	$\sigma EWMA_{tq}^2 = 0.06\sum 0.94^{j+1}(R_t$
□	VolatilitySquared	Daily volatility (for day k)- high low range esti...	
□	VolatilityHighLowEst...	Daily volatility (for day k)- high low range esti...	
□	VolatilityBollenInder	Daily volatility (for day k)- Bolen & Inder range ...	
□	VolatilityGarmanKlass	Daily volatility (for day k)- Garman & Klass rang...	
□	ClsBid	Closing best bid (only available on Sirca Real Ti...	
□	ClsAsk	Closing best ask (only available on Sirca Real Ti...	

Parameters

Output line count

● Generate all file

○ Limited line count

count 0

Data block size

Input block size 100

Min. output block size 10

Time Restrictions

Time interval duration 0

Time interval unit sec

□ Start and Stop time □ Use GMT

Start trading time

0 : 0 : 0 : 0

Stop trading time

3 : 9 : 9 : 99

Daily timeseries (Adjusted) parameters

Max gap 0

□ Cumulative

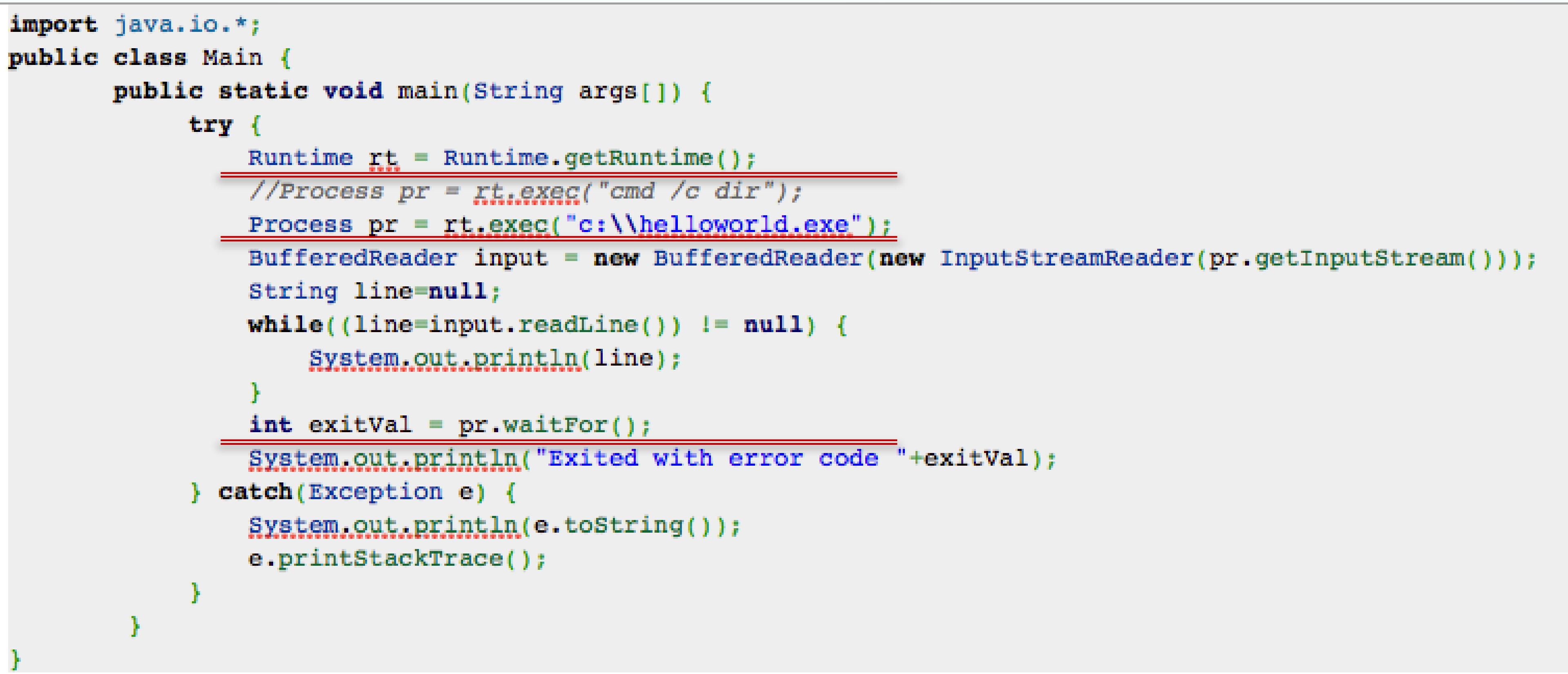
Index file

2. Back-end Service in Web App

- Put .jar into: “tomcat6/webapps/axis2/WEB-INF/services/“
- (Re)Start tomcat
- The link to the service interface (WSDL) will be available at:
  - http://HOST:PORT/axis2/services/listServices
  - E.g.: http://adage.cse.unsw.edu.au:8080/axis2/services/listServices
- Web app: http://adage.cse.unsw.edu.au:8080/ppp/

Call Dos/Linux commands from Java

- ***Process*** or ***ProcessBuilder***



```
import java.io.*;
public class Main {
        public static void main(String args[]) {
             try {
                Runtime rt = Runtime.getRuntime();
                //Process pr = rt.exec("cmd /c dir");
                Process pr = rt.exec("c:\\helloworld.exe");
                BufferedReader input = new BufferedReader(new InputStreamReader(pr.getInputStream()));
                String line=null;
                while((line=input.readLine()) != null) {
                    System.out.println(line);
                }
                int exitVal = pr.waitFor();
                System.out.println("Exited with error code "+exitVal);
             } catch(Exception e) {
                System.out.println(e.toString());
                e.printStackTrace();
             }
         }
}
```

Call Haskell from Java

- Method 1: Deploy a Haskell program as a "service" and call its functions when needed.
- Method 2: Haskell → C → Java

```
import com.googlecode.javacpp.*;
import com.googlecode.javacpp.annotation.*;
@Platform
public class Main {
    public static class Fibonacci extends FunctionPointer {
        public @Name("fibonacci") int call(int n) {
            return n < 2 ? n : call(n - 1) + call(n - 2);
        }
    }
}
```