Programming in Java - Packaging

Paolo Vercesi
Technical Program Manager

© 2019 ESTECO SpA

Agenda

© 2019 ESTECO SpA

Packages

© 2019 ESTECO SpA

Packages

esteco.Television

```
package esteco;

class Television {
    String model;
    boolean on;
    int channel;
    int volume;

    Television(String model) {
        this.model = model;
    }
}
```

units.Television

```
package units;

class Television {
    String model;
    boolean on;
    int channel;
    int volume;

    Television(String model) {
        this.model = model;
    }
}
```

Java classes can be organized in different
namespaces by defining different packages

© 2019 ESTECO SpA

Hierarchical packages

com.esteco.sdm.Television

```
package com.esteco.sdm;

public class Television {
    String model;
    boolean on;
    int channel;
    int volume;

    public Television(String model) {
        this.model = model;
    }
}
```

it.units.sdm.Television

```
package it.units.sdm;

public class Television {
    String model;
    boolean on;
    int channel;
    int volume;

    public Television(String model) {
        this.model = model;
    }
}
```

Packages can be organized in hierarchies. Each package is separate from the parent using the dot notation

The package hierarchy must be reflected in the file system

© 2019 ESTECO SpA

The fully-qualified name

HelloTelevision.java

```java
class HelloTelevision {

    public static void main(String args[]) {
        com.esteco.sdm.Television tv1 = new com.esteco.sdm.Television("LG121");
        it.units.sdm.Television tv2 = new it.units.sdm.Television("LG121");

        System.out.println("Hello tv1: " + tv1.getClass().getName());
        System.out.println("Hello tv2: " + tv2.getClass().getName());
    }
}
```

fully-qualified name = package name + '.' + simple name

To reference classes in other packages they must be declared as public. The constructor must be declared as public, too!

© 2019 ESTECO SpA

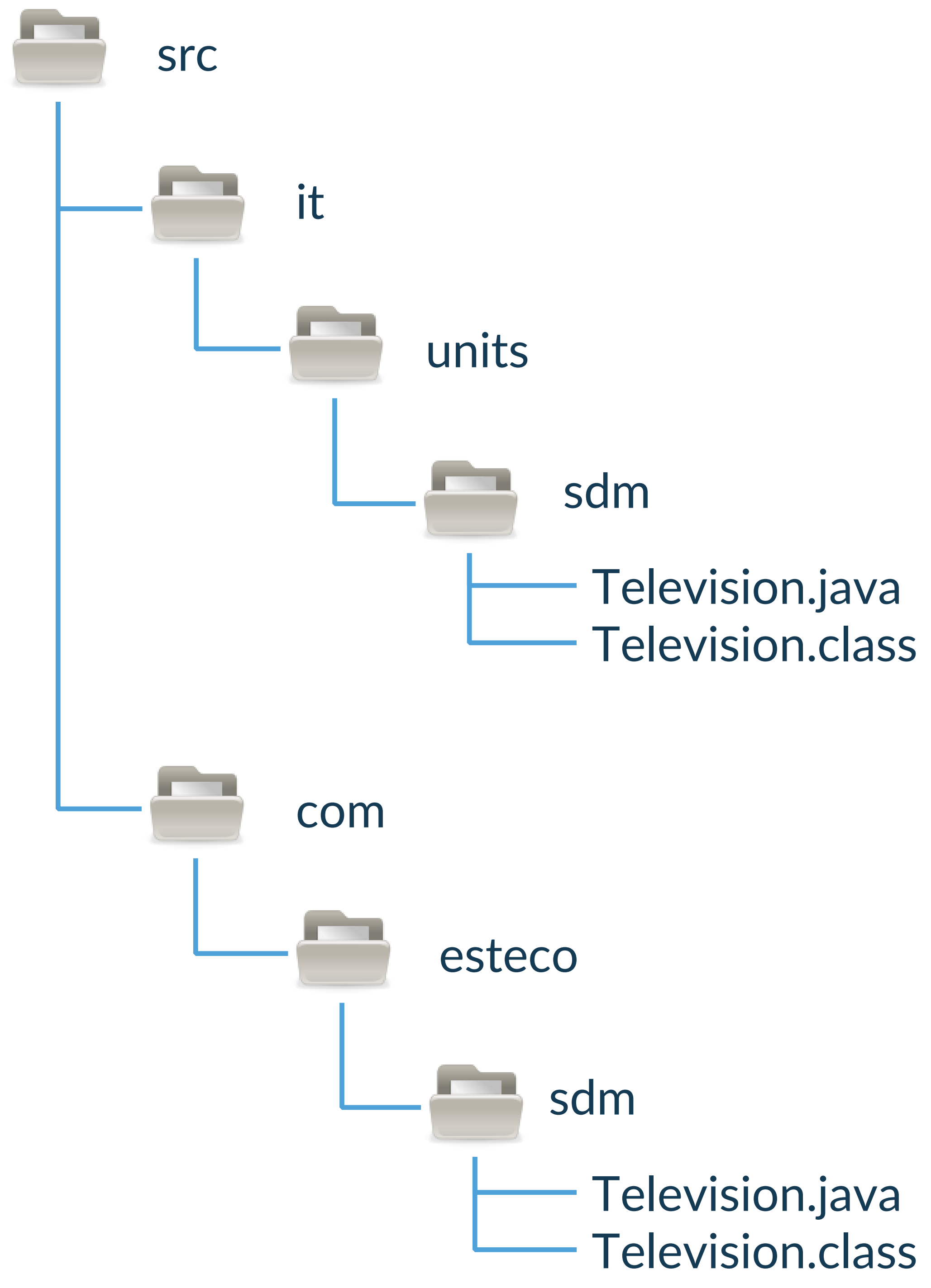
Location of Java files

```
com.esteco.sdm.Television
```

```java
package com.esteco.sdm;

class Television {
    String model;
    boolean on;
    int channel;
    int volume;

    Television(String model) {
        this.model = model;
    }
}
```



The package hierarchy must be reflected in the file system. This is not compulsory for Java source files, but it is for class files

© 2019 ESTECO SpA

Compilation & execution

```
path-to-src$ javac it/units/sdm/Television.java

path-to-src$ ls it/units/sdm/
Television.class  Television.java

path-to-src$ java it.units.sdm.Television
```

To compile a Java file we must specify its path. Either relative or absolute. The path can be independent from the declared package, but this practice is discouraged

To run a Java class we must specify its name, inclusive of the package. Java will search the class file converting the package into a path relative to the current location

I let you discover how single file launch works

© 2019 ESTECO SpA

Compilation of multiple sources

it.units.calculator.Calculator

```
package it.units.calculator;

class Calculator {

    final Display display;

    Calculator(Display display) {
        this.display = display;
    }

    void zeroPressed() {
        //...
    }

    //...
}
```

it.units.calculator.Display

```
package it.units.calculator;

class Display {
    void display(String text) {
        System.out.println(text);
    }
}
```

javac is able to find dependencies if they are in the path reflected by the package

For example

```
javac it/units/calculator/Calculator.java
```

Causes the compilation of the Display class

© 2019 ESTECO SpA

The import declaration

HelloTelevision.java

```java
import com.esteco.sdm.Television;

class HelloTelevision {

    public static void main(String args[]) {
        Television tv1 = new Television("LG121");
        it.units.sdm.Television tv2 = new it.units.sdm.Television("LG121");

        System.out.println("Hello tv1: " + tv1.getClass().getName());
        System.out.println("Hello tv2: " + tv2.getClass().getName());
    }
}
```

Each class can be referenced by using its simple name or the fully-qualified name. The simple name can be used

1. when the referenced class is in the same package of the current class
2. when the referenced class has been imported by using the import declaration

© 2019 ESTECO SpA

`java.lang` package

Classes in the java.lang package are imported by default

Notable classes in the java.lang package

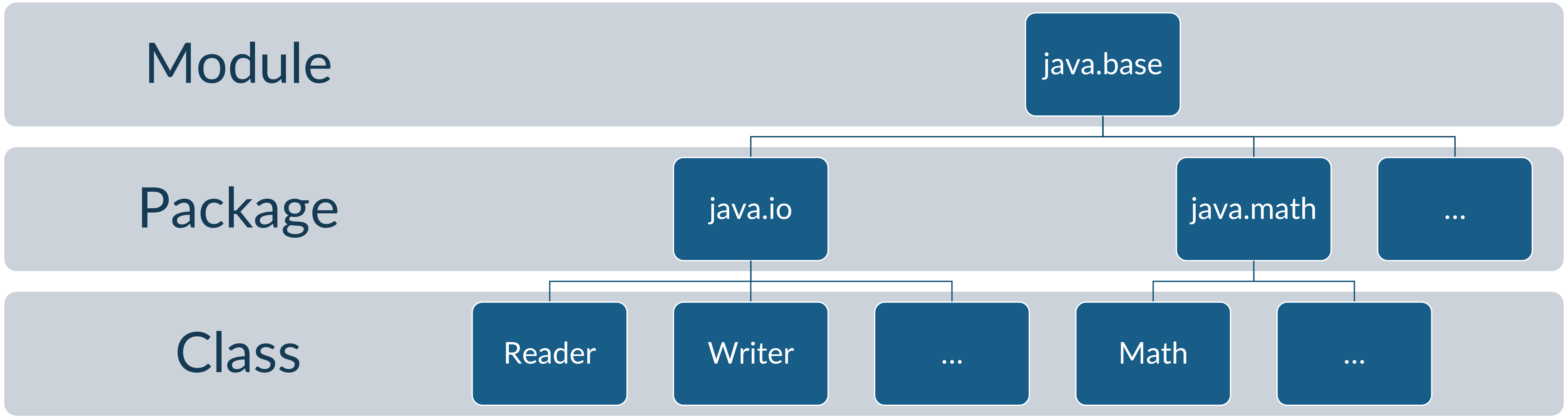
```
Class
Exception
Math
Object
Process/ProcessBuilder
Runnable
Runtime
String/StringBuilder
System
Thread
```

© 2019 ESTECO SpA

The Java library

© 2019 ESTECO SpA

Modules – Packages - Classes



Module reusable group of related packages

Package namespace that organizes a set of related classes and interfaces

Class template defining the fields and the methods of objects of that class

© 2019 ESTECO SpA

The Java library – java.base module

Package	Description
java.io	I/O through data streams
java.lang	Fundamental classes & interaction with environment
java.math	Arbitrary-precision arithmetic
java.net/javax.net	Networking applications
java.nio	New I/O
java.text	Handling of text, date, numbers, and messages
java.time	API for times, dates, instances, and durations
java.util.concurrent	Classes for concurrency
java.util.function	Classes for functional programming
java.util.stream	Classes for streams
java.util.zip	Reading and writing ZIP and GZIP files

© 2019 ESTECO SpA

The Java library – Other modules

Module	Package	Description
java.desktop	javax.swing	The GUI library
java.logging	java.util.logging	Classes for logging
java.net.http	java.net.http	HTTP Client and WebSocket API
java.prefs	java.util.prefs	Classes to store and retrieve system preferences
java.rmi	java.rmi	Support for Remote Method Invocation
java.sql	java.sql/javax.sql	Support for JDBC
java.xml	javax.xml	The Java API for XML Processing (JAXP)

© 2019 ESTECO SpA

Using external libraries

© 2019 ESTECO SpA

Using external libraries

- External libraries are organized in the so-called Jar-files
- Jar-files are collections of classes
- You can imagine a Jar file like a ZIP file with some metadata
- The classpath property is used by java to specify the places where to search for class files
- The classpath can be defined on the command line of java and javac, by using the –cp option
    - or in the environment variable CLASSPATH (not recommended)

© 2019 ESTECO SpA

Example JSON and JSON-java

https://www.json.org/json-en.html

© 2019 ESTECO SpA

Using JSONObject

JsonTest

```java
import org.json.JSONObject;

public class JsonTest {
    public static void main(String[] args) {
        JSONObject jsonObject = new JSONObject();
        jsonObject.put("pi", 3.14);
        jsonObject.put("a", new int[] {1, 2, 3});
        System.out.println(jsonObject.toString());
    }
}
```

```
>javac -cp lib\json-20220320.jar JsonTest.java
>java -cp .;lib\json-20220320.jar JsonTest
{"a":[1,2,3],"pi":3.14}
```

© 2019 ESTECO SpA

Demo

© 2019 ESTECO SpA

Assigment

© 2019 ESTECO SpA

Assignment

Implement a Java program to run scripts in a language of your choice , e.g., Python or JavaScript.

RunScript

```
public class RunScript {
    public static void main(String[] args) {
        PythonInterpreter interpreter = new PythonInterpreter();
        interpreter.runScript("print('Hello, World!')");
    }
}

class PythonInterpreter {
    void runScript(String script) {
      …
    }
}
```

© 2019 ESTECO SpA

The Intellij IDE

© 2019 ESTECO SpA

Demo

© 2019 ESTECO SpA

Thank you!

esteco.com

Read the ESTECO Copyright Policy © 2019 ESTECO SpA