



Ministerio de Producción
Presidencia de la Nación

Ministerio de Educación y Deportes

Subsecretaría de Servicios Tecnológicos y Productivos



**PROGRAMACIÓN ORIENTADA
A OBJETOS**



Ministerio de
Educación y Deportes
Presidencia de la Nación



Ministerio de Producción
Presidencia de la Nación

Programación Orientada a Objetos

Temas:

- Paquetes en Java
- Declaración y uso
- La sentencia **import**



Paquetes (Packages)

- Como ya hemos visto, existen clases que:
 - Están definidas y vienen con la distribución de Java
 - Están definidas por terceras partes
 - Fueron definidas por nosotros
- Todas las clases se organizan de acuerdo a un criterio de clasificación
- Los **paquetes** agrupan las clases siguiendo algún criterio



Paquetes - Sintaxis

- Todas las clases relacionadas con el “core” de Java (ej. String, Integer, Math) están en el paquete **java.lang**
- Para indicar en que paquete esta una clase, se debe definir al comienzo de la clase siguiendo la sintaxis:

```
package <nombrePaquete>[.<nombreSubPaquete>] ;
```

Ejemplos de Paquetes

- Cuando definimos una clase, podemos indicar a que paquete pertenece, usando la palabra clave **package**

```
package figuras.geometricas;  
public class Triangulo {  
    public void dibujar(){  
        System.out.println("dibujo del Triangulo");  
    }  
}
```

```
package figuras.geometricas;  
public class Circulo {  
    public void dibujar(){  
        System.out.println("dibujo del Circulo");  
    }  
}
```

```
package figuras.geometricas;  
public class Cuadrado {  
    public void dibujar(){  
        System.out.println("dibujo del Cuadrado");  
    }  
}
```

Las 3 clases pertenecen al
paquete
figuras.geometricas

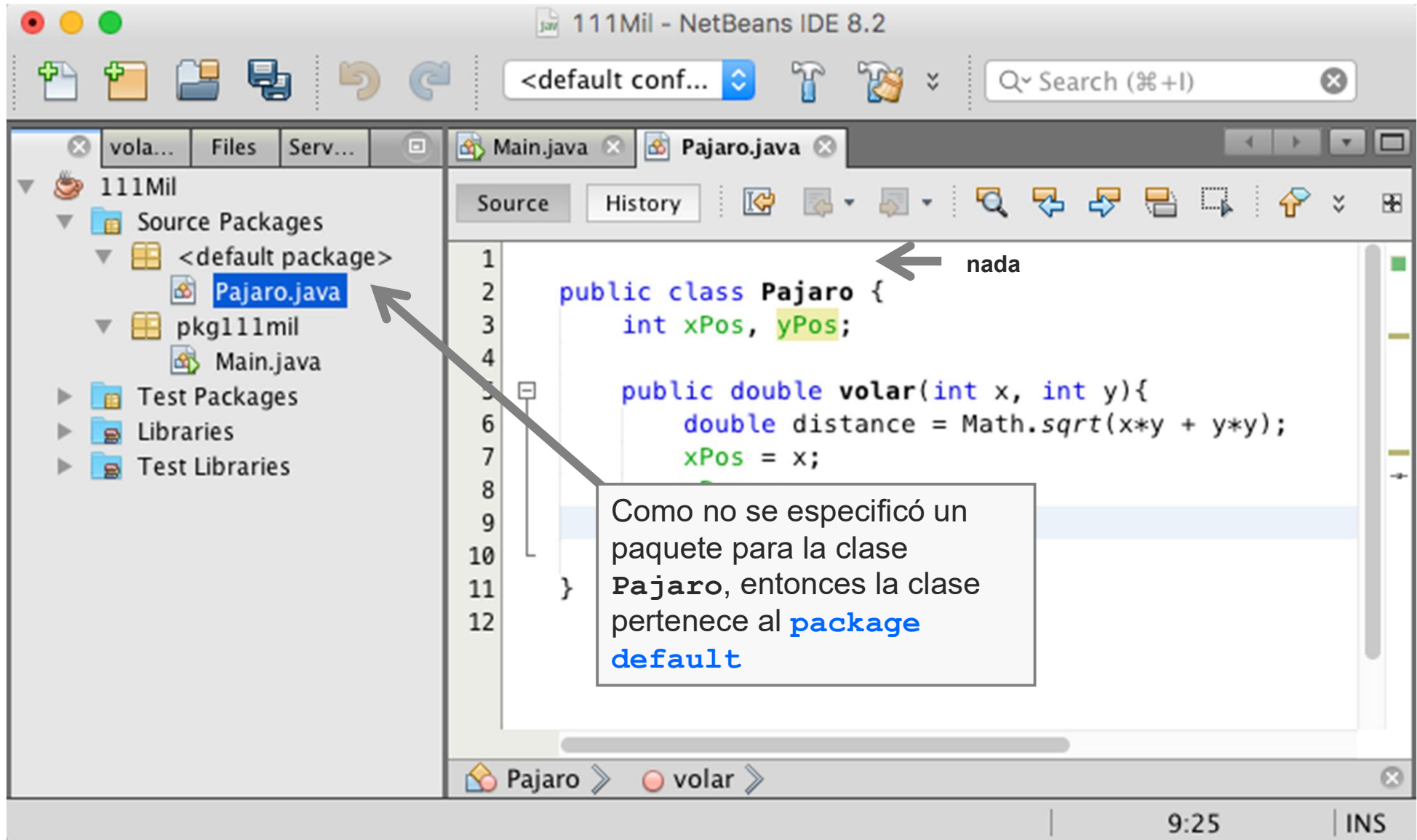
Paquetes – Declaración

- Una clase puede o no tener la declaración de un paquete.
- En caso de existir, debe estar al principio del archivo fuente.

```
package animales;  
class Pajaro {  
    int xPos,yPos;  
  
    double volar( int x, int y ) {  
        double distance = Math.sqrt( x*x + y*y );  
        xPos = x; yPos = y;  
        return distance;  
    }  
    ...  
}
```

- Si en un archivo fuente se omite la declaración de un **package**, entonces la clase declarada pertenecerá al paquete sin nombre, o sea al package **default**

Paquetes – Declaración



111Mil - NetBeans IDE 8.2

<default conf... Search (%+l)

Files Serv... Main.java Pajaro.java

111Mil

- Source Packages
 - <default package>
 - Pajaro.java
 - pkg111mil
 - Main.java
- Test Packages
- Libraries
- Test Libraries

```
1 public class Pajaro {
2     int xPos, yPos;
3
4     public double volar(int x, int y){
5         double distance = Math.sqrt(x*y + y*y);
6         xPos = x;
7     }
8
9
10
11 }
12
```

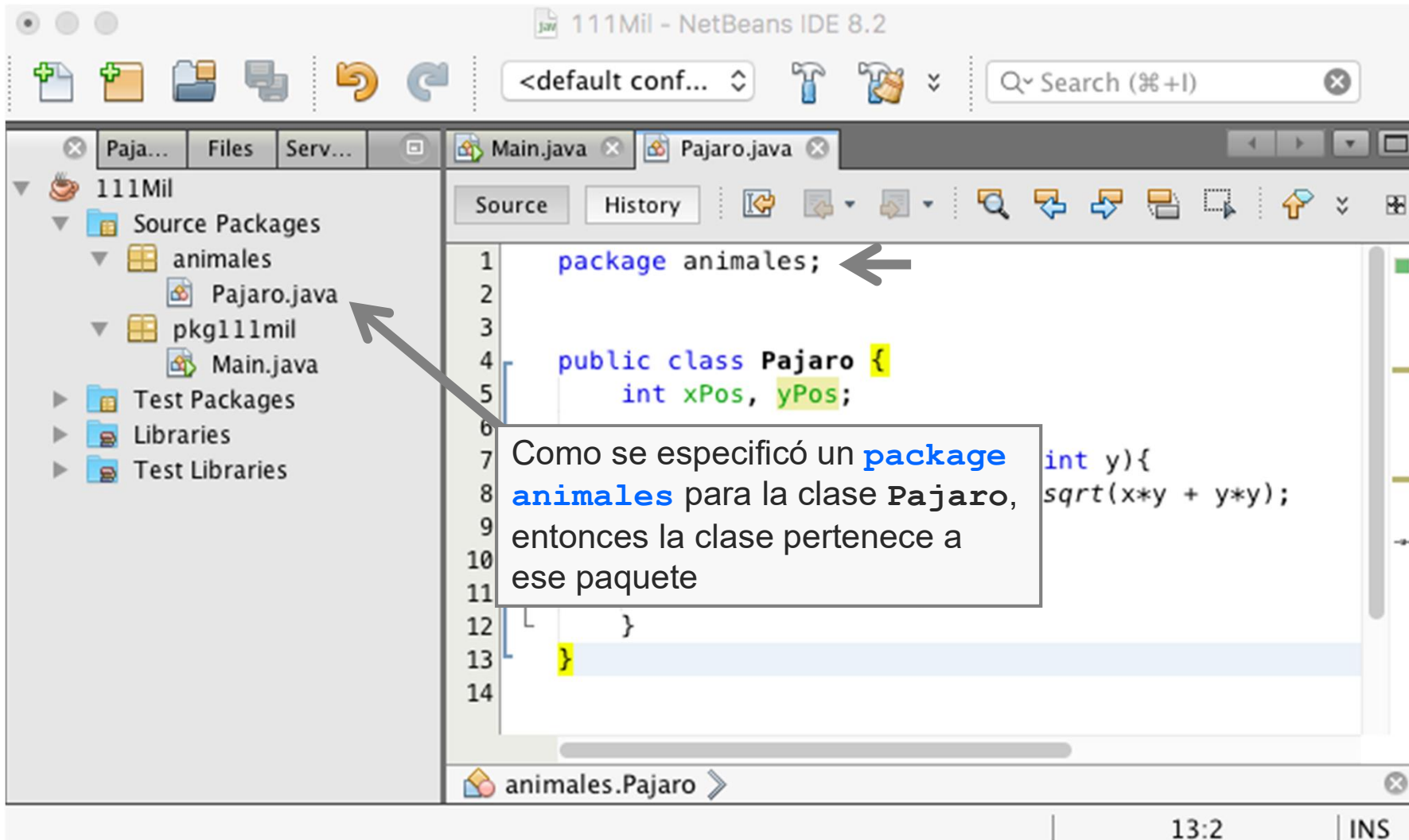
nada

Como no se especificó un paquete para la clase Pajaro, entonces la clase pertenece al **package default**

Pajaro > volar >

9:25 INS

Paquetes – Declaración

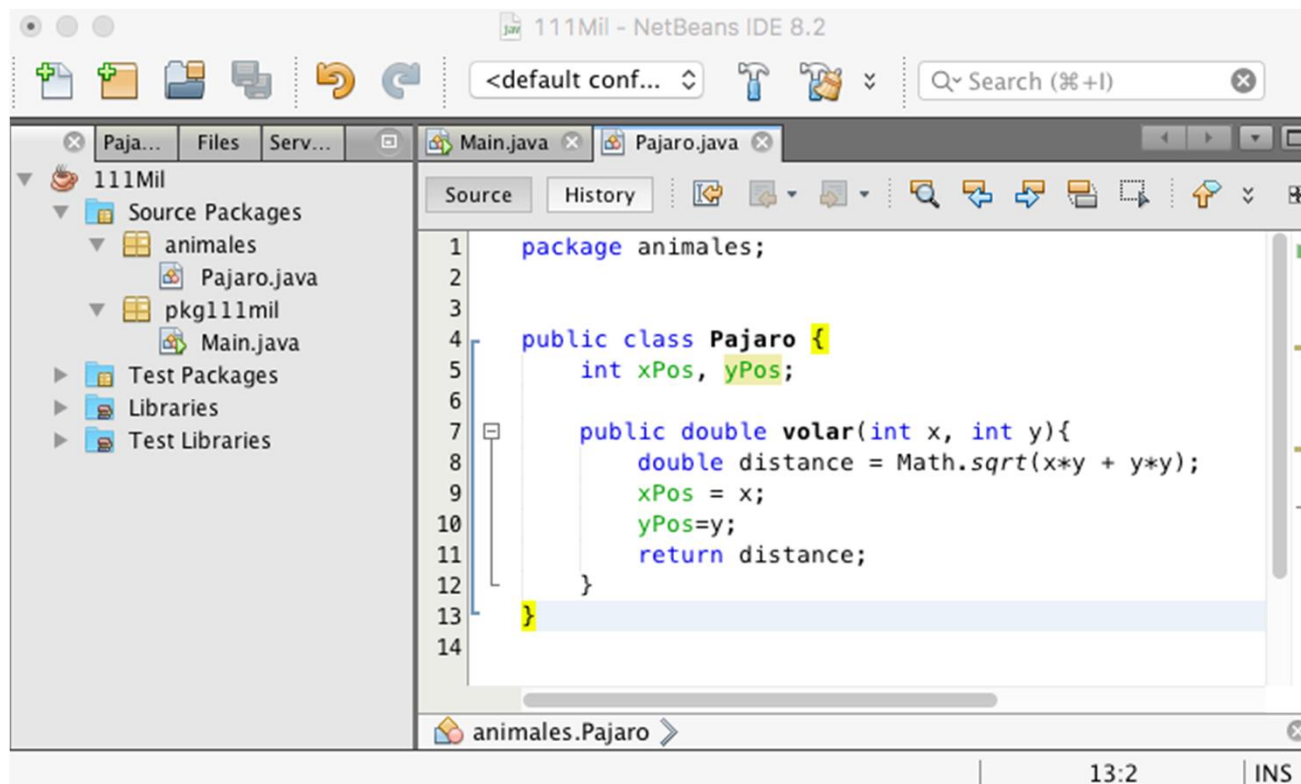


The screenshot shows the NetBeans IDE 8.2 interface. On the left, the Project Explorer displays the project structure: 111Mil, Source Packages, animales (containing Pajaro.java), pkg111mil (containing Main.java), Test Packages, Libraries, and Test Libraries. Arrows point from the 'animales' package and the 'Pajaro.java' file to the code editor. The code editor shows the following code:

```
1 package animales; ←
2
3
4 public class Pajaro {
5     int xPos, yPos;
6
7     int y){
8         sqrt(x*y + y*y);
9
10
11
12 }
13
14
```

A tooltip box contains the text: "Como se especificó un **package animales** para la clase **Pajaro**, entonces la clase pertenece a ese paquete". The status bar at the bottom shows "animales.Pajaro" and "13:2 | INS".

Paquetes – Declaración

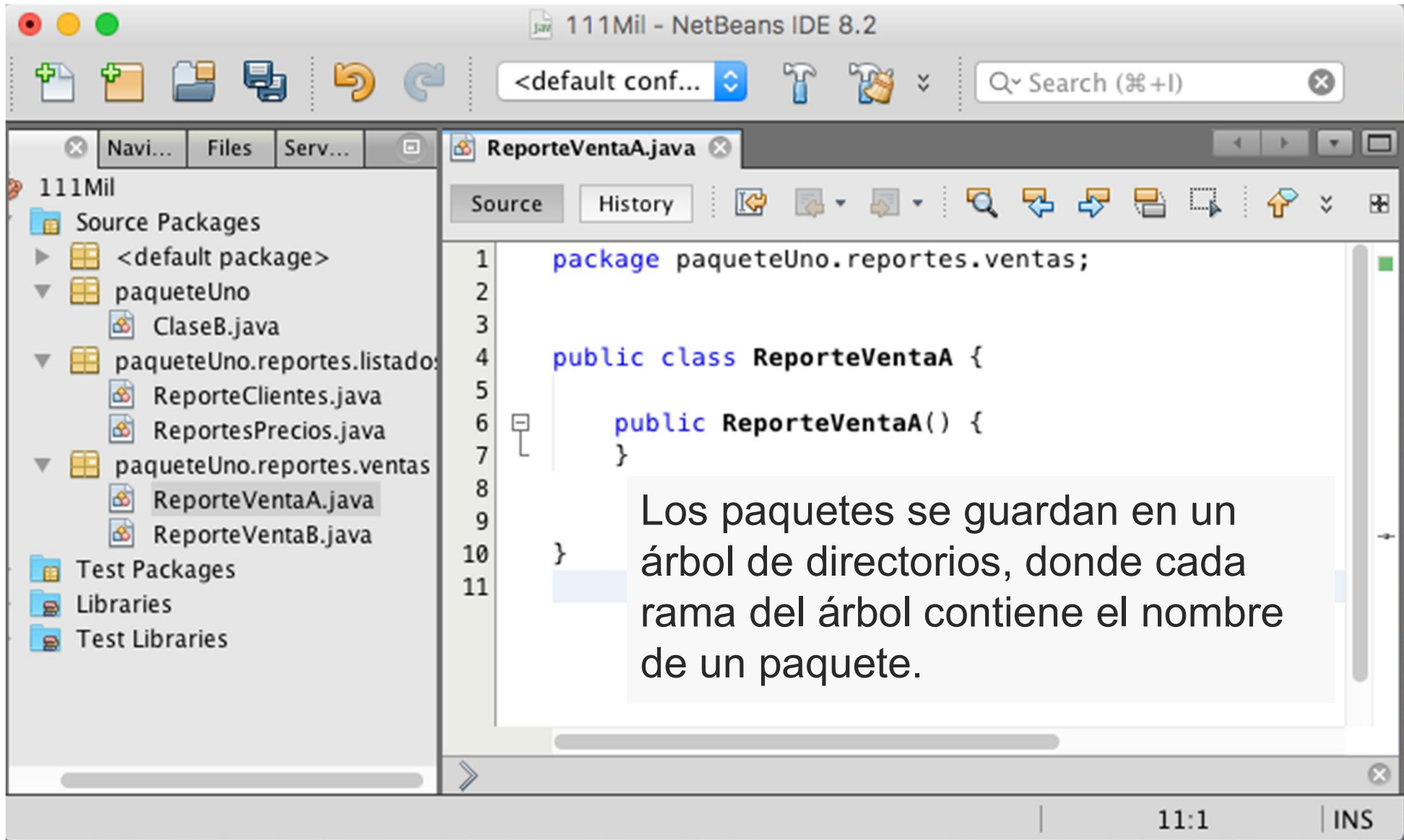


¿Cuál es el nombre
completo de la clase
Pajaro ?

animales.Pajaro



Paquetes – Carpetas



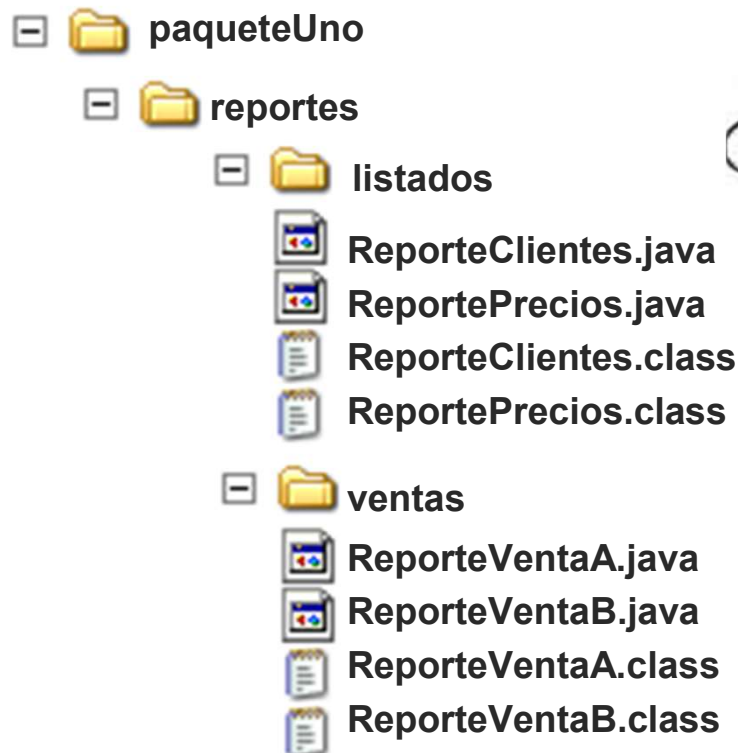
The screenshot shows the NetBeans IDE 8.2 interface. On the left, the 'Project Explorer' (Navi...) displays a project named '111Mil'. Under 'Source Packages', there is a hierarchy: '<default package>' (expanded), 'paqueteUno' (expanded), and 'paqueteUno.reportes.ventas' (expanded). Inside 'paqueteUno.reportes.ventas', the files 'ReporteVentaA.java' and 'ReporteVentaB.java' are listed. The 'ReporteVentaA.java' file is selected, and its code is displayed in the 'Source' tab of the 'Code Editor'. The code defines a package 'paqueteUno.reportes.ventas' and a public class 'ReporteVentaA' with a public constructor 'ReporteVentaA()'. A text box is overlaid on the code editor, containing the text: 'Los paquetes se guardan en un árbol de directorios, donde cada rama del árbol contiene el nombre de un paquete.'

```
1 package paqueteUno.reportes.ventas;
2
3
4 public class ReporteVentaA {
5
6     public ReporteVentaA() {
7     }
8
9
10 }
11
```

Los paquetes se guardan en un árbol de directorios, donde cada rama del árbol contiene el nombre de un paquete.

Paquetes - Carpetas

- Los paquetes se guardan en un árbol de directorios en el disco de la computadora, donde cada rama del árbol contiene una parte del nombre del paquete.



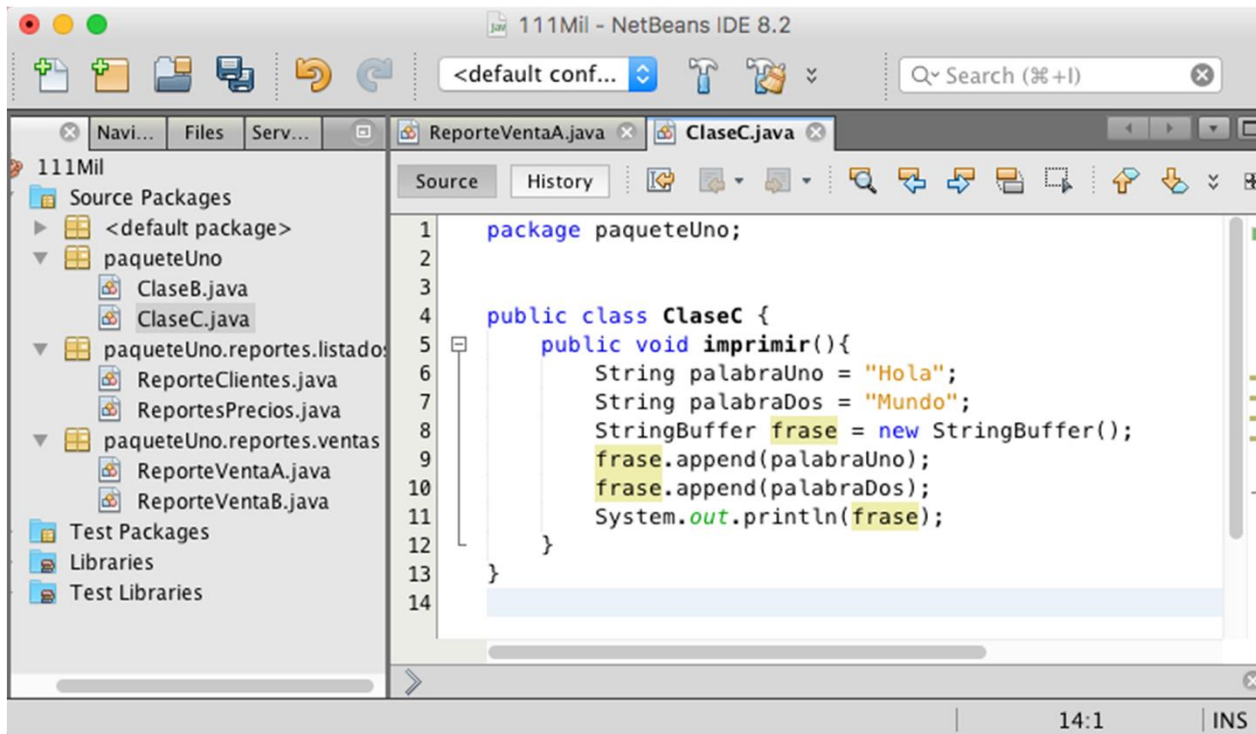
¿Cuál es el nombre completo de la clase ReportePrecios ?

`paqueteUno.reportes.listados.ReportePrecios`



Paquetes - Uso

- Cuando queremos usar una clase que pertenece al “core” o al paquete `java.lang`, no es necesario especificarlo, éstas clases se importan automáticamente.



Estamos utilizando la Clase `String`, `StringBuffer` y `System` y no indicamos en ningún momento a que paquete pertenece. Esto es porque están en el package `java.lang`.

Toda clase que pertenece al paquete `java.lang` se puede usar sin necesidad de indicar de dónde proviene.



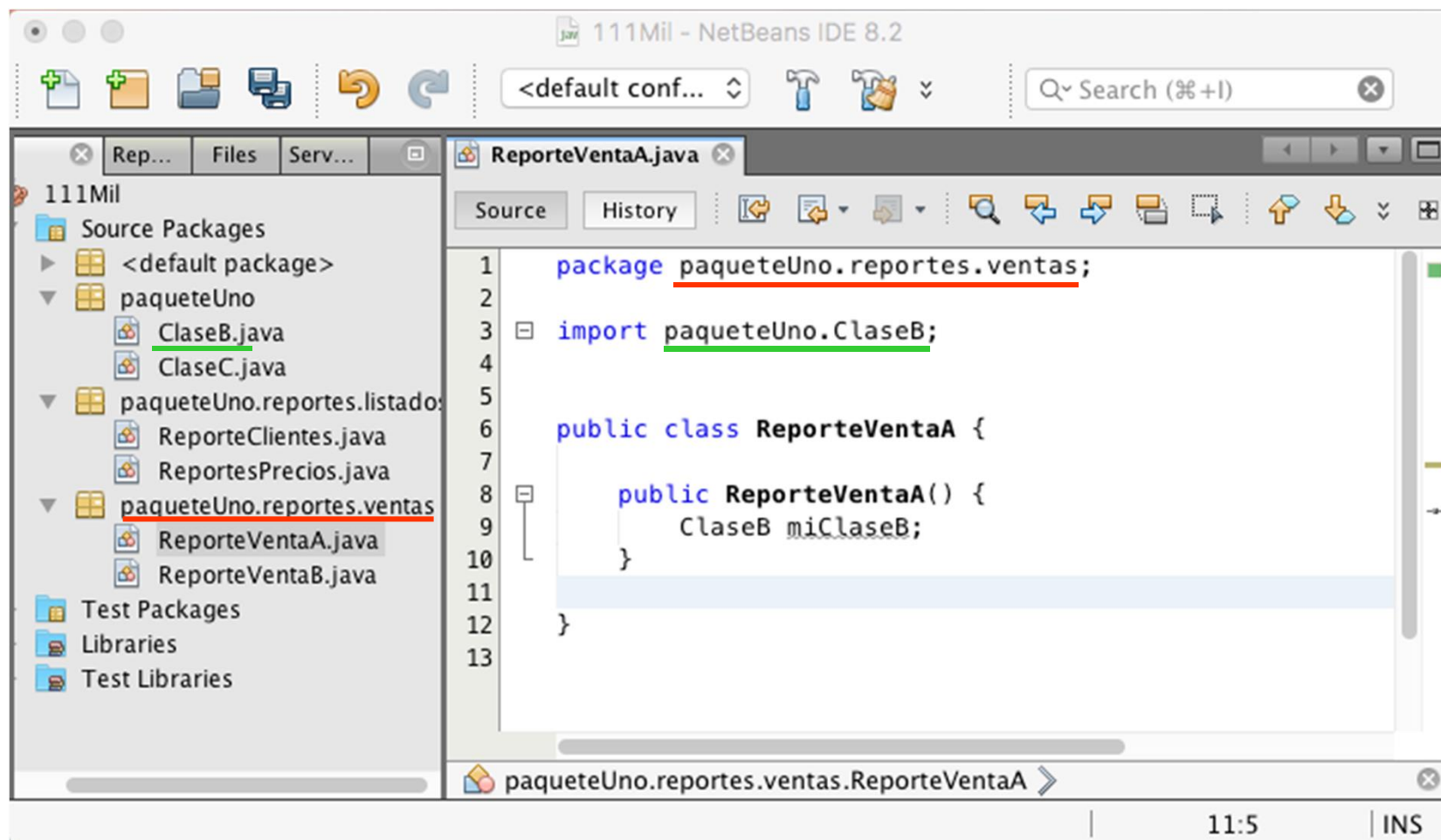
Paquetes - Uso

- Si queremos utilizar una clase que no pertenece al package **java.lang** es necesario indicar que vamos a usar una clase que pertenece a un paquete en particular, y eso se puede realizar de dos maneras:
 - utilizando la sentencia **import**
 - escribiendo el nombre completo de la clase (nombre del paquete + el nombre de la clase).

Paquetes - Uso

Sintaxis del **import**

```
import <nombrepaquete>[.<nombresubpaquete>].<nombreClase>;
```



Paquetes - Uso

Si quiero usar 30 clases de un paquete, ¿tengo que poner 30 sentencias **import**?

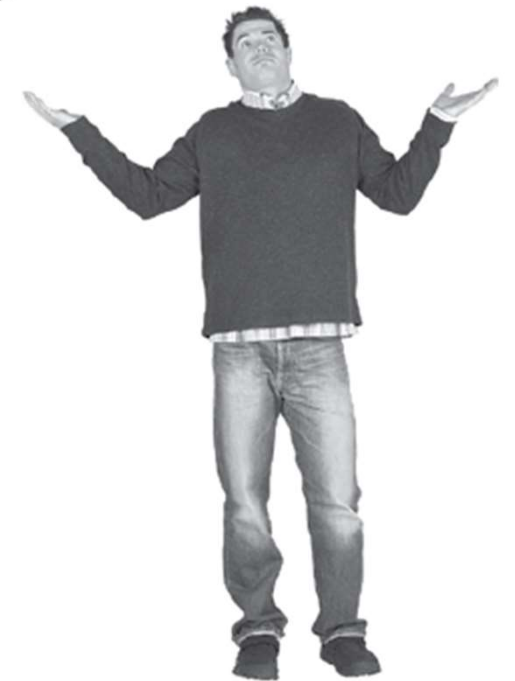
No!!, es posible importar todas las clases pertenecientes a un paquete con un solo **import**, usando **"*"**:

```
import <nombrepaquete>[.<nombresubpaquete>] .*;
```

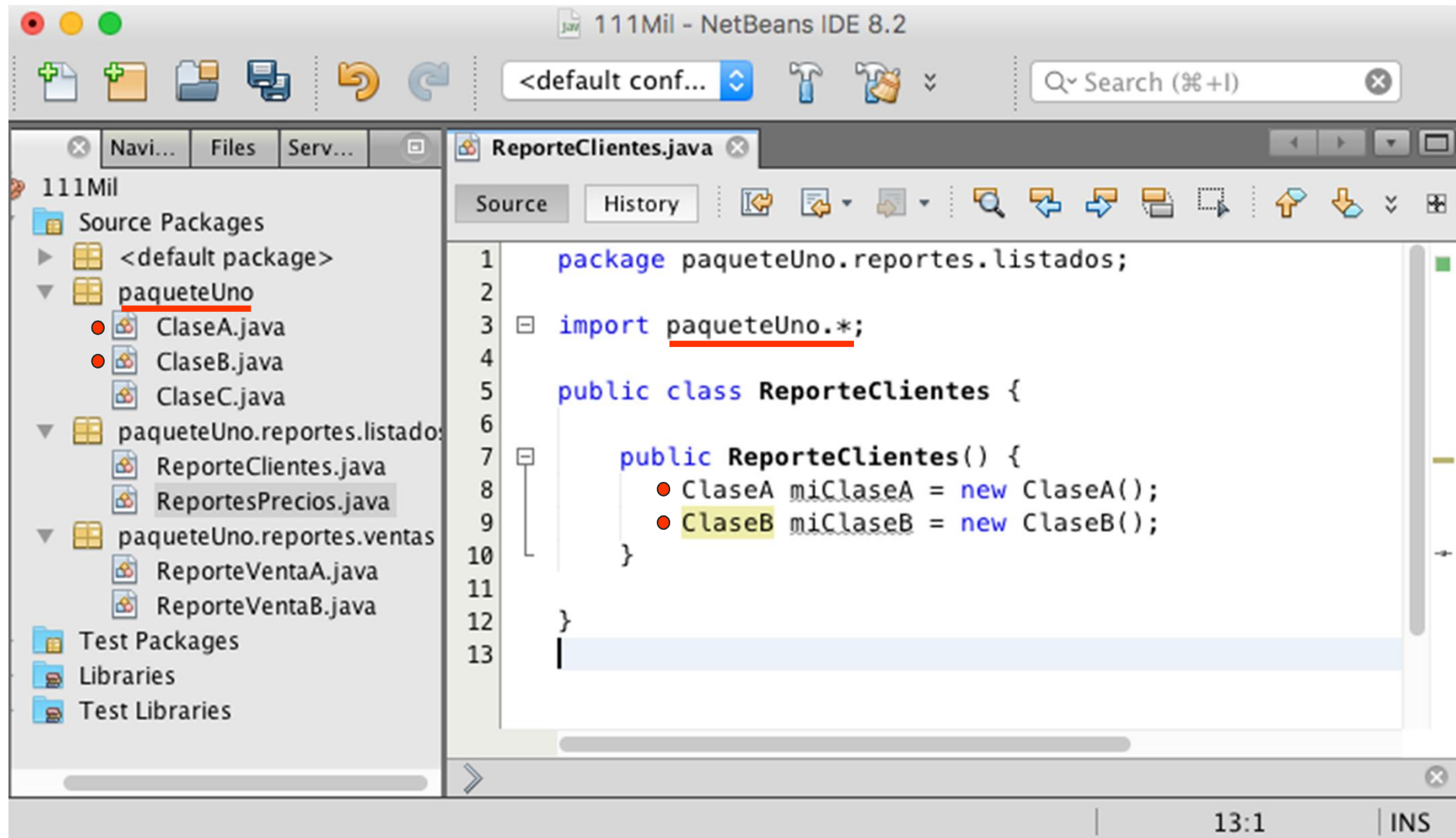
Por ejemplo:

```
import paqueteuno.reportes.listados.*;
```

De esta manera, TODAS las clases públicas del paquete **paqueteuno.reportes.listados** están disponibles para usar.



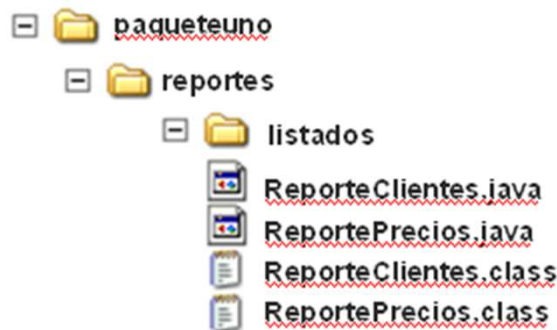
Paquetes - Uso



Paquetes – Uso canónico

Otra manera de utilizar una clase sin hacer uso de la palabra clave **import**, es utilizando el nombre completo de la clase (también conocido como nombre canónico).

Cada vez que se define una variable de un tipo referencia o se crea una instancia de una clase, se lo debe hacer poniendo el nombre canónico.

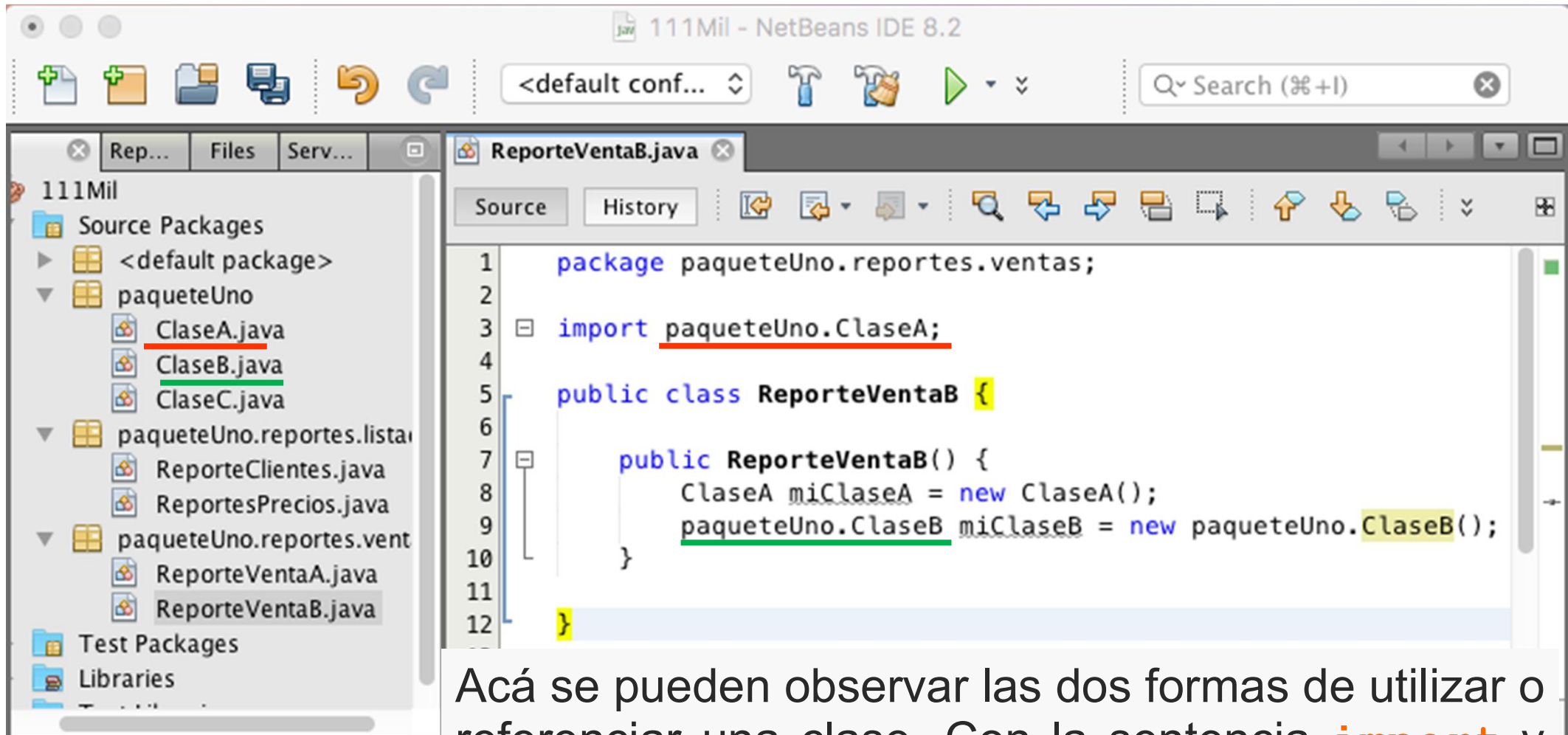


Para referenciar a la clase

ReportePrecios, debemos poner:

`paqueteuno.reportes.listados.ReportePrecios`

Paquetes – Uso canónico

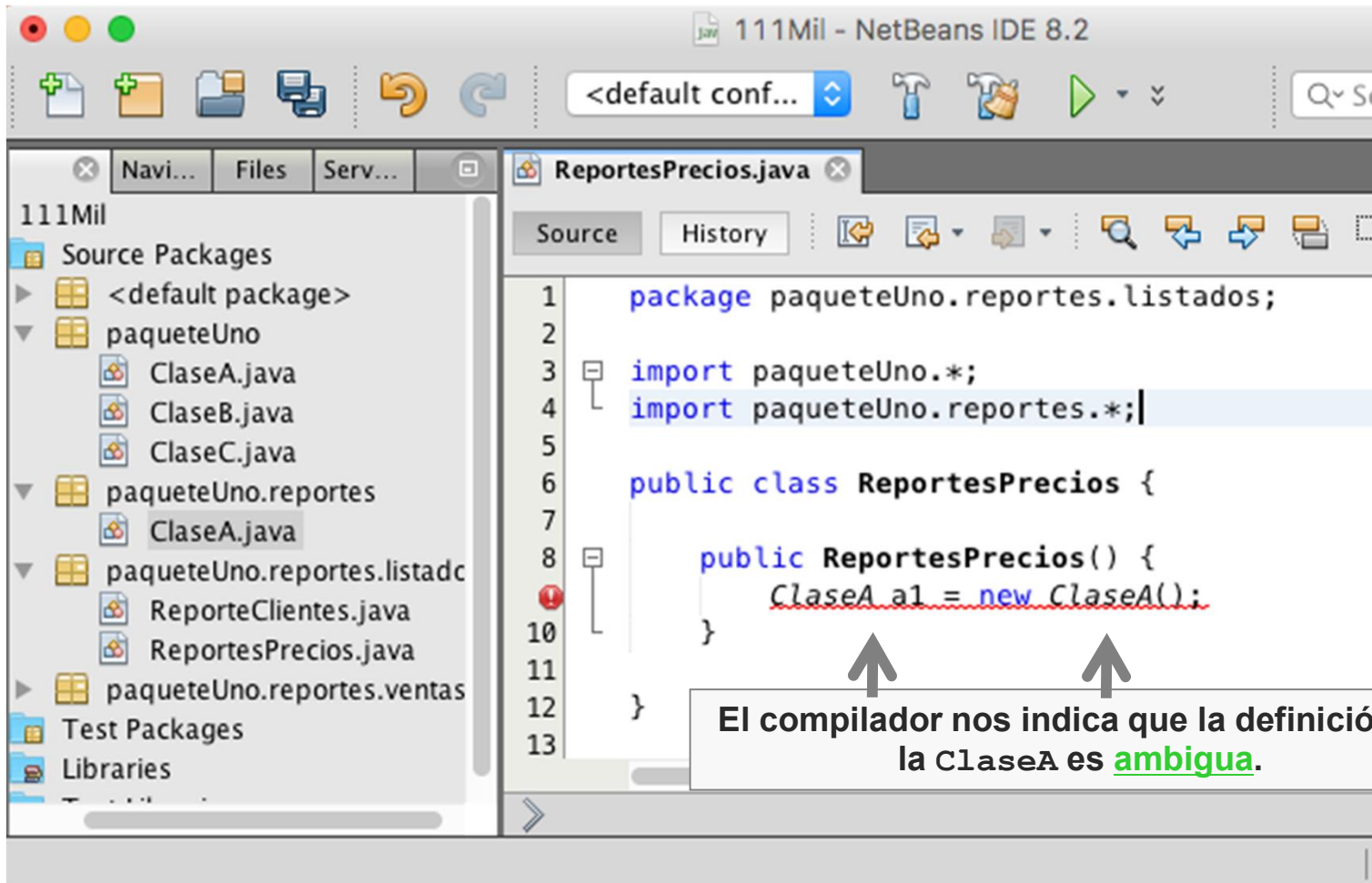


The screenshot shows the NetBeans IDE 8.2 interface. On the left, the 'Project Explorer' displays the project structure for '111Mil'. It includes 'Source Packages' with a '<default package>', a 'paqueteUno' package containing 'ClaseA.java', 'ClaseB.java', and 'ClaseC.java', and a 'paqueteUno.reportes.ventas' package containing 'ReporteVentasA.java' and 'ReporteVentasB.java'. The 'Test Packages' and 'Libraries' sections are also visible. The main editor window shows the 'ReporteVentasB.java' file. The code in the editor is as follows:

```
1 package paqueteUno.reportes.ventas;
2
3 import paqueteUno.ClaseA;
4
5 public class ReporteVentasB {
6
7     public ReporteVentasB() {
8         ClaseA miClaseA = new ClaseA();
9         paqueteUno.ClaseB miClaseB = new paqueteUno.ClaseB();
10    }
11
12 }
```

Acá se pueden observar las dos formas de utilizar o referenciar una clase. Con la sentencia **import** y haciendo mención al nombre completo de la clase.

Paquetes – Conflictos

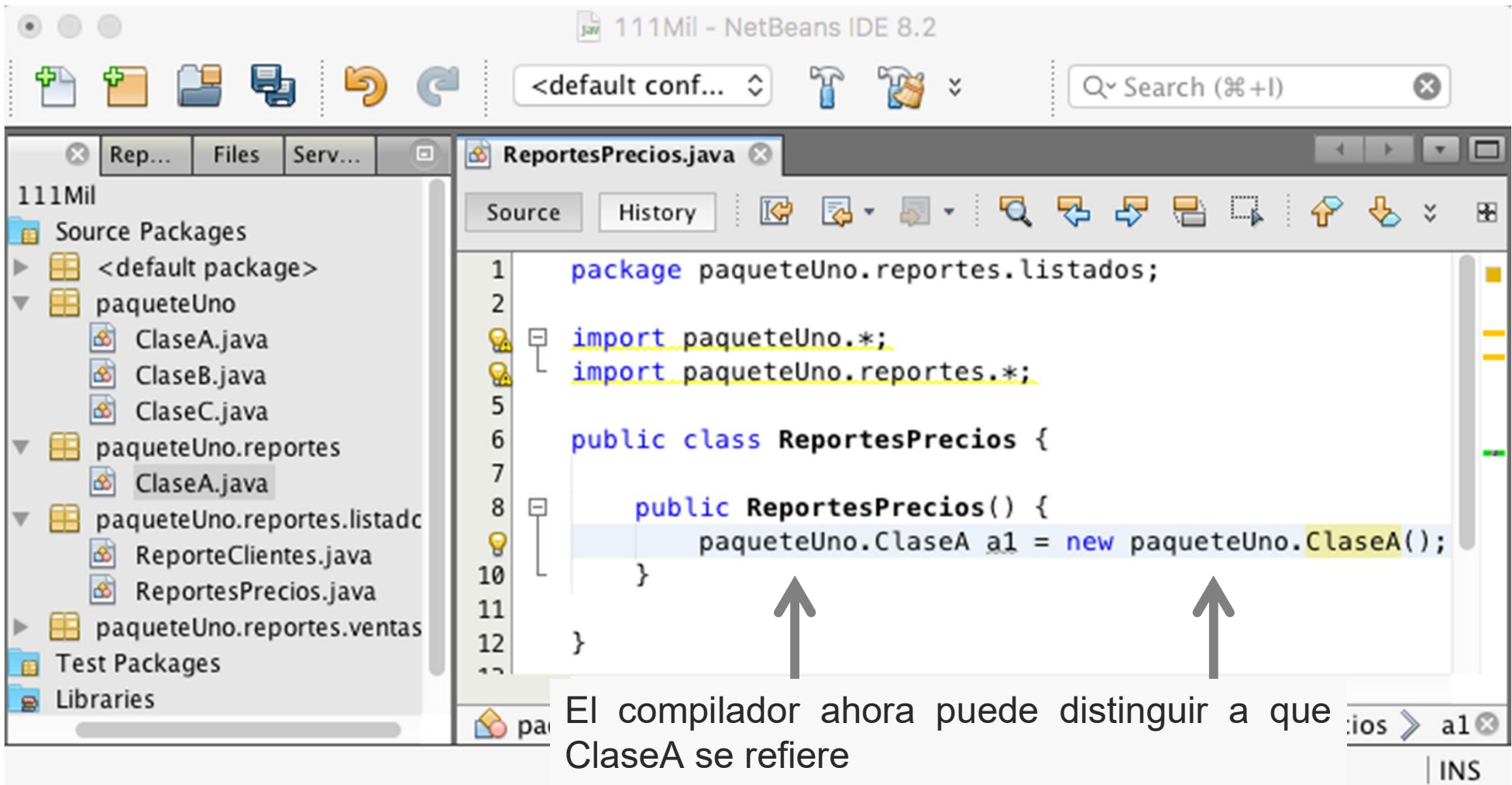


¿qué pasa ahora?
¿cómo hago?



Paquetes – Conflictos

Para resolver la **ambigüedad**, se debe utilizar el nombre completo de la clase.



```
1 package paqueteUno.reportes.listados;
2
3 import paqueteUno.*;
4 import paqueteUno.reportes.*;
5
6 public class ReportesPrecios {
7
8     public ReportesPrecios() {
9         paqueteUno.ClaseA a1 = new paqueteUno.ClaseA();
10    }
11
12 }
```

El compilador ahora puede distinguir a que ClaseA se refiere