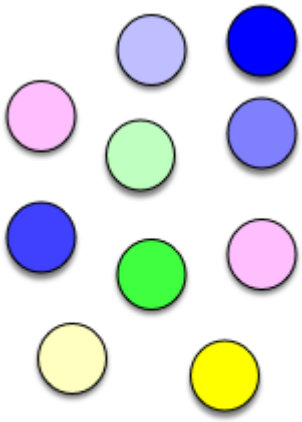
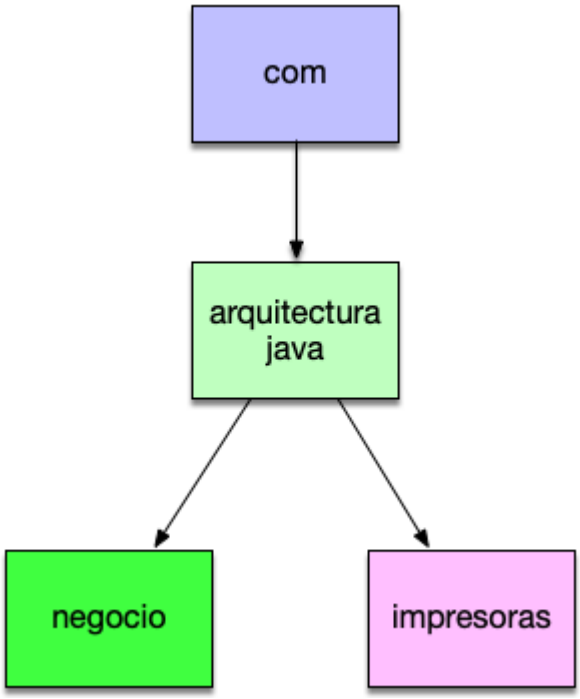
Java Packages vs JARs . ¿Qué diferencia existe entre un Java JAR y un Java Package? . Este es una pregunta que es bastante habitual entre desarrolladores y siempre surgen dudas en cuando a como se pueden abordar las cosas. El uso de paquetes esta muy enfocado al hecho de organizar un conjunto de clases. Es decir según el programa crece el numero de clases del que dispondremos será cada vez mayor . Esto pronto o tarde generará la necesidad de organizarlas

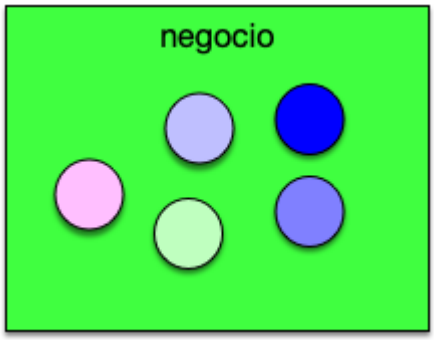


Clases Java

Ahora bien este solo es el punto de partida ya que las posibilidades de organizar las clases son muy diversas. Uno de los puntos de partida más habituales a la hora de organizar las clases es partir del dominio de internet de tu empresa y darle la vuelta . Es decir en mi caso arquitecturajava.com daríamos la vuelta al dominio “com.arquitecturajava” y de esta forma comenzaríamos a organizar nuestros packages a nivel de estructura externa para luego definir con claridad que organización interna queremos en nuestro programa.



Cada uno de nuestros packages contendrá un conjunto de clases .



En este caso vamos a tener dos packages principales uno que define las clases de negocio. Este contendrá la clase Documento y otro que define las clases que se encargan de imprimir documentos en nuestro caso el package de Impresoras. Así pues el código podría ser algo del siguiente estilo.

```
package com.arquitecturajava.negocio;
public class Documento {
```

```
        public String getTexto() {
                return texto;
        }

        public void setTexto(String texto) {
                this.texto = texto;
        }

        private String texto;

        public Documento(String texto) {
                super();
                this.texto = texto;
        }
}
```

```
package com.arquitecturajava.impresoras;
import com.arquitecturajava.negocio.Documento;
public interface Impresora {
        public void imprimir(Documento documento);
}
```

```
package com.arquitecturajava.impresoras;

import com.arquitecturajava.negocio.Documento;

public class ImpresoraWord implements Impresora {

        @Override
        public void imprimir(Documento documento) {
                System.out.println("imprimiendo en word");
                System.out.println(documento.getTexto());
```

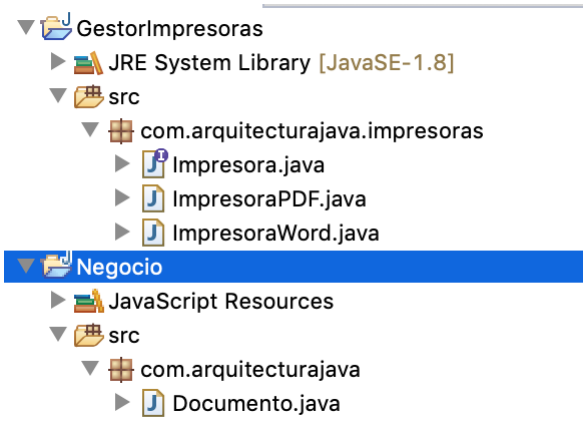
```
        }

}
```

```
package com.arquitecturajava.impresoras;

import com.arquitecturajava.Documento;

public class ImpresoraPDF implements Impresora {

        @Override
        public void imprimir(Documento documento) {
                System.out.println("imprimiendo en pdf");
                System.out.println(documento.getTexto());
        }

}
```

Ya tenemos el código de nuestras clases . Es momento de abordar el concepto de JAR o libreria.

Java Packages vs JARs

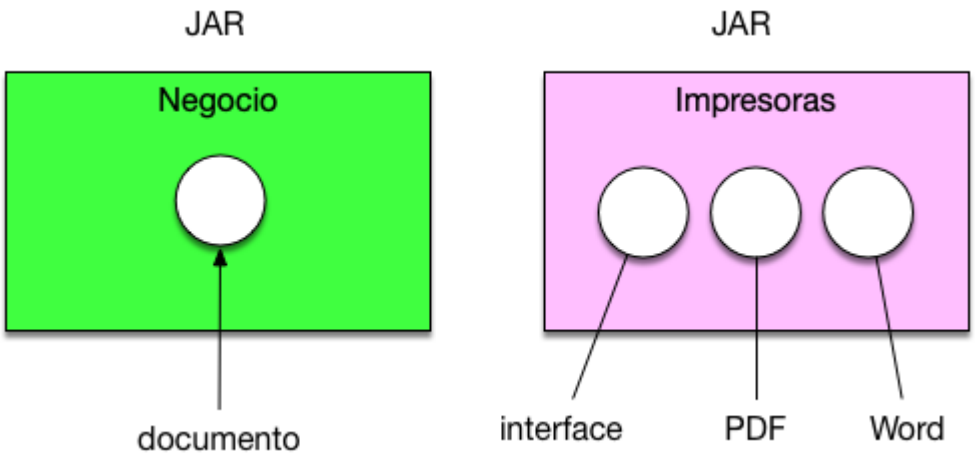
Un JAR esta diseñado como una unidad de reutilización de código. Es decir esta orientado a como vamos a querer reutilizar un conjunto de clases a futuro. En este caso hemos definido dos JARs o Librerías como Eclipse Utility Proyects.



Uno de los proyectos incluye la clase Documento y el otro incluye las clases que se encargan de de manejar las impresoras para imprimir esos Documentos . ¿ Es esta organización la más correcta a la hora de favorecer la reutilización del código?. La respuesta es que depende. En principio puede ser una solución perfectamente válida. Pero la organización puede ser diferente y esta es una de las características fundamentales de los JARs

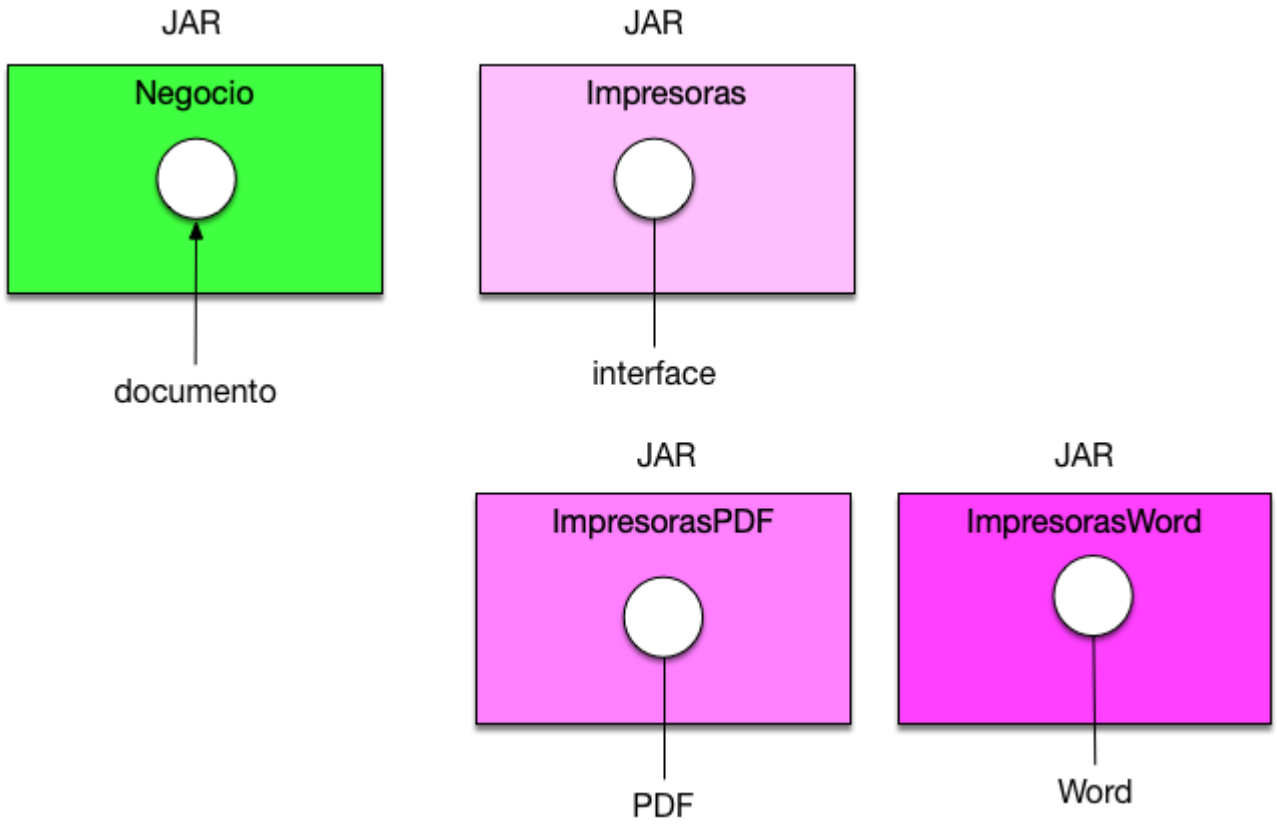
Java Packages y enfoques

¿Como ubicamos las clases para su futura reutilización? . En nuestro caso parece bastante razonable el como hemos dividido las cosas.



Ahora bien pueden existir enfoques diferentes. Por ejemplo puede haber aplicaciones para

las cuales sea interesante dividir nuestras clases en un conjunto mayor de librerías porque en unos casos instalaremos las impresoras para documentos word y en otros casos instalaremos las impresoras para documentos PDF.



Por lo tanto la organización de Packages esta mucho más orientada a definir cual es la estructura a nivel lógico de la aplicación de tal forma que el desarrollador sea capaz de orientarse de una manera natural . Mientras tanto la organización a nivel de JARs o librerías esta mucho más orientada a las capacidades de reutilización de nuestras clases y la modularización de nuestra arquitectura. Tengámoslo siempre en cuenta cuando hablemos de Java Packages vs JARs

Otros artículos relacionados

1. Java EAR
2. Java WAR