

## Tabla de Contenidos



- [Combinando Cadenas](#)
- [Java StringJoiner](#)
- [Otros artículos relacionados](#)

El concepto de StringJoiner es uno de esos conceptos que muchas personas no conocen del API fundamental de Java y que nos permite manejar agrupaciones de cadenas de forma sencilla . Vamos a explicar este concepto partiendo de un Array de cadenas.

```
package com.arquitecturajava;
```

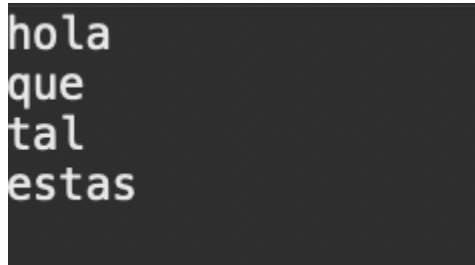
```
public class Principall {
```

```
    public static void main(String[] args) {  
        String[] lista=new String[] {"hola","que","tal","estas"};  
        for( String texto:lista) {  
            System.out.println(texto);  
        }  
    }
```

```
}
```

```
}
```

Esta lista de cadenas esta construida con un Array de Java y simplemente tenemos que recorrerla con un bucle forEach para que nos imprima los datos en la consola .



```
hola
que
tal
estas
```

## Combinando Cadenas

En muchos casos necesitamos combinar cadenas y mostrarlas en una única línea separadas por algún tipo de caracter especial .

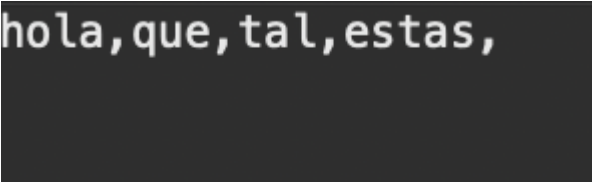
```
package com.arquitecturajava;
```

```
public class Principal2 {
```

```
    public static void main(String[] args) {
        String[] lista=new String[] {"hola","que","tal","estas"};
        String textoTotal="";
        for( String texto:lista) {
            textoTotal+=texto+",";
        }
        System.out.println(textoTotal);
    }
```

```
}
```

En este caso el resultado imprimirá las cadenas separadas por comas:



hola,que,tal,estas,

No es el resultado que nosotros deseamos al combinar un conjunto de cadenas ya que nos sobra la última coma así como nos gustaría tener delimitadores al principio y final de la cadena. Una solución más elaborada usando substring sería:

```
package com.arquitecturajava;
```

```
public class Principal3 {
```

```
    public static void main(String[] args) {
        String[] lista=new String[] {"hola","que","tal","estas"};
        String textoTotal="";
        textoTotal+="(";
        for( String texto:lista) {
            textoTotal+=texto+",";
        }
        textoTotal=textoTotal.substring(0,textoTotal.length()-1);
        textoTotal+=")";
        System.out.println(textoTotal);
    }
```

```
}
```

```
public class Principal3 {
```

```
    public static void main(String[] args) {
        String[] lista=new String[] {"hola","que","tal","estas"};
```

```

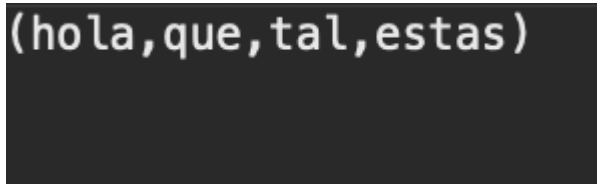
String textoTotal="";
textoTotal+="(";
for( String texto:lista) {
    textoTotal+=texto+",";
}
textoTotal=textoTotal.substring(0,textoTotal.length()-1);
textoTotal+=")";
System.out.println(textoTotal);

}

}

```

El resultado queda más elegante:



```
(hola,que,tal,estas)
```

## Java StringJoiner

El único problema que tenemos es que el código queda bastante enrevesado y no es sencillo de entender al echarle un vistazo. Es aquí donde la clase `StringJoiner` brilla y nos permite simplificar este código de un plumazo:

```

package com.arquitecturajava;

import java.util.StringJoiner;

public class Principal4 {

```

```

public static void main(String[] args) {

    String[] lista = new String[] { "hola", "que", "tal", "estas" };

    StringJoiner joiner = new StringJoiner(",", "(", ")");

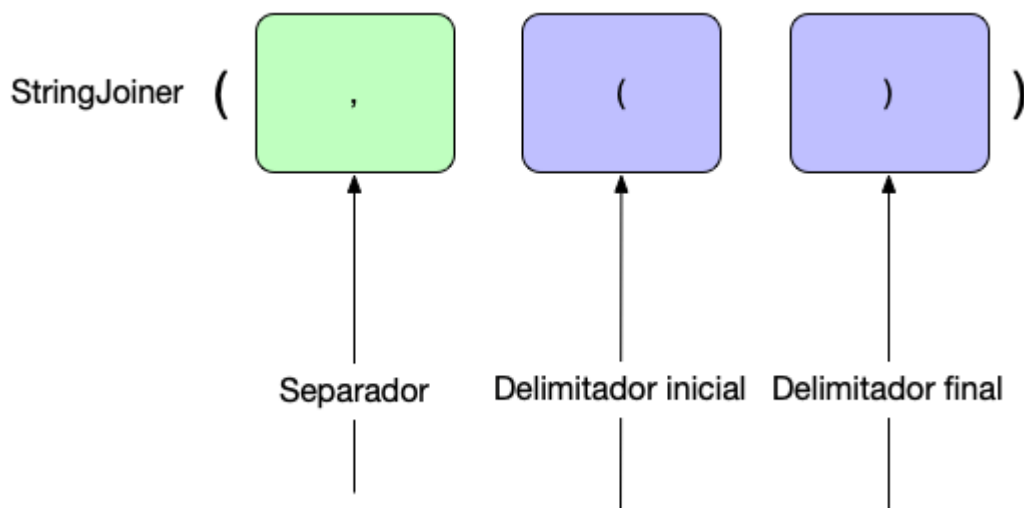
    for (String texto : lista) joiner.add(texto);

    System.out.println(joiner.toString());

}
}

```

Como se puede ver esta clase define el separador y los delimitadores de inicio y fin permitiendonos realizar la misma operación de una forma mucho mas elegante .



El resultado es idéntico:

```
(hola,que,tal,estas)
```

## Otros artículos relacionados

1. [Java String to Date utilizando Java 8](#)
2. [File to String Java 8 y manejo de ficheros](#)
3. [Java toString overriding y Eclipse](#)
4. [Java Stream String y Java 8](#)
5. [Oracle StringJoiner](#)