El uso de Java Stream String es muy común cuando utilizamos Java 8. La realidad es que en muchos casos necesitamos transformar una cadena de texto en otra cosa y que mejor que usar los Streams y la programación funcional para hacerlo. Así que todos nosotros rápidamente vamos a revisar el API de Java y ver si los Strings soportan Streams para empezar a usarlo. Pronto nos encontramos con el método chars() que nos devuelve un String de caracteres. Esto nos permite hacer operaciones como la siguiente.

```
import java.util.stream.Stream;

public class Principal2 {

	public static void main(String[] args) {

		String texto = "hola que tal estas ";

		long total=texto.chars()
		 .mapToObj(i ->(char) i)
		 .filter((l)->l=='a')
		 .count();

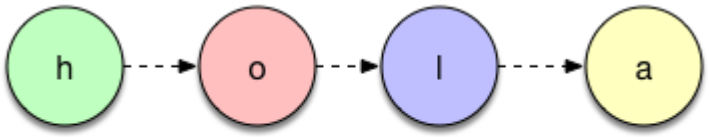
		System.out.println(total);

	}
```

Acabamos de contar el numero de letras ‘a’ que tiene una cadena y lo hemos impreso por la consola.



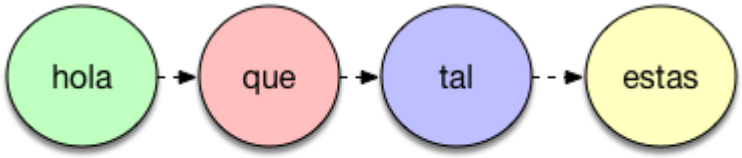
La verdad es que esta muy bien pero es poco útil , no nos engañemos. Lo que nosotros queremos no es un Stream de chars.



Stream caracteres

java stream caracteres

Lo que queremos es un Stream de Strings.



Stream String

Lamentablemente por mucho que revisemos el API de la clase String no lo encontraremos. ¿Donde se encuentra ubicado este método?. Este método se encuentra ubicado en el API de expresiones regulares lo cual tiene también sentido , aunque a veces nos resulte extraño. Vamos a utilizar este API para partir una cadena y filtrarla.

```
package com.arquitecturajava;

import java.util.regex.Pattern;
import java.util.stream.Stream;

public class Principal {

        public static void main (String[] args) {

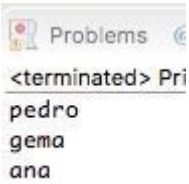
                String texto="nombre:pedro,nombre:gema,nombre:ana";

                Stream<String> bloques=
Pattern.compile(",").splitAsStream(texto);
                bloques.map(cadena->cadena.substring(7,
```

```
cadena.length())).
                forEach(System.out::println);

        }
}
```

En este caso hemos leído una cadena de texto la hemos convertido en un Stream utilizando el api de expresiones regulares partiendola por el caracter “,” . Una vez hecho esto simplemente realizamos una operación de transformación con map y nos quedamos con los nombres que es lo que imprimimos por la consola.



Acabamos de construir un par de ejemplos con Java Stream String:

Otros artículos relacionados:

1. Java 8 FlatMap y Streams
2. El concepto de Java infinite Stream
3. Java 8 Stream y workflows
4. Oracle Streams