«Talento Tech»

Back-End

Java

Clase 03

Buenos Aires Ciudad

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Clase 3: Cadenas y listas

Índice

Objetivos de la Clase

- Comprender la naturaleza de las **cadenas de caracteres** (strings) en Java y las principales operaciones disponibles para manipularlas.
- Aprender métodos comunes de las cadenas, como `length()`, `substring()`, `toUpperCase()`, `toLowerCase()`, `trim()`, `indexOf()`, entre otros.
- Introducir el concepto de **arrays** y comprender sus limitaciones y usos.
- Conocer las **listas dinámicas**, en particular `ArrayList`, y aprender a agregar, eliminar, acceder y modificar elementos de manera flexible.
- Aplicar estos conocimientos en el contexto de TechLab, resolviendo problemas concretos del catálogo de productos.

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Introducción:

¡Les damos la bienvenida nuevamente! Tras aprender sobre variables, operadores, estructuras de control y modularización del código, hoy daremos un paso hacia el manejo de datos textuales y colecciones más complejas. Veremos cómo tratar cadenas de caracteres para limpiarlas, formatearlas y extraer información, y cómo organizar conjuntos de datos en arrays y listas para trabajar con ellos de manera más estructurada y flexible.

Las cadenas (strings) son la base para representar información textual, como nombres de productos o descripciones. Las colecciones (arrays y listas) nos permiten gestionar grupos de elementos: productos, categorías, pedidos. Estos conceptos son fundamentales para construir aplicaciones más robustas y escalables.

Manipulación de cadenas de caracteres

En Java, las cadenas se representan con la clase `String`. Son inmutables, lo que significa que una vez creada una cadena, su contenido no puede modificarse. En su lugar, los métodos que "modifican" cadenas devuelven nuevas instancias con los cambios aplicados.

Algunos métodos comunes:

- `length()`: Retorna la cantidad de caracteres en la cadena.
- `substring(int inicio, int fin)`: Extrae una porción de la cadena desde `inicio` hasta `fin-1`.
- `toUpperCase()` / `toLowerCase()`: Convierte todos los caracteres a mayúsculas o minúsculas.
- `trim()`: Elimina espacios en blanco al principio y al final de la cadena.
- `indexOf(String s)`: Retorna la posición de la primera aparición de `s` dentro de la cadena.
- `replace(String viejo, String nuevo)`: Crea una nueva cadena reemplazando todas las apariciones de `viejo` por `nuevo`.

Estas operaciones resultan esenciales para estandarizar datos textuales.

Métodos comunes de cadenas

Imaginemos que tenemos un nombre de producto `" Café Premium MOLIDO "`. Queremos presentarlo en un formato uniforme: sin espacios extra, con la primera letra en mayúscula y el resto en minúsculas y sin palabras extra.

Secuencia posible:

```
String producto = "  Café Premium MOLIDO  ";

// Eliminar espacios extras

producto = producto.trim(); // "Café Premium MOLIDO"

// Pasar todo a minúsculas

producto = producto.toLowerCase(); // "café premium molido"


// Podríamos luego capitalizar la primera letra manualmente si fuese necesario.

// O reemplazar "molido" por "Molido" con substring y concatenaciones.
```

Estas técnicas nos dan control total sobre la presentación del texto.

Problema: formateo de nombres de productos

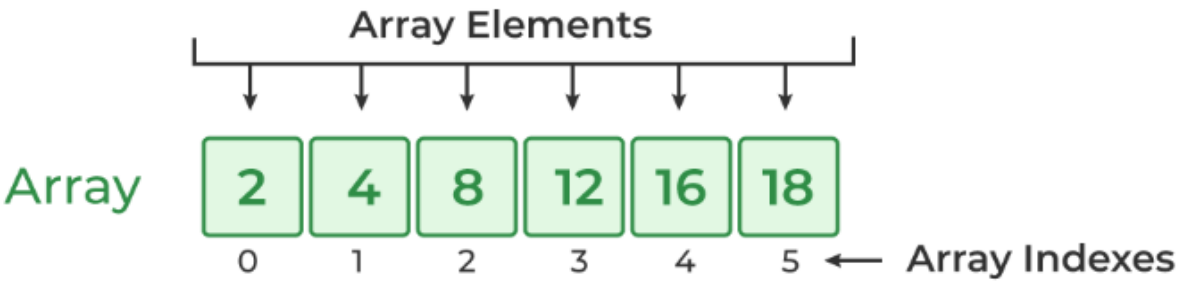
Contexto: Silvia detecta que algunos productos fueron cargados en el sistema con espacios y mayúsculas inconsistentes. Antes de mostrarlos en la tienda online, hay que formatearlos: eliminar espacios al principio y fin, convertirlos a "Title Case" (la primera letra de cada palabra en mayúscula).

Solución propuesta:

```
public static String formatearNombreProducto(String nombre) {

    nombre = nombre.trim().toLowerCase();

    // Dividimos en palabras

    String[] palabras = nombre.split(" ");

    StringBuilder sb = new StringBuilder();

  for (int i = 0; i < palabras.length; i++) {

        if (!palabras[i].isEmpty()) {

            String primeraLetra =
palabras[i].substring(0,1).toUpperCase();

            String resto = palabras[i].substring(1);

            sb.append(primeraLetra).append(resto);

            if (i < palabras.length - 1) {

                sb.append(" ");

            }

        }

    }

    return sb.toString();

}
```

Este método toma un nombre desprolijo y lo devuelve limpio y presentable.

Introducción a arrays y listas.



Cuando necesitamos manejar múltiples valores, como 100 nombres de productos, sería impráctico usar 100 variables independientes. Para esto existen las colecciones.

Arrays

Un array es una estructura de datos de tamaño fijo. Se declara con un tipo, seguido de `[]`, por ejemplo:

```
String[] productos = new String[3];

productos[0] = "Café Premium Molido";

productos[1] = "Té Verde Orgánico";

productos[2] = "Chocolate Amargo 80%";
```

Las limitaciones del array son:

- Tamaño fijo: no se puede agregar ni quitar elementos una vez creado.
- Operaciones como búsqueda o eliminación requieren manejo manual.

Listas y ArrayList

Buenos Aires aprende
Agencia de Habilidades para el Futuro

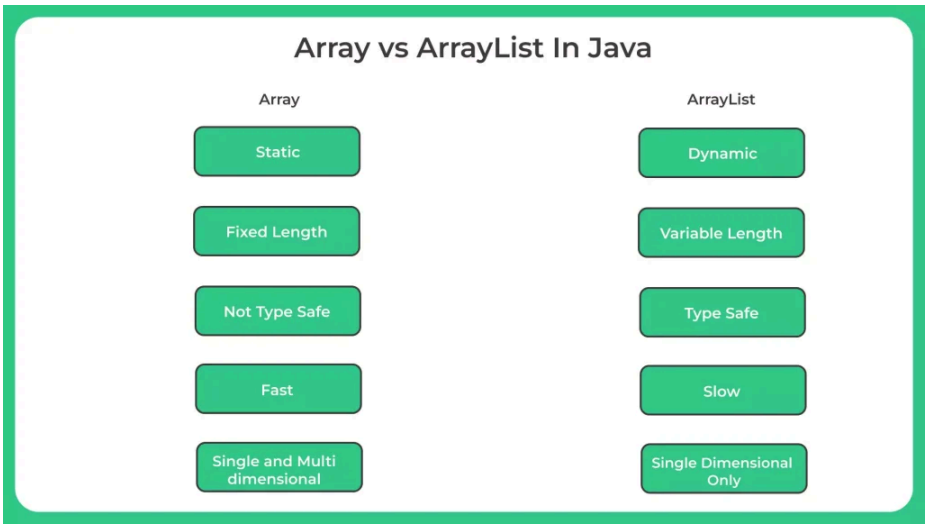
Para superar las limitaciones del array, Java ofrece `ArrayList`, una lista dinámica que crece o decrece según nuestras necesidades. Algunas ventajas:

- Podés agregar (`add()`), eliminar (`remove()`) y acceder (`get()`) elementos fácilmente.
- No tenés que preocuparte por el tamaño fijo.

Ejemplo:

```
import java.util.ArrayList;

ArrayList<String> listaProductos = new ArrayList<>();

listaProductos.add("Café Premium Molido");

listaProductos.add("Té Verde Orgánico");

listaProductos.add("Chocolate Amargo 80%");
```

Se pueden insertar, eliminar, recorrer con bucles `for` o `for-each`, y consultar el tamaño con `size()`.



Buenos Aires aprende
Agencia de Habilidades para el Futuro

Problema: gestión de un catálogo de productos

Contexto: Silvia quiere listar todos los productos disponibles. Inicialmente, Matías y Sabrina piensan en un array fijo, pero se dan cuenta de que el catálogo crecerá. Comienzan usando un array, notan su rigidez y luego considerarán `ArrayList`.

Solución con Arrays (primera aproximación):

```
String[] productos = new String[3];

productos[0] = "Café Premium Molido";

productos[1] = "Té Verde Orgánico";

productos[2] = "Chocolate Amargo 80%";


for (int i = 0; i < productos.length; i++) {

   System.out.println("Producto: " + productos[i]);

}
```

Esta solución funciona con 3 productos pero ¿y si necesitamos agregar un cuarto producto?

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Uso de ArrayList y sus métodos

`ArrayList` es una clase genérica. Para listas de Strings:

ArrayList es básicamente un arreglo de tamaño variable. Casi siempre, vas a querer usar ArrayList en lugar de arreglos regulares, ya que brindan muchos métodos útiles.

```
ArrayList<String> listaProductos = new ArrayList<>();

listaProductos.add("Café Premium Molido");

listaProductos.add("Té Verde Orgánico");

listaProductos.add("Chocolate Amargo 80%");
```

Métodos comunes de `ArrayList`:

- `add(elemento)`: Agrega un elemento al final.
- `remove(indice)` o `remove(objeto)`: Elimina un elemento por índice o por objeto.
- `get(indice)`: Obtiene el elemento en la posición dada.
- `size()`: Retorna la cantidad de elementos.
- `contains(objeto)`: Indica si la lista contiene el objeto especificado.
- `clear()`: Vacía la lista.

Podés iterar la lista con un `for-each`:

```
for (String producto : listaProductos) {

   System.out.println(producto);

}
```

Problema: actualización dinámica del catálogo

Contexto: Ahora Silvia agrega un nuevo producto "Café Descafeinado" y decide que "Chocolate Amargo 80%" ya no estará disponible. Además, el cliente pregunta si existe cierto té especial. Con un `ArrayList`, Matías y Sabrina pueden hacerlo sin problemas.

Solución con `ArrayList`:

```
ArrayList<String> listaProductos = new ArrayList<>();

listaProductos.add("Café Premium Molido");

listaProductos.add("Té Verde Orgánico");

listaProductos.add("Chocolate Amargo 80%");

// Agregar nuevo producto

listaProductos.add("Café Descafeinado");

// Eliminar Chocolate Amargo

listaProductos.remove("Chocolate Amargo 80%");

// Verificar si existe Té Verde Orgánico

if (listaProductos.contains("Té Verde Orgánico")) {

   System.out.println("El catálogo contiene Té Verde Orgánico");

} else {

   System.out.println("No se encuentra ese producto.");

}
```

De esta forma, la lista se adapta a los requerimientos cambiantes del catálogo.

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Materiales y recursos adicionales

- Documentación Oficial de Java: String
- Documentación Oficial de Java: ArrayList
- Tutoriales recomendados en YouTube sobre manipulación de cadenas y uso de colecciones.

Preguntas para reflexionar

- ¿Por qué las cadenas son inmutables en Java y qué ventajas o inconvenientes trae esto?
- ¿Cuándo convendría usar un array en lugar de un `ArrayList` y viceversa?
- ¿Cómo la posibilidad de manipular cadenas y colecciones de manera flexible contribuye a la escalabilidad de un proyecto como el de TechLab?

Próximos pasos

En la próxima clase, seguiremos avanzando en la construcción de nuestro backend. Con el conocimiento de **cadenas y colecciones**, estaremos preparados para manejar datos más complejos, integrarlos con lógica de negocio más refinada y encarar desafíos mayores.

¡Felicitaciones por llegar hasta aquí! El manejo de cadenas y listas es fundamental en cualquier aplicación, y ahora contás con mayor preparación para enfrentar las necesidades del proyecto en TechLab.

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Situación Inicial en TechLab

Talento Lab

Silvia, la Product Owner, tiene en el **backlog*** una serie de pedidos del cliente Sibelius:

- Los nombres de algunos productos vienen con espacios extra o en mayúsculas y minúsculas inconsistentes. Es necesario limpiar y formatear esos nombres antes de mostrarlos al cliente.
- El catálogo de productos empieza a crecer y almacenar los nombres en variables sueltas ya no es práctico. Necesitan una estructura que permita agruparlos. Primero probarán con arrays, pero pronto notarán las limitaciones.
- Finalmente, para agregar y quitar productos fácilmente del catálogo, un `ArrayList` será la solución. Además, Sabrina y Matías quieren filtrar, buscar y modificar productos de forma dinámica.

***Backlog:** *(El backlog es una lista de trabajo ordenado por prioridades para el equipo de desarrollo que se obtiene de su hoja de ruta y sus requisitos.)*

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Ejercicios prácticos.

Silvia

Product Owner

Silvia, la Product Owner, tiene en el **backlog*** una serie de pedidos del cliente:

- Los nombres de algunos productos vienen con espacios extra o en mayúsculas y minúsculas inconsistentes. Es necesario limpiar y formatear esos nombres antes de mostrarlos al cliente.
- El catálogo de productos empieza a crecer, y almacenar los nombres en variables sueltas ya no es práctico. Necesitan una estructura que permita agruparlos. Primero probarán con arrays, pero pronto notarán las limitaciones.
- Finalmente, para agregar y quitar productos fácilmente del catálogo, un `ArrayList` será la solución. Además, Sabrina y Matías quieren filtrar, buscar y modificar productos de forma dinámica.

Para poder llegar con los tiempos de entrega dispuestos por Silvia, será necesario realizar las siguientes acciones:

1. **Manipulación de cadenas:**
   - Dada una cadena `" té CHAi "`, formateala para que quede `"Té Chai"`.
   - Mostrá su longitud, su primera letra y verificá si contiene la palabra `"Chai"`.
2. **Arrays:**
   - Creá un array de 5 productos.
   - Imprimí sus elementos y luego intentá agregar un sexto producto (analizá el resultado).
3. **ArrayList:**
   - Creá un `ArrayList<String>` para productos.
   - Agregá varios productos, eliminá uno, verificá si otro existe y luego imprimí la lista final.

4. **Combinar cadenas y listas:**
   - Tené una lista de productos con nombres desprolijos.
   - Creá un método que recorra la lista y aplique el formateo a cada nombre, volviendo a imprimir la lista con nombres prolijos.

Buenos Aires Ciudad