


Open · Opened 2 weeks ago by  **Rubén Montero**

Almacenando preferencias en atributos

Resumen

- Añadiremos dos atributos privados y dos **getters** a **HomeCinemaPreferences**
- Escribiremos un nuevo método privado que *parsee* una línea del fichero
- Usaremos el método para *parsear* todas las líneas del fichero

Descripción

Continuemos trabajando en nuestro **HomeCinemaPreferences**.

Ahora, vamos a añadir dos atributos privados a la clase:

```
private String username;  
private boolean darkModePreferred;
```

Añadamos también dos **getters** llamados **getUsername** e **isDarkModePreferred**. Recuerda que puedes hacerlo automáticamente.

Un método que interprete una línea

Vamos a añadir un nuevo método **privado** a la clase:

```
private void parseLine(String line) {  
  
}
```

Lo invocaremos desde el método **constructor** y *delegaremos* en él la responsabilidad de interpretar la línea.

```
public HomeCinemaPreferences() {  
    String file ="assets\\cinemaPrefs.txt";  
    try {  
        FileReader fileReader = new FileReader(file);  
        BufferedReader bufferedReader = new BufferedReader(fileReader);  
        String newLine = null;  
        do {  
            newLine = bufferedReader.readLine();  
            System.out.println(newLine);  
+            // Invocamos el método que trabaja línea a línea  
+            parseLine(newLine)  
        } while (newLine != null);  
    } catch (IOException e) {  
        throw new RuntimeException(e);  
    }  
}
```

Este método recibe por parámetro un **String**:

1. Si es la preferencia *username*, la guarda en el atributo **username**
2. Si es la preferencia *prefersDarkMode*, la guarda en el atributo **darkModePreferred**

Condición de guarda

El primer paso será verificar que el parámetro tipo **String** que recibimos **no** es **nulo**.

En caso de que lo sea, *retornaremos* de forma *temprana*.

Es lo que se conoce como una condición de guarda:

```
private void parseLine(String line) {  
    if (line == null) {  
        return;  
    }  
}
```

```
    }  
    // ...  
}
```

Usando `.split()`, de `String`

Ahora vamos a separar la línea en 2 partes. Esto lo conseguimos con `.split()`, que *recibe un carácter* y devuelve el `String` *troceado* en piezas separadas por ese *carácter*.

```
private void parseLine(String line) {  
    if (line == null) {  
        return;  
    }  
    String[] separatedString = line.split("=");  
    String firstHalf = separatedString[0];  
    String secondHalf = separatedString[1];  
    // ...  
}
```

En este momento debemos ser capaces de responder a:

- ¿Qué valor tiene `firstHalf` para la **primera** línea del fichero? ¿Y `secondHalf` ?
- ¿Qué valor tiene `firstHalf` para la **segunda** línea del fichero? ¿Y `secondHalf` ?

Si no es así, escribe sentencias `System.out.println` que arrojen algo de luz.

Si la línea empieza por `username=` ...

Entonces, al atributo `this.username` queremos asignarle el valor que aparece en el fichero.

Esto se escribe en código así:

```
if (firstHalf == "username") { // OJO: MAL  
    this.username = secondHalf;  
}
```

`.equals()`

¡Un momento! **No** podemos comparar `Strings` (ni cualquier `Objeto` complejo) usando `==`.

Está **mal**.

En Java, `==` sólo sirve para **tipos primitivos**. Los objetos se comparan con `.equals()`. Así:

```
- if (firstHalf == "username") { // OJO: MAL  
+ if (firstHalf.equals("username")) { // Bien  
    this.username = secondHalf;  
}
```

Y si la línea empieza por `prefersDarkMode=` ...

Haremos algo parecido:

```
if (firstHalf.equals("username")) {  
    this.username = secondHalf;  
}  
if (firstHalf.equals("prefersDarkMode")) {  
    this.darkModePreferred = secondHalf;  
}
```

Parece que IntelliJ IDEA marca un **error**, ¿no?

Es esperable.

- `secondHalf` es de tipo `String`.
- `this.darkModePreferred` es de tipo `boolean`

¡No podemos asignarlos!

Pensemos en una solución

Seremos capaces de llegar a ella.

Una vez corrijamos este problema, podemos verificar los valores de los atributos de `HomeCinemaPreferences` con `.getUsername()` y `.isDarkModePreferred()`.

Por último

Verifica que el `test` funciona correctamente.

Haz `commit` y `push` para subir los cambios al repositorio.



Rubén Montero [@ruben.montero](#) changed milestone to [%Sprint 1](#) 2 weeks ago



Ania Blanco [@ania.blanco](#) mentioned in commit [3042ac1d](#) 2 weeks ago