

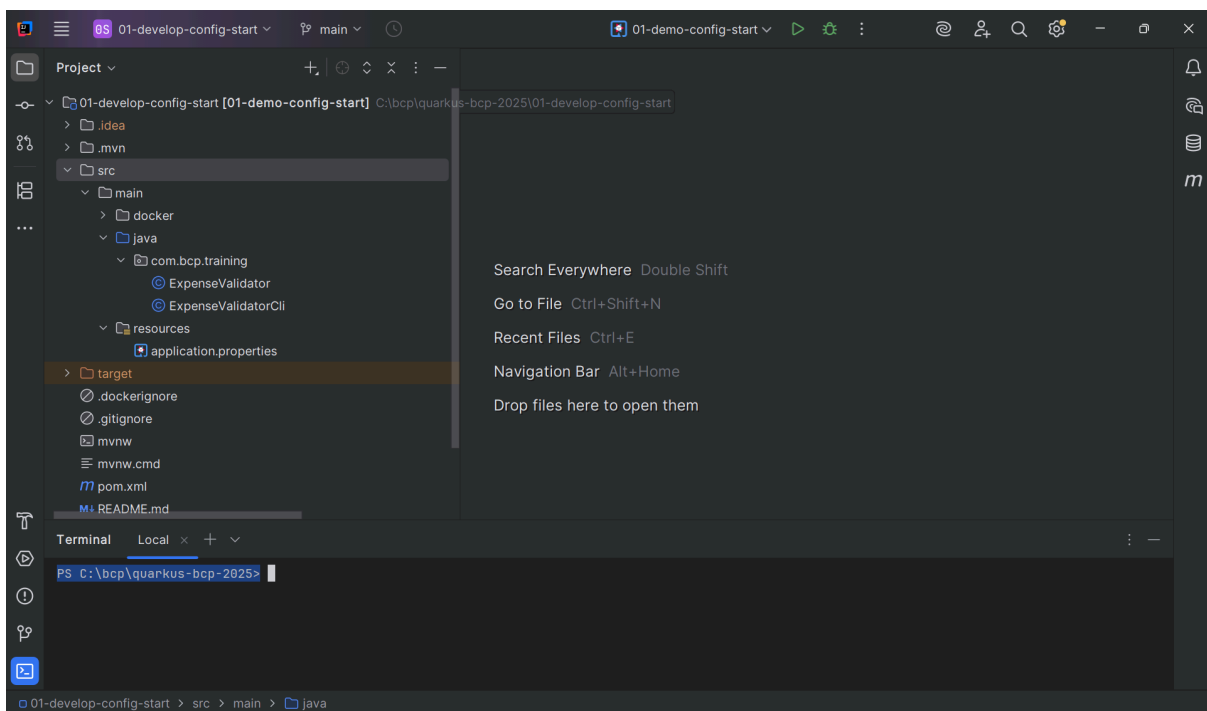
LAB 2: QUARKUS CONFIG

Autor: José Díaz

Apoyo: Juan Ramirez

Github Repo: <https://github.com/joedayz/quarkus-bcp-2025.git>

1. Cargar en su IDE el proyecto **01-develop-config-start**:



2. Examina la clase `com.bcp.training.ExpenseValidatorCli`.
 - Esta clase define una aplicación de línea de comandos que recibe un argumento.
 - La aplicación utiliza la clase `ExpenseValidator` para determinar si el argumento proporcionado está dentro de un rango definido de valores enteros y muestra un mensaje en la salida de la consola.
3. Ejecuta el comando `mvn quarkus:dev` para iniciar la aplicación Quarkus en modo de desarrollo.

NOTA: No se ve el número que se ingresa, pero, coloca 255. El resultado debería ser lo que se muestra a continuación.

Range - High: 1000

Range - Low: 250

Valid amount: 255

5. Reemplaza los valores codificados de forma fija por propiedades de configuración.

5.1 Abre la clase **ExpenseValidator** e importa la anotación **@ConfigProperty**.

```
© ExpenseValidator.java x
1 package com.bcp.training;
2
3 import jakarta.enterprise.context.ApplicationScoped;
4 import org.eclipse.microprofile.config.inject.ConfigProperty;
5
6 @ApplicationScoped 1 usage  Jose Amadeo Diaz *
7 public class ExpenseValidator {
8
9     @ConfigProperty(name = "debug-enabled", defaultValue = "false") 1 usage
10     boolean debug;
11
12     private static final int RANGE_HIGH = 1000; 2 usages
13
14     private static final int RANGE_LOW = 250; 2 usages
15
16     public void debugRanges() { 1 usage  Jose Amadeo Diaz
17         System.out.println("Range - High: " + RANGE_HIGH);
18         System.out.println("Range - Low: " + RANGE_LOW);
19     }
20
21     public boolean isValidAmount(int amount) { 1 usage  Jose Amadeo Diaz *
22         if (debug) {
23             debugRanges();
24         }
25
26         return amount >= RANGE_LOW && amount <= RANGE_HIGH;
27     }
28 }
29
```

```
@ConfigProperty(name="debug-enabled",
```

```
defaultValue = "false")  
boolean debugEnabled;
```

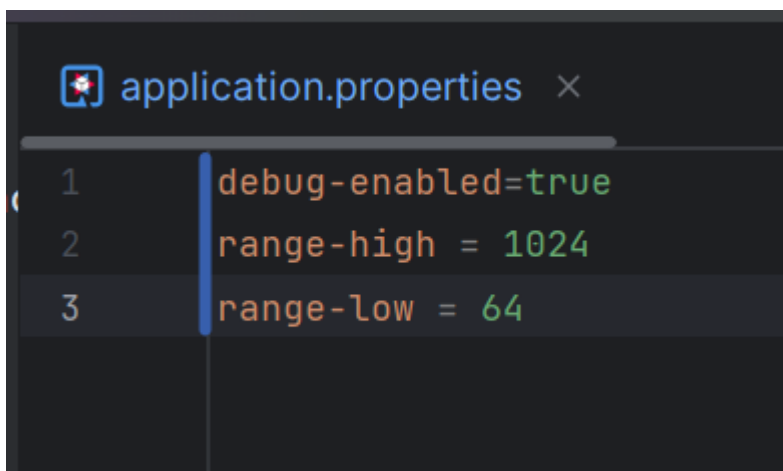
6. Realizar el cambio para las demás propiedades y asegurate de que termine así:

```
package com.bcp.training;  
  
import jakarta.enterprise.context.ApplicationScoped;  
import org.eclipse.microprofile.config.inject.ConfigProperty;  
  
@ApplicationScoped 1 usage  👤 Jose Amadeo Diaz *  
public class ExpenseValidator {  
  
    @ConfigProperty(name = "debug-enabled", defaultValue = "false") 1 usage  
    boolean debug;  
  
    @ConfigProperty(name = "range-high") 2 usages  
    int targetRangeHigh;  
  
    @ConfigProperty(name = "range-low") 2 usages  
    int targetRangeLow;  
  
    public void debugRanges() { 1 usage  👤 Jose Amadeo Diaz *  
        System.out.println("Range - High: " + targetRangeHigh);  
        System.out.println("Range - Low: " + targetRangeLow);  
    }  
  
    public boolean isValidAmount(int amount) { 1 usage  👤 Jose Amadeo Diaz *  
        if (debug) {  
            debugRanges();  
        }  
        return amount >= targetRangeLow && amount <= targetRangeHigh;  
    }  
}
```

```
@ConfigProperty(name="range-high")
```

```
int targetRangeHigh;  
@ConfigProperty(name="range-low")  
int targetRangeLow;
```

7. Abre el archivo `src/main/resources/application.properties`. Luego define las propiedades `range-high = 1024` y `range-low = 64`.



The screenshot shows a code editor window titled "application.properties" with a close button. The file contains three lines of configuration: `debug-enabled=true`, `range-high = 1024`, and `range-low = 64`. Line numbers 1, 2, and 3 are visible on the left side of the editor.

```
debug-enabled=true  
range-high=1024  
range-low=64
```

8. Regresa a la ventana del terminal, presiona **Espacio** para reiniciar la aplicación. Luego verifica que la salida de la consola no muestre los mensajes de depuración del rango:

[illegible]

...output omitted...

Valid amount: 255

...output omitted...

El método **ExpenseValidator#isValidAmount()** utiliza el valor de la propiedad **debug-enabled** para imprimir mensajes de depuración.

En ausencia de un valor para esa propiedad, el valor predeterminado definido en el código es **false**, y la aplicación no ejecuta el método **ExpenseValidator#debugRanges()**.

9. Agrupa las propiedades de configuración en una sola interfaz.

9.1. Crea una interfaz con la configuración mínima de metadatos.

- Llama a la interfaz **ExpenseConfiguration**.
- Crea la interfaz en el paquete **com.bcp.training**.
- Anota la interfaz con la anotación **io.smallrye.config.ConfigMapping**.
- Define **expense** como el prefijo de mapeo.

- Usa la anotación **io.smallrye.config.WithDefault** para definir **false** como el valor predeterminado de la propiedad de configuración **debug-enabled**.

```
package com.bcp.training;

import io.smallrye.config.ConfigMapping;
import io.smallrye.config.WithDefault;

@ConfigMapping(prefix = "expense")
public interface ExpenseConfiguration {
    @WithDefault("false")
    boolean debugEnabled();
    int rangeHigh();
    int rangeLow();
}
```

10. Abre la clase **ExpenseValidator** y reemplaza la importación de **@ConfigProperty** por la importación de **@Inject**.

```
© ExpenseValidator.java ×
1 package com.bcp.training;
2
3 import jakarta.enterprise.context.ApplicationScoped;
4 import jakarta.inject.Inject;
5
6 @ApplicationScoped 1 usage  Jose Amadeo Diaz *
7 public class ExpenseValidator {
8
9     @Inject 5 usages
10    ExpenseConfiguration config;
11
12    public void debugRanges() { 1 usage  Jose Amadeo Diaz *
13        System.out.println("Range - High: " + config.rangeHigh());
14        System.out.println("Range - Low: " + config.rangeLow());
15    }
16    public boolean isValidAmount(int amount) { 1 usage  Jose Amadeo Diaz *
17        if (config.debugEnabled()) {
18            debugRanges();
19        }
20        return amount >= config.rangeLow() && amount <= config.rangeHigh();
21    }
22 }
23
```

11. Abre el archivo **application.properties** y agrega el prefijo **expense** a todas las propiedades de configuración.

```
expense.range-high = 1024
expense.range-low = 64
expense.debug-enabled = true
```

```
application.properties ×
1 expense.range-high = 1024
2 expense.range-low = 64
3 expense.debug-enabled = true
4
```


Luego agrega un método para la propiedad **debug-message** que retorne un objeto **Optional<String>**.

```
① ExpenseConfiguration.java ×
1 package com.bcp.training;
2
3 import io.smallrye.config.ConfigMapping;
4 import io.smallrye.config.WithDefault;
5
6 import java.util.Optional;
7
8 @ConfigMapping(prefix = "expense") 1 usage new *
9 ↵ public interface ExpenseConfiguration {
10     @WithDefault("false") 2 usages new *
11     boolean debugEnabled();
12
13     Optional<String> debugMessage(); no usages new *
14
15     int rangeHigh(); 3 usages new *
16     int rangeLow(); 3 usages new *
17 }
```

14. Abre la clase **ExpenseValidator** y actualiza el método **debugRanges()**.
Reemplaza el contenido del método con una impresión del valor de la propiedad **debug-message**.

```
public void debugRanges() {
    config.debugMessage().ifPresent(System.out::println);
}
```

La salida de la consola muestra los siguientes mensajes:

```

Terminal Local x + v
- / / / / / _ _ | / , _ / / \ \
-- \ \ \ \ _ _ _ / / | _ / | _ / | _ \ _ _ _ / _ _ /
2025-09-02 17:19:35,280 INFO [io.quarkus] (Quarkus Main Thread) expense-validator 1.0.0-SNAPSHOT on JVM (powered by Quarkus 3.10.0)
2025-09-02 17:19:35,282 INFO [io.quarkus] (Quarkus Main Thread) Profile dev activated. Live Coding activated.
2025-09-02 17:19:35,282 INFO [io.quarkus] (Quarkus Main Thread) Installed features: [cdi]
Range [64, 1024]
Valid amount: 255
2025-09-02 17:19:35,289 INFO [io.quarkus] (Quarkus Main Thread) expense-validator stopped in 0.003s

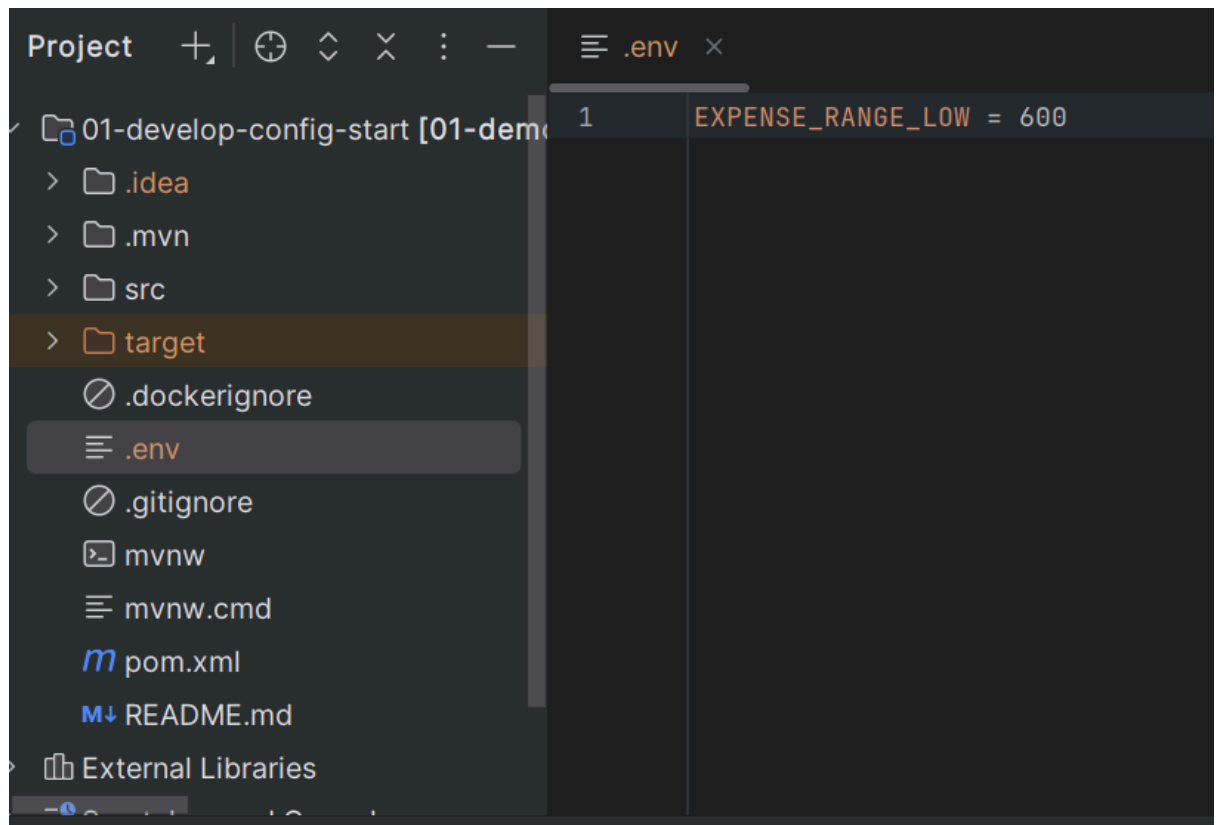
--
Tests paused
Press [space] to restart, [e] to edit command line args (currently '255'), [r] to resume testing, [o] Toggle test output for more options>

```

Range [64, 1024]

17. Usa un archivo de entorno (.env) para definir y sobrescribir propiedades de configuración.

17.1. Crea un archivo **.env** en la raíz del proyecto y sobrescribe la propiedad **expense.range-low**. Establece **600** como el valor de la propiedad.

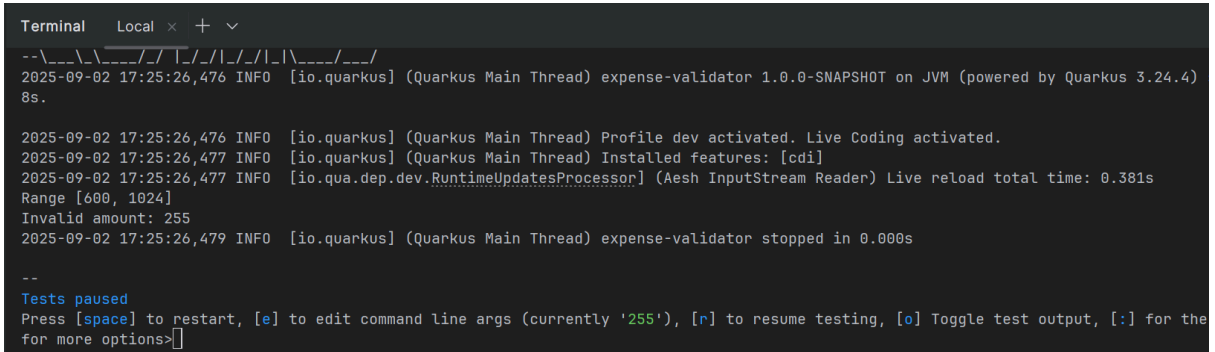


```
EXPENSE_RANGE_LOW = 600
```

17.2 Regresa a la ventana del terminal, presiona **Espacio** para reiniciar la aplicación y verifica los cambios.

La salida de la consola muestra los siguientes mensajes:

```
...output omitted...  
Range [600, 1024]  
Invalid amount: 255  
...output omitted...
```



Range [700, 1024]
Invalid amount: 255



```
Terminal  Local x + v
[INFO] -----
[INFO] Total time: 5.708 s
[INFO] Finished at: 2025-09-02T17:30:14-05:00
[INFO] -----
--/  _  \// //  _  \// // // //  _  \
-/ // // //  _  \// // // //  _  \
--\  _  \// //  _  \// // // //  _  \
2025-09-02 17:30:16,110 INFO  [io.quarkus] (main) expense-validator 1.0.0-SNAPSHOT on JVM (powered by Quarkus 3.24.4) started in 0.239s.
2025-09-02 17:30:16,111 INFO  [io.quarkus] (main) Profile dev activated.
2025-09-02 17:30:16,111 INFO  [io.quarkus] (main) Installed features: [cdi]
Range [700, 1024]
Invalid amount: 255
2025-09-02 17:30:16,124 INFO  [io.quarkus] (main) expense-validator stopped in 0.008s
PS C:\bcp\quarkus-bcp-2025\01-develop-config-start>
```

Enjoy!

Joe