Atelier 3 – Les expressions Lambda / la programmation fonctionnelle

Dans le code du fichier d'aide de Volley , on retrouve la structure de code suivante :

*// Instantiate the RequestQueue.*

**RequestQueue** queue **=** **Volley.**newRequestQueue**(this);**

**String** url **=** "https://www.google.com"**;**

*// Request a string response from the provided URL.*

**StringRequest** stringRequest **=** **new** **StringRequest(Request.**Method**.**GET**,** url**,**

**new** **Response.**Listener**<String>()** **{**

**@Override**

**public** **void** **onResponse(String** response**)** **{**

*// Display the first 500 characters of the response string.*

textView**.**setText**(**"Response is: " **+** response**.**substring**(**0**,**500**));**

**}**

**},** **new** **Response.**ErrorListener**()** **{**

**@Override**

**public** **void** **onErrorResponse(VolleyError** error**)** **{**

textView**.**setText**(**"That didn't work!"**);**

**}**

**});**

*// Add the request to the RequestQueue.*

queue**.**add**(**stringRequest**);**

Le code est un peu compliqué; on est aux prises avec deux écouteurs anonymes mettant en œuvre des [interfaces fonctionnelles](https://blog.ippon.fr/2014/03/18/java-8-interfaces-fonctionnelles/#:~:text=Le%20principe%20d'interface%20fonctionnelle,du%20code%20est%20grandement%20am%C3%A9lior%C3%A9e%20.)

On peut modifier / simplifier le code en utilisant une expression lambda.

**Une expression lambda est caractérisée par la formulation :**

**paramètre -> expression / retour**

**ou**

**( paramètre1, paramètre2 ) -> expression / retour**

Une expression lambda sert à permet d’écrire du code plus court, c’est une fonction anonyme ( pas de type de retour explicite, pas de modificateur d’accès, pas de nom ) . Plus précisément…

* C’est un raccourci syntaxique qui permet de définir une méthode exactement à l’endroit où elle est utilisée, ce raccourci est particulièrement utile quand le traitement à faire le sera une seule fois, il n’y a donc par d’avantage à coder une méthode « normale ».
* **Permet de remplacer une classe qui met en œuvre une interface fonctionnelle**

On pourrait donc remplacer le code de la première page par :

EX 1. : on veut gérer le clic d’un bouton

EX 2 La méthode forEach de la classe Vector (api 24 ou plus pour l’utiliser )

* Bon exemple car cette méthode prend en paramètre un objet mettant en oeuvre une interface fonctionnelle ( Consumer )

data.forEach((n) -> System.out.println(n));

Site très complet : <https://www.jmdoudoux.fr/java/dej/chap-lambdas.htm>