



Projet, partie 1 : compositeur de menus simplistes

Thibaut Septon

# [INFOM451] Conception d'applications mobiles

October 27, 2020

Jacquet Jean-Marie, Vanhoof Wim

# Table des matières

Introduction	2
Les différentes parties	2
Les entrées utilisateurs . . . . .	3
L'algorithme de composition . . . . .	3
Difficultés et remarques	4

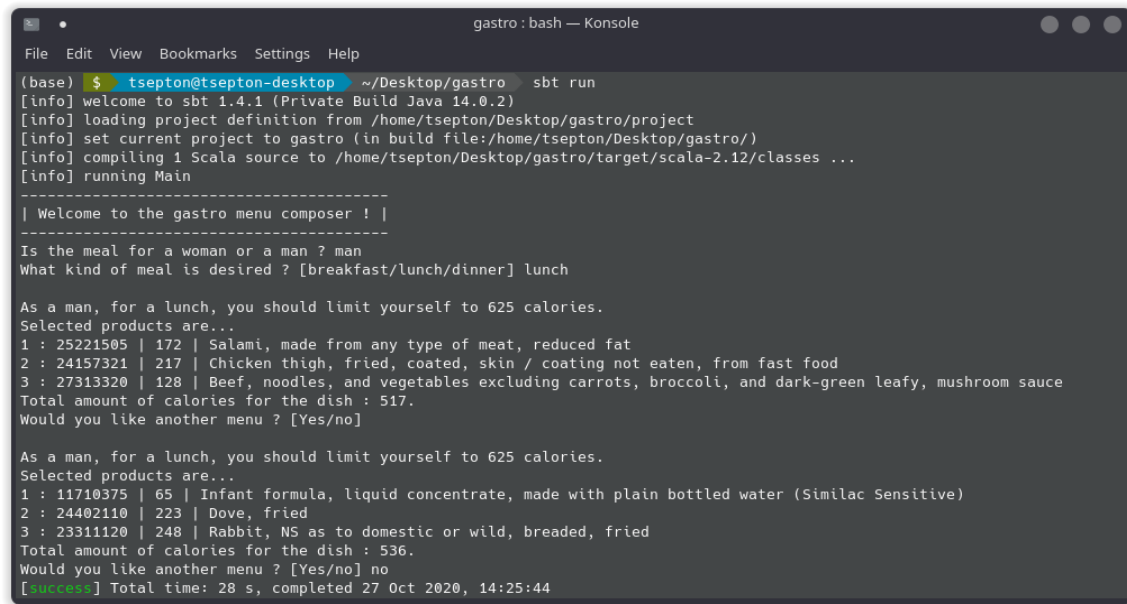


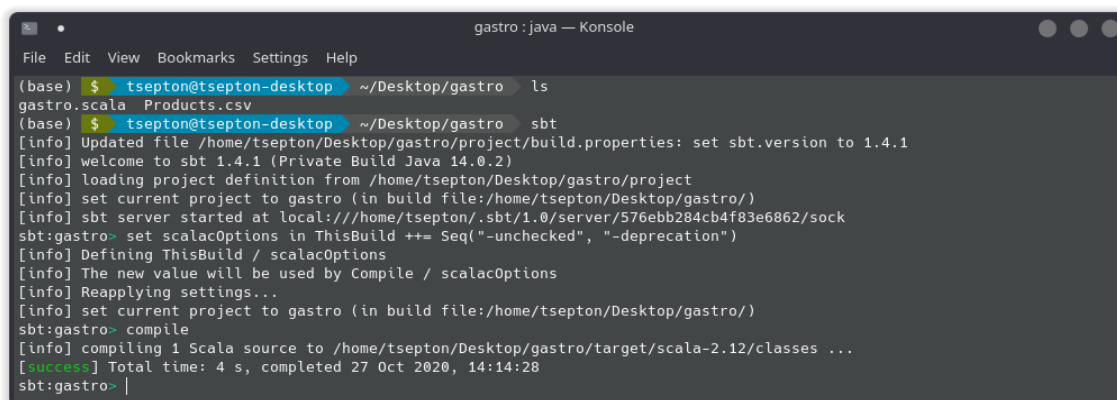
Figure 1: Aperçu du programme.

# Introduction

Etant donné que je ne suis pas un expert en alimentation, et que l'objectif transversale du projet était de montrer la compréhension des concepts du langage Scala (à savoir un mélange de fonctionnel et d'orienté objet), mon script pourra vous paraître simpliste.

En effet, il ne se base que sur le nombre de calories présent dans le repas afin de déterminer si celui-ci convient ou pas. Cependant, j'ai préféré garder un programme simple afin de mettre en oeuvre les deux paradigmes de Scala et ainsi mieux appréhender l'apprentissage de ce langage.

Le code étant lisible et commenté, je ne m'attarderai pas sur l'aspect technique du programme dans ce rapport, mais seulement sur la partie logique.



```
gastro:java — Konsole
File Edit View Bookmarks Settings Help
(base) $ tsepton@tsepton-desktop ~/Desktop/gastro ls
gastro.scala  Products.csv
(base) $ tsepton@tsepton-desktop ~/Desktop/gastro sbt
[info] Updated file /home/tsepton/Desktop/gastro/project/build.properties: set sbt.version to 1.4.1
[info] welcome to sbt 1.4.1 (Private Build Java 14.0.2)
[info] loading project definition from /home/tsepton/Desktop/gastro/project
[info] set current project to gastro (in build file:/home/tsepton/Desktop/gastro/)
[info] sbt server started at local:///home/tsepton/.sbt/1.0/server/576ebb284cb4f83e6862/sock
sbt:gastro> set scalacOptions in ThisBuild += Seq("-unchecked", "-deprecation")
[info] Defining ThisBuild / scalacOptions
[info] The new value will be used by Compile / scalacOptions
[info] Reapplying settings...
[info] set current project to gastro (in build file:/home/tsepton/Desktop/gastro/)
sbt:gastro> compile
[info] compiling 1 Scala source to /home/tsepton/Desktop/gastro/target/scala-2.12/classes ...
[success] Total time: 4 s, completed 27 Oct 2020, 14:14:28
sbt:gastro> |
```

Figure 2: La compilation n'émet aucun *Warning*.

## Les différentes parties

Afin d'expliciter la logique de mon programme, je vous propose de suivre l'exécution de la méthode *main*. Nous commençons donc par charger les produits en mémoire, et nous en profitons pour les trier (ce qui n'a, pour cette échéance ci, aucun intérêt. Cependant, j'aime à croire que je pourrais améliorer la vitesse d'exécution de l'algorithme proposé sur base de ça pour la prochaine échéance. C'est, de plus, l'occasion d'utiliser une méthode de haut niveau ainsi qu'une fonction anonyme).

## Les entrées utilisateurs

Nous arrivons ensuite à la première partie du programme, à savoir, les informations provenant de l'utilisateur. Puisque nous cherchons à calculer la quantité de calories maximum que celui-ci doit ingérer par repas, nous devons donc savoir s'il s'agit d'un homme ou d'une femme, en plus du type de repas (déjeuner, dîner ou souper), mais nous y reviendrons plus tard ; concentrons nous dans un premier temps sur le sexe de l'utilisateur. Pour ce faire, j'ai donc décidé de déclarer un *Trait, Human*, et de faire ainsi jouer l'héritage de Scala en déclarant deux classes implémentant ces traits (à savoir *Man* et *Woman*). La même logique fut utilisé pour le type de repas.

Sur bases de ces informations, nous pouvons obtenir la quantité de calories maximum par type de repas. Pour ce faire, je me suis basé sur une règle explicitant qu'il faille manger 1/6 des calories de la journée (2500 pour un homme, 2000 pour une femme) au matin, 1/4 lors du dîner et 1/4 au souper (le tiers restant étant ingéré via les gourmandises et les boissons).

Maintenant que nous avons représenté le sexe et le type de repas, nous devons demander à l'utilisateur de nous indiquer ces informations. C'est pourquoi, en suivant la même nomenclature que le singleton déjà déclaré dans la *code base* (*GastroExtractor*), j'ai implémenté l'objet *GastroUser*. Les méthodes récursives *get\_sex* et *get\_meal*, sur base de *pattern matching*, nous permettent ainsi de récupérer ces informations.

## L'algorithme de composition

Une fois ces divers informations récupérées, nous pouvons maintenant nous atteler à la composition du menu. C'est pourquoi nousinstancions la classe *MenuComposer*. Dedans, nous retrouvons 4 méthodes privées en plus de la principale, à savoir *compose*. Ici, nous indiquons à l'utilisateur la valeur maximum qu'il devrait respecter, puis nous lui indiquons un mélange de 3 ingrédients choisis via la méthode *not\_so\_random\_products*. Cette dernière est très simple, elle ne fait que de choisir 3 produits aléatoirement, mais ne dépassant pas - individuellement - le maximum autorisé pour le repas. Si les 3 produits pris ensembles excèdent le nombre maximum de calories autorisées, celle-ci se *ré-appelle* elle même, autrement, elle aura pour valeur de retour ces trois produits choisis.

Nous finissons par présenter les 3 produits à l'utilisateur et lui demandons s'il veut obtenir un autre résultat. A noter que je me suis permis de remplacer la boucle *while* par une fonction récursive afin d'emprunter moins à la programmation déclarative.

## Difficultés et remarques

Dans un premier lieu, j'aimerais ajouter que mon algorithme est loin d'être optimal. Il est clair qu'il aurait pu être plus poussé. J'aimerais continuer d'apprendre Scala en proposant un algorithme modifié pour la deuxième échéance du projet.

Sinon, aucune difficulté majeure ne s'est fait ressentir une fois Scala pris en main. Cependant, j'aimerais émettre une remarque concernant le projet : Le fait de devoir se plonger dans un code déjà existant n'est, pour moi, pas un problème mais il aura tendance à guider et influencer l'entièreté de nos décisions prises par la suite.

Ainsi, si le but du projet est de voir si l'étudiant a bien compris les principes vu au cours théorique, pourquoi ne pas proposer d'implémenter un projet au choix, *from scratch* ? En effet, trouver une idée sur base des contraintes déjà émises est, au final, ce qui m'aura pris le plus de temps, l'inspiration ne venant pas.