|  |  |
| --- | --- |
| structure mvc  Projet 9 – Le Baluchon | Rapport  Présentation de la structure MVC et comment communiquent-ils entre eux.  Mickaël HORN  Etudiant sur le parcours “Développeur d’applications iOS” - OpenClassrooms |

# Schéma

# Détail

Comme nous pouvons le constater, il existe trois principaux MVC dans l’application LeBaluchon.

Nous allons regarder en détail chacun d’entre eux.

## 1 – MVC Exchange

Tout d’abord, le MVC concernant la fonctionnalité du taux de change.

Quand l’utilisateur arrive sur l’application, c’est le premier écran auquel il a à faire.

Dès son arrivée, il se voit afficher le taux de change EUR vers USD, et vice versa.

Il peut également rafraîchir ce taux en appuyant sur le bouton RATE.

La Vue communique alors avec le Contrôleur via une Action.

Le Contrôleur (ExchangeVC) va alors demander au Model (ExchangeService) le nouveau taux de change.

ExchangeService va alors lancer l’appel API grâce à ses fichiers de supports tels que APIKeys (contenant les clés API) et ExchangeResponse (qui contient le format JSON du résultat que l’on attend, nécessaire pour la réception de celui-ci).

Une fois le résultat obtenu, le Model l’envoi au Contrôleur via un callback.

Le Contrôleur va finalement, au moyen d’Outlets, afficher les nouveaux taux de change.

L’utilisateur peut bien évidemment demander, via un montant en euros, son équivalent en dollars grâce au bouton CALCULATE.

Le contrôleur va alors se servir du taux de change précédemment obtenu et afficher le résultat du calcul à l’utilisateur.

## 2 – MVC Traduction

Cette fois, l’utilisateur rentre le texte de son choix en français dans une textView et appuie sur le bouton TRANSLATE pour obtenir la traduction en anglais (l’inverse est également possible en cliquant sur le bouton avec les flèches afin d’échanger les langues).

La Vue communique avec le Contrôleur (TraductionVC) via une Action lancée par le bouton CALCULATE pour demander la traduction, en transmettant le contenu de la textView.

Le Contrôleur réceptionne l’Action et l’envoie au Model (TraductionService) qui va réaliser l’appel API.

Pour se faire, comme pour les autres MVC, il disposera du fichier APIKeys pour sa clé API et du fichier TraductionResponse afin d’assurer la bonne réception du JSON.

La traduction étant obtenue, le Model le renvoi au Contrôleur au moyen d’un callback, qui va afficher grâce à un Outlet, la traduction que l’utilisateur a initialement demandée.

## 3 – MVC Météo

A son arrivée sur l’onglet Météo, l’utilisateur se voit normalement afficher la météo de Paris ainsi que New-York.

Le bouton WEATHER lui permet de rafraîchir les données.

Comme pour les autres MVC, le principe est le même.

Le bouton WEATHER déclenche l’Action qui demandera au Contrôleur (WeatherVC) la nouvelle météo.

Le contrôleur demandera au Model (WeatherService) qui réalisera l’appel API afin d’obtenir la météo la plus récente.

Il s’appuiera de ses fichiers APIKeys et WeatherService pour chercher sa clé API ainsi que le bon format du fichier JSON qu’il s’apprête à recevoir.

Une fois les données reçues, le Model, grâce au callback, envoie la réponse au Contrôleur qui modifie la Vue grâce aux Outlets qui afficheront la météo à l’utilisateur.