PROJET FOREX

ETUDE PREALABLE

Simon BLOEMENDAL

TSDI 2015
Pôle Formation CCI Colmar

Ou trouver l'information : site web du Forex.

Le web service du site Internet du Forex affiche dans une page web un tableau récapitulant toutes les données que l'on a sélectionné pour chaque couple de devises que l'on a choisie.

Investing.com

Nom	Achat	Vente	Cours	Ouverture	+ Haut	+ Bas	Heure
EUR/USD	1,0957	1,0959	1,0958	1,0925	1,0974	1,0901	10:28:57
EUR/CHF	1,0490	1,0494	1,0492	1,0467	1,0510	1,0461	10:28:57

Cours des Devises fournis par Investing.com France.

L'adresse internet (URL) par laquelle on va accéder au web service est composée de plusieurs parties :

http://fxrates.fr.forexprostools.com/index.php?force_lang=5&pairs_ids=1;10;&header-text-color=%23FFFFFF&curr-name-color=%230059b0&inner-text-color=%23000000&green-text-color=%232A8215&green-background=%23B7F4C2&red-text-color=%23DC0001&red-background=%23FFE2E2&inner-border-color=%23CBCBCB&border-color=%23cbcbcb&bg1=%23F6F6F6&bg2=%23ffffff&bid=show&ask=show&last=show&change=hide&change_in_percents=hide&last_update=show

Nous allons nous intéresser à chacune des parties mises en surbrillance. Le reste de l'URL concerne l'affichage proprement dit de la page web et ne nous concerne pas.

La première partie, en jaune, correspond au site auquel nous allons accéder. Nous verrons que l'on peut le remplacer par d'autres pour accéder aux sites d'autres pays. Cela peut nous être utile si l'un des sites est en maintenance ou bloqué pour différentes raisons.

La deuxième partie, en vert et commençant par « *pairs_id* », va indiquer au web service quelles couples de devises nous voulons voir afficher, par le biais d'identificateur uniques (ici 1 correspond au couple EUR/USD et 10 au couple EUR/CHF).

La dernière partie, en bleu, commence à partir de « *bid* » et va jusqu'à la fin de l'url. Cette partie va indiquer au web service quelles informations nous voulons voir afficher pour chaque couple de devises. Ainsi « *bid* » va correspondre à la valeur d'achat, « *ask* » à la valeur de vente, « *change_in_percent* » au taux de variation et « *last_update* » à l'heure de la dernière cotation.

Ainsi, en modifiant ces trois parties à notre convenance, nous obtenons une url qui permettra de recevoir du web service les valeurs que nous voulons. Nous pourrons même omettre les autres informations de l'url car elles ne nous seront pas utiles dans le cadre de notre projet.

Ainsi cette url nous permet d'obtenir le même résultat que précédemment depuis le site anglais :

http://fxrates.investing.com/index.php?pairs_ids=1;10;&bid=show&ask=show&last=show&change=hide&change_in_percents=hide&last_update=show

Nom	Achat	Vente	Cours	Ouverture	+ Haut	+ Bas	Heure
EUR/USD	1,0874	1,0876	1,0875	1,0823	1,0899	1,0785	24/04
EUR/CHF	1,0377	1,0380	1,0378	1,0332	1,0381	1,0318	24/04

Le résultat de notre requête personnalisée

Le code qui nous sera retourné par le web service est une page web écrite en langage HTML.

Afin de récupérer les valeurs qui nous intéressent, il faudra traverser le document à la recherche des bonnes balises, puis identifier chaque valeur par leur identifiant et/ou les classes qui leur sont attribué.

```
<body>
  <div id="cross_rates_container" style="clear:both">
    <table id="cross_rate_1" class="ftq_4 ftq_5_1" cellspacing="0" cellpaddin
      <tbody>
        <tr class="arial_11_white_b ftq_5"></tr>
        <tr id="pair_1" class="arial_11 ftqtr1" style="background-color:;">
          <td class="ftqbb ftqw1 ftqrl" nowrap="nowrap">
            <nobr>
              <span class="vertical arrow_red_down" style="margin-right:5px;" t
              <span class="ftqa11bb arial_11_b">
                EUR/USD
              </span>
            </nobr>
          </td>
          <td class="ftqbb ftqw2 pid-1-bid" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-1-ask" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-1-last" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-1-high" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-1-low" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-1-time ftqac"></td>
        </tr>
        <tr id="pair_10" class="arial_11 ftqtr2" style="background-color:;">
          <td class="ftqbb ftqw1 ftqrl" nowrap="nowrap"></td>
          <td class="ftqbb ftqw2 pid-10-bid" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-10-ask" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-10-last" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-10-high" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-10-low" style="direction:ltr;"></td>
          <td class="ftqbb ftqw2 pid-10-time ftqac"></td>
        </tr>
      </tbody>
    </table>
  </div>
```

On observe une balise <*table*> contenant deux lignes <*tr*>. Ces deux lignes correspondent justement aux deux couples de devises que l'on a choisies et on voit apparaitre les valeurs de chacun dans des cellules <*td*>, à part le nom du couple qui se trouve dans une balise <*span*>.

Ainsi, pour chaque couple de devises, il faudra rechercher son identifiant dans chaque ligne <*tr*> (ici, « *id = pair_1* » pour le couple EUR/USD). Puis rechercher la classe unique attribuée au titre dans sa balise <*span*>. Et enfin rechercher pour chaque valeur sa classe unique dans les balises <*td*> correspondantes.

Envoyer des requêtes HTTP et récupérer leur résultat en C++ avec Qt

La classe **QNetworkAccessManager** permet d'envoyer des requêtes vers le réseau. Cette classe permet aussi de recevoir des réponses depuis le réseau. C'est la classe principale qui s'occupe de l'accès au réseau et est nécessaire pour utiliser les classes suivantes.

Une requête vers le réseau est construite par le biais de la classe **QNetworkRequest**, qui contiendra toutes les informations nécessaires à l'envoie de la requête vers le réseau. Elle contient notamment une URL.

La réponse à une requête doit être appréhendée par la classe **QNetworkReply**. Cette classe contiendra les données et métadonnées relative à la requête envoyée par **QNetworkRequest**. Elle contient elle aussi une URL, des informations relatives à l'état de la réponse, et surtout le contenu de la réponse elle-même.

La classe **QUrl** permet de manipuler les URL de façon plus pratique. Elle peut décoder et construire une URL sous plusieurs formes.

Avec ces quatre classes, on peut facilement envoyer notre requête au site Forex, récupérer la réponse HTML qu'il nous renverra, et manipuler ce code pour accéder aux informations qui nous intéressent.

Exploitation des données reçues

Identifions maintenant plus précisément les balises et identifiants qui nous permettront d'accéder aux informations voulues.

```
<tr id="pair_1" class="arial_11 ftqtr1" style="background-color:;">
  <td class="ftqbb ftqw1 ftqrl" nowrap="nowrap">
    <nobr>
      <span class="vertical arrow_red_down" style="margin-right:5px;
      <span class="ftqa11bb arial_11_b">
        EUR/USD
      </span>
    </nobr>
  </td>
  <td class="ftqbb ftqw2 pid-1-bid" style="direction:ltr;"></td>
  <td class="ftqbb ftqw2 pid-1-ask" style="direction:ltr;"></td>
  <td class="ftqbb ftqw2 pid-1-last" style="direction:ltr;"></td>
  <td class="ftqbb ftqw2" style="direction:ltr;"></td>
  <td class="ftqbb ftqw2 pid-1-high" style="direction:ltr;"></td>
  <td class="ftqbb ftqw2 pid-1-low" style="direction:ltr;"></td>
  <td class="ftqbb ftqw2 pid-1-time ftqac"></td>
</tr>
```

Il nous faut d'abord cibler la ligne <*tr*> qui nous intéresse. On fera donc la recherche d'une balise <*tr*> ayant comme identifiant « *pair_* » suivi du numéro du couple de devises voulue. Il faudra donc au préalable connaitre la correspondance des couples de devises et des numéros que leur a attribués le site du Forex.

On cherchera ensuite à l'intérieur de la balise <*tr*> identifiée les informations qui nous intéressent :

- Nom du couple de devises :

  On recherche la balise <*span*> ayant pour classe « *ftqa11bb* ». En effet, cette classe est uniquement attribuée à la balise contenant le nom des couples de devises.

- Valeur d'achat :

  On recherche la balise <*td*> ayant une classe se terminant par « *bid* ».

- Valeur de vente :

  On recherche la balise <*td*> ayant une classe se terminant par « *ask* ».

- Valeur du taux de variation :

  On recherche la balise <*td*> ayant une classe se terminant par « *pcp* ».

Dans le cahier des charges, on nous demande aussi la date et l'heure de chaque cotation. Cependant ces valeurs pourront être plus facilement récupérées depuis la base de données elle-même.

Avec Qt, une manière simple est efficace de gérer du contenu HTML est d'utiliser la classe **QWebView**. Cette classe permet en effet de stocker et d'interpréter une réponse HTML.

Si sa première utilité est d'afficher et d'éditer un site web, nous allons pouvoir nous servir d'un mécanisme bien pratique qu'elle applique à chaque réponse HTML. En effet, elle découpe et stocke chaque balise et son contenu dans des **QWebElement**, rendant d'autant plus simple la manière d'y accéder. Une fois le **QWebView** chargé, il suffira d'interroger chaque **QWebElement** qui le compose à la recherche des classes que l'on a identifiées plus haut. On utilisera la méthode « *findFirstElement* » qui nous renverra le premier **QWebElement** contenant le critère de recherche.

Enregistrement des données lues dans une base de données

Pour utiliser une base de données, nous aurons besoin de deux classes :

1. La classe **QSqlDatabase** :

```
QSqlDatabase db = QSqlDatabase::addDatabase("QSQLITE");
db.setDatabaseName(":memory:");
```

La classe **QSqlDatabase** représente une connexion à une base de données. Elle procure une interface permettant d'accéder à une base de données. Elle y accède par le biais de drivers héritant de la classe **QSqlDriver**.

Une fois la connexion créée, la méthode « *setDatabaseName* » permet d'indiquer le nom du fichier ou sera stocké la base de données. Ici on la stocke en mémoire.

Il suffit ensuite d'appeler la méthode open pour pouvoir accéder à la base de données.

2. La classe **QSqlQuery** :

```
QSqlQuery query;
query.exec(« Commande SQL ») ;
```

La classe **QSqlQuery** donne un moyen de manipuler et d'exécuter des requêtes SQL.

Une fois instanciée, la méthode « *exec* » reçoit en argument un Q**String** contenant la commande SQL et l'envoie à la base de donnée ouverte.

Dans le cas d'une commande SELECT, le résultat est stocké dans l'objet **QSqlQuery** et est accessible grâce à la méthode « *value* ».

Ensuite, pour afficher les données depuis la base de données, nous aurons besoin de deux classes supplémentaires :

3. La classe **QSqlTableModel** :

```
QSqlTableModel *model)
model->setTable("person");
model->select();
```

La classe **QSqlTableModel** procure un modèle de données pour une seule base de données. Ce modèle est éditable. Une fois un objet **QSqlTableModel** créé, on utilise la méthode « *setTable* » pour lui indiquer à quelle table se connecter. C'est la base de données ouverte à ce moment qui sera utilisée.

La méthode « *select()* » permet ensuite de sélectionner toutes les données. On peut aussi passer une commande SQL par la méthode « *setQuery* ».

4. La classe **QTableView** :

```
QTableView *view = new QTableView;
view->setModel(model);

view1->show();
```

La classe **QTableView** procure une vue sous forme de tableau permettant d'afficher les données fournies par un **QSqlTableModel**. Une fois un objet **QTableView** créé, la méthode « *setModel* » permet de lui attribuer un **QSqlTableModel** déjà créé, et la méthode « *show()* » permet de l'afficher. Il est à noter qu'un **QSqlTableModel** peut alimenter plusieurs **QTableView**. Une modification dans le modèle sera répercuté par chaque vue connectée à ce modèle.

En l'état actuel des choses, Qt ne permet pas d'accéder à une base de données MySql, du fait de drivers défectueux. Mais étant donné que la base de données dont on aura besoin pour le projet est relativement simple et petite, on pourra avantageusement utiliser une base SQLite.

On procédera donc de même que dans l'exemple de ce début de chapitre, avec une différence néanmoins. La base de données devra être enregistrée dans un fichier, donc on indiquera le chemin et nom du fichier dans la méthode « *setDatabaseName* ». On pourra d'ailleurs lui fournir une variable dans laquelle on aura stocké le choix de l'utilisateur :

```
QString nomBdd ;
db.setDatabaseName(nomBdd);
```