

Travail pratique #6  
Interfaces Utilisateurs et Exceptions

---

<b>Objectifs :</b>	Permettre à l'étudiant de se familiariser avec le concept de modèle MVC par l'entremise de la librairie QT.
<b>Remise du travail :</b>	Mardi le 3 décembre 2019, 8h.
<b>Références :</b>	Notes de cours sur Moodle et documentation QT <a href="http://doc/qt/io/">http://doc/qt/io/</a>
<b>Documents à remettre :</b>	Les fichiers .cpp et .h complétés sous la forme d'une archive au format .zip.
<b>Directives :</b>	<a href="#">Directives de remise des Travaux pratiques sur Moodle</a>  Les travaux dirigés s'effectuent obligatoirement en équipe de deux personnes faisant partie du même groupe.  <a href="#">Veuillez suivre le guide de codage</a>

Le travail effectué dans ce TP continue celui amorcé par les TP précédents soit une plateforme de gestion de membres d'une compagnie AirPoly.

Les interfaces graphiques sont souvent conçues selon un modèle MVC, soit modèle vue contrôleur. Par contre, QT fonctionne sur une base simplifiée de ce modèle, le modèle vue. Le contrôleur est intégré aux différentes vues et il faut utiliser les fonctions propres à l'api QT pour les modifier.

Pour réussir ce TP il faudra vous familiariser avec les concepts de programmation événementielle. QT fonctionne avec des signaux et des « slots » permettant d'accepter ces signaux et de les traiter. La documentation complète de l'API QT est disponible en ligne.

Pour vous aider, des fichiers .cpp et .h vous sont fournis. Vous n'avez qu'à implémenter les nouvelles méthodes décrites plus bas. Les attributs ou méthodes et les classes qui ne sont plus nécessaires ont été supprimés. Et les méthodes à modifier vous ont été indiquées.

**ATTENTION : Tout au long du TP, assurez-vous d'utiliser les opérateurs sur les objets et non sur leurs pointeurs ! Vous devez donc déréférencer les pointeurs si nécessaires.**

**ATTENTION : Vous serez pénalisés pour les utilisations inutiles du mot-clé *this*. Utilisez-le seulement où nécessaire.**

**ATTENTION : Vous devez utiliser les fichiers fournis pour ce TP.**

**Remarque : Pour plus de précision sur le travail à faire et les changements à effectuer, veuillez-vous référer aux fichiers .h. TOUTE fonction dans les fichiers .h doivent être implémentées.**

---

### Classe ExceptionArgumentInvalide

---

Cette classe représente une exception QT. Elle peut se propager à travers différents thread utilisateurs et doit être soulevée si on tente de créer un usager sans avoir complété tous les champs. Elle fonctionne comme n'importe quelle autre exception C++.

#### Attributs :

**QString s\_ :** Attribut contenant le message de l'erreur.

#### Méthodes :

- **ExceptionArgumentInvalide :** Constructeur par paramètre prenant un QString et l'assignant à l'attribut de la classe.
- **what :** Override de la fonction what héritée de QException. Cette fonction retourne l'attribut QString par copie et ne prend aucun argument.

---

### Classe MainWindow

---

Cette classe est la classe principale du TP. C'est dans cette classe que se situent toutes les informations sur l'interface graphique. Il s'agit de la vue de notre application. Des captures d'écrans sont fournis plus bas pour vous donner une idée de ce à quoi l'application devrait ressembler et interagir.

#### Attributs :

- **vector<Coupon\*> coupons\_ :** un vecteur contenant tous les coupons créés dans le main().
- **vector<Membre\*> membres\_ :** un vecteur contenant tous les membres créés dans le main().

#### Attributs Locaux QObject :

- **QWidget\* listeBillets\_ :** La liste des billets affichés dans l'interface.
- **QWidget\* listeCoupons\_ :** La liste des coupons affichés dans l'interface.
- **QWidget\* listeMembres\_ :** La liste des membres affichés dans l'interface.

#### Affichage de Billet :

- **vector<QRadioButton\*> boutonsRadioTypeBillets\_ :** Boutons dans la création d'un billet pour déterminer le type du billet (FlightPass, Regulier, etc).
- **QComboBox\* choixMembreBillet\_ :** Liste déroulante des membres créés dans le main().
- **QLineEdit\* editeurPNR\_ :** boîte de texte pour le PNR du billet.
- **QLineEdit\* editeurPrixBillet\_ :** boîte de texte pour le prix du billet.
- **QLineEdit\* editeurOD\_ :** boîte de texte pour l'origine et la destination du billet.

- **QComboBox\* choixTarifBillet\_** : Liste déroulante des différents tarifs de Billet (Premiere, Affaire, etc).
- **QLineEdit\* editeurDateVol\_** : boîte de texte pour la date de vol du billet (FlightPassSolde ou RegulierSolde).
- **QLineEdit\* editeurUtilisationsRestantesFlightPass\_** : boîte de texte pour le nombre d'utilisation restante du billet (FlightPass ou FlightPassSolde).

#### Affichage de Coupon :

- **QLineEdit\* editeurCodeCoupon\_** : boîte de texte pour le code du coupon.
- **QLineEdit\* editeurRabaisCoupon\_** : boîte de texte pour le rabais du coupon.
- **QLineEdit\* editeurCoutCoupon\_** : boîte de texte pour le cout du coupon.

#### Affichage de Membre :

- **QLineEdit\* editeurPoints\_** : boîte de texte pour les points du membre.
- **QLineEdit\* editeurPointsCumules\_** : boîte de texte pour les points cumulés du membre.
- **QLineEdit\* editeurJoursRestants\_** : boîte de texte pour les jours restants d'un membre premium.

#### **Méthodes déjà implémentées :**

- **void setup()** : Méthodes qui construit l'interface :
  - o Elle ajoute la barre des menus.
  - o Elle charge les objets initialisés dans le main().
  - o Fait appel à nettoyerVue() pour initialiser les champs.
- **void chargerCoupons/Membres/Billets()** : Initialise les objets QListWidget qui permettent l'affichage des coupons, membres, billets.
- **void ajouterMembresDansComboBox()** : ajoute les membres existants dans la liste déroulante pour la partie Billet.

### Méthodes à modifier:

- **void setUI()** : Cette fonction principale génère tous les éléments de la vue. Il s'agit d'une longue fonction dans laquelle vous devez ajouter les éléments manquants de la vue aux endroits indiqués et connecter les différents éléments de la vue avec les fonctions que vous implémenterez. Vous devez donc compléter la vue en ajoutant :
  - **Partie milieu de l'interface réservée aux coupons (à partir de la ligne 156 dans le code) :**
    - Pour la liste des coupons, connecter le signal « sélection de coupon » avec la fonction `MainWindow ::SelectionnerCoupon()`.
    - Ajouter un bouton « Ajouter Coupon » et connecter le signal « Clic sur Ajouter Coupon » avec la fonction `MainWindow ::AjouterCoupon()`. Le bouton doit être situé juste avant la ligne de séparation (cf captures d'écrans plus bas).
  - **Partie inférieure de l'interface réservée aux membres à partir de la ligne 205 dans le code) :**
    - La liste déroulante située au-dessus de la liste des membres sert à trier les membres à afficher on peut : Tout afficher, afficher les membres réguliers, afficher les membres premiums. En fonction de l'option choisie, connecter le signal avec la fonction `MainWindow :: filtrerListe()`.
    - Pour la liste des membres, connecter le signal « sélection de membre » avec la fonction `MainWindow ::SelectionnerMembre()`.
  - **Partie supérieure de l'interface réservée aux billets à partir de la ligne 65 dans le code) :**
    - Pour la liste des billets, connecter le signal « sélection de billet » avec la fonction `MainWindow ::SelectionnerBillet()`.
    - Ajouter un bouton « Ajouter Billet » et connecter le signal « Clic sur Ajouter Billet » avec la fonction `MainWindow ::AjouterBillet()`. Le bouton doit être situé juste avant la ligne de séparation (cf captures d'écrans plus bas).
- **Void afficherMessage(QString msg)**: Cette fonction affiche une fenêtre de message qui 'pop up' à l'écran et affiche le message 'msg' passé en paramètre.
- **Void filtrerListe(int index)** : Cette fonction filtre la liste des membres selon l'index donné en paramètre et affiche seulement les membres qui correspondent à ce champ (Tout Afficher à l'index 0, Afficher Membres Réguliers l'index 1,

Afficher Membres Premiums l'index 2). À l'aide d'une boucle, on parcourt le contenu de l'attribut `listeMembres_`, on stocke un item de la liste dans un objet de type `QListWidgetItem`. À chaque item, on appelle la méthode `setHidden()` qui prend comme paramètre la méthode `filtrerMasque()`.

- **bool filtrerMasque(Membre\* membre, int index)** : En fonction de l'index passé en paramètres, retourne un booléen pour savoir si le type du membre passé en paramètre correspond au critère sélectionné. (Tout Afficher à l'index 0, Afficher Membres Réguliers l'index 1, Afficher Membres Premiums l'index 2)
- **nettoyerVueBillets/ Coupons/ Membres**: Pour chacune des méthodes de « nettoyage » de vue, réinitialiser la valeur des champs pour chaque catégorie. Une fois cette méthode appelée, visuellement, l'interface doit être comme au démarrage. Autrement dit, les champs doivent être vides et il doit être possible de les remplir.
- **selectionnerCoupon/ Membre/ Billet**: Pour chacune des méthodes de sélection, remplissez les champs associés à l'objet. Exemple si on sélectionne un billet, les champs supérieurs droits doivent contenir les attributs de l'élément sélectionné. Aussi, il ne doit pas être possible de cliquer, ou modifier ces champs.
- **getTarifBillet()**: En fonction du type de billet sélectionné dans la liste déroulante, retourne le TarifBillet associé.
- **trouverMembreParNom(const string& nom)**: Recherche le membre dans `membres_` en fonction de son nom et le retourne.
- **ajouterBillet()**: Cette fonction ajoute un billet directement à un membre (renseigné à l'aide de la liste déroulante). Il sera créé à partir des informations renseignées et ajouté seulement si tous les champs de la vue ont été complétés. On utilise donc un try throw catch dans la fonction en affichant un message d'erreur approprié en fonction du champ manquant. Par exemple, si le champ Origine-Destination pour un billet n'a pas été rempli, le message d'erreur devrait être 'Erreur : L'origine-Destination n'est pas rempli'. Enfin, une fois le billet créé, utiliser la méthode `Membre::ajouterBillet()`.
- **ajouterCoupon()**: En suivant le même principe que pour `ajouterBillet()`, ajouter le coupon au vecteur de coupons en attribut (`coupons_`).
- **void setMenu()** : Elle permet d'ajouter un onglet « Fichier » à la barre de navigation. Dans l'attribut `QMenu fileMenu` (obtenu de l'héritage), ajoutez les options (`QAction`) suivantes :
  - « **Quitter** » : Si on sélectionne cette option, le programme se termine (la fenêtre se ferme).
  - « **Nettoyer vue** » : Si on sélectionne cette option, le programme nettoie la vue. Il réinitialise les valeurs des champs et la sélection des `QList`. Lorsque qu'on choisit cette option, le programme doit faire appel à la fonction `MainWindow::nettoyerVue()`.

Des membres, des coupons et des billets sont créés pour pouvoir être manipulés via l'interface.

## Captures d'écran

### Apparence au démarrage du programme

The screenshot shows a window titled "Bienvenue sur PolyAir !". The interface is divided into two main sections: a left sidebar and a main content area.

**Left Sidebar:**

- Fichier** (Menu)
- Billets :** A list of ticket codes: A1A1A1, B4Y6S1, GHJ6U2V, HH6T3R, O8I9P0.
- Coupons :** A list of coupon percentages: 10%, 20%, 5%.
- Tout Afficher** (Dropdown menu)
- Membres :** A list of member names: Alex, John, Marc, Robert.

**Main Content Area:**

- Membres** (Dropdown menu)
- Radio buttons for membership types: ☐ Regulier, ☐ Regulier Solde, ☐ FlightPass, ☐ FlightPass Solde.
- Input fields for: PNR, Prix, Od.
- Tarif Billet** (Dropdown menu)
- Input fields for: Date de Vol, Pourcentage Solde Billet, Utilisations Restantes.
- Ajouter Billet** (Button)
- Input fields for: Code, Rabais, Cout.
- Ajouter Coupon** (Button)
- Input fields for: Points (N/a), Points Cumules (N/a), Jours Restants (N/a).

## Affichage lorsqu'on sélectionne un billet :

Bienvenue sur PolyAir !

Fichier

**Billets :**

- A1A1A1
- B4Y6S1
- GHJ6U2V
- HH6T3R
- O8I9P0

**Coupons :**

- 10%
- 20%
- 5%

**Membres :**

- Alex
- John
- Marc
- Robert

**John**

☒ Régulier ☐ Régulier Solde ☐ FlightPass ☐ FlightPass Solde

PNR : A1A1A1

Prix : 3000.000000

Od : YUL - CDG

Première

Date de Vol : 2019-12-21

Pourcentage Solde Billet : N/a

Utilisations Restantes : N/a

Ajouter Billet

Code :

Rabais :

Cout :

Ajouter Coupon

Points : N/a

Points Cumules : N/a

Jours Restants : N/a

## L'affichage lorsqu'on sélectionne un coupon :

Bienvenue sur PolyAir !

Fichier

**Billets :**

- A1A1A1
- B4Y6S1
- GHJ6U2V
- HH6T3R
- O8I9P0

**Coupons :**

- 10%
- 20%
- 5%

**Membres :**

- Alex
- John
- Marc
- Robert

**Membres**

☒ Régulier ☐ Régulier Solde ☐ FlightPass ☐ FlightPass Solde

PNR :

Prix :

Od :

Tarif Billet

Date de Vol :

Pourcentage Solde Billet :

Utilisations Restantes :

Ajouter Billet

Code : 10%

Rabais : 0.100000

Cout : 900

Ajouter Coupon

Points : N/a

Points Cumules : N/a

Jours Restants : N/a



L'affichage lorsqu'on sélectionne un membre :

Bienvenue sur PolyAir !

Fichier

Billets :

A1A1A1  
B4Y6S1  
GHJ6U2V  
HH6T3R  
O8I9P0

Coupons :

10%  
20%  
5%

Tout Afficher

Membres :

Alex  
John  
Marc  
Robert

Membres

☒ Regulier ☐ Regulier Solde ☐ FlightPass ☐ FlightPass Solde

PNR :

Prix :

Od :

Tarif Billet

Date de Vol :

Pourcentage Solde Billet :

Utilisations Restantes :

Ajouter Billet

Code :

Rabais :

Cout :

Ajouter Coupon

Points : N/a

Points Cumules : 175

Jours Restants : 30

Affichage de tous les membres premiums uniquement.

Bienvenue sur PolyAir !

Fichier

Billets :

A1A1A1  
B4Y6S1  
GHJ6U2V  
HH6T3R  
O8I9P0

Coupons :

10%  
20%  
5%

Afficher Membres Premium

Membres :

Alex

Membres

☒ Regulier ☐ Regulier Solde ☐ FlightPass ☐ FlightPass Solde

PNR :

Prix :

Od :

Tarif Billet

Date de Vol :

Pourcentage Solde Billet :

Utilisations Restantes :

Ajouter Billet

Code :

Rabais :

Cout :

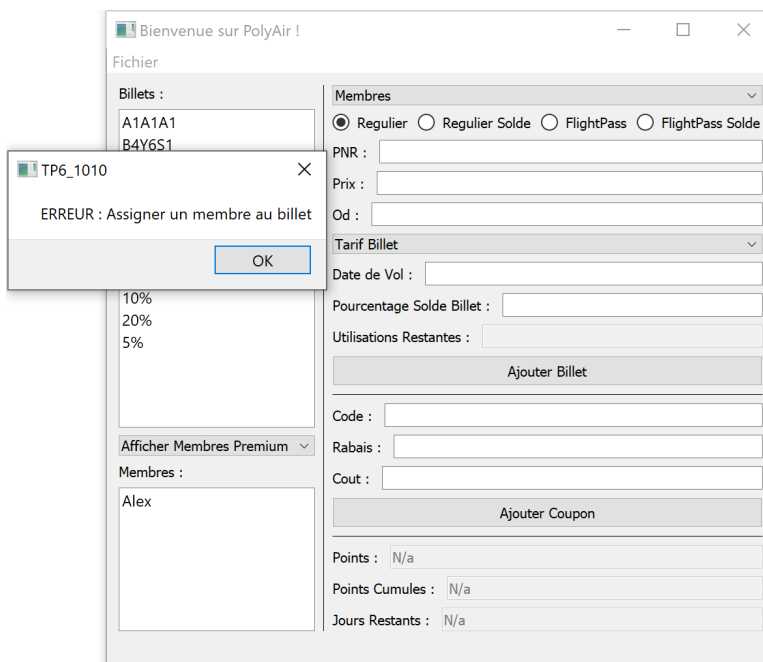
Ajouter Coupon

Points : N/a

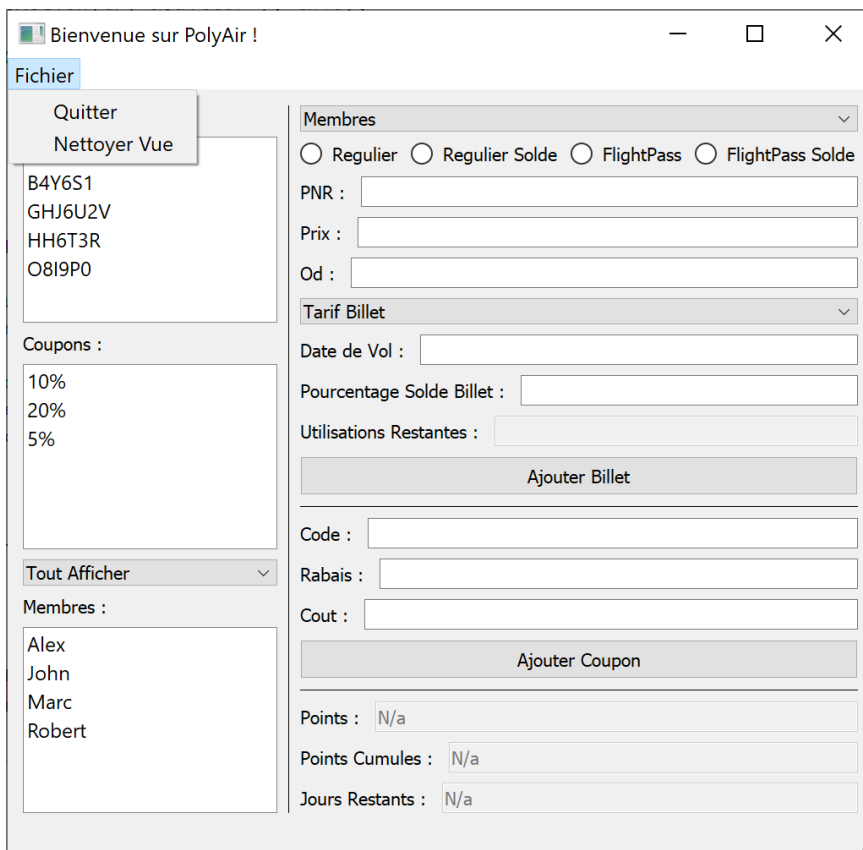
Points Cumules : N/a

Jours Restants : N/a

Erreur produite lorsque l'on clique sur 'Ajouter' mais qu'il reste des champs vides.



Menu « Fichier » pour quitter ou nettoyer la vue :



- Ajouter un destructeur pour chaque classe chaque fois que cela vous semble pertinent
- Utilisez la liste d'initialisation pour l'implémentation de vos constructeurs
- Ajouter le mot-clé const chaque fois que cela est pertinent
- Documenter votre code source
- Note : Lorsqu'il est fait référence aux valeurs par défaut, pour un string cela équivaut à la chaîne vide, et pour un entier au nombre 0
- Sentez-vous libre de personnalisez l'interface, beaucoup de choses sont possibles avec QT !

## Correction

---

La correction du TP se fera sur 20 points.

Voici les détails de la correction :

- (3 points) Compilation du programme ;
- (3 points) Exécution du programme ;
- (4 points) Comportement exact
- (4 points) Gestion adéquate des signaux/slot QT
- (2 points) Gestion des exceptions;
- (1.5 points) Utilisation adéquate des widgets QT (Boîte de message, TableWidget)
- (1.5 points) Documentation du code ;
- (1 points) Allocation / désallocation appropriée mémoire