



À remettre le 8 décembre 2022 avant 23h30.

<b>Objectifs</b>	Permettre à l'étudiant de se familiariser avec les concepts d'interface graphique, de la librairie QT, et les exceptions.
<b>Remise du travail</b>	<ul style="list-style-type: none"><li>• Date : <b>8 décembre 2022 à 23h30</b></li><li>• <b>Une note de 0 sera attribuée aux équipes qui remettent leur travail en retard.</b></li><li>• Remettre <b>tous</b> les fichiers directement sous une archive <b>.zip</b> (sans dossier à l'intérieur et sans autres fichiers).</li><li>• <b>⚡ Toute archive autre que zip ou remise ne respectant pas ces consignes sera refusée et se méritera une note de 0. Attention, une archive .7z ne compte pas comme une archive zip. ⚡</b></li><li>• <u>Respectez le format de la remise!</u> Nom de l'archive zip : <b>matricule1_matricule2_groupe.zip</b> Exemple : 1234567_1234568_1.zip</li></ul>
<b>Références</b>	<ul style="list-style-type: none"><li>• Notes de cours sur Moodle</li><li>• Documentation QT (<a href="https://doc.qt.io/qt-5/classes.html">https://doc.qt.io/qt-5/classes.html</a>)</li></ul>
<b>Directives</b>	<ul style="list-style-type: none"><li>• Les travaux s'effectuent obligatoirement en équipe de deux personnes faisant partie du même groupe.</li><li>• Les entêtes de fichiers sont obligatoires.</li><li>• Les fonctions que vous décidez d'ajouter au programme doivent être documentées.</li></ul>
<b>Conseils</b>	<ul style="list-style-type: none"><li>• Lisez les conseils, l'aperçu et les spécifications avant de commencer!</li><li>• Ayez lu vos notes de cours!</li><li>• Si vous avez des difficultés au cours du TP, rappelez-vous que de nombreux problèmes demandés sont assez couramment rencontrés en C++. Consulter la documentation sur cplusplus, les notes de cours, et les forums sur Google peuvent vous donner de bonnes pistes de résolution!</li><li>• Veuillez utiliser la version v17 de C++.</li><li>• Si votre programme ne compile pas, veuillez mettre en commentaires les instructions qui ne compilent pas.</li></ul>

## Spécifications générales

---

⌘ Le non-respect des spécifications générales entraînera des pénalités. ⌘

- Sur Visual Studio, vous devez activer C++17.
- **L'inclusion des fichiers d'en-tête (.h) doit être sensible aux majuscules et minuscules.** Malheureusement, sur Windows les fichiers ne sont pas sensibles aux majuscules et minuscule, donc des erreurs d'inclusions pourraient passer inaperçues. Relisez-vous bien.
- Utilisez le plus possible la liste d'initialisation des constructeurs. L'utilisation du corps des constructeurs à la place de la liste d'initialisation entraînera une pénalité de style. L'ordre des variables (attributs) dans la liste d'initialisation doit être la même que celle dans la liste des attributs de la classe dans le fichier d'en-tête.
- Suivez le guide de codage sur Moodle.
- Modifications/ajouts au guide de codage à respecter :
  - Vous pouvez utiliser le style d'accolades que vous désirez, **tant que vous êtes uniformes**. Sachez cependant qu'il est généralement attendu d'un(e) développeur(se) suivre la convention établie par le projet, ou la convention établie par la langue (si elle existe) en créant un nouveau projet. Dans le cas du TP fourni, les accolades ouvrantes sont sur leur propre ligne (style Allman).
  - Mettez un espace avant la parenthèse ouvrante des énoncés de contrôle comme if, for, while et switch, mais pas avant les parenthèses d'un appel de fonction. Ne mettez jamais d'espace tout juste après une parenthèse ouvrante ou tout juste avant une parenthèse fermante.
  - Le deux-points (:) et le signe égal (=) devraient être entourés d'espaces, sauf dans le cas des étiquettes de case et les spécificateurs d'accès (private, protected et public) qui ne devraient pas avoir d'espace avant le deux-points. Un espace devrait toujours suivre les éléments de ponctuation comme la virgule (,), le deux-points (:) et le point-virgule (;).
  - N'utilisez pas NULL ou 0 pour les pointeurs, mais bien nullptr. Dans le cas de test de nullptr avec un pointeur, soyez explicite : préférez if (p != nullptr) à if (p).
  - N'utilisez pas de else après un return.

## Avant de commencer

---

- Vous devez installer QT. La version open source est gratuite et disponible pour téléchargement sur le site de QT : <https://www.qt.io/download-open-source>
- Si vous voulez utiliser Visual Studio, suivez les étapes spécifiées sur la page de QT : <https://doc.qt.io/qtvstools/qtvstools-getting-started.html>

## Mise en contexte

---

Vous devez compléter l'application TP5 en utilisant les concepts de programmation événementielle et de gestion des exceptions. Cette application rend vie au travail que vous avez réalisé au cours des quatre premiers travaux pratiques en utilisant la librairie QT.

## Aperçu des classes du projet

---

### 1. Le modèle

- *Tournoi* : Classe qui définit la logique du tournoi de hockey.
- *Équipe* : Classe qui définit une équipe de hockey.
- *Joueur et ses dérivées* : Classe qui définit un joueur de hockey.
- *Match* : Classe qui définit un match de hockey.

### 2. La vue/contrôleur

- *TP5* : Classe qui présente l'interface graphique et capte les événements. De plus, elle gère les événements en mettant à jour les données du modèle et l'affichage des données

## Travail à réaliser

---

On vous demande de compléter les fichiers qui vous sont fournis pour pouvoir implémenter le programme décrit ci-dessous (se référer à la section **Résultat**). Notez que l'interface graphique a déjà été implémentée pour vous ainsi que les différents gestionnaires d'événements. Dans ce travail pratique, vous n'aurez qu'à rendre l'application utilisable en créant et connectant les différents signaux à leur gestionnaire. De plus, il vous faudra gérer les différents cas d'exception décrit ci-dessous (se référer à la section **Gestion des exceptions**). Vous travaillerez dans les classes *TP5*, *Tournoi* et *Équipe*.

## Classes Équipe et Tournoi

---

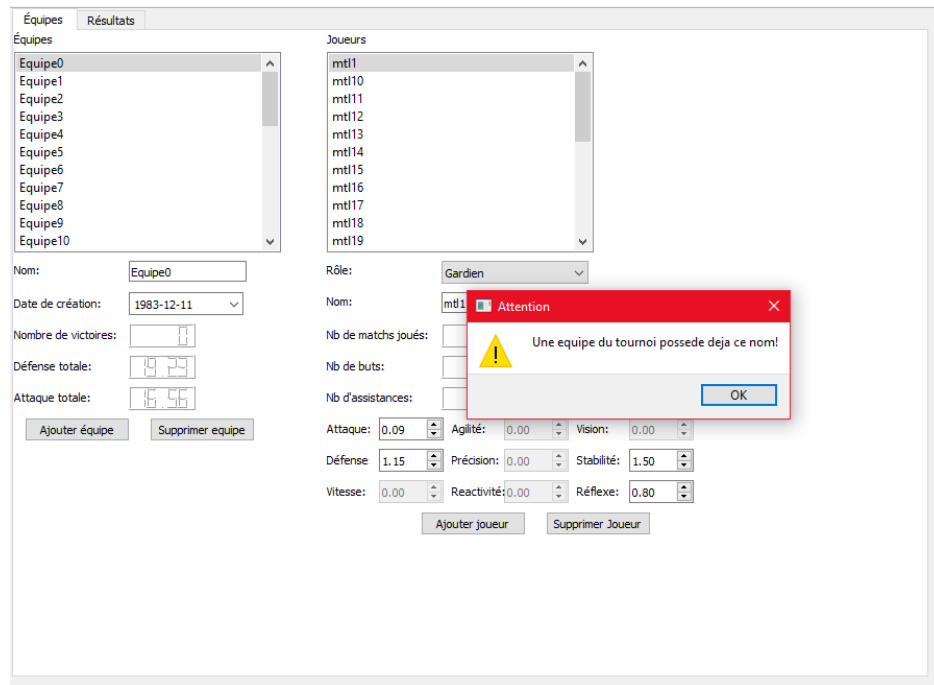
Définissent la logique de l'application. Vous devez ajouter ces deux classes au modèle QT afin de pouvoir ajouter et émettre les signaux nécessaires. De plus, vous devrez lancer les exceptions spécifiées. Fiez-vous aux TODO dans les fichiers .h et .cpp de ces classes pour les spécificités d'implémentation.

### Classe TP5

---

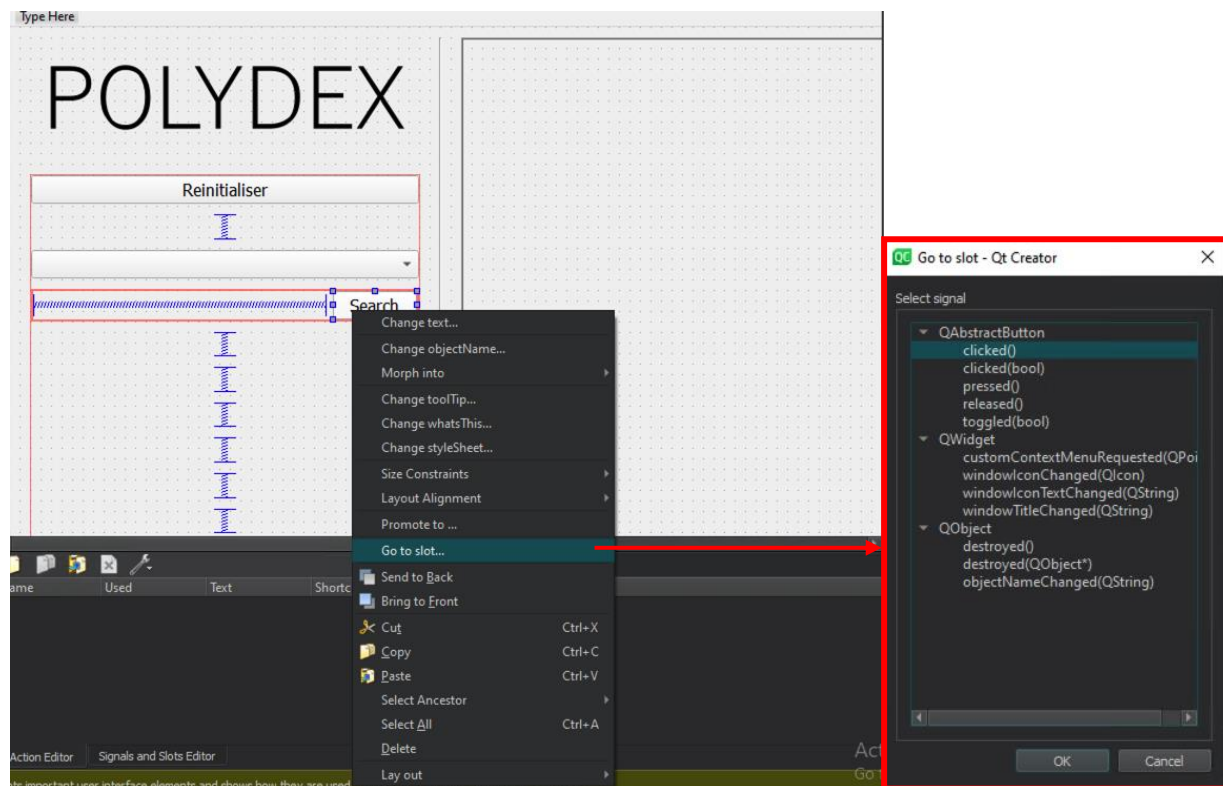
Cette classe correspond à la fois à la vue et au contrôleur de l'application. Elle englobe les composantes d'affichage (GUI) et s'occupe de la gestion d'événements à la suite d'un signal envoyé par les composantes d'affichage ou le modèle. Elle met à jour le modèle et met à jour l'affichage lors de changement d'état du modèle. On a déjà implémenté pour vous le GUI dans le fichier de l'interface graphique (.ui), mais vous êtes libre de modifier l'interface ou d'ajouter des fonctionnalités.

Vous devez ajouter les appels à la fonction *connect* nécessaires au fonctionnement de l'application. De plus, vous devrez implémenter la logique de gestion des cas d'exceptions de l'application. Fiez-vous au TODO dans TP5.cpp. Aussi, vous devez implémenter un mécanisme pour notifier l'utilisateur lorsqu'un cas d'exception survient. L'utilisateur devra être notifié au travers du GUI, par exemple :



**Remarque :** Si vous utiliser QT Creator, il est possible d'auto-générer les signatures des méthodes liées aux écoutes d'évènements (*event listeners*) dans un private slots.

1. Cliquer droit sur le UI element et sélectionner *Go to slot*
2. Vous pouvez à présent choisir le type de *event listeners*



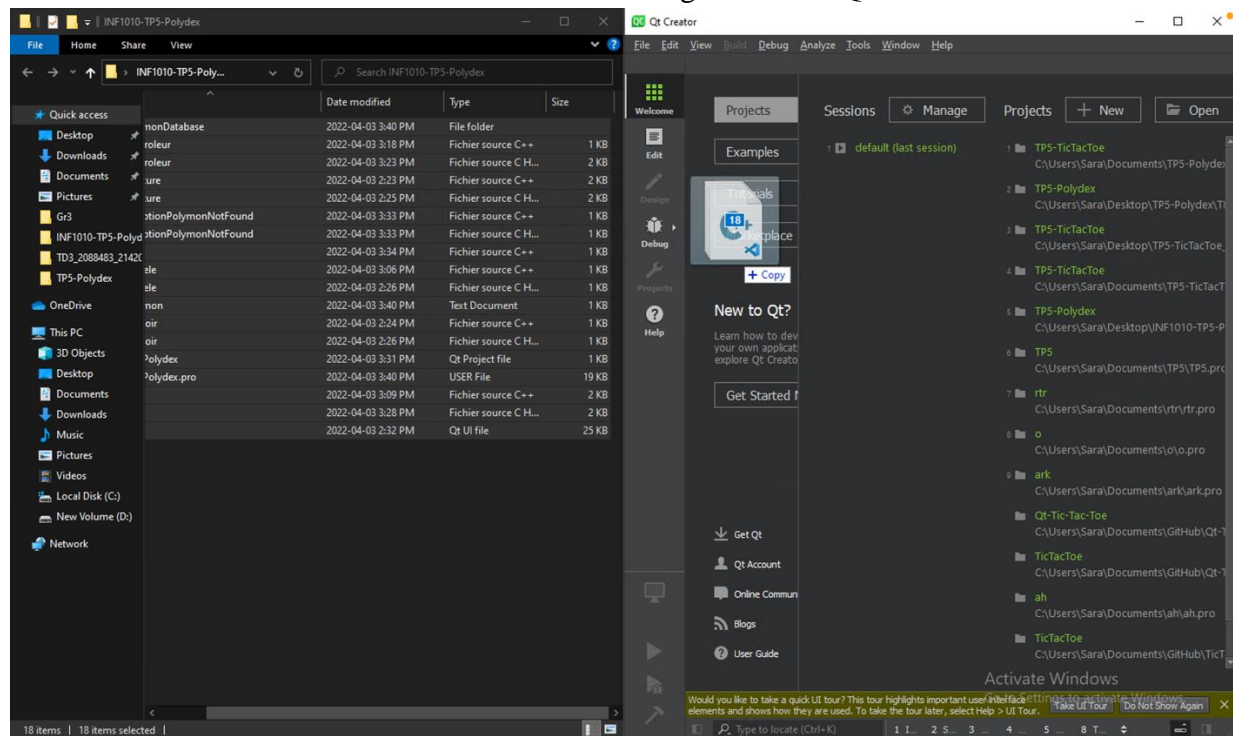
## Remarque


1. Ce TP ne comporte pas de tests, nous allons simplement tester le comportement de l'application. Fiez-vous aux descriptions ci-dessous avoir une idée du comportement souhaité.
2. Il est possible de modifier la vue, mais tous les éléments de base doivent s'y retrouver.
3. Pour la remise, vous devez compresser (.zip) la solution au complet (inclure tous les fichiers).

## Démarrage

Voici comment démarrer le projet dans QT Creator :

1. Sélectionner tous les fichiers du dossier et les glisser dans Qt



2. Cliquer sur Ok
3. Vous pouvez à présent cliquer sur *Run* (  )

Voici comment démarrer le projet dans Visual Studio:

1. Créer un nouveau projet de type QWidgetApplication
2. Copier les fichiers du TP dans le dossier du projet
3. Sélectionner tous les fichiers du dossier et les glisser dans le projet sur Visual Studio. Le fichier TP5.ui se trouvera sous le dossier *Form Files* du projet.
4. Assurez-vous que la version du compilateur du projet est bien C++ 17.

## Gestion des exceptions

1. Cas d'exceptions liés à la gestion des équipes :
  - Tentative d'ajout d'une équipe qui a le même nom qu'une autre équipe du tournoi
  - Tentative de retrait d'une équipe qui n'existe pas dans le tournoi
  - Tentative d'affichage d'une équipe qui n'existe pas dans le tournoi
2. Cas d'exceptions liés à la gestion des joueurs:
  - Tentative d'ajout d'un joueur qui a le même nom qu'un autre joueur de l'équipe
  - Tentative d'ajout d'un joueur sans avoir sélectionné un rôle
  - Tentative de retrait d'un joueur qui n'existe pas dans l'équipe
  - Tentative d'affichage d'un joueur qui n'existe pas dans l'équipe
3. Cas d'exceptions liés à la simulation du tournoi:
  - Tentative de simulation du tournoi avec un nombre impair d'équipes
  - Tentative de simulation du tournoi avec un nombre d'équipe qui n'est pas une puissance de 2. Le nombre d'équipes doit respecter la condition suivante :  $nb\ d'equipes \in \{2^1, 2^2, 2^3, \dots, 2^\infty\}$ .

## Résultat

Voici le comportement désiré de l'application qui comporte deux onglets :

1. L'onglet **Équipes** permet de gérer les équipes du tournoi et leurs joueurs associés :

- a) La **QListWidget** liste\_equipements affiche les noms des équipes inscrites au tournoi. Lorsqu'une nouvelle équipe est ajoutée au tournoi, son nom est ajouté à la liste. Lorsqu'une équipe est retirée du tournoi, son nom est retiré de la liste. Lorsque l'utilisateur double clique sur une équipe, celle-ci est affichée en mettant à jour les champs correspondants.
- b) Le **QPushButton** ajouter\_equipe ajoute une nouvelle équipe au tournoi avec le nom et la date de création spécifiés dans les champs correspondants.
- c) Le **QPushButton** supprimer\_equipe retire l'équipe spécifiée par le champ nom. Tous les champs liés à l'équipe, y compris la liste et les informations des joueurs, sont réinitialisés.
- d) La **QListWidget** liste\_joueurs affiche les noms des joueurs qui jouent pour l'équipe sélectionnée préalablement. Lorsqu'un nouveau joueur est ajouté à l'équipe, son nom est ajouté à la liste. Lorsqu'un joueur est retiré de l'équipe, son nom est retiré de la liste. Lorsque l'utilisateur double clique sur un joueur, celui-ci est affiché en mettant à jour les champs correspondants.
- e) La **QComboBox** role spécifie le type du joueur qui peut être soit un Attaquant, soit un Défenseur, soit un Gardien. Le rôle détermine quels statistiques peuvent être mises à jour. L'attaque et la défense sont toujours modifiables.
- f) Les **QDoubleSpinBox** attaque, défense, vitesse, etc. permettent de modifier les statistiques correspondantes. Elles peuvent prendre n'importe quelle valeur entre 0 et 2.
- g) Le **QPushButton** ajouter\_joueur ajoute un nouveau joueur à l'équipe du type spécifié avec le nom et les statistiques spécifiés dans les champs correspondants.
- h) Le **QPushButton** supprimer\_joueur retire de l'équipe préalablement sélectionnée le joueur spécifié par le champ nom. Tous les champs liés aux informations du joueur sont réinitialisés.
- i) Les champs Nombre de victoires, Défense totale, Attaque totale, Nb de matchs joués, Nb de buts et Nb d'assistances ne sont pas modifiables au travers du GUI.



2. L'onglet **Résultats** permet de simuler le tournoi et de consulter les résultats des matchs qui ont eu lieu lors de la simulation :

Équipes Résultats

Simuler tournoi

Matches

	2022-11-30	2022-11-31	2022-12-1	2022-12-2	2022-12-3
1	Equipe2 vs Equipe28 Scores : 7 - 1	Equipe2 vs Equipe7 Scores : 3 - 3	Equipe2 vs Equipe25 Scores : 5 - 5	Equipe2 vs Equipe9 Scores : 7 - 7	Equipe2 vs Equipe24 Scores : 7 - 7
2	Equipe20 vs Equipe23 Scores : 0 - 2	Equipe23 vs Equipe24 Scores : 1 - 3	Equipe24 vs Equipe1 Scores : 8 - 8	Equipe24 vs Equipe22 Scores : 9 - 8	
3	Equipe15 vs Equipe10 Scores : 4 - 3	Equipe15 vs Equipe22 Scores : 7 - 8	Equipe22 vs Equipe8 Scores : 6 - 6		
4	Equipe9 vs Equipe17 Scores : 5 - 3	Equipe9 vs Equipe14 Scores : 5 - 4	Equipe9 vs Equipe12 Scores : 4 - 4		
5	Equipe6 vs Equipe11 Scores : 6 - 6	Equipe6 vs Equipe12 Scores : 3 - 7			
6	Equipe8 vs Equipe30 Scores : 4 - 1	Equipe8 vs Equipe26 Scores : 3 - 0			
7	Equipe1 vs Equipe4 Scores : 6 - 6	Equipe1 vs Equipe19 Scores : 8 - 8			
8	Equipe25 vs Equipe31 Scores : 5 - 3	Equipe25 vs Equipe27 Scores : 5 - 2			
9	Equipe27 vs Equipe0 Scores : 2 - 0				

Équipe gagnante: Equipe2

- Le **QPushButton** Simuler\_tournoi permet sans surprise de simuler le tournoi. Les informations liées aux matchs et à l'équipe gagnante sont par la suite mises à jour au fur et à mesure qu'elles sont disponibles.
- Le **QTableWidget** matchs affiche chaque match sous forme de QString dans la colonne liée à leur date respective.
- Le **QLabel** equipe\_gagnante affiche le nom de l'équipe sortant gagnante du tournoi.

## Correction

La correction du TP se fait sur 20 points :

- [3 pt] Toutes les exceptions sont gérées et elles sont affichées dans le GUI
- [4 pt] Exécution du programme.
- [12 pt] Comportement exact de l'application
- [1 pt] Absence de fuites de mémoire.

**Relisez bien les spécifications générales et les consignes de remise au début du TP avant de remettre votre travail!**

**Bonne fin de session!**