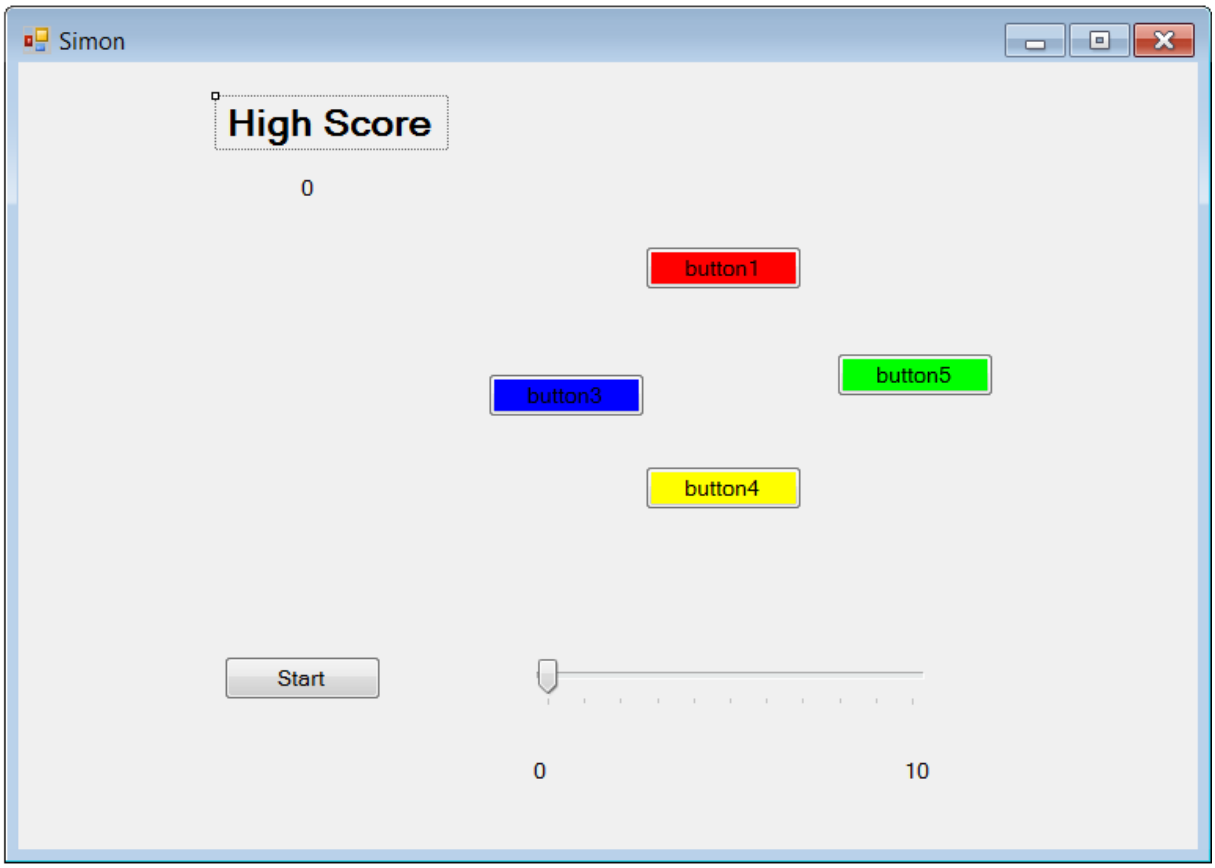
Exercices C# - Semaine 2

Objectifs

- ✓ Etre capable de créer des interfaces graphiques de type Windows Forms
- ✓ Pouvoir utiliser les éléments suivants de C# :
  - delegate
  - Propriétés
  - Classes partielles

Exercice à réaliser en Windows Forms

Le but de l'exercice de cette semaine est de créer un jeu du type « Simon ». Nous vous donnons une classe représentant la logique du jeu, à vous de créer l'interface graphique qui ressemblera à ceci :



Téléchargez l'archive Simon.7z et décompressez-la.
Un fichier .sln représente une solution applicative globale dans Visual Studio, qui peut contenir plusieurs projets.

Créez dans cette solution un nouveau projet de type « Windows Forms » en C#.

En cliquant droit sur le projet créé, définissez-le comme projet de démarrage.

Design de la Form

Vous allez voir apparaître la fenêtre de votre Form1.
Pour pouvoir y ajouter des contrôles nous allons afficher la boite à outils (Affichage -> Boîte à outils).

Ajouter sur votre forme :

- ✓ 5 Button
- ✓ 1 TrackBar
- ✓ 2 Labels

Cliquez sur un des boutons destiné à être en couleur et changez ses propriétés (Clic droit -> Propriétés) :

- ✓ Renommez-le en bouttonVert ou greenButton selon vos préférences linguistiques
- ✓ Changez la couleur de fond (propriété BackColor)
- ✓ Changez la propriété Text

Faites de même pour tous les éléments sur la Form.

Ouvrez le fichier Form1.Designer.cs. Ne le modifiez pas mais observez le code généré.

Points à noter : la classe Form1 est « partial » et le pragma #region

Logique du jeu

Il est maintenant temps de faire la liaison avec la logique du jeu. Ouvrez la classe Simon dans le répertoire Domaine et lisez-en le code et la documentation.

Etat initial de la Form

Double cliquez n'importe où sur la Form (mais pas sur un contrôle). Le designer va automatiquement vous créer un handler pour l'évènement de chargement de la forme (Load). Comme état initial du jeu, nous allons disabler tous les boutons sauf le bouton "start".

Créez également une instance de la classe Simon. Pour pouvoir utiliser la classe, vous devrez faire une référence au projet existant Simon et écrire ensuite using Simon.Domaine.

Démarrage du jeu

Double cliquez sur le bouton « start ». Le designer va automatiquement vous créer un handler pour l'évènement click. Enablez tous les boutons et disablez le start.

Appelez la méthode qui lance le jeu.

Allumage des boutons

En observant la classe Simon, vous avez sûrement remarqué qu'elle possède deux délégués. Il y en a un pour dire à la forme quels boutons elle doit allumer et un autre en cas de time out.

La classe possède deux propriétés correspondant à chaque délégué (là où l'on définit get et set). Si cela vous pose problème, lisez la partie « properties » du document de comparaison de Java et C#.

Créez dans la forme une méthode appelée « OnAllumerBoutons » cette méthode doit respecter la signature du délégué `AllumerBoutonsDelegate`.

Dans la méthode gérant le chargement de la forme, enregistrez cette méthode auprès du jeu en l'assignant à la propriété `AllumerButtonCallBack`.
Coder la logique qui en parcourant l'Enumerator change la couleur des boutons.
Testez votre code.

Réponse du joueur

Le joueur va devoir répéter la séquence des boutons allumés en cliquant sur les boutons.
Codez le gestionnaire de l'évènement click pour chaque bouton de couleur. Chaque gestionnaire fera les choses suivantes :

- ✓ Dire au jeu quelle couleur a été poussée
- ✓ En fonction du résultat
  - Demander une étape en plus
  - Afficher un message de fin de partie

Gestion du time out

Examinez dans la classe Simon comment se passe la gestion du time out.
Nous pensons que vous voyez maintenant comment coder le gestionnaire de time out.
Testez votre code. Il y aura sûrement une erreur du type « opération inter-threads non valide ».
Nous vous laissons chercher sur Internet l'origine et une solution à ce problème.

High score

Une fois que tout marche, gérez le high score. Vous devrez sans doute modifier la classe Simon.