Manuel technique du convertisseur romain

**Cette documentation technique est destinée au professeur en charge de l’évaluation du projet.**

Atelier TPI –Préparation au TPI

Table des matières

[Introduction 2](#_Toc481159687)

[Étude d’opportunité 3](#_Toc481159688)

[Analyse fonctionnelle 4](#_Toc481159689)

[Fonctionnalité 4](#_Toc481159690)

[Maquette 4](#_Toc481159691)

[Analyse organique 5](#_Toc481159692)

[Diagramme de classes 5](#_Toc481159693)

[Points techniques 5](#_Toc481159694)

[Protocole de test 6](#_Toc481159695)

[Plan de tests 6](#_Toc481159696)

[Rapport de tests 6](#_Toc481159697)

[Améliorations possibles 7](#_Toc481159698)

[Conclusion 8](#_Toc481159699)

[Table des illustrations 9](#_Toc481159700)

# Introduction

Je suis actuellement en train de réaliser un certificat fédéral de compétence au Centre de Formation Professionnelle Technique de Genève.  Dans le cadre de cette formation, je suis l’Atelier-TPI et je dois réaliser un projet.

Il s'agit ici de la documentation technique, cette dernière regroupe les analyses fonctionnelles et organiques, ainsi que le plan et le rapport de test. En outre, le planning du projet, les principales procédures et fonctions, ainsi que le diagramme d'états sont contenu dans ce manuel technique.

L’objectif de ce travail est comprendre et mettre en pratique ce que l'on attend de moi  dans le cadre d'un projet tel que le TPI.

Pendant ces prochaines semaines, je dois élaborer un programme en C#. Le sujet du projet a été donné par le professeur. Il s’agit de programmer un programme qui convertira les nombres décimaux en chiffre romain.

Avant l'introduction des chiffres arabes, qui remonte à la fin du Xème siècle, les Romains employaient les sept lettres suivantes, dites chiffres romains.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| I | V | X | L | C | D | M |
| 1 | 5 | 10 | 50 | 100 | 500 | 1000 |

* Les chiffres placés à la droite d'un autre ajoutent leur valeur à celle de cet autre s'ils lui sont égaux.

**II = 1 + 1 = 2**

**XXX = 10 + 10 + 10 = 30**

**CC = 100 + 100 = 200**

* Les chiffres V, L et D ne sont jamais répétés; les autres le sont rarement plus de trois fois.
* Tout chiffre placé à la droite d'un autre plus fort ajoute sa valeur à celle de cet autre.

**VI = 5 + 1 = 6**

**XV = 10 + 5 = 15**

**LX = 50 + 10 = 60**

* Tout chiffre placé à la gauche d'un autre plus fort retranche sa valeur de celle de cet autre.

**IV = 5 - 1 = 4**

**IX = 10 - 1 = 9**

**LD = 500 - 50 = 450**

# Étude d’opportunité

Après quelque recherche sur internet, l’idée de programmer un jeu du pendu peut paraitre inutile, vu qu’il en existe déjà de nombreux sites et applications disponible. Cependant, la plus part des applications existantes, selon mes recherches, sont remplies de publicités. De plus, ces applications ont des couleurs chaotiques et un look qui pique les yeux.

Prenons celui-ci en exemple:

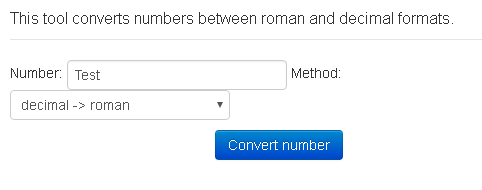


Figure 1 – Exemple de convertisseur

Le texte box n’est pas filtré, ce qui veut dire qu’il accepte tout (chiffres, lettres et ponctuation). A mon avis, c’est un point négatif, car le but du programme est de convertir les chiffres décimaux entiers en chiffre romain.

Pourtant, celui-ci n’a pas que des points négatifs. L’application est propre et harmonieuse.

Grâce à cet exemple, je peux vous dire que :

* Mon application aura des couleurs douces et accordé.
* Le texte box sera filtré (que des chiffres entiers acceptés)

# Analyse fonctionnelle

Cette partie de la documentation technique tiens à ce que vous compreniez les règles du programme. Je vais donc vous donner ici les consignes, c'est-à-dire vous expliquer comment le programme fonctionne précisément du point de vue de l’utilisateur final de l’application.

## Fonctionnalité

Le projet C# « Numérateur romain » permet de calculer le chiffre romain qui correspond à une valeur décimale (compris impérativement entre 1 et 3999) saisie par l’utilisateur.

## Maquette

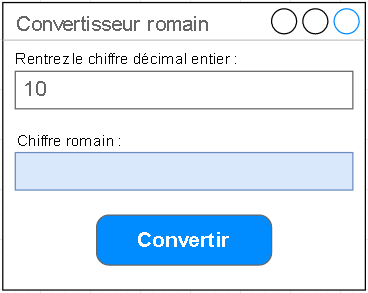


Figure 2 – Maquette

L’interface du programme contient :

* Un texte box filtré
  + Accepte que des chiffres entre 1 et 3999
* Un label
  + Affiche le chiffre décimal converti en romain
* Un bouton
  + Convertie le chiffre décimal en romain

# Analyse organique

L’analyse organique décrit le fonctionnement interne de l’application. Il s'agit d'analyser ce qui se passe en interne, du point de vue du développeur.

## Diagramme de classes

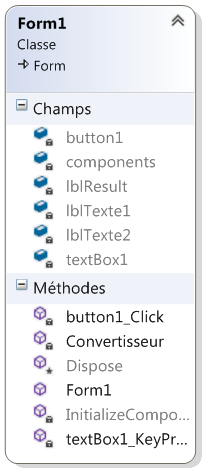


Figure 3 – Diagramme de classe

## Points techniques

* **TextBox**

Paramètre : Touche de clavier pressé

Chaque fois que l’utilisateur presse une touche sur le clavier un test est réalisé. Ce test filtre les caractères et n’accepte que les chiffres.

* **Bouton**

En cliquant sur le bouton « Convertir », le chiffre rentré dans le TextBox est récupéré et stocké. Ensuite, un test détermine si le chiffre est compris entre 1 et 3999.

Si oui : La fonction « Convertisseur » est appeler et son retour est affiché.

Si non : Un message d’erreur s’affichera.

* **Méthode « Convertisseur »**

Paramètre : Chiffre rentré converti en uint (0 à 4294967295).

Il y a deux tableaux, romain et décimale. Le programme prend le premier caractère du chiffre et ajoute les bons caractères romains en fonction.

# Protocole de test

## Plan de tests

|  |  |  |
| --- | --- | --- |
| **N° de test** | **Description test** | **Résultat attendu** |
| 1 | Rentrer des mots dans le TextBox. | Le TextBox reste vide, car des lettres ne sont pas acceptées. |
| 2 | Rentrer des caractères spéciaux dans le TextBox. | Le TextBox reste vide, car des caractères spéciaux ne sont pas acceptés. |
| 3 | Rentrer des chiffres a virgule dans le TextBox. | Le TextBox affichera seulement les chiffres et pas la virgule (ni un point). |
| 4 | Rentrer des chiffres entiers supérieurs à 3999 dans le TextBox. | Message d’erreur s’affiche. |
| 5 | Rentrer des chiffres entiers inférieurs à 1 dans le TextBox. | Message d’erreur s’affiche. |
| 6 | Qu’en cliquant sur le bouton le chiffre est converti en chiffre romain. | Afficher le chiffre romain approprié. |

## Rapport de tests

|  |  |  |  |
| --- | --- | --- | --- |
| **N° de test** | **Date test** | **Résultat obtenu** | **OK** |
| 1 | 28.04.2017 | Le TextBox reste vide, car des lettres ne sont pas acceptées. | ok |
| 2 | 28.04.2017 | Le TextBox reste vide, car des caractères spéciaux ne sont pas acceptés. | ok |
| 3 | 28.04.2017 | Le TextBox affichera seulement les chiffres et pas la virgule (ni un point). | ok |
| 4 | 28.04.2017 | Message d’erreur s’affiche. | ok |
| 5 | 28.04.2017 | Message d’erreur s’affiche. | ok |
| 6 | 28.04.2017 | Afficher le chiffre romain approprié. | ok |

# Améliorations possibles

C’est envisageable d’ajouter une fonctionnalité. Cette fonctionnalité permettra l’inverse. Pour le moment le programme converti les chiffres arabes en chiffre romain. Donc cette nouvelle fonction convertira les chiffres romains en chiffre arabe.

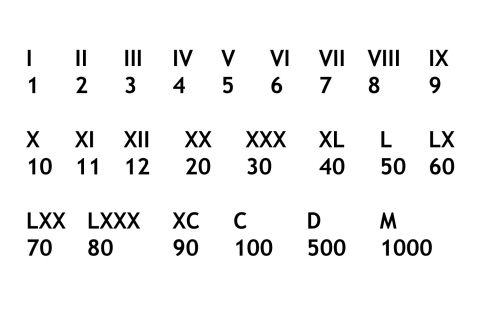


Figure 4 - Tableau de chiffre Romain et arabe

# Conclusion

Ce projet a été l’occasion pour moi d’affiner mes compétences de programmation en C# et aussi mes connaissance en chiffre romain. Mais ce projet a surtout aidé mes compétences de documenter un projet du début à la fin. La rédaction d’une documentation est un exercice qui est sensé découler de manière naturelle et la seule manière d’arriver à ce niveau est de pratiquer.

# Table des illustrations

[Figure 1 – Exemple de convertisseur 3](#_Toc481157873)

[Figure 2 – Maquette 4](#_Toc481157874)

[Figure 3 – Diagramme de classe 5](#_Toc481157875)

[Figure 4 - Tableau de chiffre Romain et arabe 7](#_Toc481157876)