intech
ECOLE SUPERIEURE
D'INGENIERIE INFORMATIQUE

Exercice : ITI.JsonParser

invenietis

1 Enoncé

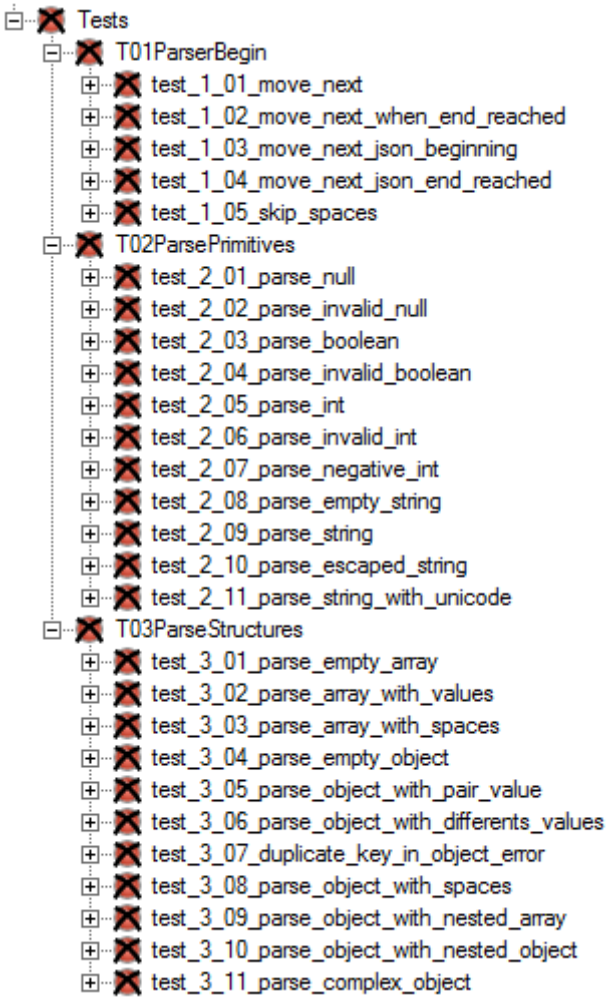
L'objectif de cet exercice est d'implémenter un parseur JSON.

Vous disposez pour cela d'une solution comprenant deux projets :

- ITI.JsonParser, est un squelette d'API à implémenter
- ITI.JsonParser.Tests, contient les tests unitaires

Pour faire tourner les tests il vous suffit de configurer le projet ITI.JsonParser.Tests en tant que projet de démarrage puis d'exécuter la solution.

Comme vous pouvez le constater, pour le moment, tous les tests sont rouges :



A vous de faire en sorte qu'ils passent en vert. Pour cela, vous avez le droit de faire ce que bon vous semble dans le projet ITI.JsonParser.

Les tests unitaires sont là pour spécifier de façon détaillée les fonctionnalités attendues. Cependant le présent document va décrire brièvement ce qui est attendu.

2 ITI.JsonParser

Le projet ITI.JsonParser contient la classe Parser dont l'API est la suivante :

```
public static class Parser
{
    public static object ParseNull( string value, ref int start, ref int count )...

    public static bool ParseBoolean( string value, ref int start, ref int count )...

    public static int ParseInt( string value, ref int start, ref int count )...

    public static string ParseString( string value, ref int start, ref int count )...

    public static object[] ParseArray( string value, ref int start, ref int count )...

    public static Dictionary<string, object> ParseObject( string value, ref int start, ref int count )...

    public static bool TryReadNextChar( string value, ref int start, ref int count, out char current )...

    public static bool TryReadNonBlankChar( string value, ref int start, ref int count, out char current )...
}
```

- les méthodes de base :
  - ParseNull
  - ParseBoolean
  - ParseInt
  - ParseString
  - ParseArray
  - ParseObject
- les méthodes utilitaires :
  - TryReadNextChar : un curseur qui tente de lire le prochain caractère de la chaine à partir de l'indice « start + 1 » inclus.
  - TryReadNonBlankChar : un curseur qui tente de lire le prochain caractère non vide/blanc de la chaine à partir de l'indice « start » inclus. Le but de cette méthode est de permettre de sauter les espaces lors du parsing

Il est recommandé d'implémenter les méthodes suivant l'ordre des tests.

3 ITI.JsonParser.Tests

Les tests sont de difficulté croissante et permettent de valider les fonctionnalités ci-dessous :

- T01ParserBegin :
  - le déplacement du curseur sur le prochain caractère
  - le déplacement du curseur à la fin de la chaine
  - le déplacement borné du curseur
  - le saut d'espaces du curseur lors du parsing
- T02ParsePrimitives
  - le parsing valide/invalide de valeur « null »
  - le parsing valide/invalide de valeur « bool »
  - le parsing valide/invalide de valeur « int »

    - le parsing valide/invalide, avec ou sans échappement/caractères Unicode, de valeur « string »
- T03ParseStructures
    - le parsing valide/invalide de valeur plate « array »
    - le parsing valide/invalide de valeur plate « object »
    - le parsing récursif de valeurs « array ou object » au sein de valeur « array ou object »

4 Barème

Chaque test rapporte un point. Sachant qu'il y a 27 tests à passer cela fait 27 points. 3 points supplémentaires seront accordés à la qualité du code. Au total, 30 points sont à acquérir qui seront transformés proportionnellement à une note sur 20.