Yassine HAMROUNI
Farès SID ATHMANE, Adem GUETTAF

RAPPORT

PROJET

INSPECTEUR JSON

Yassine HAMROUNI
Farès SID ATHMANE, Adem GUETTAF

Sommaire

Introduction……3

Développement……...4 - 10

Fonctionnalités……4 - 6
Présentation structure……..……...7
Explication Arbre……..…….8-9
Énumération structure de données…….10

Conclusion…….....11

Conclusion générale……..11
Conclusions personnelles……...….11

Yassine HAMROUNI
Farès SID ATHMANE, Adem GUETTAF

Introduction

En groupe de trois personnes :

- Yassine HAMROUNI
- Farès SID ATHMANE
- Adem GUETTAF

Le projet « Inspecteur de JSON » consiste à proposer une alternative pour faciliter les tests lors de développement de WebAPI.

En utilisant le langage Java pour améliorer l'affichage de données structurées en format JSON, et en fournissant une application qui permet d'afficher de manière claire les données sous format JSON (des retours à la ligne, des espacements et des séparateurs appropriés) afin de faciliter la lisibilité visuelle.

Fonctionnalités

Nous avons donc réussi à réaliser les fonctionnalités demandées comme :

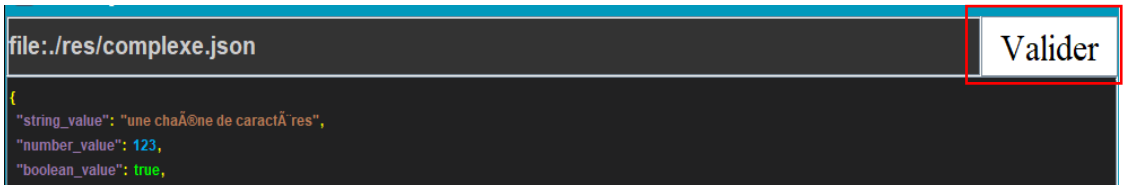
➢ Exécution du programme

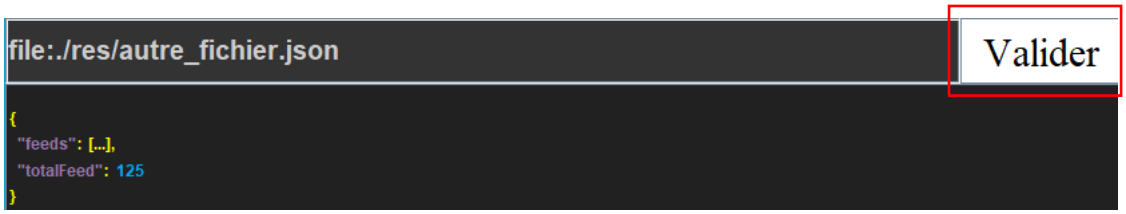
L'exécution du programme est possible via la console et via l'interface graphique. Le bouton *Valider* fonctionne et rafraîchit la zone d'affichage.

Console

```
$ java -jar Inspecteur.jar 'file:./res/complexe.json'
{
  "string_value": "une chaîne de caractères",
  "number_value": 123,
  "boolean_value": true,
  "array_value": [
    1,
    2,
    3,
    4
  ],
```

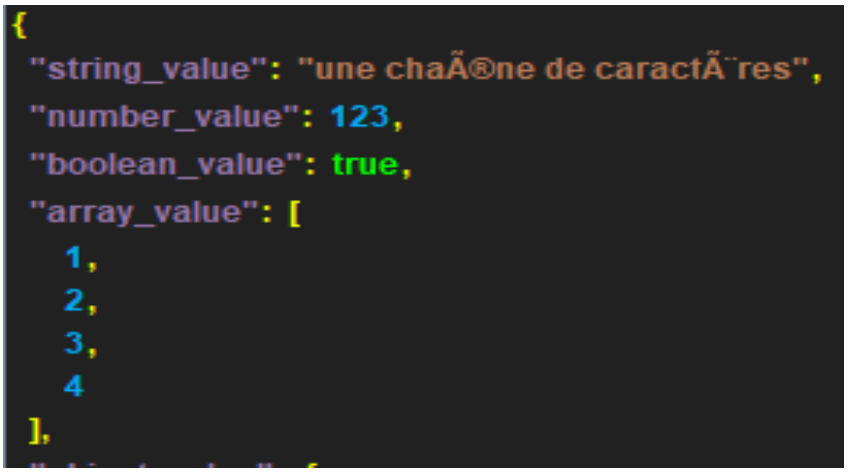
Interface graphique





➢ La coloration syntaxique

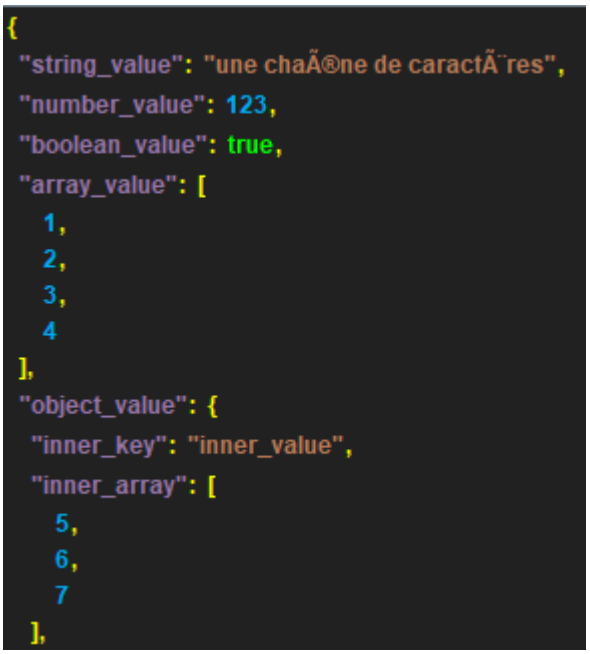
Pour que le contenu soit facilité au niveau de la lecture, nous avons réussi à distinguer les différentes demandées (et d'autres) par couleurs :



- ✓ Séparateurs : Jaune
- ✓ Noms des membres d'un objet : Violet
- ✓ Valeurs de type nombre : Bleu
- ✓ Valeurs de type chaîne : Marron
- ✓ Autres : Vert foncé

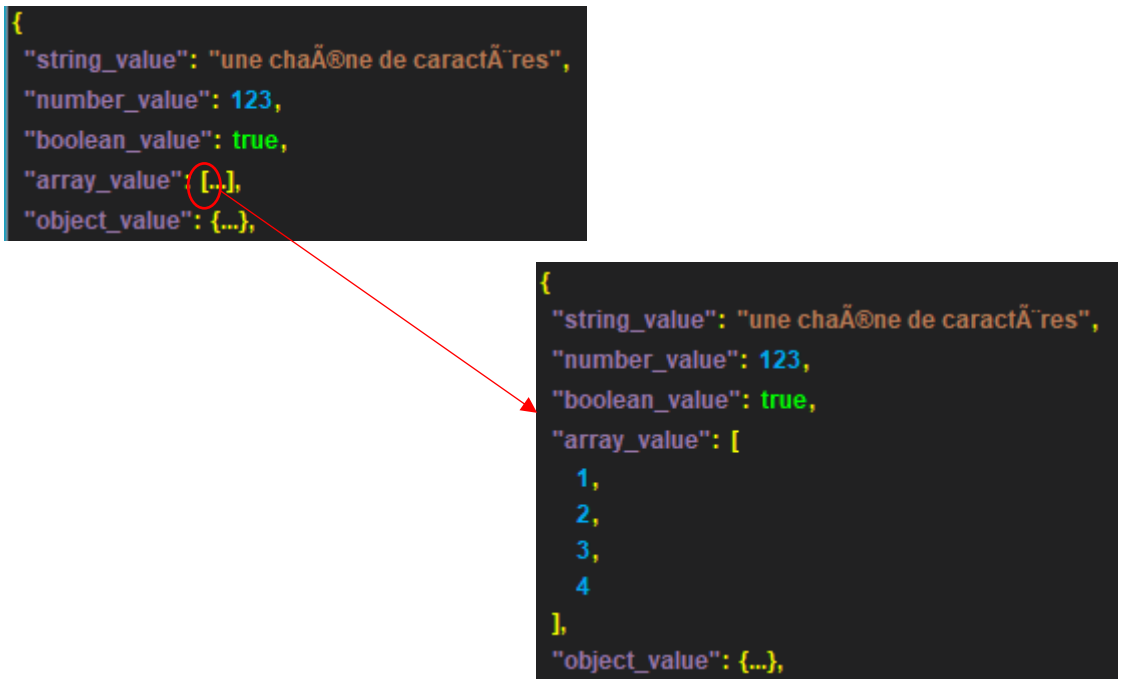
➢ Le repli syntaxique

L'affichage initial est replié et l'utilisateur peut déplier ou replier avec un clic de souris mais également déplier tout d'un coup avec le bouton *Déplier* :



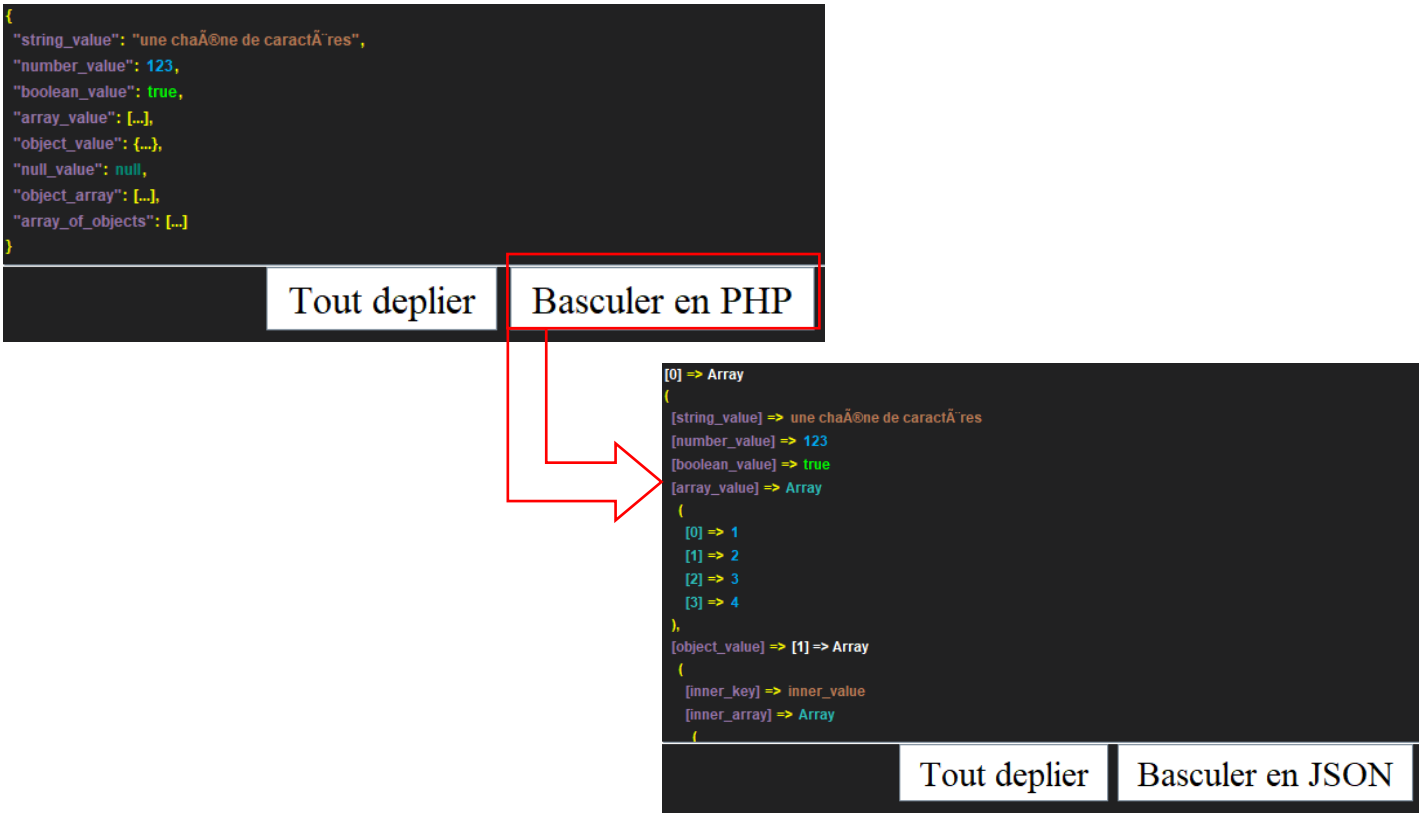
➢ Repli avec clic de souris

L'utilisateur peut cliquer sur la valeur qu'il souhaite déplier ou replier.



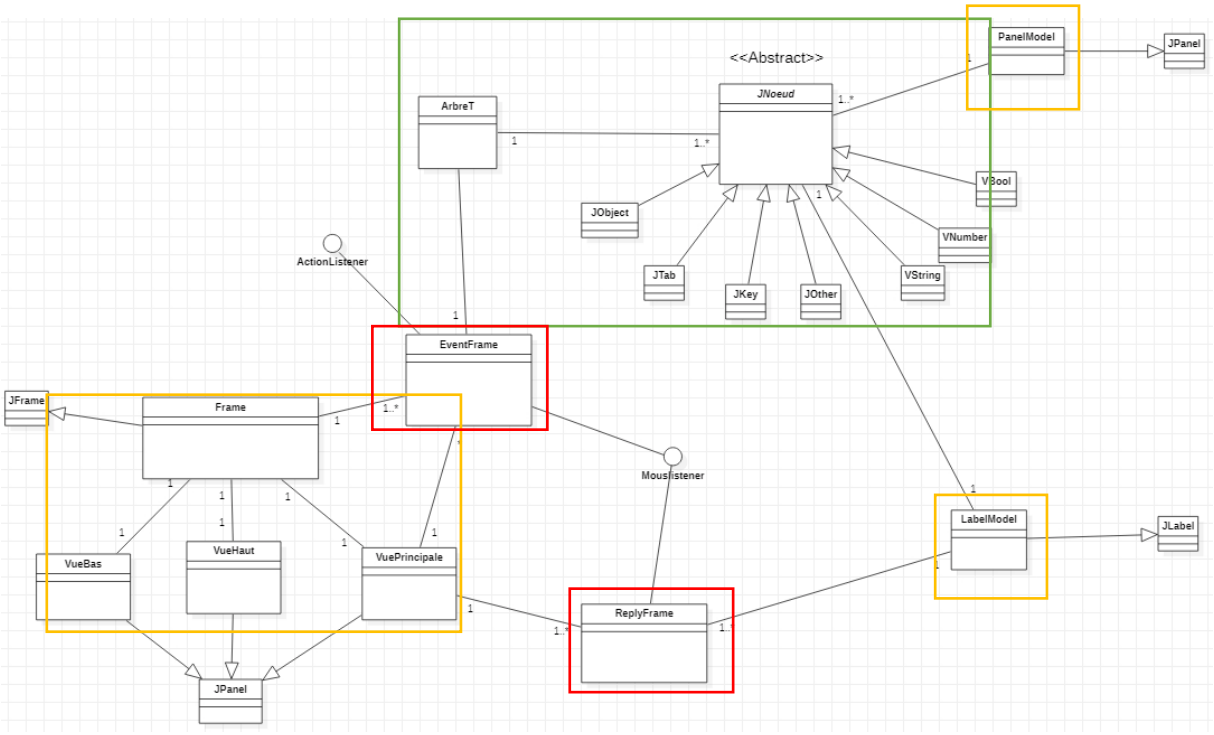
➢ Traduction en PHP

La possibilité de basculer de langage entre JSON et PHP est possible via le bouton *Basculer en PHP / JSON* :



Yassine HAMROUNI
Farès SID ATHMANE, Adem GUETTAF

Structure du programme



L'encadrement en vert représente le modèle :

JNoeud est une classe abstraite qui représente chaque nœud de l'arbre. Les différentes classes qui héritent de cette dernière, ont chacun une catégorie propre à eux pour permettre la bonne disposition du fichier JSON.

L'encadrement en orange représente la vue :

VueHaut contient la barre de recherche d'un fichier JSON, ainsi qu'un bouton *Valider* qui permet donc de rafraîchir la *VuePrincipale*. Cette VuePrincipale contient donc l'affichage JSON avec la possibilité de basculer en PHP via la VueBas. La VueBas contient les boutons *Tout déplier* / *replier* et *Basculer en PHP* / *JSON* .

L'encadrement en rouge représente le contrôleur :

EventFrame assure la fonctionnalité des boutons de *VueHaut* et *VueBas. ReplyFrame* fait fonctionner le *repli* de *VuePrincipale* .

Yassine HAMROUNI
Farès SID ATHMANE, Adem GUETTAF

Arbre de syntaxe abstraite

Notre arbre de syntaxe abstraite a été fait de la sorte :

Il possède des nœuds, chaque nœud est une classe qui hérite de JNoeud

ArbreT: Classe de l'arbre, l'arbre est créé avec la méthode createTree()

JNoeud: Classe abstraite représentant les nœuds de l'arbre, contenant les méthodes des nœuds

Les classes héritant de JNoeud sont :

JObject: Classe représentant les nœuds d'un objet JSON, récupérée par la méthode takeObject() de l'arbre. Ses fils principaux sont les JKey.

JKey: Classe représentant les nœuds de clé d'un objet JSON, elle récupéré en même temps que l'objet par la méthode getString() de l'arbre. Peut avoir différents types de fils, comme un JTab, récupérés avec la méthode getValue().

JTab: Classe représentant les nœuds d'un tableau JSON, récupérée dans l'arbre avec la méthode takeTab(). Ses fils sont ses valeurs (JObject, autre JTab, VNumber, VString ou JOther).

VNumber: Classe représentant les nœuds de valeur numérique.

VString: Classe représentant les nœuds de chaîne de caractères.

VBool: Classe représentant les nœuds de booléen.

JOther: Classe représentant les nœuds de valeurs qui ne sont ni numériques, ni des chaînes de caractères, ni des booléens.

L'arabe s'affiche de manière récursive, Chaque nœud écrit sa valeur.

Voici un exemple de l'utilisation de l'arbre :

{"status":"ok","size":3,"values":[0.5,null,1e1]}

Tout d'abord le Nœud JObject de la racine est créé,

Ensuite un Nœud JKey (status) est créé, puis un nœud VString (ok) est créé.

Maintenant, VString (ok) devient un fils de JKey (status) et JKey (JKey) devient un fils de JObject.

La procédure est la même pour le suivant mais là c'est un VNumber qui est créé.

Pour finir, un JKey(values) est créé puis un nœud JTab est créé.

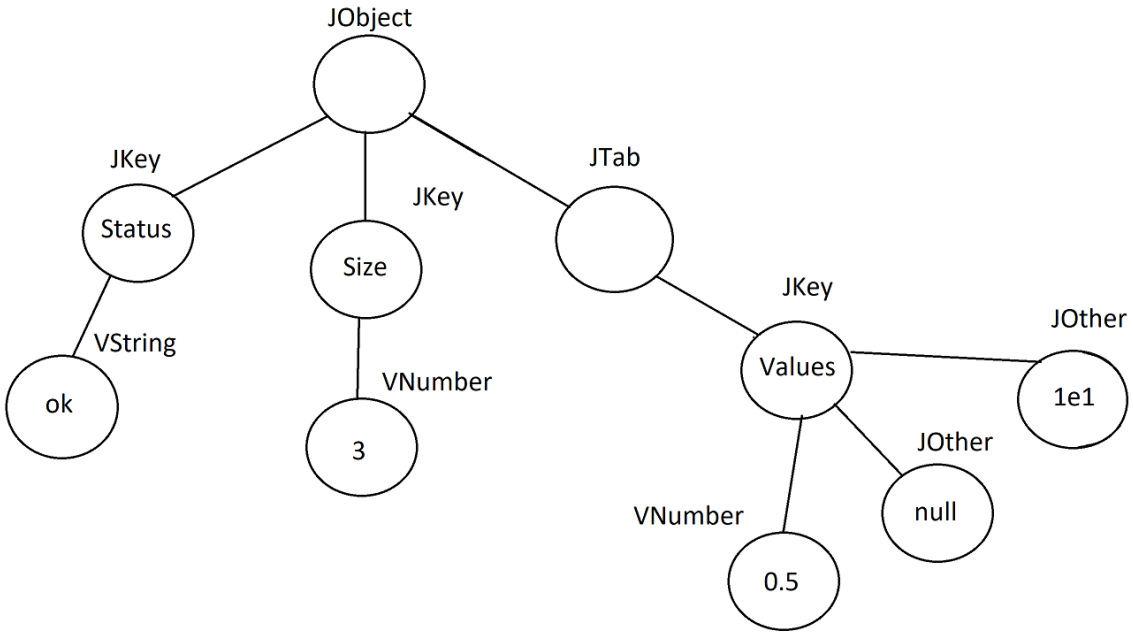
Un nœud VNumber (0.5) est créé puis ajouter au JTab,

Un nœud JOther (null) est créé puis ajouter au JTab,

Un nœud JOther (1e1) est créé puis ajouter au JTab.

Le JTab est ajouté à la JKey (values) qui elle-même est ajouté à la racine JObject.

Voici l'arbre à la fin de cet exemple :



Yassine HAMROUNI
Farès SID ATHMANE, Adem GUETTAF

Structures de données abstraites

Nous avons utilisé l'arbre de syntaxe abstrait qui était à faire et une List qui est une LinkedList de JNoeud dans notre classe abstraite JNoeud. Elle permet de stocker les fils des différents nœuds de notre arbre. Nous avons choisi d'utiliser cette structure de données car nous sommes habitués à l'utiliser.

Yassine HAMROUNI
Farès SID ATHMANE, Adem GUETTAF

Conclusion

Conclusion générale :

Ce projet a été une expérience enrichissante pour notre groupe, nous avons été amenés à apprendre et améliorer notre compréhension de la grammaire JSON ainsi que nos compétences en développement Java. Bien que des difficultés aient été rencontrées sur la compréhension du sujet et les complexités techniques, nous avons apprécié la réalisation de ce projet. En dehors de l'aspect technique, nous sommes des étudiants de la même classe depuis la première année du BUT. Nous avons donc réussi à s'adapter en fonction des facilités et difficultés de chacun pour se répartir correctement le travail à faire.

Nos remerciements.

Conclusions personnelles :

Yassine :

Etant donné que je suis en « burn-out », je n'ai pas pu exploiter mes capacités lors de ce projet. Cependant, j'ai donné le meilleur de moi-même pour que mes camarades et moi apportons un projet concret. Je tiens à remercier mes camarades et les enseignements reçus de l'enseignant en charge du projet.

Farès :

J'ai apprécié réaliser ce projet qui m'a permis d'utiliser les compétences acquises durant ce semestre et de les renforcer. J'ai pu découvrir la grande utilité d'un arbre et d'une classe abstraite et je suis persuadé que cela me sera utile.

Adem :

Malgré les difficultés rencontrées, j'ai pu développer certaines de mes compétences en langage java grâce à mon groupe qui m'ont accompagné dans ce travail. J'ai également pu me documenter et comprendre de manière plus claire le langage JSON qui m'était inconnu. J'ai trouvé ce projet assez épuisant et compliqué mais reste tout de même une bonne expérience.