

WHILE - Un interpréteur visuel pour les langages WHILE(A)

Guide de l'utilisateur

Jonathan Ferreira

Septembre 2020

Table des matières

1	Introduction	3
1.1	Prérequis	3
2	Lancement du logiciel	3
2.1	Lancement en ligne	3
2.2	Téléchargement pour un usage hors ligne	3
3	Utilisation du logiciel	4
3.1	Entrée d'un programme WHILE	4
3.2	Entrée de la valeur de départ V0	4
3.3	Exécution du programme	5
3.3.1	Lancer la prochaine instruction	5
3.3.2	Lancer le reste du programme	5
3.3.3	Arrêt du programme	5

1 Introduction

Ce guide explique l'utilisation de l'interpréteur visuel (ci-après nommé "logiciel" ou "application") pour la famille de langages WHILE(A) (ci après nommé "langage WHILE" ou "WHILE").

1.1 Prérequis

Comme l'utilisation de WHILE est une partie intégrante du logiciel, il est nécessaire de connaître la syntaxe du langage WHILE afin de pouvoir utiliser le logiciel.

2 Lancement du logiciel

2.1 Lancement en ligne

Le logiciel consiste en une simple page web. En tant que tel, le lancement de celui-ci est extrêmement simple. Un accès à Internet est requis. Il suffit à l'aide d'un navigateur web moderne (Firefox, Chrome, Edge, etc.) d'aller à l'URL <https://alphajon.github.io/WhileInterpret/> et le logiciel se chargera directement dans le navigateur. Vous verrez ainsi quelque chose ressemblant à la figure 1 suivante



FIGURE 1 – Logiciel WHILE

2.2 Téléchargement pour un usage hors ligne

Il est possible de télécharger le logiciel pour un usage ultérieur en local sans accès à Internet. Pour télécharger le logiciel, ouvrez l'application en suivant les instructions de la section 2.1, puis effectuez un clic

droit sur une section vide de la page web de l'application. Sélectionnez ensuite "Enregistrer sous..." puis choisissez un dossier de destination pour le téléchargement.

Une fois l'application téléchargée, il suffit d'ouvrir le fichier nouvellement créé avec un double-clic, ce qui devrait charger automatiquement l'application dans un navigateur web.

Cette méthode n'est cependant pas recommandée, car les mises à jour pour le logiciel ne seront pas disponibles via la version hors ligne.

3 Utilisation du logiciel

Le logiciel est conçu de manière à être très simple d'utilisation. En effet, il ne comporte qu'une poignée de boutons, et un faible nombre de champs texte.

Pour pouvoir lancer l'interpréteur, l'application a besoin de connaître le programme WHILE à exécuter, ainsi que la valeur d'entrée V_0 qui doit être assignée à ce programme. Des programmes exemples sont cependant fournis via les différents boutons « Lancer la démo » présents sur la partie inférieure de l'application, qui pré-remplissent les champs requis.

3.1 Entrée d'un programme WHILE

Le programme WHILE que vous souhaitez exécuter doit être entré dans le champ nommé « Code à exécuter » présent dans l'application. Le programme doit être syntaxiquement valide. Si le programme n'est pas syntaxiquement valide, l'interpréteur ne se lancera pas, et une erreur sera affichée à l'écran, comme sur la figure 2.

3.2 Entrée de la valeur de départ V_0

La valeur d'entrée doit être indiquée dans le champ nommé « Entrée (V_0) ». Afin d'éviter un usage excessif d'imbrications, de parenthèses et d'expressions « cons », une syntaxe spécifique pour l'entrée est fournie.

Les atomes doivent être entourées de guillemets doubles. Ceci permettra de différencier par exemple entre "*nil*" et la valeur spéciale *nil*. Pour indiquer des couples, les virgules , et les points-virgules ; peuvent être utilisés pour séparer des valeurs (des atomes ou d'autres couples), et ont un comportement différent :

La virgule sépare des valeurs successives de telle sorte à ce que l'entrée (" $t1$ ", " $t2$ ", " $t3$ ", " $t4$ ") représente l'arbre $(t1, (t2, (t3, (t4, nil))))$. Le point-virgule sépare deux valeurs, et représente directement un couple. (" $t1$ "; " $t2$ ") représente donc $(t1, t2)$. (" $t1$ ", " $t2$ ") aurait à la place représenté $(t1, (t2, nil))$. Il ne peut y avoir qu'une seule valeur (atome ou couple) du côté droit du point-virgule.

Il est donc aussi possible d'écrire (" $t1$ ", " $t2$ ", " $t3$ "; " $t4$ "), avec un point-virgule entre les deux derniers éléments d'un groupe de valeurs **uniquement**, qui permet de placer ces deux éléments en fin d'arbre dans le même couple. Ainsi, (" $t1$ ", " $t2$ ", " $t3$ "; " $t4$ ") représente l'arbre $(t1, (t2, (t3, t4)))$, mais (" $t1$ ", " $t2$ "; " $t3$ ", " $t4$ "; " $t5$ ") est une syntaxe invalide. Pour corriger cela, on peut utiliser (" $t1$ ", (" $t2$ "; " $t3$ ")", " $t4$ "; " $t5$ ") si le point-virgule ne doit s'appliquer que sur l'élément suivant, ou (" $t1$ ", " $t2$ "; (" $t3$ ", " $t4$ "; " $t5$ ")") si le point-virgule doit s'appliquer sur le reste des éléments

Comme pour l'entrée du programme, si l'entrée de la valeur V_0 n'est pas syntaxiquement valide, l'interpréteur ne se lancera pas, et une erreur sera affichée à l'écran.

3.3 Exécution du programme

Une fois que la valeur d'entrée et le programme donnés au logiciel sont entrés, et si tous deux sont valides, le logiciel débutera l'interprétation du programme. L'application ressemblera alors à la figure 3.

Le programme en cours d'exécution est représenté, avec une mise en surbrillance de la prochaine ligne à exécuter sur la moitié gauche de l'application, et le contenu de la mémoire du programme en cours est affichée sur la moitié droite de l'application. Vous pouvez alors interagir avec le programme via les boutons proposés. Si aucune ligne du programme n'est mise en surbrillance, cela signifie que le programme a terminé son exécution.

3.3.1 Lancer la prochaine instruction

Exécute une seule instruction (celle mise en surbrillance par l'application). L'instruction suivante (s'il y en a une) est alors mise en surbrillance, indiquant l'instruction qui sera exécutée en appuyant sur le bouton une fois de plus.

3.3.2 Lancer le reste du programme

Ordonne au logiciel de lancer le reste du programme, à hauteur d'une instruction toutes les X secondes, où X représente la valeur qui était présente dans le champ "Délai" au moment de l'appui du bouton. Une valeur de 0 lance instantanément tout le reste du programme en cours jusqu'à la fin. Le bouton "Pause" interrompt l'exécution automatique.

3.3.3 Arrêt du programme

Interrompt le programme en cours, et retourne sur le formulaire d'entrée du programme de la figure 1.

Entrée (V_0): "a", "b", "c", "d"

Code à exécuter:

```
{  
  V1 := nil;  
  While V0 {  
    V1 := (cons (hd V0) V1);  
    V0 := (tl V0)  
  }  
}
```

Expected "if", "while", "{", "}", or variable name but end of input found.

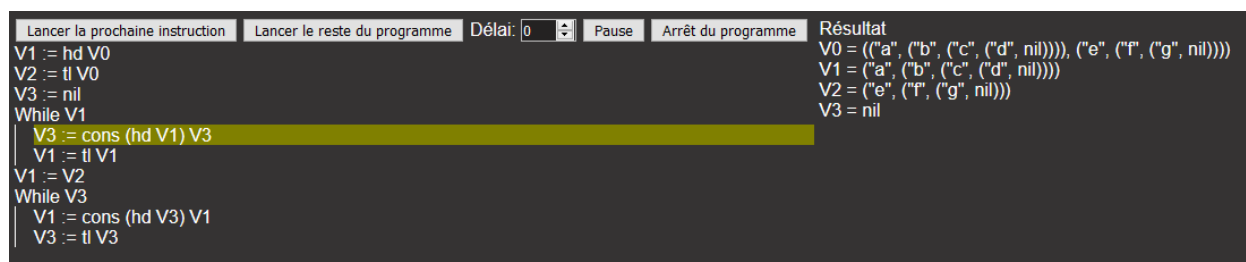
Démarrer l'exécution

Charger la démo "Reverse"

Charger la démo "Concat"

■ Surlignage récursif (?)

FIGURE 2 – Exemple d'erreur



The image shows a screenshot of a program interpreter interface. At the top, there are four buttons: "Lancer la prochaine instruction", "Lancer le reste du programme", "Délai: 0" (with a spinner), and "Pause". To the right of these buttons is a button labeled "Arrêt du programme". Below the buttons, the program code is displayed on the left, and the current state of variables is shown on the right under the heading "Résultat".

Code:

```
V1 := hd V0
V2 := tl V0
V3 := nil
While V1
  V3 := cons (hd V1) V3
  V1 := tl V1
V1 := V2
While V3
  V1 := cons (hd V3) V1
  V3 := tl V3
```

The line `V3 := cons (hd V1) V3` is highlighted in yellow.

Résultat:

```
V0 = (("a", ("b", ("c", ("d", nil)))), ("e", ("f", ("g", nil))))
V1 = ("a", ("b", ("c", ("d", nil))))
V2 = ("e", ("f", ("g", nil)))
V3 = nil
```

FIGURE 3 – Interprétation d'un programme en cours