GuideBook Mcadalang

Introduction

Mcdalang est un langage de programmation pédagogique inspiré des langages classiques comme C, Java ou Python, conçu pour apprendre les bases de la programmation structurée. Il prend en charge les fonctions, les variables, les structures de contrôle, les expressions arithmétiques, et plus encore.

Fonctionnalités de l'interface Mcdalang

Présentation

L'interface Mcdalang est un outil permettant de traduire du code écrit en Mcdalang vers plusieurs langages de programmation, tels que Python, C, C++, Rust, etc.

Comment l'utiliser ?

Pour commencer, saisissez votre code Mcdalang dans l'éditeur de gauche. Une fois votre code prêt, sélectionnez le langage cible grâce aux boutons situés à droite de l'écran. Cliquez ensuite sur le bouton "Run", en bas de l'interface : votre code sera traduit dans le langage sélectionné, et le résultat s'affichera dans l'éditeur de droite.

Fonctions de l'interface

- Switch (en bas à gauche) : Permet d'inverser les deux éditeurs de code. Cela vous donne la possibilité de changer le sens de traduction, par exemple pour convertir du Python vers Mcdalang.
- Mcdabot (en bas à droite) : Ce bouton ouvre une fenêtre pour vous aider à apprendre à coder en Mcdalang.
- Options (en haut à gauche) : Accédez à divers paramètres de personnalisation, comme l'activation du mode "Auto Run". Ce mode permet de lancer automatiquement la traduction dès que vous modifiez le code dans l'éditeur de gauche, sans avoir besoin de cliquer sur "Run".
- Import / Export (à côté du bouton Options) :
    - Import : Chargez facilement un fichier ou un extrait de code dans l'éditeur de gauche.
    - Export : Sauvegardez le code traduit dans tous les langages pris en charge.

Syntaxe et utilisation du langage Mcdalang

Types de données

Mcdalang prend en charge les types suivants :

- entier : nombre entier (ex : 42)
- flottant : nombre à virgule (ex : 3.14)
- char : caractère unique (ex : 'a')
- chaine : chaîne de caractères (ex : "Bonjour")
- bool : booléen (true ou false)
- vide : pour les fonctions ne retournant rien

Déclaration de variables et constantes
var entier x
var chaine message = “Salut”
const flottant PI = 3.1415

- var : pour une variable modifiable
- const : pour une constante (valeur fixe)
- L’affectation (=) est optionnelle lors de la déclaration

Assignation
x = 10
message = “Nouveau message”

Expressions et opérations
Opérations supportées :
+ : addition
- : soustraction
* : multiplication
/ : division flottante
// : division entière
% : modulo
^ : puissance

Comparaisons :
== : égal à
!= : différent de
< : inférieur
<= : inférieur ou égal
> : supérieur
>= : supérieur ou égal

Logiques :
ET : conjonction (ex : (a > 0) ET (b < 10))
OU : disjonction (ex : (x == 0) OU (y == 0))
NON : négation (ex : NON(a == b))

Autres :
& : concaténation de chaînes
++ : incrément
-- : décrément

Affichage
afficher(“Bonjour”)
afficher(x + 5)

Fonctions (méthodes)

Déclaration :

```
methode entier addition(entier a, entier b) {
        return a + b
}
```

- Les paramètres sont typés et séparés par des virgules
- return : permet de renvoyer une valeur
- Utilisez “vide” comme type de retour si la méthode ne retourne rien

Appel de fonction :

```
resultat = addition(3, 4)
```

Structures conditionnelles

```
si (x > 0) {
   afficher("x est positif")
} sinon {
   afficher("x est négatif ou nul")
}

si (a > b) {
   afficher("A est supérieur à B")
} snsi (a < b) {
   afficher("A est inférieur à B")
} sinon {
   afficher("A et B sont égaux")
}
```

Boucles

```
tantque (x < 5) {
   afficher(x)
   x = x + 1
}

faire {
   afficher(x)
   x = x + 1
} tantque (x < 5)

pour (i = 0; i < 3; i = i + 1) {
   afficher(i)
}
```

Blocs

Les blocs sont définis entre accolades :

```
{
   var entier x = 0
   afficher(x)
}
```

Commentaires

- Ligne unique :

```
|| Ceci est un commentaire sur une ligne
```

- Multiligne :

```
/**
Ceci est un
commentaire multiligne
*/
```

Exemple complet

```
|| Fonction pour calculer la moyenne de deux notes
methode flottant calculerMoyenne(entier note1, entier note2) {
   return (note1 + note2) / 2.0
}

|| Fonction pour déterminer si l'élève a réussi
methode chaine resultat(flottant moyenne) {
   si (moyenne >= 10) {
      return "Admis"
   } snsi (moyenne >= 8) {
      return "Rattrapage"
   } sinon {
      return "Échec"
   }
}

|| Fonction principale
methode vide principal() {
   const entier NB_ELEVES = 3
   var entier i = 0

   tantque (i < NB_ELEVES) {
      afficher("Élève numéro " & i + 1)

      var entier note1 = i * 3 + 8
      var entier note2 = i * 2 + 9

      afficher("Note 1 = " & note1)
      afficher("Note 2 = " & note2)

      var flottant moyenne = calculerMoyenne(note1, note2)
```

```
        afficher("Moyenne = " & moyenne)

        var chaine decision = resultat(moyenne)
        afficher("Résultat : " & decision)

        i = i + 1
        afficher("")
    }
}

|| Appel de la fonction principale
principal()
```