

GUIDE POUR CONFIGURER DE VOTRE ENVIRONNEMENT GOLANG

Prérequis

Je vais commencer par vous apprendre à lancer votre tout premier programme Go !

Mais avant de d'exécuter votre programme Go il vous faut :

- Un éditeur de texte
- Un compilateur GO

Éditeur de Texte

Pour ma part j'utilise un éditeur de texte gratuit et open source à savoir [Visual studio Code](#) avec l'extension [Go](#), c'est une extension qui permet entre autres de :

- Faire de l'auto complétions
- Obtenir des informations quand vous passez la souris sur votre code
- Mettre en forme votre code
- Déboguer votre code
- Importer des paquets automatiquement
- etc ...

Libre à vous d'utiliser autre chose, tant que votre éditeur de texte vous permet d'écrire du texte et de le sauvegarder alors c'est suffisant et vous pouvez passer à

l'étape suivante !

Téléchargez le compilateur Go

Pour transformer votre code source en langage machine afin que votre CPU puisse exécuter votre programme il faut **installer le compilateur GO**. Le compilateur Go est disponible sur différents OS (Linux, Mac OS X et Windows)

Téléchargez la dernière version du compilateur [en cliquant ici](#).

Installation du compilateur Go sous Linux

Téléchargez l'archive et extrayez l'archive dans le dossier `/usr/local/` avec la commande suivante

```
sudo tar -C /usr/local -xzf [NOM_ARCHIVE].tar.gz
```

Ensuite il faut ajouter le binaire de votre compilateur `/usr/local/go/bin/` à votre variable d'environnement **PATH** en tapant la commande suivante :

```
echo "export PATH=\$PATH:/usr/local/go/bin" >> ~/.bashrc  
source ~/.bashrc
```

Installation du compilateur Go sous Windows

Exécuter le fichier MSI et suivez les instructions de votre fenêtre pour installer les outils Go.

L'architecture de Go

Avant de lancer notre programme, il serait intéressant de comprendre l'architecture de go.

Déjà lors de votre installation des outils, go vous a créé plusieurs variables d'environnement dont deux variables d'environnements importantes nommées respectivement :

- **GOROOT** : contient comme valeur un dossier destiné à votre compilateur
- **GOPATH** : contient comme valeur un répertoire d'espace de travail, c'est ici qu'il cherche les packages que vous importez

Si vous ne connaissez pas la valeur de variable d'environnement **GOPATH** alors vous pouvez utiliser la commande suivante :

```
go env
```

Si vous vous placez sur votre variable d'environnement **GOPATH**, vous observerez l'arborescence suivante :

```
$GOPATH
|_____ bin
|_____ pkg
|_____ src
```

Voici la description de chaque dossier :

- **src** : contient les sources de votre projet et des librairies utilisées
- **pkg** : contient des fichiers avec l'extension .a (a pour archivé), ces fichiers sont une version compilée de votre code source original,
- **bin** : contient des commandes exécutables.

Tester votre compilateur GO

Maintenant que vous avez compris le fonctionnement de l'architecture de GO, placez vous sur le répertoire `$GOPATH/src/`.

A fin de tester votre compilateur, il faut créer un programme GO et l'exécuter.

Créez un fichier et nommez le `test.go` et mettez le code suivant puis sauvegardez :

```
package main

import "fmt"

func main() {
    fmt.Println("Hello, World!")
}
```

Testez ensuite votre programme avec la commande suivante :

```
go run test.go
```

Si tout se passe bien vous devriez avoir comme sortie :

```
Hello, World!
```

Bravo vous avez appris à lancer votre premier programme en GO "clap clap"? !

Dans le prochain chapitre je vais vous expliquer un peu plus en détail le code que vous venez d'exécuter.