Tableaux

<https://steemit.com/utopian-io/@raptorjesus/golang-10-les-tableaux>

Def

* Tableaux : collection d’éléments qui appartiennent à un même type (collections d’entiers n : 1, 2, 3…)
* Tableaux Golang ne tolèrent pas le mélange des types dans le même tableau

1. Déclaration

* ***n[T]***
* ***n***: nombre d’éléments
* ***T*** : type de chaque élément
* Plusieurs façons de déclarer un tableau
* *package main*

*import ‘’fmt ‘’*

*func main () {*

*var a [3]int*

*fmt.Println(a)*

*}*

* ***var a[3]*** déclare un tableau d’entiers de longueur 3 (0 affecté automatiquement à tous les éléments du tableau)
* L’exécution du programme va nous donner ***[0 0 0]***

*package main*

*import ‘’fmt ‘’*

*func main () {*

*var a [3]int*

*a[0] = 23*

*a[1] = 28*

*a[2] = 99*

*fmt.Println(a)*

*}*

* L’exécution du programme va donner ***[23 28 99]***
* *package main*

*import ‘’fmt’’*

*func main() {*

*a := [3]int{23, 28, 99}*

*fmt.Println(a)*

*}*

* Déclaration concise (pas nécessaire d’affecter une valeur à tous les éléments)
* *package main*

*import ‘’fmt’’*

*func main() {*

*a := [3]int{12}*

*fmt.Println(a)*

*}*

* une seule valeur est fournie donc les 2 autres valeurs seront 0
* l’exécution du programme donnera ***[12 0 0]***
* *package main*

*import ‘’fmt’’*

*func main() {*

*a := […]int{12, 102, 85, 99}*

*fmt.Println(a)*

*}*

* ***[…]***: pour que le compilateur trouve lui-même la longueur du tableau

1. Les tableaux sont des types de valeurs

* Quand un tableau est affecté à une nouvelle variable une copie du tableau original est affectée à la nouvelle variable
* Si des changements touchent la nouvelle variable cela ne modifiera pas le tableau original
* *package main*

*import ‘’fmt’’*

*func main() {*

*a := […]string{‘’USA’’, ‘’China’’, ‘’India’’, ‘’Germany’’, ‘’France’’}*

*b := a*

*b[0] = ‘’Singapore’’*

*fmt.Println(‘’a is’’, a)*

*fmt.Println(‘’b is’’, b)*

*}*

* ***a is [USA China India Germany France]***
* ***b is [Singapore USA China India Germany France]***
* *package main*

*import ‘’fmt’’*

*func changeLocal(num [5]int) {*

*num[0] = 55*

*fmt.Println(‘’dans la fonction’’, num)*

*}*

*func main() {*

*num := […]int{5, 7, 8, 8, 9}*

*fmt.Println(‘’avant de passer le tableau à la fonction’’, num)*

*changeLocal(num)*

*fmt.Println(‘’après avoir passé le tableau à la fonction’’, num)*

*}*

* Le tableau ***num***est passé par valeur à la function ***changeLocal***
* L’exécution du programme donne

***# avant de passer le tableau à la fonction [5 7 8 8 9]***

***# dans la fonction [5 7 8 8 9]***

***# après avoir passé le tableau à la fonction [5 7 8 8 9]***

1. Longueur tableau

* *package main*

*import ‘’fmt’’*

*func main() {*

*a := […]float64{67.7, 89.8, 21, 78}*

*fmt.Println(‘’la longueur est’’, len(a))*

*}*

* ***len*** : pour trouver la longueur du tableau
* l’exécution du programme donne : longueur est 4

1. Itération d’un tableau

* *package main*

*import ‘’fmt’’*

*func main() {*

*a := […]float64{67.7, 89.8, 21, 78}*

*for i := 0 ; i < len(a) ; i++ {*

*fmt.Printf(‘’%d ème élément de a est %.2f\n’’, i, a[i])*

*}*

*}*

* La boucle for peut être utilisée pour itérer sur tous les éléments d’un tableau
* L’exécution du programme donne

# 0 ème élément de a est 67.70

# 1 ème élément de a est 89.80

# 2 ème élément de a est 21.00

# 3 ème élément de a est 78.00

* On a le même résultat en utilisant range
* *package main*

*import ‘’fmt’’*

*func main() {*

*a := […]float64{67.7, 89.5, 21, 78}*

*sum := float64(0)*

*for i, v := range a {*

*fmt.Printf(‘’%d ème element de a est%.2f\n”, I, v)*

*sum += v*

*}*

*Fmt.Println(‘’\n somme de tous les éléments de a est », sum)*

*}*

1. Types de tableaux

* ***var[…]int{}***: entiers naturels
* ***var[…]float{}*** : décimaux (float32 : moins précis mais plus rapide/float64 : plus précis mais moins rapide)
* ***var[…]string{}*** : chaine de caractères
* ***var[…]bool{}***
* : boléen (true/false)

1. Tableaux multidimensionnels

* *package main*

*import ‘’fmt’’*

*func printarray(a [3][2]string) {*

*for \_, v1 := range a {*

*for \_, v2 := range v1 {*

*fmt.Printf(“%s’’, v2)*

*}*

*fmt.Printf(“\n”)*

*}*

*}*

*func main () {*

*a := [3][2]string{*

*{‘’jaune ‘’, ‘’rouge’’},*

*{‘’vert, ‘’mauve’’},*

*{‘’noir’’, ‘’blanc’’},* //virgule importante pour que le compilateur ne lance pas de message d’erreur

*}*

*printarray(a)*

*var b [3][2]string*

*b[0][0] = ‘’prada’’*

*b[0][1] = ‘’lacoste’’*

*b[1][0] = ‘’louboutin’’*

*b[1][1] = ‘’gucci’’*

*b[2][0] = ‘’dolce & gabanna’’*

*b[2][1] = ‘’louis vuiton’’*

*fmt.Printf(“\n”)*

*printarray(b)*

*}*

* l’exécution du programme donne

# jaune rouge

# vert mauve

# noir blanc

# prada lacoste

# louboutin gucci

# dolce & gabanna Louis vuiton

II Tableaux dynamiques : utiliser les slices

1. Déclarer une slice
2. Ajouter des éléments dans une slice
3. Supprimer des éléments dans une slice
4. Copier le contenu d’une slice