

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

main frame

BONUS_PERCENTAGE	10
------------------	----

Heap

P1 mcache

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

Employee:new

name	"John"
salary	5000
sales	5
bonus	0

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache



Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name    string
9     salary  int
10    sales   int
11    bonus   int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21
22 func findEmployeeBonus(salary, noOfSales int) int {
23     bonusPercentage := getBonusPercentage(salary)
24     bonus := bonusPercentage * noOfSales
25     return bonus
26 }
27
28 func main() {
29     var john = Employee{"John", 5000, 5, 0}
30     john.bonus = findEmployeeBonus(john.salary, john.sales)
31     fmt.Println(john.bonus)
32 }
```

Goroutine stack

findEmployeeBonus

salary	5000
noOfSales	5

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache



Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

findEmployeeBonus

salary	5000
noOfSales	5
bonusPercentage	0

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

getBonusPercentage

salary	5000
--------	------

findEmployeeBonus

salary	5000
noOfSales	5
bonusPercentage	0

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

getBonusPercentage

salary	5000
percentage	500

findEmployeeBonus

salary	5000
noOfSales	5
bonusPercentage	0

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

getBonusPercentage

salary	5000
percentage	500
return	500

findEmployeeBonus

salary	5000
noOfSales	5
bonusPercentage	0

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

findEmployeeBonus

salary	5000
noOfSales	5
bonusPercentage	500

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

findEmployeeBonus

salary	5000
noOfSales	5
bonusPercentage	500
bonus	2500

main frame

john	●
BONUS_PERCENTAGE	10

Heap

P1 mcache

Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack

findEmployeeBonus

salary	5000
noOfSales	5
bonusPercentage	500
bonus	2500
return	2500

main frame

john	●
BONUS_PERCENTAGE	10

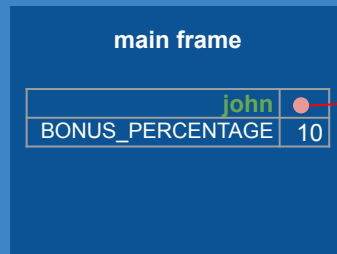
Heap

P1 mcache

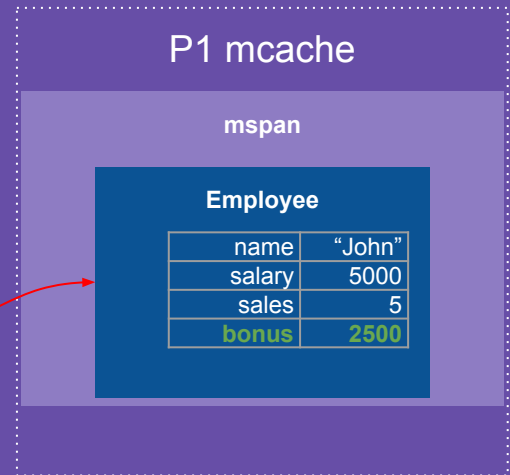
Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack



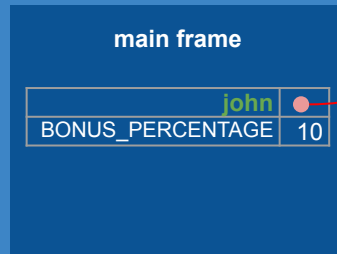
Heap



Go Memory usage

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 type Employee struct {
8     name string
9     salary int
10    sales int
11    bonus int
12 }
13
14 const BONUS_PERCENTAGE = 10
15
16 func getBonusPercentage(salary int) int {
17     percentage := (salary * BONUS_PERCENTAGE) / 100
18     return percentage
19 }
20
21 func findEmployeeBonus(salary, noOfSales int) int {
22     bonusPercentage := getBonusPercentage(salary)
23     bonus := bonusPercentage * noOfSales
24     return bonus
25 }
26
27 func main() {
28     var john = Employee{"John", 5000, 5, 0}
29     john.bonus = findEmployeeBonus(john.salary, john.sales)
30     fmt.Println(john.bonus)
31 }
32
```

Goroutine stack



Heap

