

Receiver Functions

01

About

02

Pointer Receivers

03

Value Receivers

Receiver Functions

- | Modified function signature which allows dot notation
- | Makes writing some types of functionality more convenient
- | Allows simple mutation of existing structures
 - | Similar to modifying a class variable in other languages

Regular Function

```
type Coordinate struct {  
    X, Y int  
}
```

```
func shiftBy(x, y int, coord *Coordinate) {  
    coord.X += x  
    coord.Y += y  
}
```

```
coord := Coordinate{5, 5}  
shiftBy(1, 1, &coord) // (6, 6)
```


Receiver Function (Pointer)

```
type Coordinate struct {  
    X, Y int  
}  
  
func (coord *Coordinate) shiftBy(x, y int) {  
    coord.X += x  
    coord.Y += y  
}  
  
coord := Coordinate{5, 5}  
coord.shiftBy(1, 1)    // (6, 6)
```

Example Continued

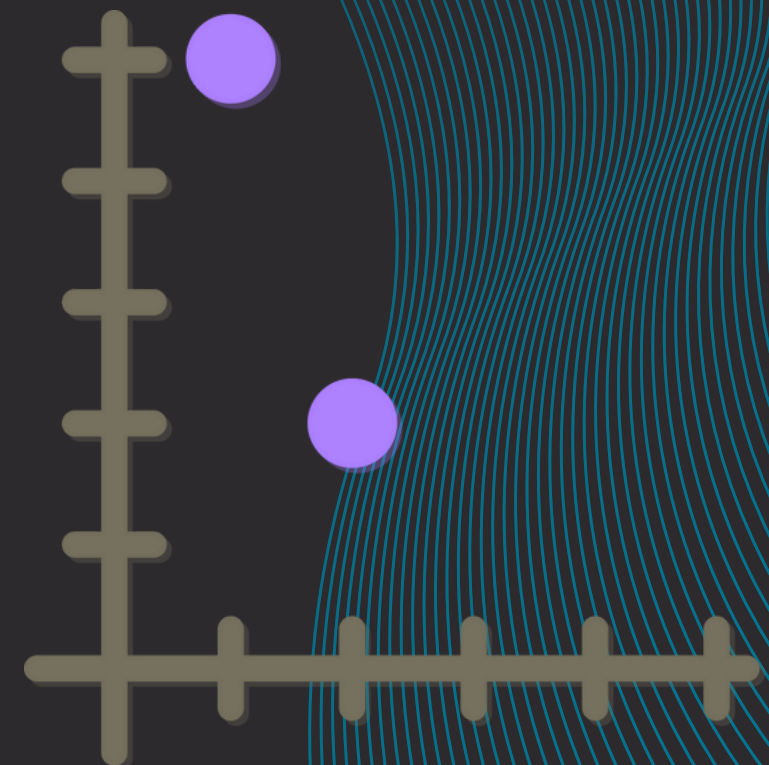
```
type Coordinate struct {  
    X, Y int  
}  
  
func shiftBy(x, y int, coord *Coordinate) {  
    coord.X += x  
    coord.Y += y  
}  
  
func (coord *Coordinate) shiftBy(x, y int) {  
    coord.X += x  
    coord.Y += y  
}  
  
coord := Coordinate{5, 5}  
shiftBy(1, 1, &coord) // (6, 6)  
coord.shiftBy(1, 1)    // (7, 7)
```


Receiver Function (Value)

```
type Coordinate struct {  
    X, Y int  
}
```

```
func (c Coordinate) ShiftDist(other Coordinate) Coordinate {  
    return Coordinate{other.X - c.X, other.Y - c.Y}  
}
```

```
first := Coordinate{2, 2}  
second := Coordinate{1, 5}  
distance := first.ShiftDist(second) // {-1 3}
```



Recap

- | Receiver functions provide the "dot" notation for structs
 - | Create more convenient APIs
- | **Pointer** receivers can modify a struct
- | **Value** receivers cannot modify a struct
- | Common to use pointer receivers

