Go fmt Paketi Cheatsheet

fmt paketi, Go programlar■nda formatl■ giri■/ç■k■■ (I/O) i■lemleri için kullan■l■r. C dilindeki printf ailesine benzer ■ekilde çal■■■r. Yazd■rma, string olu■turma, input alma ve hata üretme i■levlerini içerir.

■ Yazd■rma Fonksiyonlar■

- **Print**: Argümanlar■ ekrana basar (bo■luk yok).
- **Println**: Argümanlar■ ekrana basar, aralara bo■luk koyar, \n ekler.
- **Printf**: Format string ile yazd■r■r.
- **Sprint / Sprintf / Sprintln**: String döndürür, ekrana yazmaz.
- **Fprint / Fprintf / Fprintln**: Bir io.Writer (ör. dosya) içine yazar.

■ Okuma Fonksiyonlar■

- **Scan**: Bo■lukla ayr■lm■■ input.
- **Scanf**: Formatl■ input.
- **Scanln**: Sat■r sonuna kadar okur.
- **Fscan / Fscanf / Fscanln**: io.Reader üzerinden okur (dosya vb.).
- **Sscan / Sscanf / Sscanln**: String içinden okur.

■ Format Verb'leri

- %v → Varsay■lan format
- %+v → Struct alan isimleriyle
- %#v → Go syntax'■na uygun
- %T → Tip
- %s → String
- %q → Çift t■rnakl■ string
- %d → Ondal■k int
- %b → Binary
- %x → Hex
- %f → Ondal■k float
- %e → Bilimsel gösterim
- %t → true/false

■ Kod Örnekleri

Printf Kullan■m■:

```
name := "Ali"
age := 30
fmt.Printf("Ad: %s, Ya■: %d\n", name, age)
```

Scan Kullan■m■:

```
var name string
var age int
fmt.Scan(&name;, &age;) // Input: "Ali 30"
```

Struct Yazd■rma:

```
type User struct{ Name string; Age int }
u := User{"Ali", 30}
fmt.Printf("%+v\n", u) // {Name:Ali Age:30}
```

■ Hata Yönetimi

```
err := fmt.Errorf("kullan■c■ bulunamad■: %s", "Ali")
fmt.Println(err)
```