Date parsing in Go

Date parsing in Go may seem unintuitive at first, especially for those coming from other languages that use formats like `YYYY-MM-DD`. In Go, the `time.Parse` function uses a specific reference format: **Mon Jan 2 15:04:05 MST 2006**. This layout serves as the example from which Go infers the format of the string to interpret.

The Reference Format

Instead of using symbols like `yyyy`, `MM` or `dd`, Go relies on fixed values and their positions. Here's the full reference value:

```
const layout = "Mon Jan 2 15:04:05 MST 2006"
```

Each part represents a specific component of the date:

- **Mon**: day of the week (e.g. "Mon")
- **Jan**: month (e.g. "Jan")
- **2**: day (e.g. "2")
- **15:04:05**: time in 24h format
- **MST**: timezone
- **2006**: year

Parsing Example

Let's look at an example of parsing a string in the `YYYY-MM-DD` format:

```
package main

import (
        "fmt"
```

```
        "time"
)

func main() {
        layout := "2006-01-02"
        input := "2025-07-23"
        t, err := time.Parse(layout, input)
        if err != nil {
                fmt.Println("Error parsing:", err)
                return
        }
        fmt.Println("Parsed date:", t)
}
```

Parsing with Time

To include the time as well, you can use a more extended layout:

```
layout := "2006-01-02 15:04:05"
input := "2025-07-23 14:30:00"
t, err := time.Parse(layout, input)
```

Timezones

To interpret dates with a timezone, it’s necessary to include it in the layout:

```
layout := "2006-01-02 15:04:05 -0700"
input := "2025-07-23 14:30:00 +0200"
t, err := time.Parse(layout, input)
```

Parsing with time.ParseInLocation

If you know the desired timezone, you can use `time.ParseInLocation`:

```
loc, _ := time.LoadLocation("Europe/Rome")
t, err := time.ParseInLocation("2006-01-02 15:04:05",
"2025-07-23 14:30:00", loc)
```

Conclusions

Date parsing in Go requires familiarity with the reference layout. Once the basic principle is understood, you can correctly parse almost any format. The `time` package documentation is an essential resource for further insights.