MAVENIR™
Trust the Future

Golang Training Pointers

24/01/2023

Golang: Strings

Overview

- It is a sequence of variable-width characters
- Every character is represented by one or more bytes using UTF-8 Encoding
- Strings are the immutable chain of arbitrary bytes, including bytes with zero values
- String is a read-only slice of bytes
- Bytes of the strings can be represented in the Unicode text using UTF-8 encoding

Golang: Strings

Overview

- **Using double quotes(“”)**
  - String literals are created using double-quotes(“”)
  - This type of string support escape character
- **Using backticks(“)**
  - String literals are created using backticks(“) and known as raw literals
  - Raw literals do not support escape characters
  - Can span multiple lines
  - Can contain any character except backtick
  - Used for writing multiple line message, in the regular expressions, and in HTML.

Golang: Strings

Overview

- **Strings are immutable:**
  - Once a string is created, It's value of the string cannot be changed
  - Strings are read-only
- String can be iterated using for range
  - range can iterate over the Unicode code point for a string

Syntax:

```
for index, chr:= range str{
    // Statement..
}
```

Golang: Strings

String length

- Find the length of the string
  - len() function is used find length of the string
    - Returns the number of bytes of the string
  - RuneCountInString() function is provided by UTF-8 package
    - Returns the total number of rune presents in the string

Golang: Strings

String Comparison

- **Using comparison operators**
  - Go strings support comparison operators ==, *!=*, >=, <=, <, >
  - **==** and **!=** operator are used to check if the given strings are equal or not
  - >=, <=, <, > operators are used to find the lexical order
- **Using Compare() method**
  - Compare() provided by the strings package
  - This function returns an integer value after comparing two strings lexicographically
- Return 0, if *str1* == *str2,* Return 1, if *str1* > *str2,* Return -1, if *str1* < *str2*.

Golang: Strings

String Functions - Trim

- Trim(str string, cutstr string) string.
- TrimLeft(str string, cutstr string) string
- TrimRight(str string, cutstr string) string
- TrimSpace(str string) string
- TrimSuffix(str, suffstr string) string
- TrimPrefix(str, suffstr string) string

Golang: Strings

String Functions - Trim

- Split(str, sep string) []string
- SplitAfter(str, sep string) []string
- SplitAfterN(str, sep string, m int) []string

Golang: Strings

String Functions – Repeat/Count

- Repeat(str string, count int) string
  - This method will panics if the value of the count is negative
- Count(str, costr string) int
  - Counts number of occurances of costr in str

Golang: Strings

Explore... strings

- String concatination
  - Using Sprintf
  - Using + operator
  - Using strings package methods

MAVENIR