**Strings and strconv Package**

## Prefixes and suffixes

strings.HasPrefix(s, prefix string) bool

strings.HasSuffix(s, suffix string) bool

## Testing whether a string contains a substring

strings.Contains(s, substr string) bool

## Indicating the index a substring or character in a string

strings.Index(s, str string) int

strings.LastIndex(s, str string) int

strings.IndexRune(s string, ch int) int.

## Replacing substring [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#replacing-substring)

strings.Replace(str, old, new string, n int)

## Counting occurrences of a substring [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#counting-occurrences-of-a-substring)

strings.Count(s, str string) int

## Repeating a string [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#repeating-a-string)

strings.Repeat(s, count int) string

## Changing the case of a string

strings.ToLower(s) string

strings.ToUpper(s) string

## Trimming a string [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#trimming-a-string)

strings.TrimSpace(s)

If you want to trim a specific string str from a string s, use:

strings.Trim(s, str)

For example:

strings.Trim(s, "\r\n")

## Splitting a string [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#splitting-a-string)

### On whitespaces [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#on-whitespaces)

The strings.Fields(s) splits the string s around each instance of one or more consecutive white space characters, and returns a slice of substrings []string of s or an empty list, if s contains only white space.

### On a separator [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#on-a-separator)

The strings.Split(s, sep) works the same as Fields, but splits around sep. The sep can be a separator character (:,;,,,-,…) or any separator string sep.

## Joining over a slice [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#joining-over-a-slice)

The strings.Join(sl []string, sep string) results in a string containing all the elements of the slice sl, separated by sep:

## Reading from a string [**#**](https://www.educative.io/courses/the-way-to-go/xl6WmMg9Z8J#reading-from-a-string)

The strings package also has a function called strings.NewReader(str). This produces a pointer to a Reader value, that provides amongst others the following functions to operate on str:

* Read() to read a []byte
* ReadByte() to read the next byte from the string.
* ReadRune() to read the next rune from the string.

## Conversion to and from a string

strconv.Itoa(i int) string

strconv.Atoi(s string) (i int, err error)

## func Map

func Map(mapping func([rune](https://golang.org/pkg/builtin/#rune)) [rune](https://golang.org/pkg/builtin/#rune), s [string](https://golang.org/pkg/builtin/#string)) [string](https://golang.org/pkg/builtin/#string)

Map returns a copy of the string s with all its characters modified according to the mapping function. If mapping returns a negative value, the character is dropped from the string with no replacement.