Golang заметки - Работа с HTML шаблонами

Created By	Ⓓ Daniil A (Kiky Tokamuro)
Last Edited	@Jul 8, 2020
Tags	golang

Пакеты `html` и `html/template` из стандартной библиотеки обеспечивают основные возможности работы с HTML разметкой, включая применение переменных и функций в шаблонах. Пакет `html/template` опирается на пакет `text/template`, предназначенный для обработки текстовых шаблонов.

Использование простого HTML шаблона

```
package main

import (
  "html/template"
  "net/http"
)

var tmpl = `
<h1>
  <a href="{{.Link}}">{{.Text}}</a>
</h1>`

type page struct {
  Text string
  Link string
}

func indexHandle(w http.ResponseWriter, r *http.Request) {
  t, _ := template.New("page").Parse(tmpl)

  p := page{
    Text: "github",
    Link: "https://github.com/",
  }

  t.Execute(w, p)
}

func main() {
  http.HandleFunc("/", indexHandle)
```

```
  http.ListenAndServe(":3000", nil)
}
```

Добавление функций для шаблонов

Механизм конвейера передает вывод одного элемента конвейера следующему в последнем аргументе.

```
package main

import (
  "html/template"
  "net/http"
  "strings"
)

var tmpl = `
<h1>
  <a href="{{.Link}}">{{.Text | ToUpper}}</a>
</h1>`

type page struct {
  Text string
  Link string
}

func indexHandle(w http.ResponseWriter, r *http.Request) {
  // Добавление функции ToUpper из стандартной библиотеки strings
  funcMap := template.FuncMap{
    "ToUpper": strings.ToUpper,
  }

  t, _ := template.New("page").Funcs(funcMap).Parse(tmpl)

  p := page{
    Text: "github",
    Link: "https://github.com/",
  }

  t.Execute(w, p)
}

func main() {
  http.HandleFunc("/", indexHandle)
  http.ListenAndServe(":3000", nil)
}
```

Использование вложенных шаблонов

Шаблон заголовка, включаемый в главный шаблон:

```
package main

import (
  "html/template"
  "net/http"
)

var header = `
{{define "header"}}
  <h1>Header</h1>
  <hr>
{{end}}`

var tmpl = `
{{template "header" .}}
<h1>
  <a href="{{.Link}}">{{.Text}}</a>
</h1>`

type page struct {
  Text string
  Link string
}

func indexHandle(w http.ResponseWriter, r *http.Request) {
  t, _ := template.New("page").Parse(header)
  t.Parse(tmpl)

  p := page{
    Text: "github",
    Link: "https://github.com/",
  }

  t.Execute(w, p)
}

func main() {
  http.HandleFunc("/", indexHandle)
  http.ListenAndServe(":3000", nil)
}
```

Когда шаблоны храняться в файлах, их можно считывать такой конструкцией: `template.Must(template.ParseFiles("index.html", "header.html"))`

```
package main

import (
```

```
  "html/template"
  "net/http"
)

type page struct {
  Text string
  Link string
}

func indexHandle(w http.ResponseWriter, r *http.Request) {
  t := template.Must(template.ParseFiles("index.html", "header.html"))

  p := page{
    Text: "github",
    Link: "https://github.com/",
  }

  t.ExecuteTemplate(w, "index.html", p)
}

func main() {
  http.HandleFunc("/", indexHandle)
  http.ListenAndServe(":3000", nil)
}
```

Для обработки шаблона используется метод `ExecuteTemplate`, чтобы можно было указать имя основного шаблона. Если вызвать метод `Execute`, как в предыдущих примерах, он обработал бы первый шаблон из перечисленных в вызове функции `ParseFiles`.