

**Autorzy:**  
Dominik Kołodziej  
Marek Matys

# Translator języka Markdown na język HTML


W poniższej dokumentacji wytłuszczoną czcionką są oznaczone nazwy **pakietów**, kursywą są zapisane nazwy *plików*, a z nawiasami napisane są nazwy funkcji().

W projekcie wykorzystany został pakiet **go-sciter** dostarczający większość funkcjonalności aplikacyjnego interfejsu programistycznego (API) Scitera – czyli osadzalnego HTML/CSS/skryptowego silnika do tworzenia nowoczesnych interfejsów użytkownika – do środowiska Golang. Pakiet **go-sciter** dostarcza między innymi takie funkcjonalności jak:

- Wczytywanie plików HTML / kodu HTML jako ciągu znaków (tekstu)
- Manipulacje drzewem DOM / wykonywanie funkcji zwrotnych/obsługę zdarzeń
- Obsługę stanu drzewa DOM/ obsługę atrybutów
- Wczytywanie zasobów niestandardowych
- Sciter Behavior
- Sciter Options
- Sciter Value support
- NativeFuncor (używany w skryptach Scitera)

Projekt składa się z 3 głównych katalogów: **frontend**, **main** i results. W pakiecie **frontend** napisany jest front-end aplikacji, w pakiecie **main** właściwy translator, a do folderu results zapisywane są pliki wynikowe (plik HTML przetłumaczony z języka Markdown). Pakiet **main** składa się z 3 elementów: pakietów: **translator** i **utils** oraz pliku *main.go*.

Uruchomienie aplikacji jest równoznaczne z uruchomieniem pliku *main.go*. Jest to, można powiedzieć, swego rodzaju kontener na wszystko. Plik *main.go* składa się z:

- funkcji `init()` – utworzenie okna programu, czytanie z pliku `translatorView.html` (katalog frontend), zaczytanie wszystkich przycisków
- funkcji `main()` – odczytanie pliku za pomocą przycisku „read the file”, przetłumaczenie go z wykorzystaniem funkcji `translate()` z pakietu `translator` (patrz niżej ) , zapisanie wyjściowego pliku i ustawienie statusu „Done”
- funkcji `closeApplication()` – zamyka aplikację
- funkcji `setStatus()` – ustawia opis statusu tłumaczenia widoczny w oknie aplikacji
- funkcji `shouldOpenExplorer()` – jeżeli jest zaznaczony checkbox „Open directory” zwraca flagę `true`, w przeciwnym przypadku – `false`
- funkcji `shouldOpenBrowser()` – jeżeli jest zaznaczony checkbox „Open file when finished” zwraca flagę `true`, w przeciwnym przypadku – `false`

Pakiet **`translator`** składa się z pakietu **`linesProcessing`**, oraz plików: *translator.go* i *translator\_test.go*. Plik *translator.go* składa się z:

- funkcji `translate()` – przyjmująca 2 argumenty - `content` i `fileName` - dzieli zawartość pliku `.md` przekazaną w pierwszym argumencie na linijki za pomocą funkcji `stringToLines()` (bo takowe się łatwiej przetwarza), następnie wykorzystując pakiet `linesProcessing` przetwarza przekazane dane i zwraca zawartość pliku `.html`
- funkcji `stringToLines()` – dzieli tekst (łańcuch znaków) na linijki
- funkcji `linesToString()` – odwrotność funkcji `stringToLines()`
- funkcji `wrapInHtml()` – wrapuje wszystko co jest w znaczniki `html`, czyli dorzuca do przetłumaczonej zawartości pliku obowiązkową otoczkę dla plików `.html`

Pakiet **`linesProcessing`** odpowiada za wykrywanie poszczególnych elementów języka Markdown i przetłumaczenie tych elementów na język HTML. W skład tego pakietu wchodzi pliki:

- *emphasis.go* – pogrubienia, pochylenia i przekreślenia
- *headers.go* – nagłówki
- *horizontal.go* – poziome linie
- *images.go* – obrazy, ich tytuły i teksty alternatywne
- *inlineCode.go* – teksty zdefiniowane jako kody komputerowe

- *linesProcessing\_test.go* – testy
- *links.go* – odsyłacze (=hiperłącza)
- *lists.go* – listy uporządkowane i listy nieuporządkowane
- *paragraphs.go* – akapity
- *references.go* – referencje (zapis referencji do mapy)

Poniżej zrzuty ekranu przedstawiające działanie naszego programu: źródłowy plik MD, wyjściowy plik HTML i wyrenderowana strona:

```

1  # H1
2  ## H2
3  ### H3
4  #### H4
5  ##### H5
6  ##### H6
7
8
9  Emphasis, aka italics, with asterisks or underscores.
10
11 Strong emphasis, aka bold, with asterisks or underscores.
12
13 Combined emphasis with asterisks and underscores.
14
15 Strikethrough uses two tildes. Scratch this.
16
17 1. qwert
18   1. 234
19   2. 2343253252355235
20   ~2. 3234
21   5. 23234234
22
23   1. First ordered list item
24   2. Another item
25   ~* Unordered sub-list.
26   1. Actual numbers don't matter, just that it's a number
27   ~1. Ordered sub-list
28   4. And another item.
29

```

```

1  <html lang="en"><head><meta charset="utf-8"><title>test.html</title></head><body><h1> H1</h1>
2  <h2> H2</h2>
3  <h3> H3</h3>
4  <h4> H4</h4>
5  <h5> H5</h5>
6  <h6> H6</h6>
7
8
9  <p>
10 Emphasis, aka italics, with <em>asterisks</em> or <em>underscores</em>.
11 </p>
12
13 <p>
14 Strong emphasis, aka bold, with <strong>asterisks</strong> or <strong>underscores</strong>.
15 </p>
16
17 <p>
18 Combined emphasis with <strong>asterisks and <em>underscores</em></strong>.
19 </p>
20
21 <p>
22 Strikethrough uses two tildes. <del>Scratch this.</del>
23 </p>
24
25 <p>
26 <ol>
27 <li>qwert</li>
28 <li>234</li>
29 <li>2343253252355235</li>
30 </ol>
31
32 <ol>
33 <li>2. 3234</li>
34 <li>23234234</li>
35 </ol>
36
37 <ol>
38 <li>1. First ordered list item</li>
39 <li>2. Another item</li>
40 <li>~* Unordered sub-list.</li>
41 <li>1. Actual numbers don't matter, just that it's a number</li>
42 <li>~1. Ordered sub-list</li>
43 <li>4. And another item.</li>
44 </ol>
45

```

