BEN FORTA

WYRAŻENIA REGULARNE OD PODSTAW

Tytuł oryginału: Learning Regular Expressions

Tłumaczenie: Marta Danch-Wierzchowska

ISBN: 978-83-283-6058-7

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/wyrepo
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

Spis treści

1

Wstęp do wyrażeń regularnych

W tej lekcji omówiono, czym są wyrażenia regularne i w jaki sposób mogą być użyteczne.

Zrozumieć potrzebę

Wyrażenia regularne (w skrócie nazywane **RegEx** lub **regexami**, od ang. *Regular Expressions*) są narzędziem i jak wszystkie narzędzia są zaprojektowane, by rozwiązywać konkretne problemy. Najlepszą metodą pozwalającą zrozumieć wyrażenia regularne i zasadę ich działania jest zrozumienie rozwiązywanego przezeń problemu.

Oto kilka problemów:

- Należy znaleźć plik zawierający tekst `oko` (niezależnie od wielkości znaków), ale bez słów zawierających `oko` w środku słowa (np. `okoliczności`, `wokoło`, `wysoko`).
- Podczas tworzenia strony internetowej trzeba wyświetlić tekst z bazy danych. Tekst może zawierać linki, które mają być hiperłączami na stworzonej stronie (tak, aby zamiast generować po prostu tekst, stworzyć funkcjonalny HTML `<a href> </a>`).
- Podczas tworzenia aplikacji z formularzem dla użytkownika zawierającym adres e-mail należy zweryfikować, czy adres jest prawidłowo sformatowany (czy składnia jest poprawna).

- Podczas edycji kodu źródłowego należy zamienić wszystkie wystąpienia `size` na `iSize`, ale jedynie słowa `size`, nie `size` jako część innego słowa.
- Należy wyświetlić listę wszystkich plików w komputerze, by znaleźć jedynie te, które zawierają tekst `Aplikacja`.
- Do aplikacji są importowane dane, które są oddzielone tabulatorem, a aplikacja wspiera format plików CSV (jeden wiersz w linii, wartości oddzielone przecinkami, każdy wpis ujęty w cudzysłów).
- Należy znaleźć specyficzny tekst w pliku, ale jedynie w konkretnym miejscu (np. na początku linii albo na końcu zdania).

Wszystkie powyższe punkty pokazują przykładowe wyzwania programistyczne i wszystkie mogą być rozwiązane w dowolnym języku, który wspiera przetwarzanie warunkowe i operacje na ciągach znaków. Jak bardzo skomplikowane byłoby jednak rozwiązanie? Należałoby przejrzeć kolejno wszystkie słowa albo znaki, przetwarzając jednocześnie wiele zapytań warunkowych, śledzić wiele flag tak, by wiedzieć, co już zostało znalezione, a co nie, sprawdzać białe znaki i znaki specjalne itd. A wszystko to trzeba by zrobić za każdym razem ręcznie.

Albo… można wykorzystać wyrażenia regularne. Każdy z przedstawionych powyżej problemów może być rozwiązany za pomocą stworzonego już wyrażenia — zwięzłego ciągu znaków zawierającego tekst i instrukcje specjalne — które mogłoby wyglądać na przykład tak:

```
\b[Oo][Kk][Oo]\b
```

> **Uwaga**
> Powyższa linia już wkrótce stanie się zrozumiała.

Jak wykorzystywane są wyrażenia regularne?

Gdyby przyjrzeć się wypisanym w poprzedniej sekcji problemom, można zauważyć, że każdy z nich można zaklasyfikować jako jeden z dwóch typów: albo informacja jest szukana (**znajdź**, ang. *search*), albo informacja jest szukana i edytowana (**zamień**, ang. *replace*). W rzeczywistości do tego właśnie wyrażenia regularne są używane: znajdź i zamień. Każde wyrażenie regularne albo dopasowuje tekst (znajdź), albo dopasowuje i zamienia tekst (zamień).

RegEx „znajdź”

Wyrażenia regularne są wykorzystywane w wyszukiwaniach (znajdź), gdy tekst, którego szukają, jest wysoce dynamiczny, jak w przypadku opisanego wcześniej poszukiwania tekstu `oko`. Na początek należy zlokalizować `oko`, `OKO`, `Oko` albo `OkO` — to ta łatwa część (wiele narzędzi do wyszukiwania potrafi znaleźć ciągi znaków niezależnie od wielkości). Trudniejsza część to natomiast upewnienie się, że `okoliczności`, `wokoło` i `wysoko` nie pasują. Niektóre, bardziej wyszukane edytory tekstu mają opcję *Dopasuj tylko całe słowa* (ang. *Match Only the Whole Word*), ale wiele edytorów jej nie ma i może nie być możliwe wprowadzenie takiej zmiany w edytowanym dokumencie. Wykorzystując wyrażenia regularne do przeszukiwania, zamiast szukać jedynie tekstu `oko` rozwiązuje się problem.

> **Wskazówka**
>
> Jak wygląda rozwiązanie powyższego problemu? Już się pojawiło w książce — to przykładowe wyrażenie przedstawione wcześniej: `\b[Oo][Kk][Oo]\b`.

Warto zauważyć, że sprawdzanie zgodności (np. *czy adres e-mail podany przez użytkownika spełnia to wyrażenie regularne*) to również operacją „znajdź”. Sprawdzany jest cały ciąg znaków wprowadzony przez użytkownika (podobnie jak poszukiwania podciągów, co jest typowym zastosowaniem operacji „znajdź”).

RegEx „zamień”

Operacje typu „znajdź” wyrażeń regularnych są nieprawdopodobnie skuteczne, bardzo praktyczne i nie aż tak trudne do nauczenia. Wiele przykładów i rozdziałów w tej książce dotyczy właśnie ich. Jednak prawdziwa siła regexów tkwi w operacjach typu „zamień”. Poniżej został omówiony znany już przykład z zamianą linków URL na hiperłącza. Na początek należy znaleźć wszystkie linki w tekście (prawdopodobnie szukając ciągów zaczynających się od `http://` lub `https://`, a kończących się przecinkiem, kropką lub znakiem białym). Następnie trzeba zamienić znaleziony link na dwa powtórzenia znalezionego linku wraz z dołączonymi komendami HTML-a. W ten sposób link:

```
http://www.forta.com/
```

jest zamieniany na:

```
<a href="http://www.forta.com">http://www.forta.com/</a>
```

Może się też zdarzyć, że znaleziony link jest tylko adresem, a nie pełnym ciągiem URL:

```
www.forta.com
```

co również należy zamienić na:

```
<a href="http://www.forta.com">http://www.forta.com/</a>
```

Opcje „znajdź” i „zamień” w większości aplikacji nie obsługują tego typu operacji, ale przy wykorzystaniu wyrażeń regularnych zadanie to staje się nieprawdopodobnie proste.

Czym tak naprawdę są wyrażenia regularne?

Teraz, gdy już wiadomo, do czego używa się wyrażeń regularnych, można wprowadzić definicje. Mówiąc wprost, **wyrażenia regularne** to ciągi znaków, które są wykorzystywane do dopasowania i operowania na tekście. Wyrażenia regularne są tworzone za pomocą języka wyrażeń regularnych, który został opracowany specjalnie po to, by możliwe było wszystko to, o czym zostało tu napisane, i wiele, wiele więcej. Jak każdy inny język, wyrażenia regularne mają specyficzną składnię i instrukcje, których należy się nauczyć, a po to jest ta książka.

Język wyrażeń regularnych nie jest w pełni językiem programowania. Zazwyczaj nie jest to nawet program czy pakiet, który można zainstalować i stosować. Najczęściej wyrażenia regularne są minijęzykami wbudowanymi w inne języki lub produkty. Dobra wiadomość jest taka, że obecnie niemal każdy porządny język lub narzędzie wspiera wyrażenia regularne. Zła wiadomość: język wyrażeń regularnych sam w sobie w niczym nie przypomina żadnego innego języka ani narzędzia, w który jest wbudowany. Język wyrażeń regularnych jest językiem samym w sobie — językiem zdecydowanie nie najbardziej intuicyjnym i oczywistym, na jaki można trafić.

> **Uwaga**
>
> Wyrażenia regularne zostały stworzone na potrzeby matematyki w latach 50. ubiegłego wieku. Wiele lat później główne zasady i koncepcje zostały przeniesione i zaadaptowane do świata unixowego Perla w poleceniach takich jak `grep`. Przez długi czas wyrażenia regularne (wykorzystywane w przypadkach podobnych jak te opisane wcześniej) były dostępne wyłącznie dla wąskiego grona społeczności unixowej. Zmieniło się to jednak, a wyrażenia regularne są obecnie wspierane w różnej formie na niemal każdej platformie obliczeniowej.

Oto funkcjonalne wyrażenia regularne (wkrótce nabiorą one sensu):

- `Ben`
- `.`
- `www\.forta\.com`
- `[a-zA-ZO-9_.]*`
- `<[Hh]1>.*</[Hh]1>`
- `\r\n\r\n`
- `\d{3,3}-\d{3,3}-\d{4,4}`

W tym miejscu należy zauważyć, że składnia to najprostsza część nauki wyrażeń regularnych. Największym wyzwaniem jest opanowanie, w jaki sposób zastosować tę składnię, by przełożyć napotkany problem na rozwiązanie regexowe. Nie wystarczy do tego przeczytać książkę, podobnie bowiem jak w przypadku każdego innego języka, biegłość przychodzi wraz z praktyką.

Użycie wyrażeń regularnych

Jak już wspomniano, nie ma programu do wyrażeń regularnych, nie ma aplikacji ani oprogramowania, które można kupić lub pobrać. Język wyrażeń regularnych jest zaimplementowany w wiele programów, języków, komend i środowisk deweloperskich.

W jaki sposób używa się wyrażeń regularnych i jak ich funkcjonalność różni się w poszczególnych aplikacjach? Niektóre aplikacje mają opcje w menu i okna dialogowe, za pomocą których można używać wyrażeń regularnych, podczas gdy języki programowania najczęściej dostarczają funkcji, klas lub obiektów, które zawierają funkcjonalności regexów.

Co więcej, nie wszystkie implementacje wyrażeń regularnych są jednakowe. Często pojawiają się subtelne (a czasem mniej subtelne) różnice w składni i cechach.

Dodatek A, „ Wyrażenia regularne w popularnych narzędziach i językach”, zawiera przydatne szczegóły i uwagi dotyczące wsparcia wyrażeń regularnych w wielu aplikacjach i językach. Przed przejściem do kolejnej lekcji warto zajrzeć do tego dodatku, by nauczyć się specyfiki odnoszącej się do aplikacji lub języka, w którym chce się wykorzystywać wyrażenia regularne.

Dla łatwiejszego startu można przejrzeć linki do narzędzi internetowych stworzonych do testowania wyrażeń regularnych zamieszczone na stronie poświęconej oryginalnej wersji tej książki:

http://forta.com/books/0134757068/

Zamieszczone tam przykładowe narzędzia są najczęściej najprostszą metodą eksperymentowania z wyrażeniami regularnymi[1].

Przed rozpoczęciem

Przed rozpoczęciem pracy należy zwrócić uwagę na kilka ważnych kwestii:

- Podczas stosowania wyrażeń regularnych można zauważyć, że niemal zawsze istnieje kilka możliwych rozwiązań danego problemu. Niektóre są prostsze, inne szybsze, jedne mogą być bardziej kompaktowe, drugie bardziej uniwersalne. Rzadko kiedy rozwiązanie jest po prostu dobre albo złe (oczywiście pod warunkiem, że działa jak należy).
- Jak już zostało powiedziane, istnieją różnice pomiędzy implementacjami regexów. Przykłady i lekcje przedstawione w tej książce zostały dostosowane możliwie najogólniej do najczęściej wykorzystywanych implementacji, a ewentualne różnice zostały również zaznaczone.
- Podobnie jak w każdym języku, tak i w tym przypadku kluczem do sukcesu jest praktyka, praktyka i jeszcze raz praktyka.

> **Uwaga**
> Zdecydowanie zalecane jest, aby w trakcie lektury książki testować każdy zawarty w niej przykład.

Podsumowanie

Wyrażenia regularne są jednym z najpotężniejszych narzędzi umożliwiających operowanie na tekście. Język wyrażeń regularnych jest wykorzystywany do tworzenia wyrażeń regularnych (stworzony ciąg znaków jest nazywany **wyrażeniem regularnym**). Wyrażenia regularne są używane w operacjach „znajdź” i „zamień”.

[1] Opisane materiały są w języku angielskim — *przyp. red.*


PROGRAM
PARTNERSKI
GRUPY HELION
1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ
Zmień swoją stronę WWW w działający bankomat!
Dowiedz się więcej i dołącz już dzisiaj!
http://program-partnerski.helion.pl
GRUPA Helion

REGEX: ZROZUM I UŻYWAJ DO WOLI!

Wyrażenia regularne (RegEx) służą do dopasowywania ciągów znaków i manipulowania tekstem. Umożliwiają proste rozwiązywanie wielu złożonych problemów programistycznych. Szczególnie często są wykorzystywane przez twórców aplikacji. Nie istnieje jakiś specjalny język do obsługiwania wyrażeń regularnych, jednak znakomita większość języków programowania wspiera ich stosowanie. Wyrażenia regularne uchodzą za niezrozumiałe i trudne w implementacji, a ich składnia bywa określana jako zawiła i nieintuicyjna. Tymczasem największą barierą okazuje się jasne zrozumienie zagadnienia oraz prawidłowe określenie sposobu wykorzystania wyrażeń regularnych w praktyce.

Ta książka jest znakomitym podręcznikiem, dzięki któremu zaczniesz szybko i poprawnie stosować wyrażenia regularne w praktyce. W przystępny sposób wyjaśniono tu, czym są RegEx i jakie problemy mogą rozwiązać, a także jak należy (i jak nie należy) się nimi posługiwać. Przedstawiono wyrażenia regularne, które faktycznie należy znać, od prostych porównań tekstu po bardziej złożone tematy, takie jak stosowanie referencji wstecznych, oceny warunkowej i procesów przewidywania. Poszczególne zagadnienia wyłożono w metodyczny i prosty sposób, bogato ilustrując materiał praktycznymi, gotowymi do wykorzystania przykładami zaprezentowanymi w różnych językach programowania.

Dzięki tej książce nauczysz się:

- rozumieć wyrażenia regularne
- stosować tekst i metaznaki do budowania potężnych wzorców
- przeprowadzać złożone operacje typu znajdź-i-zamień
- dodawać wyrafinowane formuły i ciągi tekstowe do aplikacji WWW
- pracować z numerami telefonów, kodami pocztowymi,
- numerami ubezpieczeń i kart kredytowych, adresami IP i e-mail oraz URL

BEN FORTA — pełni funkcję dyrektora ds. inicjatyw edukacyjnych w Adobe Systems. Jest również autorem ponad 40 popularnych książek dotyczących różnych zagadnień technicznych, w tym kilku bestsellerów. Od ponad 30 lat zajmuje się przemysłem komputerowym, rozwojem, marketingiem i szkoleniami.

Pearson
Addison-Wesley

Helion
helion.pl
HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

ISBN 978-83-283-6058-7
9 788328 360587

INFORMATYKA W NAJLEPSZYM WYDANIU

Cena: 39,90 zł