

Bartłomiej Przybylski

Lua i LaTeX

Dynamiczne tworzenie
dokumentów



Lua i LaTeX

Bartłomiej Przybylski

Lua i LaTeX

**Dynamiczne tworzenie
dokumentów**



[Kup książkę](#)

Projekt okładki: **Hubert Zacharski**

Wydawca: **Łukasz Łopuszański**

Redaktor prowadzący: **Adam Kowalski**

Redakcja techniczna: **Maria Czekaj**

Korekta: **Agnieszka Jaworska**

Koordynator produkcji: **Anna Bączkowska**

Zastrzeżonych nazw firm i produktów użyto w książce wyłącznie w celu identyfikacji.

Książka, którą nabyłeś, jest dziełem twórcy i wydawcy. Prosimy, abyś przestrzegał praw, jakie im przysługują. Jej zawartość możesz udostępnić nieodpłatnie osobom bliskim lub osobiście znanym. Ale nie publikuj jej w internecie. Jeśli cytujesz jej fragmenty, nie zmieniaj ich treści i koniecznie zaznacz, czyje to dzieło. A kopiując jej część, rób to jedynie na użytek osobisty.

Szanujmy cudzą własność i prawo.
Więcej na www.legalnakultura.pl
Polska Izba Książki

Wydanie tej książki zostało częściowo sfinansowane ze środków
Uniwersytetu im. Adama Mickiewicza w Poznaniu.

Copyright © by Wydawnictwo Naukowe PWN SA
Warszawa 2018

ISBN 978-83-01-19696-7

Wydanie I
Warszawa 2018

Wydawnictwo Naukowe PWN SA
02-460 Warszawa, ul. Gottlieba Daimlera 2
tel. 22 69 54 321, faks 22 69 54 288
infolinia 801 33 33 88
e-mail: pwn@pwn.com.pl; reklama@pwn.pl
www.pwn.pl

Skład i łamanie: **Bartłomiej Przybylski**

Druk i oprawa: OSDW Azymut Sp. z o.o.

[Kup książkę](#)

Spis treści

Przedmowa	11
I. Język Lua	
Rozdział 1. Zanim zaczniemy	17
1.1. Środowisko pracy	18
1.2. Programy i biblioteki	21
Rozdział 2. Elementarz	23
2.1. Tryb interaktywny	25
2.2. Anatomia programu	27
2.2.1. Wartości i typy danych	27
2.2.2. Wyrażenia	29
2.2.3. Instrukcje	30
2.2.4. Porcje kodu i programy	31
2.3. Zmienne	31
2.3.1. Wartość domyślna	32
2.3.2. Typ zmiennej	33
2.3.3. Instrukcja przypisania	34
2.4. Komentarze	35
Rozdział 3. Typ podstawowy boolean	41
3.1. Operatory	41
3.1.1. Operatory relacji	42
3.1.2. Operatory logiczne	44
3.1.3. Priorytety operatorów	45
3.2. Instrukcja warunkowa if	46
3.3. Pętle	48
3.3.1. Pętla while	48
3.3.2. Pętla repeat-until	48
3.3.3. Instrukcja break	49
Rozdział 4. Typ podstawowy number	52
4.1. Wartości liczbowe	53
4.2. Operatory	54
4.2.1. Operatory relacji	54
4.2.2. Operatory arytmetyczne	55
4.3. Biblioteka matematyczna	56
4.3.1. Stałe	56
4.3.2. Funkcje trygonometryczne	57
4.3.3. Funkcje liczbowe	58
4.3.4. Generator liczb pseudolosowych	59

4.3.5.	Inne funkcje	60
4.3.6.	Przejście pomiędzy wartościami <i>integer</i> i <i>float</i>	60
4.4.	Numeryczna pętla <i>for</i>	61
Rozdział 5.	Typ podstawowy <i>string</i>	65
5.1.	Stałe wartości znakowe	65
5.1.1.	Anatomia ciągu znaków	67
5.1.2.	Długie ciągi znaków	69
5.2.	Operatory	71
5.2.1.	Operator <i>#</i>	71
5.2.2.	Operatory relacji	72
5.2.3.	Operator konkatencji (łączenia)	72
5.2.4.	Na styku typów	72
5.3.	Biblioteka tekstowa	74
5.3.1.	Podstawowe funkcje	75
5.3.2.	Podciągi	75
5.3.3.	Reprezentacje liczbowe	76
5.3.4.	Przeszukiwanie i podstawienia	77
5.3.5.	Formatowanie ciągu	78
5.4.	Kodowanie UTF-8	81
Rozdział 6.	Typ obiektowy <i>table</i>	87
6.1.	Tabele i klucze	87
6.1.1.	Klucze	88
6.1.2.	Konstruktory tabel	91
6.2.	Tabele sekwencyjne	93
6.2.1.	Sekwencje	93
6.3.	Biblioteka dla sekwencji	95
6.3.1.	Dodawanie, usuwanie i przemieszczanie elementów sekwencji	95
6.3.2.	Sortowanie	97
6.3.3.	Funkcje ogólnego przeznaczenia	98
6.4.	Tabela <i>arg</i>	100
Rozdział 7.	Typ obiektowy <i>function</i>	103
7.1.	Argumenty i wartości	105
7.1.1.	Argumenty funkcji	105
7.1.2.	Wartości zwracane przez funkcje	106
7.1.3.	Funkcje o nieznannej liczbie argumentów	108
7.2.	Zasięg zmiennych	110
7.2.1.	Bloki kodu	111
7.2.2.	Zmienne lokalne	111
7.3.	Funkcje jako wartości pierwszoklasowe	113
7.3.1.	Funkcje anonimowe	114
7.3.2.	Funkcje lokalne	115
7.4.	Iteratory	116
7.4.1.	Domknięcia i zmienne nielocalne	117
7.4.2.	Iteratory	118
7.4.3.	Ogólna pętla <i>for</i>	119
7.4.4.	Funkcje iteracyjne dla tabel	120

Rozdział 8. Obsługa wejścia i wyjścia	124
8.1. Podstawowy model obsługi wejścia i wyjścia	124
8.1.1. Funkcja <code>io.write</code>	125
8.1.2. Funkcja <code>io.read</code>	127
8.1.3. Iterator <code>io.lines</code>	128
8.2. Pełen model obsługi wejścia i wyjścia	129
8.2.1. Funkcja <code>io.open</code>	129
8.2.2. Wywoływanie funkcji na strumieniach danych	131
8.2.3. Iterator <code>io.lines</code> raz jeszcze	132
8.2.4. Inne operacje na plikach	132
Rozdział 9. Dopasowywanie do wzorców tekstowych	137
9.1. Wzorce	138
9.1.1. Funkcja <code>string.match</code>	140
9.1.2. Iterator <code>string.gmatch</code>	140
9.1.3. Klasy symboli	141
9.1.4. Modyfikatory wielokrotności	143
9.1.5. Dopasowywanie podciągów ograniczonych	146
9.1.6. Kotwice	147
9.2. Przechwytywanie fragmentów dopasowań	148
9.2.1. Podstawowy mechanizm przechwytywania fragmentów dopasowań	149
9.2.2. Wewnętrzne odwołania do przechwyconych fragmentów	150
9.3. Funkcja <code>string.gsub</code>	151
9.3.1. Podstawienia stałej wartości	152
9.3.2. Podstawienia wartości zależnej od fragmentu dopasowania	152
9.3.3. Podstawienia pochodzące z tabeli	154
9.3.4. Podstawienia zwracane przez funkcje	154
9.4. Prosty konwerter z języka $\text{\LaTeX}$ do języka XHTML	155
Rozdział 10. Moduły i pakiety	160
10.1. Funkcja <code>require</code>	160
10.1.1. Ścieżki do plików modułów	162
10.2. Własne moduły i pakiety	163
10.2.1. Prosty moduł	163
10.2.2. Inne zwyczaje	165
10.2.3. Podmoduły i pakiety	167
10.3. Oprogramowanie LuaRocks	168
10.3.1. Instalacja programu <code>luarocks</code>	168
10.3.2. Instalacja modułów	169
10.3.3. Dostęp do modułów z poziomu interpretera Lua	169
Czego nie znajdziesz w tej książce	171

II. LuaLaTeX

Rozdział 11. Wprowadzenie	177
11.1. Silniki i formaty	178
11.2. Czym się różnią pdfTeX i LuaTeX?	179
11.3. Jak przejść od pdfTeX do LuaTeX?	180

Rozdział 12. Wbudowany interpreter Lua	182
12.1. Polecenie <code>\directlua</code>	182
12.1.1. Polecenie <code>\directlua</code> w trzech krokach	183
12.1.2. Potencjalne problemy	183
12.1.3. Umieszczanie kodu Lua w zewnętrznych plikach	187
12.2. $\TeX$ i symbole specjalne	188
12.3. Pakiet <code>luacode</code>	191
12.3.1. Nowe polecenia i otoczenia	191
12.3.2. Tabliczka mnożenia	192
12.3.3. Debugowanie kodu	194
12.4. Podstawowe funkcje z biblioteki <code>tex</code>	194
12.4.1. Funkcje wypisujące kod $\TeX$ a	195
12.4.2. Funkcja <code>tex.error</code>	196
12.5. Biblioteka <code>texio</code>	197
Rozdział 13. Wywołania zwrotne	200
13.1. Wywołania zwrotne i biblioteka <code>callback</code>	200
13.1.1. Punkty kompilacji i ich identyfikatory	201
13.1.2. Podstawowe funkcje z biblioteki <code>callback</code>	201
13.1.3. Przykład – automatyczne pogrubienie liczb	202
13.2. Wywołania zwrotne i biblioteka <code>luatexbase</code>	203
13.2.1. Przykład – automatyczne formatowanie liczb	205
13.3. Wybrane punkty dla wywołań zwrotnych	206
13.3.1. Poszukiwanie plików	206
13.3.2. Przetwarzanie danych	207
Rozdział 14. Biblioteka <code>luautils</code>	210
14.1. Moduł <code>luautils-boolean</code>	211
14.2. Moduł <code>luautils-number</code>	212
14.3. Moduł <code>luautils-string</code>	213
14.4. Moduł <code>luautils-table</code>	214
14.5. Moduł <code>luautils-math</code>	216
14.6. Moduł <code>luautils-io</code>	216
14.7. Moduł <code>luautils-os</code>	218
14.8. Moduł <code>luautils-gzip</code>	218
14.9. Moduł <code>luautils-md5</code>	219
14.10. Moduł <code>luautils-dir</code>	219
14.11. Moduł <code>luautils-unicode</code>	221
Czego nie znajdziesz w tej książce	223
 III. Praktyczne przykłady	
Przykład 1. Quiz	229
1.1. Wymagania	229
1.2. Plik z bazą pytań	230
1.3. Moduł <code>quiz-module</code>	232
1.3.1. Zmienne wykorzystywane przez moduł	232
1.3.2. Funkcja sygnatury	232

1.3.3.	Funkcje związane z wczytywaniem danych	232
1.3.4.	Funkcje związane z generowaniem testu	236
1.3.5.	Funkcje związane z odczytywaniem testu	239
1.3.6.	Pełen kod źródłowy pliku <code>quiz-module.lua</code>	239
1.4.	Przykładowy plik <code>LuaTeXa</code>	243
Przykład 2.	Tabliczki	251
2.1.	Wymagania	251
2.2.	Moduł <code>csv</code>	253
2.2.1.	Funkcje <code>csv.open</code> i <code>csv.openstring</code>	255
2.2.2.	Iterator <code>lines</code>	256
2.2.3.	Parametr <code>columns</code>	258
2.3.	Tabliczki	260
2.3.1.	Wczytywanie danych z pliku	260
2.3.2.	Sortowanie listy pracowników	261
2.3.3.	Generowanie dokumentu	262
2.3.4.	Pełen kod źródłowy dokumentu	264
Przykład 3.	Rachunek	269
3.1.	Opracowane rozwiązanie	269
3.2.	Moduł <code>luasql.mysql</code>	271
3.2.1.	Środowisko MySQL i połączenie z bazą danych	272
3.2.2.	Operacje na bazie danych	274
3.2.3.	Odczytywanie pozyskanych danych	275
3.3.	Rachunek	277
3.3.1.	Numer rachunku	277
3.3.2.	Pobieranie danych instytucji	278
3.3.3.	Lista produktów	279
3.3.4.	Pełen kod źródłowy dokumentu	282
Dodatki		
Dodatek A.	Polecane lektury	291
Dodatek B.	Tabela znaków ASCII	292
Dodatek C.	Odpowiedzi do wybranych zadań	294
Indeks		315

Przedmowa

Kiedy mniej więcej w połowie 1977 r. Donald E. Knuth, amerykański akademik i ceniony specjalista w zakresie matematycznych podstaw informatyki, postanowił stworzyć zupełnie nowe oprogramowanie do cyfrowego składu tekstu, dawał sobie około sześciu miesięcy na ukończenie prac nad nim. I chociaż pierwsza wersja oprogramowania $\text{T}_{\text{E}}\text{X}$, bo o nim mowa, ujrzała światło dzienne już w 1978 r., to jednak dopiero 11 lat później, w 1989 r., uznano je za ukończone. Od tego momentu są poprawiane w nim jedynie nieliczne napotkane błędy.

Trzydzieści minionych lat wystarczyło, aby $\text{T}_{\text{E}}\text{X}$ stał się nieodłącznym kompanem w codziennej pracy naukowej matematyków, informatyków, fizyków oraz inżynierów. Trudno snuć domysły, ale wiele wskazuje na to, że istotny wpływ na popularność $\text{T}_{\text{E}}\text{X}$ a miało opublikowanie w 1985 r. przez Lesliego Lamport'a, innego amerykańskiego informatyka, pierwszego pokaznego zbioru makr dla języka $\text{T}_{\text{E}}\text{X}$. Dziś zbiór ten, nazwany $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$, jest na stałe wkomponowany w niemal wszystkie dystrybucje systemu $\text{T}_{\text{E}}\text{X}$, znacząco ułatwiając pracę z nawet najbardziej skomplikowanymi dokumentami.

W 1993 r., kilka lat po zakończeniu prac rozwojowych nad programem $\text{T}_{\text{E}}\text{X}$, opublikowano pierwszą implementację nowego języka programowania o nazwie Lua. Język ten, tworzony przez brazylijski zespół specjalistów z Robertem Ierusalimsem na czele, od początku miał być szybki, przyjazny i łatwy do integracji z istniejącym oprogramowaniem. Z perspektywy czasu można śmiało stwierdzić, że cel ten udało się osiągnąć, a sam język rozwijany jest do dzisiaj. Obecnie jest wykorzystywany przez twórców programów komercyjnych (np. Adobe Photoshop Lightroom) i takich o otwartym kodzie źródłowym (np. Audacity), ale także do usprawniania procesów prototypowania, szczególnie w branży gier komputerowych. W tej ostatniej kategorii, a więc wśród języków interpretowalnych wykorzystywanych do wsparcia procesu produkcji gier wideo, Lua zajmuje czołową pozycję.

Ta książka dotyczy jednak tego, co łączy Lua i $\text{T}_{\text{E}}\text{X}$ a (a właściwie $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ a). Te dwie na pozór odległe technologie – z jednej strony zaawansowany system składu tekstu, a z drugiej prosty i uniwersalny język programowania – skrzyżowały się jednak dopiero w 2007 r. To właśnie wtedy w San Diego, przepięknym mieście na zachodzie Stanów Zjednoczonych, tuż przy granicy z Meksykiem, odbywał się coroczny zjazd członków Grupy Użytkowników $\text{T}_{\text{E}}\text{X}$ a (*TeX Users Group*). Taco Hoekwater, z pochodzenia i zamieszkania Holender, zaprezentował tam wówczas efekty dwuletniej pracy nad programem $\text{LuaT}_{\text{E}}\text{X}$. Pozwalał on kompilować dokumenty napisane w języku $\text{T}_{\text{E}}\text{X}$, a przy tym osadzać w nich dodatkowy kod napisany w języku Lua, wykonywany automatycznie w czasie kompilacji dokumentu. Zastosowane w nim rozwiązania umożliwiały też dwustronną komunikację na linii $\text{T}_{\text{E}}\text{X}$ –Lua.

Zespół projektu Lua $\TeX$ planował wówczas, że pierwsza oficjalna wersja ich programu zostanie opublikowana pod koniec 2009 r. Tak też się stało, a już w 2010 r. została opublikowana jego pierwsza wersja stabilna. Mimo bardzo dynamicznego rozwoju, Lua $\TeX$ w wersji 1.00 (a więc w tej, która wypełniła wszystkie początkowe założenia autorów) został opublikowany dopiero we wrześniu 2016 r.

Można więc stwierdzić, że trzymasz w ręku, drogi Czytelniku, książkę, która staje na granicy dwóch na pozór odległych światów – programowania i składu tekstu. Po jednej stronie Lua – szybki, łatwy i użyteczny język programowania o licznych zastosowaniach. Po drugiej $\LaTeX$ – zaawansowany system składu tekstu o rosnącej z roku na rok popularności. Ich połączenie otwiera bez wątpienia wiele nowych możliwości, dla których inspirację tu znajdziesz.

Co znajdziesz w tej książce?

Pierwsza część książki, zatytułowana *Język Lua*, stanowi łagodne wprowadzenie do tego właśnie języka. Zależało mi na tym, aby było ono dostępne nawet dla osób, których dotychczasowe doświadczenie z programowaniem jest niezbyt duże. Odpowiednie rozdziały wprowadzają więc podstawowe pojęcia związane z programowaniem, prezentują najważniejsze konstrukcje języka Lua i szczegółowo wyjaśniają ich działanie. Każdy rozdział kończy się zestawem ćwiczeń i zadań. Większość ćwiczeń nie wymaga dostępu do komputera – mają one zmotywować Cię do zastanowienia się nad zdobytą wiedzą. Zadania mają jednak charakter bardziej praktyczny – polegają na napisaniu lub zmodyfikowaniu kodu w języku Lua tak, aby spełniał on pewne wymagania. Rozwiązanie niektórych zadań może wymagać od Ciebie odrobiny kreatywności, ale wierzę, że świetnie sobie z nimi poradzisz. Gdyby tak nie było, to w dodatku C znajdziesz wiele przykładowych rozwiązań, szczególnie do zadań z początkowych rozdziałów.

Powinieneś wiedzieć, że część pierwsza ma charakter progresywny. Oznacza to, że kolejne rozdziały opierają się silnie na rozdziałach wcześniejszych. Kiedy będziesz więc ją czytać pierwszy raz, powinieneś to robić od początku do końca. W każdej chwili możesz wrócić do poznanego już materiału, jeśli będziesz akurat potrzebować informacji dotyczących konkretnego zagadnienia. Już w pierwszych rozdziałach zakładałam jednak, że rozumiesz intuicyjnie pojęcia, takie jak zmienna, wartość i funkcja. Wydaje mi się, że nie jest to założenie przesadzone.

Druga część książki, zatytułowana *LuaLaTeX*, prezentuje możliwości programu o tej samej nazwie. Zależało mi, aby dopiero w rozdziale 11. pojawiły się pierwsze odwołania do $\TeX$ a i $\LaTeX$ a. Dzięki temu pierwsza część książki stanowi niezależny materiał, z którego możesz skorzystać bez żadnej straty nawet wtedy, gdy nie interesuje Cię dynamiczne tworzenie dokumentów. Zależność w drugą stronę niestety nie zachodzi – przed przystąpieniem do lektury części drugiej powinieneś zapoznać się z materiałem zebrany w części pierwszej. Przyjmuję też, że Twoja znajomość $\LaTeX$ a

jest wystarczająca, aby czytać i rozumieć bez problemu kod napisany w tym języku. Jeśli tak nie jest, to zrozumienie niektórych przykładów może wymagać od Ciebie sięgnięcia do innych książek (sugerowaną listę znajdziesz w dodatku A).

Podobnie jak to było w przypadku części pierwszej, tak i w części drugiej poszczególne rozdziały kończą się zestawami ćwiczeń i zadań. Ich rola nie różni się co prawda od roli ćwiczeń i zadań w części dotyczącej języka Lua, chociaż jest ich trochę mniej. Wierzę, że czytając o możliwościach Lua $\TeX$ a sam wpadniesz na wiele interesujących pomysłów, które zechcesz jak najszybciej wcielić w życie.

W trzeciej części książki znajdziesz kilka rozbudowanych przykładów wykorzystania Lua $\TeX$ a do dynamicznego tworzenia dokumentów. Przykłady te osadzone są w kontekście fikcyjnych problemów, chociaż wykorzystywane do ich rozwiązania techniki mają wyjątkowo ogólny charakter. Jeśli więc chcesz się dowiedzieć, czy nauczysz się z tej książki czegoś przydatnego – zajrzyj właśnie tam. Praktyczne przykłady mogą Ci z jednej strony posłużyć jako motywacja na początku przygody z Lua $\TeX$ em, a z drugiej jako inspiracja pod koniec czytania tej książki.

Podziękowania

Chciałbym podziękować osobom, których wsparcie miało nieoceniony wpływ na treść i formę tego wydania. Dziękuję Marcinowi Borkowskiemu i Pawłowi Mleczo za niezwykle inspirujące rozmowy o zawartości tej książki, jeszcze zanim na dobre zabrałem się do jej pisania. Gdyby nie ich doświadczenie, cierpliwość i otwartość, ta książka zapewne nigdy by nie powstała. Dziękuję również tym wszystkim, którzy poświęcili swój czas, żeby przeczytać fragmenty wstępnej wersji maszynopisu i podzielić się ze mną swoimi cennymi uwagami. Byli to (w kolejności alfabetycznej): Marcin Borkowski, Stanisław Gawiejnowicz, Mateusz Hromada, Marek Kaluba, Paweł Mleczo, Łukasz Pawluczuk, Piotr Rzonkowski i Paweł Urbanek. Dziękuję też dziekanowi Wydziału Matematyki i Informatyki Uniwersytetu im. Adama Mickiewicza w Poznaniu, który zgodził się dofinansować wydanie tej książki.

Bartłomiej Przybylski

Poznań, sierpień 2017