



Lua

Jakub Kuš



Co to jest Lua?

Lua to prosty ale wydajny język programowania.

Nazwa znaczy “Księżyc” po portugalsku,
a język pochodzi z Brazylii

Cechy języka:

- ★ funkcyjny
- ★ dynamicznie typowany
- ★ prosta składnia
- ★ 8 typów danych



Typy danych w Lua

The First Point: Basic Lua data types

Type	Example
nil	<code>a = nil</code>
boolean	<code>b = true</code>
number	<code>n = 1</code>
string	<code>s = 'abc'</code>
table	<code>t = {a, b, c}</code>
function	<code>foo = function(x) return 2*x end</code>
userdata	<code>u = some_C_struct</code>
thread	<code>co = coroutine.create(function () print("hi") end)</code>



nil

Wartość podobna
do null; brak
"normalnej" wartości.

main.lua X

main.lua > ...

```
1  x = nil
2  print(x)
3  print(type(x))
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS



powershell



```
PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> lua main.lua
nil
nil
PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> █
```

boolean

main.lua ×

main.lua > ...

```
1  x = true -- lub false
2  print(x)
3  print(type(x))
```


PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

 powershell

+ v



PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> lua main.lua

true

boolean

PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik>

number

main.lua X

main.lua > ...

```
1  x = 5 -- dynamiczne typowanie
2  local y = 4.7 -- zmienne domyślnie są globalne
3  print(x, y)
4  print(type(x), type(y))
5  print(x - y, y - x)
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS



powershell



PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> lua main.lua

5 4.7

number number

0.3 -0.3

PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik>

string

main.lua X

main.lua > ...

```
1  x = 'Izabela'
2  print(x)
3  print(type(x))
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS



powershell



PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> lua main.lua

Izabela

string

PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> █

table

main.lua ×

main.lua > ...

```
1  x = {} --pusta tablica
2  print(x[1]) --nil bo index nie istnieje
3  x = {4, 5, 7, 8}
4  print(x[1]) --4 bo tablice w Lua są indeksowane od 1 (a nie od 0)
5  x = {
6      ["key"] = "value" -- pary klucz wartość
7  }
8  print(x["key"])
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS



powershell



PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> lua main.lua

nil

4

value



table cd

Table w Lua służy do implementowania tych typów danych:

- tablice
- słowniki
- zbiory (ang. sets)
- kolejki
- Programowanie obiektowe:
 - Klasy
 - Dziedziczenie itp.

function

main.lua ×

main.lua > ...

```
1  function add(a, b)
2      return a + b
3  end
4  print(add(5, 6)) --11
5  print(type(add)) --function
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS



powershell



PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> lua main.lua

11

function

PS C:\Users\kubak\OneDrive\Pulpit\LuaPlik> █



userdata

Typ userdata umożliwia przechowywanie dowolnych danych C w zmiennych Lua. Nie ma żadnych predefiniowanych operacji do tych danych, z wyjątkiem przypisania i testu równości. Dane użytkownika służą do reprezentowania nowych typów tworzonych przez aplikację lub bibliotekę napisaną w C; na przykład standardowa biblioteka I/O używa ich do reprezentowania plików.

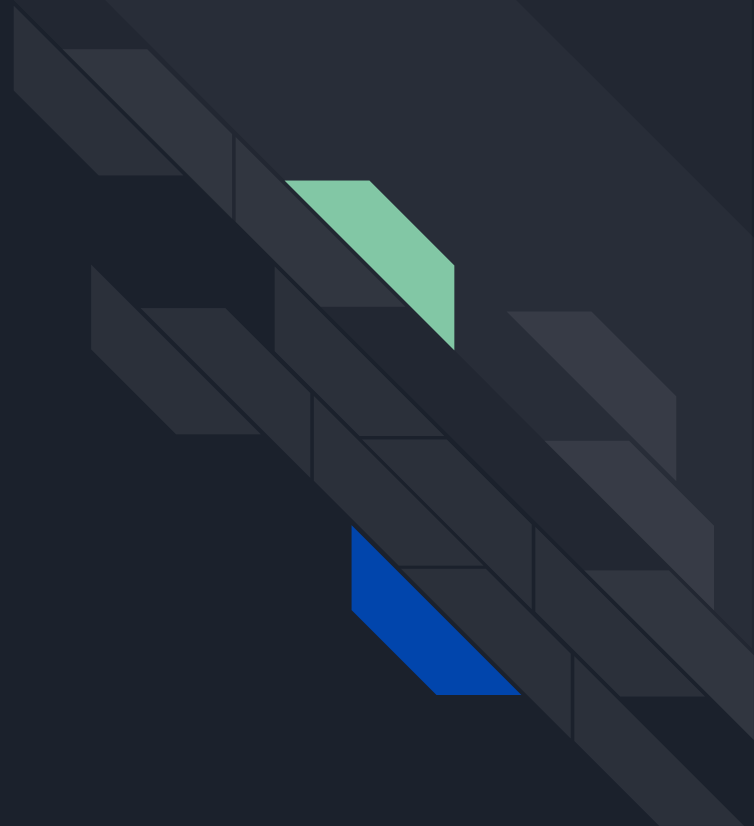


thread

Typ danych pozwalający na obsługę wątków programu przez sam program

Dziękuję za uwagę

Jakub Kuś





Bibliografia

<https://www.lua.org>

<https://scriptinghelpers.org>