Implementering av en Lua parser

Oskar Schöldström

Examensarbete
Informationsteknik
2012

EXAMENSARBETE	
Arcada	
Utbildningsprogram:	Informations- och medieteknik
Identifikationsnummer:	1234
Författare:	Oskar Schöldström
Arbetets namn:	Implementering av en Lua parser
Handledare (Arcada):	...
Uppdragsgivare:	

Sammandrag:
a a

Nyckelord:	parser, javascript, lua, lexer, tolk
Sidantal:	13
Språk:	Svenska
Datum för godkännande:	

DEGREE THESIS	
Arcada	
Degree Programme:	Information and Media Technology
Identification number:	1234
Author:	Oskar Schöldström
Title:	Implementing a Lua parser
Supervisor (Arcada):	...
Commissioned by:	

Abstract: a a

Keywords:	parser, javascript, lua, lexer, lexical analysis, tokenizer, scanner, lexical analyzer, syntactic analysis
Number of pages:	13
Language:	Swedish
Date of acceptance:	

OPINNÄYTE	
Arcada	
Koulutusohjelma:	Informaatio- ja mediatekniikka
Tunnistenumero:	1234
Tekijä:	Oskar Schöldström
Työn nimi:	Implementing a Lua parser
Työn ohjaaja (Arcada):	...
Toimeksiantaja:	

Tiivistelmä:
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas molestie, nisi sed consectetur tincidunt, ante libero euismod nulla, eget commodo ligula purus eu nibh. Ut enim risus, congue sed viverra ut, tempor vel velit. Quisque consectetur tincidunt scelerisque. Donec nec mollis leo. Donec laoreet purus a massa ultrices egestas. Ut quis neque nisi. In hac habitasse platea dictumst. Suspendisse suscipit, ante eget venenatis molestie, elit nulla aliquam ligula, sit amet consectetur tortor magna eu ipsum. Pellentesque tempor tincidunt arcu, eget ultrices lorem commodo a.

Avainsanat:	parser, javascript, lua, lexer, lexical analysis, tokenizer, scanner, lexical analyzer, syntactic analysis
Sivumäärä:	13
Kieli:	Ruotsi
Hyväksymispäivämäärä:	

INNEHÅLL

TABELLER

FIGURER

FÖRKORTNINGAR OCH DEFINITIONER

LL parser	asdasdasda
LR parser	asdasdasda
Syntax	asdasdasda
BNF	asdasdasda
Semantik	asdasdasda

FÖRORD

....

KÄLLOR

BILAGA - LUA GRAMMATIK

⟨*chunk*⟩ ::= ⟨*block*⟩

⟨*block*⟩ ::= {⟨*stat*⟩} [⟨*retstat*⟩]

⟨*stat*⟩ ::= **;**
| ⟨*varlist*⟩ = ⟨*explist*⟩
| ⟨*functioncall*⟩
| ⟨*label*⟩
| **break**
| **goto** Name
| **do** ⟨*block*⟩ **end**
| **while** ⟨*exp*⟩ **do** ⟨*block*⟩ **end**
| **repeat** ⟨*block*⟩ **until** ⟨*exp*⟩
| **if** ⟨*exp*⟩ **then** ⟨*block*⟩ {**elseif** ⟨*exp*⟩ **then** ⟨*block*⟩} [**else** ⟨*block*⟩] **end**
| **for** Name = ⟨*exp*⟩ **,** ⟨*exp*⟩ [**,** ⟨*exp*⟩] **do** ⟨*block*⟩ **end**
| **for** ⟨*namelist*⟩ **in** ⟨*explist*⟩ **do** ⟨*block*⟩ **end**
| **function** funcname ⟨*funcbody*⟩
| **local** ⟨*function*⟩ Name ⟨*funcbody*⟩
| **local** ⟨*namelist*⟩ [= ⟨*explist*⟩]

⟨*retstat*⟩ ::= **return** [⟨*explist*⟩] [**;**]

⟨*label*⟩ ::= **::** Name **::**

⟨*funcname*⟩ ::= Name {**.** Name} [**:** Name]

⟨*varlist*⟩ ::= ⟨*var*⟩ {**,** ⟨*var*⟩}

⟨*var*⟩ ::= Name | ⟨*prefixexp*⟩ [⟨*exp*⟩] | ⟨*prefixexp*⟩ **.** Name

⟨*namelist*⟩ ::= Name {**,** Name}

⟨*explist*⟩ ::= ⟨*exp*⟩ {**,** ⟨*exp*⟩}

⟨*exp*⟩ ::= **nil** | **false** | **true** | Number | String | **...** | ⟨*functiondef*⟩ | ⟨*prefixexp*⟩ | ⟨*tableconstructor*⟩ | ⟨*exp*⟩ ⟨*binop*⟩ ⟨*exp*⟩ | ⟨*unop*⟩ ⟨*exp*⟩

⟨*prefixexp*⟩ ::= ⟨*var*⟩ | ⟨*functioncall*⟩ | **(** ⟨*exp*⟩ **)**

⟨*functioncall*⟩ ::= ⟨*prefixexp*⟩ ⟨*args*⟩ | ⟨*prefixexp*⟩ **:** Name ⟨*args*⟩

⟨*args*⟩ ::= **(** [⟨*explist*⟩] **)** | ⟨*tableconstructor*⟩ | String

⟨*functiondef*⟩ ::= **function** ⟨*funcbody*⟩

⟨*funcbody*⟩ ::= **(** [⟨*parlist*⟩] **)** ⟨*block*⟩ **end**

⟨*parlist*⟩ ::= ⟨*namelist*⟩ [**,** **...**] | **...**

⟨*tableconstructor*⟩ ::= **{** [⟨*fieldlist*⟩] **}**

⟨*fieldlist*⟩ ::= ⟨*field*⟩ {⟨*fieldsep*⟩ ⟨*field*⟩} [⟨*fieldsep*⟩]

⟨*field*⟩ ::= [⟨*exp*⟩] = ⟨*exp*⟩ | Name = ⟨*exp*⟩ | ⟨*exp*⟩

⟨*fieldsep*⟩ ::= **,** | **;**

⟨*binop*⟩ ::= + | **-** | * | / | ^ | % | **..** | < | <= | > | >= | == | ~= | **and** | **or**

⟨*unop*⟩ ::= **-** | **not** | #

BILAGA - KÄLLKOD