

Esoterische Programmiersprachen

Bachblütentee gets Brainf*cked

Sven Gregori, CN8

<gregori@hs-furtwangen.de>

UnFUG SS 2008
Hochschule Furtwangen

3. April 2008



Esoterische Programmiersprachen

- haben nichts mit Esoterik an sich zu tun
- verfolgen Konzepte fern der konventionellen Programmiersprachen
- sind nicht für den praktischen Einsatz entwickelt worden
- Anwendung z.B. als Proof of Concept
- oft nur Theorien/Ideen ohne wirkliche Implementierung
- können hohen akademischer Wert haben
- sind aber oftmals einfach nur ein Witz

Absichten und Ziele

- kreative, verwirrende oder schlichtweg abnormale Syntax und Konzepte
- ein Ziel kann sein, selbst reproduzierende Programme (Quines) zu ermöglichen
- manche Sprachen sind sogar mit einem gewissen Eigenleben ausgestattet

INTERCAL

- 1972, Donald R. Woods und James M. Lyon
- gilt als die erste esoterische Sprache
- Implementierung für IBM S/360
- 1990 Unix Portierung durch Eric S. Raymond
- INTERCAL steht für "Compiler Language With No Pronounceable Acronym"

INTERCAL Operatoren Namen (Auszug)

.	spot
:	two-spot
,	tail
;	hybrid
"	rabbit ears
-	bookworm
'	hookworm
∇	overpunched worm on a V
¢	change
\$	big money
!	wow
?	what
~	sqiggle
#	mesh
=	half mesh

INTERCAL Sprachkonzepte

- COME FROM statt GOTO
- PLEASE Operator
 - allen Operatoren vorangestellt
 - wird er nicht oft genug verwendet, gilt das Programm als zu unfreundlich und wird nicht kompiliert
 - wird es zu oft verwendet, ist das Programm übermäßig freundlich und wird ebenfalls nicht kompiliert
 - diese Tatsache war undokumentiert

FALSE

- 1993, Wouter van Oortmerssen
- Stack orientiert
- Absicht möglichst verwirrenden und unlesbaren Code zu haben
- 1024 Byte großer Compiler (Motorola 68k Assembler)
 - inspirierte Urban Müller eine Sprache mit kleinerem Compiler zu entwickeln

Brainfuck

- 1993, Urban Müller
- Absicht eine Turing-vollständige Sprache mit kleinstmöglichem Compiler zu entwickeln
 - 240 Byte für Amiga (Urban Müller)
 - 171 Byte für x86 Linux (Brian Raiter)
- 8 Befehle
- Operation auf Byte-Array
- Pointer innerhalb des Arrays
- Input- und Outputstream

Brainfuck Befehle

Befehl	C Äquivalenz	Bedeutung
<	<code>++ptr;</code>	increment pointer
>	<code>--ptr;</code>	decrement pointer
+	<code>++*ptr;</code>	increment array value
-	<code>--*ptr;</code>	decrement array value
.	<code>putchar(*ptr);</code>	output current byte
,	<code>*ptr = getchar();</code>	read into current byte
[<code>while(*ptr) {</code>	loop begin
]	<code>}</code>	loop end

Hello Brainfuck World

```

+++++++ [ > ++++++ > ++++++
++>+++>+<<<<- ] >+ . >+ . ++++++
. . +++. >+ . <<+++++++ .
> . +++. ----- . ----- . >+ . > .

```

Hello Brainfuck World

```

+++++++
[>++++++>+++++++>+++>+<<<<-] initialize array
>++.          Print 'H'
>+.          Print 'e'
++++++.      Print 'l'
.            Print 'l'
+++         Print 'o'
>++.       Print ' '
<<+++++++ Print 'W'
>.         Print 'o'
+++       Print 'r'
----- .  Print 'l'
----- .  Print 'd'
>+.       Print '!'
>.        Print newline

```

Brainfuck Beispiele

- Zelle auf Null setzen
[-]
- Input Echo auf Output
, [.,]
- Destruktives Addieren zweier Zahlen
[->+<]
- Addieren zweier Zahlen
[->+>+<<]>>[-<<+>>]
- ASCII Tabelle ausgeben
.+[.+]]

Brainfuck Praxis

Einfache Struktur ermöglicht allmögliche Implementierungen

- Brainfuck CPU in VHDL von Clifford Wolf
- Brainfuck Computer von Robert Östling
- nativer PHP Brainfuck Interpreter

Brainfuck Variationen

- variable Arraygrößen (30000 als Standard)
- Kommentar Handhabung
- Brainfuck 2D
 - Zweidimensionales Array
- Brainfork
 - Ausführung von mehreren Threads
- Brainloller und Braincopter
 - Befehle sind definierte RGB Werte in PNG Bild
- Boolfuck und Smallfuck
 - Zellen sind Bits statt Bytes
- etc.

Ook!

- Brainfuck für Orang Utans
- Übersetzung der Befehle

Brainfuck	Ook!
<	Ook. Ook?
>	Ook? Ook.
+	Ook. Ook.
-	Ook! Ook!
.	Ook! Ook.
,	Ook. Ook!
[Ook! Ook?
]	Ook? Ook!

- für Kühe existiert auch *COW* mit "boviner syntax"

Whitespace

- hat 3 Sprachelemente:
 - Leerzeichen
 - Tabulator
 - Newline
- Kombinationen ermöglichen
 - Stack und Heap Operationen und Manipulationen
 - Arithmetik Operationen
 - Flusskontrolle (Subroutinen, Jumps)
 - Input/Output

Hello Whitespace World

IRC

- basierend auf IRC Syntax in Form einer Unterhaltung verschiedener Personen
 - unvoiced Kommentar
 - voiced Variable
 - op Befehl
 - channel Thread (eigenes sourcefile)
- /join und /part startet bzw. endet Thread
- /quit endet Programm
- topics dienen als Label

IRP

- Internet Relay Programming
- kooperativer Programmier Ansatz
- wird in #IRP auf irc.freenode.net interpretiert
- Befehle bestehen aus freundlichen Fragen in Plain English

```
<GregorR> Please say "Hello, World!"
```

```
<jix> Hello, World!
```

```
<GregorR> Please, some one write the first 16 numbers  
of the Fibonacci Sequence.
```

```
<calamari> 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
```

```
<GregorR> Please, write the 99 bottles lyrics
```

```
<memonic> go to hell
```

LOLCODE

- Sprache basierend auf Lolcats Bilder
- alles in CAPSLOCKED Internet Slang
- beginnend mit "HAI", endet mit "KTHXBYE"
- wächst durch Vorschläge aus Forum Community
- Beispiele

```
foo, WTF?      switch (foo) {
OMG, "bar"     case bar:
GTFO           break;
LOL foo R bar  foo = bar;
IZ foo? YARLY  if (foo) {
NOWAI         } else {
KTHX         }
```

Java2K

- nicht-deterministische, probabilistische Sprache
- Funktionen machen nur mit einer gewissen Wahrscheinlichkeit was sie wirklich machen sollen
→ es existieren jeweils zwei verschiedene Implementierungen
- anhand von PRNG wird zur Laufzeit tatsächliche Implementierung gewählt
- fast alle Builtin Funktionen haben 90% Wahrscheinlichkeit richtig zu laufen
- Reiz ist es Techniken zu entwickeln die eine höhere Wahrscheinlichkeiten erreichen

TMMLPTEALPAITAFNFAL

- "The Multi-Million Language Project To End All Language Projects And Isn't That A Fine Name For A Language"
- feste Syntax in Art eines "obsoleten BASIC Dialekts" - aber
- ändert jeden Tag Restriktionen in Bezug auf erlaubten Befehlen und Identifiern
→ Programme müssen täglich umgeschrieben werden

TMMLPTEALPAITAFNFAL - Tag n

- You can use GOSUB, but not GOTO
- You can use DIV, but not MOD
- Only the following high-level constructs are supported:
IF-THEN-ELSE, IF-THEN-UNLESS, IF-THEN-PROVIDED,
WHILE-DO-PROIDED, REPEAT-UNLESS.
- Names must be at least 15 Characters long
- Names must be less than 110 Characters long
- Identifier characters must be in ASCII range 33 .. 68 (! .. D),
or 'R'.
- The character 82 ('R') must be used 57 Times modulo 58 per
identifier.

TMMLPTEALPAITAFNFAL - Tag n+1

- You can use GOTO, but not GOSUB
- You can use MOD, but not DIV
- You have only UNTIL-DO as a high-level construct
- Names must be less than 21 characters long.

Unnecessary

- Absicht einfach zu lernen und implementieren
- Programme < 1 Bit
- Interpreter
 - macht ein NOP Befehl wenn Quelldatei nicht existiert
 - wirft Fehlermeldung wenn Quelldatei existiert
- "Run file 'example.unn' with your Unnecessary interpreter. (Note: First make sure there isn't a file called 'example.unn'. If there is, remove it before executing.)"

Whenever

"Whenever is a programming language which has no sense of urgency. It does things whenever it feels like it, not in any sequence specified by the programmer."

Wierd

- graphische (ASCII), Stack orientierte Sprache
- unterscheidet nur zwischen zwei Symbolen
 - Whitespace Zeichen
 - Non-Whitespace Zeichen
- eigentliches Programm ist eine Kette von beliebigen Zeichen
- Richtung bestimmt Befehl
- abhängig von Winkeln (in 45 Grad Abständen)
- direkte Manipulation nur durch "Push data value of '1' onto the Stack" und Subtraktion

Tree

- ebenfalls graphisch (ASCII) und Stack orientiert
- Aufbau als Baumstruktur
- kennt drei Konstrukte
 - Zweige (bestimmen Programmfluss)
 - Blätter (Stack Manipulation und I/O)
 - Insekten (Flusskontrolle)
 - alles andere wird als direkter Wert genommen
 - Zahlen werden erst zusammengefügt und dann als Wert genommen
- Programm beginnt bei der Wurzel und traversiert dann durch die Baumstruktur (vgl. Baum als Datenstruktur)

Cvleamar

- Aufgebaut als textuelle Representation eines gerichteten Graphen
- Eingangsknoten schickt etwas an Ausgangsknoten
- schicken mehrere Knoten etwas an einen Ausgangsknoten, werden die Werte zusammenaddiert
- Syntax:
(Eingangsknoten) (Ausgangsknoten) (Befehl) (Parameter)

Cvleamar Befehle

C	CONST	Put n to out node if in node is zero, otherwise out zero
V	VAR	n is initial value. Put old value to out node, set new value to in value
L	LESS	Output in value if is less than n, otherwise zero
E	EQUAL	Output in value if is equal to n, otherwise zero
M	MULTIPLY	Output n times in value
A	ARRAY	Put value of ARRAY[n OR i] to output (o), then flip bit stored in array.
R	REGISTER	Set stored value to $(i * v + 1) \text{ MOD } n$, then send stored value to output. (No modulus if $n=0$)

Piet

- graphische, Stack orientierte Sprache
- GIF Bilddatei als Quellcode
- benannt und Konzept von Piet Mondrian, Pionier der abstrakten Kunst
- "Codel" als semantische Einheit (Code + Pixel)
- 20 Farben

hellrot	rot	dunkelrot
hellgelb	gelb	dunkelgelb
hellgrün	grün	dunkelgrün
hellcyan	cyan	dunkelcyan
hellblau	blau	dunkelblau
hellmagenta	magenta	dunkelmagenta
weiss	schwarz	

Piet

- Grad des Farbton- und Helligkeitwechsels bestimmt auszuführenden Befehl
- Farbtöne und Helligkeiten sind zyklisch angeordnet
- weiss ist neutrale Farbe
- schwarz ist Programmflussgrenze
- alle anderen Farben sind Implementierungsabhängig
- zusätzlich "Direction Pointer" und "Codel Choser"
→ bestimmen den Programmfluss durch das Bild, sprich welches Codel als nächstes gewählt wird

Prelude und Fugue

Prelude

- musikalischer Hintergrund, unterteilt in mehrere "Stimmen"
- jede Stimme hat eigenen Stack und wird parallel ausgeführt

Fugue

- Schwestersprache von Prelude
 - Prelude ist in ASCII dargestellt
 - Fugue mit gleicher Semantik als polyphone Musikdatei (MIDI)
- ASCII Befehle werden in ansteigenden/abfallenden Intervallen repräsentiert

Chef

- Programme in Form von Rezepten
- "sollen nicht nur richtige Ergebnisse liefern, sondern auch leicht zuzubereiten und lecker sein"
- Sprachelemente
 - Variablendeklarationen über Zutatenliste
 - Zutaten - numerische Werte
 - flüssige Zutaten sind Unicode Daten
 - Maßeinheiten z.B. ml, l, dash(es)
 - andere Zutaten sind direkt Zahlen
 - Maßeinheiten z.B. kg, g, pinch(es)
 - Schüssel - Stack (mehrere verfügbar)
 - Kühlschrank - stdin
 - ...

Haifu

- adressiert die asiatische Philosophie
- respektiert die Natur und beachtet ihre Schönheit
- Code ist in Form eines Haikus - allerdings in englischer Form bezüglich Silbentrennung
→ 3 Zeilen mit 5-7-5 Silben
- Programm kann mehrere, sequenzielle Haikus enthalten
- true und false ersetzt durch Yin und Yang
- gilt auch für Integer und Gleitkommazahlen (gerade Yin, ungerade Yang)
- Yin und Yang müssen ausgeglichen sein, damit das Programm wirklich funktioniert

Haifu

- Stack Struktur auf spiritueller und nicht materieller Ebene
- Sprache basiert auf 5 Elemente - Erde, Feuer, Wasser, Holz und Metall
- physikalische und emotionale Beziehungen untereinander
- Variablen-Namen haben Bezug zu ihrem Element
→ rock, flame, rain, tree, iron
- arithmetische Operatoren
 - rain creates tree (Addition)
 - flame destroys tree (Subtraktion)
 - tree fears flame (Division)
 - iron loves flame (Multiplikation)

.Gertrude

- benannt nach der Schriftstellerin Gertrude Stein
- benutzt Poesie als Quelltext
 - Aneinanderreihung von Sätzen in natürlicher Sprache
- von jedem Satz wird berechnet
 - die durchschnittliche Wortlänge
 - das Verhältnis [Wörter länger als Durchschnitt] zu [Wörter die kürzer sind als Durchschnitt]
- bestimmte Längenverhältnisse bedeuten bestimmte Befehle
- Jeder Satz steht also für einen Befehl

.Gertrude - Beispiele Längenverhältnisse

14/10	Addition
10/20	Addition mit Zuweisung
11/7	Subtraktion
14/8	Modulo Operation mit Zuweisung
22/23	OR Verknüpfung
19/22	AND Verknüpfung mit Zuweisung
2/7	Funktionsaufruf
16/15	Variablen Definition
7/14	Integerwert als Zeichen ausgeben
6/10	if-then
1/x	Integerwert x

insgesamt 43 Befehle

Petrovich

- sogesehen ein esoterisches Betriebssystem
- benannt nach Ivan Petrovich Pavlov (Pawlowscher Hund)
- System basiert auf dessen Forschungen
- lernfähiges System mit "reward" und "punishment" Methodik
- man sagt dem System, es soll irgendetwas (mit irgendwas) machen, und es handelt eigenständig
- nach Bestrafung wird es sich hüten, nochmal das falsche zu machen
- "And in case you think this is entirely a joke, imagine a Petrovich layer over another operating system, such as Microsoft Windows (TM). Every time Windows does something you don't like, you could punish it, and it would *never do it again...*"

Links

<http://esolangs.org/>

<http://esoteric.sange.fi/>

<http://www.dangermouse.net/esoteric/>

http://p-nand-q.com/humor/programming_languages.html

Wikipedia

Fragen?