Dynamische Programmiersprachen

Von Lisp und Smalltalk zu Perl, Python, Ruby, Groovy ...

WS 2008/2009

Themen, betreut von

Univ.-Prof. em. Dr. H.-J. Hoffmann

(Stand 9. November 2008)

Die Literaturangaben in den einzelnen Themenblättern dienen als Anleitung zur jeweiligen Thematik. Die Bearbeiter müssen sich um einen darüber hinausgehenden Überblick bemühen.

Angaben noch unvollständig !

Alle Internet-Zugriffe September – Oktober 2008 !
Irrtum und Tippfehler vorbehalten !
Alle Wikipedia-Referenzen sind von 2008 !

Leider sind in einigen angegebenen Literaturstellen immer wieder Tippfehler zu finden.

Anleitung zur Vorbereitung Ihres Vortrags bzw. der Ausarbeitung:

- S.I.P. Jones et al.: *How to give a good research talk*; ACM SIGPlan Notices 28 (1993) 11, 9 - 12
- M. Deininger et al.: *Studien-Arbeiten, ein Leitfaden* ...; (u.a.) Teubner, 1992

Beides kann in der Bibliothek des FB Informatik eingesehen werden !

Inhaltliche Vorbereitung für alle:

L. Tratt, R. Wuyts: *Dynamically typed languages* ; IEEE Software 24 (2007) 5, 28–30 – als „Guest editor's introduction" für dieses IEEE Software Themenheft über „Dynamically typed languages" –

H. Erdogmus: *So many languages, so little time* ; IEEE Software 25 (2008) 1, 4–6 – als derzeitiger „Editor in Chief" von IEEE Software –

D. Ascher: *Dynamic languages: Ready for the next challenges, by design* ; White paper, 2004, **www.activestate.com/company/newsroom/whitepapers_ADL.plex**

C. Albrecht, S. Tilkov: *OOP 2008 – Dynamic languages shootout* ; 2008, u.a. **www.heeg.de/images/OOP2008/sigs-datacom%20-fuer%20Webseite/albrecht_tilkov_JS_02_08.pdf**

siehe weiter (u.a.)

en.wikipedia.org/wiki/Comparison_of_programming_languages
en.wikipedia.org/wiki/Dynamic_programming_language
en.wikipedia.org/wiki/Dynamic_binding

Auszug aus
en.wikipedia.org/wiki/Dynamic_programming_language

Stand aus 2004 mit einigen Ergänzungen

* APL
* Befung
* ChucK
* ColdFusion
* Curl
* D
* dBASE (dBL)
* ECMAScript
  - o ActionScript
  - o DMDScript
  - o E4X
  - o **JavaScript**
  - o JScript
* Eiffel
* **Erlang**
* Forth
* **Groovy**
* HyperCard/HyperTalk and Derivatives
  - o HyperCard/HyperTalk
  - o Revolution/Transcript
  - o SuperCard/SuperTalk
* Io
* **Lisp**
  - o **Common Lisp**
  - o Dylan
  - o Emacs Lisp
  - o Logo
  - o Lisp_Machine_Lisp
  - o **Scheme**
* **Lua**
* Maude system
* Oberon
* Objective Modula-2
* Objective-C
* **Perl**
* PHP
* Pliant
* POP-11
* Poplog
* Pike
* Prolog
* **Python**
* REALbasic
* REBOL
* **Ruby**
* Scratch

aktuell fehlend: **Seaside**

* **Smalltalk**
  - o Bistro
  - o **Self**
  - o Slate
  - o Squeak
  - o StrongTalk
* **Snobol**
* SuperCollider
* Tcl
  - o XOTcl
* TeX macro language
* VBScript
* Visual Basic 9 or 10
* Visual FoxPro
* Water
* Windows PowerShell
* xHarbour

Übersicht

- Einführung
  - #01 Was heißt *dynamisch* im Zusammenhang mit Programmierung und Programmiersprachen?
  - #02 Wurzeln: *Lisp/Scheme, Snobol/Icon, Smalltalk-80/VisualWorks*
- Umfeld
  - #03 Zeitpunkt des Bindens
  - #04 Web-Programmierung – *Seaside*
  - #05 Web-Programmierung – *JavaScript* *
- *Smalltalk*-Familie
  - #06 *Smalltalk* an sich
  - #07 *Smalltalk*-Weiterentwicklung: *"Byte code"/"Just-in-time compilation"*
  - #08 *Smalltalk*-Weiterentwicklung: u.a. *Self, Perl*
- Funktionale Sprachen a la *LISP*
  - #09 *"Closures" / "Continuations"*
- Moderne Entwicklungen
  - #10 Dynamik bei *Python*
  - #11 Dynamik bei *Ruby*
  - #12 Dynamik bei *Groovy**
  - #13 Was bietet *Lua* *
- Nicht vergessen
  - #14 Parallelverarbeitung: *Erlang* *
  - #15 Entwicklungsaufwand und Ausführungszeit *

Mindestteilnehmerzahl 10. Es werden **maximal 12 Themen** ausgegeben !

Die mit * markierten Themen stehen am Ende zur Auswahl,

Themen #01 bis #04 und #06 bis #11 werden zuerst vergeben !

Es handelt sich jeweils nicht um einen Programmierkurs der angesprochenen Programmiersprachen !

Es geht darum herauszuarbeiten, wie dynamische Effekte (was immer diese sind) beim Programmieren bzw. Ausführen zu erreichen sind, für was sie nützlich sind und wie sie implementiert werden können.

HJH#01
2008/2009

Was heißt *dynamisch* im Zusammenhang mit Programmierung und Programmiersprachen?

- S. Tilkov: Dynamische Sprachen im Aufwind ; ObjektSpektrum ?? (2008) 4, 28-29
- O. Nierstrasz et al.: *On the revival of dynamic languages* ; *Proc. Software Composition 2005*, Springer LNCS 3628, 2005, 1-13
- NN (Wikipedia): *Type system ;* `en.wikipedia.org/wiki/Dynamically_typed`
- D. Ungar/E. Ernst: *Dynamic languages ... unleash creativity* versus *Explicitly declared static types, the missing link* ; point → ← counter point, IEEE Software 24 (2007) 5, 72 – 75
- Luca Cardelli: *Object-based vs class-based languages* ; ACM PLDI 1996 Tutorial, 1996
- A.Savidis: *An enhanced form of dynamic untyped object-based inheritance* ; 2008, `www.jot.fm/issues/issue_2008_05/article2/index.html`

HJH#02
2008/2009

Wurzeln: *Lisp/Scheme, Snobol/Icon, Smalltalk-80/VisualWorks*

- F. Armbruster et al.: *The History of Programming Languages* ; 2001, `www2.lv.psu.edu/ojj/courses/ist-240/reports/spring2001/fa-cb-bc-kf/1951-1970.html` + `.../1971-1990.html` + `.../1991-2001.html`
- R. Toal: *Programming Languages* ; undatiert, `www.cs.lmu.edu/~ray/notes/intropl/`
- S.E. Keene: *Object-oriented programming in Common Lisp* ; Addison Wesley, 1989

Lisp/Scheme siehe auch Referat #09

- NN (Wikipedia): *(Über Snobol und Icon)* ; `de.wikipedia.org/wiki/Benutzer:WStephan/Dodge_Ram`
- H. Fosdick: *Programming Languages for Library and Textual Processing* ; Amer Soc Inform Science & Techn 31 (2005) 6 `www.asis.org/Bulletin/Aug-05/fosdick.html`

Smalltalk-80/VisualWorks/VisualAge siehe Referate #06 - #08

Zeitpunkt des Bindens

- NN (Wikipedia): *Name binding* ; **en.wikipedia.org/wiki/Late_binding**
- NN (Wikipedia): *Dynamic binding* ; **en.wikipedia.org/wiki/Dynamic_binding**
- D.C. Schmidt: *Dynamic binding C++* ; 2006, **www.cs.wustl.edu/~schmidt/PDF/C++-dynamic-binding4.pdf**
- L. Dai: *Static and Dynamic Behavior* ; 2007, **ac-staff.seattleu.edu/dingle/web/510/chapter11.ppt**
- O. Kiselyov et al.: *Dynamic binding, binding evaluation contexts, and (delimited) control effects* ; 2007, **okmij.org/ftp/Computation/dynamic-binding.html**
- H.-C. Stotts: *Dynamic Method Binding, Inheritance* ; 2003, **rockfish.cs.unc.edu/COMP144/lect24a.ppt**

HJH#04
2008/2009

Web-Programmierung – *Seaside*

- J.K. Ousterhout: *Scripting - Higher-level programming for the 21th century* ; IEEE Computer, 31 (1998) 3, 23-30

Referate über *Perl*, *Python* und *Ruby (on Rails)* beachten und verbinden !

- NN (Wikipedia): *Seaside (software)* ; `en.wikipedia.org/wiki/Seaside_(software)`
- S. Ducasse et al.: *Seaside –A multiple control flow Web application framework* ; ESUG Conf Research Track, 2004, 231-254
- S. Ducasse et al.: *Seaside – A flexible environment for building dynamic web applications* ; IEEE Software 24 (2007) 5, 56-63
- L. Renggli: *Seaside – Dynamische Web-Entwicklung ganz nach Ihrem Geschmack* ; Video, 2008, über `www.slideshare.net/`
- NN (Cincom): *Seaside for Cincom Smalltalk* ; `www.cincomsmalltalk.com/userblogs/cincom/blogView?content=seaside_info`

- R. Leon: *Rails vs Seaside* ; 2007, `onsmalltalk.com/programming/smalltalk/rails-vs-seaside/`

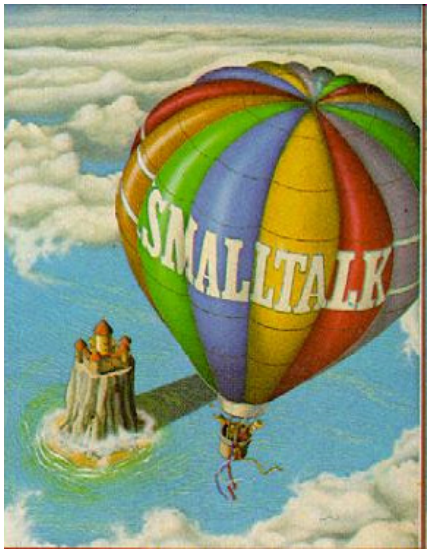
Web-Programmierung – *JavaScript*

- M. Lubkowitz: *Webseiten programmieren und gestalten* ; Galileo Computing, 2005
- M. Doernhoefer: *Surfing the net for software engineering notes* ; ACM SIGSoft 31 (2006) 4, 16-24

„Dynamisch“ in unserem Sinn ?

- M. Lubkowitz: -- siehe oben --; Teil 4, 371 ff.
- NN (Markt & Technik): *Wozu JavaScript ?* ; undatiert, `84.113.22.230:7980/books/DynamicWeb_in_21Tagen/WP_kap08.html`
- J. Perry: *JavaScript Dynamic Document Creation* ; 1999, `www.webdevelopersjournal.com/articles/dynamic_doc_creation.html`
- NN (): *JavaScript – Dynamic content* ; 2002, `www.mcli.dist.maricopa.edu/tut/tut27b.html`

Smalltalk an sich I



- NN (Wikipedia): *Smalltalk* ;
  en.wikipedia.org/wiki/Smalltalk_programming_language
- A.C. Kay: *The Early History of Smalltalk* ; ACM History of Programming Languages II (1996) 1996, 511-578
- A. Goldberg, D. Robson: *Smalltalk-80 – The language* ; Addison Wesley,1989 [*]
- NN: *JPMorgan leitet klare Vorteile aus Cincom Smalltalk ab* ; Firmenschrift Cincom Systems, Inc., 2006
- D. Ingalls et al.: *Back to the future – The story of Squeak, a practical Smalltalk written in itself* ; ACM SIGPLAN Notices 32 (1997) 11, 318-326 falsch! 10, 318-326 richtig !
- G. Giorgi: *Smalltalk tutorial for Java programmers!* ; 2002.
  daitanmarks.sourceforge.net/or/squeak/ST4J.pdf

Fortsetzung

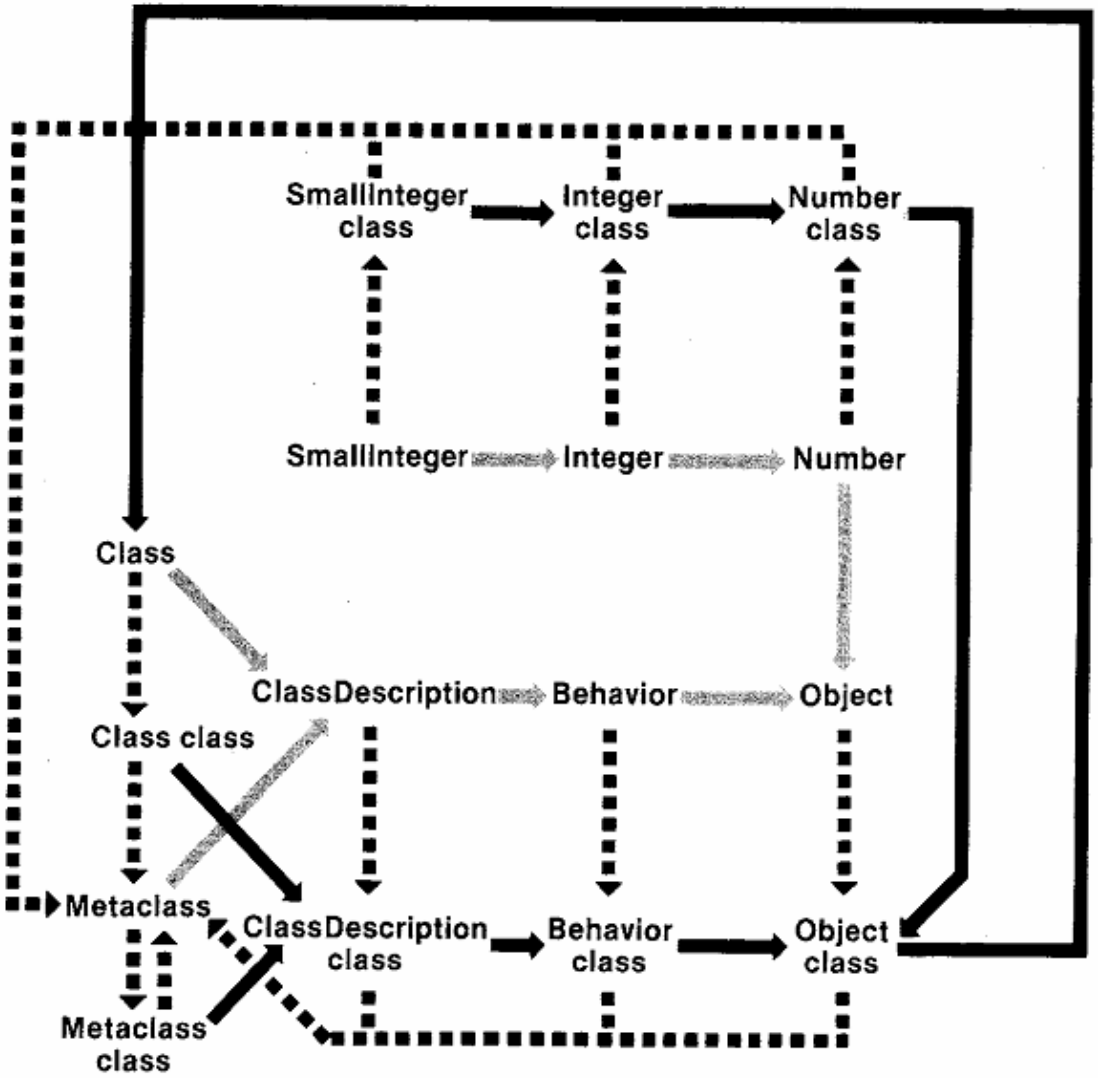
Smalltalk an sich IIa

Figure 16.5 aus [*, p 272]:
Metaklassen- und Klassen-hierarchie bei *Smalltalk-80*

A ⟶ B Klasse A ist Unterklasse der (Meta-)Klasse B

A ⟹ B Klasse A ist Unterklasse der Klasse B

A ⇢ B A ist Ausprägung der Klasse B (auf diesem Blatt sind A bzw. B Klassenobjekte !)



Smalltalk an sich IIb

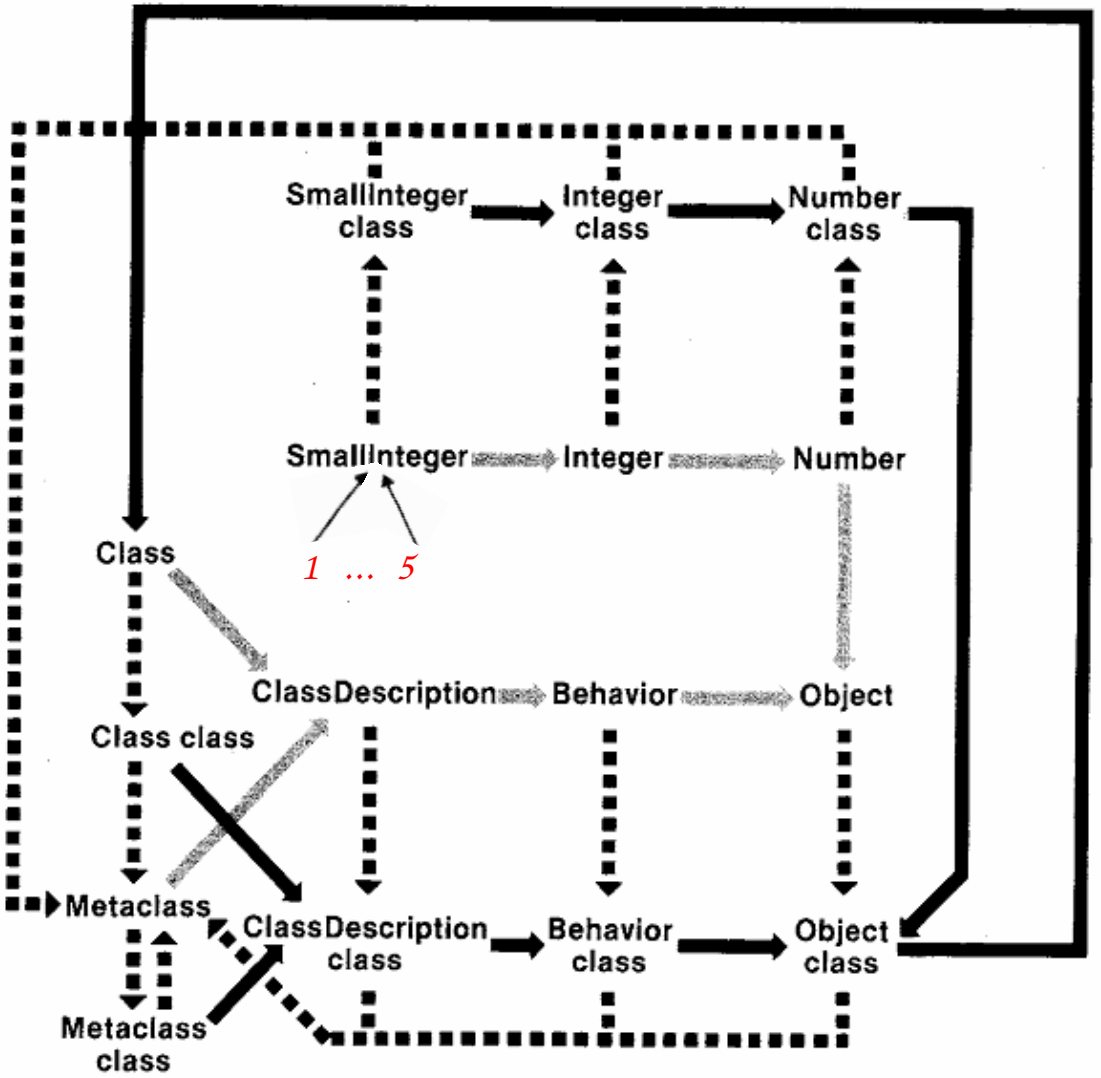
A ⟶ B Klasse A ist Unterklasse der (Meta-)Klasse B

A ⟹ B Klasse A ist Unterklasse der Klasse B

A ⇢ B A ist Ausprägung der Klasse B (auf diesem Blatt sind A bzw. B Klassenobjekte !)

A → B A ist Ausprägung der Klasse B (1 ... 5 sind Beispiele von eigentlichen Objekten)

(im Speziellen anders gelöst)



Smalltalk an sich III

- NN (Wikipedia): Metaclass ; **en.wikipedia.org/wiki/Metaclasses**
- B. Foote, R.E. Johnson: *Reflective Facilities in Smalltalk-80 ;* ACM Sigplan Notices 24 (1989) 10, 327-335
- F. Rivard: *Smalltalk – A reflective language* ; 1996, **www2.parc.com/csl/groups/sda/projects/reflection96/docs/rivard/rivard.html**
- R. Razavi et al.: *Language support for adaptive object-models using metaclasses* ; Computer Languages, Systems & Structures 31 (2005) 3-4, 199-218
- G. Brose: *Reflection in Java, CORBA and JacORB*. In C. Cap, (ed.); Proc. JIT'98. Springer,1998, 238 ff. [Inform Bib A2/JIT/98]
- F. Bühler: *Reflection in Java, Lisp und Smalltalk* ; 2006, ***seal.ifi.uzh.ch/fileadmin/User_Filemount/Vorlesungs_Folien/Seminar_SE/SS06/buehler_reflection.pdf***

Smalltalk –Weiterentwicklung I: *"Byte code"/"Just-in-time compilation"*

- NN (Wikipedia): *Virtual machine* ;
  en.wikipedia.org/wiki/Virtual_machine
- NN (Wikipedia): *Bytecode* ;
  en.wikipedia.org/wiki/Bytecode
  bzw. **de.wikipedia.org/wiki/Bytecode**
- A. Kind: *Bytecode-Interpretierung* ;
  www.gi-ev.de/service/informatiklexikon/informatiklexikon-detailansicht/meldung/48/
- A. Goldberg, D. Robson: *Smalltalk-80: The Language and Its Implementation* (Auszug); Addison-Wessley, 1983/85
- NN (Cincom): *Drilling down to bytecode* ; 2008,
  www.cincomsmalltalk.com/blog/blogView?showComments=true&entry=3378011312
- C. Baumann: *Smalltalk – ein Blick hinter die Kulissen* ; 2006,
  www.complang.tuwien.ac.at/anton/lvas/sem06w/baumann.pdf

Fortsetzung

HJH#07b
2008/2009

Smalltalk –Weiterentwicklung II: ***"Byte code"/****"Just-in-time compilation"*

- NN (Wikipedia): *Java bytecode* ;
  en.wikipedia.org/wiki/Java_bytecode
- NN (PMST GmbH): *(Java) Byte-code-Versionen* ; 2008,
  www.javahowto.de/systeme/byte-code-versionen.html
- NN (Wikipedia): *(.Net) Common Language Runtime* ;
  en.wikipedia.org/wiki/Common_Language_Runtime bzw.
  en.wikipedia.org/wiki/Common_Language_Infrastructure

Fortsetzung

Smalltalk –Weiterentwicklung III: *"Byte code"*/**"Just-in-time compilation"**

- NN (Wikipedia): *Just-in-time compilation* ; `en.wikipedia.org/wiki/Just-in-time_compilation`
- L.P. Deutsch, A.M. Schiffman: *Efficient implementation of the Smalltalk-80 system* ; ACM POPL 11 (1984) 297-302
- J. Aycock: *A brief history of Just-In-time* : ACM Computing Surveys 35 (2003) 2, 97-113
- O. Agesen: *Design and implementation of Pep, a Java Just-in-time translator* ; TAPOS – Theory & Practice Object Systems 3 (1997) 3, 127-155 (siehe `research.sun.com/java-topics/pubs/97-pep.ps`)

Smalltalk –Weiterentwicklung III: *Self, Perl*

- D. Ungar, R.B. Smith: *Self - The power of simplicity* ; ACM SIGPLAN Notices 22 (1987) 12, 227-241
- R.B.Smith, D.Ungar: *Programming as an experience – the inspiration for Self* ; Proc ECOOP'95, 1995, 303-330
- C. Chambers et al.: *An efficient implementation of Self, a dynamically-typed object-oriented language based on prototypes* ; ACM SIGPLAN Notices 24 (1989) 10, 49-70
- O. Agesen et al.: *Type inference of Self: Analysis of objects with dynamic and multiple inheritance* ; Proc. ECOOP'93, 1993, 247-267
- O. Agesen et al.: *The Self Programmers Reference Manual* ; 2000 (wohl nicht mehr von SUN unterstützt und dort zu finden? Daher Google-Suche über >self sun „reference manual"<), **research.sun.com/self/language.html**

Fortsetzung

HJH#08b
2008/2009

Smalltalk –Weiterentwicklung IV: *Self, Perl*

- R.L. Schwartz et al.: *Einführung in Perl* ; O'Reilly, 2005
- T. Fahle: *Perl – Stop saying „script", say „dynamic language"* ; 2007, thomas-fahle.blogspot.com/2007/09/perl-stop-saying-script-say-dynamic.html
- M. Lubkowitz: *Perl – dynamisch und interaktiv* ; Galileo Computing, Teil 5, 2005, 475 ff.
- NN (DseWiki): *Sprache Perl* ; 2007, www.wikiservice.at/dse/wiki.cgi?action=browse&id=SprachePerl&oldid=Perl
- B.D. Foy: *Mastering Perl – Dynamic subroutines*; 2008, www252.pair.com/comdog/mastering_perl/Chapters/09.dynamic_subroutines.html
- A. Tang: *Perl 6 – reconciling the irreconcilable* ; ACM SIGPlan Notices 42 (2007) 1, 1-1 (knapp, eingeladener Vortrag, nur Zusammenfassung)
- empfohlen: Google-Suche >Perl<
- L. Titchkosky et al.: *A performance comparison of dynamic Web technologies* ; ACM SIGMETRICS Performance Evaluation Review 31 (2003) 3, 2 - 11

Lisp - "*Closures*" | "*Continuations*"

- NN (Wikipedia): *Common Lisp Object System* ; **en.wikipedia.org/wiki/Common_Lisp_Object_System**
- G. Kiczales et al.: *The art of the Metaobject protocol* ; MIT Press, 1991
- A. Paepcke (ed.): *Object-oriented programming – The CLOS perspective* ; MIT Press, 1993

Fortsetzung

- C.T. Haynes: *Scheme standardization* ; ACM SIGPLAN Lisp Pointers, III (1990), Issue 2-4
- A.K. Wright, R. Cartwright: *A practical soft type system for Scheme* ; ACM Trans. Programming Languages & Systems, 19 (1997) 1, 87-152
- S. Tobin-Hochstadt, M Felleisen: *The design and implementation of typed Scheme* ; ACM SIGPLAN Notices 43 (2008) 1, ??-??

Lisp - *"Closures" / "Continuations"*

- NN (Wikipedia): *Closure (computer science)* ; `en.wikipedia.org/wiki/Closure_(computer_science)`
- M. Fowler: Closure ; 2004, `martinfowler.com/bliki/closure.html`
- J. Walnes: *The power of closures in C# 2.0 (Zusatz zu Fowler-Artikel)* ; 2004, `joe.truemesh.com/blog//000390.html`
- S. Roock: *Closures in Common Lisp (Zusatz zu Fowler-Artikel)* ; 2006, `stefanroock.blogspot.com/2006/08/closures-in-common-lisp.html`
- N. Gafter: *A definition of closures* ; 2007, `gafter.blogspot.com/2007/01/definition-of-closures.html`

Fortsetzung

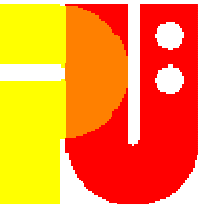
Lisp - *"Closures" / "**Continuations**"*

- NN (Wikipedia): *Continuation* ;
  en.wikipedia.org/wiki/Continuation
- H. Thielecke: *Continuations, functions, and jumps* ; ACM SIGACT News 30 (1999) 2, 33-42
- P. Goodwin: *Continuation explanation* ; 2005,
  c2.com/cgi/wiki?ContinuationExplanation
- E. Kiser: *Call with current continuation* ; 2008,
  c2.com/cgi/wiki?CallWithCurrentContinuation
- C. Barker: *Continuations in natural languages* ; 4th ACM-SIGPLAN Continuation Workshop, 2004,
  www.cs.bham.ac.uk/~hxt/cw04/barker.pdf

Dynamik bei *Python*

- NN (Wikipedia): *Python (programming language)* ; 2008, `en.wikipedia.org/wiki/Python_(programming_language)`
- NN (DseWiki): *Sprache Python* ; 2005, `www.wikiservice.at/dse/wiki.cgi?action=browse&id=SprachePython&oldid=Python`
- F. Lundh: *Compiling Python code* ; 2003, `effbot.org/zone/python-compile.htm`
- I. DeJong (?): *Python compared to Java* ; 2008, `www.razorvine.net/python/PythonComparedToJava`
- S. Schwarzer: *Perl und Python – zwei Skriptsprachen im Vergleich* ; 2000, `www.sschwarzer.net/python/perlpy_vortrag.html`
- S. Karabuk, F.H. Grant: *A common medium for programming operations research models* ; IEEE Software 24 (2007) 5, 39-47

Dynamik bei *Ruby*



- NN (Wikipedia): *Ruby* ; de.wikipedia.org/wiki/Ruby_(Programmiersprache)
- NN (DseWiki): *Sprache Ruby* ; 2007, www.wikiservice.at/dse/wiki.cgi?SpracheRuby
- J. Fox: *Ruby for the Java world* ; 2006, www.javaworld.com/javaworld/jw-07-2006/jw-0717-ruby.html
- K.C. Baird: *Ruby by example – Concepts and code* ; No Starch Press, 2007, www.linuxsecurity.com/content/view/128840/171/
- R.A. Olsen: *Design patterns in Ruby* ; Addison-Wesley Longman, 2008
-  NN (Wikipedia): *Ruby on Rails* ; de.wikipedia.org/wiki/Ruby_on_Rails

Dynamik bei *Groovy*

- NN (Wikipedia): *Groovy* ; de.wikipedia.org/wiki/Groovy
- K. Henry: *A crash overview of Groovy* ; ACM Student Magazine Crossroads 12 (2006) 3, 5-5
- NN (Codehouse Foundation): *Groovy - An agile dynamic language for the Java platform* ; 2008, groovy.codehaus.org/
- M. Tilly: *Die Skriptsprache Groovy* ; JavaSpektrum ?? (2004) 5, 44-48 (siehe www.sigs.de/publications/js/2004/05/tilly_JS_05_04.pdf)
- T. Kamann: *Groovy* ; 2006, thorque.wikispaces.com/space/showimage/groovy_rocks_or_not.pdf
- C. Vollrath; *Java wird Groovy* ; 2006, events.ccc.de/congress/2006/Fahrplan/attachments/1193-JavaWirdGroovy_final.pdf

Was bietet *Lua*

- NN (Wikipedia): *Lua* ; de.wikipedia.org/wiki/Lua
- R. Ierusalimschy: *Programming in Lua* ; 2003, www.lua.org/pil/
- J. Quigley: *A look at Lua* ; Linux Journal, 2007, issue 158
- A. Hirschi: *Traveling light, the Lua way* ; IEEE , Software 24 (2007) 5, 31 - 38
- R. Ierusalimschy et al.: *Lua - an extensible extension language*; Software - Practice & Experience 26 (1996) 6, 635 – 652
- R. Ierusalimschy et al.:: *The implementation of Lua* ; Journal of Universal Computer Science 11 (2005) 7, 1159 – 1176
- N. Gammon: *Writing dynamic web pages using Lua* ; *2006,* www.gammon.com.au/forum/bbshowpost.php?bbsubject_id=6498

HJH#14
2008/2009

Parallelverarbeitung: *Erlang*

- NN (Wikipedia): *Erlang (programming language)* ; en.wikipedia.org/wiki/Erlang_(programming_language)
- J. Armstrong, Schöpfer von *Erlang*, siehe seine Startseite www.sics.se/~joe/, undatiert, insbesondere Menü-Punkt „Publications"
- J. Armstrong: *Functions + Messages + Concurrency = Erlang* ; Vortragsfolien und Video, *Erlang* eXchange 2008, skillsmatter.com/podcast/erlang/erlang-concurrent-applications-in-a-concurrent-world bzw. www.erlang-exchange.com/conference
- V. Barr, C. Böckmann : *Moderne Programmierung nebenläufiger und verteilter Anwendungen mit Erlang* ; 2006, www.traveling-light.de/Erlang_Papier-v2.2.pdf
- S.-O. Nyström: *A soft-typing system for Erlang* ; Proc. ACM SIGPLAN Workshop on *Erlang*, 2003
- J. Armstrong: *A history of Erlang* ; Proc. 3rd ACM SIGPLAN Conf. History Programming Languages, 2007, 6-1 – 6-26

Entwicklungsaufwand und Ausführungszeit

- (verschiedene „Blogger"): *Diskussion über dynamische Sprachen und Java - "the fud continues„* ; 2007/2008 (Einstieg über pab-data.blogspot.com/2008/07/dynamic-languages-fud-continues.html)
- P. Beust: *A guide to modern languages and interesting concepts for the busy Java programmer* ; 2008, jazoon.com/jazoon08/en/conference/presentationdetails.html?type=sid&detail=5249
- NN (Cincom): *Cincom Smalltalk und Seaside gewinnen vor Ruby den Dynamic Language Shootout auf der OOP 2008* ; 2008, www.softguide.de/presse/pm/104.htm
- R.W. Sebesta: *Support for Object-Oriented Programming* ; ftp://ftp.aw.com/cseng/authors/sebesta/concepts8e/. Teil 12 aus www.aw-bc.com/sebesta/, 2008, Addison-Wesley
- J. Sutherland: *Smalltalk, C++, and OO COBOL - The Good, the Bad, and the Ugly* ; 1995 (also etwas überholt !), jeffsutherland.com/papers/oocobol.html

Entwicklungsaufwand und Ausführungszeit

http://www.goodstart.com/whousessmalltalk.php

http://www.smalltalkindustrycouncil.org/companies/companies.htm

http://www.cincom.com/pdf/CS040819-1G-A4.pdf

“Allgemeine Geschäftsbedingungen”

- Referat maximal 40 Minuten
- Datenprojektor („*beamer*") ist vorhanden
- Anschließend Diskussion
  - inhaltlich
  - Vortragsstil
- Auf Anforderung, mindestens fünf Tage vorher, kann PC für Präsentation mit PowerPoint bereitgestellt werden
- Ausarbeitung maximal acht bis zehn Seiten DIN A4;
  (in der Regel) nach einer Woche abzugeben (Papier);
  Literaturangaben nicht vergessen!
- Nach Bestätigung (eventl. Korrekturwünsche) durch Betreuer Abgabe der endgültigen Ausarbeitung innerhalb einer Woche als doc-/pdf-Datei
  (maximal 2 Mbyte groß, mit Namen “WS08-09-#xx-version-ii.<ext>”, “xx” elem {01 ÷ 15}, “ii” elem {1 ...})

Ein Hinweis des Dekans:

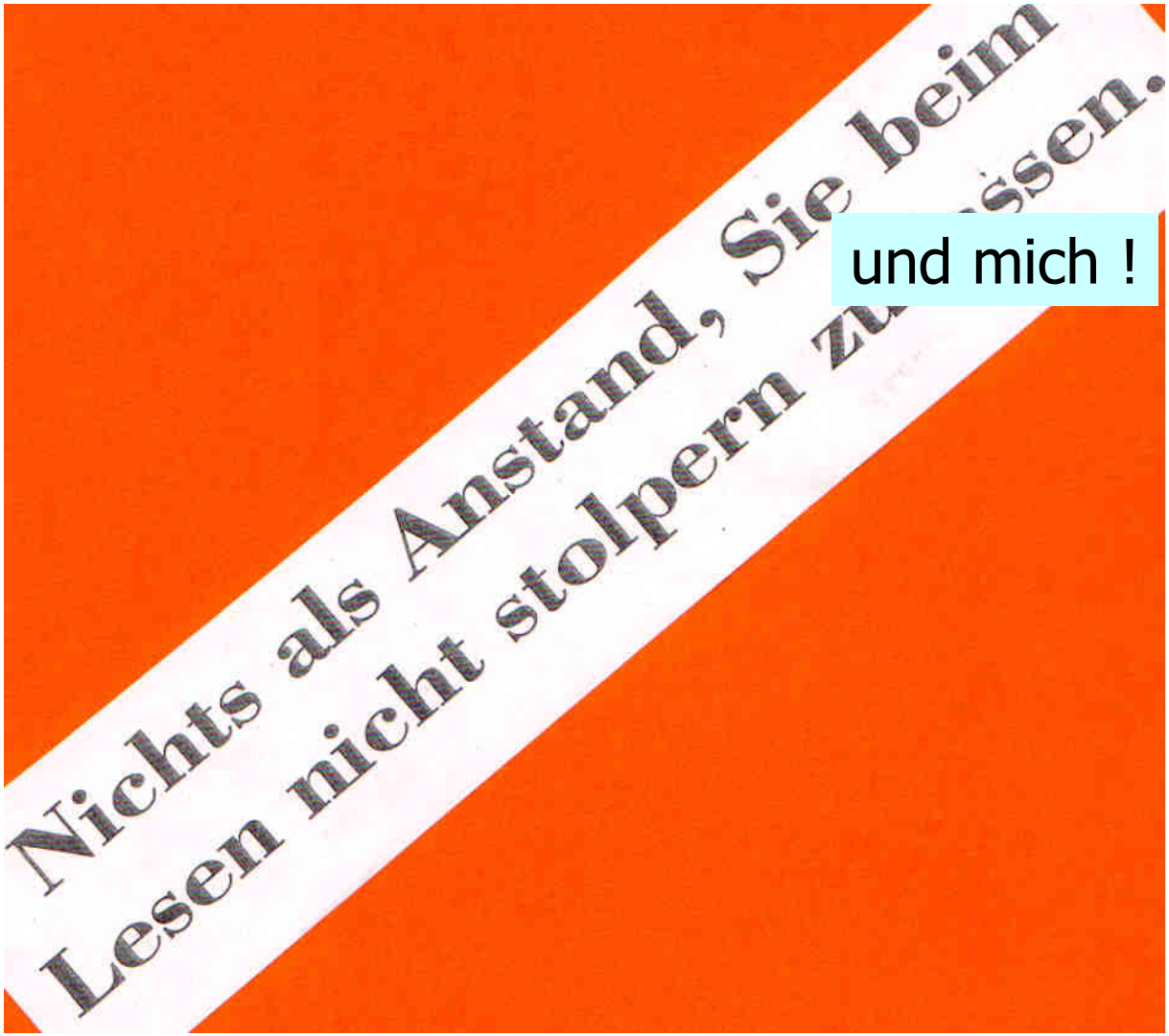
"Der Fachbereich Informatik mißt der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Zu diesen gehört auch das strikte Verfolgen von Plagiarismus. Mit der Abgabe einer Lösung (Hausaufgabe, Programmierprojekt, Seminararbeit Diplomarbeit, etc.) bestätigen Sie, daß (Sie/Ihre Gruppe) der alleinige Autor/die alleinigen Autoren des gesamten Materials sind. Falls Ihnen die Verwendung von Fremdmaterial gestattet war, so müssen Sie dessen Quellen deutlich zitiert haben.

Bei Unklarheiten zu diesem Thema finden Sie weiterführende Informationen unter **`www.informatik.tu-darmstadt.de/Plagiarism`** oder sprechen Sie Ihren Betreuer an."

Web-Seite des Seminars:

http://www.pu.informatik.tu-darmstadt.de/Seminar-Dynamik/

(dort findet sich u.a. die Vorlage für das
Deckblatt Ihrer Ausarbeitung
und der **Terminplan** !)


Nichts als Anstand, Sie beim
Lesen nicht stolpern
und mich !

Fortsetzung

HJH#??
2008/2009

??