

Objektorientierung in Reinkultur

- Was uns Smalltalk gelehrt hat und weiterhin lehrt -

WS 2010/2011



Themen, betreut von
Univ.-Prof. em. Dr. H.-J. Hoffmann

(Stand 9. November 2010)

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 August – Oktober 2010 !

Irrtum und Tippfehler vorbehalten !

Zur kürzeren Schreibweise von URLs

ist „http://“ bzw. „https://“ immer weggelassen!

Alle Wikipedia-Referenzen sind von 2010 !

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:

- Wikipedia: [de.wikipedia.org/wiki/Smalltalk-80_\(Programmiersprache\)](http://de.wikipedia.org/wiki/Smalltalk-80_(Programmiersprache))
- J. Tuchel: *Lebendiges Objekt - Smalltalk, ein aktueller Klassiker* ;
c't, 2003, Heft 3, 188 ff (nicht mehr ganz aktuell bzgl. des Umfelds), auch zu finden unter
[www2.informatik.hu-berlin.de/swt/lehre/PR_MTI_0910/
restricted/literature/smalltalk/smalltalk_1.htm](http://www2.informatik.hu-berlin.de/swt/lehre/PR_MTI_0910/restricted/literature/smalltalk/smalltalk_1.htm)
- G. Heeg: www.heeg.de/smalltalk/smalltalk.html, um 2006,
- F. Dominicus: *Nachrichten-Zustellung* ; Linux-Magazin, 2004,
[www.linux-magazin.de/Heft-Abo/Ausgaben/2004/04/
Nachrichten-Zustellung](http://www.linux-magazin.de/Heft-Abo/Ausgaben/2004/04/Nachrichten-Zustellung) (ich bevorzuge „**Botschaft**“ statt „**Nachricht**“)
- P. Kreußel et al.: *PHP, Java, Perl, Python Und Java FX im Wettstreit - plus Lisp und Smalltalk, Sprachwahl* ; Linux-Magazin, Online-Artikel, 2008,
www.linux-magazin.de/Online-Artikel/Sprachwahl?category=0
- 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
- European Smalltalk User Group (ESUG): [www.esug.org/wiki/pier/
Smalltalk?_s=_RfQwsthZdtVI2D&_k=bmkTons2K-X5c8NS&_n&17](http://www.esug.org/wiki/pier/Smalltalk?_s=_RfQwsthZdtVI2D&_k=bmkTons2K-X5c8NS&_n&17), 2010

!!!

Objektorientierung in Reinkultur
Wintersemester 2010/2011

Inhaltliche Vorbereitung für alle (Forts.):

----- nicht so ernst gemeint -----

Für absolute **Neulinge**, von L. Hellmig: [di.informatik.uni-rostock.de/
fileadmin/didaktik/files/Fortbildungen/
Objektorientierung_im_Informatikunterricht_der_OS.pdf](http://di.informatik.uni-rostock.de/fileadmin/didaktik/files/Fortbildungen/Objektorientierung_im_Informatikunterricht_der_OS.pdf), 2010

Und zur **Abschreckung** (mit vielen Schreibfehlern): [www.gutefrage.net/frage/
welchen-sinn-machen-klassen-in-der-programmierung](http://www.gutefrage.net/frage/welchen-sinn-machen-klassen-in-der-programmierung), 2010

----- und noch etwas „Lokalkolorit“ -----

- ∇ L. Heuser: *Heinz´ Life, 1962 – 2032, Kleine Geschichten vom Kommen und Gehen des Computers, auch aus der damaligen THD und jetzigen TUD*; Hanser-Verlag, 2010 (Seiten 99 – 102 über Smalltalk **1983 an der THD**)

Übersicht



- Objekte und Objektorientierung (OO)
 - # 01 Objekte der realen Welt
 - # 02 Objekte und OO in Programmen
 - # 06 *Typisierung* ?
 - # 07 einfaches vs. mehrfaches *Erben*
- Smalltalk und Vorläufer / Umfeld
 - # 03 Smalltalk-80, Allgemein + Syntax
 - # 04 Smalltalk-80, Klassengebäude beispielhaft
 - # 05 Smalltalk, heutiger Stand

 - # 11 Simula-67 *
 - # 12 Eiffel / Objective C *
- OO in anderen Beispielen
 - # 08 Einfluß in C++ / Java / C#
 - # 10 Einfluß in sog. Skriptsprachen, Ruby, JavaScript u.ä.
- Implementierungsfragen
 - # 09 Webanwendung (z.B. Seaside)

 - # 13 Bytecode *
 - # 14 Speicherbereinigung *
 - # 15 Entwicklungsaufwand und Ausführungszeit *

Mindestteilnehmerzahl 10. Es werden **maximal 15 Themen** ausgegeben !

Themen #01 bis #10 werden zuerst in dieser Reihenfolge vergeben !

Die grauen, mit * markierten Themen stehen am Ende zur Auswahl,

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

Es geht darum herauszuarbeiten, wie weit
„Objektorientierung in Reinkultur“
(nicht) zu finden ist,
wie damit umgegangen werden kann,
was einem Programmierer zu empfehlen ist
um diese zu erreichen,
Vergleich mit der Lösung von Smalltalk u.ä.



Was sind Objekte der realen Welt?

Identität, Zustand und Verhalten eines Objekts. Noch mehr?

In welcher Beziehung zueinander können Objekte stehen?

Objekte können sich verändern? Wie?

Zusammenfassen von Objekten

Denkanstöße: Semantische Netze der KI / „*Robotics*“

Einige Literaturanregungen, weitere zu finden:

- Wikipedia: *Reality*; en.wikipedia.org/wiki/Reality
- R. Joehanes: *Object oriented programming philosophy*; 2000,
www.baskent.edu.tr/~tkaracay/etudio/ders/prg/pascal/PasHTML3/pas/pas13001.html • hat gefehlt
- I. Schmiedecke: *Modellierung von Klassen- und Objektbeziehungen*; undat.,
www.schmiedecke.info/SE1/folien_archiv/SE1-7-Klassenbeziehungen.pdf
(ich bevorzuge „**Botschaften**“ statt „**Nachrichten (messages)**“, **Folie 2 des Artikels**)
- Wikipedia: *Semantisches Netz*; de.wikipedia.org/wiki/Semantisches_Netz



Einige Literaturanregungen, weitere zu finden:

- D.G. Aliaga: *Virtual objects in the real world* ; Comm. ACM 40 (1997) 3, 49-54
- G. Zabala et al.: *Physical Etoys* ; 2010 (1. Preisträger bei diesjähriger ESUG-Konferenz in Barcelona),
tecnodacta.com.ar/gira/projects/physical-etoys/
(siehe auch – zum Teil in Spanisch – :
picasaweb.google.com/105521672761794492373/PhysicalEtoys?feat=flashalbum#)



Objekte und Objektorientierung in Programmen



Was sind Objekte, die in einem Programm angesprochen werden können?

Was sichert Identität, Zustand, Verhalten?

Wann im Programmablauf und wie werden sie angesprochen?

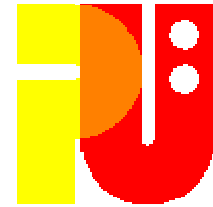
Wie manifestiert sich ihr „Zustand“?

In welcher Beziehung zueinander können Sie stehen?

Zusammenfassen von Objekten

Einige Literaturanregungen, weitere zu finden:

- Wikipedia: *Object-oriented programming* ;
en.wikipedia.org/wiki/Object-oriented_programming
- Wikipedia: *Variable (Programmierung)* ;
[de.wikipedia.org/wiki/Variable_\(Programmierung\)](http://de.wikipedia.org/wiki/Variable_(Programmierung))
- H.W. Lang: *Variablen und Ausdrücke* ; undatiert,
www.inf.fh-flensburg.de/lang/prog/variablen.htm
- C. Gaßner: *OOP – die objektorientierte Programmierung (C++)* ;
2009, stubber.math-inf.uni-greifswald.de/logik/gassner/lehre/OOP/OOP2009.pdf

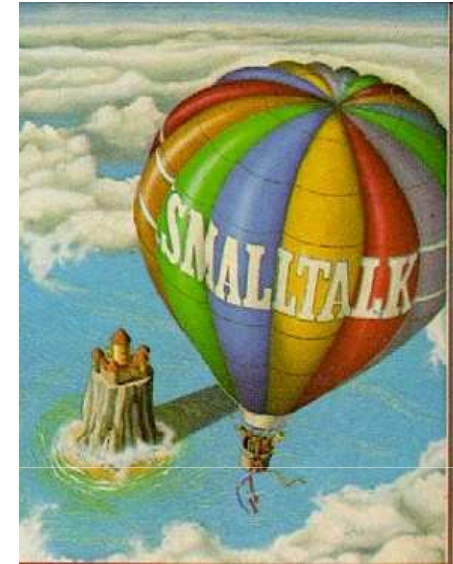


Entwicklungsgeschichte

Herausragende Eigenschaften

Syntax, Ablaufkonstrukte, Blöcke; Beispiel

Stöberer (engl. *Browser*), Tester (engl. *Debugger*),
Sitzungsverlauf, Datenhaltung



Literaturanregungen:

- NN (Wikipedia): *Smalltalk*; en.wikipedia.org/wiki/Smalltalk_programming_language
- A.C. Kay: *The early history of Smalltalk*; ACM SIGPlan Notices, 28 (1993) 3, 69 - 95
- D. Thomas: *From research PARC to main street – An historical perspective*; 2005, www.mojowire.com/TravelsWithSmalltalk/DaveThomas-TravelsWithSmalltalk.htm
- A. Goldberg, D. Robson: *Smalltalk-80–The language*; Addison Wesley, 1989 (hieraus Blätter #04k1 und #04k2 [*] entnommen)

Fortsetzung siehe www.pu.informatik.tu-darmstadt.de/Seminar-Smalltalk/Smalltalk-Anregungen.pdf,
Folien #03b ff.

Smalltalk-80, Klassengebäude beispielhaft



Beschreibung von Objekten und Zugriff darauf
Entstehen und Vergehen von Objekten, Lebensdauer
Metaobjekte
Klassen- und Metaklassenhierarchie
Beispielhaft Magnitude, Collection und Class / Metaclass
Entwurfsmuster, z.B. MVC

Literaturanregungen:

- R. Lutze: *Die Implementierung von Smalltalk*. In: H.-J. Hoffmann: *Smalltalk verstehen und anwenden*; 1987, Carl Hanser Verlag, 129-187 (hieraus Blätter #05s1 bis #05s3 [******] entnommen)
- L. Richter: (*verschiedenes*); www.ifi.uzh.ch/richter/Classes/oose2/
- Wikipedia: *Metaclass*; en.wikipedia.org/wiki/Metaclass

Fortsetzung siehe www.pu.informatik.tu-darmstadt.de/Seminar-Smalltalk/Smalltalk-Anregungen.pdf, Folien #04b ff.

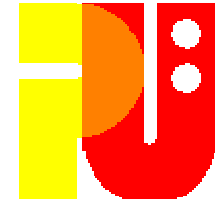
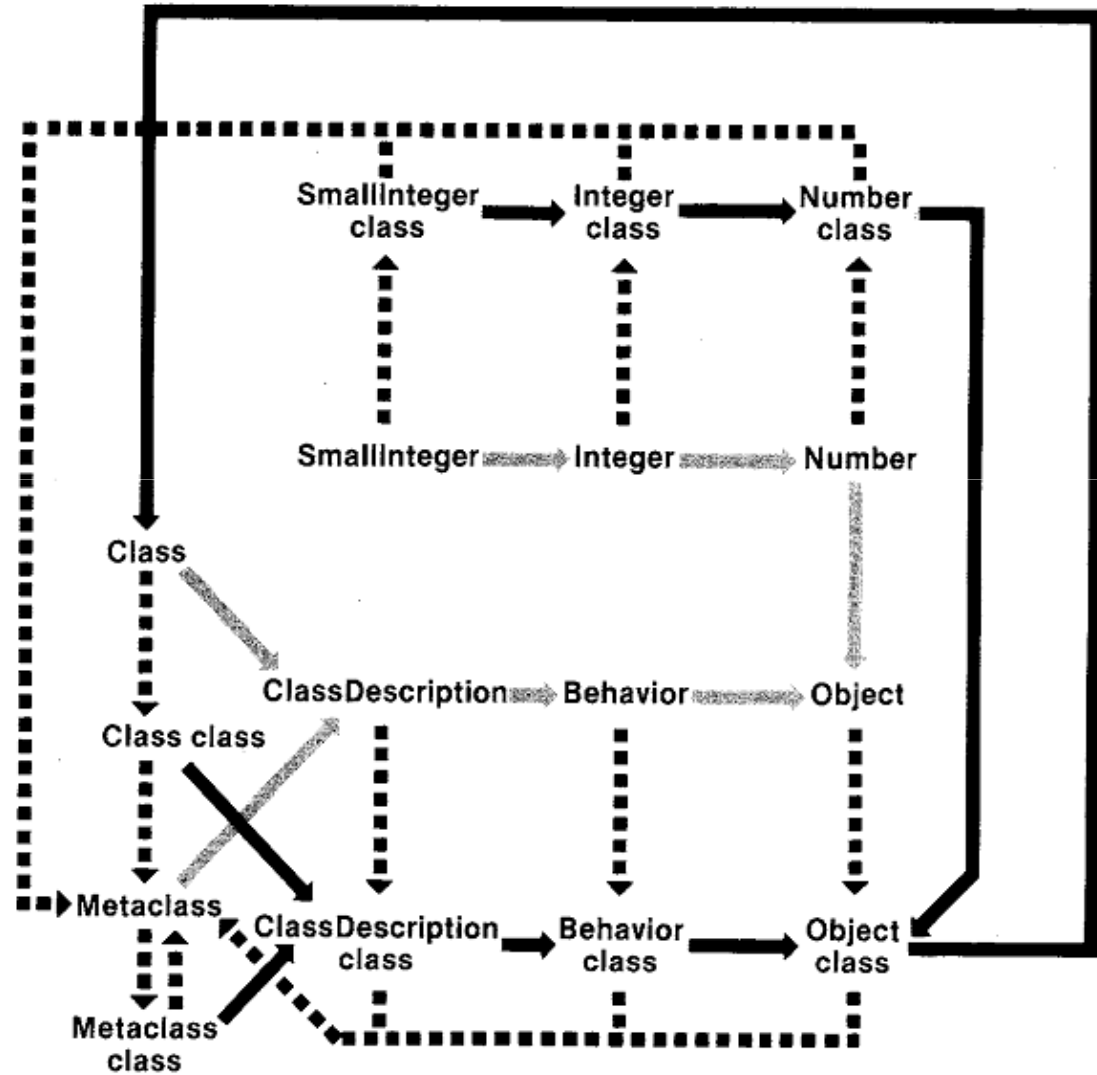
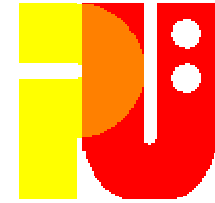


Figure 16.5 aus Folie #03a
[* , p 272]:

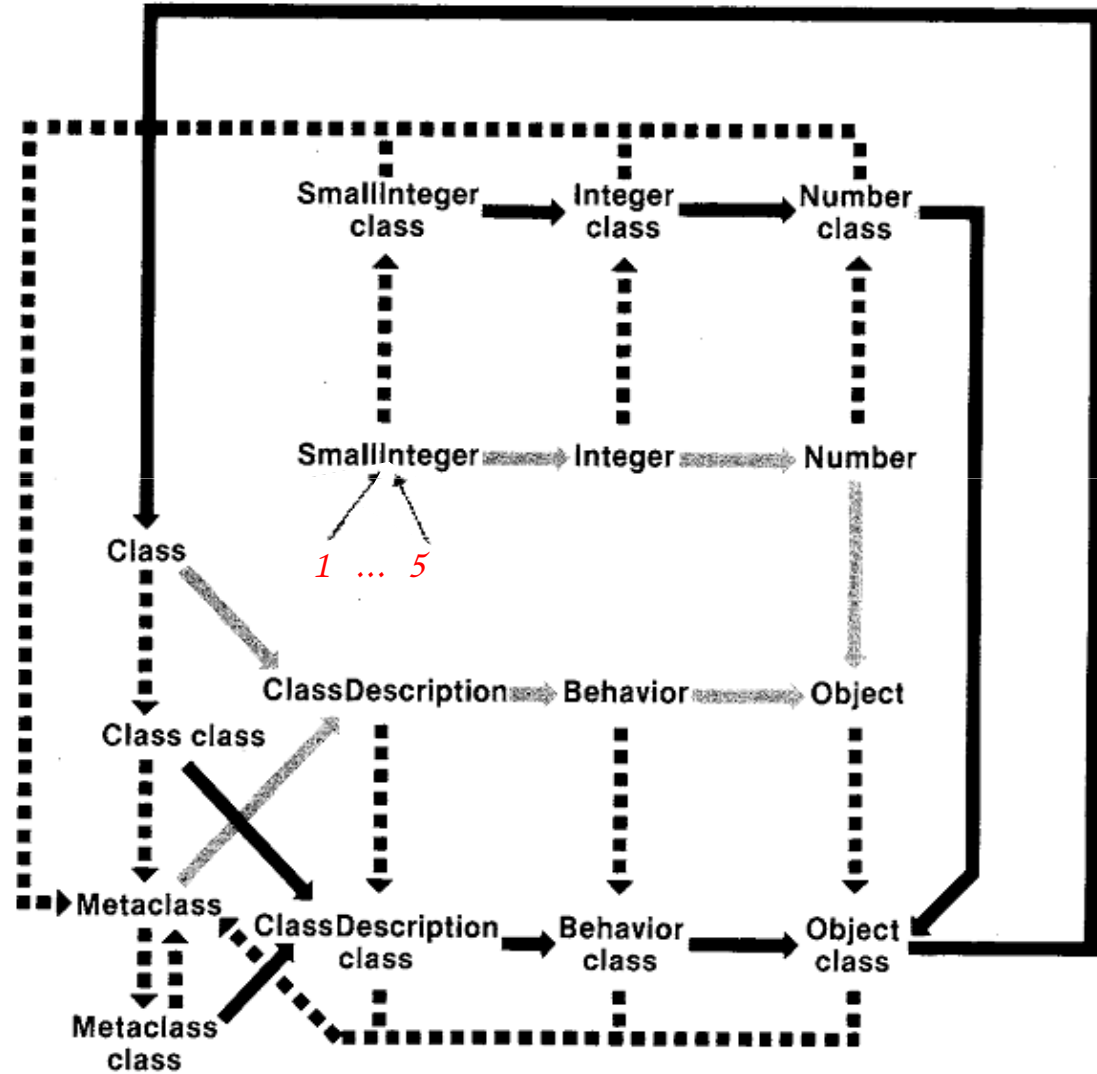
Metaklassen- und Klassen-
hierarchie bei *Smalltalk-80*

- A \longrightarrow B Klasse A ist Unterklasse der (Meta-)Klasse B
- A \dashrightarrow B Klasse A ist Unterklasse der Klasse B
- A \dashrightarrow B A ist Ausprägung der Klasse B (auf diesem Blatt sind A bzw. B Klassenobjekte !)





- 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)



(Wesentliche) Implementierungen

Typische Anwendungen, Beispiele

Smalltalk, eine dynamische Sprache, warum? Sog. *reflection*

Einblick in Klassenhierarchie z.B. von Cincom *VisualWorks*

Einblick in Benutzungsoberfläche von *Squeak*

Zusammenfassung

Fortsetzung

Literaturanregungen:

- L.J. Pinson, R.S. Wiener: *An introduction to object-oriented programming and Smalltalk*; Addison-Wesley, 1988
- T. Korson, J.D. McGregor: *Understanding object-oriented: A unifying paradigm*; Comm. ACM, 33 (1990) 9, 40-60
- T. Korson, J.D. McGregor: *Heuristiken für den objektorientierten Entwurf*; 2010, [/www.iste.uni-stuttgart.de/se/links/links-se/entwurfsregeln-fuer-den-objektorientierten-entwurf/heuristics.html](http://www.iste.uni-stuttgart.de/se/links/links-se/entwurfsregeln-fuer-den-objektorientierten-entwurf/heuristics.html)

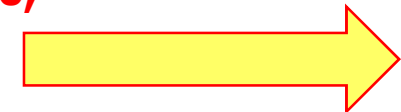
Fortsetzung siehe www.pu.informatik.tu-darmstadt.de/Seminar-Smalltalk/Smalltalk-Anregungen.pdf, Folien #05d ff.



Übung macht den Meister:

Und was kann man dafür bekommen?

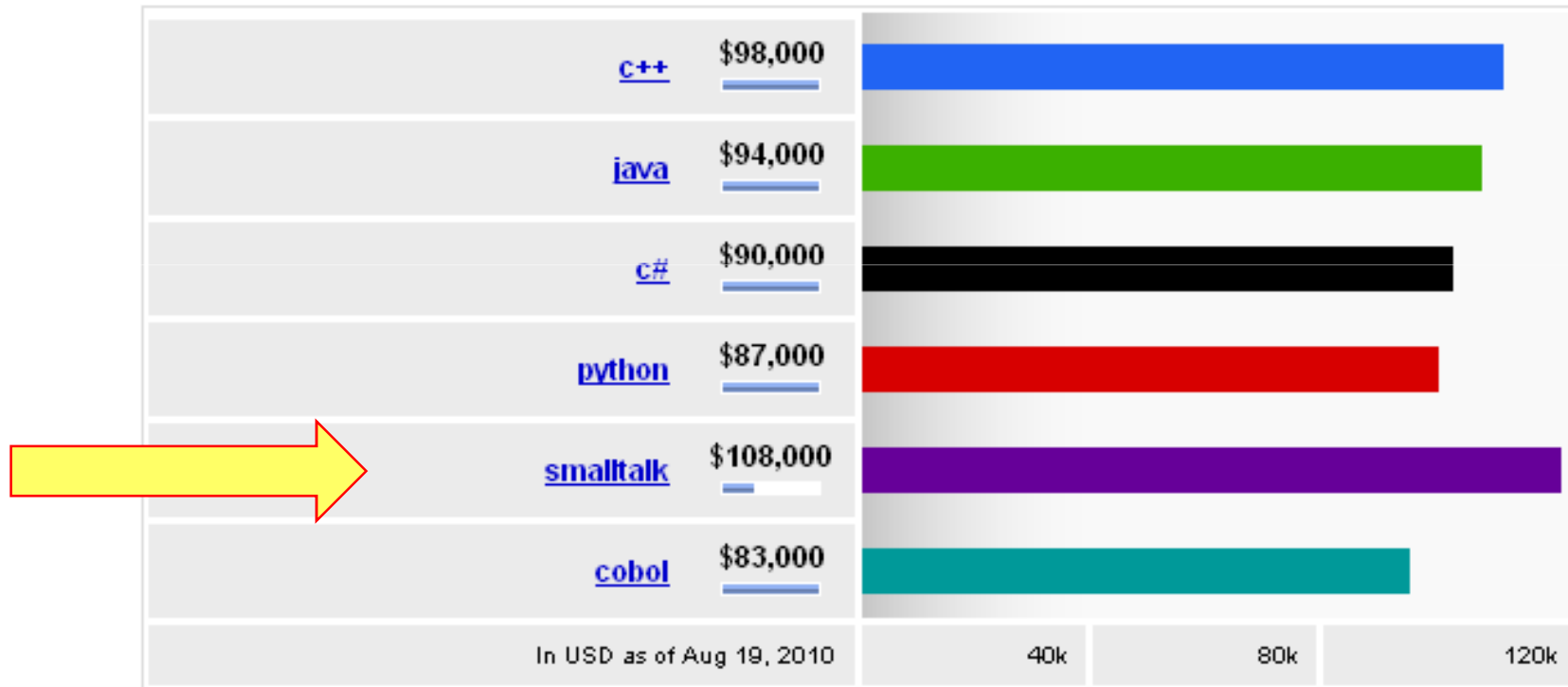
- Cincom: (download VisualWorks – free after registration) ; actual;
www.cincomsmalltalk.com/main/developer-community/non-commercial/
- Cincom: *Basic libraries guide* ; www.cincomsmalltalk.com/documentation/current/BasicLibraries.pdf
- Wikipedia: *IBM Visual Age* ; de.wikipedia.org/wiki/IBM_Visual_Age
- NN: *Instantiations is now 100% Smalltalk Focused* ; 2010,
st.instantiations.com/company/google-transition.html
- Squeak: (*download – free*) ; actual , squeak.org/Download/
- **NN: ..., C++, Java, C#, Python, Ruby, Smalltalk, ... salary; 2010,**
ww.indeed.com/salary?q1=lisp&l1=&q2=c%2B%2B&l2=&q3=java&l3=&q4=c%23&l4=&q5=python&l5=&q6=ruby&l6=&q7=smalltalk%0B&l7=&q8=cobol&l8=&tm=1



Aus: *..., C++, Java, C#, Python, Ruby, Smalltalk, ... salary ; 2010,*
www.indeed.com/salary?q1=lisp&l1=&q2=c%2B%2B&l2=&q3=java&l3=&q4=c%23&l4=&q5=python&l5=&q6=ruby&l6=&q7=smalltalk%0B&l7=&q8=cobol&l8=&tm=1

In den USA:

Average Salary of Jobs with Titles Matching Your Search



Average smalltalk salaries for job postings nationwide are 29% higher than average cobol salaries for job postings nationwide.

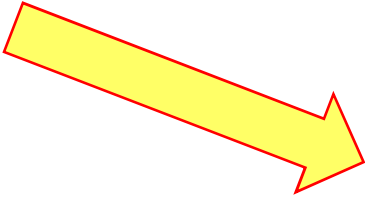
Auch in Deutschland sind „Smalltalker“ gesucht !

Betreff: [Esug-list] **Looking for Smalltalkers**

Von: Georg Heeg
<georg@heeg.de>

An: ESUG Members <esug-list@lists.esug.org>

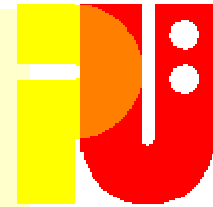
Datum: **19.08.10 12:33:15**



- Dear Smalltalkers,
- We are looking to enlarge our Smalltalk team in Köthen (Anhalt) or Dortmund.
- Georg Heeg
- Georg Heeg eK, Dortmund und Köthen, HR Dortmund A 12812
- Tel. +49-3496-214328, Fax +49-3496-214712

Smalltalk, heutiger Stand III

Implementierung



- **Literaturanregungen (Forts.):**
- NN (Wikipedia): *Bytecode*; en.wikipedia.org/wiki/Bytecode
- Baumann: *Smalltalk, ein Blick in die Vergangenheit*; 2005, www.complang.tu-bs.de/~anton/lvas/sem06w/baumann.pdf (hieraus Blätter 10-12 [***] entnommen)
- NN (Wikipedia): *Java bytecode*; en.wikipedia.org/wiki/Java_bytecode
- 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: en.scientificcommons.org/42723525)

Falls Thema #13 nicht vergeben wird.

Fortsetzung
in Folien #05s1 bis #05s3
und Folien #05t1 und 05t2

Smalltalk, Implementierungsbeispiel I



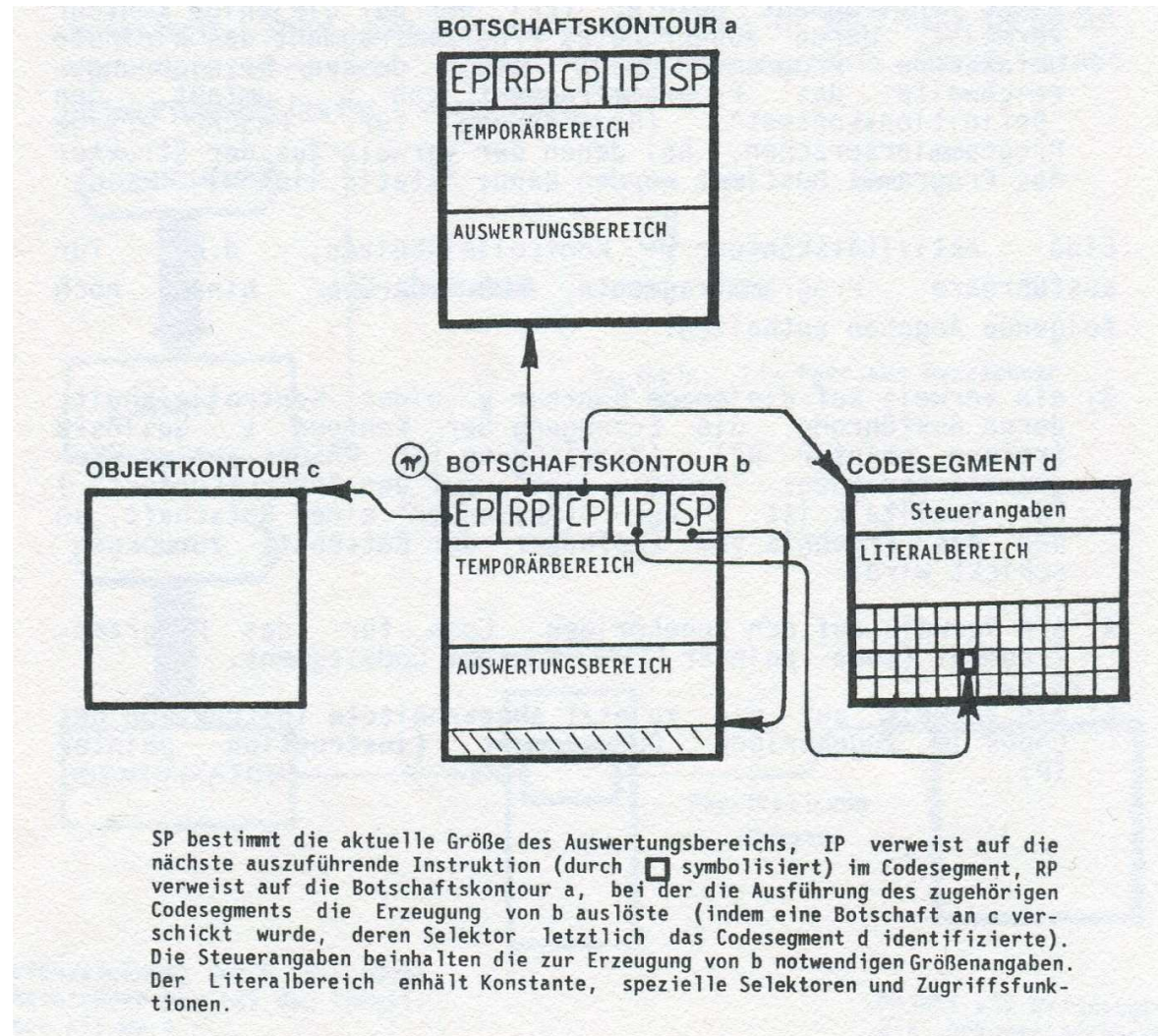
Abb. 7.2 aus

[**, p 134]:

Bearbeitung einer Botschaft gesendet an Objekt *c* bei *Smalltalk-80*.

Beachte die Kellerstruktur zwischen den Kontouren *a* (Sender) und *b* (Empfänger).

In Codesegment *d* befindet sich der Bytecode für die aktivierte Methode.



Fortsetzung

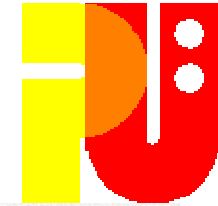


Abb. 7.3 aus
[**, p 136]:

Vorgehen zur sog. Methodenbestimmung beim Zusenden einer Botschaft an (nicht Klassen-) Objekt c (zeitlich vor der Abb. auf vorangehender Folie zu verstehen).

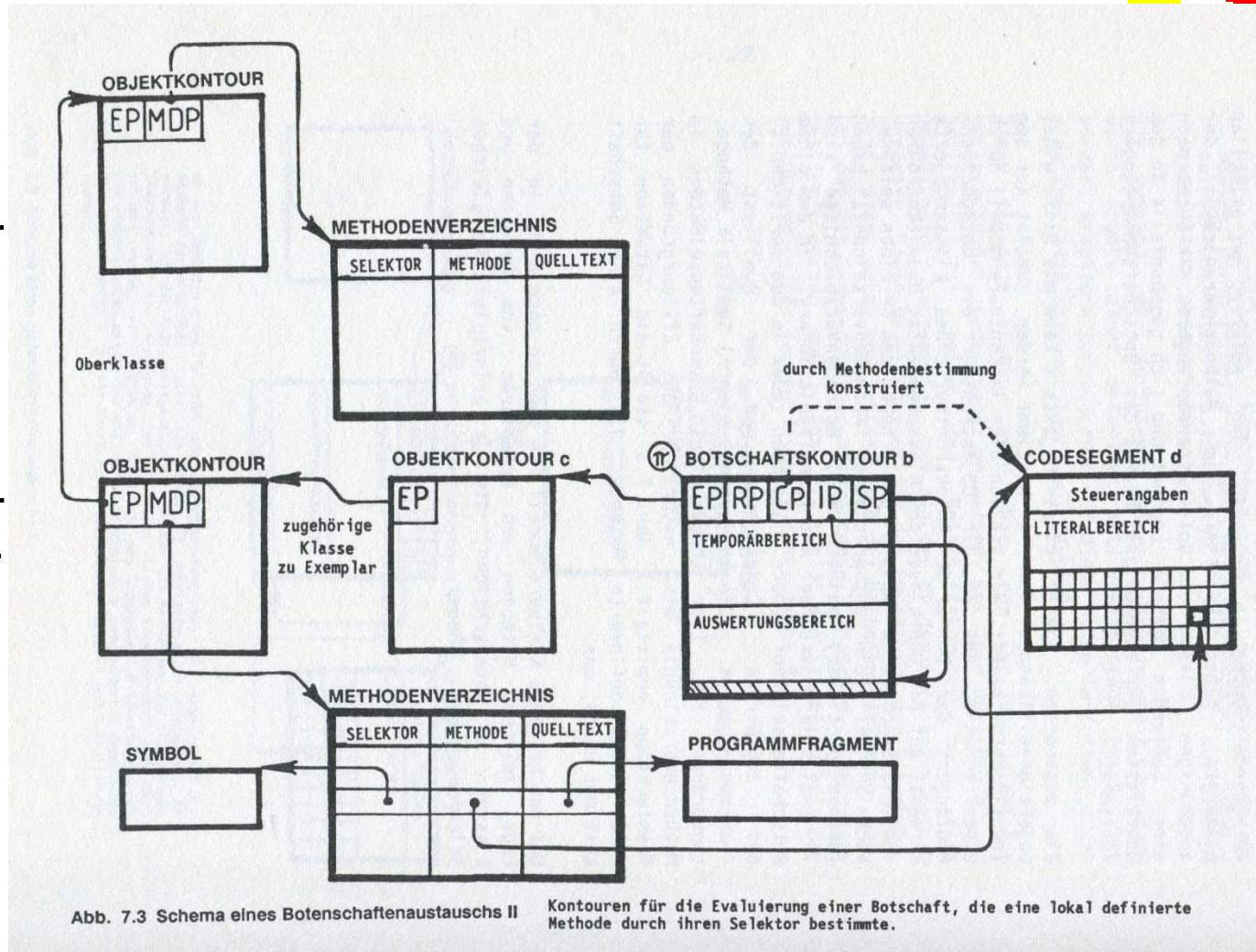


Abb. 7.3 Schema eines Botenschaftenaustauschs II Kontouren für die Evaluierung einer Botschaft, die eine lokal definierte Methode durch ihren Selektor bestimmte.

Fortsetzung

Smalltalk, Implementierungsbeispiel III

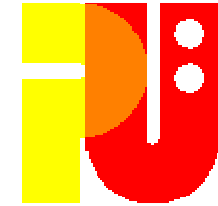
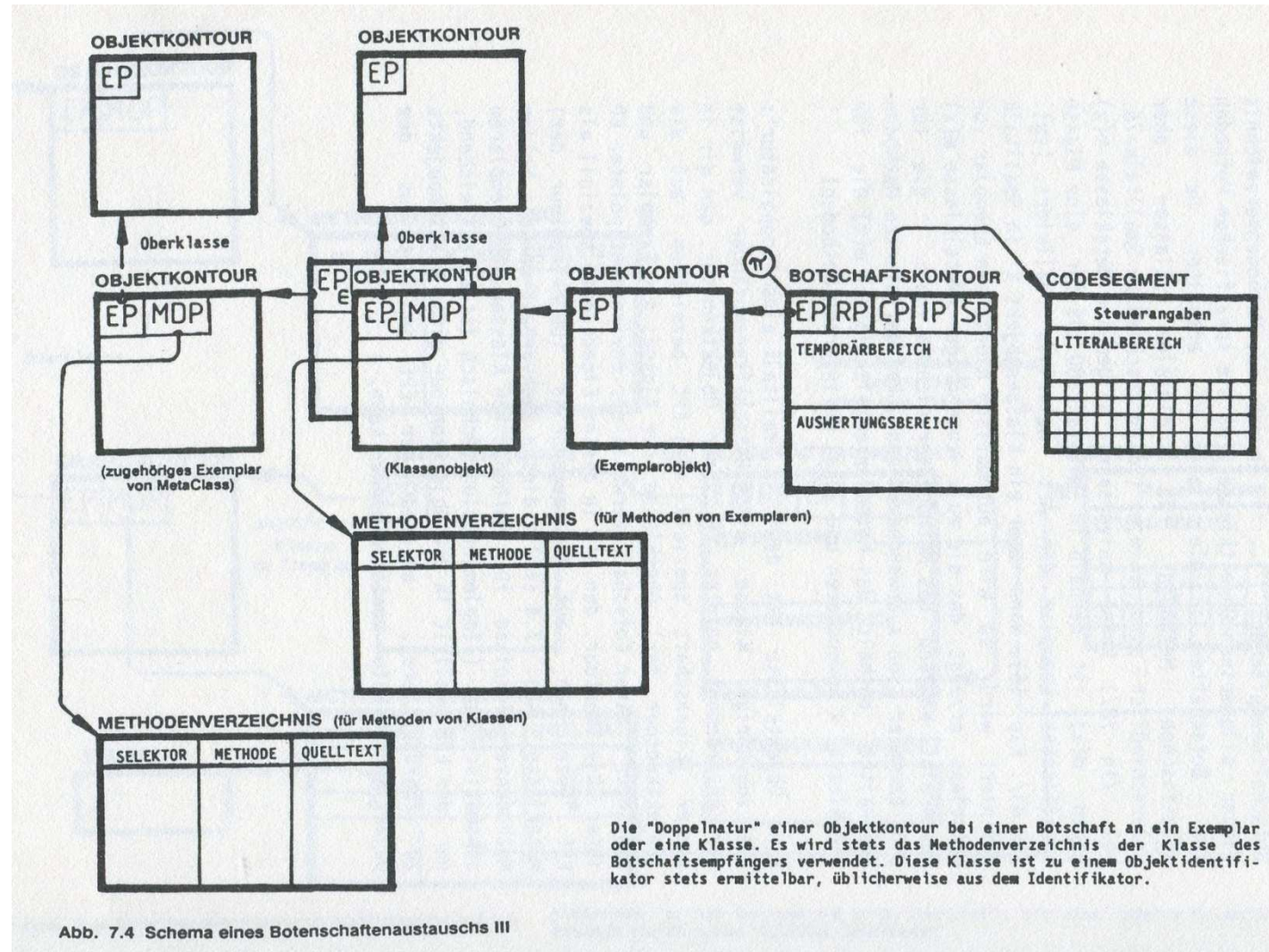


Abb. 7.4 aus [**, p 138]:
Vorgehen zur sog. Methodenbestimmung beim Zusenden einer Botschaft an Klassenobjekt von *c* zu verstehen.



Fortsetzung

Smalltalk, Implementierungsbeispiel IV



Abb. 1 aus
[***, Kap. 1.1]:
Aufbau des Smalltalk-
Objekts „Aktivierung
einer Botschaft“ im
Laufzeitkeller
(bei [**] eine
„Botschaftskontour“)

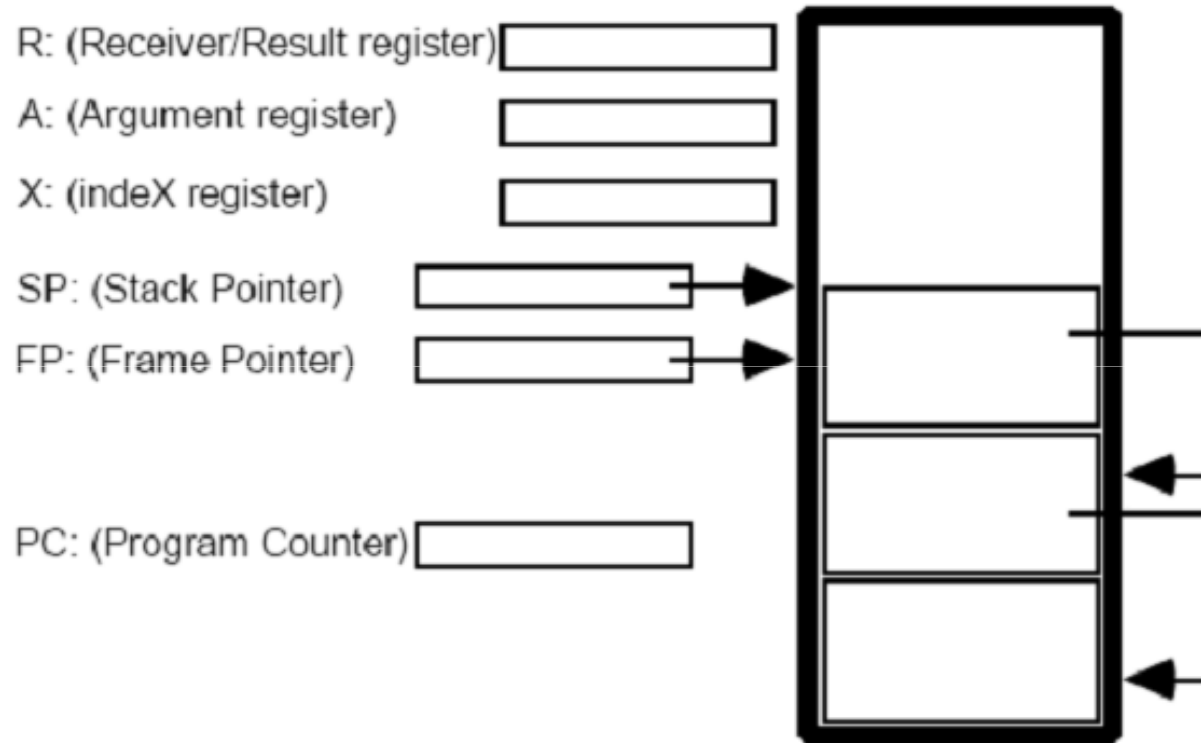


Abbildung 1: Aufbau des Smalltalk-Stacks

Fortsetzung

Smalltalk Implementierungsbeispiel V



Abb. 2 aus
[***, Kap. 1.1]:
Aufbau des Smalltalk-
Objekts „Aktivierung
einer Botschaft“ im
Laufzeitkeller
- Einzelheiten -
(bei [**] einer
„Botschaftskontour“)

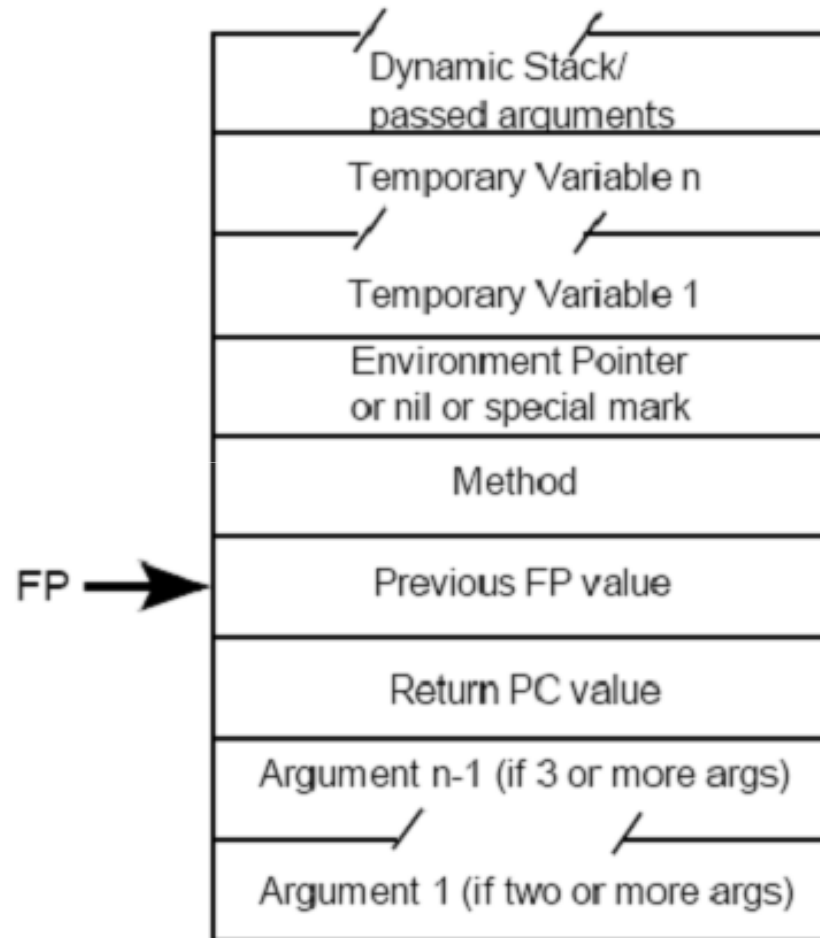


Abbildung 2: Stack-Frame Format



Geht es ohne Typisierung der Objekte? Sog. *Strong typing* ?

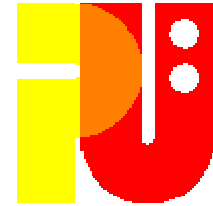
Klassenzugehörigkeit legt Typisierung fest. Typkonflikte ?

Verwenden von Bezeichnern für verschiedene Objekte, Objekte verschiedenen Typs, zu verschiedenen Ausführungszeitpunkten?

Geht es auch anders? Siehe u.a. *Self*.

Literaturanregungen, weitere zu finden:

- Wikipedia: *Type system* ; en.wikipedia.org/wiki/Type_system
- L. Cardelli, P. Wegner: *On understanding types, data abstraction and polymorphism* ; ACM Computing Surveys 17 (1985) 4, 471-523
- P. Grogono, A. Bennett: *Polymorphism and type checking in object-oriented languages* ; ACM SIGPLAN Notices 24 (1989) 11, 109-115
- A.P. Black: *Object-oriented languages: the next generation* ; ACM Computing Surveys, 28 (1996) Special issue 4es, article #149
- D. Buck; *Strongly typed, weakly understood* ; 2004, www.cincomsmalltalk.com/userblogs/buck/blogView?showComments=true&entry=3252458583
- Wikipedia: *Self*; [de.wikipedia.org/wiki/Self_\(Programmiersprache\)](http://de.wikipedia.org/wiki/Self_(Programmiersprache))



Erben (engl. *to inherit*) -- nicht *Vererben* (engl. *to leave*)!

Siehe z.B. dict.tu-chemnitz.de/deutsch-englisch/erben.html bzw.
[.../vererben.html](http://dict.tu-chemnitz.de/deutsch-englisch/vererben.html)

Ein Erblasser, viele Erblasser

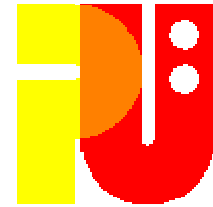
Konflikt bei mehreren Erblinien

Umfang eines Erbes?

Andere Lösungen?

Literaturanregungen, weitere zu finden:

- Wikipedia: *Vererbung (Programmierung)*; - **Begriff falsch! Siehe oben!**
[de.wikipedia.org/wiki/Vererbung_\(Programmierung\)](http://de.wikipedia.org/wiki/Vererbung_(Programmierung))
- Wikipedia: *Multiple inheritance*;
en.wikipedia.org/wiki/Multiple_inheritance
- G.B. Singh: *Single versus multiple inheritance in object oriented programming*; ACM SIGPLAN OOPS Messenger 5(1994) 1, 34-43



Was hätte man von *Smalltalk* lernen können und sollen?

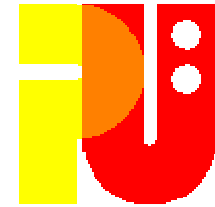
Was fehlt, was gibt es mehr?

Probleme beim Einsatz von *C++*, *Java* bzw. *C#* aus diesem Sichtwinkel?

Literaturanregungen:

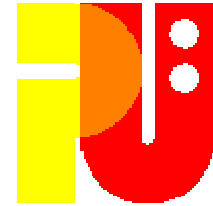
- W. Lalonde: *I can read C++ and Java but I can't read Smalltalk*; Journal of Object-Oriented Programming, February 2000, pp. 40-45
- B. Laffra: *Java for Smalltalk programmers*; 1999, www.developer.com/java/ent/article.php/614371/Java-for-Smalltalk-programmers.htm
 - G. Giorgie: *SmallTalk tutorial for Java programmers!*; 2002, daitanmarks.sourceforge.net/or/squeak/ST4J.pdf
 - S. Machari et al.: *Java VS. Smalltalk*; www.c2.com/cgi/wiki?JavaVsSmalltalk

Fortsetzung



- **Literaturanregungen (Forts.):**
- 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
- B. Schäling: *Programmieren in C# - Objektorientierung*; 2010, Kap. 2, www.highscore.de/csharp/einfuehrung/objektorientierung.html
- T. Howitz: *C# - typsicher, einfach, modern, objektorientiert*; undatiert, www.imn.htwk-leipzig.de/~waldmann/edu/ss04/se/howitz/c%23.pdf (in meinen Augen ein abschreckendes Beispiel)
- NN (Goldmann): *Webseiten programmieren in C# mit dem Visual Web Developer 2.0*; 2010, www.goldmann.de/webseiten-programmieren-in-c%23-mit-dem-visual-web-developer-20_tipp_365.html
- NN: *Smalltalk (.NET) implementation*; undated, www.refactory.com/Software/SharpSmalltalk/Implementation.html

Webanwendung (z.B. *Seaside*)



Was ist bei einer Webanwendung anders?

Programmablauf im Server oder/und beim Klienten?

Objektorientierung dabei?

Eigenschaften der *Smalltalk*-Umgebung *Seaside*

Zum Vergleich: Kurze Betrachtung auch von *AIDA/Web*

Literaturanregungen:

- (a panel, mehrere): *Smalltalk and the Web* ; 1996 (veraltet, trotzdem:),
c2.com/doc/web96.html
- R. Johnson: *Ideal Web application server* ; 2000,
www.swazoo.org/cs00-ideal_web_application_server.html
- Wikipedia: *AIDA/Web* ; en.wikipedia.org/wiki/AIDA/Web
- J. Mivšek et al. : *Swazoo Smalltalk Web server* ; 2009, www.swazoo.org/
- J. Mivšek: *AIDA/Web tutorial* ; 2009, www.aidaweb.si/tutorial.html

Fortsetzung



Literaturanregungen (Forts.):

- S. Ducasse et al.: *Seaside, a flexible environment for building dynamic web applications*; IEEE Software 24 (2007) 5, 56-63
- S. Ducasse et al.: *Dynamic Web development with Seaside*; 2009, book.seaside.st/book/
(rechts oben auf Introduction klicken für kostenloses Lesen !)
- 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
- D. Pollack: *Rails and Seaside – two new, great web frameworks*; 2006, blog.lostlake.org/index.php?/archives/7-Rails-and-Seaside-Two-new,-great-web-frameworks.html
- R. Leon: *Seaside for the .Net developer*; 2006, onsmalltalk.com/seaside-for-the-net-developer
- NN: *Hello World*; www.wikiservice.at/dse/wiki.cgi?HelloWorld#SmalltalkHelloWorld



Einfluß in sog. Skript- sprachen, *Ruby*, *JavaScript* u.ä.



Charakteristisches von sog. Skriptsprachen

Was hätte man von *Smalltalk* lernen können und sollen?

Was fehlt, was gibt es mehr?

Ausführung beim Server bzw. Klienten?

Betrachtung von *Ruby* und *Ruby on Rails*

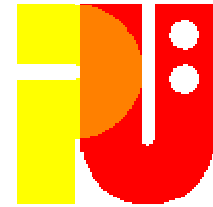
Fortsetzung

Literaturanregungen:

- Wikipedia: *Ruby (programming language)*; [en.wikipedia.org/wiki/Ruby_\(programming_language\)](http://en.wikipedia.org/wiki/Ruby_(programming_language))
- NN: *Ruby Tutorial, Teil 2*; undatiert, pc.de/software/ruby-tutorial-teil-914 (an einer Stelle entschieden zu kritisieren)
- C. Glanz: *Ein kurzer, nicht vollständiger Überblick ... (ZU Ruby)*; 2007, users.informatik.haw-hamburg.de/~sarstedt/AKOT/Ruby.pdf
- NN: *Ruby VERSUS Smalltalk VERSUS Objective-C VERSUS C++ VERSUS Java VERSUS Python VERSUS CLOS VERSUS Perl5 VERSUS XOTcl*; www.approximity.com/ruby/Comparison_rb_st_m_java.html



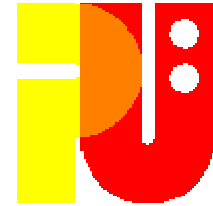
Einfluß in sog. Skript- sprachen, *Ruby*, *JavaScript* u.ä.



Literaturanregungen (Forts.):

- Wikipedia: *Ruby on Rails* ; en.wikipedia.org/wiki/Ruby_on_Rails
- T. Baustert, R. Wirdemann: *Ruby on Rails, Einführung & Überblick* ; 2006, www.b-simple.de/download/rails-overview.pdf
- S. Röhl, S. Schmiedel: *Webanwendungen mit Ruby und Rails , wie auf Schienen* ; 2004, www.linux-magazin.de/Heft-Abo/Ausgaben/2004/12/Wie-auf-Schienen?category=0
- NN (Cincom): *Comparing Seaside and Ruby on Rails* ; 2010, www.cincomsmalltalk.com/main/2010/05/comparing-seaside-and-ruby-on-rails/
- Wikipedia: *JavaScript* ; de.wikipedia.org/wiki/JavaScript
- J. Gregorio: *JavaScript is the new Smalltalk* ; 2002 -2009, bitworking.org/news/290/JavaScript-is-the-new-Smalltalk
- NN (Lshift): *Smalltalk vs. Javascript; Diff and Diff3 for Squeak Smalltalk* ; 2000 – 2009, www.lshift.net/blog/2008/07/01/smalltalk-vs-javascript-diff-and-diff3-for-squeak-smalltalk





Umfeld aus dem Zeitraum 1965 – 1970
Eigenschaften der Programmiersprache *Simula-67* aus der
Algol-60 -Familie
Objektorientierung bei *Simula-67* ?

Literaturanregungen:

- Wikipedia: *Simula* ; de.wikipedia.org/wiki/Simula
- O.-J. Dahl, K. Nygaard: *How object-oriented programming started* ; undat., heim.ifi.uio.no/~kristen/FORSKNINGSKORT_MAPPE/F_OO_start.html
- O.-J. Dahl, K. Nygaard: *IBM System 360/370 compiler and historical documentation* ; undatiert, www.edelweb.fr/Simula/
- J. Mitchell: *Simula and Smalltalk* ; undatiert, carlstrom.com/stanford/cs242/www-03/slides/simula-smalltalk.pdf



Umfeld aus dem Zeitraum 1985 – 1995

Eigenschaften der Programmiersprachen *Eiffel* bzw. *Objective C*

Können diese auch als eine Form einer „Reinkultur“ angesprochen werden?

Beachtenswerte Beiträge von *Eiffel* bzw. *Objective-C* zur „Reinkultur“?

Literaturanregungen:

- H. Gall et al.: *Objektorientierte Konzepte in Smalltalk, C++, Objective-C, Eiffel und Modula-3*; Informatik Spektrum 18 (1995) 195-202
- Wikipedia: *Eiffel*; [de.wikipedia.org/wiki/Eiffel_\(Programmiersprache\)](http://de.wikipedia.org/wiki/Eiffel_(Programmiersprache))
- B. Meyer: *Eiffel - The language*; Prentice-Hall, 1992.
- Wikipedia: *Objective-C*; de.wikipedia.org/wiki/Objective-C
- B. J. Cox,: *Object-oriented programming, an evolutionary approach*; Addison-Wesley, 1986.
- M. Gasbichler, H. Gast: *Smalltalk VS. Objective-C*; 2006, www-pu.informatik.uni-tuebingen.de/oopl-0506/material/st-objc-4.pdf





Was ist *bytecode* ?

Smalltalk-60-bytecode?

Vorläufer (0-Adress-Architektur)

Andere Lösungen? Umwandlung?

Wie zeigt sich „Objektorientierung in Reinkultur“ in einem *bytecode* ?

Literaturanregungen:

- NN (Wikipedia): *Bytecode* ; en.wikipedia.org/wiki/Bytecode
- NN: *Bytecode* ; www.economicexpert.com/a/Bytecode.htm
- NN (Wikipedia): *Burroughs large systems* ;
en.wikipedia.org/wiki/Burroughs_large_systems
- NN (Wikipedia); *P-code machine* ;
en.wikipedia.org/wiki/P-code_machine
- S.A. Moore: *Pascal-P: The portable Pascal compiler* ;
www.moorecad.com/standardpascal/PascalP.html

Fortsetzung



Literaturanregungen (Forts.):

- A. Goldberg, D. Robson: *Smalltalk-80 – The language*; Addison-Wesley, chapter 21 (verschiedene Auflagen!) (siehe auch: stephane.ducasse.free.fr/FreeBooks/BlueBook/Bluebook.pdf)
- Baumann: *Smalltalk, ein Blick hinter die Kulissen*; 2005, www.complang.tuwien.ac.at/anton/lvas/sem06w/baumann.pdf (hieraus Blätter #05t1 bis #05t2 [***] entnommen)
- NN (Cincom): *Drilling down to bytecode*; 2008, www.cincomsmalltalk.com/blog/blogView?showComments=true&entry=3378011312
- NN (Wikipedia): *Java bytecode* bzw. *java virtual machine*; en.wikipedia.org/wiki/Java_bytecode bzw. [.../Java_Virtual_Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine)
- B. Venners: *Bytecode basics – A first look at the bytecodes of the JVM*; 1996, www.artima.com/underthehood/bytecode.html
- NN (Wikipedia): *(.Net) Common Language Runtime*; en.wikipedia.org/wiki/Common_Language_Runtime bzw. [.../Common_Language_Infrastructure](http://en.wikipedia.org/wiki/Common_Language_Infrastructure)
- NN: *Bytecode design?*; 2009, stackoverflow.com/questions/1142848/bytecode-design

Speicherbereinigung



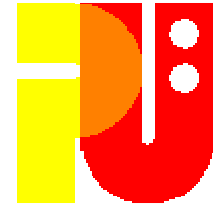
Eine Erfordernis von Objektorientierung, warum?

Überblick über Verfahren der Speicherbereinigung (engl. *garbage collection*)

Literaturanregungen:

- R.E. Jones, R. Lins: *Garbage collection – Algorithms for automatic dynamic memory management*; Wiley, 1996, ISBN 0-471-94148-4
- NN (Wikipedia): *Garbage collection*; [en.wikipedia.org/wiki/Garbage_collection_\(computer_science\)](http://en.wikipedia.org/wiki/Garbage_collection_(computer_science))
- NN (Wikipedia): *Reference counting*; en.wikipedia.org/wiki/Reference_counting
- NN (Cincom): *Memory management*; www.cincomsmalltalk.com/CincomSmalltalkWiki/DOWNLOAD/DOCS/memory-VW3.x.pdf
- J.M. McIntosh: *Intro to garbage collection in Smalltalk*; 1997/2001, www.smalltalkconsulting.com/papers/GCPaper/GCTalk%202001.htm
- NN (Microsoft): *Garbage collection*; 2007, msdn.microsoft.com/de-de/library/0xy59wtx.aspx
- A. Roehrl et al.: *Garbage collection (bei Ruby)*; 2002, www.approximity.com/rubybuch2/node189.html

Entwicklungsaufwand und Ausführungszeit



Bedeutung des Entwicklungsaufwands bei Entscheidung für einen Ansatz zum Vorgehen beim Programmieren

Aktuelle Einordnung (engl. *ranking*) von *Smalltalk* als Beispiel einer dynamischen Programmiersprache

Bedeutung von Ausführungszeit bei Entscheidung für ein Programmiersystem

Die sog. *just-in-time-compilation*

Die sog. *Skript-Sprachen*

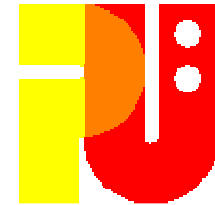
Literaturanregungen:

- NN (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, jazon.com/jazon08/en/conference/presentationdetails.html?type=sid&detail=5249



- **Literaturanregungen (Forts.):**
- NN: *JPMorgan leitet klare Vorteile aus Cincom Smalltalk ab*; Firmenschrift Cincom Systems, Inc., 2006
- 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
- E. Klimas: *Getting the biggest bang for your buck*; 1998, VisualAge Magazine, Mai 1998, www.lineaengineering.com/Resources/Productivity_/productivity_.html
- 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

“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 “WS10-11-#xx-version-ii.<ext>”,
“xx” elem {01 ÷ 15}, “ii” elem {1 ...})

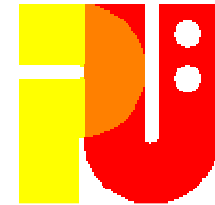
Ein Hinweis des Dekans:

“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

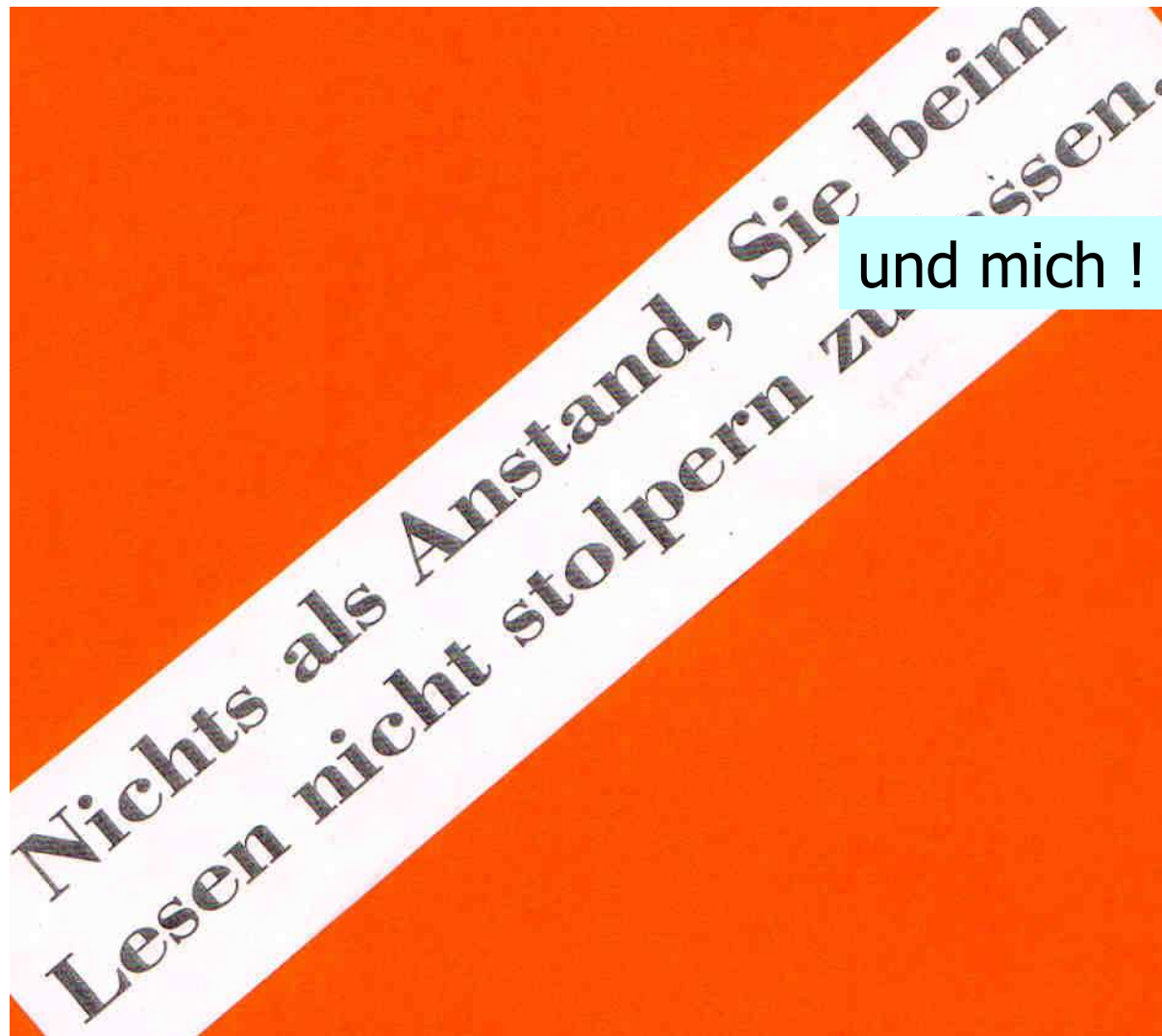
oder sprechen Sie Ihren Betreuer an.”



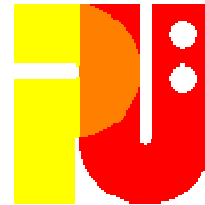
Web-Seite des Seminars:

www.pu.informatik.tu-darmstadt.de/Seminar-Smalltalk/

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



Werkstatt



Fortsetzung