

```
;;;;;;;;;;;;;;
;;
;; teh lisp workshop
;;

;;;;;;;;;;;;;;
;;
;; part4: CLOS und diverses
;;

;;
;; Common Lisp Object System
;;

; teil von ANSI Common Lisp
; * alles ist ein objekt
;   (auch funktionen, listen, strings)
; * klassenhierarchie:
;   t ist wurzel-klasse
; * reflektiv: MetaObjectProtocol
; * multiple dispatch
; * multiple inheritance

;;
;; defclass
;;

(defclass point-2d ()
  (x y))

;initargs, accessors machen

;mehrfachvererbung ist moeglich:
(defclass line (drawable-thing)
  ((start
    :initarg :start
    :accessor start)
   (end
    :initarg :end
    :accessor end)))

(defclass glowing-thing (drawable-thing)
  ...)

(defclass glowing-line (line glowing-thing)
```

```

... )

;
; methoden und generische funktionen
;

(defgeneric distance (thing1 thing2))

(defmethod distance ((p1 point-2d) (p2 point-2d))
  ;code this
)

;methoden "gehören" nicht einer klasse:
(defmethod distance ((p1 point-2d) (l1 line))
  ...)

; die spezialisierteste methode wird angewendet.
;
; mit call-next-method kann die naechst
; weniger spezialisierte aufgerufen werden

(defmethod draw ((thing drawable-thing))
  ...)

(defmethod draw ((my-line line))
  ...
  (call-next-method))

; before, after, around

(defmethod draw :before ((my-anim animation))
  (do-animation-step my-anim))

;;
;; mehr zu macros
;;
```

```
; macros sind wichtig
;
; * man bemerkt wiederholung eines
; musters im code -> macro draus
; machen -> aaah :)
; * die sprache entwickelt sich au
; das problem zu (siehe loop)
; * code wird lesbarer und kompakter,
; weil abstrakte ideen direkt
; ausgedrueckt werden koennen
; * macros haben lisp alle trends
; assimilieren lassen
; * "syntactic sugar" ist immer
; nur mit mitteln der sprache
; implementiert

; nachteile von lisp
; * generische funktionen duerfen nicht
; wie bestehende "normale" funktionen
; heissen
; * inkonsistenzen durch lange geschichte
; * executables erzeugen stinkt

; ich habe euch vorenthalten:
; * garbage collector
; * closures
; * lisp ist kompiliert: (disassemble #'foo)
; * libraries
; * builtin: hashtabellen, bignums,
; arrays, komplexe zahlen
; * multiple values
; * conditions
; * reader macros

#I(1 + 2 * (2 - 3))

;;
;; die gruende, lisp zu benutzen
;;
;
; lesbar und kompakt
; "Q: how can you tell you've reached lisp
; enlightenment?
; A: the parentheses disappear"
```

```
;
; leetness
; "just because we lisp programmers are better
;   than everyone else is no excuse for us to be
;   arrogant"
;
; macros machen lisp einzigartig
; "lisp is a programmable programming language"
; "lisp isn't a language, it's building material"
```

```
;;
;; und wie weiter?
;;
;
; * "Practical common lisp" lesen
;   http://gigamonkeys.com/book/
; * http://cliki.net
; * mich fragen :)
```