Überblick über *SML*

	Auswertung	
	statisch (Compilezeit)	dynamisch (Laufzeit)
Kernsprache		
expression e	$C \vdash exp \Rightarrow \tau$	$E \vdash exp \Rightarrow v \mid p$
declaration d	$C \vdash dec \Rightarrow E$	$E \vdash dec \Rightarrow E' \mid p$
pattern p	$C \vdash pat \Rightarrow (VE, \tau)$	$E, v \vdash pat \Rightarrow VE \mid Fail$
type ty	$C \vdash ty \Rightarrow \tau$	—
Modulsprache		
structure A	$B \vdash strexp \Rightarrow E$	$B \vdash strexp \Rightarrow E \mid p$
	$B \vdash strdec \Rightarrow E$	$B \vdash strdec \Rightarrow E \mid p$
signature S	$B \vdash sigexp \Rightarrow E$	$B \vdash sigexp \Rightarrow I$
	$B \vdash sigdec \Rightarrow G$	$B \vdash sigdec \Rightarrow G$
functor f	$B \vdash fundec \Rightarrow F$	$B \vdash fundec \Rightarrow F$

Dabei sind

$C = T, U, E$	$\in$ Context	= TyNameSet x TyVarSet x Env
$E = (SE, TE, VE)$	$\in$ Env	= StrEnv x TyEnv x ValEnv
τ	$\in$ Type	
$B = T, F, G, E$	$\in$ Basis	= TyNameSet x FunEnv x SigEnv x Env
p	$\in$ Pack	= ExVal
$I = (SI, TI, VI)$	$\in$ Interface	= StrInt x TyInt x ValInt

Die Kernsprache

Ausdrücke von *SML*

Expression

e	$:=$	a	*Atom*
	$\mid$	$e : \tau$	*getypter Ausdruck*
	$\mid$	$e\ a$	*Anwendung*
	$\mid$	$\texttt{fn}\ m$	*Abstraktion*
	$\mid$	$\texttt{raise}\ e$	*Ausnahme*
	$\mid$	$e\ \texttt{handle}\ m$	*Ausnahmebehandlung*

Atomic Expression

a	$:=$	c	*(Wert-)Konstante*
	$\mid$	x	*(Wert-)Variable*
	$\mid$	$A.x$	*Lange (Wert-)Variable*
	$\mid$	$\{l_1 = e_1, \ldots, l_n = e_n\}$	*Feld (record),* $n \geq 0$
	$\mid$	$(\texttt{let}\ d\ \texttt{in}\ e\ \texttt{end})$	*lokale Deklaration*
	$\mid$	(e)	*geklammerter Ausdruck*

Constant

c	$:=$	`nil`	*leere Liste*
	$\mid$	`::`	*Listenerweiterung*
	$\mid$	`true`	*Wahrheitswert*
	$\mid$	`false`	*Wahrheitswert*
	$\mid$	`=`	*Gleichheit*
	$\mid$	`ref`	*Zeigerbildung*
	$\mid$	`:=`	*Zeigerbelegung*
	$\mid$	`!`	*Zeigerwert ablesen*
	$\mid$	`Match`	*Ausnahme*
	$\mid$	`Bind`	*Ausnahme*

Ausdrücke von *SML* (Forts.)

Value Identifier

x	$:=$	$x_1 \mid x_2 \mid \ldots$	*Wertnamen*

Label

l	$:=$	$l_1 \mid l_2 \mid \ldots$	*Feldnamen*

Match

m	$:=$	p_1 `=>` $e_1 \mid \cdots \mid p_n$ `=>` e_n	*Regeln* $(n \geq 1)$

Pattern

p	$:=$	ap	*atomares Muster*
	$\mid$	$x\ ap$	*komplexes Muster*
	$\mid$	$A.x\ ap$	
	$\mid$	$p : \tau$	*getyptes Muster*
	$\mid$	x `as` p	*geschachteltes Muster*
ap	$:=$	`_`	*namenloses Muster*
	$\mid$	c	*Konstante*
	$\mid$	x	*Wertvariable*
	$\mid$	$A.x$	*lange Wertvariable*
	$\mid$	$\{l_1 = p_1, \ldots, l_n = p_n\}$	*Feldmuster* $(n \geq 0)$
	$\mid$	(p)	*geklammertes Muster*

Deklarationen von *SML*

Declaration			
d	$:=$	`val` p `=` e	*einfacher Wert* *)
	$\mid$	`val rec` x `=` e	*rekursiver Wert*
	$\mid$	`type` $(\alpha_1, \ldots, \alpha_n)$ tyc $=$ τ	*Typ*
	$\mid$	`datatype` $(\alpha_1, \ldots, \alpha_n)$ tyc $=$	*Datentyp*
		$x_1 \mid \ldots \mid x_{k+1}$ `of` $\tau_{k+1} \mid \ldots$	*mit Konstruktoren*
	$\mid$	`abstype` $(\alpha_1, \ldots, \alpha_n)$ tyc $=$	*abstrakter Datentyp*
		$x_1 \mid \ldots \mid x_{k+1}$ `of` $\tau_{k+1} \mid \ldots$	*mit Konstruktoren*
		`with` d `end`	*und Werten*
	$\mid$	`exception` x `of` τ	*Ausnahme*
	$\mid$	`open` A	*Öffnen einer Struktur*
	$\mid$	d; d	*sequentielle Dekl.*
	$\mid$	`infix` k $x_1 \ldots x_n$	*Infix-Erklärung* $(k <$ 1
	$\mid$	`infixr` k $x_1 \ldots x_n$	- " -, *rechtsassoziative*
tyc	$:=$	$t_1 \mid t_2 \mid \ldots$	*Typkonstruktor*

ABKÜRZUNG:

`fun` x p_1 `=` $e_1 \mid \ldots \mid x$ p_n `=` e_n statt
`val rec` x `=` `fn` $p_1 \Rightarrow e_1 \mid \ldots \mid p_n \Rightarrow e_n$

KONVENTION: Die gebundenen Variablen eines Ausdrucks oder einer Deklaration sind paarweise verschieden und verschieden von den freien Variablen.

$FV(e)$ ist die Menge der *freien Wertvariablen von e*

*) $FV(p) \cap FV(e) = \emptyset$

Typausdrücke von *SML*

Grundtypen

ι	$:=$	`unit`	*Einheitstyp*
	$\mid$	`bool`	*Wahrheitswerte*
	$\mid$	`int`	*ganze Zahlen*
	$\mid$	`word`	*8-bit (32-bit) Blöcke*
	$\mid$	`real`	*reelle Zahlen*
	$\mid$	`string`	*Zeichenreihen*
	$\mid$	`char`	*Zeichen*
	$\mid$	`exn`	*Ausnahmen*

Monotypen

τ	$:=$	ι	*Grundtyp*
	$\mid$	α	*Typvariable*
	$\mid$	$A.\alpha$	*lange Typvariable*
	$\mid$	$\{l_1 : \tau_1, \ldots, l_n : \tau_n\}$	*Feldtyp*
	$\mid$	$(\tau_1 \to \tau_2)$	*Funktionstyp*
	$\mid$	τ `list`	*homogene Liste*
	$\mid$	τ `ref`	*homogener Zeiger*
	$\mid$	$(\tau_1, \ldots, \tau_n)\ \mathit{Tyc}$	*definierter Typ*

Polytypen

$\overline{\tau}$	$:=$	τ	*Monotyp*
	$\mid$	$\forall\alpha\overline{\tau}$	*Typschema*

Typkonstruktoren

Tyc	$:=$	tyc	*einf. Typkonstruktor*
	$\mid$	$A.\mathit{tyc}$	*langer Typkonstruktor*

$FV(\overline{\tau})$ ist die Menge der *freien Typvariablen von* $\overline{\tau}$

Abkürzungen

Ausdrücke:	
$()$	$\{\,\}$
$(e_1, \dots, e_n)$	$\{1 = e_1, \dots, n = e_n\}$
$[e_1, \dots, e_n]$	$e_1 :: \dots :: e_n :: \texttt{nil}$
$\#\, l$	$\texttt{fn}\ \{l = x, \dots\}\ \texttt{=>}\ x$
$\texttt{case}\ e\ \texttt{of}\ m$	$(\texttt{fn}\ m)(e)$
$(e_1; \dots; e_n)$	$\texttt{case}\ e_1\ \texttt{of}\ \texttt{\_} =>$ $(\dots(\texttt{case}\ e_n\ \texttt{of}\ \texttt{\_} => e_n)\dots)$
$\texttt{if}\ e_1\ \texttt{then}\ e_2\ \texttt{else}\ e_3$	$\texttt{case}\ e_1\ \texttt{of}\ \texttt{true} => e_2$ $\mid \texttt{false} => e_3$
$\texttt{while}\ e_1\ \texttt{do}\ e_2$	$\texttt{let}\ \texttt{fun}\ x = \texttt{fn}\ () =>$ $\texttt{if}\ e_1\ \texttt{then}\ (e_2; x())\ \texttt{else}\ ()$ $\texttt{in}\ x()\ \texttt{end}$
Typen:	
$\tau_1 * \dots * \tau_n$	$\{1 : \tau_1, \dots, n : \tau_n\}$
Muster:	
$(p_1, \dots, p_n)$	$\{1 = p_1, \dots, n = p_n\}$
$[p_1, \dots, p_n]$	$p_1 :: \dots :: p_n :: \texttt{nil}$
Deklarationen:	
$\texttt{fun}\ x\ p_1 \texttt{=} e_1$ $\mid \quad \dots$ $\mid x\ p_n \texttt{=} e_n$	$\texttt{val}\ \texttt{rec}\ x \texttt{=} \texttt{fn}\ p_1 => e_1$ $\mid \quad \dots$ $\mid p_n => e_n$