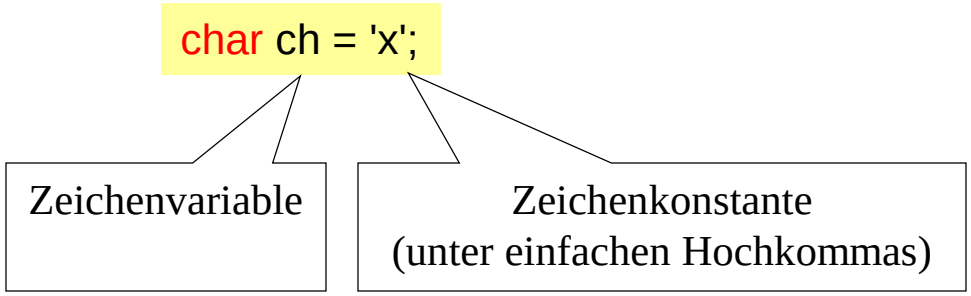
6. Zeichen

Datentyp char



Zeichen braucht man zur Verarbeitung von Texten.

Zeichencodes

ASCII (American Standard Code for Information Interchange)

- 1 Zeichen = 1 Byte (128 oder 256 Zeichen darstellbar)
- z.B. in Pascal und C verwendet

Unicode (www.unicode.org)

- 1 Zeichen = 2 Bytes (65536 Zeichen darstellbar)
- auch Umlaute, griechische, arabische Zeichen etc.
- z.B. in Java und C# verwendet
- ASCII ist Teilmenge von Unicode

ASCII

0	NUL	16	DLE	32	space	48	0	64	@	80	P	96	´	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

Wichtige Steuerzeichen

BS	*backspace*	löscht Zeichen vor Cursor	CR	*carriage return*	Zeilenvorschub
HT	*horizontal tab*	Tabulatorsprung	LF	*line feed*	folgt auf CR
ESC	*escape*		FF	*form feed*	Seitenvorschub

Unicode

0000 - 007F	ASCII-Zeichen
0080 - 024F	Umlaute, Akzente, Sonderzeichen
0370 - 03FF	griechische Zeichen
0400 - 04FF	cyrillische Zeichen
0530 - 058F	armenische Zeichen
0590 - 05FF	hebräische Zeichen
0600 - 06FF	arabische Zeichen
...	...

Details siehe
http://www.unicode.org

Deutsche Umlaute

00E4	ä	00C4	Ä	00DF	ß
00F6	ö	00D6	Ö		
00FC	ü	00DC	Ü		

Unicode-Zeichenkonstanten

Alle Zeichen können auch mit ihrem Unicode-Wert angegeben werden:

'\u*dddd*'

Beispiele:

'\u0041'	'A'
'\u000d'	CR (carriage return)
'\u0009'	TAB
'\u03c0'	p

Spezielle Zeichen

'\n'	LF (line feed, newline)
'\r'	CR (carriage return)
'\t'	TAB
'\"'	'
'\\'	\

Zeichen-Operationen

Zuweisungen

```
char ch1, ch2 = 'a';
ch1 = ch2;          // ok, gleicher Typ
int i = ch2;        // ok, char kann int zugewiesen werden
ch1 = (char) i;     // Zuweisung nach Typumwandlung möglich
```

double Ê float Ê long Ê int Ê short Ê byte Ê char

Vergleiche (==, !=, <, <=, >, >=)

Zeichen sind nach Unicode-Wert geordnet;
Buchstaben und Ziffern liegen aufeinanderfolgend

```
if ('a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z') ...
```

Arithmetische Operationen (+, -, *, /, %)

```
10 + (ch - 'A')      // Ergebnistyp: int
```

Zeichenarrays

```
char[] s = new char[20];    // initialisiert alle Elemente mit '\u0000'
char[] t = {'a', 'b', 'c'};
```

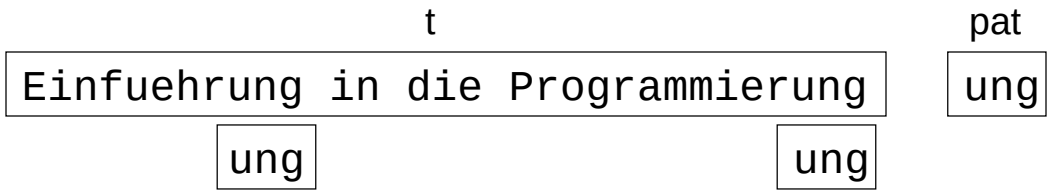
Standardfunktionen mit Zeichen

if (Character.isLetter(ch)) ...	true, wenn *ch* ein Unicode-Buchstabe ist
if (Character.isDigit(ch)) ...	true, wenn *ch* eine Ziffer ist
if (Character.isLetterOrDigit(ch)) ...	
if (Character.isLowerCase(ch)) ...	true, wenn *ch* ein Kleinbuchstabe ist
if (Character.isUpperCase(ch)) ...	true, wenn *ch* ein Großbuchstabe ist
ch1 = Character.toUpperCase(ch2);	wandelt *ch2* in einen Großbuchstaben um
ch1 = Character.toLowerCase(ch2);	wandelt *ch2* in einen Kleinbuchstaben um

6. Zeichen

Beispiel: Textsuche

geg.: Text *t*, Muster *pat*
ges.: erstes Vorkommen von *pat* in *t*



Ergebnis: *pos* ³ 0: *pat* in *t* an Stelle *pos*
pos < 0: *pat* kommt nicht in *t* vor

```
char[] t = ...;              // Text
char[] pat ...;              // Pattern
int pos = -1;                // initialisiert mit "nicht gefunden"

int last = t.length - pat.length;
for (int i = 0; i <= last; i++) {
   if (t[i] == pat[0]) {
      int j = 1;
      while (j < pat.length && pat[j] == t[i+j]) j++;
      // j == pat.length || pat[j] != t[i+j]
      if (j == pat.length) {pos = i; break;}
   }
}
```