

Variable, konstanter, blokker  
og skop

# Vårt første javaprogram

```
1  
2  
3  
4  
5  
6 public class FirstProgram {  
7  
8     public static void main(String[] args) {  
9  
10        System.out.println("Hello, World!");  
11    }  
12  
13 }
```

# Terminologi

- Variabel
  - En plassholder i minnet, hvor vi kan lagre noe av den ønskede og forhåndsdefinerte typen
- Deklarasjon
  - Deklarasjon av en variabel. Vi lager en variabel og gir den en datatype den kan lagre

# Terminologi

- Tilordning
  - Vi tilordner en variabel en verdi. Dette gjør vi med tilordningsoperatoren ( = )

# Eksempler

- Deklarasjon
  - `int integerVariabel;`
  - `double enVariabelForFlyttall;`
- Tilordning
  - `integerVariabel = 2; // Vi lagrer tallet 2 i plassen "integerVariabel"`
  - `enVariabelForFlyttall = 4.4; // Vi lagrer 4.4`

# Eksempler

- Deklarasjon og tilordning på en og samme linje
  - `String tekstVariabel = "En ny linje med tekst";`

# Variabel-teori

- Vi kan lage en "formel" som beskriver dette
  - `<datatype> <variabelNavn> <tilordning> <uttrykk>`
  - `int valgfrittNavn = 4 + 4;`
  - `double nyttValgfrittNavn = 4.2 - 5.0;`
  - `char bokstav = 'b';`
  - `String tekst = "Hei på deg";`

# Terminologi

- Primitive datatyper
  - En innebygget datatype som kun består av en binærrepresentasjon.
  - Det viktige for dere å kunne er hvilke datatyper som er primitive.

# Eksempel

- Primitive datatyper

- int
- float
- double
- char
- byte
- short

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	<b>NUL</b> (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	<b>SOH</b> (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	<b>STX</b> (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	<b>ETX</b> (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	<b>EOT</b> (end of transmission)	36	24	044	&#36;	\$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	<b>ENQ</b> (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	<b>ACK</b> (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	<b>BEL</b> (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	<b>BS</b> (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	<b>TAB</b> (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	<b>LF</b> (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	<b>VT</b> (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	<b>FF</b> (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	<b>CR</b> (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	<b>SO</b> (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	<b>SI</b> (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	<b>DLE</b> (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	<b>DC1</b> (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	<b>DC2</b> (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	<b>DC3</b> (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	<b>DC4</b> (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	<b>NAK</b> (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	<b>SYN</b> (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	<b>ETB</b> (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	<b>CAN</b> (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	<b>EM</b> (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	<b>SUB</b> (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	<b>ESC</b> (escape)	59	3B	073	&#59;	:	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	<b>FS</b> (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	
29	1D	035	<b>GS</b> (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	<b>RS</b> (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	<b>US</b> (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Source: [www.LookupTables.com](http://www.LookupTables.com)

Source: [www.LookupTables.com](http://www.LookupTables.com)

3T	1E	03D	<b>02</b> (numeric separator)	e3	3E	03D	&#e3;	3	82	2E	13D	&#82;	-	T3D	3E	13D	&#T3D;	DEL
30	1E	03E	<b>03</b> (numeric separator)	e3	3E	03E	&#e3;	>	8f	2E	13E	&#8f;	v	T3E	3E	13E	&#T3E;	~
3a	1D	032	<b>02</b> (numeric separator)	eT	3D	032	&#eT;	=	82	2D	132	&#82;	]	T32	3D	132	&#T32;	]

# Terminologi

- Typekonvertering
  - Å endre fra en datatype til en annen. Det finnes to former for typekonvertering
- Implisitt typekonvertering
- Eksplisitt typekonvertering

# Typekonvertering

- Implisitt:
  - Når javakompilatoren gjør det av seg selv fordi det er naturlig
  - `double flyttallsVariabel = 4;`
  - `System.out.println(flyttallsVariabel);` // Utskrift i java

```
1x-193-157-242-6:GL-INF1000-4 stene$ javac Test.java
1x-193-157-242-6:GL-INF1000-4 stene$ java Test
4.0
1x-193-157-242-6:GL-INF1000-4 stene$ █
```

```
1x-193-157-242-6:GL-INF1000-4 stene$ █
```

# Typekonvertering

- Eksplisitt:
  - Eksplisitt typekonvertering er når vi selv tvinger kompilatoren til å endre fra en type til en annen
  - `int heltall = (int) 'A';`

```
4.0  
1x-193-157-242-6:GL-INF1000-4 stene$ javac Test.java  
1x-193-157-242-6:GL-INF1000-4 stene$ java Test  
65  
1x-193-157-242-6:GL-INF1000-4 stene$ █
```

```
JX-J23-J25-545-0:07-INF1000-4 2f6U6? █
```

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	<b>NUL</b> (null)	32	20	040	&#32;	<b>Space</b>	64	40	100	&#64;	<b>@</b>	96	60	140	&#96;	<b>`</b>
1	1	001	<b>SOH</b> (start of heading)	33	21	041	&#33;	<b>!</b>	65	41	101	&#65;	<b>A</b>	97	61	141	&#97;	<b>a</b>
2	2	002	<b>STX</b> (start of text)	34	22	042	&#34;	<b>"</b>	66	42	102	&#66;	<b>B</b>	98	62	142	&#98;	<b>b</b>
3	3	003	<b>ETX</b> (end of text)	35	23	043	&#35;	<b>#</b>	67	43	103	&#67;	<b>C</b>	99	63	143	&#99;	<b>c</b>
4	4	004	<b>EOT</b> (end of transmission)	36	24	044	&#36;	<b>\$</b>	68	44	104	&#68;	<b>D</b>	100	64	144	&#100;	<b>d</b>
5	5	005	<b>ENQ</b> (enquiry)	37	25	045	&#37;	<b>%</b>	69	45	105	&#69;	<b>E</b>	101	65	145	&#101;	<b>e</b>
6	6	006	<b>ACK</b> (acknowledge)	38	26	046	&#38;	<b>&amp;</b>	70	46	106	&#70;	<b>F</b>	102	66	146	&#102;	<b>f</b>
7	7	007	<b>BEL</b> (bell)	39	27	047	&#39;	<b>'</b>	71	47	107	&#71;	<b>G</b>	103	67	147	&#103;	<b>g</b>
8	8	010	<b>BS</b> (backspace)	40	28	050	&#40;	<b>(</b>	72	48	110	&#72;	<b>H</b>	104	68	150	&#104;	<b>h</b>
9	9	011	<b>TAB</b> (horizontal tab)	41	29	051	&#41;	<b>)</b>	73	49	111	&#73;	<b>I</b>	105	69	151	&#105;	<b>i</b>
10	A	012	<b>LF</b> (NL line feed, new line)	42	2A	052	&#42;	<b>*</b>	74	4A	112	&#74;	<b>J</b>	106	6A	152	&#106;	<b>j</b>
11	B	013	<b>VT</b> (vertical tab)	43	2B	053	&#43;	<b>+</b>	75	4B	113	&#75;	<b>K</b>	107	6B	153	&#107;	<b>k</b>
12	C	014	<b>FF</b> (NP form feed, new page)	44	2C	054	&#44;	<b>,</b>	76	4C	114	&#76;	<b>L</b>	108	6C	154	&#108;	<b>l</b>
13	D	015	<b>CR</b> (carriage return)	45	2D	055	&#45;	<b>-</b>	77	4D	115	&#77;	<b>M</b>	109	6D	155	&#109;	<b>m</b>
14	E	016	<b>SO</b> (shift out)	46	2E	056	&#46;	<b>.</b>	78	4E	116	&#78;	<b>N</b>	110	6E	156	&#110;	<b>n</b>
15	F	017	<b>SI</b> (shift in)	47	2F	057	&#47;	<b>/</b>	79	4F	117	&#79;	<b>O</b>	111	6F	157	&#111;	<b>o</b>
16	10	020	<b>DLE</b> (data link escape)	48	30	060	&#48;	<b>0</b>	80	50	120	&#80;	<b>P</b>	112	70	160	&#112;	<b>p</b>
17	11	021	<b>DC1</b> (device control 1)	49	31	061	&#49;	<b>1</b>	81	51	121	&#81;	<b>Q</b>	113	71	161	&#113;	<b>q</b>
18	12	022	<b>DC2</b> (device control 2)	50	32	062	&#50;	<b>2</b>	82	52	122	&#82;	<b>R</b>	114	72	162	&#114;	<b>r</b>
19	13	023	<b>DC3</b> (device control 3)	51	33	063	&#51;	<b>3</b>	83	53	123	&#83;	<b>S</b>	115	73	163	&#115;	<b>s</b>
20	14	024	<b>DC4</b> (device control 4)	52	34	064	&#52;	<b>4</b>	84	54	124	&#84;	<b>T</b>	116	74	164	&#116;	<b>t</b>
21	15	025	<b>NAK</b> (negative acknowledge)	53	35	065	&#53;	<b>5</b>	85	55	125	&#85;	<b>U</b>	117	75	165	&#117;	<b>u</b>
22	16	026	<b>SYN</b> (synchronous idle)	54	36	066	&#54;	<b>6</b>	86	56	126	&#86;	<b>V</b>	118	76	166	&#118;	<b>v</b>
23	17	027	<b>ETB</b> (end of trans. block)	55	37	067	&#55;	<b>7</b>	87	57	127	&#87;	<b>W</b>	119	77	167	&#119;	<b>w</b>
24	18	030	<b>CAN</b> (cancel)	56	38	070	&#56;	<b>8</b>	88	58	130	&#88;	<b>X</b>	120	78	170	&#120;	<b>x</b>
25	19	031	<b>EM</b> (end of medium)	57	39	071	&#57;	<b>9</b>	89	59	131	&#89;	<b>Y</b>	121	79	171	&#121;	<b>y</b>
26	1A	032	<b>SUB</b> (substitute)	58	3A	072	&#58;	<b>:</b>	90	5A	132	&#90;	<b>Z</b>	122	7A	172	&#122;	<b>z</b>
27	1B	033	<b>ESC</b> (escape)	59	3B	073	&#59;	<b>;</b>	91	5B	133	&#91;	<b>[</b>	123	7B	173	&#123;	<b>{</b>
28	1C	034	<b>FS</b> (file separator)	60	3C	074	&#60;	<b>&lt;</b>	92	5C	134	&#92;	<b>\</b>	124	7C	174	&#124;	<b> </b>
29	1D	035	<b>GS</b> (group separator)	61	3D	075	&#61;	<b>=</b>	93	5D	135	&#93;	<b>]</b>	125	7D	175	&#125;	<b>}</b>
30	1E	036	<b>RS</b> (record separator)	62	3E	076	&#62;	<b>&gt;</b>	94	5E	136	&#94;	<b>^</b>	126	7E	176	&#126;	<b>~</b>
31	1F	037	<b>US</b> (unit separator)	63	3F	077	&#63;	<b>?</b>	95	5F	137	&#95;	<b>_</b>	127	7F	177	&#127;	<b>DEL</b>

# Terminologi

- Tekststreng ("streng")
  - En "spesiell" datatype i Java for å lagre tekst.
  - Klassen String lagrer tekst.
  - `String tekststreng;`

# Eksempel

- String:
  - `String tekst1, tekst2, tekst3;`
  - `tekst1 = "Hei på deg";`
  - `tekst2 = "Hei" + "på" + "deg";`
  - `tekst3 = "Hei " + "på " + "deg";`

# Terminologi

- Tester
- En **if-test** er en boolsk test. Boolsk betyr at den enten kan være sann eller usann.
- I java heter dette **true** eller **false**
- `if (<uttrykk>) { ... }`

# Eksempel

```
if (true) { ... }
```

```
if (false) { ... }
```

# Terminologi

- Boolske operatorer
- Vi har seks boolske operatorer:
  - `==` // Er det samme som / Equals
  - `!=` // Ikke det samme som / Not equals
  - `<` // Mindre enn
  - `>` // Større enn
  - `<=` // Mindre enn eller lik
  - `>=` // Større enn eller lik

# Eksempel

```
if (1 < 2) { ... }
```

```
if (a != b) { ... }
```

```
if (q >= z) { ... }
```

```
if (heltallVariabel < 60) { ... }
```

# Terminologi

- En siste primitiv datatype
  - `boolean`
- En boolean er en variabeltype som kun kan inneholde `true` eller `false`.

# Eksempel

```
boolean testVariabel;
```

```
testVariabel = true;
```

```
if (testVariabel) {
```

```
    ... // Noe kode
```

```
}
```

# Eksempel

```
boolean nyVariabel = 10 <= 20;
```

```
if (nyVariabel) {
```

```
    ... // Noe kode
```

```
} else {
```

```
    ... // Noe annen kode
```

```
}
```

# Terminologi

- If, else if og else
  - Alle "if-tester" begynner med en `if (<uttrykk>)`
  - Hvis den feiler kan vi hvis vi vil legge til en `else if (<uttrykk>)`
  - Til slutt kan det også legges til en `else` som slår inn for alle andre tilfeller

# Eksempel

```
int a = 10;  
int b = 11;
```

```
if (a > b) {  
    System.out.println("A er størst"); // Utskrift  
} else if (a == b) {  
    System.out.println("A og B er like"); // Utskrift  
} else {  
    System.out.println("B er størst"); // Utskrift  
}
```

# Terminologi: Java-program

- Hva består et ferdig javaprogram av?
  - En klassedefinisjon (Klasse)
  - En main-metode (Metode)
  - Programkode (Kode)

# Klasser i java

- Vi kommer kun til å jobbe med en enkelt klasse de to første dagene
- En klasse lager vi ved å skrive setningen:
  - `public class Klassenavn { ... }`
  - `class Klassenavn { ... }`

# Main-metoden

- Metoden "main" er en veldig spesiell metode
- Det er den metoden som får beskjed om å kjøre først når vi ønsker å kjøre programmene våre.
- `public static void main (String[] args) { ... }`

# Programkode

- Programkoden er det vi vil at programmet vårt skal gjøre.
- For eksempel å summere to heltall:

```
int heltall1 = 10;
```

```
int heltall2 = 33;
```

```
int sum = heltall1 + heltall2;
```

# Om kompilator, skop og Livekoding