



Java

TDDC77

Objektorienterad Programmering

Föreläsning 1

Sahand Sadjadee
IDA, Linköpings Universitet
Hösttermin, 2020

Outline

- Introduktion till Java
- Variabler
- Literaler
- Utmatning

Introduktion till Java

Java nyckelord

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

Java nyckelord

Nyckelord gör **grundläggande** uppgifter i varje programmeringsspråk. Några exempel i Javas fall:

- Att tilldela utrymme i minnet för att lagra data. `int`, `float`, `double`, `char`, ...
- Att styra programflödet. `if`, `while`, `switch`, ...
- ...

Det är inget nyckelord till exempel för att göra inmatning och utmatning!!! Hur ska vi göra massor av andra grejer som kommer att behövas i programmet? Ska vi implementera de själva?

ETT STORT NEJ!

Ett enkelt program

- Filen måste ha samma namn som klassen.
- Kommentarer med `/* ... */` och `//...`
- Filen måste innehålla en klass, `class Namn { /* body */ }`
- En main metod ligger i klassen som följande
 - `public static void main(String[] args){`
 - `/* body */`
 - `}`
- Metoden main gör klassen körbar, Runnable. Den anropas automatiskt av virtuella maskinen när klassen körs genom att använda kommandot `java KlassNamnet`

Det är viktigt att använda samma signatur för main metoden hela tiden.

```
/* Hej.java
 * Demonstrera hur man kan mata ut texter
 * till den standard outputen
 */
class Hej
{
    // Skriv ut ett enkelt meddelande
    public static void main(String[] args)
    {
        System.out.println("Hej");
        System.out.println("... IT1 studenter!");
    }
}
```

Förutsättning

- Vi använder följande kod tillfälligt:

```
class Name {  
    public static void main(String[] args){  
        //vår kod ligger här  
    }  
}
```

- Java filen som innehåller koden ska ha samma namn som klassen.
- Vår kod ligger alltid i `main`.

Task 1 - 3 minuter

I terminalen:

1. Skapa en katalog, säg `lec01` med `mkdir lec01`.
2. Kom in i katalogen med `cd lec01`.
3. Skapa/öppna `HelloWorld.java` i `atom` med `atom HelloWorld.java`
4. Skriv följande koden i filen:

```
public class HelloWorld
{
    public static void main(String[] args)
    {
        System.out.println("Hello World");
    }
}
```

5. Spara filen.
6. Avsluta.
7. Kompilera javakoden med `javac HelloWorld.java`
8. Kör programmet med `java HelloWorld`

Ett enkelt program

- `class`, `Hej`, `public`, `main`, `System`, `String`, `println` är identifierare.
- Några har vi valt själva (`Hej`, `args`).
- Vissa har andra programmerare valt (`String`, `System`, `out`, `println`). **Andra programmerare!!?**
- Vissa är nyckelord (`class`, `public`, `static`, `void`) http://docs.oracle.com/javase/tutorial/java/nutsandbolts/_keywords.html

```
/* Hej.java
 * Demonstrera hur man kan mata ut texter
 * till den standard outputen
 */
class Hej
{
    // Skriv ut ett enkelt meddelande
    public static void main(String[] args)
    {
        System.out.println("Hej");
        System.out.println("... IT! studenter!");
    }
}
```

Egna identifierare(identifiers)

- Kombination av bokstäver, siffror, _, och \$.
- Kan inte börja med en siffra.
- Exempel: Hej, total, numberOfElements, \$var, NUM_SEATS
- Men inte: 2th, var#5
- Det finns skillnad mellan stora och små bokstäver: sum, Sum och SUM är inte samma identifierare.

```
/* Hej.java
 * Demonstrera hur man kan mata ut texter
 * till den standard outputen
 */
class Hej
{
    // Skriv ut ett enkelt meddelande
    public static void main(String[] args)
    {
        System.out.println("Hej");
        System.out.println("... IT1 studenter!");
    }
}
```

Blanktecken(White spaces)

- Java använder blanktecken för att separera ord.
- Man kan formatera sin text på olika sätt.
- Man ska sträva efter att göra programmet så läsbart som möjligt

```
/* Demonstrera hur man kan mata ut texter
 * till den standard outputen
 */
class Hej{ // Skriv ut ett enkelt meddelande
public static void main(String[] args){ System.out.println("Hej");
System.out.println("... IT1 studenter!"); }}
```

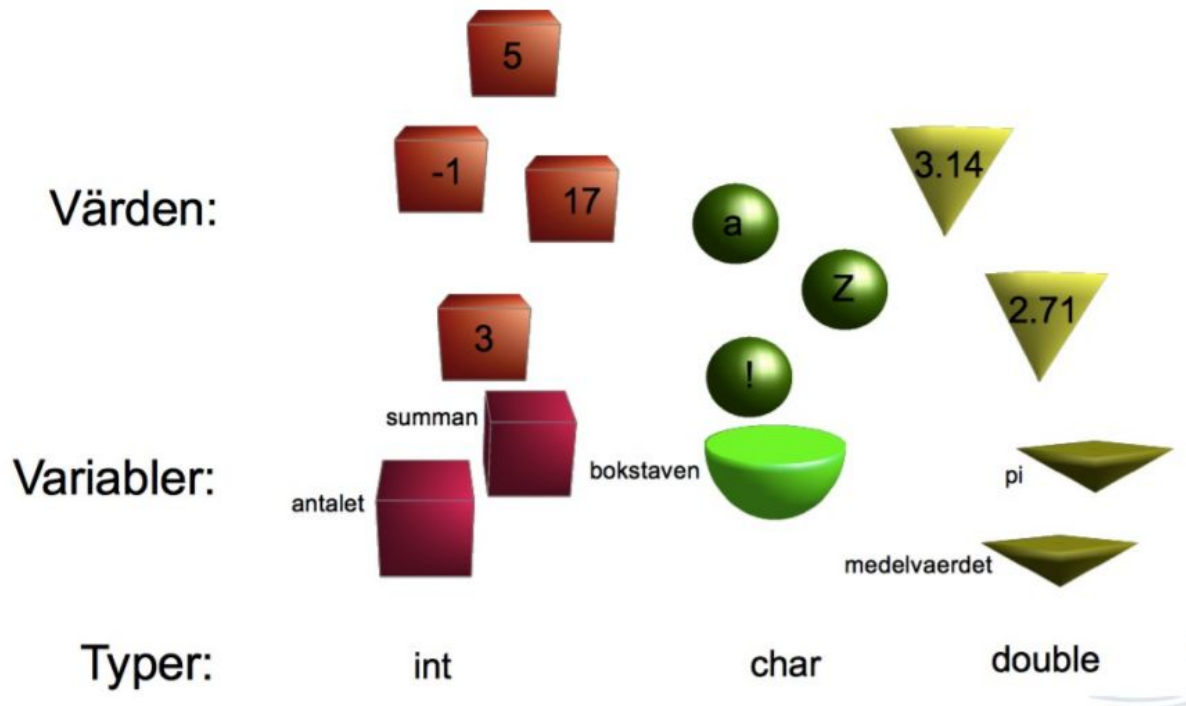
```
/* Hej.java
 * Demonstrera hur man kan mata ut texter
 * till den standard outputen
 */
class Hej
{
    // Skriv ut ett enkelt meddelande
    public static void main(String[] args)
    {
        System.out.println("Hej");
        System.out.println("... IT1 studenter!");
    }
}
```

Variabler

Variabler

- En variabel är en plats i primärminnet.
- Den används för att lagra ett värde.
- Varje variabel har ett namn/identifierare.
- Skilj på variabelns namn och värdet den lagrar.
- Varje variabel har också en typ.

Variabler och Datatyper



Primitiva Datatyper

Type	Description	Default	Size
boolean	true or false	false	1 bit
byte	two complement integer	0	8 bits
char	Unicode character	\u0000	16 bits
short	two complement integer	0	16 bits
int	two complement integer	0	32 bits
long	two complement integer	0	64 bits
float	IEEE 754 floating point	0.0	32 bits
double	IEEE 754 floating point	0.0	64 bits
void	None/Nothing		

Deklaration

- Variabler måste deklarerar.
- Deklarationer associerar en typ(till exempel, `int`, `char`, `String`) till ett namn(till exempel, `age`, `height`, `savingsBalance`) .
 - `int age;`
- Namnet ska vara meningsfullt.
- Namnet ska vara en giltig identifierare.
- Alla ord i namnet börjar med stora bokstäver förutom första ordet. Till exempel, `String myCar`, `int totalValue`.
- Man använder `final` för att deklarerar en konstant. Namnen på konstanter skrivs i stora bokstäver, t.ex: `final double PI`, `final int MAXPLACES`.

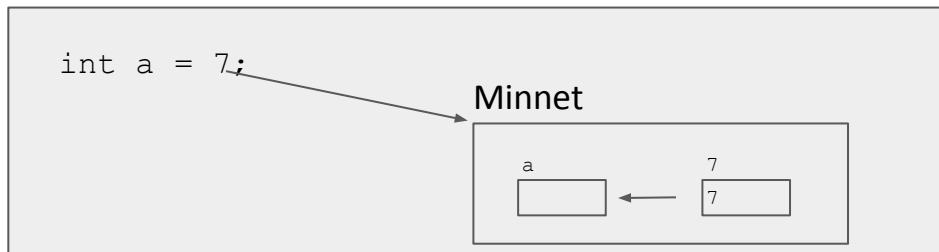
Task 2 - 3 minuter

I terminalen:

1. Kom in i katalogen med `cd lec01`.
2. Skapa/öppna `Variables.java` i Atom med `atom Variables.java`
3. Deklarera följande variablerna i main metoden:
 - a. `name`
 - b. `height`
 - c. `weight`
 - d. `age`
 - e. `dateOfBirth`
4. Bestäm typerna själva.

Literaler

- En literal är ett konstant värde som skrivs direkt i koden.
- Liksom variabler lagras literaler också i primärminnet.
- En literal har också en typ.
- När virtuella maskinen vill lagra en literal i minnet reserverar den ett block i minnet. Storleken på blocket bestäms av typen av literalen. Sedan lagrar den literalen i blocket och nämner också blocket samma som literalen.
- Tilldelning är ett sätt att ge en variabel ett värde.

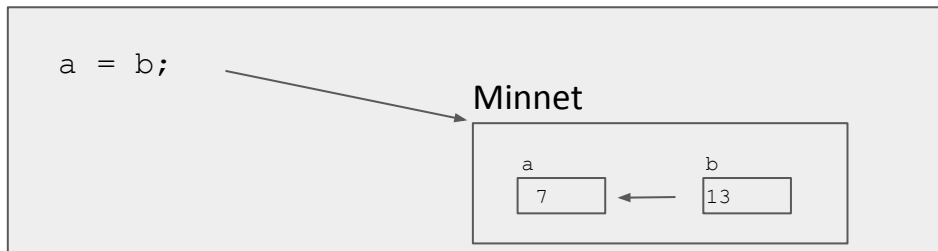


Typer på literaler

Literal	Type
<code>'A', 'c', 'z'</code>	<code>char</code>
<code>11, 15, 2018</code>	<code>byte</code> eller <code>short</code> eller <code>int</code>
<code>3.141519</code>	<code>double</code>
<code>3.141519f</code>	<code>float</code>
<code>3.141519d</code>	<code>double</code>
<code>true, false</code>	<code>boolean</code>
<code>"Hello TDDC77!"</code>	<code>String</code>

Tilldelning (assignment)

- ett sätt att ge en variabel ett värde.
 - `int a = 7;` eller `int a; a = 7;`
 - `int b = 13;` eller `int b; b = 7;`
 - `a = b;`
- En variabel ska **initialiseras** med något värde innan den kan användas.
- Storleken på typen på högersidan ska vara alltid mindre än storleken på typen på vänstersidan.



Task 3 - 2 minuter

I terminalen:

1. Kom in i katalogen med `cd lec01`.
2. Öppna `Variables.java` i atom med `atom Variables.java`
3. Initialisera alla skapade variablerna med lämpliga värden.

Task 4 - 2 minuter

1. Kom in i katalogen med `cd lec01`.
2. Öppna `Variables.java` i atom med `atom Variables.java`
3. Testa vilka av följande tilldelningar är giltiga?
 - 1) `int a = 3.3;`
 - 2) `int b = 3.3d;`
 - 3) `char c = 3;`
 - 4) `short d = 3;`
 - 5) `boolean e = 1;`
 - 6) `String f= 'a';`

Tilldelning

Left-side	Right-side
boolean	boolean
char	char, short
byte	byte
short	char, byte
int	byte, char, short
long	byte, char, short, int
float	byte, char, short, int
double	byte, char, short, int, float

Utmatning: `System.out.print` och `System.out.println`

- Att skriva till skärmen, en fil, nätverk ... alltså data som kommer från programmet.
- Brukar vara ganska lätthanterligt.
- `System.out.print` och `System.out.println` kommer ni att använda oftast.

Utmaning: println

```
/* StraengVar.java
 *
 * Illustrera konceptet av en sträng variabeln
 */
public class StraengVar{

    // Metoden deklarerar och tilldelar en sträng variabel
    public static void main(String[] args){
        String exampleNamn="TDDC77";
        String otherName = exampleNamn;
        System.out.println("Kursen heter: ");
        System.out.println(exampleNamn);
        System.out.println(otherName);
    }
}
```

Strängar

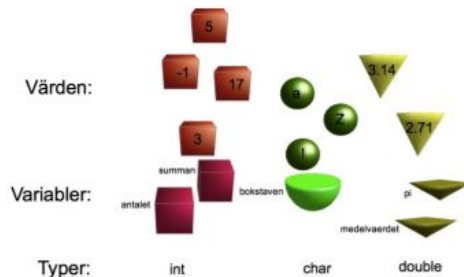
- Exempel på en icke primitiv datatyp, eller en så kallade referenstyper
- “Rader av tecken”, “TDDC77:are”
- Man kan inte ändra på en sträng efter att ha skapat en (immutable, diskuteras senare i kursen)

Att konkatenera sträng litteraler (string literals)

```
/* KlaraPlus.java
 *
 * Illustrera Strängskonkatenering
 */

public class KlaraPlus
{
    // Metoden konkatenera och skriver ut "klara färdiga gå!"
    public static void main(String[] args){
        System.out.print("Klara" + "... färdiga" + "... .. gå!!!");
    }
}
```

Variabler och Datatyper



```
public class DeklarationOchTilldelning
{
    public static void main(String[] args)
    {
        int talet=4, sum=0;
        double pi=3.14159265359, meddelvaerde;
        char bokstaven='!';
        String namnet;

        System.out.println("talet värde är: " + talet);
    }
}
```

Strängar

```
/* Str.java
 * Programmet demonstrerar hur man kan använda sig
 * av Java APIet för att manipulera strängar
 */
class Str{

    // Metoden anropar på olika Sträng metoder från APIet
    static public void main(String[] args){
        String kursnamn = "TDDC77", are = ":are!";

        System.out.println("char at index 1: " + kursnamn.charAt(1));

        System.out.println("compare to \"TDDC77\" gives: " + kursnamn.compareTo("TDDC77"))

        System.out.println("compare to \"Z\" gives: " + kursnamn.compareTo("Z"));

        System.out.println("Hej " + kursnamn.concat(are));
        System.out.println("Hej " + kursnamn + are);

        System.out.println("TDDC77 equals tddc77: " + kursnamn.equals("tddc77"));

        System.out.println("TDDC77 equals ignore case: "
            + kursnamn.equalsIgnoreCase("tddc77"));

        System.out.println("Length of TDDC77: " + kursnamn.length());

        System.out.println("Replace D by d in TDDC77: "
            + kursnamn.replace('D', 'd'));

        System.out.println("Substring of TDDC77 from 2 to 5: "
            + kursnamn.substring(2,5));

        System.out.println("To lower case: " + kursnamn.toLowerCase());
```



Task 5 - 2 minuter

1. Kom in i katalogen med `cd lec01`.
2. Skapa/öppna `TestPrint.java` i atom med `atom TestPrint.java`
3. Skapa main metoden.
4. Skriv ut `"1" + 1` i main metoden? är utskrivna meddelandet exakt som vad du förväntade?



Tack för att du lyssnade!