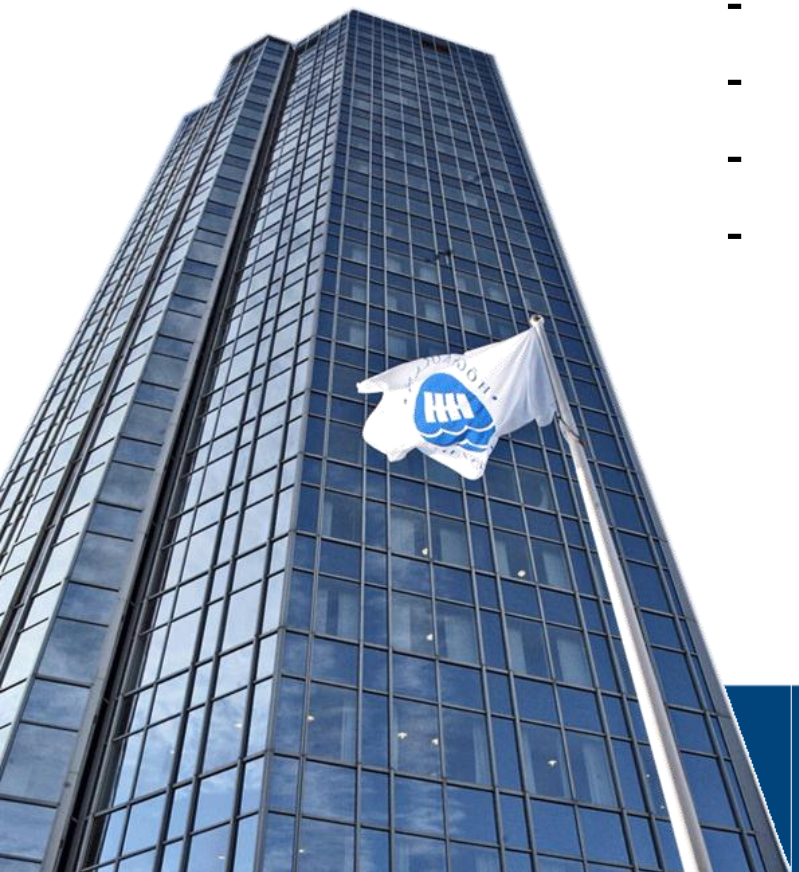
Webbprogrammering 15hp



- Variabler, datatyper, objekt och collections.
- Struktur
- Funktioner
- Objektorientering

Nu är det dags för att kika på kuggarna i maskineriet…

- Hittills har vi i lab 1 och 2 kikat på HTML, CSS samt masterpages och navigering, mestadels genom drag & drop. Nu går vi in på hur vi kan använda C# och kod för att skapa funktioner, klasser och annat kul i ASP.NET…

- Vi börjar med variabler, datatyper och struktur…

Datatyper och variabler

- Variabler är data som tillfälligt lagras i datorns internminne (RAM)
- Variabler kan vara av många olika typer för att optimera hanteringen av minnet
  (se utdelat blad över olika datatyper)
- Variabler kan även lagras i vektorer (arrays) som håller många variabler av samma typ

Olika datatyper

bool	True/False
byte	Whole numbers 0-256
Int	Whole numbers, in the range ± 2147483647
uint	Whole unsigned numbers 0-4294967295
decimal	With up to 29 significant digits, most accurate for floating-point numbers
double	As Decimal, but with a more narrow range Faster, often preferred
single	As Double but with a more narrow range.
char	A single caracter
string	Storing strings (text) – which can be almost anything Up to 2 billion characters
datetime	hh:mm:ss MM:DD:YYYY
object	Parent of all data types

More datatypes exist but these are the most used ones…

Deklarera variabler

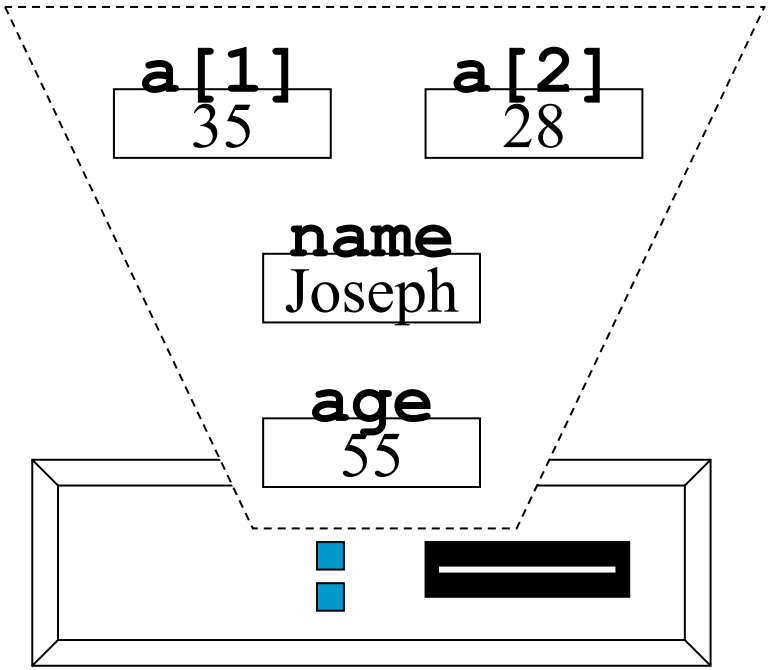
- Allokera plats och ett värde åt en variabel kallas att deklarera variabeln:

```
datatype myVariableName
```

- Skapa variabelnamn som är enkla att förstå (och gärna korta)

```
// deklarera en variabel för att lagra tal
och sträng räckvidd inom klassen
        int age;
        string name;

// tilldela värden i de olika variablerna
                age = 55;
                name = ”Joseph”;
```

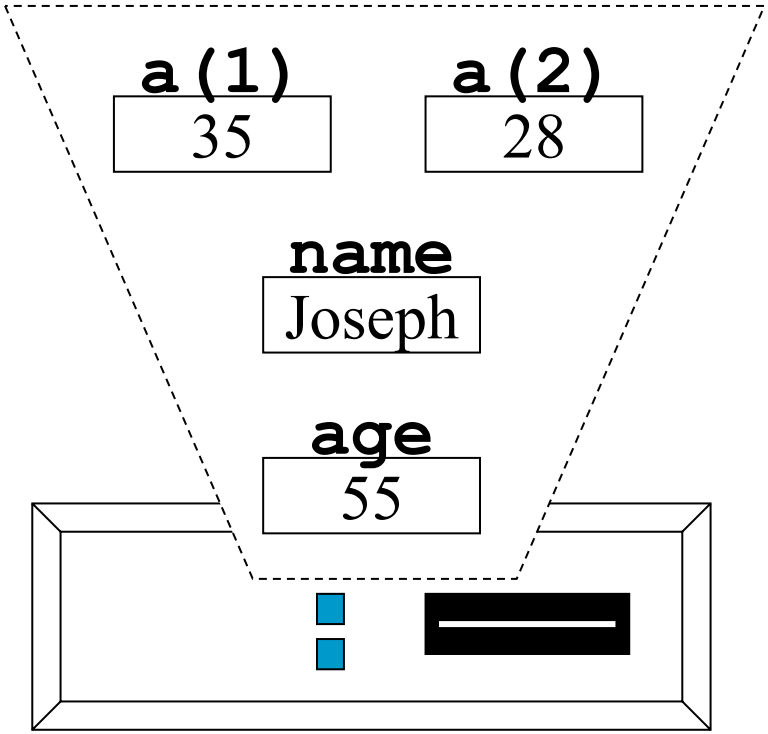


Variabelnamn, mm

- Tidigare användes vanligen prefix innan ett variabelnamn (ungersk notation):

```
string strName;
int intAge;
```

- Detta anses numera föråldrat eftersom vi har IntelliSense i Visual Studio. En annan fördel är att det helt enkelt är lättare att läsa utan prefix.
- Ge variabler meningsfulla namn som stämmer överens med dess syfte.
- Tänk på att versal och gemen bokstav i variabelnamn behandlas olika i C#.
- F1 ger tillgång till hjälp då vi står i koden, hjälp för det markören står på…



Konvertera datatyper

- Ibland behöver data konverteras ifrån en datatyp till en annan för att undvika onödiga fel i applikationen, t ex är värden ifrån textbox och dropdownlist strängar. Ska beräkningar med tal göras behöver vi konvertera från sträng till tal.

- All data kan konverteras till strängar, t ex datum som vi arbetat med tidigare:

```
label1.text =
System.DateTime.Now.ToString();
```

- ToString är en metod som konverterar till strängformat t ex för en variablel.

- Ett annat sätt är att använda klassen convert:

```
bool myBool =
Convert.ToBoolean(”True”);
```

- Ett tredje sätt är att använda casting med det fungerar bara för kompatibla typer, t ex fungerar det inte att konvertera datetime till int medan double till int fungerar

```
object myobj = 1;
int myint = (int)myobj;
```

- Parantes med önskad datatyp innan variabeln som ska konverteras

Arrays

- En array som hanterar tal kan deklareras såhär:

```
/* skapar en array med två platser
a[0] och a[1] */
Int[] a = new int[2];
```

- Tilldelning kan ske såhär:

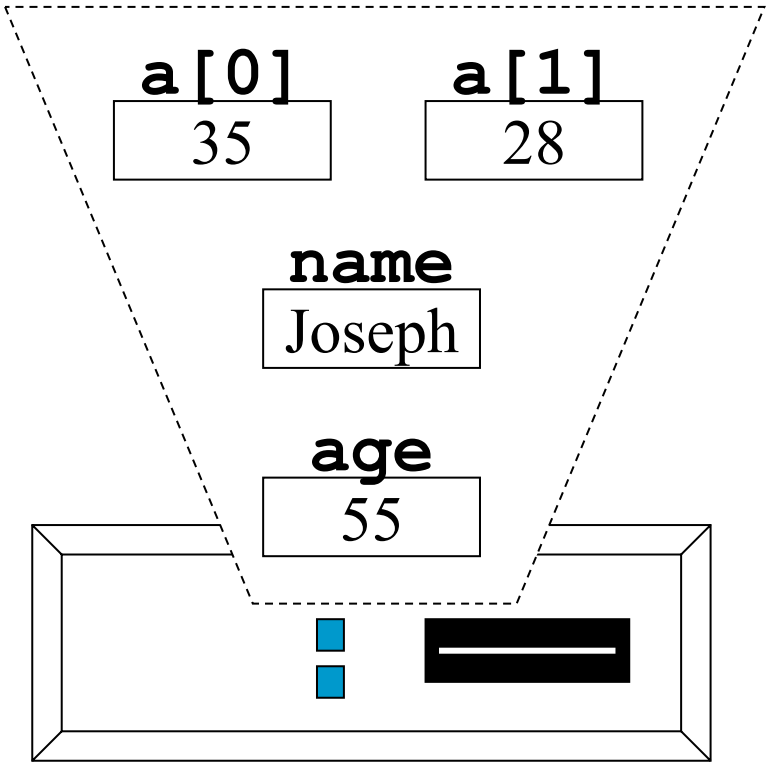
```
a[0] = 35;
a[1] = 28;
```

- Arrays har fast storlek

```
a[2] = 56; // fel uppstår
```

- Arrayen kan dock omdefineras

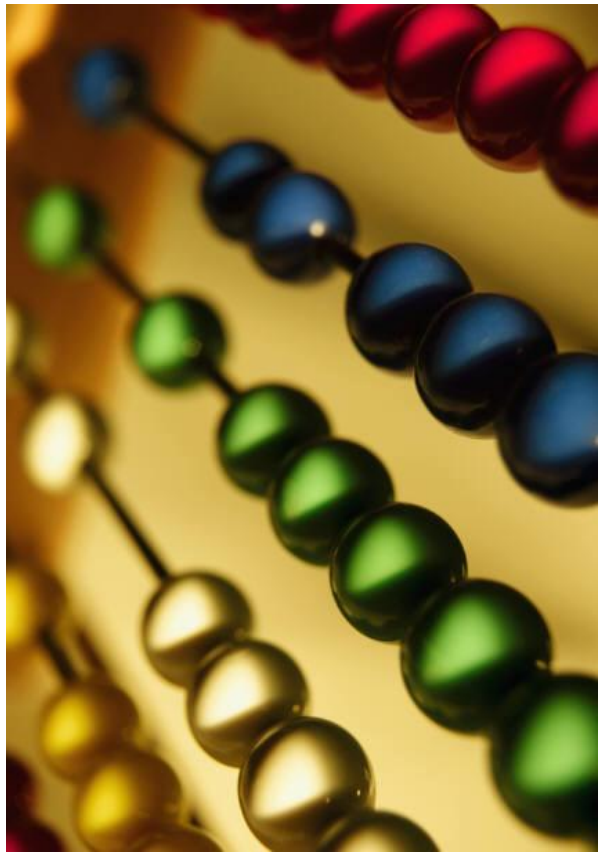
```
array.Resize<int>(ref a, 3);
a[2] = 56; // fungerar nu
```



Collections

- En collection låter dig lagra fler än ett object i en variabel, som att samla saker i en och samma väska
- För en collection finns det metoder som add(), remove() och clear() som möjliggör en enkel hantering

```
// skapar en arraylist & lägger
   till 3 olika roller
ArrayList roles = new ArrayList();
roles.Add(”Administrators”);
roles.Add(”ContentManagers”);
roles.Add(”Members”);
```



Collections 2

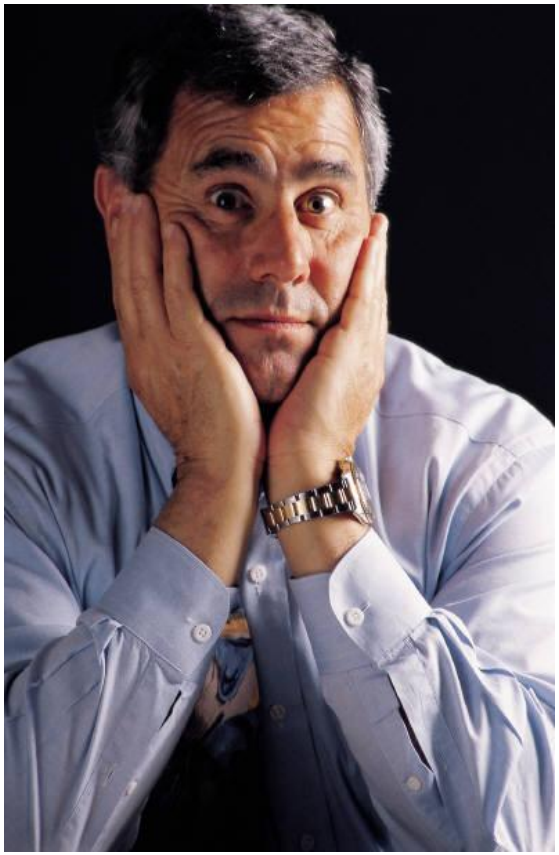
- Finns i Systems.Collections namespace laddas med: `using System.Collections;`
  - ArrayList
    - General collection for objects
  - HashTable
    - Storing key/value pairs
    - Keys must be unique
  - Queue
    - First-in, first-out collection
  - SortedList
    - Ordered storage for key/value pairs

- Stack
  - Last-in, first-out
- StringCollection
  - A collection of strings

Väl mellan array och collections

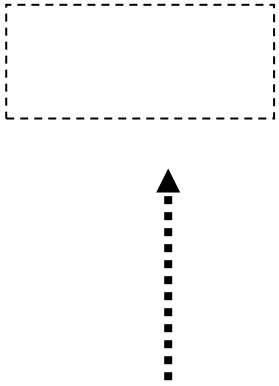
- Use arrays when the number of elements don’t change
  - Faster, more efficient…and perhaps easier to understand
- Use collections when you need dynamic allocation of elements, and special functionalities (search, sort, etc) of collections
- Both can handle data binding, tex:

```
GridView1.DataSource = ShoppingCart.getProducts();
GridView1.DataBind();
```



Att arbeta med variabel utan värde (null-värde)

- Värdet 0 (noll) är som alla andra nummer ett värde
- En variabel som saknar värde håller ett värde som kallas null – är ej detsamma som 0
  - T ex kan det finnas fält som ska in i en databas där värde avsaknas eller ej går att tillämpa



(null = värde saknas eller går ej att tillämpa)

Stränghantering

- Mycket av användarrelaterad input och output är string-värden. Finns det många olika sätt att hantera och förändra string-värden på.
- Tänk på att blanksteg också är ett tecken

- Jämförelser:

  Är båda identiska är villkoret sant och koden körs…

```
If (name1 == name2)
{// kod som gör nåt
}
```

- Konkatenering (sätta samman):

```
string first = ”Hej ”;
string second = ”hopp!”;
string result;
result = first + second;
```

Vad blir resultatet av ovanstående?

Operatorer

- Aritmetiska operatorer
  + - * /
- Lite mer advancerade
  - Exponent ^
    Math.Pow(2, 3); // =8
  - Modulus (remainder)
    1 % 2 = 1 // rest 1
    2 % 2 = 0 // ingen rest

- Jämförelseoperatorer
  - Also, rather easy
    - ==, !=, <, >, <=, >=
    - is (jämför objekt)

- Logiska operatorer
  - Allows combinations of expressions
    && (And) ||(Or) !(Not)
  - Special cases
    && (AndAlso) || OrElse

Operatorers exekveringsordning

- Beräkningar exekveras (körs) från vänster till höger utifrån operatorernas prioriteringar som kan förändras genom att använda paranteser

  1 + 2 * 3 = 7, men (1 + 2) * 3 = 9
- Operatorernas exekveringsordning:

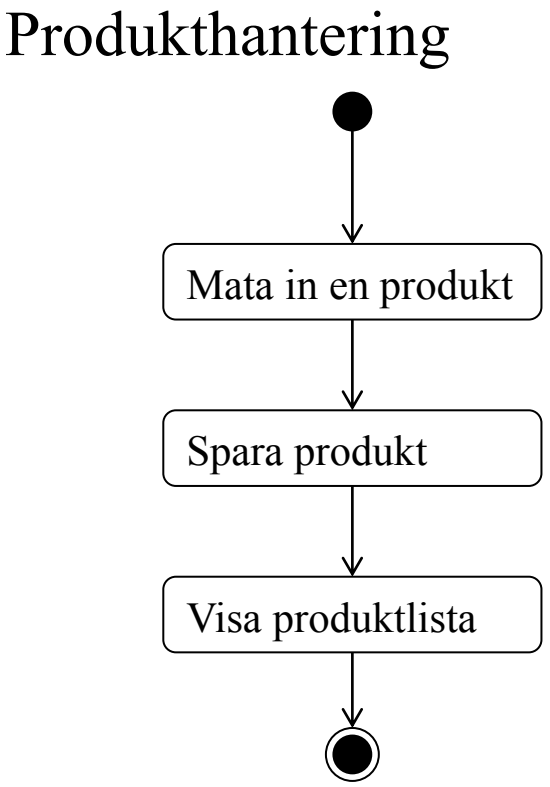
  * / % + -

Tre viktiga strukturer

- Logik kan struktureras med hjälp av tre olika grundstrukturer:
  - Sekvens (sequence)
    - Gör först A, sen B, sen C
  - Val (selection)
    - Om ett villkor uppfylls gör vi en sak, annars gör en annan sak
  - Upprepning (loop)
    - Så länge villkor är uppfyllt upprepas processen…

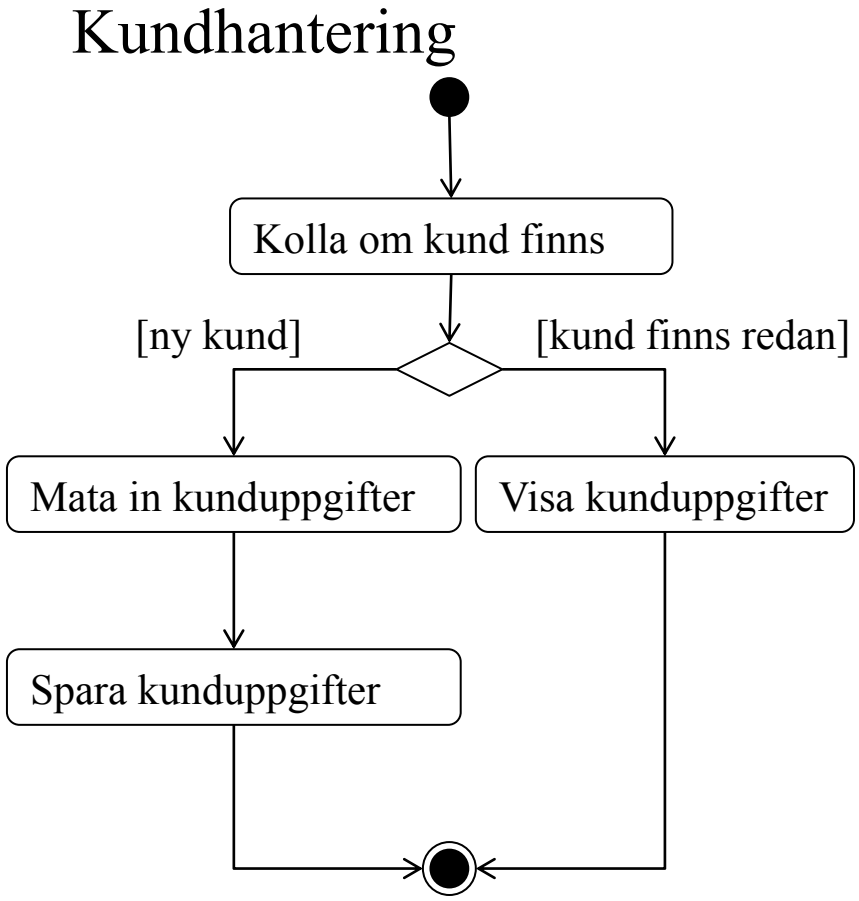
Sekvens: Att göra saker i ordning

- En sekvens beskriver händelser som utförs i steg för steg. T ex först A, sen B, sist C…
- Om vi beskriver detta med hjälp av beskrivningsteknik UML, ett aktivitetsdiagram kan det se ut som i figuren



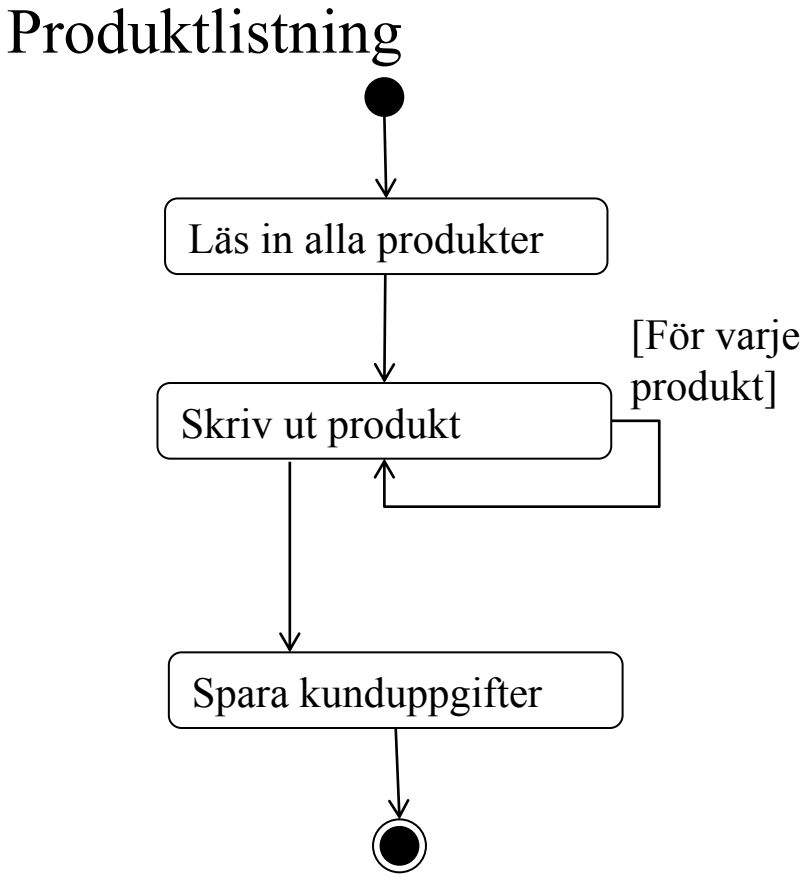
Selection: att göra val

- En selection är ett val mellan olika alternativ
- Else används för att samla upp alla andra alternativ
- Villkor skrivs inom hakparanteser, se figuren



Iteration: att upprepa något

- Iterationer är en process som upprepas.
- Iteration markeras med villkor inom hakparanteser, […] till samma aktivitet, se figur. Så länge villkor är uppfyllt sker iterationen. Ibland är antal iterationer noll.



Val

- Om ett villkor är sant körs koden

```
If (user.IsInRole(”User”)
{
deletebutton.visible=true;
}
```

- alternativ körning med else om det första villkoret ej var uppfyllt

```
If (user.IsInRole(”User”)
{
deletebutton.visible=true;
}
else
{
deletebutton.visible=false;
}
```

- Testa flera alternativ med else if

```
If (user.IsInRole(”User”)
{
deletebutton.visible=true;
}
else if user.IsInRole(”Admin”)
{
deletebutton.visible=true;
}
else
{
deletebutton.visible=false;
}
```

- Alternativa metoder finns, tex

```
switch
```

Switch

- Om flera alternativ existerar kan en switch-struktur vara lättare att använda:

```
switch (lstOperator.selectedValue)
{case "+":
   result=value1+value2;
   break;
 case "-":
   result=value1-value2;
   break;
}
resultlabel.text=result.ToString();
```

- Strukturen ger stor flexibilitet…
- Strängar inom citat: ”…”

- …t ex kan du fånga flera alternativ samtidigt
  - Simple test
    ```
    case 3:
    ```
    …
  - Multiple test
    ```
    case 3:
    case 5:
    case 8:
    ```
    …
  - Range test
    ```
    case 6 - 11:
    ```
    …
  - Logic test
    ```
    case < 25:
    ```
    …
  - Boolean test
    ```
    case True:
    ```
    …
  - Default test
    ```
    case default:
    ```
    …

Iterationer(loopar)

- for

```
 // for-loop (startvillkor, slutvillkor, steg)
for (int loopCount=0;loopCount<roles.Count;loopCount ++)
                {
                    Label2.Text += roles[loopCount]+"<br />";
                }
```

- foreach (väldigt användbart vid collections)

```
 foreach (string role in roles)
                {
                    Label2.Text += role + "<br />";
                }
```

- while

```
 bool lyckades = false;
                int i=0;
                while (!lyckades | i==3)
                {
                    lyckades = SendMailMessage();
                    i++;
                }
```

Iterationer (loopar)

- do

```
 // for-loop (startvillkor, slutvillkor, steg)
for (int loopCount=0;loopCount<roles.Count;loopCount ++)
                {
                    Label2.Text += roles[loopCount]+"<br />";
                }
```

- do while (sen testning)

```
        int x = 0;
        do
        {
                  Label2.Text += x.ToString();
                  x++;
        } while (x < 5);
```

Base pages

- Alla aspx-sidor baseras på klassen System.Web.UI.Page.
- Om funktionalitet i den klassen ej räcker till kan ytterligare funktionalitet läggas till utöver System.Web.UI.Page genom att skapa en bassida.
- Alla sidor för websajten kan därefter baseras på denna bassida som i sin tur baseras på System.Web.UI.Page
- En base page klass hamnar i App_Code mappen i rooten på din webbsajt

Themes

- Ett theme är en samling filer som definierar hur en sida ska se ut.
- Den hamnar i App_Themes mappen i rootkatalogen för din webbsajt.
- I mappen APP_Themes skapas sedan en underkatalog för varje theme
- Css-filer, bilder och skins kan ingå som en del av ett theme

- Kan sättas på sidnivå
  <% Page … Theme=”DarkGrey” %>

- Kan sättas för site i web.config
  <pages theme=“DarkGrey”>
  …
  </pages>

- Kan även sättas i programkod
  (se exempel s222 i kurslitteratur)

Skins

- Ett sätt att ge enhetligt utseende på exempelvis alla knappar för webbsajten i ASP.NET
- Placeras i foldern App_Themes och har extension .skin
- Får ej ha id attributet eftersom ett skin appliceras på alla kontroller av samma typ
- Bara visuella attribut kan användas för ditt skin. Attribut som enable går ej att ange.

Innehåll i en .skin fil för utseende på buttons:

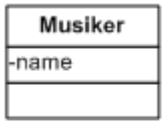
```
<asp:Button BackColor=”#cccccc”
        ForeColor=”#308462” runat=”server” />
```

Jämför med CSS:

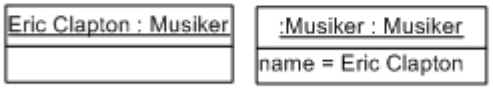
```
<asp:Button CssClass=”MyButton” runat=”server” />

.MyButton
{
  color: #308462;
  background-color: #cccccc;
}
```

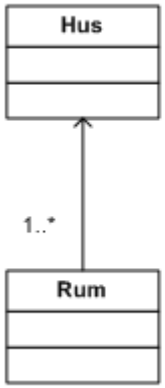
Lösningsförslag



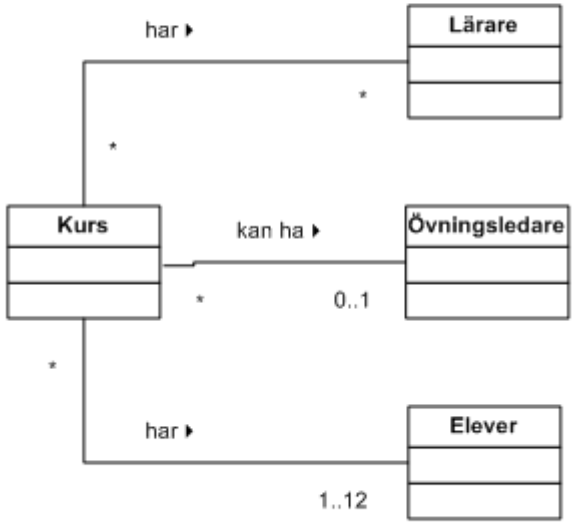
1. Eric Clapton är en musiker



2. Hus består av rum

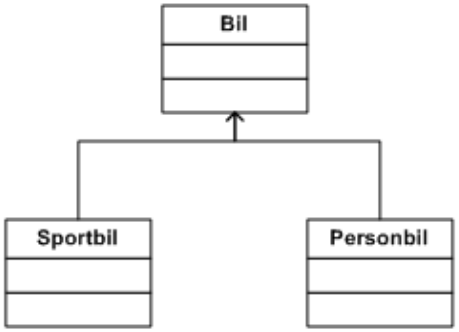


6. En kurs har en lärare, eventuellt en övningsledare och maximalt 12 elever



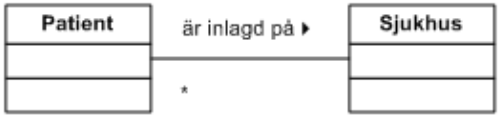
lösningsförslag

3. En bil är en sportbil eller en personbil



7. Ett äktenskap förenar 2 personer

4. En patient är inlagd på sjukhus



5. En livsform kan vara ett djur eller en växt