5.3 Foreach-satsen, en ny kontrollstruktur

```
<!-- Foreach.html -->
<!-- Summering av en arrays element med Foreach-satsen (13-14) -->
<head>
   <title>Foreach-satsen</title>
   <script>
      function start()
      {
         MinArray = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
         sum1 = 0, sum2 = 0
         for (index = 0; index < MinArray.length; index++)
             sum1 += MinArray[index]       // Summering med vanlig for-sats
         document.writeln("<h2>Sum med for-satsen: " + sum1 + "</h2>")
         for (element in MinArray)         // För ALLA element i arrayen:
             sum2 += MinArray[element]     // summera över elementen
         document.writeln("<h2>Sum med Foreach-satsen: " + sum2 + "</h2>")
      }
   </script>
</head>
<body onload="start()">
</body>
```

Scriptet **Foreach** summerar värdena i arrayen **MinArray** på två olika sätt: en gång med en vanlig **for**-sats på raderna **10-11** och en annan gång med en annan sats på raderna **13-14** som vi kallar för **Foreach**-satsen. Dess syntax skiljer sig från den vanliga **for**-satsen. Den nya satsen är i själva verket en ny kontrollstruktur som vi inte använt hittills. Att den nu introduceras i samband med arrays är inte en tillfällighet. Vi kommer att se varför. Ovan visas ett körexempel.

Foreach-*satsen*

```
for (element in MinArray)
    sum2 += MinArray[element]
```

Översatt till svenska:

För varje **element** av arrayen **MinArray**
summera elementen till **sum2**.

Denna sats som används på raderna **13-14** summerar värdena i arrayen **MinArray**. Vi kunde inte ta upp den i kapitlet om kontrollstrukturer därför att den förutsätter array-begreppet eller liknande sammansatta datatyper, som vi då inte hade hunnit gå igenom. Den är t.o.m. idealisk för att hantera arrays. Den gör samma sak som

for-satsen, men har en lite annorlunda – ja t.o.m. enklare syntax, om man är förtrogen med arrays.

element – ett namn som är valt av oss – kallas för **Foreach**-satsens *iterationsvariabel*. **element** pekar på värdet (innehållet) som står i arrayen. Iteration betyder upprepning och innebär här att satsens kropp upprepas: Programflödet fortskrider från element till element tills alla element är genomgångna. Det reserverade ordet **in** betyder *av* eller *element av*. **MinArray** pekar på arrayen som ska loopas igenom. Därför: ” För varje **element** *av* arrayen **MinArray**”.

Foreach-satsens enkelhet består i att den till skillnad från **for**-satsen varken behöver ett start-, steg- eller slutvärde resp. avslutningsvillkor. Den går helt enkelt igenom arrayens *alla* element, från det första till det sista. Det är själva arrayen som bestämmer start-, steg- och slutvärdena. Variabeln **element** pekar i varje varv av loopen på resp. arrayelementets värde och kan sedan användas i loopens kropp för att göra det man önskar. I vårt exempel för att summera arrayens element.

En viktig egenskap av iterationsvariabeln är att den inte kan ändra arrayelementens värden i **Foreach**-satsens kropp. Den är så att säga *read only*. I praktiken innebär detta att iterationsvariabeln inte får förekomma till vänster om tilldelningsoperatorn (**=**) i någon sats i **Foreach**-satsens kropp. Vill man i **Foreach**-satsens kropp ändra på arrayelementens värden måste man använda **for**-satsen istället med arrayens index som räknare.

Iterationsvariabel istället för räknare

Att **Foreach**-satsens syntax är trots enkelhet mer sofistikerad än så, kan man se när man tittar på **Foreach**-satsens kropp på rad **14**:

```
sum2 += MinArray[element]
```

Här står iterationsvariabeln **element** på en plats som egentligen är reserverad för räknaren (indexet). Men **element** är ju inte något index, utan en variabel som i varje varv av loopen pekar på resp. arrayelementets värde, utan att vara identisk med själva värdet. Just här hade det varit kanske bättre att välja ett annat namn för iterationsvariabeln än **element**, för att framhäva skillnaden.

I själva verket tilldelas **element** i varje varv av loopen det aktuella arrayelementets värde – dvs ett annat värde i varje varv. Därför antar **element** här *rollen* av en ”räknare”. Man kan också säga att vi har att göra med en ny betydelse av hakparentesen: Den kan innehålla **Foreach**-satsens iterationsvariabel, för att iterera över en arrays alla element.