Vad vi går igenom

Vad är ett scriptspråk ?

Vad är Javascript?

Hur körs koden?

Javascript editorer

Kodbibliotek

Vad vi går igenom

Vad är ett en variabel ?

Typer av variabler i Javascript

var och let

undefined och null

Aritmetiska operatorer

Kommentarer

Vad vi går igenom

Logiska beslut

If satser

Logiska operatorer

Switch

Vad är ett kodblock ?

Vad är kompilering?

En kompilator är ett program som går igenom koden som man skrivit och kontrollerar att den är riktigt skriven.

När kod testas av kompilatorn kallas det för att den kompileras

Finns det fel i koden får man upp en lista på alla fel som måste fixas innan koden kan bli körbar.

Finns det inga fel i koden bygger kompilatorn en körbar fil av koden.

Vad är ett scriptspråk?

Man skiljer på kompilerande och interpreterande språk.

Kod skriven i tex C# eller Java kompileras innan den kan köras. Det innebär att den omvandlas till maskinnära kod som skiljer sig från källkoden. Detta kallas kompilerande språk .

Ett scriptspråk kompileras inte innan, utan koden där körs direkt när den exekveras. Då behövs någon form av scripttolk (interpretator) som kan omvandla koden till instruktioner. Detta kallas för interpreterande språk .

Vad är Javascript?

Ett scriptspråk som togs fram för att kunna **köra** kod på klienten, direkt i webbläsaren.

Första versionen kom 1995 i Netscape (föregångare till Firefox)

Förvaltas idag av ECMA som är ett standardiserings-organ.

Inte samma sak som Java, men det finns en del likheter.

Webbsidan- Tre lager som samverkar

Det som händer i en webbläsare när en sida visas består av 3 lager som samverkar.

HTML - beskriver strukturen på sidan dvs vilka rubriker som finns, hur menyer/navigeringen ser ut, vilka texter som skall stå på sidan.

CSS - beskriver layouten på sidan dvs färg och form

Javascript - interaktiviteten dvs så fort en sida inte skall vara statisk utan det skall hända något i webbläsaren. Det kan tex vara när användaren gör något på sidan.

Hur körs koden ?

Javascript togs från början fram för att köras i en webbläsare. Alla stora webbläsare har en inbyggd javascriptmotor (interpretator) som gör att det går att köra kod .

Detta förändrades 2009 när Node.js kom. Det är en runtimemiljö för att köra javascriptkod utanför en webbläsare . Den används oftast på webbservern.

Google har tagit fram en javascriptmotor till Chrome som heter V8. Den är open source dvs fri att använda. Den används även av Node.JS för att köra kod.

JS - Editorer

Finns en stor mängd editorer där det går att skriva och testa JavaScript-kod. Här är de vanligaste:

- Sublime Text
- Atom
- WebStorm
- Visual Studio Code
- JSFiddle

Versioner av Javascript

Det har hänt mycket sedan första versionen kom 1995. Inte minst har sättet som vi använder webben på har förändrats och vilken typ av lösningar som byggs.

Kan variera mellan olika webbläsare vilket stöd dom har för olika funktionalietet som finns i olika versioner

ECMAScript 10 senaste versionen som kom 2019

https://en.wikipedia.org/wiki/ECMAScript

Hur används Javascript?

Används ofta tillsammans med CSS för att ge en bättre användarupplevelse. Dessutom för att minska trafiken mellan klient och server.

Idag byggs ofta Single Page Applications där nästan all programkod ligger i webbläsaren och man bara kommunicerar med webbservern för att skicka/hämta data

Nackdelar är tex att det inte är typsäkert, fungerar ibland olika i olika webbläsare och att det kan vara svårt att debugga.

Ett språk i förändring

Tanken från början var bara att ha ett språk för att göra enklare saker i webbläsaren

Idag byggs den en annan typ av applikationer och det sätt som webben används på har förändrats

De senaste versionerna av Javascript innehåller därför mycket nya saker och språket förändras en hel del.

JavaScript kodbibliotek och ramverk

Det finns bibliotek med färdig JS kod som man kan ladda ned och använda sig av för att bygga lösningar.

För att ”slippa uppfinna hjulet på nytt” och snabbare bygga lösningar.

Bibliotek har ofta färdiga funktioner inom ett specifikt område.

Ramverk har ofta färdiga modeller för hur man ska bygga hela applikationer. Exempel på ramverk: AngularJS, React, Knockout, Ember, Vue.

Vad är en variabel?

När man programmerar arbetar man ofta med olika värden . Dessa måste lagras i programkoden och det görs i variabler. Tänk dig en behållare som det går att lägga saker i,

En variabel kan variera dvs värdet som den har kan ändras. Att sätta ett värde på en variabel kallas tilldelning .

Innan en variabel kan tilldelas ett värde måste den deklareras och få ett namn. Då avsätts ett visst minnesutrymme för att kunna hålla värden på variabeln.

Deklarera en variabel- var och let

En variabel måste deklareras i koden för att kunna användas. Det första man gör i koden är ofta att deklarera de variabler som skall användas,

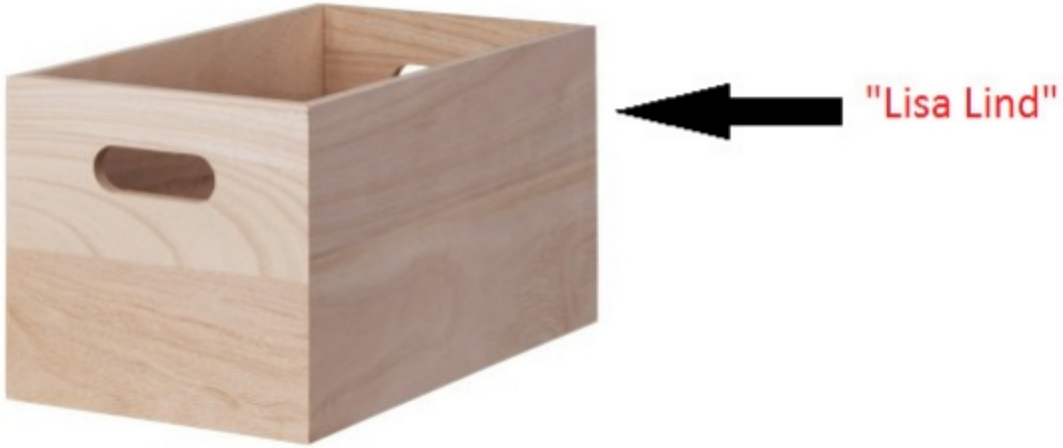
Detta kan göras med antingen nyckelordet var eller let . Deklarareras den med var blir den tillgänglig i hela koden.

Om let används blir tillgängligheten begränsad till den del av koden som den deklareras i.

Deklarera en variabel

Här deklareras en variabel och tilldelas samtidigt ett värde.
Det gör att den kan användas senare i programkoden.

```
var studentNamn= ”Lisa Lind” ;
```



Datatyper- Sträng, Nummer, Boolean

De värden som lagras i en variabel kan vara av olika typ. Javascript är ett otypat språk dvs det går inte att bestämma typen direkt när variabeln deklareras utan den får den typ som värdet den tilldelas har.

Det kan vara tex nummer som hanterar heltal och decimaltal

Strängar som hanterar alla typer av tecken som en text

Boolean som hanterar värden av typen sant/falskt tex rökare-ickerökare, har körtkort- har inte körkort, gift inte gift.

Deklarera en variabel

```
//Här deklareras en variabel och tilldelas ett strängvärde
var namn = "Lisa Lind";

//Här deklareras en variabel som är ett tal
var alder = 28;

//Här deklareras en variabel som är en boolean
var korkort = true ;
var rokare  = false ;
```

Undefined och null

När en variabel är deklarerad men inte har tilldelats ett värde har den värdet *undefined*

```
var namn ;      //Har värdet undefined
```

Värdet null är nästan samma sak. Det beskriver att en variabel saknar värde. Det kan vara att den inte är deklarerad eller att den har tilldelats värdet null.

```
namn;    eller  var  namn =  null ;
```

Aritmetiska operatorer

Operator	Description	Example
+	Addition	x = y + 2
-	Subtraction	x = y - 2
*	Multiplication	x = y * 2
/	Division	x = y / 2
%	Modulus (division remainder)	x = y % 2
++	Increment	x = ++y
		x = y++
--	Decrement	x = --y
		x = y--

Exempel

```
var mittTal= 2;

mittTal = 4 + 2;                //Ger värdet 6

mittTal = mittTal + 4; eller mittTal += 4;   //Ger också värdet 6

mittTal = 5 – 1;                //Ger värdet 4

mittTal = mittTal– 1;           //Ger värdet 1

mittTal = mittTal * 8;          //Ger värdet 16

mittTal = 8 / mittTal;          //Ger värdet 4
```

Exempel

För upp och nedräkning

```
var mittTal = 2;

mittTal ++ ;        //Ger värdet 3
mittTal -- ;        //Ger värdet 1
```

Plustecknet kan också användas för att sätta samman (konkatenera) strängar

```
var minString1 = ”Jag studerar på” ;
var minString2 = ”skolan” ;
var totalString = minString1 + ” ” + minString2 ;
//Ger värdet- Jag studerar på skolan
```

Konvertera datatyper

När en variabel tilldelas ett värde får den en typ. Det går att konvertera en variabel till en annan typ genom att tex använda funktionerna String() eller Number() .

```
var x = "10" ;
var y= x + 20;                  //Hanteras som strängar. Ger 1020
var z = Number (x) + 20;  //Hanteras som tal. Ger 30

var x2 =10;
var y2= x2 + 20;   //Hanteras som tal. Ger 30
var z2 = String (x2) + 20; //Hanteras som sträng ger 1020
```

Kommentarer

Används för att förklara en viss kodrad eller för att markera att vissa rader inte skall köras.

En kommentar kan gälla antingen en rad eller markera flera rader samtidigt

```
//Detta är en kommentar på en rad

/* Här skapas en
   kommentar
   över flera
   rader */
```

Logiska beslut

Om bussen är mer än 5 minuter försenad ta tunnelbanan annars ta bussen.

Om det finns mindre än 10 exemplar kvar av en vara beställ hem nya varor.

Om det regnar ta med paraplyet annars lämna det hemma.

Om en faktura inte är betalad efter 30 dagar skicka påminnelse

IF sats

En programmeringslösning innehåller i regel många logiska beslut.

För att kunna skapa logiska villkor för hur koden körs används IF-satser.

För att kunna skapa villkor i en IF sats används olika logiska operatorer. Vi skall titta på några av dem nu

Jämförelse operatorer

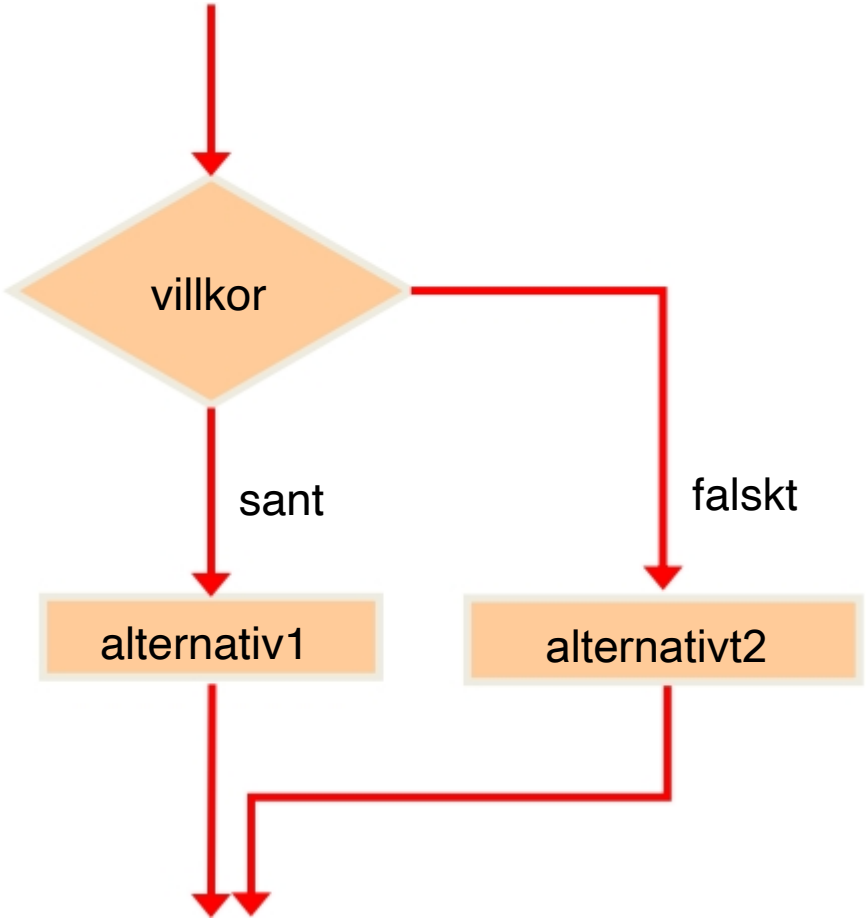
Operator	Beskrivning
<	Mindre än
>	Större än
<=	Mindre än eller lika med
>=	Större än eller lika med
==	Lika med
!=	Ej lika med

Logiska operatorer

Operator	Beskrivning
&&	OCH
||	ELLER
!	INTE

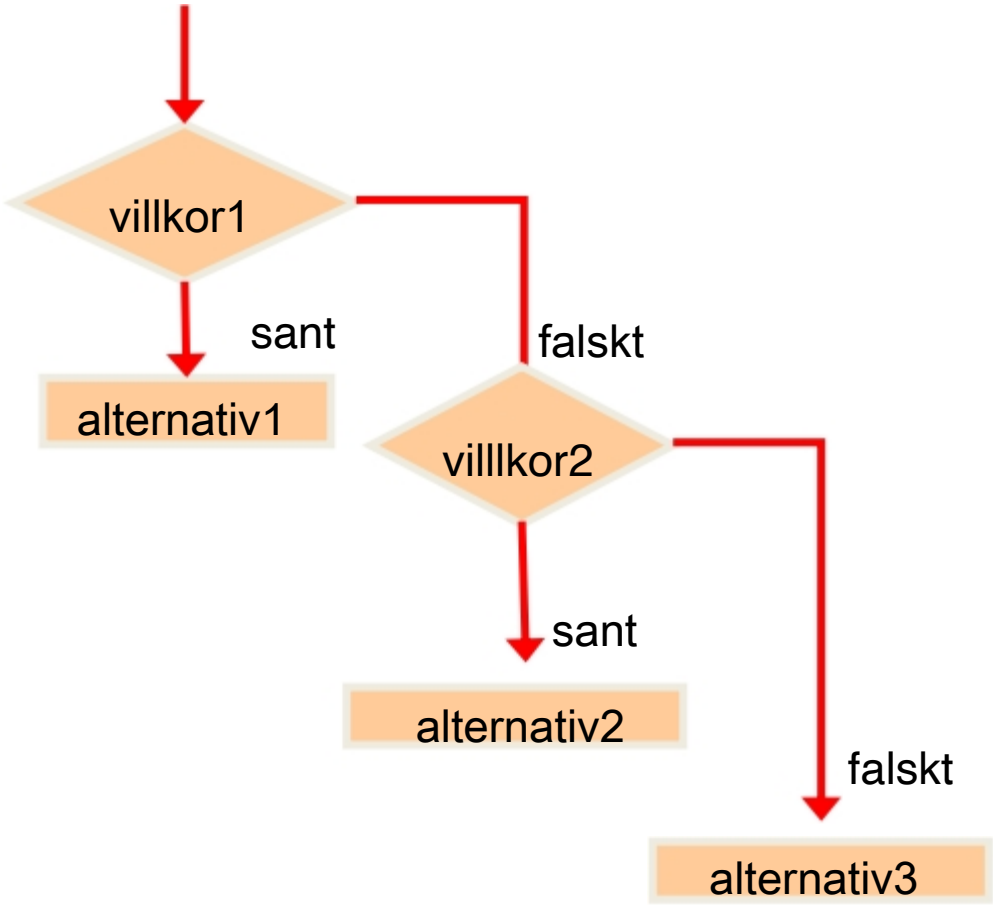
IF sats - grunden

```
if (villkor)
{
    alternativ1;
}
else
{
    alternativ2;
}
```



IF - ELSE IF - ELSE

```
if (villkor1)
{
   alternativ1;
}
else if (villkor2)
{
   alternativ2;
}
else
{
   alternativ3;
}
```



Kodblock

Kod som hör samman på något sätt läggs i sk kodblock. Ett kodblock har alltid en rubrik som talar om vad som finns inne i blocket. Ett exempel på det är villkoret i en IF sats.

Blocket börjar med detta tecken { och slutar med } . Mellan dessa tecken finns koden i blocket.

Rubriken i kodblocket talar om vad blocket innehåller. Den har aldrig ett semikolon efter sig .

Rader som finns inne i blocket anger kod som körs. Dessa rader har alltid ett semikolon efter sig .

Exempel

Här är ett exempel på en if-sats. Detta säger att om det är mindre än eller lika med 8 grader och vinter skall man ha mössa annars keps

```
var arstid = ”sommar” ;
var attHaPaHuvudet ;
var temperatur=15 ;

if (temperatur <= 8 && arstid == “vinter” )
{
      attHaPaHuvudet = “Mössa” ;
}
else
{
      attHaPaHuvudet = “Keps” ;
}
```

Logiska villkor med typkontroll

En if sats kan ha ett villkor som även kontrollerar om datatyperna matchar i villkoret. Det gör med operatorer som === eller ==!

Utan kontroll av datatyper

```
if (3 == "3" )          // ger true
if (3 == 3)             // ger true
```

I dessa fall kontrolleras även datatypen i villkoret

```
if (3 === "3" )         // ger false.
if (3 === 3)            // ger true
```

Switch

Ett alternativ till att skriva en IF sats om ett villkor har många värden är att göra en switch. Det testar många altivernativ på samma villkor. Här visas aktuell dag.

```
var day;

switch ( new Date().getDay()) {
      case 1:
            day = “Today is monday" ;
            break;
      case 2:
             day = “Today is tuesday" ;
             break;
}
```

Länkar

Bra sidor med mycket exempel och kod

http://www.w3schools.com/js/default.asp
http://www.tutorialsteacher.com/javascript

Grattis online böcker om Javascript

eloquentjavascript.net
https://github.com/getify/You-Dont-Know-JS

Online övningar och exempel

http://www.codecademy.com
http://www.w3resource.com/javascript-exercises/
https://www.freecodecamp.org/

Länkar

- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference
- https://www.youtube.com/playlist?list=PL46F0A159EC02DF82

Övningar

https://www.w3schools.com/js/exercise_js.asp

- **JS Variables**
- **JS Operators**
- **JS Data Types**

Länkar

https://www.w3schools.com/js/js_if_else.asp

https://www.tutorialspoint.com/javascript/javascript_switch_case.htm