

Frontend <Lektion class="sju">

Utbildare: Mahmud Al Hakim

NACKADEMIN

Lektionstillfällets mål

Mål med lektionen

- Att förstå JavaScript
- Att länka JavaScript till HTML
- Introduktion till objekt och metoder
- Syntax
- Satser (statements)
- Kommentarer
- Variabler och värde
- Datatyper (typ)
- Arrayer

NACKADEMIN

Kort summering av föregående lektion

Föregående lektion:

- Bootstrap
- CSS Mallar (Templates)

Frontend-teknologier

HTML



CSS



JS



HTML

Struktur

CSS

Utseende

JavaScript

Beteende

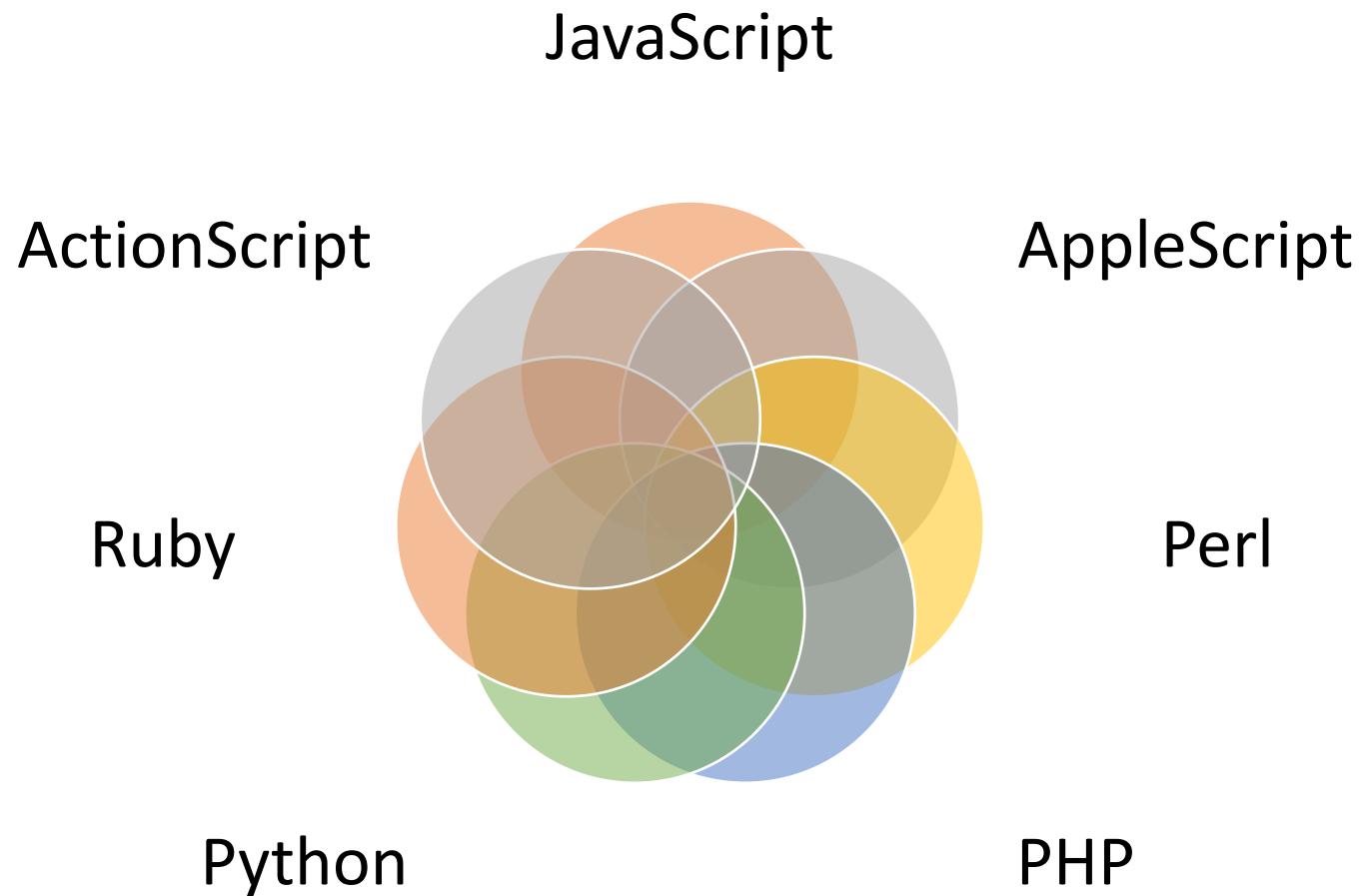
Olika slags programmering

- Programmering kan ske på olika abstraktionsnivåer.
- Närmast hårdvaran, förutom maskinkod, ligger assemblerprogrammering där man använder ett assemblerspråk.
- På nästa nivå kommer allmänna programspråk (engelska: general purpose programming languages)
Till exempel Java, C++, C# och JavaScript.
- Källkoden i dessa språk kompileras till maskinkod av en kompilator eller översätts (interpretaras) av en tolk.

Vad är skriptspråk?

- Skriptspråk är inom datavetenskap en benämning på "små" högnivåspråk inriktade på specialiserade uppgifter inom redan befintliga miljöer, i motsats till systemspråk, som används för programmering av tillämpningsprogram.
- Det finns ingen exakt avgränsning mellan systemspråk och skriptspråk, men skriptspråk är ofta interpreterande programspråk som använder dynamisk typning.
- Källa
<https://sv.wikipedia.org/wiki/Skriptspr%C3%A5k>

Exempel på skriptspråk



Vad är JavaScript?

- JavaScript är ett skriptspråk som används oftast tillsammans med HTML och CSS för att lägga till beteende och interaktivitet på webben.
- Koden körs/översätts direkt i webbläsaren (interpreteras) och behöver inte kompileras som till exempel är fallet för andra programspråk såsom C# och Java.
- Tolkning av JavaScript-kod sker i besökarens webbläsare.



JavaScript vs ECMAScript

December 1995 släppte Netscape sin webbläsare Navigator 2 som stödde skriptspråket Livescript.

Livescript utvecklades av Brendan Eich, som var anställd vid Netscape och sedan grundade Mozilla år 1998.

Efter ett samarbete med Sun (som utvecklade Java) ändrades namnet till JavaScript. Kort efteråt kom Microsoft med sin variant som kallades Jscript.

JavaScript och JScript skilde sig på några inbyggda objekt och funktioner vilket ledde fram till behovet av en gemensam standard för skriptspråket, standarden kom att kallas för ECMA-262, även känt som **ECMAScript**.

Att länka JavaScript till HTML

- Hur hittar webbläsaren JavaScript-kod?
- Mellan <script> och </script> placerar man sin JavaScript-kod.

```
<script>  
    Här skrivs JavaScript-kod!  
</script>
```

JavaScript i <head>

```
<html>
<head>
  <meta charset="UTF-8">
  <title>JavaScript i head</title>
  <script>
    Här kan du skriva JavaScript-kod
  </script>
</head>
<body>
</body>
</html>
```

JavaScript i <body>

```
<html>
<head>
  <meta charset="UTF-8">
  <title>JavaScript i body</title>
</head>
<body>
  <script>
    Här kan du skriva JavaScript-kod
  </script>
</body>
</html>
```

Ett första JavaScript-exempel

```
alert("Hello World!");
```

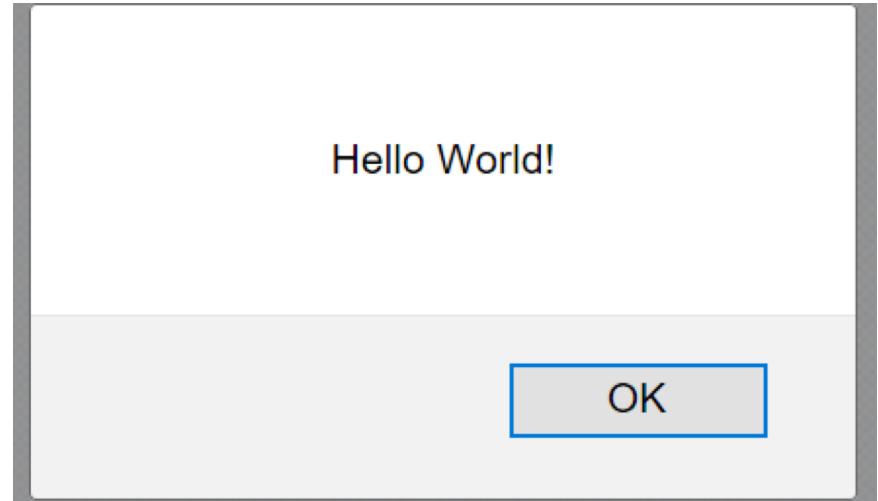
alert() är en funktion som visar ett meddelande på skärmen.

Besökaren måste klicka på "OK"-knappen på för att fortsätta.

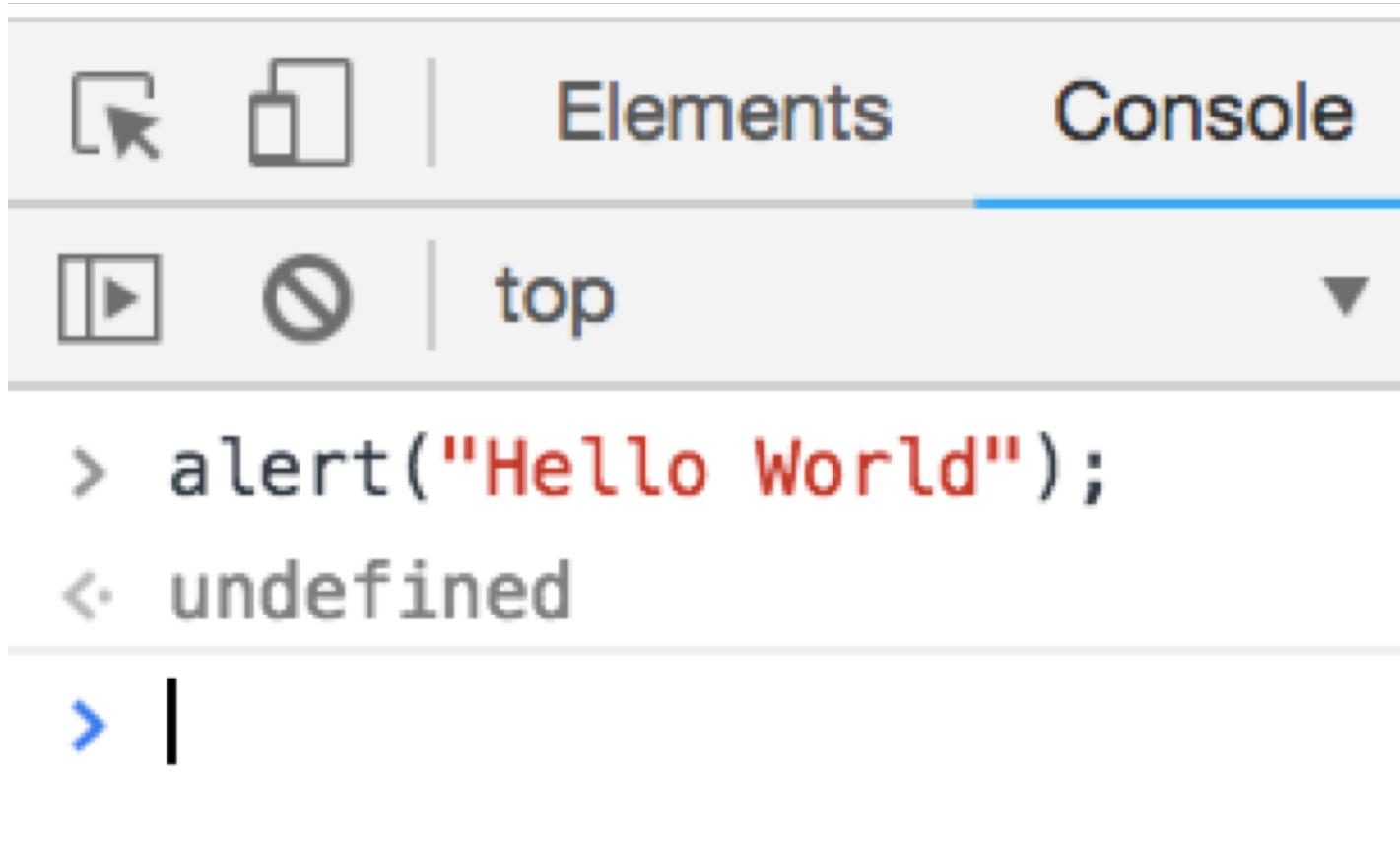
Typiskt användningsområde är om du vill försäkra dig att information kommit fram till besökaren.

Hello World!

```
<body>  
<script>  
    alert("Hello World!");  
</script>  
</body>
```

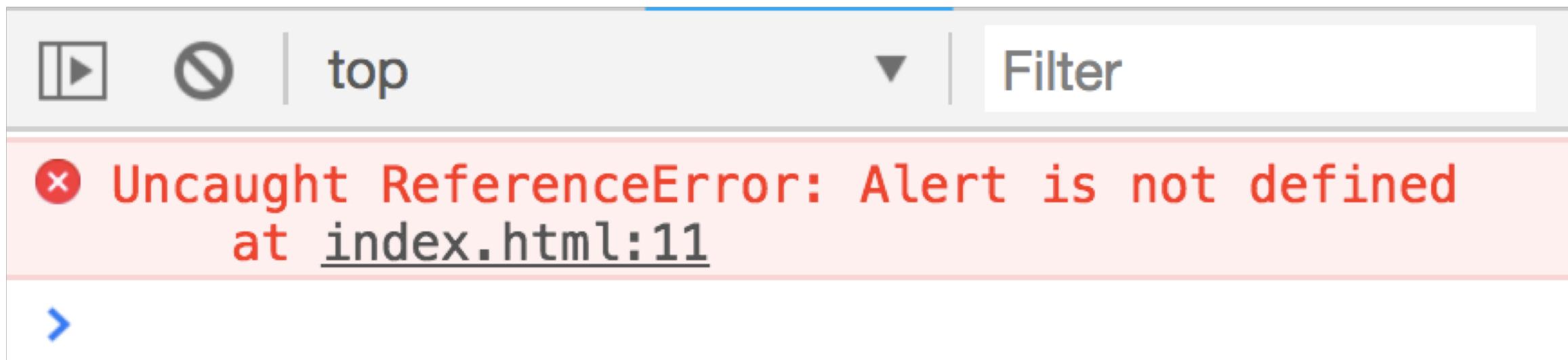


Starta JS-konsolen i Chrome DevTools
ctrl+shift+J (Windows)
cmd+alt+J (Mac)



OBS! JavaScript är Case Sensitive

- JavaScript är Case Sensitive (skiftlägeskänsligt)
d.v.s. språket är känslig för stora och små bokstäver!
- Exempel:
Alert och alert är två olika ord i JavaScript!



Viktiga grundregler

1. JavaScript kod skall läggas in mellan <script> och </script>
2. JavaScript är *skiftlägeskänslig (Case Sensitive)*
3. Lägg alltid in en text mellan raka citationstecken
(dubbla " " eller enkla ' ')

Externa JavaScript-filer

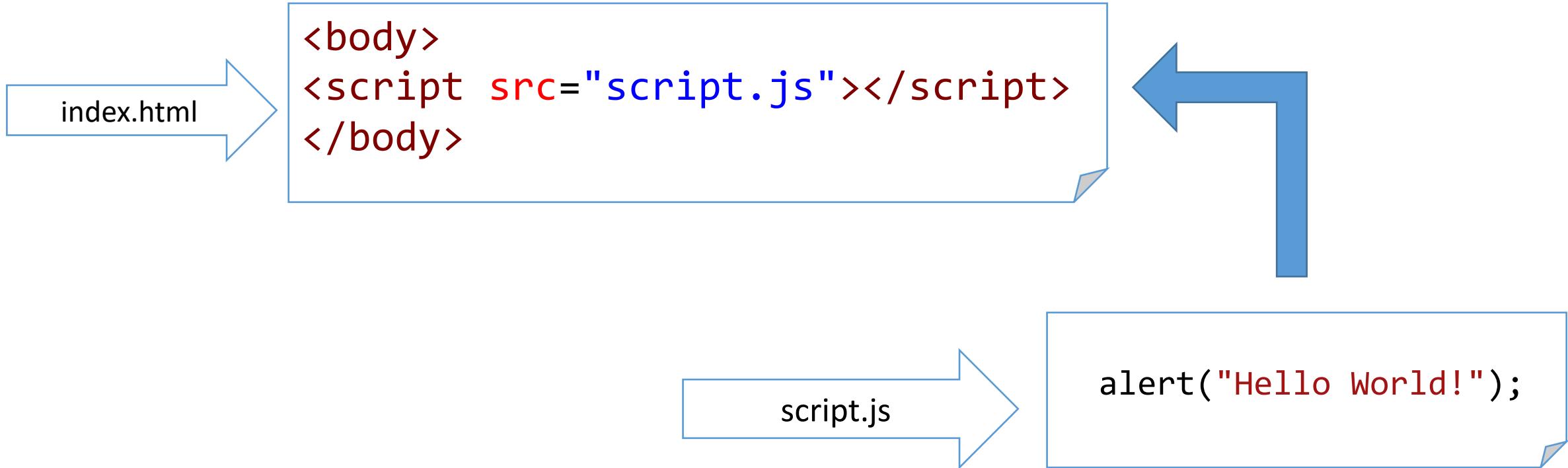
- Externa JavaScript-filer kan kopplas till flera HTML-dokument.
- JavaScript-kod skrivas i en separat fil med filnamnstillägget .js
(t.ex. script.js)
- Script-taggen med attributet src infogar skriptet i HTML-dokumentet:

```
<script src="script.js"></script>
```

OBS!

Man får inte länka till externa JavaScript-filer och skriva kod inne i script-taggen samtidigt.

Externa JavaScript-filer



Introduktion till objekt och metoder

- Objektorienterad programmering (Object Oriented Programming, OOP) är en programmeringsmetod i vilken ett program kan innehålla en varierande uppsättning objekt som interagerar med varandra.
- Den grundläggande idén med OOP är att vi använder objekt för att modellera saker som vi vill representera i våra program och ge ett enkelt sätt att komma åt metoder (funktioner).
- Här finns en lista på alla objekt som finns i JavaScript
<https://www.w3schools.com/jsref/default.asp>

Objektet Console

- Console är ett objekt som finns i JavaScript.
- Med hjälp av detta objekt kan vi bl. a. skriva ut information till webbläsarens konsol.
- En viktig metod i objektet är log()
- https://www.w3schools.com/jsref/met_console_log.asp
- Exempel

```
console.log("Hello world!");
```

Objektet document

- När ett HTML-dokument laddas i en webbläsare blir det ett dokument-objekt.
- Detta objekt kallas **document**.
- **write()** är metod i detta objekt.
- OBS! Metoden write() används enbart vid testning i webbläsaren.
- Exempel

```
document.write("Hello World!");
```

Metoden getElementById()

- `getElementById()` är en viktig metod som finns i objektet `document`
- Metoden hämtar ett HTML-element som har ett specifikt **id**
- Exempel: HTML-kod

```
<div id="demo"></div>
```

- Exempel: JS-kod

```
document.getElementById("demo");
```

Objektet Element

- Ett HTML-objekt representerar ett HTML-element t.ex. body eller div
- En viktig egenskap (property) i detta objekt är `innerHTML`

```
let el = document.getElementById("demo");
// OBS! el blir ett objekt av typen HTML-element
el.innerHTML = "<h1>TEST</h1>";
```

- En annan viktig egenskap i detta objekt är `textContent`
- ```
el.textContent = "OBS! Enbart text, ej HTML";
```

# Syntax

---

Syntax är språkets regler och grammatik

---

Syntax handlar om att skriva korrekt källkod

---

En duktig programmera måste känna till språkets syntax!

# Satser (statements)

- JavaScript-applikationer består huvudsakligen av satser med lämplig syntax.
- Enkla satser avslutas vanligen med semikolon men detta är inte obligatoriskt i JavaScript.
- JavaScript har "**Automatic semicolon insertion**".
  - \* se länken nedan
- Flera satser kan förekomma på en enda rad men då måste varje sats avslutas med ett semikolon.

\* [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Lexical\\_grammar#Automatic\\_semicolon\\_insertion](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Lexical_grammar#Automatic_semicolon_insertion)

# Satser – Exempel

```
alert("Hello World!")
```

Semikolon  
saknas här!  
Helt okej!

```
console.log("Hello World");
```

```
console.log("Hello 1"); console.log("Hello 2");
```

Obs!  
Semikolon  
obligatoriskt här



# Kommentarer

- Att kommentera källkod är en konst.
- Skriv i kommentaren VAD som görs och inte HUR det görs.
- Kommentera i en sammanhängande längre kommentar före ett avancerat block vad som görs.
- Undvik ”Papegoja -kommentarer”!

# Kommentarer – Exempel

```
// Detta är en kommentar på en rad

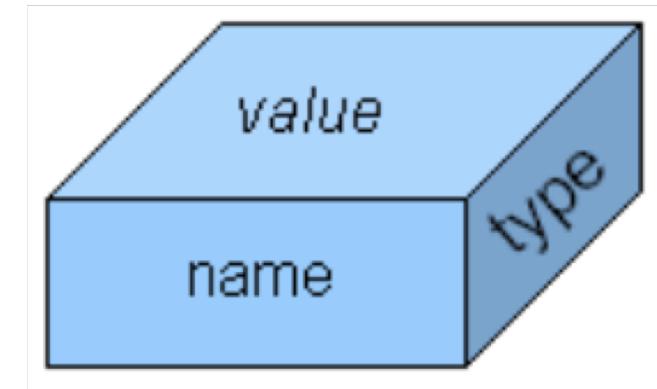
// Nedanstående sats är bortkommenterad
// alert("Hello World!");

alert("Hello World!"); // visar en alert
// Ovanstående kommentar är en "Papegoja-kommentarer"

/* Detta är en kommentar som sträcker sig
 över flera rader.
*/
```

# Variabler och värde

- En variabel är en platshållare för ett värde som kan ändras under programmets gång.
- I datorprogram använder man variabler för att lagra data.
- Data kan vara av olika slag (typer)  
t.ex. text eller tal.
- I JavaScript kan en variabel innehålla data av en valfri typ.
- Innan man använder en variabel i ett program måste man deklarera den.



# Deklarera en variabel

Deklarera en ny  
variabel med  
nyckelordet **let**

Du måste ange ett  
**variabelnamn**

**let namn;**

Du kan även skapa variabler med  
nyckelordet **var** (föråldrat)



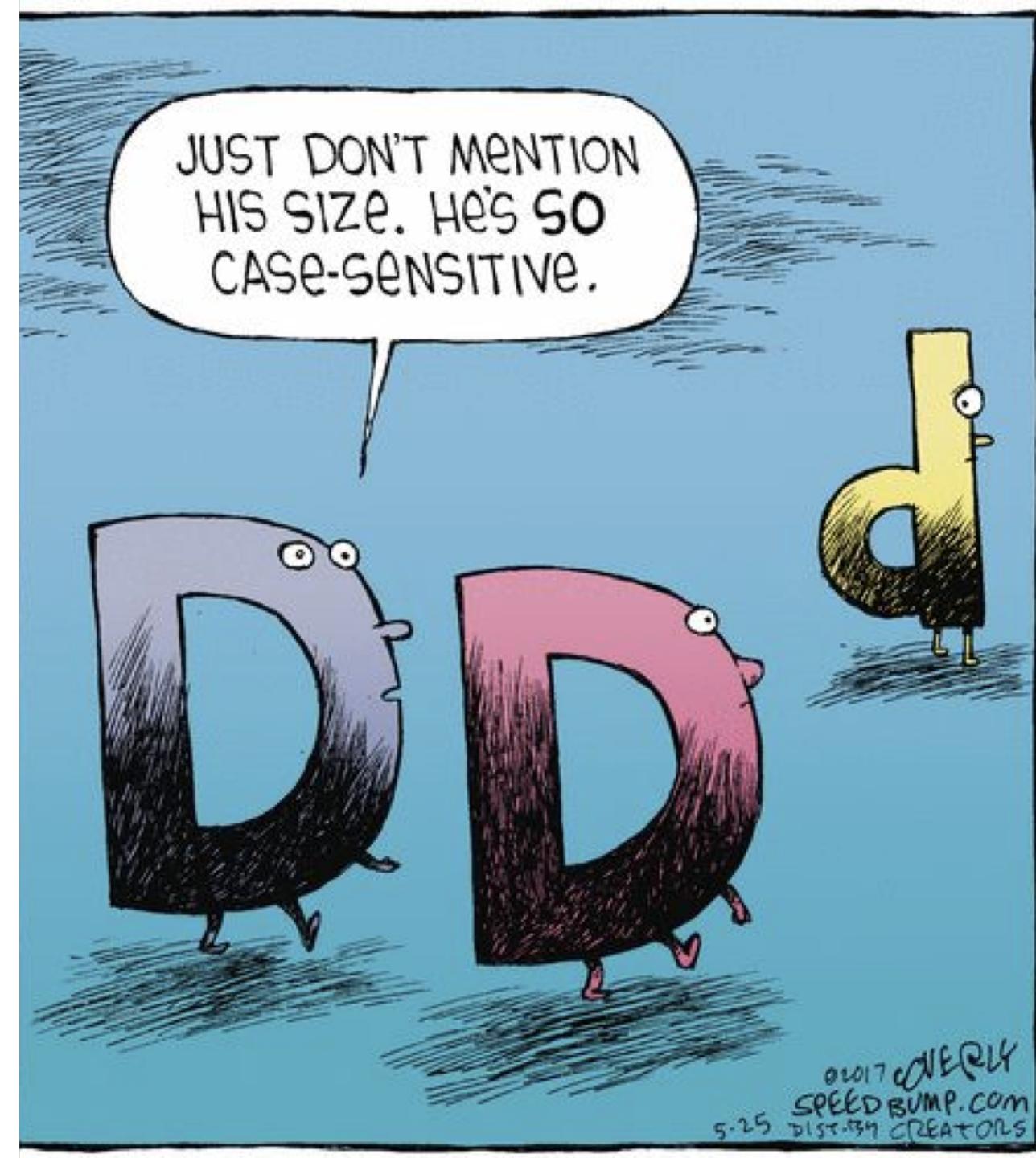
Du måste ge dina  
variabler **unika** namn

Kan du ge  
mig  
kassen?

Vilken?

# JavaScript är Case Sensitive

```
// OBS! tre olika variabler
let firstname;
let firstName;
let FIRSTNAME;
```



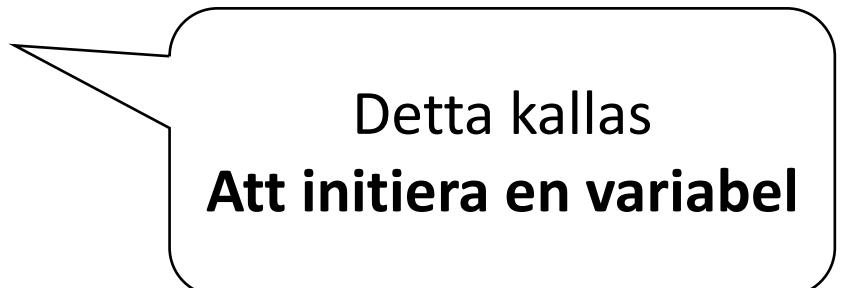
# Tilldelning

- Vill du ge en variabel ett värde använder du likhetstecknet (=) som kallas **tilldelningsoperator (assignment operator)**.
- Då du ger en variabel ett värde kallas det att du **tilldelar (assign)** variabeln ett värde t.ex.

```
namn = "Mahmud";
```

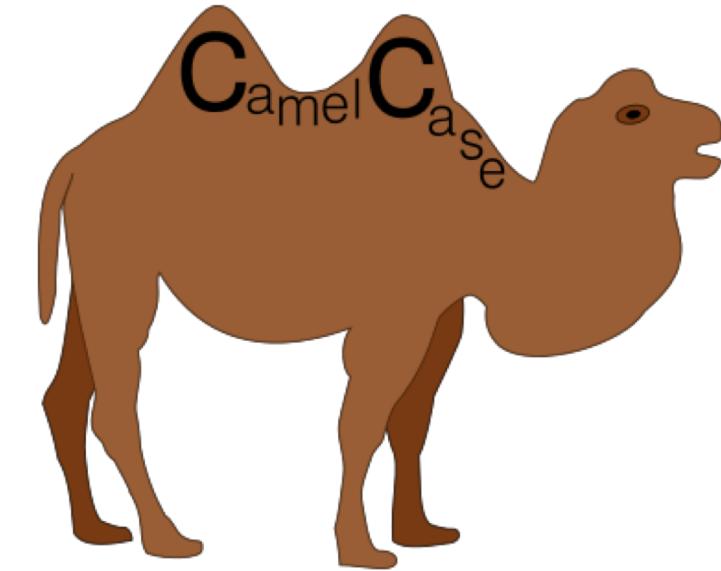
- Du kan deklarera och tilldela en variabel ett värde på en och samma gång t.ex.

```
let age = 45;
```



Detta kallas  
**Att initiera en variabel**

# camelCase (*kamelNotation*)



I JavaScript används camelCase för att skriva variabelnamn utan bindestreck eller mellanslag.

```
let firstname; // Ok men...
```

```
let firstName; // mycket bättre variabelnamn
```

# Variabler – Exempel

```
let firstName;
firstName = "Mahmud";
```

```
let lastName = "Al Hakim";
```

```
let tel , mobile;
tel = "08-1234567";
mobile = "073-123456";
```

# Variabler – Flera exempel

```
let firstName = "Mahmud" ,
lastName = "Al Hakim",
tel = "08-1234567",
mobile = "073-123456";
```

# \$ och \_

- Variabelnamn måste börja med en bokstav, \$ eller \_

```
let x = "x är en variabel";
```

```
let $ = "dollar kan användas i variabelnamn";
```

```
let _ = "underscore kan användas i variabelnamn ";
```

```
let $_TEST = "detta är ett giltigt variabelnamn";
```

```
let MAX_VALUE = 10;
```

— och .

- Variabelnamn får inte innehålla minustecken (-) eller punkt (.)

```
let first-name; // Unexpected token -
```

```
let first.name; // SyntaxError: Unexpected token .
```

// Mellanslag och komma då? Testa själv!

```
let first name;
```

```
let first,name;
```

# Siffror i variabelnamn

- Variabelnamn får innehålla siffror men inte påbörja med en siffra!

```
let x1;
```

```
let x2;
```

```
let 123; // OBS! SyntaxError!
```

# ReferenceError

- Vad händer om du försöker använda en **odeklarerad** variabel?

```
console.log(x);
```

---

✖ ➔ Uncaught ReferenceError: x is not defined  
at j.js:1

>

---

# Reserverade ord

- Variabelnamn får inte vara nyckelord t.ex. **var** eller reserverade ord t.ex. **super**
- Här finns en lista över alla reserverade ord

[https://www.w3schools.com/js/js\\_reserved.asp](https://www.w3schools.com/js/js_reserved.asp)

- OBS! Var försiktig med att använda namn på inbyggda JavaScript-objekt t.ex. console

```
let console = "TEST"; // OBS!!!
console.log("TEST"); // console.log is not a function
https://www.w3schools.com/jsref/default.asp
```

# Tips! JavaScript variable name validator

<https://mothereff.in/js-variables>

A screenshot of a web browser window showing the "JavaScript variable name validator" tool. The page title is "JavaScript variable name validator". The URL in the address bar is "https://mothereff.in/js-variables". The main content area contains the heading "JavaScript variable name validator", a descriptive paragraph about the tool, and a form where the user has entered the identifier "σ\_σ". A green button labeled "permalink" is visible next to the input field. Below the input field, a message states "That's a valid identifier according to ECMAScript 6 / Unicode 8.0.0." At the bottom, there is a footer note: "Made by [@mathias](#) — [fork this on GitHub!](#)".

JavaScript variable name validator

Wondering if you can use a given string as a variable name in JavaScript? [Learn how it works](#), or just use this tool.

Enter a variable name: [permalink](#)

σ\_σ

That's a valid identifier according to ECMAScript 6 / Unicode 8.0.0.

Made by [@mathias](#) — [fork this on GitHub!](#)

# Datatyper (typ)

## Vad är en datotyp?

- En datotyp (eller bara **typ**) är ett attribut för data som berättar för datorn (och programmeraren) vilken sorts typ data bär på t.ex. tal eller text.

## Primitiva datatyper

- Med **primitiva datatyper** menas de grundläggande typer som tillhandahålls direkt av programspråket.

## Dynamiska typer (dynamic typing )

- JavaScript har **dynamiska typer**, vilket innebär att samma variabel kan användas för att lagra olika typer.

# Primitiva datatyper i JavaScript

String

Number

Boolean

Undefined

Null

# Typen string

```
let firstName = "Mahmud";
let lastName = 'Al Hakim';
console.log(firstName,lastName);

console.log("It's nice");
```

# Typen number

```
let price = 5.5;
let quantity = 10;
let total = price * quantity;
console.log(total);
```

# Typen boolean

```
let ok = false;
let again =true;
if(again)
 console.log("Fortsätt...");
```

# Typen undefined

```
// En variabel som har deklarerats
// men inte tilldelats ett värde
// får ett odefinierat värde (undefined)
let name;
console.log(name); // ger undefined
```

# Typen null

```
let nothing = "Something"
nothing = "";
console.log(nothing);
nothing = null;
console.log(nothing);
// OBS!
// En variabel med värdet "" är inte samma sak som NULL
// Värdet "" är en tom sträng medan null representerar ingenting.
// Förväxlingen kan i vissa fall leda till logiska fel!
```

# Inmatningsfält

- Metoden **prompt()** finns i objektet window (enbart i webbläsare).
- Exempel

```
let person = prompt("Please enter your name");
alert("Hello " + person + "!");
```

- OBS! Viktigt  
Prompt returnerar en sträng om man trycker på OK, annars null  
[https://www.w3schools.com/jsref/met\\_win\\_prompt.asp](https://www.w3schools.com/jsref/met_win_prompt.asp)

# Typkonvertering från string till number

```
let tal1 = prompt("Ange tal 1");
let tal2 = prompt("Ange tal 2");
let summa = tal1 + tal2;
alert("Summa: " + summa);
```

```
// OBS! Vi måste konvertera string till number
summa = Number(tal1) + Number(tal2);
alert("Summa efter konvertering: " + summa);
```

[https://www.w3schools.com/Js/tryit.asp?filename=tryjs\\_global\\_number](https://www.w3schools.com/Js/tryit.asp?filename=tryjs_global_number)

# Övning 1

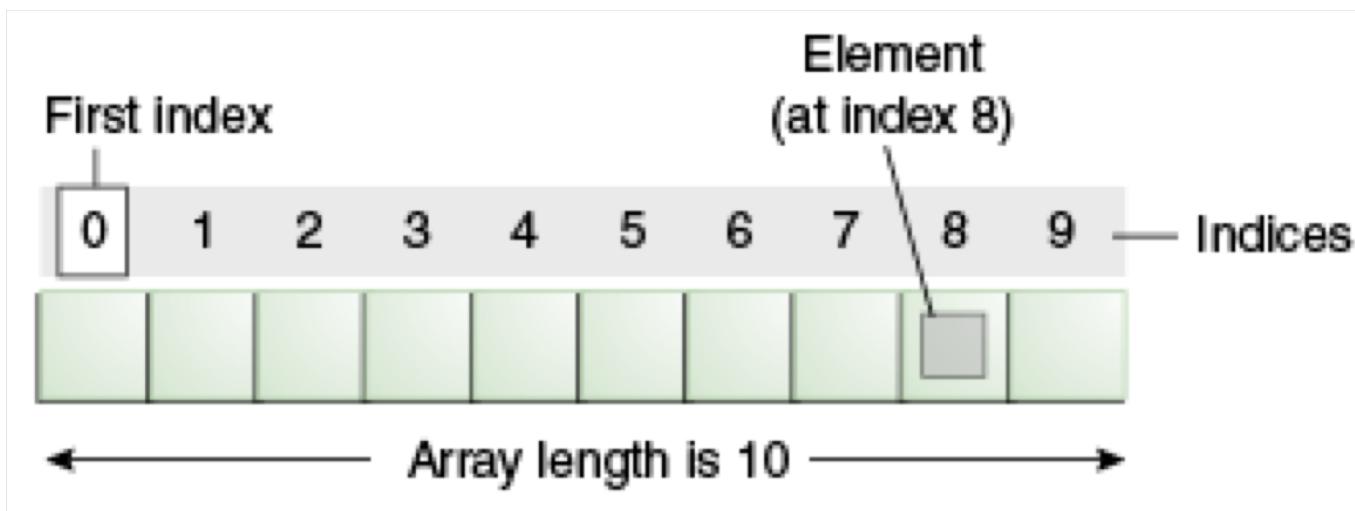
- Skapa ett program som ber användaren mata in 3 tal via prompt.
- Programmet ska beräkna summan av dessa tal.
- Visa resultatet i en alert-ruta.

# Övning 2

- Skapa ett program som ber användaren mata in 2 tal via prompt.
- Programmet ska beräkna medelvärdet av dessa tal.
- Visa resultatet i en alert-ruta.

# Vad är en array (fält)?

- En array eller ett fält är en datastruktur som består av en samling av element som identifieras med ett eller flera heltaliga index.
- Till skillnad från en vanlig variabel som bara innehåller ett värde kan ett fält alltså innehålla ett godtyckligt antal värden.



Bildkälla: <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html>

# Skapa en array

## 1. Array literal

```
let cars = ["Saab", "Volvo", "BMW"];
```

## 2. Array constructor

```
let cars = new Array("Saab", "Volvo", "BMW");
```

# Indexering

- Adressen till varje element kan räknas ut via dess index.

```
let cars = ["Saab", "Volvo", "BMW"];
console.log(cars[0]);
console.log(cars[1]);
console.log(cars[2]);
```

# Ändra ett element

```
let cars = ["Saab", "Volvo", "BMW"];
cars[0] = "Opel";
console.log(cars);
```

# Antal element

```
let cars = ["Saab", "Volvo", "BMW"];
console.log(cars.length);
```

# Sortera en array

```
let cars = ["Saab", "Volvo", "BMW"];
console.log(cars);

cars.sort();
console.log(cars);
```

# Övning 1

1. Skapa en array som innehåller en lista över dina kurser.
2. Skriv ut alla kurser i konsolen i sorterad ordning.
3. Skriv därefter ut antal kurser.

# Övning 2

1. Skapa en array som innehåller en lista över dina favoritböcker.
2. Skriv ut den första boken.
3. Skriv ut den sista boken.
4. Skriv ut alla böcker i konsolen i sorterad ordning.
5. Skriv därefter ut antal böcker.

# Summering av dagens lektion

- Vi har gått igenom
  - Att förstå JavaScript
  - Att länka JavaScript till HTML
  - Introduktion till objekt och metoder
  - Syntax
  - Satser (statements)
  - Kommentarer
  - Variabler och värde
  - Datatyper (typ)
  - Arrayer
- Reflektioner kring dagens lektion?

**NACKADEMIN**

# Framåtblick inför nästa lektion

**Under nästa lektion kommer vi att jobba med**

- Funktioner
- Parametrar och Argument
- Selektioner
- Iterationer
- Arbeta med objekt
- Egenskaper och Metoder