JAVASCRIPT

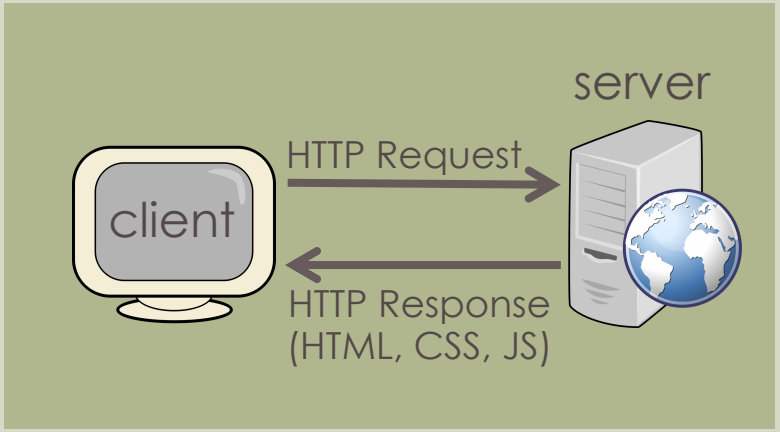
Beteende

OVERVIEW

- HTML Formulär
- JavaScript
- Syntax
- Events
- DOM
- Validering av fomulär

WEBBPROGRAMMERING PÅ KLIENTSIDAN

- Skriptspråk som körs på klientsidan (klientskript)
  - Koden exekveras i webbläsaren efter att sidan har skickats från servern
- Kontrollera/manipulera en sidas innehåll utan att behöva skicka tillbaka till servern
- Svara på ”actions” från användaren som inmatning av text, klick och knapptryckningar



HTML FORMULÄR

- Används för att ta emot information från användaren och skickar den till en webbserver
- Uppbyggd av en grupp av gränssnittskontroller
  - T.ex. textfält, checkboxar, radioknappar etc.
- Informationen från användaren skickas till servern som
  - URL variabler i en **query string** (get)
    - http://www.google.se/search?**q=html+formulär**
    - http://..../formular_ex.html?**user=namn&pwd=12345**
  - HTTP post request (post)

HTML FORMULÄR

- Formulärkontroller, text och övriga element placeras inom `<form>` taggen
- Egenskapen `action` är åtgärden som utförs när formuläret skickas
  - Den ger webbadressen till den sida som kommer att behandla formulärdata

```
<form action="URL">
    <div>
        Formulärkontroller
    </div>
</form>
```

INPUT KONTROLLEN

- `input` är det vanligaste formulärelementet
  - Används för att skapa många kontroller
  - Inline element och måste öppnas/stängas i samma tag <input ... />
- `name` attributen definierar namnet på parametern som ska skickas till webbservern
- `value` ger en initial text värde till kontrollen
- `type` anger typ av kontroll
  - `Button`, `checkbox`, `color`, `date`, `file`, `hidden`, `password`, `radio`, `reset`, `search`, `submit`, `text`, ...

Exempel: js_formular_1

FORMULÄRKONTROLLER

`<input>`	Input kontroll
`<textarea>`	Flera raders textfält
`<label>`	Etikett för en input kontroll beskriver vad som ska anges i fältet
`<fieldset>`	Gruppera formulärkontroller
`<legend>`	Anger rubrik till en fieldset
`<select>`	Definierar en drop-down lista
`<option>`	Definierar ett värde i drop-down listan
`<optgroup>`	Grupperar värden i en drop-down lista
`<button>`	Klickbar knapp
`<datalist>`	Definierar ett set av förbestämda option element som är tillgängliga för input kontroller
`<output>`	Definierar resultatet av en beräkning

Exempel: js_formular_2

JAVASCRIPT

- Skriptspråk: tolkat programmeringsspråk
  - Exekveras i webbläsare av javascriptsmotor
- Används för att göra webbsidor interaktiva
  - Ändra innehåll och presentation
  - Validera formulär
- Händelsestyrd (event-driven) programmering
- Stödjs inte identiskt av alla webbläsare

JAVASCRIPT

- ”Avslappnade” syntaxregler
  - Svagt typat
  - Variabler behöver inte deklareras
  - Fel är ofta tysta
- Inkluderas i en webbsida och integrerar med sidans HTML/CSS innehåll

INKLUDERA JAVASCRIPT KOD

- Direkt i HTML sidans `head` eller `body` innanför `script` taggar
  - Bör undvikas! → Separation av innehåll, presentation och beteende!

```
<script> alert("message"); </script>
```

- I separat `.js` fil och inkluderas via `script` taggen
  - `script` taggen placeras helst i HTML sidans `head`
  - Alternativt längst ner i HTML sidans `body`

```
<script src="filnamn.js" type="text/javascript"></script>
```

Exempel: js_include

JAVASCRIPT VARIABLER

- Deklareras med `var`
  - Kan deklareras implicit genom att tilldela de ett värde
- Svagt *typade* och dynamiskt definierade
  - Har typer men de behöver inte anges vid deklaration
  - Typen bestäms eller konverteras beroende på värdet som variabeln tilldelas
- 8 huvudtyper:
  - `Number`, `Boolean`, `String`, `Array`, `Object`, `Function`, `Null`, `Undefined`
  - Kan ta reda på typen genom `typeof` operator

```
var namn = "John";
var antal = 2;
procent = 0.5;
```

JAVASCRIPT FUNKTIONER

- Funktioner som i C++

```
function funktionsNamn( parameter1, parameter2, ...)
{
    kod som ska evekveras;

    return värde; /* ej obligatoriskt */
};
```

```
var x = mult(4, 3);

function mult(a, b) {
    return a * b;
}
```

- Funktioner kan också vara anonyma och läggas in direkt i variabler, objekt eller event

JAVASCRIPT OBJEKT

- Objekt är behållare av värden
  - Egenskaper
  - Metoder

```
var namn = {
    attributNamn: värde,
    ...,
    metodNamn: function(parametrar) {
        kod som ska exekveras;
    }
};
```

```
var person = {
    firstName: "Sven",
    lastName: "Svensson",
    age: 40,
    getAge: function() {return this.age;}
};
```

Exempel: js_syntax

JAVASCRIPT SCOPE

- Set av variabler, objekt, funktioner som jag kan nå
  - Lokala variabler nås enbart inuti funktioner
  - Globala variabler nås från var som helst i JS koden
- Variabler deklarerade utanför funktioner blir global
  - Nås även från kod i andra .js filer
  - Obs! om samma variabelnamn → variabeln skrivs över

```
var a = 3; //global variabel
function test() {
    a = 2*2;
    b = 6; // implicit deklaration (variabeln har inte deklarerats med var)
    var c = 2; //lokal variabel
    // inuti funktionen: a=4, b=6, c=2
}
test();
// utanför funktionen: a=4, b=6, c=undefined
```

Exempel: js_scope

HÄNDELSESTYRD PROGRAMMERING

- JavaScript program svarar på *actions* från användaren (event-driven)
  - Ingen `main`
- Events
  - Sidan har laddats färdigt
  - musklick, tangenttryck etc.

JAVASCRIPT SYNTAX

- Kommentarer
  - `//` kommentar
  - `/*` kommentar `*/`
- Skriva ut i konsol
  - Bra vid debugging & inspektera koden

```
var namn = "John";
console.log(namn);

console.log("John");
```

JAVASCRIPT SYNTAX

- String
  - `""` eller `''`
  - Metoder: `charAt`, `indexOf`, `replace`, `split`, `substring`, `capitalize`, `toLowerCase`,...
  - Egenskaper: `length`
  - *Concatenate* med **+** : `2+1= 3` MEN `"2"+1="21"`
  - *Escape* med \ : `"Han heter \"John\""`
  - Nästla med `'"'`: `'Han heter "John"'`
  - Konvertera till sträng med: `""` eller `''`

JAVASCRIPT SYNTAX

- Number
  - Samma typ för heltal (`int`) och reella tal (`double`)
  - Operatorer: `+`, `-` , `*`, `/`, `%`, `+=`, `-=`, `*=`, `/=`, `%=`
  - Konvertera med: `parseInt`, `parseFloat`
- `Math` objektet ger tillgång till matematiska metoder och egenskaper
  - `min`, `max`, `abs`, `random`, `ceil`, `sqrt`, ...
  - `PI`, `E`

JAVASCRIPT SYNTAX

- Boolean
  - `true`, `false`
  - "Falska" värden: `0`, `0.0`, `NaN`, `""`, `null`, `undefined`
- Logiska operatorer
  - >, <, >=, <=, `&&`, `||`, `!`, ==, `!=` (~ samma som i C++)
    - Konverterar till "rätt" typ vid jämförelser
    - "3.0" == 3 → true
    - 4.5 >= "3" → true
  - ===, `!==` : strikta operatorer
    - Konverterar ej typen, kolla både typ och värde av variabler
    - "3.0" === 3 → false

JAVASCRIPT SYNTAX

- Många satser och loopar samma som i C++
  - `if/else` satser
  - `for` loopar
  - `while`, `do/while` loopar
  - `break` och `continue`
- Arrays
  - initieras på 2 sätt

```
var minarray1 = ["element1", "element2", "element3"];
//eller
var minarray2 = [];
minarray2[0] = "el1";
minarray2[3] = "el2";
minarray2[5] = "el3";
```

Exempel: js_functions

HÄNDELSESTYRD PROGRAMMERING

- JavaScript program svarar på HTML event
- HTML element kan inkludera JavaScript funktioner som event hanterare (event handlers)

```
<html-element event=“ function(); “>...</html-element>
```

- Exempel på event
  - onload: webbsidan är färdigladdad
  - onclick: HTML element klickas
  - onmouseover: mus över elementet
  - onsubmit: när <submit> knapp klickas

Lista över HTML event: http://www.w3schools.com/jsref/dom_obj_event.asp

HÄNDELSESTYRD PROGRAMMERING

1. Givet ett html element (t.ex. knapp)

```
<button> Tryck här! </button>
```

2. och en matchande event (t.ex mus-klick)

```
onclick
```

3. Skriv en funktion som ska exekveras när knappen trycks

```
function knappTryck() {
    alert("Hej!");
    alert("Vad heter du?");
}
```

4. Länka funktionen till event och knapp

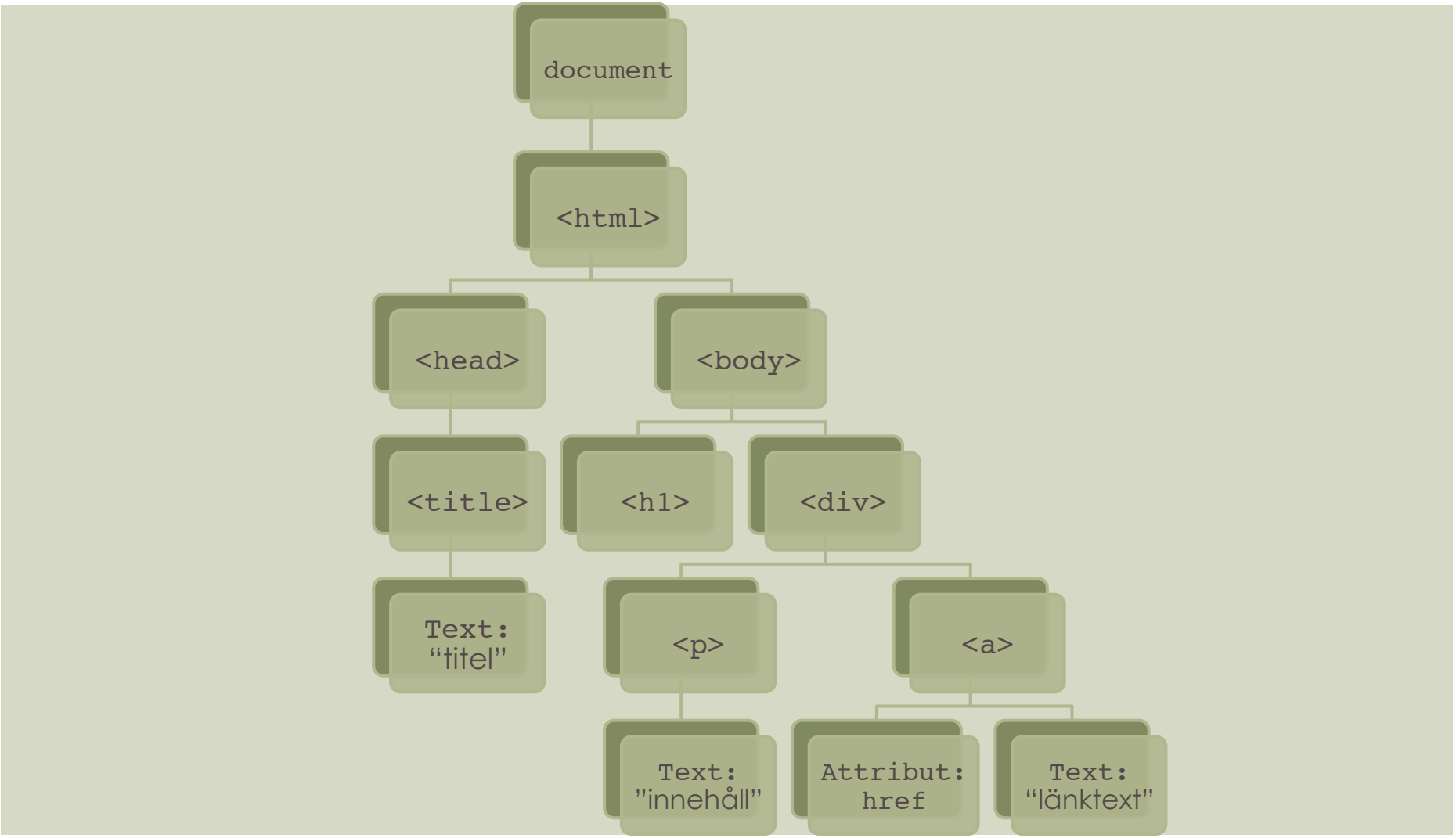
```
<button onclick="knappTryck();"> Tryck här! </button>
```

Exempel: http://jsfiddle.net/tnmk30/9zkmjxhv/

DOM

- Document Object Model
- Uppsättningen av JavaScript (JS) objekt som representerar innehållet (div, p, h etc.) för en HTML sida
- En DOM objekt per HTML element
- Via JS kod kan man interagera med DOM objekt och dynamiskt ändra på HTML element
  - Byta utseende, status, innehåll

DOM


document
<html>
<head>
<body>
<title>
<h1>

Text:
“titel”
<p>
<a>
Text:
”innehåll”
Attribut:
href
Text:
“länktext”

ACCESS DOM OBJEKT

- **`document`** objektet representerar hela html dokumentet och används för att komma åt alla andra DOM objekt.
- Åtkomst DOM objekt via HTML elementens **`id`** attribut

```
<button id="mybtn" onclick="knappTryck();”>Tryck mig</button>
```

- **`getElementById`** metoden accepterar en id sträng och returnerar motsvarande DOM objekt eller `null`
- DOM objektets attribut kan sedan manipuleras via ***objectNamn.attributNamn***

```
var button_obj = document.getElementById("mybtn");
button_obj.innerHTML = "Tack!”;
```

Exempel: http://jsfiddle.net/tnmk30/LwmkL3Lh/

DOM OBJEKT EGENSKAPER

- Text innehåll kan hämtas/ändras oftast med
  - `innerHTML`: ändrar text eller HTML taggar i DOM noder
  - `value`: ändrar värden i *widgets* (t.ex. textfält)

- Vanliga egenskaper
  - `tagName`: Elementets HTML tagg
  - `className`: Elementets CSS klass
  - `innerHTML`: Elementets Innehåll
  - `src`: URL adressen av en bild

```
<button id="mybtn" onclick="bytKnappText();"> Tryck
mig </button>
```

FIXA STILEN MED DOM

- `style` egenskapen gör att man kan ändra på CSS regler för ett element
- Innehåller alla attribut som finns i CSS
- Attribut namnen är samma men istället för gemener och bindestreck (-) skrivs namnen ihop med *kamelNotation*:
  - background-color → backgroundColor
  - font-family → fontFamily
- Värdena som ska tilldelas skrivs som strängar
- Enheter angivs som i CSS

Exempel: js_events

VALIDERA FORMULÄRDATA

- Använd JavaScript för att validera den information som användaren har matat in i en formulär
- Skriv en funktion som körs när `<submit>` knappen trycks och `onsubmit` händelsen anropas
- Funktionen borde kontrollera om de inmatade värdena har rätt typ och format och returnera `true/false`
- Bara om `true` ska informationen skicka till webbservern

Exempel: js_validering

ATT TÄNKA PÅ: JAVASCRIPT

- Vissa har JavaScript avaktiverat på sina webbläsare
- Använd inte JavaScript till sånt som är viktigt
- Använd **`noscript`** element för att bestämma innehåll som kommer att visas i det här fallet
- Sidan borde vara användbar även utan JavaScript

ATT TÄNKA PÅ: JAVASCRIPT

- Separera mellan
  - Innehåll → HTML
  - Presentation → CSS
  - Beteende → JavaScript
- Deklarera alltid variabler explicit med var
- Kolla om en variabel har ett värde eller är `undefined` innan du använder den
- Använd semikolon även fast det funkar utan

LÄNKAR TILL MATERIAL/LÄSNING

- HTML formulärtaggar:
  http://www.w3schools.com/html/html_forms.asp
- W3schools JavaScript tutorial:
  http://www.w3schools.com/js/default.asp
- W3schools JavaScript exempel:
  http://www.w3schools.com/js/js_examples.asp
- HTML DOM event:
  http://www.w3schools.com/jsref/dom_obj_event.asp
- HTML DOM objekt egenskaper och metoder:
  http://www.w3schools.com/jsref/dom_obj_all.asp

På måndag skickar vi data!

TP1 kl. 09:15