



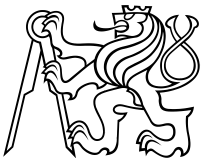
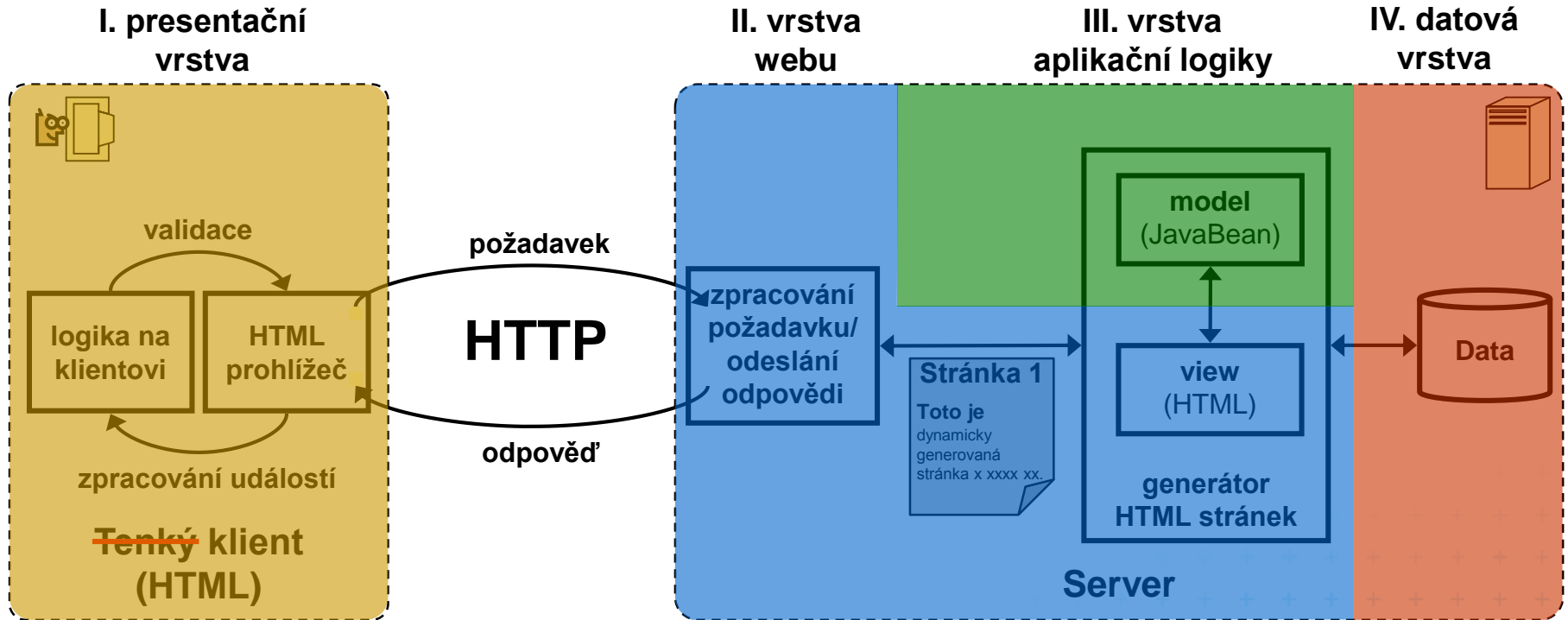
**DCGI**

**KATEDRA POČÍTAČOVÉ GRAFIKY A INTERAKCE**

# Logika na straně klienta, skriptovací jazyky

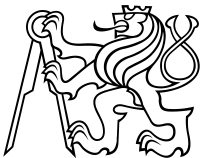
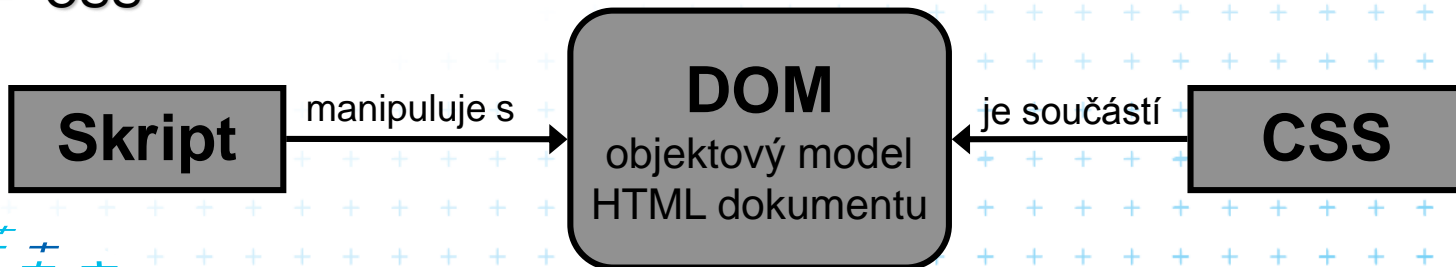
**Martin Klíma**

# Architektura web aplikace: dynamický web



# Co je to DHTML?

- **Cíl:** HTML dokument reaguje na události bez nutnosti spolupráce se serverovou stranou web aplikace
  - změna obsahu a prezentace stránky, validace formulářů, atd.
- **Řešení:** umožnit vytvářet klientský program manipulující s obsahem HTML dokumentu
- **DHTML je směs následujících technologií:**
  - DOM (Document Object Model)
  - klientské skriptování
  - CSS



# Charakteristika JavaScriptu

- skriptovací jazyk
  - interpretovaný klientem
  - nezávislý na platformě (nutné pro použití všude)
  
- skripty v prohlížeči pracují v definovaném prostředí
  - omezené možnosti (sandbox)
  - datový model DOM
  - UI+prezentace dat: řeší HTML prohlížeč
  - události

# K čemu skripty slouží a k čemu ne?

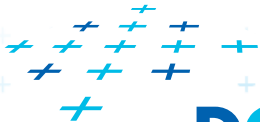
## ■ ANO

- kontrola a předzpracování vstupních dat (formuláře)
- manipulace s malými objemy dat
- dynamické změny obsahu HTML
  - událost => změna HTML elementu (např. obrázků, položek ve formuláři), generování HTML do nových oken prohlížeče
- AJAX (samostatná přednáška)

## ■ NE

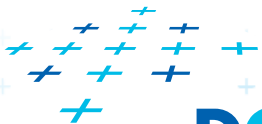
- spouštění aplikací na klientském počítači
- manipulace se soubory a adresáři

**POZOR!** Není-li zaručeno, že prohlížeč všech uživatelů umí spouštět skripty, vaše stránky by měly fungovat i bez nich.

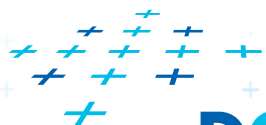
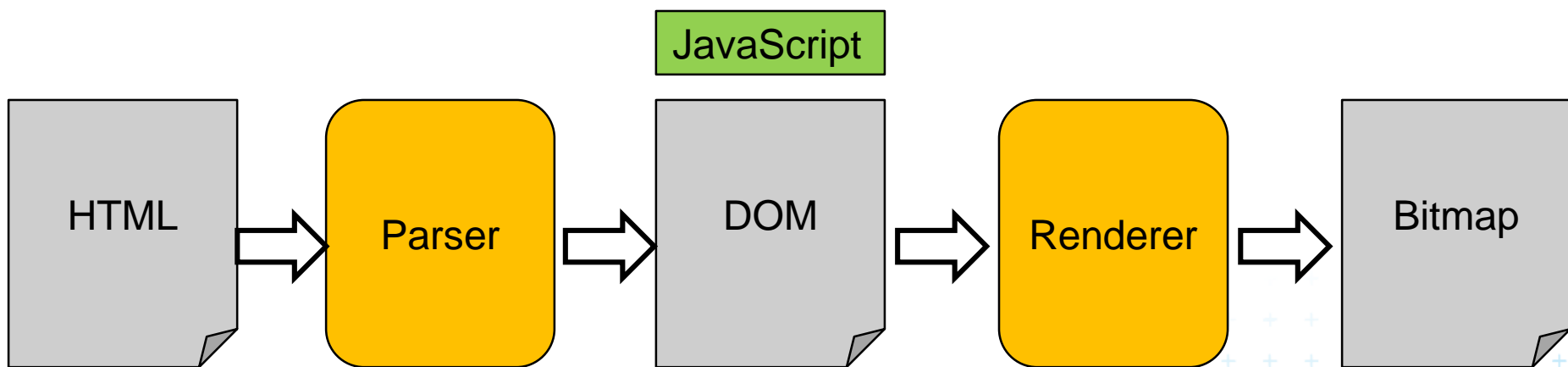


# Vývoj JavaScriptu

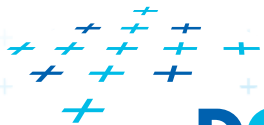
- Dosti divoká historie
- Vede k řadě nekompatibilit
- V současnosti se situace stabilizuje
  
- Řada nových rozšíření v souvislosti s HTML5
  - Přístup k vykreslené stránce
  - Síťová komunikace
  - Periferie: repro/mikrofon, OpenGL, senzory (mobil), GPS
  - Lokální ukládání dat
  - Omezená práce se souborovým systémem



# Zpracování dokumentu



# KAM SKRIPT NAPSAT



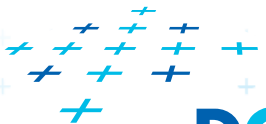
**DCGI**





# Ukázka JavaScriptu

```
2 <html>
3 <head>
4   <meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
5   <title>Ukázka JavaScriptu</title>
6   <script type="text/javascript">
7     function dokumentNacten()
8     {
9       alert("Dokument byl načten.");
10    }
11  </script>
12 </head>
13
14 <body onLoad="dokumentNacten()">
15   <h1>Následující text je generovaný skriptem</h1>
16   <script type="text/javascript">
17     document.write("Toto je napsáno pomocí skriptu.");
18   </script>
19   <form onReset="return confirm('Opravdu vymazat obsah formuláře?')">
20     <input type="text" value="" />
21     <input type="submit" value="Odeslat" />
22     <input type="reset" value="Vymazat" />
23   </form>
24 </body>
25 </html>
```



# Skripty a HTML: jak ho zapsat

```
<script language="JavaScript" type="text/javascript">
```

```
<!--
```

**tady je skript**

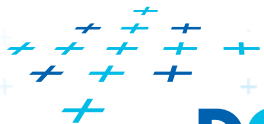
```
// -->
```

```
</script>
```

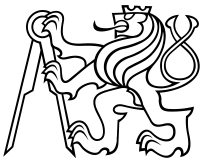
schová skript  
před prohlížeči,  
které ho neumí

```
<script language="JavaScript" type="text/javascript" src="skript.js"></script>
```

```
<input type="button" onClick="tady je skript"/>
```



**DCGI**



# Skripty a HTML: kam ho zapsat

```
<html>
```

```
<head>
```

```
<title>Jednoduchý dokument</title>
```

```
<script type="text/javascript">tady je skript</script>
```

```
</head>
```

reakce  
na  
události

```
<body>
```

```
<h1>Tělo dokumentu</h1>
```

```
<script type="text/javascript">tady je skript</script>
```

```
<form>
```

```
<input type="text"/>
```

```
<input type="button" onClick="tady je skript"/>
```

```
</form>
```

```
</body>
```

reakce na události

tvorba obsahu při  
načítání

```
</html>
```



# Kdy se skripty spouští

- při načítání dokumentu: uvnitř *body*

**<body>**

```
<h1>Tělo dokumentu</h1>
```

```
<script type="text/javascript">tady je skript</script>
```

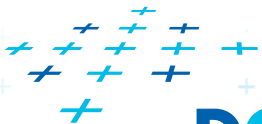
**<body>**

- řízení událostmi

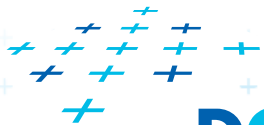
```
<input type="button" onClick="tady je skript"/>
```

Takto to ale neděláme :-

- spuštění jiným skriptem



# SYNTAXE A ZÁKLADY JAZYKA



**DCGI**



# Vlastnosti JavaScriptu

## ■ proměnné

```
var prom; // deklarace, lokální prom.
```

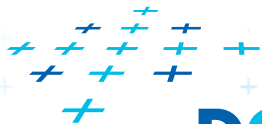
```
prom2 = "ahoj"; // deklarace a definice, globální prom.
```

## ■ netypový jazyk

```
var prom = 12; // prom je Number  
prom = "text"; // prom je String
```

## ■ datové typy

- String: "řetězec" – řetězec znaků
- Number: 4.5e-12 – libovolné číslo (celé i desetinné; decimální, oktál, hexadec)
- Boolean: true, false – logická hodnota
- Null: null – žádná hodnota
- Object definován svými vlastnostmi a metodami
- Function: function provedKontrolu() – definice funkce
- Undefined



# Syntaxe

```
var pole = ["foo", "bar", 3.14];
```

```
var data = {  
  jmeno: "Petr",  
  "prijmeni": "Novák"  
}
```

```
var plus = function(a, b) {  
  return a+b;  
}
```



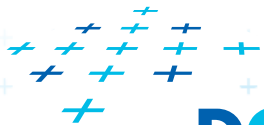
# Syntaxe pokr.

```
if (a && b) { ... }
```

```
for (var i=0; i<pole.length; i++) { ... }
```

```
try { ... } catch (e) { ... }
```

```
switch (x) {  
    case y:  
        break;  
}
```





# Vlastnosti JavaScriptu

## ■ konverze datových typů

```
vysledek = 2 + 3           // vysledek = 5
vysledek = 2 + "3"        // vysledek = "23"
vysledek = 2 + 2 + "3"    // vysledek = "43"
"12" < 3                  // numerické srovnání;false
```

## ■ pole

- nemají souvislý index, každá položka jiný typ

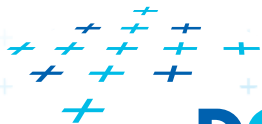
```
p[0]= 1;                   // p.length==1
p[10]="prvek s indexem 10"; // p.length==11; v paměti 2 prvky
```

- asociativní pole

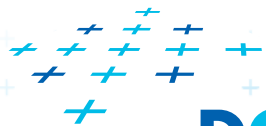
- metody

```
p.join(,);                // konverze do String, oddělovač ",", ""
p.reverse();              // řazení pozpátku
p.sort();                 // alfanumerické řazení
function ciselne_razeni(a,b){return a-b}
p.sort(ciselne_razeni);   // numerické seřazení
```

- build-in pole: např.: forms[], elements[]



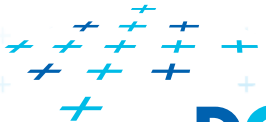
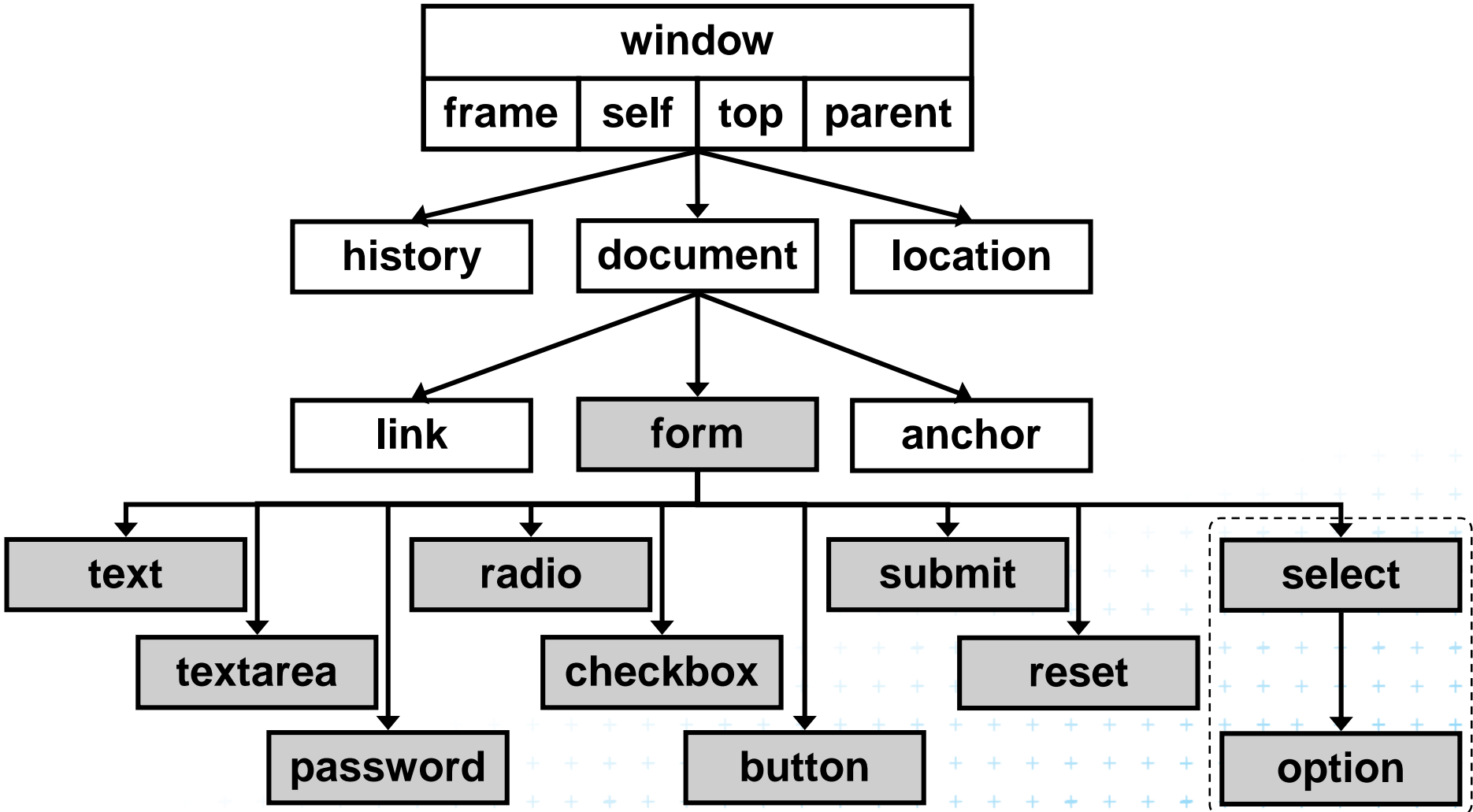
# SPOJENÍ S PROHLÍŽEČEM



**DCGI**

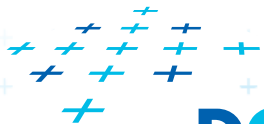
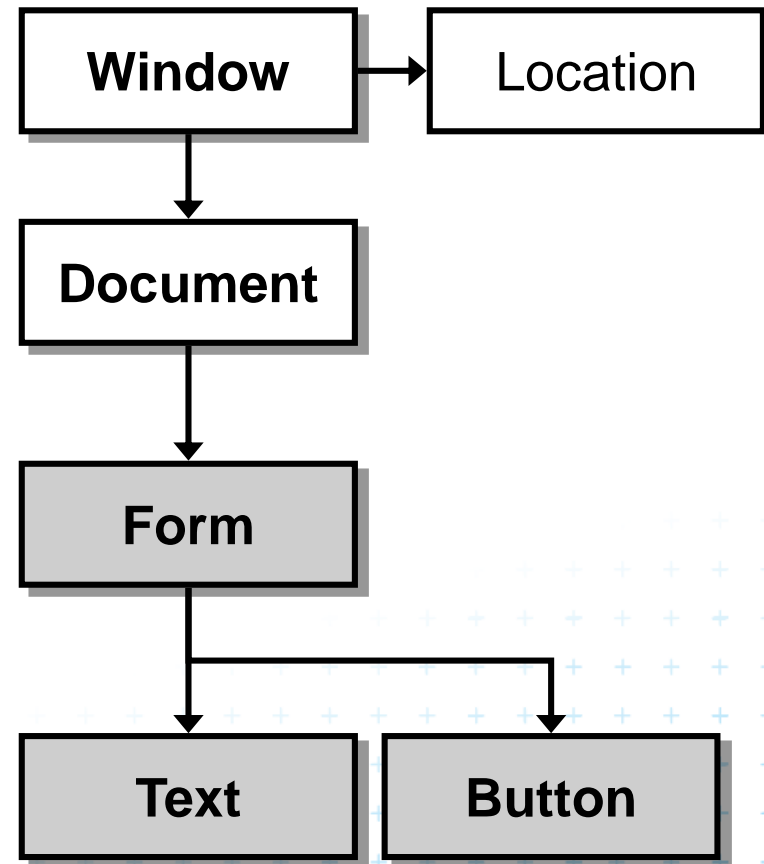


# DOM - hierarchie



# DOM: Ukázka

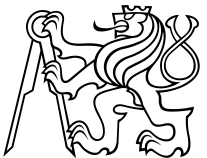
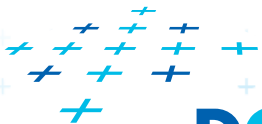
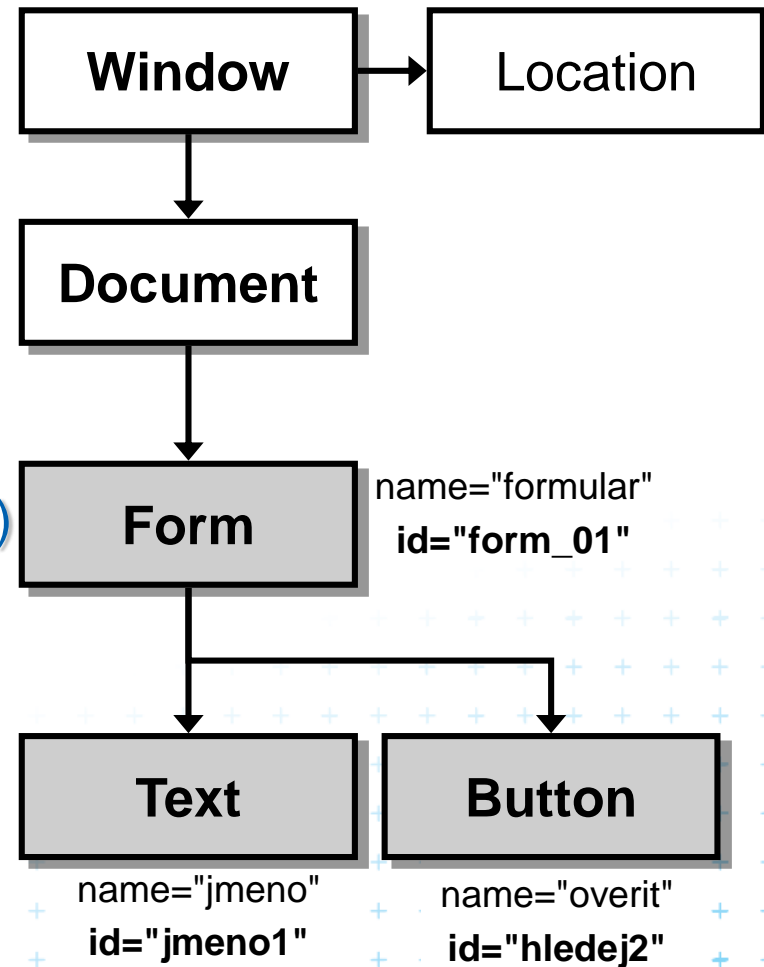
```
<html>
<head>
  <title>Jednoduchý dokument</title>
</head>
<body>
<h1>Tělo dokumentu</h1>
<form>
  <input type="text"/>
  <input type="button"/>
</form>
</body>
</html>
```



# DOM adresování

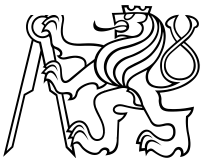
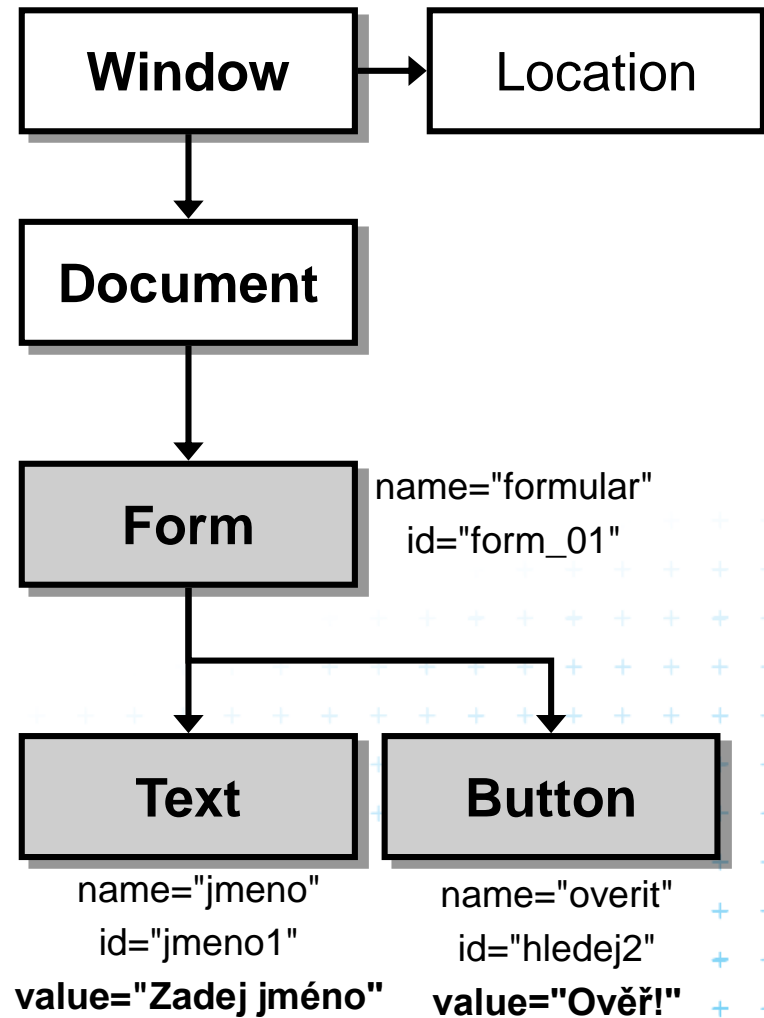


- `document.getElementById("form_01")`
- `document.getElementById("jmeno1")`
- `document.getElementById("hledej2")`



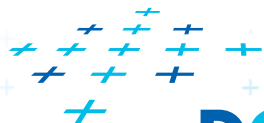
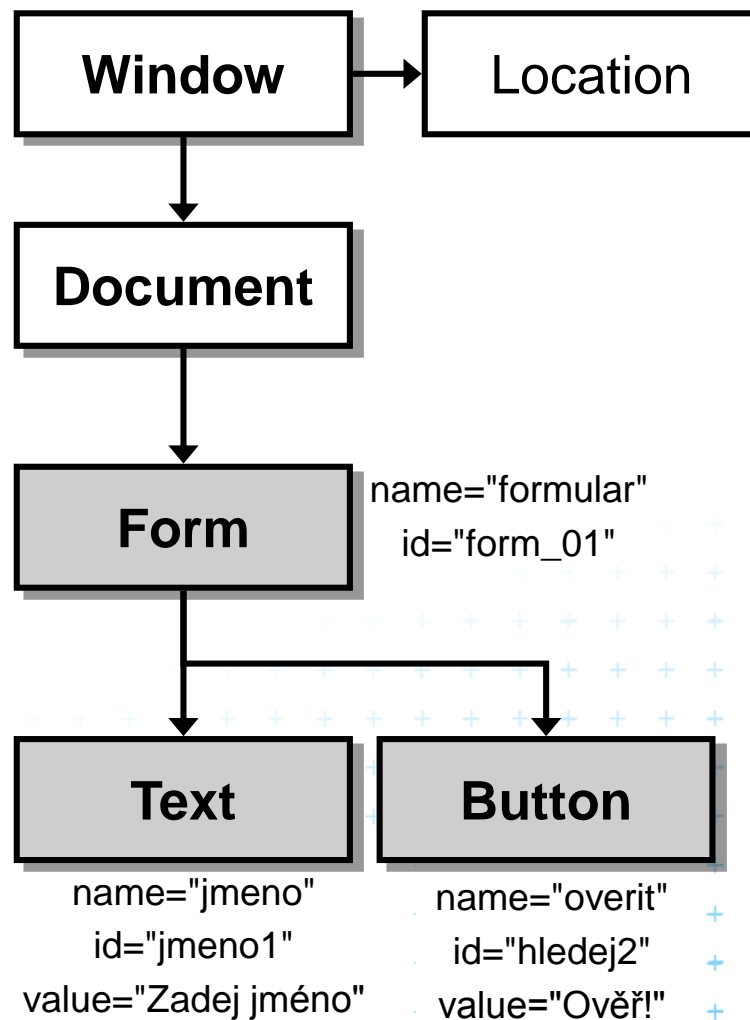
# DOM: Vlastnosti (properties)

- `document.getElementById("jmeno1").value`



# DOM: Metody

- `window.moveTo(30,50)`
- `document.write("Nějaký text")`
- `document.getElementById("form_01").submit()`
- `document.formular.jmeno.select()`



# DOM - dotazování

```
var druhyOdkaz = document.querySelectorAll("a")[1];
```

```
var obrazek = document.querySelector("p > img");
```

```
var odstavec = obrazek.parentNode;
```

```
var potomek = odstavec.firstChild;
```

```
alert(odstavec.innerHTML);
```



# DOM - manipulace

```
var odstavec = document.querySelector("p");
```

```
odstavec.innerHTML = "Generováno <em>skriptem</em>";
```

```
var obrazek = document.createElement("img");
```

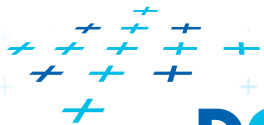
```
obrazek.src = "...";
```

```
odstavec.appendChild(obrazek);
```

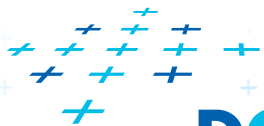


# Javascript – pokročilé programování

- Vytváření objektů
- Prototypy a dědičnost
- Události
  - Event Handler
  - Event Objekt
  - Proublávání
  - Ošetřování chyb



# UDÁLOSTI

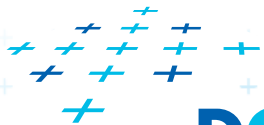


**DCGI**



# Události v javascriptu

- Implicitní definice Event-handleru
- Explicitní definice Event-handleru
- Objekt event
- Životní cyklus události
- Probublávání události



# Události

- Události jsou generovány v uživatelském rozhraní
- Máme možnost je odchytnout a napojit na nějaký vlastní kód
- Část programu, která ošetřuje události se nazývá **Event-Handler**



# Ošetření událostí

- Některé události mají přiřazené implicitní akce
- Tyto akce jsou volány, pokud neřekneme jinak
- Příklad:
  - click na odkazu způsobí přechod na jinou stránku
  - click na tlačítko submit způsobí odeslání formuláře
- Jestliže definujeme vlastní akci, je pořadí vykonání
  1. vlastní definované akce
  2. implicitní akce



# Registrace Event-handlerů podle standardu W3C

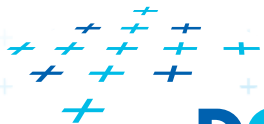
```
element.addEventListener('click',  
    doSomething, false);
```

```
element.addEventListener('click',  
    doSomethingElse, false);
```

- Registrují se oba
- Poslední argument určuje, zda se událost má odchytit ve fázi capture nebo bubble (false=bubble)

```
element.removeEventListener('click', doSomethingElse, false)
```

- Odstranění Event-handleru



# Způsoby zachytávání událostí

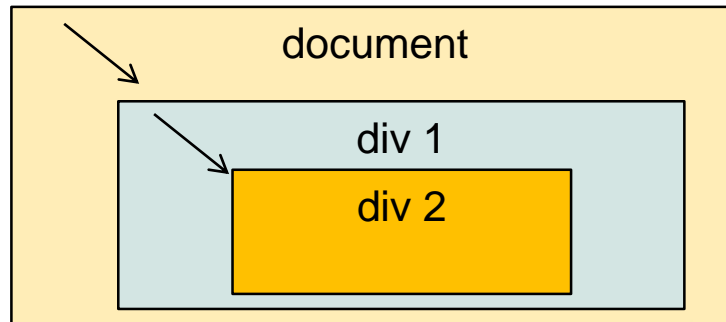
- Otázka: jestliže mám vnořený element který odchyťává stejnou událost jako jeho nadřazený element, kdo to má odchyťit první?

- Event Capture

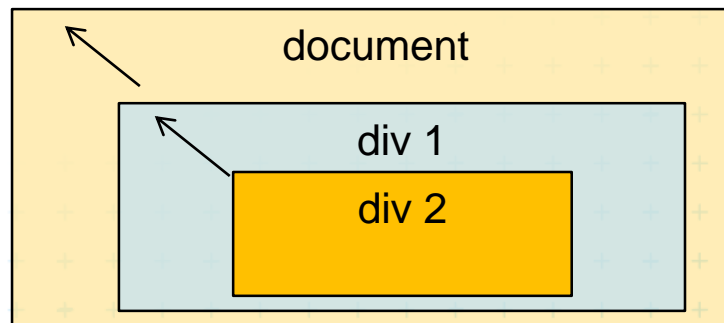
1. document
2. div 1
3. div 2

- Event Bubbling

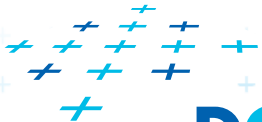
- div 2
- div 1
- document



Netscape



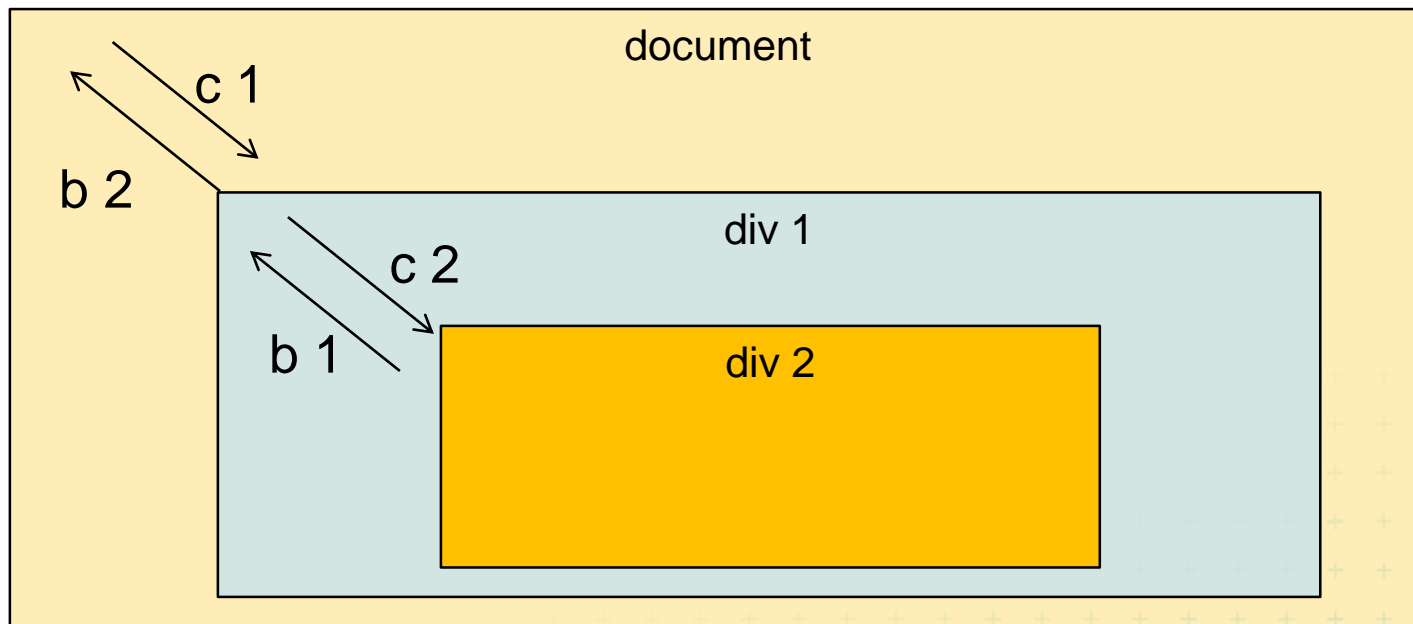
Microsoft





# Propagování události – W3C

- Kombinace capture a bubble propagace
- Nejprve capture (c), pak zpětné bubble (b)



kuk: capture.htm

# Události - příklad

```
var p = document.querySelector("p");
```

```
var posluchac = function() {  
    alert("kliknuto");  
}
```

```
p.addEventListener("click", posluchac); //  
předáno odkazem
```

```
/* ... */
```

```
p.removeEventListener("click", posluchac);
```



# Event objekt

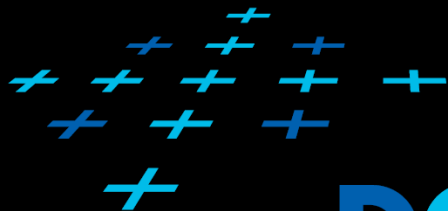
```
document.body.addEventListener("click",  
function(e) {  
    alert(e.type);        // "click"  
  
    alert(e.clientX);    // souřadnice kurzoru  
    alert(e.target);    // ?  
  
    alert(e.ctrlKey);    // informace o  
klávesách  
});
```



# Výchozí akce a zastavení akce

```
var form = document.querySelector("form");  
form.addEventListener("submit", function(e) {  
    e.preventDefault(); // formulář nebude odeslán  
});
```

```
var odkaz = document.querySelector("a");  
odkaz.addEventListener("click", function(e) {  
    e.stopPropagation(); // nikdo další se nedozví  
});
```



**DCGI**

**KATEDRA POČÍTAČOVÉ GRAFIKY A INTERAKCE**

**Děkuji za pozornost**

**Martin Klíma**