

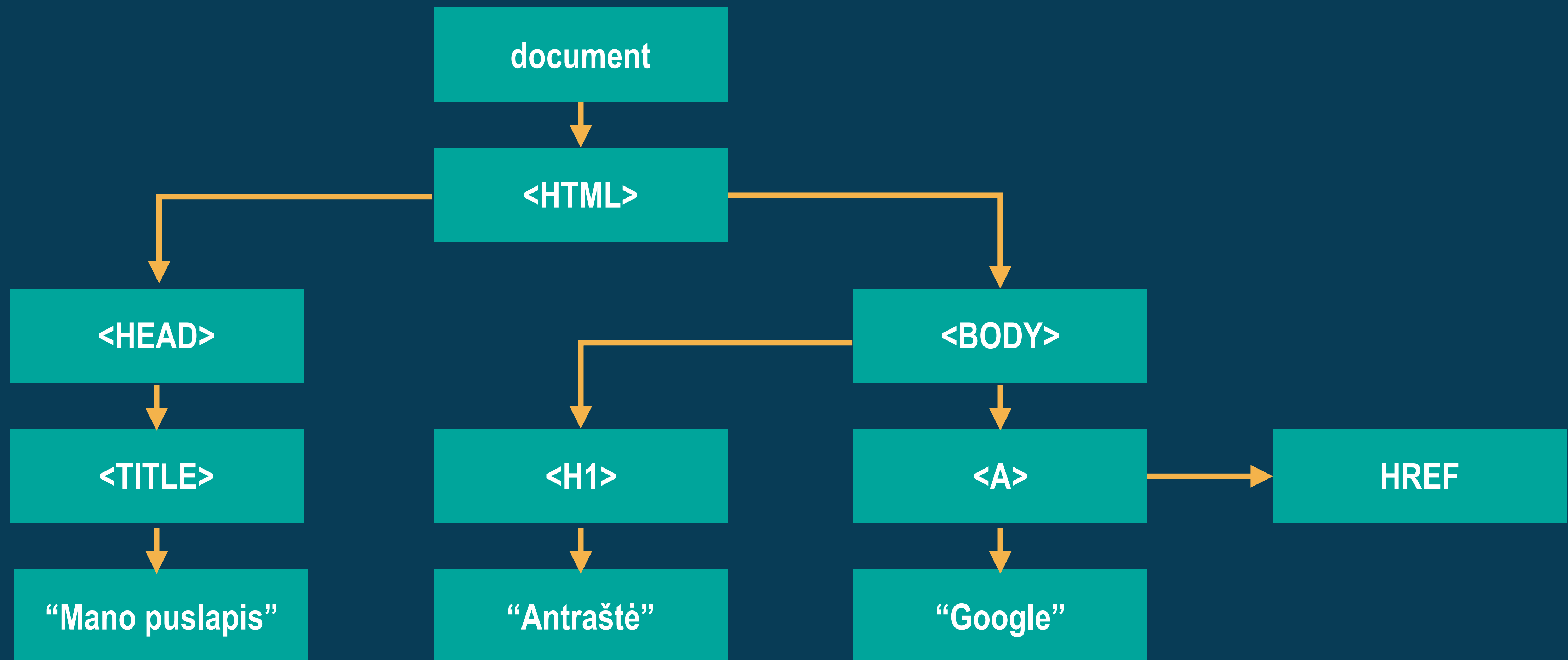
BALTIC TALENTS ACADEMY

JAVASCRIPT

KARTOJIMAS

- ▶ DOM - Document Object Model
- ▶ DOM elemento **style** atributas
- ▶ DOM elemento **className** atributas
- ▶ DOM elemento **classList** atributas

DOM - DOCUMENT OBJECT MODEL



DOM ELEMENTŲ CSS

- ▶ HTML:

```
<p id="p2">Tekstas</p>
```

- ▶ JavaScript:

```
var elementas = document.getElementById("p2");  
elementas.style.color = "white";  
elementas.style.backgroundColor = "green";
```

DOM ELEMENTŲ KLASĖS

- ▶ HTML:

```
<p id="p2" class="abc">Tekstas</p>
```

- ▶ JavaScript:

```
var elementas = document.getElementById("p2");  
elementas.className = "abc raudona";
```

- ▶ HTML:

```
<p id="p2" class="abc raudona">Tekstas</p>
```

DOM ELEMENTŲ KLASĖS

- ▶ HTML:

```
<p id="p2" class="abc">Tekstas</p>
```

- ▶ JavaScript:

```
var elementas = document.getElementById("p2");  
var klases = elementas.classList;  
klases.add("raudona");
```

- ▶ HTML:

```
<p id="p2" class="abc raudona">Tekstas</p>
```

- ▶ JavaScript:

```
klases.remove("abc");
```

- ▶ HTML:

```
<p id="p2" class="raudona">Tekstas</p>
```

DATOS / LAIKO OBJEKTAS

- ▶ `new Date()`
- ▶ `new Date(value)`
value - milisekundės nuo 1970 Sausio 1d. 00:00:00 UTC
- ▶ `new Date(dateString)`
dateString - "2016-01-01 12:45:00" - nenaudoti
- ▶ `new Date(year, month[, date[, hours[, minutes[, seconds[, milliseconds]]]])`

DATOS / LAIKO OBJEKTO METODAI (FUNKCIJOS)

- ▶ `.getFullYear()` - metai
- ▶ `.getMonth()` - mėnuo: 0..11 !!!
- ▶ `.getDate()` - mėnesio diena: 1..31
- ▶ `.getDay()` - savaitės diena: 0..6 (0 - sekmadienis)
- ▶ `.getHours()` - valandos: 0..23
- ▶ `.getMinutes()` - minutės: 0..59
- ▶ `.getSeconds()` - sekundės: 0..59
- ▶ `.getTime()` - milisekundės nuo 1970 Sausio 1d. 00:00:00 UTC
- ▶ analogiškai yra ir 'set' funkcijos - pvz. `.setMonth(4)`

OBJEKTAI

- ▶ `var figura = {ilgis: 2, plotis: 5, aukstis: 3};`
- ▶ Reikia pasirašyti funkciją, kuri suskaičiuoja figūros tūrį:
- ▶ `figura.turis = function () {
 return this.ilgis * this.plotis * this.aukstis;
};`
- ▶ `console.log (figura.turis());`
- ▶ `figura.ilgis = 4;`
- ▶ `console.log (figura.turis());`

DINAMINIAI OBJEKTAI

Dabar mes žinome, kad jei reikia sukurti kelis objektus turinčius tas pačias funkcijas, tai reikia funkcijas aprašyti kiekviename objekte, pvz:

```
var a1 = {  
  numeris: 'LRS001',  
  atstumas: 5000,  
  laikas: 150,  
  greitis: function() { return this.atstumas / this.laikas * 3.6; }  
}
```

```
var a2 = {  
  numeris: 'ABC222',  
  atstumas: 5000,  
  laikas: 125,  
  greitis: function() { return this.atstumas / this.laikas * 3.6; }  
}
```

```
console.log( a1.greitis() );  
console.log( a2.greitis() );
```

DINAMINIAI OBJEKTAI

- ▶ Tai objektai kurie kuriami su **new** operatoriumi naudojant specialią funkciją vadinamą objekto konstruktoriumi arba tiesiog konstruktoriumi

OBJEKTO KONSTRUKTORIUS

```
function Automobilis(nr, s, t) {  
  this.numeris = nr;  
  this.atstumas = s;  
  this.laikas = t;  
  this.greitis = function () {  
    return this.atstumas / this.laikas * 3.6;  
  };  
}  
  
var a1 = new Automobilis('LRS001', 5000, 150);  
var a2 = new Automobilis('ABC222', 5000, 125);  
  
console.log( a1.greitis() );  
console.log( a2.greitis() );
```

OBJEKTO PROTOTIPAS

- ▶ Kiekvienas objektas turi taip vadinamą to objekto prototipą, t.y. to objekto tėvyninį “objektą” iš kurio jis paveldi metodus ir atributus.
- ▶ Norint sukurti naują arba perrašyti seną objekto metodą (funkciją) arba objekto atributo pradinę reikšmę reikia juos aprašyti objekto prototipe.

OBJEKTO PROTOTIPAS

```
function Automobilis(nr, s, t) {  
  this.numeris = nr;  
  this.atstumas = s;  
  this.laikas = t;  
  this.greitis = function () {  
    return this.atstumas / this.laikas * 3.6;  
  };  
}
```

```
var a1 = new Automobilis('LRS001', 5000, 150);  
var a2 = new Automobilis('ABC222', 5000, 125);
```

```
Automobilis.prototype.greitisms = function() { return this.atstumas / this.laikas };
```

```
console.log( a1.greitisms() );  
console.log( a2.greitisms() );
```

FORMOS

HTML:

```
<form name="forma" onsubmit="return onFormSubmit()">  
  <input name="atstumas" type="number">  
  <input name="laikas" type="number">  
  <button>Saugoti</button>  
</form>
```

JS:

```
function onFormSubmit() {  
  // ... document.forms['forma']['atstumas'].value ...  
  return false;  
}
```

Funkcija **onFormSubmit** grąžina **true** jei formą siųsti į serverį arba **false** - jei ne.

p.s. Tas pats galioje visiems įvykiams - jei grąžinama true - įvykis toliau apdorojamas, false - ne.

TRY - CATCH - FINALLY BLOKAS

Jei tam tikroje kodo vietoje tikėtina, kad gali būti klaida, tai tokį kodo bloką naudinga apglėbti try-catch sakiniu:

```
try {  
    ...  
} catch (error) {  
    ...  
} finally {  
    ....  
}
```


UŽDAVINYS

Sukurkite slaptažodžio keitimo formą, kurioje būtų du įvedimo laukai (naujas slaptažodis, pakartotinai suvestas naujas slaptažodis) ir mygtukas 'Keisti'

Paspaudus tą mygtuką reikia patikrinti ar abu slaptažodžiai tokie patys - jei ne tai parodyti tekstą 'Slaptažodžiai nesutampa - pakartokite dar kartą'. Jei tokie patys - rodykite pranešimą 'Slaptažodis keičiamas...' ir animuokite kažkaip parodydami kad vyksta pakeitimas ir laukiamas atsakymas iš serverio