

JavaScript dióhéjban

Siki Zoltán



Böngészőkben futtatható szkript nyelv

A Netscape-nél fejlesztették ki (Brendan Eich)

HTML-be ágyazható

A helyi gép erőforrásait **korlátozottan** érheti el

Prototípus alapú objektum orientált nyelv

Böngészőben bekövetkezett események kezelése (pl. klikk)

Bár a JavaScriptet szabványosították, a különböző böngésző programokban eltérő implementációk készültek (IE!)

A böngészőkhöz kapcsolódó fejlesztőkörnyezetek

Firefox

Chrome

Internet Explorer

beépített fejlesztőeszközök,
debugger

OpenLayers ismertető

- [JavaScript és OpenLayers 3 gyorstalpaló](#), Siki Zoltán, 2015.
- [Webes térképezés gyorstalpaló](#), OpenLayers 2.x, Padányi-Gulyás Gergely, 2013.
- [OpenLayers API alapjai](#), OpenLayers 2.x, Gede Máttyás, 2013.

Mintapéldák

A mintapélda oldalakon nézze meg az oldal forrását a böngészőben, hogy a JavaScript kód is megjelenjen.

- [Heron mintapéldák](#), Padányi-Gulyás Gergely, 2014.
- [BME térkép](#), Siki Zoltán 2011
- [Ortofotók összehasonlítása](#), Siki Zoltán 2013
- [MapFish \(OpenLayers\), OSM](#), Berényi Attila/Siki Zoltán, 2010.
- [OpenLayers Cookbook](#) mintapéldái, Antonio Santiago Perez, 2012. ([letöltés](#))

További információk: siki@agt.bme.hu

A gyűjtemény karbantartásához segítőtársakat keresünk!



Google
Keresés az oldalon
található dokumentumokban:

 Keresés

[További angol nyelvű
oktatóanyagok és videók](#)



Az FDL csak a szerverünkön megtalálható
dokumentumokra vonatkozik!

Frissítés időpontja: 2015.09.28

Menüből Developer és Debugger (Ctrl + Shift + S)

Inspector Console Debugger Style Editor Performance Memory Network DOM



Search scripts (Ctrl+P)

Sources Call Stack

1

This page has no sources.

Változók

Egyszerű változók szám, szöveg és logikai érték tárolására
Különböző típusú változók közötti műveleteknél automatikus
típus konverzió

Az utasítás után írt ; (pontosvessző) nem kötelező, ha csak
egy utasítás van a sorban

Interaktív JavaScript parancsok bevitele

Clear Persist Profile All Errors Warnings Info Debug Info

```
> a = 5;
5
> b = 6;
6
> a + b
11
> s = 'alma';
"alma"
> t = "körte";
"körte"
> s + " " + t
"alma körte"
> c = 12.34
12.34
```

Mi lesz x és y értéke?
v1 = '112';
v2 = 76;
x = v1 + v2;
y = v2 + v1;

Logikai konstansok:
true
false
v1 == v2;

Kis és nagybetűk
különböznek!

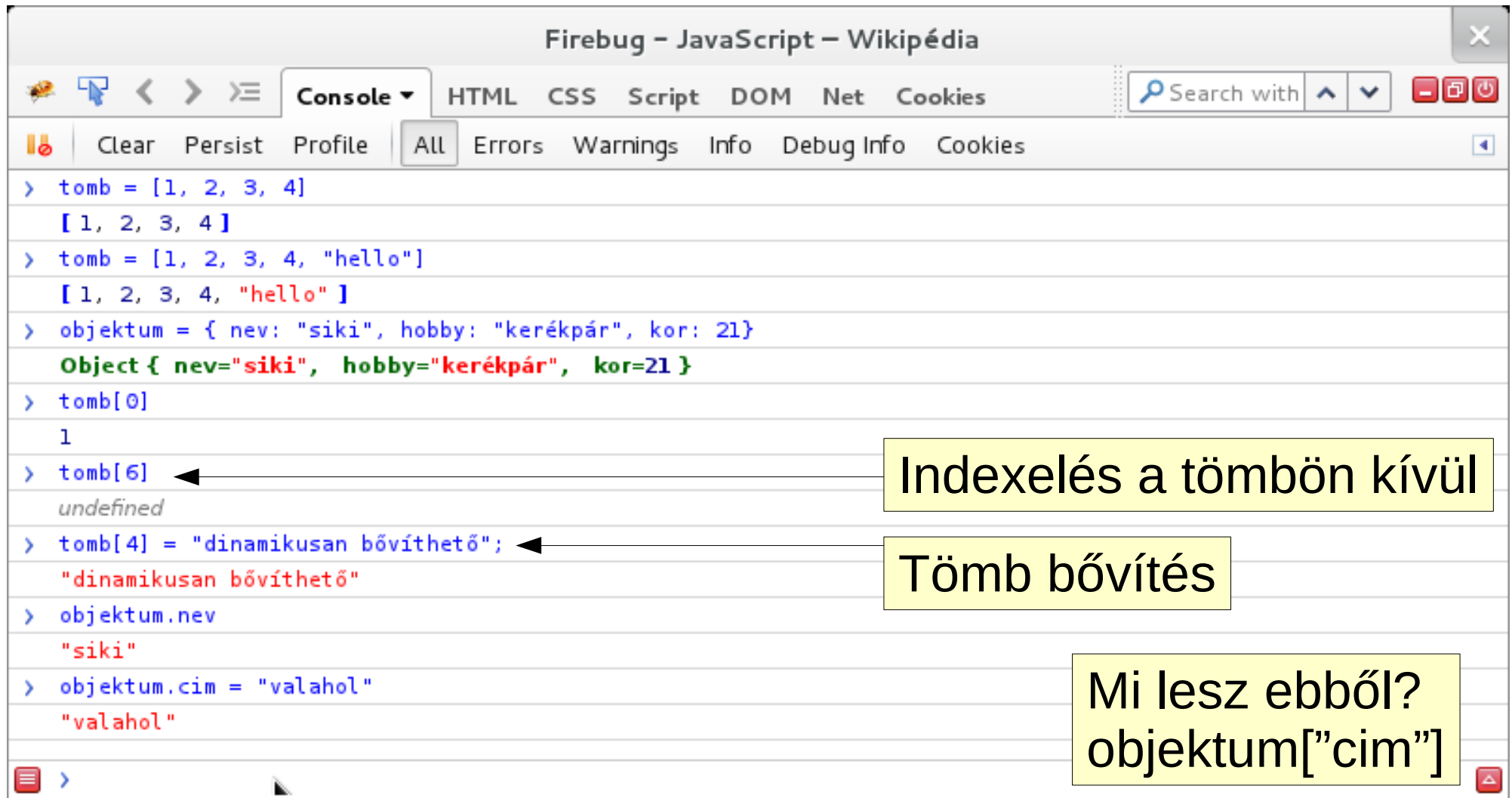
Mi az eredmény?
6 / 0

Összetett adattípusok

Tömb és objektum

Tömb – rendezett adattípus, indexelés 0-tól

Objektum – asszociatív tömb (függvényeket is tartalmazhat)



The screenshot shows the Firebug JavaScript console with the following code and output:

```
> tomb = [1, 2, 3, 4]
[ 1, 2, 3, 4 ]
> tomb = [1, 2, 3, 4, "hello"]
[ 1, 2, 3, 4, "hello" ]
> objektum = { nev: "siki", hobby: "kerékpár", kor: 21 }
Object { nev="siki", hobby="kerékpár", kor=21 }
> tomb[0]
1
> tomb[6]
undefined
> tomb[4] = "dinamikusan bővíthető";
"dinamikusan bővíthető"
> objektum.nev
"siki"
> objektum.cim = "valahol"
"valahol"
```

Annotations in yellow boxes:

- Indexelés a tömbön kívül (points to `tomb[6]`)
- Tömb bővítés (points to `tomb[4] = "dinamikusan bővíthető";`)
- Mi lesz ebből? objektum["cim"] (points to `objektum.cim = "valahol"`)

Fontosabb beépített objektumok

Math, Date, RegExp, Function

Függvények (igen, a JavaScriptben a függvények is objektumok)

Szabályos kifejezések

Dátum és időpont kezelés

Matematikai függvények, konstansok

```
> Math.sin(0.5)
0.479425538604203
> Math.PI
3.141592653589793
> Date()
"Tue Sep 08 2015 22:15:15 GMT+0200 (CEST)"
> d = new Date()
Date { Tue Sep 08 2015 22:15:22 GMT+0200 (CEST) }
> d.getDay()
2
> r = /^a.*x$/
RegExp /^a.*x$/
> "asterix".match(r)
[ "asterix" ]
> "alma".match(r)
null
```

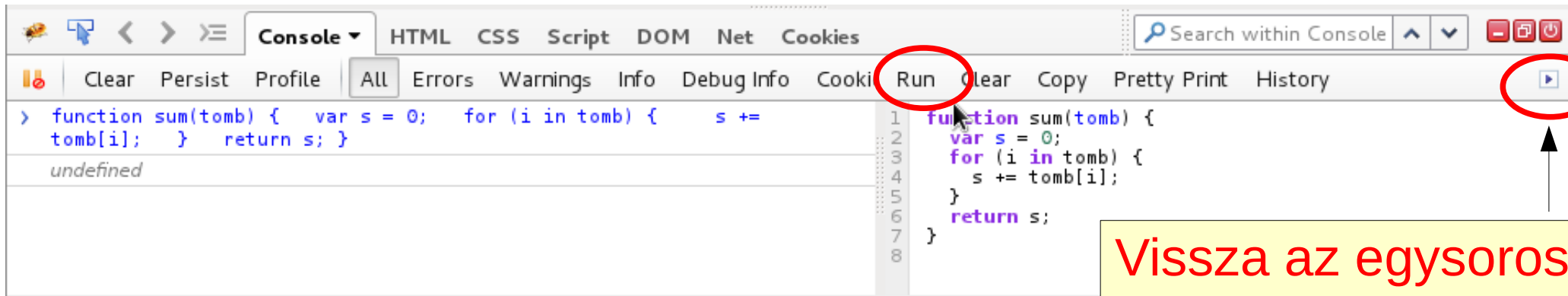
Math statikus objektum

Új objektum létrehozása

Többsoros adatbevétel

Ennek mi lesz az eredménye:
r.test("asterix")

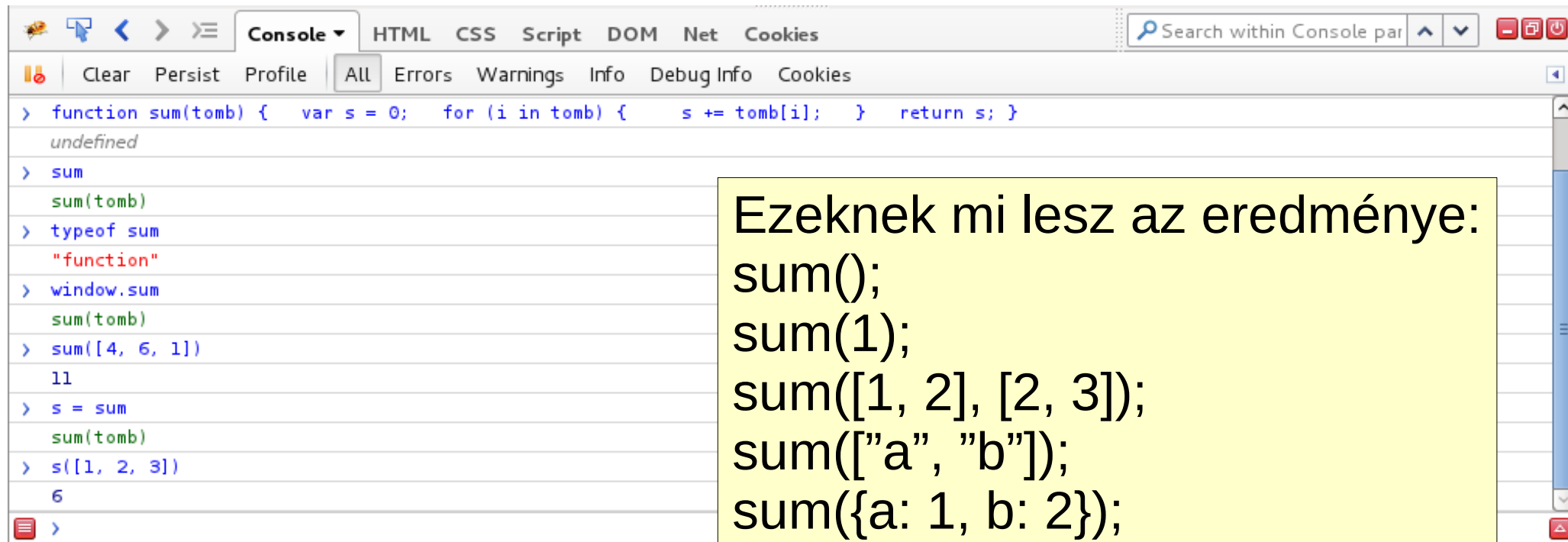
Függvények



```
> function sum(tomb) { var s = 0; for (i in tomb) { s += tomb[i]; } return s; }
```

```
1 function sum(tomb) {  
2   var s = 0;  
3   for (i in tomb) {  
4     s += tomb[i];  
5   }  
6   return s;  
7 }  
8
```

Vissza az egysoros adatbevitelhez



```
> function sum(tomb) { var s = 0; for (i in tomb) { s += tomb[i]; } return s; }
```

```
undefined
```

```
> sum  
sum(tomb)
```

```
> typeof sum  
"function"
```

```
> window.sum  
sum(tomb)
```

```
> sum([4, 6, 1])  
11
```

```
> s = sum  
sum(tomb)
```


```
> s([1, 2, 3])  
6
```

Ezeknek mi lesz az eredménye:
sum();
sum(1);
sum([1, 2], [2, 3]);
sum(["a", "b"]);
sum({a: 1, b: 2});
Miért?

Függvények 2

A sum függvény másik megfogalmazása
Változó hosszúságú paraméterlista kezelése

```
function sum() {  
  var s = 0;  
  var i;  
  for (i in arguments) {  
    if (typeof arguments[i] == 'number') {  
      s += arguments[i];  
    }  
  }  
  return s;  
}
```



Rekurzió

```
function fact(n) {  
  var f;  
  if (n == 0) {  
    return 1;  
  }  
  return n * fact(n-1);  
}
```

Hasznos további beépített függvények:

parseInt("123")

parseFloat("1.123")

isNaN(NaN)

isFinite(Infinity)

eval("1+2")

Feladat:

Írj egy függvényt az első n természetes szám összegére

Időzítés JavaScripttel

<http://www.agt.bme.hu/tantargyak/web/timer.html>

```
counter = 0; // kiírandó rész pozíciója msg-ben
width = 70; // mező szélessége a szöveghez
var spc = " *** "; // elválasztó a hírek közé
var msg = "Javascript programozás" + spc + // görgetendő szöveg
        "Weblap tervezés/készítés" + spc +
        "MySQL/Postgres adatbázisok" + spc +
        "Internetes térképek" + spc;
```

Változók inicializálása
Az oldal HEAD részében

```
function gorget()
{
```

onLoad eseményre

Globális változó

```
var out;
counter++;
if (counter > msg.length) counter = 0;
out = msg;
while (out.length <= width) {
    out += spc + msg;
}
```

Lokális változó

Rövid szöveg ismétlése

```
out = out.substr(counter, width);
document.getElementById("banner").innerHTML = out;
window.setTimeout(gorget, 500); // 0.5 másodperc után újra
```

Banner id-jű div-be írás

```
}
```

Időzítés beállítása újra

További egyszerű példák

<http://www.agt.bme.hu/tantargyak/web/prim.html>

http://www.geod.bme.hu/on_line/hossztorzulas.html

<http://www.agt.bme.hu/tantargyak/web/animacio.html>

<http://www.agt.bme.hu/tantargyak/web/eov.html>

<http://www.agt.bme.hu/tantargyak/web/eovxy.htm>

<http://www.w3schools.com/js/>



Objektumok

Nincsenek osztályok csak objektumok (prototípus alapú)

Összezártság (függvények és adatok)

Öröklődés (prototípuson keresztül, trükkös)

Többértelműség (nincs paraméter stamp)

```
function személy(first, last, birth) {  
  this.firstName = first;  
  this.lastName = last;  
  this.birth = birth;  
  this.age = function () {  
    return new Date().getFullYear() - this.birth;  
  }  
}
```

Konstruktor függvény (prototípus)

Objektum literális

```
me = { firstName: 'Zoltán',  
      lastName: 'Siki', age: 1958,  
      age: function () {  
        return new Date().getFullYear() - this.birth;  
      }  
}
```

Névtelen függvény

```
me = new személy('Zoltán', 'Siki', 1958);  
me.age();
```

```
szemely.prototype.fullname = function() {  
  return this.lastName + ' ' + this.firstName;  
}
```

objektum példány létrehozás

Mi a különbség a konstruktor függvény és az objektum literális között?

minden példány bővítése új függvénnyel!

Komplex szám objektum példa

```
* komplex számok osztálya */
```

```
/* konstruktor */
```

```
function Complex(r, i) {  
  if (r) { this.real = r; } else { this.real = 0.0; }  
  if (i) { this.imag = i; } else { this.imag = 0.0; }
```

Opcionális paraméter
kezelés

```
/* metódusok */
```

```
this.setReal = function(r) { this.real = r; };  
this.setImag = function(i) { this.imag = i; };  
this.getReal = function() { return this.real; };  
this.getImag = function() { return this.imag; };  
this.abs = function() {  
  return Math.sqrt(this.real * this.real + this.imag * this.imag);
```

Getter, setter metódusok

```
};  
this.conj = function() {  
  var w = this.real; this.real = this.imag; this.imag = w;
```

Eredeti objektum
módosítása

```
};  
this.inc = function(c) {  
  this.real += c.real; this.imag += c.imag;
```

Eredeti objektum
módosítása

```
}  
this.toString = function() {  
  return this.real + '+' + this.imag + 'i';  
}
```

```
}
```

Komplex szám objektum használata

```
/* komplex osztály használata */
```

```
var a = new Complex();  
document.write('a=' + a.toString() + '<br>');  
var b = new Complex(1.0);  
document.write('b=' + b.toString() + '<br>');  
var c = new Complex(0.5, 2.4);  
document.write('c=' + c.toString() + '<br>');  
var d = new Complex();  
d = c; d.conj();    // vigyázat ez elrontja c értékét is! REFERENCIA  
                  // d = Complex(c.real, c.imag) esetén c nem változik!  
document.write('d=' + d.toString() + '<br>');  
document.write('de !!! c=' + c.toString() + '<br>');  
var e = new Complex();  
e.setReal(4.1); e.setImag(1.8);  
document.write('e=' + e.toString() + '<br>');  
d.inc(e);  
document.write('d=' + d.toString() + '<br>');
```

Megoldható olyan conjugalt függvény készítése, mely az eredeti objektumot nem rontja el?

Megoldható olyan összeg függvény készítése, mely egy új objektumba teszi két komplex szám összegét?

Objektumon kívüli megoldás

```
function conjugalt(c) {  
    return new Complex(c.getImag(), -c.getReal())  
}
```

Mi lesz, ha a paraméter nem Complex típusú objektum?

```
function add(c1, c2) {  
    return new Complex(c1.getReal() + c2.getReal(), c1.getImag() + c2.getImag());  
}
```

Objektumon belüli megoldás

```
Complex.prototype.conjugate = function () {  
    return new Complex(this.getImag(), -this.getReal())  
}
```

Itt nem lehet paraméter típus probléma

```
Complex.prototype.add = function (c) {  
    if (typeof c == "Number") { return new Complex(this.getReal() + c, this.getImag()); }  
    if (typeof c == "Object" && c.constructor == Complex) {  
        return new Complex(this.getReal() + c.getReal(), this.getImag() + c.getImag());  
    }  
    return new Complex(this.getReal(), c.getImag());  
}
```

Complex + szám

Complex + Complex

Minden más esetben az eredeti Complex

OpenLayers



2006 óta

Nyíl forráskódú, BSD 2 licenz, OSGeo projekt

Vékony kliens, JavaScript könyvtár, AJAX

Szabványos térkép szolgáltatások (WMS, WMTS, WFS, ...)

Nem szabványos szolgáltatások (Google)

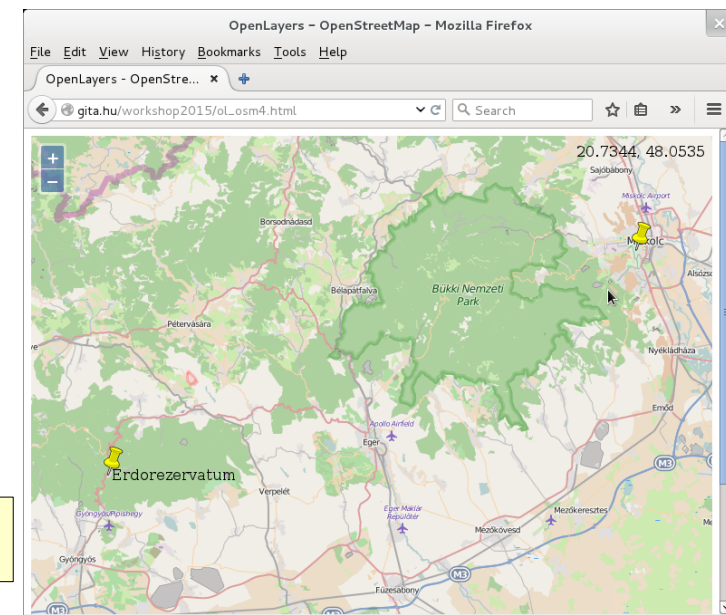
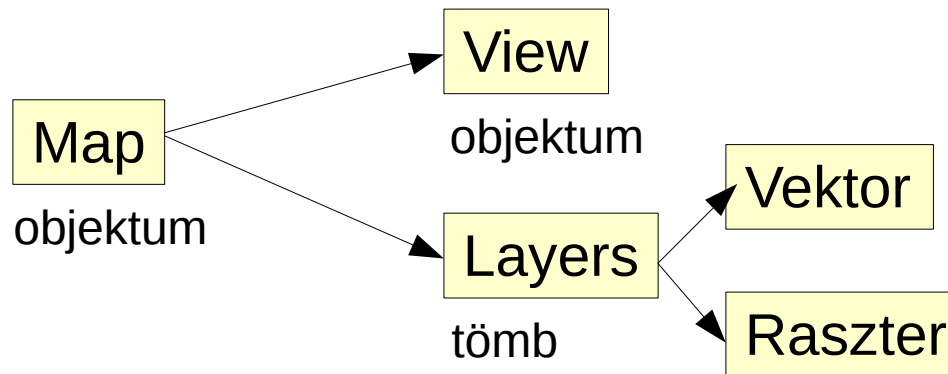
Vektor rétegek (KML, GPX, GeoJSON, GML)

<http://openlayers.org/en/v3.8.2/examples/>

<http://openlayers.org/en/v3.8.2/apidoc/>

<http://openlayers.org/en/v3.8.2/doc/>

Logikai
struktúra



OpenLayers példa

```
<!doctype html>
```

```
<head>
```

```
<title>OpenLayers - OpenStreetMap</title>
```

```
<link rel="stylesheet" href="http://openlayers.org/en/v3.9.0/css/ol.css" type="text/css" />
```

```
<!-- OpenLayer letöltése -->
```

```
<script src="http://openlayers.org/en/v3.9.0/build/ol.js"></script>
```

```
</script>
```

```
// a térkép inicializálás
```

```
function initialize() {
```

```
  // OSM réteg
```

```
  var osmLayer = new ol.layer.Tile({  
    source: new ol.source.OSM()  
  });
```

```
  // pont a térkép középpontjához
```

```
  var eger = ol.proj.transform([20.376004, 47.897257], 'EPSG:4326', 'EPSG:3857');
```

```
  // nezet a térképhez
```

```
  var view = new ol.View({  
    center: eger, // középpont  
    zoom: 14     // nagyítási szint  
  });
```

```
  // térkép
```

```
  var map = new ol.Map({  
    target: 'map' // térképet tartalmazó htm elem  
  });
```

```
  // réteg hozzáadása a térképhez
```

```
  map.addLayer(osmLayer);
```

```
  // nezet hozzáadása a térképhez
```

```
  map.setView(view);
```

```
}
```

```
</script>
```

```
</head>
```

```
<body onLoad="initialize()">
```

```
<!-- térképet tartalmazó elem -->
```

```
<div id="map" class="map"></div>
```

```
</body>
```

```
</html>
```

Stílusok

Forrás

Tile réteg létrehozás (OSM)

WGS84 -> Mercator

Kompakt megoldás

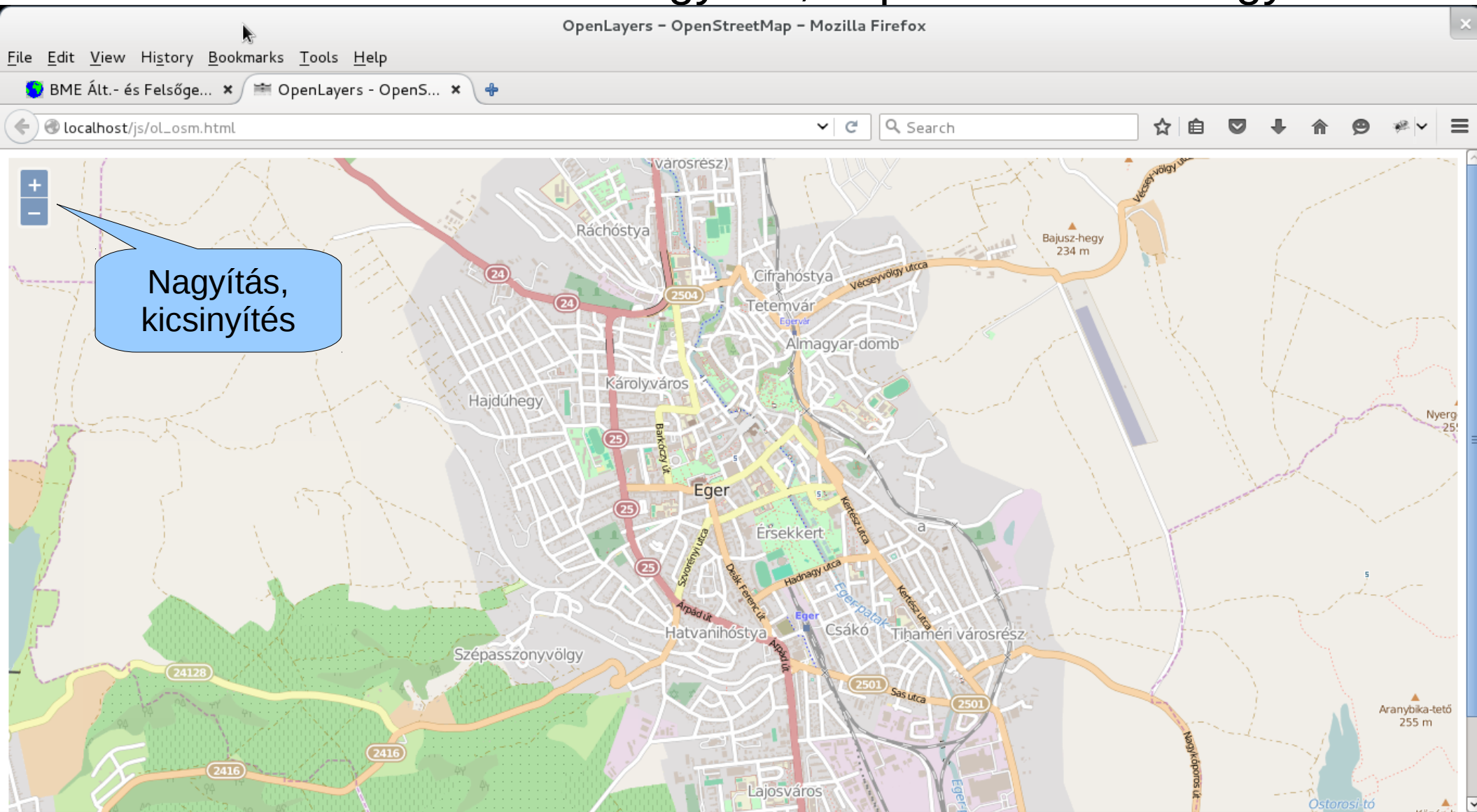
```
var map = new ol.Map({  
  target: 'map',  
  layers: [  
    new ol.layer.Tile({  
      source: new ol.source.OSM()  
    })  
  ],  
  view: new ol.View({  
    center: ol.proj.transform([20.376004, 47.897257],  
                              'EPSG:4326', 'EPSG:3857'),  
    zoom: 14  
  })  
});
```

Alapfunkciók

Egérműveletek: egérgörgő – nagyítás/kicsinyítés

megragadás és húzás – térkép eltolás

shift + húzás – ablakos nagyítás, dupla kattintás - nagyítás



Billentyűzetről vezérlés

Módosítsuk a map div-et:

```
<div id="map" class="map" tabindex="0">
```

Ettől szelektálható lesz a térkép

Az oldal újratöltése után kattintson a térképre (fókusz)

The screenshot shows a Mozilla Firefox browser window displaying a map from OpenLayers. The browser's address bar shows the URL `localhost/js/ol_osm4.html`. The map displays a region in Hungary, including Salgótarján, Mátra, and Eger. A keyboard overlay is shown in the center, with red boxes highlighting the arrow keys (for movement) and the '+' and '-' keys (for zooming). Three callout boxes provide instructions: 'Térkép forgatás' (Map rotation) with a circular arrow icon, 'Nagyítás, kicsinyítés' (Zoom in, zoom out) with a double-headed arrow icon, and 'Térkép mozgatás' (Map movement) with a hand-drawn hand icon. The text 'Érintő képernyő' (Touchscreen) is also present on the map area.

KML réteg

Az előző kódot bővítjük

[Http://www.geod.bme.hu/siki/H-BotanicalGardens.kml](http://www.geod.bme.hu/siki/H-BotanicalGardens.kml) letölteni

Új vektor réteg

```
var projection = ol.proj.get('EPSG:3857');
var kmlLayer = new ol.layer.Vector({
  source: new ol.source.Vector({
    projection: projection,
    format: new ol.format.KML(),
    url: 'H-BotanicalGardens.kml'
  })
});
```

```
map.addLayer(kmlLayer);
```

Próbáljuk ki!

Találd ki hová kerüljenek a fenti program sorok!

A KML mellett további fájl formátumok is használhatók, pl. GPX, GeoJSON

Koordináta kiírás

```
<div id="map" class="map">
  <div id="popup"></div>
</div>
```

```
var element = document.getElementById('popup');
var popup = new ol.Overlay({
  element: element
});
map.addOverlay(popup);

map.on('click', function(evt) {
  var coordinate = evt.coordinate;
  popup.setPosition(coordinate);
  element.innerHTML = coordinate[0].toFixed(0) +
    ',' + coordinate[1].toFixed(0);
});
```

Próbáljuk ki ezt is!

Attribútum kiírás

Az előző kódot bővítjük, koordináta kiírás helyett legyen az arborétum neve felirat

// függvény a felirat megjelenítéshez

```
var displayFeatureInfo = function(pixel, coordinate) {
  var features = [];
  // a kattintas kozeleben levo elemek
  map.forEachFeatureAtPixel(pixel, function(feature, layer) {
    features.push(feature);
  });
  if (features.length > 0) {
    var info = [];
    // megtalalt elemek attributumainak osszefuzese
    for (var i = 0, ii = features.length; i < ii; ++i) {
      info.push(features[i].get('name'));
    }
    // kiiras a terkepre
    element.innerHTML = info.join(', ') || '(ismeretlen)';
    // felirat pozicionalasa
    popup.setPosition(coordinate);
  } else {
    // korabbi felirat torlese
    element.innerHTML = '&nbsp;';
  }
};
// az eger kattintas es a felirat megjelenites osszekapcs.
map.on('click',function(evt){
  var coordinate = evt.coordinate;
  displayFeatureInfo(evt.pixel, coordinate);
});
```

// szelekcio bealítása a KML retegre

```
var selectClick = new ol.interaction.Select({
  layers: [kmlLayer]
});
map.addInteraction(selectClick);
```

Talált elemek hozzáadása a tömbhöz

name attribútumok beszerzése

div tartalom módosítása

popup pozíció beállítása

Vezérlők

Az előző kódot bővítjük

```
// terkep vezerlok  
// koordinata kijelzes a jobb felső sarokban  
var mousePosition = new ol.control.MousePosition({  
  coordinateFormat: ol.coordinate.createStringXY(4),  
  projection: 'EPSG:4326'  
});  
map.addControl(mousePosition);  
// lépték bal alsó sarok  
map.addControl(new ol.control.ScaleLine());  
// toloka nagyitas  
map.addControl(new ol.control.ZoomSlider());  
// attekinto terkep  
map.addControl(new ol.control.OverviewMap({  
  layers: [osmLayer]  
}));
```

A vezérlők látható
elemek

Egy kis CSS

```
<style>
  #popup {
    background: yellow;
  }

  .ol-scale-line {
    position: absolute;
    left: 50px;
  }
</style>
```

Módosítsuk az arborétum nevek megjelenítését, legyen sárga a háttér.

A stílus beállítások kerüljenek a ol.css betöltése után!

A lépték vonalzó és az áttekintő térkép ikonja eltakarja egymást. Mozdítsuk el a lépték vonalzót !
(*.ol-scale-line az OpenLayers-ben létrehozott stílus osztály*)

Az eredmény

3.1. Displaying a Scale Li... x OpenLayers - OpenS... x CSS Layout - Horizo... x +

localhost/js/ol_osm4.html ol.control.ScaleLine

20.0384, 47.8316

Egér pozíció WGS84

Nagyítás tolóka

Erdőrezervátum

Áttekintő és lépték

QGIS2WEB modul

- TODO

Feladatok

- A [html://www.geod.bme.hu/on_line](http://www.geod.bme.hu/on_line) oldalon található JavaScriptek átírása jQuery használatára
- A GPS hét és másodperc átszámító program kibővítése az visszafelé történő átszámítással
- Honlapba beágyazható digitális vagy analóg óra
- Szétnyitható/összezárható lista/al-lista (jQuery hide)
- Egyszerű (hasznos) számítás űrlap adatokból
pl. hőmérséklet átszámítás (Celsius, Fahrenheit, Kelvin)
- Navigációs GPS-ről letöltött GPX állomány megjelenítése OSM térképen
- Homogén koordinátás síkbeli transzformációk grafikus megjelenítéssel (svg vagy WebGL)