**Лекция Ивайло - Document Object Model DOM API** се състои от обекти и методи за правене на по-атрактивни HTML страници. - добавяне и отстраняване на HTML елементи - прилагане на стилове динамично - добавяне и отстраняване на HTML атрибути document.documentElement document.body За всеки HTML таг си има съответния ДОМ обектен тип. - HTMLLIElement -

 - HTMLAudioElement Всеки от тези обекти има подходящите пропертита. HTMLAnchorElement си имат съответно href property HTMLImageElement - src property document обектът е специален обект, който ни дава вход за достъпа до всички тези елементи. Другото е какви пропертита имат самите обекти. Всички атрибути, които имат съответните HTML елементи, са рефлектирали като пропертита. Стантартните неща като id, class, dragable, style, onclick и т.н. ги имат във всеки единт обект. var li = document.getElementById("ivanli")

* Ivan

Ако напишем: li.innerHTML = "Pesho" - променя съдържанието на елемента и при ъпдейт на страницата се показва Пешо вместо Иван. По този начин могат да се правят динамични странички. HTML елементите също така имат пропертита, съответстващи на тяхното съдържание. Тези са важните неща: .innetHTML - връща стринг със съдържание на елемента без самия елемент .outerHTML - връща като стринг съдържанието на елемента заедно с елемента – с таговете. .innerText/textContent - връща като стринг текстовото съдържание на елемента без таговете. Може да селектира и навътре в този таг, ако има и под него и връща само съдържанието без таговете. var li = document.getElementById('ivanli'); li.innerHTML += ' Another Click'; и въвежда този анкър в страницата безпроблемно. Ако иска обаче да се изпечата кода, а не самата визуализация като препратка, използва li.innerText. - li.innerHTML = ''; //това пък ще изтрие съдържанието на ли елемента. Най-бързия начин да изтрием нещо - селектираме го и му даваме да е празен стринг. li.id = 'Pesho' - можем да сменим id-то на елемента li.style.color = 'red'; - може да промени стила на елемента.

Get Name //oncklick - изпълнява някаква функция, която можем да закачим тук.

**Лекция Дончо 20.07.2015 г.**

**SVG,     kineticJS,**

По-бавно е от Canvas. Но дава по-добро качество на изображението на вякаква резолюция.

Точки, линии, криви, правоъгълници.

<svg>

<rect x = ""......>

<line x1 = "".....>

</svg>

stroke = ""  като атрибут на line - даваме цвета на линията.   stroke-width = "5" за дебелината на линията.

Points, line x1, y1, x2, y2,  stroke - начална и крайна точка, но трябва и stroke !!

rect - x1, y1, width, height, fill, stroke   fill = "none" - тогава е празна фигурата. Горния ляв ъгъл на правоъгълника се задава с точката.

<circle cx = "150" cy="250" radius="50" />

Друго е

<path d="M 50 50 L 175 310 H210 V10 Z" fill = "none" stroke = "purple" />

Когато е с малки букви вътре, задава релативни позиции - Главни букви - абсолютни координати;

М си е винаги главна за да не е релативно тръгването. Z накрая прави линията затворена. М - move  L може да замени H  или V, но се искат две координати.

Image to SVG - да го разгледаме този сайт.

Криви на Безие - кубична Q и  C  При кубичната имаме 2 точки в средата са които са точки за издърпване при другата крива имаме 1 точка в средата.

Ако използаме буквата Т и кривата ни се удължава - продължава се кривата. Можем да продължим и с нова буква С.

SVG & DOM

document.createElement(svgNs, 'rect ')

path.setAttribute('d', 'M 100 100 L 160 140 .......') Цялото поставя в една функция със setTimeout

за анимация ползва във функция  requestAnimationFrame(animFrame);

raphaelJS - библиотека

Лекция втора /след почивката/ Дончо

 Библиотеки за HTML5 Canvas и SVG

KineticJS - библиотека за работа с Кенвас.

Canvas

npm install -g bower

bower search kinetic

bower install -g kineticjs

Така може да я инсталираме библиотеката.

Продължение на лекцията Дончо. 20.07.15

Сега за анимации. Топче, което се отразява от стените на Кенваса

requestAnimationFrame

**21.07.15 Лекция Ивайло**

**DOM Operations  - Операции върху DOM дървото**

Как можем да обикаляме DOM дървото. Как можем да манитулираме DOM -а Добавяне, триене, и т.н.

Всяко парче от страницата може да се селектира и променя. Това е Ванила JS Обикновено се ползват различни библиотеки, които улесняват работата.

През JS може да направим много повече неща, отколкото в html-a;

Как обикаляме DOM дървото. Структкура от данни, които има разклонения. Всеки един елемент има точно 0 или 1 родител и 0 или много деца. HTML-a е такава дървовидна структура от данни.

Няма как да знаем кой е по-първи елемент. Подредбата може да е различна, Различни начини има по които може да се обикаля едно дърво. Как може да вземем нещо, което ни върши работа.

Как да вземем или родителя, или някой от децата му. Братя или сестри на съответния елемент. Намиращи се на същото ниво в дървото. Sibling Elements

 - element.parentNode.parentNode......    така вървим нагоре по йерархията

- element.childNodes - но дава и празните интервали.

- element.nodeName - дава името

По - правилно е да ползваме getElementBy.....

Имаме пропертита firstChild, lastChild, firstElementChild  nextSibling, previousElementSibling    null  ако не намери нищо връща null. Така може да го проверим.

Манипулиране на DOM  дървото -

Създаване на елементи - Имаме няколко варианта Най-бързия вариант за добавяне е ul.innerHTML +=  content. Като перди това в content сме добавили елементите с цикъл.

document.createElement(elementName). li.innerHTML = '123jkl;js';   Но така създаден елементът трябва да бъде добавен към DOM  дървото, т.е към . Иначе не се визуализира.

ul.appendChild(li). Преди това може да се стилизира, може да му се добави съдържание ( .innerHTML) Добавянето с appendChild e сравнително бавен начин.

Имаме също така командата   parent.insertBefore(new Node, specificElement)

Триене на елементи - Можем да трием нещо, което е дете на друго нещо. Не можем да селектираме елемента и да кажем да се изтрие-

ul.parentNode.removeChild(ul) - така може да изтрием ul елемента. За триене на съдържание ul.innerHTML = "".  ul.outherHTML = "".

Промяна на елементите - добавяне и триене на атрибути, от Гугъл може да се намери бързо инфо как да се изтрие или промени в nativeJS.

Когато местим някакъв елемент, той си върви със стиловете и референцията към него.

div.style...... - променя ин лайн стилове генерирани с JS. Когато по този начин сетнем стила, той оверрайдва стилизирането със CSS. Може да се постави imprtant! ако е необходимо.

При добавяне на CSS да не се забравят px, em  и т.н. т.е. units. Те се добавят като стрингове към някакви стойности.

По.добрия начин е да апендваме класове, и стилизирането през тях.

DocumentFragment - с него може да се добави наведнъж много елементи. Работи като StringBulder = Това е бързия начин за добавяне Важно е да добавяме по този начин За изпита също!!!

Също бавна операция е document.createElement('element')

DOMElement.cloneNode(true)  - по-бързо работи. от горното.

Трите оптимизации са това - за създаване на елементи - създаваме си, добавяме му характеристиките, после с горния ред.

JS event Model

Продължение на лекцията Евент моделът в JS

Създаване на евенти.

Какво представлява евент обекта

Cross-Browser EventHandler

Когато юзъра направи нещо, да се изпълни някакво парче код. Може да се закачан евенти върху всеки дом елемент. Повечето евенти може да се прихващат автоматично.

При hover, click, touch events, Form events

Mouse Events - click, hover, mouseup, ousedown, mouseover, mouseout

Keyboard Events keydown, keypress, keyup

UI Events - load - да се изпълни когато се зареди цялото DOM  дърво. change, select, resize,

Focus Events

Touch Events

Закачване на Евенти на елементите  onclick = "function()" onchange = "function()" onkeyup = "function()" As HTML Attributes

Най-адекватния вариант за закачване на евенти е с   domElement.addEventListener(eventType(onclick), eventHandler(function()), isCaptureEvent(true/false))

ev.target - върху какво изпълняваме евента.   ev e като аргумент на функцията и го ползваме после.

if(!ev) event......от лекцията. Там има и как през някаква функция да закакчваме, така че да работи на по-старите браузъри. Така няма да има проблеми при по-стари браузъри.

EventChain

Custom Events