|  |
| --- |
| **Laboratorio 3 – Mercoledì 21/10/2020** |
| * Per modificare il documento utilizziamo il cosiddetto DOM, un albero che viene creato al momento del caricamento del browser. * Javascript permette la modifica di elementi della pagina: prima cosa che dobbiamo saper fare è recuperare questi elementi all'interno dell'albero DOM. * **Metodi per ottenere gli elementi DOM**: per modificare il documento utilizzeremo l'oggetto document      * Proprietà e metodi per cambiare gli elementi DOM:     **Attenzione**: come già detto con Marcelloni i nomi degli stili sono diversi (si rimuovono gli scores e si pongono lettere maiuscole per distinguere le varie parole)    Nell’esempio vediamo font-size -> fontSize   * **Proprietà e metodi per aggiungere o cancellare elementi**:     Nel seguente esempio abbiamo creato un elemento button, abbiamo impostato il suo contenuto e infine lo abbiamo inserito in fondo all’elemento body |
| **Eventi**   * Gli eventi sono importantissimi all’interno di Javascript. * Abbiamo visto con Marcelloni che esistono vari modi per creare *listener*. * Il metodo suggerito dal prof. Tesconi è l’addEventListener.      * Gli eventi più utilizzati sono i seguenti   + click, che può essere utilizzato non solo su bottoni ma anche su anchors, paragrafi, su un qualunque elemento HTML…   + change, si intercetta il cambiamento di una casella di input   + scroll, quando si fa scrolling della pagina (si usa l’ascensore per scendere o salire)   + drag, relativo allo spostamento di elementi   + DOMContentLoaded, evento che si scatena con il caricamento completato del DOM. Può essere utile quando la pagina da caricare è estremamente complessa: il DOM potrebbe richiedere più tempo per essere caricato e Javascript, nel frattempo, potrebbe provare a modificare un albero non ancora completato. La cosa può provocare errori.   + resize, ridimensionamento della pagina. Utile se vogliamo fare schermate che si adattano alla dimensione della finestra del browser. * **Lista completa degli eventi**: *https://www.w3schools.com/jsref/dom\_obj\_event.asp* |
| **Riscaldamento**     * Il codice pone un input di testo in cui noi andiamo a indicare un nome * L’addEventListener associa all’elemento button con id mybtn un *listener*: quando qualcuno clicca l’elemento si modifica il contenuto dell’elemento avente id output. * Questo contenuto include il nome che abbiamo indicato nell’input * **Output**: |

|  |
| --- |
| **setInterval() e setTimeout()**   * Le seguenti funzioni sono importantissime per realizzare animazioni:   + setInterval() chiama una funzione ogni tot millisecondi e continua ad invocarla fino a quando non viene fermata con clearInterval() o viene chiusa la finestra   + setTimeout() chiama una funzione dopo tot millisecondi, salvo chiusura della finestra o chiamata della funzione clearTimeout(). * **Esempio**: |
| **Progetto *libretto universitario***   * Tesconi ha introdotto il progetto Libretto universitario creato dal prof. Tanganelli  (docente che ha curato i laboratori degli anni passati) * **File index.html**   <!DOCTYPE hmtl>  <html>  <head>  <meta charset="utf-8">  <link rel="shortcut icon" type="image/x-icon" href="./css/img/favicon.ico" />  <title>Libretto universitario</title>  <link rel="stylesheet" href="./css/libretto.css" type="text/css" media="screen">  <script type="text/javascript" src="./js/libretto.js"></script> <!-- js -->  </head>  <body onLoad="main()">  <h2> Analisi statistica dei voti riportati negli esami </h2>  </body>  </html> L’index è molto semplice: è presente il riferimento al CSS e al Javascript, mentre nel body è presente solo il titolo della pagina. Il contenuto vero e proprio verrà inserito dalla funzione main(), triggerata dall’evento onLoad. Tesconi è insofferente sull’inserimento di tutto questo codice HTML attraverso Javascript: avrebbe preferito avere larga parte del codice direttamente nella index.   * **File js/libretto.js**   Andiamo ad analizzare i vari aspetti del progetto:   * + **Gestione di messaggi rivolti all’utente e egli errori**: possiamo raccogliere messaggi stampati all’interno del documento in un array. Questo permetterà di recuperarli in modo veloce. Tra questi messaggi possiamo porre anche quelli di errore: segue un esempio di funzione.   *// Array di messaggi utente, istanziati prima che la pagina venga visualizzata*  MESSAGGI\_UTENTE = [  "Inserisci una sequenza di voti tra 18 e 33",  " non e` un numero",  " e` un numero minore di 18 o maggiore di 33"  ];  *// gestore dei messaggi di errore*  function stampaErrore(dato, codMess) {  window.alert("Errore: '" + dato + "' " + MESSAGGI\_UTENTE[codMess]);  }   * + **Programmazione ad oggetti**: Tanganelli ha creato un oggetto di nome Statistico. Questo oggetto presenta attributi con i dati richiesti (quelli che stamperemo: minimo, massimo, media, variabilità) e funzioni utilizzate per calcolare i vari dati. L’associazione delle funzioni avviene con l’attributo prototype.   function Statistico(dati) {  this.voti = this.analizzaDati(dati);  this.min = 0;  this.mas = 0;  this.med = null;  this.variab = null;  }  [… per le funzioni vedere più avanti]   * + analizzaDati() La funzione viene eseguita quando si inizializza un’istanza dell’oggetto. Data una stringa in ingresso restituisce     - un array con i voti se tutto va bene     - *null* nel caso siano stati trovati problemi (caratteri non numerici o voti non validi)   Statistico.prototype.analizzaDati = function(datiInput) {  *// Array contenente i voti degli esami inseriti dall'utente (gli elementi sono delle strighe)*  var dati = datiInput.split(";");  *// Array da restituire contenenti i voti degli esami (gli elementi sono degli interi)*  var voti = new Array();    *// Per ogni voto inserito dall'utente*  for (var i = 0; i < dati.length; i++) {  var voto = Number(dati[i]);  if (isNaN(voto)) { *// controllo se è un numero valido*  stampaErrore(dati[i],1);  return null;  }   *// controllo se è compreso tra 18 e 33*  else if (voto<18 || voto>33){  stampaErrore(voto,2);  return null;  }    voti[i] = voto;  }    return voti;  }   * + - Con la split genero un array (poniamo come separatore il punto e virgola)     - Creo un *Array* voti     - Col for controllo gli elementi dell’array generato precedentemente       * Se il singolo voto non è un numero stampo l’errore 1 e restituisco null.       * Se il singolo voto non rispetta le convenzioni universitarie stampo l’errore 2 e restituisco null.       * Se non si hanno errori il voto è valido e può essere incluso nell’array voti.     - Restituisco l’array voti.   + **Output**: riprendiamo il codice della funzione main()   function main() {  var voti = window.prompt(MESSAGGI\_UTENTE[0]);  if (voti == null)  return;    var stat = new Statistico(voti);  if (!stat.datiOk())  return;    stat.calcolaMinimo();  stat.calcolaMassimo();  stat.calcolaMedia();  stat.calcolaVariabilita();  stat.stampa();  }   * + - Si richiede l’inserimento di una lista di voti con la funzione prompt()  (Il testo è preso dall’array citato all’inizio).     - Si verifica che sia stato inserito del contenuto: in caso contrario ci si ferma subito.     - Si crea un istanza di Statistico ponendo in ingresso la stringa voti.     - SI verifica con la funzione datiOk() che tutti i dati siano corretti.  In caso contrario mi fermo.     - Eseguo una serie di funzioni per calcolare minimo, massimo, media e variabilità.     - Eseguo un’ultima funzione per stampare i risultati e il contenuto vero e proprio della pagina HTML.   + datiOk() semplice funzione che verifica il valore in uscita della analizzaDati(). Dobbiamo solo verificare che l’uscita non sia *null*.   Statistico.prototype.datiOk = function() {  return this.voti != null;  }   * + calcolaMinimo()   Si scorrono gli elementi dell’array membro this.voti, dall’ultimo al primo elemento. Ogni volta si aggiorna il minimo utilizzando la funzione Math.min(). Alla fine si aggiorna la variabile membro this.min ponendo il minimo trovato.  Statistico.prototype.calcolaMinimo = function() {  var minimo = this.voti[0];  for (var i = this.voti.length-1; i > 0 ; i--)  minimo = Math.min(minimo, this.voti[i]);    this.min = minimo;  }   * + calcolaMassimo() Stesso meccanismo di prima per trovare il massimo. Si utilizza la funzione Math.max()   Statistico.prototype.calcolaMassimo = function() {  var massimo = this.voti[0];  for (var i = this.voti.length-1; i > 0 ; i--)  massimo = Math.max(massimo, this.voti[i]);    this.mas = massimo;  }   * + calcolaMedia() funzione che restituisce un array contenente due valori:     - Media quantitativa (valore numerico)     - Media qualitativa (stringa contenente un giudizio deciso sulla base   Statistico.prototype.calcolaMedia = function() {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  CALCOLO MEDIA QUANTITATIVA  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  var i = 0, media = 0;  while (i < this.voti.length) {  media += this.voti[i];  i++;  }  media /= this.voti.length;  media = Math.round(media\*100)/100;    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  CALCOLO MEDIA QUALITATIVA  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  var mediaQual = null;  switch(Math.floor((media-18)/3)) {  case 0:  mediaQual = "sufficiente";  break;  case 1:  mediaQual = "discreta";  break;  case 2:  mediaQual = "buona";  break;  case 3:  mediaQual = "distinta";  break;  case 4:  mediaQual = "ottima";  break;  default:  mediaQual = "eccellente";  }    this.med = { numerica: media, qualitativa: mediaQual };  }   * + - Per il calcolo della media quantitativa:       * Sommo tutti i voti presenti nell’array this.voti       * Divido per il numero di voti (lunghezza dell’array, this.voti.length)       * Con la Math.round()arrotondo a due cifre decimali.         + Moltiplico per 100, spostando la virgola di due posizioni avanti.         + Arrotondo con la round (cambia eventualmente l’ultima cifra intera)         + Divido per 100, spostando la virgola di due posizioni indietro.         + Ottengo un numero che ha solo due cifre decimali.     - Per la media qualitativa:       * Prendo la media quantitativa, sottraggo 18, divido per 3 e arrotondo per difetto con la floor.       * Ottengo un numero compreso tra 0 e 5. Vediamo i casi limite:         + 0: (18-18)/3=0         + 5: (33-18)/3=5       * Con la switch individuo quale valore qualitativo si addice alla mia media.   + calcolaVariabilita() funzione che restituisce un array contenente due valori     - Variabilità quantitativa (valore numerico)     - Variabilità qualitativa (stringa)   Con variabilità intendiamo la seguente formula  Dove consiste nell’i-esimo elemento, nella media ed nel numero di elementi coinvolti. Si considera una somma di valori assoluti per evitare che alcuni termini si annullino.  Statistico.prototype.calcolaVariabilita = function() {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  CALCOLO VARIABILTA' QUANTITATIVA  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  var i = 0, varia = 0;  do {  varia += Math.abs(this.voti[i]-this.med.numerica);  i++;  } while (i < this.voti.length);    varia /= this.voti.length;  varia = Math.round(varia\*100)/100;    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  CALCOLO VARIABILTA' QUALITATIVA  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  var variabQual = null;  switch(Math.ceil(varia/7.5\*3)) { *// 7.5 massima variabilita*  case 1:  variabQual = "bassa";  break;  case 2:  variabQual = "normale";  break;  case 3:  variabQual = "alta";  default:  variabQual = "nessuna";  break;  }  this.variab = { numerica: varia, qualitativa: variabQual };  }   * + - Per la varianza quantitativa:       * Sommo i valori assoluti (utilizzando la Math.abs) delle differenze viste prima.       * Divido per il numero di elementi dell’array.       * Arrotondo alla seconda cifra esattamente come abbiamo fatto prima.     - Per la varianza qualitativa:       * Divido la varianza quantitativa per 7.5, moltiplico per 3 e arrotondo per eccesso con la funzione ceil.       * Con la switch individuo un valore qualitativo che si addice alla mia varianza quantitativa.   + stampa() ultima funzione eseguita nella main(), permette di stampare il contenuto della pagina includendo i dati inseriti e i risultati calcolati poco prima. All’interno sono chiamate due ulteriori funzioni: stampaTabellaVoti() e stampaDatiStatistici()   Statistico.prototype.stampa = function() {  *// Stampa intestazione pagina HTML (doctype e head)*  document.writeln("<!DOCTYPE hmtl>");  document.writeln("<html><head><meta charset=\"utf-8\">");  document.writeln("<link rel=\"shortcut icon\" type=\"image/x-icon\" href=\"./css/img/favicon.ico\"/>");  document.writeln("<title>Libretto universitario</title>");  document.writeln("<link rel=\"stylesheet\" href=\"./css/libretto.css\" type=\"text/css\" media=\"screen\"> <!-- css --></head>");    *// Qua inizia il <body>*  document.writeln("<body>");  document.writeln("<div id=\"wrapper\">");  document.writeln("<div id=\"topnav\"><img src=\"./css/img/unipi\_logo.png\" alt=\"Logo\"></div>");  document.writeln("<p>Libretto Universitario</p>");    this.stampaTabellaVoti();  this.stampaDatiStatistici();    document.writeln("</div>");  document.writeln("</body>");  document.writeln("</html");  }   * + stampaDatiStatistici() stampa dei risultati calcolati con le funzioni spiegate prima della stampa()   Statistico.prototype.stampaDatiStatistici = function() {  document.writeln("<div id=\"datiStatistici\">");  document.writeln("Minimo: " + this.min + "<br>");  document.writeln("Massimo: " + this.mas + "<br>");  document.writeln("Media: " + this.med.numerica + " (" + this.med.qualitativa + ")<br>");  document.writeln("Variabilit&agrave;: " + this.variab.numerica + " (" + this.variab.qualitativa +")");  document.writeln("</div>");  }   * + stampaTabellaVoti() stampa dei valori inseriti con la prompt() sottoforma di tabella   Statistico.prototype.stampaTabellaVoti = function() {  document.writeln("<div id=\"tabellaVoti\">");  document.writeln("<table>")  document.writeln("<caption>Elenco Esami</caption>");  document.writeln("<tr><th>Voti");    for (var i = 0; i < this.voti.length; i++)  document.writeln("<tr><td>" + this.voti[i]);    document.writeln("</table>");  document.writeln("</div>")  } |

|  |
| --- |
| **Esercizio: bottoni che aumentano/riducono la dimensione del font**     * Abbiamo la frase Hello world!!! che inizialmente ha dimensione di 20px (impostata da CSS) * Abbiamo inserito due bottoni: uno con id plus e uno con id minus. * Abbiamo associato delle funzioni a due eventi:   + *click* sul bottone con id plus: funzione che incrementa la dimensione del font;   + *click* sul bottone con id minus: funzione che decrementa la dimensione del font; * Entrambe le funzioni agiscono tenendo conto di una variabile fontSize inizializzata a 20 (la stessa dimensione del carattere impostata attraverso CSS. Dopo aver incrementato o decrementato la variabile aggiornano la proprietà del CSS col nuovo valore (si concatena la variabile aggiornata a px) * **Esempio di output**: |

|  |
| --- |
| **Esercizio: bar chart**  Il seguente esercizio mira a creare un bar chart, cioè un diagramma a barre il cui contenuto è stabilito da un input. Precisamente: si pone una lista di numeri, separati da virgola, in ingresso; si crea una barra per ciascun numero; ciascuna barra avrà una lunghezza in px pari al numero indicato.     * Associo al click del pulsante con id button una funzione:   + Genero un array dividendo la stringa posta in ingresso con l’input (la virgola è il separatore)   + Per ogni numero presente nell’array generato:     - Creo un nuovo div     - Associo al div appena creato la classe bar (che presenta certe proprietà grafiche - CSS)     - Imposto la lunghezza della barra alterando il width.     - Aggiungo la barra nel contenuto del div output. * **Effetto collaterale**: con la funzione append concateniamo nuovo codice con quello già presente all’interno del div class output. Questo significa che nuovi dati comporteranno l’aggiunta di nuove barre senza eliminare le precedenti (a meno che non si ricarichi la pagina). * **Esempio di output**: |
| **Esercizio: indovina il numero**  Proviamo a realizzare un gioco molto semplice dove l’utente deve indovinare un numero. Il computer, al caricamento della pagina, genera un numero random. L’utente prova a indovinarlo attraverso un input: la pagina risponde dicendo se ha indovinato (mostrando il num. di tentativi) o se ha indicato un numero maggiore o minore.     * Con le funzioni Math.floor e Math.random() generiamo il numero random.  Ricordiamoci quanto detto durante la lezione di Marcelloni relativamente al generare numeri random. * Si utilizza la getElementById per ottenere gli oggetti del DOM da manipolare (il div con id output dove stamperemo le risposte del gioco e il bottone con id button che premiamo per inviare gli input) * Si inizializza con valore 0 una variabile contatore num\_tentativi per ricordarsi il numero di tentativi usati (sarà stampata in caso di vittoria) * Si associa una funzione all’evento *onclick*:   + Incremento i tentativi   + Stampo il numero di tentativi nella console (questo è per comodità nostra, per debuggare)   + Con getElementById(“numero”).value estraiamo il valore posto nell’input   + Con una serie di *if-statement* confronto il numero inserito con quello generato da Javascript e agisco di conseguenza modificando il contenuto del div output. * **Esempio di output**: |