**EL3-10\_4ST TIMER**

**Autore: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Data: \_\_\_\_\_\_\_\_\_Classe: \_\_\_\_**

**ATTENZIONE**

Il significato dei simboli usati in questa e nelle prossime esercitazioni è spiegato dettagliatamente nella guida <http://www.classiperlo.altervista.org/Materiale/Generale/Simboli.doc> (scaricala e consultala in caso di dubbi).

|  |
| --- |
|  |

Il simbolo della manina in colore blu indica una domanda alla quale bisogna OBBLIGATORIAMENTE rispondere scrivendo in colore blu.

|  |
| --- |
|  |

Il simbolo della manina in colore blu con la scritta Cou New indica un codice che va incollato usando Courier New in colore blu.

|  |
| --- |
|  |

Il simbolo della manina in nero indica una o più immagini o schermate da incollare (protette col tuo watermark, le tue iniziali di Nome e Cognome)

Il simbolo della manina con colori attenuati indica un'operazione che bisogna svolgere, senza rispondere a nessuna domanda (non vuol dire che non devi fare nulla - significa solo che non devi scrivere niente!).

Il simbolo della manina col ciak video indica un video da registrare con *Gif Recorder* (<http://gifrecorder.com/>)e da salvare in formato gif nella cartella dell'esercitazione.

recupero.

E:\Dropbox\Doc\Sito Web\classiperlo\immagini\video.gifQuesto simbolo indica un video di esempio o di spiegazioni da guardare su YouTube

**A) OPERAZIONI PRELIMINARI**

A1) Compila l'intestazione del documento scrivendo Nome, Cognome e Classe

A2) Crea una sottocartella di ES3 con nome uguale a quello di questa esercitazione (*EL3-10\_4ST Timer*)

A3) All'interno della sottocartella *EL3-10\_4ST Timer* salva questo file Word

**B) AVVIARE UN TIMER**

Spesso si ha la necessità di eseguire ripetutamente una funzione a intervalli di tempo prefissati. Per esempio, potrei voler ripetere qualcosa (una funzione) ogni 60 secondi.

Questo è molto utile per visualizzare un orologio o un cronometro che si aggiorna automaticamente o per far muovere un oggetto su una pagina.

Javascript possiede una comoda funzione di library per realizzare questo: *setInterval*. Ecco come si usa:

setInterval(funzione, milli);

dove *funzione* è il nome della funzione che si vuole ripetere e *milli* è un intervallo di tempo espresso in millisecondi.

**FUNZIONI DI CALLBACK**

Nel nostro esempio alla funzione setInterval vengono passati due parametri:

1) l’intervallo di tempo milli

2) la function funzione

Osserva che, mentre milli è una normale variabile, funzione è una function. Cioè, alla function setInterval noi passiamo un’altra funzione…

Questo meccanismo in base al quale una function riceve come parametro un’altra function, prende il nome di callback. Nel nostro caso funzione è una **funzione di callback**.

Si noti che le funzioni di callback, quando vengono passate a un’altra funzione, NON DEVONO mai avere le parentesi tonde. Dunque è sbagliato scrivere per esempio

setInterval(calcola(), 2500);

mentre questo è il modo corretto di richiamare la funzione di callback calcola (senza le parentesi):

setInterval(calcola, 2500);

Vediamo subito un esempio pratico:

B1) → *timer1.html (HTML + CSS + JS, insieme)*

Nella parte riservata al codice JS, copia e incolla il seguente codice:

function avviaTimer()

{

setInterval(visualizza, 1500);

}

function visualizza()

{

alert("Ciao!");

}

B2) Nell’evento onLoad sull’elemento body inserisci una chiamata alla funzione *avviaTimer* e poi osserva cosa succede…

**IN ALTERNATIVA**

Per eseguire qualcosa al caricamento di una pagina esiste un metodo alternativo rispetto a onLoad. In pratica basta eliminare la funzione avviaTimer e lasciare solo il suo codice interno, in questo modo:

setInterval(visualizza, 1500);

function visualizza()

{

alert("Ciao!");

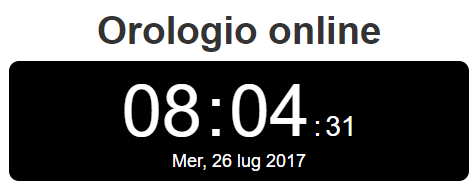
}

Così facendo, l'istruzione setInterval viene eseguita automaticamente al caricamento dell'elemento <script> che la contiene. Prova!

Come vedi la funzione *visualizza* viene chiamata automaticamente ogni 1500 millisecondi, cioè ogni 1,5 secondi. In pratica è come se *setInterval* generasse un evento automatico (una specie di onClick virtuale), eseguito allo scadere dell'intervallo fissato.

**C) UN OROLOGIO**

Un esempio interessante di uso di questa funzione è la creazione di un orologio che si aggiorna automaticamente:



C1) → *timer2.html, timer2.css, timer2.js*

Usando gli stili crea una pagina che mostra un orologio simile a quello precedente (non deve per forza essere uguale, fai come piace a te). Lascia pure vuoto il contenuto dei P o dei DIV che hai usato.

E:\Dropbox\Doc\Sito Web\classiperlo\immagini\video.gif<https://youtu.be/eVJgO4tTEEs>

Usando *setInterval* avvia un timer che si aggiorna ogni mezzo secondo (l'orologio deve aggiornarsi ogni secondo, ma conviene controllare l'ora ogni mezzo secondo, per evitare di perdere gli istanti di aggiornamento):

setInterval(funzioneAssociataTimer, 500);

C2) Nella funzione associata al timer, usa i metodi dell'oggetto Date() per leggere ora, minuti, secondi e data correnti. Visualizza quindi tutto dentro i paragrafi o div che hai usato nella tua pagina.

|  |
| --- |
|  |

C3) → Schermata browser

È probabile che la tua pagina visualizzi i valori minori di 10 in questo modo:



invece che così:



Per formattare i valori in modo da aggiungerci davanti uno zero quando sono minori di 10, puoi usare questa funzione di elaborazione già scritta da me:

// Funzione di servizio per formattare i numeri minori di 10  
function checkTime(i)   
    {  
    if (i < 10) {i = "0" + i}; // aggiunge uno zero davanti ai numeri < 10  
    return i;  
    }

|  |
| --- |
|  |

C4) → Codice JS

C5) → *timer2.gif*

**D) FERMARE IL TIMER: UN CONTASECONDI**

Se si vuole interrompere l'esecuzione di un timer avviato precedentemente, occorre usare una variabile in cui memorizzare il timer al momento dell'avvio, in questo modo:

memTimer = setInterval(funz, milli);

Tale variabile (che io ho chiamato memTimer ma che tu puoi chiamare come vuoi) potrà poi essere usata per interrompere il timer usando la funzione *clearInterval* come nell'esempio seguente:

clearInterval (memTimer);

La variabile memTimer, che collega fra loro l'avvio del timer (setInterval) al suo stop (clearInterval), dev'essere dichiarata **globale**.

|  |
| --- |
|  |

D1) Spiegami perché deve essere dichiarata globale:

D2) → *timer3.html, timer3.css, timer3.js*

Vogliamo usare una pagina che mostra un contasecondi (da 0 in avanti). La pagina visualizza i secondi in una casella di testo o in un paragrafo e ci sono due pulsanti, uno per avviare il contasecondi e l'altro per fermarlo.

E:\Dropbox\Doc\Sito Web\classiperlo\immagini\video.gif<https://youtu.be/ZehQc91YFnE>

Usa *setInterval* per incrementare i secondi a intervalli di 1000 ms (non servono in questo caso i metodi dell'oggetto Date).

|  |
| --- |
|  |

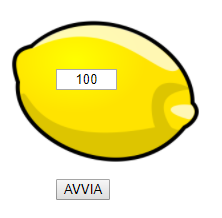
D3) → Codice JS

|  |
| --- |
|  |

D4) → Schermata browser

**E) UN CONTASECONDI A DECREMENTO**

Vogliamo adesso realizzare un contasecondi da cucina tipo quello mostrato in figura:



Il contasecondi si compone di tre DIV:

1. un DIV con un'immagine di sfondo (io ho scelto un limone, ma potete scegliere un'immagine a piacere - scaricatela nella cartella di questa esercitazione)
2. un DIV con una casella di testo (sovrapposta al limone), nel quale l'utente scrive il numerod i secondi da contare
3. un DIV con il pulsante AVVIA

E1) → *timer4.html, timer4.css, timer4.js*

Usa il posizionamento con gli stili (absolute per esempio). Non dimenticarti di assegnare una dimensione ai blocchi da posizionare. Utilizza gli Strumenti per sviluppatori di Chrome per ottenere un posizionamento preciso.

|  |
| --- |
|  |

E2) → Schermata browser

E:\Dropbox\Doc\Sito Web\classiperlo\immagini\video.gif<https://youtu.be/M1Zkx7hrz7k>

Premendo sul pulsante AVVIA, si avvia un timer con un conto alla rovescia a decremento, partendo dal valore fornito nella casella di testo. Quando tale valore si azzera, il timer si ferma automaticamente (per farlo ripartire bisogna scrivere un nuovo valore e premere AVVIA).

|  |
| --- |
|  |

E3) → Codice JS

Per migliorare l'aspetto del nostro timer, vogliamo fare in modo che all'avvio del contasecondi l'immagine (il limone, nel mio caso) si metta a ruotare e si fermi quando i secondi sono scaduti.

**ANIMAZIONI CSS IN JAVASCRIPT**

Per far ruotare l'immagine usa le animazioni CSS (<http://www.programmiamo.altervista.org/HTML/dinamiche/din2c.html> ).

Usa una classe in cui inserisci i parametri dell'animazione, come per esempio:

.ruota

{

animation-name: animazione;

animation-duration: 4s;

animation-iteration-count: infinite;

animation-timing-function: ease-in-out;

}

Per far partire l'animazione, devi aggiungere la classe all'elemento DIV con il metodo:

document.getElementById("limone").className = "ruota";

Per fermare l'animazione devi rimuovere la classe, con il metodo:

document.getElementById("limone").classList.remove("ruota");

|  |
| --- |
|  |

E4) → Codice CSS

|  |
| --- |
|  |

E5) → Codice JS

**F) OPERAZIONI FINALI**

F1) Controlla di aver risposto a tutte le domande e incollato tutte le schermate. Tutte le caselline dovrebbero avere un segno X, per indicare che hai risposto 

F2) Comprimi le immagini contenute in questo file Word (seleziona un'immagine, scheda *Formato* e poi *Comprimi immagini* e infine *Applica a tutte le immagini del documento*) in modo da ridurne le dimensioni.

F3) Controlla che la cartella di questa esercitazione contenga i seguenti file con i nomi qui indicati:

|  |  |  |
| --- | --- | --- |
| **Nome del file** | **Tipo del file** | **Descrizione** |
| *EL3-10\_4ST Timer* | Word | Il file di questa esercitazione |
| *timer1.html* | HTML+CSS+JS |  |
| *timer2.html* | HTML |  |
| *timer2.css* | CSS |  |
| *timer2.jss (cambia estensione)* | JS |  |
| *timer3.html* | HTML |  |
| *timer3.css* | CSS |  |
| *timer3.jss (cambia estensione)* | JS |  |
| *timer4.html* | HTML |  |
| *timer4.css* | CSS |  |
| *timer4.jss (cambia estensione)* | JS |  |
| *...* | IMG | Immagine sfondo contasecondi |
| *timer2.gif* | GIF |  |

F4) Chiudi tutti i file, zippa la cartella di questa esercitazione e inviala all'insegnante su Classiperlo.