Web development & Azure

Evento Microsoft Student Partners Italia

WebApp on Azure

Hey! Ciao a tutti!

- Leonardo Piccoli

- Microsoft Student Partner
- Leonardo.Piccoli@studentpartner.com
- Twitter: @3lpleo

- Federico Parezzan

- Microsoft Student Partner
- Federico.Parezzan@studentpartner.com
- Twitter: @FParezzan

Microsoft Student Partners

Cosa sono gli MSP

- I Microsoft Student Partner sono degli studenti con un particolare background tecnico.
- Offrono il loro tempo per insegnare, sperimentare e presentare eventi riguardanti l'utilizzo delle nuove tecnologie.
- Sono presenti in tutta Italia ed organizzano eventi come questo!

Microsoft per gli studenti

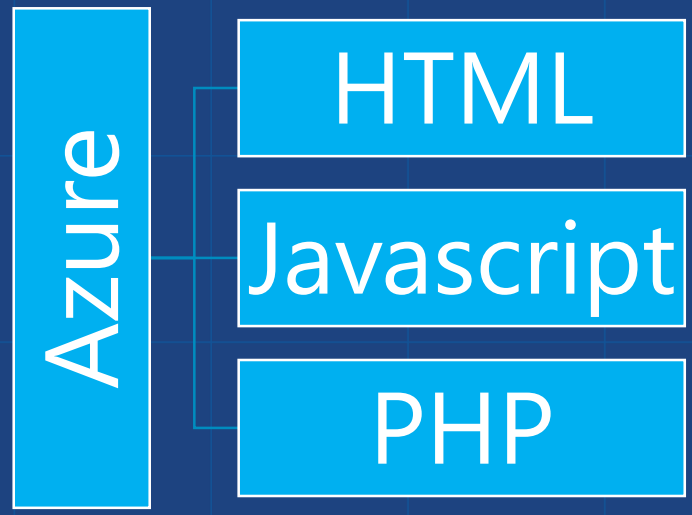
Tecnologie gratuite

- Un programma che sostiene l'istruzione tecnica fornendo accesso a software Microsoft a scopi di apprendimento, didattica e ricerca.
  - Microsoft Imagine Azure
  - Account sviluppatore su Dev Center
  - Xamarin for Students
  - E molto altro!

Tecnologie utilizzate per il laboratorio

‘Front End’ & ‘Back End’

- Il Front End, cioè il codice eseguito sulla macchina client è rappresentato dal javascript e dall’HTML.
- Il Back End, cioè il codice eseguito sul server è il PHP.
- L’hosting, cioè dove viene ospitato il tutto, è dato dal server Azure.



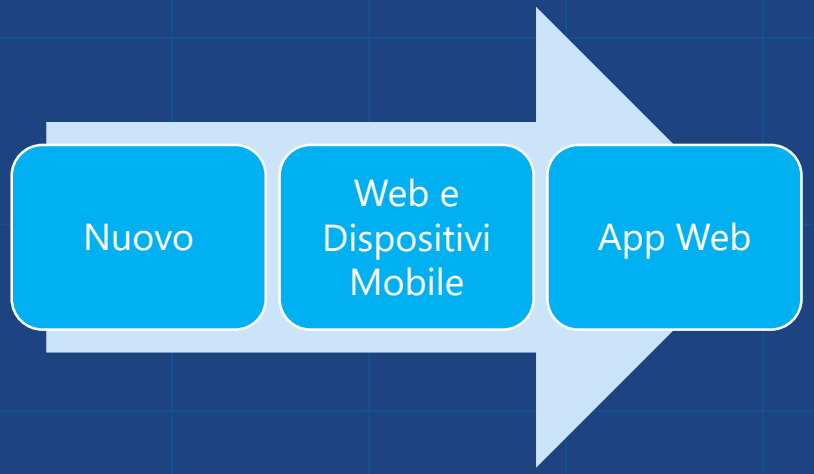
Let’s GO!

1 - Accedere ad AZURE

- Ora vi daremo il codice che vi permetterà di accedere ad Azure, dove potrete caricare il vostro progetto una volta finito e metterlo online.
- Quindi per prima cosa dobbiamo andare sul sito di azure -> https://azure.microsoft.com/it-it/

2 – Inside Azure

- Procediamo ora a creare una nuova applicazione Web premendo su questi menu:



Collegare FileZilla a Azure

- Nella schermata dell'applicazione cliccare Get Publish Profile e aprire il file appena scaricato con un editor di testo.
- Una volta aperto, aprire il software «FileZilla» e cliccare file -> gestore siti…
- Una volta che si è aperto il gestore siti, creare un nuovo sito.

- Immettere nel campo «Host» del nuovo sito la stringa simile a questa presente nel file scaricato precedentemente:
- publishUrl="ftp://waws-prod-db3-029.ftp.azurewebsites.windows.net/site/wwwroot"
- Della stringa trovata teniamo solo la parte tra virgolette: ftp://waws-prod-db3-029.ftp.azurewebsites.windows.net
- NB cancellare la parte finale /site/wwwroot!

Collegare FileZilla a Azure – Parte II

- Identificare nel file scaricato al stringa userName="univrprova\$univrprova. Copiare la parte univrprova\$univrprova (che dovrebbe avere lo stesso nome della vostra applicazione web.) nel campo Utente di Filezilla.
- Cambiare Il campo «Criptazione» di FileZilla in «Usa solo FTP non sicuro».
- Cambiare il campo «Tipo di Accesso» di FileZilla in «Richiedi Password»
- Cliccare il pulsante «connetti» in FileZilla
- Quando richiede la password, identificare nel file scaricato la stringa userPWD="LPBa1iz6cPeJfddnsyRAq2wwekPuvFwK2EVVmLBytvcdTFHTcjaYytkMEoozxP« e copiare la parte tra virgolette nel campo password.
- Infine, cliccare ok! Se tutto è stato configurato correttamente, FileZilla dovrebbe essere collegato alla cartella del vostro nuovo progetto!

Iniziamo il progetto!

- Cancelliamo tutto il contenuto della cartella /site/wwwroot… Inizieremo da una pagina bianca!
- Bene siamo pronti! Adesso possiamo spostarci sul desktop e scaricare all'indirizzo https://github.com/lpleo/galileiWebApp l'applicazione. Vi troverete davanti 3 cartelle, una contenente queste slide, una contenente le soluzioni e la cartella che ci interessa, denominata Esercizio.

- Dentro la cartella Esercizio troviamo tre file:

1. Index.html (completo)
2. Stile.css (completo)
3. Logica.js (il vero esercizio!)

Le funzioni

- Definiremo l’intero programma all’interno del file Logica.js.
- Iniziamo dalla prima funzione (chiamata rules()) che come vedete è richiamata tramite la stringa <body onload="rules()"> nel file index.html.

Funzione Rules()

```
function rules() {
        var rules="Premere sui pulsanti nella stessa sequenza in cui si spengono\n Il gioco inizierà appena chiusa questa finestra!\n Inserisci il tuo nome:";
        var nome = prompt(rules);
        nome = ((nome==null) ? 'Ospite' : nome);
        document.getElementById("nome").innerHTML = document.getElementById("nome").innerHTML.concat( nome);
        inizio_gioco();
}
```

- Osserviamo la funzione:

1. Nella variabile rules definiamo il testo che apparirà a schermo.
2. Nella variabile nome salviamo l'input dell'utente, cioè il nome scritto nella finestra che è apparsa all'inizio del gioco.
3. Testiamo che il nome non sia vuoto, nel caso lo sia inseriamo la stringa «Ospite»
4. Scriviamo il nome all'interno della pagina html Index.
5. Richiamiamo la funzione che effettivamente da inizio al gioco!

Funzione inizio_gioco()

```
function inizio_gioco() {
        punteggio=0;
        volte=3;
        aggiungiListener();
        inizia(volte);
}
```

- La funzione di inizio gioco si occupa solamente di inizializzare le variabili esterne e fare dei controlli sui **Listener**.
- Cosa sono il **Listener**? I **listener** sono ‘ascoltatori’, cioè porzioni di codice che hanno come obiettivo intercettare determinati input da parte degli utenti.
- Dopo aver richiamato la funzione aggiungiListener(), richiamo la funzione inizia passando come parametro il numero di volte che voglio far lampeggiare un quadrato.

Funzione aggiungiListener()

```
function aggiungiListener() {
    document.getElementById("q1").addEventListener("click", function() {aggiungi_seq_ins("1");});
    document.getElementById("q2").addEventListener("click", function() {aggiungi_seq_ins("2");});
    document.getElementById("q3").addEventListener("click", function() {aggiungi_seq_ins("3");});
    document.getElementById("q4").addEventListener("click", function() {aggiungi_seq_ins("4");});
}
```

- All'interno della funzione aggiungi_listener() andiamo ad aggiungere l'evento 'click' alle div identificate dagli id q1,q2,q3,q4. Questo ci permette di associare il click su una di queste div (per capirsi, uno dei quattro quadrati) ad un evento.
- Nel nostro caso l'evento è la chiamata della funzione aggiungi_seq_ins() e passa come parametro una stringa.
- Come vedremo prossimamente, la funzione aggiungi_seq_ins() serve per tenere traccia dei quadrati premuti dall'utente.

Funzione inizia()

```
function inizia(volte) {
        sequenza_inserita="";
        sequenza="";
        inizializza_array(volte);
        colora(volte);
}
```

- All'interno di questa funzione proseguiamo con l'inizializzazione delle variabili che ci serviranno prossimamente.
- Inizializziamo le stringhe «sequenza_inserita» e «sequenza» a vuoto.
- Richiamiamo le funzioni inizializza_array() e colora() passando come parametro il numero di volte.

Funzione inizializza_array()

```
function inizializza_array(volte) {
        finito = new Array();
        for(var i=0;i<volte;i++) {
                finito.push(false);
        }
}
```

- All'interno di questa funzione andremo a costruire un array contenente un numero 'volte' di «**false**».
- Questo array ci servirà più tardi per controllare se è terminata la sequenza di inserimento, poiché la sequenza di inserimento pone l'intero array a **true** quando termina.

Funzione colora() Prima Parte

```
function colora(volte) {
        var millis = 0;
        for(var i=0;i<volte;i++) {
                millis=1000*i;
                setTimeout(function() {colora_quadrato();},
millis);
                setTimeout(function() {colora_originale();},
millis+500);
        }
        setTimeout(function(){alert('Tocca a
te!');},millis+1000);
}
```

- All'interno della funzione colora() coloriamo i vari quadrati prima di bianco e dopo li riportiamo al colore di partenza.
- Questo serve a dare l'illusione di accensione/spegnimento che crea la sequenza.
- Questo avviene tramite la funzione colora_quadrato e colora_originale che sono **asincrone.**

Funzione colora() Seconda Parte

- Cos'è una funzione **asincrona?**

1. Una funzione asincrona è una funzione che viene eseguita in maniera indipendente dal normale corso del programma.
2. Il programma cioè non aspetta la terminazione della funzione per continuare con le successive operazioni.

- Per questo motivo abbiamo inizializzato prima un array di controllo.
- Ci servirà per testare che effettivamente la sequenza abbia terminato, prima di permettere all'utente di cliccare sui quadrati e quindi di registrare la sua sequenza.

Funzione colora_quadrato() Prima Parte

```
function colora_quadrato() {
        var quad = (Math.random()*100);
        var id="q";
        if(quad<25) {
                id+="1";
                sequenza += "1";
                }
        if(quad>=25 && quad <50) {
                id+="2";
                sequenza+="2";
        }
        if(quad>=50 && quad <75) {
                id+="3";
                sequenza+="3";
        }
        if(quad>=75) {
                sequenza+="4";
                id+="4";
        }
        var div = document.getElementById( id );
        div.style.backgroundColor = 'white';
        ritorna_true();
}
```

Funzione colora_quadrato() Seconda Parte

- All'interno di questa funzione andiamo a colorare un quadrato scelto a random tramite la funzione Math.random().
- Questo ci permette di riempire la sequenza generata dal computer (variabile 'sequenza') e di colorare il quadrato di bianco ogni volta che questa funzione viene richiamata.
- Al termine della funzione andiamo a richiamare la funzione ritorna_true(), che serve per mettere a **true** il primo valore a false dell'array 'finito'.
- Questo ci serve per controllare che l'utente non clicchi e generi parti della sequenza prima che la sequenza generata dal computer sia effettivamente finita.

Funzione colora_originale()

```
function colora_originale() {
        document.getElementById( "q1" ).style.backgroundColor = "red";
        document.getElementById( "q2" ).style.backgroundColor = "green";
        document.getElementById( "q3" ).style.backgroundColor = "blue";
        document.getElementById( "q4" ).style.backgroundColor = "yellow";
}
```

- Tramite la funzione colora_originale(), ricoloriamo tutti i quadrati del loro colore originale.
- Concludendo, in tutto il processo, viene colorato un quadrato di bianco e dopo mezzo secondo vengono ricolorati tutti i quadrati dello stesso colore. Questo da l'impressione che il quadrato «lampeggi».

Funzione aggiungi_seq_ins() Prima Parte

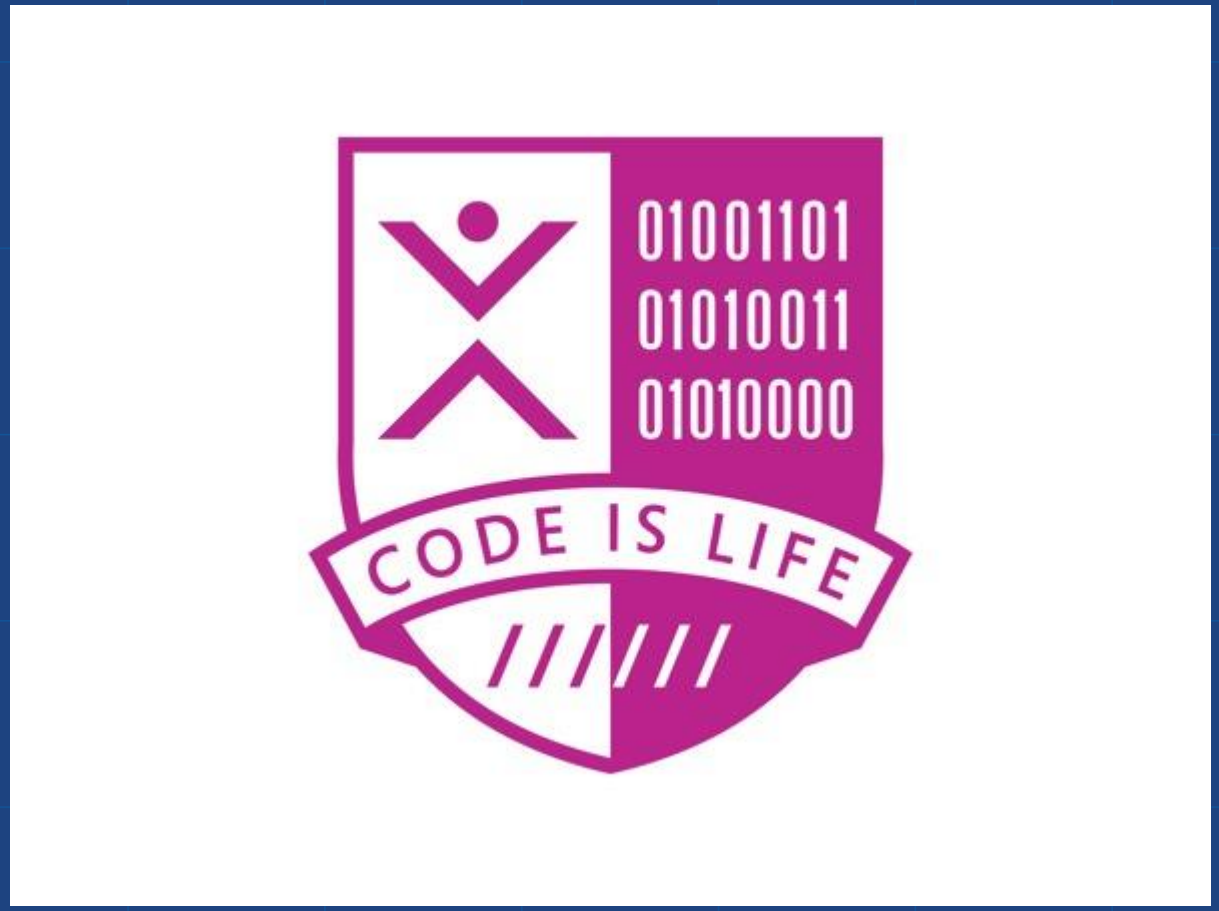
```
function aggiungi_seq_ins(numero) {
        if(testa_array()) {
                sequenza_inserita+=numero;
                if(sequenza_inserita.length == sequenza.length) {
                        if(sequenza_inserita==sequenza) {
                                alert("Perfetto sequenza indovinata!");
                                punteggio += 100;
                                var testo = "Punteggio Attuale: ";
                                document.getElementById("p_attuale").innerHTML =
testo.concat(punteggio);
                                volte+=1;
                                inizia(volte);
                        } else {
                                 alert("Hai perso!");
                                 document.getElementById('button_restart').style.display =
'block';
                        } } } }
```

Funzione aggiungi_seq_ins() Seconda Parte

- Questa funzione è la funzione che racchiude un po' tutti i ragionamenti fatti in precedenza .
- Per prima cosa controlliamo se la funzione testa_array() ritorna **true**. Se non lo fa, vuol dire che è ancora in corso la «stampa» della sequenza da parte del computer e quindi non accettiamo input.
- Invece, se riceviamo **true** vuol dire che la sequenza di stampa da parte del computer è terminata. Possiamo registrare il numero del quadrato premuto.
- Dopo aver salvato il numero del quadrato premuto all'interno della variabile 'sequenza_inserita', ne confrontiamo la lunghezza della stringa 'sequenza' (generata dal computer). Se le lunghezze sono uguali vuol dire che sono stati premuti tanti quadrati quanti sono stati quelli stampati dal computer.
- Quindi… Adesso possiamo confrontare le sequenze! Se le sequenze sono uguali l'utente ha passato il turno, poiché ha ripetuto correttamente la sequenza generata dal computer.
- Continua…

Funzione aggiungi_seq_ins() Terza Parte

- Quindi se la sequenza è stata indovinata, aumento e stampo il punteggio, aumento il numero di volte (cioè il numero di volte che i quadrati devono lampeggiare) e richiamo la funzione Inizia passando come parametro il nuovo numero di volte.
- Se invece la sequenza non è stata indovinata, compare un avviso e un pulsante che ti permette di ricominciare la partita. Per farlo, il pulsante non fa nient'altro che ricaricare la pagina. Tutto si azzera e ricomincia il procedimento!



GRAZIE PER L'ATTENZIONE!
And now… Question time!