1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO

UNIVERSITÀ DEGLI STUDI DI URBINO CARLO BO

Dipartimento di Scienze Pure e Applicate
Corso di Laurea in Informatica Applicata

Tesi di Laurea

COS'È BLAZOR?

Relatore:
Chiar.mo Prof. Emanuele Lattanzi

Candidato:
Francesco Belacca

Anno Accademico 2018-2019

A tutti quelli che mi hanno detto almeno una volta di non mollare.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

1.1 Contesto

A seguito della crescita esponenziale del web in questo secolo e dell'abituarsi di coloro che ne usufruiscono ad un livello grafico sempre migliore e ad una esperienza mano a mano più interattiva e vicina all'utente medio, i siti web e le tecnologie utilizzate si sono adattati per permettere uno sviluppo sempre più rapido di codice più facilmente testabile e mantenibile.

Di conseguenza nel frontend si sono susseguiti una serie di framework a partire da JQuery nel 2006, che per primo si è occupato di risolvere il problema della compatibilità tra browsers, permettendo ai developers di scrivere una volta, e poter eseguire su tutti i browsers.

AngularJS nel 2010 è stato il primo MVC framework ad offrire in un unico pacchetto, two-way data binding, dependency injection, routing facilitato e altri strumenti utili per rendere più standard lo sviluppo nel frontend [1]. Dopo la riscrittura di questo framework nel 2013 che è diventato Angular 2 (e recentemente Angular) senza mantenere retrocompatibilità e senza offrire un modo preciso per migrare alla nuova versione agli utilizzatori di AngularJS, React, un nuovo framework più leggero e modulare sviluppato dagli sviluppatori di Facebook, ha preso il posto di Angular come framework più utilizzato.

Vue infine é il terzo dei principali framework che ha provato a prendere piede proponendo una versione intermedia tra il fortemente opinionato Angular e il piú flessibile React.

Oltre a questi, ciascuno con la propria semantica, organizzazione logica dei folder, spesso una CLI dedicata, ad un developer frontend viene solitamente richiesto di conoscere HTML, CSS e Javascript su cui si basano poi i vari framework.

Oltre a Javascript, se si vuole scrivere degli unit test facilmente mantenibili, bisogna conoscere TypeScript(specialmente se si utilizza Angular, che rende il suo utilizzo obbligatorio) e degli altri framework che facilitino i test(Enzyme, Karma + Jasmine, ...).

1.2 Problema

I continui cambiamenti nei molti framework utilizzati, la diversità degli strumenti stessi tra loco, che spesso realizzano in modo diverso tutti la stessa cosa(framework concorrenti), rendono specialmente per un junior developer molto ampia la curva di apprendimento e lo studio necessario, per essere anche solo operativo.

Oltretutto un developer ad oggi finisce per essere costretto a scegliere se diventare uno sviluppatore frontend o backend, dato che rimanere al passo e aggiornarsi già in uno di questi due campi richiede tempo e non è scontato che venga concesso di poterlo fare in orario lavorativo, pur essendo fondamentale.

Microsoft ha reso nel tempo il framework .NET e le sue implementazioni(.NET Core, .NET Framework e Mono) utilizzabili nei vari linguaggi supportati, C#, F# e VB. Scrivendo ad esempio in C# è possibile sviluppare vari tipi di applicazioni, ma se si decide di sviluppare codice per un applicazione client web ad oggi si è ancora costretti a scrivere utilizzando Javascript e un suo framework se si vuole essere utili e competitivi a livello enterprise.

Razor ha provato a risolvere parzialmente il problema di generare codice HTML e CSS in modo dinamico utilizzando C#, ma è utilizzabile solo lato server e quindi ad esempio la cattura di un evento client side come il click di un utente su un bottone senza contattare il server non è gestibile utilizzando il solo C#.

Capitolo 2

Wireless LAN

2.1 Stato Attuale delle Wireless Network

Oggigiorno non esistono più dispositivi isolati che non comunicano con altri, tutti i PC prevedono la connessione ad internet, o in generale comunicano con altri dispositivi elettronici grazie alle reti di comunicazione. Inizialmente queste reti erano basate sui cavi (*wired network*), poi si sono rivelate più pratiche le reti senza filo (*wireless network*); da ciò si è dovuto affrontare il problema di permettere l'utilizzo delle tecnologie attualmente utilizzate nelle reti wired anche in reti wireless, che per loro natura sono intrinsecamente meno sicure a causa del mezzo trasmissivo. Le informazioni non vengono più convertite sotto forma di impulso elettronico e spedite su un filo metallico, ma essendo l'etere il mezzo di propagazione sono convertite in onde radio; queste ultime hanno una portata limitata non sempre ben determinabile, sono soggette a disturbi che degradano le informazioni inviate fino, a volte, a farle perdere; ed infine cosa a noi maggiormente sgradita l'energia consumata e direttamente connessa alla potenza trasmessa, ciò significa che aumentare la portata di trasmissione e la qualità del segnale comporta un consumo maggiore dell'energia consumata.

2.2 Dynamic Power Management

Essendo la parte a radio frequenza colpevole del maggior consumo di energia, in particolare gli amplificatori utilizzati immediatamente prima dell'invio del segnale o dopo una ricezione, le stazioni 802.11 possono allungare la vita delle batterie spegnendo i transceiver radio e mettendo la schada wireless in sleep periodicamente.

Nella figura sottostante (Figura **??**), si può osservare una finestra di 20 secondi del consumo di energia di una scheda wireless senza il power management abilitato:

Attivando il power management si può notare un sensibile risparmio energetico, a fronte di un diverso consumo energetico dovuto ad un diverso modo di operare:

In questa modalità la stazione resta per la maggior parte del tempo nello stato di sleep, risvegliandosi periodicamente (nella figura ogni 100 ms) per controllare se nel frattempo c'è stato traffico per essa. Durante il periodo di sleep, sarà demandato all'Access Point il compito di bufferizzare i frame destinati verso la scheda momentaneamente in sleep; al risveglio i frame bufferizzati saranno annunciati da un *Beacon frame*, la ricezione da parte della stazione wireless dei frame bufferizzati inizierà subito dopo con la loro richiesta tramite il *PS-Poll frame* (Figura 2.1).

Quando viene attivato il power management l'Access Point e la stazione wireless devono sincronizzarsi per risvegliarsi ad intervalli prestabiliti, che nel caso della figura soprastante è 100 ms; all'inizio di ogni risveglio l'Access Point invierà il beacon contenente la *traffic indication map - TIM* che dice alla stazione se vi sono pacchetti bufferizzati destinati ad essa; in tal cosa la stazione risponderà con un PS-Pool contenente il suo Association ID. Grazie all'Association ID l'Access Point sarà in grado di selezionare i frame destinati alla stazione e provvederà ad inviarli.

Bibliografia

[1] M. Wanyoike Disponibile: https://blog.logrocket.com/history-of-frontend-frameworks

Ringraziamenti

Vorrei ringraziare il professor Lattanzi per avermi aiutato a scrivere questa tesi, i miei genitori per avermi permesso di procedere facendo di testa mia su un percorso a loro estraneo, e la mia ragazza Ivana per aver creduto in me e nelle mie capacità anche quando non l'ho fatto io.