Node.js e i suoi concorrenti

Node.js ha sostanzialmente tre concorrenti tra i linguaggi di solito scelti per implementare soluzioni web molto complesse.

PHP

A parte l'esperimento HHVM compiuto da Facebook, il design di PHP non è mutato anche nelle versioni più recenti. Il modello I/O resta di tipo blocking, e l'abbinamento con MySQL rende praticamente impossibile concepire un approccio asincrono. Ne consegue che la performance di PHP richiede un enorme lavoro a livello sistemistico per poter affrontare le sfide di un web sempre più orientato verso prestazioni di elevato livello.

Ad aggravare la situazione si aggiunge un ecosistema in cui accanto a prodotti di livello come Laravel e Zend si affiancano implementazioni scadenti sia sotto il profilo del codice che sotto il profilo dell'aderenza ad uno standard comune che dovrebbe garantire scalabilità e sicurezza.

PHP ha subito notevoli cambiamenti a livello di caratteristiche introdotte e di ottimizzazione delle prestazioni a partire dalla versione 5.4 in poi.

Tuttavia il core design del linguaggio non è cambiato: PHP è un linguaggio **sincrono** basato su un modello I/O di tipo **blocking**.

Ciò significa che se PHP effettua ad esempio una query ad un database MySQL non può compiere altre operazioni finché i risultati della query non sono disponibili.

Anche nella manipolazione del file system le operazioni compiute da PHP sono sincrone senza offrire alcuna possibilità di temporizzare in modo asimmetrico i task svolti.

Ripetiamo: tutte le operazioni sono sincrone. Il tempo di esecuzione del codice cresce in modo direttamente proporzionale al numero di file e directory presenti. Quando si raggiunge e si supera l'ordine delle centinaia, anche su una macchina con risorse adeguate (ad esempio 32 GB di RAM, processore 3,2 GHz Intel Core i5) è necessario modificare le risorse a disposizione di PHP sul file `php.ini` per evitare crash.

PHP resta fondamentale

A dispetto di quanto affermino i suoi detrattori, PHP non è affatto morto ma al contrario gode di ottima salute. Varie sono le ragioni alla base di questa sua condizione.

Innanzitutto PHP è un linguaggio **maturo** con più di 20 anni di sviluppo alle spalle. Questo lo rende un linguaggio stabile e consolidato.

PHP è anche un linguaggio **estremamente diffuso** e facilmente installabile e configurabile: le maggiori distro Linux forniscono tutte i pacchetti fondamentali di PHP già ottimizzati per il sistema operativo di destinazione in uso sui server.

Inoltre PHP funziona molto bene in combinazione con i principali web server sul mercato, ossia **Apache ed nginx**.

PHP, in definitiva, si rivela essere lo strumento adatto per la maggior parte delle richieste da parte dei clienti: quando queste richieste sono particolari a livello di requisiti e quando l'accento viene posto sulle prestazioni, Node.js invece è un'alternativa da considerare. In altre parole, è diverso realizzare un semplice blog dallo sviluppare un CMS complesso.

Inoltre PHP è un ottimo modo per acquisire le competenze lato server necessarie in Node.js, come la gestione delle richieste HTTP, i cookie, le sessioni, l'autenticazione, i database e così via.

È più difficile imparare a conoscere Node se non si ha alcuna esperienza di programmazione lato server. La differenza non sta tanto nel linguaggio

usato, quanto piuttosto nella conoscenza dello specifico ambiente di sviluppo.

Python

Sicuramente Python è il linguaggio che combina più di altri funzionalità e prestazioni. Il numero di feature built-in supportate dal linguaggio è quasi pari a quello di Java e le prestazioni in termini di performance sono di gran lunga superiori a quelle di PHP.

Python ha la stessa maturità di Java e un design che, rispetto a quanto accade con PHP, favorisce e promuove la scrittura di codice di qualità.

Java

Java è un linguaggio che, come Python, ha enormi potenzialità per il fatto stesso di essere stato concepito come linguaggio di programmazione per soluzioni software molto più complesse di quelli che in media vengono richieste sul web.

Possiamo dire che Java sul web può esprimere il suo potenziale solo in minima parte: gestire un form, visualizzare documenti web, usare i template sono compiti di routine che non rendono appieno l'idea di quello che Java può fare quando elabora dati complessi.

Usare Java per il frontend di un sito poteva essere un'alternativa quando altri linguaggi specifici per il web erano ancora all'inizio della loro storia, ma oggi Java deve tornare al posto di rilievo che gli compete: se dobbiamo sviluppare un applicativo di tipo finanziario, la nostra scelta sarà orientata verso linguaggi come Java o Python. Ciò non impedisce di utilizzare Node.js per i livelli intermedi e di dialogare con Java tramite API REST basate su JSON. In questo modo potremo avere una soluzione potente sia sul frontend che sul backend.