

# SVILUPPARE APPLICAZIONI WEB CON IL FRAMEWORK PHP SYMFONY

Luca Saba, HAL Software, Cagliari, IT, lucasaba@gmail.com

## Abstract

*Lo sviluppo di applicazioni web offre diversi vantaggi rispetto allo sviluppo delle classiche applicazioni stand-alone. Tra i maggiori vantaggi ci sono quello di non dover distribuire aggiornamenti su tutte le macchine che utilizzano l'applicazione, l'indipendenza della piattaforma utilizzata e la facilità di integrazione con altre applicazioni web (Twitter, Facebook, YouTube, solo per citare le più famose).*

*Utilizzare un framework per lo sviluppo di applicazioni web porta ulteriori vantaggi. Se nella maggior parte dei casi, questi vantaggi sono legati al risparmio di tempo e alla possibilità di non reinventare la ruota ogni volta che si crea una nuova funzionalità, nel caso di Symfony si aggiungono altri diversi importantissimi aspetti tra cui*

- *la qualità della documentazione*
- *un'organizzazione del codice che favorisce ottime pratiche di programmazione*
- *facile estensibilità tramite plugin*
- *e ancora molto altro.*

*In sintonia con la filosofia di Symfony dell'imparare facendo, utilizzeremo come spunto una banale applicazione, un piccolo blog dove gli autori possono pubblicare i loro articoli e gli utenti possono commentarli, per vedere come l'utilizzo di Symfony ci permetta di scrivere, in poco tempo, del codice elegante, leggibile, ben organizzato, sicuro e testabile.*

*Per l'accesso alla base dati, Symfony permette di scegliere tra due principali O.R.M. (Object-Relational Mapping): Propel e Doctrine. Questi si occupano di automatizzare l'accesso alla base dati, trasformando i record del DB in oggetti corredati dai relativi getter e setter per l'accesso ai singoli campi. L'ORM permette anche di tener conto delle relazioni tra le tabelle della base dati, facilitando così l'accesso ai dati collegati (ad esempio, per recuperare tutti i post di un determinato autore). Inoltre, sia Propel che Doctrine, permettono di accedere in modo trasparente ai più comuni gestionali di basi dati, così che lo sviluppatore non si debba minimamente occupare di questi aspetti. Dall'interazione tra Symfony e l'ORM, vedremo come sia facile generare, all'interno dei nostri progetti, degli scheletri di moduli che gestiscano il CRUD degli oggetti (ossia la generazione automatica di pagine per i tipici accessi Create, Read, Update e Delete).*

*Molti dei moduli interni di Symfony sono dei mini-framework utilizzabili in modo indipendente dal resto dell'ambiente (come già succede in Zend). Tra questi vi è il framework dedicato alla gestione dei form e dei filtri. Perché scrivere la validazione dei campi quando la maggior parte delle informazioni sono già scritte nella definizione della base dati? Ecco che il form framework di symfony ci permette di non preoccuparci di questi aspetti. Abbiamo bisogno di controlli più raffinati sui dati? Nessun problema: basterà estendere gli oggetti a nostra disposizione nell'ambiente.*

*Il framework Symfony è estensibile tramite plugin. Esistono moltissimi plugin per Symfony che si occupano dei compiti più disparati. Vedremo come l'installazione e l'utilizzo di questi plugin sia semplice e ci dia immediato accesso a diverse funzionalità.*

*Per concludere, daremo uno sguardo all'ambiente di test di Symfony. La gestione dei test è stato uno degli aspetti più centrali nello sviluppo di Symfony ma anche, troppo spesso, uno degli elementi meno seguiti da chi programma. Con gli strumenti messi a disposizione da Symfony, diventa facile diventare test-dipendenti. La semplicità con la quale è possibile definire l'interazione dell'utente con l'applicazione, consente di scrivere dei test estremamente complessi ed esaustivi. Così che, quando avvengono delle modifiche nel nostro codice, l'avvio dei test ci da una documentata informazione sull'affidabilità del codice, facendo sì che il nostro "svn update" non sia più un'azione temeraria.*

*Parole Chiave: PHP, Symfony, MVC, test driven, best practice, CRUD, ORM, framework*