

03/11/08 - Programación Rápida de Webs

Cada lenguaje que es potencialmente útil para el desarrollo web, termina teniendo un *framework* para el desarrollo de aplicaciones web de forma rápida. A continuación pongo un listado de los lenguajes con sus respectivos entornos (los más usados y/o conocidos):

- **Ruby**: este lenguaje dio el salto al desarrollo web con su *framework* denominado *on Rails*. Es un sistema que ayuda al desarrollador para realizar un sitio web dinámico que requiera de acceso a datos y características típicas de los CMS, foros, webmails, etc.
- **Java**: el desarrollo de aplicaciones empresariales siempre ha estado marcado por Java. En el desarrollo web, Java usa tanto servlets como JSP. Un entorno muy conocido que se comenzó a usar en el patrón MVC con Java fue *Struts*. Se considera de los primeros. Luego hay otros que se consideran más completos, como por ejemplo *Spring*.
- **PHP**: en PHP realmente no haría falta ningún entorno en sí, pero ciertamente, para marcar el camino de un desarrollo grande para una empresa, debería de existir un método que permita a varios programadores no realizar código difícil de mantener. En este sentido, hay entornos como *CakePHP* o como *Symfony*, que facilitan pensar todos los aspectos relativos al desarrollo de una aplicación web grande y pone en manifiesto pocos patrones de desarrollo a seguir. En BosqueViejo tenemos nuestro propio framework para desarrollo de CRM: [Oak Framework](#).
- **Erlang**: dispone también de *ErlyWeb* que al igual que los *frameworks* de los demás lenguajes, permite el desarrollo rápido de elementos basados en base de datos y similares.
- **Python**: es un lenguaje que se ha basado, para la web, casi siempre en la existencia de Zope para el desarrollo de aplicaciones (o productos) de forma fácil y rápida. No obstante, hay mucha gente que comienza a usar otros *frameworks* como Django.
- **Groovy**: este lenguaje, concebido como lenguaje *embebido* para Java, se ha comenzado a usar como simplificación del mismo y uso en sistemas de *scripting* y con el *framework* Grails en la web.
- **Perl**: este lenguaje script, de los más rápidos en ejecución y con una sintaxis que, hace tiempo, se ganó el apodo de *write-only*, tiene también su *framework* para el desarrollo web, cuyo nombre es *Catalyst*, y se encuentra disponible en su propio CPAN.

Cada uno de estos lenguajes tiene su potencial y es más indicado para un tipo de desarrollo que los otros. Hay que tener en cuenta que, los lenguajes más estructurados (Python y Java, por ejemplo) tienen la propiedad de poder estructurar de forma correcta el código, con lo que son más indicados para proyectos grandes y muy configurables.

En cambio, para proyectos pequeños, el uso de otros lenguajes rápidos como Groovy, PHP, Perl o Ruby, está mejor indicado puesto que, aunque son lenguajes perfectamente válidos para desarrollos grandes, su potencia radica en la posibilidad de escribir código rápido y, muchas veces, de cualquier forma, y terminan convirtiéndose en códigos difíciles de mantener.

Sin embargo, hay muchos proyectos desarrollados en estos lenguajes que han demostrado que, con un poco de organización y una política algo estricta a la hora de escribir código, permiten al equipo de desarrollo mantener y hacer que un sistema web se haga grande a lo largo del tiempo, como han sido: WordPress, Drupal, eGroupware, LiveJournal, Radiant, etc.

Otros lenguajes de alta escalabilidad, como Erlang, que permite el desarrollo de aplicaciones en varios nodos (equipos) de forma fácil, sin necesidad de emplear aplicaciones y servidores extra, que al fin y al cabo, resultan siendo un SPOF más en nuestro sistema.

Como conclusión, podemos sacar que, para la web, cualquier lenguaje es válido, y sus respectivas comunidades nos lo demuestran día a día. Si el proyecto que hay que realizar es pequeño, introducirse en Java o Python (a menos que sea Plone y/o Zope) es algo costoso, mientras que si se hace en otro lenguaje de scripting, suele ser más rápido. Pero cabe recordar que, si el proyecto es grande, lo que hay que hacer, es un planteamiento y una línea general, una política de actuación y llevar siempre el código claro y limpio, sobre todo si lo queremos mantener y reutilizar en el futuro.

Entradas relacionadas de Programación