Ruby on Rails

Ruby e uma linguagem de programação voltada para web que visa viabilizar o desenvolvimento de softwares de forma simplificada dinamizando o projeto como um todo. O Rails e uma ferramenta usada com a linguagem Ruby que vem sendo bastante utilizada atualmente. Ruby é uma Linguagem de programação interpretada, com tipagem dinâmica e forte, orientada a objetos e, com várias semelhanças com Perl, Python e SmallTalk. Projetada tanto para a programação em grande escala quanto para codificação rápida, tem um suporte a orientação a objetos simples e prático. A linguagem foi criada pelo japonês Yukihiro Matsumoto, que aproveitou as melhores idéias das outras linguagens da época. Esta linguagem possui vastos repositórios de bibliotecas disponíveis em sites como Ruby Forge e Ruby Application Archive (RAA). Existe, ainda, uma ferramenta bastante útil para instalação de bibliotecas, chamada Ruby Gems, o software mais famoso desenvolvido em Ruby é o Ruby on Rails.

Para melhorar a interação com o usuário nas aplicações web, várias tecnologias continuam surgindo e evoluindo, como Ajax [Stamey e Richardson, 2006] JavaFX1, AdobeFlex2. Já para melhorar a organização da arquitetura de aplicações web no servidor, vários frameworks têm sido propostos: Struts, Spring, CakePHP, RIFE, Microsoft.NET (ASP.NET), entre outros. Estes frameworks têm o objetivo de auxiliar o desenvolvimento e a manutenção de aplicações voltadas para a web.

Observa­se que, atualmente, as tecnologias predominantes nas principais aplicações web são da Sun (Java) e Microsoft (plataforma .NET). Os principais frameworks para Java são Struts, Java Server Faces, Hibernate (persistência), Spring, e outros, utilizados por grandes empresas que desenvolvem as aplicações utilizando a plataforma Java Enterprise Edition (JEE). A Microsoft propõe uma solução completa em sua plataforma .NET .

Entretanto, todas as tecnologias são relativamente novas, e um grupo de desenvolvedores de software começou a contestar as soluções abertas feitas em Java por serem muitas vezes complicadas de utilizar, com muitas configurações para serem feitas, bem como a solução da Microsoft devido a questões comerciais. Com o objetivo de simplificar o desenvolvimento de aplicações web, começaram o desenvolvimento de um framework completamente novo, numa linguagem aberta – Ruby ­, o que resultou no Ruby on Rails, que foi lançado em meados de 2004.

História

Ruby se tornou reconhecida no meio especializado desde que Dave Thomas, conhecido como um dos "Programadores Pragmáticos", adotou-o como uma de suas linguagens preferidas e acabou por escrever um dos mais completos livros sobre a linguagem, o Programming Ruby. Com o advento desta publicação, a linguagem passou a contar com uma boa fonte de iniciação e referência em inglês, aumentando consequentemente o número de adeptos da linguagem no Ocidente.

Ultimamente, devido a grande exposição de um framework web feito em Ruby, o Ruby on Rails desenvolvido por David Heinemeier Hansson, a linguagem tem sido foco da mídia especializada justamente pela sua praticidade.

Esta mesma praticidade inclusive é um dos conceitos básicos desta linguagem. É possível fazer algoritmos que resolvam seus problemas, não necessitando se preocupar com as limitações da linguagem ou do interpretador.

Com o avanço da Internet, é possível observar um movimento de migração de aplicações Desktop para aplicações web, além da criação de novos tipos de aplicações que não existiam no mundo sem Internet, como as aplicações de comércio eletrônico e serviços governamentais oferecidos pela rede e diversas outras [Jazayeri, 2007].

Neste contexto, também evoluíram as tecnologias e técnicas para o desenvolvimento de aplicações web. Inicialmente, a maior parte das aplicações era desenvolvida em linguagem Perl ou como scripts CGI [Lerner, 1997] desenvolvidos em linguagem C e outras linguagens procedurais, sem utilização do paradigma de Orientação a Objetos. Posteriormente, foram desenvolvidas linguagens bastante voltadas para o desenvolvimento de aplicações web, como PHP [Riddell, 2000]. Entretanto, foi apenas após o ano 2000, época em que o acesso à Internet começou a se popularizar mais em todo o mundo, que começaram a ser desenvolvidas soluções com o objetivo de resolver os principais problemas existentes no desenvolvimento deste tipo de aplicações: as aplicações web são extremamente dinâmicas, não guardam estado, utilizam um protocolo simples para comunicação (HTTP [Fielding et al., 1999]) que, por definição, permite apenas tipos simples de interação e que deve ser iniciada sempre do lado do cliente, entre outros. A Figura 3.1 mostra a arquitetura básica de uma aplicação web, destacando as características de comunicação do protocolo HTTP.

Características

Para manter a praticidade, a linguagem possui algumas características interessantes:

A sintaxe é enxuta, quase não havendo necessidade de colchetes e outros caracteres.

Todas as variáveis são objetos, onde até os "tipos primitivos" (tais como inteiro, real, entre outros) são classes.

Estão disponíveis diversos métodos de geração de código em tempo real, como os "attribute accessors".

Através do Ruby Gems, é possível instalar e atualizar bibliotecas com uma linha de comando, de maneira similar ao APT do Debian Linux.

Code blocks (blocos de código), ajudam o programador a passar um trecho de instruções para um método. A idéia é semelhante aos "callbacks" do Java, mas de uma forma extremamente simples e bem implementada.

Mixins, uma forma de emular a herança múltipla, sem cair nos seus problemas.

Tipagem dinâmica, mas forte. Isso significa que todas as variáveis devem ter um tipo (fazer parte de uma classe), mas a classe pode ser alterada dinamicamente. Os "atalhos" citados acima, por exemplo, se beneficiam da tipagem dinâmica para criar os métodos de acesso/alteração das propriedades.

O Padrão de Projeto MVC

O framework Ruby on Rails, além de ter sido desenvolvido com o objetivo de aumentar a produtividade e manter baixa a curva de aprendizagem, também foi pensado como um framework que incentiva as boas práticas de codificação e organização das aplicações. Neste âmbito, deve ser destacada a arquitetura das aplicações desenvolvidas sobre Rails, que seguem o Padrão de Projeto (design pattern) MVC: Model­View­Controller [Sasine e Toal, 1995].

MVC é um padrão de projeto no qual as aplicações são desenvolvidas em camadas. Neste caso, existem as camadas de modelo, visão e controle. Em cada uma dessas camadas, deve­ser alocar diferentes partes da aplicação em desenvolvimento, conforme o seguinte critério:

● Modelo – representação das informações operadas pela aplicação. Por exemplo, author, title e abstract fazem parte do domínio de um sistema para revisão sistemática. Esta camada opera com os dados crus, e não possui a lógica necessária para o tratamento dos dados. Em Ruby on Rails, esta camada é manipulada pelo Active Record, que é uma implementação do design pattern Active Record.

● Visão – camada responsável por exibir os dados (modelo) em uma forma específica para a interação – no caso do Ruby on Rails, páginas web disponibilizadas no navegador. Rails fornece o framework Action View para compor a camada de visão.

● Controle – a camada de controle é responsável por processar e responder aos eventos, podendo invocar alterações na camada de modelo. Em Ruby on Rails, o Action Controller possui as funcionalidades para esta camada.

Entre algumas desvantagens do uso do Ruby on Rails estariam :

**Segurança:** o armazenamento de determinados dados da programação e feito atraves de cookies. Possíveis vulnerabilidades com inserções de javascript por usuários maliciosos.

**Sistemas de tipagem dinâmica:** apesar de dar mais prudutividade, aumenta-se o risco de uma exceção em tempo de execução.

**Modelagem :** em projetos de grande escala podem haver problemas com a produção dos modelos relacionais.

Esta ferramenta em conjunto com a linguagem tendem a encurtar a parte de engenharia do software, fazendo com que o projeto decorra de modo mais dinâmico mas também menos configurável e com algumas vulnerabilidades que podem ser contornadas. Ele vem sendo bem aceito pelos desenvolvedores por tornar mais aberta a parte criativa da programação menos formal, favorecendo claramente esta etapa do projeto.

Na verdade, Ruby on Rails é formado por um conjunto de cinco (5) outros frameworks, a saber: Action Mailer, Action Pack, Action Web Service, Active Record e Active Support. A seguir, uma breve descrição de suas funções:

1. **Action Mailer** é um framework para o desenvolvimento de camadas que contenham serviços de e­mail. Estas camadas são utilizadas para criar código para enviar senhas perdidas, mensagens de boas vindas durante a assinatura de determinado site ou serviço, e quaisquer outros casos de uso que requeiram uma notificação escrita a uma pessoa ou sistema. Além disso, uma classe Action Mailer pode ser utilizada para processar mensagens de email recebidas, como por exemplo permitir que um weblog receba postagens diretamente do email.

2. **Action Pack** é responsável por fazer a separação da resposta de uma requisição web em uma parte de controle (que faz a lógica) e outra de visão (processando um template). Esta abordagem é realizada em dois passos, para normalmente criar (create), ler (read), atualizar (update) ou deletar (delete) (CRUD – create, read, update, delete) a parte do modelo, que geralmente tem um banco de dados por trás, antes de escolher entre renderizar um template ou redirecionar para outra action. Neste framework são implementados o Action Controller e a Action View.

3. **Action Web Service** fornece mecanismos para publicar APIs de web services interoperacionais com Rails sem dispender muito tempo aprofundando nos detalhes de protocolo.

4. **Active Record** é responsável por fazer o mapeamento entre os objetos da aplicação e as tabelas da base de dados para criar um modelo de persistência do domínio onde a lógica e os dados são apresentados e tratados de uma única forma. É uma implementação do padrão (design pattern) Active Record.

5. **Active Support** é uma coleção de várias classes utilitárias e extensões de bibliotecas padrão que foram consideradas úteis para Rails. Todas estas bibliotecas facilitadoras de desenvolvimento foram empacotadas juntas. Embora não seja feito um detalhamento explícito das facilidades disponibilizadas por este framework, algumas de suas características são apresentadas.

Além de o framework ser escrito na linguagem Ruby, todas as aplicações desenvolvidas com o framework também devem ser desenvolvidas nesta linguagem. Conforme será apresentado mais adiante, mesmo os templates utilizam uma variação da linguagem, batizada de eRuby.

Criação de aplicações em Ruby on Rails podem ser feitas considerando-se:

● Criação do banco de dados: esta etapa consiste em criar o banco de dados da aplicação. Com isso, deve ser feito o mapeamento do modelo conceitual da aplicação para um banco de dados relacional.

● Criação da estrutura padrão: após instalado o framework na máquina de desenvolvimento, o comando “rails” pode ser utilizado para criar uma aplicação. A utilização deste comando é da seguinte forma: rails <nome\_da\_aplicação>. Desta forma, será criada uma aplicação contendo a estrutura padrão de qualquer aplicação desenvolvida em Rails. Além de poder ser criada com o comando “rails”, a IDE Aptana, indicada neste capítulo, auxilia o trabalho de criação inicial da aplicação. Basta solicitar a criação de um novo projeto na IDE para que a estrutura padrão da aplicação seja gerada.

● Configuração do banco de dados: após realizados os passos de criação do banco de dados e criação da estrutura padrão, deve ser configurado o arquivo config/database.yml para conter a base de dados de desenvolvimento. Os dados de conexão devem ser informados neste arquivo. Nesta etapa, devem ser informados os dados do banco de desenvolvimento.

● Criação de scaffolding: é sugerida a utilização de scaffolding para agilizar a criação das aplicações. Após configurado o banco de dados, pode­se utilizar o script de geração para se criar um scaffold da aplicação, da seguinte forma: script/generate scaffold papers. Como resultado disso, são criados o modelo, visão e controle para os artigos da aplicação de revisão sistemática tomada como exemplo, e as operações básicas de inserção, visualização, edição e listagem de artigos já passam a ser disponibilizadas. Os arquivos de modelo podem, então, ser editados. Este passo é ainda mais detalhado nesta seção.

As aplicações desenvolvidas em Ruby on Rails devem ter seus arquivos organizados segundo a estrutura sugerida pelo framework, na qual os arquivos específicos da aplicação são colocados abaixo do diretório app. Além disso, as camadas de modelo, visão e controle devem ter seus arquivos separados nos respectivos diretórios.

Para o desenvolvimento de aplicações com tal organização, Ruby on Rails disponibiliza um gerador automático de código bastante útil, o denominado scaffold. O que este gerador faz é criar arquivos de “exemplo” para aplicação, mas que são funcionais o suficiente para permitir a execução das ações de criação, edição, visualização, listagem e exclusão de registros das tabelas criadas no banco de dados.

Fontes:

<http://www.andersondamasio.com.br/post/2009/04/15/Ruby-on-Rails-Historia-Caracteristicas.aspx>

http://www2.icmc.usp.br/~renata/RoR-capitulo\_webmedia07.pdf