

Frameworks para Desenvolvimento em PHP

Elton Luís Minetto



CAPÍTULO 1

Introdução

Uma das grandes vantagens do PHP é sua facilidade de aprendizado. Ao ler poucas páginas de tutoriais ou de algum livro, um programador já é capaz de montar um formulário HTML e de criar um script PHP que processe os dados fornecidos pelo usuário. Isso favoreceu o rápido aumento do número de programadores e o surgimento de grandes softwares.

Contudo, essa facilidade de aprendizado permitiu o surgimento de programas, digamos, pouco eficientes, programas vulneráveis a ataques, com péssima manutenibilidade, os denominados *spaggeti code*, com PHP e HTML inseridos no mesmo script, e também a dificuldade de se trabalhar em equipes, entre outros. Um dos argumentos usados por programadores de outras linguagens era que “PHP facilita a geração de programas ruins”.

Argumentos como esse não são mais válidos. Vários fatores contribuíram para esta mudança: as novas e robustas características de orientação a objetos advindas do PHP5, o apoio de grandes empresas como a Oracle e a IBM, a profissionalização dos programadores, graças a certificações como a da Zend, e o surgimento do Ruby on Rails.

Sei que escrever esse parágrafo anterior em um livro sobre PHP é uma blasfêmia e, provavelmente, um exagero. Mas com o surgimento do Rails, vários programadores da comunidade *open source* em consenso de que programar para “o ambiente web”, muitas vezes, era uma tarefa repetitiva e maçante, perceberam que vários aspectos poderiam ser reaproveitados, e, principalmente, que programar poderia ser divertido. Não estou dizendo que não existissem frameworks para PHP antes disso, mas a reação de todos foi olhar com mais atenção para os existentes e criar novos, pensando nessas idéias.

Neste livro, pretende-se descrever alguns frameworks que estão sendo desenvolvidos e utilizados ativamente por uma comunidade novamente entusiasmada com a programação para um renovado ambiente web. Inicialmente comenta-se sobre o framework e suas vantagens. Comenta-se também sobre algumas tecnologias usadas pelos frameworks que serão apresentados.

Os capítulos podem ser lidos na ordem em que o leitor achar mais interessante, mas seguir a ordem em que aparecem facilita a comparação e, conseqüentemente, a identificação do leitor com framework que mais lhe agrada. Isso porque é muito difícil, em qualquer comparação deste porte, apontar o melhor. Para o programador, é sempre importante conhecer várias opções antes de escolher uma para utilizar ou contribuir, pois como dizem “a melhor ferramenta é a que você conhece e confia”.

1.1 PHP 5

O PHP foi criado inicialmente como uma linguagem de script estruturada, mas, com o passar dos anos, novos recursos foram sendo adicionados com o intuito de transformá-la em uma linguagem orientada a objetos. Na versão 4 da linguagem, já existia a possibilidade de se utilizar classes e objetos, mas de uma maneira ainda rudimentar.

Com o lançamento da versão 5 esse objetivo foi atingido com sucesso. a partir desse “amadurecimento” o PHP começou a chamar a atenção de programadores acostumados a utilizar o paradigma de programação orientada a objetos, além de favorecer a criação de aplicações mais robustas e de melhores componentes de software. Lógico que outras características importantes foram adicionadas ao PHP 5, como o SQLite, exceções e o SimpleXML, mas o fortalecimento da orientação a objetos foi de grande importância para o surgimento dos frameworks que serão demonstrados no decorrer deste livro.

No tópico 1.1.1 serão descritos alguns conceitos da orientação a objetos com exemplos em PHP. Vale lembrar que esse tópico é só uma revisão e, caso o leitor queira se aprofundar no maravilhoso mundo da orientação a objetos, existem vários livros que discorrem muito melhor sobre o assunto.

1.1.1 Orientação a objetos

1.1.1.1 Objetos

Um objeto é uma representação de um conceito real, possui estados, operações (métodos) e dados (atributos). Um objeto em PHP é definido da seguinte forma:

```
<?php
    //cria uma novo objeto
    $php = new LinguagemProgramacao;
    //um objeto possui atributos
    $php->versao = 5;
    //e um objeto possui um método
    $php->showVersao();
?>
```

1.1.1.2 Classes

Uma classe pode ser considerada como uma descrição, um molde para criar objetos. Ela define todos os dados e comportamentos possíveis daquele tipo de objetos. Dessa forma, um objeto é considerado uma instância de uma classe.

Para definir uma classe em PHP:

```
<?php
class LinguagemProgramacao {
    $versao; //atributo
    //método construtor da classe. é executado sempre que um novo
    //objeto é criado
    function __construct() {
        $this->versao = 0;    //acessando o atributo internamente
    }

    //método destrutor da classe. é executado sempre que um objeto é destruído
    function __destruct() {
    }

    function showVersao() {
        echo $this->versao;
    }
}
?>
```

1.1.1.3 Herança

Podemos definir uma classe como sendo uma classe-filha de outra. Isto significa que a nova classe herda todas as características da classe-pai. Este conceito é muito importante e utilizado nos frameworks que veremos no decorrer deste livro. Exemplos:

```
<?php
class LinguagemProgramacaoEstruturada extends LinguagemProgramacao {
    function goto($linha) {
        echo "Movendo para linha $linha";
        //implementação;
    }
}

$cobol = new LinguagemProgramacaoEstruturada;
$cobol->showVersao(); //usando um método herdado da classe-pai
$cobol->goto(100);    //usando um método da classe-filha
?>
```

1.2 Frameworks

1.2.1 O que é um framework?

Parafraseando um professor que tive, framework é um “arcabouço de software”. Sempre gostei dessa definição, mais pela forma como foi dita do que pelo seu significado. Em outras palavras, um framework de desenvolvimento é uma “base” de onde se pode desenvolver algo maior ou mais específico. É uma coleção de códigos-fonte, classes, funções, técnicas e metodologias que facilitam o desenvolvimento de novos softwares.

1.2.2 Mas afinal, porque é vantagem usar um framework?

Esta pergunta foi uma das primeiras que eu e meus colegas fizemos ao professor que citei anteriormente. Quando um desenvolvedor começa a estudar um novo framework, ele se depara muitas vezes, com uma forma diferente de programar ou até de pensar um sistema. É necessário aprender uma sintaxe diferente, convenções para nomes de arquivos, variáveis e tabelas de banco de dados. Além disso, muitas vezes surge a sensação de estar “engessado”, pois é preciso fazer as coisas da forma que o framework trabalha, de modo que

qualquer coisa diferente requer um empenho melhor. Contudo, as vantagens a médio e longo prazo fazem esse pequeno esforço inicial valer.

Como todos os desenvolvedores que usam determinado framework programam usando as mesmas convenções, classes e bibliotecas, a manutenção de um programa é muito mais fácil. Mesmo que determinado script tenha sido escrito por outra equipe há vários meses, não será necessário passar horas tentando entender a serventia da classe `Foo`, a variável `$bar`, ou, ainda, se a tabela onde estão armazenados os registros de clientes se chama `Cliente`, `Clientes` ou `CLIENTE` (em razão de os vários gerenciadores de bancos de dados serem *case sensitive*, `Cliente` e `CLIENTE` são bem diferentes). Isto contribui para que novos desenvolvedores ingressantes em uma equipe possam rapidamente se inteirar da forma como o framework é trabalhado, diminuindo custos e tempo para treinamento.

Outra vantagem da grande parte dos frameworks é que tarefas repetitivas podem ser automatizadas. É o conceito conhecido como DRY – Don't Repeat Yourself, que pode ser traduzido para “não se repita”. Em uma aplicação que tenha de manipular dados vindos de uma tabela na base de dados, as operações de inclusão, exclusão e alteração são praticamente iguais para todas as tabelas envolvidas. Não teria sentido repetir o esforço para desenvolver esse código-fonte várias vezes, e a geração dessas funções poderia ser automatizada por alguma ferramenta contida no framework.

Além das já mencionadas, existem outras vantagens como separação de apresentação e lógica, facilidade de geração de testes automatizados e geração de documentação. Algumas delas ficarão mais claras no decorrer da leitura dos capítulos.

1.3 Tecnologias

Ao olharmos atentamente para os principais frameworks de desenvolvimento para PHP, podemos observar que algumas tecnologias e técnicas são comuns à maioria deles. O tópico 1.3 apresenta algumas delas.

1.3.1 Padrões de projeto

Padrões de projeto, ou *design patterns*, são formas já testadas e documentadas de se resolver certo tipo de problemas. Essas soluções ou metodologias

podem ser aplicadas a vários problemas encontrados no desenvolvimento de aplicações. A grande maioria dos frameworks existentes baseia seu funcionamento em um ou mais desses padrões, dos quais serão descritos os dois padrões mais importantes no contexto deste livro.

1.3.1.1 MVC

MVC é um acrônimo para Model, View, Controller (Modelo, Visão e Controlador). A idéia é separar todo o desenvolvimento de uma aplicação nestas três partes, ou camadas:

- **Model** – gerencia o comportamento dos dados da aplicação.
- **View** – gerencia a saída gráfica e textual da parte da aplicação visível ao usuário.
- **Controller** – interpreta as entradas de mouse e teclado do usuário, comandando a Visão e o Modelo para se alterarem de forma apropriada.

Todas as requisições feitas pelo usuário são enviadas ao Controller. Este manipula os dados usando o Model e invoca a View correta, de acordo com a ação executada ou com os resultados vindos do Model.

A grande vantagem de se utilizar o padrão MVC é a separação de lógica e apresentação, sendo que isso favorece o trabalho em equipe. Um designer poderia trabalhar na apresentação, definindo o HTML, CSS, Flash, enquanto um Data Base Administrator (DBA), administrador de banco de dados, poderia trabalhar com o modelo e outro programador poderia se concentrar nas regras de negócio inseridas no controlador. Dessa forma, qualquer mudança, por exemplo, na apresentação, teria pouco ou nenhum impacto nas demais camadas da aplicação.

1.3.1.2 ActiveRecord

Este padrão de design facilita a manipulação de dados contidos em uma base de dados por aplicações desenvolvidas com o paradigma orientadas a objetos. Uma tabela de uma base de dados é vista na forma de uma classe, enquanto cada linha da tabela é considerada um objeto desta classe. Quando um objeto é criado, alterado ou excluído, esta ação é automaticamente refletida na base de dados. Desta forma, não é necessário que o desenvolvedor conheça uma linguagem de manipulação de dados como SQL, além de manter toda a aplicação desenvolvida no paradigma orientado a objetos.

1.3.2 AJAX

O AJAX (Asynchronous Javascript and XML) não é uma tecnologia nova, mas ganhou holofotes a partir do momento em que o Google começou a usá-la em seus aplicativos, como o Gmail, o qual usa uma mistura de HTML, Javascript e CSS, possibilitando a criação de aplicações que aparentam ser executadas localmente na máquina do usuário e não em um ambiente web. Um exemplo simples seria a apresentação da descrição de um item após a digitação do seu código-fonte em um campo text de um formulário HTML, sem aparentar para o usuário que a página realizou uma requisição ao servidor para buscar a informação.

Toda a página do AJAX está em um objeto chamado XMLHttpRequest, o qual permite que sejam feitas requisições assíncronas ao servidor web, não precisando de atualizações na página ou espera por parte do usuário. Esse objeto foi desenvolvido pela Microsoft para o Internet Explorer 4.0, mas posteriormente outros navegadores como o Firefox e o Opera adicionaram suporte a ele.

Existem diversas bibliotecas para se utilizar as funcionalidades do AJAX em suas aplicações. Entre elas, pode-se citar o Sajax, Cpaint, Script.aculo.us, Dojo, Prototype etc. Alguns frameworks utilizam estas bibliotecas para facilitar o desenvolvimento de aplicações. O CakePHP e o Symfony, por exemplo, utilizam o Script.aculo.us e o Prototype.

1.3.3 Internacionalização

Internacionalização é o processo de possibilitar que seja facilmente alterada a linguagem da interface e de mensagens de uma determinada aplicação. Para facilitar o desenvolvimento deste tipo de aplicações, é necessário que a ferramenta, biblioteca ou framework utilizado possua suporte a essa característica. Alguns frameworks como o Symfony possuem suporte à internacionalização, enquanto o CakePHP ainda não.

1.3.4 PEAR

PEAR consiste em um repositório de componentes reutilizáveis para PHP. Estes componentes abrangem vários tópicos, como autenticação, geração de imagens, conexão com bases de dados etc. Além de ser um repositório de componentes, PEAR também é um sistema de distribuição, que facilita

a instalação, a atualização e a remoção de componentes instalados. Alguns frameworks como o Symfony possuem suporte ao sistema de distribuição PEAR, o que facilita a instalação e atualização do framework.

1.4 Aplicação de demonstração

Para facilitar a visualização das principais características de cada framework e também para facilitar uma comparação entre os mesmos, é interessante modelar uma aplicação que será desenvolvida utilizando se de cada um dos frameworks apresentados.

A aplicação consiste em um sistema onde o usuário pode cadastrar suas coleções de livros. Alguns requisitos desta pequena aplicação são:

- visitantes podem realizar comentários e sugestões para cada item;
- visitantes podem acompanhar a coleção usando um feed RSS;
- deve haver uma área administrativa onde o usuário pode alterar as suas coleções;
- a interface deve ser simples e fácil de utilizar.

Esta aplicação foi escolhida por não ser muito complexa, e mesmo assim possuir características que possibilitam demonstrar algumas necessidades de outras aplicações como controle de acessos, validação de campos, acesso a bancos de dados entre outras.

O ambiente usado no decorrer deste livro foi:

- Ubuntu Linux 6.10
- Apache 2.0.55
- PHP 5.1.6
- MySQL 5.0.24a-9
- DBDesigner 4
- Eclipse com o plug-in PHPTIDE

Esse ambiente foi utilizado, pois é meu ambiente de trabalho. Nada impede que sejam utilizadas outras opções como:

- Microsoft Windows como substituto do Linux.
- Apache para plataforma Windows ou outro servidor web que suporte PHP.
- A maioria dos frameworks precisa do PHP 5 para funcionar, com exceção do Cake, que também funciona com o PHP 4.
- Pode ser utilizada qualquer base de dados suportada pelos frameworks.
- Qualquer ferramenta de modelagem pode ser utilizada.
- Pode ser utilizada qualquer IDE que suporte PHP. O interessante é que tenha suporte mínimo à highlighting. Outra boa ferramenta é o Zend Studio.

A modelagem das tabelas pode ser visualizada na Figura 1.1.

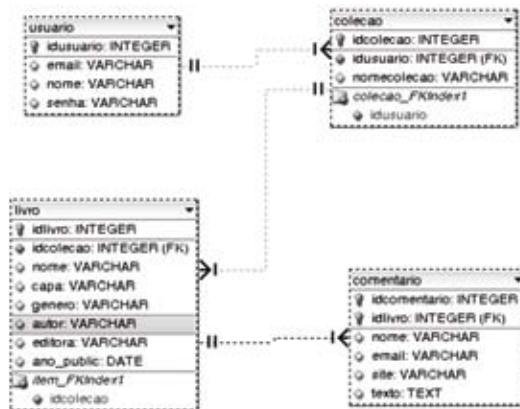


Figura 1.1 – Modelagem da aplicação.

Na tabela usuario, serão armazenados os utilizadores da aplicação. Serão utilizados os campos e-mail e senha para autenticação. Nas demais tabelas, serão armazenados os dados da coleção, dos livros e dos comentários de cada livro.

No decorrer dos próximos capítulos, essa aplicação será desenvolvida em cada um dos frameworks.