

Padrões, PEAR e Frameworks PHP

Professor: FLÁVIO GOMES DA SILVA LISBOA (FGSL)

Introdução a Padrões e Frameworks

Padrões, PEAR e Frameworks PHP

Curriculum

Bacharel em Ciência em Computação, trabalha como Analista de Desenvolvimento de Sistemas na Coordenação Estratégica de Tecnologia do Serviço Federal de Processamento de Dados – SERPRO, empresa do Ministério da Fazenda, citada em pesquisa da revista Info Exame de agosto de 2008 como uma das 200 maiores empresas de tecnologia do Brasil.

Já trabalhou na Diretoria Internacional do Banco do Brasil, com o Sistema de Gerenciamento de Operações utilizado pela Mesa de Operações para o controle dos financiamentos de importação e exportação (ACC, ACE, Carta de Crédito, Finimp e Operações Estruturadas).

Programa em PHP desde a versão 4, e conduziu no SERPRO o trabalho de prospecção e avaliação de frameworks para desenvolvimento em PHP, donde resultou a indicação do Zend Framework. Participa da comunidade Zend Brasil e é autor do livro “Zend Framework – Desenvolvendo em PHP 5 Orientado a Objetos com MVC”.

Plano de Aulas

Dia	Conteúdo
1	Motivação para o uso de frameworks. Instalação e uso do Eclipse com plugin PDT. Padrão de Projeto MVC. Apresentação do Zend Framework. Projeto Mínimo. Padrões de Projeto Singleton, Controller Front e Controller Page. Controle de Erros.
2	Componente de acesso ao banco. Mapeamento Objeto-Relacional. Abstração da camada do banco X Uso de funções específicas. Encapsulamento da sessão como objeto. Padrões de Projeto Factory, Gateway, Iterator e Active Record.
3	Implementação de código seguro com componentes do framework. Filtros e Validadores. Listas de Controle de Acesso. Autenticação. Segurança no acesso ao banco de dados.
4	Separação da aplicação em módulos. Uso de templates e subtemplates de página. Criação de formulários dinâmicos.
5	Encapsulamento de componentes de terceiros (PEAR, Smarty). Criação de novos componentes.

Pré-requisitos

- ✓Conhecimentos básicos em HTML;
- ✓Conhecimentos básicos em PHP;
- ✓Conhecimentos básicos de Orientação a Objetos;
- ✓Conhecimentos básicos de bancos de dados relacionais.

Padrões

O que é
um
PADRÃO?



Padrões

Padrão¹

sm (lat patronu) **1** Modelo oficial de pesos e medidas. **2** Modelo. **3** Desenho de estampa. **4** Título autêntico. *P. de cor, Telev*: cada um dos três padrões internacionais usados para descrever como as cores da TV e imagens de vídeo são exibidas e transmitidas: NTSC, PAL e SECAM. *P. de fato, Inform*: um projeto, método ou sistema tão amplamente utilizado que se tornou padrão, apesar de não ter sido reconhecido oficialmente por qualquer comitê. *P. -ouro*: moeda-ouro. *Pl: padrões-ouros e padrões-ouro*.

Padrão²

sm (pedra+ão) **1** Monumento de pedra que os portugueses erigiam em terras de sua descoberta. **2** Monumento monolítico; marco.

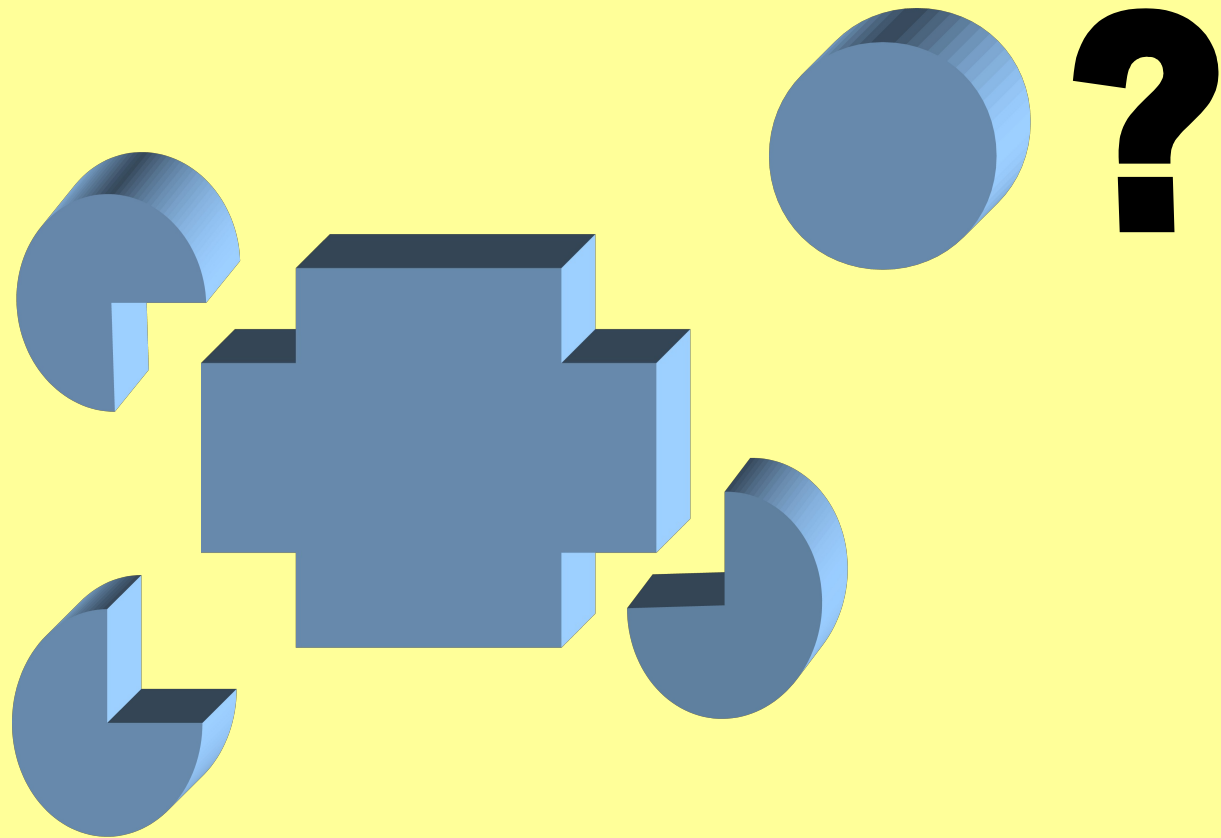
Padrões

P. de fato, Inform: um projeto, método ou sistema tão amplamente utilizado que se tornou **padrão**, apesar de não ter sido reconhecido oficialmente por qualquer comitê.

DEFINIÇÃO RECURSIVA!

Mas traz uma pista para o conceito *comercial* e *industrial* de padrão: o reconhecimento oficial por um **comitê**.

Para quê servem padrões?



PARA QUE AS COISAS SE ENCAIXEM!

Para quê servem padrões?



Padrões, PEAR e Frameworks PHP

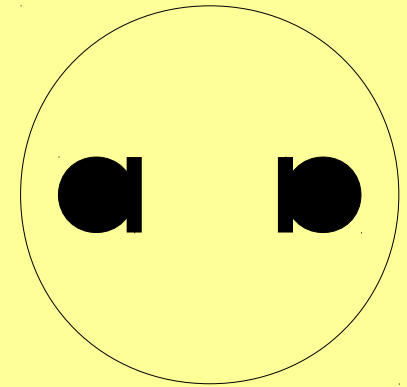
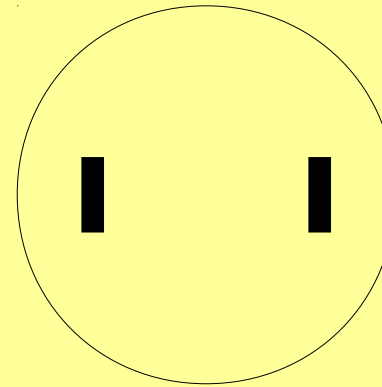


Não seria maravilhoso se todas as tomadas fossem padronizadas e eu não precisasse comprar adaptadores e extensões só pra ligar um aparelho?

Deve ser um sonho...

O'Reilly Media All rights reserved.

Para quê servem padrões?



Ô, Zé, tem um
benjamim aí?

Padrões, PEAR e Frameworks PHP



Não seria maravilhoso se todos os celulares tivessem o mesmo tipo de carregador e eu pudesse emprestar de outra pessoa quando eu esquecer o meu?

Deve ser um sonho...

O'Reilly Media All rights reserved.

Padrões, PEAR e Frameworks PHP



Não seria
maravilhoso se o
mecânico não
pudesse dizer que as
peças do meu carro
são caras e difíceis
de encontrar?

Deve ser um sonho...

O'Reilly Media All rights reserved.

Para quê servem padrões?

أحبك العادل
الطريقة أنت أنت
عندما لا يجري فقط
الطريق انت

☆☆☆ ❄❄❄
▲❄□ ❄●❄ □
❄○□ ❄❄❄ □❄
■▼❄ ✂



PARA QUE A COMUNICAÇÃO SEJA POSSÍVEL!

Para quê servem padrões?



PARA SEREM RECONHECIDOS!

Marvel Comics All rights reserved. Spiderman created by Stan Lee and Steve Ditko.

O QUE SÃO *PADRÕES DE PROJETO?* (*DESIGN PATTERNS*)

No inglês, na falta de um padrão, existem três:

default
pattern
standard

E na contramão da riqueza de verbetes do Português, os três são geralmente traduzidos como... **padrão!**

O QUE SÃO *PADRÕES DE PROJETO?* (*DESIGN PATTERNS*)

default

Um caminho de ação tomado por um computador quando não é dada a ele nenhuma outra instrução.

O QUE SÃO *PADRÕES DE PROJETO?* (*DESIGN PATTERNS*)

pattern

Modo pelo qual algo ocorre, é desenvolvido ou é feito (rotina); arranjo de linhas, formas, cores, etc; **um projeto, um conjunto de instruções ou a forma como você corta caminho pelo que você usa, de modo a fazer alguma coisa.**

O QUE SÃO *PADRÕES DE PROJETO?* (*DESIGN PATTERNS*)

standard

Nível de qualidade; um nível de qualidade com o qual você compara alguma coisa com outra; nível de comportamento aceitável moralmente; normal ou médio; não especial ou não incomum.

O QUE SÃO *PADRÕES DE PROJETO?* (*DESIGN PATTERNS*)

"Cada padrão descreve um problema no nosso ambiente e o núcleo da sua solução, de tal forma que você possa usar esta solução mais de um milhão de vezes, sem nunca fazê-lo da mesma maneira."

Cristopher Alexander

Interessante... mas ele está falando de edificações e cidades!

O QUE SÃO *PADRÕES DE PROJETO?* (*DESIGN PATTERNS*)

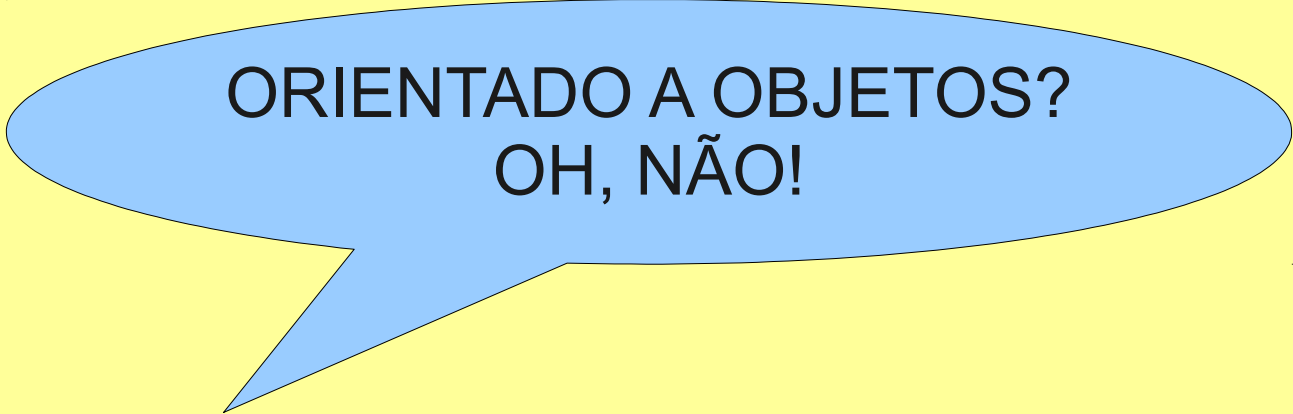
Todo padrão de projeto consiste essencialmente em:

- *Um nome;*
- *Um problema;*
- *Uma solução;*
- *Conseqüências (vantagens e desvantagens).*

O QUE SÃO *PADRÕES DE PROJETO?* (*DESIGN PATTERNS*)

Mas o que são afinal?

Podemos dizer que padrões de projeto são “soluções reutilizáveis de software orientado a objetos” (Gamma, 2000)



ORIENTADO A OBJETOS?
OH, NÃO!

O QUE SÃO *PADRÕES DE PROJETO?* (*DESIGN PATTERNS*)

Podemos ainda dizer que padrões de projeto são “soluções simples para problemas específicos no projeto de software orientado a objetos” (Gamma, 2000)



**TEM QUE SER
ORIENTADO A OBJETOS?**

Revisão: Orientação a Objetos

Segundo Martin(1995), a tecnologia orientada a objetos traz vários benefícios:

- **Reusabilidade;**
- **Estabilidade** (obtida com o reuso);
- ***"A ignorância é uma benção"*** (ocultação do baixo nível de implementação);

Revisão: Orientação a Objetos

- Classes **complexas** feitas a partir de classes **simples** (linha de montagem);
- **Confiabilidade** (menor número de falhas);
- **Novos mercados** (classes especialistas que usam classes genéricas);
- **Projeto mais rápido** (obtido com o reuso);

Revisão: Orientação a Objetos

- **Projeto de qualidade mais elevada** (obtido com o reuso);
- **Integridade** (dados protegidos);
- **Programação mais fácil** (há controvérsias);
- **Manutenção mais fácil** (mude em um só lugar);

Revisão: Orientação a Objetos

- **Estímulo da criatividade** (o que pode ser perigoso sem disciplina);
- **Ciclo de vida dinâmico** (tudo muda toda hora);
- **Refinamento durante a construção** (refatoração);
- **Modelagem mais realística** (será?);

Revisão: Orientação a Objetos

- **Melhor comunicação entre programador e analista de negócio** (será?);
- **Manutenção mais fácil** (mude em um só lugar);
- **Independência de projeto** (independente de plataforma, hardware e ambiente de software);
- **Interoperabilidade** (as classes conversam entre si);

Revisão: Orientação a Objetos

- **Computação cliente-servidor;**
- **Computação maciçamente distribuída;**
- **etc.**

Por que padrões de projeto são orientados a objeto?

Palavra-chave: reuso

Reuso não carece necessariamente de orientação a objetos, mas ela oferece um conjunto de ferramentas que facilita essa prática.

Quando eu uso os padrões? Quais eu uso? Como eu uso?

O primeiro passo é identificar o problema.

Depois deve-se pesquisar em um catálogo de padrões se existe (provavelmente existe) um que possa ser aplicado para o problema em questão.

Eventualmente, será necessário usar vários padrões juntos para resolver um problema.

Quando eu uso os padrões? Quais eu uso? Como eu uso?

O matemático Stephen Wolfram, criador do software *Mathematica*, afirma que problemas complexos podem ser resolvidos pela combinação de soluções simples.

Um problema simples exige uma solução simples.

Um problema complexo exige a combinação de várias soluções simples.

Nem sei usar um padrão, quanto mais vários!

Seus problemas acabaram!

Chegou o novo e revolucionário modo de usar padrões de projeto para solucionar problemas de software:

FRAMEWORK

*Disponível em várias linguagens de programação,
com mais de uma opção para cada!*

Como aprenderemos padrões

A experiência mostrou através das décadas, que o melhor modo de aprender a programar é programando.

Com base nesse caso de sucesso, iremos aprender padrões de projeto vendo sua implementação dentro de um *framework*.

Mas o que é um framework?

É um conjunto de classes cooperantes que constroem um projeto reutilizável para uma específica classe de software

Um framework típico contém vários padrões de projeto, mas a recíproca nunca é verdadeira.

Frameworks podem ser materializados em código, mas apenas exemplos de padrões de projeto podem ser materializados em código.

Gamma (1995)

E pra que serve um framework?

- ▶ Para diminuir custos e tempo de treinamento para novos ingressantes nas equipes de desenvolvimento, graças aos componentes reutilizáveis;
- ▶ Para automatizar de tarefas repetitivas, diminuindo o esforço do desenvolvimento;
- ▶ Para separar as camadas de apresentação e lógica, permitindo o desenvolvimento simultâneo por equipes distintas;
- ▶ Para facilitar a documentação dos sistemas desenvolvidos, o que contribui para a facilidade de manutenção;
- ▶ Para facilitar a geração de testes automatizados, o que contribui para a garantia de qualidade do produto.

Qual framework utilizaremos?

Gamma (1995) enuncia algumas características importantes para um *framework*:

Inversão de Controle: O *framework* chama seu código, ao invés de seu código chamar sub-rotinas. Isso obriga o programador a escrever operações com nomes e convenções de chamada já especificadas, porém isso reduz as decisões de projeto – já existe um **padrão** de nomes!

Qual framework utilizaremos?

Flexível e extensível: A arquitetura deve funcionar para todas as aplicações do domínio.

Acoplamento fraco: Pequenas mudanças no *framework* não podem gerar grandes repercussões na aplicação.

Qual framework utilizaremos?

Além disso, temos alguns problemas no mundo **PHP**:

- Sistemas que usam classes mas não são orientados a objetos (frutos do PHP 3 e 4).
- Classes com nomes e convenções de chamada completamente diferentes.
- Um exército de programadores sem formação adequada.

Padrões, PEAR e Frameworks PHP



Não seria maravilhoso se todos os programadores PHP do mundo trabalhassem de uma forma padronizada, de modo que pudessem compartilhar seus componentes e reutilizar milhares de linhas de código fonte?

Deve ser um sonho...

O'Reilly Media All rights reserved.

Não é um sonho!

É o **PHP Collaboration Project**:

Iniciativa *open source* através da qual a comunidade PHP e seus parceiros pretendem criar um padrão de ambiente de distribuição e desenvolvimento de aplicações Web em PHP.

Consiste em duas frentes:

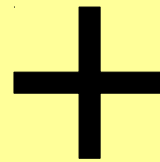
► **Zend PHP Framework**: com o objetivo de padronizar a forma como as aplicações PHP são construídas.

► **Engajamento com a Eclipse Foundation**: a Zend juntou-se à fundação como Desenvolvedor Estratégico. O objetivo é liderar um projeto focado em PHP dentro do Eclipse.

Zend Framework e Eclipse: Como Goiabada com Queijo



Eclipse Foundation All rights reserved.



Zend Technologies All rights reserved.

Pausa para instalação do ambiente...

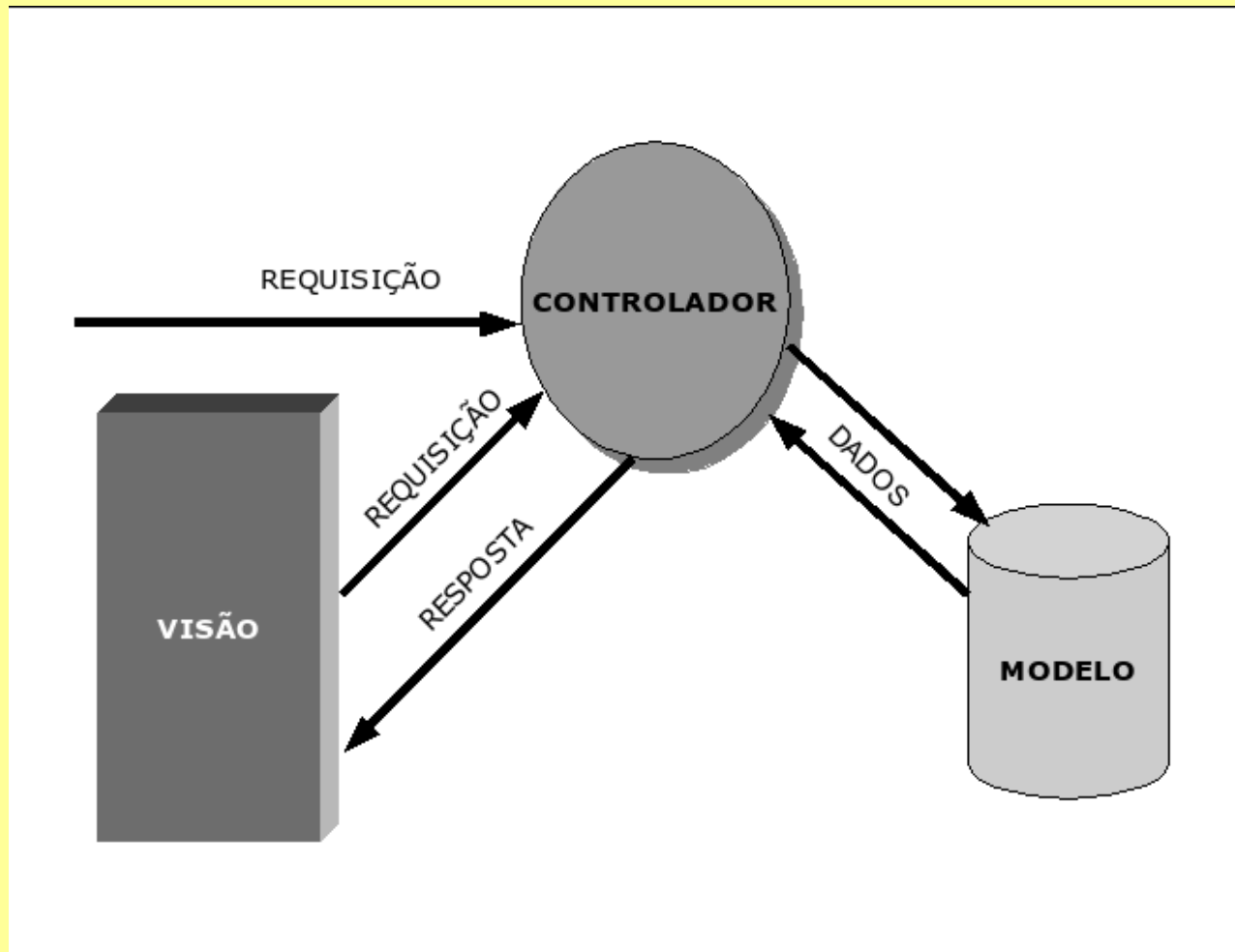


Projeto Mínimo do Zend Framework

Vamos fazer um “Alô Mundo” com Zend Framework e aprender com isso quatro padrões de projeto:

- **MVC;**
- **Singleton;**
- **Front Controller;**
- **Page Controller.**

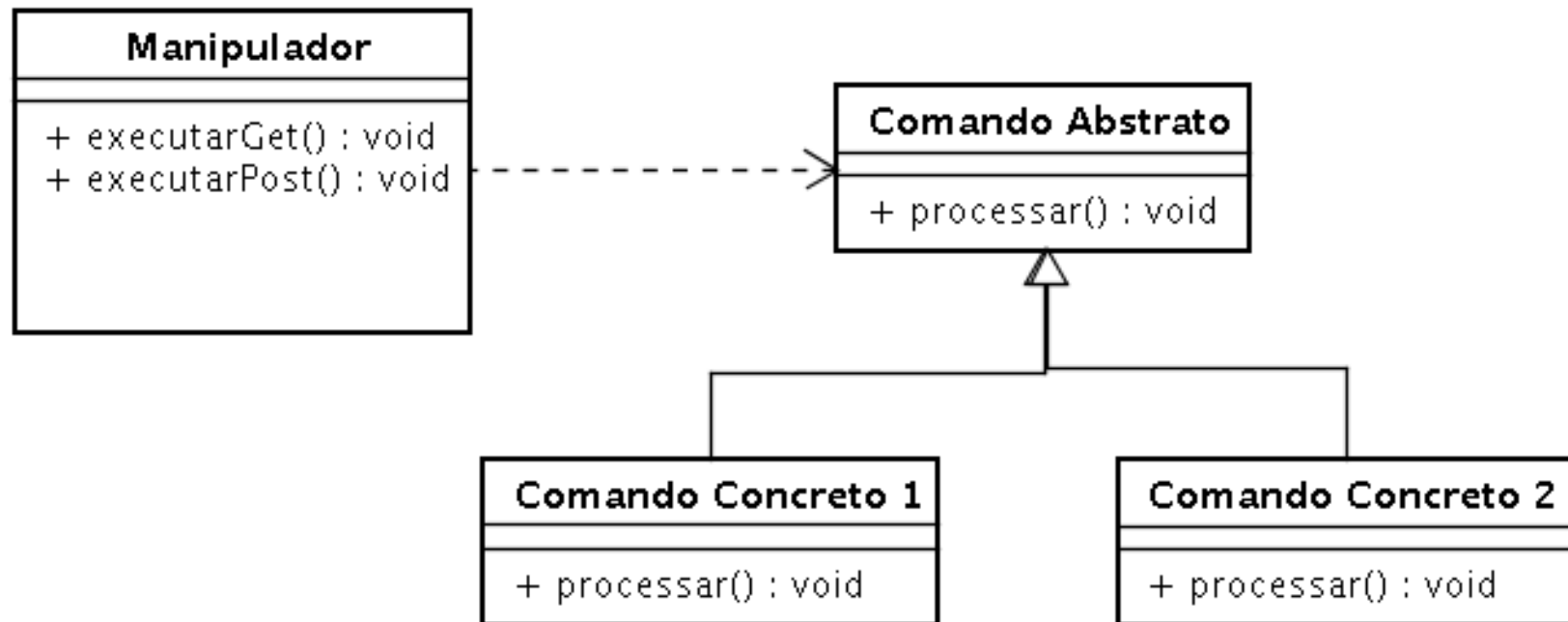
MVC – Model View Controller



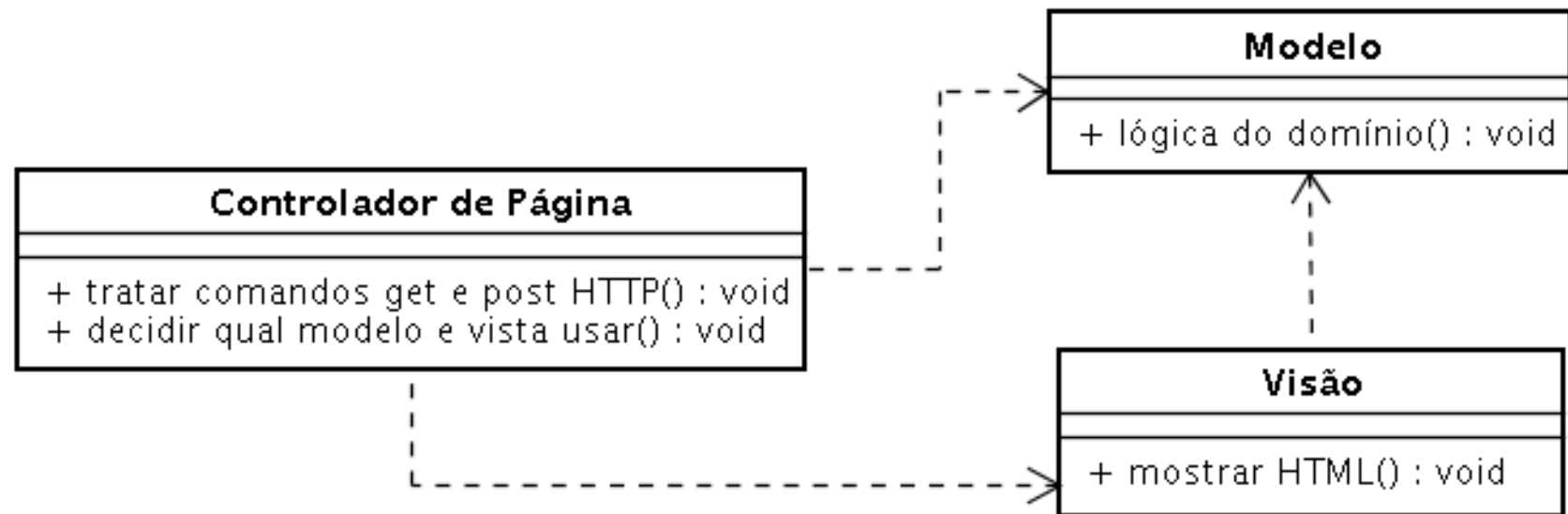
Singleton

Singleton	
<u>- instance : Singleton</u>	
- __construct() : void	
+ getInstance() : Singleton	

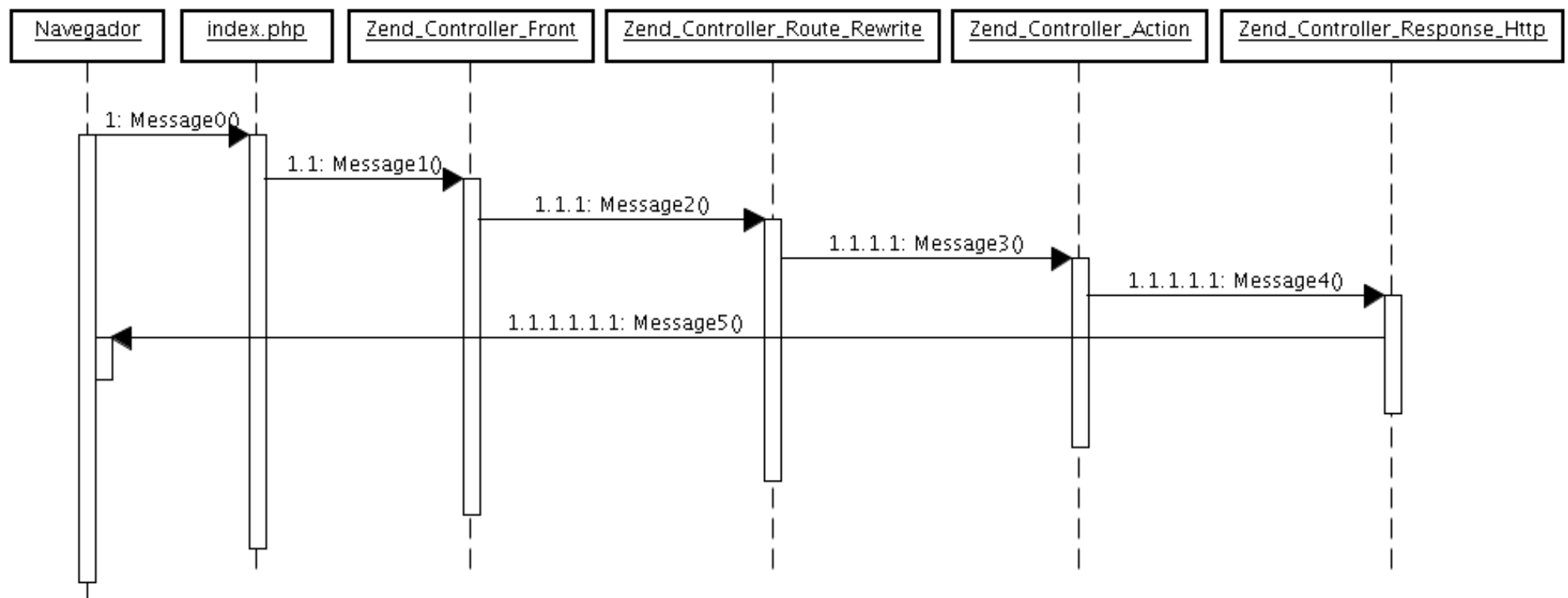
Front Controller



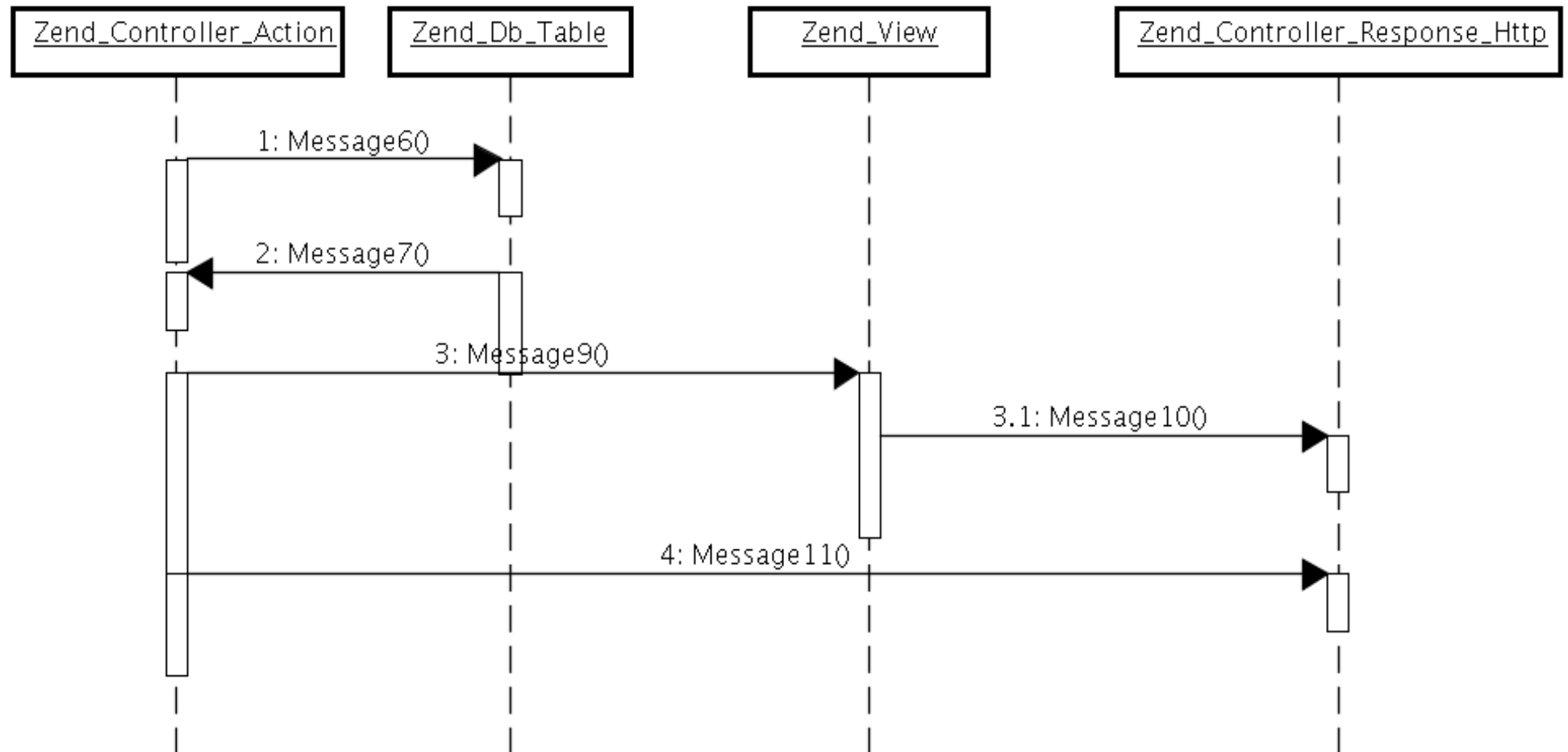
Page Controller



Arquitetura Geral Zend Framework



Arquitetura Geral Zend Framework



Referências Bibliográficas

Fowler, M. *Padrões de Projeto de Aplicações Corporativas*. Porto Alegre. Bookman, 2006.

Freeman, E. e Freeman, E. *Use a Cabeça! Padrões de Projetos*. Rio de Janeiro. AltaBooks, 2005.

Gamma, E. et alli. *Padrões de Projeto: Soluções Reutilizáveis de Software Orientado a Objetos*. Porto Alegre. Bookman, 2000.

Martin, J. e Odell, J. J. *Análise e Projeto Orientados a Objeto*. São Paulo. Makron Books, 1995.

[http://framework.zend.com](http://framework zend.com)