

# Oraculum

## Guia utilização do Oraculum Framework

**Autor:** Patrick Kaminski

**Última atualização:** 31 de Agosto de 2009.

**Versão do Documento:** 1

**Versão do Framework:** 0.1 Alpha

### Resumo

Atualmente existem ótimos Frameworks PHP no mercado, porém, até hoje não surgiu nenhum Framework nacional que tivesse um grande alcance mundial. Além disso, existem muitas necessidades que poderiam ser mais exploradas pelos frameworks, convertendo-as em um menor esforço dos programadores ao desenvolver sistemas, e conseqüentemente em softwares de melhor qualidade e desempenho, além da criação de um padrão de desenvolvimento. Com isso, a área de desenvolvimento Web, tornaria-se mais preparada e evoluída, fazendo com que a própria linguagem PHP ocupasse mais espaço no mercado. Este também é um dos focos deste projeto.

**Palavras-chave:** *framework, php, mvc, orm, scaffolding.*

### Por que utilizar o Oraculum Framework?

Um dos diferenciais do Oraculum, é que há uma preocupação em procurar o que é melhor para a maioria.

Atualmente o que mais foi identificado - através de pesquisas - que seria importante melhorar em comparação a outros Frameworks está listado abaixo:

- Documentação
- Ajax
- Tutoriais e Exemplos
- Scaffolding

- Cache
- Internacionalização
- Segurança

Por isso esses itens estão sendo analisados com maior cuidado.

O nome Oraculum vem do Latim, e significa Oráculo. O Latim foi escolhido por ser uma das línguas básicas que deu origem a várias outras linguagens como o próprio Português, e porque é uma língua interessante que merece o seu respeito, além de passar uma imagem mais séria. Já a palavra Oraculum foi escolhida para representar um sistema que assim como os Oráculos antigos, como o I Ching, Delphos, etc, deve fornecer algo, baseado em situações externas conhecidas ou desconhecidas por que o consulta/utiliza.

## **Foco do projeto**

O projeto Oraculum Framework possui como objetivos principais:

- Dar liberdade e incentivar a sua comunidade para que façam com que a força do PHP seja mostrada.
- Fazer o PHP evoluir mais.
- Atender as necessidades (atuais e futuras) de seus usuários.
- Apresentar diferenciais e soluções inovadoras às necessidades existentes.
- Fornecer uma ferramenta de qualidade para o desenvolvimento web.
- Facilidade de uso.
- Apresentar um novo paradigma de desenvolvimento, uma nova filosofia.

## **Vantagens**

Um dos focos do Oraculum Framework é apresentar diferenciais e soluções inovadoras às necessidades existentes, por este motivo, o Oraculum Framework apresenta as seguintes características:

- Desenvolvido para ser utilizado com PHP5.
- Focado na compatibilidade com as próximas versões do PHP (6, etc).
- Seu código segue os padrões da Zend.
- Implementa os padrões de desenvolvimento Front Controller, MVC, KISS, DRY, etc
- Implementa um conceito de desenvolvimento (Workspace's) para facilitar o trabalho de equipes heterogêneas.
- Possui Scaffolding (Gerador de Códigos).
- Uso de URL Amigáveis.
- Sua camada de modelo é integrada com o projeto Doctrine ORM.

- Possui componentes (Helpers) próprios, além de componentes baseados em projetos já existentes.
- Caracteriza-se como um framework horizontal, pois pode ser utilizado para diversas aplicações.
- Possui o objetivo de evoluir de acordo com as necessidades dos usuários, porém mantendo a simplicidade e organização.
- Possui sistema de internacionalização.
- Possui sistema de logs.
- Etc.

## Estrutura de um Framework

O Oraculum Framework possui uma estrutura dividida em duas partes denominadas frozenspots e hotspots.

Frozenspots tratam-se de partes fixas que já são implementadas pelo Framework e que não necessitam de implementações do desenvolvedor.

Hotspots tratam-se de partes flexíveis do framework e que necessitam de complementação.

Esta parte deve ser flexível para que possam ser desenvolvidos vários tipos de sites e sistemas.

## Arquitetura do Oraculum Framework

O Oraculum Framework adota uma arquitetura que utiliza os padrões mais utilizados no desenvolvimento web orientado a objetos, como Front Controller, MVC, KISS, DRY, etc.

Para conter estes padrões, é utilizada uma estrutura de diretórios de acordo com a tabela abaixo (Tabela 1):

<b>Diretório</b>	<b>Descrição / O que é armazenado</b>
Oraculum Framework	Base do Framework
@project/	Diretórios para armazenar arquivos de projeto da aplicação, como diagramas UML, DFD, MER, arquivos SQL, etc
apps/	Diretório que contém as aplicações
[PROJETO]	Aplicação
config/	Configurações da aplicação
workspaces/	Configurações específicas de cada ambiente de trabalho
controllers/	Camada de Controle da aplicação
modulos/	Classes da camada de Controle da aplicação
models/	Camada de Modelo da Aplicação

<b>Diretório</b>	<b>Descrição / O que é armazenado</b>
views/	Camada de Visualização da Aplicação
modulos/	Arquivos com extensão shtml, html ou php, contendo tags HTML e PHP para gerar a interface Web para o usuário
controllers/	Contem o controlador monolítico do Framework (Front Controller). Esta diretório é um diretório fixo do Framework, e não necessita de interferência do desenvolvedor [ <i>Frozenspot</i> ]
layout/	Diretório contendo arquivos de layout das aplicações
[PROJETO]	Diretório do layout da aplicação
css/	Arquivos CSS da aplicação
img/	Arquivos de imagem da aplicação
js/	Arquivos Javascript da aplicação
swf/	Arquivos SWF da aplicação
library/	Diretório de bibliotecas do Framework. Também é um diretório fixo do Framework [ <i>Frozenspot</i> ]
components/	Componentes baseados em projetos já existentes em software livre e que são integrados ao Oraculum Framework para uma maior evolução dos projetos
core/	Diretório que contém as classes que fazem parte do núcleo do Oraculum Framework.
logs/	Diretório de Logs
models/	Diretório da camada de modelo do Framework, que contém como base o Doctrine ORM, para realizar o gerenciamento das bases de dados das aplicações
doctrine/	Diretório do Doctrine ORM (versão 1.0 atualmente)
[PROJETO].php	Arquivo de configuração contendo os parâmetros de conexão com bases de dados da aplicação
start/	Diretório contendo o sistema gerador de códigos (Scaffolding) [ <i>Frozenspot</i> ]
	Diretório temporário
index.php	Arquivo principal que chama as aplicações

Tabela 1 – Estrutura de Diretórios do Oraculum Framework

## Instalação e uso

Para obter o Oraculum Framework basta realizar o download da última versão estável disponível no site do projeto. Atualmente o mesmo está disponível no endereço <http://code.google.com/p/oraculum-php/>.

Os requisitos para instalar e utilizar o Oraculum Framework são os seguintes:

- Servidor Web preferencialmente Apache 2 ou superior.
- PHP5 ou superior.
- Recomenda-se habilitar a opção de reescrita de URL no Apache, porém, os testes com a mesma desabilitada foram bem sucedidos.
- Sistema Operacional Windows ou Linux. Ainda não foram realizados testes em outras plataformas como Mac OS.

Após obter o arquivo, descompacte-o com um programa descompactador na pasta de publicações (*document\_root*) do servidor Web. Acesse o endereço de acesso do servidor web no seu navegador, seguido do nome da pasta, ou subpastas onde o framework foi incluído dentro da pasta de publicações.

Se aparecer uma tela informando que o Oraculum Framework foi inicializado com sucesso ocorreu tudo certo, caso contrário, procure verificar se você descompactou o framework na pasta correta, e que está acessando o endereço correto.

Se ocorreu tudo corretamente, você pode acessar o sistema gerador de códigos, e gerar a estrutura básica do seu sistema, baseando-se em algum banco de dados que você deseje utilizar.

## **Desenvolvendo com o Oraculum Framework**

### **Criando um projeto com o Oraculum Framework**

Para criar um projeto utilizando o Oraculum Framework, deve-se seguir um padrão de organização que acaba por facilitar em caso de manutenções futuras. O Framework possui a capacidade de comportar várias aplicações em sua estrutura, todas separadas e capazes de interagir, como por exemplo, uma aplicação servindo para gerar um Website, e outra aplicação para controlar este Website, como um sistema gerenciador de conteúdo.

Para iniciar o desenvolvimento de uma aplicação, existem duas formas, criá-la manualmente ou automaticamente através de um sistema gerador de código do Framework, que também pode ser chamado de Scaffolding.

### **Criando uma aplicação manualmente**

Para se criar uma aplicação manualmente, deve-se criar um diretório dentro do diretório **apps**, contendo a seguinte estrutura de pastas que pode ser visualizada através da Imagem 1.

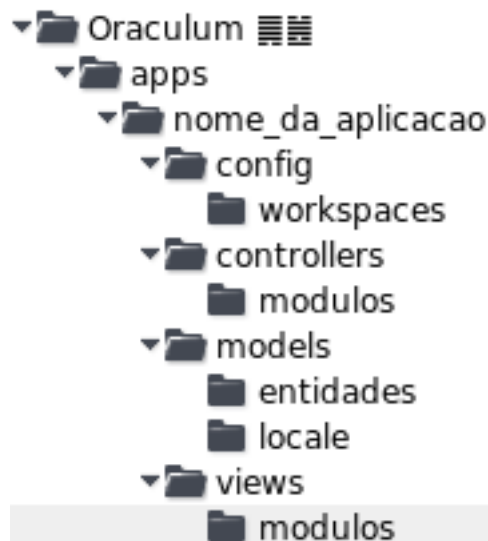


Imagem 1 - Estrutura de diretórios de uma aplicação do Oraculum Framework

Essa estrutura serve para manter o projeto organizado de acordo com o padrão MVC, que será melhor explicado item 4.6.1.2. Para não precisar criar todos os arquivos iniciais, você pode copiar ou renomear a pasta default dentro da pasta **apps**, que é a aplicação padrão que vem junto com o Oraculum Framework. Após isso, você deve alterar as configurações da aplicação através do arquivo `init.php` localizado no diretório `config` da sua aplicação.

Outra coisa importante que você deve fazer é configurar os Workspace's (ambientes de trabalho). Dentro do diretório `config` você deve encontrar também uma pasta chamada `workspaces`, onde existirá um arquivo chamado `list.php`, nele há um vetor, contendo um endereço de e-mail e uma string "arquivo", isto define que se na configuração do servidor Web estiver configurado como e-mail de administração o e-mail listado, deverá ser carregado o arquivo de configuração de nome "arquivo.php" no mesmo diretório. Altere conforme as suas configurações.

Esta estrutura de `workspaces` é muito útil para equipes e servidores heterogêneos, onde podem existir vários sistemas operacionais e servidores com configurações diferentes, ou que necessitam de configurações específicas, como por exemplo, evitar a exibição de erros para o usuário.

Após isso, você deve acessar o arquivo `index.php` localizado na raiz do Framework, e alterar a linha que faz a inclusão do arquivo da aplicação, para que a sua aplicação seja carregada corretamente.

Outro detalhe importante, e que infelizmente não pode ser alterado dinamicamente pelo framework, são as configurações de reescrita de Url. Estas configurações ficam armazenadas em um arquivo texto padronizado, para ser interpretado pelo servidor Web, chamado `.htaccess`. Este arquivo é responsável por habilitar a reescrita de Url's, para gerar Url's amigáveis que possam ser compreendidas pelo usuário, além de manter suas aplicações

seguras.

Fazendo essas alterações a sua aplicação já deve estar pronta para ser visualizada no navegador através da Url correspondente de acordo com o seu servidor Web.

### **Criando uma aplicação automaticamente**

O Oraculum Framework possui um sistema muito comum na maioria dos frameworks, chamado scaffolding, que possui a utilidade de gerar os códigos de sistemas através de uma estrutura pré-definida.

O sistema de scaffolding do Oraculum Framework utiliza o mapeamento da estrutura de bancos de dados para gerar toda a estrutura básica dos sistemas, que se trata das seguintes operações:

- Criação
- Listagem
- Atualização
- e Exclusão

Essas operações básicas são também conhecidas pelo acrônimo CRUD – Create, Retrieve, Update e Delete. Com isso, boa parte dos sistemas e sites podem ser geradas automaticamente através das Bases de Dados, por este motivo, se o banco de dados estiver bem modelado, o sistema converterá esta modelagem para formulários e ações fazendo com que o sistema possua uma boa qualidade, através de validações de campos entre outras coisas. Desta forma as operações mais repetitivas são eliminadas, fazendo com que os desenvolvedores se preocupem mais com outras fases do desenvolvimento, como a modelagem, os testes, a criação de designs mais elaborados para as telas dos sistemas, etc. Para realizar a criação automática de aplicações você deve acessar o endereço /start e acessar e clicar em CRUD. Ao fazer isto, será possível selecionar as entidades (tabelas) do banco de dados e quais ações você deseja que sejam geradas.

Ao clicar em executar, todos os arquivos serão gerados automaticamente.

### **ORM: Acessando tabelas como objetos**

"ORM (Object relational mapping) é uma técnica utilizada em linguagens de programação para realizar a manipulação com bases de dados para traduzir os tipos de dados incompatíveis em bancos de dados relacionais. Isto permite essencialmente ter um "objeto de dados virtual", que pode ser utilizado a partir da linguagem de programação. Muitos pacotes livres e comerciais existentes permitem a utilização de ORM's, porém, muitos programadores optam por desenvolver os seus próprios ORM's."

## Doctrine ORM

Ao iniciar o desenvolvimento do projeto Oraculum Framework, foi optado por utilizar algum ORM como base para a camada de modelo, pois atualmente, a maioria dos ORM's são utilizados em conjunto com a maioria dos Frameworks, porém, não o Oraculum Framework terá um ORM como base da camada de modelo, para fornecer uma maior integração com as demais camadas do Framework. Para integrar o Oraculum Framework, foi escolhido como ORM, o ORM Doctrine, por ter suporte à vários tipos de sistemas gerenciadores de bancos de dados como FreeTDS, Microsoft SQL Server, Sybase, Firebird, Interbase 6, Informix, MySQL, Oracle, Odbc, PostgreSQL, Sqlite. E por ter várias ferramentas úteis, facilitando a criação de componentes integrados.

## Controlando as requisições do cliente

Através do padrão chamado Front Controller, é possível utilizar um controlador para controlar todas as requisições do cliente baseando-se nas regras de negócio do sistema/site, e assim, instanciando os controladores correspondentes.

Desta forma, todas as requisições passam pelo mesmo controlador, tornando a estrutura mais controlada e conseqüentemente mais segura e equilibrada.

## Camadas de uma Aplicação

As aplicações do Oraculum Framework seguem o padrão MVC, fazendo com que haja uma maior organização para a aplicação, além de um maior controle.

## Padrão MVC

"Model View Controller (MVC) é o design pattern mais conhecido de todos. Seus conceitos remontam à plataforma Smaltalk na década de 1970. Basicamente uma aplicação que segue o pattern Model View Controller é dividida em três camadas. As letras que compõem o nome deste pattern representam cada um desses aspectos." (Lasater, 2006)

Através da Imagem 2 é possível analisar o funcionamento deste padrão.

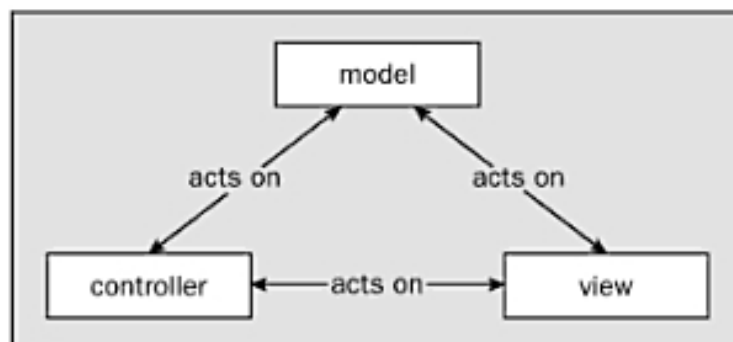


Imagem 2 – Padrão MVC (Horton, 2004)

## **Model**

Model significa Modelo, e é esta a camada responsável por controlar as requisições e manipulações ao sistema gerenciador de banco de dados. Através desta camada, é possível a utilização de classes para realizar a interação com as entidades, fazendo assim, com que as entidades sejam manipuladas como objetos.

## **View**

View significa Visualização, e é esta a camada responsável por controlar o que é exibido para o usuário. Na área do desenvolvimento Web, esta camada é composta basicamente de tags HTML, e algumas chamadas de métodos, sem conter regras de negócios.

## **Controller**

Controller significa Controlador, e é esta a camada que é a principal responsável pela interação entre as camadas View e Model. Além disso, é ela a responsável pelo tratamento das requisições do cliente.

## **Considerações finais sobre o Framework**

Analisando o resultado obtido a partir deste projeto, pode-se concluir que o Oraculum Framework pode muito bem ser utilizado por outras pessoas, e apresenta um grande potencial na área de desenvolvimento web, oferecendo aos desenvolvedores uma forma de realizar um desenvolvimento ágil e talvez o mais importante, um desenvolvimento controlado.

Desta forma, este projeto deverá ter continuidade como software livre, disponível através da licença LGPL, permitindo que o mesmo seja utilizado também com softwares comerciais, dando uma maior liberdade para desenvolvedores web e empresas. Através desta ação, e de algumas estratégias de marketing é possível reunir uma comunidade de usuários e desenvolvedores capazes de fazer com que o projeto evolua por conta própria e tenha um alcance global.

Sem dúvida, os objetivos gerais e específicos do projeto foram alcançados.

## **Referências**

### **Design Patterns: Model-View-Controller**

<<http://java.sun.com/blueprints/patterns/MVC.html>> Acessado em 18/02/2009.

Lasater Christopher G.; **Design Patterns**. Wordware Publishing, Inc.. 2006.