

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA



@fgsl

Padrões de Projeto e Boas Práticas em PHP



Flávio Gomes da Silva Lisboa
@fgsl
www.fgsl.eti.br

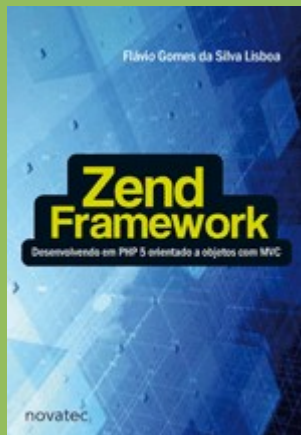


Quem sou eu

2008

2009

2007



25 A 28 DE NOVEMBRO



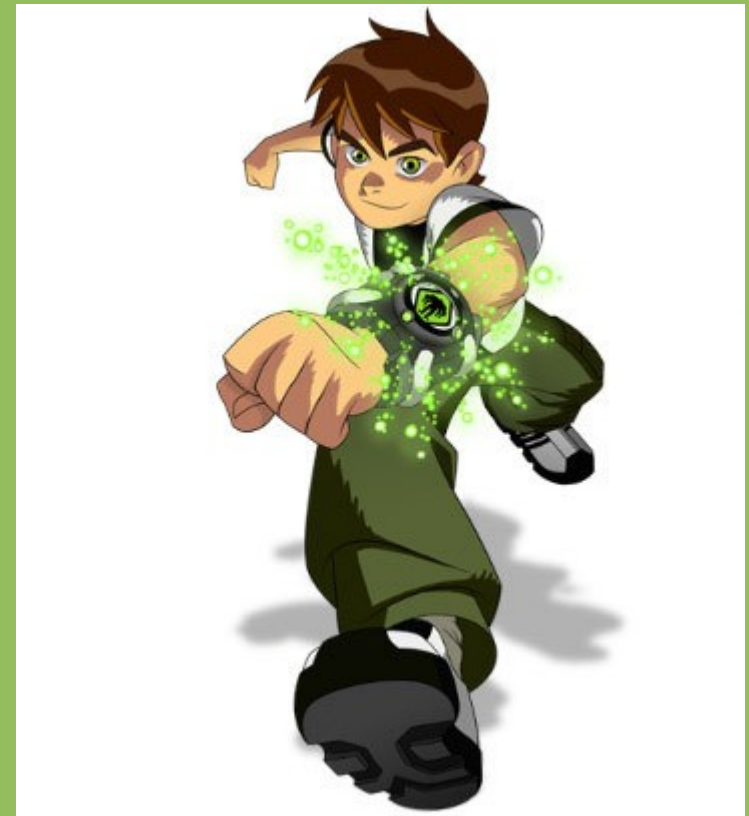
PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Qual nosso objetivo?



toddler-gift.blogspot.com



ondecomprar.net

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Qual nosso objetivo?



disneypedia.com.br

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Padrão de Projeto



ci-columbia.com.br

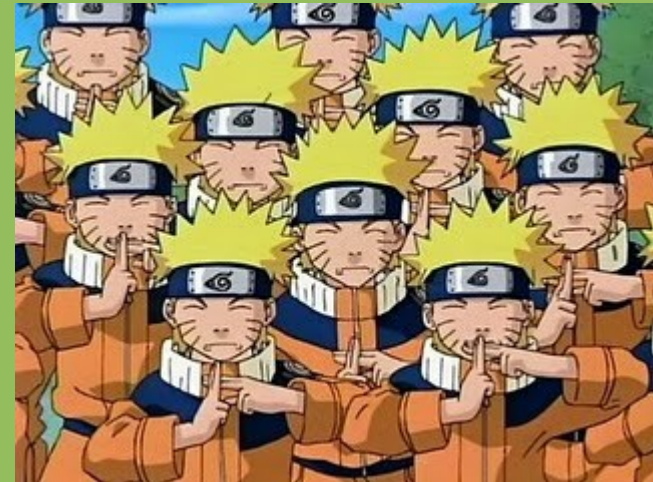
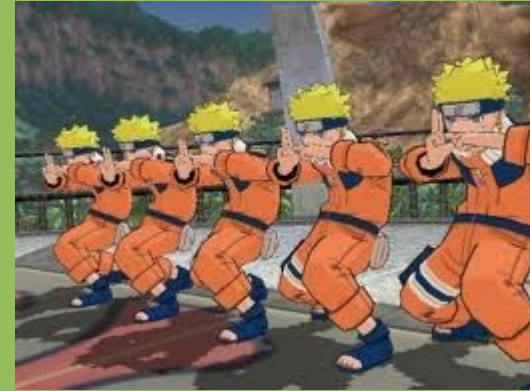
25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Cada padrão
descreve um
problema que
ocorre
repetidamente
em nosso
ambiente...



Naruto, by Masashi Kishimoto

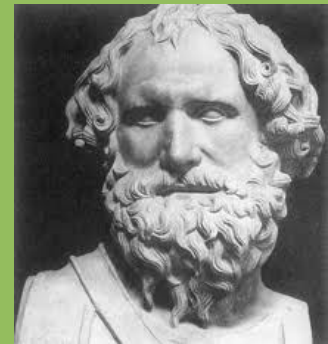
25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Cada
padrão
descreve o
núcleo da
solução
para esse
problema...



Arquimedes de Siracusa

atomosybits.com

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

**... de
forma tal
que você
pode usar
essa
solução
milhões de
vezes...**



bracreditesequiser.blogspot.com

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

**... sem
nunca
fazê-la da
mesma
forma
duas
vezes.**



enologia.org.br

** Christopher Alexander*

O QUE UM PADRÃO DE PROJETO É

- Um modelo de solução **comprovada**
- Uma linguagem **comum** para desenvolvedores
- Útil para compreender **frameworks**

SOLUÇÕES REUTILIZÁVEIS



O QUE UM PADRÃO DE PROJETO NÃO É

- A solução para todos os problemas de projeto
- Implementações finais
- **Panacéia**



www1.folha.uol.com.br



helenaeabelezadostextos.blogspot.com

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Padrão de projeto tem
a ver com
arquitetura de
software.

Referência: Gangue dos Quatro

Gang of Four at trial, 1981.



Yao Wenyuan



Jiang Qing



Zhang Chunqiao



Wang Hongwen

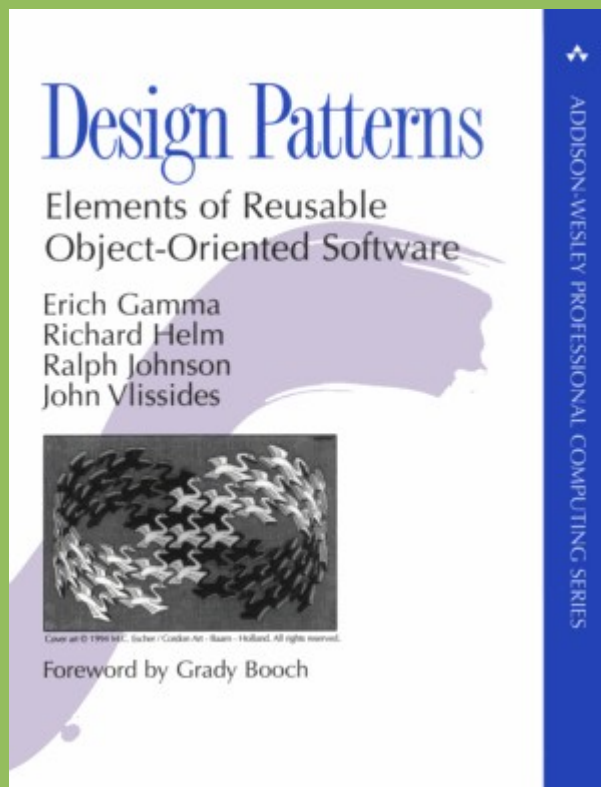
http://en.wikipedia.org/wiki/Gang_of_Four

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



1995



Ralph Johnson, Erich Gamma, Richard Helm e John Vlissides



O tempo passa para todos...

Eles descrevem **23 padrões de projeto** em seu livro, agrupados em três categorias: de criação, estruturais e comportamentais.

Veremos aqui **10** deles.



doggies.com

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Padrões de Criação



25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

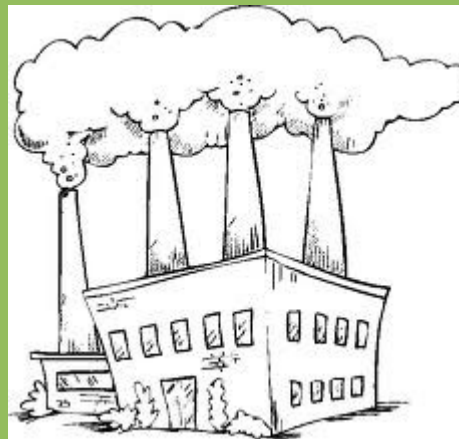


Abstract Factory & Factory Method

Classes



minhainfancia.com.br



colorirdesenhos.com

Objetos



faberludens.com.br

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Para que servem?

Para criar objetos complexos **facilmente** sem **copiar e colar**.



cmota.no.sapo.pt

Para que servem?

Para **desacoplar** a criação de determinados objetos de um sistema.



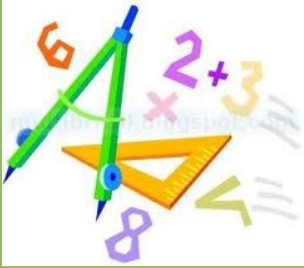
blog.brasilacademico.com

Para que servem?

Quando você **não sabe** qual objeto deve ser criado em determinado momento.



flickrriver.com



Qual a diferença entre os dois?

A diferença é *sutil*. **Abstract Factory** é usado quando você não sabe qual classe irá iniciar a solicitação para criação do objeto.

Factory Method é aplicado quando se tem uma classe determinada para quem os pedidos de criação serão dirigidos.

Exemplo: conexão com banco de dados

A aplicação pode trabalhar com diferentes “marcas” de bancos de dados.

Como saber qual objeto de conexão instanciar e ter a mesma interface para manipular os dados?

Assim?

```
if ($database == MYSQL)
{
    ...
}
if ($database == POSTGRESQL)
{
    ...
}
if ($database == SQLSERVER)
{
    ...
}
if ($database == ORACLE)
{
    ...
}
```



sargentolago.blogspot.com

Cubrid

dBase

DB++

FrontBase

filePro

Firebird/InterBase

Informix

IBM DB2 — IBM DB2, Cloudscape
and Apache Derby

Ingres — Ingres DBMS, EDBC, and
Enterprise Access Gateways

MaxDB

Mongo

mSQL

Mssql — Microsoft SQL Server



estou-sem.blogspot.com

MySQL

Mysqli — Extensão MySQL
Melhorada

Mysqlnd — MySQL Native Driver

mysqlnd_qc — Mysqlnd query
result cache plugin

OCI8 — Oracle OCI8

Ovrimos SQL

Paradox — Paradox File Access

PostgreSQL

SQLite

SQLite3

Sybase

tokyo_tyrant



É possível adotar uma **interface única** para a manipulação de dados com o uso de camadas de abstração de dados.

DBA — Database (dbm-style) Abstraction Layer

dbx

ODBC

PDO — PHP Data Objects



Com uma camada de abstração, uma **única classe** pode, por configuração, delegar a uma terceira, em tempo de execução, a criação do objeto de conexão.

```
$conn = Zend_Db::factory($adapter,$config);
```



25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



Singleton

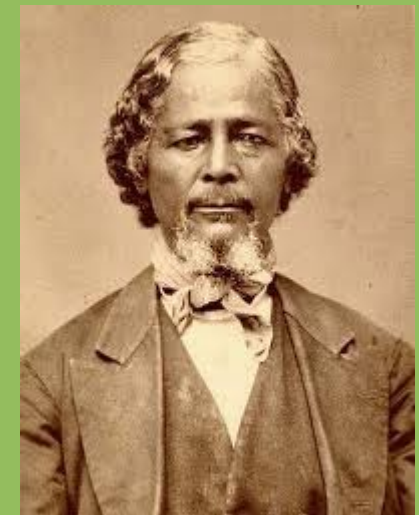


John

viswiki.com



marketingassassin.wordpress.com



Benjamin

kshs.org

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

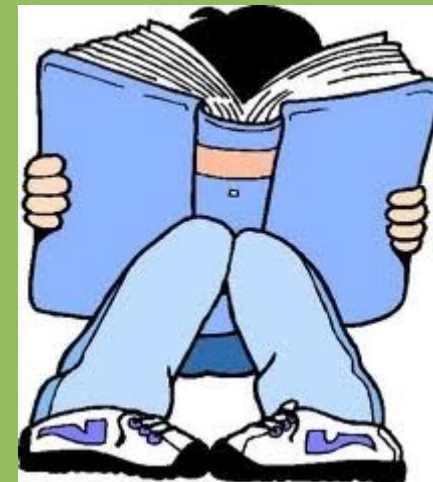
@fgsl

Para que serve?

Para garantir que uma instância seja **exclusiva** e com acesso **somente para leitura**.



digitaldrops.com.br



sarakertesz.blogspot.com

Como implementar?

Primeiro, torne o construtor da classe **privado**.

```
private function __construct()
```

Depois, crie um método **estático público** que retorne sempre a mesma instância (armazenada em um atributo **estático** da classe).

Como implementar?

```
class SingletonSample
{
    private static $_instance = null;

    private function __construct()
    {
    }

    public static function getInstance()
    {
        if (self::$_instance == null)
        {
            self::$_instance = new SingletonSample();
        }
        return self::$_instance;
    }
}
```

Como implementar?

```
$singleton = SingletonSample::getInstance();
```



25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Padrões Estruturais



portoalegre.olx.com.br

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



Adapter



Para que serve?

Para **proteger** sua aplicação de **mudanças** na API de bibliotecas externas.

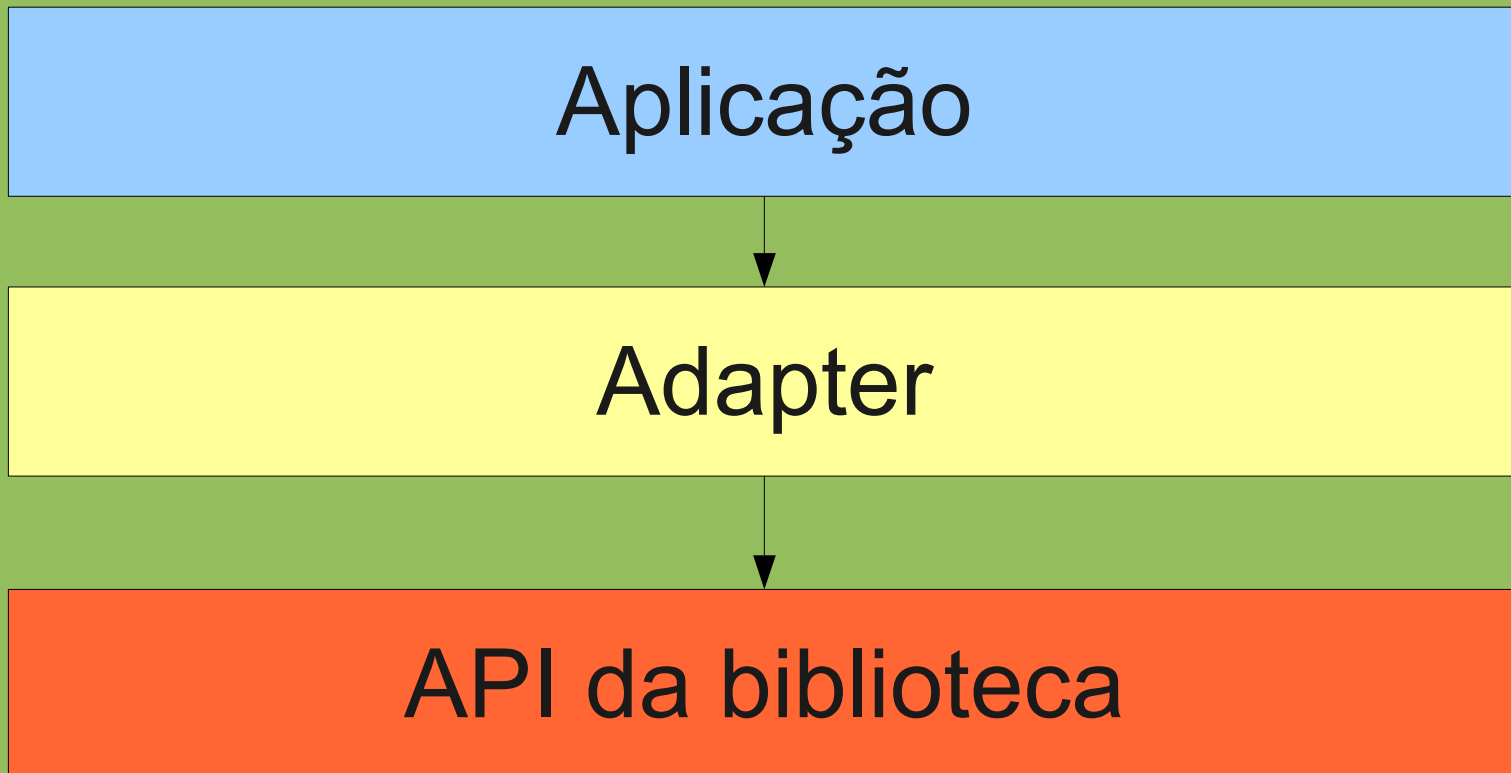
Para permitir que você crie bibliotecas que sejam fáceis de serem atualizadas pelos usuários, mesmo quando a API sofrer modificações.

Vantagens

Adapter desloca a manutenção de vários pontos da aplicação, ou de várias aplicações, para um único componente de software.

Permite evoluir a API de uma biblioteca **sem afetar o funcionamento** de uma aplicação existente.

Esquema geral



25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Dica



facilita a criação de adaptadores e já está
preparado para trabalhar com **namespaces**.

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



Decorator



rumahcons.blogspot.com



boards.nbc.com

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Para que serve?

Para estruturar seu código de modo a **adicionar facilmente** características usadas sob **certas condições** ou **raramente** sem colocar o código extra diretamente em sua classe.

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Como funciona?

Um objeto **incorpora** outros objetos em tempo de execução, **encapsulando** seus métodos. Os métodos do objetos encapsulado são chamados como se fosse do objeto encapsulador.

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Como funciona?



comicsofrhodey.blogspot.com



<http://www.marveldirectory.com/individuals/r/rogue.htm>



<http://www.strayhair.com>

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Como funciona?



figurerealm.com



thecomicsforums.com



dc.wikia.com

25 A 28 DE NOVEMBRO



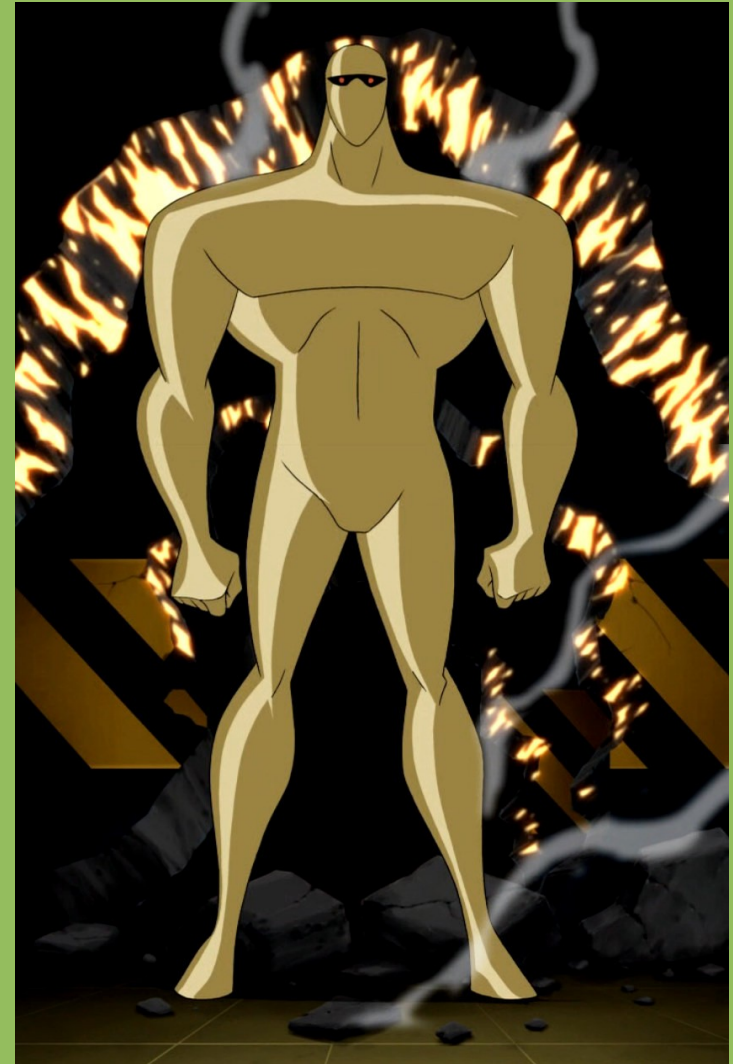
PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Como funciona?



forum.daz3d.com



outskirtsbattledome.wikispaces.com

Herança Múltipla

PHP não suporta herança múltipla. Mas a aplicação de Decorator permite que uma classe “herde” os métodos de várias classes ao mesmo tempo.

TER UM É MELHOR QUE SER UM!

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Mudança de Comportamento

Decorator também permite que o comportamento de um objeto seja alterado **depois que ele foi instanciado**, sem modificar o comportamento dos **outros objetos** da mesma classe.

Implementação

Você pode manter um atributo na classe a ser “decorada”, que guarde uma coleção de “decoradores”.

PHP possui o método **__call()**, que é invocado quando um método não pertencente à classe é chamado.

O método **__call()** pode procurar na coleção qual objeto tem o método chamado, e executá-lo.

Implementação

A classe “decorada” possui métodos para adicionar e remover “decoradores”.

Pode-se adotar uma convenção, em os métodos de adição e remoção dos “decoradores” sejam compostos pelos nomes das classes. Esses métodos não precisam ser definidos, pois o método **`__call()`** da classe “decorada” pode tratar isso.

Implementação

```
class Decorated
{
    const DEFAULT_PREFIX = 'addDecorator';
    private $_decorators = array();

    public function __call($method,$arguments)
    {
        if (substr($method,0,strlen(DEFAULT_PREFIX)-1) == DEFAULT_PREFIX)
        {
            $className = substr($method,strlen(DEFAULT_PREFIX));
            $this->_decorators[] = new $className();
        }
        else
        {
            foreach($this->_decorators as $decorator)
            {
                if (method_exists($decorator,$method))
                {
                    return $decorator->$method();
                }
            }
        }
    }
}
```

Implementação

Outra forma de usar Decorator é definir via interface dois métodos: um para adicionar “decoradores” e outro que, sempre que é executado, também executa o método (de mesmo nome) dos “decoradores”.

Implementação

```
class Decorated implements DecoratorInterface
{
    private $_decorators = array();

    public function addDecorator($decorator)
    {
        $this->_decorators[] = $decorator;
    }

    public function removeDecorator($decoratorSearched)
    {
        foreach($this->_decorators as $key => $currentDecorator)
        {
            $class = get_class($currentDecorator);
            if ($decoratorSearched instanceof $class)
            {
                unset($this->_decorators[$key]);
                break;
            }
        }
    }

    public function execute()
    {
        foreach($this->_decorators as $decorator)
        {
            $decorator->execute();
        }
    }
}
```

25 A 28 DE NOVEMBRO

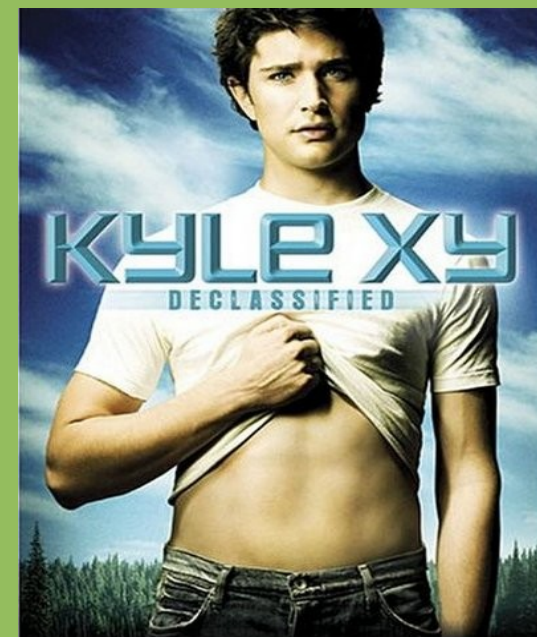


PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



Proxy



seriesecia-br.blogspot.com



fundamig.org.br

Para que serve?

Para prover acesso a um objeto sem fornecer o objeto **diretamente**.

Para adiar a criação de um objeto que usa recursos **caros** e nem sempre é necessário.

Para **restringir** acesso aos métodos de um objeto.

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

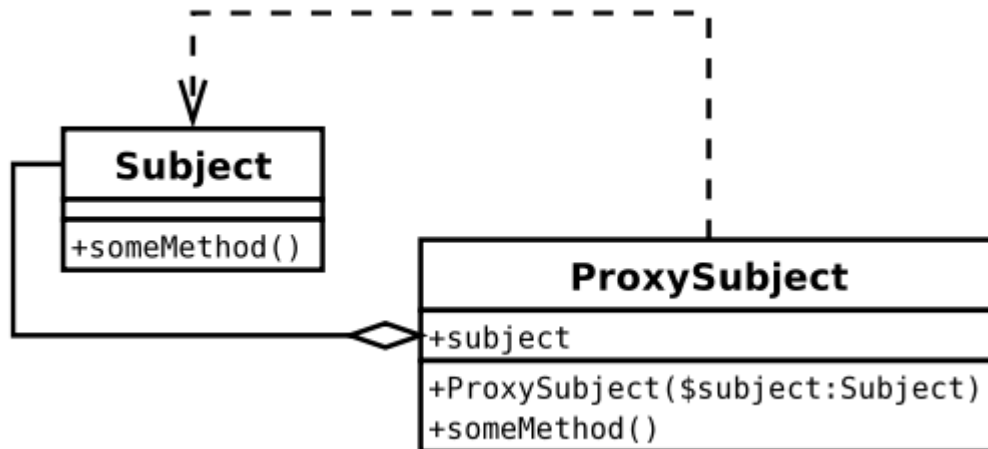
Para domar o programador



liquidsilver.org

caozen.blogspot.com

Como funciona?



PHP Architects Guide to PHP Design Patterns

```
class ProxySubject{
    private $_subject;

    private function lazyLoad()
    {
        if (!$this->_subject instanceof Subject) {
            $this->_subject = new Subject();
        }
    }

    public function execute()
    {
        $this->lazyLoad();
        return $this->_subject->execute();
    }
}

$proxy = new ProxySubject();
```

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Padrões Comportamentais



filipesoad.wordpress.com

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



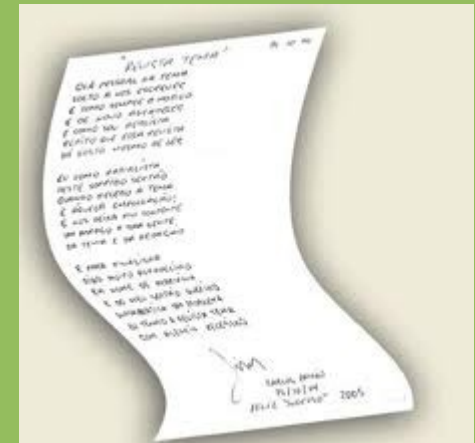
Chain of Responsibility



Spiderman, by Stan Lee & Steve Ditko

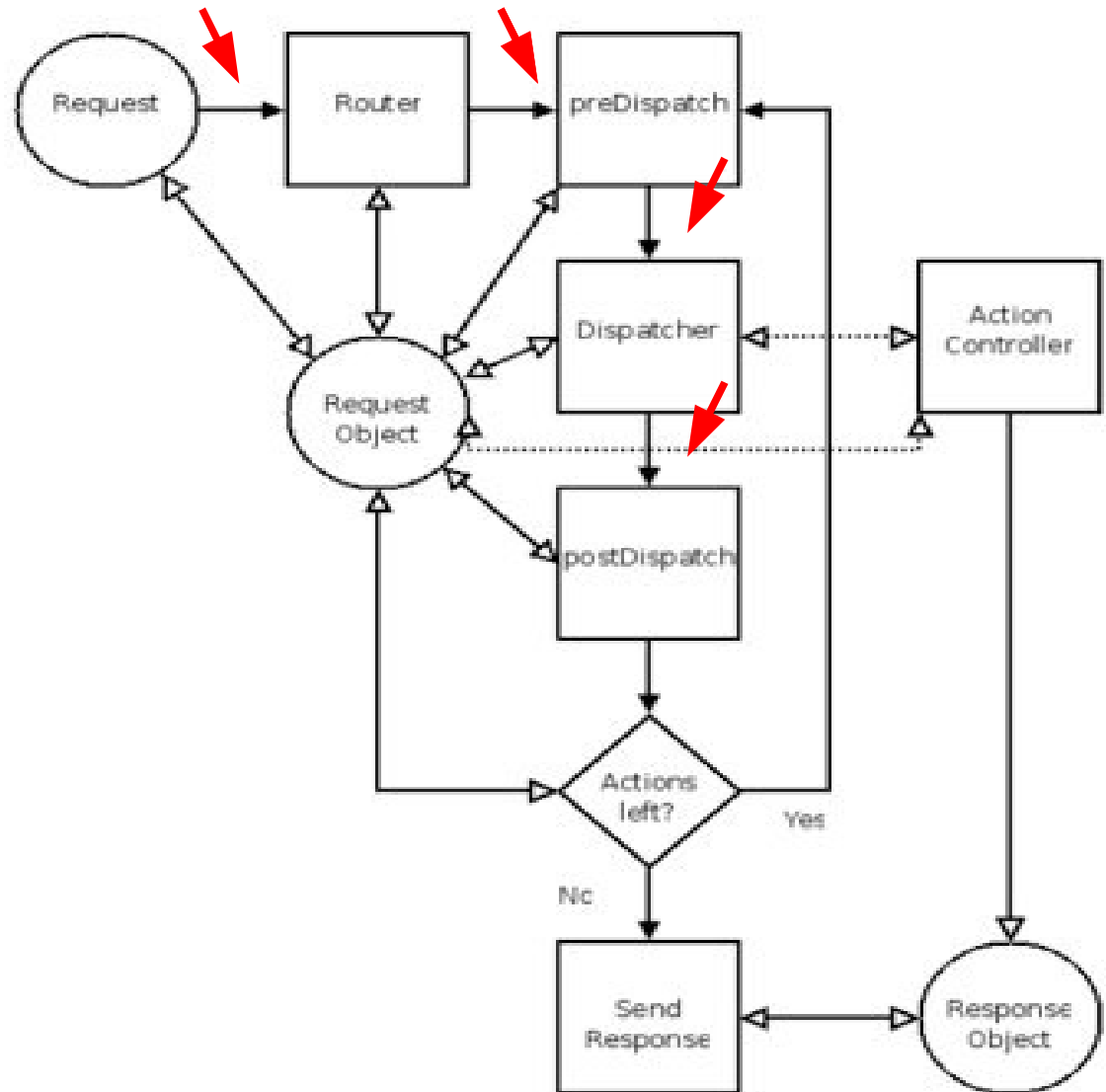
Para que serve?

Para evitar o **acoplamento** do **remetente** de uma solicitação ao seu **destinatário**, dando a mais de um objeto a chance de tratar a solicitação.



Estudo de Caso

Tratamento de requisição HTTP no Zend Framework



From Zend Framework in Action

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



Observer



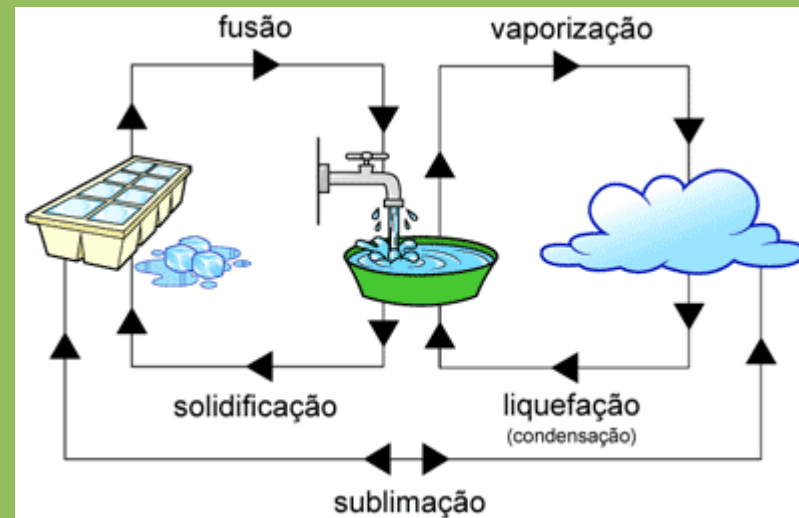
bakashihzonho.blogspot.com



pt.dreamstime.com

Para que serve?

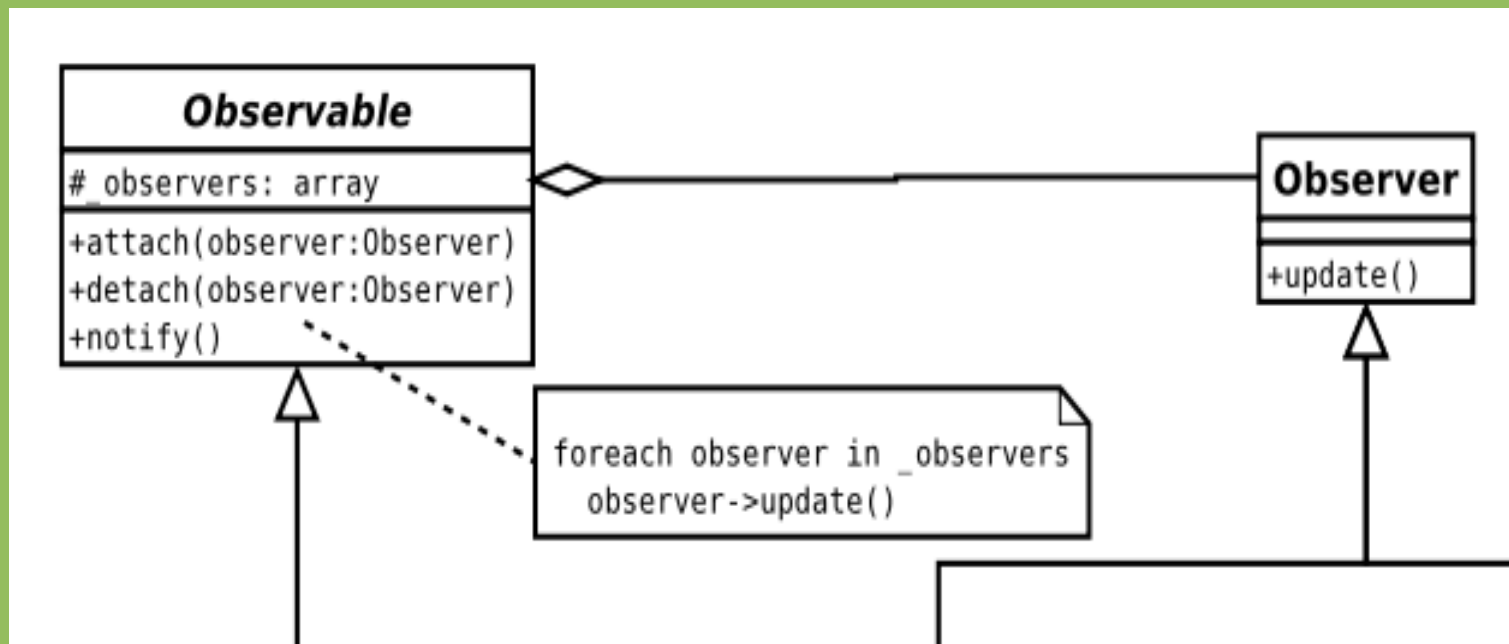
Para alertar **muitos** objetos quando o **estado** de **um** certo objeto mudar.



physikbr.blogspot.com

Como funciona?

Um objeto **Observável** registra **Observadores**. Quando o estado de Observável muda, ele notifica os Observadores.



PHP Architects Guide to PHP Design Patterns

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



Strategy



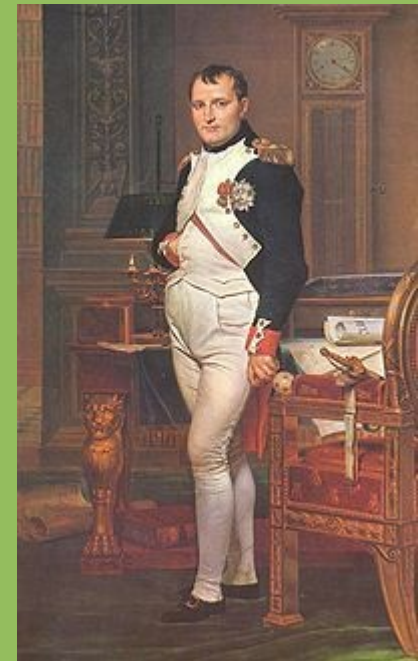
ndr.de



rajputbrotherhood.com



enciclopedia.com.pt



tudoehistoria.spaceblog.com.br

Para que serve?

Para alterar facilmente a **implementação interna** de um objeto, escolhendo uma implementação para ser usada no momento em que o script é **executado**.


Para codificar um conjunto de implementações fáceis de manter e estender.

Exemplo: algoritmos genéticos

Um algoritmo genético é uma técnica de busca, usada para encontrar soluções aproximadas em problemas de otimização.

Todos os algoritmos genéticos consistem dos mesmos passos. O que muda (além dos argumentos) é a **função-objetivo**, que irá avaliar os indivíduos.

Strategy

A red arrow pointing downwards from the word "Strategy" to the code block.

```
class GeneticAlgorithm
{
protected $_population = null;
protected $_args = array();
protected $_fitness = null;

public function __construct($population, array $args, $fitness)
{
$this->_population = $population;
$this->_args = $args;
$this->_fitness = $fitness;
}

public function execute()
{
if (null == $this->_fitness)
return $this->_fitness->execute($population, $args);
else
return $this->executeDefault();
}
}
```

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl



Iterator



miniempresafloredelotus.blogspot.com

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Para que serve?

Para manipular facilmente qualquer **coleção** de objetos.



monsores.net

Exemplo: iterando sobre resultados de consulta

Para manipular facilmente qualquer **coleção** de objetos.

```
$keys = Zend_Registry::get('keys');  
  
$dmc = new Categoria();  
  
$categorias = $dmc->find($keys);  
  
do  
{  
    $categoria = $categorias->current();  
    printf("%s", $categoria->nome);  
}while ($categorias->next())
```

25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Boas Práticas

- 1) Teste seu código
- 2) Automatize seus testes
- 3) Integre continuamente a aplicação



Boas Práticas

- 4) Refatore o código
- 5) Simplifique o código
- 6) Aplique padrões de projeto



25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Boas Práticas

- 7) Documente o código
- 8) Controle as versões



Boas Práticas

9) Use um ambiente de desenvolvimento **integrado**

10) Use **frameworks**, pois eles implementam os **padrões de projeto**



25 A 28 DE NOVEMBRO



PHP CONFERENCE BRASIL:
O PRINCIPAL EVENTO DE PHP DA AMÉRICA LATINA

@fgsl

Sugestões



Obrigado!

www.fgsl.eti.br



Little Einsteins by Walt Disney