

Profiling

Nosso código ainda pode ser melhorado, basta olharmos que existe repetição em nossos modelos, é bem provável que todas as nossas requisições irão acessar o banco de dados, neste caso é comum que queiramos carregá-lo automaticamente a cada requisição, fazer um autoload.

Dentro do diretório config, temos um `autoload.php`, lá podemos pedir para carregar automaticamente helpers, bibliotecas e afins.

Para isto vamos alterar a seguinte linha:

```
$autoload['libraries'] = array();
```

Ficará assim:

```
$autoload['libraries'] = array('database');
```

Vamos remover as linhas do model que fazem o load do database.

Verifique na página que continua tudo funcionando.

Agora repare no `produtos.php`, carregamos 3 helpers lá

```
public function index()
{
    $this->load->helper(array("url"));
    $this->load->helper(array("currency"));
    $this->load->helper(array("form"));
    $this->load->template("produtos/index.php", $dados);
}
```

Podemos carregar os três em uma única linha, passando todos separados por vírgula:

```
public function index()
{
    $this->load->helper(array("url", "currency", "form"));
    $this->load->template("produtos/index.php", $dados);
}
```

Já fica melhor certo? Isso que tínhamos apenas três helpers, imagine com mais.

Legal, agora, queremos ver o que está acontecendo no nosso servidor, no nosso PHP, vamos então ativar o profiling da nossa aplicação, vamos no controller que queremos ativar isto, neste caso no nosso `produtos.php`, no método `index`, vamos usar a nossa saída(output) para ativar o profiler:

```
public function index()
{
    $this->output->enable_profiler(TRUE);
    //resto do código
}
```

Atualize sua página e veja que temos diversas outras informações por exemplo:

Quanto tempo demorou para carregar e executar:

BENCHMARKS	
Loading Time: Base Classes	0.0057
Controller Execution Time (Produtos / Index)	0.0123
Total Execution Time	0.0180

Quais os parâmetros recebidos:

DADOS GET
Não existem dados no GET

Qual a classe e o método chamado:

CLASSE/MÉTODO
produtos/index

O uso de memória no servidor:

USO DE MEMÓRIA
2,127,856 bytes

Quais as queries executadas no banco:

BANCO DE DADOS: mercado QUERIES: 1 (Esconder)	
0.0004	SELECT * FROM (`produtos`)

Vamos ativar o profiling também no método novo do usuarios

Agora iremos inserir um novo usuário e verificamos que a query de insert foi executada, que os parâmetros post foram corretos.

Veja que o próprio code igniter já faz o escape das strings para evitar SQL injection:

- BANCO DE DADOS: mercado QUERIES: 1 (Esconder)

```
0.0009  INSERT INTO `usuarios` (`nome`, `email`, `senha`) VALUES ('Joana d\'arc', 'joana@dark.com', '3954805f195c592d88d7852a3df41402')
```