Módulo I

Uma requisição Drupal

Ao acessar uma página como https://www.drupal.org/about:

O browser requisita ao servidor:

```
GET /about HTTP/1.1
Host: www.drupal.org
```

O Apache transforma /about em /index.php?q=about

O index.php contém o seguinte:

```
require_once DRUPAL_ROOT . '/includes/bootstrap.inc';
drupal_bootstrap(DRUPAL_BOOTSTRAP_FULL);
menu_execute_active_handler();
```

Durante o bootstrap:

1. o settings.php é carregado
2. se a página solicitada está armazenada em cache, ela é retornada e a requisição é encerrada
3. o banco de dados é inicializado
4. a sessão do usuário é carregada
5. a hook hook_boot é rodada
6. descobre-se qual a língua ativa do site, com
7. todos os módulos são carregados (só agora!)
8. inicialização de path! O `$_GET['q']` é lido, *about*, se torna *node/1*, após uma consulta rápida à tabela *url_alias*
9. o tema é carregado
10. hook_init é rodado

Após o Drupal ter sido inicializado, roda-se o `menu_execute_active_handler()`, que por sua vez chama `menu_get_item()`. Aí, consultando a tabela *menu_router* (tabela que guarda um cache dos resultados dos hook_menus da instalação), descobrimos que o responsável por responder por "node/1" é o *item de menu* "node/%node", declarado por `node_menu()`.

```
$items ['node/%node'] = array(
    'title callback' => 'node_page_title',
    'title arguments' => array(1),
    'page callback' => 'node_page_view',
    'page arguments' => array(1),
    'access callback' => 'node_access',
    'access arguments' => array('view', 1),
```

```
);
```

De posse dessa informação, a requisição parte para rodar:

```
node_access('view', node_load(1));
```

Que vai dizer se nós temos acesso a ver o node 1. Uma vez decidido que sim, a função de *page callback* é rodada:

```
node_page_view('view', node_load(1));
```

Ela por sua vez vai preparar a exibição do node, e passar o resultado para a camada de tema.

Por fim, a camada de tema vai produzir o HTML final do site, e esta requisição será encerrada.

A criação de módulos no Drupal é essencialmente uma de duas coisas:

1. Criação de novos itens de menu; ou
2. Captura e alteração do processo de execução em algum momento da requisição descrita acima.

Hello World

Arquivo .info

Crie em sites/all/modules/custom/drupal_training, um arquivo drupal_training.info com o seguinte código:

```
name = Drupal Traning
description = Creates a module for Chuva's Drupal training.
core = 7.x
```

Crie um arquivo drupal_training.module com o seguinte código:

```
<?php

/**
 * Implements hook_menu().
 */
function drupal_training_menu() {
  $items = array();

  $items['hello-world'] = array(
    'title' => 'Hello world',
    'access callback' => TRUE,
    'page callback' => 'drupal_training_hello_world',
    'file' => 'drupal_training.pages.inc',
    'type' => MENU_CALLBACK,
  );

  return $items;
}
```

E por fim, um arquivo drupal_training.pages.inc, com o código:

```
<?php
/**
 * Page callback for hello-world.
 */
function drupal_training_hello_world() {
  return 'Hello World';
}
```

Criando o seu primeiro hook

Crie dentro do .module as seguintes funções:

```
/**
 * Implements hook_node_view().
 */
function drupal_training_node_view($node, $view_mode, $langcode) {
  if ($node->type == 'article' && !$node->status) {
    drupal_set_message(t('You are viewing the draft of an
article.'));
  }
}
```

Hooks úteis

`hook_init()`
Chamado no início da requisição, com todos os módulos já carregados.

`hook_form_alter(&$form, &$form_state, $form_id)`
Altera um formulário.

`hook_menu()`
Declara novos itens de menu.

`hook_menu_alter(&$items)`
Permite alterar itens de menu declarados por outros módulos.

`hook_cron()`
Permite rodar tarefas agendadas.

`hook_node_access($node, $op, $account)`
Define permissão de acesso para um node.

`hook_node_insert($node)`
Roda logo após um node novo ter sido salvo.

`hook_node_update($node)`
Roda logo após um node existente ter sido salvo.

`hook_node_presave($node)`
Roda logo antes de um node novo ser salvo.

`hook_node_delete($node)`
Roda logo antes de um node ser removido do banco de dados.

`hook_node_view($node, $view_mode, $langcode)`
Roda logo antes de um node ser exibido.

`hook_user_insert(&$edit, $account, $category)`
`hook_user_update(&$edit, $account, $category)`
`hook_user_presave(&$edit, $account, $category)`
`hook_user_delete($account)`
`hook_user_view($account, $view_mode, $langcode)`
Idem, para um usuário.

`hook_update_N()` (onde N é um número)
Permite executar algum script ao rodar update.php

`hook_drupal_goto_alter()`
Permite alterar o destino de um drupal_goto(). (use com cuidado!!)

Banco de dados

db_query

```
global $user;
$result = db_query("
  SELECT n.nid, n.title, n.created
    FROM {node} n
    WHERE n.uid = :uid
      AND n.type = 'article'
    ORDER BY created",
  array(':uid' => $user->uid));

foreach ($result as $article) {
  $titles[] = check_plain($article->title);
}
```

db_select

```
global $user;
$result = db_select('node', 'n')
  ->fields('n', array('nid', 'title', 'created'))
  ->condition('n.uid', $user->uid)
  ->condition('n.type', 'article')
  ->execute();

foreach ($result as $article) {
  $titles[] = check_plain($article->title);
}
```

db_update

```
db_update('system')
```

```
  ->fields(array('weight' => 10))
  ->condition('name', 'mymodule')
  ->condition('type', 'module')
  ->execute();
```

db_delete

```
db_delete('example_table')
  ->condition('id', 10)
  ->execute();
```

hook_query_alter

```
$query->addTag('micro_limit');

(...)

function drupal_training_query_alter(QueryAlterableInterface
$query) {
  if ($query->hasTag('micro_limit')) {
    $query->range(0, 2);
  }
}
```

transactions

```
function my_transaction_function() {
  // The transaction opens here.
  $transaction = db_transaction();

  try {
     $id = db_insert('example')
     ->fields(array(
     'field1' => 'mystring',
     'field2' => 5,
     ))
     ->execute();

     my_other_function($id);

     return $id;
  }
  catch (Exception $e) {
     $transaction->rollback();
     watchdog_exception('my_type', $e);
  }
```

```
  // $transaction goes out of scope here.  Unless it was rolled
back, it
  // gets automatically commited here.
}

function my_other_function($id) {
  // The transaction is still open here.

  if ($id % 2 == 0) {
     db_update('example')
     ->condition('id', $id)
     ->fields(array('field2' => 10))
     ->execute();
  }
}
```

Blocos

Declaração de um bloco

Adicionar no .module:

```
function drupal_training_block_info() {
  $blocks['my_block'] = array(
    'info' => t('My Drupal Training block'),
    'cache' => DRUPAL_NO_CACHE,
  );

  return $blocks;
}
```

Opções de cache

DRUPAL_CACHE_PER_ROLE (*default*): O cache do bloco é feito por papel de usuário.
DRUPAL_CACHE_PER_USER: O cache do bloco é por usuário.
DRUPAL_CACHE_PER_PAGE: O bloco pode mudar por página em que é visualizado.
DRUPAL_CACHE_GLOBAL: O bloco é o mesmo em todas as páginas do site em que é exibido
DRUPAL_NO_CACHE: O bloco não será cacheado.

Exibição do bloco

Adicionar no .module:

```
/**
 * Implements hook_block_view().
```

```
 */
function drupal_training_block_view($delta = '') {
  $block = array();

  switch ($delta) {
    case 'my_block':
      $block['subject'] = t('Block title');
      $block['content'] = t('Hello world');
      break;
  }

  return $block;
}
```

Formulários

Adicionar no .module:

```
/**
 * Implements hook_menu().
 */
function drupal_training_menu() {
  $items = array();

  $items['node/%node/hello-form'] = array(
    'title' => 'Hello forms',
    'access callback' => TRUE,
    'page callback' => 'drupal_get_form',
    'page arguments' => array('drupal_training_hello_form', 1),
    'file' => 'drupal_training.pages.inc',
    'type' => MENU_LOCAL_TASK,
  );

  return $items;
}
```

Adicionar no drupal_training.pages.inc:

```
/**
 * Callback function for hello-form.
 */
function drupal_training_hello_form($form, &$form_state, $node) {
  $form['#node'] = $node;

  $form['new_title'] = array(
    '#type' => 'textfield',
```

```
    '#title' => t('New title'),
    '#default_value' => $node->title,
    '#required' => TRUE,
  );

  $form['accept'] = array(
    '#type' => 'checkbox',
    '#title' => t('I accept the terms and conditions'),
    '#required' => TRUE,
  );

  $form['submit'] = array(
    '#type' => 'submit',
    '#value' => t(Change title'),
  );

  return $form;
}

/**
 * Validate callback for drupal_training_hello_form.
 */
function drupal_training_hello_form_validate($form, &$form_state) {
  if (strlen($form_state['values']['new_title']) < 3) {
    form_set_error(
      'new_title',
       t('The new title must have at least 3 characters'));
  }
}

/**
 * Submit callback for drupal_training_hello_form.
 */
function drupal_training_hello_form_submit($form, &$form_state) {
  $form['#node']->title = $form_state['values']['new_title'];
  node_save($form['#node']);
  $form_state['redirect'] = 'node/' . $form['#node']->nid;
}
```

Nesting

Você pode colocar o new_title dentro de um fieldset:

```
$form['fieldset'] = array(
  '#type' => 'fieldset',
);
```

```
$form['fieldset']['new_title'] = array(
  '#type' => 'textfield',
  '#title' => t('New title'),
  '#required' => TRUE,
);
```

Como acessar o valor submetido? Da mesma maneira que antes:

```
$form_state['values']['new_title']
```

Caso eu queira ter uma estrutura diferentes no values, ela pode ser conseguida com o seguinte código:

```
$form['my_group'] = array(
  '#type' => 'fieldset',
  '#tree' => TRUE,
);
```

E o valor submetido passa a ser acessado com:

```
$form_state['values']['my_group']['new_title']
```

Módulo II

Metodologias para interação de forma correta e proveitosa dos principais módulos, redes sociais, formulários e informações integradas com dados de terceiros.

- Interação e extensão com as APIs dos principais módulos contribuídos:
    - Views na manipulação da listagem de conteúdo;
    - Context na manipulação dinâmica de conteúdo contextual;
- Implementação avançada do Form API;
- Uso de Entity API na extensão de funcionalidades com aproveitamento de toda a flexibilidade do Drupal;
- Integração de Drupal com Webservices, criação e consumo de API's
- Criação de hooks personalizados para prover extensibilidade;

Entity

Conceito

Criando uma tabela para sua Entity

No .install:

function drupal_training_

```
hook_entity_insert($entity, $type)
hook_entity_update($entity, $type)
hook_entity_presave($entity, $type)
hook_entity_view($entity, $type, $view_mode, $langcode)
```

Idem, para qualquer entity.

Módulo III

Ferramentas para desenvolvimento e organização do site modular com Features e Code-Driven Development, para a colaboração entre membros da equipe de desenvolvimento, reutilização de código e para rastrear precisamente suas mudanças. Utilização do Drush.

- **Features**
    - **Conceituação de Features**

      O módulo Features permite que funcionalidades existentes no banco de dados sejam exportadas para código e reutilizadas em outros projetos.

      A nomenclatura do módulo pode causar um pouco de confusão, mas tenha em mente que a principal diferença entre uma feature e um módulo normal é que uma feature é criada para atender determinado caso de uso, como por exemplo a necessidade de uma página de noticias em um site.

      A página de noticias irá possuir uma view, um content type e múltiplos fields que se encontram no banco de dados e que queremos utilizar em outros projetos. Para fazer isso, nós exportamos todas essas informações para código utilizando o módulo Features.

    - **Criação e atualização de features**

      Considerando que já tenhamos alguma funcionalidade em nosso site que deve ser exportada, siga os seguintes passos para criar nossa primeira feature:

      Fazer o download do módulo features

Acesse drupal.org/project/features e baixe o módulo em sites/all/modules/contrib

ou

drush dl features

Ativar o módulo features

Acesse admin/modules e procure o módulo features na lista de módulos

ou

drush en features

Criar uma feature

Acesse admin/structure/features

Durante a criação da feature, tenha atenção ao seguinte:

Machine name: Apesar de se chamar feature, na pratica estamos criando um módulo, só que usando a interface, por essa razão, tente utilizar um machine name que não conflite com outros módulos ou features, geralmente prefixar o nome da feature com a sigla ou nome do projeto pode ser uma boa opção.

Components: Aqui são definidas as partes que vão compor a feature, imagine que uma feature é como um LEGO, onde cada componente é como se fosse uma pecinha que será usada para monta-lo, basta escolher entre as opções disponiveis (content types, views, etc..)

Download Feature: Até agora nada realmente foi feito, nós apenas definimos quais serão as peças que irão compor a nossa feature. Ao clicar no botão Download Feature, o módulo Features irá pegar todas as peças que definimos e criar um módulo por você. Ele irá escrever o PHP e todos os arquivos necessários para que ele seja um módulo do Drupal.

Descompacte o arquivo gerado em sites/all/modules/features

Ative seu novo módulo

Acesse admin/structure/features, marque a checkbox e salve.

ou

drush en nome_do_modulo

Pronto, você criou sua primeira feature, que pode ser reutilizada em múltiplos projetos.

Atualizar uma feature

É um processo bem simples porem é importante que seja feito com atenção. SEMPRE que alguma funcionalidade exportada anteriormente for alterada no site, é necessário exporta-la novamente atualizando a feature, isso não é um processo automático.

Após alterar algo que foi exportado para feature anteriormente (uma view, content type, context, etc..) acesse **admin/structure/features** e confirme o status da feature. Se

estiver **Default** então a feature não precisa ser atualizada, porem se estiver como **Overriden** será necessário atualiza-la para exportar as alterações para código.

É possível atualizar pela interface mesmo, clicando primeiramente no link **overriden** e depois em **recreate**.

Caso sejam apenas alterações em funcionalidades previamente exportadas, apenas clique em **Download Feature** que um novo arquivo será gerado com todas as modificações. Caso também queira adicionar novas funcionalidades na sua feature, basta selecionar os componentes novos antes de clicar no botão de Download.

Descompacte a feature e sobreescreva o diretório original. Se tudo foi feito corretamente sua feature irá aparecer com status **Default** na listagem de features.

- **Versionando e deploy com features**

Uma das grandes vantagens de se utilizar Features é se beneficiar de ferramentas de controle de versão como o GIT. Sem a utilização de features o único modo de compartilhar e/ou reutilizar uma funcionalidade custom (página de noticias, blog, etc..) seria exportando e importando o banco de dados.

Como uma feature é nada mais que um módulo, é possível adiciona-la, altera-la e remove-la de um repositório com facilidade.

Veja o capitulo sobre GIT para mais informações sobre como versionar e deployar arquivos.

- **Manipulação de features com Drush**

features-list: Lista todas as features disponíveis no seu site
Aliases: fl, features
Exemplo: `drush features / drush fl`

features-update: Re-exporta alterações na feature para código.
Aliases: fu
Exemplo: `drush features-update nome_da_feature / drush fu nome_da_feature`

features-update-all: Re-exporta alterações de todas as features do seu site para código.
Aliases: fu-all, fua
Exemplo: `drush features-update-all / drush fua`

features-revert: Reverte qualquer alterações na feature
Aliases: fr
Exemplo: `drush features-revert nome_da_feature / drush fr nome_da_feature`

features-revert-all: Reverte todas as features do seu site. (muito cuidado!)
Aliases: fr-all, fra

Exemplo: `drush features-revert-all / drush fra`

features-diff: Exibe um diff da feature
Aliases: fd
Exemplo: `drush fd nome_da_feature`

- **Gerenciamento de Configurações**
  - Quando usar
  - Como interagir com features
  - Melhorando a sua estratégia de deploy

Drush

Drush é uma ferramenta para executar certas tarefas administrativas do Drupal via linha de comando, agilizando certas tarefas administrativas. É especialmente útil para lidar com ambientes de produção, onde algumas linhas de comando drush por SSH são uma maneira bastante sólida de se executar atualizações.

Instalação

Instalação do composer (caso já não tenha instalado):

```
curl -sS https://getcomposer.org/installer | php
sudo mv composer.phar /usr/local/bin/composer
```

Instalação do drush:

```
composer global require drush/drush:7.*
```

Comandos úteis

`drush cc`
limpa vários tipos de cache, com as variantes

`drush cc all`
limpa todos os caches do Drupal

`drush dl nome_do_projeto ou drush dl nome_do_projeto-7.x-1.3`
Baixa um projeto (p.ex. oauth) do drupal.org. Se não for especificada a versão, ele baixa a versão suportada mais recente.

`drush en nome_do_modulo`
Habilita o módulo nome_do_modulo

`drush dis nome_do_modulo`
Desabilita o módulo module_name.

`drush pm-uninstall nome_do_modulo`
Desinstala o módulo module_name (i.e., remove todas as suas tabelas no banco de dados).

`drush fr nome_da_feature`
Reverte qualquer alterações na feature, voltando ao estado padrão do código.

`drush fra`
Reverte todas as features do seu site. (muito cuidado!)

`drush fu nome_da_feature`
Re-exporta alterações na feature para código.

`drush uli nome_de_usuario`
Gera um link de recuperação de senha (extremamente conveniente para se logar em um site sem saber a senha dele)

`drush upwd nome_de_usuario --password=nova_senha`
Atualiza a senha do usuário.

`drush vget nome_da_variável`
Retorna o resultado de uma variável.

`drush vset nome_da_variável valor_novo_da_variável`
Define o valor de uma variável.

`drush vdel nome_da_variável`
Apaga uma variável.

`drush sql-cli`
Acessa a linha de comando MySQL.

`drush sql-cli < nome_do_arquivo.sql`
Roda um arquivo SQL (muito cuidado!)

`drush sqlq "SELECT * FROM users"`
Roda um comando SQL diretamente sem ter que entrar na linha de comando MySQL.

drush si
Cria uma nova instalação do Drupal.

```
drush sql-dump > nome_do_arquivo.sql
```
Faz um dump do banco de dados.

```
drush php-eval "print_r(node_load(3));"
```
Roda um comando PHP. (muito cuidado!)

```
drush scr arquivo.php
```
Roda um script php no arquivo. Nesse script, todas as funções do Drupal estarão disponíveis e o banco de dados do site estará carregado.

```
drush updb
```
Roda as atualizações de banco de dados. (equivalente a rodar /update.php)

```
drush cron
```
Roda o cron.

Criação de comandos drush

TODO