Drupal Book ≡

Menus

views 100

Dynamically change menu items with hook_preprocess_menu

In the example below, we're changing the labels of items in the user menu. The labels are changed from "login" and "log out" to "log the flock in" and "log the flock out." This can be implemented in a theme file as it is here.

```
/**
 * Implements hook_preprocess_menu().
 */
function pega_academy_theme_preprocess_menu(&$vars, $hook) {
  if ($hook == 'menu__account') {
    $items = $vars['items'];
    foreach ($items as $key => $item) {
      if ($key == 'user.page') {
        $vars['items'][$key]['title'] = [
          '#markup' => 'Log the <i>flock</i> in!',
          '#allowed_tags' => ['i'],
        ];
      }
      if ($key == 'user.logout') {
        $vars['items'][$key]['title'] = [
          '#markup' => 'Log the <i>flock</i> out!',
          '#allowed_tags' => ['i'],
        ];
      }
    }
  }
}
```

Permanently update menu links in a hook_update using entityQuery

To update menu item links, you can use the following code (from a .install file).

```php
function pdq_academy_core_update_8002() {

  $mids = \Drupal::entityQuery('menu_link_content')
    ->condition('menu_name', 'pdq-wide-utility')
    ->execute();

  foreach($mids as $mid) {
    $menu_link = \Drupal::entityTypeManager()->getStorage('menu_link_content')->load($mid);

    $title = $menu_link->getTitle();
    if ($title === 'Support') {
      $menu_link->set('weight',2);
      $menu_link->set('expanded', TRUE);
//      $menu_link->set('title','yomama');
      $menu_link->set('link', 'https://www.google.com');
      $menu_link->save();

    }
  }
}
```

Add menu items with hook_update

Menus are config entities while menu items are content entities. Here, a hook_update creates some menu items and adds them to an existing menu.

```php
function pdq_archive_core_update_8001() {
  $items = [
    '1' => ['Training', 'https://abc.pdq.com/', -51, TRUE],
    '2' => ['Support', 'https://abc.pdq.com/search/product-support', -50, TRUE],
  ];

  foreach($items as $item) {
    $menu_link = Drupal\menu_link_content\Entity\MenuLinkContent::create([
      'title' => $item[0],
      'link' => ['uri' => $item[1]],
      'menu_name' => 'pdq-wide-utility',
      'weight' => $item[3],
      'expanded' => $item[4],
    ]);
    $menu_link->save();
  }
}
```

Permanently modify or delete menu items with hook_update

Below, we use hook_update to grab all the menu items in the menu called pdf-wide-utility, then loop thru them, delete some, and change the weight of some.

```
function pdq_archive_core_update_8001() {

  // Update and remove unused menu items
  $mids = \Drupal::entityQuery('menu_link_content')
    ->condition('menu_name', 'pdq-wide-utility')
    ->execute();

  foreach($mids as $mid) {

    $menu_link = \Drupal::entityTypeManager()->getStorage('menu_link_content')->load($mid);

    $title = $menu_link->getTitle();
    if ($title === 'Pdq.com' || $title === 'PDQ Community' || $title === 'Careers') {
      $menu_link->delete();
    }
    if ($title === 'Support') {
      $menu_link->set('weight',2);
      $menu_link->set('expanded', TRUE);
      //    $menu_link->set('title','yomama');
      //    $menu_link->set('link', 'https://www.google.com');
      $menu_link->save();
    }
  }
}
```

If you need to get the parent value so you can make a menu item a child, use:

```
$parent_id = $menu_link->getPluginId();
```

and

```
$menu_link->set('parent', $parent_id);
```

Peer up a menu to its parents to see if it is a child of a content type

Here, we need to display a sidebar if the current node is both a page and a child (or any level of offspring, e.g., grandchild, great-grandchild, etc.) of a content type "unit." This means we need to "peer" up the menu chain to see what kind of parent the node has.

This example was implemented in a .theme file.

When hook_preprocess_node is called for my content type, and we are viewing a full node, we grab the nid and call the _check_ancestry_for_unit(), which looks up the menu chain for a menu item that points to a node of type "unit". If there is one, we display the sidebar, i.e., we set the $variables ['show_sidebar_menu'] to TRUE.

```
function txg_preprocess_node(&$variables) {
  $node = $variables['node'];

  if (($node->getType() == 'page') && ($view_mode == 'full')) {
    // This will only be true if it is a node route.
    if ($node = \Drupal::request()->attributes->get('node')) {
      if (is_string($node)) {
        $node_id = $node;
      }
      else {
        $node_id = $node->id();
      }
    }
    $show_sidebar = _check_ancestry_for_unit($node_id);
    $variables['show_sidebar_menu'] = $show_sidebar;
  }
```

Here is the function _check_ancestry_for_unit():

```php
/**
 * Go look up the menu chain for a unit.
 *
 * @param int $node_id
 *   Start with this node.
 * @param int $unit_nid
 *   Fill in the unit nid that you found.
 *
 * @return bool
 *   Return TRUE if there is a unit in the lineage.
 *
 * @throws \Drupal\Component\Plugin\Exception\PluginException
 */
function _check_ancestry_for_unit(int $node_id, &$unit_nid = 0, &$menu_item_title = '') {
  /** @var \Drupal\Core\Menu\MenuLinkManagerInterface $menu_link_manager */
  $menu_link_manager = \Drupal::service('plugin.manager.menu.link');
  $links = $menu_link_manager->loadLinksByRoute('entity.node.canonical', ['node' => $node_id]);

  /** @var \Drupal\Core\Menu\MenuLinkContent $link */
  $link = reset($links);
  if (!empty($link)) {
    while ($parent_menu_id = $link->getParent()) {

      // Create a menu item.
      /** @var \Drupal\Core\Menu\MenuLinkInterface $parent */
      $parent_menu_item = $menu_link_manager->createInstance($parent_menu_id);
      $parent_url = $parent_menu_item->getUrlObject();

      // Is the parent an External link e.g. nytimes.com?
      $external = $parent_url->isExternal();
      if ($external) {
        return FALSE;
      }

      // Internal links (could be <nolink>)
      $internal = $parent_url->isRouted();
      $parent_route = $parent_url->getRouteParameters();
      // When $parent_route is <nolink>.
      if (empty($parent_route)) {
        return FALSE;
      }

      // Parent is an internal link.
      // Does parent menu item refer to a unit?
      if (isset($parent_route['node'])) {
        $parent_nid = $parent_route['node'];
        $parent_node = Node::load($parent_nid);
        $parent_type = $parent_node->getType();
        if ($parent_type == 'unit') {
          $unit_nid = $parent_nid;
          $menu_item_title = $parent_menu_item->getTitle();
          return TRUE;
        }
        $links = $menu_link_manager->loadLinksByRoute('entity.node.canonical', ['node' => $parent_nid]);
        $link = reset($links);
      }
      else {
        // Some other kind of internal link.
        return FALSE;
      }
    }
  }
  return FALSE;
}
```

Find all the children of a menu

From a .module file, I needed to load a dropdown with items from the main menu.

You can load the menu up with this:

```
use Drupal\Core\Menu\MenuTreeParameters;


function get_menutree($menu_name) {
  $parameters = new MenuTreeParameters();

  // Only enabled items.
  $parameters->onlyEnabledLinks();

  // Load the tree.
  $tree = \Drupal::menuTree()->load($menu_name, $parameters);
  return $tree;
}
```

This is basically the same as:

```
$sub_nav = $menu_tree->load('main', new \Drupal\Core\Menu\MenuTreeParameters());
```

And here we call get_menutree() and then loop through the menu items to build some arrays to be rendered via a twig template. We loop through the top level, skipping non-nodes:

```php
function get_offices() {

  //Get drupal main menu
  $tree = get_menutree('main');
  $menu_tree = \Drupal::menuTree();

  // Build a renderable array from the tree which fills in all the children in
  // item['below']
  $menu_render_array = $menu_tree->build($tree);
  $storage = [];
  $nid = 0;
  foreach ($menu_render_array['#items'] as $item) {
    $url = $item["url"];
    $route = $url->getRouteParameters();
    // Skip menu items that aren't nodes.
    if (!isset($route['node'])) {
      continue;
    }
    $nid = $route["node"];
    $node = Node::load($nid);
    $type = $node->getType();
    if ($type = "area_of_focus") {
      $storage[] = [
        'title' => $item['title'],
        'value' => $nid
      ];
    }

    // Process children:
    foreach ($item['below'] as $child) {
      $url = $child["url"];
      $route = $url->getRouteParameters();
      $nid = 0;
      $type = '';
      if (isset($route['node'])) {
        $nid = $route["node"];
        $node = Node::load($nid);
        $type = $node->getType();

      }
      // Only add units.
      if ($type == 'unit') {
        $storage[] = [
          'title' => '--' . $child['title'] . ' (' . $type . ')',
          'value' => $nid
        ];
      }
    }
  }
  return $storage;
}
```

And the template that is used to display the dropdown, node--news-stories-landing-page.html.twig:

```html
<form action="#" class="filter-form">
 <div class="search-form">
  <input type="search" placeholder="Search Events by Keyword" title="Type search text here">
  <button type="button"><i class="icon icon-search"></i><span class="show-for-sr">Search</span></button>
 </div>
 <div class="grid-x grid-margin-x align-justify">

 </div>
</form>
```

And the form looks like this:

| Search Events by Keyword | 🔍 |

By Office
Select Office ⇕

By Topic
Select Topic ⇕

By Continent
Select Continent ⇕

By Country
Select Country ⇕

Build a menu and all its children

This also looks pretty interesting, but I haven't tried it. It is from https://stackoverflow.com/questions/54245942/drupal-8-menulinkcontent-get-all-children-via-loadbyproperties/54254491

```php
function generateSubMenuTree(&$output, &$input, $parent = FALSE) {
  $input = array_values($input);
  foreach ($input as $key => $item) {
    //If menu element disabled skip this branch
    if ($item->link->isEnabled()) {
      $key = 'submenu-' . $key;
      $name = $item->link->getTitle();
      $url = $item->link->getUrlObject();
      $url_string = $url->toString();

      //If not root element, add as child
      if ($parent === FALSE) {
        $output[$key] = [
          'name' => $name,
          'tid' => $key,
          'url_str' => $url_string
        ];
      }
      else {
        $parent = 'submenu-' . $parent;
        $output['child'][$key] = [
          'name' => $name,
          'tid' => $key,
          'url_str' => $url_string
        ];
      }

      if ($item->hasChildren) {
        if ($item->depth == 1) {
          generateSubMenuTree($output[$key], $item->subtree, $key);
        }
        else {
          generateSubMenuTree($output['child'][$key], $item->subtree, $key);
        }
      }
    }
  }
}
```

It is called with:

```php
//Get drupal menu
$sub_nav = \Drupal::menuTree()->load('main', new \Drupal\Core\Menu\MenuTreeParameters());

//Generate array
generateSubMenuTree($menu_tree2, $sub_nav);
```

Create custom Twig extension for rendering a menu

Note. The module twig_tweak module can do all this with one line of code:

```twig
{{ drupal_menu('main', 2, 3, TRUE) }}
```

More at: https://www.drupal.org/docs/8/modules/twig-tweak/cheat-sheet

From https://www.drupal.org/forum/support/theme-development/2015-01-29/rendering-a-menu-in-twig-drupal-8, Peter from Dusseldorf shows how to render a menu into a render array. He runs through the whole load, transform (with manipulators) and build. He does this through the magic of a twig extension. So, in modules/custom/custom_module/src/Twig/RenderMenuExtension.php:

```php
namespace Drupal\custom_module\Twig;

class RenderMenuExtension extends \Twig_Extension {

  /**
   * @return array
   */
  public function getFunctions() {
    return [
      new \Twig_SimpleFunction('renderMenu', [$this, 'renderMenu']),
    ];
  }

  /**
   * Provides function to programmatically rendering a menu
   *
   * @param String $menu_name
   *   The machine configuration id of the menu to render
   */
  public function renderMenu(string $menu_name) {
    $menu_tree = \Drupal::menuTree();

    // Build the typical default set of menu tree parameters.
    $parameters = $menu_tree->getCurrentRouteMenuTreeParameters($menu_name);

    // Load the tree based on this set of parameters.
    $tree = $menu_tree->load($menu_name, $parameters);

    // Transform the tree using the manipulators you want.
    $manipulators = [
      // Only show links that are accessible for the current user.
      ['callable' => 'menu.default_tree_manipulators:checkAccess'],
      // Use the default sorting of menu links.
      ['callable' => 'menu.default_tree_manipulators:generateIndexAndSort'],
    ];
    $tree = $menu_tree->transform($tree, $manipulators);

    // Finally, build a renderable array from the transformed tree.
    $menu = $menu_tree->build($tree);

    $menu['#attributes']['class'] = 'menu ' . $menu_name;

    return ['#markup' => drupal_render($menu)];
  }

}
```

Oh, and you do need to implement the getFunctions() in your class. It looks something like this:

```php
public function getFunctions() {
  $context_options = ['needs_context' => TRUE];
  $all_options = ['needs_environment' => TRUE, 'needs_context' => TRUE];
  return [
    new \Twig_SimpleFunction('drupal_render', [$this, 'renderMenu']),
  ];
}
```

In custom_module.services.yml:

```yaml
services:
  custom_module.render_menu_extension:
    class: Drupal\custom_module\Twig\RenderMenuExtension
    tags:
      - { name: twig.extension }
```

To render your menu in the template via this twig function call:

```twig
{{ renderMenu('main') }}
```

Active Trail

From
https://api.drupal.org/api/drupal/core%21lib%21Drupal%21Core%21Menu%21MenuTreeParameters.php/property/MenuTreeParameters%3A%3AactiveTrail/9.3.x

The IDs from the currently active menu link to the root of the whole tree.

Active trail is an array of menu link plugin IDs, representing the trail from the currently active menu link to the ("real") root of that menu link's menu. This does not affect the way the tree is built. It is only used to set the value of the inActiveTrail property for each tree element.

In the code below, I grab the active trail for an item that is in a menu. Then, I grab all the links for the current node_id, pull off the first one, grab the plugin (which is menu_link_content:957297e4-38eb-4502-868a-668407c71a44 – the id from the menu_tree table), and get the parameters (all the juicy goodness about this menu item). You can find a nice trail back up the menu chain along the active_trail. See activeTrail in the debug variable dump below.

```php
// Get current item's menu
/** @var \Drupal\Core\Menu\MenuLinkManagerInterface $menu_link_manager */
$menu_link_manager = \Drupal::service('plugin.manager.menu.link');
$links = $menu_link_manager->loadLinksByRoute('entity.node.canonical', ['node' => $node_id]);
/** @var \Drupal\Core\Menu\MenuLinkContent $currentMenuItem */
$currentMenuItem = reset($links);

// grab plugin for the current item
$pluginId = $currentMenuItem->getPluginId();


/** @var \Drupal\Core\Menu\MenuLinkTreeInterface $menu_tree */
$menu_tree = \Drupal::service('menu.link_tree');
$parameters = $menu_tree->getCurrentRouteMenuTreeParameters($menu_name);

// grab the active trail.
$active_trail = array_keys($parameters->activeTrail);
```


Extracting out the active trail gives this useful information:

```
$active_trail = {array} [6]
    0 = "menu_link_content:957297e4-38eb-4502-868a-668407c71a44"
    1 = "menu_link_content:8e5abbf7-0e94-42be-993f-2d1b3d538071"
    2 = "menu_link_content:ea010649-7a2a-40c3-8673-687a5fc04878"
    3 = "menu_link_content:ba000658-e485-41b5-b2eb-eaa10cf1f713"
    4 = "menu_link_content:d1b2acfb-7438-4be8-8a61-63a3bc090d03"
    5 = ""
```

Get a node's menu item and more

Here we get the current route's menu item using its nid then pull the link from the array using reset, and we can extract the URL as well as other exciting things. Mostly we want to check its children, parents etc. In the code below, we grab its URL as well as its parent and its title.

```
/** @var \Drupal\Core\Menu\MenuLinkManagerInterface $menu_link_manager */
$menu_link_manager = \Drupal::service('plugin.manager.menu.link');
$links = $menu_link_manager->loadLinksByRoute('entity.node.canonical', ['node' => $nid]);
$link = reset($links);
$urlobject = $link->getUrlObject();
$url_string = $urlobject->toString();
$x = $link->getParent(); //get the parent menu item GUID.
$y = $link->getTitle(); // get the title of the menu item.

$menu_name = $link->getMenuName(); // get the menu name e.g. "main"
```

Create menu items in your custom module

When creating a menu for your module, you need a YAML file like this from dev1 pageexample. The names (e.g. pageexample.description and pageexample.simple) are arbitrary, the title is the menu text, and the route_name comes from the routing yml file (web/modules/custom/pageexample/pageexample.routing.yml). The parent value is interesting.

```
pageexample_description:
  title: 'Page Example'
  route_name: pageexample_description
  parent: system.admin_reports
pageexample.simple:
  title: 'Simple page example - no arguments'
  route_name: pageexample_simple
  parent: system.admin_reports
```

The parent values are defined in other *.links.menu.yml files and especially the Structure link, which is defined in the core System module in system.links.menu.yml. Here we are adding a link to appear under the Reports menu of Drupal.

To create menu links programmatically, see https://drupal.stackexchange.com/questions/197073/how-do-i-create-menu-links-programmatically/197076#197076

To edit menu links programmatically, see https://drupal.stackexchange.com/questions/235516/how-do-i-programmatically-update-or-delete-menu-items

First you will have to load the entity. Either way works:

```
$menu_link = MenuLinkContent::load($menu_link_id);
```

or ...

```
$menu_link = \Drupal::entityTypeManager()->getStorage('menu_link_content')->load($menu_link_id);
```

Next you can update value using set() method or through the magic method __set...

```
$menu_link->expanded = TRUE;
```

To save, simply call the save() method. To delete, call the delete() method.

Resources

- To create menu links programmatically see https://drupal.stackexchange.com/questions/197073/how-do-i-create-menu-links-programmatically/197076#197076

- To edit menu links programmatically, see https://drupal.stackexchange.com/questions/235516/how-do-i-programmatically-update-or-delete-menu-items

- Active Trail
  https://api.drupal.org/api/drupal/core%21lib%21Drupal%21Core%21Menu%21MenuTreeParameters.php/property/MenuTreeParameters%3A%3AactiveTrail/9.3.x

- The Twig_Tweak module can do some great menu magic. Cheat sheet at https://www.drupal.org/docs/8/modules/twig-tweak/cheat-sheet

- #! code: Drupal 9: Creating A Category Menu Using Derivers, August 2022

Back to top

Page last modified: Apr 13 2023.

Edit this page on GitHub

This site uses Just the Docs, a documentation theme for Jekyll.