Drupal 8: Building Drupal 8 CMS Modules and Adding Hooks.

Drupal Modules now use YML
YML uses spaces for Indentation

SPACES VS TABS

YML uses spaces, period.

Do not use tabs in your YML files! If strange errors are coming up in rendering YML files, make sure to check that no tabs have crept in!

In Vim, after enabling search highlighting with: :set hlsearch, you can check with the following key sequence in normal mode(you can hit ESC twice to be sure): /, Ctrl-v, Tab, then hit Enter.

Also, you can convert tabs to 2 spaces by these commands in Vim: :set tabstop=2 expandtab and then :retab.

YML or YAML Indentation

Why does YML forbid tabs?

Tabs have been outlawed since they are treated differently by different editors and tools. And since indentation is so critical to proper interpretation of YAML, this issue is just too tricky to even attempt. Indeed Guido van Rossum of Python has acknowledged that allowing TABs in Python source is a headache for many people and that were he to design Python again, he would forbid them.

YML or YAML Indentation

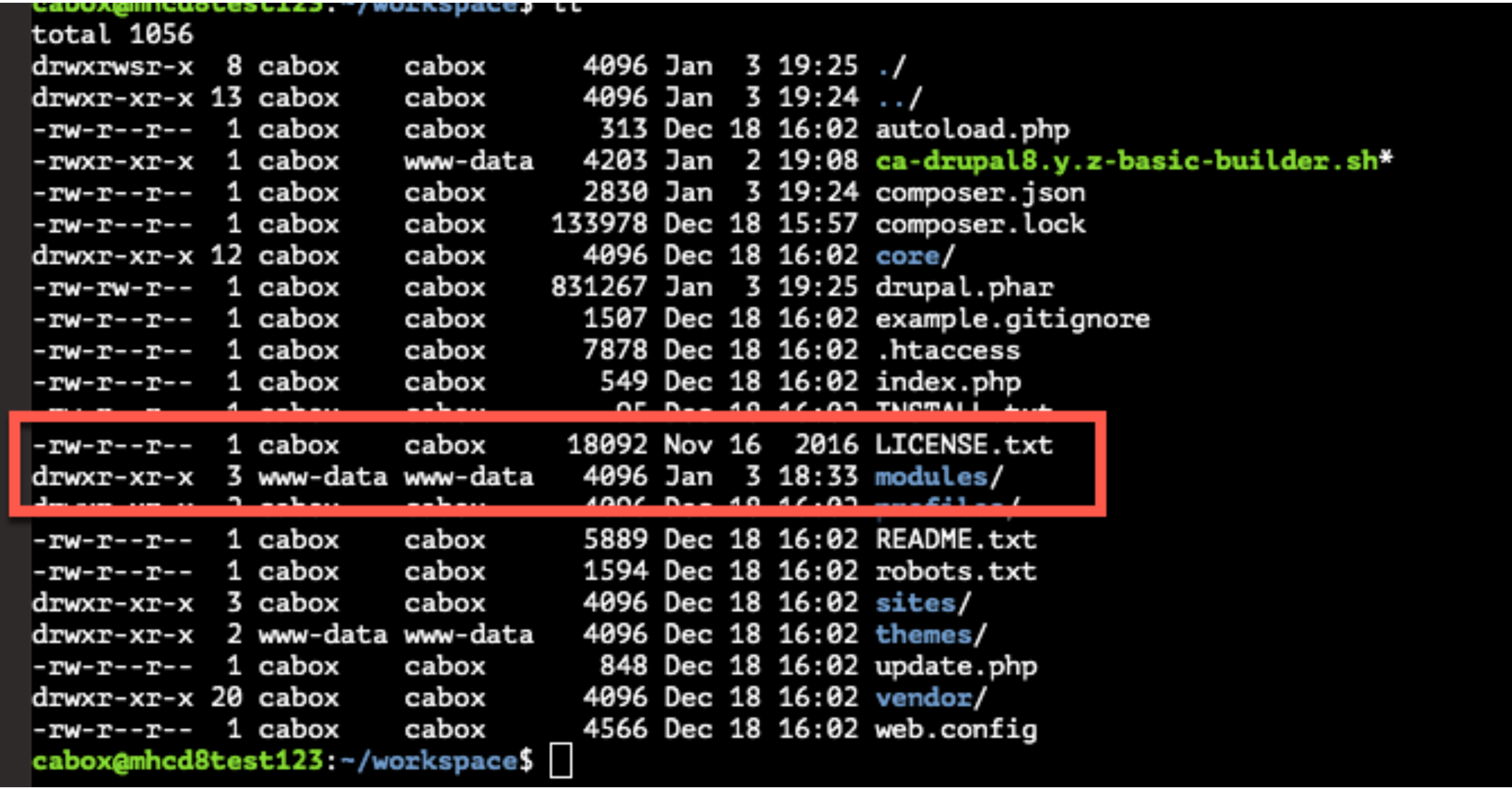
INDENTATION

The suggested syntax for YML files is to use 2 spaces for indentation, but YML will follow whatever indentation system that the individual file uses. Indentation of two spaces works very well for YML files given the fact that the data is uniform and not deeply nested.

Issues with D8 Custom Modules on Codeanywhere

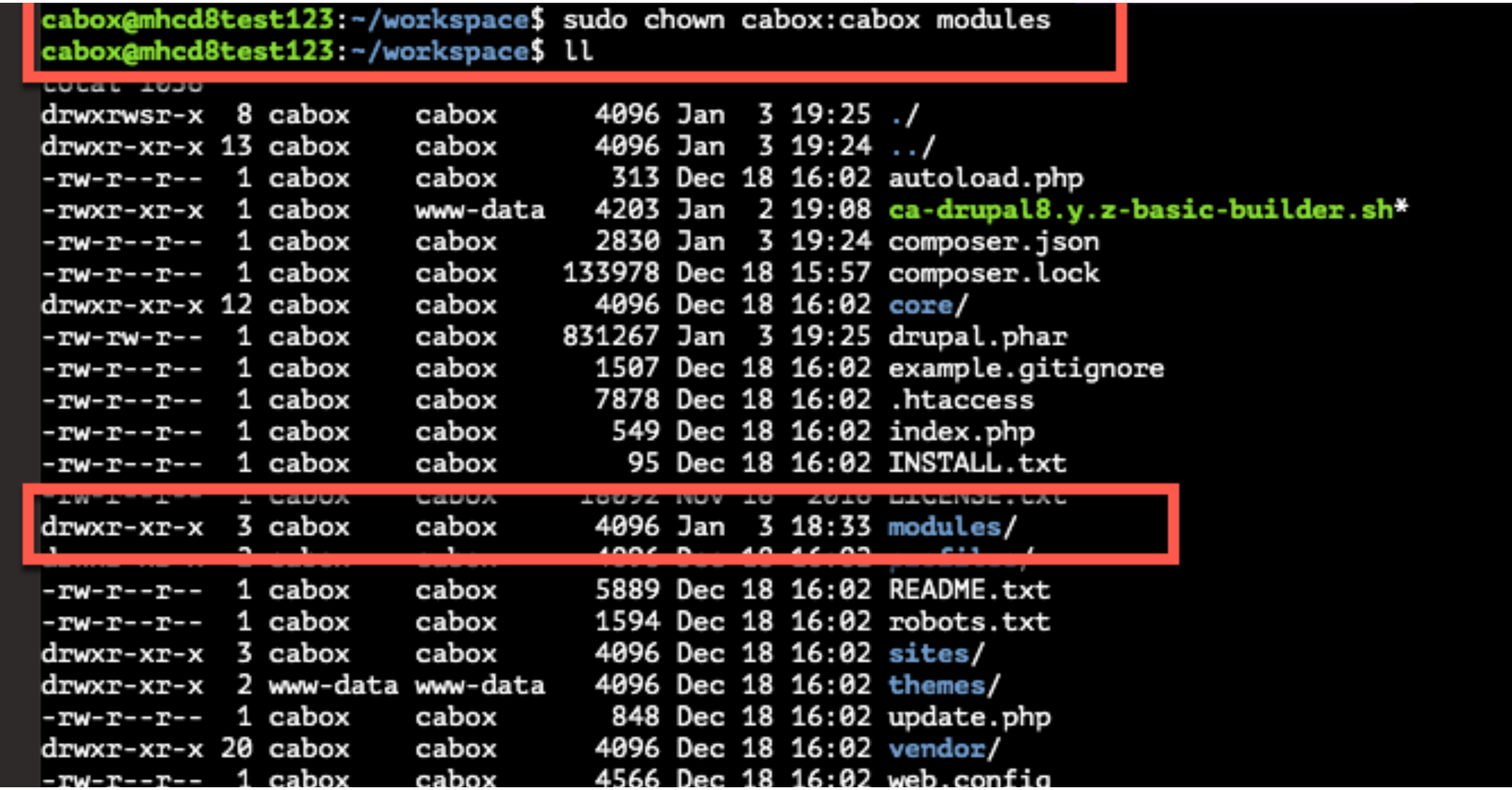
- The modules folder ownership is set to "www-data" at the end of the Codeanyhwere D8 build script.
- This change is required if we wan to install modules as a site admin within the Drupal website.
- However, when we do this, we can no longer create folders or files form the Codeanywere IDE, as the use is cabox, not www-data.
  So in order to build the D8 custom module we have to temporarily set the modules folder ownership back to cabox:cabox
- And remember to set it back to "www-data:www-data" when we are done configuring the D8 custom module

Set folder ownership of the modules folder to cabox:cabox



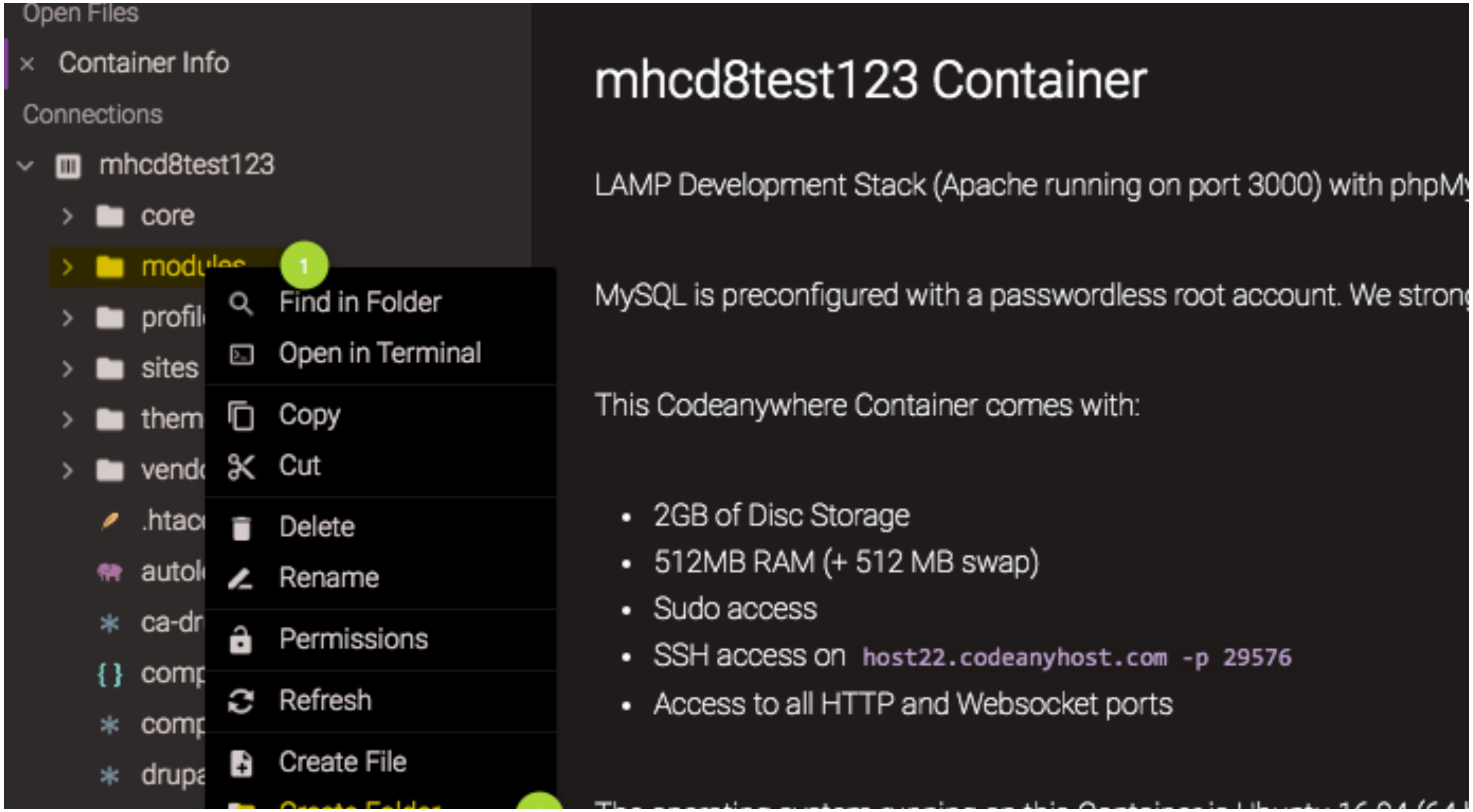
```
cabox@mhcd8test123:~/workspace$ ll
total 1056
drwxrwsr-x  8 cabox    cabox      4096 Jan  3 19:25 ./
drwxr-xr-x 13 cabox    cabox      4096 Jan  3 19:24 ../
-rw-r--r--  1 cabox    cabox       313 Dec 18 16:02 autoload.php
-rwxr-xr-x  1 cabox    www-data   4203 Jan  2 19:08 ca-drupal8.y.z-basic-builder.sh*
-rw-r--r--  1 cabox    cabox      2830 Jan  3 19:24 composer.json
-rw-r--r--  1 cabox    cabox    133978 Dec 18 15:57 composer.lock
drwxr-xr-x 12 cabox    cabox      4096 Dec 18 16:02 core/
-rw-rw-r--  1 cabox    cabox    831267 Jan  3 19:25 drupal.phar
-rw-r--r--  1 cabox    cabox      1507 Dec 18 16:02 example.gitignore
-rw-r--r--  1 cabox    cabox      7878 Dec 18 16:02 .htaccess
-rw-r--r--  1 cabox    cabox       549 Dec 18 16:02 index.php
-rw-r--r--  1 cabox    cabox        95 Dec 18 16:02 INSTALL.txt
-rw-r--r--  1 cabox    cabox     18092 Nov 16  2016 LICENSE.txt
drwxr-xr-x  3 www-data www-data   4096 Jan  3 18:33 modules/
drwxr-xr-x  2 cabox    cabox      4096 Dec 18 16:02 profiles/
-rw-r--r--  1 cabox    cabox      5889 Dec 18 16:02 README.txt
-rw-r--r--  1 cabox    cabox      1594 Dec 18 16:02 robots.txt
drwxr-xr-x  3 cabox    cabox      4096 Dec 18 16:02 sites/
drwxr-xr-x  2 www-data www-data   4096 Dec 18 16:02 themes/
-rw-r--r--  1 cabox    cabox       848 Dec 18 16:02 update.php
drwxr-xr-x 20 cabox    cabox      4096 Dec 18 16:02 vendor/
-rw-r--r--  1 cabox    cabox      4566 Dec 18 16:02 web.config
cabox@mhcd8test123:~/workspace$ 
```

Command
sudo chown cabox:cabox modules



```
cabox@mhcd8test123:~/workspace$ sudo chown cabox:cabox modules
cabox@mhcd8test123:~/workspace$ ll
total 1056
drwxrwsr-x  8 cabox    cabox      4096 Jan  3 19:25 ./
drwxr-xr-x 13 cabox    cabox      4096 Jan  3 19:24 ../
-rw-r--r--  1 cabox    cabox       313 Dec 18 16:02 autoload.php
-rwxr-xr-x  1 cabox    www-data   4203 Jan  2 19:08 ca-drupal8.y.z-basic-builder.sh*
-rw-r--r--  1 cabox    cabox      2830 Jan  3 19:24 composer.json
-rw-r--r--  1 cabox    cabox    133978 Dec 18 15:57 composer.lock
drwxr-xr-x 12 cabox    cabox      4096 Dec 18 16:02 core/
-rw-rw-r--  1 cabox    cabox    831267 Jan  3 19:25 drupal.phar
-rw-r--r--  1 cabox    cabox      1507 Dec 18 16:02 example.gitignore
-rw-r--r--  1 cabox    cabox      7878 Dec 18 16:02 .htaccess
-rw-r--r--  1 cabox    cabox       549 Dec 18 16:02 index.php
-rw-r--r--  1 cabox    cabox        95 Dec 18 16:02 INSTALL.txt
-rw-r--r--  1 cabox    cabox     18092 Nov 16  2016 LICENSE.txt
drwxr-xr-x  3 cabox    cabox      4096 Jan  3 18:33 modules/
drwxrwxr-x  2 cabox    cabox      4096 Dec 18 16:02 profiles/
-rw-r--r--  1 cabox    cabox      5889 Dec 18 16:02 README.txt
-rw-r--r--  1 cabox    cabox      1594 Dec 18 16:02 robots.txt
drwxr-xr-x  3 cabox    cabox      4096 Dec 18 16:02 sites/
drwxr-xr-x  2 www-data www-data   4096 Dec 18 16:02 themes/
-rw-r--r--  1 cabox    cabox       848 Dec 18 16:02 update.php
drwxr-xr-x 20 cabox    cabox      4096 Dec 18 16:02 vendor/
-rw-r--r--  1 cabox    cabox      4566 Dec 18 16:02 web.config
```

Create the folder named hello_world under the modules folder



mhcd8test123 Container

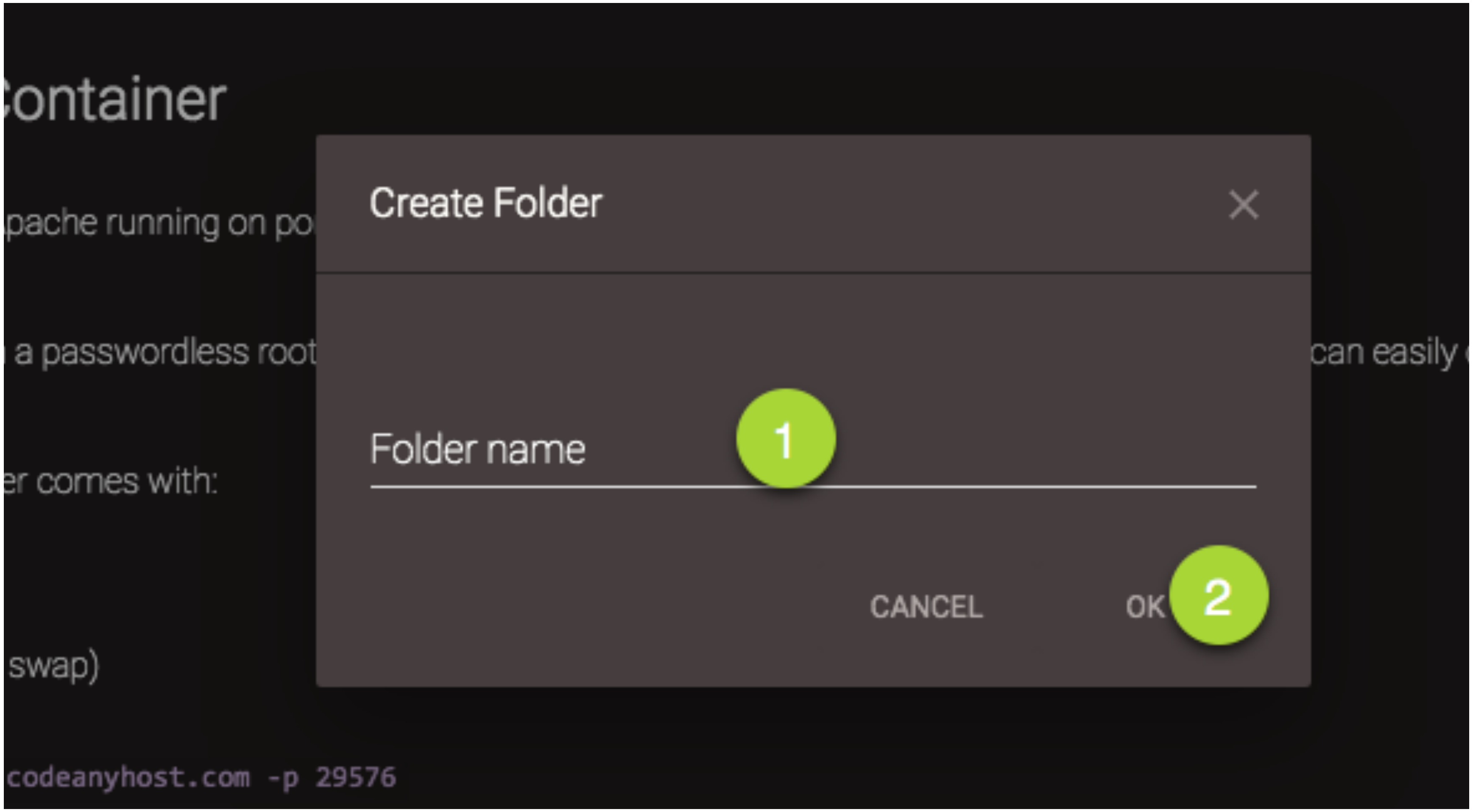
LAMP Development Stack (Apache running on port 3000) with phpMy

MySQL is preconfigured with a passwordless root account. We strong

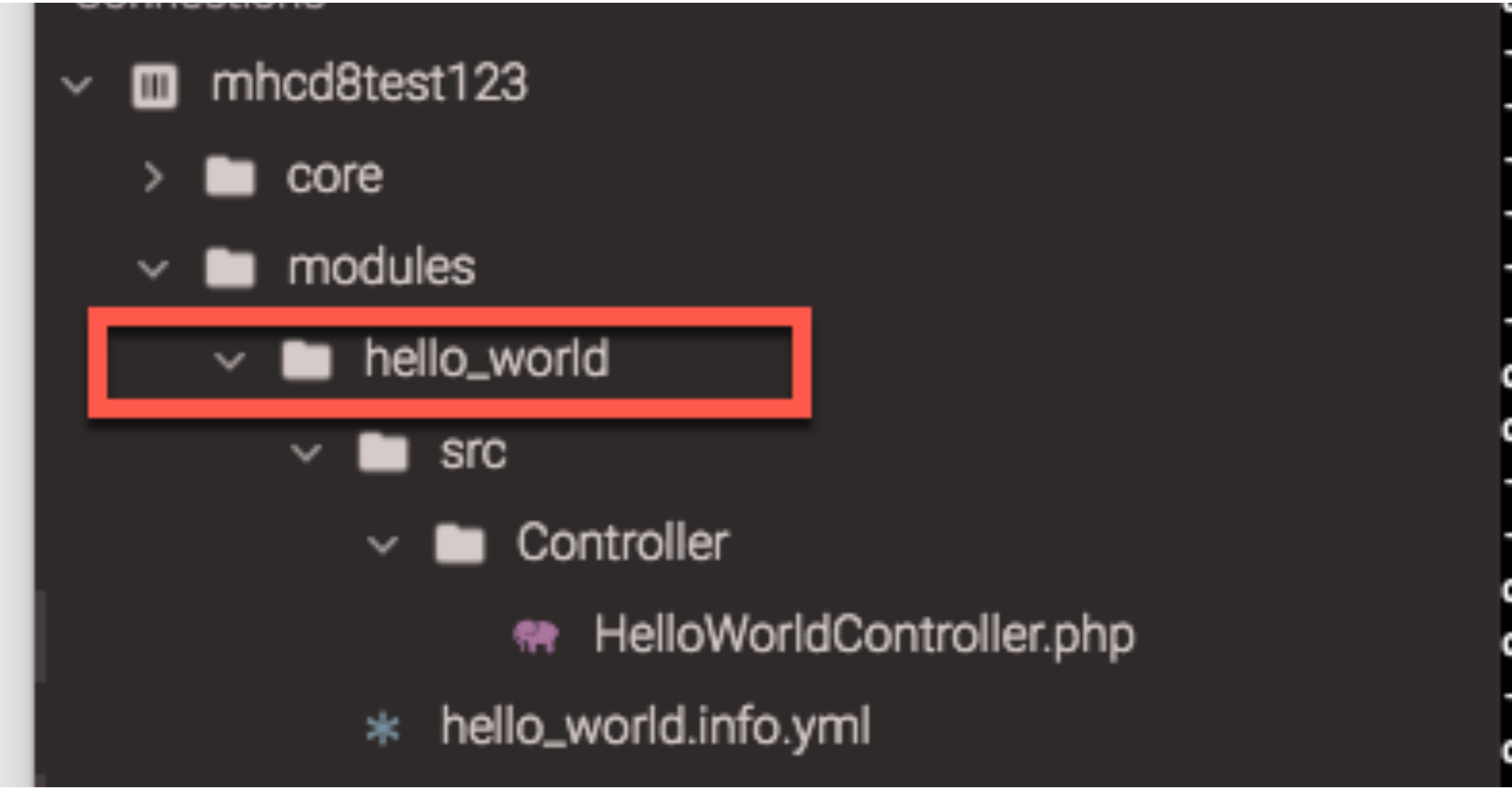
This Codeanywhere Container comes with:

- 2GB of Disc Storage
- 512MB RAM (+ 512 MB swap)
- Sudo access
- SSH access on `host22.codeanyhost.com -p 29576`
- Access to all HTTP and Websocket ports

Enter folder name and press OK



Create folder hello_world under the modules folder

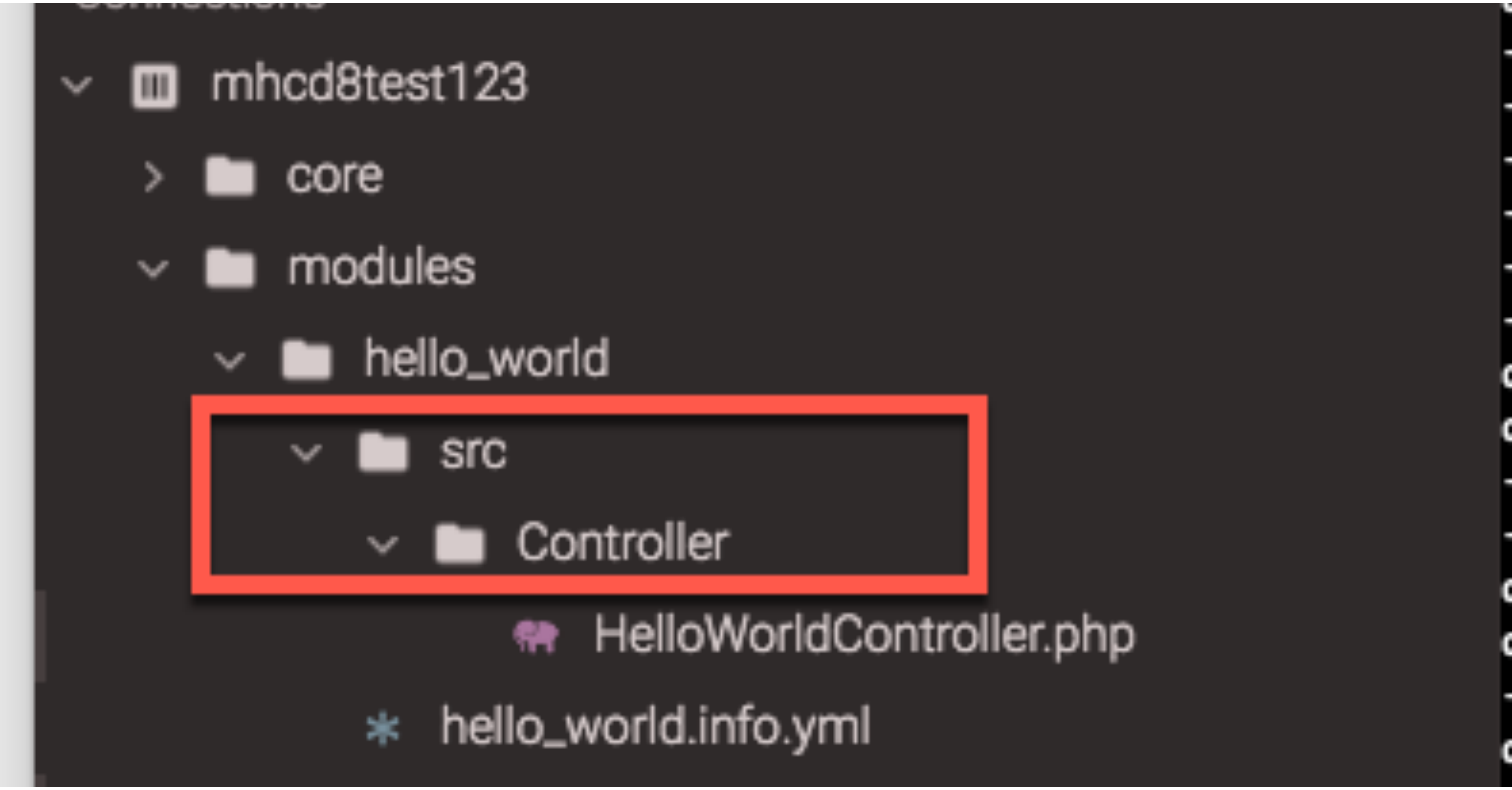


```
mhcd8test123
  core
  modules
    hello_world
      src
        Controller
          HelloWorldController.php
      hello_world.info.yml
```

The module sub-folder name serves as the machine name for the module

- Out folder is named hello_world so our Drupal 8 custom module machine name (a unique name) is hello_world

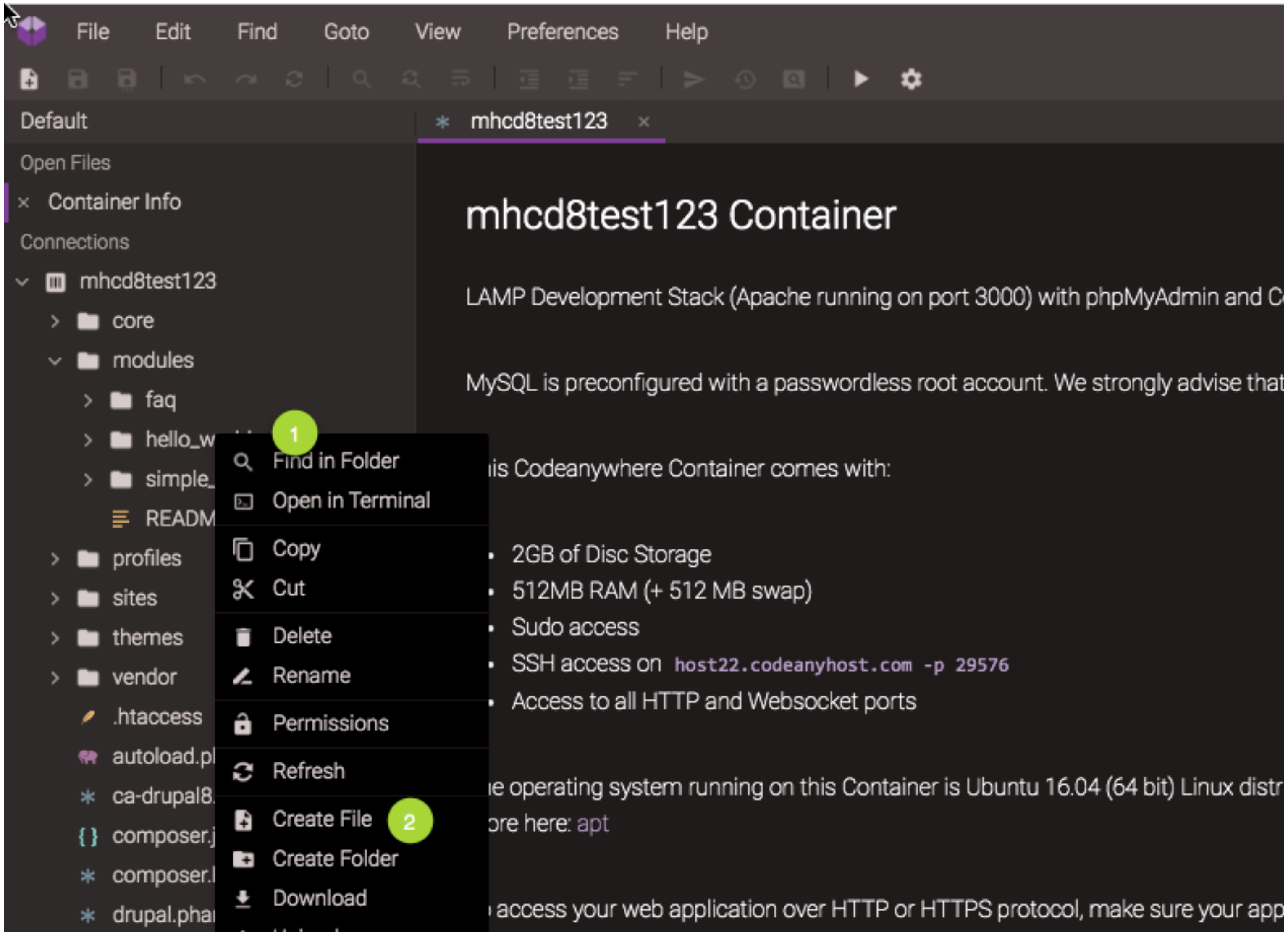
Create src & Controller folders



Create empty files or copy up existing

- In the Tutorial I copy up existing custom module files to the new folders.
- You can also copy the existing files to the folder or create your own versions of the YML and Controller files
- When creating your own custom modules, you can create the files on your computer and upload them to the Drupal file system as long as they are created on an ASCII editor, like Brackets and not on Microsoft Word or NotePad.

Create empty files or copy up existing



mhcd8test123 Container

LAMP Development Stack (Apache running on port 3000) with phpMyAdmin and C

MySQL is preconfigured with a passwordless root account. We strongly advise that

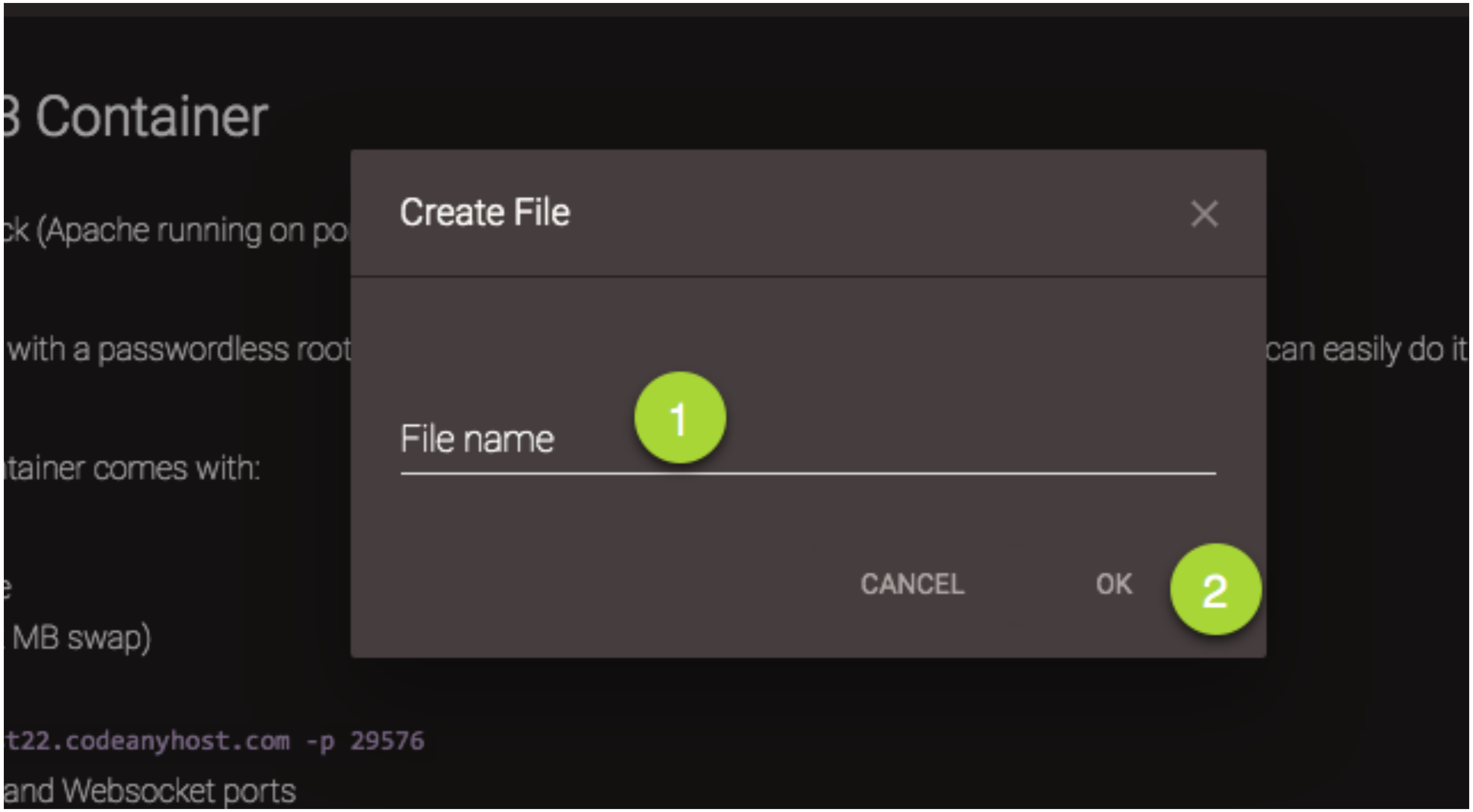
is Codeanywhere Container comes with:

- 2GB of Disc Storage
- 512MB RAM (+ 512 MB swap)
- Sudo access
- SSH access on `host22.codeanyhost.com -p 29576`
- Access to all HTTP and Websocket ports

e operating system running on this Container is Ubuntu 16.04 (64 bit) Linux distr
ore here: apt

access your web application over HTTP or HTTPS protocol, make sure your app

Enter file name + extension and press OK



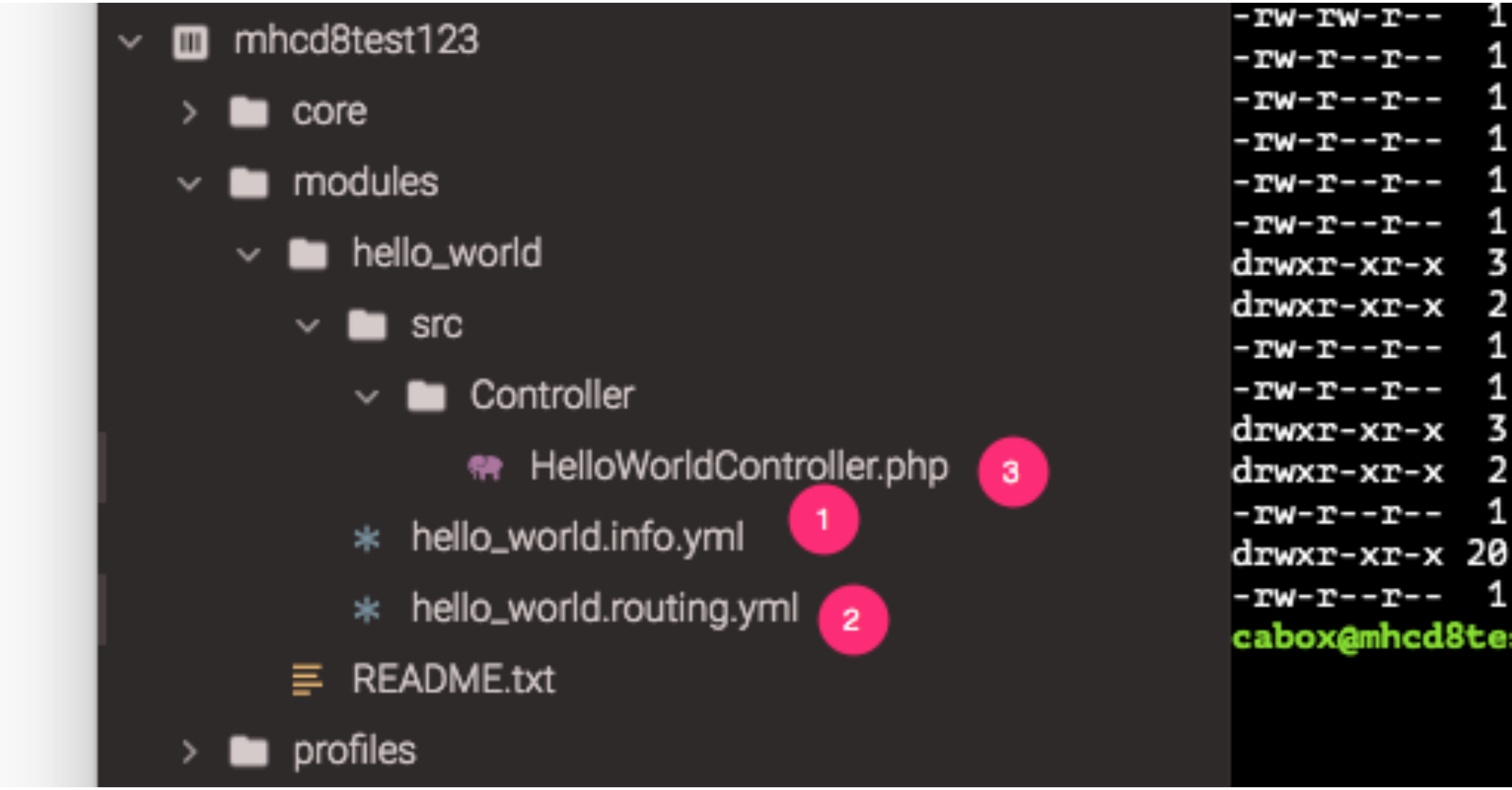
```
t22.codeanyhost.com -p 29576
```

Create the following empty files

- Create YML files in the hello_world folder
  - hello_world.info.yml ← To define our site parameters
  - hello_world.routing.yml

- Create a PHP file in the hello_world**\src\Controller** folder
  - HelloWorldController.php

Create the files in the folders as follows



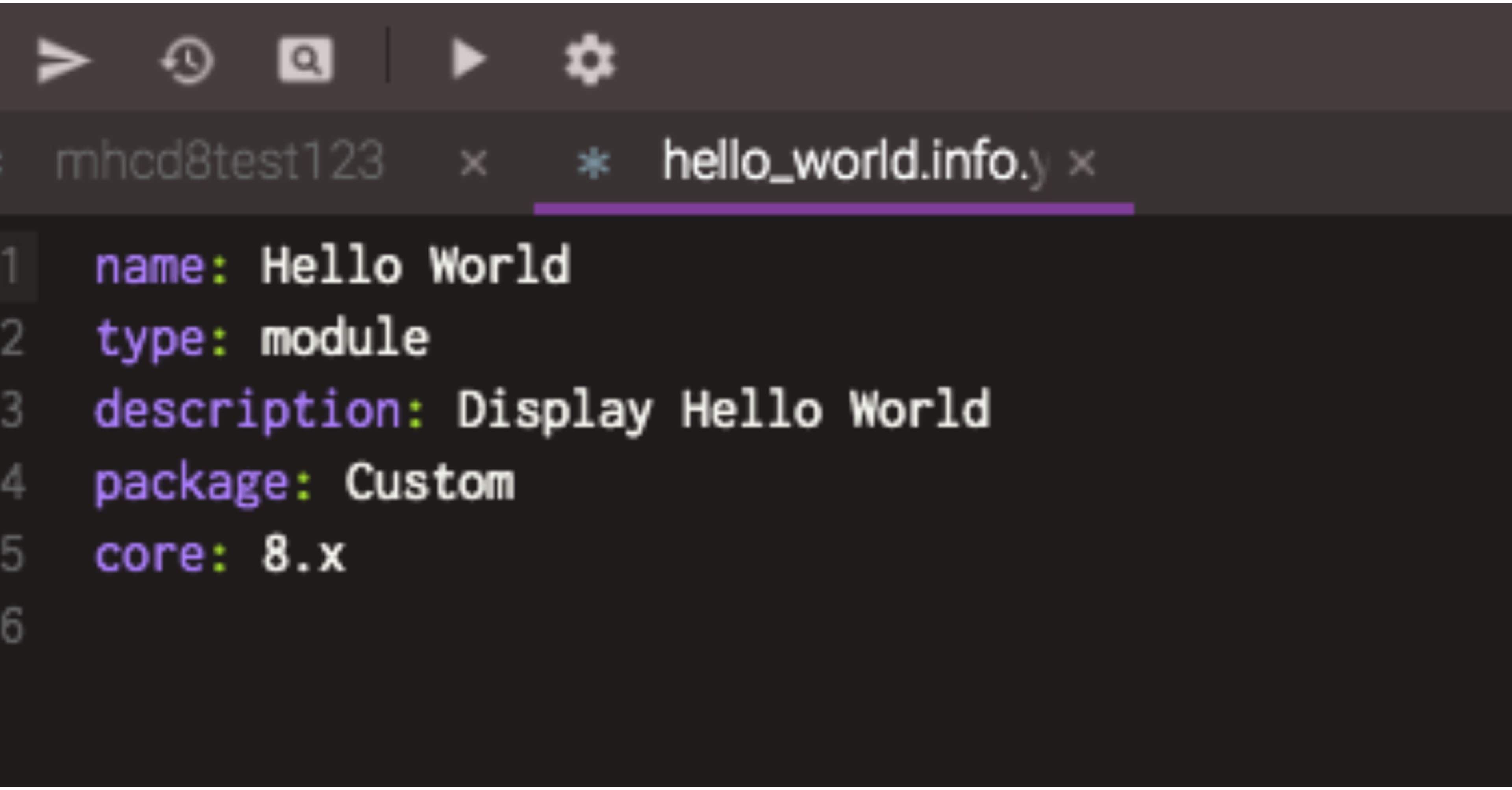
```
mhcd8test123
  core
  modules
    hello_world
      src
        Controller
          HelloWorldController.php   3
      hello_world.info.yml   1
      hello_world.routing.yml   2
    README.txt
  profiles
```

```
-rw-rw-r--  1
-rw-r--r--  1
-rw-r--r--  1
-rw-r--r--  1
-rw-r--r--  1
-rw-r--r--  1
drwxr-xr-x  3
drwxr-xr-x  2
-rw-r--r--  1
-rw-r--r--  1
drwxr-xr-x  3
drwxr-xr-x  2
-rw-r--r--  1
drwxr-xr-x 20
-rw-r--r--  1
cabox@mhcd8te
```

.info.yml [INSTALL PROFILE] contains parameters

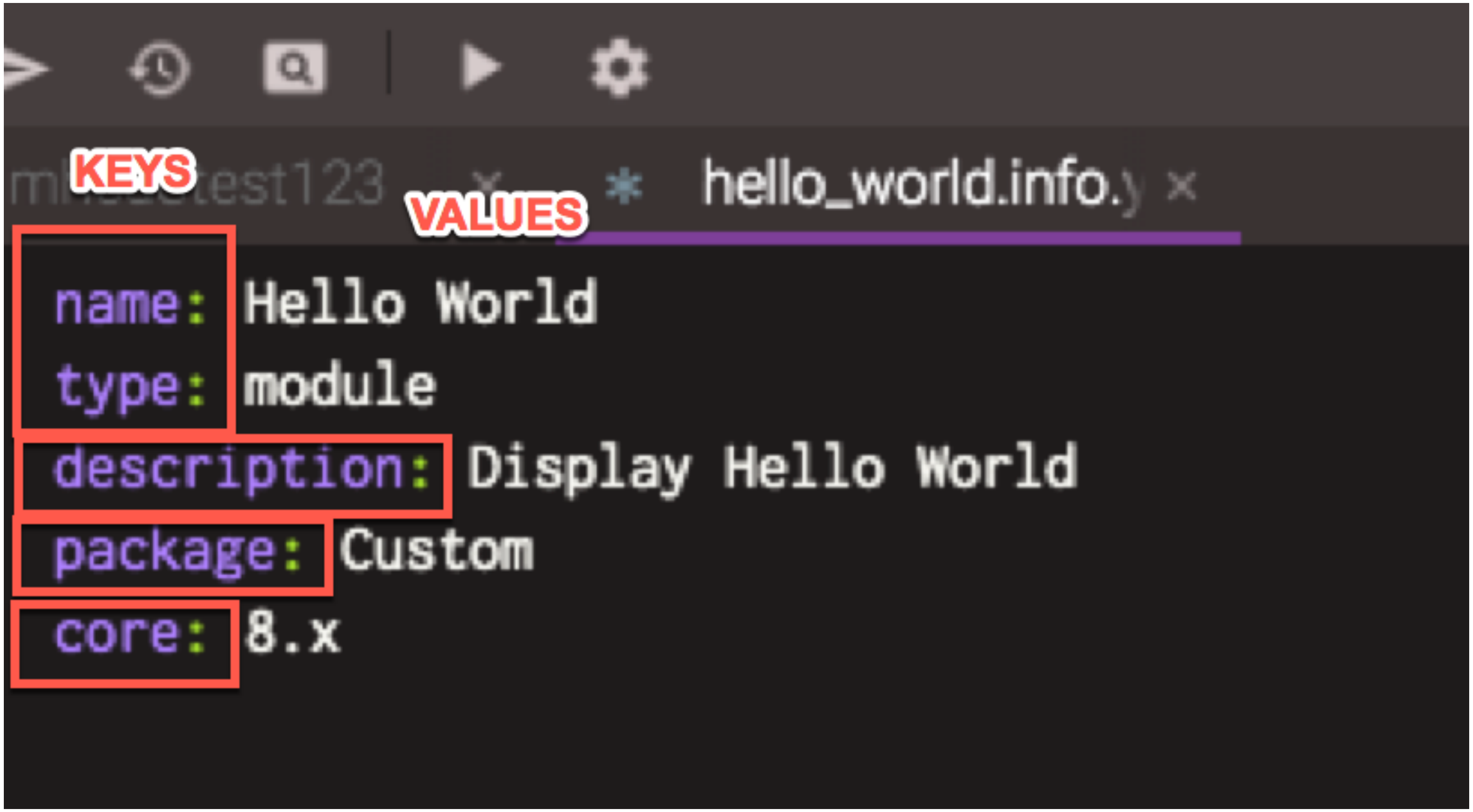
- An .info.yml file (aka. "info yaml file") is an essential part of a Drupal 8 module, theme, or install profile to store metadata about the project.
- These .info.yml files are required to:
  - Notify Drupal about the existence of a module, theme, or install profile.
  - Differ theme, module by type.
  - Provide information for the Drupal Web UI administration pages.
  - Provide criteria to control module activation and deactivation and Drupal version compatibility.
  - General administrative purposes in other contexts.

Create hello_world.info.yml



```
name: Hello World
type: module
description: Display Hello World
package: Custom
core: 8.x
```

Create hello_world.info.yml

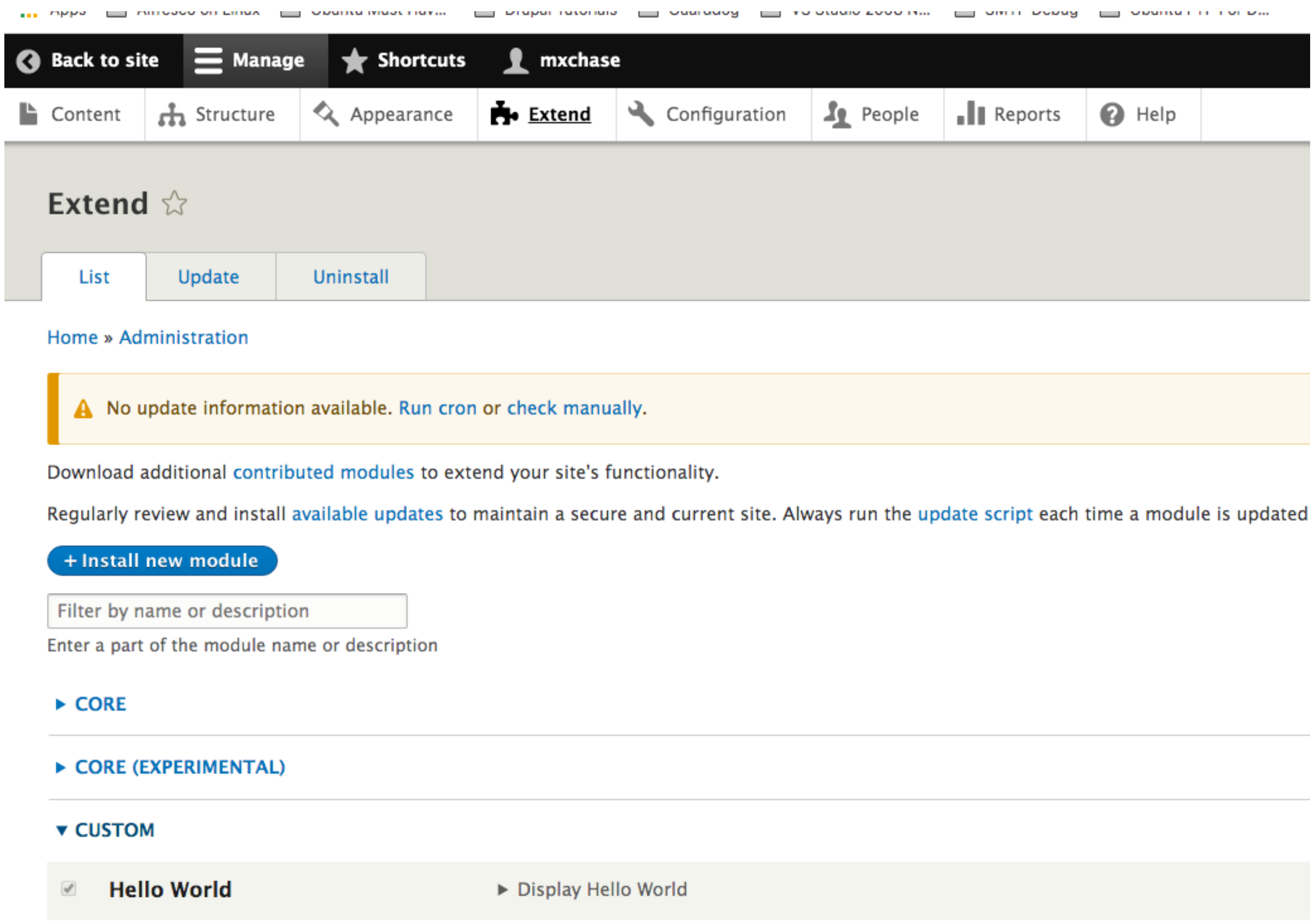


```
name: Hello World
type: module
description: Display Hello World
package: Custom
core: 8.x
```

Info.yml decoded
name, type and core are required keys.

- The first three lines are primarily used in the administration UI when allowing users to enable or disable your module.
  - The name and description keys provide the text that is shown on the module administration page and the package key allows you to group like modules together.
  - Core, for example, uses package: Core to group all of the modules provided with Drupal 8 together, likewise you might use package: Custom to group all of your projects custom modules together making them easier to locate and enable.
- The type key, which is new in Drupal 8, indicates the type of extension, e.g. module, theme, or profile.
- The core key specifies with which Drupal core version your module is compatible.

Module should now be visible on the Extends page



Extend ☆

List | Update | Uninstall

Home » Administration

No update information available. Run cron or check manually.

Download additional contributed modules to extend your site's functionality.

Regularly review and install available updates to maintain a secure and current site. Always run the update script each time a module is updated

+ Install new module

Filter by name or description

Enter a part of the module name or description

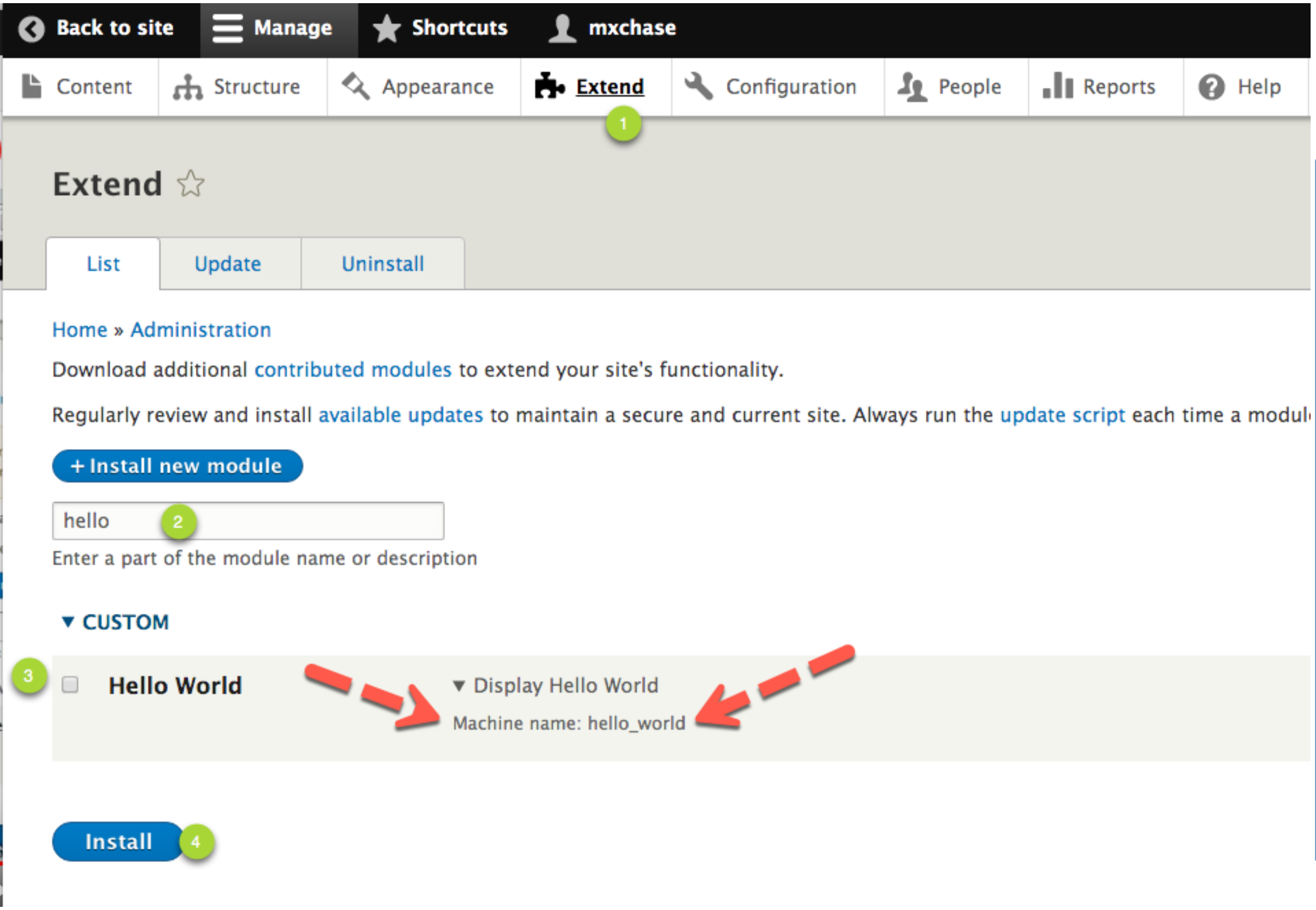
▸ CORE

▸ CORE (EXPERIMENTAL)

▾ CUSTOM

✓ Hello World ▸ Display Hello World

Find an Enable the "Hello World" module with machine name hello_world on the Extends page



Home » Administration

Download additional contributed modules to extend your site's functionality.

Regularly review and install available updates to maintain a secure and current site. Always run the update script each time a modul

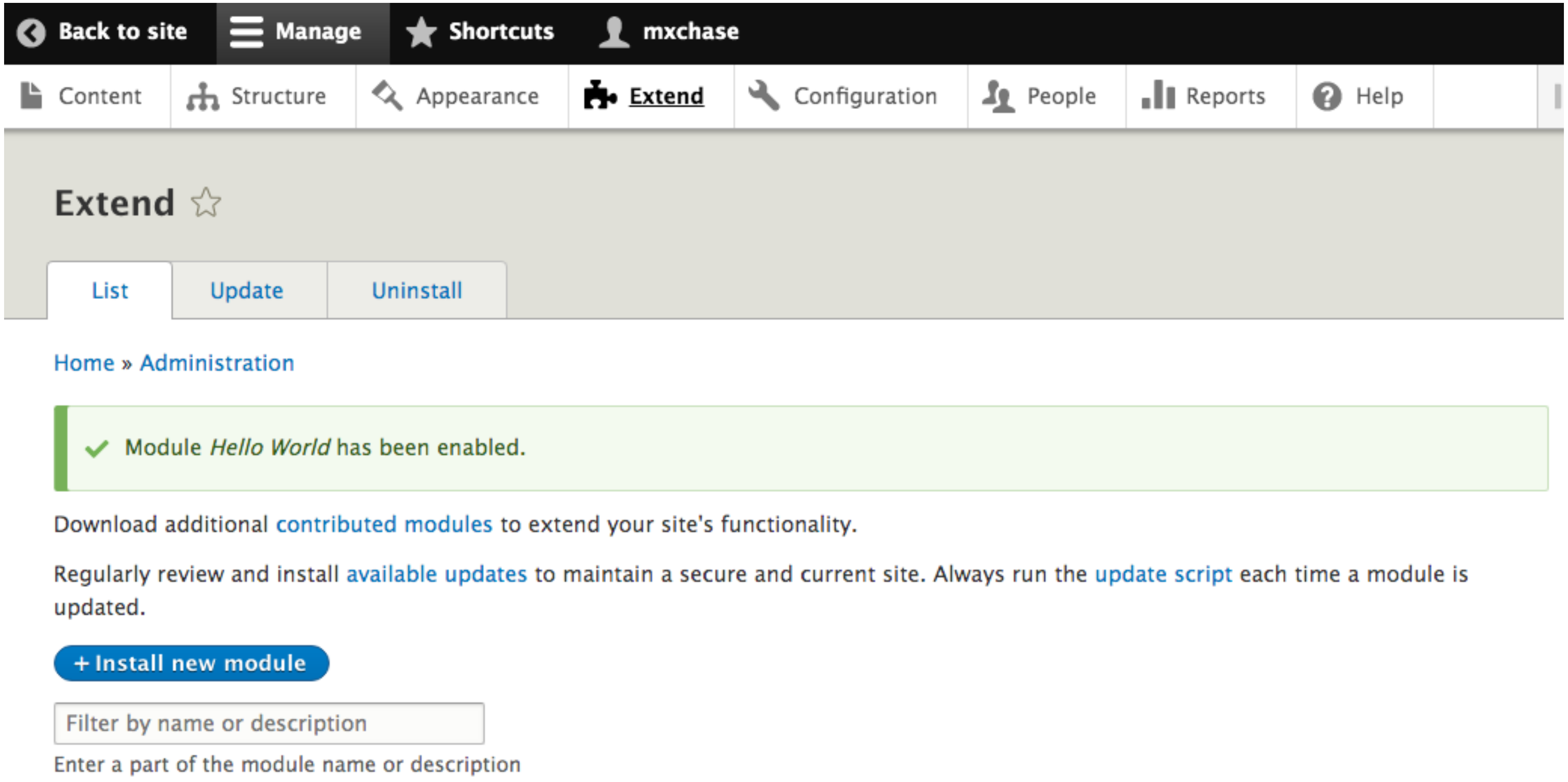
Enter a part of the module name or description

▼ Display Hello World

Machine name: hello_world

1. Select Admin toolbar Extend
2. Enter hello in the search
3. Select (check) Hello World module
4. Select (check) the Install button

We have successfully enabled the "Hello World" module



Extend ☆

List | Update | Uninstall

Home » Administration

✓ Module *Hello World* has been enabled.

Download additional contributed modules to extend your site's functionality.

Regularly review and install available updates to maintain a secure and current site. Always run the update script each time a module is updated.

+ Install new module

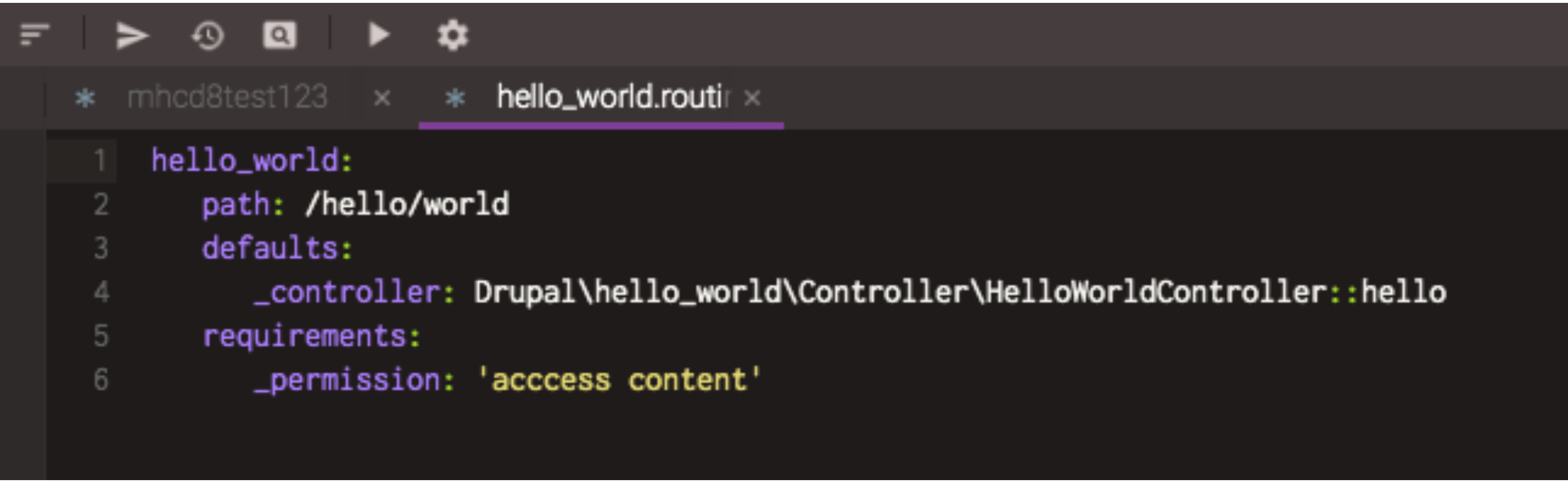
Filter by name or description

Enter a part of the module name or description

Next steps – Define Routs

- We need to define the routs that will activate our module
- To do this we will create a new file named hello_world.routing.yml
- The routing file starts with our machine name of hello_world
- The routing file has the path "/hello/world" that if accessed we set some defaults to which controllers should be accessed or called
- Next we set permissions or access content

Once the “Hello World” module is enabled Create the file hello_world.routing.yml



```
hello_world:
   path: /hello/world
   defaults:
      _controller: Drupal\hello_world\Controller\HelloWorldController::hello
   requirements:
      _permission: 'acccess content'
```

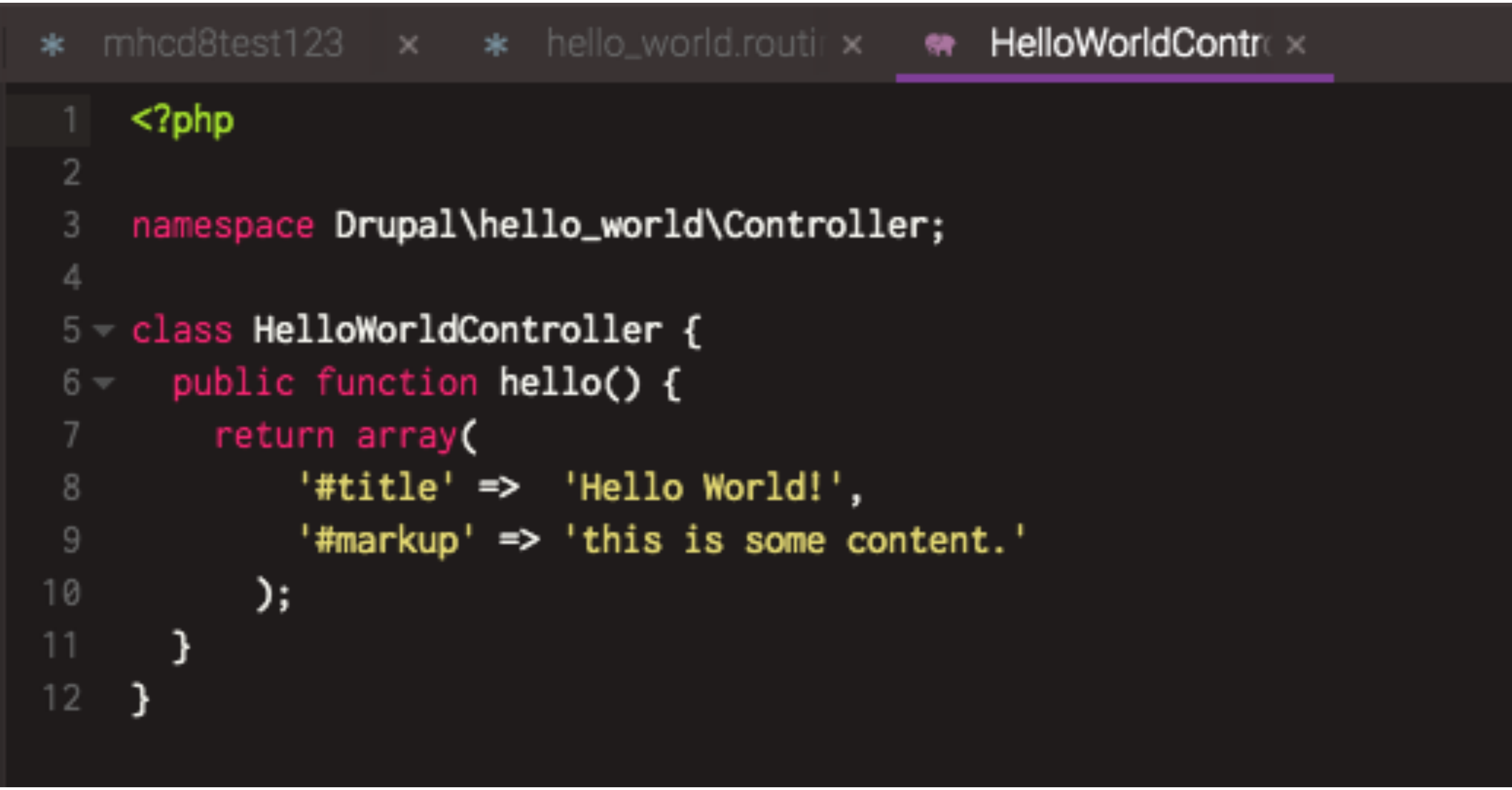
Create the Controller in the src folder

- Under the hello_world folder create a new folder named src
- src is the source folder and it is where all the code for this module resides
- User the src folder create a new sub-folder named Controller, where all the Controllers will reside (we will have only 1 Controller)
- Create a new file PHP empty file named HelloWorldController.php

The Contoller PHP File

- The Controller PHP file starts with a start PHP tag <?php
- Create the class and add a namespace, getting it form the routing file
- Create the hello method as a public method
- Return a rendered array – Drupal 8 supports returning symphony responses, that Drupal is built on top of or a rendered array.

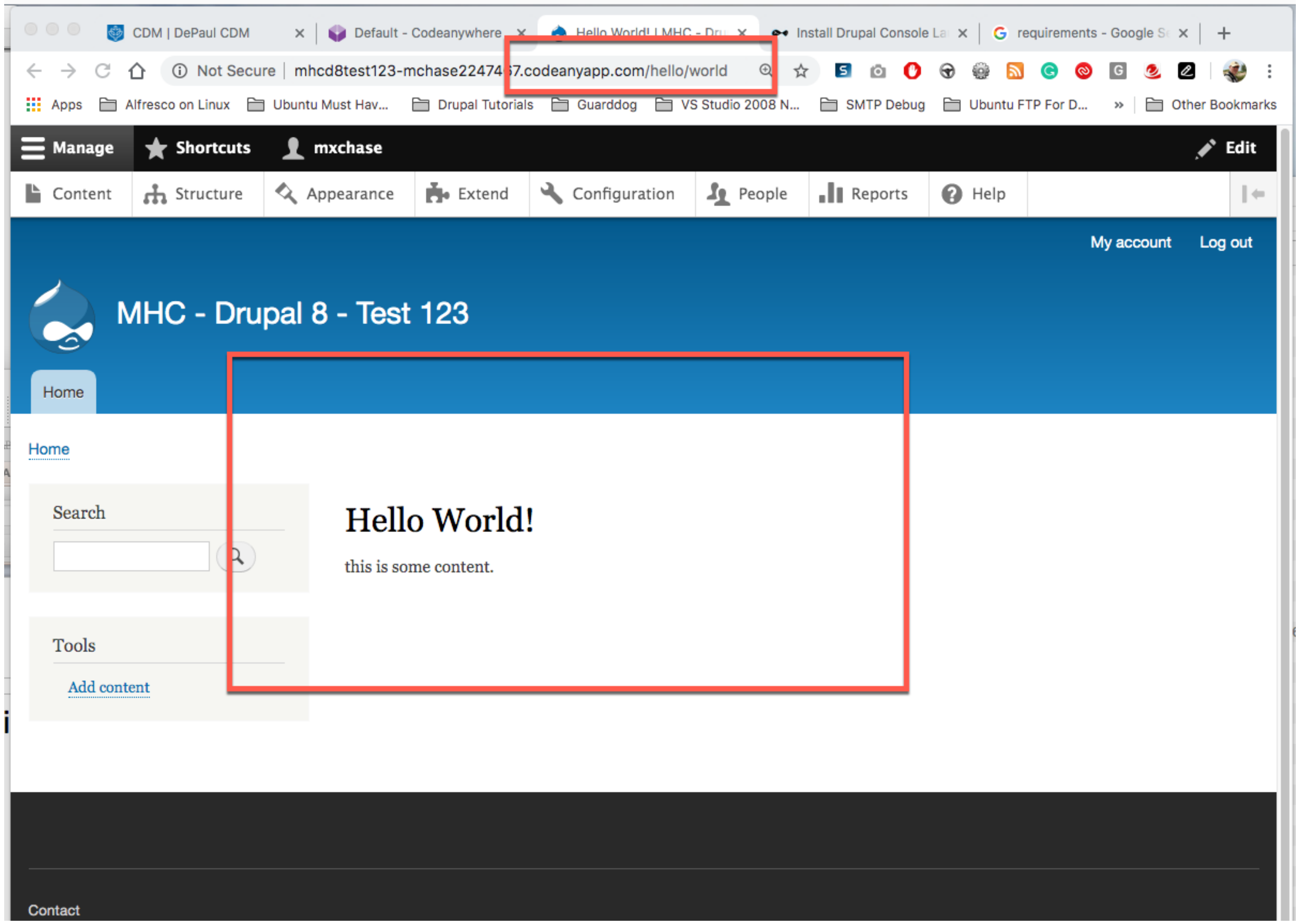
Also Create HelloWorldController.php



```php
<?php

namespace Drupal\hello_world\Controller;

class HelloWorldController {
  public function hello() {
    return array(
        '#title' => 'Hello World!',
        '#markup' => 'this is some content.'
      );
  }
}
```

Access the Hello World custom plugin

[D8 site URL]/hello/world



mhcd8test123-mchase224747.codeanyapp.com/hello/world

Hello World!

this is some content.

Drupal 8 Hooks API

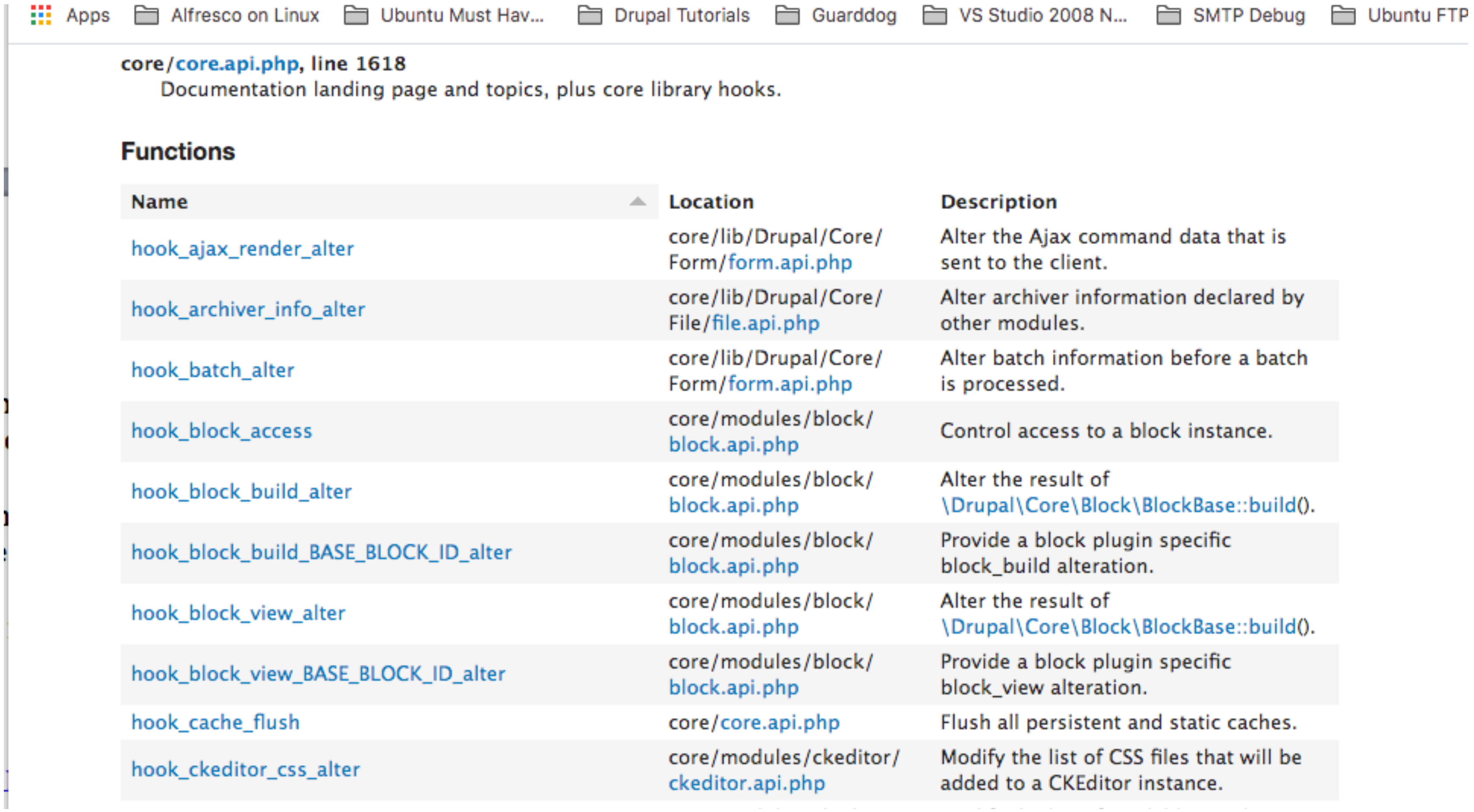
- Hooks define functions that alter the behavior of Drupal core.
- Hooks are a way modules can alter the core behavior of Drupal (or another module.
- Hooks are specially-named functions that a module defines (this is known as "implementing the hook"), which are discovered and called at specific times to alter or add to the base behavior or data (this is known as "invoking the hook").
- Each hook has a name (example: hook_batch_alter()), a defined set of parameters, and a defined return value.
- Your modules can implement hooks that are defined by Drupal core or other modules that they interact with.
- Your modules can also define their own hooks, in order to let other modules interact with them.

To implement a Drupal 8 hook

- Locate the documentation for the hook. Hooks are documented in *.api.php files, by defining functions whose name starts with "hook_" (these files and their functions are never loaded by Drupal -- they exist solely for documentation).
  - The function should have a documentation header, as well as a sample function body.
  - For example, in the core file system.api.php, you can find hooks such as hook_batch_alter().
  - Also, if you are viewing this documentation on an API reference site, the Core hooks will be listed in this topic.

Drupal 8 Hooks List

https://api.drupal.org/api/drupal/core!core.api.php/group/hooks/8.2.x



Apps · Alfresco on Linux · Ubuntu Must Hav... · Drupal Tutorials · Guarddog · VS Studio 2008 N... · SMTP Debug · Ubuntu FTP

core/core.api.php, line 1618

Documentation landing page and topics, plus core library hooks.

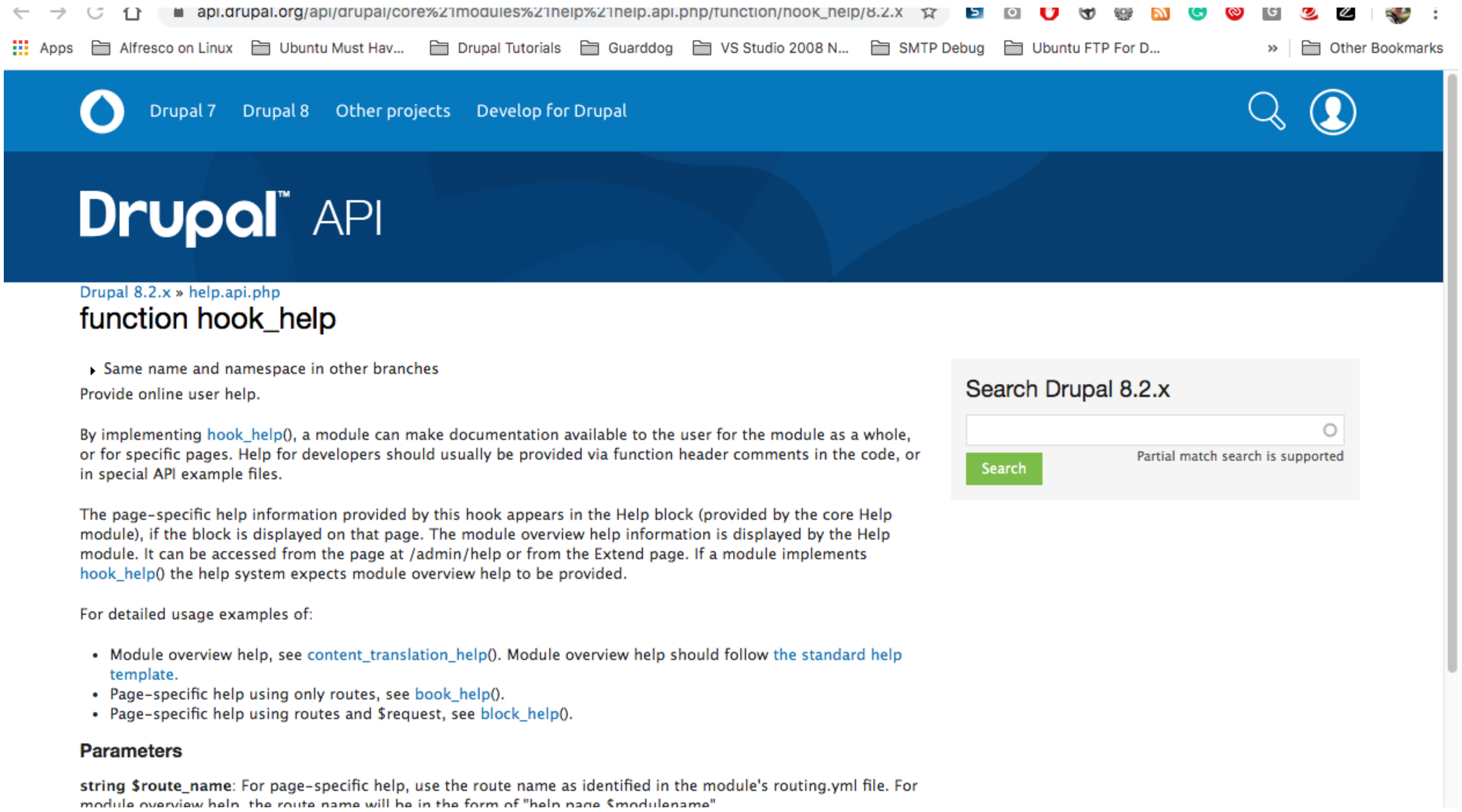
Functions

Name	Location	Description
hook_ajax_render_alter	core/lib/Drupal/Core/Form/form.api.php	Alter the Ajax command data that is sent to the client.
hook_archiver_info_alter	core/lib/Drupal/Core/File/file.api.php	Alter archiver information declared by other modules.
hook_batch_alter	core/lib/Drupal/Core/Form/form.api.php	Alter batch information before a batch is processed.
hook_block_access	core/modules/block/block.api.php	Control access to a block instance.
hook_block_build_alter	core/modules/block/block.api.php	Alter the result of \Drupal\Core\Block\BlockBase::build().
hook_block_build_BASE_BLOCK_ID_alter	core/modules/block/block.api.php	Provide a block plugin specific block_build alteration.
hook_block_view_alter	core/modules/block/block.api.php	Alter the result of \Drupal\Core\Block\BlockBase::build().
hook_block_view_BASE_BLOCK_ID_alter	core/modules/block/block.api.php	Provide a block plugin specific block_view alteration.
hook_cache_flush	core/core.api.php	Flush all persistent and static caches.
hook_ckeditor_css_alter	core/modules/ckeditor/ckeditor.api.php	Modify the list of CSS files that will be added to a CKEditor instance.

Drupal 8 Hooks List

https://api.drupal.org/api/drupal/core!core.api.php/group/hooks/8.2.x

Scroll down and find hook_help, click the link



Drupal 8.2.x » help.api.php

function hook_help

▸ Same name and namespace in other branches

Provide online user help.

By implementing hook_help(), a module can make documentation available to the user for the module as a whole, or for specific pages. Help for developers should usually be provided via function header comments in the code, or in special API example files.

The page-specific help information provided by this hook appears in the Help block (provided by the core Help module), if the block is displayed on that page. The module overview help information is displayed by the Help module. It can be accessed from the page at /admin/help or from the Extend page. If a module implements hook_help() the help system expects module overview help to be provided.

For detailed usage examples of:

- Module overview help, see content_translation_help(). Module overview help should follow the standard help template.
- Page-specific help using only routes, see book_help().
- Page-specific help using routes and $request, see block_help().

Parameters

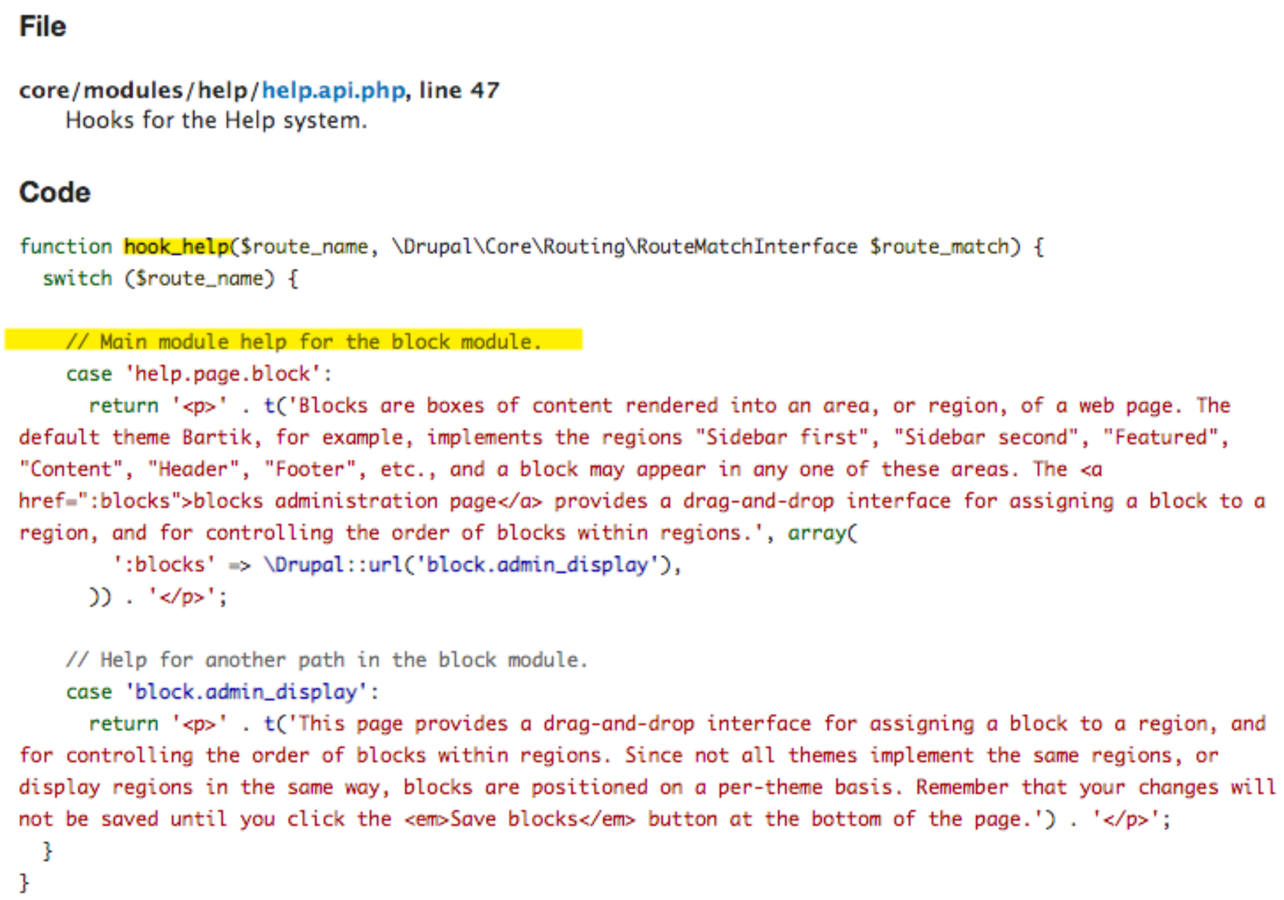
string $route_name: For page-specific help, use the route name as identified in the module's routing.yml file. For module overview help, the route name will be in the form of "help.page.$modulename"

Search Drupal 8.2.x

Search

Partial match search is supported

Copy the code example into your .module file



File

core/modules/help/help.api.php, line 47
Hooks for the Help system.

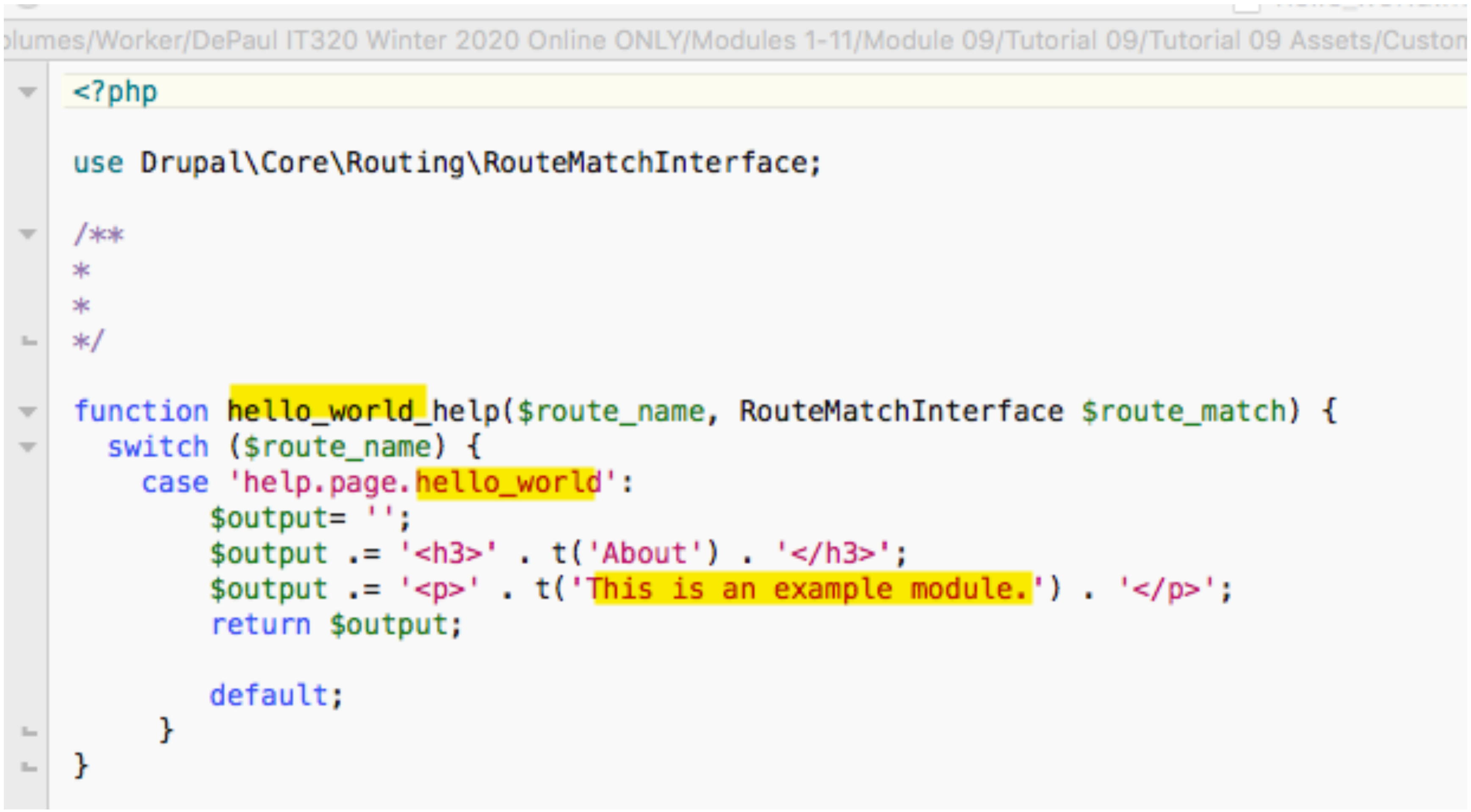
Code

```
function hook_help($route_name, \Drupal\Core\Routing\RouteMatchInterface $route_match) {
  switch ($route_name) {

    // Main module help for the block module.
    case 'help.page.block':
      return '<p>' . t('Blocks are boxes of content rendered into an area, or region, of a web page. The
default theme Bartik, for example, implements the regions "Sidebar first", "Sidebar second", "Featured",
"Content", "Header", "Footer", etc., and a block may appear in any one of these areas. The <a
href=":blocks">blocks administration page</a> provides a drag-and-drop interface for assigning a block to a
region, and for controlling the order of blocks within regions.', array(
        ':blocks' => \Drupal::url('block.admin_display'),
      )) . '</p>';

    // Help for another path in the block module.
    case 'block.admin_display':
      return '<p>' . t('This page provides a drag-and-drop interface for assigning a block to a region, and
for controlling the order of blocks within regions. Since not all themes implement the same regions, or
display regions in the same way, blocks are positioned on a per-theme basis. Remember that your changes will
not be saved until you click the <em>Save blocks</em> button at the bottom of the page.') . '</p>';
  }
}
```

To implement a hook: After finding the hook

- Copy the function to your module's .module file.
- **Change the name of the function, substituting your module's short name** (name of the module's directory, and .info.yml file without the extension) for the "hook" part of the sample function name. For instance, to implement hook_batch_alter(), you would rename it to my_module_batch_alter().
- Edit the documentation for the function (normally, your implementation should just have one line saying "Implements hook_batch_alter().").
- Edit the body of the function, substituting in what you need your module to do.

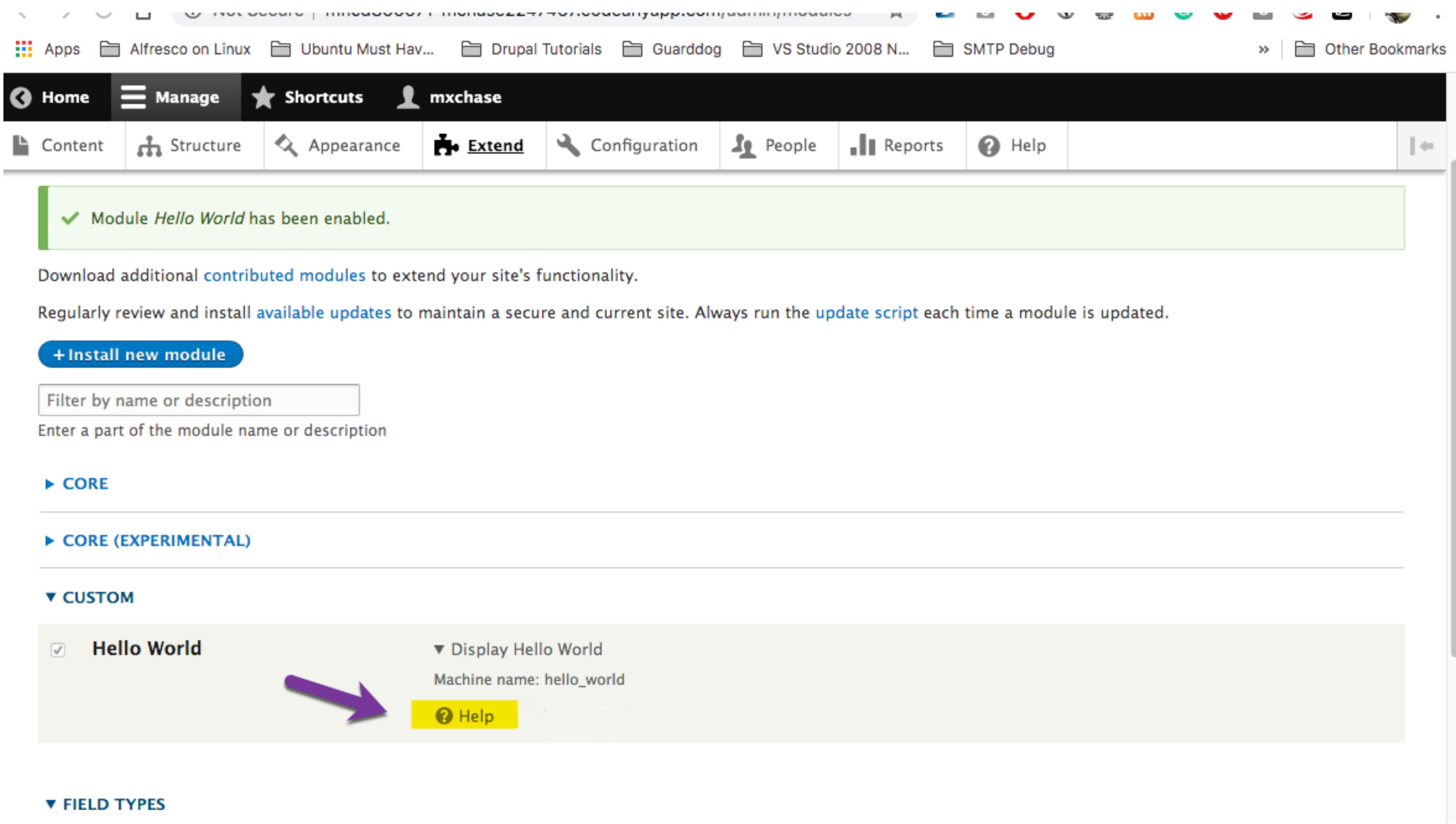
Change the Hook code to reflect the custom module namespace



lumes/Worker/DePaul IT320 Winter 2020 Online ONLY/Modules 1-11/Module 09/Tutorial 09/Tutorial 09 Assets/Custon

```
<?php

use Drupal\Core\Routing\RouteMatchInterface;

/**
*
*
*/

function hello_world_help($route_name, RouteMatchInterface $route_match) {
  switch ($route_name) {
    case 'help.page.hello_world':
        $output= '';
        $output .= '<h3>' . t('About') . '</h3>';
        $output .= '<p>' . t('This is an example module.') . '</p>';
        return $output;

        default;
     }
}
```

Test the Hook Help Implementation
module must be enabled to see the Help Link



Module *Hello World* has been enabled.

Download additional contributed modules to extend your site's functionality.

Regularly review and install available updates to maintain a secure and current site. Always run the update script each time a module is updated.

+Install new module

Filter by name or description

Enter a part of the module name or description

▸ CORE

▸ CORE (EXPERIMENTAL)

▾ CUSTOM

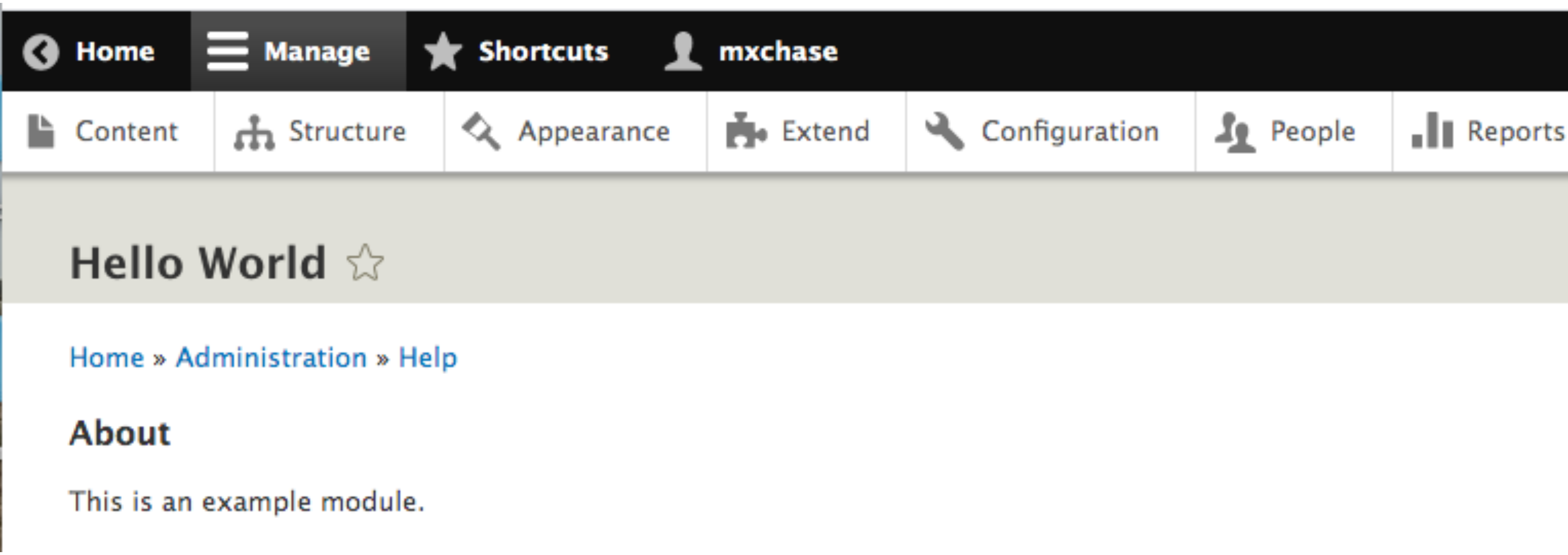
Hello World — ▾ Display Hello World

Machine name: hello_world

Help

▾ FIELD TYPES

Hook Help works



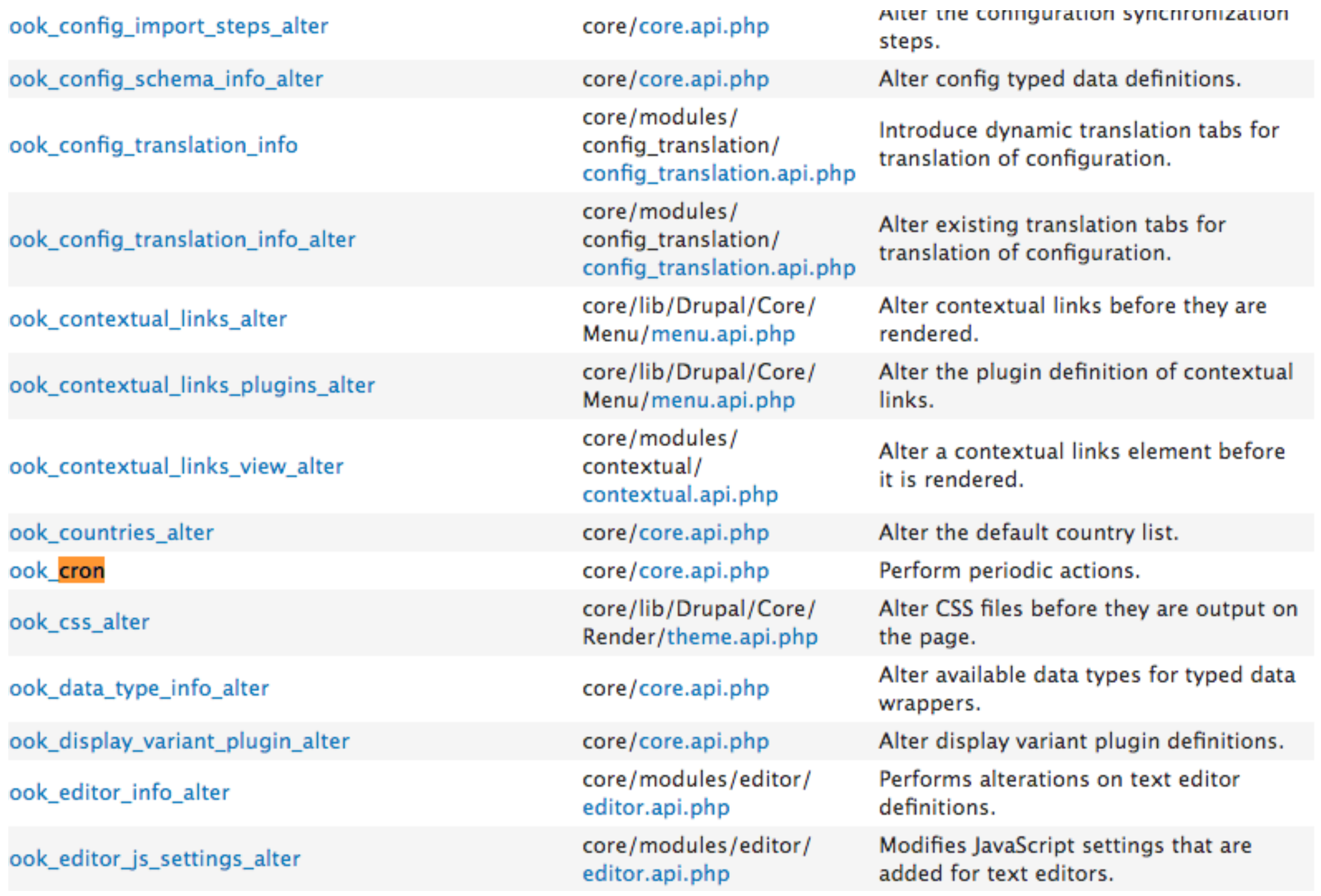
Hello World ☆

Home » Administration » Help

About

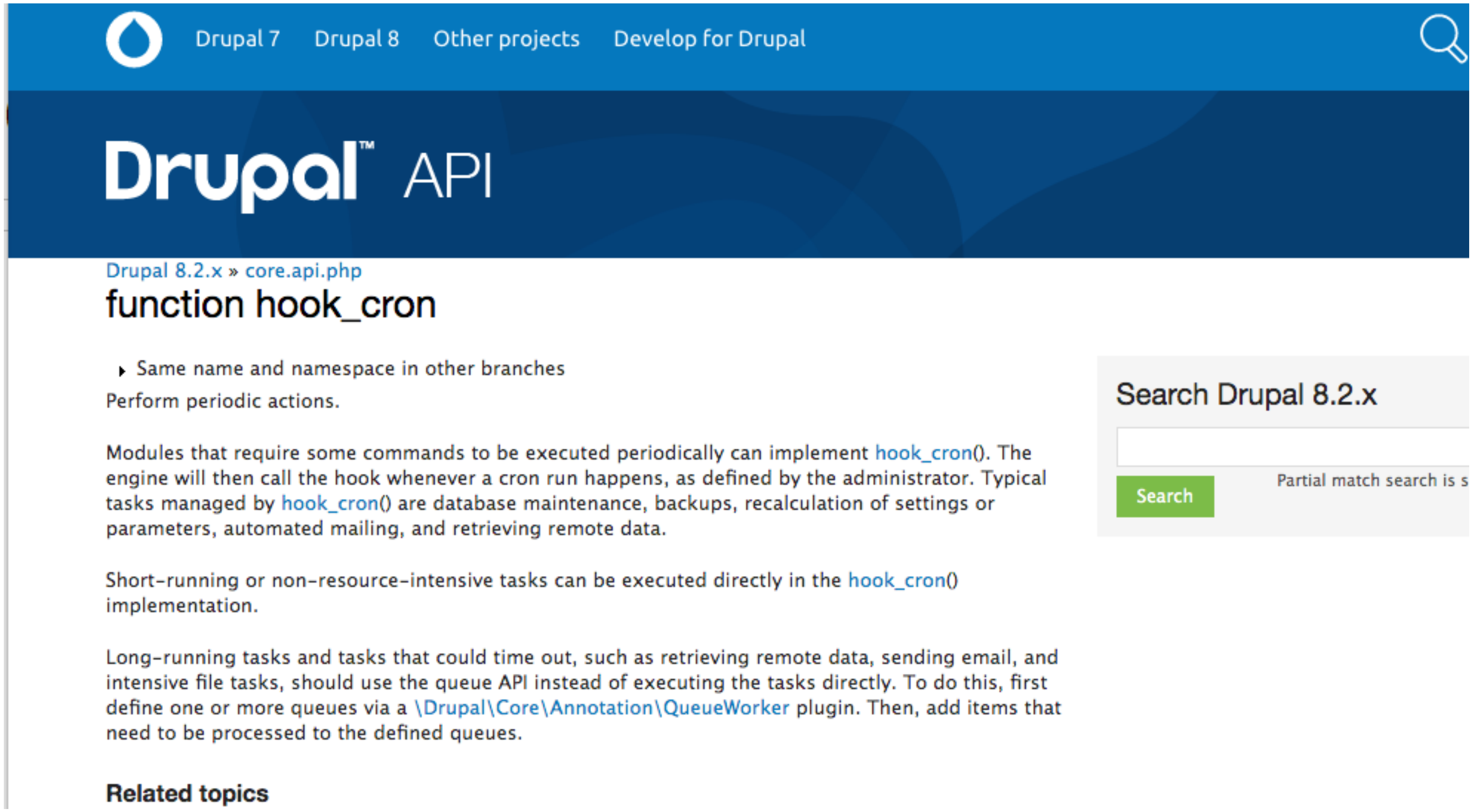
This is an example module.

Hook Cron link on list of Hooks



ook_config_import_steps_alter	core/core.api.php	Alter the configuration synchronization steps.
ook_config_schema_info_alter	core/core.api.php	Alter config typed data definitions.
ook_config_translation_info	core/modules/config_translation/config_translation.api.php	Introduce dynamic translation tabs for translation of configuration.
ook_config_translation_info_alter	core/modules/config_translation/config_translation.api.php	Alter existing translation tabs for translation of configuration.
ook_contextual_links_alter	core/lib/Drupal/Core/Menu/menu.api.php	Alter contextual links before they are rendered.
ook_contextual_links_plugins_alter	core/lib/Drupal/Core/Menu/menu.api.php	Alter the plugin definition of contextual links.
ook_contextual_links_view_alter	core/modules/contextual/contextual.api.php	Alter a contextual links element before it is rendered.
ook_countries_alter	core/core.api.php	Alter the default country list.
ook_cron	core/core.api.php	Perform periodic actions.
ook_css_alter	core/lib/Drupal/Core/Render/theme.api.php	Alter CSS files before they are output on the page.
ook_data_type_info_alter	core/core.api.php	Alter available data types for typed data wrappers.
ook_display_variant_plugin_alter	core/core.api.php	Alter display variant plugin definitions.
ook_editor_info_alter	core/modules/editor/editor.api.php	Performs alterations on text editor definitions.
ook_editor_js_settings_alter	core/modules/editor/editor.api.php	Modifies JavaScript settings that are added for text editors.

Function hook_cron



Drupal 8.2.x » core.api.php

function hook_cron

▸ Same name and namespace in other branches

Perform periodic actions.

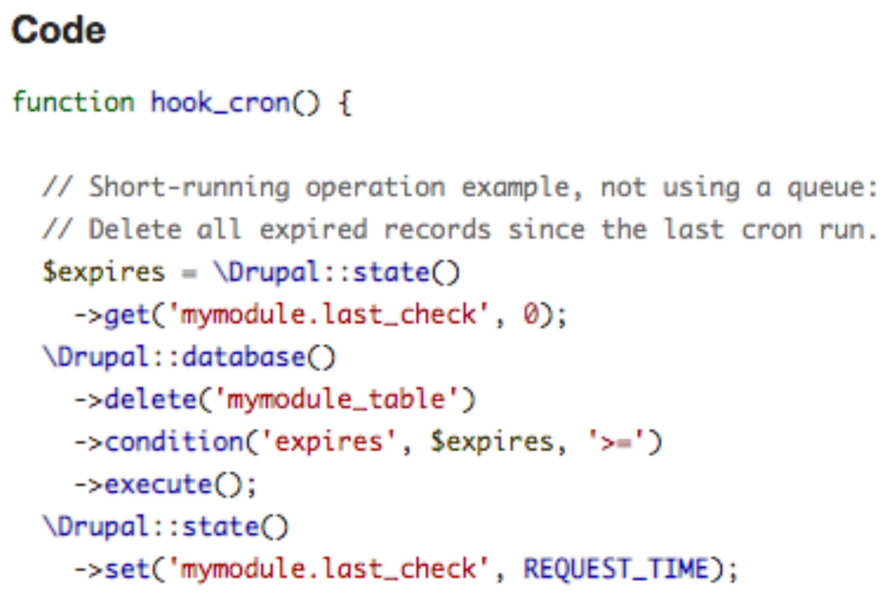
Modules that require some commands to be executed periodically can implement hook_cron(). The engine will then call the hook whenever a cron run happens, as defined by the administrator. Typical tasks managed by hook_cron() are database maintenance, backups, recalculation of settings or parameters, automated mailing, and retrieving remote data.

Short-running or non-resource-intensive tasks can be executed directly in the hook_cron() implementation.

Long-running tasks and tasks that could time out, such as retrieving remote data, sending email, and intensive file tasks, should use the queue API instead of executing the tasks directly. To do this, first define one or more queues via a \Drupal\Core\Annotation\QueueWorker plugin. Then, add items that need to be processed to the defined queues.

Related topics

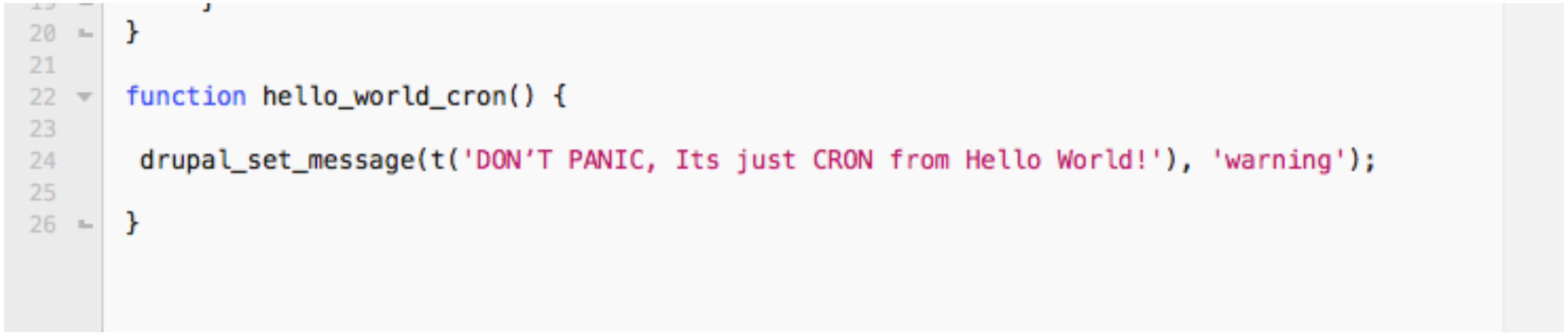
Function hook_cron example code



Code

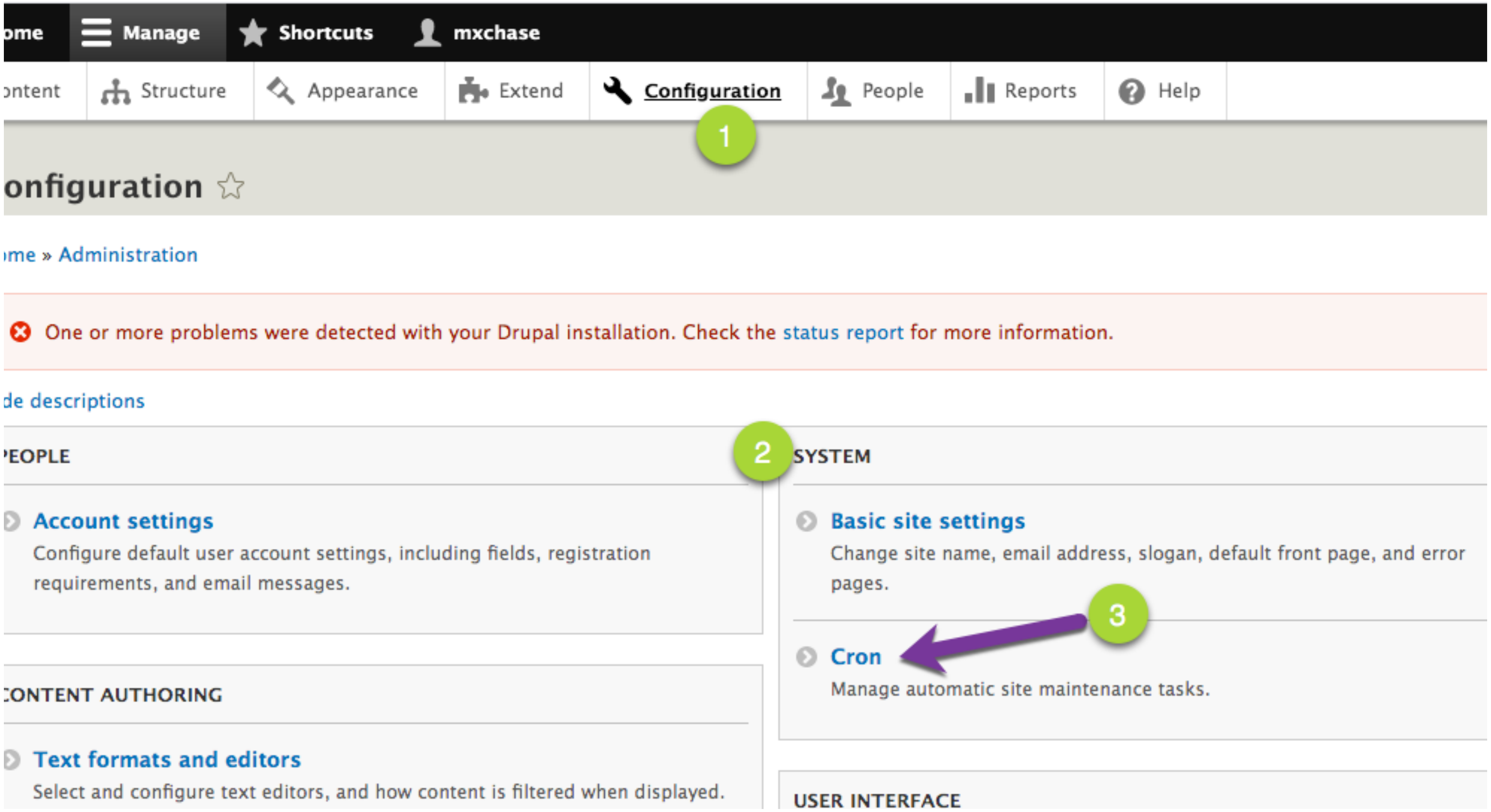
```
function hook_cron() {

  // Short-running operation example, not using a queue:
  // Delete all expired records since the last cron run.
  $expires = \Drupal::state()
    ->get('mymodule.last_check', 0);
  \Drupal::database()
    ->delete('mymodule_table')
    ->condition('expires', $expires, '>=')
    ->execute();
  \Drupal::state()
    ->set('mymodule.last_check', REQUEST_TIME);
```

A more basic implementation of hook_cron, just post a message



```
}

function hello_world_cron() {

 drupal_set_message(t('DON'T PANIC, Its just CRON from Hello World!'), 'warning');

}
```

Test by running CRON
1-3 Access the "Run cron" button



onfiguration ☆

ome » Administration

One or more problems were detected with your Drupal installation. Check the status report for more information.

de descriptions

EOPLE

Account settings
Configure default user account settings, including fields, registration requirements, and email messages.

ONTENT AUTHORING

Text formats and editors
Select and configure text editors, and how content is filtered when displayed.

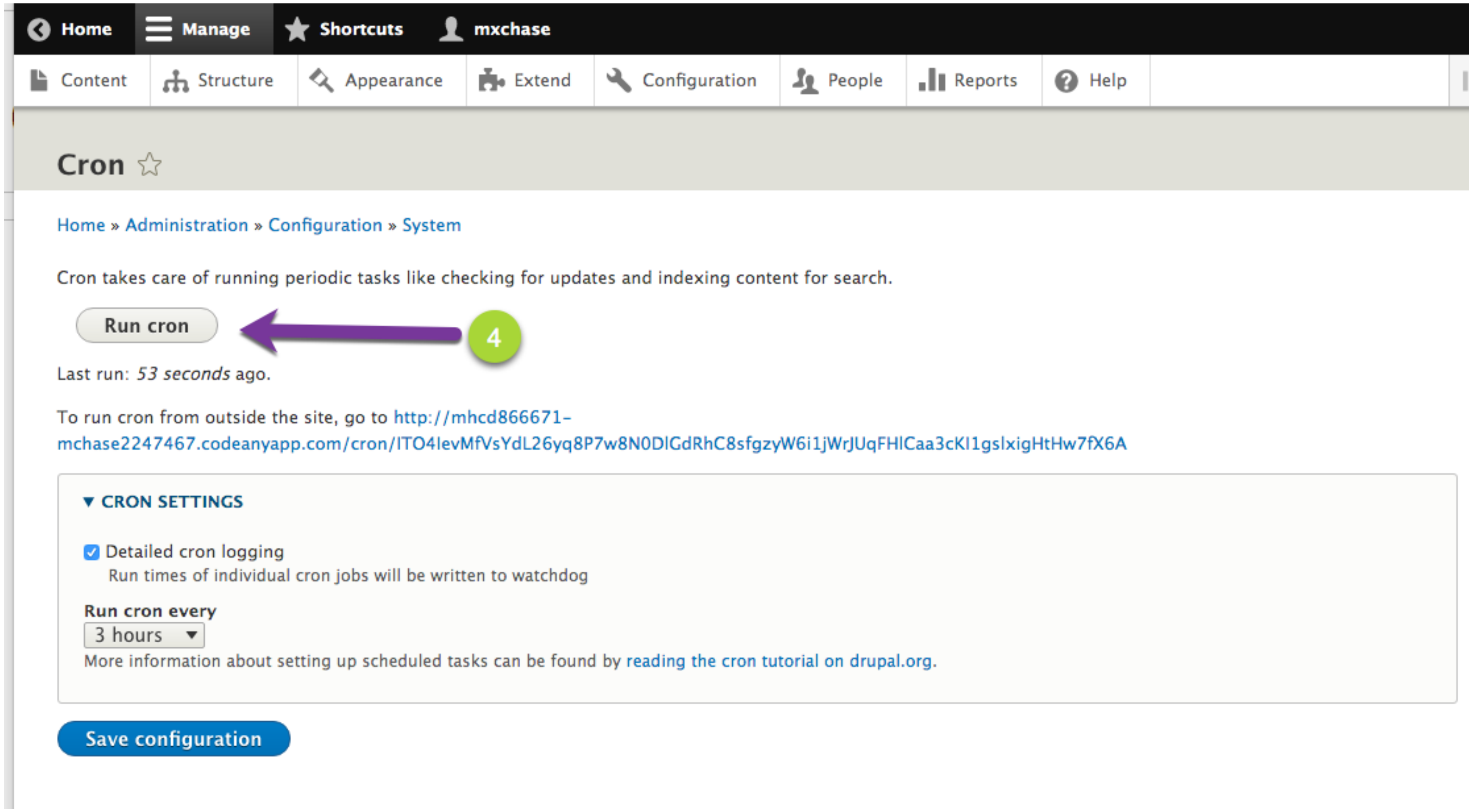
SYSTEM

Basic site settings
Change site name, email address, slogan, default front page, and error pages.

Cron
Manage automatic site maintenance tasks.

USER INTERFACE

(4) Press the "Run cron" button



Cron

Home » Administration » Configuration » System

Cron takes care of running periodic tasks like checking for updates and indexing content for search.

Run cron

Last run: *53 seconds* ago.

To run cron from outside the site, go to http://mhcd866671-mchase2247467.codeanyapp.com/cron/ITO4levMfVsYdL26yq8P7w8N0DlGdRhC8sfgzyW6i1jWrJUqFHlCaa3cKl1gslxigHtHw7fX6A

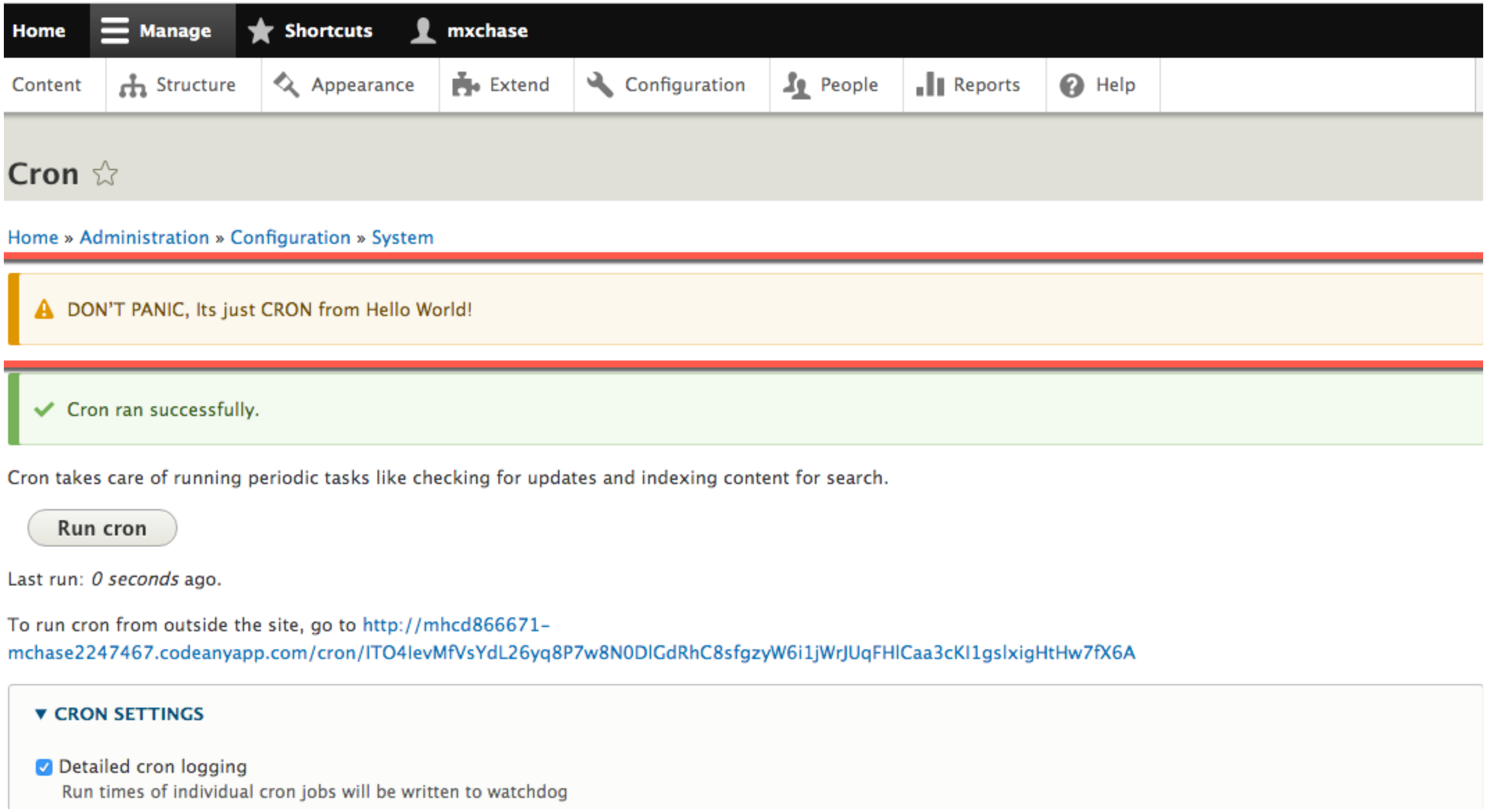
▾ CRON SETTINGS

☑ Detailed cron logging
Run times of individual cron jobs will be written to watchdog

Run cron every
3 hours ▾
More information about setting up scheduled tasks can be found by reading the cron tutorial on drupal.org.

Save configuration

Our hello world Cron message



Cron ☆

Home » Administration » Configuration » System

DON'T PANIC, Its just CRON from Hello World!

Cron ran successfully.

Cron takes care of running periodic tasks like checking for updates and indexing content for search.

Run cron

Last run: *0 seconds* ago.

To run cron from outside the site, go to http://mhcd866671-mchase2247467.codeanyapp.com/cron/ITO4levMfVsYdL26yq8P7w8N0DlGdRhC8sfgzyW6i1jWrJUqFHlCaa3cKl1gslxigHtHw7fX6A

▼ CRON SETTINGS

☑ Detailed cron logging
Run times of individual cron jobs will be written to watchdog

We need a command line interface (CLI) to clear cache

We need a way to clear the Drupal 8 cache in case we break our Drupal 8 site with our custom module code.

If we do break out site, we will most probably break it so bad that we will not not be able to access the admin backend, so we need a way to clear cache outside of the Drupal admin website.

Our 2 CLI options are Drush and Drupal Console and

What are command-line (CLI) tools?

- Command-line tools provide an alternative to using the administrative interface for various operations on your site. Many site builders and maintainers have invested the time to install and learn a command-line tool, because:
  - Administrative tasks are typically faster and less tedious when performed at the command line than in the user interface.
  - You can write scripts that combine site-related commands with other commands on the server, to automate more complicated tasks.
  - Command-line tools provide additional functionality not available via the administrative interface; for example, running database queries.

CLI(s) Drush

- Drush has been available longer, and has commands for both the core software and contributed modules; this guide documents the Drush commands for many tasks.
- To use DRUSH, you will need to have command-line terminal access to the server where your website will be hosted, and you will need to install Composer first in order to install Drush.

Drush CLI

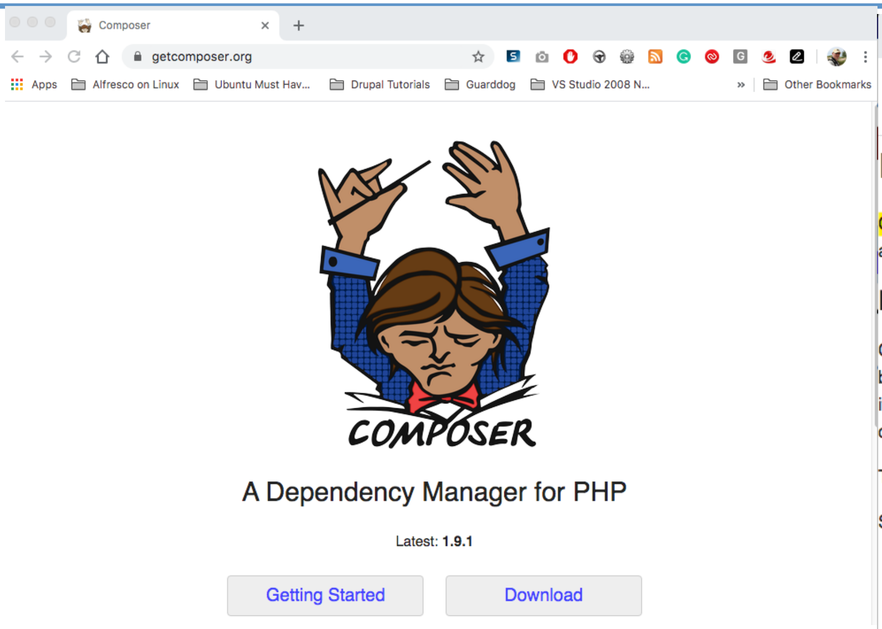
Drush is a computer software shell-based application used to control, manipulate, and administer Drupal websites. On the surface, drush is a tool for updating site modules, however Drush has a more comprehensive list of features.

We install Drush using a composer command.

We can use Drush instead of Drupal console to clear the cache.

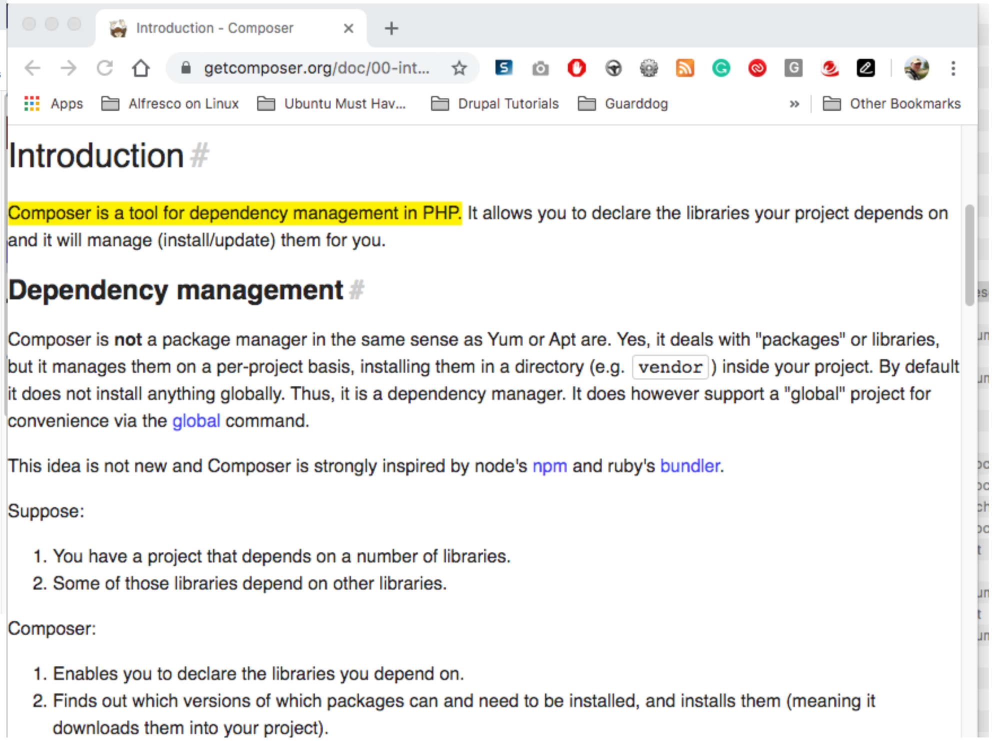
What is Composer?

- Composer is an application-level package manager for the PHP programming language that provides a standard format for managing dependencies of PHP software and required libraries.



A Dependency Manager for PHP

Latest: **1.9.1**



Introduction

Composer is a tool for dependency management in PHP. It allows you to declare the libraries your project depends on and it will manage (install/update) them for you.

Dependency management

Composer is **not** a package manager in the same sense as Yum or Apt are. Yes, it deals with "packages" or libraries, but it manages them on a per-project basis, installing them in a directory (e.g. `vendor`) inside your project. By default it does not install anything globally. Thus, it is a dependency manager. It does however support a "global" project for convenience via the global command.

This idea is not new and Composer is strongly inspired by node's npm and ruby's bundler.

Suppose:

1. You have a project that depends on a number of libraries.
2. Some of those libraries depend on other libraries.

Composer:

1. Enables you to declare the libraries you depend on.
2. Finds out which versions of which packages can and need to be installed, and installs them (meaning it downloads them into your project).

To install Drush using composer

1. Open an SSH terminal

2. Set ownership of the /usr and all its subfolders to cabox:cabox

```
?>  sudo chown  -R  cabox:cabox /usr
```

3. Set ownership of the composer to cabox:cabox

```
?> sudo chown cabox:cabox /usr/local/bin/composer
```

4. Run the composer update command

```
?> composer self-update
```

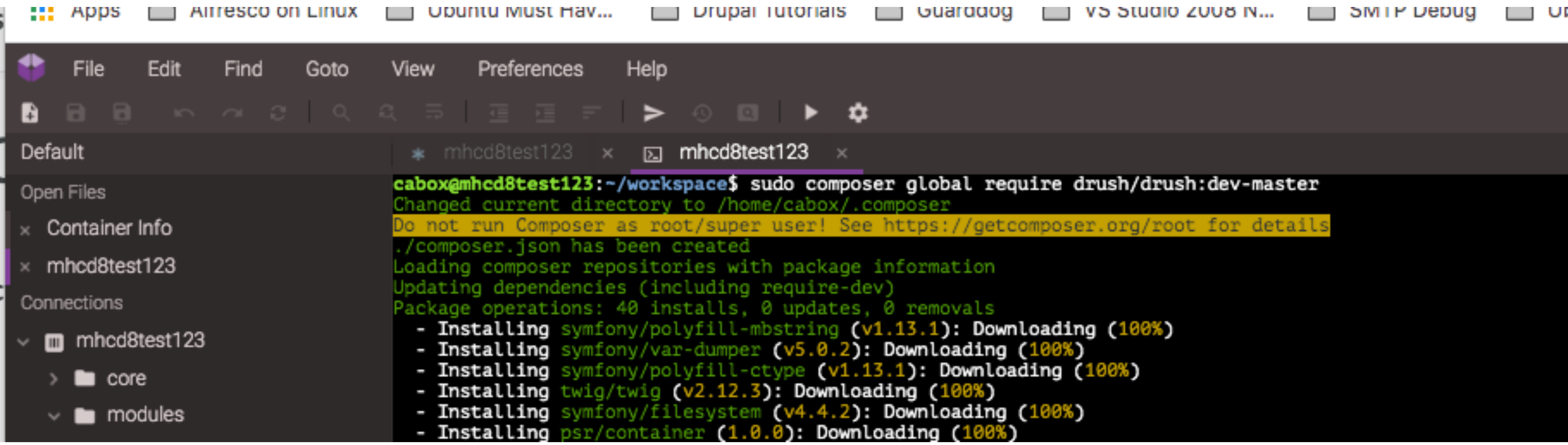
5. Step 4 – check the composer version

```
?> composer –version
```

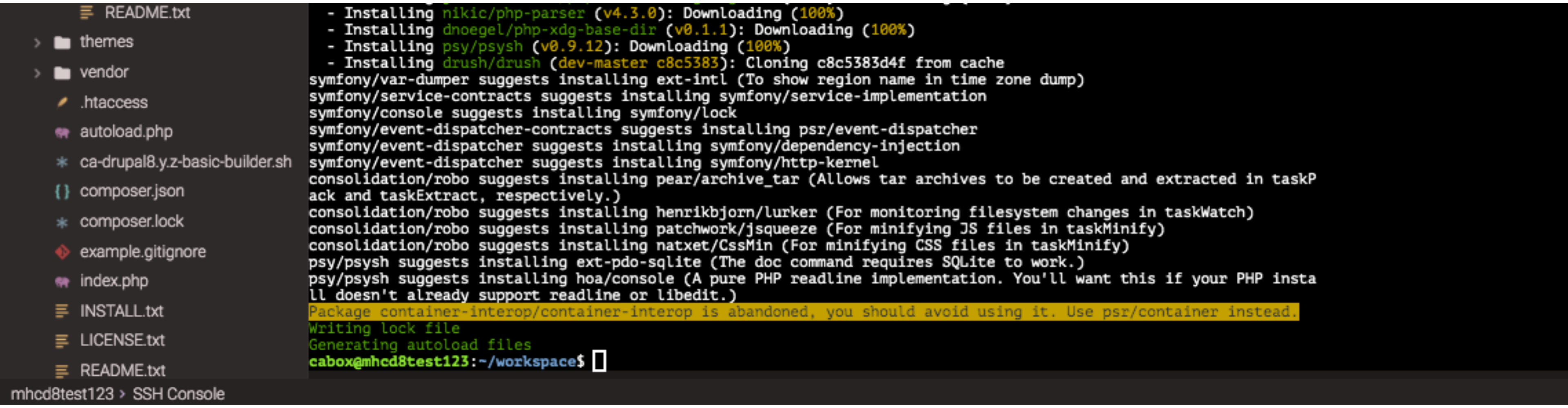
6. Run the command

```
composer global require drush/drush:dev-master
```

Installing Drush via composer

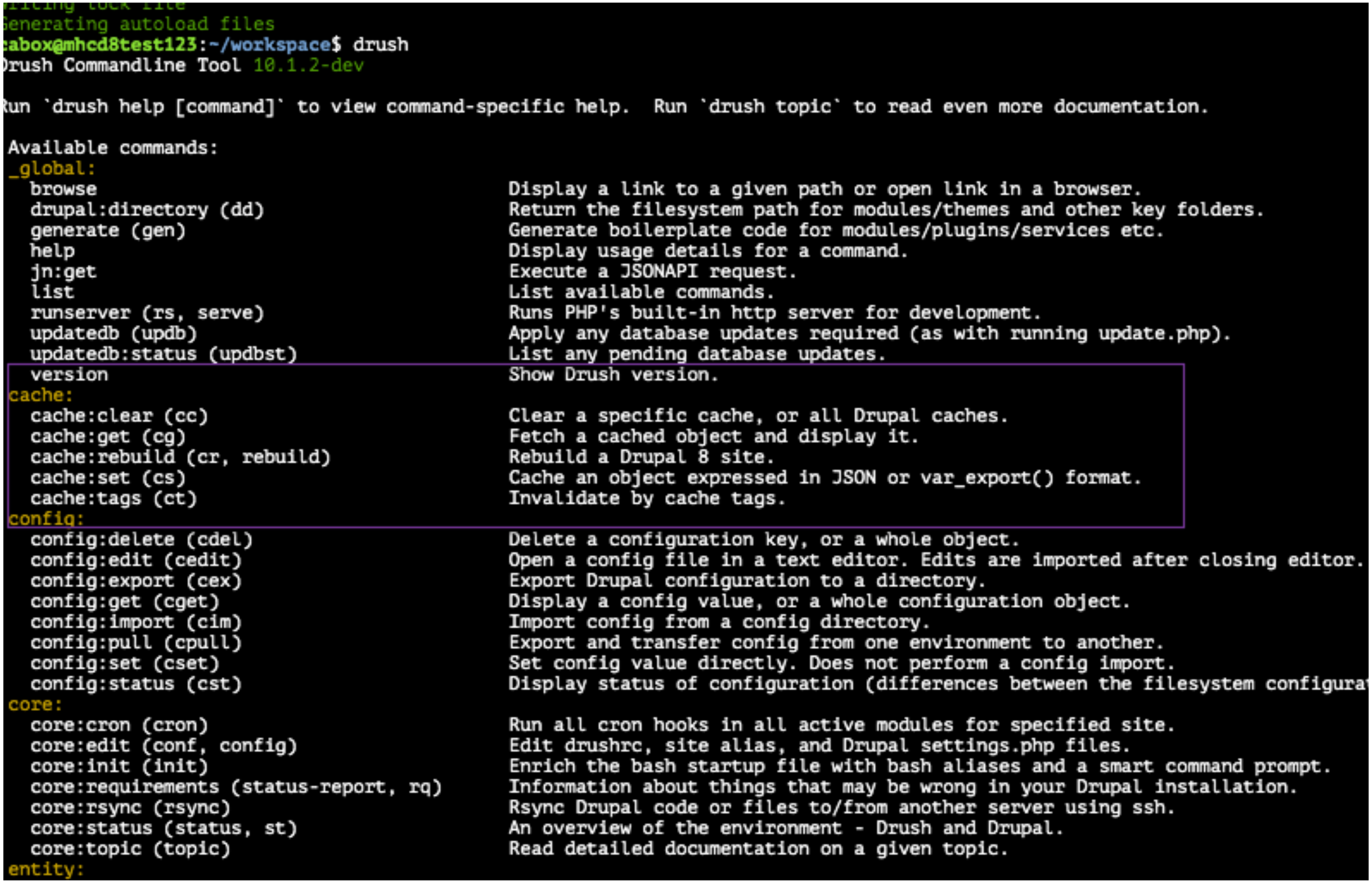


```
cabox@mhcd8test123:~/workspace$ sudo composer global require drush/drush:dev-master
Changed current directory to /home/cabox/.composer
Do not run Composer as root/super user! See https://getcomposer.org/root for details
./composer.json has been created
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 40 installs, 0 updates, 0 removals
  - Installing symfony/polyfill-mbstring (v1.13.1): Downloading (100%)
  - Installing symfony/var-dumper (v5.0.2): Downloading (100%)
  - Installing symfony/polyfill-ctype (v1.13.1): Downloading (100%)
  - Installing twig/twig (v2.12.3): Downloading (100%)
  - Installing symfony/filesystem (v4.4.2): Downloading (100%)
  - Installing psr/container (1.0.0): Downloading (100%)
```



```
  - Installing nikic/php-parser (v4.3.0): Downloading (100%)
  - Installing dnoegel/php-xdg-base-dir (v0.1.1): Downloading (100%)
  - Installing psy/psysh (v0.9.12): Downloading (100%)
  - Installing drush/drush (dev-master c8c5383): Cloning c8c5383d4f from cache
symfony/var-dumper suggests installing ext-intl (To show region name in time zone dump)
symfony/service-contracts suggests installing symfony/service-implementation
symfony/console suggests installing symfony/lock
symfony/event-dispatcher-contracts suggests installing psr/event-dispatcher
symfony/event-dispatcher suggests installing symfony/dependency-injection
symfony/event-dispatcher suggests installing symfony/http-kernel
consolidation/robo suggests installing pear/archive_tar (Allows tar archives to be created and extracted in taskP
ack and taskExtract, respectively.)
consolidation/robo suggests installing henrikbjorn/lurker (For monitoring filesystem changes in taskWatch)
consolidation/robo suggests installing patchwork/jsqueeze (For minifying JS files in taskMinify)
consolidation/robo suggests installing natxet/CssMin (For minifying CSS files in taskMinify)
psy/psysh suggests installing ext-pdo-sqlite (The doc command requires SQLite to work.)
psy/psysh suggests installing hoa/console (A pure PHP readline implementation. You'll want this if your PHP insta
ll doesn't already support readline or libedit.)
Package container-interop/container-interop is abandoned, you should avoid using it. Use psr/container instead.
Writing lock file
Generating autoload files
cabox@mhcd8test123:~/workspace$ 
```

At the command line, enter drush to get available commands



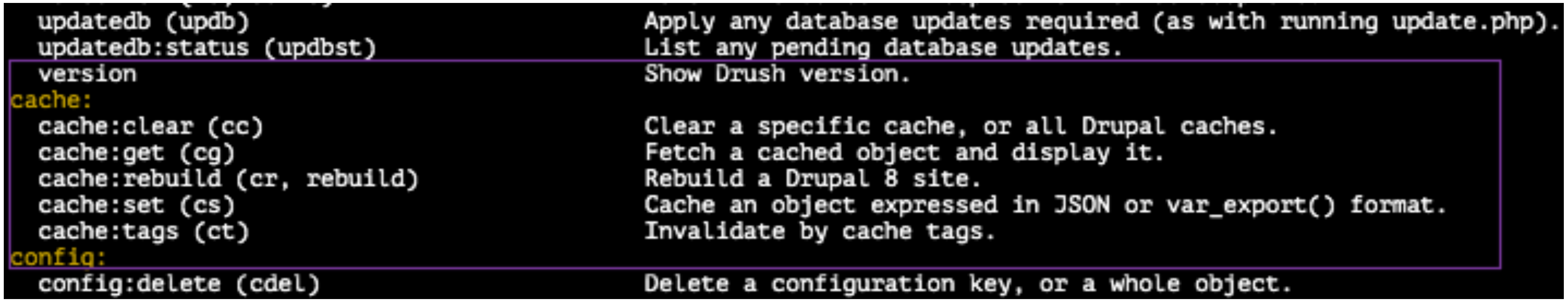
```
enerating autoload files
abox@mhcd8test123:~/workspace$ drush
rush Commandline Tool 10.1.2-dev

un `drush help [command]` to view command-specific help.  Run `drush topic` to read even more documentation.

 Available commands:
 _global:
   browse                                      Display a link to a given path or open link in a browser.
   drupal:directory (dd)                       Return the filesystem path for modules/themes and other key folders.
   generate (gen)                              Generate boilerplate code for modules/plugins/services etc.
   help                                        Display usage details for a command.
   jn:get                                      Execute a JSONAPI request.
   list                                        List available commands.
   runserver (rs, serve)                       Runs PHP's built-in http server for development.
   updatedb (updb)                             Apply any database updates required (as with running update.php).
   updatedb:status (updbst)                    List any pending database updates.
   version                                     Show Drush version.
 cache:
   cache:clear (cc)                            Clear a specific cache, or all Drupal caches.
   cache:get (cg)                              Fetch a cached object and display it.
   cache:rebuild (cr, rebuild)                 Rebuild a Drupal 8 site.
   cache:set (cs)                              Cache an object expressed in JSON or var_export() format.
   cache:tags (ct)                             Invalidate by cache tags.
 config:
   config:delete (cdel)                        Delete a configuration key, or a whole object.
   config:edit (cedit)                         Open a config file in a text editor. Edits are imported after closing editor.
   config:export (cex)                         Export Drupal configuration to a directory.
   config:get (cget)                           Display a config value, or a whole configuration object.
   config:import (cim)                         Import config from a config directory.
   config:pull (cpull)                         Export and transfer config from one environment to another.
   config:set (cset)                           Set config value directly. Does not perform a config import.
   config:status (cst)                         Display status of configuration (differences between the filesystem configura
 core:
   core:cron (cron)                            Run all cron hooks in all active modules for specified site.
   core:edit (conf, config)                    Edit drushrc, site alias, and Drupal settings.php files.
   core:init (init)                            Enrich the bash startup file with bash aliases and a smart command prompt.
   core:requirements (status-report, rq)       Information about things that may be wrong in your Drupal installation.
   core:rsync (rsync)                          Rsync Drupal code or files to/from another server using ssh.
   core:status (status, st)                    An overview of the environment - Drush and Drupal.
   core:topic (topic)                          Read detailed documentation on a given topic.
 entity:
```

We want to use drush to either clear the router cache or rebuild cache

$drush cache:clear

or

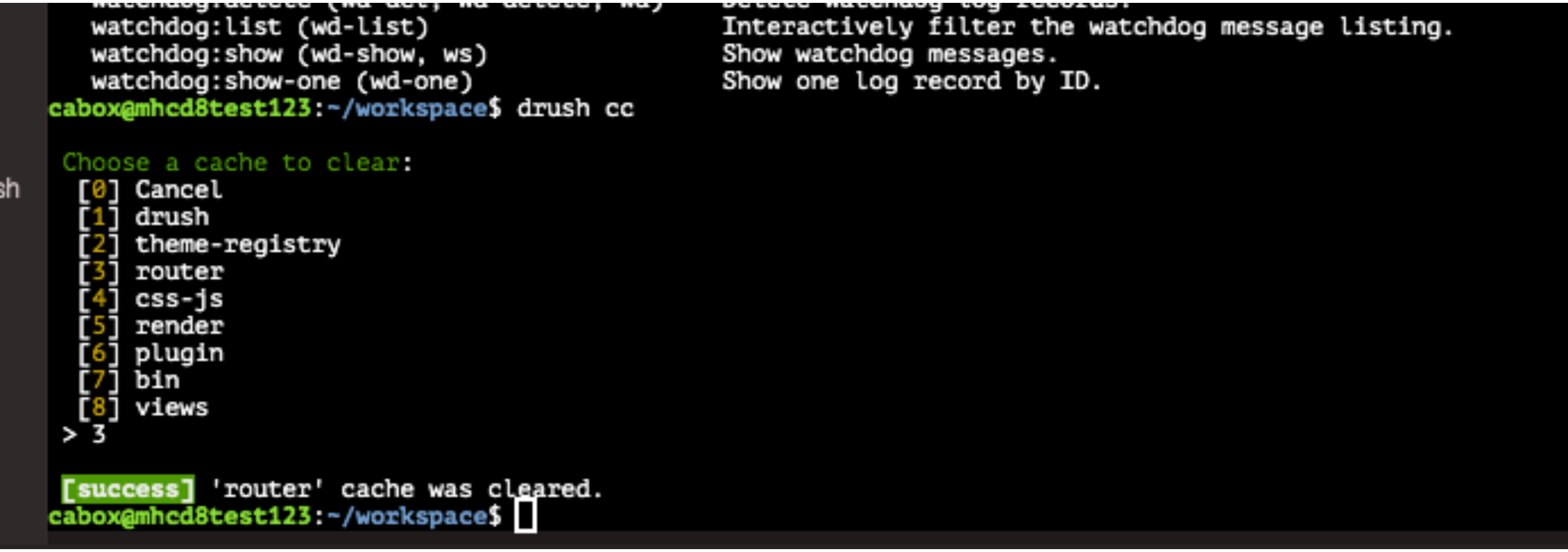
$drush cc



```
  updatedb (updb)                           Apply any database updates required (as with running update.php).
  updatedb:status (updbst)                  List any pending database updates.
  version                                   Show Drush version.
cache:
  cache:clear (cc)                          Clear a specific cache, or all Drupal caches.
  cache:get (cg)                            Fetch a cached object and display it.
  cache:rebuild (cr, rebuild)               Rebuild a Drupal 8 site.
  cache:set (cs)                            Cache an object expressed in JSON or var_export() format.
  cache:tags (ct)                           Invalidate by cache tags.
config:
  config:delete (cdel)                      Delete a configuration key, or a whole object.
```

To clear cache

$drush cc – then select 3



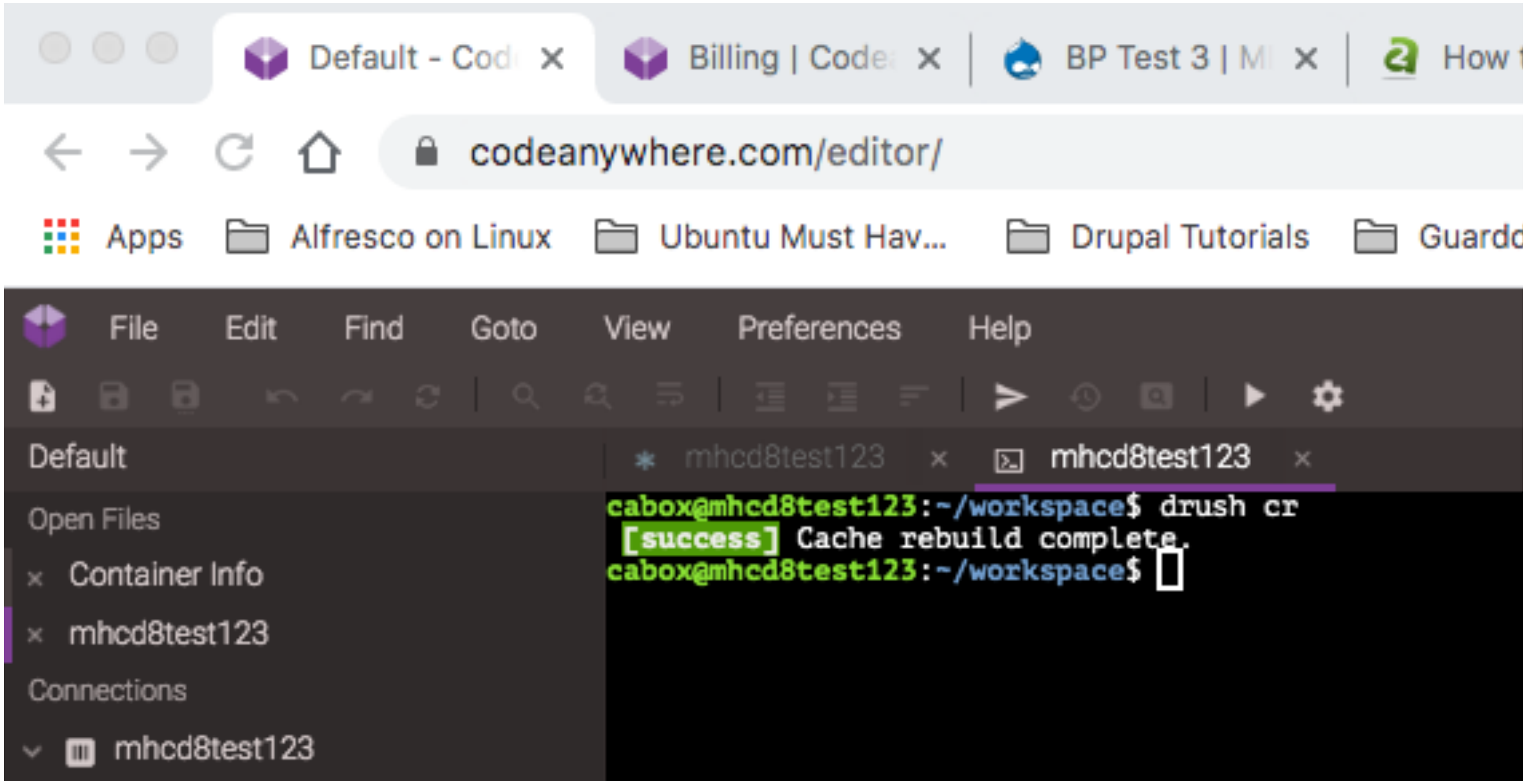
```
   watchdog:list (wd-list)                        Interactively filter the watchdog message listing.
   watchdog:show (wd-show, ws)                    Show watchdog messages.
   watchdog:show-one (wd-one)                     Show one log record by ID.
cabox@mhcd8test123:~/workspace$ drush cc

 Choose a cache to clear:
  [0] Cancel
  [1] drush
  [2] theme-registry
  [3] router
  [4] css-js
  [5] render
  [6] plugin
  [7] bin
  [8] views
 > 3

 [success] 'router' cache was cleared.
cabox@mhcd8test123:~/workspace$
```

To rebuild cache

$drush cache:rebuild

or

$drush cr



```
cabox@mhcd8test123:~/workspace$ drush cr
 [success] Cache rebuild complete.
cabox@mhcd8test123:~/workspace$ 
```