

Getting Started, Part 2

Objectives

1. Create a new, local Drupal site using Drush.
2. Create a new local Git repository, add and commit code to it.
3. Understand the basics of Drupal's "Big 5" concepts:
 - a. Nodes
 - b. Taxonomy
 - c. Menus
 - d. Blocks/regions/themes
 - e. Users/roles/permissions

Schedule

Assumes a 3.5-hour block.

- 0:00-0:45 Homework review
- 0:45-1:15 Create a new local site using Drush
- 1:15-1:45 Create a new local Git repository, add and commit code to it
- 1:45-2:00 Break
- 2:00-3:15 Tour of Drupal's "Big 5" concepts
- 3:15-3:30 Homework assignments

Materials

A modern computer with the following downloaded and installed:

- Acquia Dev Desktop 2 - <http://acquia.com/downloads>
- Git version control system - <http://git-scm.com>

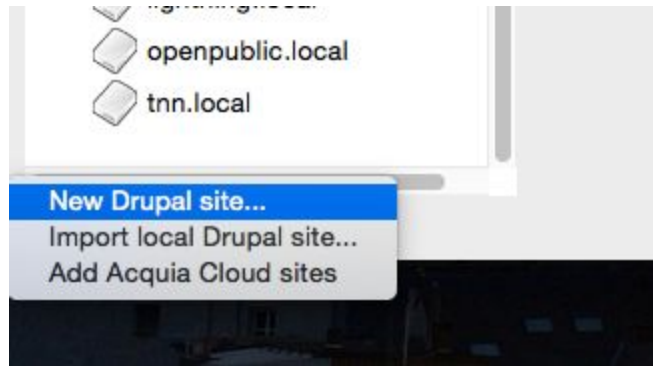
If the student is already familiar with *AMP stacks, then they are welcome to use the *AMP stack of their choice.

Prerequisites

Students should have some familiarity with the command-line interface. Drupalize.me (<https://drupalize.me/videos/moving-around-command-line?p=1149>) and Modules Unraveled (<https://modulesunraveled.com/command-line-beginners>) have a nice series of free screencasts with the basics. Additional resources can be found on the DrupalEasy Academy's web site on the "Satellite topics" page.

Discussion

1. Create a new, local site using Drush.
 - a. In the previous lesson, we (git) cloned an existing Drupal web site code base to our local machine and got it up-and-running.
 - b. Acquia Dev Desktop provides the ability to create new Drupal sites using the "+" button - but we're not going to use it.



- c. Instead we're going to create a new local Drupal site using Drush. Drush is a command-line interface for working with Drupal sites. In this example, we're going to use the `drush dl` command. `dl` is an alias for `pm-download`, where "pm" is an acronym for "package manager".
 - d. Open up your local terminal (if you're using Acquia Dev Desktop, open the command line via the "Open Drush Console" button) and navigate to your `/sites/` directory. For this example, we're going to download the current release of Drupal 8 using `drush dl drupal-8`. Drush will download the codebase and place it in a directory inside of your `sites/` directory. Once complete, rename the directory to "drupal8" (either from your file browser or via the command-line `mv` command).
 - e. Configure your AMP stack to recognize the new drupal8 site. If using Acquia Dev Desktop, this is via the "Import local Drupal site..." command. **Be sure the version of PHP is 5.6.x or better.** For the local site name, use "drupal8".
 - f. Once complete, click the "Local site" link on the main Acquia Dev Desktop window for the "drupal8" site. This will launch the site in your default web browser and bring you to the main "Install Drupal" page. Go through and complete the installation process.
2. Create a new local Git repository, add and commit code to it.
 - a. Now that we've created a new site, let's go ahead and create a new, local Git repository for it and commit the codebase to it. The repository that we're going to create will exist only on our local machines.
 - b. From the terminal (if you're on a Windows machine, you'll need to use the Git Bash command window for any Git commands), navigate into your `/sites/drupal8/` directory.

- c. Use the `git status` command to confirm that a repository does not already exist.

```
~/Sites/drupal8 $ git status
fatal: Not a git repository (or any of the parent
directories): .git
```

- d. Use the `git init` command to initialize a new, empty repository in the `sites/drupal8/` directory.

```
~/Sites/drupal8 $ git init
Initialized empty Git repository in
/Users/michael/Sites/drupal8/.git/
```

- e. Next, we have to decide exactly what we want to put in the new Git repository. Drupal 8 core includes an "example.gitignore" file that lists files for Git to ignore - meaning files that shouldn't be in the repository. In order to use this file, we need to copy and rename it to `.gitignore` using `cp example.gitignore .gitignore`. *Note: Drupal 7 comes with a .gitignore file ready-to-go, already named .gitignore.*
- f. Looking at the contents of `.gitignore`, we can see that it is suggested that environment-specific configuration settings, anything in the "files" directory, as well as several other files are set to be ignored by Git. If your local AMP stack is Acquia Dev Desktop, then it is recommended to add the following to the `.gitignore` file:

```
# Ignore Acquia Dev Desktop stuff
sites/*.dd
```

This can be accomplished by editing the `.gitignore` file either in a command-line text editor (pico, nano, or vim) or a desktop code editor (TextWrangler (<http://www.barebones.com/products/textwrangler/>) for Mac OS X and Notepad++ (<https://notepad-plus-plus.org/>) for Windows are good, free choices).

- g. From the `sites/drupal8/` directory, we can use the `git status` command to get the current status of the repository. Since this is a new, empty repository, this command will return:

```
~/Sites/drupal8 $ git status
On branch master
```

```
Initial commit
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be
committed)
```

```
.
.
.
```

```
nothing added to commit but untracked files present (use
"git add" to track)
```

- h. Committing code to a Git repository is a two-step process. First, we need to add it to the "staging area", then we can commit everything in the staging area to the repository.
- i. Add the entire codebase to the Git staging area (or "stage") using the `git add .` command from inside the `sites/drupal8/` directory. The `.` means to add everything that has not already been added as well as any changes that have been made (which ignoring any file changes for files listed in `.gitignore`). This command will not return anything (and may take several seconds to run).
- j. Running the `git status` command again will display a very long list of all the files in Drupal 8, all prepended with "new file:":

```
~/Sites/drupal8 $ git status
On branch master
```

```
Initial commit
```

```
Changes to be committed:
```

```
(use "git rm --cached <file>..." to unstage)
```

```
new file:   .csslintrc
new file:   .editorconfig
...
```

This is Git's way of telling us that each of the files will be added to the repository at the next commit.

- k. If you are new to Git, and/or you just installed Git on your machine, you'll need to configure it with your name and email address so that commits you make are credited to you. You can do this from the command-line with the following commands:

```
git config --global user.name "Your name"
git config --global user.email "your@emailaddress.com"
```

- I. The final step is to commit everything on the stage to our local repository. This will be done using the `git commit` command. Whenever `git commit` is used, we must also provide a useful commit message that becomes part of the repository's "permanent record". The commit message is often used to explain why a change was made. To make the initial commit, use `git commit -m 'Initial commit of Drupal 8 for the DCO!'`. The "-m" tells Git that we're including a commit message - this is required. If omitted, Git will automatically open the default command-line editor prompting you to add a commit message.
- m. Upon a successful commit, run `git status` again to ensure that your site is "clean".

```
~/Sites/drupal8 [master*]$ git status
On branch master
nothing to commit, working directory clean
```

A "clean" working directory means that there are no code changes that are not committed to the repository.

3. Introduction to Drupal's "Big 5" concepts.
 - a. As with other content management systems, Drupal is built on a set of fundamental concepts that are crucial to understand in order to fully leverage and extend. These concepts are often extended using contributed and custom modules and themes, so a solid understanding of the "Big 5" is key to building a strong foundational knowledge of Drupal.
 - b. Drupal's "Big 5" are all concepts implemented in Drupal core, and are normally present in production Drupal sites.
4. Understand the basics of Drupal's "Big 5" concepts:
 - a. Nodes
 - i. Nodes are the default unit of content on a Drupal site. In general, "content" can be considered anything that appears on a public facing page of this site (this includes user profiles, menus, etc...), but in general, when talking about just Drupal core, "content" normally refers to nodes.
 - ii. Nodes stem from various "content types". A content type is a structural definition of fields that describe the content. For example, an "article" content type may be comprised of a title, body, and category fields while a "movie" content type may be comprised of a title, year of release, lead actor, lead actress, trailer, movie poster, and director fields. When a new node is created, the first step is to select which type of content to create.
 - iii. Drupal core, by default, provides two content types: article and basic page. While similar, they have several differences including an image field on the article content type.
 - iv. One of the most powerful features of Drupal core (and something that makes it stand out from other content management systems) is the ability

to create new (custom) content types from within the administrative user interface. This allows site builders to create content types that match an organization's information architecture needs.

- v. *Exercise: as a class, brainstorm 1-2 examples of a content type and its fields.*
- vi. Nodes created from a content type have a great deal of functionality built-in by default. Their data can be indexed by Drupal's search system, comment forms can be associated with them, and they can be leveraged and extended by many, many contributed modules. The vast majority of Drupal sites utilize nodes as their main source of content.
- vii. Drupal 7 walkthrough
 1. As a site administrator, click on the main admin bar "Structure" link, then select "Content types".
 2. Two content types are listed: "Article" and "Basic page". While these are the default content types that Drupal core provides, neither is mandatory for use. They can both be removed or their structure modified using the drop button options.
 3. Content types are "entities". Entities will be covered in more detail later, but for now it is sufficient to understand that some entities are "fieldable", meaning fields can be added to them. When an entity is fieldable, the "Manage fields" and "Manage display" options are available. These configuration pages allow site-builders to add/edit/delete the information architecture of the content type by adding/editing/deleting fields attached to the content type.



- a. Manage fields - provides an interface for a site builder to add/edit/delete fields, including what type of data is stored (text, numbers, images, etc...) Drupal core provides several available field types (text, image, etc...) while contributed and custom modules can provide additional ones.
- b. Manage display - provides an interface for a site builder to control how the field data is displayed to an end-user (text-only, full HTML, image size, etc...)

4. The operations for each content type also provide access for the site builder to edit the basic configuration of the content type (its name, default published state, etc...) as well as to delete the content type from the system.
 5. *Exercise: what differences are there between the "Article" and "Basic page" content type by default?*
 6. Nodes, or site content, can be created via the main "Content" link in the menu bar. From this page, the user can select which type of content to create from a list of defined content types using the "Add content" button.
 7. *Exercise: create a node of type "Article" and type "Basic page" and confirm the differences from the previous exercise.*
- viii. Drupal 8 walkthrough
1. As a site administrator, click on the main admin bar "Structure" link, then select "Content types".

Content types ☆

[Home](#) » [Administration](#) » [Structure](#)

[+ Add content type](#)

NAME	DESCRIPTION	OPERATIONS
Article	Use <i>articles</i> for time-sensitive content like news, press releases or blog posts.	Manage fields ▼
Basic page	Use <i>basic pages</i> for your static content, such as an 'About us' page.	Manage fields ▼

2. Two content types are listed: "Article" and "Basic page". While these are the default content types that Drupal core provides, neither is mandatory for use. They can both be removed or their structure modified using the drop button options.
3. Content types are "entities". Entities will be covered in more detail later, but for now it is sufficient to understand that some entities are "fieldable", meaning fields can be added to them. When an entity is fieldable, the "Manage fields", "Manage form display", and "Manage display" options are available. These configuration pages allow site-builders to add/edit/delete the information architecture of the content type by adding/editing/deleting fields

attached to the content type.

Content types ☆

[Home](#) » [Administration](#) » [Structure](#)

[+ Add content type](#)

NAME	DESCRIPTION	OPERATIONS
Article	Use <i>articles</i> for time-sensitive content like news, press releases or blog posts.	Manage fields ▼
Basic page	Use <i>basic pages</i> for your static content, such as an 'About us' page.	Manage fields Manage form display Manage display Edit Delete

- a. Manage fields - provides an interface for a site builder to add/edit/delete fields, including what type of data is stored (text, numbers, images, etc...) Drupal core provides several available field types (text, image, etc...) while contributed and custom modules can provide additional ones.
 - b. Manage form display - provides an interface for a site builder to control how the field appears on a node add/edit page. This includes the widget used to add/edit data (text field, select box, radio buttons, etc...)
 - c. Manage display - provides an interface for a site builder to control how the field data is displayed to an end-user (text-only, full HTML, image size, etc...)
4. The drop buttons for each content type also provide access for the site builder to edit the basic configuration of the content type (its name, default published state, etc...) as well as to delete the content type from the system.
 5. *Exercise: what differences are there between the "Article" and "Basic page" content type by default?*
 6. Nodes, or site content, can be created via the main "Content" link in the menu bar. From this page, the user can select which type of content to create from a list of defined content types using the "Add content" button.
 7. *Exercise: create a node of type "Article" and type "Basic page" and confirm the differences from the previous exercise.*

b. Taxonomy

- i. Drupal's core taxonomy system allows site administrators to define and use a categorization system for all content on the site. The taxonomy system is separate from the node system, so other types of entities (including users and entities defined by custom and contributed modules) can be categorized as well.
- ii. The taxonomy system allows site administrators to create any number of "vocabularies". Each vocabulary contains a set of "terms" that can either be pre-defined by site administrators or extended by content authors (normally called "free-tagging").
 - 1. Example of a pre-defined vocabulary: Movie genres
 - a. Horror
 - b. Comedy
 - c. Action/adventure
 - d. Romance
 - 2. Example of a free-tagging vocabulary: Tags (see Drupal core default "Article" content type).
- iii. Once vocabularies and terms are defined and applied to content, they can be used to display content by category.
- iv. Vocabularies, like content types, are "fieldable entities". This means that additional fields can be added to each vocabulary to capture additional data about each term (in addition to just the term name).
- v. Once a vocabulary is defined, it can be applied to a content type (or any fieldable entity) via a "Taxonomy term reference" field. If it is necessary for content authors to be able to create new terms on the content add/edit page (free tagging), then the "Create referenced entities if they don't already exist" option should be selected on the taxonomy term reference field configuration page. The "Number of values" option for taxonomy term reference fields controls the number of terms that can be applied to a single piece of content. For example, if a "Movie genre" topic has a "Number of values" value of 2, then a content author can select up to 2 different genres to each movie node.
- vi. Taxonomy terms can be hierarchical.
- vii. Drupal 7 walkthrough
 - 1. As a site administrator, click on the main admin bar "Structure" link, then select "Taxonomy".
 - 2. A single vocabulary is available, "Tags". While this is the default vocabulary that Drupal core provides, it is not mandatory for use. It can be removed or its structure modified using the "Operations" links.
 - 3. Vocabularies also have an "edit vocabulary" option that allows site builders to change the basic configuration of the vocabulary (name, description) as well as the ability to delete the vocabulary from the system.

4. Like content types, vocabularies have the "Manage fields" and "Manage display" pages, available from the "edit vocabulary" page as tabs. These pages provide the same functionality as they do for content types - they allow site administrators to modify the information architecture for each vocabulary.

5. Vocabularies also have a "List terms" option that displays a list of all terms that have been defined for a given vocabulary.
6. Once a vocabulary is defined, it can be applied to a content type (or any fieldable entity) via a "Term reference" field. If it is necessary for content authors to be able to create new terms on the content add/edit page (free tagging), then the "Autocomplete term widget (tagging)" widget should be used.

LABEL	MACHINE NAME	FIELD TYPE	WIDGET	OPERATIONS
✚ Title	title	Node module element		
✚ Tags	field_tags	Term reference	Autocomplete term widget (tagging)	edit delete

7. Review the settings of the existing "Tags" vocabulary.
8. Review the settings of the existing "Tags" field on the "Article" content type. Note the "Autocomplete term widget (tagging)" widget.
9. Create a new vocabulary named "Topic". Create the following terms for it:
 - a. Drupal 6
 - b. Drupal 7
 - c. Drupal 8
10. Add a new "Term reference" field to the "Article" content type with the following configuration (leave default values for anything not specified):
 - a. Label: Topic

- b. Widget: Select list
 - c. Vocabularies: Topic
 - d. Required field: selected
- 11. *Exercise: Add a new node of type "Article", and confirm the new Topic vocabulary and field work as expected.*
- viii. Drupal 8 walkthrough
 1. As a site administrator, click on the main admin bar "Structure" link, then select "Taxonomy".
 2. A single vocabulary is available, "Tags". While this is the default vocabulary that Drupal core provides, it is not mandatory for use. It can be removed or its structure modified using the drop button options.
 3. Like content types, vocabularies have the "Manage fields", "Manage form display", and "Manage display" drop button options. These options provide the same functionality as they do for content types - they allow site administrators to modify the information architecture for each vocabulary as well as control the input and output configuration for each field's data.

Taxonomy ☆

[Home](#) » [Administration](#) » [Structure](#)

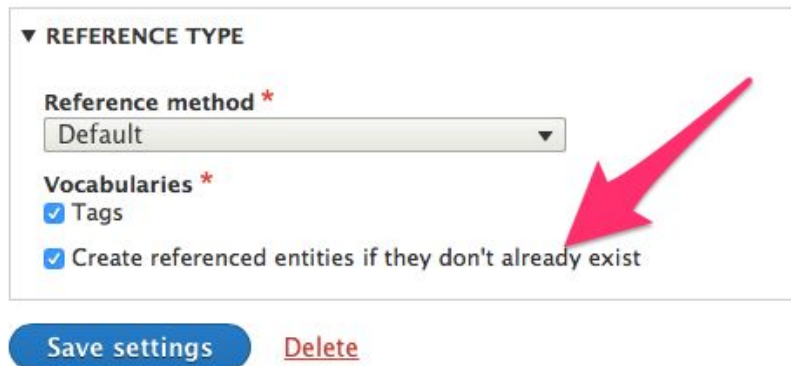
Taxonomy is for categorizing content. Terms are grouped into vocabularies. For example, a vocabulary called "Fruit" would contain the terms "Apple" and "Banana".

[+ Add vocabulary](#)

VOCABULARY NAME	OPERATIONS
Tags	<div> List terms Edit vocabulary Add terms Manage fields Manage form display Manage display </div>

4. Vocabularies also have "edit" and "delete" drop button options that allow site builders to change the basic configuration of the vocabulary (name, description) as well as the ability to delete the vocabulary from the system.

5. Vocabularies also have a "List terms" drop button option that displays a list of all terms that have been defined for a given vocabulary.
6. Once a vocabulary is defined, it can be applied to a content type (or any fieldable entity) via a "Taxonomy term reference" field. If it is necessary for content authors to be able to create new terms on the content add/edit page (free tagging), then the "Create referenced entities if they don't already exist" option should be selected on the taxonomy term reference field configuration page.
7. Review the settings of the existing "Tags" vocabulary.
8. Review the settings of the existing "Tags" field on the "Article" content type. Note the "Create referenced entities if they don't already exist" option.



▼ REFERENCE TYPE

Reference method *

Default

Vocabularies *

☒ Tags

☒ Create referenced entities if they don't already exist

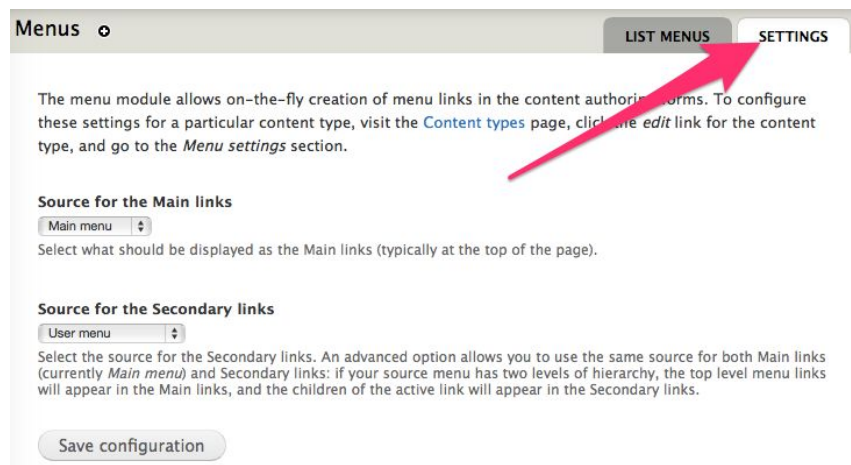
Save settings Delete

9. Create a new vocabulary named "Topic". Create the following terms for it:
 - a. Drupal 6
 - b. Drupal 7
 - c. Drupal 8
10. Add a new "Taxonomy term reference" field to the "Article" content type with the following configuration (leave default values for anything not specified):
 - a. Label: Topic
 - b. Required field: selected
 - c. Vocabularies: Topic
11. *Exercise: Add a new node of type "Article", and confirm the new Topic vocabulary and field work as expected.*

c. Menus

- i. Menus are collections of links. Links in menus can be internal (links within the Drupal site) or external (links to external sites). Links can be manually generated or automatically generated by contributed and/or custom modules.
- ii. Menus are **not** fieldable entities.

- iii. Drupal core provides several menus by default. Each of these menus contains default links also provided by Drupal core modules.
 - 1. Drupal 7 default menus:
 - a. Main menu - contains links for the main site navigation.
 - b. Management - contains links to administrative functions.
 - c. Navigation - contains links useful for content administrators.
 - d. User menu - contains user account related links.
 - 2. Drupal 8 default menus
 - a. Administration - contains links to administrative functions.
 - b. Footer - contains links for the site footer.
 - c. Main navigation - contains links for the main site navigation.
 - d. Tools - contains links for site visitors.
 - e. User account menu - contains user account related links.
- iv. Menus are commonly exposed on a Drupal site by placing a menu block (provided by default by Drupal core) in the desired region. In Drupal 7, some themes also provide an alternate method to place a menu in a predetermined menu location. This method is not available in Drupal 8.
- v. Menu links can be created and added to existing menus from various parts of Drupal core, including node add/edit pages.
- vi. Menu links can be hierarchical.
- vii. Drupal 7 walkthrough
 - 1. As a site administrator, click on the main admin bar "Structure" link, then select "Menus".
 - 2. Review the four default menus provided by Drupal core.
 - 3. From the home page of a default Drupal 7 install (with the Bartik theme active), identify the location of each of the four menus.
 - 4. Review the Menu settings (Structure | Menu | Settings) as an example of placing menus directly in predetermined theme locations.



5. Exercise:

- Create a new "Featured articles" menu.
- Modify the "Article" content type to allow authors to place links in this new menu.
- Edit two existing articles and add links to them to the new "Featured articles" menu.
- Visit the "List links" page of the "Featured articles" menu and confirm the new links are present.
- We'll place the menu so it is visible to site visitors in the next section.

viii. Drupal 8 walkthrough

- As a site administrator, click on the main admin bar "Structure" link, then select "Menus".
- Review the five default menus provided by Drupal core.

Menus ☆

[Home](#) » [Administration](#) » [Structure](#)

Each menu has a corresponding block that is managed on the [Block layout](#) page.

[+ Add menu](#)

TITLE	DESCRIPTION	OPERATIONS
Administration	Contains links to administrative tasks.	Edit menu ▼
Footer	Use this for linking to site information.	Edit menu ▼
Main navigation	Use this for linking to the main site sections.	Edit menu ▼
Tools	Contains links for site visitors. Some modules add their links here.	Edit menu ▼
User account menu	Links related to the user account.	Edit menu ▼

- From the home page of a default Drupal 8 install (with the Bartik theme active), identify the location of each of the four menus.

4. Exercise:

- Create a new "Featured articles" menu.
- Modify the "Article" content type to allow authors to place links in this new menu.
- Edit two existing articles and add menu links to them to the new "Featured articles" menu.
- Visit the "Edit menu" page of the "Featured articles" menu and confirm the new links are present.
- We'll place the menu so it is visible to site visitors in the next section.

d. Blocks/regions/themes

- i. Drupal core's block system allows site administrators to place (normally small) units of content (called "blocks") on various parts of the page with distinct visibility conditions.
- ii. Block content can be provided by core, custom, and contributed modules. For example, the Drupal core "Menu" module provides a block for each menu defined on the site.
- iii. Custom blocks with custom content can also be created through the administrative area.
- iv. In Drupal 8, blocks are fieldable entities.
- v. Blocks are assigned to regions of a page. Multiple blocks can be placed in a single region.
- vi. Regions are defined by the site's theme. For example, a custom theme might define a region called "Promotions" that is located in a prominent part of the page. Any defined block can be placed in the "Promotions" region.
- vii. Since regions are defined by themes, special care must be taken to ensure block placement when changing themes.
- viii. When a block is placed in a region, visibility settings can be defined to set the conditions for when a block is visible. For example, consider a "Sale items" block that is placed in the "Promotions" region. The block visibility settings can be configured so that the "Sale items" block only appears on the home page of the site.
- ix. Drupal 7 walkthrough
 1. As a site administrator, click on the main admin bar "Structure" link, then select "Blocks".
 2. The main "Blocks" page displays a list of regions and each of the blocks placed in each region. At the bottom of the list is a section of "Disabled" blocks that aren't currently placed in a region (and therefore not visible on the site). *Note: contributed and custom modules can expose blocks on the site using additional methods.*

Disabled		
 Development	<input type="text" value="- None -"/>	configure
 Main menu	<input type="text" value="- None -"/>	configure
 Management	<input type="text" value="- None -"/>	configure

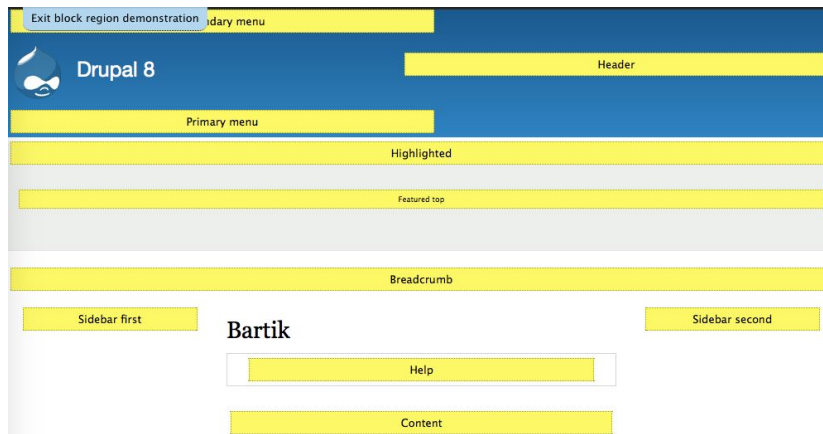
3. Click on the "Demonstrate block regions" link to view the relative locations of each region on a user-facing page.
4. Note that blocks can be easily moved between regions (or re-ordered within a single region) via drag-and-drop or by modifying the selected "Region" for each block (once the "Save blocks" button at the bottom of the page is clicked).

5. Visibility settings for each block are available on its "configure" page. In addition, a block's region and title can be set here as well.

Visibility settings

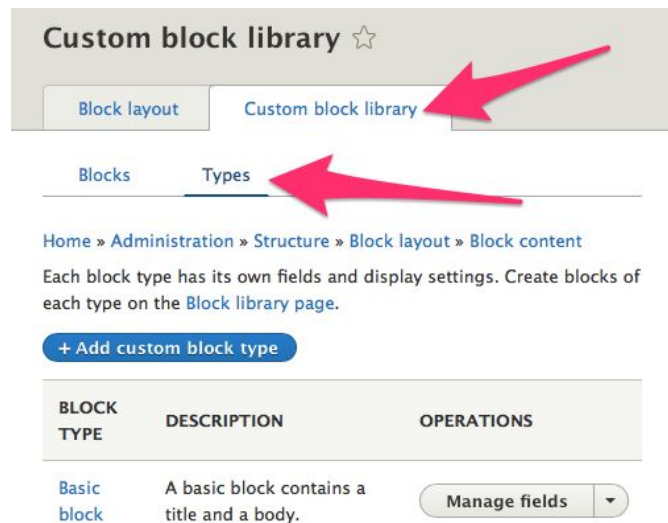
6. Custom blocks can be created via the "Add block" link on the main "Blocks" page.
7. *Exercise:*
- Create a new block called "Welcome".
 - Add some welcome text to the "Block body".
 - Configure the block so that it appears in the "Highlighted" region of the Bartik theme, but only on the home page of the site (hint: use "<front>").
8. *Exercise:*
- Place the "Featured articles" block from the previous exercise in the "Sidebar first region", and configure its visibility settings to only appear on pages that display nodes of type "Article".
- x. Drupal 8 walkthrough
- As a site administrator, click on the main admin bar "Structure" link, then select "Block layout".
 - The main "Block layout" page displays a list of regions and each of the blocks placed in each region. At the bottom of the list is a section of "Disabled" blocks that aren't currently placed in a region (and therefore not visible on the site). *Note: contributed and custom modules can expose blocks on the site using additional methods.*

3. Click on the "Demonstrate block regions" link to view the relative locations of each region on a user-facing page.



4. Note that blocks can be easily moved between regions (or re-ordered within a single region) via drag-and-drop or by modifying the selected "Region" for each block (once the "Save blocks" button at the bottom of the page is clicked).
5. Visibility settings for each block are available on its "configure" page. In addition, a block's region and title can be set here as well.
6. In Drupal 8, blocks are fieldable entities, and various "block types" can be defined for a site, each one with a different set of fields. To view a list of custom blocks, click on the "Custom block library" tab on the main "Block layout" page.
7. The "Custom block library" has two sub-tabs:
 - a. Blocks
 - i. The "Blocks" sub-tab displays a list of all custom blocks on the site. From here, existing custom blocks can be edited or deleted, or new ones created via the "Add custom block" link.
 - ii. Drupal core does not provide any default custom blocks.
 - b. Types
 - i. The "Types" sub-tab displays a list of default block types provided by Drupal core. The "Basic block" block type is a simple block type similar to Drupal 7 blocks.
 - ii. The drop button options for each block type provide access to the familiar field configuration pages, "Manage fields", "Manage form display", and "Manage display".
 - iii. New block types can be created using the "Add custom block type" button and additional fields can

be added to each block type.



8. Exercise:

- i. Create a new custom block called "Welcome".
- ii. Add some welcome text to the "Block body".
- iii. Configure the block so that it appears in the "Highlighted" region of the Bartik theme, but only on the home page of the site (hint: use "<front>").

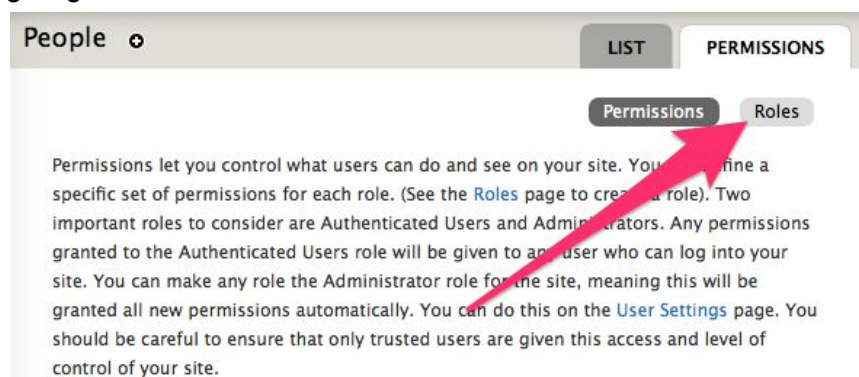
9. Exercise:

- i. Place the "Featured articles" block from the previous exercise in the "Sidebar first region", and configure its visibility settings to only appear on pages that display nodes of type "Article".

e. Users/roles/permissions

- i. Drupal core provides a robust user authentication, management, and permission system that is flexible enough to be configured for most situations.
- ii. Drupal core utilizes a common users/roles/permissions style system, where:
 1. Authenticated, logged in users are granted one or more roles.
 2. Roles are granted any number of permissions.
 3. Permissions provide granular access to functionality as defined by the modules that provide the given functionality. For example, the core "Search" module exposes the following permissions:
 - a. Administer search
 - b. Use advanced search
 - c. Use search
- iii. Any user that is visiting the site, but not logged in is granted the "anonymous user" role (the name of this role is editable).

- iv. Any user that is visiting the site and logged in is **always** granted the "authenticated user" role.
- v. Users may be granted additional (normally custom) roles by site administrators to provide additional access permissions **above and beyond** permissions already granted by the "authenticated user" role.
- vi. For example, consider a custom role called "search admin". This role is granted the "Administer search" permission, and is therefore allowed to modify search settings on site. When this role is granted to user "Amelia", when she is logged in, she has access to all of the permissions granted to the "authenticated user" role as well as all permissions granted to the "search admin" role - which includes the "Administer search" permission.
- vii. In addition to the "anonymous user" and "authenticated user" roles, Drupal core provides an "Administrator" role by default. This role is configured (by default) to automatically be granted all permissions (although in Drupal 7, this isn't always the case).
- viii. Drupal 7 walkthrough
 - 1. As a site administrator, click on the main admin bar "Structure" link, then select "People".
 - 2. The main "People" page contains a list of site users. Each user can be edited. Review configuration options for a user (be sure to note the "Roles" section).
 - 3. The "Permissions" tab leads to the main "Permissions" page that displays a list of all permissions exposed by all the enabled modules on the site. From this page, permissions may be granted to roles.
 - 4. The "Roles" sub-tab leads to the main "Roles" page that lists all of the active user roles on the site. Defining a user role only involves giving it a name.



- 5. Exercise:
 - a. Create a new "content editor" role.
 - b. Give the new role the "Article: edit any content" permission.

- c. *Create a new user and grant it the new "content editor" role.*
 - d. *Log out of the site as admin and log back in using the new user (or, alternatively, use a different browser to login as the new user) and confirm the new user can edit nodes of type "article".*
- ix. **Drupal 8 walkthrough**
 1. As a site administrator, click on the main admin bar "Structure" link, then select "People".
 2. The main "People" page contains a list of site users. Each user can be edited. Review configuration options for a user (be sure to note the "Roles" section).
 3. The "Permissions" tab leads to the main "Permissions" page that displays a list of all permissions exposed by all the enabled modules on the site. From this page, permissions may be granted to roles.
 4. The "Roles" tab leads to the main "Roles" page that lists all of the active user roles on the site. Defining a user role only involves giving it a name.

Add role ☆

[Home](#) » [Administration](#) » [People](#) » [Roles](#)

Role name *

The name for this role. Example: "Moderator", "Editorial board", "Site architect".

[Save](#)

5. **Exercise:**
 - a. *Create a new "content editor" role.*
 - b. *Give the new role the "Article: edit any content" permission.*
 - c. *Create a new user and grant it the new "content editor" role.*
 - d. *Log out of the site as admin and log back in using the new user (or, alternatively, use a different browser to login as the new user) and confirm the new user can edit nodes of type "article".*

Homework

1. Weekly activity report.
2. "Getting Started" assessment.
3. The following exercises are to be completed on your local Drupal 7 site (from Lesson 1, Part 1):
 - a. Create a new vocabulary called "Genre" and add the following terms: Action, Comedy, Romance, Drama, Horror, Documentary, Sci-fi.
 - b. Create a new menu called "Favorite movies".
 - c. Place the block created by Drupal for the "Favorite movies" menu in the "Sidebar second" region. Set its visibility settings to only display to authenticated users.
 - d. Create a new content type called "Movie" (content type names should always be singular).
 - i. Add some relevant fields (including a term reference field to the "Genre" vocabulary to the "Movie" content type.
 - ii. Configure the "Movie" content type so that nodes of type "Movie" are allowed to be added to the "Favorite movies" menu.
 - iii. Add five new "Movie" nodes to your site - add at least two of them to the "Favorite movies" menu.
 - e. Create a new "movie editor" role.
 - i. Grant this role the ability to add/edit/delete all nodes of type "Movie".
 - ii. Grant this role the ability to use the administration menu.
 - iii. Grant this role the ability to use the administration theme.
 - iv. Grant this role any other permissions you feel are necessary for a "movie editor" to do their job successfully. Do not grant any unnecessary permissions that allow users with the "movie editor" role to perform tasks not specified above.
 - v. Create a new user, grant them the "movie editor" role and confirm that they can do their job.
 - f. Be prepared to demonstrate your work to the class!
4. Extra credit
 - a. Repeat the previous exercise using Drupal 8. When complete, continue with the steps below.
 - i. Create a new block type called "Movie theater".
 - ii. Remove the "Body" field from the "Movie theater" block type.
 - iii. Add fields for "Name", "Location", and "Number of screens". Use appropriate field types and configurations.
 - iv. Add two new custom blocks of type "Movie theater", use actual names of your local movie theaters. Place each block in a different region (maybe two different footer regions) and set each to be visible only on nodes of type "Movie".

5. Additional exercise (only complete if specifically requested)
 - a. Create a fresh Drupal site on your local machine.
 - b. You have a large, multinational donut/coffee shop chain as a client and they want to build a site that lists all of their locations. Design and build out a "location" content type, adding fields to describe a typical "location". Do not use any custom or contributed modules. Add several locations using the new content type. Put yourself in the shoes of the client to ensure that you capture as much relevant data as possible for each location. Extra time? Also design and build out content types for "coffee" and "donut".