

# Chapter 1

## Introduction

E-commerce has been around for 20 years. Since the release of the Netscape browser in 1994 companies have been selling their products online. From 1995 to 2000 the number of companies who were selling their products online increased dramatically. Venture capitalists invested huge amounts of money in these new internet companies. Sadly, the excitement wasn't well founded. The world wasn't ready for a shift from traditional retailing to e-commerce. People where wary about ordering products online, online payment methods weren't very widely excepted and parcel services weren't optimized for delivering the products to the people.

The excitement about internet companies led to the dotcom bubble bursting in 2000. A lot of new internet companies went bankrupt, others had to downsize significantly. A lot of survivors of the dotcom bubble are still around today. They have grow to be huge internet companies, some examples are: amazon.com, ebay.com and dell.com.

In the last 15 years online retailing has been steadily increasing. People where getting more comfortable buying their products online, online payment methods improved significantly and parcel services optimized their distribution network. These improvements give small companies the opportunity to start selling their products online. It is expected that the online retailing market will overtake traditional retailing within ten years.

### 1.1 What's in the course

In this course you will learn to create and manage a web store. There are a lot of platforms which allow you to create a web store. In this course we chose to use Drupal. Drupal is a content management system (CMS) written in php. It has many features that will make it easy to create a web store.

First we'll learn how to create, manage and edit a basic Drupal site. Afterwards we add the web store functionality through the use of modules. During the course you'll also learn how to write php code, manage your site through

**git**, collaborate on your site and deploy it to a server.

## 1.2 What's a CMS

Drupal is a content management system or CMS. The name is a good description of what it actually does. It's an application which makes it easy to manage content. The content can be of different types. For example, a blog can use a CMS. The type of content on the blog will be articles. The CMS makes it easy to create, edit, remove and manage articles. Another example where a CMS can be useful is when you create a **web store**. The type of content we use here are products. The CMS makes it easy to manage the products, for example: add a product, change the price or add a shipping method.

A lot of big website use a CMS as their background application. Examples include: [www.engadget.com](http://www.engadget.com), [www.puma.com](http://www.puma.com), [www.societegenerale.com/](http://www.societegenerale.com/).

Figure 1.1 shows different CMS systems.



Figure 1.1: Examples of CMS sytems

## 1.3 What's Drupal

As mentioned before, Drupal (logo figure 1.2) is an open source CMS written in **php**. Drupal is web based so it uses HTML, CSS and JavaScript as application front end. A Drupal application is completely customizable. We can change **how our site looks** by changing the HTML and CSS (Drupal calls this theming).

The application back end can be extended by adding our own code or code other people made available online (Drupal does this through its module system).

Drupal was created in 2001 by Dries Buytaert. He got his degree in computer science at the university of Antwerp and his phd at Ghent university. In 2007 he started the company Acquia which provides services to organisations using Drupal as their CMS. In 2009 Acquia helped with the relaunch of `whitehouse.gov`. Next to `whitehouse.gov` a lot of other sites use Drupal. At <https://www.drupal.com/showcases> you can find a list of different sites build with Drupal.



Figure 1.2: Drupal logo

## 1.4 Drupal 8

Even though Drupal 8 is still in beta we'll be using Drupal 8 to create our applications in this course. Drupal 8 is still in beta but it has been feature frozen so no new features will be added. The Drupal core developers are now just performing some bugfixes prior to the final release. We do not expect you to encounter any of these bugs but if you do, you should report them to the Drupal community.

## 1.5 Review exercises

1. **Lookup** the definition of a CMS on wikipedia.
2. Think of five content types that could be managed by a CMS.
3. Find five sites which use Drupal as CMS.
4. Name three other content management systems.

## Chapter 2

# Tools installation

### 2.1 Acquia Dev Desktop

The Acquia Dev Desktop provides an easy way to run a Drupal site on your local computer. The tool includes an AMP (Apache, MySQL, PHP) stack which allows your Drupal application to be executed. You can use it to create a new Drupal site and manage it.

#### 2.1.1 Installation

- Download the version of Acquia dev desktop for your operating system at <https://www.acquia.com/downloads>.
- When your download is complete, run the installer.
- Keep clicking Next, accept the licence agreement and select the installation location for dev desktop as well as the location where your Drupal sites will be stored. In this course we use the C:\devpal\_sites directory but you can change this to a directory of your preference. It is advised to choose a directory that is short and has no spaces.

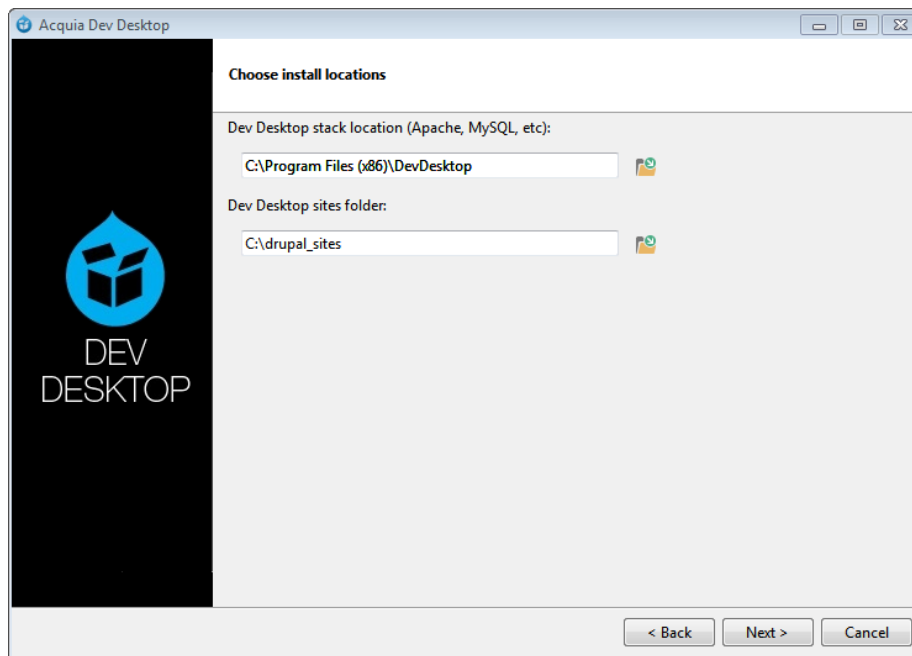


Figure 2.1: Installation location

- When the installer is complete you are ready to create your first site. Before we do that we'll install some useful tools that will help us during development.

## 2.2 NetBeans

When we are developing our Drupal site we'll write php, html and css code. In this course we use the Netbeans ide to do all this. You can download the [en](https://netbeans.org/downloads/) NetBeans editor at: <https://netbeans.org/downloads/>. You can choose to download the version for HTML5 an PHP or the ALL version.

### 2.2.1 installation

Run the NetBeans installer. You can use the default settings or change them to your liking.

### 2.2.2 configuration

Next we'll configure NetBeans to associate the Drupal file extensions to the correct file types. For example `.module` files should be opened as a php file. Go to **Tools > Preferences > Miscellaneous > Files** there you can add new file

extensions and associate the file type. Associate the following extensions with the *text/x-php5* MIME type:

- module
- install
- test
- profile
- theme
- engine

Associate the following extensions with the *text/plain* MIME type:

- info
- po

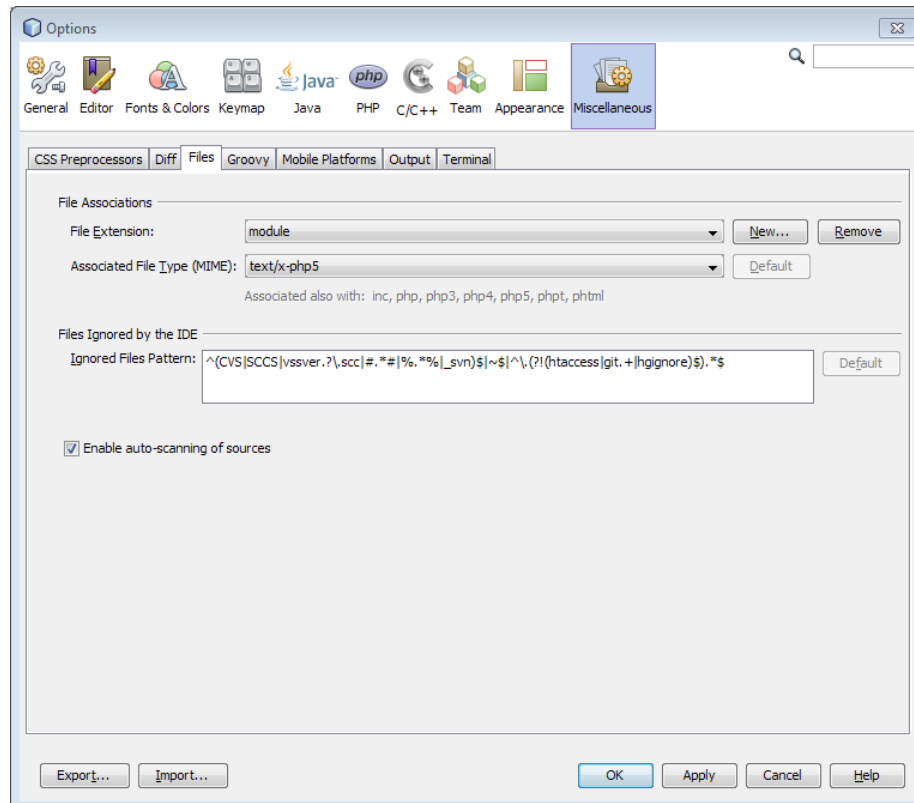


Figure 2.2: Extension configuration

## 2.3 WinLess

WinLess is a less compiler for windows. The less language makes it easier to write css code. In this course we assume you are familiar with css and less so we won't go into depth on how to write less code. To install the less compiler, download the latest version of WinLess from <http://winless.org/> and run the installer.

WinLess allows you to select a less file and automatically compile it to a css file in the directory you specify. (see Figure: 2.3)

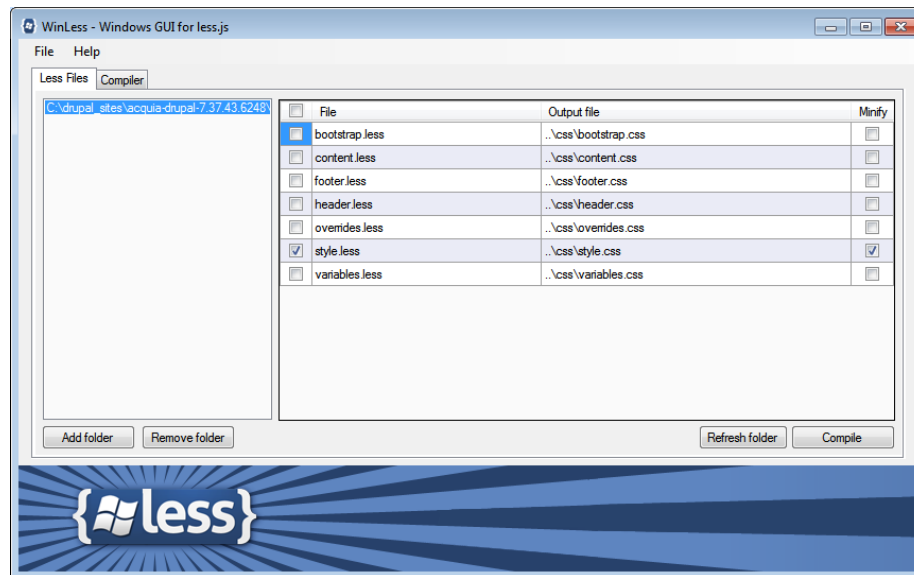


Figure 2.3: WinLess interface

## 2.4 Drush

Drush <sup>a</sup> a command line tool which makes it easy to manage your site. Since Drush is included in the Acquia dev desktop installation you don't need to install it separately. You can test your Drush installation by opening a command line window and typing *drush* and hitting enter.

## 2.5 GIT

GIT is a versioning system, it enables you to share and manage your code. You can download the git installer here: <https://git-scm.com/downloads>

### 2.5.1 Installation

- Run the installer.
- In the window *Adjusting your PATH environment* select the option **Use Git from the Windows Command Prompt**
- In the next window select **UseOpenSSH**
- Configure the line endings as **Checkout Windows-style, commit Unix-style line endings**
- Finish the installer wizard.



## Chapter 3

# Create your first Drupal site

### 3.1 Acquia Dev Desktop

As mentioned before we will be managing our local Drupal installation through the Acquia Dev Desktop tool. The tool contains a php hosting environment which includes: the php runtime, an Apache web server, a MySQL database server and phpMyAdmin for database management.

Figure 3.1 shows the Acquia Dev Desktop interface. On the left you can see an overview of the sites you created. The plus and minus signs in the bottom left corner allow you to add or remove a Drupal site. In the right panel you can see the information of the site you selected. It contains the following:

**Local site** The local url to your site. Refers to the local Apache server running on port 8083.

**Local code** Shows the path to the local codebase. All the code your site uses is stored in this location on your computer.

**Local database** A link to the phpMyAdmin page of your local database.

**PHP version** The PHP version your site uses, we will use the default version (5.5.27).

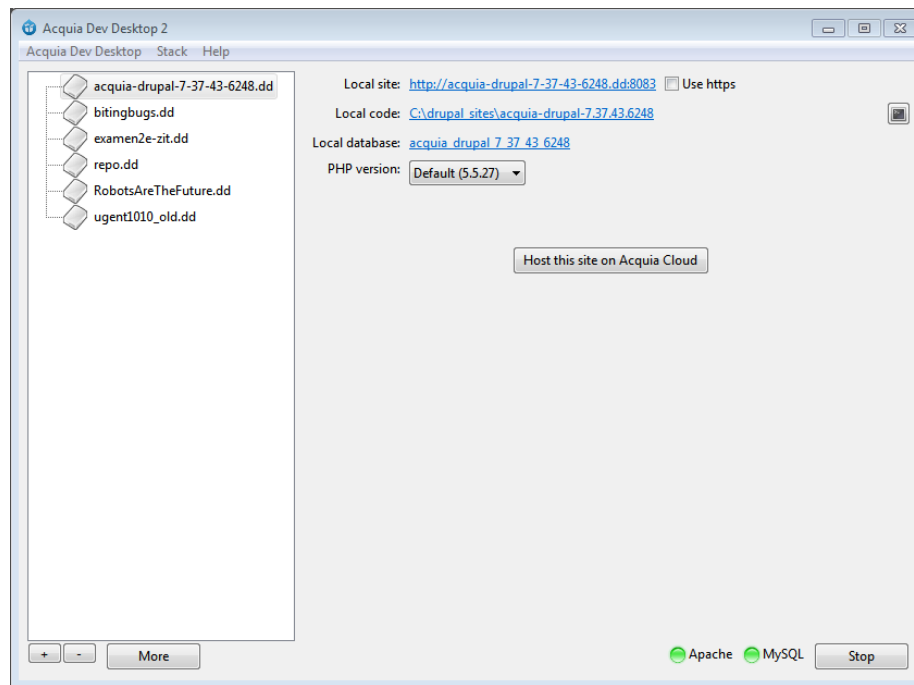


Figure 3.1: The Acquia Dev Desktop interface

## 3.2 Creating a new site

To create a new site through Acquia Dev Desktop, click the plus button in the bottom left corner. Select *New Drupal site...* in the context menu (Figure 3.2)

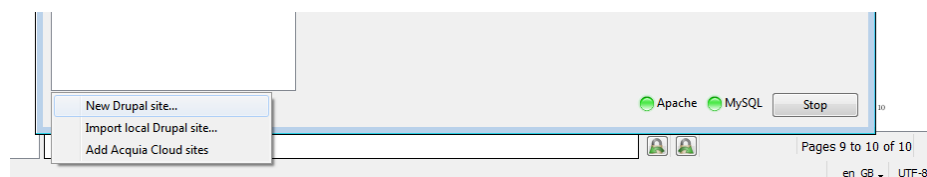


Figure 3.2: New site menu

In the next window you will see an overview of different standard Drupal installations. Select the Drupal 8 version (Figure 3.3)



Figure 3.3: Drupal version selection

Fill out the following site information in the next window (Figure 3.4). Note that if you have configured Acquia Dev Desktop to store your sites in a different location the default path might not be `C:\drupal_sites\`.

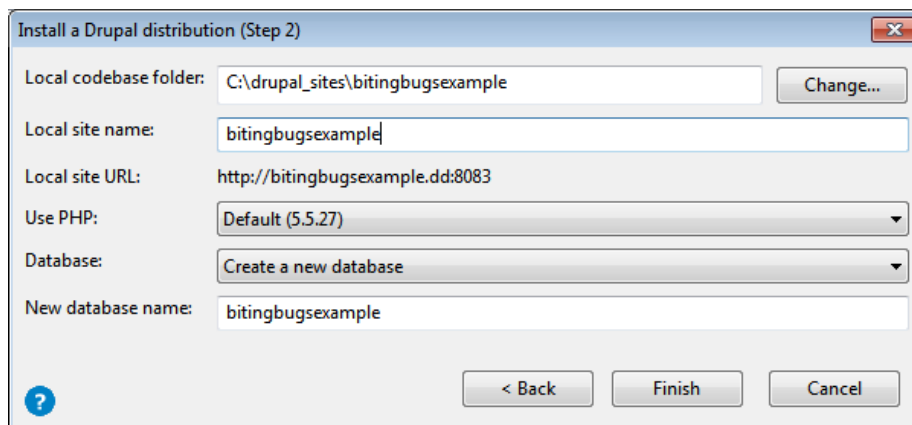


Figure 3.4: Site configuration

Click *Finish*. Acquia Dev Desktop unpacks the selected Drupal 8 archive into the local codebase folder. When the process finishes click the link to your local site in the right panel of the Acquia Dev Desktop, this will open the installation page for your site in your default browser (Figure 3.5).

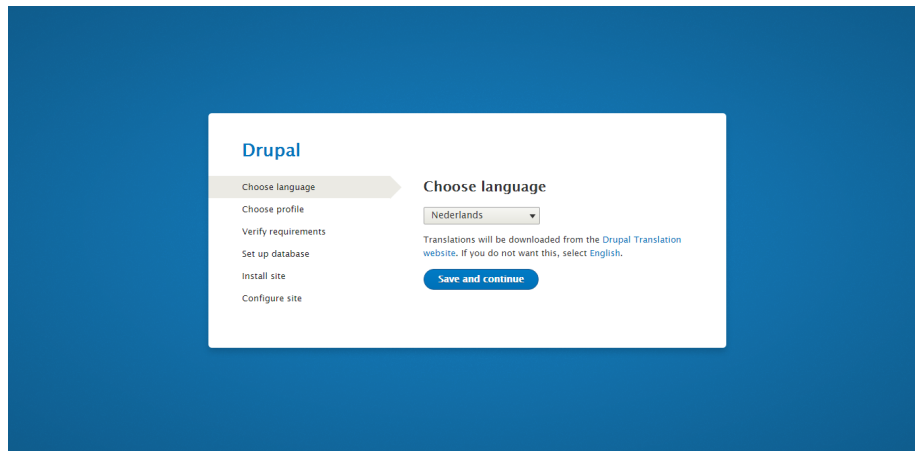


Figure 3.5: Site installation: Choose language

Since this course is in English we'll select it as our default site language. Click *Save* and on the next page select the *Standard* installation profile (Figure 3.6).

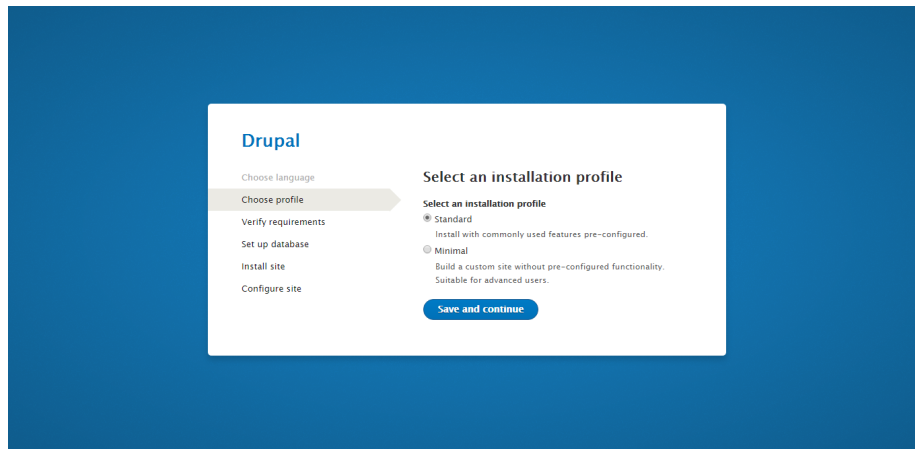


Figure 3.6: Site installation: installation profile

Click *Save and continue*. This step might take a few minutes. After the installation has finished we need to configure some site properties. Enter the following information (Figure 3.7 and 3.8):

**Site name:** bitingbugsexample

**Site email address:** noreply@bitingbugsexample.com

**Username:** drupal

**Password:** Drupal

**Confirm password:** Drupal

**Email address:** info@bitingbugsexample.com

**Default country:** Belgium

**Default time zone:** Europe/Brussels

After entering the right configuration you can click the *Save and continue* button.

Drupal

- Choose language
- Choose profile
- Verify requirements
- Set up database
- Install site
- Configure site

### Configure site

#### SITE INFORMATION

**Site name \***  
bitingbugsexample

**Site email address \***  
noreply@bitingbugsexample.com

Automated emails, such as registration information, will be sent from this address. Use an address ending in your site's domain to help prevent these emails from being flagged as spam.

#### SITE MAINTENANCE ACCOUNT

**Username \***  
drupal

Spaces are allowed, punctuation is not allowed except for periods, hyphens, and underscores.

**Password \***  
.....

■ Password strength: Weak

**Confirm password \***  
.....

Passwords match: yes

Figure 3.7: Site installation: installation configuration 1

\* Make it different from your username

**Email address \***  
info@bitingbugsexample.com

#### REGIONAL SETTINGS

**Default country**  
Belgium

Select the default country for the site.

**Default time zone**  
Europe/Brussels

By default, dates in this site will be displayed in the chosen time zone.

#### UPDATE NOTIFICATIONS

**Update notifications**

☒ Check for updates automatically

☐ Receive email notifications

The system will notify you when updates and important security releases are available for installed components. Anonymous information about your site is sent to [Drupal.org](https://drupal.org).

**Save and continue**

Figure 3.8: Site installation: installation configuration 2

Et voila, you have your first Drupal site. It's pretty basic, you only have a welcome page, no content yet. By default, after completing the installation, you are logged in as administrator (figure 3.9). To see the non administrator layout click the *Log out* link in the top right corner. As you can see we only have a basic one page site with our site title at the top. Now log back into your site with username: drupal and password: Drupal.

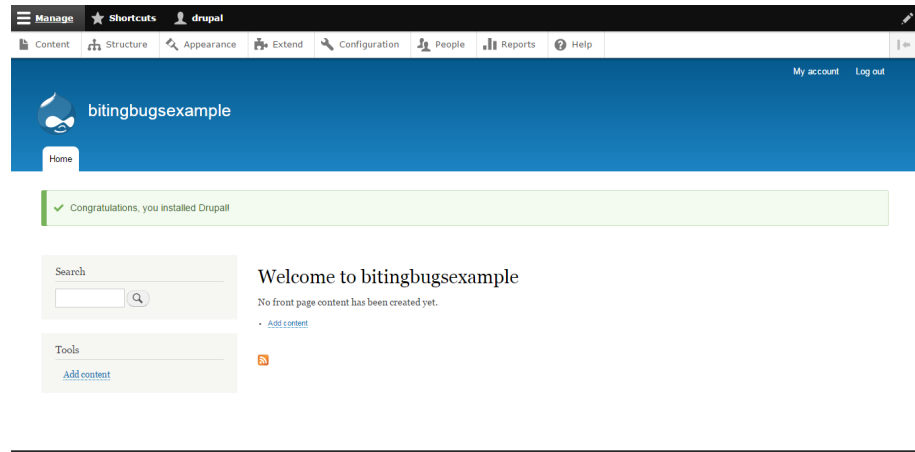


Figure 3.9: Your first Drupal site.

### 3.3 review exercises

Create a new Drupal 8 site with the following properties:

- Sitename: exploringdrupal8
- database: exploringdrupal8
- Language: English
- Username: drupal, password: Drupal
- Timezone: Europe/Brussels
- Country: Belgium

## Chapter 4

# The Drupal structure

### 4.1 Structural elements

Drupal 8 core defines eight types of structural elements. To see an overview of these elements click on the *Structure* button in the administrative menu (Figure 4.1).

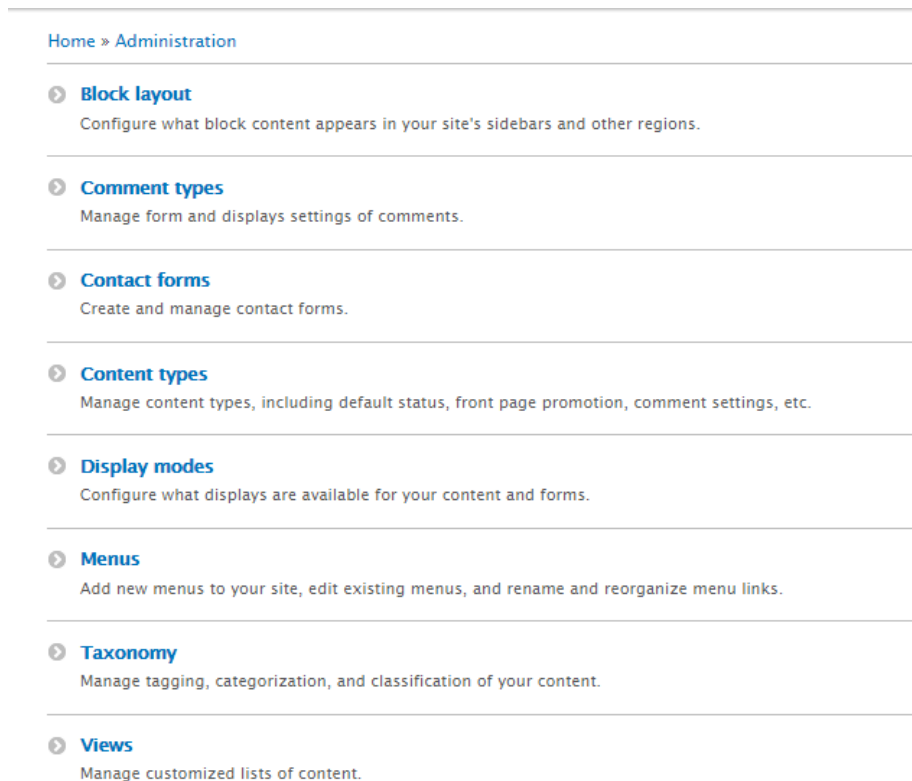


Figure 4.1: Structural Elements

Each of these elements has a certain use within Drupal. The following list describes each of the structural elements. Don't worry if not all the elements are clear to you, when you start changing and using your site the structure will become clear.

**Block layout:** Blocks are site elements which can be positioned in different places on your site. For example the search field, which is visible on the left side of your first Drupal page, is a Drupal block. You can put a block in different places on your page, these places are called **regions**. These regions depend on the Drupal theme you are using. When you click the *Block layout* link you will see the following page (Figure 4.2).



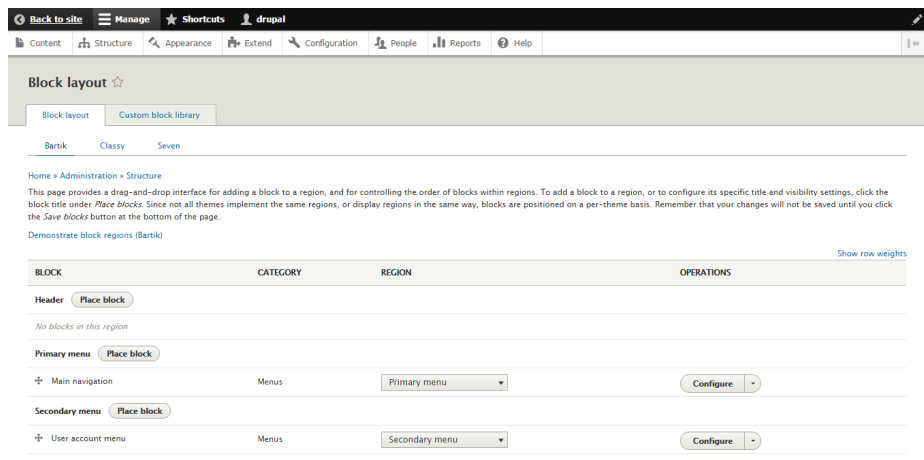


Figure 4.2: Block layout

This page shows a list of the different regions and allows you to add blocks to each of these regions. You can click the link *Demonstrate block regions(Bartik)* to view the regions of the current theme (Bartik) (Figure 4.3).

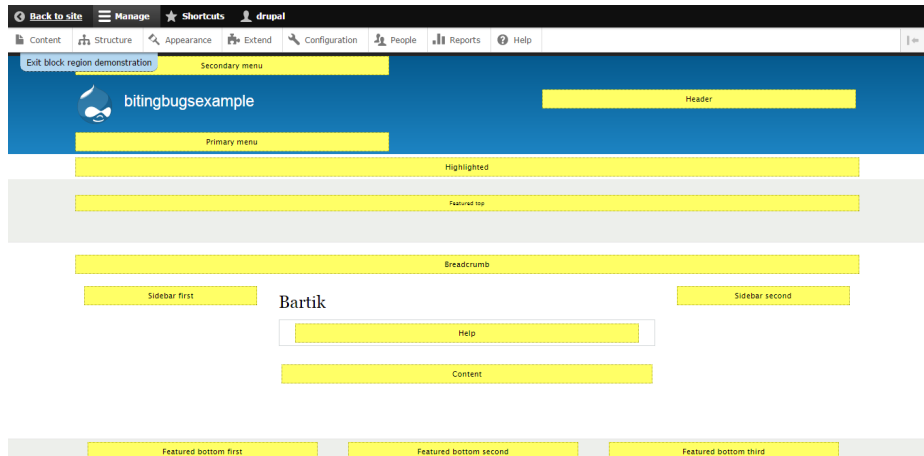


Figure 4.3: Bartik regions

**Comment types** The comment types page allows you to create and manage different comment types. When you add content to your site you can allow people to comment on the new content. The comment type defines the fields that the commenter has to fill out when he's writing his comment. The default comment type has only one field: the comment body. We

could, for example, create a new comment type which includes a field for the name and age of the commenter.

**Contact form** The Personal contact form is the form for site visitors to contact registered users; the name and recipients of this form cannot be edited. Other forms listed here are your configured site-wide contact forms, which site visitors can use to send mail to a centralized email address or addresses. You can edit the name and recipients of site-wide forms by choosing the Edit operation. You can also configure the fields and display of both personal and site-wide forms.

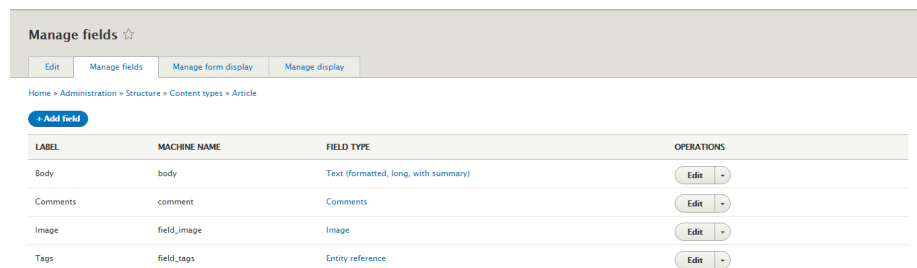
**Content types** The content types page is very important. The content types define which kind of information your CMS will manage. A content type has different **field**. These fields define the information that is stored in the content type. By default Drupal has two content types: Article and Basic page (Figure 4.4);



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

Figure 4.4: Default content types

When you click the *Manage fields* button on the right you can see what kind of information is stored in this content type. (Figure 4.5). As you can see, the Article content type has four fields: Body, Comments, Image and Tags.



LABEL	MACHINE NAME	FIELD TYPE	OPERATIONS
Body	body	Text (formatted, long, with summary)	<a href="#">Edit</a>
Comments	comment	Comments	<a href="#">Edit</a>
Image	field_image	Image	<a href="#">Edit</a>
Tags	field_tags	Entity reference	<a href="#">Edit</a>

Figure 4.5: Article content type fields

Next to the *Manage fields* menu item tab you have the *Manage form display* and *Manage display* tabs. These allow you to edit which fields are displayed when an element is created/edited or viewed.

**Display modes** Display modes define different ways in which information is displayed. There are two types of display modes: form modes and view modes. Form modes are used when the content is created or edited, view modes when the content is viewed. When you go to *Display modes* → *View modes* (Figure 4.6) you will see the different ways in which a content type can be displayed.

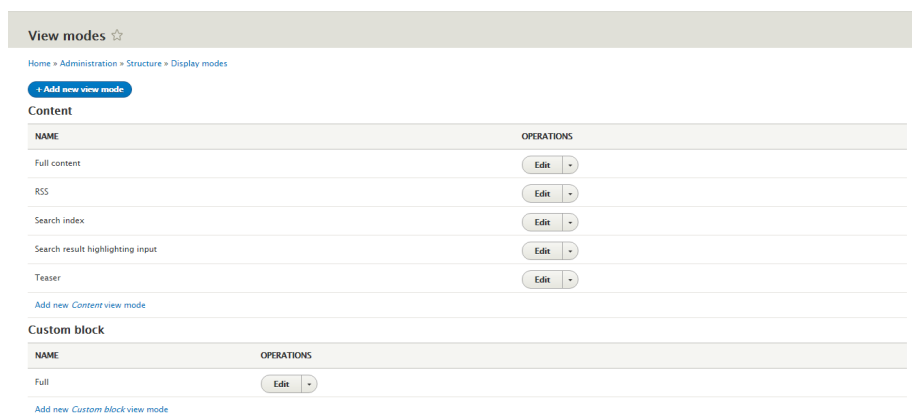


Figure 4.6: Default view modes

When you go to *Structure* → *Content types* → *Article* → *Manage display* you will see the following page:

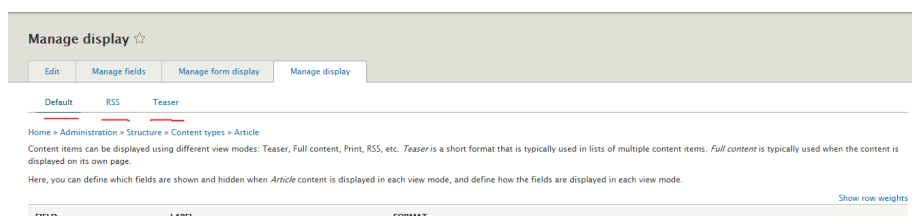


Figure 4.7: Display modes

There you see the different display modes that you can use for your *Article* content type. At the bottom of the page you can enable other display modes in the custom display settings dropdown (Figure 4.8).

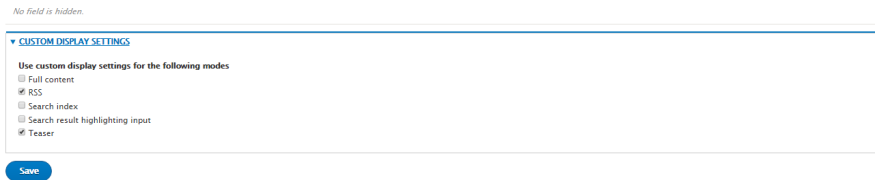


Figure 4.8: Custom display settings

**Menus** Menus are easy, they define a menu with different menu items. Each menu has a corresponding block that is managed on the Block layout page.

**Taxonomy** A taxonomy defines **a lists of terms**, these terms can be associated with the content. A list of terms is called a vocabulary. For example: if we have a site with news articles about sports we could add a tag to each article to tell what the article is **about. these** tags can be defined in a vocabulary.

**Views** A view enables you to create a display based on different content types. This course has a **hole** chapter dedicated to Drupal Views so we won't go into details here.

## 4.2 Bitingbugs example

In the following sections we will review the content of this chapter by applying it to an example website. In the previous chapter we created the bitingbug-**sexample** website through the Acquia Dev Desktop. In this and the following chapters we will keep adding features to this site. Our goal is to create a web-store for selling edible insects. The site will also provide a database with recipes so people know how to cook with the insects.

### 4.2.1 Change the site title and logo

To make our site a little bit prettier we will change the logo and title. To change the site title go to **Configuration** → **Site information**. There change the site name to *Biting Bugs* (Figure 4.9).

Site information ☆

Home » Administration » Configuration » System

▼ SITE DETAILS

**Site name \***  
Biting Bugs

**Slogan**

How this is used depends on your site's theme.

**Email address \***  
noreply@bitingbugs.com

The From address in automated emails sent during registration and new password requests, and other notifications. (Use an address ending in your site's domain to help prevent this email being flagged as spam.)

Figure 4.9: Changing the site title

The logo is part of the Drupal theme you are using. To change the logo go to: **Appearance** → **Settings** → **Logo image settings**. Uncheck *Use the default logo supplied by the theme* and upload the file `bitingbugs_logo_transp_white_right_small.png` (Available in the course files zip). Click **Save configuration**.

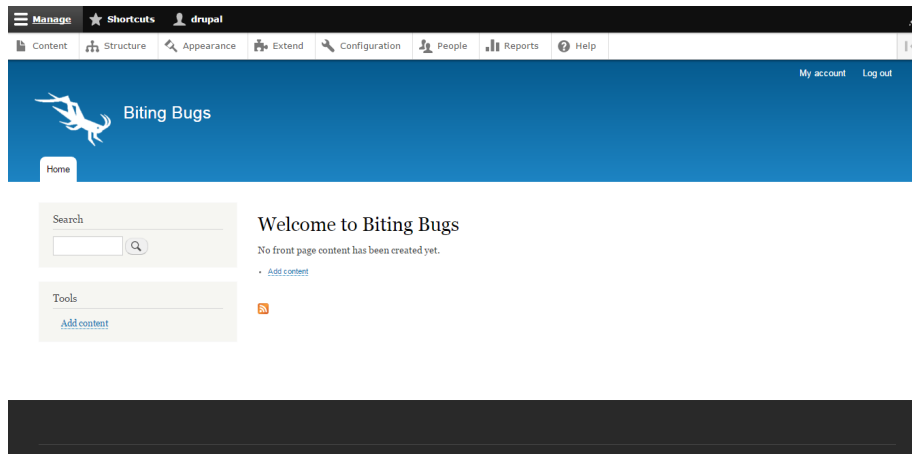


Figure 4.10: Changed logo and title

### 4.2.2 Removing the Search and Tools block

On the right side of the page we have the *Search* and *Tools* block. We don't need them for now so you can remove them by going to **Structure** → **Block layout** and moving them from the *Sitebar first* to the *None* region (Figure 4.11). Click **Save blocks**.

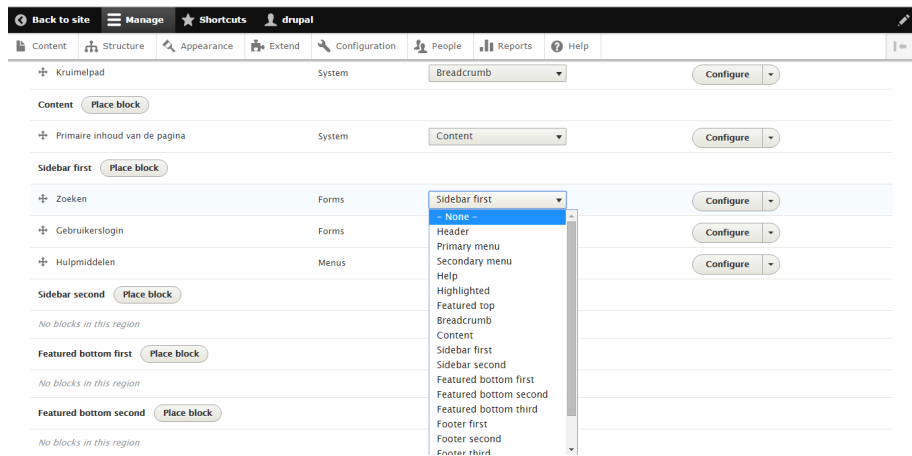


Figure 4.11: Remove a block from a region

### 4.2.3 Adding a Taxonomy

To categorise the recipes on our site we will add a taxonomy. This taxonomy will contain different types of dishes. Go to **Structure** → **Taxonomy** → **Add vocabulary**. Use the following settings:

**Name:** Dish types

**Description:** Describes the type of dish we will add.

**Vocabulary language** English

**Default language** Site's default language (English)

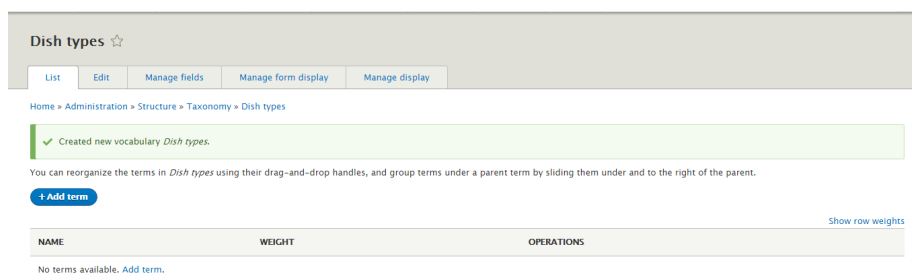


Figure 4.12: Adding a vocabulary terms

Click **Save**. In Figure 4.12 you can see the empty vocabulary. Next we will add some terms to the vocabulary. Click the **Add term** button and add the following terms:

- candy
- curry
- dessert
- pasta
- salad
- sandwiches
- sauces
- vegetarian
- vegan

Add term ☆

Home » Administration » Structure » Taxonomy » Dish types

✓ Created new term *dessert*.

Name \*  
pasta

The term name.

Description

B I [link icon] [list icon] [source icon]

Text format Basic HTML

About text formats ?

Figure 4.13: Adding a vocabulary term

To see an overview of the terms you have added go to **Structure** → **Taxonomy** and click the **List items** button next to your vocabulary. (Figure 4.14)

[Show row weights](#)

NAME	OPERATIONS
+ Candy	Edit ▾
+ curry	Edit ▾
+ dessert	Edit ▾
+ pasta	Edit ▾
+ salad	Edit ▾
+ sandwiches	Edit ▾
+ sauces	Edit ▾
+ vegan	Edit ▾
+ vegetarian	Edit ▾

Figure 4.14: Vocabulary terms

#### 4.2.4 Adding the *Recipe* content type

Since we are going to store recipes in our CMS we will need to add the *Recipe* content type. Go to: **Structure** → **Content types** → **Add content type**. Give it the following properties:

**Name:** Recipe

**Description:** A recipe for cooking bugs!

At the bottom of the page you can see some **setting** for this content type. Explore the settings, the names are very descriptive so most of them should be clear without further explanation. These are general settings that apply to all instances of this content type. You are able to change these for each instance individually when you create them.

Click **Save and manage fields**.

Manage fields ☆

Edit
Manage fields
Manage form display
Manage display

Home » Administration » Structure » Content types » Recipe

✓ The content type *Recipe* has been added.

+ Add field

LABEL	MACHINE NAME	FIELD TYPE	OPERATIONS
Body	body	Text (formatted, long, with summary)	Edit ▾

Figure 4.15: Manage recipe fields

Add the following fields to the *Recipe* content type:

- Name (Text/plain, Maximum length = 255, number of values = 1)



- Ingredients (Text/plain, Maximum length = 255, number of values = unlimited)
- Body (already there)
- Plate image (Image, number of values = 1)
- Estimated time (Number(decimal), number of values = 1, Help text = Estimated time to cook the recipe in minutes).
- Type (Taxonomy term, number of values = unlimited, reference method = default, Vocabularies: Dish types, Create referenced entities if they don't already exist).

In figure 4.16 you can see an overview of the fields.

✓ Saved Type configuration.

+ Add field

LABEL	MACHINE NAME	FIELD TYPE	OPERATIONS
Body	body	Text (formatted, long, with summary)	<div>Edit</div>
Estimated time	field_estimated	Number (decimal)	<div>Edit</div>
Ingredients	field_ingredients	Text (plain)	<div>Edit</div>
Name	field_name	Text (plain)	<div>Edit</div>
Plate image	field_plate_image	Image	<div>Edit</div>
Type	field_type	Entity reference	<div>Edit</div>

Figure 4.16: Recipe fields

## 4.3 review exercises

1. Log in to your *exploringdrupal8* site (created in the previous chapter). Add the search block to the *sidebar second* region and disable the *powered by Drupal* block.
2. Add a comment type **Answer** to your *exploringdrupal8* site. We will use the comment type to allow users to answer a question. The comment type has two fields: answer (a number between 0 and 100000) and motivation (a textual explanation describing how they got the answer).
3. Add a new content type **recipe** to your *exploringdrupal8* site. The new content type has the following fields: Title, PlateImage (Image), ingredients (list), description. Make sure the teaser only displays the Title and Image fields.
4. Add a vocabulary **Food types** to your *exploringdrupal8* site. Add the following terms to the vocabulary: Indian, Chinese, vegetarian, vegan.

## Chapter 5

# Creating content

### 5.1 Creating a basic page

In the previous chapter we learned how to create a content type. In this chapter we will learn how to create the content itself. An instance of a content type (a.k.a. a piece of content) is called a **Node** in Drupal. Creating content is easy, just go to **Content** → **Add content**. On this page you will see an overview of the different content types (Figure 5.1)



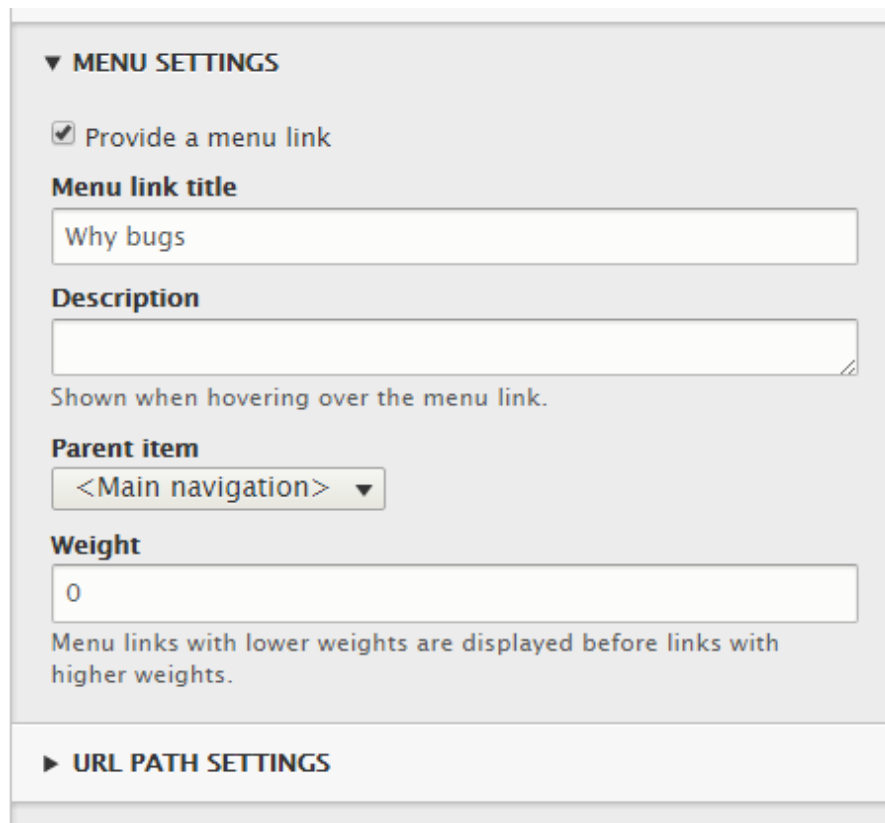
Figure 5.1: Add content, overview content types

We are going to add a basic page. This page will explain why it is good to eat bugs. Go to **Content** → **Add content** → **Basic page**. Use the following settings:

Title Why bugs

Body See file `chapter5/reasonstoeatbugs.txt` included in the course attachments.

On the right side you have several menu items. These allow you to change your posts properties. Under **Menu settings** add a menu link to the *Main navigation* (figure 5.2)



▼ MENU SETTINGS

☒ Provide a menu link

**Menu link title**

Why bugs

**Description**

Shown when hovering over the menu link.

**Parent item**

<Main navigation> ▼

**Weight**

0

Menu links with lower weights are displayed before links with higher weights.

► URL PATH SETTINGS

Figure 5.2: Basic page publishing options

Click **Save and publish**

## 5.2 Creating a recipe

Go to **Content** → **Add content** → **Recipe**. Figure ?? shows the editing page for our recipe. As you can see we have a *Name* and *Title* element. We don't need **te** title element. Remove it by going to: **Structure** → **Content types**

→ **Recipe** → **Manage fields** → **Manage form display**. Move the edit fields like in figure 5.3

FIELD	WIDGET
✚ Name*	Textfield ▾
✚ Ingredients*	Textfield ▾
✚ Body	Text area with a summary ▾
✚ Plate image	Image ▾
✚ Estimated time	Text field ▾
✚ Type	Autocomplete ▾
✚ Promoted to front page	Check boxes/radio buttons ▾
<b>Disabled</b>	
✚ Sticky at top of lists	- Hidden - ▾
✚ Authored on	- Hidden - ▾
✚ Authored by	- Hidden - ▾
✚ URL alias	- Hidden - ▾
✚ Title	- Hidden - ▾
✚ Language	- Hidden - ▾

Figure 5.3: Manage recipe form display.

Now go back to the content creation page: **Content** → **Add content** → **Recipe**. Use the following information:

Name: Dry Roasted Crickets

Ingredients 25 50 live crickets

Body See file `recipe_roasted_crickets.txt`

Image See file `roasted_crickets.jpg`

Alternative text Plate of bugs

Estimated time 10

Type snack

Click **Save and publish**. You should see a page like figure 5.4.

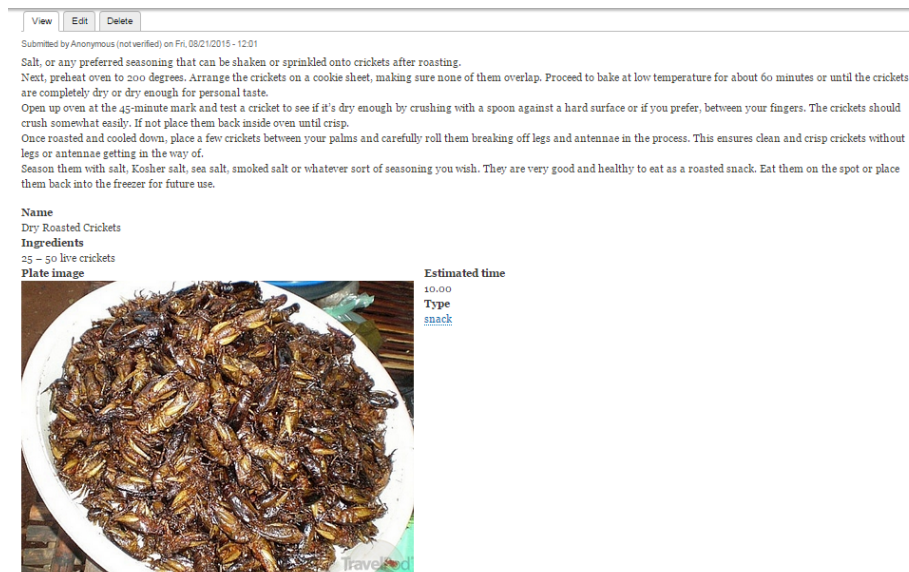


Figure 5.4: Roasted crickets recipe.

We would like to change the way that our recipe is displayed. To do this go to: **Structure** → **Content types** → **Recipe** → **Manage display**. Change it to look like figure 5.5

FIELD	LABEL	FORMAT	
⊕ Name	- Hidden -	Plain text	⚙
⊕ Ingredients*	Above	Plain text	⚙
⊕ Body	- Hidden -	Default	
⊕ Plate image	- Hidden -	Image	Original image ⚙
⊕ Estimated time	Above	Default	1234:12 Display with prefix and suffix. ⚙
⊕ Type	Above	Label	Link to the referenced entity ⚙
<b>Disabled</b>			
⊕ Links		- Hidden -	
⊕ Language	Above	- Hidden -	
▶ CUSTOM DISPLAY SETTINGS			

Figure 5.5: Manage recipe display

Click **Save**. The new recipe page should look like figure 5.6

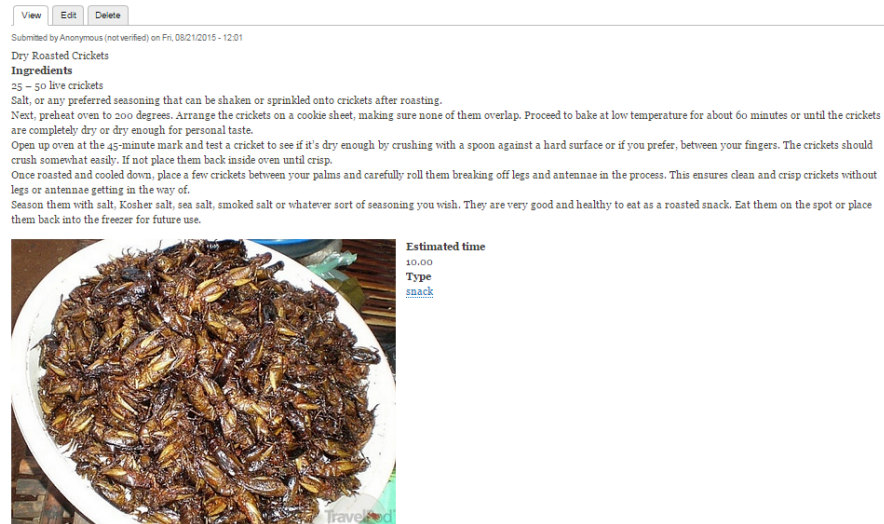


Figure 5.6: Recipe roasted crickets

To be honest, it looks a bit better **then** before but still not great. In the chapter about theming we will learn how to further change the layout by adding custom CSS.

## 5.3 Review exercises

1. Go to the homepage of the Biting Bugs site. You should see a summary of the recipe we added earlier in this chapter (Figure 5.7).

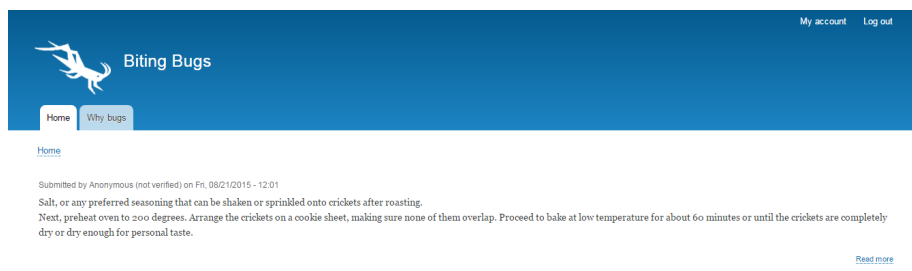


Figure 5.7: Recipe teaser

Change the display settings of the recipe content type teaser display mode so it looks like figure 5.8. Make sure that when you click the image you go to the full recipe.

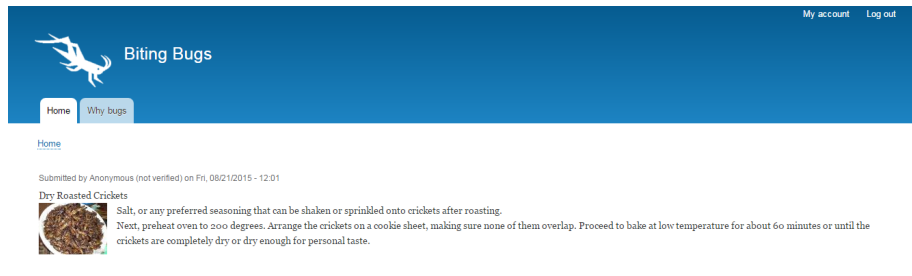


Figure 5.8: Recipe teaser updated

HINT: You can edit the image size by clicking on the gear next **ot** the image field.

2. Add three recipes to the bitingbugs site. You can find the recipes in the file `chapter5/recipes_with_bugs.txt`. Use the images: `cricket-flour.jpg`, `cricket_fried_rice.jpg` and `cricket_pad_thai.jpg`.
3. Logout of the bitingbugs site. On the homepage you can see a login form on the left side of the page. Make sure the login form does not appear on the homepage. (HINT: after removing the login form from the homepage you can always login by going to *site hostname/user* for example `bitingbugs.dd:8083/user`)
4. Change the display settings for the recipe teaser title so it links to the content.
5. Change the Recipe content type so the author information is not displayed.
6. Add a new basic page to the main menu. The basic page contains the information found in file `chapter5/edible_bugs.txt` and has the **tile** *Edible bugs*. (Put the bug names in bold font.)

## Chapter 6

# Drupal Views



## Chapter 7

# Collaborating on a Drupal site

Throwing your codebase into Git or an other versioning system will not be enough to enable you to collaborate on your Drupal site. Since Drupal uses a database to store its contents this database should be shared as well. In this course we recommend using the OpenShift cloud for hosting your site. OpenShift is a platform as a service product from Red Hat. The software that runs the service is open-sourced under the name OpenShift Origin, and is available on GitHub. Developers can use Git to deploy web applications in different languages on the platform. A version for cloud computing is named OpenShift Enterprise. OpenShift also supports binary programs that are web applications, so long as they can run on Red Hat Enterprise Linux. This allows the use of arbitrary languages and frameworks. OpenShift takes care of maintaining the services underlying the application and scaling the application as needed.

### 7.1 Creating an openshift account

- First go to <https://www.openshift.com/>.
- Click on **Sign Up**.
- Fill out your information.
- Click **Sign Up**
- Verify your account
- Accept the terms of service
- Click **Create your fist application now**

OpenShift gives you the option to create an instant Drupal 7 app. Since we are using Drupal 8 we will not use this option. We are going to create a PHP 5.4 application. This is a Linux server running a PHP web hosting environment. Click the PHP 5.4 application link and enter the following information:

Public url: `drupalsite-[firstname][lastname].rhcloud.com`

Source code: leave empty

Scaling: No scaling

Region: `aws-us-east-1`

Click the **Create application** button.

When OpenShift asks if you will be changing the code for the application, click **Not now, continue**. You will see a page like figure 7.1.

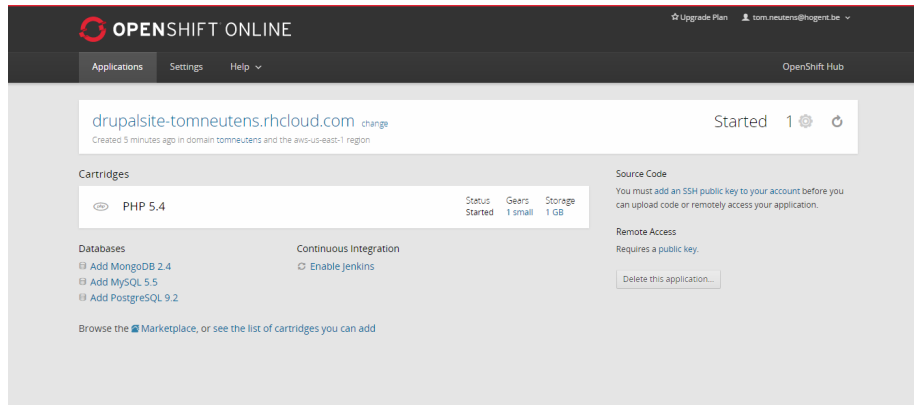


Figure 7.1: OpenShift application overview page.

Since we are going to use a MySQL database, click the **Add MySQL 5.5** link. On the next page click **Add Cartridge**. Make note of the database credentials, it's a good idea to take a screenshot! After adding the MySQL cartridge you should also add the PHPMyAdmin cartridge to manage the database.

## 7.2 Installing the OpenShift client tools

To be able to manage your OpenShift site you have to install some tools. The following link (<https://developers.openshift.com/en/managing-client-tools.html>) gives step by step instructions on how to install them.

## 7.3 Putting your site on openshift