Day 01 - Configuration Management in D8

In this course today, we will be dealing with the basics of Configuration Management in D8. We will not be talking about Features but about the CM capabilities provided in core, which would be required on any D8 project that we would work with.

Objective

At the end of this course, you will be able to

- export configuration from a local or Dev environment
- transfer the same by copying the files or through version control to another environment

Thereby, moving version controlled configuration across various environments.

Primary Tutorial

https://www.youtube.com/watch?v=-c-G53dTGy4

Exercise

1. Setup 2 local D8 instances. Call one site - Dev and the other - Prod (Clone the dev site to prod, rather than setting them up individually. Research issues with mismatching UUIDs)
2. Create a simple content type (say Book) with 2 fields (Title, ISBN) and create 2 nodes of content on Dev
3. Build a simple view (of page type, accessible at the url */list)* listing the book nodes
4. Export the site configuration using Drush (which includes the content type and the view) using *drush cex* command or Drupal Console using *drupal ce*
5. Copy the exported files to same directory on Prod site simulating a code pull from upstream repo.
6. Import the configuration using *drush cim* on prod site or *drupal ci*. Verify the content type and view are available on Prod.

   Estimated Time : 30min

Additional Resources

- **https://www.drupal.org/documentation/administer/config**
- **Using Drupal UI**
- **Using Drush**
- **Changing the Sync Directory Location**

Day 02 - Paragraphs Module

In this course today, we will be trying out the Paragraphs Module.

Objective

At the end of this course, you will be able to

- Create paragraph types
- Use paragraphs in content types to display grouped fields appropriately

Primary Tutorial

- https://www.youtube.com/watch?v=YqtCv7KEUis

Exercise

1. Create a Paragraph type - "Social Media" with the following fields - 1) Embed Code (Text field that accepts any long text and is rendered without any filters (Hint: Create a text format called "Raw") and 2) Link
2. On the Article Content type - Add a new field referring to the above paragraph type created (Multiple values allowed).
3. Configure display of the paragraph type and the article content type such that the embed code and link are displayed as on the screengrab attached.

Screengrabs:
https://www.evernote.com/l/ASmw_AMKx2dFaqaeh3jyTA4icDft3dhvopM

Estimated Time : 30min

Additional Resources

- https://www.drupal.org/node/2444893
- http://drupal.stackexchange.com/questions/181991/how-to-get-the-type-of-paragraphs-entity-in-the-parent-node-field

Bonus Exercise

- Theme the paragraph type such that the 2 fields (Embed code and link, wherever they are shown on any content type, are rendered in 2 columns as below:
  *<table><tr><td>**Embed Code**</td><td>**Link**</td></tr></table>*

Day 03 - Building Configuration forms

In this course today, we will be looking at building a very basic configuration form on a custom page.

Objective

At the end of this course, you will be able to

- Create a very basic configuration form on a custom page
- Provide Default values for the configuration on the form
- Save the configuration values on form submission.

Primary Tutorial

- Creating a Custom Page: https://www.drupal.org/node/2282025
- Defining and using your own configuration in Drupal 8: https://www.drupal.org/node/2116839
- Providing Default configuration: https://www.drupal.org/node/2087879
- Creating a custom Form: http://www.trellon.com/content/blog/how-create-custom-form-in-drupal-8
- Simple Config API: https://www.drupal.org/node/1809490

Exercise

1. Create a custom module that provides a configuration form available at the url “mymodule/config” to all users with the permission “administer content”
2. The form shall have 3 fields. Labels and Types of fields are arbitrary. Ensure you have diverse fields - Text, Select, Radio
3. Get the form to work such that the form values are saved and persisted on the form on reload.
4. The values submitted on the form should be accessible elsewhere on the same or a different module using the Config API.

Estimated Time : 30min

Additional Resources

- NA

Bonus Exercise

- Create your own permission and use that to restrict this page

Day 04 - Migration 101

So Migrate API is now in core. Let's attempt a simple exercise to migrate a set of MySQL tables to Drupal Content.

Objective

At the end of this course, you will be able to

- Import content from MySQL/CSV dump to content types, taxonomies in Drupal.

Primary Tutorial

- http://www.sitepoint.com/your-first-drupal-8-migration/
- http://webikon.com/cases/migrating-to-drupal-8#disclaimer (Could be outdated)

Exercise

1. Create a movie content type with fields - Title (text), Plot(Formatted Text), Actors (Node Reference), Genre(Term Reference)
2. Actors and Genres are simple content type and vocab respectively with no additional fields
3. Import the CSV dumps from **here** to populate the movie nodes in Drupal using Migrate API.

Preview Content: Movies, Actors

Estimated Time : 90m

Additional Resources

- Using CSV - https://www.drupal.org/node/2574707
- Process Plugins - https://www.drupal.org/node/2129651
- https://github.com/amitgoyal/migrate_d6_metatag_custom/blob/master/config/install/migrate.migration.d6_node__page_custom.yml
- https://www.drupal.org/project/migrate_source_csv
- https://github.com/heddn/d8_custom_migrate
- https://www.drupal.org/project/migrate_tools
- Explode Process - https://www.drupal.org/node/2674504
- https://codedrop.com.au/blog/writing-custom-process-plugins-using-drupal-8-migrate-api

Bonus Exercise

- Let's try some preprocessing while migrating. In the plot field of the movie, remove the word "the" (wherever it occurs, irrespective of the letter case).
- Add an Image field to the Movie content type. Update the migration script to use the dump file with images. Re-run the migration such that the images are downloaded and attached to the movie nodes.

Day 05 - Drupal 8 Block System

In this session, we will take a look at the basics of block system in D8.

Objective

At the end of this exercise, you will be able to

- Create "Block Types" with fields
- Place Multiple Instances of the block type with different field values
- Programmatically update block instance values / content

Primary Tutorial

- https://drupalize.me/blog/201403/block-system-finally-useful-drupal-8

Exercise

- Create a Block type called **"Stock Exchange Rate Card"**
  - Company Name
  - Symbol
  - Last Price
  - Change
- Create Instances of the block with different sets of values and place them at different spots on the site. Example values - ***(Apple, AAPL), (Bank of America,BAC), (Transocean, RIG), (Freeport, FCX)***. (Ignore the Last Price and Change fields' values, we will be dealing them in next step).
- Below API can be used to retrieve Last Price and Change values [http://dev.markitondemand.com/MODApis/Api/v2/Quote/jsonp?symbol=**BAC**&callback=myFunction](http://dev.markitondemand.com/MODApis/Api/v2/Quote/jsonp?symbol=BAC&callback=myFunction)
- Build a custom module, which on Cron run,
  - Iterates through each instance of blocks of type "Stock Exchange Price Card"
  - For each block, take the symbol value and call the API to retrieve the Last Price and Change values
  - Update the block field values programmatically with the values of Last Price and Change retrieved from the API and save the block

Day 06 - Services and Dependency Injection

Objective

At the end of this exercise, you ~~will~~ might be able to understand

- What is dependency injection
- Building Services and why build services?
- Core Services

Primary Tutorial

- https://ffwagency.com/blog/drupal-8-services-dependency-injection-and-decoupling-your-code

Exercise

This is not a Drupal API to have an exercise for itself. But a way of writing code. And if we learn it right all further exercises would themselves be an exercise for this topic. So take some additional time in going through this video series instead of an exercise dedicated for this topic. I have done this series and found it to be the best one I could come across in explaining the concepts of Dependency Injection and Services to anyone who is new to Object Oriented Programming!

Don't be scared by the long list below. All of them are 3 minutes or shorter. And d.me allows you to watch the videos at 1.5x speed which is probably the speed at which this group will watch most of their videos at. The whole playlist should not take more than 20 mins.

- **Install Webprofiler and Identify Controllers**
- **Understand the Service Container**
- **Create a Service**
- **Configure a New Service**
- **Get a Service out of a Controller**
- **Understand How Shortcut Methods Use Services**
- **Access a Service within a Service**
- **Add Configuration to a Service**
- **Override Drupal Core Services**
- **Understand Drupal Events Versus Hooks**
- **Create an Event Subscriber with Tags**
- **Utilize Event Arguments and Request Objects**
- **Inspect the Event Listener Behind Render Arrays**

Day 07 - Cron Queuing

Objective

At the end of this exercise, you will be able to

- Understand and utilize the Queue API to add tasks to the queue and process them on cron runs

Primary Tutorial

- http://www.sitepoint.com/drupal-8-queue-api-powerful-manual-and-cron-queueing/

Exercise

- Build a module that sends a welcome email to registered users
- Use the Cron Queue by adding to the queue the user id whenever a user registers
- Create a queue worker that picks these queue items (uids) during the cron and sends a welcome email (can be any simple text) to the registered email address.

Additional Resources

- NA

Bonus Exercise

- NA

Day 08 - Plugin System - Text Filters

Objective

At the end of this exercise, you will be able to

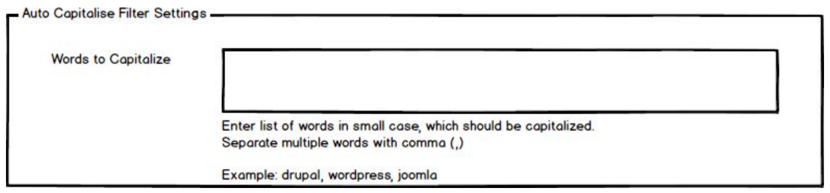
- Understand a bit of the plugin system
- Create a new Custom Text filter using the Plugin system

Primary Tutorial

- https://drupalize.me/blog/201409/unravelling-drupal-8-plugin-system (Outdated at places, a bit long, but definitely worth a read)
- https://www.lullabot.com/articles/creating-a-custom-filter-in-drupal-8

Exercise

- Create a custom text filter that could be added to any text format, which auto-capitalizes pre-configured words anywhere they occur in the filtered text
- The filter has a configuration form that allows to configure the list of words that should be auto-capitalized


Auto Capitalise Filter Settings

Words to Capitalize

Enter list of words in small case, which should be capitalized.
Separate multiple words with comma (,)

Example: drupal, wordpress, joomla

Additional Resources

- https://www.drupal.org/developing/api/8/filter

Day 09 - Attaching assets (css/js) on D8

Objective

- We are all familiar with the multiple ways of adding JS/CSS to a module or a theme to be available on select / all pages on D7. In this exercise we will check out how we could do the same on D8.

Primary Tutorial

- https://www.drupal.org/theme-guide/8/assets
- https://www.drupal.org/developing/api/8/assets
- https://www.drupal.org/node/2605130

Exercise

- Create a custom module and add a CSS and JS file in appropriate subdirectories in the module.
- Create a libraries.yml file and define 2 libraries - one for each of the CSS and JS
- Attach the CSS such that it is loaded for all Table elements shown anywhere on the site. Hint
- Take any custom block that you have built over the previous exercises. Modify the build() to attach the JS to be loaded whenever the block is displayed. Hint

Additional Resources

- N/A

Day 10 - Configuring your local site for Development

Objective

- Over the earlier tutorials, you would have noticed that Drupal would not always listen to your code changes although you had turned off caching. Looks like in D8, there is more that you would have to do to your D8 site to turn off caching completely, than just turning it off on the performance settings page.

  In this session, we will learn about:
  - Setting up your settings.local.php
  - Null Cache Service
  - Disabling Twig Cache
  - rebuild.php

Primary Tutorial

- https://www.youtube.com/watch?v=rRsOxSuJ4OU

Exercise

- On your local D8 site, perform all the configuration changes mentioned on the primary video tutorial

Additional Resources

- https://www.drupal.org/node/2598914

Day 11 - Creating a Custom D8 Content Entity Type

Objective

- In this course today, we will explore creating content entity type, with administration management pages
- We will also try building the same using boilerplate code generated by Drupal Console

Primary Tutorial

- https://www.drupal.org/node/2192175
- Console Command - https://hechoendrupal.gitbooks.io/drupal-console/content/en/commands/generate-entity-content.html
- Creating a custom content entity the easy way using Drupal Console - http://blog.oskoui-oskoui.com/?p=8218 (Note: One of the screengrabs in this post is incorrect)

Exercise

- Create a custom content entity type called “Contact” with the following fields:
  - Name
  - Email Address
  - Telephone
  - Address
- Build / Provide a minimal CRUD management page to manage the custom contact entities
- Explore why would you even want to define a custom entity type vs using a Content Type

Additional Resources

- http://www.agoradesign.at/blog/how-create-custom-entities-bundle-support-drupal-8

Day 12 - Theming 101

Objective

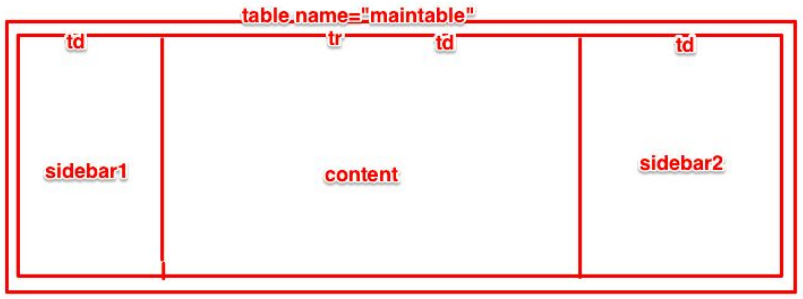
- In this course today, we will build a very basic 3 column D8 theme using “stable” as the base theme
- Explore the different base themes available for use out of the box in D8 and how they are different.
- Use Drupal Console to generate a new theme

Primary Tutorial

- https://www.lullabot.com/articles/a-tale-of-two-base-themes-in-drupal-8-core
- https://hechoendrupal.gitbooks.io/drupal-console/content/en/commands/generate-theme.html

Exercise

- Generate a new theme with “stable” as the base theme using Drupal console.
- Define 3 regions (sidebar1, content, sidebar2 while generating the theme)
- Turn on Twig Debugging to see the template names in use, as well as template name suggestions (This is covered in our Day 10 course) in the view source.
- Override page.html.twig in your theme such that the 3 columns in the page are rendered as below:



Day 13 - Logging in D8

Objective

- In this course today, we will be looking at the replacement of watchdog() in D8

Primary Tutorial

- https://drupalize.me/blog/201510/how-log-messages-drupal-8 (Also refer to comments on the post about how this should ideally be implemented via dependency injection)
- https://www.drupal.org/node/2595985

Exercise

- Build a small custom module, which adds to the "Recent Log Messages" a message of type "Node Updates" with the message "Node with title %NODE_TITLE% of type %NODE_TYPE% has been updated" whenever a node is updated.

Day 14 - Features Module in D8

Objective

- In this course today, we will be looking at the D8 Features Module

Primary Tutorial

- Hand Book - https://www.drupal.org/node/2404427
- https://www.drupal.org/project/features

Additional Resources

- https://www.phase2technology.com/blog/announcing-features-for-drupal-8/
- https://events.drupal.org/losangeles2015/sessions/features-drupal-8

Exercise

- The UI of Features is NOT any drastically different on D8 from what it was in D7. This should be an easy one. - Enable Features UI module. Export a content type to a Feature Module.
- Although Features in D7 allowed mentioning the Package Name for the Feature export, the D8 Features has more sophisticated Bundles. Explore.
- Discuss and make an attempt to understand what are the assignment plugins and how they work with bundles. (The Phase2 Blog link in Addl Resources Below is a great read for this).
- And most importantly discuss and understand when you would use CMI vs When you would use Features, and if they can co-exist on a site?
- Read through the help text on /admin/config/development/features/bundle . Try creating a new bundle on this page.
- Make an attempt to attach the feature you created earlier to the bundle that you just created.
- Understand how “Download Archive” is different from “Write”. Nothing new here. Just the names were changed from D7.
- Explore the “Include install profile” option available in Bundle Management
- Explore the assignment methods available for bundles @ admin/config/development/features/bundle and their significance.

Day 15 - Creating a Custom Field Formatter

Objective

- In this course today, we will be revising the plugin system by creating a custom field formatter

Primary Tutorial

- http://www.sitepoint.com/creating-custom-field-formatters-drupal-8/

Additional Resources

- https://www.drupal.org/node/2620966
- https://www.webwash.net/drupal/tutorials/how-create-custom-field-formatter-drupal-8

Hints for Exercise

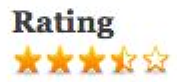
- The primary tutorial and webwash link should give you almost everything you need for the exercise
- Attaching the library to the field formatter might be a bit tricky. Check out this Stack Exchange question for hint.
- Use Drupal Console to generate the Field Formatter Plugin. You life will suddenly become so easy!

- **The complete code** for this module is available here. However, refrain from referring to the repo, and use only the hints for at least an hour once you start the exercise!

Exercise

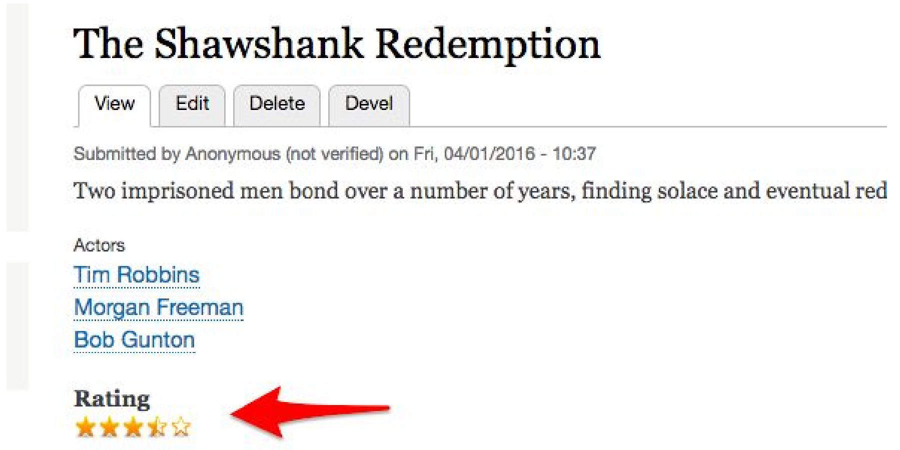
Remember the “Movie” Content type that we created during the migration exercise? Don’t worry if you don’t have that content type. On any content type, create a Decimal field called Rating that accepts any decimal value between 0 and 5.

Now, we would like this field to be displayed in the form of stars (just like you would see the movie rating on any movie site).



The CSS and Star image that we would like to use are as described on http://www.webcodingeasy.com/Webdesign/Display-simple-CSS-star-rating

In this exercise, you will build a custom field formatter for decimal fields, which when selected for the display, will show the decimal value (between 0 and 5) as a collection of stars. (Of course quantized at 0.5 stars, as limited by the CSS we chose to implement).



Submitted by Anonymous (not verified) on Fri, 04/01/2016 - 10:37

Two imprisoned men bond over a number of years, finding solace and eventual red

Day 16 - Dependency Injection Example / Service Container

Objective

- We have read about Dependency Injection on Day-06. We will be trying out an example today, in an attempt to use Dependency Injection by modifying an existing code without changing any functionality.

Primary Tutorial

- http://achilleskal.com/blog/service-container-for-drupal-geeks

Hints for Exercise

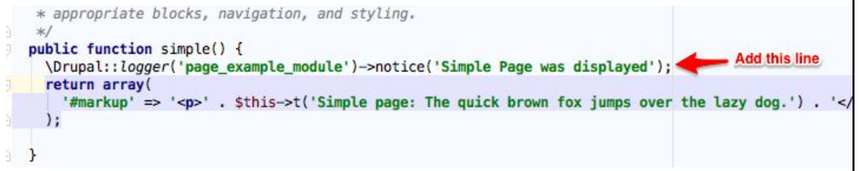
- https://www.drupal.org/node/2203931
- http://drupal.stackexchange.com/questions/109849/drupal-8-get-the-updated-content-of-a-block-via-an-ajax-call
- https://www.drupal.org/node/2168121

Solution

- Refrain from referring to the solution until you have attempted for an hour, and till you have read through Tutorial and Hints. Solution @ **https://github.com/saitanay/d8study/blob/master/modules/examples/page_example/src/Controller/PageExampleController.php**

Exercise

- Download the examples module from https://www.drupal.org/project/examples . It has a bunch of modules but we will be interested in the ***page_example*** module only, during the course of this exercise.
- The module has a ***PageExampleController*** defined in src/Controller/PageExampleController.php . The ***simple()*** method in this controller is responsible for rendering the url **“examples/page_example/simple”** when the module is enabled.
- Let’s modify this method by adding a line such that a log entry is made whenever this url is opened



```
  * appropriate blocks, navigation, and styling.
  */
 public function simple() {
   \Drupal::logger('page_example_module')->notice('Simple Page was displayed');
   return array(
     '#markup' => '<p>' . $this->t('Simple page: The quick brown fox jumps over the lazy dog.') . '</
   );

 }
```

- **Now, this is the code we will start with for our exercise.**
- Modify the code to use dependency injection to give access to logger.factory service from our controller, which will be used to do the logging, instead of \Drupal::logger.
- The links in additional resources are a great read towards achieving this.

Day 17 - Composer in your module to load PHP libraries

Objective

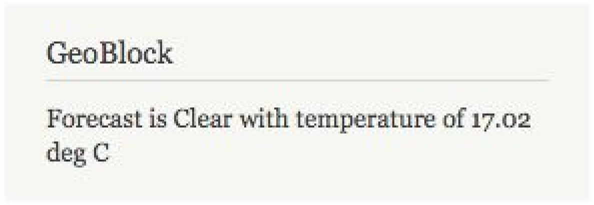
- In this session, we will see how we could leverage composer, and composer_manager to load PHP libraries / SDKs in our custom modules

Primary Tutorial

- https://drupalcommerce.org/blog/31782/managing-d8-module-dependencies-new-composer-manager
- https://bojanz.wordpress.com/2015/09/18/d8-composer-definitive-intro/

Exercise

- On a fresh D8 instance, create a custom module
- Update the module's composer.json file to include this library - https://packagist.org/packages/guhelski/forecast-php
- Install Composer Manager Module, run the init script. Run *composer drupal-update* such that the mentioned library is fetched to the vendor folder and autoloaded and hence is available for use in your module (This might not be required post 8.1.x??)
- Post 8.1.x, you could directly run *composer update* from docroot (?)
- Build a custom block with a configuration form that takes latitude and longitude in the configuration form
- The block, when enabled should show the forecast for the configured location by in simple text as "Forecast is XXXXX with temperature of XXX dec C".



GeoBlock

Forecast is Clear with temperature of 17.02 deg C

This forecast information is retrieved using Forecast wrapper library that we included.

- API Key you could use = **7411b0e6d5e0c99fbd7405fd6de00cd5** (Alternatively, you could register on forecast.io for the key)

Bonus Exercise

- Explore why you would need composer_manager module in the first place since you could just simply run *composer install* from the module folder
- Explore what the init script of composer_manager does

Day 18 - Events and Subscribers

Objective

- In this session, we will take a look into the Events and Subscribers which is a mechanism very similar to the Drupal's hook system which allows one component of code to be triggered when something else is triggered.

Primary Tutorial

- http://www.sitepoint.com/drupal-8-hooks-symfony-event-dispatcher/

Additional Resources

- https://drupalize.me/blog/201502/responding-events-drupal-8

- https://knpuniversity.com/screencast/drupal8-under-the-hood/events-versus-hooks

- https://www.previousnext.com.au/blog/alter-or-dispatch-drupal-8-events-versus-alter-hooks

Exercise

- **Discuss with your group :** When you would register an event dispatcher vs when would you define a custom hook
  Notes: Stop further propagation?

- **Discuss with your group :** Are there any core events that you could subscribe to in your custom module? Ie, if your custom module has to act on node deletion, would your custom module implement hook_delete, or would it subscribe to a core event dispatch?

- **Discuss with your group :** Is there a place where you could find list of all event dispatchers available in core?
  Notes: https://www.drupal.org/project/webprofiler , https://api.drupal.org/api/drupal/core!core.api.php/group/events/8.2.x

- **Implement a custom Event Dispatch and Subscription:**
  - Modify the page_example module such that whenever the "examples/page_example/simple" page is loaded, an event "simple_page_load" is dispatched
  - In your custom module, subscribe to the earlier event. Implement some custom code in your subscriber (Say make an entry to database logging under "Simple Page" type with the message "Simple Page Loaded".

Day 19 - Twig Templating

Objective

- In this session, we will take a look at basics of Twig Templating, that has replaced the PHPTemplate engine in Drupal.

Primary Tutorial

- https://youtu.be/V0_US_YZSxU?t=597
  (From **9:57 to 16:10** in the video)
  The video is a long one over 40mins, however the Twig section lasts for around 7 mins, which covers most parts of what is showcased in the exercise.

Solution

- https://gist.github.com/saitanay/97255a8db0a1d8bff0ffe2fa0a018dff (Probably not the best, Suggestions welcome)

Debug

- If you don't see your template changes affecting the block on page refresh, refer to Day-10 card above to find out how you can disable caching.

Exercise

- Download and install **this module** . This module provides a very basic block whose template is *`day19/templates/day19-twig-test.html.twig`* . The build() of the block provides variables - ***`var1, var2, classes, myclasscount`*** - that are available in the template to be used.
- Place the block provided by the module *(My Block)* onto a region on your site from block admin interface, so you can view the block
- If you see the block display "`Your template goes here..`" - you are all set now. Go further.
- **Now modify the template as below**
  - If var1 is set, print the value of var1
  - Move the output of var1 to a <span> whose class names are those provided by *classes* variable passed to the template (Hint - Ensure that the class names are cleaned enough so they deserve to be class names)
  - Print the variable **$var2[3]['g']**
  - Print the word "Hello", such that it can be translated using the admin translation interface
  - Move the *Hello string* into a <div> such that the div has all classes in the pattern minion0, minion1, minion2, minion3, minion4…..till minionX, where X is provided by the value of *myclasscount* variable
  - Try out {{ dump() }} and {{ dump(var1) }} on the template to see how the printed variables show up on the screen
- With the above changes, the markup of the rendered block is expected to be something like **this**.

Bonus Exercise

- The provided solution has a bug wherein the classes added to the span enclosing var1, are also added as classes to the div enclosing the translatable *Hello* String. Fix the template.

Day 20 - Cache API and Cache Tags

Objective

- Drupal 8's cache API, although fairly similar in its usage to D7, has a few fascinating additions to it. Mainly the cache tags, that would allow you to invalidate caches automatically whenever related content has changed.

Primary Tutorial

- http://www.sitepoint.com/exploring-cache-api-drupal-8/
- https://dev.acquia.com/blog/drupal-8-performance-render-caching

Additional Resources

- Cache contexts are worth exploring as well. Check out **this writeup**. As another bonus exercise, extend the block such that it displays the titles of current logged in user's latest 5 published nodes (instead of all published nodes) and apply the cache context such that the block is cached for each user than globally for all users.

Debug

- If your block is not caching, it is probably because you have disabled your caching and have unset cache bins as prescribed on card Day-10.

Exercise

- Task in hand is to build a small custom block. The block is fairly simple whose content is a concatenated string with the titles of the latest 5 published nodes (of any type). Title could be set to anything arbitrary. The body of the block should be of the format - [Title of Node A]-[Title of Node B]-[Title of Node C]-[Title of Node D]-[Title of Node E] where A, B, C, D, E are the latest 5 published nodes on the site.
- Additional requirement to improve performance is that this block should be cached and this cache should be automatically invalidated when any of these 5 nodes are updated. We are not bothered about new nodes added, but the block should not display an outdated title at any point of time.
- Hint: Cache the output of the block's build() such that function doesn't compute the string if it already exists in cache. Set cache tags as node:a, node:b, node:c, node:d, node:e where a,b,c,d,e are nids of the retrieved nodes - A,B,C,D,E
- Explore a more cleaner way of implementing this. You could avoid all the logic of checking if the cache exists, and if the cache tags are valid etc, by just passing the cache tags as "#cache" to the block's render array.

Bonus Exercise

- Modify the cache tags to add something more such that the block cache is invalidated even when a new node is added (published). As of the earlier implementation, the block cache is implemented only when any of the nodes represented in the cached block are modified.

Day 21 - Replacing hook_init / Doing something on every page

Objective

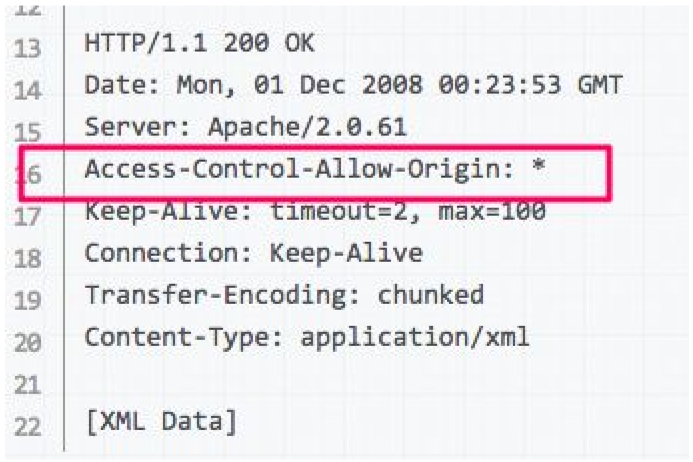
- Although doing stuff in hook_init is avoided as there is always a better way of doing the same thing elsewhere. We have taken a look at the event subscriber system on Day-18. We will be extending the same to hook into the page response to do something before every response is rendered, just as you did with hook_init()

Primary Tutorial

- www.qed42.com/blog/porting-hookinit-drupal8

Exercise

- Build a custom module such that the below highlighted header is added to the page response **only for Anonymous users**.



```
HTTP/1.1 200 OK
Date: Mon, 01 Dec 2008 00:23:53 GMT
Server: Apache/2.0.61
Access-Control-Allow-Origin: *
Keep-Alive: timeout=2, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: application/xml

[XML Data]
```