

DAN

- * Welcome everyone
- * We're going to be talking about site building
- * Specifically when it comes to translations, layout builder,
- * and regionaliz•ing it all.
- * This will include configuration for the technical
- * As well as usage for content teams

forward slide

- * Since this really is incredibly easy and fast to do, we're gonna save you time.

Dan Arbello

Account + Tech Lead @ Chromatic

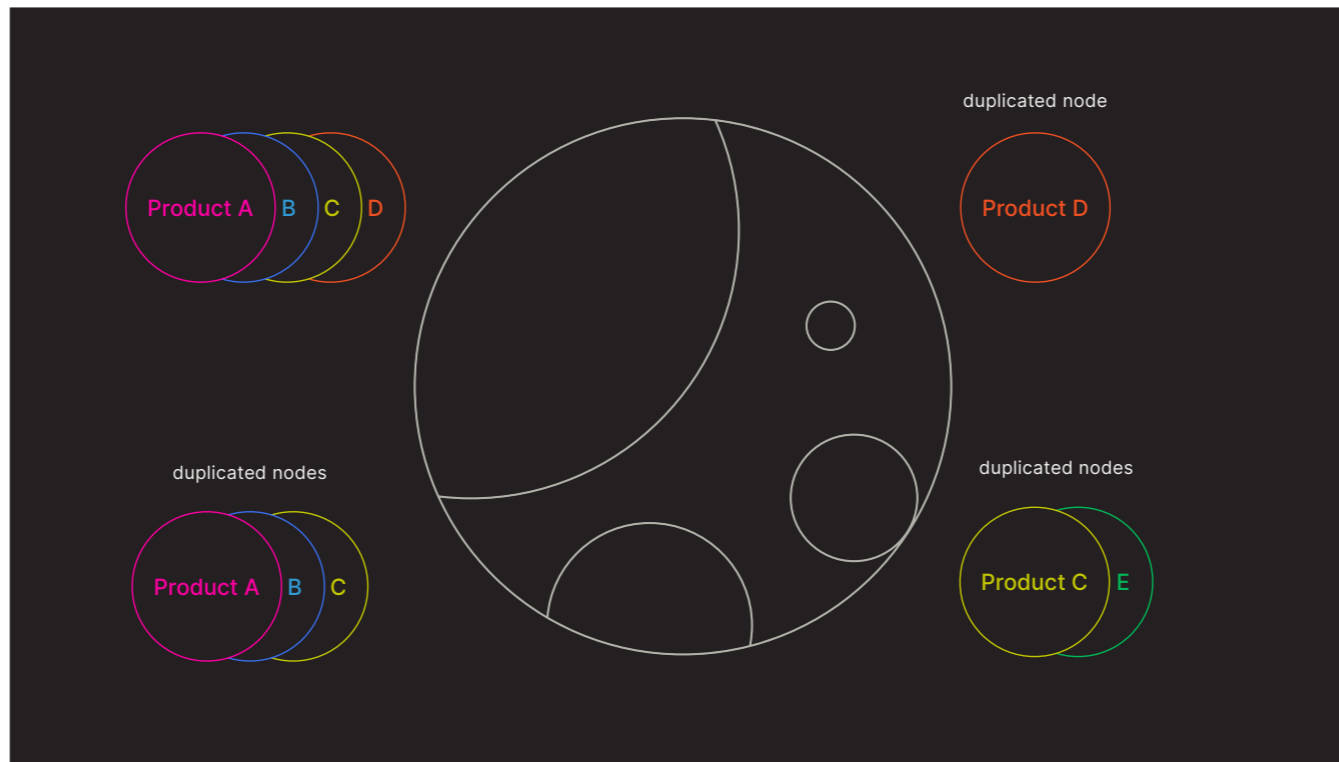


Matt Daniel

Senior Developer @ Chromatic

CHROMATIC

- * I'm Dan, account lead at Chromatic
- * I'm Matt, senior developer at Chromatic



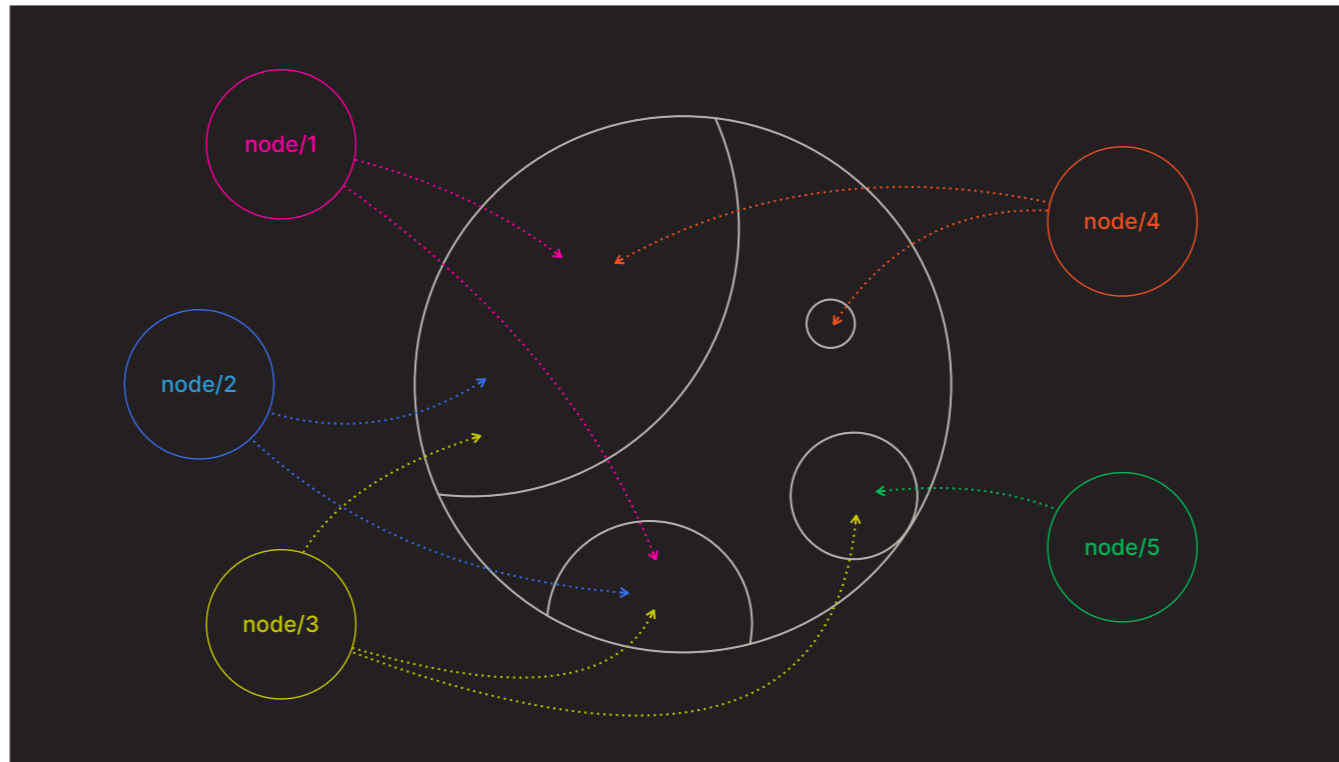
DAN

- * The foundation of this talk was inspired by a client that was using a multi-site setup to manage regionalization.
- * With a suite of several products and regulations around which product can be marketed to which regions, source of truth and duplication became a real problem.
- * Applicably, this meant they had to recreate nodes over and over again.



MATT

- * The business logic was actually quite simple however the Drupal setup was not.
- * We realized we could simplify their setup by moving to a single site but with the addition of allowing them to group their content by region
- * This made life a lot easier for site editors



MATT

- * With regionalization in place, we are still able to control the content presented to visitors, so that they see what is relevant to them, whilst improving the site building experience our client was used to
- * This talk is going to cover initial translation setup, tips for working with translations alongside layout builder, and also grouping content by geographical region and how our CDN might be able to help us

Business Value

1. Reach a wider audience
2. Full control over who sees what content where
3. Full control of the actual translated content (e.g., idiomatic translations)
4. Gain higher authority in target markets/regions
5. Multi-lingual multi-site without an actual multi-site
6. Content team can manage everything in one single place

CHROMATIC

DAN

- * To quickly wrap up the whys.
- * The value of all of this spans across teams and users.
- * For one, this automatically helps you reach a wider audience.
- * You'll get a more granular control over content and who sees it, and this extends to site content crawling, as well.
- * This means you can gain SEO authority in regions you may not have previously had authority in.
- * Lastly, this makes source of truth and content aggregation more simple and trustworthy.

Basic Requirements

→ Modules Required

Layout Builder ([layout_builder](#))

Layout Discovery ([layout_discovery](#))

Layout Builder Browser ([layout_builder_browser](#))

Layout Builder Asymmetric Translation ([layout_builder_at](#))

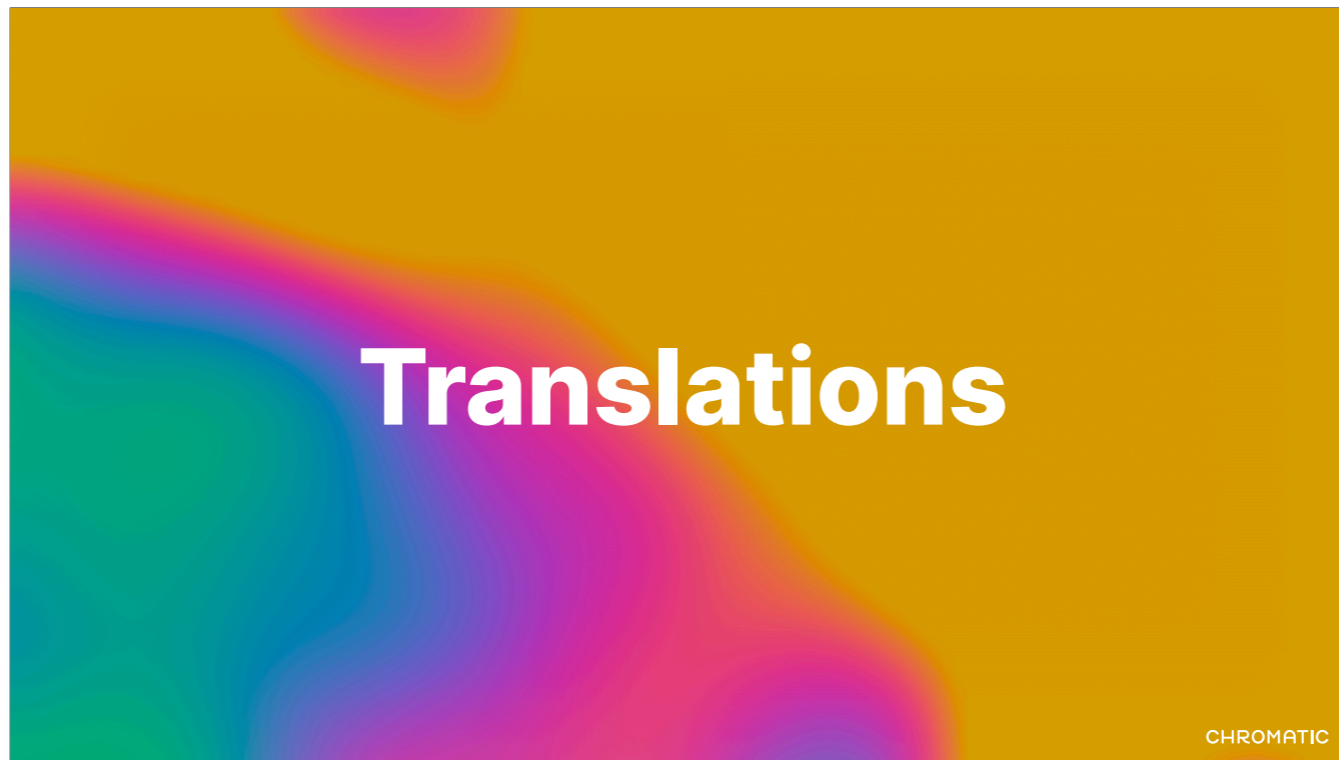
→ Audience groups in at least two different geographic regions

→ At least 2 languages configured for translation

CHROMATIC

DAN

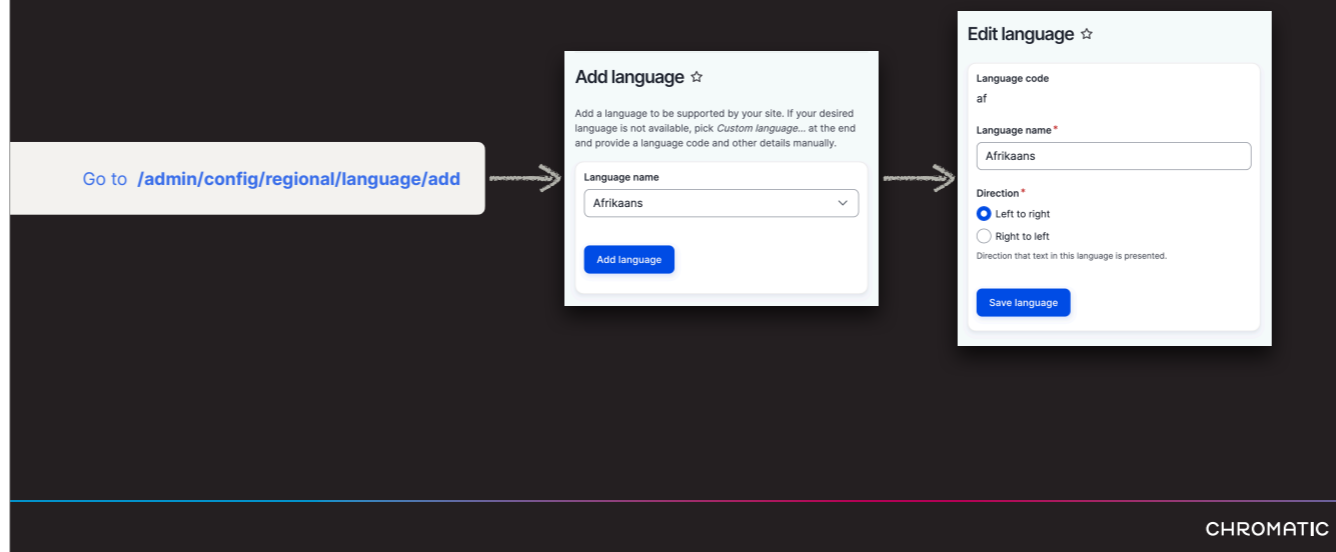
- * Before moving on, let's cover some basic requirements.
- * Of course, you'll need Layout Builder installed
- * But in addition, you'll need three other modules that will make site building and translations with Layout Builder possible.
- * Those are Layout Discovery, Browser, and Asymmetric Translation
- * You'll need at least two languages configured in Drupal
- * And an audience that accesses the site from two different geographic regions (or more).



DAN

- * We've broken this talk into three major sections with the first being Translations.
- * Next will be Layout Builder
- * And finally we'll wrap up with how to regionalize all of this.

Add a language



DAN

- * You'll need more than your default language to be configured.
- * Drupal comes with a preset of over 100 languages.
- * Selecting from this list will automatically provide its corresponding ISO standard language code, along with a customizable name label and writing direction.

Custom languages

Add language ☆

Add a language to be supported by your site. If your desired language is not available, pick *Custom language...* at the end and provide a language code and other details manually.

Language name
Custom language... ▾

Language code *

Use language codes as [defined by the W3C](#) for interoperability.
Examples: "en", "en-gb" and "zh-hant".

Language name *

Direction *
 Left to right
 Right to left
Direction that text in this language is presented.

Add custom language

DAN

- * You can also add your own custom language (available at the absolute bottom of the list).
- * However, it is crucial to note that you must not set a language code that conflicts with any other ISO standard lang code.
- * This will cause plenty of conflicts both within the admin UI and for the end users.
- * Thankfully, Drupal provides a helpful link in the field's help text that will go into more detail.

Negotiate the language(s)

Accept header → `langcode="fr-FR"`
Path-based → `/fr`

Detection method	Description	Enabled	Operations
URL	Language from the URL (Path prefix or domain).	<input checked="" type="checkbox"/>	Configure
Session	Language from a request/session parameter.	<input checked="" type="checkbox"/>	Configure
User	Follow the user's language preference.	<input checked="" type="checkbox"/>	
Browser	Language from the browser's language settings.	<input checked="" type="checkbox"/>	Configure
Selected language	Language based on a selected language.	<input checked="" type="checkbox"/>	Configure
Account administration pages	Account administration pages language setting.	<input checked="" type="checkbox"/>	

Go to </admin/config/regional/language/detection>

MATT

- * Don't forget to configure your preferred language detection methods, and the order of them as well.
- * You can prioritize and enable different detection methods, such as:
 - URL, or path based. (recommended as primary method)
 - and
 - Browser language preference
 - works by examining the HTTP "Accept-Language" header that browsers send with each request

Configuring *what* is translatable.

Go to </admin/config/regional/content-language>

The screenshot shows the 'Content language and translation' configuration page in Drupal. The page title is 'Content language and translation' with a star icon. Below the title is a brief instruction: 'Change language settings for content types, taxonomy vocabularies, user profiles, or any other supported element on your site. By default, the language selector and the language is the site's default language.' The page is divided into two main sections: 'Custom language settings' and 'Content'.

Custom language settings

- Comment
- Contact message
- Content
- Content block
- Custom menu link
- File
- Shortcut link
- Taxonomy term
- URL alias
- User

Content

Translatable	Content type	Configuration
<input type="checkbox"/>	Article	<p>Default language: Site's default language (English)</p> <p><small>Explanation of the language options is found on the languages list page.</small></p> <p><input checked="" type="checkbox"/> Show language selector on create and edit pages</p>
<input checked="" type="checkbox"/>	Basic page	<p>Default language: Site's default language (English)</p> <p><small>Explanation of the language options is found on the languages list page.</small></p> <p><input checked="" type="checkbox"/> Show language selector on create and edit pages</p> <p><input type="checkbox"/> Hide non translatable fields on translation forms</p>
<input checked="" type="checkbox"/>	Published	
<input checked="" type="checkbox"/>	Authored by	
<input checked="" type="checkbox"/>	Title	
<input checked="" type="checkbox"/>	Authored on	
<input checked="" type="checkbox"/>	Channel	

MATT

- * Now that we have languages configured, we need to specify what editors can translate.
- * By default, Drupal Translations will provision nodes and most of their fields as translatable, but you can adjust this as you please.
- * This is also where you specify what language should be the default.

Ensure templated text is translatable

Go to </admin/config/regional/translate>

```
$ajax_error_string->t();  
{%t 'An AJAX HTTP error occurred.' %}
```

Back to site | Administration | Configuration | Region and language

User interface translation

Translate Import Export Settings

This page allows a translator to search for specific translated and untranslated strings, and is used when creating or editing translations. (Note: Because translation tasks involve many strings, it may be more convenient to [export](#) strings for offline editing in a desktop Gettext translation editor.) Searches are limited to strings in a specific language.

Filter translatable strings

String contains
An AJAX HTTP error occurred.
Leave blank to show all strings. The search is case sensitive.

Translation language
Spanish

Search in
Both translated and untranslated strings

Filter Reset

* Changes made in this table will not be saved until the form is submitted.

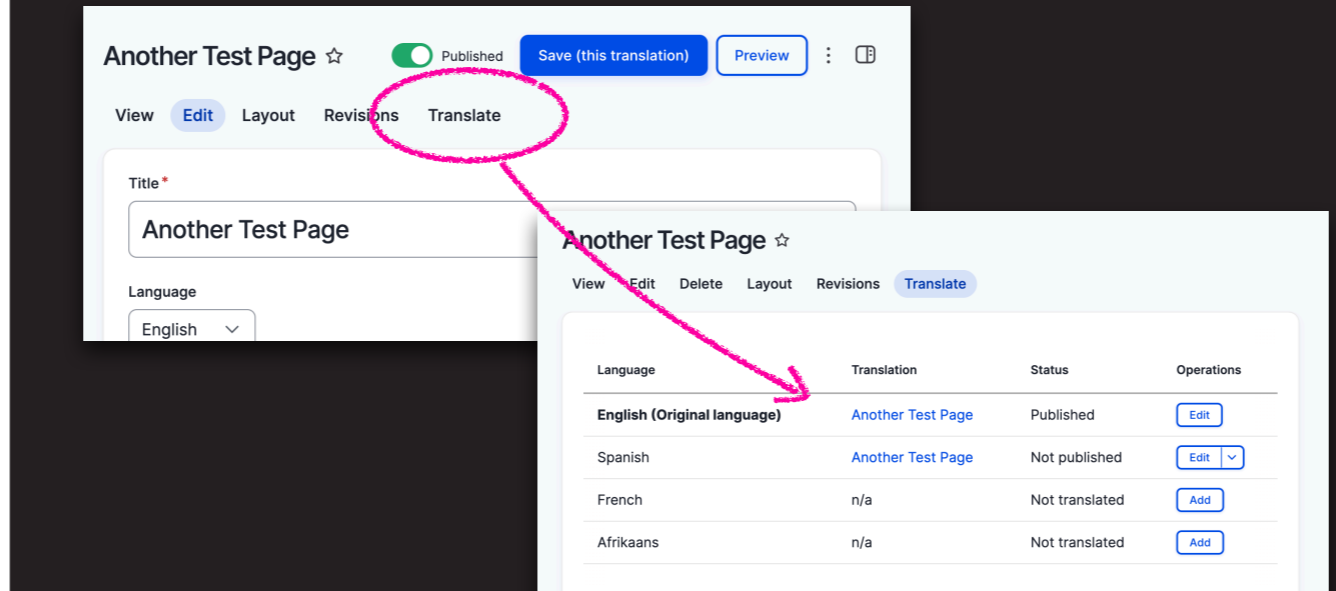
Source string	Translation for Spanish
An AJAX HTTP error occurred. *	Se ha producido un error AJAX HTTP.

Save translations

DAN

- * So long as you're making use of the trans function in templating and preprocessing, Drupal will be able to log these strings in the user interface translation section.
- * Your content team can then provide translation for any and all of this baked in text.

Translate a node

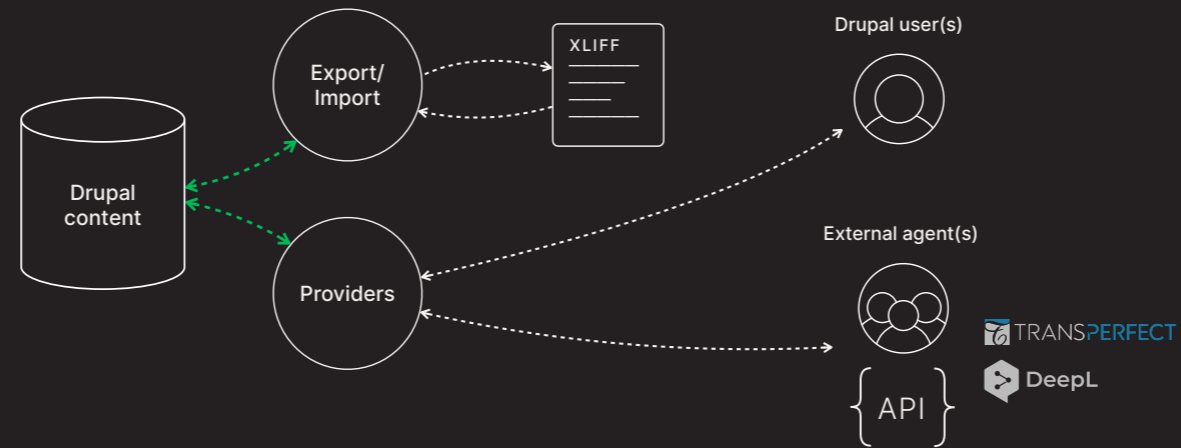


MATT

- * This is in addition to translating actual content
- * Which exhibits the same UI as your usual node edit view

Translation Management & Automation

Translation Management Tool ([tmgmt](#)) / GlobalLink Connect for Drupal ([globallink](#))



CHROMATIC

DAN

- * However, if you're getting to translations just now, and you already have content, this all might seem insurmountable.
- * As usual, there is module for that: Translation Management Tool.
- * Or, commonly acronymed as "TMGMT".
- * This will allow you to, essentially, project manage translations all within Drupal admin UI.

Content types ☆ [+ 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 form display Manage display Edit Translate Manage permissions Delete

↓

[Request translation](#) [Add to cart](#) There are 0 items in the [translation cart](#).

Language	Pending Translations	Operations
<input type="checkbox"/> English (original)	Source	Edit
<input checked="" type="checkbox"/> Spanish	None	Add
<input type="checkbox"/> French	None	Edit ↓
<input type="checkbox"/> Afrikaans	None	Edit ↓

→

Create a translation job

Label: Custom job name

Source language: English | Target language: Spanish | Total words: 15 | Total HTML tags: 2

Provider: File exchange

Export to: HTML | XLIFF

Submit to provider | Save job | Delete

State: In progress | Needs review

DAN

- * TMGMT will let you assign translation jobs (whether entire content types or individual entities) to a provider
- * A “provider” can either be a Drupal user, a third-party API (like TransPerfect, Google Translate, DeepL, et cetera), or an XLIFF file.

Leave the heavy lifting to XLIFF

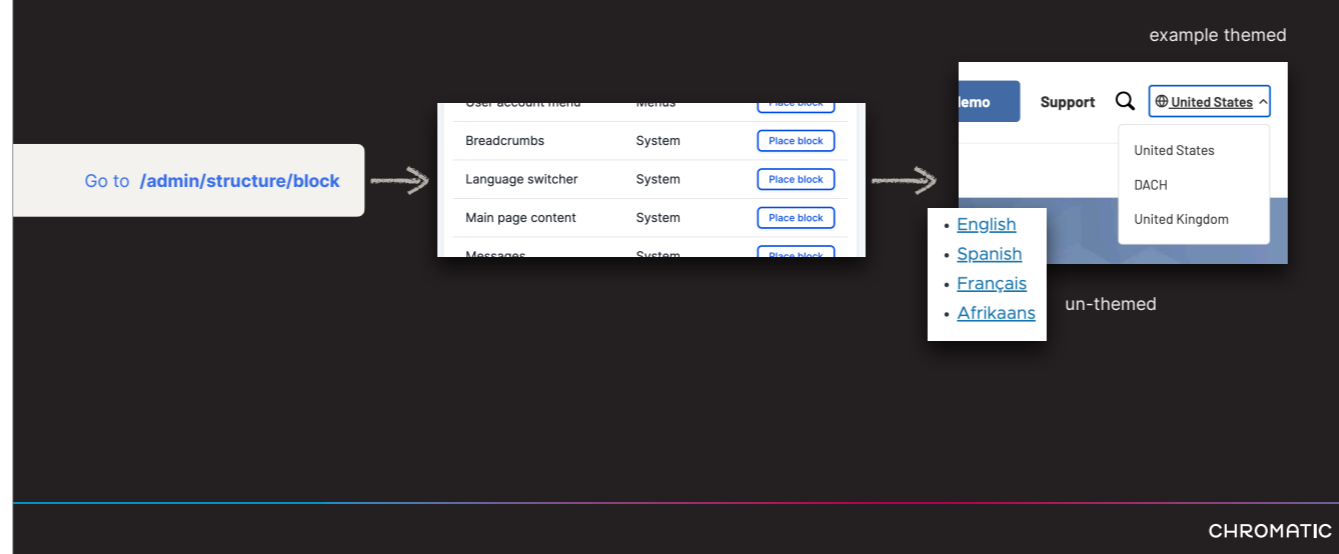
Status message
Exported file can be downloaded [here](#).

```
<?xml version="1.0" encoding="UTF-8"?>
<xliff version="1.2" xmlns="urn:oasis:names:tc:xliff:document:1.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xliff:document:1.2 xliff-core-1.2-strict.
xsd">
  <file original="xliff-core-1.2-strict.xsd" source-language="en"
target-language="es" datatype="plaintext" date="2025-03-24T03:03:53Z">
    <header>
      <phase-group>
        <phase tool-id="tmgmt" phase-name="extraction" process-name="extraction"
job-id="1"/>
      </phase-group>
      <tool tool-id="tmgmt" tool-name="Drupal Translation Management Tools"/>
    </header>
    <body>
      <group id="1">
        <note>Article content type</note>
        <trans-unit id="1"[node__type__article][name" resname="1"][node__type__article
[name">
          <source xml:lang="en">Article</source>
          <target xml:lang="es"/>
          <note>Name</note>
        </trans-unit>
        <trans-unit id="1"[node__type__article][description" resname="1"
[node__type__article][description">
          <source xml:lang="en">Use &lt;em>articles</em> for time-sensitive
content like news, press releases or blog posts.</source>
          <target xml:lang="es"/>
          <note>Description</note>
        </trans-unit>
        <trans-unit id="1"[core__base_field_override__node__article__title][label"
resname="1"][core__base_field_override__node__article__title][label">
          <source xml:lang="en">Title</source>
          <target xml:lang="es"/>
          <note>Label</note>
        </trans-unit>
      </group>
    </body>
  </file>
</xliff>
```

DAN

- * An XLIFF file can then be sent to a translator, who can iterate on it, and return it for import back into Drupal.
- * **[CHEEKY] The takeaway here is: "Delegation is elevation"**
- * **Because why struggle when someone else can do it better?**

Add a language switcher



MATT

- * Before we move to Layout Builder,
- * We wanted to note that Drupal comes out-of-the-box with a handy language switcher UI element.
- * This will let your users change the language on the front-end easily.
- * You might find your specific business logic requires unique configuration of how this handles language negotiation, but if you followed our suggestions earlier,
- * When a user switches to a language on a node without the target translation layer (whether in moderation or just not existing) the origin language node will be loaded in its place

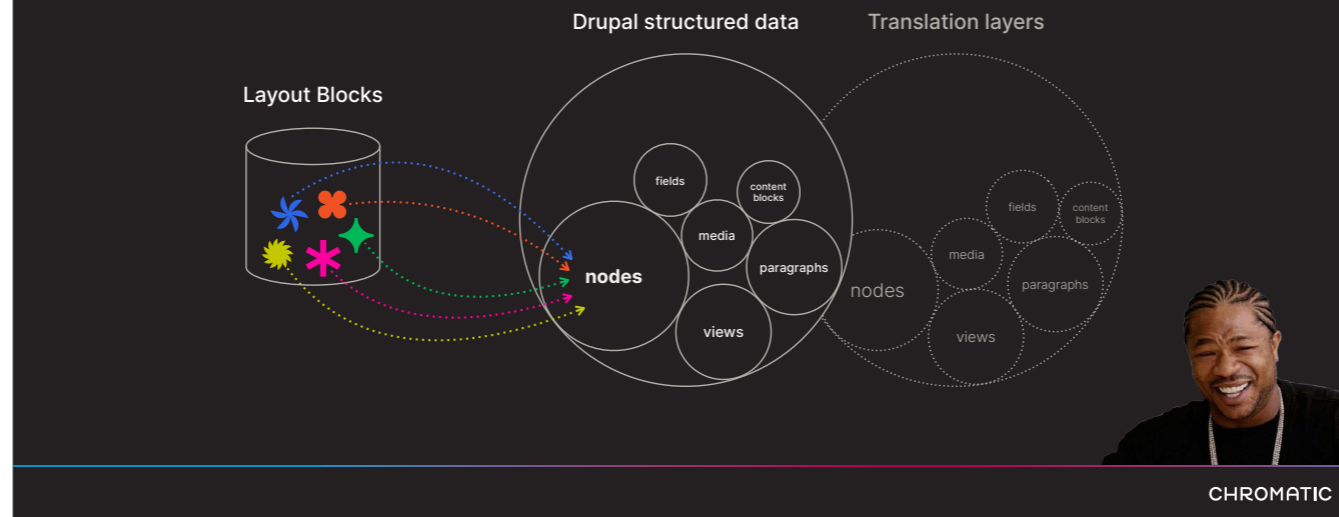


DAN

If you aren't familiar with Layout Builder then let's provide a super quick intro...

Layout Builder's Translatability

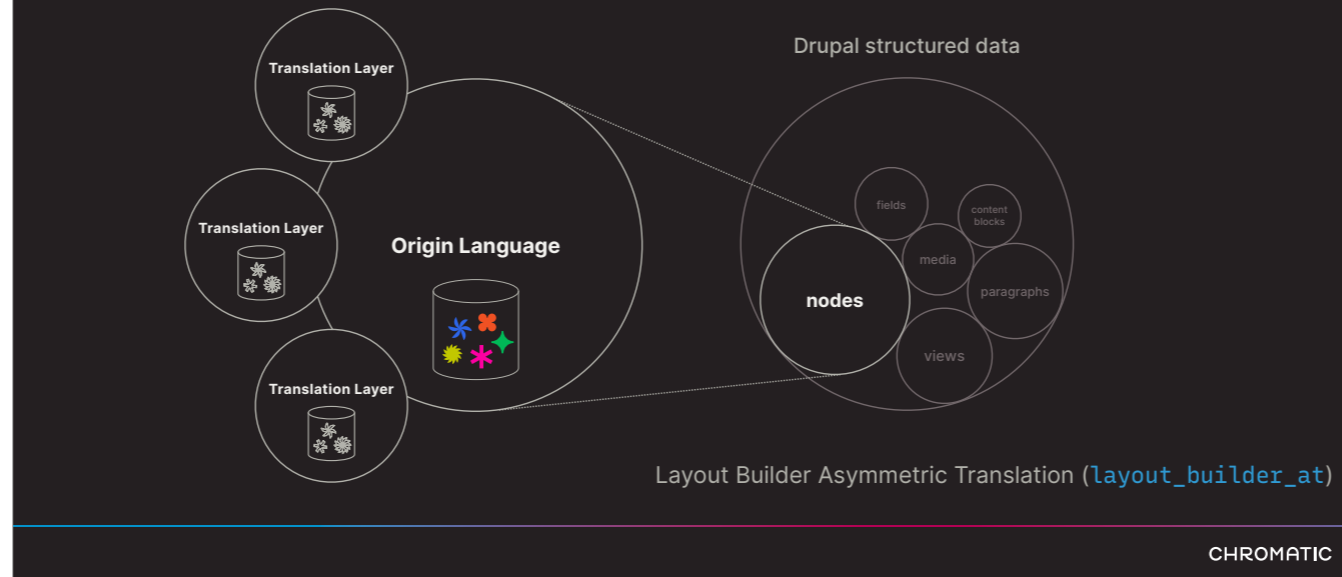
...or, rather, lack thereof.



DAN

- * Though the vast majority of entities in Drupal are translatable
- * — Layout Builder does not come with that same sort of translatability.
- * Since they're more like floating entities that get **attached** to nodes, their contents are kind of like attachments on attachments.
- * That means, out of the box, you cannot have two translation layers with different Layout blocks.
- * That is, **without** the Asymmetric Translation module.

Preparing Layout Builder for Asymmetry



DAN

- * This module allows you to have a completely different set of Layout contents (and their block instances) between translation layers.
- * To make Layout translatable is to asymmetrically divide the translation layer's connection to a single Layout with a separate Layout instance.
- * Two translation layers then require two Layout instances.

The image shows a screenshot of the Drupal administration interface for configuring content translatability. On the left, a table lists content types and their configurations. The 'Basic page' content type is selected, and its configuration is shown. A green arrow points from the 'Layout' checkbox in the configuration table to a modal window that lists fields: 'URL alias', 'Body', and 'Layout'. The 'Layout' checkbox is checked in this modal. On the right, a dark overlay contains the text 'Configure Layout as translatable' and a button that says 'Go to /admin/config/regional/content-language'.

Translatable	Content type	Configuration
<input type="checkbox"/>	Article	Default language: Site's default language (English) Show language selector on create and edit pages: <input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	Basic page	Default language: Site's default language (English) Show language selector on create and edit pages: <input checked="" type="checkbox"/> Hide non translatable fields on translation forms: <input type="checkbox"/>
<input checked="" type="checkbox"/>	Published	
<input checked="" type="checkbox"/>	Authored by	
<input checked="" type="checkbox"/>	Title	
<input checked="" type="checkbox"/>	Authored on	
<input checked="" type="checkbox"/>	Changed	
<input checked="" type="checkbox"/>	Promoted to front page	
<input checked="" type="checkbox"/>	Sticky at top of lists	
<input checked="" type="checkbox"/>	URL alias	
<input checked="" type="checkbox"/>	Body	
<input type="checkbox"/>	Layout	

Configure Layout as translatable

Go to </admin/config/regional/content-language>

URL alias

Body

Layout

DAN

- * Once you've installed and enabled Asymmetric Translation you'll need to configure Content's Layout field as translatable
- * This is done via the "Content language and translation" section
- * As we've seen earlier, you simply check "Layout" in Content's configuration.

Allow layout cloning for translation layers

Content type(s) making use of Layout Builder > [Manage form display](#)

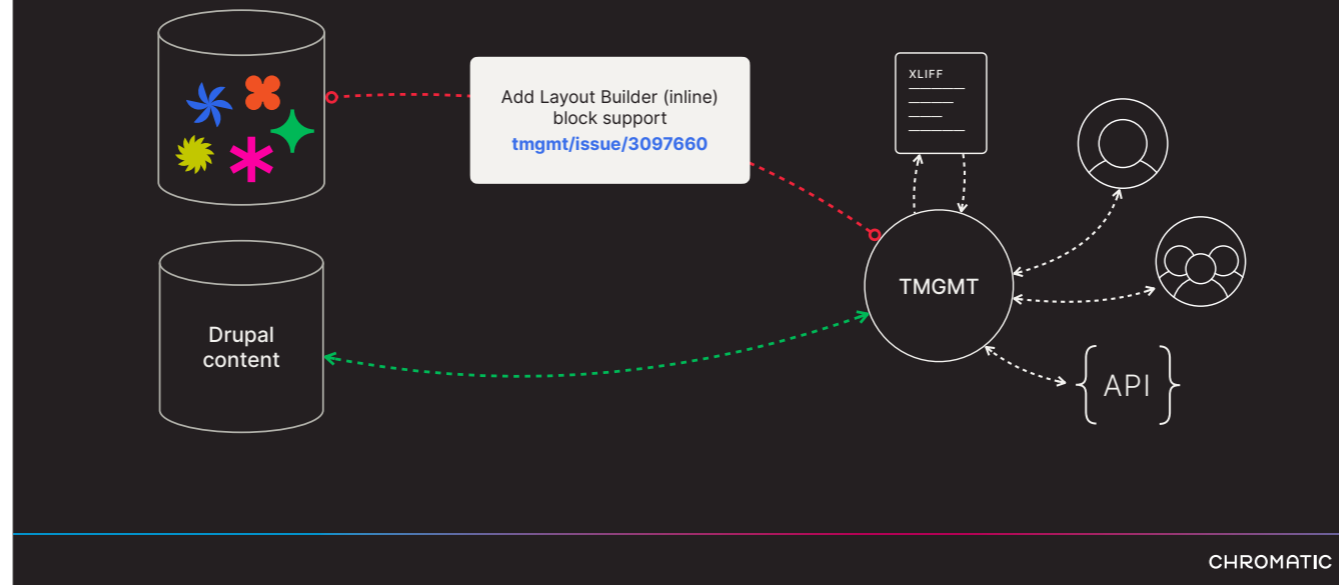
The screenshot shows the 'Manage form display' interface for a content type using Layout Builder. A modal window titled 'Widget settings: Layout Builder Asymmetric Translation' is open, showing the 'Checkbox appearance' set to 'Checked'. Below the modal, the 'Layout' field is visible, and the 'Copy blocks into translation' checkbox is highlighted with a pink circle. The background shows the form display editor with the title 'Page de test' and the text 'Il s'agit d'une page de test'.

No you can toggle layout cloning when adding a translation layer to a node.

DAN

- * You'll also want to make sure that the Layout field is enabled for the node's form display.
- * Bonus here is that you can set it so editors don't have to start from scratch on the new translation layer.
- * Now you can go into the Layout and update the text for all the blocks (and their fields) as desired without having to recreate all of it from the ground up.

TMGMT is not quite ready for Layout Builder



DAN

- * Cool, this make it much easier for content editors to plug in translations.
- * Bad news is that you currently cannot include Layout Builder block contents with Translation Management jobs.
- * Good news is that 🙌 **this** issue — that aims to solve this — is very close to being resolved.

Our options for now...



Manual translation

Make translators Drupal users.
Ease fatigue with AI tools like
DeepL's browser extension.



Build an API translator

Fuck it! Build your *own* API
connector!
Probably could even connect
up to third-party translation
services to support glossaries.



Manipulate the DOM

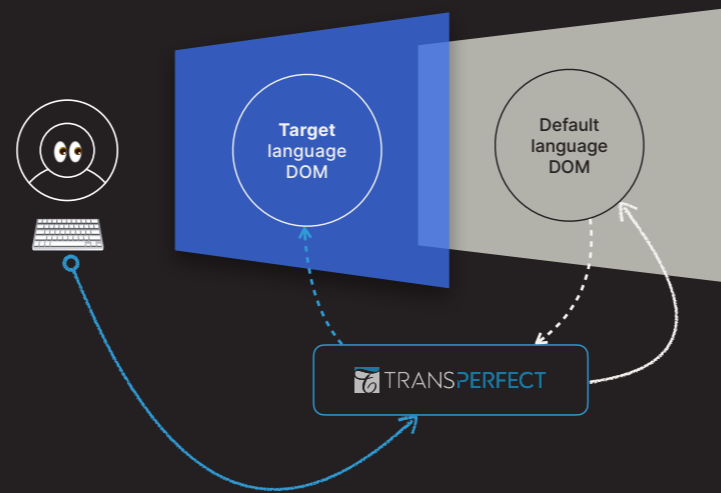
Make use of services like
GlobalLink Web or Localize.js.
Translates DOM for crawlers
and all site-internal interactions
like search queries and filtering.

CHROMATIC

DAN

- * Ok. What can you do in the meantime?
- * Well, you still have manual translations and if your translators are Drupal users there isn't really much of a problem to solve.
- * On the other hand, if you have a lot of nodes to translate, even a translation team can get worn out doing so.
- * There **is** another option, outside of just grabbing API wires and connecting them manually with some custom hand-made aggregator.

Dominate the DOM



CHROMATIC

DAN

- * Third-party services like GlobalLink Web can translate your content on the user's side by translating it at the DOM level.
- * The translations can also be customized by your team and can even be integrated with AI services like DeepL and its glossary system.
- * This tool will even translate the content for SEO crawling as well as Drupal form interactions (like search) by creating a separate DOM layer for the translations.

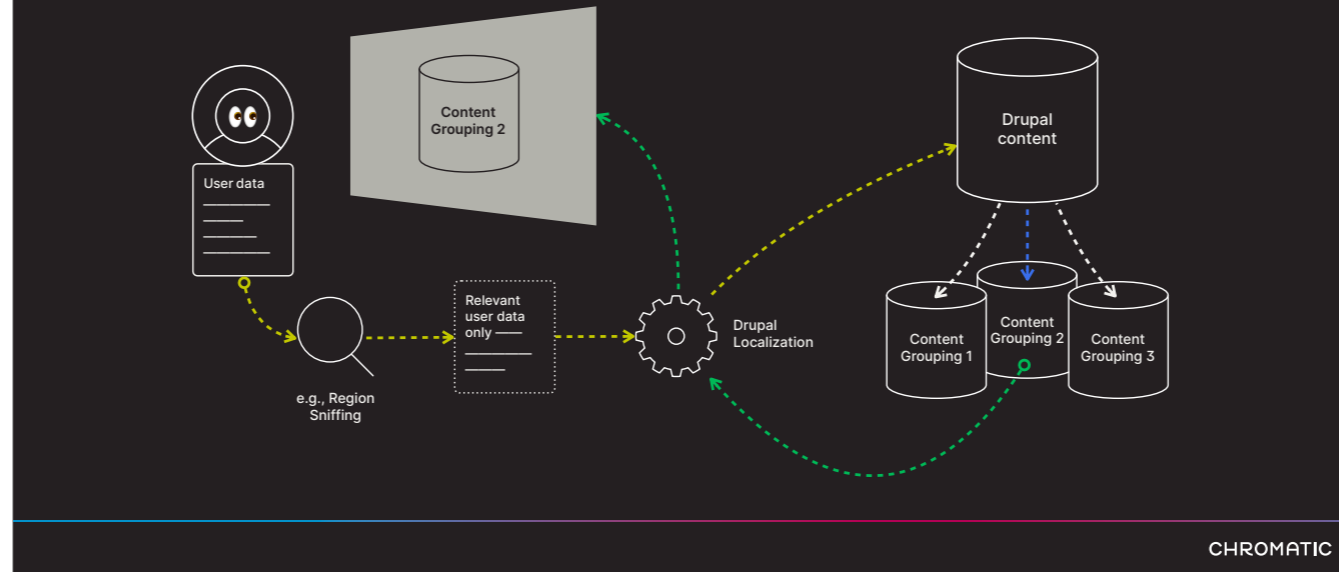
Regionalization

CHROMATIC

MATT

- * Lastly, we have regionalization
- * Which might also be described as "localization" or "regional customization"

What is regionalization?



MATT

- * While translation deals with more than one language, regionalization focuses on adapting content and functionality for specific geographic regions while potentially keeping the same language.
- * For example, we might need to display a different product range on our North American site compared to our UK site, however both require English as an available language option.

Group Regions

This is an example content group switching custom module

This is a taxonomy vocab!

Content Group ☆ [+ Add term](#)

List

You can reorganize the terms in *Content Group* using their drag-and-drop handles, and group terms under a parent term by sliding them under and to the right of the parent.

[Show row weights](#)

Name	Status	Operations
Argentina	Published	Edit
Australia	Published	Edit
Belgium	Published	Edit
Canada	Published	Edit
France	Published	Edit
France	Published	Edit

Configurable settings for Benz content groupings.

Enable
Enable content grouping functionality.

Modal Title

The modal is the page overlay that appears on front page, when no cookie is yet set. Leave blank if you do not want a title.

Switcher Dropdown First Option

The switcher is the dropdown used to navigate between the content groupings, usually located in the Header region. This sets the dropdown's first option text. (E.g. "Select country")

Search API Content Index Name

The matching machine name of the index field that represents the content grouping code in the Search API "Benz" index. See `admin/config/search/search-api/index/benz/fields`. It will likely be `field_content_grouping_code` or `field_content_grouping_code_1`. This is used in `benz_content_grouping_search_api_query_alter()` to alter the search query and return only content appropriate for the given content grouping.

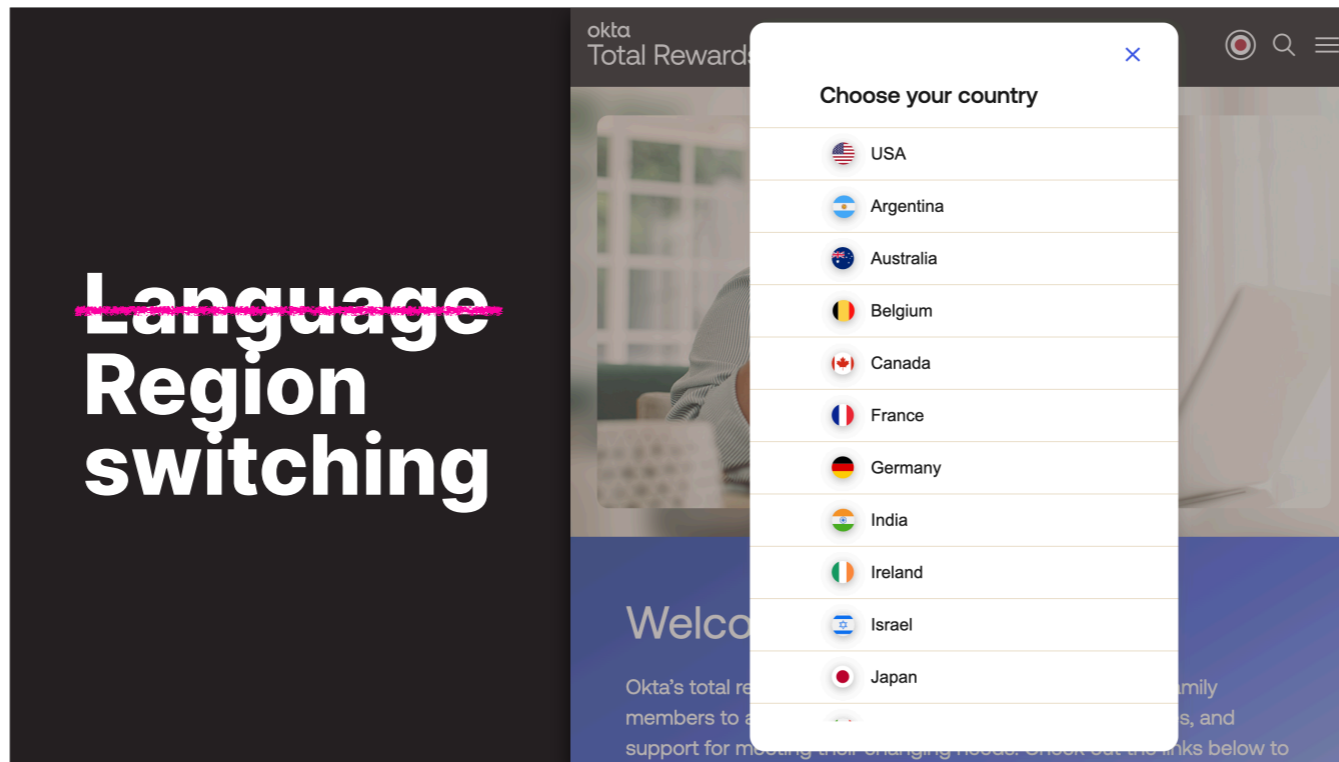
Search API Media Index Name

The matching machine name of the index field that represents the media grouping code in the Search API "Benz" index. See `admin/config/search/search-api/index/benz/fields`. It will likely be `field_content_grouping_code` or `field_content_grouping_code_1`. This is used in `benz_content_grouping_search_api_query_alter()` to alter the search query and return only content appropriate for the given content grouping.

USA
 Argentina

MATT

- * So, how can we handle this in Drupal?
- * It might be possible to only use translation to handle regions if it fits your use case (where one region = one language).
- * If you need more flexibility, it would be worth building the ability to group by region, and translate too, if necessary.
- * A taxonomy vocabulary for grouping by region, combined with the Context module for controlling visibility, would be one approach, and would still allow us to layer on one or more language translations



MATT

- * Remember the language switcher from before?
- * Considering everything so far, we can use a custom switcher to change content based on taxonomy instead. This could also be a combined region & language switcher.
- * You may also want to consider setting a cookie when the user updates their selection, which we will come back to in a minute

Different content for different regions

Home

Product A (USA)

A more colorful world, one refill at a time.

Join more than 85,000 people that have helped collect plastic and stopped this from entering the ocean.

And that's just the start: register your bottle on our Ocean Bottle App and we collect more plastic every time you use it.

Home

Product A (UK)

A more colourful world, one refill at a time.

Join more than 85000 citizens that have helped collect plastic and stopped this from entering our oceans.

That's just the start: register your bottle on our Ocean Bottle App and we collect more plastic every time you use it.

MATT

- * Putting this setup together, this demo product page now shows two different sets of content, both are in English and each region now also has the ability to be translated as needed.
- * Notice how some of the content is a translation (color vs colour) while other content is completely different

Region Detection

If we have multiple regions, we should try to improve visitor experience by automating the region selection for them by geo targeting

CDN

Fastly can add geolocation headers to requests

Don't use geolocation for language: If possible, use the Accept-Language header to customize language preferences.

Smart IP Module

Integrates with various geolocation databases to determine user location

Browser Language Detection

Read the [Accept-Language](#) header to infer both language and region

Why not both? We can IP based geotargeting plus Browser headers to be more accurate.

CHROMATIC

MATT

- * We should try to improve user experience by sending visitors to the region that appears to be the most relevant to them
- * To do this, we can use the users location, obtained using our Content Delivery Network, or by using the users IP address and a geolocation database.
- * If we have multiple languages, in addition to regions, we can combine the location data with the language negotiation options we covered earlier, such as the browser's Accept-Language header, to be as accurate as possible for both region and language.
- * The user can always override the automated selection, and we can track their preference.
- * Also, a note here to avoid using geolocation for language: The ability to speak a language is a property of a person, not the place where they are currently located.

```
as_number": 16509,  
"area_code": 0,  
"city": "tokyo",  
"conn_speed": "broadband",  
"conn_type": "wired",  
"continent": "AS",  
"country_code": "JP",  
"country_code3": "JPN",  
"country_name": "japan",  
"latitude": 35.68,  
"longitude": 139.75,  
"metro_code": 392001,  
"postal_code": "100-0001",  
"proxy_description": "cloud",  
"proxy_type": "hosting",  
"region": "13",  
"utc_offset": 900
```

Receive user data from CDN

A CDN, like Fastly, should provide all of the regionalization, geoIP detection, redirection, and cookie handling you'll need.

MATT

- * We often work with Platform SH, who provide a Fastly CDN but other options such as Cloudflare and CloudFront also provide IP geolocation data such as continent, country and city
- * This data is typically included for no additional cost with your CDN service, so it can make a good alternative to using a separate paid database

Configure Fastly VCL

```
sub vcl_recv {
    # Skip redirect if user has set a preference cookie
    if (req.http.Cookie ~ "user_region_preference=") {
        # User has explicitly set their preference, respect it
        return;
    }

    # Otherwise check if user is in Europe
    if (client.geo.country_code ~ "(DE|FR|IT|ES|GB|NL)") {
        # Redirect to European version if not already there
        if (!req.url ~ "^/eu/") {
            # Set redirect using standard HTTP redirect status
            set req.http.Location = "https://drupalsite.com/eu"
            req.url;
            error 302 "Redirect to EU site";
        }
    }
}
```

MATT

- * Continuing the Fastly example, for a high performance approach, Fastly can help us to handle geolocation redirection too
- * Fastly VCL is a programming language which is part of their platform architecture, allowing us to code custom configuration
- * The vcl_recv subroutine is executed when a client request is received by Fastly, this brief example shows a redirect to our European region, (/eu), if the visitor is located in selected countries

```
sub vcl_recv {
    # Skip redirect if user has set a preference cookie
    if (req.http.Cookie ~ "user_region_preference=") {
        # User has explicitly set their preference, respect it
        return;
    }

    # Otherwise check if user is in Europe
    if (client.geo.country_code ~ "(DE|FR|IT|ES|GB|NL)") {
        # Redirect to European version if not already there
        if (!req.url ~ "^/eu/") {
            # Set redirect using standard HTTP redirect status
            set req.http.Location = "https://drupalsite.com/eu"
            req.url;
            error 302 "Redirect to EU site";
        }
    }
}
```

Check for a cookie

MATT

- * To not annoy our visitors, we should respect the cookie we dropped earlier, which we can check for right in the CDN, only performing a redirect if necessary
- * There are a lot of details and code examples in the Fastly documentation covering available data and implementation examples.

Alternatives

Smart IP (`smart_ip`)

Detects visitor's geographic location (country, region/state, city) based on IP address.

Provides multiple geolocation data source options including:

- + MaxMind GeoIP (free and commercial databases)
- + IP2Location (commercial database)

GeoIP (`geoip`)

Provides an API for geolocating an IP address, plugins for Maxmind, and some CDNs

Custom





Patch together 🐡 those module and/or a custom module

CHROMATIC

MATT

- * There are alternative IP geolocation options if you don't want to rely completely on your CDN, or if you don't use one
- * The most obvious contrib modules include Smart IP and GeoIP which are both powerful tools for location-based customization
- * We could also implement a custom solution, for example we can obtain an IP database from Maxmind and use a custom Drupal module or JavaScript to handle the lookup and redirection.
- * Whatever option you go with, keep in mind that IP databases are not always up to date, and may be more inaccurate, the more granular you get

How PII is defined across the world

Regime	What is considered PII	IP collection parameters	Notice/ Consent
 General Data Protection Regulation (GDPR)	Names, IDs, location data, online identifiers (aka "traditional identifiers") that can directly identify an individual	Treated as processing personal data	✓
African Union Convention on Cyber Security & Personal Data Protection	May include traditional identifiers	Appears to be leaning towards GDPR definitions	🤖
 Personal Information Protection Law (PIPL)	Traditional identifiers, as well as any kind of location and/or behavioral data	Any location data (including IP or not)	✓
 Act on the Protection of Personal Information (APPI)	Focused on direct identifiers but evolving to include traditional identifiers	+ any other personal data	✓
 United States of America	"Take me to court first"	+ clear association with any other personal data	⊘
California Consumer Privacy Act	Similar to GDPR with some broad extensions to location data	Any location data (including IP or not)	✓

DAN

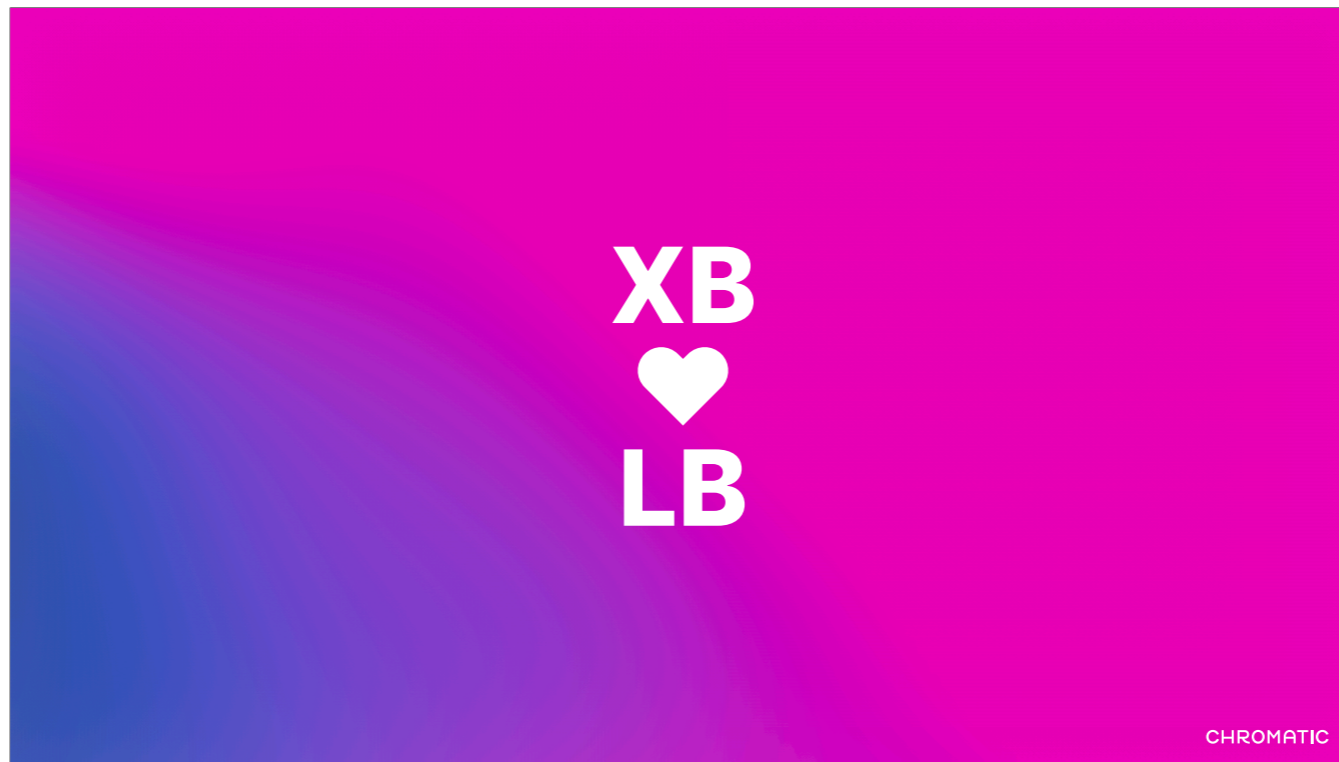
- * Now to Personally Identifiable Information (PII)
- * Global policies are an incredibly esoteric subject and I'd be baffled if any of us had a firm, confident understanding of the rules for all regimes.
- * So, we highly suggest verifying any and all functionality you have in place that collects any sort of data about your users (location or otherwise) with your legal team.
- * Having said that, out of all of these regimes listed, we've found two common parameters that suggest you should be letting your users know **if** you are collecting information about them and dropping a cookie.
- * The obvious individual identifiers like name and ID numbers are unanimously considered PII.
- * Location, especially IP address and sometimes region, is considered PII when it is collected in **addition** to any other user data that could help you **individually** identify a person (and this could include behavior).
- * Therefore, one could even argue that time and flow across the site (i.e., user analytics) counts as behavior.
- * Interestingly, our read of China and California's policies makes it seem like they consider just knowing a person's location is enough to count the data as PII.

Paragraphs VS Layout Builder

CHROMATIC

DAN

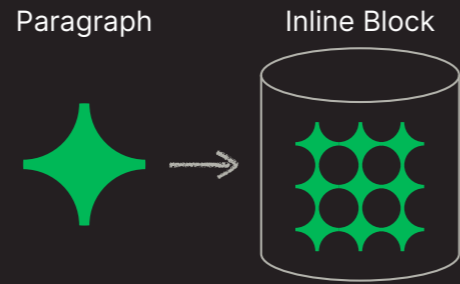
- * Alright, let's wrap up with comparing the two ways we typically provision site building for content teams.
- * Which do you choose? Paragraphs or Layout Builder? Or both?



DAN

- * First off, it should be noted that Experience Builder looks to be layered on top of or at minimum an extension or evolution of Layout Builder.
- * Sort of like how Layout Builder was to Panels.
- * Therefore, if you as an agency or engineer are looking to start working more in XB it might help to get used to LB in the meantime.

Paragraphs still have a place



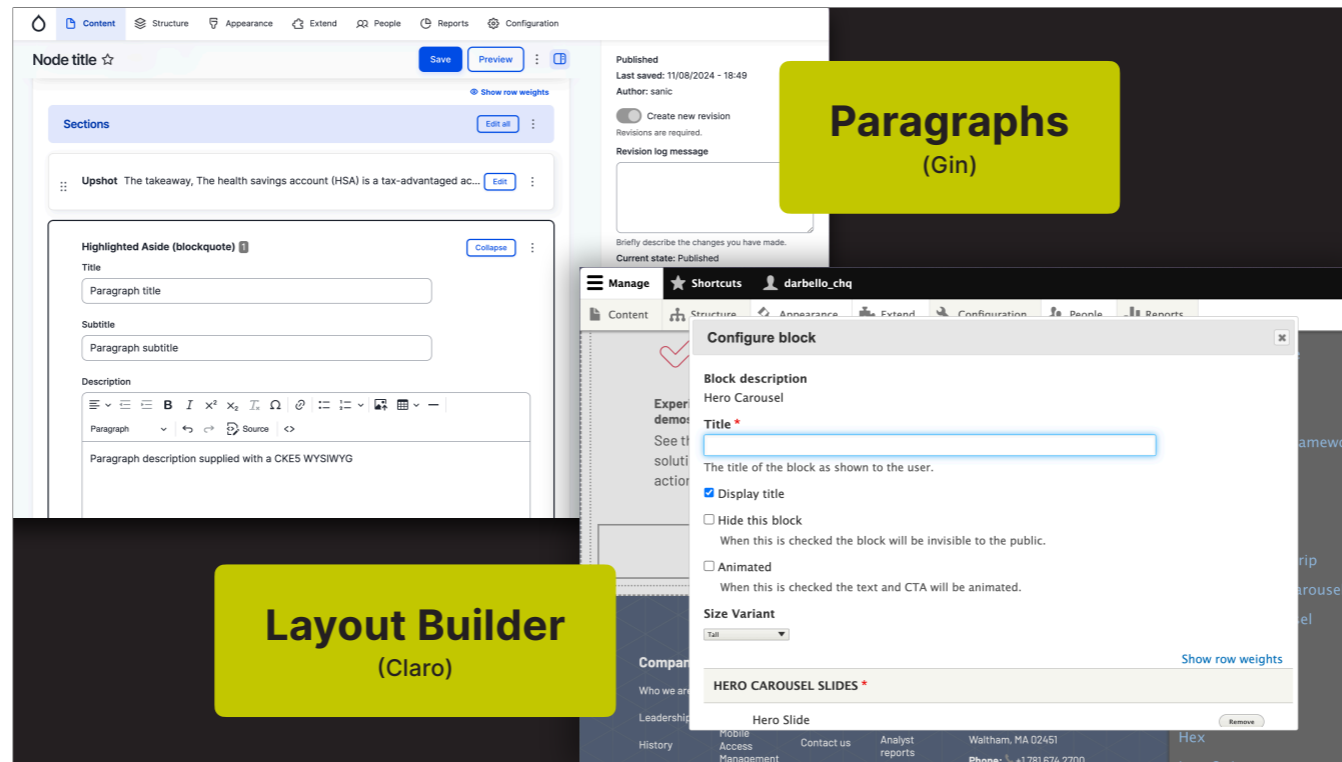
Slide	slide	Inline Image, Background Image
Hero Slide	hero	Title, Description, Image, CTA, Background Image, Lottie animations
Hub Node	hubnode	Category, Flag,

Animated	hero_b_animated	Boolean
Hero Carousel Slides	field_b_hero_slide	Entity reference revisions Reference type: Paragraph Paragraph type: Hero Slide
Hide this block	field_b_visibility	Boolean

CHROMATIC

DAN

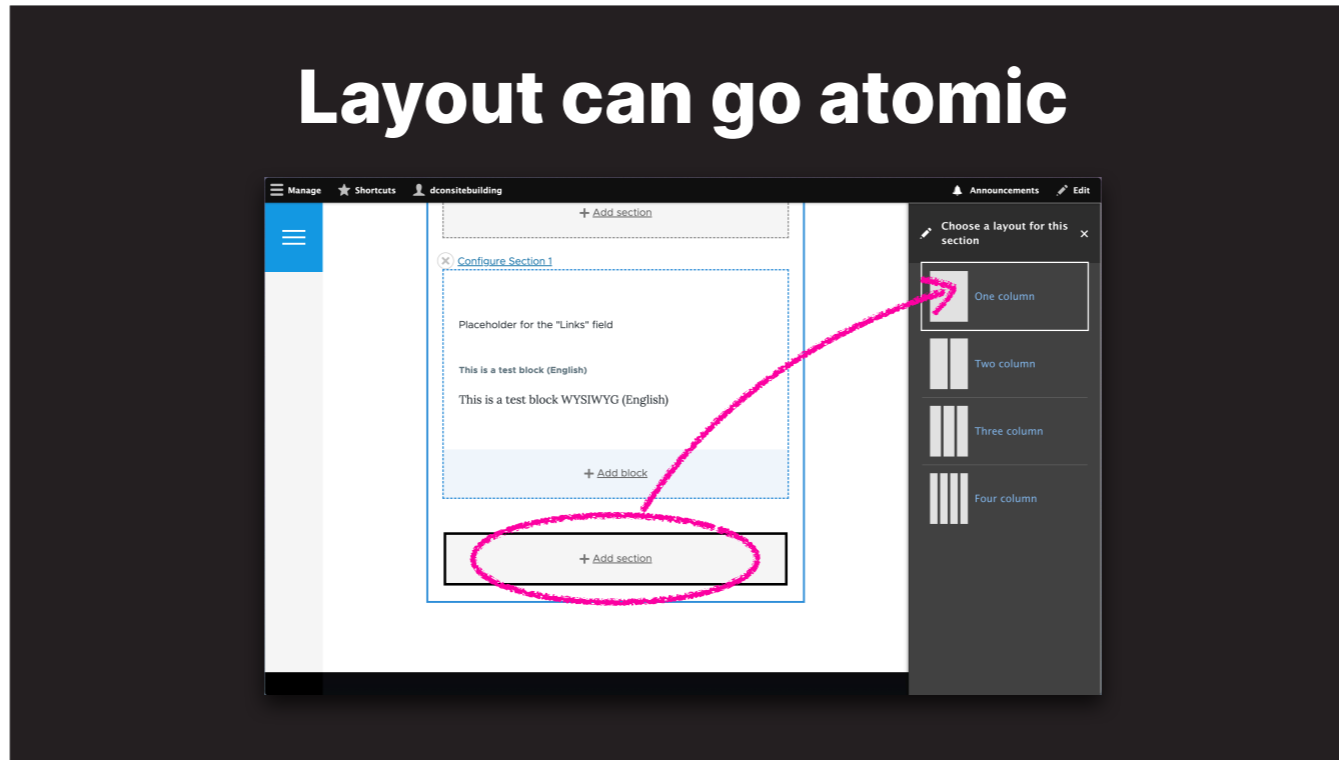
- * Secondly, you'll likely find that using Layout Builder does not inherently mean scrapping paragraphs.
- * Some inline blocks will still benefit from paragraphs' unique usage and structure.
- * For instance, if you're creating a carousel slider that accepts an array of slides,
- * the slides might be best served as a nested paragraph type.



DAN

- * Admin themes aside, the actual form display UI is exactly the same between paragraphs in the Node edit view and inline blocks in the Layout edit view.
- * While you can generate a preview link for nodes, Layout shows the themed blocks right up front.
- * This saves you from having to switch back and forth between preview link and edit view for paragraphs.

Layout can go atomic



DAN

- * Layout provides a way to go atomic right out-of-the-box.
- * You can create inline blocks that are singular (like a Card vs Cards)
- * And then allow content editors to decide how many columns (or what precisely those columns are: like aside vs main) they want and place blocks in each.
- * To be fair, it looks like you can do this for Paragraphs with the Layout Paragraphs module, as well.

Paragraphs are translatable


Translations of Card Paragraphs type ☆

Edit Manage fields Manage form display Manage display Translate paragraphs type

Operations

Request translation Add to cart There are 0 items in the [translation cart](#).

<input type="checkbox"/> Language	Pending Translations	Operations
<input type="checkbox"/> English (original)	Source	Edit
<input type="checkbox"/> Spanish	▲	Add
<input checked="" type="checkbox"/> French	None	Add
<input type="checkbox"/> Afrikaans	None	Add
<input type="checkbox"/> United Kingdom	None	Add



DAN

- * Lastly, and probably more importantly when comparing the two,
- * Paragraph can actually be translated via TMGMT.
- * That might wrap all of this talk up for you if you're on paragraphs and trying to decide whether to wait for XB in a year or so or LB right now.



Download a PDF of the talk
at chromatichq.com

 **Thank you.**