2018 GoalGorilla

Maintaining a Drupal Module: Keep It Small and Simple

goalgorilla

Who am I?

drupal.org/user/1868952

- Alexander Varwijk
- Developer at GoalGorilla for Open Social

- On Drupal.org for 6 years 17 days

- @Kingdutch on Twitter

- Site building since shortly after the Drupal 7 release
- Author of the IMCE Plupload module (integrating Plupload with IMCE)

- Maintainer of the Real-Time SEO module

Real-Time SEO for Drupal

What is it? Where did it come from?

drupal.org/project/yoast_seo

Originally built in a collaboration with Yoast.
Maintained by GoalGorilla

Analyses your content to provide feedback about common issues affecting search ranking (SEO)

Initial module release in September 2015
I took on maintenance in April 2017

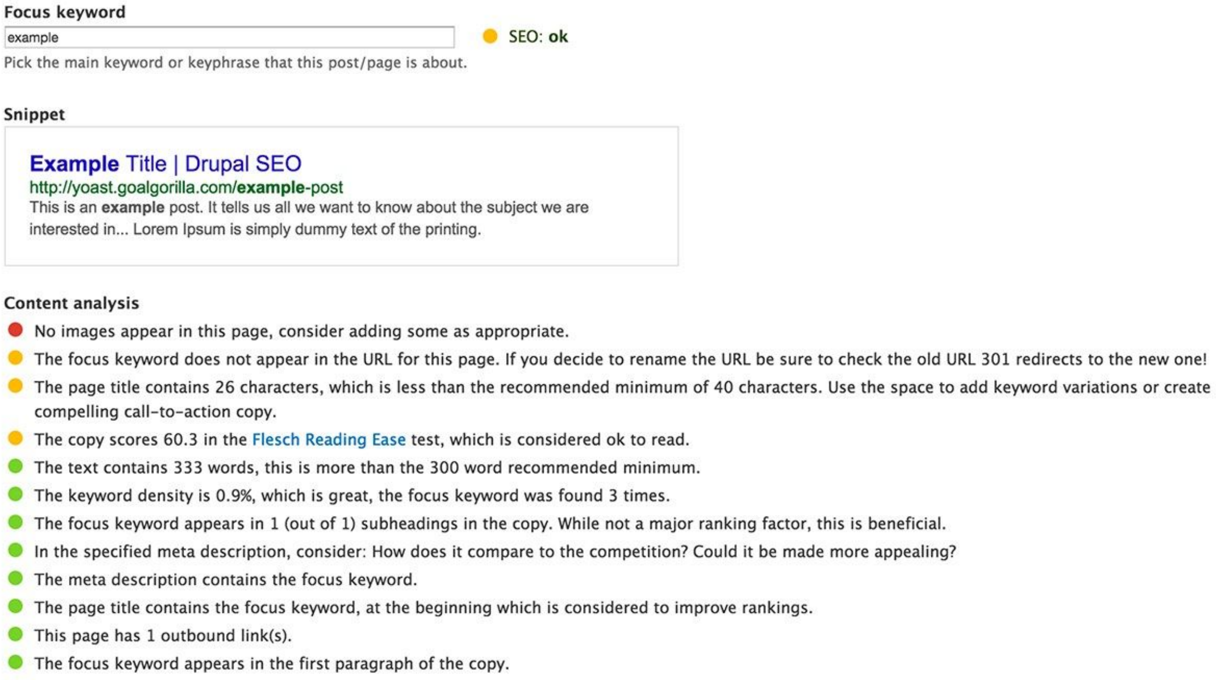
Real-Time SEO for Drupal

How does it work?

YoastSEO.js analysis library

Pass field data to analysis library using javascript

Fetches data from the fields directly on the form



Focus keyword

example ● SEO: **ok**

Pick the main keyword or keyphrase that this post/page is about.

Snippet

Example Title | Drupal SEO
http://yoast.goalgorilla.com/**example**-post
This is an **example** post. It tells us all we want to know about the subject we are interested in... Lorem Ipsum is simply dummy text of the printing.

Content analysis

- ● No images appear in this page, consider adding some as appropriate.
- ● The focus keyword does not appear in the URL for this page. If you decide to rename the URL be sure to check the old URL 301 redirects to the new one!
- ● The page title contains 26 characters, which is less than the recommended minimum of 40 characters. Use the space to add keyword variations or create compelling call-to-action copy.
- ● The copy scores 60.3 in the Flesch Reading Ease test, which is considered ok to read.
- ● The text contains 333 words, this is more than the 300 word recommended minimum.
- ● The keyword density is 0.9%, which is great, the focus keyword was found 3 times.
- ● The focus keyword appears in 1 (out of 1) subheadings in the copy. While not a major ranking factor, this is beneficial.
- ● In the specified meta description, consider: How does it compare to the competition? Could it be made more appealing?
- ● The meta description contains the focus keyword.
- ● The page title contains the focus keyword, at the beginning which is considered to improve rankings.
- ● This page has 1 outbound link(s).
- ● The focus keyword appears in the first paragraph of the copy.

Version 1.x

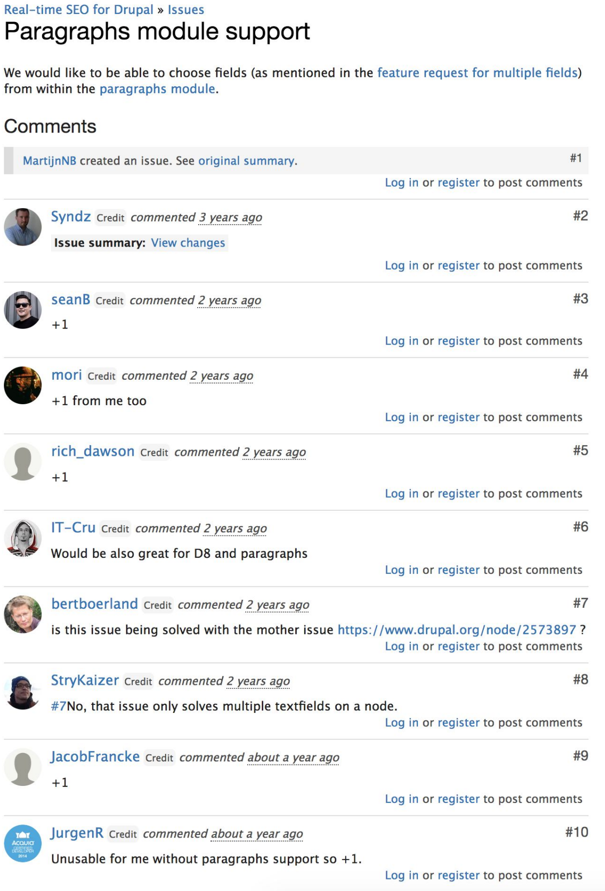
- Nodes only
- Needed a text field as body
- No support for popular modules such as Paragraphs

- No support for images embedded with Media (e.g. through CKEditor)
- No support for images outside of the main content field

- Worked only with the Path and Metatag field *visible* on the Node create/edit form

- Enabled by default (for all content types)
- Enabled/disabled with a settings page

Plenty of work to be done



Real-time SEO for Drupal » Issues

Paragraphs module support

We would like to be able to choose fields (as mentioned in the feature request for multiple fields) from within the paragraphs module.

Comments

MartijnNB created an issue. See original summary. #1
Log in or register to post comments

Syndz Credit *commented 3 years ago* #2
Issue summary: View changes
Log in or register to post comments

seanB Credit *commented 2 years ago* #3
+1
Log in or register to post comments

mori Credit *commented 2 years ago* #4
+1 from me too
Log in or register to post comments

rich_dawson Credit *commented 2 years ago* #5
+1
Log in or register to post comments

IT-Cru Credit *commented 2 years ago* #6
Would be also great for D8 and paragraphs
Log in or register to post comments

bertboerland Credit *commented 2 years ago* #7
is this issue being solved with the mother issue https://www.drupal.org/node/2573897 ?
Log in or register to post comments

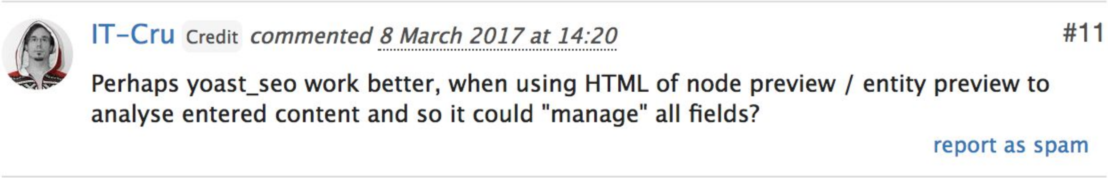
StryKaizer Credit *commented 2 years ago* #8
#7No, that issue only solves multiple textfields on a node.
Log in or register to post comments

JacobFrancke Credit *commented about a year ago* #9
+1
Log in or register to post comments

JurgenR Credit *commented about a year ago* #10
Unusable for me without paragraphs support so +1.
Log in or register to post comments

Support for the Paragraphs module

Analyse rendered content, not fields

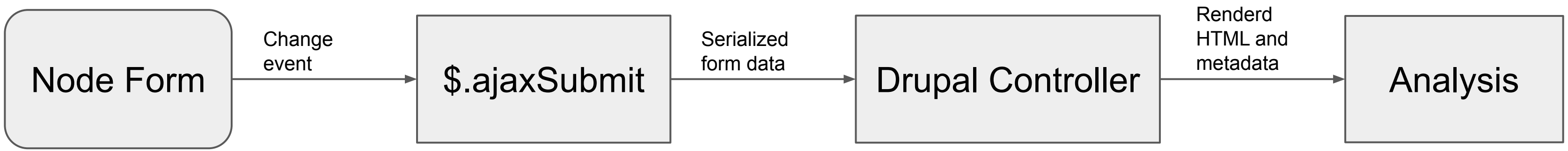


IT-Cru Credit *commented 8 March 2017 at 14:20* #11

Perhaps yoast_seo work better, when using HTML of node preview / entity preview to analyse entered content and so it could "manage" all fields?

report as spam

YoastSEO.js (the underlying library) is pretty great
It can handle an HTML snippet with some metadata



EntityPreviewController

How not to use the Drupal 8 Form API

commit: c2c8e2b

yoast_seo.routing.yml

```yaml
yoast_seo.entity_preview:
 path: '/yoast_seo/preview'
 defaults:
   _title: 'Entity Preview'
   _controller: '\Drupal\yoast_seo\Controller\EntityPreviewController::json'
 methods: ['POST']
```

EntityPreviewController

```php
public function json(Request $request) {
   $entity = $this->getEntityForRequest($request);
   $preview_data = $this->entityPreviewer->createEntityPreview($entity);

   // ... snip alias override ...

   return new JsonResponse($preview_data);
}
```

EntityPreviewController

```php
/**
 * Returns an instantiated preview entity for the request.
 *
 * @param \Symfony\Component\HttpFoundation\Request $request
 *   The request that contains the POST body for this preview.
 *
 * @throws \Symfony\Component\HttpKernel\Exception\BadRequestHttpException
 *   Too many reasons for this slide.
 *
 * @return \Drupal\Core\Entity\Entity
 *   The instantiated entity.
 */
protected function getEntityForRequest(Request $request) {
   // Fetch all our post data.
   $content = $request->request->all();

   // … snip lots of array existence checking ...

   $preview_context = $content['yoast_seo_preview'];
   unset($content['yoast_seo_preview']);

   $form_data = $content;

   return $this->entityPreviewer->entityFromFormSubmission(
      $preview_context['action'], $preview_context['method'], $form_data
   );
}
```

Intermezzo - Form API

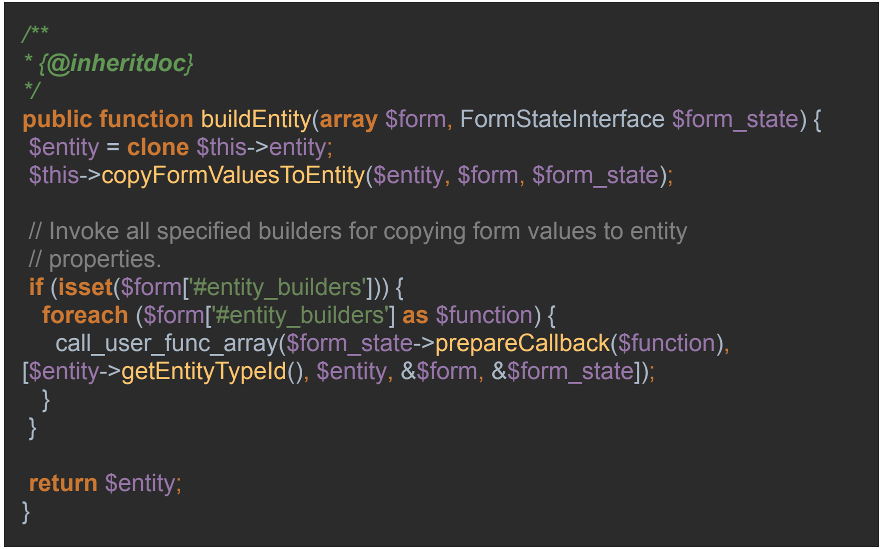
What does it do?

```
Request
-> HtmlEntityFormController::getContentResult
  -> construct form object and form state
  -> FormBuilder::buildForm
    -> Prepare user input
    -> Handle caching depending on request type
    -> FormBuilder::retrieveForm - get’s the form array from the form callback
    -> FormBuilder::prepareForm - Adds required elements, executes any hook_form_alter functions, and optionally inserts a validation token to prevent tampering.
    -> FormBuilder::processForm
      -> FormBuilder::doBuildForm - finishes building the form by calling element #process functions and mapping user input properties and $form_state->getValues()
        -> Complex recursive input handling
      -> Only if isProcessingInput()
        -> validateForm and if validation is correct then submit
  -> Return Drupal Render array
```

Intermezzo - EntityForm

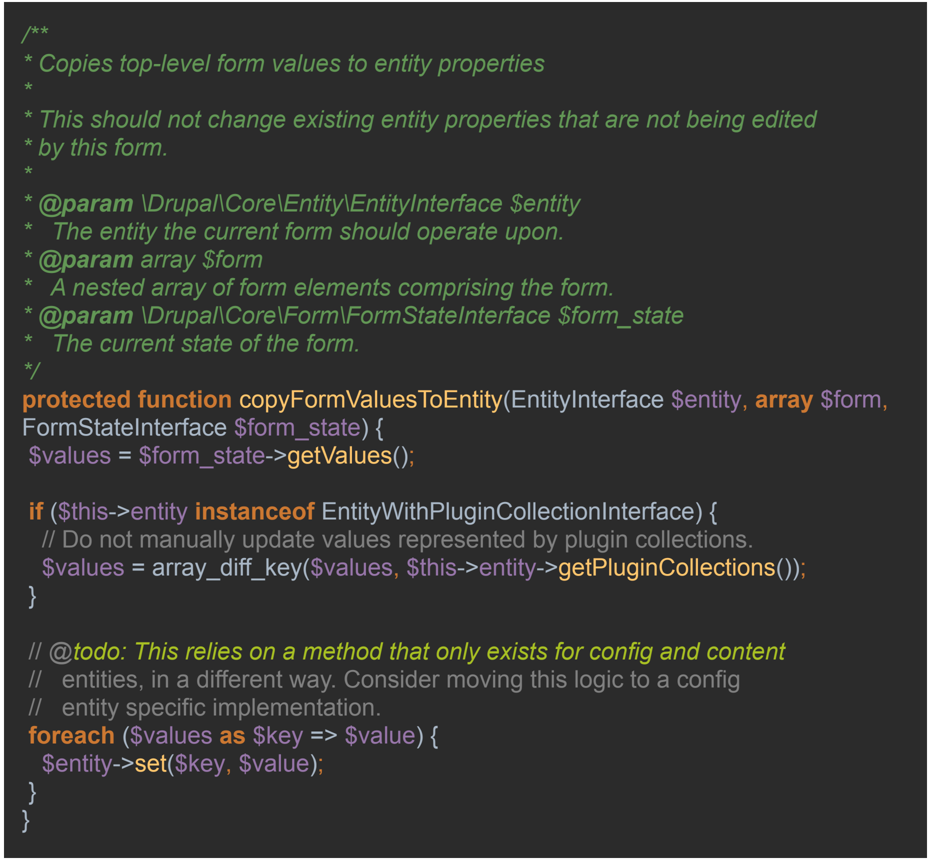
From form to entity

EntityForm



```
/**
* {@inheritdoc}
*/
public function buildEntity(array $form, FormStateInterface $form_state) {
 $entity = clone $this->entity;
 $this->copyFormValuesToEntity($entity, $form, $form_state);

 // Invoke all specified builders for copying form values to entity
 // properties.
 if (isset($form['#entity_builders'])) {
   foreach ($form['#entity_builders'] as $function) {
     call_user_func_array($form_state->prepareCallback($function),
[$entity->getEntityTypeId(), $entity, &$form, &$form_state]);
   }
 }

 return $entity;
}
```

EntityForm



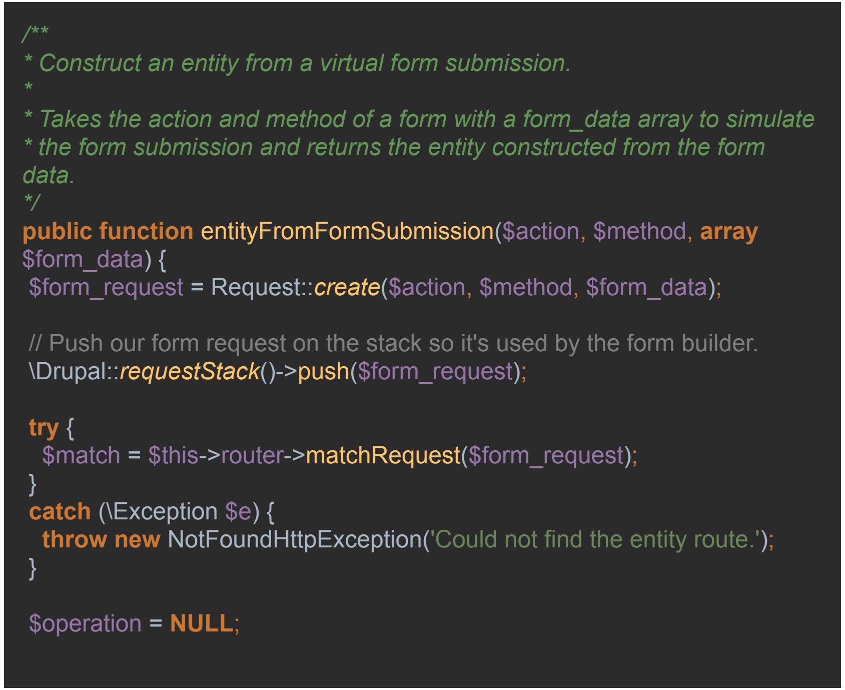
```
/**
* Copies top-level form values to entity properties
*
* This should not change existing entity properties that are not being edited
* by this form.
*
* @param \Drupal\Core\Entity\EntityInterface $entity
*   The entity the current form should operate upon.
* @param array $form
*   A nested array of form elements comprising the form.
* @param \Drupal\Core\Form\FormStateInterface $form_state
*   The current state of the form.
*/
protected function copyFormValuesToEntity(EntityInterface $entity, array $form,
FormStateInterface $form_state) {
 $values = $form_state->getValues();

 if ($this->entity instanceof EntityWithPluginCollectionInterface) {
   // Do not manually update values represented by plugin collections.
   $values = array_diff_key($values, $this->entity->getPluginCollections());
 }

 // @todo: This relies on a method that only exists for config and content
 //   entities, in a different way. Consider moving this logic to a config
 //   entity specific implementation.
 foreach ($values as $key => $value) {
   $entity->set($key, $value);
 }
}
```

EntityPreviewController

commit: c2c8e2b

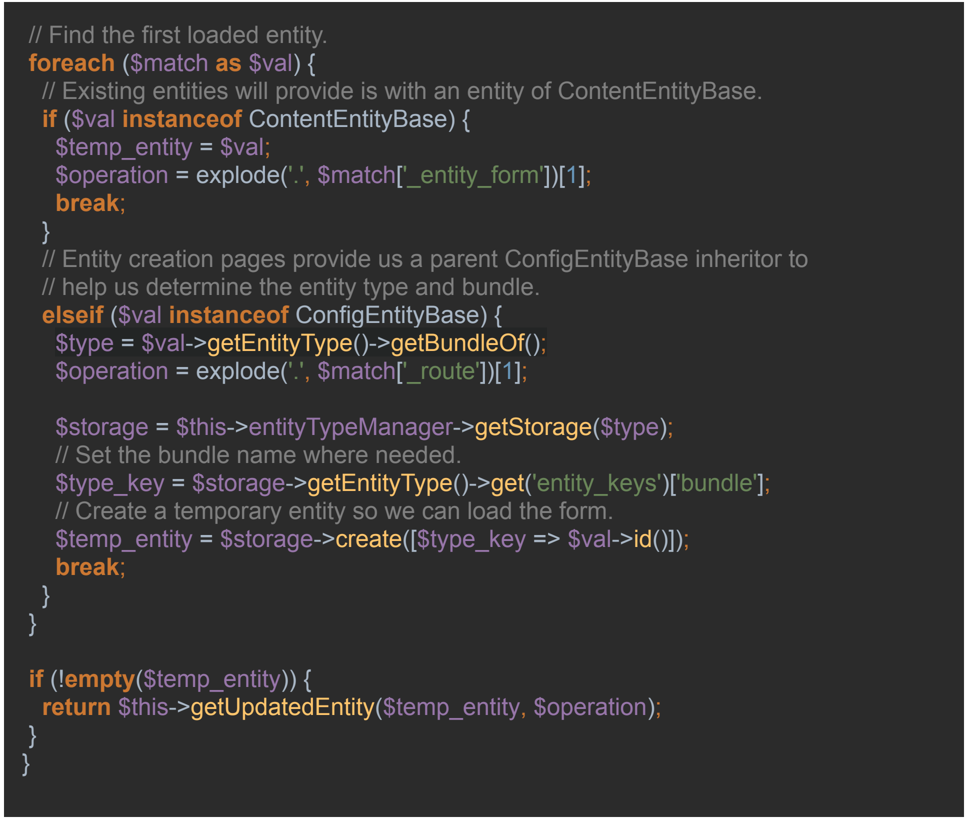
How not to use the Drupal 8 Form API

EntityPreviewController



```
/**
* Construct an entity from a virtual form submission.
*
* Takes the action and method of a form with a form_data array to simulate
* the form submission and returns the entity constructed from the form
data.
*/
public function entityFromFormSubmission($action, $method, array
$form_data) {
  $form_request = Request::create($action, $method, $form_data);

  // Push our form request on the stack so it's used by the form builder.
  \Drupal::requestStack()->push($form_request);

  try {
    $match = $this->router->matchRequest($form_request);
  }
  catch (\Exception $e) {
    throw new NotFoundHttpException('Could not find the entity route.');
  }

  $operation = NULL;
```

contd.



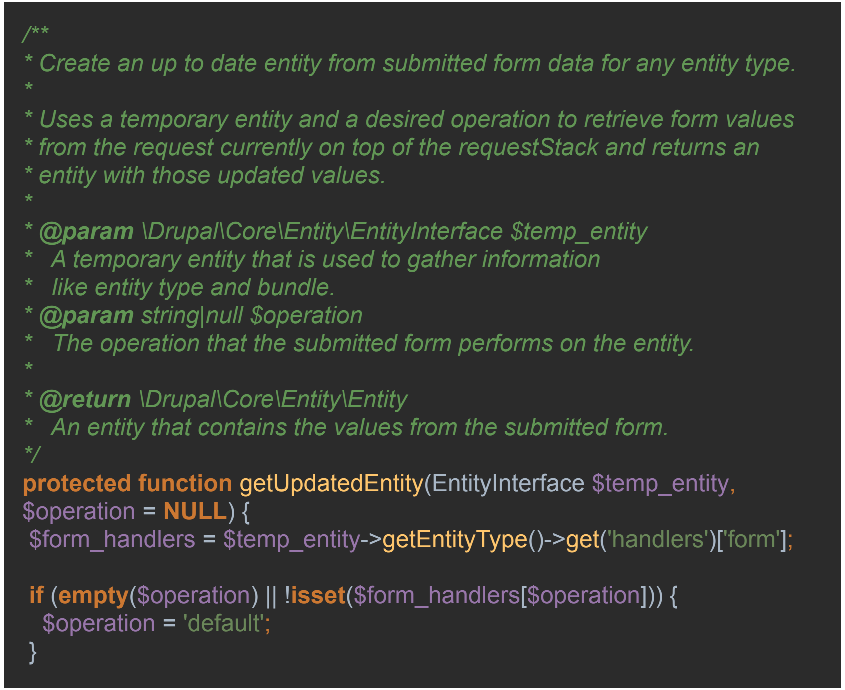
```
// Find the first loaded entity.
foreach ($match as $val) {
  // Existing entities will provide is with an entity of ContentEntityBase.
  if ($val instanceof ContentEntityBase) {
    $temp_entity = $val;
    $operation = explode('.', $match['_entity_form'])[1];
    break;
  }
  // Entity creation pages provide us a parent ConfigEntityBase inheritor to
  // help us determine the entity type and bundle.
  elseif ($val instanceof ConfigEntityBase) {
    $type = $val->getEntityType()->getBundleOf();
    $operation = explode('.', $match['_route'])[1];

    $storage = $this->entityTypeManager->getStorage($type);
    // Set the bundle name where needed.
    $type_key = $storage->getEntityType()->get('entity_keys')['bundle'];
    // Create a temporary entity so we can load the form.
    $temp_entity = $storage->create([$type_key => $val->id()]);
    break;
  }
}

if (!empty($temp_entity)) {
  return $this->getUpdatedEntity($temp_entity, $operation);
}
}
```

EntityPreviewController

commit: c2c8e2b

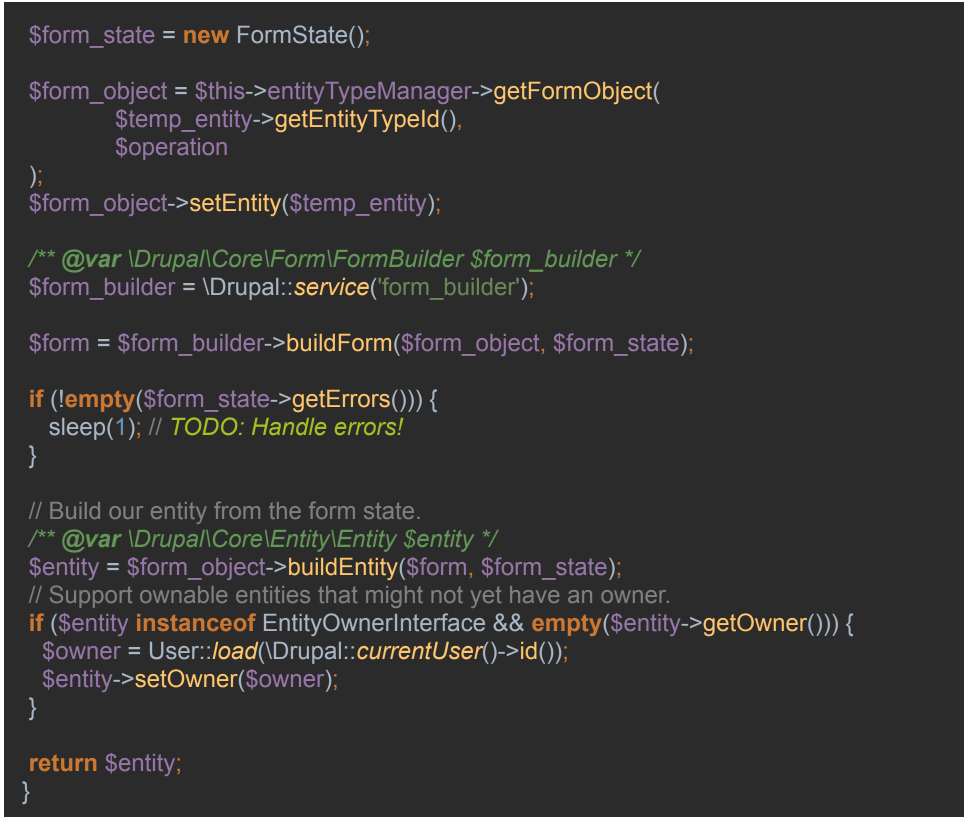
How not to use the Drupal 8 Form API

EntityPreviewController



```
/**
* Create an up to date entity from submitted form data for any entity type.
*
* Uses a temporary entity and a desired operation to retrieve form values
* from the request currently on top of the requestStack and returns an
* entity with those updated values.
*
* @param \Drupal\Core\Entity\EntityInterface $temp_entity
*   A temporary entity that is used to gather information
*   like entity type and bundle.
* @param string|null $operation
*   The operation that the submitted form performs on the entity.
*
* @return \Drupal\Core\Entity\Entity
*   An entity that contains the values from the submitted form.
*/
protected function getUpdatedEntity(EntityInterface $temp_entity,
$operation = NULL) {
  $form_handlers = $temp_entity->getEntityType()->get('handlers')['form'];

  if (empty($operation) || !isset($form_handlers[$operation])) {
    $operation = 'default';
  }
```

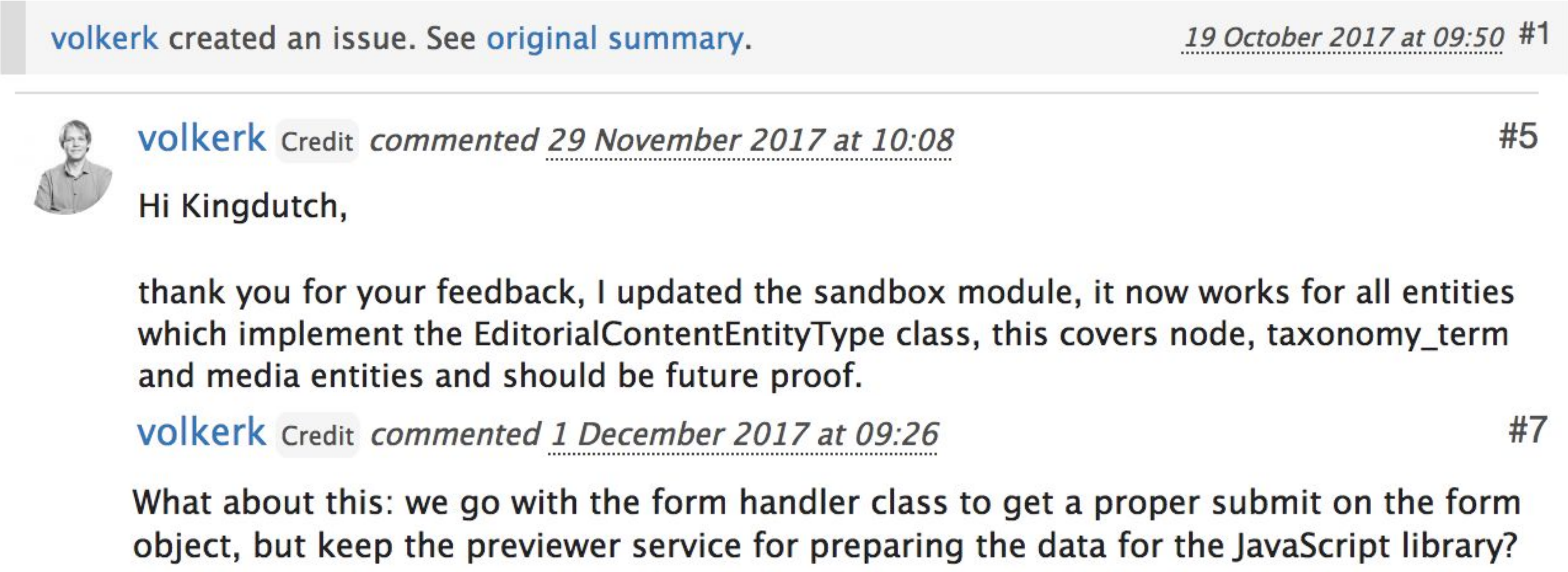
contd.



```
$form_state = new FormState();

$form_object = $this->entityTypeManager->getFormObject(
        $temp_entity->getEntityTypeId(),
        $operation
);
$form_object->setEntity($temp_entity);

/** @var \Drupal\Core\Form\FormBuilder $form_builder */
$form_builder = \Drupal::service('form_builder');

$form = $form_builder->buildForm($form_object, $form_state);

if (!empty($form_state->getErrors())) {
  sleep(1); // TODO: Handle errors!
}

// Build our entity from the form state.
/** @var \Drupal\Core\Entity\Entity $entity */
$entity = $form_object->buildEntity($form, $form_state);
// Support ownable entities that might not yet have an owner.
if ($entity instanceof EntityOwnerInterface && empty($entity->getOwner())) {
  $owner = User::load(\Drupal::currentUser()->id());
  $entity->setOwner($owner);
}

return $entity;
}
```

Not that kind of Thunder

A suggestion for a better solution



volkerk created an issue. See original summary. *19 October 2017 at 09:50* #1

volkerk Credit *commented 29 November 2017 at 10:08* #5

Hi Kingdutch,

thank you for your feedback, I updated the sandbox module, it now works for all entities which implement the EditorialContentEntityType class, this covers node, taxonomy_term and media entities and should be future proof.

volkerk Credit *commented 1 December 2017 at 09:26* #7

What about this: we go with the form handler class to get a proper submit on the form object, but keep the previewer service for preparing the data for the JavaScript library?

KISS: Keep It Simple Stupid

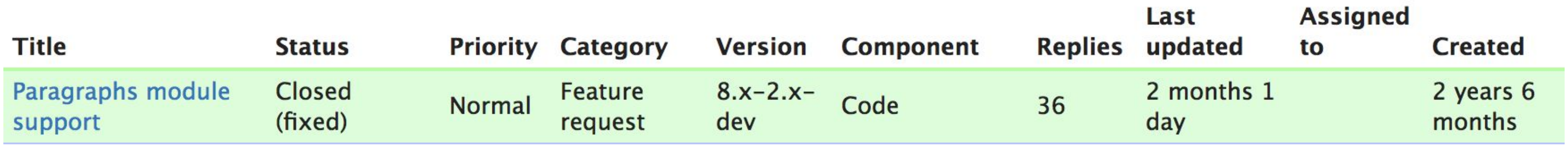
Don’t do it yourself. Use Drupal 8 APIs - The New Solution

Use a Form Handler and Form API!

```
// It starts with a class.
class AnalysisFormHandler implements EntityHandlerInterface;

// In a hook_entity_type_build() implementation.
$entity_type->setHandlerClass(
        'yoast_seo_preview',
        AnalysisFormHandler::class
);

// In the Real-Time SEO field widget.
$target_type = $this->fieldDefinition->getTargetEntityTypeId();
if ($this->entityTypeManager->hasHandler($target_type, 'yoast_seo_preview')) {
 $form_handler = $this->entityTypeManager
                                        ->getHandler($target_type, 'yoast_seo_preview');

 if ($form_handler instanceof AnalysisFormHandler) {
   $element['yoast_seo_preview_button'] = [
     '#type' => 'button',
     '#ajax' => [ 'callback' => [$this, 'analysisSubmitAjax'] ],
   ];
 }
}
```

AnalysisFormHandler

```
public function analysisSubmitAjax(array &$form, FormStateInterface
$form_state) {
 $preview_entity = $form_state->getFormObject()
                                                  ->buildEntity($form, $form_state);
 $preview_entity->in_preview = TRUE;

 $entity_data = $this->entityAnalyser->createEntityPreview($preview_entity);

 // The current value of the alias field, if any,
 // takes precedence over the entity url.
 $user_input = $form_state->getUserInput();
 if (!empty($user_input['path'][0]['alias'])) {
   $entity_data['url'] = $user_input['path'][0]['alias'];
 }

 drupal_get_messages();

 $response = new AjaxResponse();
 $response->addCommand(
     new InvokeCommand('body', 'trigger', ['updateSeoData', $entity_data])
 );
 return $response;
}
```



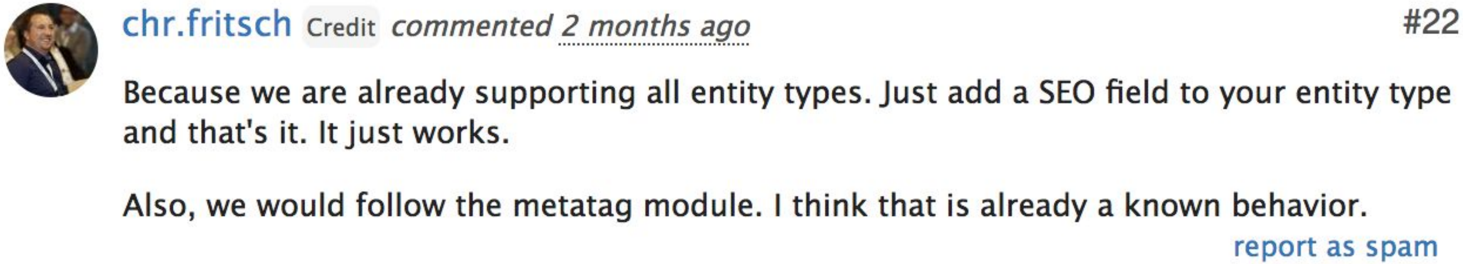
Title	Status	Priority	Category	Version	Component	Replies	Last updated	Assigned to	Created
Paragraphs module support	Closed (fixed)	Normal	Feature request	8.x-2.x-dev	Code	36	2 months 1 day		2 years 6 months

From configuration page to Field API

Don't do it yourself. Use Drupal 8 APIs

Configuration form + custom FieldManager class to manage which entities (nodes) were analysed

Suggested by Christian Fritsch at Thunder



chr.fritsch Credit *commented 2 months ago* #22

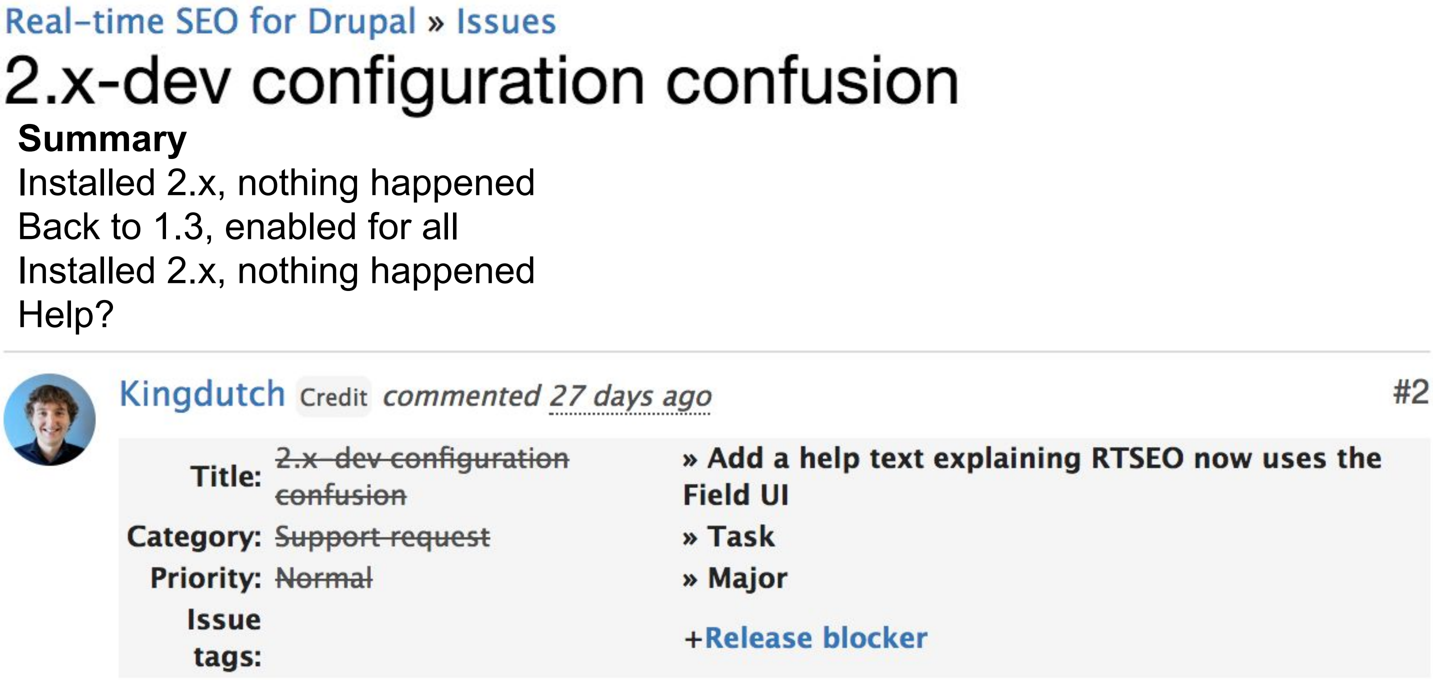
Because we are already supporting all entity types. Just add a SEO field to your entity type and that's it. It just works.

Also, we would follow the metatag module. I think that is already a known behavior.

9 files changed, 67 insertions(+), 531 deletions(-)
(Including a new Empty FieldFormatter)

From configuration page to Field API

You have 1 new issue



Real-time SEO for Drupal » Issues

2.x-dev configuration confusion

Summary

Installed 2.x, nothing happened
Back to 1.3, enabled for all
Installed 2.x, nothing happened
Help?

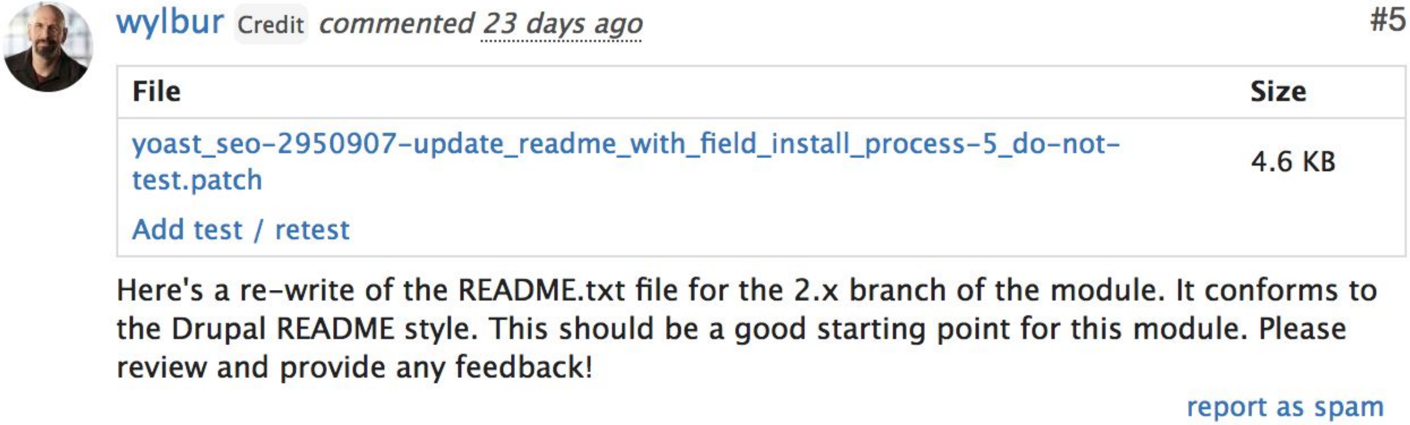
Kingdutch Credit *commented 27 days ago* #2

Title:	~~2.x-dev configuration confusion~~	**» Add a help text explaining RTSEO now uses the Field UI**
Category:	~~Support request~~	**» Task**
Priority:	~~Normal~~	**» Major**
Issue tags:		**+Release blocker**

From configuration page to Field API

Drupal issue queue to the rescue

Just 4 days later
Contribution from Wilbur Ince



wylbur Credit *commented 23 days ago* #5

File	Size
yoast_seo-2950907-update_readme_with_field_install_process-5_do-not-test.patch Add test / retest	4.6 KB

Here's a re-write of the README.txt file for the 2.x branch of the module. It conforms to the Drupal README style. This should be a good starting point for this module. Please review and provide any feedback!

report as spam

Version 2.x

- Disabled by default
- Add using the Field API
- Supports any fieldable Entity
- Supports any field type out of the box
- ~~Worked only with the Path and Metatag field *visible* on the Node create/edit form~~

- Support for images embedded with Media
- Support for images outside of the main content field
- Support for popular modules such as Paragraphs

Summary

Try the Drupal way! Issue queue collaboration is awesome.

- If you’re writing a lot of code, it could be that you missed something
- Think about whether you’re not duplicating Drupal functionality
- The Drupal way can be difficult but it’s very rewarding (new field support for free)
- Use the Drupal issue queue, it helps!
    - Contribute to existing issues
    - Ask questions the maintainer may not have thought of
    - Suggest new features

Things may take a while, but we will get there!

What’s next?

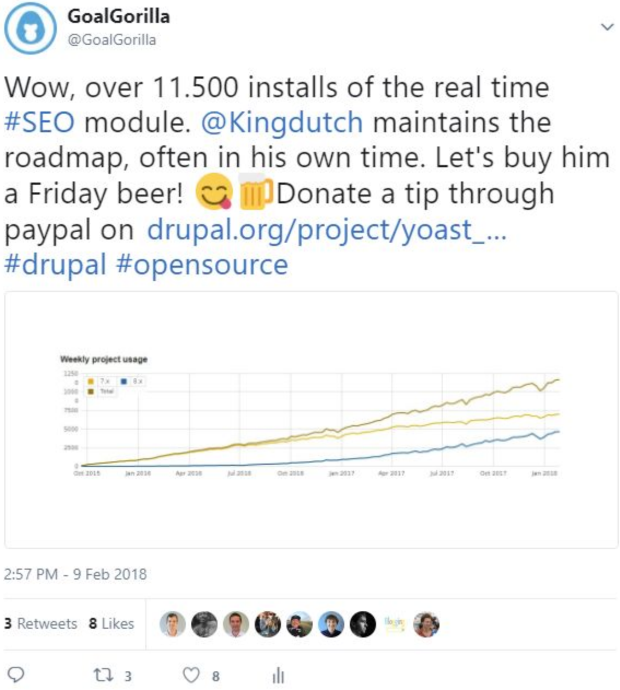
Roadmap

- Support for Social Media
  - Preview/adjust how your content looks when shared
- schema.org support
  - Preview rich snippets that are found in your content such as recipes, reviews or contact information
- Feedback in languages other than English
  - Provide a better experience to content editors who don’t use English for their content

Help out! Contribute in the issue queue or ...

Funding open source is hard

Make it possible for me to spend more time on this



GoalGorilla
@GoalGorilla

Wow, over 11.500 installs of the real time #SEO module. @Kingdutch maintains the roadmap, often in his own time. Let's buy him a Friday beer! 😋🍺Donate a tip through paypal on drupal.org/project/yoast_... #drupal #opensource

2:57 PM - 9 Feb 2018

3 Retweets 8 Likes

open collective
A NEW FORM OF ASSOCIATION, TRANSPARENT BY DESIGN
Open your finances to your community

https://opencollective.com/realtimeseo/

Become a backer for $100 / year

Thanks!

- To everyone who contributes in the issue queue
- To everyone who uses the module
- Everyone who helped build the first version of the module
- My colleagues in preparing for this presentation
- You for attending!