DRUPAL CAMP ASHEVILLE

Custom Block Types

Custom Block Types

Jonathan Daggerhart

- **Developer at Hook 42**
- **Organizer for Drupal Camp Asheville**

Drupal.org: daggerhart

Twitter: @daggerhart

Blog: https://www.daggerhart.com

Drupal Camp Asheville

Site: https://drupalasheville.com

Twitter: @drupalasheville

Custom Block Types

What we will cover

1. A Complete Module w/ Custom Block Type
2. Annotations & Plugins
3. Annotations vs Drupal 7 Hooks
4. Block Configuration Form

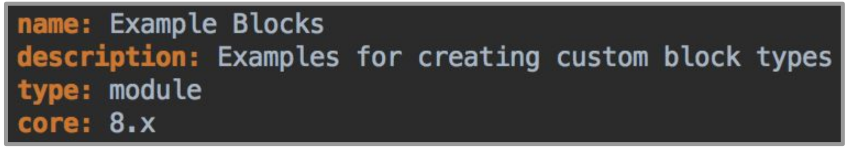
Custom Block Types

A Complete Module & Block

In Drupal 8 Blocks are “Plugins”.

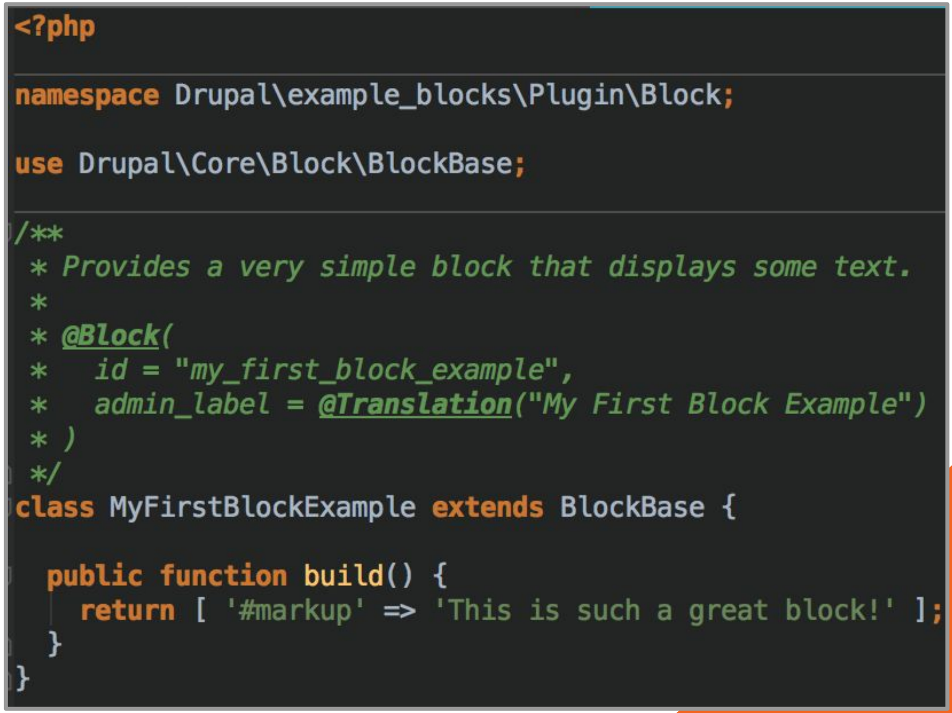
Plugins are a new concept in Drupal 8. They are classes that are automatically discovered by the system if they are in the correct location & have the correct Annotation. Generally, Plugins with Annotations have replaced the `hook_something_info` concept from Drupal 7.

`example_blocks.info.yml`



```yaml
name: Example Blocks
description: Examples for creating custom block types
type: module
core: 8.x
```

`src/Plugin/Block/MyFirstBlockExample.php`



```php
<?php

namespace Drupal\example_blocks\Plugin\Block;

use Drupal\Core\Block\BlockBase;

/**
 * Provides a very simple block that displays some text.
 *
 * @Block(
 *   id = "my_first_block_example",
 *   admin_label = @Translation("My First Block Example")
 * )
 */
class MyFirstBlockExample extends BlockBase {

  public function build() {
    return [ '#markup' => 'This is such a great block!' ];
  }
}
```

Custom Block Types

Annotations & Plugins

Annotations tell the plugin system about the plugin you're creating.

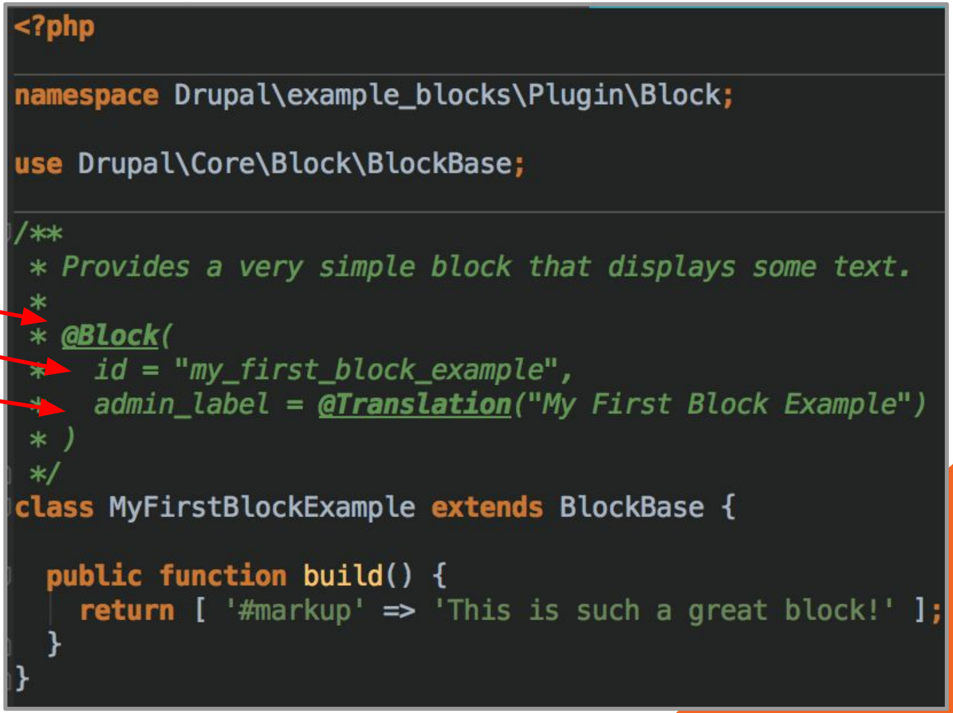
This annotation is telling Drupal that:

- This is a Block
- Its ID is my_first_block_example
- It's label My First Block Example

This block class does two important things:

- Extends the BlockBase
- Contains a build() method

`src/Plugin/Block/MyFirstBlockExample.php`



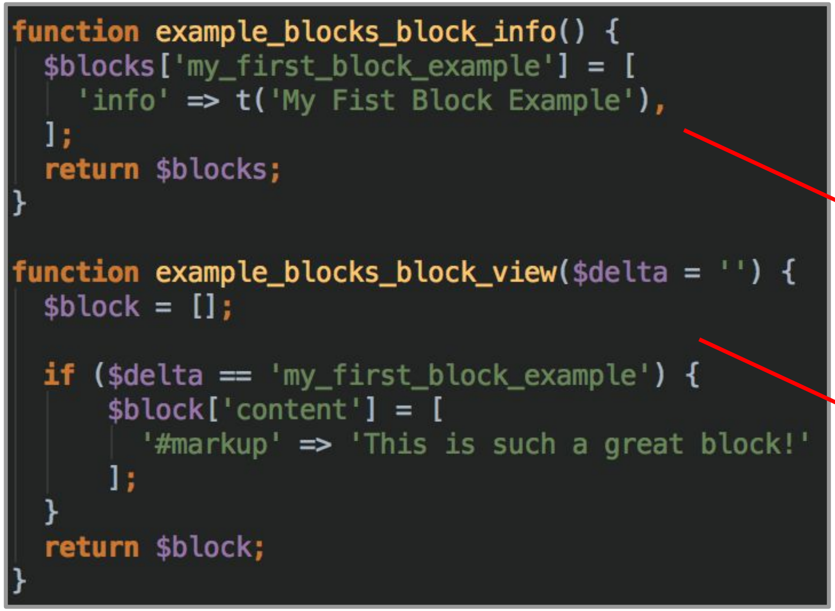
```php
<?php

namespace Drupal\example_blocks\Plugin\Block;

use Drupal\Core\Block\BlockBase;

/**
 * Provides a very simple block that displays some text.
 *
 * @Block(
 *   id = "my_first_block_example",
 *   admin_label = @Translation("My First Block Example")
 * )
 */
class MyFirstBlockExample extends BlockBase {

  public function build() {
    return [ '#markup' => 'This is such a great block!' ];
  }
}
```

Custom Block Types

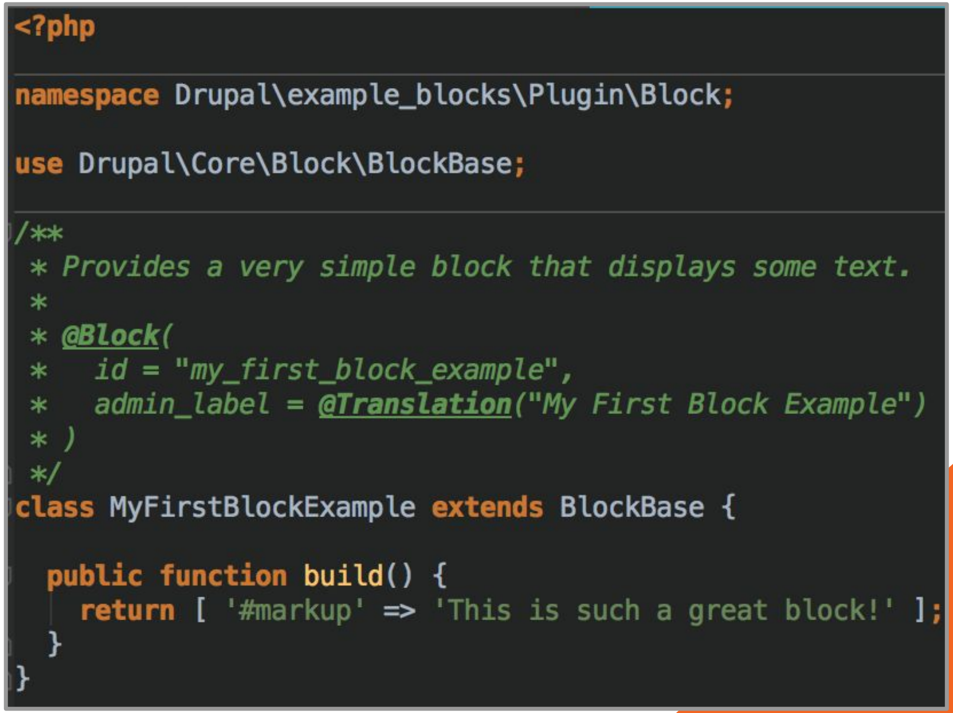
Annotations vs Drupal 7 Hooks

src/Plugin/Block/MyFirstBlockExample.php

Drupal 7 custom block



```php
function example_blocks_block_info() {
  $blocks['my_first_block_example'] = [
    'info' => t('My Fist Block Example'),
  ];
  return $blocks;
}

function example_blocks_block_view($delta = '') {
  $block = [];

  if ($delta == 'my_first_block_example') {
      $block['content'] = [
        '#markup' => 'This is such a great block!'
      ];
  }
  return $block;
}
```



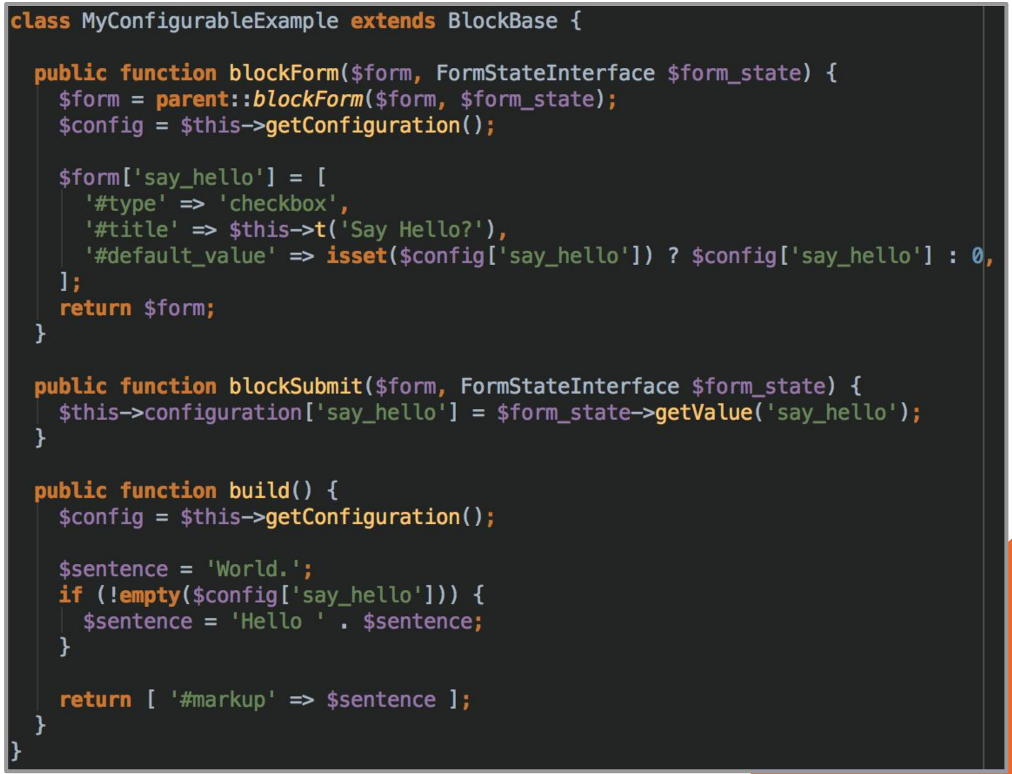
```php
<?php

namespace Drupal\example_blocks\Plugin\Block;

use Drupal\Core\Block\BlockBase;

/**
 * Provides a very simple block that displays some text.
 *
 * @Block(
 *   id = "my_first_block_example",
 *   admin_label = @Translation("My First Block Example")
 * )
 */
class MyFirstBlockExample extends BlockBase {

  public function build() {
    return [ '#markup' => 'This is such a great block!' ];
  }
}
```

Custom Block Types

Block Configuration Forms

Blocks can provide configuration forms to the end user by implementing a few additional methods:

- `blockForm()`
  Return a Drupal form array
- `blockSubmit()`
  Get the submitted form values and set them within the configuration array.



```
class MyConfigurableExample extends BlockBase {

  public function blockForm($form, FormStateInterface $form_state) {
    $form = parent::blockForm($form, $form_state);
    $config = $this->getConfiguration();

    $form['say_hello'] = [
      '#type' => 'checkbox',
      '#title' => $this->t('Say Hello?'),
      '#default_value' => isset($config['say_hello']) ? $config['say_hello'] : 0,
    ];
    return $form;
  }

  public function blockSubmit($form, FormStateInterface $form_state) {
    $this->configuration['say_hello'] = $form_state->getValue('say_hello');
  }

  public function build() {
    $config = $this->getConfiguration();

    $sentence = 'World.';
    if (!empty($config['say_hello'])) {
      $sentence = 'Hello ' . $sentence;
    }

    return [ '#markup' => $sentence ];
  }
}
```