[Step by step method to create a custom form in Drupal 8](http://valuebound.com/resources/blog/step-by-step-method-to-create-a-custom-form-in-drupal-8)

1. Step by step method to create a custom form in Drupal 8

In Drupal 8 form API is similar to Drupal 7 Form API. forms still uses array structure to render the data. but having separate validation and submission form.Drupal 8 has some new (HTML 5) elements available.

New HTML 5 elements like

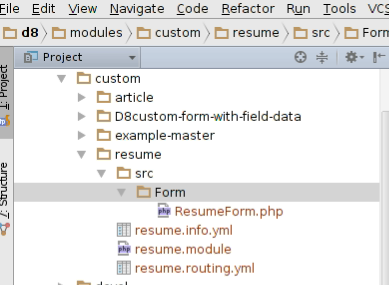
'tel','email','number','date','url','search','range' etc.

In Drupal 8 Form classes implement the \Drupal\Core\Form\FormBuilderInterface and the basic workflow of a form is defined by the buildForm, validateForm, and submitForm methods of the interface.

There are a different classes to choose depending on the type of form you are creating.

* **ConfigFormBase** : For creating system configuration forms like the one found at admin/config/system/site-information.
* **ConfirmFormBase** : For providing users with a form to confirm an action such as deleting a piece of content.
* **FormBase** : The most generic base class for generating forms.

FIle structure:



**Step 1:** Create .info.yml file

An essential part of a Drupal 8 module, theme, or install profile is the .info.yml file (aka, "info yaml file") to store metadata about the project.

In Drupal 8, .info file changes to .info.yml. Old .info files have been converted to YAML.

Added name, description, core, package, dependencies, type (The type key, which is new in Drupal 8, is required and indicates the type of extension, e.g. module, theme or profile.

|  |  |
| --- | --- |
|  | name: Resume |
|  | description: A Resume form to enter user details. |
|  | core: 8.x |
|  | package: Custom |
|  | type: module |

[**view raw**](https://gist.github.com/xaiwant/3dc9c3d7f2725b1e970f/raw/8dbb245a988acab80225b930cf46790dff3f7152/resume.info.yml)[**resume.info.yml**](https://gist.github.com/xaiwant/3dc9c3d7f2725b1e970f#file-resume-info-yml) hosted with ❤ by **[GitHub](https://github.com/)**

**Step 2:** Creating .routing.yml

This routing system replaces the routing parts of hook\_menu() in Drupal 7. The parts of hook\_menu() that were used for creating menu entries, tabs.

A route is a path which return some sort of content on.

For example, the default front page, '/node/2' is a route. When Drupal receives a request, it tries to match the requested path to a route it knows about. If the route is found, then the route's definition is used to return content. Otherwise, Drupal returns a 404.

|  |  |
| --- | --- |
|  | resume.form: |
|  | path: '/resume/myform' |
|  | defaults: |
|  | \_title: 'Application form' |
|  | \_form: '\Drupal\resume\Form\ResumeForm' |
|  | requirements: |
|  | \_permission: 'access content' |

[**view raw**](https://gist.github.com/xaiwant/957e41f55f65740a995a/raw/4ae97dd9dfd44746ff2852ae312bc58ca29a6812/resume.routing.yml)[**resume.routing.yml**](https://gist.github.com/xaiwant/957e41f55f65740a995a#file-resume-routing-yml) hosted with ❤ by **[GitHub](https://github.com/)**

In above code:



1. {name} element in the URL is a path parameter and is available as $name in the controller method. e.g: resume.form

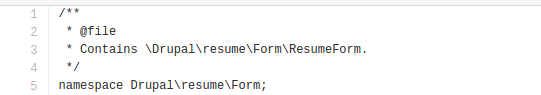
2. {path name} is url to access the page. e.g; /resume/my-form

3.  The {title} is title of the page. e.g: Application Form

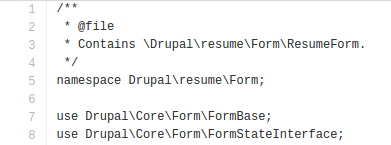
4.  {module-name} is the name of our module. e.g: resume

**Step 3:** ResumeForm in the Drupal\resume\Form namespace, which implements the FormInterface and is defined in the file: modules/resume/src/Form/ResumeForm.php

In modules/resume/src/Form/ResumeForm.php. First we declare the namespace, the other classes we want to use, and extend the FormBase class. Some code:



Next we use to get code from some other classes, using the use PHP keyword and then the namespace, using the PSR-4 standard, which will autoload the classes in the files that correspond to these namespaces



Now that we've our namespace, we'd like to use, we can declare our own class, ResumeForm extends the class FormBase, FormBase is the class we brought in as a dependency with this line: use Drupal\Core\Form\FormBase;

|  |  |
| --- | --- |
|  | <?php |
|  | /\*\* |
|  | \* @file |
|  | \* Contains \Drupal\resume\Form\ResumeForm. |
|  | \*/ |
|  | namespace Drupal\resume\Form; |
|  | use Drupal\Core\Form\FormBase; |
|  | use Drupal\Core\Form\FormStateInterface; |
|  | class ResumeForm extends FormBase { |
|  | /\*\* |
|  | \* {@inheritdoc} |
|  | \*/ |
|  | public function getFormId() { |
|  | return 'resume\_form'; |
|  | } |

[**view raw**](https://gist.github.com/xaiwant/ef354ce17333205c8dc4/raw/a88f091ae200ebeb662292e4d8734f767933a11f/ResumeGetformId.php)[**ResumeGetformId.php**](https://gist.github.com/xaiwant/ef354ce17333205c8dc4#file-resumegetformid-php) hosted with ❤ by **[GitHub](https://github.com/)**

**Form Creation:**Which returns a Form API array that defines each of the elements your form is composed of.

|  |  |
| --- | --- |
|  | <?php |
|  | /\*\* |
|  | \* {@inheritdoc} |
|  | \*/ |
|  | public function buildForm(array $form, FormStateInterface $form\_state) { |
|  | $form['candidate\_name'] = array( |
|  | '#type' => 'textfield', |
|  | '#title' => t('Candidate Name:'), |
|  | '#required' => TRUE, |
|  | ); |
|  | $form['candidate\_mail'] = array( |
|  | '#type' => 'email', |
|  | '#title' => t('Email ID:'), |
|  | '#required' => TRUE, |
|  | ); |
|  | $form['candidate\_number'] = array ( |
|  | '#type' => 'tel', |
|  | '#title' => t('Mobile no'), |
|  | ); |
|  | $form['candidate\_dob'] = array ( |
|  | '#type' => 'date', |
|  | '#title' => t('DOB'), |
|  | '#required' => TRUE, |
|  | ); |
|  | $form['candidate\_gender'] = array ( |
|  | '#type' => 'select', |
|  | '#title' => ('Gender'), |
|  | '#options' => array( |
|  | 'Female' => t('Female'), |
|  | 'male' => t('Male'), |
|  | ), |
|  | ); |
|  | $form['candidate\_confirmation'] = array ( |
|  | '#type' => 'radios', |
|  | '#title' => ('Are you above 18 years old?'), |
|  | '#options' => array( |
|  | 'Yes' =>t('Yes'), |
|  | 'No' =>t('No') |
|  | ), |
|  | ); |
|  | $form['candidate\_copy'] = array( |
|  | '#type' => 'checkbox', |
|  | '#title' => t('Send me a copy of the application.'), |
|  | ); |
|  | $form['actions']['#type'] = 'actions'; |
|  | $form['actions']['submit'] = array( |
|  | '#type' => 'submit', |
|  | '#value' => $this->t('Save'), |
|  | '#button\_type' => 'primary', |
|  | ); |
|  | return $form; |
|  | } |

**Validation:** After a user fills out the form and clicks the submit button it's common to want to perform some sort of validation on the data that's being collected. To do this with Drupal's Form API we simply implement the validateForm method from

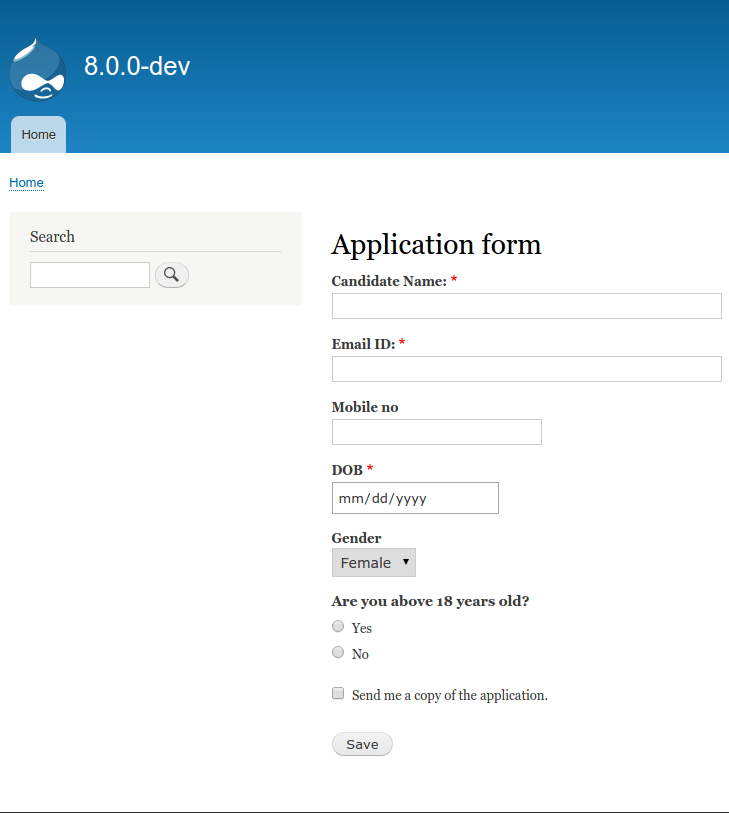
\Drupal\Core\Form\FormBuilderInterface in our ResumeForm class.

|  |  |
| --- | --- |
|  | <?php |
|  |  |
|  | /\*\* |
|  | \* {@inheritdoc} |
|  | \*/ |
|  | public function validateForm(array &$form, FormStateInterface $form\_state) { |
|  |  |
|  | if (strlen($form\_state->getValue('candidate\_number')) < 10) { |
|  | $form\_state->setErrorByName('candidate\_number', $this->t('Mobile number is too short.')); |
|  | } |
|  |  |
|  | } |
|  | } |

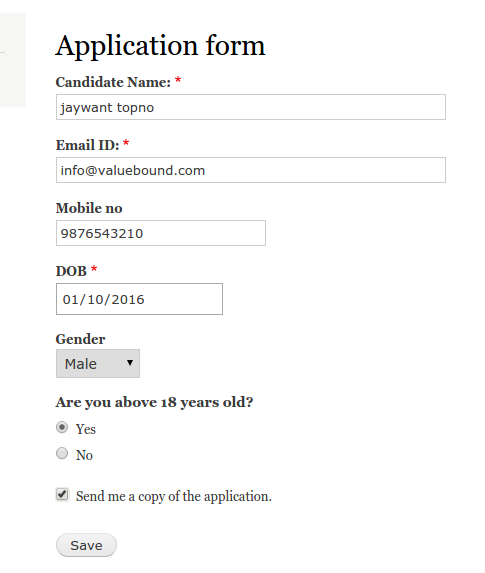
**Submission:** Finally, we build the submit method. According to the FormInterface, we use: public function submitForm(array &$form, FormStateInterface $form\_state).

|  |  |
| --- | --- |
|  | <?php |
|  |  |
|  | /\*\* |
|  | \* {@inheritdoc} |
|  | \*/ |
|  | public function submitForm(array &$form, FormStateInterface $form\_state) { |
|  | // drupal\_set\_message($this->t('@can\_name ,Your application is being submitted!', array('@can\_name' => $form\_state->getValue('candidate\_name')))); |
|  | foreach ($form\_state->getValues() as $key => $value) { |
|  | drupal\_set\_message($key . ': ' . $value); |
|  | } |
|  | } |

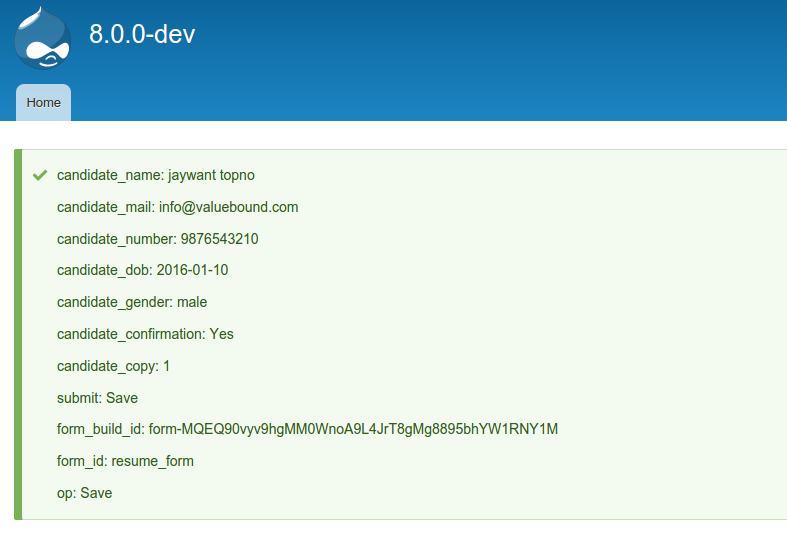
[**view raw**](https://gist.github.com/xaiwant/39bf6262c15b079a8468/raw/967594a929fdabfa0a8f674eac9c5541a366de67/ResumeForm_submit.php)[**ResumeForm\_submit.php**](https://gist.github.com/xaiwant/39bf6262c15b079a8468#file-resumeform_submit-php) hosted with ❤ by **[GitHub](https://github.com/)**

**Step 4:** we are done with our drupal 8 custom form. we can review the page by typing  
<http://d8/resume/myform>  
  


Fill up all the required field and and click on save button.



once you click on save button, you’ll get saved value info. as we are displaying this in the form.



As per our Submit form source code it has to display all keys and values

s30

**Conclusion:** Going through from Drupal 7 form module to Drupal 8, you learned about more than just the Drupal 8 Form API. You have learned how hook\_menu got replaced by routes and controllers, in YAML files and Controller.php files. you also learned how to use classes, and find the methods