

# Assignment Hooks and Theme

**Q1:- Write the filename where we use and write hooks.**

**Ans:-** Hooks are present in \*.api.php files, & functions whose name starts with "hook\_" contain the code of hooks.

Original hook function in "core/lib/Drupal/Core/Form/form.api.php":- hook\_batch\_alter()

Custom hook function in ".module":- my\_hook\_batch\_alter()

**Q2:-Write any 10 hooks name and their definition which is used by drupal 8.**

**Ans:-**

(1) **hook\_ajax\_render\_alter**:- Alter the Ajax command data that is sent to the client.

## **File core/lib/Drupal/Core/Form/form.api.php**

```
function hook_ajax_render_alter(array &$data) {  
  
    // Inject any new status messages into the content area.  
    $status_messages = array(  
        '#type' => 'status_messages',  
    );  
    $command = new \Drupal\Core\Ajax\PrependCommand('#block-system-main .content',  
        \Drupal::service('renderer')  
            ->renderRoot($status_messages));  
    $data[] = $command  
        ->render();  
}
```

(2) **hook\_block\_view\_alter**:- Alter the result of  
`\Drupal\Core\Block\BlockBase::build()`.

This hook is called after the content has been assembled in a structured array and may be used for doing processing which requires that the complete block content structure has been built.

## File `core/modules/block/block.api.php`

```
function hook_block_view_alter(array &$build, \Drupal\Core\Block\BlockPluginInterface $block)
{

    // Remove the contextual links on all blocks that provide them.

    if (isset($build['#contextual_links'])) {

        unset($build['#contextual_links']);

    }

}
```

**(3) hook\_block\_build\_alter:-** Unlike `hook_block_view_alter()`, this hook is called very early, before the block is being assembled. Therefore, it is early enough to alter the cacheability metadata (change `#cache`), or to explicitly placeholder the block (set `#create_placeholder`).

## File

## `core/modules/block/block.api.php`

**(4) hook\_cache\_flush:-** Flush all persistent and static caches. This hook asks your module to clear all of its static caches, in order to ensure a clean environment for subsequently invoked data rebuilds.

Do NOT use this hook for rebuilding information. Only use it to flush custom caches.

## File

### core/core.api.php

```
function hook_cache_flush() {  
    if (defined('MAINTENANCE_MODE') && MAINTENANCE_MODE == 'update') {  
        _update_cache_clear();  
    }  
}
```

(5) **hook\_ckeditor\_css\_alter:-** Modify the list of CSS files that will be added to a CKEditor instance. Modules may use this hook to provide their own custom CSS file without providing a CKEditor plugin. This list of CSS files is only used in the iframe versions of CKEditor

## File

### core/modules/ckeditor/ckeditor.api.php

```
function hook_ckeditor_css_alter(array &$css, Editor $editor) {  
    $css[] = drupal_get_path('module', 'mymodule') . '/css/mymodule-ckeditor.css';  
}
```

(6) **hook\_config\_import\_steps\_alter:-** Alter the configuration synchronization steps.

## File

### core/core.api.php

```
function hook_config_import_steps_alter(&$sync_steps, \Drupal\Core\Config\ConfigImporter  
$config_importer) {  
    $deletes = $config_importer  
        ->getUnprocessedConfiguration('delete');  
    if (isset($deletes['field.storage.node.body'])) {  
        $sync_steps[] = '_additional_configuration_step';  
    }  
}
```

**(7) hook\_config\_translation\_info:-** introduce dynamic translation tabs for translation of configuration.

This hook augments MODULE.config\_translation.yml as well as THEME.config\_translation.yml files to collect dynamic translation mapper information. If your information is static, just provide such a YAML file with your module containing the mapping.

## File

**core/modules/config\_translation/**[config\\_translation.api.php](#)

**(8) hook\_contextual\_links\_alter:-** Alter contextual links before they are rendered. This hook is invoked by [\Drupal\Core Menu\ContextualLinkManager::getContextualLinkPluginsByGroup\(\)](#). The system-determined contextual links are passed in by reference. Additional links may be added and existing links can be altered.

## File

**core/lib/Drupal/Core/Menu/**[menu.api.php](#)

```
function hook_config_translation_info(array &$links, $group, array $route_parameters) {  
  
    if ($group == 'menu') {  
  
        // Dynamically use the menu name for the title of the menu_edit contextual  
  
        // link.  
  
        $menu = \Drupal::entityManager()  
  
            ->getStorage('menu')  
  
            ->load($route_parameters['menu']);  
  
        $links['menu_edit']['title'] = t('Edit menu: @label', array(  

```

```

        '@label' => $menu

        ->label(),

    ));
}
}

```

**(9)hook\_extension:-** Declare a template file extension to be used with a theme engine.

This hook is used in a theme engine implementation in the format of ENGINE\_extension().

## File

**core/lib/Drupal/Core/Render/[theme.api.php](#)**

```

function hook_extension() {

    // Extension for template base names in Twig.

    return '.html.twig';
}

```

**(10) hook\_field\_info\_alter:-** Perform alterations on Field API field types.

## File

**core/modules/field/[field.api.php](#)**

```

function hook_field_info_alter(&$info) {

```

```
// Change the default widget for fields of type 'foo'.

if (isset($info['foo'])) {

    $info['foo']['default widget'] = 'mymodule_widget';

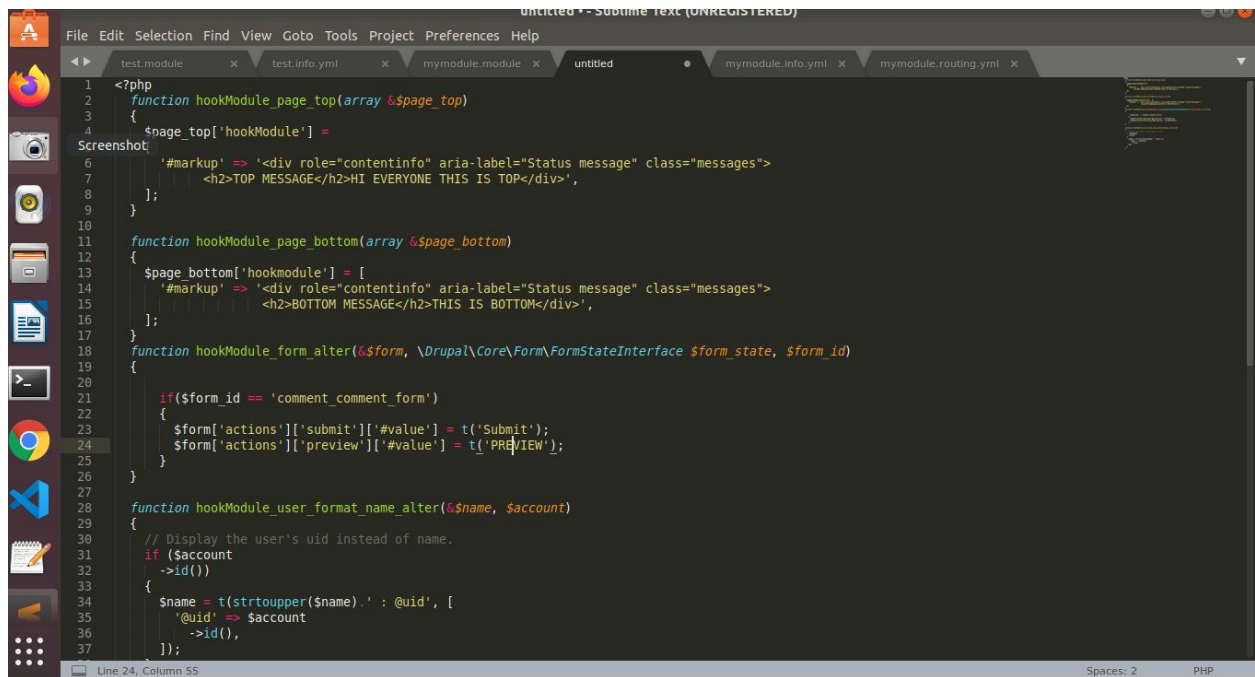
}

}
```

**Q3:-** Create a custom module and use any 3 hooks and their implementation.

**Ans:-** implementing of three hooks in the below.

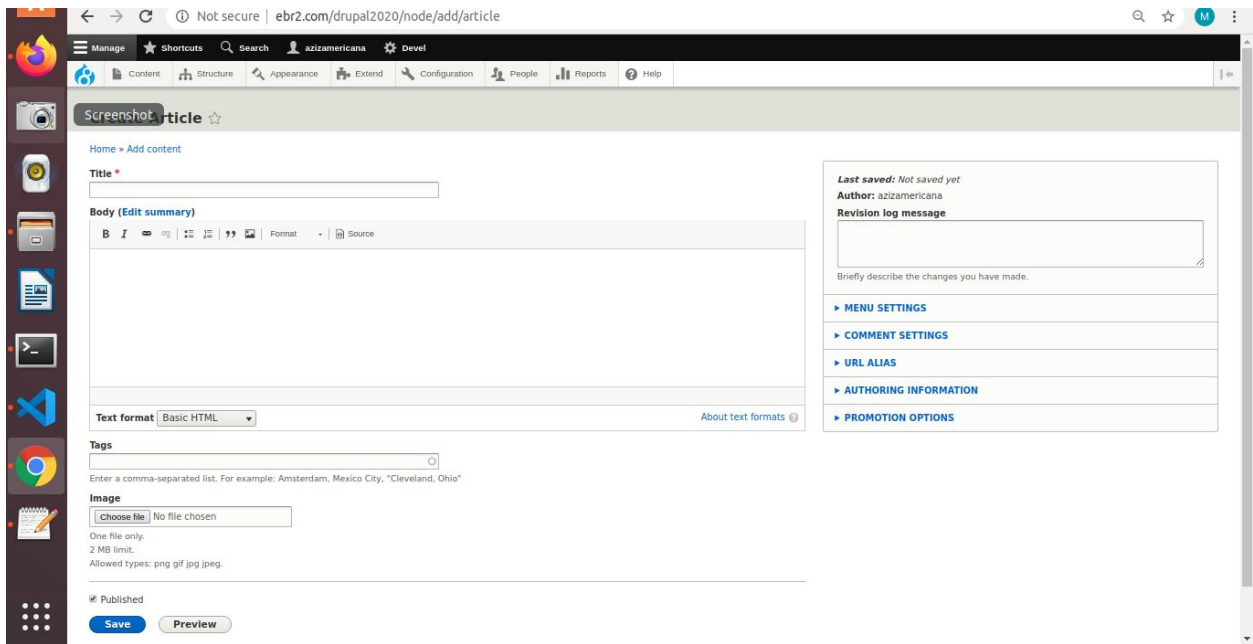
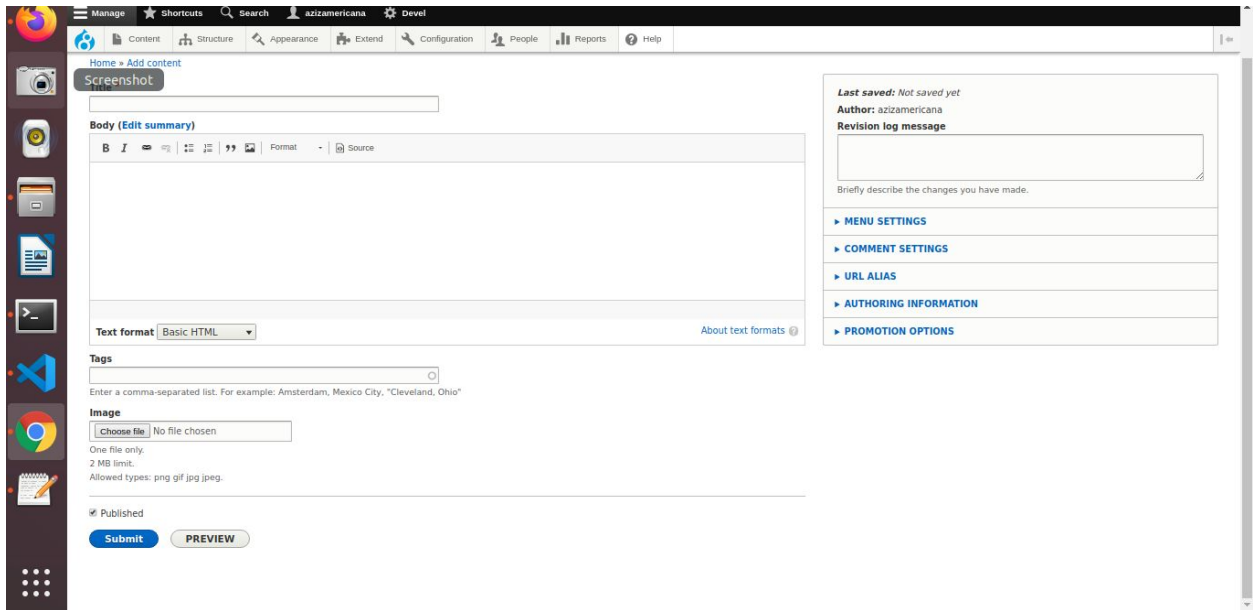
change save and preview button using hook\_form\_alter.



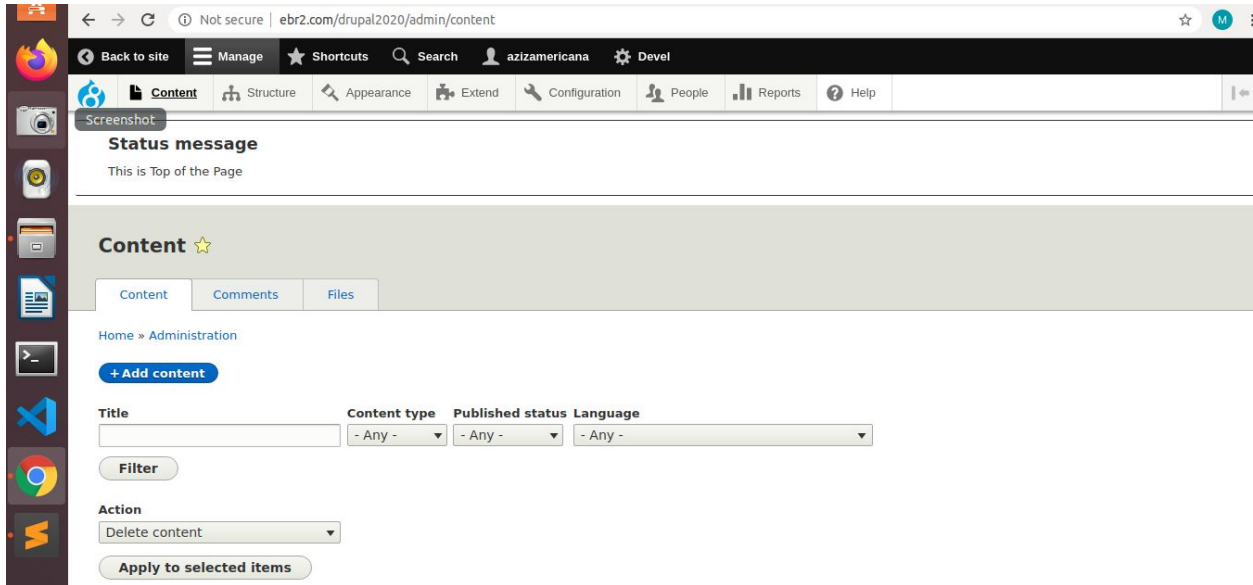
The screenshot shows a code editor with the following PHP code implementing three Drupal hooks:

```
1 <?php
2 function hookModule_page_top(array &$page_top)
3 {
4     $page_top['hookModule'] =
5     [
6         '#markup' => '<div role="contentinfo" aria-label="Status message" class="messages">
7             <h2>TOP MESSAGE</h2>HI EVERYONE THIS IS TOP</div>',
8     ];
9 }
10
11 function hookModule_page_bottom(array &$page_bottom)
12 {
13     $page_bottom['hookModule'] = [
14         '#markup' => '<div role="contentinfo" aria-label="Status message" class="messages">
15             <h2>BOTTOM MESSAGE</h2>THIS IS BOTTOM</div>',
16     ];
17 }
18 function hookModule_form_alter(&$form, \Drupal\Core\Form\FormStateInterface $form_state, $form_id)
19 {
20     if($form_id == 'comment_comment_form')
21     {
22         $form['actions']['submit']['#value'] = t('Submit');
23         $form['actions']['preview']['#value'] = t('PREVIEW');
24     }
25 }
26
27 function hookModule_user_format_name_alter(&$name, $account)
28 {
29     // Display the user's uid instead of name.
30     if ($account
31         =>id())
32     {
33         $name = t(strtoupper($name).': @uid', [
34             '@uid' => $account
35             =>id(),
36         ]);
37     }
38 }
```

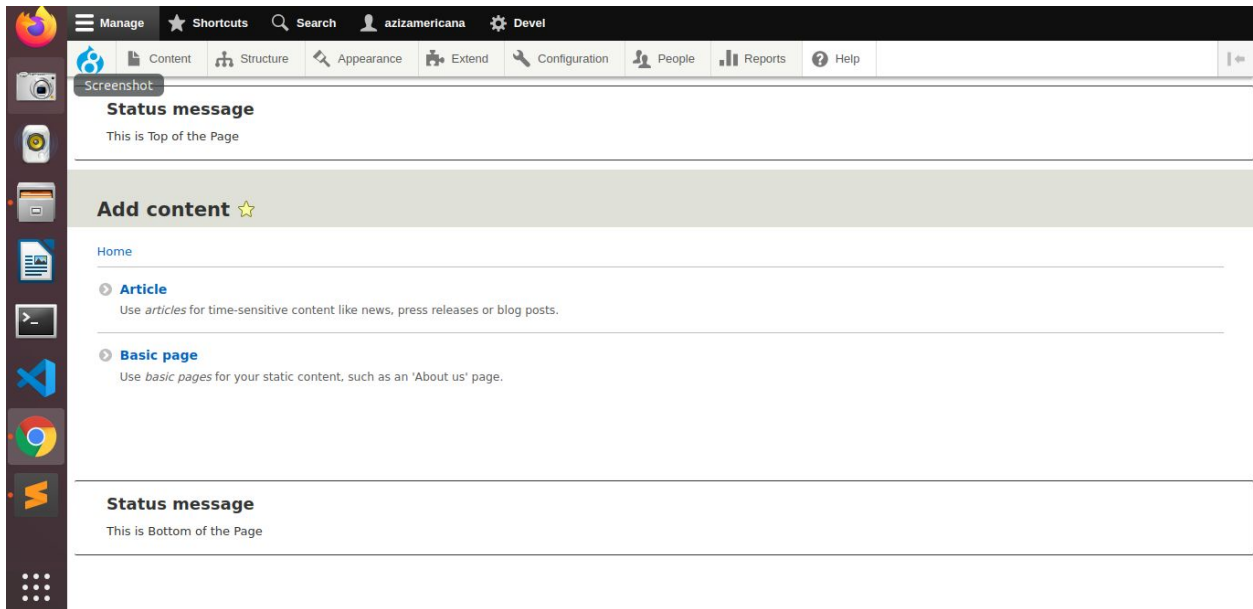
The editor interface includes a menu bar (File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, Help), a tab bar with open files (test.module, test.info.yml, mymodule.module, untitled, mymodule.info.yml, mymodule.routing.yml), and a status bar at the bottom showing 'Line 24, Column 55', 'Spaces: 2', and 'PHP'.



There is implementing hook\_page\_top



There is implementing hook\_page\_bottom hook.



Q4:- What is the use of .theme file?

Ans:- The .theme file is a PHP file that contains all the conditional logic and data (pre)processing of the output. It may also extend basic theme settings.



**Q5:- What is the theme() ?**

**Ans:-**The theme() function allows a theme to have nearly complete control over the appearance of the site, which includes both the markup and the CSS used to style the markup

**Q6:- Why do we use preprocess functions and how many types of preprocess functions in drupal 8?**

**Ans:-** Preprocess functions allow Drupal themes to manipulate the variables that are used in Twig template files by using PHP functions to preprocess data before it is exposed to each template. ... Understanding how preprocess functions work, and the role they play, is important for both module developers and theme developers.

Types of preprocess function in drupal 8 is given below.

**1.template\_preprocess().**This function creates a default set of variables for all theme hooks with template implementations provided by Drupal Core.

**2.tepmplate\_preprocess\_HOOK().**This function should be implemented by the module that registers the theme hook, to set up default variables.

**3.MODULE\_preprocess().**hook\_preprocess() is invoked on all implementing modules.

**4.MODULE\_preprocess\_HOOK().** hook\_preprocess\_HOOK() is invoked on all implementing modules, so that modules that didn't define the theme hook can alter the variables.

**5.ENGIN\_engine\_preprocess().** This function allows the theme engine to set necessary variables for all theme hooks with template implementations.

**6.ENGIN\_engin\_preprocess\_HOOK().**This function allows the theme engine to set necessary variables for all theme hooks with template implementations.

**7.THEME\_preprocess().** This function allows the theme to set necessary variables for all theme hooks with template implementations.

**8.THEME\_preprocess\_HOOK().** This function allows the theme to set necessary variables specific to the particular theme hook.

**Q7:- In which file do we write the preprocess functions?**

**Ans:-** to work with a preprocess function.

1.Create or edit a file in your theme suppose the directory name is “mytheme.theme” and create a function such as mytheme\_preprocess\_HOOK() where HOOK refers to the item you wish to affect which is present in “\*.theme” file.

**Q8:- What is template engine in drupal 8?**

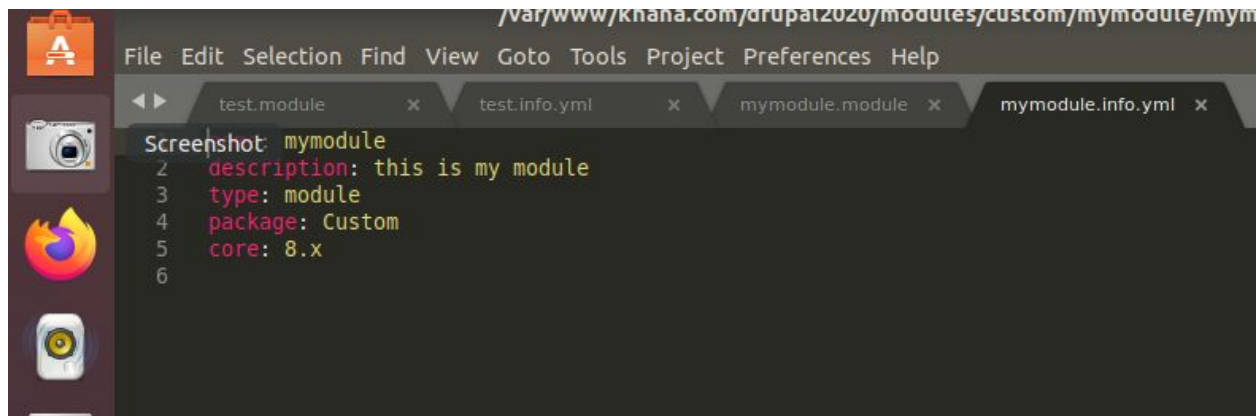
**Ans:-** A template engine is a software component which is used to combine data with templates from themes and gives the result in the form of html to the user. Template engine also known as template processor or template parser and theme engine.

Drupal 8 uses the templating engine Twig. Twig offers developers a fast, secure, and flexible method for building templates for Drupal 8 sites.

Twig also offers substantial usability improvements over PHPTemplate, and does not require front-end developers to know PHP to build and manipulate Drupal 8 themes.

**Q9:-Add one text field programmatically at the last in any one content type with your name and save your name and add a validation for alphabet**

**Ans:-**



```

Screenshot
2  /**
3   * Implementation of hook_form_alter
4   */
5  function mymodule_update_1234(&$form, \Drupal\Core\Form\FormStateInterface $form_state, $form_id)
6  {
7      \Drupal\field\Entity\FieldStorageConfig::create(array(
8          'field_name' => 'field_ebrahim',
9          'entity_type' => 'node',
10         'type' => 'text',
11         'cardinality' => -1,
12     ))->save();
13
14     \Drupal\field\Entity\FieldConfig::create([
15         'field_name' => 'field_ebrahim',
16         'entity_type' => 'node',
17         'label' => 'qwerty',
18     ])->save();
19
20     entity_get_form_display('node', 'hello', 'default')
21     ->setComponent('field_ebrahim', array(
22         'type' => 'text_textfield',
23     ))
24     ->save();
25
26
27 }
28
29 function mymodule_form_alter(&$form, \Drupal\Core\Form\FormStateInterface $form_state, $form_id)
30 {
31     if ($form_id == 'node_type_edit_form') {
32         $form['#validate'][] = 'qwerty';
33         // kint($form['#validate']);
34
35         // die;
36     }
37 }
38
39
40 function qwerty($form, \Drupal\Core\Form\FormStateInterface $form_state){
41     // kint($form_state->getValue('field_akash'));die;
42     $vall=$form_state->getValue('field_akash')[0]['value'];
43     //kint($vall);die;
44     if(!preg_match("/^[A-Z][a-zA-Z]$/", $vall)){
45         // kint($vall);die;
46         $form_state->setErrorByName('field_ebrahim', t('Name must start with capital letter'));
47     }
48     // die;
49 }

```

Text format

Basic HTML

About

QWERTY

ad2

Text format

Basic HTML

About

[Home](#) » [Add content](#)

 Name must start with capital letter

**Title \***