Error: User warning: The following module is missing from the file system:

```
drush sql-query "DELETE FROM key_value WHERE collection='system.schema' AND
name='MODULE_NAME';"
```

==========

Read a page's GET URL variables and return them as an associative array.

```
function getUrlVars()
{
  var vars = [], hash;
  var hashes = window.location.href.slice(window.location.href.indexOf('?') + 1).split('&');
  for (var i = 0; i < hashes.length; i++)
  {
    hash = hashes[i].split('=');
    vars.push(hash[0]);
    vars[hash[0]] = hash[1];
  }
  return vars;
}
if (getUrlVars()['field_date_value%5Bmin%5D']) {
    var selectedYear = getUrlVars()['field_date_value%5Bmin%5D'].split('-')[0];
    $("select#reports-filter").val(selectedYear);
  }
```

==========

Run JS only once

```
(function($, Drupal) {

  var initialized;

  function init() {
    if (!initialized) {
      initialized = true;
      // Add your one-time only code here
```

```
  }
 }

 Drupal.behaviors.someKey = {
  attach: function() {
   init();
  }
 };

}(jQuery, Drupal));
```

==========

No available releases found (In Modules or themes etc)

While checking updates Drupal creates some rows inside the key_value table which should be deleted after checking is complete but looks like they doesn't for some reason. So deleting the related rows manually solved my problem:

```
delete from key_value where collection='update_fetch_task'
```

==========

Retrieving block by id, name, uuid etc

```
{{ drupal_block('<plugin_id>', {label: 'Example'|t, some_setting: 'example'}) }}
```

```
{{ drupal_block('<plugin_id>', wrapper=false) }}
```

```
drush ev
"print_r(array_keys(\Drupal::service('plugin.manager.block')->getDefinitions()));"
```

```
{{ drupal_block('system_breadcrumb_block') }}
```

```
{{ drupal_entity('block', '<block_id>') }}
```

```
{{ drupal_entity('block', '<block_id>', check_access=false) }}
```

```
drush ev 'print_r(\Drupal::configFactory()->listAll("block.block."));'
```

```
{{ drupal_entity('block_content', '<content_block_id>') }}
```

```
drush sqlq 'SELECT id, info FROM block_content_field_data'
```

```
<div class="block">
  <h2>{{ 'Example'|t }}</h2>
  {{ drupal_entity('block_content', content_block_id) }}
</div>
```

```
{{ drupal_block('block_content:<uuid>', {label: 'Example'|t}) }}
```

You need to expose the fields from Node to Block variables.

```
function MYTHEME_preprocess_block(&$variables) {
    if(isset($variables['elements']['content']['#view'])){
        $node = $variables['elements']['content']['#view']->result[0]->_entity;

         if($node){
            if ($node instanceof \Drupal\node\Entity\Node) {

              if ($node->hasField('field_name_1')) {
                    $variables['field_name_1'] = $node->get('field_name_1')->value;
              }

              if ($node->hasField('field_name_2')) {
                   $variables['field_name_2'] = $node->get('field_name_2')->value;
              }
            }
        }
    }
}
```

To use in your block--views-block--block-custom.html.twig

```
<div>
     {{field_name_1}}
</div>

<div>
     {{field_name_2}}
</div>
```

==========

Basically there are two type of renders.

1. When there is an existing instance of the block in the layout. The the block can be rendered in twig using preprocess as
2.
```
$block = Block::load('mediablock');
  $variables['social_links'] = \Drupal::entityTypeManager()
    ->getViewBuilder('block')
    ->view($block);
```

3. There is no instance or configurations for the block. Then in the preprocessor we need to create the instance, build the block and then render it
4.
```
$block_manager = \Drupal::service('plugin.manager.block');
  $config = [];
  $plugin_block = $block_manager->createInstance('farmjournal_social_sharing', $config);
  $render = $plugin_block->build();
  $variables['farmjournal_social_sharing'] = render($render);
```

==========

Important Cammands

composer update --dry-run - to see what will be updated.
drush entity-updates - to see there are any updates for entity or not

Composer outdated 'drupal/*' - to check outdated modules
Composer update drupal/modulename - update the specific module
Composer require drupal/modulename - install the module
Composer remove drupal/modulename - remove the module

Drupal moi modulename - install the module
Drupal mou modulename - uninstall the module
$ drupal generate:plugin:block
$ drupal generate:form:config

==========

List updates

Use Composer's built-in command for listing packages that have updates available:

```
composer outdated 'drupal/*'
```

You can get the same information with Composer's `show` command.

List security updates

The security status from Drupal.org isn't available through Composer. Luckily Drush comes to the rescue:

```
drush pm:security
```

Install updates

For a given Drupal module/project use

```
composer update drupal/modulename --with-dependencies
```

Finally, run any database updates and rebuild the cache:

```
drush updatedb
drush cr
```

Checking Route name to run conditionals
==========
{% set url = url('<current>') %}

{% if '/contact-us' in url|render|render %}
ghdfghdf
{% else %}
uityityu
{% endif %}
==========

Managing users with Drush

Changing passwords for an existing user

```
drush upwd <username> —password=<new password>
```

Examples

Change password for test1 to letmein:

```
drush upwd test1 —password=letmein
```

Assign a role

```
drush urol <rolename> <username>
```

Examples

Add the role editor to username test1:

```
drush urol editor test1
```

Add the role editor to user with uid of 3:

```
drush urol editor 3
```

Add the editor role to users with ids 3 and 4:

```
drush urol editor --uid=3,4
```

Remove a role

```
drush unrol <rolename> <username>
```

Examples

Remove the role editor to username test1:

```
drush unrol editor test1
```

Add editor role to user with uid of 3:

```
drush unrol editor 3
```

Remove the editor role to users with ids 3 and 4

```
drush unrol editor --uid=3,4
```

Create users

```
drush ucrt <username>
```

Examples

Create a user with username test1:

```
drush ucrt test1
```

Add an email address and password for the user at the same time:

```
drush ucrt test1 --mail=test@example.com -- password=password1
```

Setting passwords for a user account

```
drush upwd <username> --password=<password>
```

Examples

Add letmein as a password to username test1:

```
$ drush upwd test1 --password=letmein
```

Display information about a user

```
drush uinf <username>
```

Examples

Display information for user with username test1:

```
drush uinf test1
```

Display information for user with uid of 3:

```
drush uinf 3
```

Display information about users with uids 3 and 4:

```
drush uinf 3,4
```

Display information for users with username test1 and test2:

```
drush uinf test1,test2
```

Display information about user with uid 2 and also user with name of test3:

```
drush uinf 2,test3
```

Block a user

```
drush ublk <username>
```

Examples

Block the user with username test1:

```
drush ublk test1
```

Block users with uids 3 and 4 and username test1:

```
drush ublk 3,4,test1
```

Unblock a user

```
drush uublk <username>
```

Examples

Unblock the user with username test1:

```
drush uublk test1
```

Unblock users with uids 3 and 4 and username test1:

```
drush unblk 3,4,test1
```

Installing a Project on local and synchronise with git

==========

If you have code on your local machine that is not under source control, you can prepare it by putting your code into a Git repository locally. From there, you push it to Bitbucket.

Add to a Git repository

1. From your terminal, change to the root directory of your existing code.
2. Initialize the directory under source control from the following command:
3. `$ git init`
4. Add the existing files to the repository you have initialized:
5. `$ git add .`
6. Commit the files:
7. `$ git commit -m "initial commit of full repository"`
8. On macOS, you can use single quotes or double quotes around the comment message. On Windows, you must use double quotes.

9. Connect your new local Git repository to the remote repository on Bitbucket. To do so, enter `git remote add origin` with the remote URL:
10. `$ git remote add origin <bitbucket_URL>`
11. You can find the URL next to the `git clone` command for the repository:

    Like << `git remote add origin https://wasim4spots@bitbucket.org/team4spots/bac014-web.git >>`

12. Push all the code in your local repo to Bitbucket with the following command:
13. `$ git push -u origin --all`