Pour les pages et bloc CMS

- Media URL (chemin vers les images uploadés dans le BO)
www.sonsite.com/pub/media/image.jpg :

{{media url='/image.jpg'}}

- Store URL (chemin vers les pages du sites) www.sonsite.com/page.html :

{{store url='page.html'}}

- View URL (chemin vers les images du thème déployé)
www.sonsite.com/pub/static/versionXYXX/frontend/Magento/luma/en_US//image.jpg :

{{view url='/image.jpg'}}

- Appelez un block :

{{block class="Magento\Framework\View\Element\Template"
template="[VendorName]_[ModuleName]::[YourTemplateFileLocation].phtml"}}

Pour les Phtml

Récupérer la fonction getLoadedProductCollection() du block;
$_productCollection=$this->getLoadedProductCollection();

Récupérer un helper :
$helper = $this->helper('{Vendor}\{Module}\Helper\Data');
$values = $helper->YourHelperMethod();

Récupérer un block enfant (block définit comme contenu dans le block actuelle via le layout de la page)
<?php echo $block->getChildHtml('name-of-block');?>

Appeler un block :
echo $this->getLayout()
->createBlock('Magento\Framework\View\Element\Template')
->setData('nom_de_ma_variable',$ma_variable)
->setTemplate('[VendorName]_[ModuleName]::[YourTemplateFileLocation].phtml')
->toHtml();

Pour récupérer dans l'enfant : $ma_variable = $this->ma_variable;

Url de base du site
<?php echo $this->getBaseUrl();?>

Url de d’une page
<?php echo $this->getUrl('contacts');?>

Url du dossier pub/media
<?php echo $this->getUrl('pub/media'); ?>

Url des fichiers images du thèmes
<?php echo $this->getViewFileUrl('images/test.png') ?>

Url des fichiers images d’un module
<?php echo $this->getViewFileUrl('Vendor_Module::images/test.png'); ?>

Dev

Affichage d’erreur :

passer en mode dev dans app/etc/env.php
'MAGE_MODE' => 'default',
et afficher les erreurs -> app/bootstrap.php
error_reporting(E_ALL);
ini_set('display_errors', 1);

L’object manager :

L’object manager permet de charger n’importe quelle classe sans avoir à compiler. Il sert à faire des tests rapide pour aller chercher un model (un entity id) ou une collection (all entity qui peuvent être filtrer et sort).

Ici on charge tous les produits et on affiche leur nom pour chacun.

```
$objectManager = \Magento\Framework\App\ObjectManager::getInstance();
$productCollection =
$objectManager->create('Magento\Catalog\Model\ResourceModel\Product\CollectionFactory');
$collection = $productCollection->create() ->addAttributeToSelect('*') ->load();
foreach ($collection as $product){ echo 'Name  =  '.$product->getName().' '; }
```

Les méthodes :

```
 ->addAttributeToSelect("entity_id") // si on veut load tous les attributs (*) ou seulement certains pour limiter la requête
->addAttributeToSort('entity_id', 'desc') // sort by entity
->setPageSize(10) // charger que les 10 premiers produits
->setCurPage(1); // on sélectionne la première page
```

Charger une entité ou une collection d’un model

Dans notre class :

```
//définir sa variable
        protected $_postFactory;
        public function __construct(
                ...
                \Vendor\Module\Model\NameFactory $postFactory,
                ...
        )
        {
                ….
                $this->_postFactory = $postFactory;
                parent::__construct(...);
        }

//appelle dans une fonction du model
$post = $this->_postFactory->create();
//si on veut une entité précise par un type d’attribut:
$post->load(1,'posts_id');

//Ensuite on peut afficher toutes ses données via
var_dump($post->getData());

//Si on veut juste une donnée précise on fait
$post->getData(‘ma_value’);

//si on veut toute la collection :
$post = $this->_postFactory->create();
$postCollection = $post->getCollection();

Puis on parcours :
   <?php
      foreach ($block->getPostCollection() as $post){
//$post->getName() == $post->getData(‘name’);
            echo  $post->getName();
      }
   ?>
```

Les commandes CLI :

All list : https://www.mageplaza.com/devdocs/magento-2-command-line-interface-cli.html

Mettre à jour Magento 2 mais juste les dépendances (cela évite que tout se mette à jour et que des conflits apparaisent sur un projet qui comporte de nombreuses dépendances)

composer install

Mettre à jour Magento 2

composer update

Installer un module par composer (après faire un setup:upgrade/compile et un static:deploy)

composer require vendor/namepackage;branch php

Activer/désactiver et désinstaller un module :

php bin/magento module:enable <Vendor>_<Module>

php bin/magento module:disable <Vendor>_<Module>

php bin/magento module:uninstall -r <Vendor>_<Module>

composer remove VendorName/VendorExtensionRepository

Créer un user admin pour le back-end de magento :
php bin/magento admin:user:create --admin-user='thibault' --admin-password='coucou1' --admin-email='thibault.lhotellier@gmail.com' --admin-firstname='titi' --admin-lastname='titi'

Vider le cache :
php bin/magento cache:clean ou php bin/magento cache:flush

Redéployer son thème :
rm -rf pub/static/*;rm -rf var/view_preprocessed/*;

php -dmemory_limit=5G bin/magento setup:static-content:deploy -f fr_FR en_US --theme Magento/backend --theme Startrocket/formation

Réindexer :
php bin/magento indexer:reindex

Lancer le cron en manuel
php bin/magento cron:run

Activer ou Mettre à jour son module (il faut redéployer son thème après cette commande)
php bin/magento setup:upgrade

Compiler son code
php -dmemory_limit=5G bin/magento setup:di:compile

Afficher en front le path des templates (ajouter l'option "add block names to hints" dans l'admin pour voir sa classe associée) :
php bin/magento dev:template-hints:enable