

Affichage du texte dans le Front office

Dans cette section vous allez apprendre à afficher du texte défilant dans la page d'accueil de la boutique en utilisant un hook.

- 1- Dans le fichier **myModule.php**, modifiez la méthode `install()` pour inscrire le hook que vous allez utiliser. Utilisez le hook `displayHome` pour afficher le texte sur la page d'accueil :

```
public function install() {  
    return parent::install() && $this->registerHook('displayHome') ;  
    // && $this->registerHook('top') && $this->registerHook('leftColumn');  
}
```

- 2- Ajoutez une méthode **hookDisplayHome()** dans le fichier `myModule.php`. Cette méthode doit récupérer la valeur de `COURSE_RATING` enregistré dans la base de données et l'afficher dans une balise `<marquee>`.

```
public function hookDisplayHome($params)  
{  
    $this->context->controller->addCSS($this->_path . 'views/css/custom_flag.css', 'all');  
    $message = Configuration::get('COURSE_RATING');  
  
    return '<div class="scrolling-text "><span><b>' . $message . '</b></span></div>';  
}
```

public function hookDisplayHome(\$params) :

- Cette méthode est appelée lorsque le hook `displayHome` est déclenché.
- Dans PrestaShop, le hook `displayHome` est utilisé pour afficher des éléments dans la page d'accueil.
- **Paramètre `$params` :** Il contient les données passées par le hook. Dans ce cas, il n'est pas utilisé explicitement.

`$this->context->controller->addCSS($this->_path . 'views/css/custom_flag.css', 'all');`

- `$this->context->controller` : Représente le contrôleur actuellement chargé (par exemple, le contrôleur de la page d'accueil).
- `addCSS()` : Méthode utilisée pour inclure un fichier CSS spécifique dans la page HTML. Ici, elle ajoute le fichier CSS situé dans le chemin `views/css/custom_flag.css` de votre module.
- `$this->_path` : Représente le chemin du module actuel. Par exemple, si le nom du module est `mymodule`, cela pointe vers `/modules/mymodule/`.

- 'all' :Indique que ce fichier CSS est destiné à tous les supports (desktop, mobile, etc.).
- Effet : Le fichier CSS custom_flag.css sera chargé sur la page où ce hook est utilisé, permettant de styliser l'élément affiché.

return '<div class="scrolling-text">' . \$message . '</div>;'

- `<div class="scrolling-text">` : Une balise `<div>` avec la classe `scrolling-text`. Cette classe sera stylisée via un fichier CSS.

3- Maintenant, ajoutez un fichier CSS dans votre module pour styliser le texte. Par exemple, créez un fichier `views/css/costum_flag.css` et ajoutez le code suivant :

```

1
2  .scrolling-text {
3      width: 100%;
4      overflow: hidden;
5      white-space: nowrap;
6      box-sizing: border-box;
7      background-color: #1e4fd4;
8      padding: 10px;
9      font-size: 18px;
10     color: #ece5e5;
11     text-align: center;
12 }
13
14 .scrolling-text span {
15     display: inline-block;
16     padding-left: 100%; /* Espace pour démarrer en dehors de l'écran */
17     animation: scroll-left 10s linear infinite; /* Animation de 15 secondes */
18 }
19
20 @keyframes scroll-left {
21     0% {
22         transform: translateX(100%); /* Commence en dehors de l'écran */
23     }
24     100% {
25         transform: translateX(-100%); /* Sort par la gauche */
26     }
27 }
```