A. Les marqueurs et les fonctions

Pour afficher les contenus de votre site WordPress, les templates utilisent des marqueurs de modèle, des *template tags* en anglais. Les marqueurs sont des fonctions PHP spécialement dédiées à WordPress. Elles vont chercher des informations dans la base de données pour permettre l'affichage de leurs valeurs dans les templates.

Par exemple, le marqueur `bloginfo('url')` permet d'afficher l'URL de votre site. Le code pourrait être celui-ci : `<p><?php bloginfo('url'); ?></p>`. Le template va afficher dans un paragraphe `<p>`, l'URL de votre site.

Les fonctions de référence de WordPress (*functions reference* en anglais) sont très similaires aux marqueurs. Contrairement aux marqueurs, les fonctions n'affichent rien, elles renvoient des valeurs que vous pouvez stocker dans des variables PHP. Ensuite, vous pouvez effectuer un traitement sur ces variables.

Par exemple, la fonction `get_bloginfo('template_url')` va renvoyer le chemin d'accès au thème actif. Cet exemple va permettre de stocker le chemin d'accès du thème actif dans une variable nommée `$chemin_theme`. Ensuite, nous allons afficher la valeur de cette variable dans un paragraphe `<p>` à l'aide de la fonction PHP `echo()` :

```
<?php
  $chemin_theme = get_bloginfo('template_url');
  echo('<p>Le chemin d\'accès au thème est :
'.$chemin_theme.'</p>');
?>
```

Les marqueurs sont principalement destinés à la création des thèmes et les fonctions sont principalement destinées à la création de plug-ins et d'applications personnalisées. Et vous trouverez bien sûr des marqueurs et des fonctions dans les thèmes ! Enfin, notez que les marqueurs sont des fonctions de référence.

Vous trouverez toute la documentation technique sur les marqueurs et les fonctions à cette URL : https://codex.wordpress.org.

Dans ce chapitre, nous n'allons bien sûr pas étudier tous les marqueurs, il faudrait plusieurs ouvrages pour le faire, mais les principaux. Ces marqueurs vous permettront d'afficher les principaux contenus de vos sites web WordPress.

B. La boucle WordPress

Dans les thèmes, vous trouverez dans de très nombreux fichiers la "fameuse" boucle WordPress (*loop* en anglais) qui est le cœur de l'affichage des sites WordPress. C'est par cette boucle `while()` que sont affichés les contenus rédactionnels dans toutes les pages de vos sites WordPress.

La boucle va interroger la base de données pour savoir s'il y a des données à afficher. Si la réponse est oui, la boucle va afficher les données que nous lui demandons d'afficher.

Voici la boucle standard simplifiée (sans les commentaires et sans les arguments de la fonction `the_posts_pagination()`) du thème par défaut de WordPress 4.7, le thème **Twenty Seventeen** :

```
<?php
  if ( have_posts() ) :
       while ( have_posts() ) : the_post();
       get_template_part( 'template-parts/post/content',
get_post_format() );
       endwhile;
       the_posts_pagination(...);
  else :
       get_template_part( 'template-parts/post/content', 'none' );
  endif;
?>
```

La première étape consiste à tester avec la fonction PHP `if()`, s'il y a du contenu disponible avec la fonction WordPress `have_posts()`. Voici l'URL de référence de cette fonction : https://developer.wordpress.org/reference/functions/have_posts/

S'il y a du contenu à afficher, la boucle PHP `while()` est appelée. Tant qu'il y a du contenu à afficher, `have_posts()`, la boucle utilise la fonction WordPress `the_post()` qui permet une itération sur le contenu. Voici l'URL de référence de cette fonction : https://developer.wordpress.org/reference/functions/the_post/

Ensuite, la boucle va chercher le bon template à utiliser pour afficher les informations. Pour utiliser le bon template, c'est la fonction WordPress `get_template_part()` qui est utilisée. Voici l'URL de référence de cette fonction : https://developer.wordpress.org/reference/functions/get_template_part/. Le premier argument de cette fonction est le chemin d'accès aux templates : `template-parts/post/content`. Le deuxième argument détermine s'il faut utiliser un format d'article spécifié, avec la fonction WordPress `get_post_format()`. Voici l'URL de référence de cette fonction : https://developer.wordpress.org/reference/functions/get_post_format/

Puis la boucle s'arrête : `endwhile;`.

Ensuite, WordPress affiche, si c'est nécessaire, la pagination des articles dans la page, avec la fonction `the_posts_pagination()`. Voici l'URL de référence de cette fonction : https://developer.wordpress.org/reference/functions/the_posts_pagination/

Ceci termine la partie du code qui affiche le contenu, s'il y a du contenu à afficher. C'est la première partie de la fonction `if( have_posts() )`.

Ensuite, nous avons `else` : qui indique ce qu'il faut faire s'il n'y a pas de contenu à afficher. Dans ce cas, WordPress utilise un template spécifique, nommé **content-none.php** pour indiquer qu'il n'y a pas de contenu à afficher. C'est à nouveau la fonction WordPress `get_template_part()` qui est utilisée.

Puis pour terminer, nous avons la fin du test avec `endif;`.

C. Créer de nouvelles boucles

1. Les objectifs

Nous venons de le voir, la boucle standard de WordPress permet d'afficher les contenus dans toutes les pages de votre site. Dans la page d'accueil, la boucle permet d'afficher la liste des X derniers articles créés, avec une configuration standard de type blog (menu **Réglages - Lecture** - options **La page d'accueil affiche** et **Les pages du site doivent afficher au plus**).

Mais pour personnaliser l'affichage en page d'accueil, vous pouvez souhaiter afficher en plus d'autres contenus selon d'autres critères. Par exemple, vous souhaiteriez afficher les X derniers articles d'une catégorie donnée, ou d'un auteur précis, ou d'un mois spécifié. Vous pouvez aussi vouloir afficher les articles d'un type de contenu personnalisé (*Custom Post Type*).

Si tel est votre objectif, vous devez alors créer une deuxième boucle WordPress à l'aide de la fonction `WP_Query()`, dont voici l'URL de référence sur le Codex de WordPress : https://codex.wordpress.org/Class_Reference/WP_Query.

2. Créer une boucle sur une catégorie

Pour ce premier exemple, nous allons créer une boucle sur une catégorie donnée. La première étape consiste à connaître l'identifiant de la catégorie voulue. Pour ce faire vous pouvez utiliser un plug-in comme **Catch IDs** (https://wordpress.org/plugins/catch-ids/) ou bien repérer cet identifiant au survol de ladite catégorie, dans la barre d'état de l'administration de votre site. Dans cet exemple, l'identifiant de la catégorie ciblée est `2`.

ID	Nom ▾	Description	Identifiant	Total
2	Voyage	Les articles de la catégorie Voyage	voyage	2

Nous allons afficher le résultat de cette nouvelle boucle dans la page d'accueil. C'est donc dans le fichier **index.php** que nous allons insérer ce code, sous la boucle principale.

La première étape consiste à définir les arguments de la nouvelle requête, avec les critères de sélection. Dans cet exemple, les critères seront d'afficher les contenus de type **Article** de la catégorie **Voyage** dont l'identifiant est **2** et d'afficher les **10** derniers. Voici la déclaration de cette requête :

```
// Définition des arguments de la requête
$args_voyage=array(
  'post_type' => 'post',
  'cat' => 2,
  'posts_per_page' => 10,
);
```

Détaillons cette déclaration :

- `$args_voyage` : nous créons une variable pour stocker les arguments.
- `array(...)` : cette variable stocke les arguments dans un tableau.
- `'post_type' => 'post'` : le type de contenu recherché est **Article**.
- `'cat' => 2` : l'identifiant de la catégorie d'article recherché est **2**.
- `'posts_per_page' => 10` : nous recherchons les **10** derniers articles publiés.

Ensuite, dans la deuxième étape, nous allons définir une nouvelle requête :

```
// Définition de la nouvelle requête
$query_voyage = new WP_Query($args_voyage);
```

- `$query_voyage` : nous créons une variable pour stocker cette nouvelle requête.
- `new WP_Query($args_voyage)` : nous créons une nouvelle requête, `new`, avec la fonction `WP_Query()`, en lui passant les arguments avec la variable `$args_voyage`, définie précédemment.

La troisième étape consiste à créer une boucle WordPress avec cette nouvelle requête :

```
//Exécution de la boucle avec la nouvelle requête
if($query_voyage->have_posts()) : while ($query_voyage->
have_posts() ) : $query_voyage->the_post();
  the_title('<h4><a href="'.get_permalink().'">','</a></h4>');
  endwhile;
endif;
```

Il n'y a aucune difficulté dans cette boucle. C'est une boucle WordPress classique qui utilise la nouvelle requête `$query_voyage`. Nous affichons le titre des articles trouvés, `the_title()`, avec leur lien, `get_permalink()`, pour les afficher en page seule.

Pour terminer, la dernière étape permet de réinitialiser la requête principale afin de remettre à zéro le compteur de contenu `post` :

```
// Réinitialisation de la requête principale
wp_reset_postdata();
```

Voici l'URL de référence de cette fonction : https://developer.wordpress.org/reference/functions/wp_reset_postdata/

Voici le code complet de cette nouvelle requête :

```
// Définition des arguments de la requête
$args_voyage=array(
  'post_type' => 'post',
  'cat' => 2,
  'posts_per_page' => 10,
);
// Définition de la nouvelle requête
$query_voyage = new WP_Query($args_voyage);
```

```
//Exécution de la boucle avec la nouvelle requête
if($query_voyage->have_posts()) : while ($query_voyage->
have_posts() ) : $query_voyage->the_post();
  the_title('<h4><a href="'.get_permalink().'">','</a></h4>');
  endwhile;
endif;
// Réinitialisation de la requête principale
wp_reset_postdata();
```

3. Créer une boucle sur un mois

Dans cet exemple, nous allons afficher les 10 derniers articles du mois d'août.

Voici la déclaration du tableau d'arguments :

```
$args_aout=array(
  'post_type' => 'post',
  'monthnum' => 8,
  'posts_per_page' => 10,
);
```

Nous utilisons le paramètre `monthnum` pour définir le numéro du mois d'août : **8**.

Ensuite nous définissons la nouvelle requête :

```
$query_aout = new WP_Query($args_aout);
```

Et vient ensuite la nouvelle boucle :

```
if($query_aout->have_posts()) : while ($query_aout->
have_posts() ) : $query_aout->the_post();
  the_title('<h4><a href="'.get_permalink().'">','</a></h4>');
   the_date();
endwhile;
endif;
```

Dans cette boucle, nous affichons le titre des articles avec leur lien et la date de publication, `the_date()`.

Nous terminons par la réinitialisation :

```
wp_reset_postdata();
```

Voici le code complet de cet exemple :

```
// Définition des arguments de la requête
$args_aout=array(
  'post_type' => 'post',
  'monthnum' => 8,
  'posts_per_page' => 10,
);
```

```
// Définition de la requête
$query_aout = new WP_Query($args_aout);
//Exécution de la boucle avec la nouvelle requête
if($query_aout->have_posts()) : while ($query_aout->
have_posts() ) : $query_aout->the_post();
  the_title('<h4><a href="'.get_permalink().'">','</a></h4>');
  the_date();
endwhile;
endif;
// Réinitialisation de la requête principale
wp_reset_postdata();
```

4. Créer une boucle sur un auteur

Vous pouvez aussi afficher tous les articles d'un auteur donné. Comme pour les catégories, il faut connaître l'identifiant de l'auteur concerné. Dans cet exemple, l'identifiant est `2`.

Voici immédiatement le code complet de cet exemple qui utilise le paramètre `'author' => 2` pour identifier l'auteur :

```
// Définition des arguments de la requête
$args_christine=array(
  'post_type' => 'post',
  'author' => 2,
  'posts_per_page' => 10,
);
// Définition de la requête
$query_christine = new WP_Query($args_christine);
// Exécution de la boucle avec la nouvelle requête
if($query_christine->have_posts()) : while ($query_christine->
have_posts() ) : $query_christine->the_post();
  the_title('<h4><a href="'.get_permalink().'">','</a></h4>');
endwhile;
endif;
// Réinitialisation de la requête principale
wp_reset_postdata();
// Affichage de l'URL de sa page d'auteur
$url_auteur = get_author_posts_url(2);
echo ('<p>URL de la page de l\'auteur : <a href="'.$url_auteur.'">
'.get_the_author_meta('first_name',2).'</a>.</p>');
```

5. Créer une boucle multicritère

Dans la déclaration du tableau des arguments de la requête, vous pouvez indiquer tous les critères de recherche que vous voulez. Vous pouvez spécifier des critères sur les auteurs, les catégories, les étiquettes, les types de contenu (article ou page), les droits des utilisateurs, les dates de publication, les statuts des publication... Reportez-vous à la page du Codex pour visualiser tous ces critères : https://codex.wordpress.org/Class_Reference/WP_Query

Voici un exemple d'un tableau d'arguments portant sur le type de contenu, la catégorie, l'auteur et la date de publication :

```
$args = array(
  'post_type' => 'post',
  'cat' => 6,
  'author' => 2,
  'monthnum' => 5,
  'posts_per_page' => 10,
);
```

6. Créer une boucle sur un type de contenu

Vous le savez, WordPress ne propose de manière standard que deux types de contenu : les articles et les pages. À l'aide d'un plug-in ou directement en mode code dans le fichier **functions.php**, vous pouvez créer d'autres types de contenu (**Custom Post Type**) et leur taxinomie associée.

Dans cet exemple, le nouveau type de contenu est nommé `livre`.

Voici la déclaration des arguments :

```
$args_livre=array(
  'post_type' => 'livre',
  'posts_per_page' => 5,
);
```

C'est avec le paramètre `post_type` que nous spécifions quel est le type de contenu sur lequel nous recherchons des contenus. Dans cet exemple, c'est **livre**.

D. Les inclusions des fichiers et des templates

1. Insérer les fichiers de structure

Nous l'avons déjà évoqué, les thèmes WordPress sont constitués de plusieurs fichiers de structure qu'il faut associer afin de constituer l'affichage complet des pages.

De manière usuelle nous avons quatre fichiers de structure :

- la page principale : **index.php** qui est le fichier maître, celui qui assemble l'ensemble des autres fichiers, des autres templates.