Documentation theme.json

Table des matières

Le theme.json, c’est quoi ?

Le fichier "theme.json" est au cœur d’un thème Full Site Editing (FSE). Il joue un rôle central dans la configuration et le contrôle du design du site WordPress. Ce fichier, au format JSON, permet de définir une grande partie de l’apparence et du comportement du thème sans avoir à toucher au code PHP ou CSS directement.

Grâce au "theme.json", on peut paramétrer les couleurs, les typographies, les espacements, les mises en page, et bien plus, d’une manière standardisée et native. C’est lui qui établit les règles globales du style, ainsi que les options disponibles dans l’éditeur de blocs pour les utilisateurs.

Ce fichier est un outil puissant qui simplifie la gestion du design d’un site WordPress en FSE, en rassemblant toutes les configurations dans un seul et même fichier clair et structuré.

Structure

Le fichier "theme.json" est organisé en plusieurs sections principales qui permettent de contrôler différents aspects du thème. Cette organisation claire facilite la personnalisation et la gestion du style global du site.

Les grandes parties qui composent "theme.json" sont principalement :

- Version : indique la version de la structure du fichier utilisée. Cela permet à WordPress de savoir comment interpréter les paramètres.
- Settings : cette section contient toutes les options de configuration du thème, comme les couleurs, la typographie, les espacements, ou encore les réglages spécifiques à certains blocs.
- Styles : ici sont définies les règles visuelles globales qui s'appliquent aux éléments du site, comme les couleurs par défaut, les polices, les marges, etc.

Chacune de ces sections se décompose elle-même en sous-parties pour affiner le contrôle, ce qui rend "theme.json" à la fois puissant et flexible.

Cette structure hiérarchique est la clé pour offrir une expérience utilisateur fluide et cohérente dans l'éditeur de blocs, tout en assurant un rendu harmonieux du site final.

version

Premièrement, pour configurer le fichier, il faut définir la version. En ce moment, la dernière version du theme.json est la 3. C'est indispensable pour que WordPress puisse interpréter les paramètres du fichier theme.json. Cette déclaration est souvent accompagnée de la déclaration « $schema », qui elle permet d'aider pendant la création du fichier avec de l'auto-complétions.

settings

C'est ici que sont définies toutes les variables CSS utiles à la création d'un site. Pour activer ou désactiver un paramètre, il faudra entrer la valeur « true » ou la valeur « false » en fonction du résultat voulu. Pour les autres, il faudra créer ces variables avec un « name », un « slug » et une valeur en fonction de la variable (code HEX pour les couleurs, px/rem pour les espacements, ...) Voici les différentes propriétés sur lesquelles il est possible d'agir :

useRootPaddingAwareAlignements

Active ou désactive l'ajustement automatique des alignements en tenant compte du padding racine. Lorsqu'elle est activée (ce qui est le cas par défaut), certains blocs comme les groupes appliquent automatiquement un padding basé sur la structure globale du site. Cela peut parfois gêner la mise en page, c'est pourquoi il est courant de la désactiver.

appearanceTools

Active plusieurs outils visuels dans l'éditeur qui sont désactivés par défaut. En activant cette option, on débloque des contrôles supplémentaires pour gérer les arrière-plans, bordures, couleurs de lien, aspect ratio, position sticky, marges, paddings, et plus encore. Cela donne à l'utilisateur plus de liberté, mais nécessite parfois un peu plus de vigilance pour garder une cohérence visuelle.

color

Gère tout ce qui concerne les couleurs dans l'éditeur. Il est possible d'y activer ou désactiver :

- La palette personnalisée ou par défaut
- Les dégradés et duotones
- Les couleurs pour le texte, les arrière-plans, les liens, les titres, les boutons, etc.

On peut aussi y définir sa propre palette de couleurs.

layout

Permet de définir les largeurs maximales du contenu du site.

- contentSize : largeur standard du contenu
- wideSize : largeur étendue (utilisée notamment dans les blocs pleine largeur)

Cela aide à contrôler l'aspect général et le comportement responsive.

shadow

Gère les ombres globales du thème.

- defaultPresets : active les ombres par défaut de WordPress
- presets : permet de créer des ombres personnalisées à l'aide de slugs (en kebab-case).

Peut être utile pour donner du relief à certains blocs ou éléments.

spacing

Contrôle tout ce qui est lié aux marges, paddings et espacements.

- spacingSizes : permet de créer ses propres tailles d'espacement
- customSpacingSize et defaultSpacingSize : activent ou désactivent les tailles personnalisées ou par défaut

Cela permet de garder un système d'espacement cohérent sur tout le site.

typography

Regroupe tous les réglages typographiques.

- fontFamilies : permet de déclarer des polices (locales ou Google Fonts)
- fontSizes : définit des tailles de texte réutilisables
- fluid : active des tailles de texte adaptatives (responsive)
- customFontSizes et defaultFontSizes : activent ou désactivent les tailles par défaut ou personnalisées

On peut aussi ajuster l'interlignage, la graisse, la casse, etc.

Avec le thème sont fournis deux fichier theme.json, un prérempli et un template. Vous pourrez donc retrouver toutes les façons de coder ces paramètres.

Documentation theme.json

styles

Maintenant que les variables sont définies, il est possible de les utiliser pour ajouter du style aux éléments d'un site dans la section « styles ». Pour récupérer une variable voici l'écriture à employer : var(--wp--preset--nom-de-la-propriété--slug). Par exemple pour récupérer la taille de texte « Title_1 » il faut écrire : var(--wp--preset--font-size--title-1).

color

Définit les couleurs appliquées globalement au site.

- background : couleur de fond
- text : couleur principale des textes

On utilise ici les variables définies dans la section « colors » de la partie « settings ».

typography

Définit les styles typographiques globaux.

- fontFamily, fontSize, fontWeight, etc.

Ces styles sont appliqués par défaut, sauf si un bloc ou un élément en définit d'autres. On utilise ici les variables définies dans la section « typography » de la partie « settings ».

elements

Permet de styliser des éléments HTML spécifiques comme :

- Les titres (h1 à h6)
- Les boutons (button)

Chaque élément peut être personnalisé avec ses couleurs, tailles de police, marges, paddings, bordures, etc. Très utile pour garder un design cohérent sans devoir passer par du CSS. On utilise ici toutes les variables utiles définies la partie « settings ».

blocks

Permet de styliser des blocs Gutenberg précis (ex: core/paragraph, core/navigation, etc.).

On peut définir pour chaque bloc des styles personnalisés qui s'appliqueront partout où ce bloc est utilisé. Très utile pour uniformiser des composants récurrents comme des boutons, des en-têtes ou des galeries. On utilise ici toutes les variables utiles définies la partie « settings ».

Puissant mais encore limité

Malgré sa puissance, le fichier theme.json ne permet pas encore de tout gérer. Pour les animations (comme hover, focus, active), les transitions ou certains comportements responsives avancés, un complément en CSS reste nécessaire.

L'ajout de classes personnalisées sur les blocs permet de cibler précisément les éléments à styliser. Il est recommandé de suivre une logique mobile-first pour le responsive : cela facilite l'adaptation aux écrans plus larges et permet un CSS plus léger.

Des outils comme Tailwind CSS peuvent être utiles pour structurer ce CSS plus efficacement, d'autant plus qu'ils acceptent l'utilisation des variables définies dans theme.json.

En résumé

Le fichier "theme.json" est donc l'élément central dans un thème Full Site Editing. Il permet de définir les styles globaux, les paramètres de l'éditeur, et les comportements du site de manière structurée, sans passer par du CSS ou du PHP. Il remplace progressivement une grande partie des personnalisations autrefois réparties dans plusieurs fichiers.

Sa structure repose sur trois parties principales :

- version : obligatoire, elle indique à WordPress quelle version du schéma est utilisée (actuellement 3).
- settings : regroupe tous les réglages disponibles dans l'éditeur (couleurs, typographies, espacements, mises en page…). C'est ici qu'on décide de ce que l'utilisateur peut modifier, et dans quelles limites.
- styles : applique les styles par défaut à l'ensemble du site (couleurs, polices, tailles, etc.) ou à des blocs spécifiques.

Chaque sous-clé de settings et styles permet d'affiner le comportement du thème, en définissant par exemple une palette de couleurs, une famille de polices, ou encore des espacements prédéfinis. Des outils comme appearanceTools permettent également d'activer des options avancées dans l'éditeur.

En parallèle, il est important de noter que theme.json ne gère pas encore certains aspects comme les animations (hover, focus, etc.) ou les règles CSS liées au responsive design. Pour ces cas, un fichier CSS complémentaire reste nécessaire. Une approche mobile-first est fortement conseillée pour une gestion responsive efficace, et l'utilisation d'un framework comme Tailwind CSS peut faciliter la mise en œuvre tout en restant compatible avec les variables de theme.json.