PUG

Par Alicia Ross

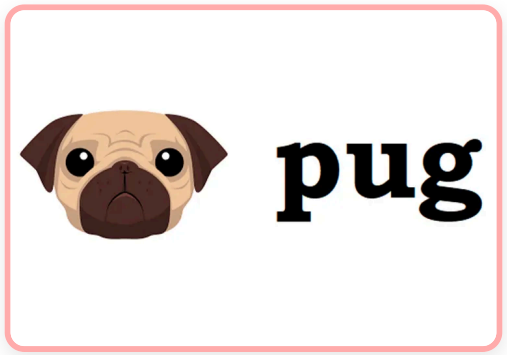
Recherche et développement - PMI 2024

QU'EST-CE QUE PUG ?

Pug est un moteur de templates principalement pour Node.js qui permet de générer du code HTML de manière efficace et concise.

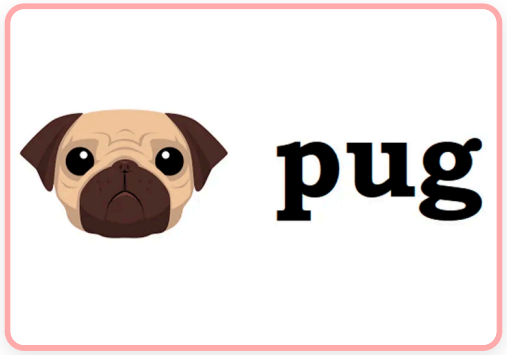
Pug simplifie la création de pages web en utilisant une syntaxe propre et lisible, qui élimine le besoin de balises de fermeture avec l'indentation.

```
div.container
    h1 Titre
    p Ceci est un paragraphe en Pug
```



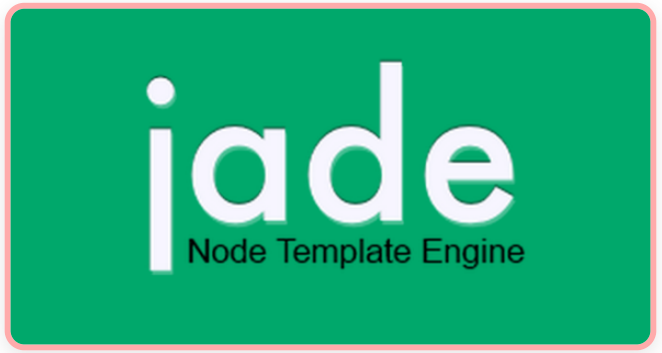
PRINCIPALES CARACTÉRISTIQUES

- Syntaxe simplifiée : Pug utilise une syntaxe basée sur l'indentation, ce qui réduit le besoin de balises HTML répétitives.
- Interpolation de données : Pug permet d'insérer facilement des données dynamiques dans le HTML en utilisant une simple syntaxe de substitution.
- Partials et mixins : Pug permet de créer des fragments réutilisables de code HTML (partials) et des fonctions pour générer du HTML (mixins), ce qui facilite la modularité et la maintenance du code.
- Prise en charge des fonctionnalités modernes : Pug prend en charge les fonctionnalités HTML5 et CSS3, ainsi que des outils comme Sass et LESS pour le style.



SON HISTOIRE

- Pug a commencé sous le nom de Jade, créé par Tim Griesser en 2010. L'idée était de proposer un moteur de templates léger et flexible pour Node.js, qui permettrait aux développeurs de générer du HTML sans avoir à écrire des balises répétitives et complexes.
- La suppression des balises de fermeture et l'usage de l'indentation pour organiser le code étaient des caractéristiques qui ont rapidement séduit les développeurs.
- En 2015, le projet a dû changer de nom, car "Jade" était un nom de marque déposée, ce qui posait un problème légal. Le projet a donc été rebaptisé Pug.
- Le changement de nom s'est accompagné d'une mise à jour majeure du moteur, offrant de nouvelles fonctionnalités et améliorations de performances. La communauté a adopté Pug assez rapidement, et le moteur a continué à croître.



INDENTATION ET STRUCTURE

Contrairement au HTML traditionnel, qui nécessite l'utilisation de balises d'ouverture et de fermeture explicites, Pug s'appuie sur des niveaux d'indentation pour organiser et imbriquer les éléments.

La position des éléments est déterminée par leur niveau d'indentation, ce qui rend le code beaucoup plus lisible. Plus un élément est imbriqué, plus il est indenté.

```
body
    header
        h1 Bienvenue
    main
        p Ceci est un paragraphe dans la section principale.
    footer
        p © 2024 Mon site web
```

- Header, main, et footer sont directement à l'intérieur de la balise body
- Les éléments h1 et p se trouvent à l'intérieur de leurs parents

COMPARAISON AVEC HTML

PUG

```
body
    header
        h1 Bienvenue
    main
        p Ceci est un paragraphe dans la section principale.
    footer
        p © 2024 Mon site web
```

HTML

```
<body>
    <header>
        <h1>Bienvenue</h1>
    </header>
    <main>
        <p>Ceci est un paragraphe dans la section principale.</p>
    </main>
    <footer>
        <p>© 2024 Mon site web</p>
    </footer>
</body>
```

VARIABLES EN PUG

Pug permet d'utiliser des variables pour rendre le code plus dynamique et réutilisable. Les variables peuvent être définies en haut du fichier Pug et utilisées partout dans le document.

Cela permet de modifier facilement des valeurs sans avoir à chercher à travers le code entier.

```
- var title = "Bienvenue"

body
    header
        h1= title
    main
        p Ceci est un paragraphe dans la section principale.
    footer
        p © 2024 Mon site web
```

- La variable `title` est définie avec la valeur `"Bienvenue"`.
- La syntaxe = permet d'afficher la valeur de la variable dans l'élément `h1`.

EXPRESSIONS & INTERPOLATIONS

Les interpolations dans Pug ne se limitent pas seulement à l'insertion de variables ; elles peuvent également inclure des expressions. Cela permet d'effectuer des calculs ou de manipuler des valeurs directement dans le code.

Pour utiliser une expression, il suffit de l'écrire entre les accolades dans l'interpolation. Par exemple :

```
p Le résultat de 5 + 3 est #{5 + 3}.
p La date d'aujourd'hui est #{new Date().toLocaleDateString()}.
```

- Dans cet exemple, l'expression 5 + 3 est évaluée et le résultat (8) est affiché.
- La deuxième interpolation utilise une méthode JavaScript pour afficher la date actuelle, montrant la flexibilité de Pug pour intégrer des opérations dynamiques.

CONDITIONS EN PUG

Les structures conditionnelles permettent d'exécuter un bloc de code en fonction de certaines conditions. En Pug, vous pouvez utiliser les conditions avec la syntaxe de JavaScript.

```
if (age >= 18)
    p Vous êtes majeur.
else
    p Vous êtes mineur.
```

Ce code affiche un message selon l'âge de la personne.

```
if (stock > 0)
    p Produit disponible
else
    p Produit épuisé
```

Ce code affiche un message indiquant si le produit est disponible ou non.

BOUCLES EN PUG

Pug permet de parcourir facilement des tableaux ou des objets grâce à des boucles comme `each`. Cela permet de générer du contenu répétitif de manière concise.

```
//- Définir un tableau d'éléments
- var fruits = ['Pomme', 'Banane', 'Cerise']

//- Boucle pour afficher chaque fruit
each fruit in fruits
    li= fruit
```

```
<ul>
    <li>Pomme</li>
    <li>Banane</li>
    <li>Cerise</li>
</ul>
```

- La boucle `each` parcourt chaque élément du tableau `fruits`.
- Pour chaque élément, un élément `<li>` est généré avec le nom du fruit.

MIXINS

Les mixins en Pug permettent de réutiliser du code, simplifiant ainsi la création de composants récurrents avec des paramètres personnalisés.

```
//- Définition d'un mixin pour créer un bouton avec un texte
mixin bouton(texte)
    button.btn= texte

//- Utilisation du mixin pour générer plusieurs boutons
+bouton('Envoyer')
+bouton('Annuler')
+bouton('Réessayer')
```

```
<button class="btn">Envoyer</button>
<button class="btn">Annuler</button>
<button class="btn">Réessayer</button>
```

- Le mixin `bouton` permet de générer un bouton avec un texte personnalisé.
- Les différents boutons sont générés à partir du même mixin, mais avec des valeurs de texte différentes.

PARTIALS

Les partials en Pug permettent de diviser des sections de code en fichiers séparés et réutilisables. Cela améliore la structure du projet et réduit la duplication de code.

```
//- Exemple d'un fichier partial (header.pug)
header
nav
    ul
    li: a(href='/') Accueil
    li: a(href='/about') À propos
    li: a(href='/contact') Contact
```

```
//- Utilisation d'un partial dans un fichier principal (index.pug)
include header.pug

main
h1 Bienvenue sur notre site
p Ceci est une page d'exemple utilisant des partials en Pug.
```

- Cela permet de mettre à jour plusieurs pages en même temps en modifiant un seul fichier.

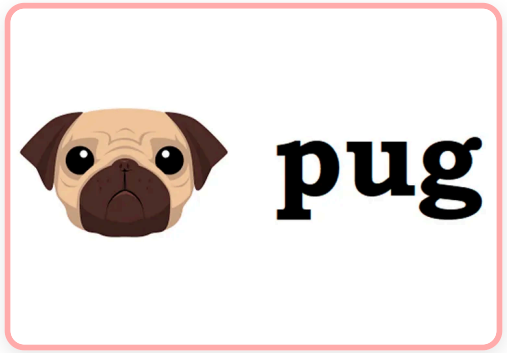
MA PRISE EN MAIN DE PUG

Points forts

- **Simplicité :** Grâce à l'indentation et à l'absence de balises fermantes, Pug offre une syntaxe propre et lisible.
- **Réutilisation du code :** Les mixins et variables permettent de rendre le code plus modulaire et évitent la répétition.

Points faibles

- **Courbe d'apprentissage :** La syntaxe spécifique de Pug peut être difficile au début, surtout pour ceux habitués au HTML traditionnel.



INSTALLATION D'UN PROJET PUG

- **Créer un nouveau projet Node.js :**

```
mkdir projet-pug
cd projet-pug
npm init -y
```

- **Installer Pug :**

```
npm install express pug
```

Express est un framework web pour Node.js qui peut être intégré avec des moteurs de templates pour générer du HTML dynamique.

LIENS UTILES

Voici quelques ressources et liens utiles pour en apprendre davantage sur Pug et améliorer votre maîtrise du langage :

- Tutoriels Pug
- Getting started
- Guide du débutant Pug
- Documentation Express