Libs Docs

Bastien Helec

09/07/2025

PHP_PAGES

```
Bastien Helec
Description : Explication des classes PAGES de la section PHP/PAGES

09-07-2025
```

Sommaire

Model

Description :

La partie PAGES du modèle PHP permet de construire dynamiquement des composants d'interface réutilisables (div, logo, bouton, champ), chacun étant représenté par une classe dédiée. Ces classes facilitent la génération structurée de blocs HTML complexes, pouvant être imbriqués ou enrichis de sous-éléments.

- Div.php ⇒ Div
- Logo.php ⇒ Logo
- Bouton.php ⇒ Bouton
- Fields.php ⇒ Glob_Fields

Div :

Description :

La classe `Div` permet de créer dynamiquement un bloc `<div>` avec un identifiant, une classe CSS et une liste d'éléments enfants (boutons, logos, champs, autres divs).

Arguments du constructeur :

- `$id` : Identifiant unique de la div.
- `$class` : Classe CSS à appliquer.

- `$element` : Tableau d'éléments enfants (instances de Logo, Bouton, Glob_Fields ou Div).

Exemple d'utilisation:

```php
<?php
$logo = new Logo('logo.png', 'logo1', 'logo-class');
$bouton = new Bouton('btn1', 'Cliquez-moi', 'btn-class', [], []);
$field = new Glob_Fields('champ1', 'champ-class', 'input', '', 'text', 'champ1');

$div = new Div('div1', 'container', [$logo, $bouton, $field]);
$dataDiv = $div->gen_div();
?>
```

Cela produit la vue :

Div.php

Méthodes principales :

- `gen_div()` : Retourne un tableau associatif décrivant la div et ses enfants, prêt à être utilisé dans la vue.

Logo :

Description :

La classe `Logo` permet de générer un logo ou une image avec une source, un identifiant et une classe CSS.

Arguments du constructeur :

- `$src` : Chemin ou URL de l'image.
- `$id` : Identifiant unique de l'image.
- `$class` : Classe CSS à appliquer.

Exemple d'utilisation :

```php
<?php
$logo = new Logo('logo.png', 'logo1', 'logo-class');
$dataLogo = $logo->gen_logo();
?>
```

Cela produit la vue :

logo.php

Méthodes principales :

- `gen_logo()` : Retourne un tableau associatif contenant les informations nécessaires à l'affichage du logo.

Bouton :

Description :

La classe `Bouton` permet de générer un bouton HTML personnalisable, avec texte, identifiant, classe CSS, attributs HTML et éventuellement des éléments enfants (pour des boutons complexes ou imbriqués).

Arguments du constructeur :

- `$id` : Identifiant du bouton.
- `$text` : Texte affiché sur le bouton.
- `$class` : Classe CSS à appliquer.
- `$element` : Tableau d'éléments enfants (Div, Glob_Fields, etc.).
- `$attributs` : Tableau associatif d'attributs HTML supplémentaires.

Exemple d'utilisation :

```php
<?php
$bouton = new Bouton('btn1', 'Envoyer', 'btn-class', [], ['type' => 'submit']);
$dataBouton = $bouton->gen_Bouton();
?>
```

Cela produit la vue :

bouton.php

Méthodes principales :

- `gen_Bouton()` : Retourne un tableau associatif décrivant le bouton et ses propriétés.

Glob_Fields :

Description :

La classe `Glob_Fields` permet de générer dynamiquement un champ de formulaire ou tout autre balise HTML, avec gestion des attributs, du texte (ou sous-éléments), du type, du nom, etc.

Arguments du constructeur :

- `$id` : Identifiant du champ.
- `$class` : Classe CSS à appliquer.
- `$balise` : Type de balise HTML à générer (input, textarea, etc.).
- `$text` : Texte ou contenu du champ (peut être un tableau ou un objet pour l'imbrication).
- `$type` : Type du champ (pour les balises input, par exemple).
- `$name` : Nom du champ.

Exemple d'utilisation :

```php
<?php
$field = new Glob_Fields('champ1', 'champ-class', 'input', 'Valeur par défaut', 'text', 'champ1');
$dataField = $field->gen_balise();
?>
```

Cela produit la vue :

Fields.php

Méthodes principales :

- `gen_balise()` : Retourne un tableau associatif décrivant la balise et ses propriétés.

Vue-Implementation

Description :

La partie vue exploite les structures générées par les classes du modèle pour afficher dynamiquement les composants HTML (`div` , `logo` , `bouton` , `champ`) sur la page. Chaque fonction de rendu prend en entrée les données structurées du modèle et produit le HTML correspondant, en gérant l'imbrication et la personnalisation.

Div.php :

Description :

Affiche une `<div>` avec ses attributs et tous ses éléments enfants (logos, boutons, champs, autres divs).

Exemple d'utilisation:

```php
<?php
echo render_element_DIV($dataDiv);
?>
```

Code Source:

```php
function render_element_DIV($div_data){
    $html = "<div id='{$div_data['id']}' class='{$div_data['class']}'>";
    $bouton_html = '';
    $bouton_class = '';

    foreach ($div_data['element'] as $element) {
        if ($element instanceof Logo) {
            $data = $element->gen_logo();
            $html .= "<img src='{$data['src']}' alt='' id='{$data['id']}' class='{$data['class']}' />";
        }
        elseif ($element instanceof Bouton) {
            $data = $element->gen_Bouton();
            $bouton_class = $data['class'];
            $bouton_html .= "<button id='{$data['id']}'>{$data['text']}</button>";
        }
        elseif ($element instanceof Glob_Fields) {
            $html .= render_Glob_Fields($element->gen_balise());
        }
        elseif ($element instanceof Div) {
            $html .= render_element_DIV($element->gen_div());
        }
    }

    if (!empty($bouton_html)) {
        $html .= "<div class='{$bouton_class}'>{$bouton_html}</div>";
    }

    $html .= "</div>";
    return $html;
}
```

---

logo.php :

Description :

Affiche un ou plusieurs logos/images à partir d'un tableau d'objets `Logo`.

Exemple d'utilisation:

```php
<?php
$logos = [$logo];
if(!empty($logos)){
    foreach ($logos as $logo) {
        $data = $logo->gen_logo();
        echo '<img src="'.$data['src'].'" alt="" id="'.$data['id'].'" class="'.$data['class'].'"/>';
    }
} else {
    echo "<p> Pas de logo a afficher </p>";
}
?>
```

Code_source:

```php
<?php
if(!empty ($logos)){
    foreach ($logos as $logo) {
        $data= $logo->gen_logo();
        echo '<img src="'.$data['src'].'" alt="" id="'.$data['id'].'" class="'.$data['class'].'"/>';
    }
} else {
    echo "<p> Pas de logo a afficher </p>";
}
```

---

bouton.php :

Description :

Affiche un ou plusieurs boutons à partir d'un tableau d'objets `Bouton`, en gérant les attributs HTML et les enfants éventuels.

Exemple d'utilisation:

```php
<?php
echo render_bouton([$bouton]);
?>
```

Code source:

```php
<?php
function render_bouton(array $boutons): string {
    $html = '';
    foreach ($boutons as $bouton) {
        if (!($bouton instanceof Bouton)) {
            $html .= "<!-- Élément non Bouton détecté -->";
            continue;
        }
        $data = $bouton->gen_Bouton();
        if ($data['attributs'] && is_array($data['attributs'])) {
            $attributes = '';
            foreach ($data['attributs'] as $key => $value) {
                $attributes .= " {$key}=\"" . htmlspecialchars($value) . "\"";
            }
        } else {
            $attributes = '';
        }
        $html .= "<button id='{$data['id']}' class='{$data['class']}' {$attributes} >";
        if (!empty($data['text'])) {
            $html .= htmlspecialchars($data['text']);
        }
        foreach ($data['element'] as $child) {
            if ($child instanceof Div) {
                $html .= render_element_DIV($child->gen_div());
            } elseif ($child instanceof Glob_Fields) {
                $html .= render_Glob_Fields($child->gen_balise());
            } elseif (is_string($child)) {
                $html .= $child;
            } else {
                $html .= "<!-- Élément enfant inconnu -->";
            }
        }
        $html .= "</button>";
```

```
    }
    return $html;
}
```

Fields.php :

Description :

Affiche dynamiquement un champ ou une balise HTML complexe à partir d'un tableau de propriétés généré par `Glob_Fields`, en gérant l'imbrication de sous-éléments ou de texte.

Exemple d'utilisation :

```php
<?php
echo render_Glob_Fields($dataField);
?>
```

Code Source :

```php
<?php
function render_Glob_Fields(array $Glob_Fields): string {
    $attrs = '';
    if (!empty($Glob_Fields['id'])) {
        $attrs .= " id='{$Glob_Fields['id']}'";
    }
    if (!empty($Glob_Fields['class'])) {
        $attrs .= " class='{$Glob_Fields['class']}'";
    }
    if (!empty($Glob_Fields['name'])) {
        $attrs .= " name='{$Glob_Fields['name']}'";
    }
    if (!empty($Glob_Fields['type'])) {
        $attrs .= " type='{$Glob_Fields['type']}'";
    }
    $html = "<{$Glob_Fields['balise']}{$attrs}>";
    $text = $Glob_Fields['text'];
    if (is_array($text)) {
        foreach ($text as $sub) {
            if (is_object($sub)) {
                if ($sub instanceof Div) {
                    $html .= render_element_DIV($sub->gen_div());
                } elseif ($sub instanceof Glob_Fields) {
                    $html .= render_Glob_Fields($sub->gen_balise());
                }
            } else {
                $html .= $sub;
            }
        }
    } elseif (is_object($text)) {
        if ($text instanceof Div) {
            $html .= render_element_DIV($text->gen_div());
        } elseif ($text instanceof Glob_Fields) {
            $html .= render_Glob_Fields($text->gen_balise());
        }
    } else {
        $html .= $text;
    }
    $html .= "</{$Glob_Fields['balise']}>";
```

```
    return
```