Différence entre parse et pparse

parse($target, $var, $append=false)

Exécute le remplacement des variables dans le template $var.
Stocke le résultat dans la variable interne $target (en mémoire dans la classe).
N'affiche rien : le contenu est juste prêt pour un usage ultérieur ou pour être affiché plus tard.

Si $append vaut true, concatène le résultat au contenu existant de $target.

pparse($target, $var, $append=false)

Fait exactement la même chose que parse (remplacement et stockage dans $target),
MAIS : affiche (echo) le résultat immédiatement (via print).

Dans cet exemple :

```
$tpl->parse('display', 'main');
$tpl->pparse('display', 'main');
```

La 1ère (parse) crée la variable interne display (optionnel ici).
La 2e (pparse) parse ET affiche (echo) ce qu'il y a dans main, et stocke aussi le résultat dans display.

Mais comme on n'utilise pas la variable interne display ailleurs,
→ On peut n'utiliser QUE $tpl->pparse('display', 'main');
(et même, souvent, juste $tpl->pparse('main', 'main'); ou $tpl->pparse('', 'main'); selon la classe).

TL;DR

- Si on veut stocker le rendu (pour traitement, logs, tests, envoi email, etc.) : on utilise parse, puis $tpl->get('display').
- Si on veut juste afficher le template : on utilise uniquement pparse.

Exemple moderne (affichage direct) :

```
$tpl->pparse('main', 'main'); // ou $tpl->pparse('', 'main');
```

Exemple stockage puis affichage personnalisé :

```
$tpl->parse('display', 'main');
$html = $tpl->get('display');
echo strtoupper($html); // Par exemple, post-traitement
```

Par rapport à l'exemple donné ci-dessus, on peut donc simplement faire :

```
$tpl->pparse('main', 'main');
```

ou même, si la classe le permet,

```
$tpl->pparse('', 'main');
```

En résumé :

- parse prépare en mémoire,
- pparse affiche (echo) directement.
- Dans 95 % des cas web/app, on n'a besoin que de pparse !