

Délimiteurs
<code>{{ Variable }}</code>
<code>{{ Var.attrib }}</code> <code>{{ Var[attrib] }}</code>
<code>{{ Variable filtre ... }}</code>
<code>{% Controle %}</code>
<code>{# Commentaires #}</code>

Controles
<code>for else endfor</code> <code>loop.xxx</code>
<code>if elif else endif</code>
<code>macro endmacro</code> <code>call</code>
<code>filter endfilter</code>
<code>set</code> <code>set block endset</code>
<code>include</code> <code>ignore missing</code>
<code>import</code> <code>from import as</code>

Opérateurs
<code>+</code> <code>-</code> <code>/</code> <code>//</code>
<code>%</code> <code>*</code> <code>**</code>
<code>==</code> <code>!=</code> <code>></code> <code>>=</code>
<code><</code> <code><=</code>
<code>and</code> <code>or</code> <code>not</code> <code>(expr)</code>
<code>in</code> <code>is</code> <code> </code> <code>~</code>
<code>./[]</code> <code>foo.bar</code> <code>foor[' bar']</code>

Espaces
<code>{%- -%}</code> supprime espaces avant ou après
<code>{%+}</code>

Filtre Chaines
<code>capitalize(s)</code> <code>center(value, width)</code>
<code>escape(s)</code> <code>forceescape(value)</code>
<code>format(value, *args, **kwargs)</code>
<code>indent(s, width=4, indentfirst=False)</code>
<code>lower(s)</code> <code>upper(s)</code>
<code>pprint(value, verbose=False)</code>
<code>replace(s, old, new, count=None)</code>

Filtre Chaines (cont)
<code>string(object)</code> <code>striptags(value)</code>
<code>title(s)</code> <code>trim(value)</code>
<code>urlencode(value)</code>
<code>urlize(value, trim_url_limit, nofollow, target)</code>
<code>wordcount(s)</code> <code>xmlattr(d, autoescape)</code>

Filtre Nombres
<code>abs(number)</code> <code>random(seq)</code>
<code>int(value, default=0)</code> <code>float(value, default=0.0)</code>
<code>sum(iterable, attribute=None, start=0)</code>

Héritage
<code>block name</code> <code>endblock</code>
<code>extends " template "</code>
<code>self.name</code> <code>super()</code>
Les endblock peuvent être nommés

Global
<code>range(start, stop, step)</code>
<code>libsum(n=5, html=True, min=20, max=100)</code>
<code>dict(**items)</code> <code>class cycler(*items)</code>
<code>reset()</code> <code>next()</code>
<code>current</code> <code>class joiner(sep=',')</code>

Filtre Divers
<code>attr(obj, name)</code> <code>reject attr()</code>
<code>default(value, default, boolean=False) d</code>
<code>filesizeformat(value, binary=False)</code>
<code>safe(value)</code>
<code>select()</code> <code>select attr()</code>
Utilisé dans <code>{{ Variable filtre ... }}</code> ou dans les commandes

Filtre Liste
<code>dictsort(value, case_sensitive='key')</code>
<code>first(seq)</code> <code>last(seq)</code>
<code>groupby(value, attr)</code> <code>map()</code>
<code>join(value, d=' u', attribute=None)</code>
<code>length(object)</code> <code>l1st()</code>
<code>reverse(value)</code>
<code>select()</code> <code>select</code>
<code>slice(value, slices, filter_width)</code>
<code>sort(value, reverse, case_sensitive, attribute)</code>
<code>sum(iterable, attribute=None, start=0)</code>

Tests
<code>callable(object)</code> <code>divisibleby</code>
<code>escaped(value)</code> <code>mapping(value)</code>
<code>iterable(value)</code> <code>sequence(value)</code>
<code>string(value)</code> <code>number(value)</code>
<code>lower(value)</code> <code>upper(value)</code>
<code>defined(value)</code> <code>undefined(value)</code>
<code>none(value)</code>
<code>even(value)</code> <code>odd(value)</code>
<code>equalto(value, other)</code> <code>sameas(value)</code>

