Shopify Liquid Cheat Sheet

by Jacques Blom (jacquesblom) via cheatography.com/217031/cs/47570/

Best Practices & Tips

- Use {% render %} for isolated scope
- Use {% comment %} for inline notes
- Avoid nested if statements
- Use {% case %} for many options
- Minimize use of filters in loops
- Use snippets for reusable code
- Separate concerns per file
- Test variables with unless nil
- Limit array iterations
- Avoid complex conditional logic
- Use forloop variables wisely
- Check for nil before accessing

Common Objects

product	Current product object
collection	Current collection object
article	Current blog article
shop	Shop/theme settings
page	Current page object
customer	Logged-in customer data
cart	Shopping cart object
settings	Theme configuration
request	Request information (URL, etc)

Access global Liquid objects like product, collection, shop, and customer. These contain data about the current context. Properties are accessed using dot notation (e.g., product.title).

Tags

Conditional statement

```
{% if condition %}
...
{% endif %}
```

Tags (cont)

Unless condition (opposite of if)

```
{% unless condition %}
...
{% endunless %}
```

Switch/Case statement

```
{% case variable %}
...
{% when value %}
...
{% endcase %}
```

Loop (for each item)

```
{% for item in collection %}
...
{% endfor %}
```

Liquid tags are used to create dynamic content and control flow in Shopify theme templates. They allow you to add conditional logic, loops, and variable assignments to your template code. Tags are enclosed in {% %} brackets.

Variable Assignment

Assign Variable

```
{% assign var = 'value' %}
```

Assign with Filter

```
{% assign var = var | filter %}
```

Capture Output

```
{% capture var %}
content
{% endcapture %}
```

Increment Counter

```
{% increment var %}
```

Decrement Counter

```
{% decrement var %}
```

Create variables using assign, capture output in variables, or use increment/decrement for counters. Variables are scoped to the template and can be reused throughout.

Variable Output

{{ **variable** }}	Output variable
{{ **'text'** }}	Output text string
{{ **5** }}	Output number
{{ **product.title** }}	Access object property
{{ **array[0]** }}	Access array element by index
{{ **'text' | filter** }}	Apply single filter to value
{{ **var1 | filter1 | filter2** }}	Chain multiple filters

Output variables, strings, numbers, and access object properties and array elements using dot notation or bracket notation. Apply filters to transform values.

Template Tags

Include Snippet

```
{% include 'snipp et- name' %}
```

Include with Variable

```
{% include 'snippet' with produc
```

Render Snippet (Isolated)

```
{% render 'snipp et- name' %}
```

Comment Block

```
{% comment %}...{% endcomment %
```

Raw/Literal Block

```
{% raw %}...{% endraw %}
```

Include snippets, pass variables to includes, or use render for isolated scope. Use comment tags for multi-line comments or raw tags to prevent Liquid processing.

By **Jacques Blom** (jacquesblom)

cheatography.com/jacquesblom/

Published 10th January, 2026.
Last updated 10th January, 2026.

Advanced Filters

base64_encode	Encode to base64
base64_decode	Decode from base64
url_encode	Encode URL parameter
url_decode	Decode URL parameter
escape	Escape HTML characters
escape_once	Escape only unescaped HTML
json	Output as JSON string
md5	MD5 hash
sha1	SHA1 hash
compact	Remove nil/null values
default: 'fallback'	Fallback if nil/empty
where_exp: 'condition'	Advanced array filtering

Encode/decode base64 and URLs, escape HTML, generate hashes (MD5, SHA1), or output JSON. Use compact to remove nil values and where_exp for complex array filtering.

String Filters

upcase	Convert to UPPERCASE
downcase	Convert to lowercase
capitalize	Capitalize first letter
size	Get string/array length
split: '|'	Split string by delimiter
join: ', '	Join array with separator
strip	Remove whitespace
append: 'text'	Add text to end
prepend: 'text'	Add text to start
remove: 'text'	Remove substring

String Filters (cont)

replace: 'a','b'	Replace 'a' with 'b'

Transform strings with upcase, downcase, capitalize, split, join, strip, and replace. Use size to get length. These filters are chainable for multiple transformations.

Math Filters

plus: n	Add n to number
minus: n	Subtract n from number
times: n	Multiply number by n
divided_by: n	Divide number by n
modulo: n	Get remainder of division
round: n	Round to n decimals
ceil	Round up to nearest integer
floor	Round down to nearest integer
abs	Absolute value

Perform mathematical operations on numbers with plus, minus, times, divided_by, and modulo. Use round, ceil, and floor for rounding, or abs for absolute value.

Array Filters

sort	Sort array alphabetically
reverse	Reverse array order
first	Get first element
last	Get last element
map: 'property'	Map/extract property from objects
where: 'key','val'	Filter array by condition
uniq	Remove duplicate values

Array Filters (cont)

concat: array	Concatenate/join two arrays
size	Get array length

Manipulate arrays with sort, reverse, first, last, and map. Filter arrays with where, remove duplicates with uniq, or join arrays with concat. Size returns array length.

By **Jacques Blom** (jacquesblom)

cheatography.com/jacquesblom/

Published 10th January, 2026.
Last updated 10th January, 2026.