attributes (/reference/attributes)

case (/reference/case)

code (/reference/code)

comments (/reference/comments)

conditionals (/reference/conditionals)

doctype (/reference/doctype)

extends (/reference/extends)

filters (/reference/filters)

includes (/reference/includes)

inheritance (/reference/inheritance)

iteration (/reference/iteration)

mixins (/reference/mixins)

plain text (/reference/plain-text)

tags (/reference/tags)

Iteration

Jade supports two primary methods of iteration, `each` and `while`.

each

Jade's first-class iteration syntax makes it easier to iterate over arrays and objects within a template:

```
ul
  each val in [1, 2, 3, 4, 5]
    li= val
```

```
<ul>
  <li>1</li>
  <li>2</li>
  <li>3</li>
  <li>4</li>
  <li>5</li>
</ul>
```

You can also get the index as you iterate:

```
ul
  each val, index in ['zero', 'one', 'two']
    li= index + ': ' + val
```

```
<ul>
  <li>0: zero</li>
  <li>1: one</li>
  <li>2: two</li>
</ul>
```

Jade also lets you iterate over the keys in an object:

```
ul
  each val, index in {1:'one',2:'two',3:'three'}
    li= index + ': ' + val
```

```
<ul>
  <li>1: one</li>
  <li>2: two</li>
  <li>3: three</li>
</ul>
```

The object or array to iterate over is just plain JavaScript so it can be a variable or the result of a function call or almost anything else. e.g.

```
- var values = [];
ul
  each val in values.length ? values : ['There are no values']
    li= val
```

```
<ul>
  <li>There are no values</li>
</ul>
```

You can also use `for` as an alias of `each`.

while

You can also use `while` to create a loop:

```
- var n = 0
ul
  while n < 4
    li= n++
```

```
<ul>
  <li>0</li>
  <li>1</li>
  <li>2</li>
  <li>3</li>
</ul>
```