Built-in template tags and filters

This document describes Django's built-in template tags and filters. It is recommended that you use the automatic documentation, if available, as this will also include documentation for any custom tags or filters installed.

Built-in tag reference

autoescape

Controls the current auto-escaping behavior. This tag takes either `on` or `off` as an argument and that determines whether auto-escaping is in effect inside the block. The block is closed with an `endautoescape` ending tag.

When auto-escaping is in effect, all variable content has HTML escaping applied to it before placing the result into the output (but after any filters have been applied). This is equivalent to manually applying the `escape` filter to each variable.

The only exceptions are variables that are already marked as "safe" from escaping, either by the code that populated the variable, or because it has had the `safe` or `escape` filters applied.

Sample usage:

```
{% autoescape on %}
    {{ body }}
{% endautoescape %}
```

block

Defines a block that can be overridden by child templates. See Template inheritance for more information.

comment

Ignores everything between `{% comment %}` and `{% endcomment %}`.

csrf_token

In the Django 1.1.X series, this is a no-op tag that returns an empty string for future compatibility purposes. In Django 1.2 and later, it is used for CSRF protection, as described in the documentation for Cross Site Request Forgeries.

cycle

Cycles among the given strings or variables each time this tag is encountered.

Within a loop, cycles among the given strings each time through the loop:

```
{% for o in some_list %}
    <tr class="{% cycle 'row1' 'row2' %}">
        ...
    </tr>
{% endfor %}
```

You can use variables, too. For example, if you have two template variables, `rowvalue1` and `rowvalue2`, you can cycle between their values like this: