# django-awesome-snippets Documentation

*Release 0.2*

**Gael Pasgrimaud**

March 16, 2015

A snippet is a class attribute that use a template to render his value. A kind of template tag but as an instance attribut.

The snippet will use the template `{app_label}/snippets/{model}_{name|func_name}.html` by default.

The current `object`, the `settings` and the `MEDIA_URL` are available in the template by default.

There is various way to use the snippet decorator:

```python
from django.db import models
from snippets import snippet

class MyModel(models.Model):

    class Meta:
        app_label = 'app'

    name = models.CharField(max_length=255)

    # simple attribute (name is required)
    h1 = snippet(name='h1')

    # as a property. Then then function may return a dict. This dict will be
    # available in the template
    @snippet
    def a_tag(self):
        return dict(url=self.get_absolute_url())

    # as a property with extra arguments
    @snippet(template='app/snippets/mymodel_a_tag.html')
    def custom_template(self):
        return dict(url=self.get_absolute_url())

    # as a cached property using another template (specified by name)
    # here the template used will be app/snippets/mymodel_a_tag.html instead of
    # app/snippets/mymodel_a_cached_tag.html
    @snippet(name='a_tag', delay=60)
    def a_cached_tag(self):
        return dict(url=self.get_absolute_url())

    def get_absolute_url(self):
        return '/mymodel/%s' % self.id
```

CHAPTER 1

# Module content

CHAPTER 2

# Bugs

Use the github tracker

CHAPTER 3

# Indices and tables

- *genindex*
- *modindex*
- *search*