django-deletion-side-effects Documentation

Release 0.1.1

Wes Kendall

May 06, 2015

Django Deletion Side Effects provides a framework for gathering side effects of deleting objects in Django.

For example, imagine your website has an interface to delete users. When deleting a user in your app, it will also delete any memberships to groups along with any API credentials of that user. This information would be useful to display before the user is deleted from the website. Django Deletion Side Effects allows you to easily populate this information across your project as it grows.

Setting Up A Side Effect Handler

Similar to registering a signal handler, this package allows you to set up deletion handlers and register them. Each deletion side effect class must define the following:

1. A *deleted_obj_class* variable. This variable denotes the class of the object being deleted.
2. A *get_side_effects* method. This method is passed a list of objects of *deleted_obj_class* type that are candidates for deletion. The method returns a tuple of objects that are affected by deletion of the objects for deletion (i.e the side effect objects) and a list of objects that will be cascade deleted if the candidate objects are deleted.
3. A *get_side_effect_message* method. This method is passed all of the side effect objects from gathering side effects with the class. The method is responsible for returning a human-readable string of the side effects.

A side effect handler can best be illustrated with an example. Assume we have the following models:

```
class GroupType(models.Model):
    name = CharField(max_length=64)


class Group(models.Model):
    group_type = models.ForeignKey(GroupType)
    name = CharField(max_length=64)
```

In this case, a *Group* will be cascade deleted with the deletion of a *GroupType*. To set up a side effect for this, do the following:

```
from deletion_side_effects import register_deletion_side_effects, BaseDeletionSideEffects


class CascadeGroupDeletionSideEffect(BaseDeletionSideEffect):
    deleted_obj_class = GroupType

    def get_side_effects(self, deleted_objs):
        """
        Given a list of deleted group types, return the list of side effect objects
        and the list of objects that will be deleted as a result. In this case, the lists are the sa
        """
        deleted_groups = Group.objects.filter(group_type__in=deleted_objs)
        return deleted_groups, deleted_groups

    def get_side_effect_message(self, side_effect_objs):
        """
        Prints out the message about groups being deleted.
        """
        return u'{0} group{1} will be deleted'.format(len(side_effect_objs), 's' if len(side_effect_
# Register the side effect. Note that this is best called in the App Config's ready() method
register_deletion_side_effects(CascadeGroupDeletionSideEffect)
```

In the above example, the side effect class inherits *BaseDeletionSideEffects*. The side effect handler is registered with the *register_deletion_side_effects* function. Note that the side effect handlers will need to be connected in the app config's *ready* method for your app with side effects.

Gathering Side Effects

In order to gather side effects for a list of deleted objects of the same type, do the following:

```
from deletion_side_effects import gather_deletion_side_effects


# Find all of the side effects of deleting all group types
side_effects = gather_deletion_side_effects(GroupType, GroupType.objects.all())

print side_effects
[{
    'msg': u'2 groups will be deleted',
    'side_effect_objs': [
        <Group: group1>,
        <Group: group2>
    ]
}]
```

This case follows with using the models defined in the example above. In this example, we retrieve the side effects of deleting every group type by passing the *GroupType* model and the iterable of all group types to *gather_deletion_side_effects*. The return value of the function has a list of all side effects. Each side effect is a dictionary that has a *msg* field for the side effect message. It also has a list of side effect objects related to the message in the *side_effect_objs* field.