Django 1.7 Updates

Some Highlights and Observations

Project Experience

- I have converted one project from 1.6.x to 1.7x

- No real issues so far

- Some difficulties with my development env.

Some Key Changes

- Schema Migrations

- App Loading

- System checks

- QuerySet.as_manager()

- Custom lookups and transforms

Schema Migrations

Finally, a baked in capability!

- About the same as South..

- Easy migration from South

- No separate datamigration command

- Migrations no longer load initial data

- Squash Migrations

Upgrade from south

- Insure your database is up-to-date

- Remove 'south' from installed apps

- Remove all existing south migrations, but leave migrations dir and __init__.py

- Run: `python manage.py makemigrations`

- Run: `python manage.py migrate`

App Loading

- This is pretty significant IMO

- No longer need to hack models.py

- Override: ready() method to add custom init, register signals, etc.

```
# rock_n_roll/apps.py
from from django.apps import AppConfig
class RockNRollConfig(AppConfig):
    name = 'rock_n_roll'
    verbose_name = "Rock 'n' roll"
```

System check framework

- Static checks for many aspects of the Django app

- models, signals, admin, compatibility

QuerySet Changes

- Resolves issue with returning query set from custom manager.

  - Cannot get back to manager

- Instead, define methods on custom queryset.

- Use as manager with "Queryset.as_manager()

- Specify queryset on reverse relation

Queryset reverse

```python
class Blog(models.Model):
  pass

class Entry(models.Model):
  blog = models.ForeignKey(Blog)
  objects = models.Manager() # Default
Manager
  entries = EntryManager() # Custom
Manager

b = Blog.objects.get(id=1)
b.entry_set(manager='entries').all()
```

Custom Lookups and Transformations

```python
from django.db.models import Lookup

class NotEqual(Lookup):
    lookup_name = 'ne'

    def as_sql(self, qn, connection):
        lhs, lhs_params = self.process_lhs(qn,
            connection)
        rhs, rhs_params = self.process_rhs(qn,
            connection)
        params = lhs_params + rhs_params
        return '%s <> %s' % (lhs, rhs), params
```

Now, register the lookup:

```python
from django.db.models.fields import Field
Field.register_lookup(NotEqual)
```

That's all I have!