








DJANGO 1.5 CHEATSHEET

Model fields	Common model field options	Grove.io Enterprise chat, backed by IRC, built by Revsys. http://grove.io/	ModelAdmin options & callbacks	
AutoField BigIntegerField BooleanField, NullBooleanField CharField max_length = 100 CommaSeparatedIntegerField max_length = 50 DateField, DateTimeField DecimalField max_digits = 10 decimal_places = 2 EmailField max_length = 75 FileField upload_to = "files/%Y/%m/%d" storage = file_storage_obj max_length = 100 FilePathField path = "/var/images" match = r"\\.pdf\$" recursive = True max_length = 100 FloatField FileField upload_to = "/uploads" storage = file_storage_obj max_length = 100 ImageField upload_to = "/uploads" storage = file_storage_obj height_field = "field_name" width_field = "field_name" max_length = 100 IntegerField, PositiveIntegerField, PositiveSmallIntegerField IPAddressField GenericIPAddressField protocol = "both" "IPv4" "IPv6" unpack_ipv4 = False SlugField max_length = 100 SmallIntegerField TextField TimeField URLField	null = False blank = False choices = [(1, "Choice 1"), ...] db_column = "column_name" db_index = False db_tablespace = "tablespace_name" default = value_or_callable error_messages = {"name": "Message", ...} help_text = "long help text" primary_key = False exclude(**lookups) = False unique_for_date = "date_field" unique_for_month = "month_field" verbose_name = "Field Name" validators = [validator_obj, ...]	 QuerySet methods all() annotate(**annotations) dates(field, "year/month/day", "ASC/DESC") defer(*fields) distinct(*fields) (1.4) exclude(**lookups) filter(**lookups) none() only(*fields) order_by(*fields) prefetch_related('field1', 'field2', ...) raw(sql, params) reverse() select_for_update(nowait=False) select_related('field', 'field2', ..., depth=1) using("db_alias") values(*fields) values_list(*fields) values_list(field, flat=True)	actions = ["method", callback, ...] actions_on_bottom = False actions_on_top = True actions_selection_counter = True add_form_template = 'add_form.html' change_form_template = 'change_form.html' change_list_template = 'changelist.html', date_hierarchy = "date_field" delete_confirmation_template = 'delete.html' delete_selected_confirmation_template = 'del.html' exclude = ["field_name", ...] fields = ["field_name", ...] fieldsets = [{"Name": ["fields": ["field_name", ...]]}, ...] filter_horizontal = ["m2m_field", ...] filter_vertical = ["m2m_field", ...] form = ModelFormClass formfield_overrides = {TextField: {"widget": MyWidget}, ...} inlines = [InlineClass, ...] list_display = ["field_name", callable, ...] list_display_links = ["field_name", ...] list_editable = ["field_name", ...] list_filter = ["field_name", ...] list_max_show_all = 200 list_per_page = 100 list_select_related = False object_history_template = 'admin/history.html', ordering = ["field_name"] paginator = Paginator prepopulated_fields = {'slug': ['title', ...]} radio_fields = {'fk_field': admin.HORIZONTAL, ...} raw_id_fields = ['fk_or_m2m_field', ...] readonly_fields = ["field_name", ...] save_as = False save_on_top = False search_fields = ["field_name"]	
	Model Meta options abstract = False app_label = "applabel" db_table = "table_name" db_tablespace = "space_name" get_latest_by = "field_name" index_together = [("field1", "field2"), ...] managed = False order_with_respect_to = "field" ordering = ["field_name", ...] permissions = [{"code", "text label"}, ...] proxy = False unique_together = [("field1", "field2"), ...] verbose_name = "verbose name" verbose_name_plural = "plural verbose names"	Not chainable aggregate(**aggregations) bulk_create(list_of_objects) count() create(**attributes) delete() exists() get(**lookups) get_or_create(**attributes, defaults={}) in_bulk(id_list) iterator() latest(field) update(**attributes)		
	ForeignKey on_delete options CASCADE cascades; default PROTECT prevent related deletion SET_NULL Set to NULL SET_DEFAULT Set to field's default SET(value) / SET(callback) Set to value / callback() DO_NOTHING No action (in Python)	Lookups exact, iexact = "string" contains, icontains = "string" startswith, endswith, istartswith, iendswith = "string" in = list / queryset gt, lt, gte, lte = date / int / float / decimal range = (lower, upper) year, month, day = int isnull = bool regex, iregex = r"^\reg\. exp\."		
	Model/Form field validators MaxLengthValidator(max_length) MinLengthValidator(min_length) MaxValueValidator(max_value) MinValueValidator(min_value) RegexValidator(regex, message=None, code=None) URLValidator(verify_exists=False) validate_email validate_slug validate_ipv4_address (also _ipv6_, _ipv46_)	Aggregation/annotation functions Avg(field) StdDev(field, sample=False) Count(field, distinct=False) Sum(field) Max(field) Variance(field, sample=False) Min(field)		
	 Read the Docs Create, host, and browse documentation. http://readthedocs.org/			
	Model relationships ForeignKey(OtherModel) related_name = "things" limit_choices_to = Q(foo="bar", ...) to_field = "key_field_name" on_delete = (see right) ManyToManyField(OtherModel) related_name = "things" limit_choices_to = Q(foo="bar", ...) to_field = "key_field_name" symmetrical = True, through = RelationshipModel db_table = "table_name" OneToOneField(OtherModel) parent_link = False related_name = "thing" limit_choices_to = Q(foo="bar", ...) to_field = "key_field_name" on_delete = (see right)	Signals django.db.models.signals pre_init, post_init(sender, args, kwargs) pre_save, post_save(sender, instance, using) pre_delete, post_delete(sender, instance, using) m2m_changed(sender, instance, action, reverse, model, pk_set, using) class_prepared(sender) post_syncdb(sender, app, created_models, verbosity, interactive) django.core.signals request_started, request_finished(sender) got_request_exception(sender, request)	django.test.signals settings_changed(sender, setting, value) template_rendered(sender, template, context) django.db.backends.signals connection_created(sender, connection) django.contrib.auth.signals user_logged_in, user_logged_out(sender, request, user) user_login_failed(sender, credentials) django.contrib.comments.signals comment_will_be_posted(sender, comment, request) comment_was_posted(sender, comment, request) comment_was_flagged(sender, comment, flag, created, request)	add_view(self, request, form_url=" ", extra_context={}) changelist_view(self, request, extra_context={}) change_view(self, request, object_id, extra_context={}) delete_model(self, request, obj) delete_view(self, request, object_id, extra_context={}) formfield_for_choice_field(self, db_field, request, **kw) formfield_for_foreignkey(self, db_field, request, **kw) formfield_for_manytomany(self, db_field, request, **kw) get_list_display(self, request) get_list_display_link(self, request, list_display) get_ordering(self, request) get_paginator(self, queryset, per_page, orphans=0, allow_empty_first_page=True) get_prepopulated_fields(self, request, obj=None) get_readonly_fields(self, request, obj=None) get_urls(self) has_add_permission(self, request) has_change_permission(self, request, obj=None) has_delete_permission(self, request, obj=None) history_view(self, request, object_id, extra_context={}) message_user(self, request, message) queryset(self, request) save_formset(self, request, form, formset, change) save_model(self, request, obj, form, change) save_related(self, request, obj=None)
			 Haystack Modular search for Django. http://haystacksearch.org/	
			 south Intelligent migrations. http://south.aeracode.org/	

Common field options		 Django Debug Toolbar Debug better, faster. http://django.me/djdt		 Tastypie Creating delicious APIs for Django apps since 2010. http://tastypieapi.org/		Datetime formatting																																										
error_messages = {"code": "Message", ...} help_text = "help text" initial = value label = "field label" localize = False required = True validators = [validator, ...] widget = WidgetClass	Form fields	Form widgets All widgets attrs = {'class': 'fancy', ...} CheckboxInput check_test = callback(value) CheckboxSelectMultiple ClearableFileInput DateInput format = "%Y-%m-%d" DateTimeInput format = "%Y-%m-%d %H:%M" FileInput HiddenInput, MultipleHiddenInput MultiWidget NullBooleanSelect PasswordInput render_value = False RadioSelect Select SelectDateWidget years = [2010, 2011, ...] SelectMultiple SplitDateTimeWidget Textarea TextInput TimeInput format = "%H:%M:%S"	Template filters add:"2" addslashes capfirst center:"15" cut:" " date:"JS F Y H:i" default:"nothing" default_if_none:"nothing" dictsort:"key" dictsortreversed:"key" divisibleby:"4" escape escapejs filesizeformat first fix_ampersands floatformat floatformat:"3" force_escape get_digit:"2" iriencode join:"/" length length_is:"4" linebreaks linebreaksbr linenumbers ljust:"10" lower make_list phonenumeric	Template tags {% block name %}...{% endblock %} {% csrf_token %} {% cycle "row1" "row2" [as varname] [silent] %} {% debug %} {% extends "base.html" %} {% filter force_escape lower %}...{% endfilter %} {% firstof var1 var2 "fallback" %} {% for i in list [reversed] %}...{% empty %}...{% endfor %} {{ forloop.counter }} {{ forloop.counter0 }} {{ forloop.revcounter }} {{ forloop.revcounter0 }} {{ forloop.first }} {{ forloop.last }} {{ forloop.parentloop }} {% if condition %}...{% elif condition %}...{% else %}...{% endif %} {% ifchanged %}...{% else %}...{% endifchanged %} {% include "other/template.html" [with who="Jane" ... [only]] %} {% load [foo bar ... from] other_library %} {% now "JS F Y H:i" %} {% regroup people by gender as gender_list %} {% spaceless %}...{% endspaceless %} {% templatetag openblock / closeblock / openvariable / closevariable / openbrace / closebrace / opencomment / closecomment %} {% url path.to.view arg1 arg2 arg3=v1 arg4=v2 [as the_url] %} {% widthratio this_value max_value 100 %} {% verbatim %} ... {% endverbatim %} {% with alpha=1 beta=2 ... %}...{% endwith %}	template strftime a a.m. / p.m. A AM / PM b jan, feb, ... B c 2008-01-02T10:30:00.000123 d %c Fri Mar 4 16:43:23 2011 D 01 - 31 e Mon, Tue, ... E UTC, CST (timezone name) F (alternate long month) f 1, 1:30 F %B January, February, ... g %m 1 - 12 G 0 - 23 h %l 01 - 12 H %H 00 - 23 i %M 00 - 59 (minutes) j 1 - 31 j 001 - 365 l %A Monday, Tuesday, ... L (leap year?) m 01 - 12 M %b Jan, Feb, ... n 1 - 12 N Jan., Feb., March, ... o (ISO-8601 week-numbering year) O +0200 P 1 a.m., noon, 2:30 p.m. r Thu, 21 Dec 2000 16:01:07 +0200 s %S 00 - 59 (seconds) S st, nd, rd, th t 28 - 31 T %Z EST, UTC, ... u (microseconds) U (unix timestamp) w %w 0 (Sun.) - 6 (Sat.) W 1 - 53 y %y 99 Y %Y 1999 x 03/04/11 X 16:43:23 z 0 - 365 Z (tz offset, seconds) DATE_FORMAT (as defined in settings) DATETIME_FORMAT (as defined in settings) SHORT_DATE_FORMAT (as defined in settings) SHORT_DATETIME_FORMAT (as defined in settings)																																											
BooleanField, NullBooleanField CharField max_length = 100 min_length = 10 ChoiceField, MultipleChoiceField choices = [(1, "Choice 1"), ...] TypedChoiceField, TypedMultipleChoiceField choices = [(1, "Choice 1"), ...] coerce = callback(value) empty_value = "" ModelChoiceField, ModelMultipleChoiceField queryset = Model.objects.all() empty_label = u"....." DateField input_formats = ["%Y-%m-%d", ...] DateTimeField input_formats = ["%Y-%m-%d %H:%M", ...] DecimalField max_value = Decimal(100) min_value = Decimal(10) max_digits = 10 decimal_places = 2 EmailField FileField, ImageField FilePathField path = "/home/images" recursive = True match = "/*.pdf" FloatField max_value = 100.0 min_value = 10.0 IntegerField max_value = 100 min_value = 10 IPAddressField GenericIPAddressField protocol = "both" "IPv4" "IPv6" unpack_ipv4 = False RegexField regex = r'\w+' max_length = 100 min_length = 10 SlugField max_length = 100 min_length = 10 TimeField input_formats = ["%H:%M:%S", ...] URLField max_length = 100 min_length = 10 verify_exists = False validator_user_agent = "Django/1.3" SplitDateTimeField input_date_formats = ["%Y-%m-%d", ...] input_time_formats = ["%H:%M:%S", ...]	Celery Distributed, asynchronous task queue. http://celeryproject.org/	<table border="1"> <thead> <tr> <th>HttpRequest</th> <th>HttpResponse</th> </tr> </thead> <tbody> <tr> <td>__iter__()</td> <td>__init__(content="", mimetype=None, status=200, content_type=None)</td> </tr> <tr> <td>body</td> <td>content_type=None</td> </tr> <tr> <td>build_absolute_uri(path)</td> <td>__delitem__(header)</td> </tr> <tr> <td>COOKIES</td> <td>__getitem__(header)</td> </tr> <tr> <td>encoding</td> <td>__setitem__(header, val)</td> </tr> <tr> <td>GET, POST, REQUEST</td> <td>delete_cookie(key, path="/", domain=None)</td> </tr> <tr> <td>FILES</td> <td>flush()</td> </tr> <tr> <td>get_full_path()</td> <td>has_header(header)</td> </tr> <tr> <td>get_host()</td> <td>set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)</td> </tr> <tr> <td>get_signed_cookie(key)</td> <td>flush()</td> </tr> <tr> <td>is_ajax()</td> <td>has_header(header)</td> </tr> <tr> <td>is_secure()</td> <td>set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)</td> </tr> <tr> <td>META</td> <td>flush()</td> </tr> <tr> <td>method</td> <td>has_header(header)</td> </tr> <tr> <td>path</td> <td>set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)</td> </tr> <tr> <td>path_info</td> <td>flush()</td> </tr> <tr> <td>read(size=None)</td> <td>has_header(header)</td> </tr> <tr> <td>readline(), readlines()</td> <td>set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)</td> </tr> <tr> <td>session</td> <td>flush()</td> </tr> <tr> <td>urlconf</td> <td>has_header(header)</td> </tr> <tr> <td>user</td> <td>set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)</td> </tr> </tbody> </table>	HttpRequest	HttpResponse	__iter__()	__init__(content="", mimetype=None, status=200, content_type=None)	body	content_type=None	build_absolute_uri(path)	__delitem__(header)	COOKIES	__getitem__(header)	encoding	__setitem__(header, val)	GET, POST, REQUEST	delete_cookie(key, path="/", domain=None)	FILES	flush()	get_full_path()	has_header(header)	get_host()	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)	get_signed_cookie(key)	flush()	is_ajax()	has_header(header)	is_secure()	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)	META	flush()	method	has_header(header)	path	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)	path_info	flush()	read(size=None)	has_header(header)	readline(), readlines()	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)	session	flush()	urlconf	has_header(header)	user	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)	View shortcuts (django.shortcuts) render (request, template, context_dict={}, context_instance=RequestContext, content_type="text/html", status=200, current_app=None) redirect (to, permanent=False, *args, **kw) get_object_or_404 (Model, **lookup) get_list_or_404 (Model, **lookup)	django.test.TestCase client_class = django.test.client fixtures = ["fix1.json", "fix2.json", ...] urls = "my.app.urls" multi_db = False assertContains, assertNotContains (resp, text, count=None, status_code=200, msg_prefix="", html=False) assertFieldOutput (cls, valid, invalid, field_args=None, field_kwargs=None, empty_value='') assertFormError (resp, form, field, errors, msg_prefix='') assertHTMLEqual, assertHTMLNotEqual (html1, html2, msg='') assertXMLEqual, assertXMLNotEqual (xml1, xml2, msg='') assertQuerysetEqual (qs, values, transform=repr, ordered=True) assertRedirects (resp, url, status_code=302, target_status_code=200, msg_prefix='') assertTemplateUsed, assertTemplateNotUsed (resp, tpl) with self.assertNumQueries (num, func): ... with self.assertRaisesMessage (exe, msg): ... with self.settings (SETTING='new_value'): ...
HttpRequest	HttpResponse																																															
__iter__()	__init__(content="", mimetype=None, status=200, content_type=None)																																															
body	content_type=None																																															
build_absolute_uri(path)	__delitem__(header)																																															
COOKIES	__getitem__(header)																																															
encoding	__setitem__(header, val)																																															
GET, POST, REQUEST	delete_cookie(key, path="/", domain=None)																																															
FILES	flush()																																															
get_full_path()	has_header(header)																																															
get_host()	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)																																															
get_signed_cookie(key)	flush()																																															
is_ajax()	has_header(header)																																															
is_secure()	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)																																															
META	flush()																																															
method	has_header(header)																																															
path	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)																																															
path_info	flush()																																															
read(size=None)	has_header(header)																																															
readline(), readlines()	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)																																															
session	flush()																																															
urlconf	has_header(header)																																															
user	set_cookie, set_signed_cookie(key, value, max_age=None, expires=None, path="/", domain=None, secure=None, httponly=True)																																															
			 Find more details in Django's official docs: <a href="http://django.me/<any object here>">http://django.me/<any object here>																																													