
django-sciweb Documentation

Release 0.9

Marcos Lopez

December 22, 2014

1	Requirements	3
2	Static Directories	5
3	Local Settings	7
4	Local Settings Template	9
5	Shopzilla Settings	11
6	Local Settings Code	13
7	External Media & Templates	17
8	Website Default and Sub Templates	19
9	Default Template Filenames	21
10	Linked Media Subdirectories	23
11	Changelog	25
11.1	v1.0 2013-07-22	25
11.2	v0.9.4 2013-03-04	25
11.3	v0.9.3 2013-02-23	25
11.4	v0.9.2 2013-01-19	26
11.5	v0.9.1 2012-12-31	26
11.6	v0.9 2012-12-31	26

django-sciweb will manage and create websites on the fly for general e-commerce and affiliate marketing purposes

Contents:

Requirements

Please install the following packages using PIP or easy_install

- Django 1.4.1
- South >= 0.7.6
- Django-lettuce >= 0.1.27
- nosetools
- Simplejson
- importlib

Static Directories

Use the test file to check for the static directories. Test file is located at `apps/mainweb/tests/test_settings.py` - `TestSettings.test_static_directories`

- `PROJECT_DIR/static/` - The main static directory
- `PROJECT_DIR/static/bootstrap` The default bootstrap css/js files

Local Settings

Local settings should be created to:

- Set your database configuration
- Allow support for multiple-dev environments
- Separate configuration for development/staging/production branches
- Set admin on or off settings to open the admin pages
- Set the path to the custom templates to serve for specific sites

Local Settings Template

The git repository is provided with a template called `local_settings.template`

Shopzilla Settings

Be sure to look out for the following variables in your `local_settings` which control the search frequency and authentication keys for shopzilla queries.

- `SHOPZILLA_TOKEN` = Your api key
- `SHOPZILLA_PUB_TOKEN` = Your pub token from shopzilla
- `SHOPZILLA_SEARCH_FREQUENCY` = how often to search in minutes
- `SHOPZILLA_OUTPUT_FILE` = `'/home/user/Desktop/json.dumps'`

Local Settings Code

You can copy paste this code into your `local_settings.py`:

```
from django.conf.urls.defaults import *
# Uncomment the next two lines to enable the admin:

ENABLE_ADMIN = True
DEBUG = True
DEBUG_CRUD = True
LOG_ON = True
TEMPLATE_DEBUG = DEBUG

# same as application_url_includes url value
# hardcoded to redirect to admin - no client login implemented
LOGIN_REDIRECT_URL = '/webadmin/yay/hooray'

# debugtoolbar IP settings (optional)
INTERNAL_IPS = ('127.0.0.1', 'localhost',)

# set the logger to writet!

# set dev-installed apps
DEV_INSTALLED_APPS = (
    'appname',
    'lettuce.django',
    'debug_toolbar',
    'mainweb',
)

# add additional custom middleware classes
DEV_MIDDLEWARE_CLASSES = (
    'django.logging.middleware.LoggingMiddleware',
    'debug_toolbar.middleware.DebugToolbarMiddleware',
    'debug_logging.middleware.DebugLoggingMiddleware',
)

# enable or disable admins - uncomment and modify with your url
if ENABLE_ADMIN:
    application_url_includes = patterns('',
        #(r'^admin/(.*)', admin.site.root),
    )
else:
    application_url_includes = patterns('',)
```

```
# choose the view for the root URL /
mastersite_rooturl = patterns('',
    #url(r'^$', 'web.views.admin_redirect'),
    url(r'^robots.txt', 'mainweb.views.robots'),
    url(r'^$', 'mainweb.views.index', name="mainweb_index"),
    url(r'^(?P<linkname>\w+)/(?P<filtername>[\w -]+)', 'mainweb.views.index'),
    url(r'^(?P<linkname>\w+)', 'mainweb.views.index'),
)

# assuming mysql db - assuming django.db.backends.mysql
DATABASES = {
    'test': {
        'ENGINE': 'django.db.backends.',
        'NAME': 'testdatabase',
        'HOST': '',
        'PASSWORD': 'testpassword',
        'USER': 'testuser'
    },
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': 'dbdata.sqlite3',
        'HOST': '',
        'PASSWORD': '',
        'USER': '',
    }
}

DATABASE_ENGINE = DATABASES.get('default').get('ENGINE')
DATABASE_NAME = DATABASES.get('default').get('NAME')
DATABASE_USER = DATABASES.get('default').get('USER')
DATABASE_PASSWORD = DATABASES.get('default').get('PASSWORD')
DATABASE_HOST = DATABASES.get('default').get('HOST')
DATABASE_PORT = ''

# dynamic shopzilla query pages (search, compare, category view
SHOP_SEARCH = 'shopsearch'
SHOP_COMPARE = 'shopcompare'
SHOP_CATEGORY = 'shopcategory'
SHOPZILLA_TOKEN = 'your-shopzilla-token'
SHOPZILLA_PUB_TOKEN = 'your-pub-token'
# how often to search the remote api (in minutes)
SHOPZILLA_SEARCH_FREQUENCY = 15
SHOPZILLA_OUTPUT_FILE = '/home/user/Desktop/jsondumps'

# define the static page names that will be used in the URL
STATIC_PAGES = ['products', 'articles']
# define static pages that require an arg
STATIC_ARG_PAGES = ['p', 'search', 'a', SHOP_SEARCH, SHOP_COMPARE, SHOP_CATEGORY]

# optional if using django-debug-toolbar
DEBUG_TOOLBAR_PANELS = (
    #'debug_toolbar.panels.version.VersionDebugPanel',
    'debug_toolbar.panels.timer.TimerDebugPanel',
    'debug_toolbar.panels.settings_vars.SettingsVarsDebugPanel',
    'debug_toolbar.panels.headers.HeaderDebugPanel',
    'debug_toolbar.panels.request_vars.RequestVarsDebugPanel',
    #'debug_toolbar.panels.template.TemplateDebugPanel',
)
```

```
'debug_toolbar.panels.sql.SQLDebugPanel',
'debug_toolbar.panels.signals.SignalDebugPanel',
'debug_toolbar.panels.logger.LoggingPanel',
)

# Additional locations of static files
STATICFILES_DIRS = (
    # Put strings here, like "/home/html/static" or "C:/www/django/static".
    # Always use forward slashes, even on Windows.
    # Don't forget to use absolute paths, not relative paths.
    "%s/static" % PROJECT_ROOTDIR,
)

# List of finder classes that know how to find static files in
# various locations.
STATICFILES_FINDERS = (
    'django.contrib.staticfiles.finders.FileSystemFinder',
    'django.contrib.staticfiles.finders.AppDirectoriesFinder',
    # 'django.contrib.staticfiles.finders.DefaultStorageFinder',
)

# PATH to your templates without trailing slash
TEMPLATE_PATH = '/home/user/htmls'
```

External Media & Templates

Here we will configure our templates, css/javascript & images directories.

Includes Twitter Bootstrap for the default CSS & JS.

When creating a custom domain folder, each subdirectory of that should be a directory named as the site stripped of all prefixes.

By default, it will search for a folder called domains/ inside the TEMPLATE_PATH directory, with a subdirectory named after the website domain (site123.com) and finally the requested template filename from the WebsitePage() data. If the directory is not found, it will search return a default template.

Example Directory Structure example:

```
home/
  user/
    htmls/
      domains/
        sitename.com/
          index.html
          search.html
          .....
        domain-cool.com/
          index.html
          search.html
          .....
        hot-a55-orgy.com/
          index.html
          search.html
          .....
```

Website Default and Sub Templates

By default, there is a set of templates located inside the mainweb app at `*mainweb/templates/`

These templates are used if there is no subdirectory `domains` inside of `TEMPLATE_PATH`

Default Template Filenames

- index.html
- show_product.html
- show_article.html
- search.html
- products.html
- articles.html

Linked Media Subdirectories

Static files are now supported. `{% static '/bootstrap/css/bootstrap.css' %}` or `{% static 'priceswagger.com/css/style.css' %}` to use the default bootstrap.css file on PROJECTROOT/static or your custom templates directory E.g: /your/templates/directory/static/domains/

Note: 1.0 is now using staticfiles which makes use of settings.STATICFILES_DIRS

Changelog

11.1 v1.0 2013-07-22

- Version bump
- Django1.5 support
- Implemented staticfiles use for static media
- Some refactors
- Removed crudstuff
- Main App 'mainweb' admin html pages
- Implemented class views crud
- Main App 'mainweb' more pluggable now
- Removed local settings URL for crud pages, using default urls.

11.2 v0.9.4 2013-03-04

- Implemented shopzilla api
- Implemented compare search
- Implemented shopzilla taxonomy service
- Static arg pages modified (added some in as well)
- TODO update docs with standard shopzilla templates and vars

11.3 v0.9.3 2013-02-23

- Implemented shopzilla search api for shopsearch pagename as static-arg type
- Returning raw json data to context return
- Implemented base shop templates
- Implemented read template which will search for custom domain templates before defaults

11.4 v0.9.2 2013-01-19

- Implemented crudstuff admin app
- Template structure app-based
- Documented static media structure
- Crud dynamic forms with bindmodels
- Processing basic POST/form-save on 'add' action

11.5 v0.9.1 2012-12-31

- Added the templates directory with base templates (index, index-carousel, login)

11.6 v0.9 2012-12-31

- Route all incoming requests to mainweb.index method
- Created a ProcessPage class to handle incoming GET/POST
- Write unit tests around the models for mainweb
- Created a Website database table
- Created a WebsitePage database table