Flask-LDAP Documentation

Release 0.1.5

Dmitry Zhiltsov

December 23, 2013

Contents

1 Features **3**

2 Installation **5**

3 Setting it up **7**
3.1 Configuration options 7

4 API **9**

5 An example app **11**

6 Indices and tables **13**

Flask-LDAP is an extension to Flask that allows you to easily add LDAP based authentication to your website. It depends on Flask, python-ldap and Flask-PyMongo(optionally).

Contents:

CHAPTER 1

Features

- support of openLDAP and Active Directory
- friendly API
- perfect integration with Flask
- save profile into MongoDB (Flask-PyMongo required)

CHAPTER 2

Installation

Install the extension with one of the following commands:

```
pip install Flask-LDAP
```

or download the source code from here and run command:

```
python setup.py build
python setup.py install
```

CHAPTER 3

Setting it up

Here's an example of how Flask-LDAP is typically initialized and configured:

```
from flask import Flask
from flask.ext.ldap import LDAP, login_required
from flask.ext.pymongo import PyMongo
app = Flask(__name__)
app.debug = True

app.config['LDAP_HOST'] = 'ldap.example.com'
app.config['LDAP_DOMAIN'] = 'example.com'
app.config['LDAP_SEARCH_BASE'] = 'OU=Domain Users,DC=example,DC=com'
app.config['LDAP_LOGIN_VIEW'] = 'custom_login'

app.config['MONGO_DBNAME'] = 'simpledb'
mongo = PyMongo(app, config_prefix='MONGO')
ldap = LDAP(app, mongo)
app.secret_key = "welfhwdlhwdlfhwelfhwlehfwlehfelwehflwefwlehflwefhlwefhlewjfhwelfjhweflhweflhwel"
app.add_url_rule('/login', 'login', ldap.login, methods=['GET', 'POST'])
```

If you not use MongoDB, you can initialization extension without MongoDB support (user profile can't be stored):

```
ldap = LDAP(app)
```

3.1 Configuration options

Flask-LDAP is configured via the standard Flask config API. Here are the available configuration options:

- **LDAP_HOST** - the hostname or IP address ldap server (Default: "127.0.0.1")
- **LDAP_PORT** - the port where ldap server is listening (Default: 389)
- **LDAP_SCHEMA** - the schema that should be used to reach the ldap server (Default: "ldap")
- **LDAP_DOMAIN** - domain name (Default: "example.com")
- **LDAP_SEARCH_BASE** - setting the search base (Default: "OU=Users,DC=example,DC=com")
- **LDAP_REQUIRED_GROUP** - If specified, the authentication must succeed AND the user must be part of this security group. (Ex: "CN=security_group_name,OU=Groups,DC=example,DC=com")

- **LDAP_LOGIN_VIEW** - function name for auth login form (Default: “login”)
- **LDAP_LOGIN_TEMPLATE** - name of template for rendering auth form (Default: “login.html”)
- **LDAP_SUCCESS_REDIRECT** - function name for redirect after success login (Default: ‘index’)
- **LDAP_PROFILE_KEY** - name of the attribute in the user’s account from LDAP to save as ‘_id’ to MongoDB (Default: ‘sAMAccountName’)
- **LDAP_AVATAR_LOC** - path of user’s avatar from LDAP. If LDAP_PROFILE_KEY is defined then its value is used as name of image file, else value of userPrincipalName is used.

CHAPTER 4

API

CHAPTER 5

An example app

```
# coding=utf-8

from flask import Flask
from flask.ext.ldap import LDAP
app = Flask(__name__)
app.debug = True
app.config['LDAP_HOST'] = 'ldap.example.com'
app.config['LDAP_DOMAIN'] = 'example.com'
app.config['LDAP_AUTH_TEMPLATE'] = 'login.html'
app.config['LDAP_PROFILE_KEY'] = 'employeeID'
# app.config['LDAP_AUTH_VIEW'] = 'login'

ldap = LDAP(app)
app.secret_key = "welfhwdlhwdlfhwelfhwlehfwlehfelwehflwefwlehflwefhlwefhlewjfhwelfjhweflhweflhwel"
app.add_url_rule('/login', 'login', ldap.login, methods=['GET', 'POST'])

@app.route('/')
@ldap.login_required
def index():
        pass


# @app.route('/login', methods=['GET', 'POST'])
# def login():
#     pass

if __name__ == '__main__':
app.run(debug=True, host="0.0.0.0")
```

CHAPTER 6

Indices and tables

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