
Django test AJAX autocomplete Documentation

Release 0.2.1

pvergain@gmail.com

Jul 16, 2017

1	Autocomplete introduction	5
1.1	Definitions	5
1.1.1	Definition 1	5
1.1.2	Definition 2	5
1.2	Arguments	6
2	Autocomplete jQuery plugins tests	7
2.1	Specification	7
2.2	Test the different jQuery plugins	8
2.2.1	django-autocomplete-light test	8
2.2.1.1	install	9
2.2.1.1.1	update the settings.py module	9
2.2.1.2	tree	10
2.2.1.3	projects/views_django_autocomplete_light.py	16
2.2.1.4	projects/api/get_users/?q (projects/urls.py)	16
2.2.1.4.1	test	16
2.2.1.5	projects/forms_django_autocomplete_light.py	16
2.2.1.6	projects/admin.py	17
2.2.1.6.1	Using autocompletes in the admin : nice !	17
2.2.1.7	Using autocompletes outside the admin	17
2.2.1.7.1	projects/views_django_autocomplete_light.py	17
2.2.1.7.2	Django template	19
2.2.1.8	Test URL http://127.0.0.1:8004/projects/project/1/update_django_autocomplete	23
2.2.1.9	Passing options to select2	23
2.2.2	jQuery EasyAutocomplete test	24
2.2.2.1	Running the local Django web server	24
2.2.2.2	Testing the JSON API view	24
2.2.2.3	Testing the form, step1	24
2.2.2.3.1	projects/urls.py	26
2.2.2.3.2	The forms.py part	26
2.2.2.3.3	The HTML and JavaScript part	26
2.2.2.4	Step2 : initialize the easyAutocomplete placeholder with the value of champion	28
2.2.2.5	Step3 : hide the select champion form field	28
2.2.2.5.1	Before the update	28
2.2.2.5.2	After the update	29
2.2.3	jQuery UI autocomplete test	29

2.2.3.1	Running the local Django web server	30
2.2.3.2	The projects/forms.py module	31
2.2.3.3	The HTML and JavaScript part (the Django Template)	31
2.2.3.4	The projects/views.py module	33
2.2.3.5	The projects/urls.py module	33
2.2.3.6	The jQuery UI scrollbar module	34
3	Actions	35
3.1	Actions 2016	35
3.1.1	2016-10-25 installing gitsome	35
3.1.1.1	install	35
3.1.2	2016-10-25 testing jquery-ui autocomplete and django-autocomplete-light	37
3.1.2.1	Conclusion	37
3.1.3	2016-10-25 Building fake users with the Python faker module	37
3.1.3.1	Motivation	37
3.1.4	2016-10-24 Learning jQuery 101	37
3.1.5	2016-10-21 Receive 2 new excellent books about jQuery: jQuery In Action and jquery UI in Action	39
3.1.6	2016-10-21 testing the jquery EasyAutocomplete (EAC) options for an AJAX call	39
3.1.6.1	Renaming views	39
3.1.6.2	Add a great number of projects into the 'project' table	39
3.1.6.3	Write an API view/urls to select projects	40
3.1.6.3.1	The view	40
3.1.6.3.2	The URL	40
3.1.6.3.3	Tests with httpie	41
3.1.6.4	Update the Django template file in order to call this view	41
3.1.6.5	6 elements is not enough	41
3.1.6.6	Ajout placeholder	44
3.1.6.7	list['onSelectItemEvent']	44
3.1.6.7.1	HTML5	44
3.1.6.7.2	Javascript	44
3.1.6.8	list['showAnimation']	45
3.1.6.9	TODO : tester categories	45
3.1.7	2016-10-21 pushing documentation on readthedocs	45
3.1.7.1	Connect to readthedocs	46
3.1.7.2	Import a projet	46
3.1.7.3	Advanced parameters	46
3.1.8	2016-10-20 2016-10-21 First step: using the jquery EasyAutocomplete plugin with Django	46
3.1.8.1	Why ? EasyAutocomplete can easily become one of the best autocomplete plugins available for free	51
3.1.8.2	EasyAutocomplete files tree (Javascript, CSS, JSON, PHP files)	54
3.1.8.3	Javascript SCSS and CSS EasyAutocomplete files	56
3.1.8.3.1	EasyAutocomplete-master/src/sass/easy-autocomplete.scss	56
3.1.8.3.2	EasyAutocomplete-master/dist/easy-autocomplete.css	64
3.1.8.3.3	EasyAutocomplete-master/dist/jquery.easy-autocomplete.js	70
3.1.8.4	EasyAutocomplete simple example	102
3.1.8.4.1	EasyAutocomplete-master/demo/example_simple.html	102
3.1.8.5	EasyAutocomplete JSON example	103
3.1.8.5.1	EasyAutocomplete-master/demo/example_json.html	103
3.1.8.5.2	EasyAutocomplete-master/demo/resources/countries.json	104
3.1.8.6	EasyAutocomplete flags example	106
3.1.8.6.1	EasyAutocomplete-master/demo/example_flags.html	106
3.1.8.6.2	EasyAutocomplete-master/demo/resources/flags.css	108
3.1.8.6.3	EasyAutocomplete-master/demo/resources/flags.png	112

3.1.8.7	EasyAutocomplete categories example	113
3.1.8.7.1	EasyAutocomplete-master/demo/example_categories.html	113
3.1.8.7.2	EasyAutocomplete-master/demo/resources/categories.json	114
3.1.8.8	EasyAutocomplete static_link example	114
3.1.8.8.1	EasyAutocomplete-master/demo/example_static_link.html	114
3.1.8.8.2	EasyAutocomplete-master/demo/resources/site.json	116
3.1.8.9	EasyAutocomplete email example	116
3.1.8.9.1	EasyAutocomplete-master/demo/example_email.html	116
3.1.8.9.2	EasyAutocomplete-master/demo/resources/people.json	117
3.1.8.10	EasyAutocomplete Django integration	121
3.1.8.10.1	Include JS and CSS files from the distribution (static files)	121
3.1.8.11	Exemples of static files in the Django World	122
3.1.8.11.1	The Django contrib module	122
3.1.8.11.2	Django_By_Example_Code/Chapter 1/mysite/blog	123
3.1.8.11.3	Django_By_Example_Code/Chapter 8	124
3.1.8.11.4	Django_By_Example_Code/Chapter 13	126
3.1.8.12	Django settings example	129
3.1.8.13	Integration in our Django test project	129
3.1.8.14	Create a simple update_easy_simple.html	131
3.1.8.14.1	Le fichier projects/templates/projects/projet/update_easy_simple.html	131
3.1.8.14.2	First problem : the libraries are not found	132
3.1.8.14.3	collectstatic	133
3.1.8.14.4	New tree	133
3.1.8.14.5	Create an easyautocomplete directory	134
3.1.8.14.6	Update the Django template file	135
3.1.8.14.7	OK it works !	136
3.1.8.14.8	Add JSON files	136
3.1.8.15	Adding the AJAX call in the Django template	136
3.1.8.15.1	URL : http://127.0.0.1:8004/projects/champion_get_json/?term=a	136
3.1.8.15.2	pip install httpie (clihttp)	136
3.1.8.15.3	http http://127.0.0.1:8004/projects/champion_get_json/?term=a	138
3.1.8.15.4	http http://easyautocomplete.com/api/countrySearch.php?phrase=co	138
3.1.8.15.5	Avec countrySearch.php	140
3.1.8.15.6	Avec Python/Django OK the first step is DONE	140
3.1.8.16	This is the happy end of the first step	142
3.1.9	2016-10-19 improve the Django templates + unobstrusive Javascript (step1)	142
3.1.9.1	The Javascript code must be in the the body element	143
3.1.9.2	Modify names	144
3.1.9.3	Test champion_get_json : http://127.0.0.1:8004/projects/champion_get_json/?term=a	144
3.1.9.4	Rendering fields manually	144
3.1.9.5	Adding django-debug-toolbar	144
3.1.9.6	First step for a better look	145
3.1.10	2016-10-18 improve the jquery-ui autocomplete look and feel	145
3.1.10.1	Last look	146
3.1.10.2	jquery-ui autocomplete options already used	149
3.1.10.2.1	Remote option	149
3.1.10.3	Try https://jqueryui.com/autocomplete/#combobox	150
3.1.10.4	Try https://jqueryui.com/autocomplete/#maxheight	150
3.1.10.5	Try https://jqueryui.com/autocomplete/#categories	151
3.1.10.6	Try https://jqueryui.com/autocomplete/#remote-jsonp	152
3.1.11	2016-10-18 try a more simple approach with jquery-ui autocomplete widget	154
3.1.11.1	Installation	155
3.1.11.2	Create a projects application	156
3.1.11.3	Add the Django model Project	156

3.1.11.4	Add the project in the INSTALLED_APPS	157
3.1.11.5	makemigrations	160
3.1.11.6	sqlmigrate	161
3.1.11.7	migrate	161
3.1.11.8	Create the projects/forms.py file and add 2 forms	162
3.1.11.9	Create the projects/urls.py file and add the champion auto complete URL	162
3.1.11.10	Add the projects.urls file to the root urls.py	163
3.1.11.11	Add the ProjectUpdateView	164
3.1.11.12	Create the templates/projects/project/update.html django template	166
3.1.11.13	Add the jquery-ui javascript code	169
3.1.11.14	Add the admin interface	169
3.1.11.15	Create some projects with the admin interface	169
3.1.11.16	Reading Project record from the commande line	169
3.1.11.17	Add some users with the admin interface	172
3.1.11.18	Add some users with the command line interface	172
3.1.11.19	Try the URL : http://127.0.0.1:8000/projects/project/1/update	172
3.1.12	2016-10-18 create some Django Templates : base.html	173
3.1.12.1	The Django Template templates/base.html	173
3.1.12.2	The Django Template singers/templates/song/update.html	176
3.1.12.3	Add the django-extensions and ipython modules	177
3.1.12.3.1	New django commands	177
3.1.12.4	Get the first song	178
3.1.12.5	Try the URL : http://127.0.0.1:8000/singers/song/1/update	178
3.1.12.6	Compared to the django admin interface	178
3.1.12.7	Conclusion	180
3.1.13	2016-10-17 create a normal Django view	180
3.1.13.1	Le fichier singers/models.py	180
3.1.13.2	Le fichier singers/forms.py	182
3.1.13.3	Le fichier singers/views.py	183
3.1.13.4	Le fichier singers/urls.py	186
3.1.14	2016-10-17 create singers SQL tables	186
3.1.14.1	makemigrations singers	187
3.1.14.2	sqlmigrate singers 0001	187
3.1.14.3	migrate singers	189
3.1.14.4	show-migrations	189
3.1.14.5	createsuperuser	190
3.1.14.6	Connect to the admin interface	190
4	Django test AJAX autocomplete development	191
4.1	Github Singers's development	191
4.1.1	Create new project	191
4.1.2	Create singers project	191
4.1.3	Git remote	191
4.2	Singers's Python virtualenv	193
4.2.1	ajax_django35 Python 3.5 virtualenv	193
4.3	Django	194
4.3.1	Django packages	194
4.3.2	Django tips	194
4.4	Python modules	197
4.4.1	Python faker module	197
4.4.1.1	Description	197
4.4.1.2	install	197
4.4.1.3	Test	198

4.5	2016-10-21 Receive 2 new excellent books about jQuery: jQuery In Action and jquery UI in Action	200
4.5.1	If you were starting a new web app tomorrow, would you use jQuery ?	201
4.5.1.1	Aurelio De Rosa	201
4.5.1.2	TJ VanToll	202
4.5.2	jQuery in Action (third edition)	202
4.5.2.1	clone	202
4.5.3	jQuery UI in Action	202
4.5.4	jquery plugins	202
4.5.4.1	jquery plugins	202
4.5.4.1.1	jquery_ui_autocomplete_scroll plugin	202
4.5.4.1.2	jquery select2 plugin	202
5	Versions	209
5.1	Creating tags from the command line	209
5.2	Versions	209
5.2.1	Version 0.2.0 (2016-10-27)	209
5.2.1.1	Added	210
5.2.1.2	Changed	210
5.2.1.3	Miscellaneous	210
5.2.1.4	Fixed	210
5.2.1.5	Test	210
5.2.2	Version 0.1.0 (2016-10-21)	210
5.2.2.1	Added	210
5.2.2.2	Changed	212
5.2.2.3	Miscellaneous	212
5.2.2.4	Fixed	212
5.2.2.5	Test	212
6	Glossary	213

See also:

- <https://django-test-autocomplete.readthedocs.io/en/latest/>
- <https://github.com/pvergain/django-test-autocomplete.git>
- [https://en.wikipedia.org/wiki/Ajax_\(programming\)](https://en.wikipedia.org/wiki/Ajax_(programming))
- <https://en.wikipedia.org/wiki/JSON>

Dango test AJAX autocomplete

Projet Dango test AJAX autocomplete

Release 0.2.1

Date Jul 16, 2017

Authors pvergain

- [genindex](#)
- [search](#)
- [Glossary](#)

Thanks <https://readthedocs.org/>

- <https://readthedocs.org/projects/django-test-autocomplete/>

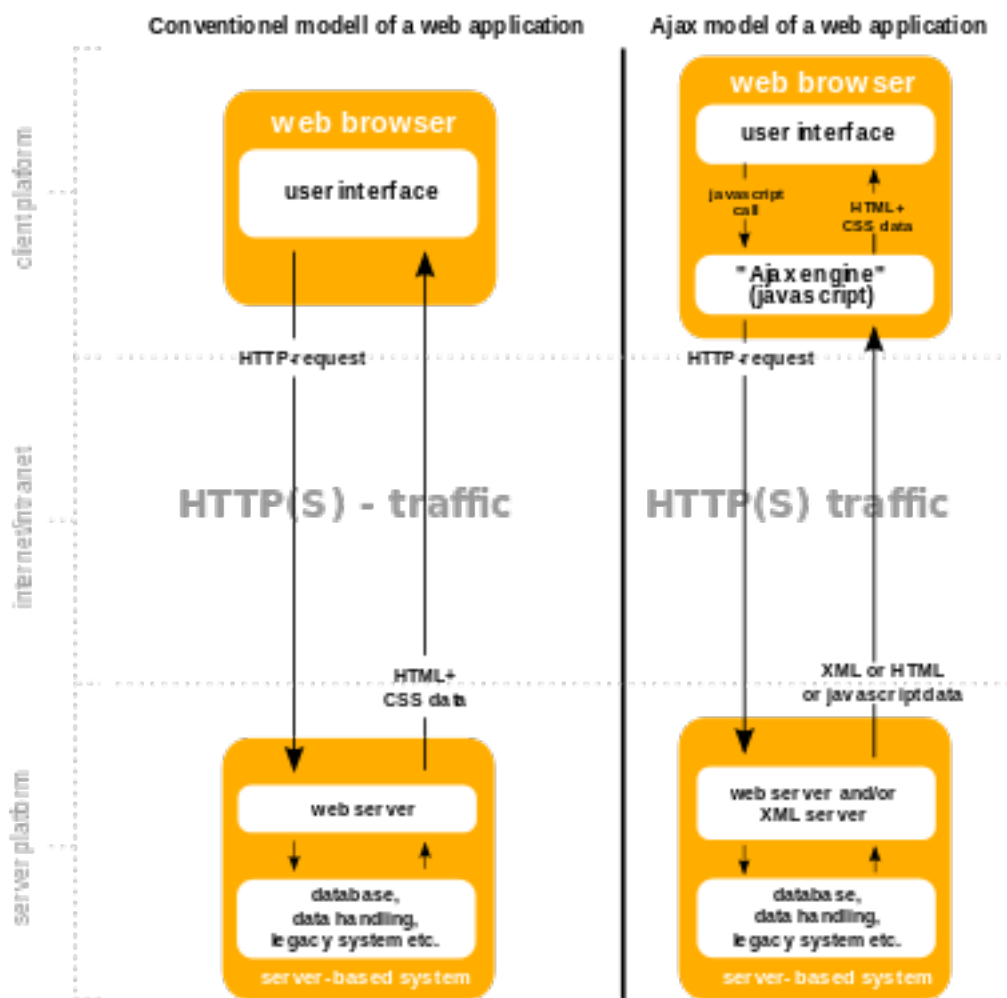


Fig. 1: *AJAX* source: [https://en.wikipedia.org/wiki/Ajax_\(programming\)](https://en.wikipedia.org/wiki/Ajax_(programming))



Fig. 2: *JSON* logo (<https://en.wikipedia.org/wiki/JSON>)



Fig. 3: httpie logo (<https://httpie.org/>)

Autocomplete introduction

Contents

- *Autocomplete introduction*
 - *Definitions*
 - * *Definition 1*
 - * *Definition 2*
 - *Arguments*

Definitions

Definition 1

See also:

- <http://blog.appliedinformaticsinc.com/autocomplete-input-field-in-django-template-with-jquery-ui/>

Auto-complete is one of the most important aspects of modern web interface.

Auto-complete feature is used to provide auto suggestion for users while entering input. We can create auto-complete using an AJAX call to make a list and display the list using javascript. Creating auto-complete with jquery and jquery-ui is the most efficient way of creating it.

Definition 2

See also:

- <http://api.jqueryui.com/autocomplete/>

- <http://api.jqueryui.com/autocomplete/#option-source>

You can pull data in from a local or remote source: Local is good for small data sets, e.g., an address book with 50 entries; remote is necessary for big data sets, such as a database with hundreds or millions of entries to select from. To find out more about customizing the data source, see the documentation for the [source option](#).

Arguments

When using `<select>` you need to retrieve *all* values from the database before displaying the form. This is a potentially expensive operation on the server and delays the time when the user see the form.

Autocomplete jQuery plugins tests

Contents

- *Autocomplete jQuery plugins tests*
 - *Specification*
 - *Test the different jQuery plugins*

Specification

Write a form for selecting ‘champion’ for a project.

```
1  #!/usr/bin/python
2  # -*- coding: utf8 -*-
3  """The project's models.
4
5  """
6  from __future__ import unicode_literals
7  from django.utils.encoding import python_2_unicode_compatible
8
9  from django.db import models
10 from django.core.urlresolvers import reverse
11
12 # https://docs.djangoproject.com/en/dev/ref/contrib/auth/#user-model
13 from django.contrib.auth.models import User
14
15
16
17 @python_2_unicode_compatible
18 class Project(models.Model):
19     """A project with a title and a champion which is the foreign key to the
```

```
20     auth user.
21
22     Documentation
23     =====
24
25     - http://guiqinqian.blogspot.fr/2012/01/using-jquery-auto-complete-in-django.html
26
27     """
28     title = models.CharField(max_length=200)
29     champion = models.ForeignKey(User)
30
31     def __str__(self):
32         return "{} {}".format(self.title, self.champion)
33
34     def get_absolute_url(self):
35         """
36         https://docs.djangoproject.com/en/dev/ref/class-based-views/generic-editing/
37         """
38         return reverse('projects:project_update',
39                        kwargs={'pk': self.pk})
```

Test the different jQuery plugins

django-autocomplete-light test

See also:

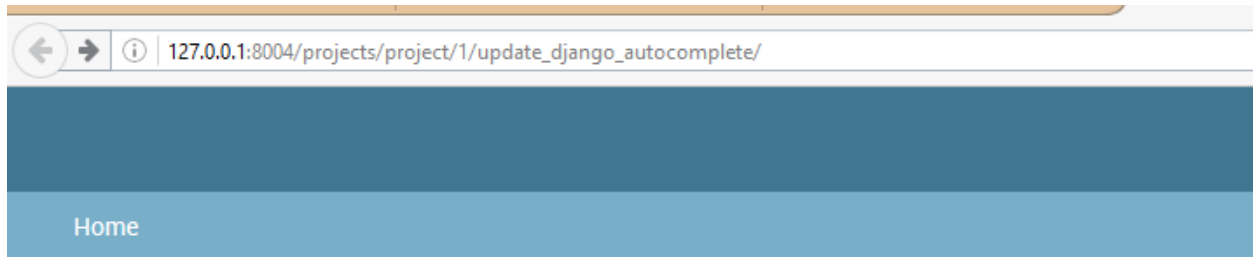
- <https://github.com/yourlabs/django-autocomplete-light.git>
- <https://github.com/select2/select2.git>
- <https://github.com/django/django/blob/master/django/contrib/admin/templates/admin/base.html>

Contents

- *django-autocomplete-light test*
 - *install*
 - * *update the settings.py module*
 - *tree*
 - *projects/views_django_autocomplete_light.py*
 - *projects/api/get_users/?q (projects/urls.py)*
 - * *test*
 - *projects/forms_django_autocomplete_light.py*
 - *projects/admin.py*
 - * *Using autocompletes in the admin : nice !*
 - *Using autocompletes outside the admin*
 - * *projects/views_django_autocomplete_light.py*

* *Django template*

- *Test URL `http://127.0.0.1:8004/projects/project/1/update_django_autocomplete`*
- *Passing options to `select2`*



Test Django autocomplete

Update of the project '(title:project test 2 champion:Jalmari_Utriainen)

Title:

Champion:

- as
- Martin_Porras
- Mats_Asplund
- Osvaldo_Piras
- Santiago_Abascal
- Silas_Jeppesen
- Tazio_Basile
- Tomas_Holmkvist

install

```
pip install django-autocomplete-light
```

update the settings.py module

Add the 2 following applications:

- dal
- dal-select2

```
# Applications definition
INSTALLED_APPS = [
    'dal',
    'dal_select2',
```

tree

```
C:\PROJECTS_ID3\DJANGO-TEST-AUTOCOMPLETE\ANNEXES\DJANGO-AUTOCOMPLETE-LIGHT
|
| .gitignore
| .gitmodules
| .travis.yml
| AUTHORS
| CHANGELOG
| CHANGES.txt
| LICENSE
| MANIFEST.in
| pytest.ini
| README
| README.rst
| setup.py
| tox.ini
|
+---.openshift
| |   README.md
| |
| | +---action_hooks
| | |   deploy
| | |   README.md
| | |
| | \---cron
| | |   README.cron
| | |
| | \---hourly
| | |   reset_database
| |
+---docs
| |   api.rst
| |   conf.py
| |   creation.rst
| |   genericm2m.rst
| |   gfk.rst
| |   gm2m.rst
| |   index.rst
| |   install.rst
| |   Makefile
| |   requirements.txt
| |   tagging.rst
| |   taggit.rst
| |   tutorial.rst
| |
+---img
| |   all.png
| |   autocomplete.png
| |   created_option.png
| |   create_option.png
| |   mine.png
```

```

|         view.png
|
| \---_ext
|        .djangodocs.py
|
+---src
|     __init__.py
|
+---dal
|     |     autocomplete.py
|     |     forms.py
|     |     forward.py
|     |     views.py
|     |     widgets.py
|     |     __init__.py
|     |
|     +---static
|     |     +---admin
|     |     |     \---js
|     |     |         jquery.init.js
|     |     |
|     |     \---autocomplete_light
|     |         autocomplete.init.js
|     |         forward.js
|     |         jquery.init.js
|     |
|     \---test
|         case.py
|         stories.py
|         utils.py
|         __init__.py
|
+---dal_contenttypes
|     fields.py
|     __init__.py
|
+---dal_genericm2m
|     fields.py
|     __init__.py
|
+---dal_genericm2m_queryset_sequence
|     fields.py
|     __init__.py
|
+---dal_gm2m
|     fields.py
|     __init__.py
|
+---dal_gm2m_queryset_sequence
|     fields.py
|     __init__.py
|
+---dal_queryset_sequence
|     |     fields.py
|     |     views.py
|     |     widgets.py
|     |     __init__.py
|

```

```
| | \---tests
| |     test_views.py
| |
| +---dal_select2
| | | apps.py
| | | checks.py
| | | fields.py
| | | models.py
| | | test.py
| | | views.py
| | | widgets.py
| | | __init__.py
| |
| | \---static
| | | \---autocomplete_light
| | | | select2.css
| | | | select2.js
| | |
| | | \---vendor
| | | | \---select2
| +---dal_select2_queryset_sequence
| | views.py
| | widgets.py
| | __init__.py
|
| +---dal_select2_tagging
| | widgets.py
| | __init__.py
|
| \---dal_select2_taggit
| | widgets.py
| | __init__.py
|
| \---test_project
| | .coveragerc
| | db.sqlite3
| | manage.py
| | requirements.txt
| | reset.json
| | urls.py
| | views.py
| | wsgi.py
|
| +---linked_data
| | | admin.py
| | | apps.py
| | | forms.py
| | | models.py
| | | test_forms.py
| | | test_functional.py
| | | urls.py
| | | __init__.py
| |
| +---migrations
| | | 0001_initial.py
| | | __init__.py
| |
| \---static
```

```
|         linked_data.js
|
+---rename_forward
| |     admin.py
| |     apps.py
| |     forms.py
| |     models.py
| |     test_forms.py
| |     test_functional.py
| |     urls.py
| |     __init__.py
|
+---migrations
| |     0001_initial.py
| |     __init__.py
|
\---static
|     linked_data.js
|
+---secure_data
| |     admin.py
| |     apps.py
| |     forms.py
| |     models.py
| |     test_functional.py
| |     urls.py
| |     views.py
| |     __init__.py
|
\---migrations
|     0001_initial.py
|     __init__.py
|
+---select2_foreign_key
| |     admin.py
| |     apps.py
| |     forms.py
| |     models.py
| |     test_functional.py
| |     urls.py
| |     __init__.py
|
\---migrations
|     0001_initial.py
|     __init__.py
|
+---select2_generic_foreign_key
| |     admin.py
| |     apps.py
| |     forms.py
| |     models.py
| |     test_forms.py
| |     test_functional.py
| |     urls.py
| |     views.py
| |     __init__.py
|
\---migrations
```

```
|         0001_initial.py
|         __init__.py
|
+---select2_generic_m2m
| |     admin.py
| |     apps.py
| |     forms.py
| |     models.py
| |     test_forms.py
| |     test_functional.py
| |     urls.py
| |     views.py
| |     __init__.py
| |
| | \---migrations
| |     0001_initial.py
| |     __init__.py
| |
+---select2_gm2m
| |     admin.py
| |     apps.py
| |     forms.py
| |     models.py
| |     test_forms.py
| |     test_functional.py
| |     urls.py
| |     views.py
| |     __init__.py
| |
| | \---migrations
| |     0001_initial.py
| |     __init__.py
| |
+---select2_list
| |     admin.py
| |     forms.py
| |     models.py
| |     test_fields.py
| |     test_functional.py
| |     test_views.py
| |     urls.py
| |     views.py
| |     __init__.py
| |
| | \---migrations
| |     0001_initial.py
| |     __init__.py
| |
+---select2_many_to_many
| |     admin.py
| |     apps.py
| |     forms.py
| |     models.py
| |     test_functional.py
| |     urls.py
| |     __init__.py
| |
| | \---migrations
```

```

|         0001_initial.py
|         __init__.py
|
+---select2_one_to_one
| |     admin.py
| |     apps.py
| |     forms.py
| |     models.py
| |     test_functional.py
| |     urls.py
| |     __init__.py
| |
| |     \---migrations
| |         0001_initial.py
| |         __init__.py
| |
+---select2_outside_admin
| |     urls.py
| |     views.py
| |     __init__.py
| |
| |     \---templates
| |         select2_outside_admin.html
| |
+---select2_tagging
| |     admin.py
| |     forms.py
| |     models.py
| |     test_forms.py
| |     test_functional.py
| |     urls.py
| |     __init__.py
| |
| |     \---migrations
| |         0001_initial.py
| |         0002_testmodel_test.py
| |         __init__.py
| |
+---select2_taggit
| |     admin.py
| |     forms.py
| |     models.py
| |     test_forms.py
| |     test_functional.py
| |     urls.py
| |     __init__.py
| |
| |     \---migrations
| |         0001_initial.py
| |         __init__.py
| |
+---settings
|     base.py
|     __init__.py
|
+---templates
| |     base.html
| |

```

```
| \---admin
|         login.html
|
| \---tests
|         admin.py
|         models.py
|         test_widgets.py
|         __init__.py
```

projects/views_django_autocomplete_light.py

```
from django.contrib.auth.models import User

from django.db.models import Q
from dal import autocomplete

class ApiUserDjangoAutocompleteLight(autocomplete.Select2QuerySetView):
    """https://django-autocomplete-light.readthedocs.io/en/master/tutorial.html"""
    def get_queryset(self):
        # Don't forget to filter out results depending on the visitor !
        users = User.objects.all()

        if self.q:
            users = User.objects.filter(Q(username__icontains=self.q)
                                         | Q(email__icontains=self.q)).order_by(
→ 'username')

        return users
```

projects/api/get_users/?q (projects/urls.py)

```
url(r'^api/get_users/$',
    ApiUserDjangoAutocompleteLight.as_view(),
    name='api_get_users'),
```

test

```
{"pagination": {"more": true}, "results": [{"text": "Aaron_De Angelis", "id": 119},
{"text": "Abdul_Parker", "id": 177}, {"text": "Adolf_Wahlberg", "id": 338},
{"text": "Adrian_Jenkins", "id": 184}, {"text": "Aim\u00e9_Roussel", "id": 42},
{"text": "Alejandro_Hoyos", "id": 240}, {"text": "Alexander_Dahlberg", "id": 335},
{"text": "Alexandria_Weiss", "id": 29}, {"text": "Alicia_Pareja", "id": 149},
{"text": "Alighieri_Coppola", "id": 216}]}
```

projects/forms_django_autocomplete_light.py

```
from django import forms
```



```
from .models import Project

from dal import autocomplete

class ProjectFormDjangoAutocomplete(forms.ModelForm):
    """https://django-autocomplete-light.readthedocs.io/en/master/tutorial.html"""
    class Meta:
        model = Project
        fields = ('__all__',)
        widgets = {
            'champion': autocomplete.ModelSelect2(url='projects:api_get_users')
        }
```

projects/admin.py

```
1  #!/usr/bin/python
2  # -*- coding: UTF-8 -*-
3  """Project Administration.
4
5  """
6
7  from django.contrib import admin
8
9  from .models import Project
10
11 from .forms_django_autocomplete_light import ProjectFormDjangoAutocomplete
12
13 @admin.register(Project)
14 class ProjectAdmin(admin.ModelAdmin):
15     """Project administration
16
17     Documentation
18     =====
19
20     - https://docs.djangoproject.com/en/dev/ref/contrib/admin/#modeladmin-objects
21
22     """
23     form = ProjectFormDjangoAutocomplete
24     list_display = ('title', 'champion')
25     search_fields = ('title', 'champion')
26     list_filter = ('title', 'champion')
```



Using autocompletes in the admin : nice !

Using autocompletes outside the admin

projects/views_django_autocomplete_light.py

```
from django.contrib.auth.models import User

from django.db.models import Q
from django.views.generic.edit import UpdateView
```

  127.0.0.1:8004/admin/projects/project/120/change/





Django administration

Home › Projects › Projects › python project:18 nigel

Change project

Title:python project:18

Champion:

nigel   

ag

Dagny_Danielsen

Luis_Paniagua

Omar_Battaglia

Santiago_Abascal

Yago_Neri

Delete

```

from dal import autocomplete

from .models import Project

from .forms_django_autocomplete_light import ProjectFormDjangoAutocomplete

# Get an instance of a logger
logger = logging.getLogger(__name__)

class ApiUserDjangoAutocompleteLight(autocomplete.Select2QuerySetView):
    """https://django-autocomplete-light.readthedocs.io/en/master/tutorial.html"""
    def get_queryset(self):
        # Don't forget to filter out results depending on the visitor !
        users = User.objects.all()
        if self.q:
            users = User.objects.filter(Q(username__icontains=self.q)
                                       | Q(email__icontains=self.q)).order_by(
↳ 'username')

        return users

class ProjectDjangoAutoCompleteUpdateView(UpdateView):
    """Update the view with the jQuery UI Autocomplete plugin.

    Documentation:

    - http://ccbv.co.uk/projects/Django/1.10/django.views.generic.edit/UpdateView/

    """
    model = Project
    form_class = ProjectFormDjangoAutocomplete
    context_object_name = 'project'
    template_name = 'projects/project/update_django_autocomplete_light.html'

    def get_object(self, queryset=None):
        """Pour mémoriser self.demande_article"""
        self.object = super(ProjectDjangoAutoCompleteUpdateView, self).get_
↳ object(queryset)
        return self.object

    def post(self, request, *args, **kwargs):
        logger.warning("Hello from ProjectDjangoAutoCompleteUpdateView !")
        return super(ProjectDjangoAutoCompleteUpdateView, self).post(request, *args,
↳ **kwargs)

```

Django template

See also:

- <https://github.com/django/django/blob/master/django/contrib/admin/templates/admin/base.html>

```

{# inherit from the django admin base.html file #}
{#####}
{% extends "admin/base_site.html" %}

```

This screenshot shows a GitHub repository for 'django-autocomplete-light'. The left sidebar displays a file tree with the following structure:

- test_project
 - linked_data
 - rename_forward
 - secure_data
 - select2_foreign_key
 - select2_generic_foreign_key
 - select2_generic_m2m
 - select2_gm2m
 - select2_list
 - select2_many_to_many
 - select2_one_to_one
 - select2_outside_admin
 - templates
 - select2_outside_admin.html

The main content area shows the file 'select2_outside_admin.html' with the following details:

- Branch: master
- File path: django-autocomplete-light / test_project / select2_outside_admin / templates / select2_outside_admin.html
- Commit message: jpic Use {% static %} for documentation
- Contributors: 1 contributor
- Stats: 20 lines (16 sloc) | 399 Bytes
- Code content (lines 1-19):

```
1 {% extends 'base.html' %}
2 {% # Don't forget that one ! #}
3 {% load static %}
4
5 {% block content %}
6 <div>
7     <form action="" method="post">
8         {% csrf_token %}
9         {{ form.as_p }}
10        <input type="submit" />
11    </form>
12 </div>
13 {% endblock %}
14
15 {% block footer %}
16 <script type="text/javascript" src="{% static 'admin/js/vendor/jquery/jquery.js' %}"></script>
17
18 {{ form.media }}
19 {% endblock %}
```

This screenshot shows a GitHub repository for 'django-autocomplete-light'. The left sidebar displays a file tree with the following structure:

- yourlabs / django-autocomplete-light
 - master
 - .openshift
 - docs
 - src
 - test_project
 - linked_data
 - rename_forward
 - secure_data
 - select2_foreign_key
 - select2_generic_foreign_key
 - select2_generic_m2m
 - select2_gm2m
 - select2_list
 - select2_many_to_many
 - select2_one_to_one
 - select2_outside_admin
 - templates
 - select2_outside_admin.html

The main content area shows the file 'base.html' with the following details:

- Branch: master
- File path: django-autocomplete-light / test_project / templates / base.html
- Commit message: jpic Added example outside the admin
- Contributors: 2 contributors
- Stats: 55 lines (52 sloc) | 1.87 KB
- Code content (lines 1-21):

```
1 {% extends "admin/base_site.html" %}
2 {% block branding %}
3 <h1>django-autocomplete-light demo: welcome</h1>
4 {% endblock %}
5
6 {% block breadcrumbs %}
7 <div class="breadcrumbs">
8 <a href="/">Home</a>
9 {% block title %}{% endblock %}
10 </div>
11 {% endblock %}
12
13 {% block content %}
14 <div style="display: inline-block; width: 49%; vertical-align: top;">
15 <h2>Welcome to django-autocomplete-light's demo project !</h2>
16 <p>
17 Here are a number of example apps which you can test manually, or let
18 robots do the work and run ./manage.py test.
19 </p>
20 <ul>
21 </ul>
```

The screenshot shows the Django admin template file `base.html` on GitHub. The left sidebar displays the file tree structure, with `base.html` selected under `templates/admin`. The main content area shows the HTML code of the template, which includes Django template tags for rendering messages, breadcrumbs, and the main content area.

```

45         {% endif %}
46         <a href="{% url 'admin:logout' %}">{% trans 'Log out' %}</a>
47     {% endblock %}
48 </div>
49     {% endif %}
50     {% endblock %}
51     {% block nav-global %}{% endblock %}
52 </div>
53 <!-- END Header -->
54 {% block breadcrumbs %}
55 <div class="breadcrumbs">
56 <a href="{% url 'admin:index' %}">{% trans 'Home' %}</a>
57 {% if title %} &rsaquo; {{ title }}{% endif %}
58 </div>
59 {% endblock %}
60 {% endif %}
61
62 {% block messages %}
63     {% if messages %}
64     <ul class="messagelist">{% for message in messages %}
65     <li{% if message.tags %} class="{{ message.tags }}"{% endif %}>{{ message|capfirst }}</li>
66     {% endfor %}</ul>
67     {% endif %}
68 {% endblock messages %}
69
70 <!-- Content -->
71 <div id="content" class="{% block coltype %}colM{% endblock %}">
72     {% block pretitle %}{% endblock %}
73     {% block content_title %}{% if title %}<h1>{{ title }}</h1>{% endif %}{% endblock %}
74     {% block content %}
75     {% block object-tools %}{% endblock %}
76     {{ content }}
77     {% endblock %}
78     {% block sidebar %}{% endblock %}
79     <br class="clear" />
80 </div>
81 <!-- END Content -->
82
83 {% block footer %}<div id="footer"></div>{% endblock %}
84 </div>
85 <!-- END Container -->
86
87 </body>
88 </html>

```

The screenshot shows a file explorer window displaying the contents of the `staticfiles` directory. The file list includes `jquery.js`, `jquery.min.js`, and `LICENSE-JQUERY.txt`.

Nom	Modifié le	Type	Taille
jquery.js	20/10/2016 15:58	Fichier de JavaScript	253 Ko
jquery.min.js	20/10/2016 15:58	Fichier de JavaScript	84 Ko
LICENSE-JQUERY.txt	20/10/2016 15:58	Fichier TXT	2 Ko

```

{% load static %}
{% load staticfiles %}

{% block branding %}
    <title>Django autocomplete example</title>
{% endblock %}

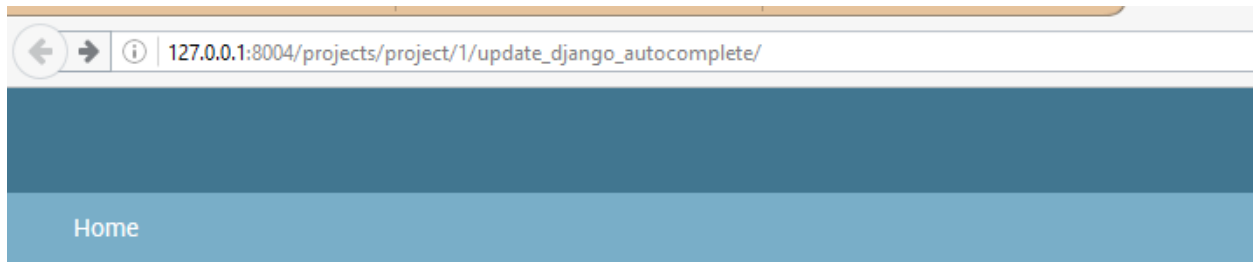
{% block content %}
<!-- STRUCTURE -> HTML5 elements -->
    <h1>Test Django autocomplete</h1>
    <h1>Update of the project '(title:{{ project.title }} champion:{{ project.
↪champion.username }})'</h1>
    <p></p>
    <p></p>
    {# https://docs.djangoproject.com/en/dev/topics/forms/ #}
    <form id="id_form_project" action="{% url 'projects:project_update' project.id %}"
↪" method="post">
        {% csrf_token %}
        <div class="forms">
            {{ form.id }}
            {{ form.non_field_errors }}
            {# Include the hidden fields #}
            {% for hidden in form.hidden_fields %}
                {# here we will have the champion filed (which is hidden) #}
                {{ hidden }}
            {% endfor %}
            <table id="id_table" class="table table-hover table-bordered table-
↪condensed">
                <tbody>
                    <tr>
                        <td class="text-right">Title:</td>
                        <td>{{ form.title }}</td>
                    </tr>
                    <tr>
                        <td class="text-right">Champion:</td>
                        <td>{{ form.champion }}</td>
                    </tr>
                </tbody>
            </table>
        </div>
        <input type="submit" name="btn_update" value="Update" class="btn btn-success_
↪btn-block" />
    </form>
<!-- end STRUCTURE-->

{% block footer %}
{# https://github.com/yourlabs/django-autocomplete-light/blob/master/test_project/
↪select2_outside_admin/templates/select2_outside_admin.html #}
<script type="text/javascript" src="{% static 'admin/js/vendor/jquery/jquery.js' %}">
↪</script>

    {{ form.media }}
{% endblock footer %}

```

Test URL http://127.0.0.1:8004/projects/project/1/update_django_autocomplete/



Test Django autocomplete

Update of the project '(title:project test 2 champion:Jalmari_Utriainen)

Title:

Champion:

Martin_Porras
 Mats_Asplund
 Osvaldo_Piras
 Santiago_Abascal
 Silas_Jeppesen
 Tazio_Basile
 Tomas_Holmkvist

Passing options to select2

Select2 supports a bunch of options. These options may be set in data-* attributes.

For example:

```
# Instantiate a widget with a bunch of options for select2:
autocomplete.ModelSelect2(
    url='select2_fk',
    attrs={
        # Set some placeholder
        'data-placeholder': 'Autocomplete ...',
        # Only trigger autocompletion after 3 characters have been typed
        'data-minimum-input-length': 3,
    },
)
```

jQuery EasyAutocomplete test

Contents

- *jQuery EasyAutocomplete test*
 - *Running the local Django web server*
 - *Testing the JSON API view*
 - *Testing the form, step1*
 - * *projects/urls.py*
 - * *The forms.py part*
 - * *The HTML and JavaScript part*
 - *Step2 : initialize the easyAutocomplete placeholder with the value of champion*
 - *Step3 : hide the select champion form field*
 - * *Before the update*
 - * *After the update*

Running the local Django web server

```
System check identified no issues (0 silenced).
October 24, 2016 - 10:41:29
Django version 1.10.2, using settings 'projet_ajax.settings'
Starting development server at http://127.0.0.1:8004/
Quit the server with CTRL-BREAK.
```

Testing the JSON API view

```
http://127.0.0.1:8004/projects/api_get_champions/?term=a
```

```
[{"id": 6, "value": "aaaa", "label": "aaaa"},
{"id": 1, "value": "admin", "label": "admin"},
{"id": 8, "value": "albert", "label": "albert"},
{"id": 2, "value": "champion_0001", "label": "champion_0001"},
{"id": 7, "value": "john", "label": "john"},
{"id": 10, "value": "nigel", "label": "nigel"},
{"id": 9, "value": "zoya", "label": "zoya"}]
```

Testing the form, step1

```
http://127.0.0.1:8004/projects/project/1/updateeasy
```

← ⓘ | 127.0.0.1:8004/projects/project/1/updateeasy/

Test jquery EasyAutocomplete

Update of the project '(title:project test champion:john)

Title:

Champion:

- admin
- champion_0001
- user_0002
- user_0003
- user_0004
- aaaa
- john
- albert
- zoya**
- nigel

projects/urls.py

```
url(r'^project/(?P<pk>\d+)/updateeasy/$',
    ProjectUpdateViewEasyAutoComplete.as_view(),
    name='project_update_easy'),
```

The forms.py part

```
#!/usr/bin/python
# -*- coding: utf8 -*-
"""The project's forms.

"""

from django import forms

from .models import Project

class ProjectChampionForm(forms.ModelForm):
    """The champion project form"""
    champions_choice_list = forms.CharField(max_length=100,
                                           help_text='type username or email')

    class Meta:
        model = Project
        fields = ('title',
                  'champions_choice_list', 'champion',)

    def __init__(self, *args, **kwargs):
        super(ProjectChampionForm, self).__init__(*args, **kwargs)
        self.fields['champions_choice_list'].label = "Update the champion"
        self.fields['champion'].widget = forms.HiddenInput()
```

The HTML and JavaScript part

```
<body>

    <!-- STRUCTURE -> HTML5 elements -->
    <h1>Test jquery EasyAutocomplete</h1>
    <h1>Update of the project '{title:{{ project.title }} champion:{{ project.
↪champion }})' </h1>
    <p></p>
    <p></p>
    {# https://docs.djangoproject.com/en/dev/topics/forms/ #}
    <form id="id_form_project_update_easy" action="{% url 'projects:project_
↪update' project.id %}" method="post">
        {% csrf_token %}
        <div class="forms">
            {{ form.id }}
            {{ form.non_field_errors }}
            {# Include the hidden fields #}
            {% for hidden in form.hidden_fields %}
                {# here we will have the champion filed (which is hidden) #}
```

```

        {{ hidden }}
    {% endfor %}
    <table id="id_table" class="table table-hover table-bordered_
↪table-condensed">
        <tbody>
            <tr>
                <td class="text-right">Title:</td>
                <td>{{ form.title }}</td>
            </tr>
            <tr>
                <td class="text-right">Champion:</td>
                <td> {{ form.champions_choice_list }} </td>
            </tr>
        </tbody>
    </table>
</div>
<input type="submit" name="btn_update" value="Update" class="btn btn-
↪success btn-block" />
</form>
<!-- end STRUCTURE-->

<!--BEHAVIOR -> Javascript scripts-->
<!-- Using jQuery with a CDN -->
<script src="//code.jquery.com/jquery-1.11.2.js"></script>
<script src="{% static 'easyautocomplete/js/jquery.easy-autocomplete.js'
↪%}" type="text/javascript"></script>

<script>
    var options_easy_autocomplete_champions = {
        {# we have to build this URL: http://127.0.0.1:8004/projects/api_
↪get_champions/?term=a #}
        url: function(term) {
            return "{% url 'projects:api_get_champions' %}" + "?term=" +
↪term;
        },
        getValue: "value",
        list: {
            maxNumberOfElements: 200,
            onSelectItemEvent: function() {
                var champion_id = $("#id_champions_choice_list").
↪getSelectedItemData().id;

                {# put the id in the champion field #}
                $("#id_champion")
                    .val(champion_id)
                    .trigger("change");
            },
            match: {
                enabled: true
            },
        },
        placeholder: "Choose your champion"
    };

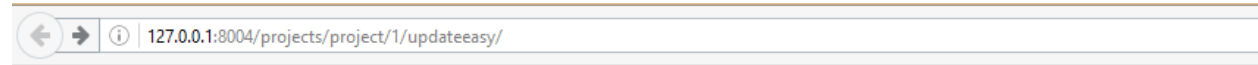
    {# build the autocomplete list for the champions_choice_list #}
    $("#id_champions_choice_list").easyAutocomplete(options_easy_
↪autocomplete_champions);

```

```
        </script>
        <!-- end BEHAVIOR -->

    </body>
```

Step2 : initialize the easyAutocomplete placeholder with the value of champion



Test jquery EasyAutocomplete

Update of the project '(title:project test 2 champion:albert)

Title:

Champion:

Champion value:

Add these jQuery lines:

```
{# build the autocomplete list for the champions_choice_list #}
$("#id_champions_choice_list").easyAutocomplete(options_easy_autocomplete_champions);
{# Get the value of the former champion (from database) #}
{# Thanks http://stackoverflow.com/questions/1643227/get-selected-text-from-a-drop-
↳down-list-select-box-using-jquery #}
var champion_name = $("#id_champion option:selected").text();
{# replace the placeholder by the value coming from database #}
$("#id_champions_choice_list").attr('placeholder', champion_name);
```

Step3 : hide the select champion form field

Before the update

in the HTML file:

```
<td class="text-right">Champion value:</td>
<td> <select id="id_champion" name="champion" required>
<option value="">-----</option>
<option value="1">admin</option>
<option value="2">champion_0001</option>
<option value="3">user_0002</option>
<option value="4">user_0003</option>
<option value="5">user_0004</option>
<option value="6">aaaa</option>
<option value="7">john</option>
<option value="8">albert</option>
<option value="9" selected="selected">zoya</option>
```

```
<option value="10">nigel</option>
</select>
```

in the projects/forms.py file:

```
def __init__(self, *args, **kwargs):
    super(ProjectChampionForm, self).__init__(*args, **kwargs)
    self.fields['champions_choice_list'].label = "Update the champion"
    # self.fields['champion'].widget = forms.HiddenInput()
```

After the update

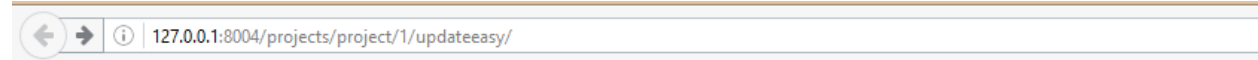
in the projects/forms.py file:

```
def __init__(self, *args, **kwargs):
    super(ProjectChampionForm, self).__init__(*args, **kwargs)
    self.fields['champions_choice_list'].label = "Update the champion"
    self.fields['champion'].widget = forms.HiddenInput()
```

New jQuery lines:

```
{# build the autocomplete list for the champions_choice_list #}
$("#id_champions_choice_list").easyAutocomplete(options_easy_autocomplete_champions);

var champion_name = "{{ project.champion.username }}"
$("#id_champions_choice_list").attr('placeholder', champion_name);
```



Test jquery EasyAutocomplete

Update of the project '(title:project test 2 champion:zoya)

Title:

Champion:

jQuery UI autocomplete test

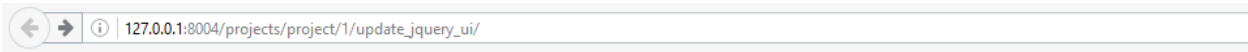
See also:

- <https://jqueryui.com/autocomplete/>
- <https://github.com/jquery/jquery-ui/tree/master/demos>
- <https://github.com/jquery/jquery-ui/blob/master/demos/autocomplete/maxheight.html>
- <https://www.npmjs.com/package/jquery-ui.autocomplete.scroll>

- <https://github.com/anseki/jquery-ui-autocomplete-scroll>

Contents

- *jQuery UI autocomplete test*
 - *Running the local Django web server*
 - *The projects/forms.py module*
 - *The HTML and JavaScript part (the Django Template)*
 - *The projects/views.py module*
 - *The projects/urls.py module*
 - *The jQuery UI scrollbar module*



Test jquery EasyAutocomplete

Update of the project '(title:project test 2 champion:Tazio_Basile)

Title:

Champion:

Abdul_Parker

Adolf_Wahlberg

Alexander_Dahlberg

Alexandria_Weiss

Amber_Taylor

Anaïs_Bonneau

André_Albert

Anette_Robertsson

Anežka_Bláhová

Asser_Sirviö

Fig. 2.1: jQuery UI with scrollbar (thanks <https://github.com/anseki>)

Running the local Django web server

```
System check identified no issues (0 silenced).
October 26, 2016 - 08:54:53
Django version 1.10.2, using settings 'projet_ajax.settings'
Starting development server at http://127.0.0.1:8004/
Quit the server with CTRL-BREAK.
```

The projects/forms.py module

```
#!/usr/bin/python
# -*- coding: utf8 -*-
"""The project's forms.

"""

from django import forms

from .models import Project

class ProjectChampionForm(forms.ModelForm):
    """The champion project form"""
    champions_choice_list = forms.CharField(max_length=100,
                                           help_text='type username or email')

    class Meta:
        model = Project
        fields = ('title',
                  'champions_choice_list', 'champion',)

    def __init__(self, *args, **kwargs):
        super(ProjectChampionForm, self).__init__(*args, **kwargs)
        self.fields['champions_choice_list'].label = "Update the champion"
        self.fields['champion'].widget = forms.HiddenInput()
```

The HTML and JavaScript part (the Django Template)

```
{% load static %}
{% load staticfiles %}

<!DOCTYPE html>
<html lang="en">
  <head>
    <title>jQuery UI complete simple example</title>

    <!-- STYLE -> CSS -->
    {# https://jqueryui.com/autocomplete/ #}
    <link rel="stylesheet" href="//code.jquery.com/ui/1.12.1/themes/base/jquery-
    ↪ui.css" type="text/css">
    <!-- end STYLE-->

  </head>
  <body>

    <!-- STRUCTURE -> HTML5 elements -->
    <h1>Test jquery EasyAutocomplete</h1>
    <h1>Update of the project '(title:{{ project.title }} champion:{{ project.
    ↪champion.username }})' </h1>
    <p></p>
    <p></p>
    {# https://docs.djangoproject.com/en/dev/topics/forms/ #}
    <form id="id_form_project_update_easy" action="{% url 'projects:project_
    ↪update' project.id %}" method="post">
      {% csrf_token %}
```

```

        <div class="forms">
            {{ form.id }}
            {{ form.non_field_errors }}
            {% Include the hidden fields %}
            {% for hidden in form.hidden_fields %}
                {% here we will have the champion filed (which is hidden) %}
                {{ hidden }}
            {% endfor %}
            <table id="id_table" class="table table-hover table-bordered_
→table-condensed">
                <tbody>
                    <tr>
                        <td class="text-right">Title:</td>
                        <td>{{ form.title }}</td>
                    </tr>
                    <tr>
                        <td class="text-right">Champion:</td>
                        <td> {{ form.champions_choice_list }} </td>
                    </tr>
                </tbody>
            </table>
        </div>
        <input type="submit" name="btn_update" value="Update" class="btn btn-
→success btn-block" />
    </form>
    <!-- end STRUCTURE-->

    <!--BEHAVIOR -> Javascript scripts-->
    <!-- Using jQuery with a CDN -->
    <script src="//code.jquery.com/jquery-1.11.2.js"></script>
    <script src="//cdnjs.cloudflare.com/ajax/libs/jqueryui/1.12.1/jquery-ui.js
→"></script>
    {# https://github.com/anseki/jquery-ui-autocomplete-scroll #}
    <script src="{% static 'jquery_ui/js/jquery.ui.autocomplete.scroll.min.js
→' %}" type="text/javascript"></script>
    <script>
        var options_jquery_ui_autocomplete_champions = {
            {# we have to build this URL: http://127.0.0.1:8004/projects/api_
→get_champions/?term=a #}
            {# calling the JSON view (this is the very famous AJAX call) the_
→+ "?term=" + term; is added by autocomplete #}
            maxShowItems: 10,
            source: "{% url 'projects:api_get_champions' %}",
            minLength: 1,
            select: function( event, ui ) {
                {# put the id in the champion field #}
                let champion_id = ui.item.id;
                $("#id_champion")
                    .val(champion_id)
                    .trigger("change");
                console.log( "Selected: " + ui.item.value + " aka " + ui.item.
→id );
            }
        };

        {# build the autocomplete list for the champions_choice_list #}
        $("#id_champions_choice_list").autocomplete(options_jquery_ui_
→autocomplete_champions);

```



```

        var champion_name = "{{ project.champion.username }}"
        $("#id_champions_choice_list").attr('placeholder', champion_name);

    </script>
    <!-- end BEHAVIOR -->

</body>

</html>

```

The projects/views.py module

```

class ProjectUpdateViewjQueryUIAutoComplete(UpdateView):
    """Update the view with the jQuery UI Autocomplete plugin.

    Documentation:

    - http://ccbv.co.uk/projects/Django/1.10/django.views.generic.edit/UpdateView/

    """
    model = Project
    form_class = ProjectChampionForm
    context_object_name = 'project'
    template_name = 'projects/project/update_jquery_ui_autocomplete.html'

    def get_object(self, queryset=None):
        """Pour mémoriser self.demande_article"""
        self.object = super(ProjectUpdateViewjQueryUIAutoComplete, self).get_
        ↪ object(queryset)
        return self.object

    def post(self, request, *args, **kwargs):
        logger.warning("Hello from ProjectUpdateViewjQueryUIAutoComplete !")
        ↪ return super(ProjectUpdateViewjQueryUIAutoComplete, self).post(request, *args,
        ↪ **kwargs)

```

The projects/urls.py module

```

urlpatterns = [
    url(r'^project/(?P<pk>\d+)/update/$',
        ProjectUpdateView.as_view(),
        name='project_update'),

    url(r'^project/(?P<pk>\d+)/updateeasy/$',
        ProjectUpdateViewEasyAutoComplete.as_view(),
        name='project_update_easy'),

    url(r'^project/(?P<pk>\d+)/update_jquery_ui/$',
        ProjectUpdateViewjQueryUIAutoComplete.as_view(),
        name='project_update_jquery_ui'),

```

The jQuery UI scrollbar module

See also:

jquery_ui_autocomplete_scroll plugin

Actions 2016

2016-10-25 installing gitsome

See also:

- *django-autocomplete-light test*
- <https://github.com/donnemartin/gitsome>
- <https://github.com/integrations/gitsome>
- https://twitter.com/donne_martin

install

```
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>pip install_
↪gitsome

Collecting gitsome
  Downloading gitsome-0.6.0.tar.gz (308kB)
    100% |#####| 317kB 819kB/s
Collecting numpydoc<1.0,>=0.5 (from gitsome)
  Downloading numpydoc-0.6.0.tar.gz
Collecting ply<4.0,>=3.4 (from gitsome)
  Downloading ply-3.9.tar.gz (150kB)
    100% |#####| 153kB 1.3MB/s
Requirement already satisfied (use --upgrade to upgrade): prompt-toolkit<1.1.0,>=1.0.
↪0 in c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from_
↪gitsome)
Requirement already satisfied (use --upgrade to upgrade): requests<3.0.0,>=2.8.1 in_
↪c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from_
↪gitsome)
```

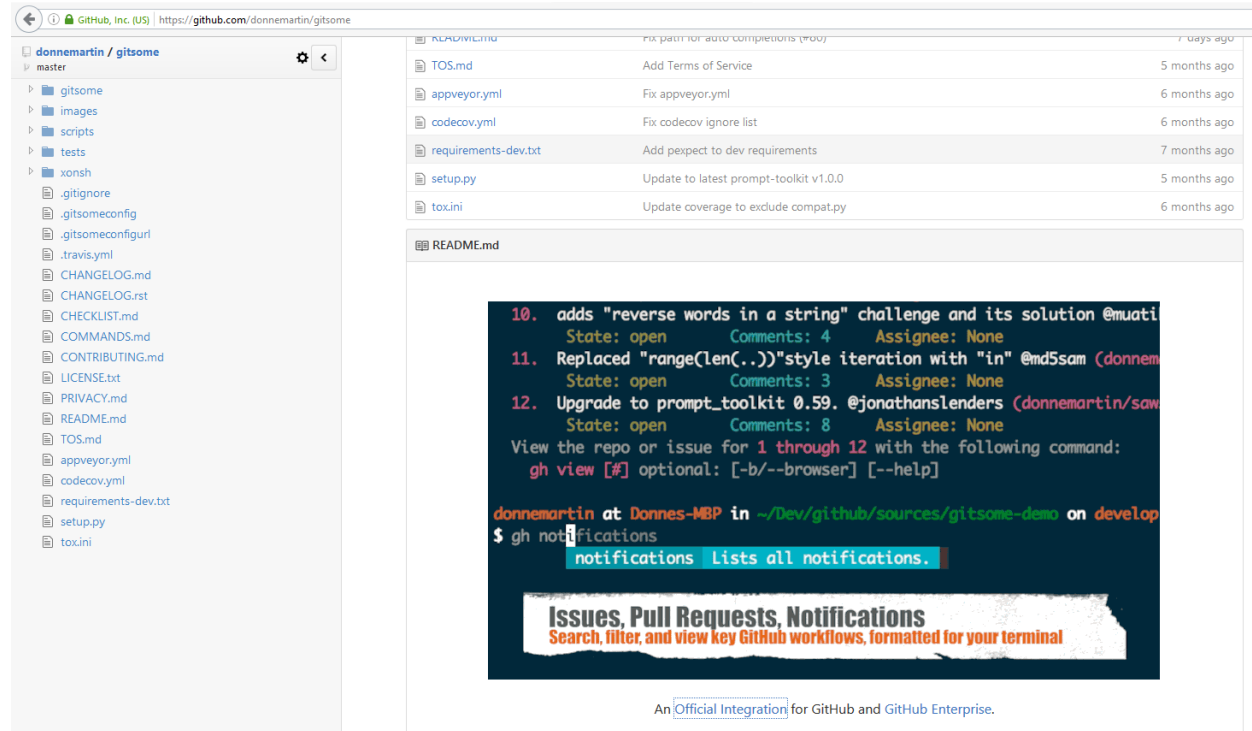


Fig. 3.1: An Official Integration for GitHub and GitHub Enterprise.

```
Requirement already satisfied (use --upgrade to upgrade): colorama<1.0.0,>=0.3.3 in_
→c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from_
→gitsome)
Collecting click<7.0,>=5.1 (from gitsome)
Requirement already satisfied (use --upgrade to upgrade): pygments<3.0.0,>=2.0.2 in_
→c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from_
→gitsome)
Collecting feedparser<6.0.0,>=5.2.1 (from gitsome)
  Downloading feedparser-5.2.1.zip (1.2MB)
    100% |#####| 1.2MB 655kB/s
Requirement already satisfied (use --upgrade to upgrade): pytz<2017.0,>=2016.3 in_
→c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from_
→gitsome)
Collecting docopt<1.0.0,>=0.6.2 (from gitsome)
Collecting uritemplate.py<1.0.0,>=0.2.0 (from gitsome)
  Downloading uritemplate.py-0.3.0.tar.gz
Requirement already satisfied (use --upgrade to upgrade): wcwidth in_
→c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from_
→prompt-toolkit<1.1.0,>=1.0.0->gitsome)
Requirement already satisfied (use --upgrade to upgrade): six>=1.9.0 in_
→c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from_
→prompt-toolkit<1.1.0,>=1.0.0->gitsome)
Building wheels for collected packages: gitsome, numpydoc, ply, feedparser,
→uritemplate.py
  Running setup.py bdist_wheel for gitsome ... done
  Stored in directory:
→C:\Users\pvergain\AppData\Local\pip\Cache\wheels\c8\4f\5a\128527ed75a5c07780bcef517c1be9a031b0f4672
  Running setup.py bdist_wheel for numpydoc ... done
  Stored in directory:
→C:\Users\pvergain\AppData\Local\pip\Cache\wheels\9d\ca\8c\9be286b195791bbbfbefbd407899de13eb12725a2
```

```
Running setup.py bdist_wheel for ply ... done
Stored in directory:
→C:\Users\pvergain\AppData\Local\pip\Cache\wheels\c1\0c\bd\306a63396decbe8353a4a056fcb97a092be0e03
Running setup.py bdist_wheel for feedparser ... done
Stored in directory:
→C:\Users\pvergain\AppData\Local\pip\Cache\wheels\15\ce\10\b500f745822ea6db6ea8ed225c06b15c000d7101
Running setup.py bdist_wheel for uritemplate.py ... done
Stored in directory:
→C:\Users\pvergain\AppData\Local\pip\Cache\wheels\f0\28\64\52c28cc4428d1c79eb7e923cade677f5e63ae1d2
Successfully built gitsome numpydoc ply feedparser uritemplate.py
Installing collected packages: numpydoc, ply, click, feedparser, docopt, uritemplate.
→py, gitsome
Successfully installed click-6.6 docopt-0.6.2 feedparser-5.2.1 gitsome-0.6.0 numpydoc-
→0.6.0 ply-3.9 uritemplate.py-0.3.0
```

2016-10-25 testing jquery-ui autocomplete and django-autocomplete-light

See also:

- *django-autocomplete-light test*

Conclusion

We adopt *django-autocomplete-light*.

Note: With jquery-ui autocomplete we have discovered <https://github.com/anseki/jquery-ui-autocomplete-scroll> and <https://github.com/anseki> who has a deep knowledge of JavaScript.

2016-10-25 Building fake users with the Python faker module

See also:

- *Python faker module*
- <http://blog.districtdatalabs.com/a-practical-guide-to-anonymizing-datasets-with-python-faker>

Motivation

We want to test the autocomplete test with a large number of users. So, we want to create *fake* users.

2016-10-24 Learning jQuery 101

See also:

- *Step2 : initialize the easyAutocomplete placeholder with the value of champion*
- *2016-10-21 Receive 2 new excellent books about jQuery: jQuery In Action and jquery UI in Action*

127.0.0.1:8004/projects/project/1/updateeasy/

Test jquery EasyAutocomplete

Update of the project '(title:project test 2 champion:zoya)

Title:

Champion:

- Alicia_Pareja
- Alighieri_Coppola
- Ana_Orozco
- Andrés_Vizcaino
- Arthur_Chauvet
- Belen_Cárdenas
- Bente_Christoffersen
- Blanca_Anguita
- Blanca_Aparicio
- Carl_Carlsen
- Carmen_Piquer
- Catrine_Petersen
- Cecco_De Angelis
- Colette_Guyot
- Corinne_Imbert
- Céline_Poulain
- Declan_Pugh
- Ebbe_Christensen
- Erna_Koch
- Francisco Jose_Pedro
- Francisco_Company
- Gabrielle_Charrier
- Gioacchino_Martini
- Guillermo_Rocha
- Henry_Bech
- Jessica_Taylor
- Jill_Clark
- Jose Francisco_Ojeda

2016-10-21 Receive 2 new excellent books about jQuery: jQuery In Action and jquery UI in Action

See also:

- *2016-10-21 Receive 2 new excellent books about jQuery: jQuery In Action and jquery UI in Action*

2016-10-21 testing the jquery EasyAutocomplete (EAC) options for an AJAX call

Contents

- *2016-10-21 testing the jquery EasyAutocomplete (EAC) options for an AJAX call*
 - *Renaming views*
 - *Add a great number of projects into the 'project' table*
 - *Write an API view/urls to select projects*
 - * *The view*
 - * *The URL*
 - * *Tests with httpie*
 - *Update the Django template file in order to call this view*
 - *6 elements is not enough*
 - *Ajout placeholder*
 - *list['onSelectItemEvent']*
 - * *HTML5*
 - * *Javascript*
 - *list['showAnimation']*
 - *TODO : tester categories*

Renaming views

Rename `champion_get_json` into `api_get_champions`

Add a great number of projects into the 'project' table

See also:

- <https://github.com/fcurella/django-fakery>
- <http://blog.districtdatalabs.com/a-practical-guide-to-anonymizing-datasets-with-python-faker>

We can create a Django command or a simple loop in the command line interface.

```
user = User.objects.last()
```

```
for i in range(20):
    title='python project:{}'.format(i)
    p = Project.objects.create(champion=user, title=title)
    print(p)
```

Write an API view/urls to select projects

The view

```
class ApiGetProjectsView(FormView):
    """
    Documentation
    =====

    - https://ccbv.co.uk/projects/Django/1.10/django.views.generic.edit/FormView/

    """
    def get(self, request, *args, **kwargs):
        """term is sent by the jquery-ui autocomplete widget.

        For the jquery-ui autocomplete widget we have to return 3 fields:

        - id
        - label
        - value

        For the jquery EasyAutocomplete we can return what we want.

        """
        term = request.GET.get("term")
        if term:
            projects = Project.objects.filter(title__icontains=term)
        else:
            projects = Project.objects.all()[:50]

        results = []
        for project in projects:
            project_json = {}
            project_json['id'] = project.id
            project_json['label'] = project.title
            project_json['value'] = project.title
            results.append(project_json)

        data = json.dumps(results)
        mimetype = 'application/json'
        return HttpResponse(data, mimetype)
```

The URL

```
# calls by jquery EasyAutocomplete (EAC)
# http://127.0.0.1:8004/projects/api_get_projects/?term=a
url(r'^api_get_projects/$',
```



```
ApiEACGetProjectsView.as_view(),
name='api_get_projects'),
```

Tests with httpie

```
http http://127.0.0.1:8004/projects/api_get_projects/?term=a
```

```
HTTP/1.0 200 OK
Content-Type: application/json
Date: Fri, 21 Oct 2016 13:03:31 GMT
Server: WSGIServer/0.2 CPython/3.5.2
X-Frame-Options: SAMEORIGIN
```

```
[
  {
    "id": 52,
    "label": "an other projec:0",
    "value": "an other projec:0"
  },
  {
    "id": 53,
    "label": "an other projec:1",
    "value": "an other projec:1"
  },
  {
    "id": 54,
    "label": "an other projec:2",
    "value": "an other projec:2"
  },
]
```

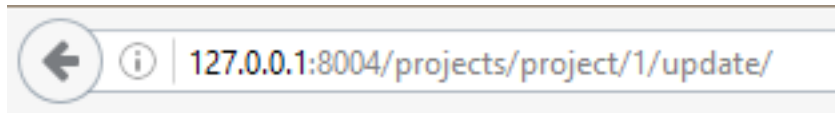
Update the Django template file in order to call this view

```
<script>
  var options_easy_autocomplete_ajax = {
    {# we have to build this URL: http://127.0.0.1:8004/projects/api_get_projects/
    ↪?term=a #}
    url: function(term) {
      return "{% url 'projects:api_get_projects' %}" + "?term=" + term;
    },
    getValue: "value",

  };
  $("#data-ajax").easyAutocomplete(options_easy_autocomplete_ajax);
</script>
```

6 elements is not enough

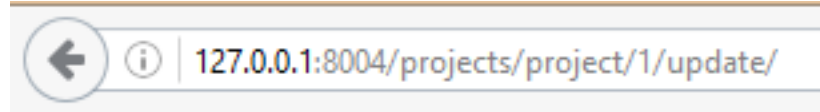
OK, no slider



EasyAutocomplete

Choose project

- python project:0
- python project:1
- python project:2
- python project:3
- python project:4
- python project:5



EasyAutocomplete

Choose project

p
project test
projet:0
projet:1
projet:2
projet:3
projet:4
projet:5
projet:6
projet:7
projet:8
projet:9
projet:10
projet:11
projet:12
projet:13
projet:14
projet:15
projet:16
projet:17
projet:18
projet:19

Ajout placeholder

```
<!-- STRUCTURE -> HTML5 elements -->
<h1>EasyAutocomplete</h1>

<input id="data-ajax" placeholder="Choose the project" size="50" />
<!-- end STRUCTURE-->
```

EasyAutocomplete

Choose the project

`list['onSelectItemEvent']`

See also:

<http://easyautocomplete.com/guide#sec-function-selected-data>

HTML5

```
<!-- STRUCTURE -> HTML5 elements -->
<h1>EasyAutocomplete</h1>

<input id="project_choice_list" placeholder="Choose the project" size="50" />
<input id="projet_holder" type="text" value="" size="50">
<!-- end STRUCTURE-->
```

Javascript

```
var options_easy_autocomplete_ajax = {
  {# we have to build this URL: http://127.0.0.1:8004/projects/api_get_projects/?
  ↪term=a #}
  url: function(term) {
    return "{% url 'projects:api_get_projects' %}" + "?term=" + term;
  },
  getValue: "title",
  list: {
    maxNumberOfElements: 200,
    onSelectItemEvent: function() {
      var project_id = $("#project_choice_list").getSelectedItemData().id;
```

```

        {# put the project.id in the projet_holder field #}
        $("#projet_holder").val(project_id).trigger("change");
    },
    match: {
        enabled: true
    },
},
},
};
$("#project_choice_list").easyAutocomplete(options_easy_autocomplete_ajax);

```

`list['showAnimation']`

TODO : tester categories

See also:

- <http://easyautocomplete.com/guide#sec-categories>

2016-10-21 pushing documentation on readthedocs

See also:

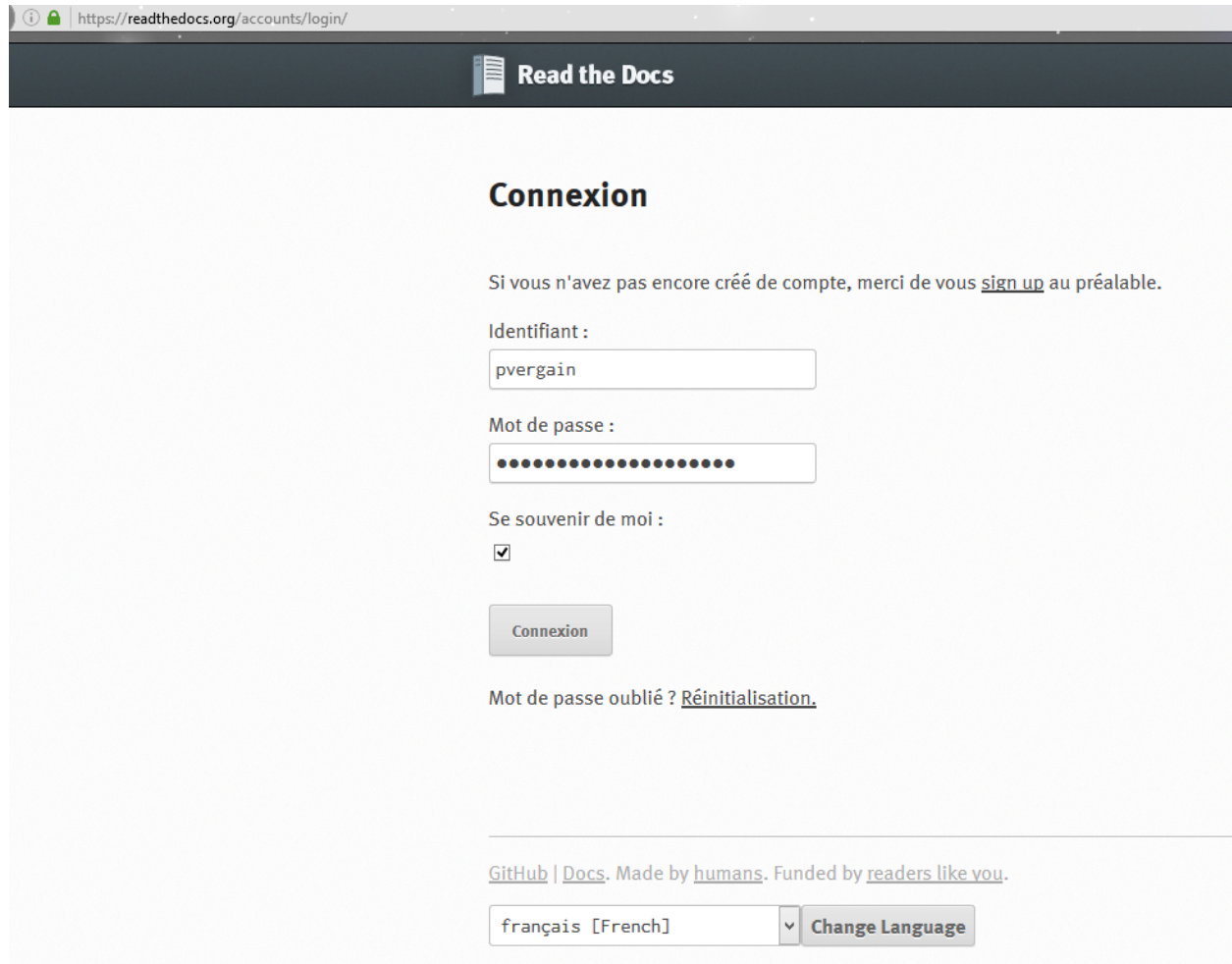
- <https://readthedocs.org/>
- <https://readthedocs.org/accounts/login/>
- <https://django-test-autocomplete.readthedocs.io/en/latest/>
- <https://readthedocs.org/projects/django-test-autocomplete/>

The screenshot shows the Read the Docs interface for the 'django-test-autocomplete' project. The top navigation bar includes the 'Read the Docs' logo and a user profile for 'pvergain'. Below the navigation bar, the project name 'django-test-autocomplete' is displayed, along with a button to 'Afficher les Docs'. A row of buttons provides access to 'Aperçu', 'Téléchargements', 'Chercher', 'Compilations', 'Versions', and 'Admin'. The 'Versions' section features a table with two entries: 'latest' and 'stable', both marked as 'Public' with a 'Modifier' button. Below this, a 'Compiler une version' section allows selecting a version (currently 'latest') and clicking a 'Compiler' button. The 'Description' section states: 'Django-test-autocomplete tests and documents the settings of jquery autocomplete plugins in a Django project.' On the right side, the 'Dépot' section provides a GitHub link, and the 'Dernière compilation' section shows a 'passed' status from 5 minutes ago. The 'Owners' section lists the project owner, and the 'Badge' section shows a 'docs latest' badge. The 'Etiquettes' section lists tags: 'Django', 'jquery', and 'autocomplete'. The 'Niveau de confidentialité du projet' is set to 'Public'.

Connect to readthedocs

See also:

- <https://readthedocs.org/accounts/login/>

A screenshot of the Read the Docs login page. The browser's address bar shows 'https://readthedocs.org/accounts/login/'. The page has a dark header with the 'Read the Docs' logo. The main content area is titled 'Connexion'. It contains a message: 'Si vous n'avez pas encore créé de compte, merci de vous [sign up](#) au préalable.' Below this are two input fields: 'Identifiant :' with the value 'pvergain' and 'Mot de passe :'. The password field is masked with dots. There is a 'Se souvenir de moi :' checkbox which is checked. A 'Connexion' button is below the inputs. At the bottom of the form, it says 'Mot de passe oublié ? [Réinitialisation.](#)'. At the very bottom of the page, there is a footer with links: 'GitHub | Docs. Made by [humans](#). Funded by [readers like you](#).' and a language selector showing 'français [French]' with a 'Change Language' button.

https://readthedocs.org/accounts/login/

Read the Docs

Connexion

Si vous n'avez pas encore créé de compte, merci de vous [sign up](#) au préalable.

Identifiant :

Mot de passe :

Se souvenir de moi : ☒

Mot de passe oublié ? [Réinitialisation.](#)

[GitHub](#) | [Docs](#). Made by [humans](#). Funded by [readers like you](#).

français [French]

Import a projet

See also:

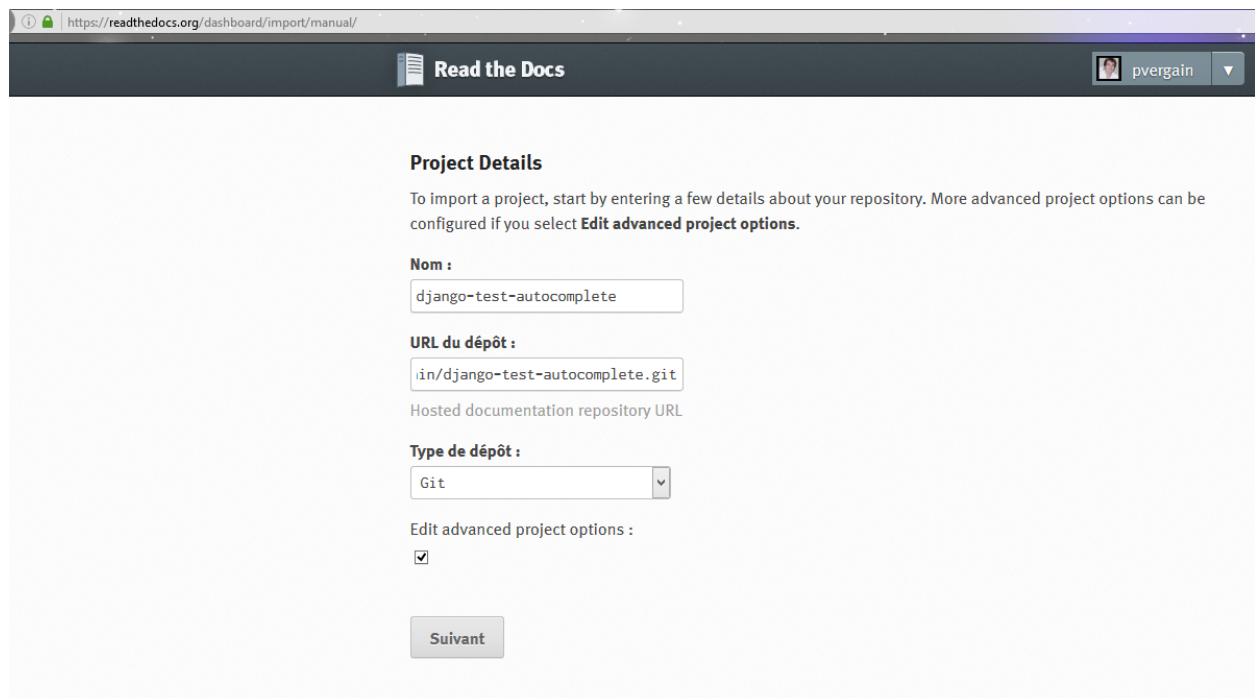
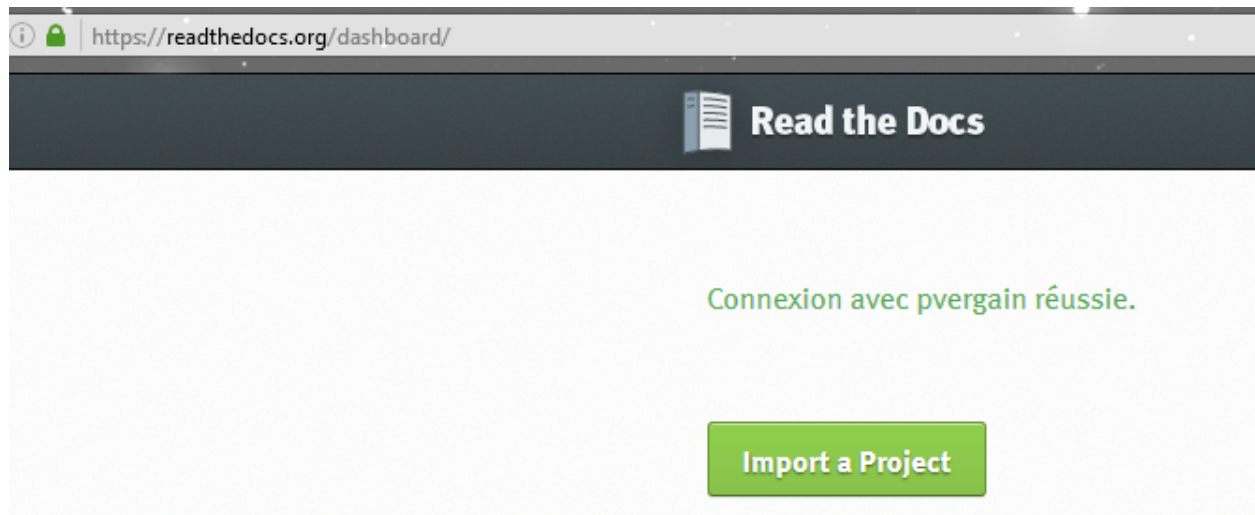
- https://docs.readthedocs.io/en/latest/getting_started.html#import-docs

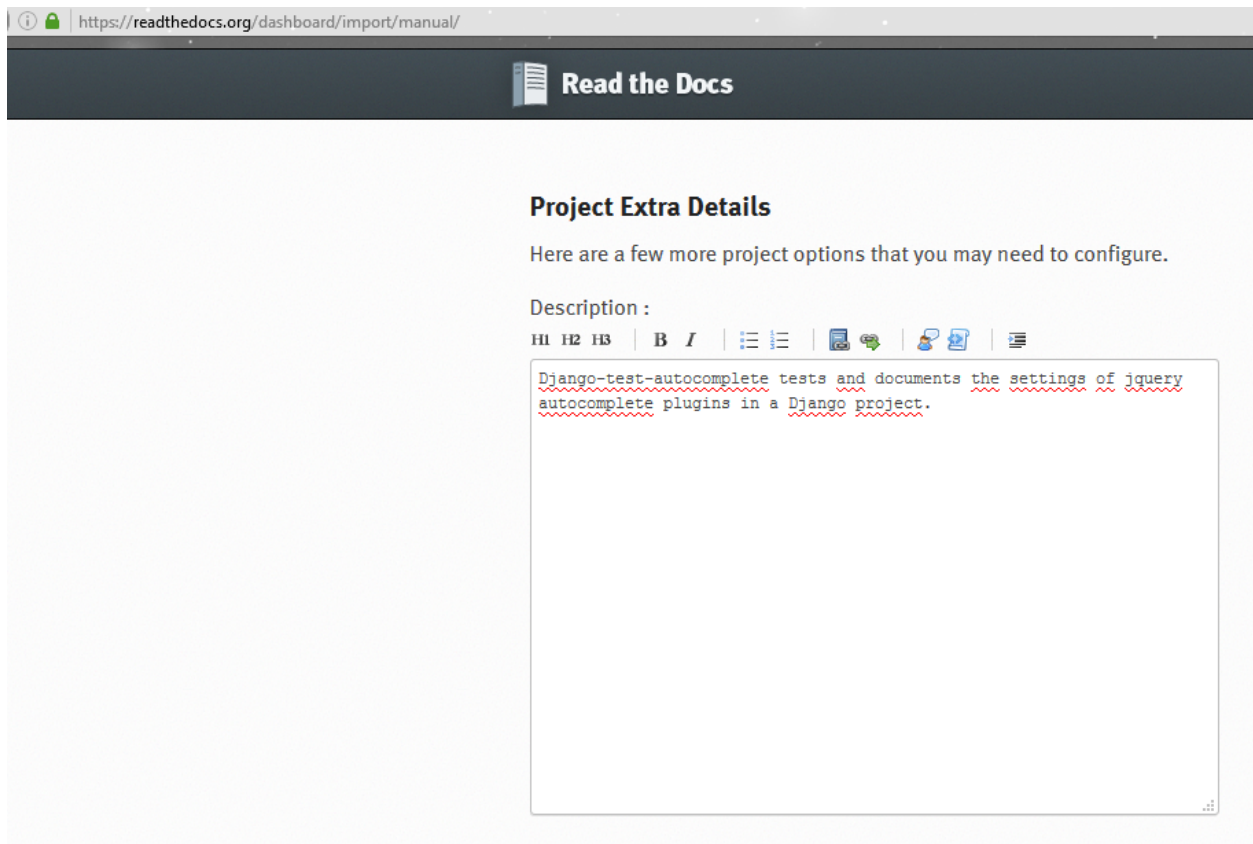
Advanced parameters

2016-10-20 2016-10-21 First step: using the jquery EasyAutocomplete plugin with Django

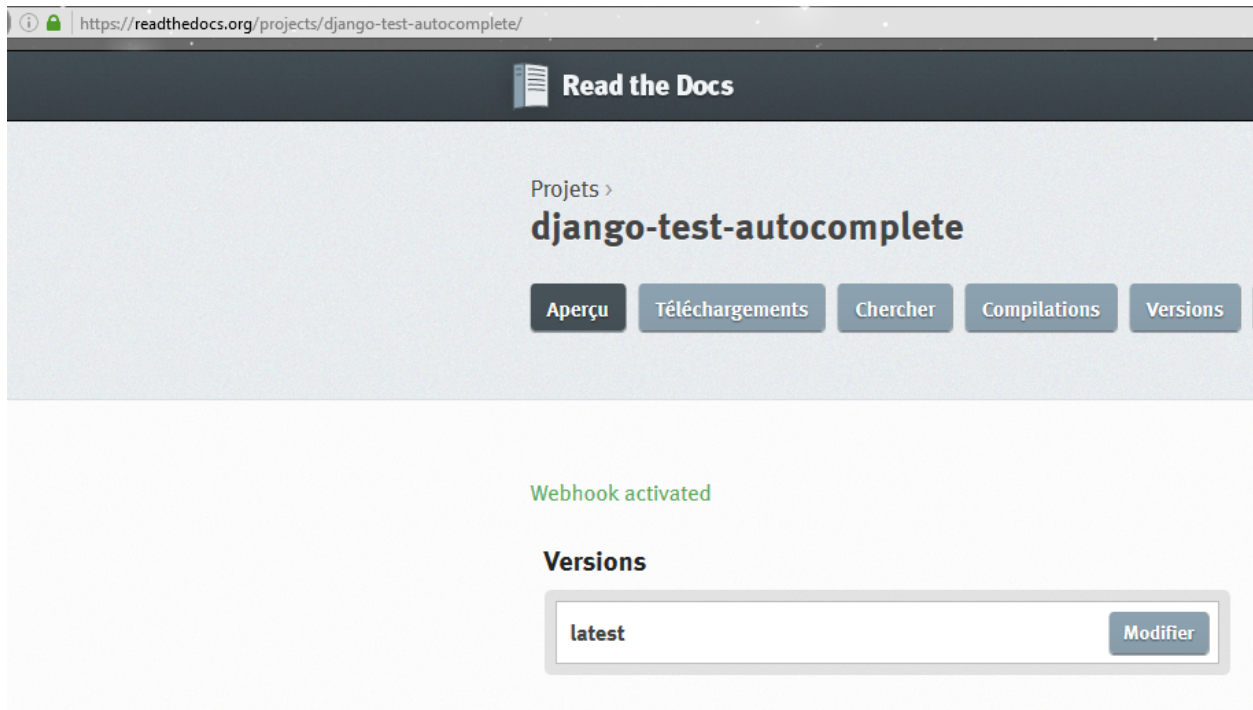
See also:

- <http://easyautocomplete.com/>
- <http://easyautocomplete.com/guide>





The screenshot shows the Read the Docs dashboard for the 'django-test-autocomplete' project. The browser address bar displays 'https://readthedocs.org/dashboard/import/manual/'. The page header features the 'Read the Docs' logo. The main content area is titled 'Project Extra Details' and includes a sub-header 'Here are a few more project options that you may need to configure.' Below this, a 'Description :' section contains a rich text editor with a toolbar (H1, H2, H3, B, I, bulleted list, numbered list, link, unlink, image, video, quote, code, fullscreen) and a text area containing the text: 'Django-test-autocomplete tests and documents the settings of jquery autocomplete plugins in a Django project.'



The screenshot shows the Read the Docs project page for 'django-test-autocomplete'. The browser address bar displays 'https://readthedocs.org/projects/django-test-autocomplete/'. The page header features the 'Read the Docs' logo. The main content area is titled 'Projets > django-test-autocomplete'. Below the title, there are five buttons: 'Aperçu', 'Téléchargements', 'Chercher', 'Compilations', and 'Versions'. A green status message 'Webhook activated' is displayed. The 'Versions' section shows a dropdown menu with 'latest' selected and a 'Modifier' button.

https://readthedocs.org/dashboard/django-test-autocomplete/advanced/

Paramètres avancés	Install Project :
Versions	<input type="checkbox"/> Install your project inside a virtualenv using <code>setup.py install</code>
Domains	Fichier des prérequis :
Mainteneurs	<input type="text"/>
Redirects	A pip requirements file needed to build your documentation. Path from the root of your project.
Traductions	Version unique :
Sous-projets	<input type="checkbox"/> Un site avec une version unique n'a pas de traductions et n'affiche que la dernière version, à la racine de votre domaine. Soyez prudent, n'activez cette option que si vous n'allez jamais avoir plusieurs versions de votre documentation.
Notifications	Fichier de configuration Python :
Advertising	<input type="text" value="doc_sphinx/conf.py"/>
	Path from project root to <code>conf.py</code> file (ex. docs/conf.py). Leave blank if you want us to find it for you.

https://readthedocs.org/dashboard/django-test-autocomplete/advanced/

Enable PDF build :
<input checked="" type="checkbox"/> Create a PDF version of your documentation with each build.
Enable EPUB build :
<input checked="" type="checkbox"/> Create a EPUB version of your documentation with each build.
Niveau de confidentialité :
<input type="text" value="Public"/>
(Beta) Niveau de confidentialité que vous souhaitez pour ce dépôt. Protégé signifie public mais absent des index.
Utiliser les paquets système :
<input type="checkbox"/> Donner accès à l'environnement virtuel au dossier global site-packages.
Python interpreter :
<input type="text" value="CPython 3.x"/>
(Beta) L'interpréteur Python utilisé pour créer l'environnement virtuel.
Code Analytics :
<input type="text"/>
Google Analytics Tracking ID (ex. UA-22345342-1). This may slow down your page loads.
<input type="button" value="Soumettre"/>

- <http://easyautocomplete.com/guide#sec-data-providers>
- <https://github.com/pawelczak/EasyAutocomplete>

Contents

- *2016-10-20 2016-10-21 First step: using the jquery EasyAutocomplete plugin with Django*
 - *Why ? EasyAutocomplete can easily become one of the best autocomplete plugins available for free*
 - *EasyAutocomplete files tree (Javascript, CSS, JSON, PHP files)*
 - *Javascript SCSS and CSS EasyAutocomplete files*
 - * *EasyAutocomplete-master/src/sass/easy-autocomplete.scss*
 - * *EasyAutocomplete-master/dist/easy-autocomplete.css*
 - * *EasyAutocomplete-master/dist/jquery.easy-autocomplete.js*
 - *EasyAutocomplete simple example*
 - * *EasyAutocomplete-master/demo/example_simple.html*
 - *EasyAutocomplete JSON example*
 - * *EasyAutocomplete-master/demo/example_json.html*
 - * *EasyAutocomplete-master/demo/resources/countries.json*
 - *EasyAutocomplete flags example*
 - * *EasyAutocomplete-master/demo/example_flags.html*
 - * *EasyAutocomplete-master/demo/resources/flags.css*
 - * *EasyAutocomplete-master/demo/resources/flags.png*
 - *EasyAutocomplete categories example*
 - * *EasyAutocomplete-master/demo/example_categories.html*
 - * *EasyAutocomplete-master/demo/resources/categories.json*
 - *EasyAutocomplete static_link example*
 - * *EasyAutocomplete-master/demo/example_static_link.html*
 - * *EasyAutocomplete-master/demo/resources/site.json*
 - *EasyAutocomplete email example*
 - * *EasyAutocomplete-master/demo/example_email.html*
 - * *EasyAutocomplete-master/demo/resources/people.json*
 - *EasyAutocomplete Django integration*
 - * *Include JS and CSS files from the distribution (static files)*
 - *Exemples of static files in the Django World*
 - * *The Django contrib module*
 - * *Django_By_Example_Code/Chapter 1/mysite/blog*
 - * *Django_By_Example_Code/Chapter 8*
 - * *Django_By_Example_Code/Chapter 13*

- *Django settings example*
- *Integration in our Django test project*
- *Create a simple update_easy_simple.html*
 - * *Le fichier projects/templates/projects/projet/update_easy_simple.html*
 - * *First problem : the libraries are not found*
 - * *collectstatic*
 - * *New tree*
 - * *Create an easyautocomplete directory*
 - * *Update the Django template file*
 - * *OK it works !*
 - * *Add JSON files*
- *Adding the AJAX call in the Django template*
 - * *URL : http://127.0.0.1:8004/projects/champion_get_json/?term=a*
 - * *pip install httpie (clihttp)*
 - * *http http://127.0.0.1:8004/projects/champion_get_json/?term=a*
 - * *http http://easyautocomplete.com/api/countrySearch.php?phrase=co*
 - * *Avec countrySearch.php*
 - * *Avec Python/Django OK the first step is DONE*
 - *The Template*
 - *The View ChampionAutoCompleteView*
 - *The projects urls.py*
- *This is the happy end of the first step*

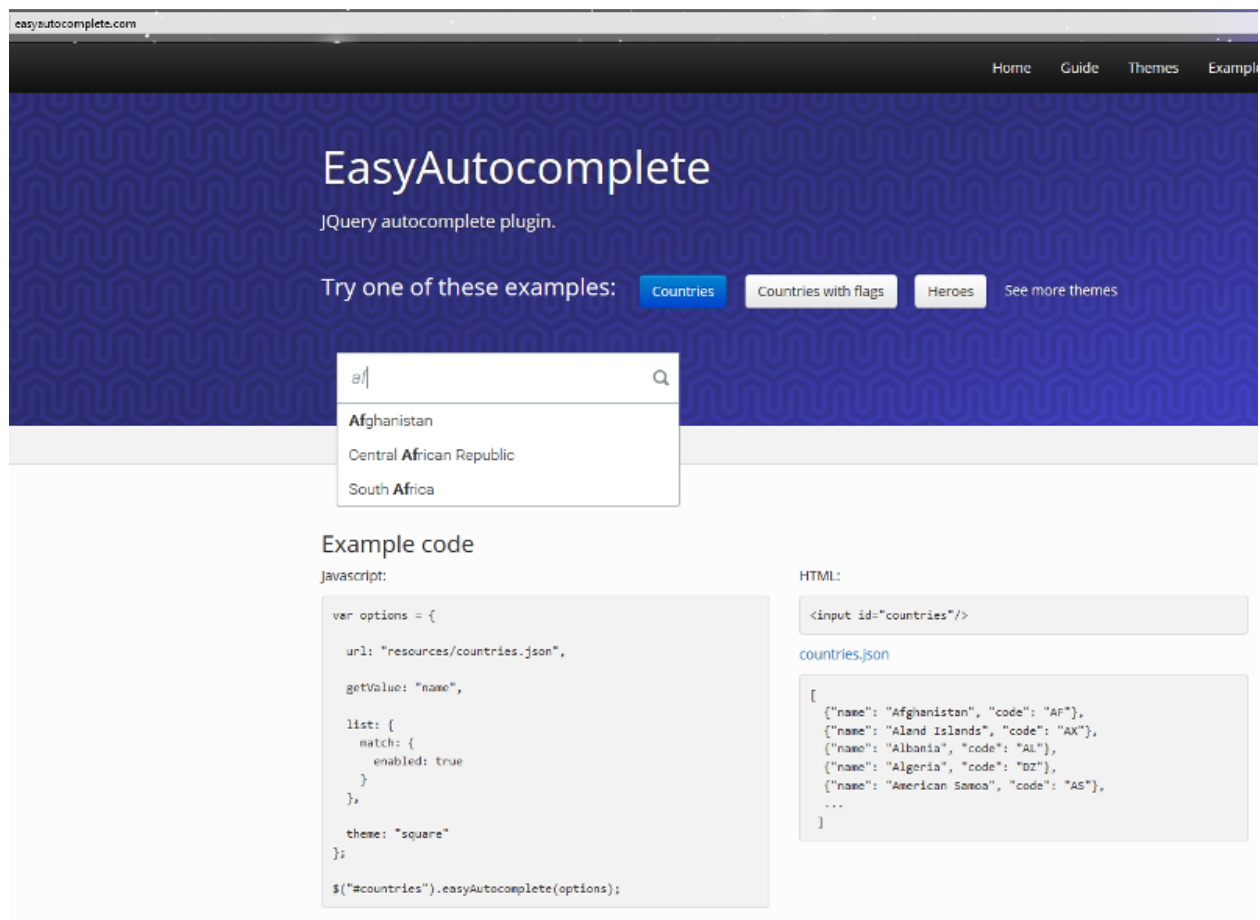


Why ? EasyAutocomplete can easily become one of the best autocomplete plugins available for free

See also:

<http://126kr.com/article/69n5fxx65il>

Because jquery Easyautocomplete can easily become one of the best autocomplete plugins available for free..



The screenshot shows the EasyAutocomplete website. The header includes the URL 'easyautocomplete.com' and navigation links: Home, Guide, Themes, and Examples. The main heading is 'EasyAutocomplete' with the subtitle 'jQuery autocomplete plugin.' Below this, it says 'Try one of these examples:' followed by buttons for 'Countries', 'Countries with flags', 'Heroes', and a link 'See more themes'. A search input field contains the text 'a|' and has a magnifying glass icon. A dropdown menu is open, showing suggestions: 'Afghanistan', 'Central African Republic', and 'South Africa'. Below the search field, there is an 'Example code' section. It contains two code blocks: 'Javascript:' and 'HTML:'. The Javascript block shows the configuration for the plugin, including the URL to the countries.json file, the value to get, the list of countries, and the theme. The HTML block shows the input field and the countries.json file content.

easyautocomplete.com

Home Guide Themes Examples

EasyAutocomplete

jQuery autocomplete plugin.

Try one of these examples: [Countries](#) [Countries with flags](#) [Heroes](#) [See more themes](#)

a|

- Afghanistan
- Central African Republic
- South Africa

Example code

JavaScript:

```
var options = {  
  url: "resources/countries.json",  
  getValue: "name",  
  list: {  
    match: {  
      enabled: true  
    }  
  },  
  theme: "square"  
};  
$("#countries").easyAutocomplete(options);
```

HTML:

```
<input id="countries"/>
```

countries.json

```
[  
  {"name": "Afghanistan", "code": "AF"},  
  {"name": "Åland Islands", "code": "ÅX"},  
  {"name": "Albania", "code": "AL"},  
  {"name": "Algeria", "code": "DZ"},  
  {"name": "American Samoa", "code": "AS"},  
  ...  
]
```

126kr.com/article/69n5fox65il

126Kr126kr.com

Google Search



Home Programmi

EasyAutocomplete - A Lightweight jQuery Autocomplete Plugin

Datetime:2016-08-29 20:49:25 Topic: jQuery Share

There are quite a few autocomplete plugins available online, but all of them have their own strength and weaknesses. Among all options, the most popular one would be [jQuery Autocomplete](#). It bundled with a lot of useful features and configuration but it's bulky. I was looking for a lightweight solution with the features I need.

I needed a few features:

- It has to be lightweight because I'm building a mobile app with offline application mode.
- It parses local JSON data and able to assign name-value combinations.
- It has relevant callback events for data manipulation when user interact with the plugin.
- It need to filter the data in client-side.

Then I found [EasyAutocomplete](#). With all the features and configurable options, it can easily become one of the best autocomplete plugins available for free.

Example

I had a great time working wit EasyAutocomplete and I customised it to my requirements.

The following example will render icon next to each selection.

JSON File - countries.json

```
1 [
2   {
3     "name": "Afghanistan", "code": "AF"},
4     "name": "Aland Islands", "code": "AX"},
5     "name": "Albania", "code": "AL"},
6     "name": "Algeria", "code": "DZ"},
7     "name": "American Samoa", "code": "AS"},
8   ...
9 ]
```

EasyAutocomplete files tree (Javascript, CSS, JSON, PHP files)

```
C:.\
| .gitattributes
| .gitignore
| .jscsrc
| .jshintrc
| authors.txt
| bower.json
| Gruntfile.js
| LICENSE.txt
| package.json
| README.md
| tree_a_f.txt
|
+---demo
| | example_categories.html
| | example_custom_template.html
| | example_duckduckgo.html
| | example_email.html
| | example_flags.html
| | example_json.html
| | example_remote.html
| | example_simple.html
| | example_static_link.html
| | example_theme.html
| | example_theme_funky.html
| | example_theme_square.html
| | example_xml.html
| |
+---api
| | countries.php
| | countrySearch.php
| |
\---resources
| | categories.json
| | countries.json
| | countries.xml
| | flags.css
| | flags.png
| | icon_search.png
| | people.json
| | site.json
| |
+---dist
| | easy-autocomplete.css
| | easy-autocomplete.min.css
| | easy-autocomplete.themes.css
| | easy-autocomplete.themes.min.css
| | jquery.easy-autocomplete.js
| | jquery.easy-autocomplete.min.js
| |
\---maps
| | easy-autocomplete.css.map
| | easy-autocomplete.min.css.map
| | easy-autocomplete.themes.css.map
| | easy-autocomplete.themes.min.css.map
```

```
+---lib
|   jquery-1.11.2.min.js
|   jquery-2.1.3.min.js
|
+---src
|   |   configuration.js
|   |   constans.js
|   |   core.js
|   |   listBuilder.js
|   |   logger.js
|   |   proccessData.js
|   |   template.js
|   |
|   \---sass
|       |   easy-autocomplete.scss
|       |   easy-autocomplete.themes.scss
|
\---test
|   |   configuration.js
|   |   listBuilder.js
|   |   modules.html
|   |   proccessData.js
|   |   template.js
|   |
|   +---core
|   |   |   build.html
|   |   |   build.js
|   |   |   categories.html
|   |   |   categories.js
|   |   |   event.html
|   |   |   event.js
|   |   |   features.html
|   |   |   features.js
|   |   |   functions.html
|   |   |   functions.js
|   |   |   handles.html
|   |   |   handles.js
|   |   |   plugin.html
|   |   |   plugin.js
|   |   |   response.html
|   |   |   response.js
|   |   |   response_json.js
|   |   |   response_remote.html
|   |   |   response_remote.js
|   |   |   response_static.js
|   |   |   response_xml.js
|   |   |   template.html
|   |   |   template.js
|   |   |
|   |   +---remote
|   |   |   |   countries.php
|   |   |   |   countrySelectService.php
|   |   |   |
|   |   \---resources
|   |       |   categories.json
|   |       |   colors.json
|   |       |   colors.xml
|   |       |   colors_caps_string.json
```

```
|      | colors_object.json
|      | colors_object.xml
|      | colors_string.json
|      | countries.json
|      | duckduckgo.json
|      | response.json
|      |
|      | \---categories
|      |     fruits.json
|      |     fruits.xml
|      |     otherFruits.xml
|      |
| \---qunit
|      qunit.css
|      qunit.js
```

Javascript SCSS and CSS EasyAutocomplete files

EasyAutocomplete-master/src/sass/easy-autocomplete.scss

Listing 3.1: EasyAutocomplete-master/src/sass/easy-autocomplete.scss

```
1
2 $blue: #5A91CB;
3 $blue-light: rgba(102, 175, 233, 1);
4 $blue-lighter: rgba(102, 175, 233, 0.6);
5 $green-light: #41DB00;
6 $green-lighter: rgba(146, 237, 107, 0.6);
7 $red-light: #ff5b5b;
8 $red-lighter: rgba(255, 90, 90, 0.6);
9 $yellow-light: #ffdb00;
10 $yellow-lighter: rgba(255, 231, 84, 0.6);
11 $dark-light: #333;
12 $dark-lighter: rgba(55, 55, 55, 0.6);
13 $dark-glass: rgba(0, 0, 0, 0.8);
14 $dark: #333;
15 $yellow: rgba(255, 212, 100, 1);
16 $purple: #c7c0de;
17
18 @mixin placeholder {
19     &::-webkit-input-placeholder {@content}
20     &::-moz-placeholder           {@content}
21     &::-ms-input-placeholder      {@content}
22 }
23
24
25 .easy-autocomplete {
26     position: relative;
27
28     input {
29         border: {
30             color: #ccc;
31             radius: 4px;
32             style: solid;
33             width: 1px;
34         }
```



```

35     box-shadow: 0 1px 2px rgba(0, 0, 0, 0.1) inset;
36     color: #555;
37     float: none;
38     padding: 6px 12px;
39
40     &:hover, &:focus {
41         box-shadow: none;
42     }
43 }
44
45 a {
46     display: block;
47 }
48
49
50 &.eac-blue-light {
51
52     input {
53
54         &:hover, &:focus {
55             border-color: $blue-light;
56             box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $blue-
↪lighter;
57         }
58     }
59
60     ul {
61         border-color: $blue-light;
62         box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $blue-lighter;
63
64         li, .eac-category {
65             border-color: $blue-light;
66
67             &.selected {
68                 background-color: lighten($blue-light, 30%);
69             }
70         }
71     }
72 }
73
74
75 &.eac-green-light {
76
77     input {
78         &:hover, &:focus {
79             border-color: $green-light;
80             box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $green-
↪lighter;
81         }
82     }
83
84     ul {
85         border-color: $green-light;
86         box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $green-lighter;
87
88         li, .eac-category {
89             border-color: $green-light;
90

```

```
91         &.selected {
92             background-color: lighten($green-light, 30%);
93         }
94     }
95 }
96 }
97 }
98
99 &.eac-red-light {
100
101     input {
102         &:hover, &:focus {
103             border-color: $red-light;
104             box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $red-
↪lighter;
105         }
106     }
107
108     ul {
109         border-color: $red-light;
110         box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $red-lighter;
111
112         li, .eac-category {
113             border-color: $red-light;
114
115             &.selected {
116                 background-color: lighten($red-light, 10%);
117             }
118         }
119     }
120 }
121 }
122
123 &.eac-yellow-light {
124
125     input {
126         &:hover, &:focus {
127             border-color: $yellow-light;
128             box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $yellow-
↪lighter;
129         }
130     }
131
132     ul {
133         border-color: $yellow-light;
134         box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $yellow-lighter;
135
136         li, .eac-category {
137             border-color: $yellow-light;
138
139             &.selected {
140                 background-color: lighten($yellow-light, 10%);
141             }
142         }
143     }
144 }
145 }
146
```

```

147     &.eac-dark-light {
148
149         input {
150             &:hover, &:focus {
151                 border-color: $dark-light;
152                 box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $dark-
153                 ⇨lighter;
154             }
155         }
156
157         ul {
158             border-color: $dark-light;
159             box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px $dark-lighter;
160
161             li, .eac-category {
162                 border-color: $dark-light;
163
164                 &.selected {
165                     background-color: lighten($dark-light, 10%);
166                     color: #fff;
167                 }
168             }
169         }
170     }
171
172     &.eac-dark {
173         color: #fff;
174
175         input {
176             background-color: lighten($dark-light, 5%);
177             border: {
178                 radius: 4px;
179             }
180             box-shadow: 0;
181             color: #f6f6f6;
182
183             &:hover, &:focus {
184                 border-color: $dark;
185                 box-shadow: 0;
186             }
187         }
188
189         ul {
190             border-color: $dark;
191
192             li, .eac-category {
193                 background-color: lighten($dark-light, 5%);
194                 border-color: $dark;
195
196                 &.selected {
197                     background-color: lighten($dark, 25%);
198                     color: #f6f6f6;
199                 }
200             }
201         }
202     }
203 }

```

```
204 &.eac-dark-glass {
205     color: #fff;
206
207     input {
208         background-color: $dark-glass;
209         border: {
210             radius: 4px;
211         }
212         box-shadow: 0;
213         color: #f6f6f6;
214
215         &:hover, &:focus {
216             border-color: $dark-glass;
217             box-shadow: 0;
218         }
219     }
220 }
221
222 ul {
223     border-color: $dark-glass;
224
225     li, .eac-category {
226         background-color: $dark-glass;
227         border-color: $dark-glass;
228
229         &.selected {
230             background-color: lighten($dark-glass, 25%);
231             color: #f6f6f6;
232         }
233
234         &:last-child {
235             border-radius: 0 0 4px 4px;
236         }
237     }
238 }
239 }
240
241
242
243
244 &.eac-blue {
245     color: #fff;
246
247     input {
248         @include placeholder {
249             color: #f6f6f6;
250         }
251         background-color: lighten($blue, 5%);
252         border: {
253             radius: 4px;
254         }
255         box-shadow: 0;
256         color: #f6f6f6;
257
258         &:hover, &:focus {
259             border-color: $blue;
260             box-shadow: 0;
261         }
262     }
263 }
```

```
262     }
263
264     ul {
265         border-color: $blue;
266
267         li, .eac-category {
268             background-color: lighten($blue, 5%);
269             border-color: $blue;
270
271             &.selected {
272                 background-color: lighten($blue, 15%);
273                 color: #f6f6f6;
274             }
275         }
276     }
277 }
278
279
280 &.eac-yellow {
281     color: #333;
282
283     input {
284         background-color: lighten($yellow, 5%);
285         border: {
286             color: #333;
287             radius: 4px;
288         }
289         box-shadow: 0;
290         color: #333;
291
292         &:hover, &:focus {
293             border-color: #333;
294             box-shadow: 0;
295         }
296     }
297
298     ul {
299         border-color: #333;
300
301         li, .eac-category {
302             background-color: lighten($yellow, 5%);
303             border-color: #333;
304
305             &.selected {
306                 background-color: lighten($yellow, 15%);
307                 color: #333;
308             }
309         }
310     }
311 }
312
313
314 &.eac-purple {
315     color: #333;
316
317     input {
318         background-color: lighten($purple, 5%);
319         border: {
```

```
320         color: darken($purple, 5%);
321     }
322     box-shadow: 0;
323     color: #333;
324
325     &:hover, &:focus {
326         border-color: #333;
327         box-shadow: 0;
328     }
329 }
330
331 ul {
332     border-color: #333;
333
334     li, .eac-category {
335         background-color: lighten($purple, 5%);
336         border-color: #333;
337
338         &.selected {
339             background-color: lighten($purple, 12%);
340             color: #333;
341         }
342     }
343 }
344
345 }
346
347 &.eac-bootstrap {
348
349     input {
350         border-color: #ccc;
351         border-radius: 4px;
352         border-style: solid;
353         border-width: 1px;
354         color: #555;
355         padding: 6px 12px;
356     }
357 }
358
359
360 }
361
362
363 .easy-autocomplete-container {
364     left: 0;
365     position: absolute;
366     width: 100%;
367     z-index: 2;
368
369     ul {
370         background: none repeat scroll 0 0 #ffffff;
371         border-top: 1px dotted #ccc;
372         display: none;
373         margin-top: 0;
374         padding-bottom: 0;
375         padding-left: 0;
376         position: relative;
377         top: -1px;
```

```
378
379     li, .eac-category {
380         background: inherit;
381         border: {
382             color: #ccc;
383             image: none;
384             style: solid;
385             width: 0 1px;
386         }
387         display: block;
388         font: {
389             size: 14px;
390             weight: normal;
391         }
392         padding: 4px 12px;
393     }
394
395     li {
396
397         &:last-child {
398             border: {
399                 radius: 0 0 2px 2px;
400                 width: 0 1px 1px;
401             }
402         }
403
404         &.selected {
405             background: none repeat scroll 0 0 #ebebeb;
406             cursor: pointer;
407
408             div {
409                 font-weight: normal;
410             }
411         }
412
413         div {
414             display: block;
415             font-weight: normal;
416             word-break: break-all;
417         }
418
419         b {
420             font: {
421                 weight: bold;
422             }
423         }
424     }
425
426     .eac-category {
427         font: {
428             color: #aaa;
429             style: italic;
430         }
431     }
432
433 }
434
435 }
```

```
436
437 .eac-description {
438
439     .eac-item {
440
441         span {
442             color: #aaa;
443             font-style: italic;
444             font-size: 0.9em;
445         }
446     }
447 }
448
449 .eac-icon-left {
450
451     .eac-item {
452         img {
453             margin-right: 4px;
454             max-height: 30px;
455         }
456     }
457 }
458
459
460 .eac-icon-right {
461
462     .eac-item {
463         margin-top: 8px;
464         min-height: 24px;
465         position: relative;
466
467         img {
468             margin-left: 4px;
469             max-height: 30px;
470             position: absolute;
471             right: -4px;
472             top: -8px;
473         }
474     }
475 }
```

EasyAutocomplete-master/dist/easy-autocomplete.css

Listing 3.2: EasyAutocomplete-master/dist/easy-autocomplete.css

```
1  /*
2   * easy-autocomplete
3   * jQuery plugin for autocompletion
4   *
5   * @author Łukasz Pawełczak (http://github.com/pawelczak)
6   * @version 1.3.5
7   * Copyright License:
8   */
9
10 .easy-autocomplete {
11     position: relative;
```



```

12 }
13 .easy-autocomplete input {
14     border-color: #ccc;
15     border-radius: 4px;
16     border-style: solid;
17     border-width: 1px;
18     box-shadow: 0 1px 2px rgba(0, 0, 0, 0.1) inset;
19     color: #555;
20     float: none;
21     padding: 6px 12px;
22 }
23 .easy-autocomplete input:hover, .easy-autocomplete input:focus {
24     box-shadow: none;
25 }
26 .easy-autocomplete a {
27     display: block;
28 }
29 .easy-autocomplete.eac-blue-light input:hover, .easy-autocomplete.eac-blue-light_
↪input:focus {
30     border-color: #66afe9;
31     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(102, 175, 233, 0.6);
32 }
33 .easy-autocomplete.eac-blue-light ul {
34     border-color: #66afe9;
35     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(102, 175, 233, 0.6);
36 }
37 .easy-autocomplete.eac-blue-light ul li, .easy-autocomplete.eac-blue-light ul .eac-
↪category {
38     border-color: #66afe9;
39 }
40 .easy-autocomplete.eac-blue-light ul li.selected, .easy-autocomplete.eac-blue-light_
↪ul .eac-category.selected {
41     background-color: #ecf5fc;
42 }
43 .easy-autocomplete.eac-green-light input:hover, .easy-autocomplete.eac-green-light_
↪input:focus {
44     border-color: #41DB00;
45     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(146, 237, 107, 0.6);
46 }
47 .easy-autocomplete.eac-green-light ul {
48     border-color: #41DB00;
49     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(146, 237, 107, 0.6);
50 }
51 .easy-autocomplete.eac-green-light ul li, .easy-autocomplete.eac-green-light ul .eac-
↪category {
52     border-color: #41DB00;
53 }
54 .easy-autocomplete.eac-green-light ul li.selected, .easy-autocomplete.eac-green-light_
↪ul .eac-category.selected {
55     background-color: #9eff75;
56 }
57 .easy-autocomplete.eac-red-light input:hover, .easy-autocomplete.eac-red-light_
↪input:focus {
58     border-color: #ff5b5b;
59     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(255, 90, 90, 0.6);
60 }
61 .easy-autocomplete.eac-red-light ul {
62     border-color: #ff5b5b;

```

```

63     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(255, 90, 90, 0.6);
64 }
65 .easy-autocomplete.eac-red-light ul li, .easy-autocomplete.eac-red-light ul .eac-
↪category {
66     border-color: #ff5b5b;
67 }
68 .easy-autocomplete.eac-red-light ul li.selected, .easy-autocomplete.eac-red-light ul .
↪eac-category.selected {
69     background-color: #ff8e8e;
70 }
71 .easy-autocomplete.eac-yellow-light input:hover, .easy-autocomplete.eac-yellow-light
↪input:focus {
72     border-color: #ffdb00;
73     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(255, 231, 84, 0.6);
74 }
75 .easy-autocomplete.eac-yellow-light ul {
76     border-color: #ffdb00;
77     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(255, 231, 84, 0.6);
78 }
79 .easy-autocomplete.eac-yellow-light ul li, .easy-autocomplete.eac-yellow-light ul .
↪eac-category {
80     border-color: #ffdb00;
81 }
82 .easy-autocomplete.eac-yellow-light ul li.selected, .easy-autocomplete.eac-yellow-
↪light ul .eac-category.selected {
83     background-color: #ffe233;
84 }
85 .easy-autocomplete.eac-dark-light input:hover, .easy-autocomplete.eac-dark-light
↪input:focus {
86     border-color: #333;
87     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(55, 55, 55, 0.6);
88 }
89 .easy-autocomplete.eac-dark-light ul {
90     border-color: #333;
91     box-shadow: 0 1px 1px rgba(0, 0, 0, 0.075) inset, 0 0 8px rgba(55, 55, 55, 0.6);
92 }
93 .easy-autocomplete.eac-dark-light ul li, .easy-autocomplete.eac-dark-light ul .eac-
↪category {
94     border-color: #333;
95 }
96 .easy-autocomplete.eac-dark-light ul li.selected, .easy-autocomplete.eac-dark-light
↪ul .eac-category.selected {
97     background-color: #4d4d4d;
98     color: #fff;
99 }
100 .easy-autocomplete.eac-dark {
101     color: #fff;
102 }
103 .easy-autocomplete.eac-dark input {
104     background-color: #404040;
105     border-radius: 4px;
106     box-shadow: 0;
107     color: #f6f6f6;
108 }
109 .easy-autocomplete.eac-dark input:hover, .easy-autocomplete.eac-dark input:focus {
110     border-color: #333;
111     box-shadow: 0;
112 }

```

```

113 .easy-autocomplete.eac-dark ul {
114     border-color: #333;
115 }
116 .easy-autocomplete.eac-dark ul li, .easy-autocomplete.eac-dark ul .eac-category {
117     background-color: #404040;
118     border-color: #333;
119 }
120 .easy-autocomplete.eac-dark ul li.selected, .easy-autocomplete.eac-dark ul .eac-
121 ↪category.selected {
122     background-color: #737373;
123     color: #f6f6f6;
124 }
125 .easy-autocomplete.eac-dark-glass {
126     color: #fff;
127 }
128 .easy-autocomplete.eac-dark-glass input {
129     background-color: rgba(0, 0, 0, 0.8);
130     border-radius: 4px;
131     box-shadow: 0;
132     color: #f6f6f6;
133 }
134 .easy-autocomplete.eac-dark-glass input:hover, .easy-autocomplete.eac-dark-glass_
135 ↪input:focus {
136     border-color: rgba(0, 0, 0, 0.8);
137     box-shadow: 0;
138 }
139 .easy-autocomplete.eac-dark-glass ul {
140     border-color: rgba(0, 0, 0, 0.8);
141 }
142 .easy-autocomplete.eac-dark-glass ul li, .easy-autocomplete.eac-dark-glass ul .eac-
143 ↪category {
144     background-color: rgba(0, 0, 0, 0.8);
145     border-color: rgba(0, 0, 0, 0.8);
146 }
147 .easy-autocomplete.eac-dark-glass ul li.selected, .easy-autocomplete.eac-dark-glass_
148 ↪ul .eac-category.selected {
149     background-color: rgba(64, 64, 64, 0.8);
150     color: #f6f6f6;
151 }
152 .easy-autocomplete.eac-dark-glass ul li:last-child, .easy-autocomplete.eac-dark-glass_
153 ↪ul .eac-category:last-child {
154     border-radius: 0 0 4px 4px;
155 }
156 .easy-autocomplete.eac-blue {
157     color: #fff;
158 }
159 .easy-autocomplete.eac-blue input {
160     background-color: #6d9ed1;
161     border-radius: 4px;
162     box-shadow: 0;
163     color: #f6f6f6;
164 }
165 .easy-autocomplete.eac-blue input::-webkit-input-placeholder {
166     color: #f6f6f6;
167 }
168 .easy-autocomplete.eac-blue input:-moz-placeholder {
169     color: #f6f6f6;
170 }

```

```
166 .easy-autocomplete.eac-blue input::-moz-placeholder {
167     color: #f6f6f6;
168 }
169 .easy-autocomplete.eac-blue input:-ms-input-placeholder {
170     color: #f6f6f6;
171 }
172 .easy-autocomplete.eac-blue input:hover, .easy-autocomplete.eac-blue input:focus {
173     border-color: #5A91CB;
174     box-shadow: 0;
175 }
176 .easy-autocomplete.eac-blue ul {
177     border-color: #5A91CB;
178 }
179 .easy-autocomplete.eac-blue ul li, .easy-autocomplete.eac-blue ul .eac-category {
180     background-color: #6d9ed1;
181     border-color: #5A91CB;
182 }
183 .easy-autocomplete.eac-blue ul li.selected, .easy-autocomplete.eac-blue ul .eac-
184 ↪category.selected {
185     background-color: #94b8dd;
186     color: #f6f6f6;
187 }
188 .easy-autocomplete.eac-yellow {
189     color: #333;
190 }
191 .easy-autocomplete.eac-yellow input {
192     background-color: #ffdb7e;
193     border-color: #333;
194     border-radius: 4px;
195     box-shadow: 0;
196     color: #333;
197 }
198 .easy-autocomplete.eac-yellow input:hover, .easy-autocomplete.eac-yellow input:focus {
199     border-color: #333;
200     box-shadow: 0;
201 }
202 .easy-autocomplete.eac-yellow ul {
203     border-color: #333;
204 }
205 .easy-autocomplete.eac-yellow ul li, .easy-autocomplete.eac-yellow ul .eac-category {
206     background-color: #ffdb7e;
207     border-color: #333;
208 }
209 .easy-autocomplete.eac-yellow ul li.selected, .easy-autocomplete.eac-yellow ul .eac-
210 ↪category.selected {
211     background-color: #ffe9b1;
212     color: #333;
213 }
214 .easy-autocomplete.eac-purple {
215     color: #333;
216 }
217 .easy-autocomplete.eac-purple input {
218     background-color: #d6d1e7;
219     border-color: #b8afd5;
220     box-shadow: 0;
221     color: #333;
222 }
223 .easy-autocomplete.eac-purple input:hover, .easy-autocomplete.eac-purple input:focus {
```

```

222     border-color: #333;
223     box-shadow: 0;
224 }
225 .easy-autocomplete.eac-purple ul {
226     border-color: #333;
227 }
228 .easy-autocomplete.eac-purple ul li, .easy-autocomplete.eac-purple ul .eac-category {
229     background-color: #d6d1e7;
230     border-color: #333;
231 }
232 .easy-autocomplete.eac-purple ul li.selected, .easy-autocomplete.eac-purple ul .eac-
233 ↪category.selected {
234     background-color: #ebe8f3;
235     color: #333;
236 }
237 .easy-autocomplete.eac-bootstrap input {
238     border-color: #ccc;
239     border-radius: 4px;
240     border-style: solid;
241     border-width: 1px;
242     color: #555;
243     padding: 6px 12px;
244 }
245 .easy-autocomplete-container {
246     left: 0;
247     position: absolute;
248     width: 100%;
249     z-index: 2;
250 }
251 .easy-autocomplete-container ul {
252     background: none repeat scroll 0 0 #ffffff;
253     border-top: 1px dotted #ccc;
254     display: none;
255     margin-top: 0;
256     padding-bottom: 0;
257     padding-left: 0;
258     position: relative;
259     top: -1px;
260 }
261 .easy-autocomplete-container ul li, .easy-autocomplete-container ul .eac-category {
262     background: inherit;
263     border-color: #ccc;
264     border-image: none;
265     border-style: solid;
266     border-width: 0 1px;
267     display: block;
268     font-size: 14px;
269     font-weight: normal;
270     padding: 4px 12px;
271 }
272 .easy-autocomplete-container ul li:last-child {
273     border-radius: 0 0 2px 2px;
274     border-width: 0 1px 1px;
275 }
276 .easy-autocomplete-container ul li.selected {
277     background: none repeat scroll 0 0 #ebebeb;
278     cursor: pointer;

```

```
279 }
280 .easy-autocomplete-container ul li.selected div {
281     font-weight: normal;
282 }
283 .easy-autocomplete-container ul li div {
284     display: block;
285     font-weight: normal;
286     word-break: break-all;
287 }
288 .easy-autocomplete-container ul li b {
289     font-weight: bold;
290 }
291 .easy-autocomplete-container ul .eac-category {
292     font-color: #aaa;
293     font-style: italic;
294 }
295
296 .eac-description .eac-item span {
297     color: #aaa;
298     font-style: italic;
299     font-size: 0.9em;
300 }
301
302 .eac-icon-left .eac-item img {
303     margin-right: 4px;
304     max-height: 30px;
305 }
306
307 .eac-icon-right .eac-item {
308     margin-top: 8px;
309     min-height: 24px;
310     position: relative;
311 }
312 .eac-icon-right .eac-item img {
313     margin-left: 4px;
314     max-height: 30px;
315     position: absolute;
316     right: -4px;
317     top: -8px;
318 }
319
320 /*# sourceMappingURL=easy-autocomplete.css.map */
```

EasyAutocomplete-master/dist/jquery.easy-autocomplete.js

Listing 3.3: EasyAutocomplete-master/dist/jquery.easy-autocomplete.js

```
1  /*
2   * easy-autocomplete
3   * jQuery plugin for autocompletion
4   *
5   * @author Łukasz Pawełczak (http://github.com/pawelczak)
6   * @version 1.3.5
7   * Copyright License:
8   */
9
```

```
10  /*
11  * EasyAutocomplete - Configuration
12  */
13  var EasyAutocomplete = (function(scope){
14
15      scope.Configuration = function Configuration(options) {
16          var defaults = {
17              data: "list-required",
18              url: "list-required",
19              dataType: "json",
20
21              listLocation: function(data) {
22                  return data;
23              },
24
25              xmlElementName: "",
26
27              getValue: function(element) {
28                  return element;
29              },
30
31              autocompleteOff: true,
32
33              placeholder: false,
34
35              ajaxCallback: function() {},
36
37              matchResponseProperty: false,
38
39              list: {
40                  sort: {
41                      enabled: false,
42                      method: function(a, b) {
43                          a = defaults.getValue(a);
44                          b = defaults.getValue(b);
45                          if (a < b) {
46                              return -1;
47                          }
48                          if (a > b) {
49                              return 1;
50                          }
51                          return 0;
52                      }
53                  },
54
55                  maxNumberOfElements: 6,
56
57                  hideOnEmptyPhrase: true,
58
59                  match: {
60                      enabled: false,
61                      caseSensitive: false,
62                      method: function(element, phrase) {
63
64                          if (element.search(phrase) > -1) {
65                              return true;
66                          } else {
67                              return false;
```

```
68         }
69     }
70 },
71
72     showAnimation: {
73         type: "normal", //normal|slide|fade
74         time: 400,
75         callback: function() {}
76     },
77
78     hideAnimation: {
79         type: "normal",
80         time: 400,
81         callback: function() {}
82     },
83
84     /* Events */
85     onClickEvent: function() {},
86     onSelectItemEvent: function() {},
87     onLoadEvent: function() {},
88     onChooseEvent: function() {},
89     onKeyEnterEvent: function() {},
90     onMouseOverEvent: function() {},
91     onMouseOutEvent: function() {},
92     onShowListEvent: function() {},
93     onHideListEvent: function() {}
94 },
95
96     highlightPhrase: true,
97
98     theme: "",
99
100     cssClasses: "",
101
102     minCharNumber: 0,
103
104     requestDelay: 0,
105
106     adjustWidth: true,
107
108     ajaxSettings: {},
109
110     preparePostData: function(data, inputPhrase) {return data;},
111
112     loggerEnabled: true,
113
114     template: "",
115
116     categoriesAssigned: false,
117
118     categories: [{
119         maxNumberOfElements: 4
120     }]
121
122 };
123
124 var externalObjects = ["ajaxSettings", "template"];
125
```



```

126     this.get = function(propertyName) {
127         return defaults[propertyName];
128     };
129
130     this.equals = function(name, value) {
131         if (isAssigned(name)) {
132             if (defaults[name] === value) {
133                 return true;
134             }
135         }
136
137         return false;
138     };
139
140     this.checkDataUrlProperties = function() {
141         if (defaults.url === "list-required" && defaults.data ===
↪ "list-required") {
142             return false;
143         }
144         return true;
145     };
146     this.checkRequiredProperties = function() {
147         for (var propertyName in defaults) {
148             if (defaults[propertyName] === "required") {
149                 logger.error("Option " + propertyName + "
↪ must be defined");
150                 return false;
151             }
152         }
153         return true;
154     };
155
156     this.printPropertiesThatDoesntExist = function(consol,
↪ optionsToCheck) {
157         printPropertiesThatDoesntExist(consol, optionsToCheck);
158     };
159
160     prepareDefaults();
161
162     mergeOptions();
163
164     if (defaults.loggerEnabled === true) {
165         printPropertiesThatDoesntExist(console, options);
166     }
167
168     addAjaxSettings();
169
170     processAfterMerge();
171     function prepareDefaults() {
172
173         if (options.dataType === "xml") {
174
175             if (!options.getValue) {
176
177                 options.getValue = function(element) {
178                     return $(element).text();
179                 };
180             }

```

```

181         }
182
183
184         if (!options.list) {
185
186             options.list = {};
187         }
188
189         if (!options.list.sort) {
190             options.list.sort = {};
191         }
192
193
194         options.list.sort.method = function(a, b) {
195             a = options.getValue(a);
196             b = options.getValue(b);
197             if (a < b) {
198                 return -1;
199             }
200             if (a > b) {
201                 return 1;
202             }
203             return 0;
204         };
205
206         if (!options.list.match) {
207             options.list.match = {};
208         }
209
210         options.list.match.method = function(element, phrase)
↪ {
211
212             if (element.search(phrase) > -1) {
213                 return true;
214             } else {
215                 return false;
216             }
217         };
218
219     }
220     if (options.categories !== undefined && options.categories_
↪ instanceof Array) {
221
222         var categories = [];
223
224         for (var i = 0, length = options.categories.length; i
↪ < length; i += 1) {
225
226             var category = options.categories[i];
227
228             for (var property in defaults.categories[0]) {
229
230                 if (category[property] === undefined)
↪ {
231
232                     category[property] = defaults.
↪ categories[0][property];
233
234                 }
235             }
236         }
237     }

```

```

234         categories.push(category);
235     }
236
237     options.categories = categories;
238 }
239
240 }
241
242 function mergeOptions() {
243
244     defaults = mergeObjects(defaults, options);
245
246     function mergeObjects(source, target) {
247         var mergedObject = source || {};
248
249         for (var propertyName in source) {
250             if (target[propertyName] !== undefined &&
↪target[propertyName] !== null) {
251
252                 if (typeof target[propertyName] !==
↪"object" ||
253
254                     target[propertyName]
↪ instanceof Array) {
255                     mergedObject[propertyName] =
↪target[propertyName];
256                 } else {
257                     mergeObjects(source[propertyName],
↪ target[propertyName]);
258                 }
259             }
260
261             /* If data is an object */
262             if (target.data !== undefined && target.data !== null
↪ && typeof target.data === "object") {
263                 mergedObject.data = target.data;
264             }
265
266             return mergedObject;
267         }
268     }
269
270     function processAfterMerge() {
271
272         if (defaults.url !== "list-required" && typeof defaults.url !
↪== "function") {
273
274             var defaultUrl = defaults.url;
275             defaults.url = function() {
276                 return defaultUrl;
277             };
278         }
279
280         if (defaults.ajaxSettings.url !== undefined && typeof
↪defaults.ajaxSettings.url !== "function") {
281             var defaultUrl = defaults.ajaxSettings.url;
282             defaults.ajaxSettings.url = function() {
283                 return defaultUrl;

```

```

284         };
285     }
286
287     if (typeof defaults.listLocation === "string") {
288         var defaultlistLocation = defaults.listLocation;
289
290         if (defaults.dataType.toUpperCase() === "XML") {
291             defaults.listLocation = function(data) {
292                 return $(data).
↪find(defaultlistLocation);
293             };
294         } else {
295             defaults.listLocation = function(data) {
296                 return data[defaultlistLocation];
297             };
298         }
299     }
300
301     if (typeof defaults.getValue === "string") {
302         var defaultsGetValue = defaults.getValue;
303         defaults.getValue = function(element) {
304             return element[defaultsGetValue];
305         };
306     }
307
308     if (options.categories !== undefined) {
309         defaults.categoriesAssigned = true;
310     }
311
312 }
313
314 function addAjaxSettings() {
315
316     if (options.ajaxSettings !== undefined && typeof options.
↪ajaxSettings === "object") {
317         defaults.ajaxSettings = options.ajaxSettings;
318     } else {
319         defaults.ajaxSettings = {};
320     }
321
322 }
323
324 function isAssigned(name) {
325     if (defaults[name] !== undefined && defaults[name] !== null) {
326         return true;
327     } else {
328         return false;
329     }
330 }
331 function printPropertiesThatDoesntExist(console, optionsToCheck) {
332
333     checkPropertiesIfExist(defaults, optionsToCheck);
334
335     function checkPropertiesIfExist(source, target) {
336         for(var property in target) {
337             if (source[property] === undefined) {
338                 console.log("Property '" + property + "
↪' does not exist in EasyAutocomplete options API.");

```

```

339         }
340
341         if (typeof source[property] === "object" && $.
↪ inArray(property, externalObjects) === -1) {
342             checkPropertiesIfExist(source[property],
↪ target[property]);
343         }
344     }
345 }
346 }
347 };
348
349 return scope;
350
351 })(EasyAutocomplete || {});
352
353
354 /*
355  * EasyAutocomplete - Logger
356  */
357 var EasyAutocomplete = (function(scope) {
358
359     scope.Logger = function Logger() {
360
361         this.error = function(message) {
362             console.log("ERROR: " + message);
363         };
364
365         this.warning = function(message) {
366             console.log("WARNING: " + message);
367         };
368     };
369
370     return scope;
371
372 })(EasyAutocomplete || {});
373
374
375 /*
376  * EasyAutocomplete - Constants
377  */
378 var EasyAutocomplete = (function(scope) {
379
380     scope.Constants = function Constants() {
381         var constants = {
382             CONTAINER_CLASS: "easy-autocomplete-container",
383             CONTAINER_ID: "eac-container-",
384
385             WRAPPER_CSS_CLASS: "easy-autocomplete"
386         };
387
388         this.getValue = function(propertyName) {
389             return constants[propertyName];
390         };
391     };
392
393     return scope;
394

```

```
395 }) (EasyAutocomplete || {});
396
397
398 /*
399  * EasyAutocomplete - ListBuilderService
400  *
401  * @author Łukasz Pawełczak
402  *
403  */
404 var EasyAutocomplete = (function(scope) {
405
406     scope.ListBuilderService = function ListBuilderService(configuration,
407     ↪processResponseData) {
408
409         this.init = function(data) {
410             var listBuilder = [],
411                 builder = {};
412
413             builder.data = configuration.get("listLocation")(data);
414             builder.getValue = configuration.get("getValue");
415             builder.maxListSize = configuration.get("list").
416     ↪maxNumberOfElements;
417
418             listBuilder.push(builder);
419
420             return listBuilder;
421         };
422
423         this.updateCategories = function(listBuilder, data) {
424             if (configuration.get("categoriesAssigned")) {
425                 listBuilder = [];
426
427                 for(var i = 0; i < configuration.get("categories").
428     ↪length; i += 1) {
429
430                     var builder =
431     ↪convertToListBuilder(configuration.get("categories")[i], data);
432
433                     listBuilder.push(builder);
434                 }
435             }
436
437             return listBuilder;
438         };
439
440         this.convertXml = function(listBuilder) {
441             if(configuration.get("dataType").toUpperCase() === "XML") {
442                 for(var i = 0; i < listBuilder.length; i += 1) {
443                     listBuilder[i].data =
444     ↪convertXmlToList(listBuilder[i]);
445                 }
446             }
447         }
448     }
449 }
```

```

448         return listBuilder;
449     };
450
451     this.processData = function(listBuilder, inputPhrase) {
452         for(var i = 0, length = listBuilder.length; i < length; i+=1)
453         ↪ {
454             listBuilder[i].data = ↪
455             ↪processResponseData(configuration, listBuilder[i], inputPhrase);
456         }
457         return listBuilder;
458     };
459
460     this.checkIfDataExists = function(listBuilders) {
461         for(var i = 0, length = listBuilders.length; i < length; i += ↪
462         ↪1) {
463             if (listBuilders[i].data !== undefined && ↪
464             ↪listBuilders[i].data instanceof Array) {
465                 if (listBuilders[i].data.length > 0) {
466                     return true;
467                 }
468             }
469         }
470         return false;
471     };
472
473     function convertToListBuilder(category, data) {
474         var builder = {};
475
476         if(configuration.get("dataType").toUpperCase() === "XML") {
477             builder = convertXmlToListBuilder();
478         } else {
479             builder = convertDataToListBuilder();
480         }
481
482         if (category.header !== undefined) {
483             builder.header = category.header;
484         }
485
486         if (category.maxNumberOfElements !== undefined) {
487             builder.maxNumberOfElements = category.
488             ↪maxNumberOfElements;
489         }
490
491         if (configuration.get("list").maxNumberOfElements !== ↪
492         ↪undefined) {
493             builder.maxListSize = configuration.get("list").
494             ↪maxNumberOfElements;

```

```

500     }
501
502     if (category.getValue !== undefined) {
503
504         if (typeof category.getValue === "string") {
505             var defaultsGetValue = category.getValue;
506             builder.getValue = function(element) {
507                 return element[defaultsGetValue];
508             };
509         } else if (typeof category.getValue === "function") {
510             builder.getValue = category.getValue;
511         }
512
513     } else {
514         builder.getValue = configuration.get("getValue");
515     }
516
517
518     return builder;
519
520
521     function convertXmlToListBuilder() {
522
523         var builder = {},
524             listLocation;
525
526         if (category.xmlElementName !== undefined) {
527             builder.xmlElementName = category.
↵ xmlElementName;
528
529         }
530
531         if (category.listLocation !== undefined) {
532             listLocation = category.listLocation;
533         } else if (configuration.get("listLocation") !==
↵ undefined) {
534
535             listLocation = configuration.get("listLocation
↵ ");
536
537         }
538
539         if (listLocation !== undefined) {
540             if (typeof listLocation === "string") {
541                 builder.data = $(data).
↵ find(listLocation);
542
543             } else if (typeof listLocation === "function
↵ ") {
544
545                 builder.data = listLocation(data);
546             }
547         } else {
548             builder.data = data;
549         }
550
551         return builder;
552     }

```



```

553
554         function convertDataToListBuilder() {
555
556             var builder = {};
557
558             if (category.listLocation !== undefined) {
559
560                 if (typeof category.listLocation === "string
561 ↪") {
562                     builder.data = data[category.
563 ↪listLocation];
564
565                 } else if (typeof category.listLocation ===
566 ↪"function") {
567                     builder.data = category.
568 ↪listLocation(data);
569
570                 }
571             } else {
572                 builder.data = data;
573             }
574             return builder;
575         }
576
577         function convertXmlToList(builder) {
578             var simpleList = [];
579
580             if (builder.xmlElementName === undefined) {
581                 builder.xmlElementName = configuration.get(
582 ↪"xmlElementName");
583             }
584
585             $(builder.data).find(builder.xmlElementName).each(function() {
586                 simpleList.push(this);
587             });
588
589             return simpleList;
590         }
591
592         return scope;
593     })(EasyAutocomplete || {});
594
595     /*
596     * EasyAutocomplete - Data process module
597     *
598     * Process list to display:
599     * - sort
600     * - decrease number to specific number
601     * - show only matching list
602     *
603     */
604     var EasyAutocomplete = (function(scope) {
605

```

```

606     scope.process = function processData(config, listBuilder, phrase) {
607
608         scope.process.match = match;
609
610         var list = listBuilder.data,
611             inputPhrase = phrase; //TODO REFACTOR
612
613         list = findMatch(list, inputPhrase);
614         list = reduceElementsInList(list);
615         list = sort(list);
616
617         return list;
618
619         function findMatch(list, phrase) {
620             var preparedList = [],
621                 value = "";
622
623             if (config.get("list").match.enabled) {
624
625                 for(var i = 0, length = list.length; i < length; i += 1) {
626
627                     value = config.get("getValue")(list[i]);
628
629                     if (match(value, phrase)) {
630                         preparedList.push(list[i]);
631                     }
632
633                 }
634
635             } else {
636                 preparedList = list;
637             }
638
639             return preparedList;
640         }
641
642         function match(value, phrase) {
643
644             if (!config.get("list").match.caseSensitive) {
645
646                 if (typeof value === "string") {
647                     value = value.toLowerCase();
648                 }
649
650                 phrase = phrase.toLowerCase();
651             }
652             if (config.get("list").match.method(value, phrase)) {
653                 return true;
654             } else {
655                 return false;
656             }
657         }
658
659         function reduceElementsInList(list) {
660             if (listBuilder.maxNumberOfElements !== undefined && list.
661                 length > listBuilder.maxNumberOfElements) {

```

```

662         list = list.slice(0, listBuilder.maxNumberOfElements);
663     }
664
665     return list;
666 }
667
668 function sort(list) {
669     if (config.get("list").sort.enabled) {
670         list.sort(config.get("list").sort.method);
671     }
672
673     return list;
674 }
675
676 };
677
678
679 return scope;
680
681
682 })(EasyAutocomplete || {});
683
684
685 /*
686  * EasyAutocomplete - Template
687  *
688  *
689  *
690  */
691 var EasyAutocomplete = (function(scope) {
692
693     scope.Template = function Template(options) {
694
695
696         var genericTemplates = {
697             basic: {
698                 type: "basic",
699                 method: function(element) { return element; },
700                 cssClass: ""
701             },
702             description: {
703                 type: "description",
704                 fields: {
705                     description: "description"
706                 },
707                 method: function(element) { return element + "
708 ↪ description"; },
709                 cssClass: "eac-description"
710             },
711             iconLeft: {
712                 type: "iconLeft",
713                 fields: {
714                     icon: ""
715                 },
716                 method: function(element) {
717                     return element;
718                 },
719                 cssClass: "eac-icon-left"

```

```

    },
    iconRight: {
        type: "iconRight",
        fields: {
            iconSrc: ""
        },
        method: function(element) {
            return element;
        },
        cssClass: "eac-icon-right"
    },
    links: {
        type: "links",
        fields: {
            link: ""
        },
        method: function(element) {
            return element;
        },
        cssClass: ""
    },
    custom: {
        type: "custom",
        method: function() {},
        cssClass: ""
    }
},

/*
 * Converts method with {{text}} to function
 */
convertTemplateToMethod = function(template) {

    var _fields = template.fields,
        buildMethod;

    if (template.type === "description") {

        buildMethod = genericTemplates.description.method;

        if (typeof _fields.description === "string") {
            buildMethod = function(elementValue, element)
↵{
                return elementValue + " - <span>" + _
↵element[_fields.description] + "</span>";
            };
        } else if (typeof _fields.description === "function")
↵{
            buildMethod = function(elementValue, element)
↵{
                return elementValue + " - <span>" + _
↵fields.description(element) + "</span>";
            };
        }
    }
}

```

```

772         return buildMethod;
773     }
774
775     if (template.type === "iconRight") {
776
777         if (typeof _fields.iconSrc === "string") {
778             buildMethod = function(elementValue, element)
↪ {
779                 return elementValue + "<img class=
↪ 'eac-icon' src='" + element[_fields.iconSrc] + "' />" ;
780             };
781         } else if (typeof _fields.iconSrc === "function") {
782             buildMethod = function(elementValue, element)
↪ {
783                 return elementValue + "<img class=
↪ 'eac-icon' src='" + _fields.iconSrc(element) + "' />" ;
784             };
785         }
786
787         return buildMethod;
788     }
789
790
791     if (template.type === "iconLeft") {
792
793         if (typeof _fields.iconSrc === "string") {
794             buildMethod = function(elementValue, element)
↪ {
795                 return "<img class='eac-icon' src='" +
↪ + element[_fields.iconSrc] + "' />" + elementValue;
796             };
797         } else if (typeof _fields.iconSrc === "function") {
798             buildMethod = function(elementValue, element)
↪ {
799                 return "<img class='eac-icon' src='" +
↪ + _fields.iconSrc(element) + "' />" + elementValue;
800             };
801         }
802
803         return buildMethod;
804     }
805
806     if(template.type === "links") {
807
808         if (typeof _fields.link === "string") {
809             buildMethod = function(elementValue, element)
↪ {
810                 return "<a href='" + element[_fields.
↪ link] + "' >" + elementValue + "</a>";
811             };
812         } else if (typeof _fields.link === "function") {
813             buildMethod = function(elementValue, element)
↪ {
814                 return "<a href='" + _fields.
↪ link(element) + "' >" + elementValue + "</a>";
815             };
816         }
817

```

```

818         return buildMethod;
819     }
820
821     if (template.type === "custom") {
822
823         return template.method;
824     }
825
826     return genericTemplates.basic.method;
827
828 },
829
830
831
832 prepareBuildMethod = function(options) {
833     if (!options || !options.type) {
834
835         return genericTemplates.basic.method;
836     }
837
838     if (options.type && genericTemplates[options.type]) {
839
840         return convertTemplateToMethod(options);
841     } else {
842
843         return genericTemplates.basic.method;
844     }
845
846 },
847
848 templateClass = function(options) {
849     var emptyStringFunction = function() {return ""};
850
851     if (!options || !options.type) {
852
853         return emptyStringFunction;
854     }
855
856     if (options.type && genericTemplates[options.type]) {
857         return (function () {
858             var _cssClass = genericTemplates[options.
859 ↪type].cssClass;
860
861             return function() { return _cssClass;};
862         })();
863     } else {
864         return emptyStringFunction;
865     }
866
867 };
868
869 this.getTemplateClass = templateClass(options);
870
871 this.build = prepareBuildMethod(options);
872
873 };
874
875 return scope;

```

```

875 }) (EasyAutocomplete || {});
876
877
878
879 /*
880  * EasyAutocomplete - jQuery plugin for autocompletion
881  *
882  */
883 var EasyAutocomplete = (function(scope) {
884
885
886     scope.main = function Core($input, options) {
887
888         var module = {
889             name: "EasyAutocomplete",
890             shortcut: "eac"
891         };
892
893         var consts = new scope.Constants(),
894             config = new scope.Configuration(options),
895             logger = new scope.Logger(),
896             template = new scope.Template(options.template),
897             listBuilderService = new scope.ListBuilderService(config,
898 ↪scope.proccess),
899             checkParam = config.equals,
900             $field = $input,
901             $container = "",
902             elementsList = [],
903             selectedElement = -1,
904             requestDelayTimeoutId;
905
906         scope.consts = consts;
907
908         this.getConstants = function() {
909             return consts;
910         };
911
912         this.getConfiguration = function() {
913             return config;
914         };
915
916         this.getContainer = function() {
917             return $container;
918         };
919
920         this.getSelectedItemIndex = function() {
921             return selectedElement;
922         };
923
924         this.getItems = function () {
925             return elementsList;
926         };
927
928         this.getItemData = function(index) {
929
930             if (elementsList.length < index || elementsList[index] ===
931 ↪undefined) {

```

```

931         return -1;
932     } else {
933         return elementsList[index];
934     }
935 };
936
937 this.getSelectedItemData = function() {
938     return this.getItemData(selectedElement);
939 };
940
941 this.build = function() {
942     prepareField();
943 };
944
945 this.init = function() {
946     init();
947 };
948 function init() {
949
950     if ($field.length === 0) {
951         logger.error("Input field doesn't exist.");
952         return;
953     }
954
955     if (!config.checkDataUrlProperties()) {
956         logger.error("One of options variables 'data' or 'url
↪ ' must be defined.");
957         return;
958     }
959
960     if (!config.checkRequiredProperties()) {
961         logger.error("Will not work without mentioned_
↪ properties.");
962         return;
963     }
964
965     prepareField();
966     bindEvents();
967
968 }
969
970 function prepareField() {
971
972
973     if ($field.parent().hasClass(consts.getValue("WRAPPER_CSS_
↪ CLASS")))) {
974         removeContainer();
975         removeWrapper();
976     }
977
978     createWrapper();
979     createContainer();
980
981     $container = $("#" + getContainerId());
982     if (config.get("placeholder")) {
983         $field.attr("placeholder", config.get("placeholder"));
984     }
985

```



```

986
987     function createWrapper() {
988         var $wrapper = $("<div>"),
989             classes = consts.getValue("WRAPPER_CSS_CLASS
↪");
990
991
992         if (config.get("theme") && config.get("theme") !== "
↪") {
993             classes += " eac-" + config.get("theme");
994         }
995
996         if (config.get("cssClasses") && config.get("cssClasses
↪") !== "") {
997             classes += " " + config.get("cssClasses");
998         }
999
1000         if (template.getTemplateClass() !== "") {
1001             classes += " " + template.getTemplateClass();
1002         }
1003
1004
1005         $wrapper
1006             .addClass(classes);
1007         $field.wrap($wrapper);
1008
1009
1010         if (config.get("adjustWidth") === true) {
1011             adjustWrapperWidth();
1012         }
1013
1014     }
1015
1016
1017     function adjustWrapperWidth() {
1018         var fieldWidth = $field.outerWidth();
1019
1020         $field.parent().css("width", fieldWidth);
1021     }
1022
1023     function removeWrapper() {
1024         $field.unwrap();
1025     }
1026
1027     function createContainer() {
1028         var $elements_container = $("<div>").addClass(consts.
↪getValue("CONTAINER_CLASS"));
1029
1030         $elements_container
1031             .attr("id", getContainerId())
1032             .prepend $("<ul>");
1033
1034
1035         (function() {
1036
1037             $elements_container
1038                 /* List show animation */
1039                 .on("show.eac", function() {

```

```

1041                                     switch (config.get("list").
↪ showAnimation.type) {
1042
1043                                     case "slide":
1044                                         var_
↪ animationTime = config.get("list").showAnimation.time,
1045                                     callback_
↪ = config.get("list").showAnimation.callback;
1046                                     $selements_
1047 ↪ container.find("ul").slideDown(animationTime, callback);
1048                                     break;
1049
1050                                     case "fade":
1051                                         var_
↪ animationTime = config.get("list").showAnimation.time,
1052                                     callback_
↪ = config.get("list").showAnimation.callback;
1053                                     $selements_
1054 ↪ container.find("ul").fadeIn(animationTime), callback;
1055                                     break;
1056
1057                                     default:
1058                                         $selements_
↪ container.find("ul").show();
1059                                     break;
1060                                     }
1061                                     config.get("list").
↪ onShowListEvent();
1062
1063                                     })
1064                                     /* List hide animation */
1065                                     .on("hide.eac", function() {
1066
1067                                     switch (config.get("list").
↪ hideAnimation.type) {
1068
1069                                     case "slide":
1070                                         var_
↪ animationTime = config.get("list").hideAnimation.time,
1071                                     callback_
↪ = config.get("list").hideAnimation.callback;
1072                                     $selements_
1073 ↪ container.find("ul").slideUp(animationTime, callback);
1074                                     break;
1075
1076                                     case "fade":
1077                                         var_
↪ animationTime = config.get("list").hideAnimation.time,
1078                                     callback_
↪ = config.get("list").hideAnimation.callback;
1079                                     $selements_
1080 ↪ container.find("ul").fadeOut(animationTime, callback);
1081

```

```

1082                                     break;
1083
1084                                     default:
1085                                         $elements_
1086
1087                                     break;
1088                                     }
1089                                     config.get("list").
1090
1091                                     })
1092                                     .on("selectElement.eac", function() {
1093                                         $elements_container.find("ul_
1094
1095                                     $elements_container.find("ul_
1096                                     config.get("list").
1097                                     })
1098                                     .on("loadElements.eac", _
1099
1100                                     function(event, listBuilders, phrase) {
1101
1102                                     var $item = "",
1103                                         $listContainer =
1104
1105                                     $listContainer
1106                                         .empty()
1107                                         .detach();
1108                                     elementsList = [];
1109                                     var counter = 0;
1110                                     for(var builderIndex = 0, _
1111
1112                                     listBuildersLength = listBuilders.length; builderIndex < listBuildersLength; _
1113                                     builderIndex += 1) {
1114
1115                                     var listData = _
1116
1117                                     if (listData.length_
1118                                     continue;
1119                                     }
1120                                     if_
1121                                     (listBuilders[builderIndex].header !== undefined && listBuilders[builderIndex].
1122                                     header.length > 0) {
1123
1124                                     $listContainer.append("<div class='eac-category' >" + listBuilders[builderIndex].
1125                                     header + "</div>");
1126                                     }
1127                                     for(var i = 0, _
1128
1129                                     listDataLength = listData.length; i < listDataLength && counter <_
1130                                     listBuilders[builderIndex].maxListSize; i += 1) {

```

1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157

```

    <li><div class='eac-item'></div></li>>";
    (function() {
        var j
        i = i,
        counter,
        listBuilders[builderIndex].getValue(listData[j]);
        $item.
        .
        .
        on("click", function() {
            $field.val(elementsValue).trigger("change");
            itemCounter;
            selected;
            selected;
            configuration;
            configuration;
        })
        .
        .
        mouseover(function() {
            itemCounter;
            selected;
            selected;
            configuration;
        })
        .
        mouseout(function() {
            configuration;
        })
        .
        html(template.build(highlight(elementsValue, phrase), listData[j]));
    })();
    $listContainer.append($item);
    elementsList.
    push(listData[i]);
    counter += 1;

```

```

1158                                     }
1159                                     }
1160                                     $elements_container.append(
1161 ↪ $listContainer);
1162                                     config.get("list").
1163 ↪ onLoadEvent();
1164                                     });
1165                                     })();
1166                                     $field.after($elements_container);
1167                                     }
1168                                     function removeContainer() {
1169                                     $field.next("." + consts.getValue("CONTAINER_CLASS")).
1170 ↪ remove();
1171                                     }
1172                                     function highlight(string, phrase) {
1173                                     if(config.get("highlightPhrase") && phrase !== "") {
1174                                     return highlightPhrase(string, phrase);
1175                                     } else {
1176                                     return string;
1177                                     }
1178                                     }
1179                                     function escapeRegExp(str) {
1180                                     return str.replace(/\-\[ \] \{ \} \[ \] \* \+ \? \. \\ \^ \
1181 ↪ \$ \[ \] /g, "\\$&");
1182                                     }
1183                                     function highlightPhrase(string, phrase) {
1184                                     var escapedPhrase = escapeRegExp(phrase);
1185                                     return (string + "").replace(new RegExp("(" + _
1186 ↪ escapedPhrase + ")", "gi"), "<b>$1</b>");
1187                                     }
1188                                     function getContainerId() {
1189                                     var elementId = $field.attr("id");
1190                                     elementId = consts.getValue("CONTAINER_ID") + elementId;
1191                                     return elementId;
1192                                     }
1193                                     function bindEvents() {
1194                                     bindAllEvents();
1195                                     }
1196                                     function bindAllEvents() {

```

```

1211         if (checkParam("autocompleteOff", true)) {
1212             removeAutocomplete();
1213         }
1214
1215         bindFocusOut();
1216         bindKeyup();
1217         bindKeydown();
1218         bindKeyPress();
1219         bindFocus();
1220         bindBlur();
1221     }
1222
1223     function bindFocusOut() {
1224         $field.focusout(function () {
1225
1226             var fieldValue = $field.val(),
1227                 phrase;
1228
1229             if (!config.get("list").match.caseSensitive) {
1230                 fieldValue = fieldValue.toLowerCase();
1231             }
1232
1233             for (var i = 0, length = elementsList.length;
1234 ↪ i < length; i += 1) {
1235
1236                 phrase = config.get("getValue
1237 ↪ ") (elementsList[i]);
1238
1239                 if (!config.get("list").match.
1240 ↪ caseSensitive) {
1241                     phrase = phrase.toLowerCase();
1242                 }
1243
1244                 if (phrase === fieldValue) {
1245                     selectedElement = i;
1246                     selectElement(selectedElement);
1247 ↪
1248                     return;
1249                 }
1250             }
1251         });
1252     }
1253
1254     function bindKeyup() {
1255         $field
1256         .off("keyup")
1257         .keyup(function(event) {
1258
1259             switch(event.keyCode) {
1260
1261                 case 27:
1262
1263                     hideContainer();
1264                     loseFieldFocus();
1265
1266                     break;
1267
1268                 case 38:
1269
1270                     event.preventDefault();

```

```

1265
1266                                     if(elementsList.length > 0 &&
↪selectedElement > 0) {
1267
1268                                     selectedElement -= 1;
1269
1270                                     $field.val(config.get(
↪"getValue")(elementsList[selectedElement]));
1271
1272                                     selectElement(selectedElement);
↪
1273
1274                                     }
1275                                     break;
1276
1277                                     case 40:
1278
1279                                     event.preventDefault();
1280
1281                                     if(elementsList.length > 0 &&
↪selectedElement < elementsList.length - 1) {
1282
1283                                     selectedElement += 1;
1284
1285                                     $field.val(config.get(
↪"getValue")(elementsList[selectedElement]));
1286
1287                                     selectElement(selectedElement);
↪
1288
1289                                     }
1290
1291                                     break;
1292
1293                                     default:
1294
1295                                     if (event.keyCode > 40 ||
↪event.keyCode === 8) {
1296
1297                                     var inputPhrase =
↪$field.val();
1298
1299                                     if (!(config.get("list
↪").hideOnEmptyPhrase === true && event.keyCode === 8 && inputPhrase === "")) {
1300
1301                                     if (config.
↪
1302                                     if
↪
1303                                     clearTimeout
1304                                     }
1305
1306                                     requestDelayTimeoutId
↪= setTimeout(function () { loadData(inputPhrase);}, config.get("requestDelay"));
1307                                     } else {
1308                                     loadData(inputPhrase);
↪

```

```

1309                                     }
1310
1311                                     } else {
1312                                     hideContainer();
1313
1314                                     }
1315
1316                                     }
1317
1318                                     break;
1319
1320                                     }
1321
1322                                     function loadData(inputPhrase) {
1323
1324
1325                                     if (inputPhrase.length < config.get(
1326                                     ↪ "minCharNumber")) {
1327                                     return;
1328                                     }
1329
1330                                     if (config.get("data") !== "list-
1331                                     ↪ required") {
1332
1333                                     var data = config.get("data");
1334
1335                                     var listBuilders = _
1336                                     ↪ listBuilderService.init(data);
1337
1338                                     listBuilders = _
1339                                     ↪ listBuilderService.updateCategories(listBuilders, data);
1340
1341                                     listBuilders = _
1342                                     ↪ listBuilderService.processData(listBuilders, inputPhrase);
1343
1344                                     loadElements(listBuilders, _
1345                                     ↪ inputPhrase);
1346
1347                                     if ($field.parent().find("li
1348                                     ↪ ").length > 0) {
1349                                     showContainer();
1350                                     } else {
1351                                     hideContainer();
1352                                     }
1353
1354                                     }
1355
1356                                     var settings = createAjaxSettings();
1357
1358                                     if (settings.url === undefined || _
1359                                     ↪ settings.url === "") {
1360                                     settings.url = config.get("url
1361                                     ↪ ");
1362                                     }
1363
1364                                     if (settings.dataType === undefined,
1365                                     ↪ || settings.dataType === "") {

```



```

1357                                     settings.dataType = config.
↪get("dataType");
1358                                     }
1359
1360
1361                                     if (settings.url !== undefined &&
↪settings.url !== "list-required") {
1362
1363                                     settings.url = settings.
↪url(inputPhrase);
1364
1365                                     settings.data = config.get(
↪"preparePostData")(settings.data, inputPhrase);
1366
1367                                     $.ajax(settings)
1368                                         .done(function(data) {
1369
1370                                             var
↪listBuilders = listBuilderService.init(data);
1371
1372                                             listBuilders
↪= listBuilderService.updateCategories(listBuilders, data);
1373
1374                                             listBuilders
↪= listBuilderService.convertXml(listBuilders);
1375                                             if
↪(checkInputPhraseMatchResponse(inputPhrase, data)) {
1376
1377                                             listBuilders
↪= listBuilderService.processData(listBuilders, inputPhrase);
1378
1379                                             loadElements(listBui
↪inputPhrase);
1380
1381                                             }
1382
1383                                             if
↪(listBuilderService.checkIfDataExists(listBuilders) && $field.parent().find("li").
↪length > 0) {
1384
1385                                             showContainer();
1386                                             } else {
1387                                             hideContainer();
1388
1389                                             }
1390
1391                                             config.get(
↪"ajaxCallback")();
1392
1393                                             })
1394                                             .fail(function() {
1395                                                 logger.
1396
1397                                                 warning("Fail to load response data");
1398
1399                                             })
1400                                             .always(function() {
1401
1402                                             });
1403
1404                                     }

```

```

1399
1400
1401
1402         function createAjaxSettings() {
1403
1404             var settings = {},
1405                 ajaxSettings = config.
↪get("ajaxSettings") || {};
1406
1407             for (var set in ajaxSettings)
↪{
1408                 settings[set] =
↪ajaxSettings[set];
1409             }
1410
1411             return settings;
1412         }
1413
1414         function
↪checkInputPhraseMatchResponse(inputPhrase, data) {
1415
1416             if (config.get(
↪"matchResponseProperty") !== false) {
1417                 if (typeof config.get(
↪"matchResponseProperty") === "string") {
1418                     return
↪(data[config.get("matchResponseProperty")] === inputPhrase);
1419                 }
1420
1421                 if (typeof config.get(
↪"matchResponseProperty") === "function") {
1422                     return
↪(config.get("matchResponseProperty")(data) === inputPhrase);
1423                 }
1424
1425                 return true;
1426             } else {
1427                 return true;
1428             }
1429
1430         }
1431
1432     }
1433
1434
1435     });
1436
1437
1438     function bindKeydown() {
1439         $field
1440             .on("keydown", function(evt) {
1441                 evt = evt || window.event;
1442                 var keyCode = evt.keyCode;
1443                 if (keyCode === 38) {
1444                     suppressKeypress = true;
1445                     return false;
1446                 }
1447             })

```

```

1448         .keydown(function(event) {
1449
1450             if (event.keyCode === 13 &&
↪selectedElement > -1) {
1451
1452                 $field.val(config.get(
↪"getValue")(elementsList[selectedElement]));
1453
1454                 config.get("list").
1455                 config.get("list").
↪onKeyEnterEvent();
1456
↪onChooseEvent();
1457
1458                 selectedElement = -1;
1459                 hideContainer();
1460                 event.preventDefault();
1461             }
1462         });
1463     }
1464
1465     function bindKeypress() {
1466         $field
1467         .off("keypress");
1468     }
1469
1470     function bindFocus() {
1471         $field.focus(function() {
1472
1473             if ($field.val() !== "" && elementsList.
↪length > 0) {
1474
1475                 selectedElement = -1;
1476                 showContainer();
1477             }
1478
1479             });
1480     }
1481
1482     function bindBlur() {
1483         $field.blur(function() {
1484             setTimeout(function() {
1485
1486                 selectedElement = -1;
1487                 hideContainer();
1488             }, 250);
1489         });
1490     }
1491
1492     function removeAutocomplete() {
1493         $field.attr("autocomplete", "off");
1494     }
1495
1496 }
1497
1498     function showContainer() {
1499         $container.trigger("show.eac");
1500     }

```

```

1501         function hideContainer() {
1502             $container.trigger("hide.eac");
1503         }
1504
1505         function selectElement(index) {
1506             $container.trigger("selectElement.eac", index);
1507         }
1508
1509         function loadElements(list, phrase) {
1510             $container.trigger("loadElements.eac", [list, phrase]);
1511         }
1512
1513         function loseFieldFocus() {
1514             $field.trigger("blur");
1515         }
1516
1517     };
1518     scope.eacHandles = [];
1519
1520     scope.getHandle = function(id) {
1521         return scope.eacHandles[id];
1522     };
1523
1524     scope.inputHasId = function(input) {
1525         if($(input).attr("id") !== undefined && $(input).attr("id").length >
1526         ↪ 0) {
1527             return true;
1528         } else {
1529             return false;
1530         }
1531     };
1532
1533     scope.assignRandomId = function(input) {
1534         var fieldId = "";
1535
1536         do {
1537             fieldId = "eac-" + Math.floor(Math.random() * 10000);
1538         } while ($("#" + fieldId).length !== 0);
1539
1540         elementId = scope.consts.getValue("CONTAINER_ID") + fieldId;
1541
1542         $(input).attr("id", fieldId);
1543     };
1544
1545     scope.setHandle = function(handle, id) {
1546         scope.eacHandles[id] = handle;
1547     };
1548
1549     return scope;
1550
1551
1552
1553
1554
1555
1556
1557

```

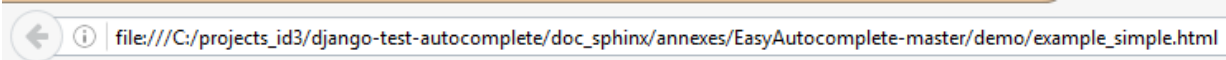
```

1558 }) (EasyAutocomplete || {});
1559
1560 (function($) {
1561
1562     $.fn.easyAutocomplete = function(options) {
1563
1564         return this.each(function() {
1565             var $this = $(this),
1566                 eacHandle = new EasyAutocomplete.main($this, options);
1567
1568             if (!EasyAutocomplete.inputHasId($this)) {
1569                 EasyAutocomplete.assignRandomId($this);
1570             }
1571
1572             eacHandle.init();
1573
1574             EasyAutocomplete.setHandle(eacHandle, $this.attr("id"));
1575
1576         });
1577     };
1578
1579     $.fn.getSelectedItemId = function() {
1580
1581         var inputId = $(this).attr("id");
1582
1583         if (inputId !== undefined) {
1584             return EasyAutocomplete.getHandle(inputId).
↪getSelectedItemId();
1585         }
1586
1587         return -1;
1588     };
1589
1590     $.fn.getItems = function () {
1591
1592         var inputId = $(this).attr("id");
1593
1594         if (inputId !== undefined) {
1595             return EasyAutocomplete.getHandle(inputId).getItems();
1596         }
1597
1598         return -1;
1599     };
1600
1601     $.fn.getItemData = function(index) {
1602
1603         var inputId = $(this).attr("id");
1604
1605         if (inputId !== undefined && index > -1) {
1606             return EasyAutocomplete.getHandle(inputId).getItemData(index);
1607         }
1608
1609         return -1;
1610     };
1611
1612     $.fn.getSelectedItemData = function() {
1613
1614         var inputId = $(this).attr("id");

```

```
1615         if (inputId !== undefined) {
1616             return EasyAutocomplete.getHandle(inputId) .
1617 ↪getSelectedItemData();
1618         }
1619
1620         return -1;
1621     };
1622
1623 })(jQuery);
```

EasyAutocomplete simple example



EasyAutocomplete - easy example



EasyAutocomplete-master/demo/example_simple.html

Listing 3.4: EasyAutocomplete-master/demo/example_simple.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3      <head>
4          <title>EasyAutocomplete simple example</title>
5
6          <!-- STYLE -> CSS -->
7          <link href="../dist/easy-autocomplete.min.css" rel="stylesheet" type=
8 ↪"text/css">
9          <!-- end STYLE-->
10
11      </head>
12      <body>
13
14          <!-- STRUCTURE -> HTML5 elements -->
15          <h1>EasyAutocomplete - easy example</h1>
16          <input id="simple" />
17          <!-- end STRUCTURE-->
18
19          <!--BEHAVIOR -> Javascript scripts-->
```

```

19     <script src="../../lib/jquery-1.11.2.min.js"></script>
20     <script src="../../dist/jquery.easy-autocomplete.min.js" type="text/
↪ javascript" ></script>
21     <script>
22
23         var options_easy_autocomplete = {
24             data: ["blue", "green", "pink", "red", "yellow"],
25
26             };
27
28         $("#simple").easyAutocomplete(options_easy_autocomplete);
29
30     </script>
31     <!-- end BEHAVIOR -->
32
33 </body>
34
35 </html>

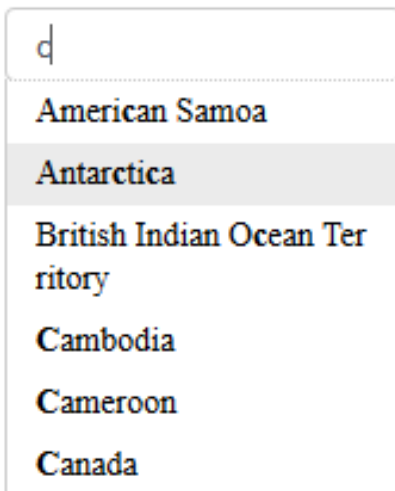
```

EasyAutocomplete JSON example

See also:

JSON

EasyAutocomplete - json example



EasyAutocomplete-master/demo/example_json.html

Listing 3.5: EasyAutocomplete-master/demo/example_json.html

```

1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>EasyAutocomplete json example</title>
5
6     <!-- STYLE -> CSS -->
7     <link href="../dist/easy-autocomplete.min.css" rel="stylesheet" type=
8 ↪ "text/css">
9     <!-- end STYLE-->
10  </head>
11  <body>
12
13    <!-- STRUCTURE -> HTML5 elements -->
14    <h1>EasyAutocomplete - json example</h1>
15    <input id="data-json" />
16    <!-- end STRUCTURE-->
17
18    <!-- BEHAVIOR -> Javascript scripts-->
19    <script src="../lib/jquery-1.11.2.min.js"></script>
20    <script src="../dist/jquery.easy-autocomplete.min.js" type="text/
21 ↪ javascript" ></script>
22    <script>
23      let options_easy_autocomplete = {
24        url: "resources/countries.json",
25
26        getValue: "name",
27
28        list: {
29          match: {
30            enabled: true
31          }
32        }
33      };
34      $("#data-json").easyAutocomplete(options_easy_autocomplete);
35    </script>
36    <!-- end BEHAVIOR -->
37  </body>
38 </html>

```

EasyAutocomplete-master/demo/resources/countries.json

Listing 3.6: EasyAutocomplete-master/demo/resources/countries.json

```

1 [
2   { "name": "Afghanistan", "code": "AF" },
3   { "name": "Albania", "code": "AL" },
4   { "name": "Algeria", "code": "DZ" },
5   { "name": "American Samoa", "code": "AS" },
6   { "name": "Andorra", "code": "AD" },
7   { "name": "Angola", "code": "AO" },
8   { "name": "Anguilla", "code": "AI" },
9   { "name": "Antarctica", "code": "AQ" },
10  { "name": "Antigua and Barbuda", "code": "AG" },

```



```

11  {"name": "Argentina", "code": "AR"},
12  {"name": "Armenia", "code": "AM"},
13  {"name": "Aruba", "code": "AW"},
14  {"name": "Australia", "code": "AU"},
15  {"name": "Austria", "code": "AT"},
16  {"name": "Azerbaijan", "code": "AZ"},
17  {"name": "Bahamas", "code": "BS"},
18  {"name": "Bahrain", "code": "BH"},
19  {"name": "Bangladesh", "code": "BD"},
20  {"name": "Barbados", "code": "BB"},
21  {"name": "Belarus", "code": "BY"},
22  {"name": "Belgium", "code": "BE"},
23  {"name": "Belize", "code": "BZ"},
24  {"name": "Benin", "code": "BJ"},
25  {"name": "Bermuda", "code": "BM"},
26  {"name": "Bhutan", "code": "BT"},
27  {"name": "Bolivia", "code": "BO"},
28  {"name": "Bosnia and Herzegovina", "code": "BA"},
29  {"name": "Botswana", "code": "BW"},
30  {"name": "Bouvet Island", "code": "BV"},
31  {"name": "Brazil", "code": "BR"},
32  {"name": "British Indian Ocean Territory", "code": "IO"},
33  {"name": "Brunei Darussalam", "code": "BN"},
34  {"name": "Bulgaria", "code": "BG"},
35  {"name": "Burkina Faso", "code": "BF"},
36  {"name": "Burundi", "code": "BI"},
37  {"name": "Cambodia", "code": "KH"},
38  {"name": "Cameroon", "code": "CM"},
39  {"name": "Canada", "code": "CA"},
40  {"name": "Cape Verde", "code": "CV"},
41  {"name": "Cayman Islands", "code": "KY"},
42  {"name": "Central African Republic", "code": "CF"},
43  {"name": "Chad", "code": "TD"},
44  {"name": "Chile", "code": "CL"},
45  {"name": "China", "code": "CN"},
46  {"name": "Christmas Island", "code": "CX"},
47  {"name": "Cocos (Keeling) Islands", "code": "CC"},
48  {"name": "Colombia", "code": "CO"},
49  {"name": "Comoros", "code": "KM"},
50  {"name": "Congo", "code": "CG"},
51  {"name": "Congo, The Democratic Republic of the", "code": "CD"},
52  {"name": "Cook Islands", "code": "CK"},
53  {"name": "Costa Rica", "code": "CR"},
54  {"name": "Cote D'Ivoire", "code": "CI"},
55  {"name": "Croatia", "code": "HR"},
56  {"name": "Cuba", "code": "CU"},
57  {"name": "Cyprus", "code": "CY"},
58  {"name": "Czech Republic", "code": "CZ"},
59  {"name": "Denmark", "code": "DK"},
60  {"name": "Djibouti", "code": "DJ"},
61  {"name": "Dominica", "code": "DM"},
62  {"name": "Dominican Republic", "code": "DO"},
63  {"name": "Ecuador", "code": "EC"},
64  {"name": "Egypt", "code": "EG"},
65  {"name": "El Salvador", "code": "SV"},
66  {"name": "Equatorial Guinea", "code": "GQ"},
67  {"name": "Eritrea", "code": "ER"},
68  {"name": "Estonia", "code": "EE"},

```

```

69  {"name": "Ethiopia", "code": "ET"},
70  {"name": "Falkland Islands (Malvinas)", "code": "FK"},
71  {"name": "Faroe Islands", "code": "FO"},
72  {"name": "Fiji", "code": "FJ"},
73  {"name": "Finland", "code": "FI"},
74  {"name": "France", "code": "FR"},
75  {"name": "French Guiana", "code": "GF"},
76  {"name": "French Polynesia", "code": "PF"},
77  {"name": "French Southern Territories", "code": "TF"},
78  {"name": "Gabon", "code": "GA"},
79  {"name": "Gambia", "code": "GM"},
80  {"name": "Georgia", "code": "GE"},
81  {"name": "Germany", "code": "DE"},
82  {"name": "Ghana", "code": "GH"},
83  {"name": "Gibraltar", "code": "GI"},
84  {"name": "Greece", "code": "GR"},
85  {"name": "Greenland", "code": "GL"},
86  {"name": "Grenada", "code": "GD"},
87  {"name": "Guadeloupe", "code": "GP"},
88  {"name": "Guam", "code": "GU"},
89  {"name": "Guatemala", "code": "GT"},
90  {"name": "Guernsey", "code": "GG"},
91  {"name": "Guinea", "code": "GN"},
92  {"name": "Guinea-Bissau", "code": "GW"},
93  {"name": "Guyana", "code": "GY"},
94  {"name": "Haiti", "code": "HT"},
95  {"name": "Heard Island and Mcdonald Islands", "code": "HM"},
96  {"name": "Holy See (Vatican City State)", "code": "VA"},
97  {"name": "Honduras", "code": "HN"},
98  {"name": "Hong Kong", "code": "HK"},
99  {"name": "Hungary", "code": "HU"},
100 {"name": "Iceland", "code": "IS"},

```

EasyAutocomplete flags example

EasyAutocomplete-master/demo/example_flags.html

Listing 3.7: EasyAutocomplete-master/demo/example_flags.html

```

1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <title>EasyAutocomplete flags example</title>
5
6      <!-- STYLE -> CSS -->
7      <link href="../../dist/easy-autocomplete.min.css" rel="stylesheet" type=
8  ↪ "text/css">
9      <link href="resources/flags.css" rel="stylesheet" type="text/css" >
10     <!-- end STYLE-->
11
12   </head>
13   <body>
14     <!-- STRUCTURE -> HTML5 elements -->
15     <h1>EasyAutocomplete - country flags example</h1>
16
17     <input id="flags" />

```

EasyAutocomplete - country flags example



```

17     <!-- end STRUCTURE-->
18
19     <!--BEHAVIOR -> Javascript scripts-->
20         <script src="../../lib/jquery-1.11.2.min.js"></script>
21         <script src="../../dist/jquery.easy-autocomplete.min.js" type="text/
→ javascript" ></script>
22         <script>
23             var options_easy_autocomplete = {
24                 url: "resources/countries.json",
25
26                 getValue: "name",
27
28                 list: {
29                     match: {
30                         enabled: true
31                     },
32                     maxNumberOfElements: 10
33                 },
34
35                 template: {
36                     type: "custom",
37                     method: function(value, item) {
38 → toLowerCase() + " " ></span>" + value;
39                     }
40                 }
41             };
42
43             $("#flags").easyAutocomplete(options_easy_autocomplete);
44         </script>
45     <!-- end BEHAVIOR -->
46 </body>
47

```

`</html>`

EasyAutocomplete-master/demo/resources/flags.css

Listing 3.8: EasyAutocomplete-master/demo/resources/flags.css

```

1  .flag {
2      width: 16px;
3      height: 11px;
4      background: url(flags.png) no-repeat;
5      display: inline-block;
6      margin-right: 5px;
7  }
8
9  .flag.flag-ad {background-position: -16px 0}
10 .flag.flag-ae {background-position: -32px 0}
11 .flag.flag-af {background-position: -48px 0}
12 .flag.flag-ag {background-position: -64px 0}
13 .flag.flag-ai {background-position: -80px 0}
14 .flag.flag-al {background-position: -96px 0}
15 .flag.flag-am {background-position: -112px 0}
16 .flag.flag-an {background-position: -128px 0}
17 .flag.flag-ao {background-position: -144px 0}
18 .flag.flag-ar {background-position: -160px 0}
19 .flag.flag-as {background-position: -176px 0}
20 .flag.flag-at {background-position: -192px 0}
21 .flag.flag-au {background-position: -208px 0}
22 .flag.flag-aw {background-position: -224px 0}
23 .flag.flag-az {background-position: -240px 0}
24 .flag.flag-ba {background-position: 0 -11px}
25 .flag.flag-bb {background-position: -16px -11px}
26 .flag.flag-bd {background-position: -32px -11px}
27 .flag.flag-be {background-position: -48px -11px}
28 .flag.flag-bf {background-position: -64px -11px}
29 .flag.flag-bg {background-position: -80px -11px}
30 .flag.flag-bh {background-position: -96px -11px}
31 .flag.flag-bi {background-position: -112px -11px}
32 .flag.flag-bj {background-position: -128px -11px}
33 .flag.flag-bm {background-position: -144px -11px}
34 .flag.flag-bn {background-position: -160px -11px}
35 .flag.flag-bo {background-position: -176px -11px}
36 .flag.flag-br {background-position: -192px -11px}
37 .flag.flag-bs {background-position: -208px -11px}
38 .flag.flag-bt {background-position: -224px -11px}
39 .flag.flag-bv {background-position: -240px -11px}
40 .flag.flag-bw {background-position: 0 -22px}
41 .flag.flag-by {background-position: -16px -22px}
42 .flag.flag-bz {background-position: -32px -22px}
43 .flag.flag-ca {background-position: -48px -22px}
44 .flag.flag-catalonia {background-position: -64px -22px}
45 .flag.flag-cd {background-position: -80px -22px}
46 .flag.flag-cf {background-position: -96px -22px}
47 .flag.flag-cg {background-position: -112px -22px}
48 .flag.flag-ch {background-position: -128px -22px}
49 .flag.flag-ci {background-position: -144px -22px}
50 .flag.flag-ck {background-position: -160px -22px}

```

```

51 .flag.flag-cl {background-position: -176px -22px}
52 .flag.flag-cm {background-position: -192px -22px}
53 .flag.flag-cn {background-position: -208px -22px}
54 .flag.flag-co {background-position: -224px -22px}
55 .flag.flag-cr {background-position: -240px -22px}
56 .flag.flag-cu {background-position: 0 -33px}
57 .flag.flag-cv {background-position: -16px -33px}
58 .flag.flag-cw {background-position: -32px -33px}
59 .flag.flag-cy {background-position: -48px -33px}
60 .flag.flag-cz {background-position: -64px -33px}
61 .flag.flag-de {background-position: -80px -33px}
62 .flag.flag-dj {background-position: -96px -33px}
63 .flag.flag-dk {background-position: -112px -33px}
64 .flag.flag-dm {background-position: -128px -33px}
65 .flag.flag-do {background-position: -144px -33px}
66 .flag.flag-dz {background-position: -160px -33px}
67 .flag.flag-ec {background-position: -176px -33px}
68 .flag.flag-ee {background-position: -192px -33px}
69 .flag.flag-eg {background-position: -208px -33px}
70 .flag.flag-eh {background-position: -224px -33px}
71 .flag.flag-england {background-position: -240px -33px}
72 .flag.flag-er {background-position: 0 -44px}
73 .flag.flag-es {background-position: -16px -44px}
74 .flag.flag-et {background-position: -32px -44px}
75 .flag.flag-eu {background-position: -48px -44px}
76 .flag.flag-fi {background-position: -64px -44px}
77 .flag.flag-fj {background-position: -80px -44px}
78 .flag.flag-fk {background-position: -96px -44px}
79 .flag.flag-fm {background-position: -112px -44px}
80 .flag.flag-fo {background-position: -128px -44px}
81 .flag.flag-fr {background-position: -144px -44px}
82 .flag.flag-ga {background-position: -160px -44px}
83 .flag.flag-gb {background-position: -176px -44px}
84 .flag.flag-gd {background-position: -192px -44px}
85 .flag.flag-ge {background-position: -208px -44px}
86 .flag.flag-gf {background-position: -224px -44px}
87 .flag.flag-gg {background-position: -240px -44px}
88 .flag.flag-gh {background-position: 0 -55px}
89 .flag.flag-gi {background-position: -16px -55px}
90 .flag.flag-gl {background-position: -32px -55px}
91 .flag.flag-gm {background-position: -48px -55px}
92 .flag.flag-gn {background-position: -64px -55px}
93 .flag.flag-gp {background-position: -80px -55px}
94 .flag.flag-gq {background-position: -96px -55px}
95 .flag.flag-gr {background-position: -112px -55px}
96 .flag.flag-gs {background-position: -128px -55px}
97 .flag.flag-gt {background-position: -144px -55px}
98 .flag.flag-gu {background-position: -160px -55px}
99 .flag.flag-gw {background-position: -176px -55px}
100 .flag.flag-gy {background-position: -192px -55px}
101 .flag.flag-hk {background-position: -208px -55px}
102 .flag.flag-hm {background-position: -224px -55px}
103 .flag.flag-hn {background-position: -240px -55px}
104 .flag.flag-hr {background-position: 0 -66px}
105 .flag.flag-ht {background-position: -16px -66px}
106 .flag.flag-hu {background-position: -32px -66px}
107 .flag.flag-ic {background-position: -48px -66px}
108 .flag.flag-id {background-position: -64px -66px}

```

```

109 .flag.flag-ie {background-position: -80px -66px}
110 .flag.flag-il {background-position: -96px -66px}
111 .flag.flag-in {background-position: -112px -66px}
112 .flag.flag-in {background-position: -128px -66px}
113 .flag.flag-io {background-position: -144px -66px}
114 .flag.flag-iq {background-position: -160px -66px}
115 .flag.flag-ir {background-position: -176px -66px}
116 .flag.flag-is {background-position: -192px -66px}
117 .flag.flag-it {background-position: -208px -66px}
118 .flag.flag-je {background-position: -224px -66px}
119 .flag.flag-jm {background-position: -240px -66px}
120 .flag.flag-jo {background-position: 0 -77px}
121 .flag.flag-jp {background-position: -16px -77px}
122 .flag.flag-ke {background-position: -32px -77px}
123 .flag.flag-kg {background-position: -48px -77px}
124 .flag.flag-kh {background-position: -64px -77px}
125 .flag.flag-ki {background-position: -80px -77px}
126 .flag.flag-km {background-position: -96px -77px}
127 .flag.flag-kn {background-position: -112px -77px}
128 .flag.flag-kp {background-position: -128px -77px}
129 .flag.flag-kr {background-position: -144px -77px}
130 .flag.flag-kurdistan {background-position: -160px -77px}
131 .flag.flag-kw {background-position: -176px -77px}
132 .flag.flag-ky {background-position: -192px -77px}
133 .flag.flag-kz {background-position: -208px -77px}
134 .flag.flag-la {background-position: -224px -77px}
135 .flag.flag-lb {background-position: -240px -77px}
136 .flag.flag-lc {background-position: 0 -88px}
137 .flag.flag-li {background-position: -16px -88px}
138 .flag.flag-lk {background-position: -32px -88px}
139 .flag.flag-lr {background-position: -48px -88px}
140 .flag.flag-ls {background-position: -64px -88px}
141 .flag.flag-lt {background-position: -80px -88px}
142 .flag.flag-lu {background-position: -96px -88px}
143 .flag.flag-lv {background-position: -112px -88px}
144 .flag.flag-ly {background-position: -128px -88px}
145 .flag.flag-ma {background-position: -144px -88px}
146 .flag.flag-mc {background-position: -160px -88px}
147 .flag.flag-md {background-position: -176px -88px}
148 .flag.flag-me {background-position: -192px -88px}
149 .flag.flag-mg {background-position: -208px -88px}
150 .flag.flag-mh {background-position: -224px -88px}
151 .flag.flag-mk {background-position: -240px -88px}
152 .flag.flag-ml {background-position: 0 -99px}
153 .flag.flag-mm {background-position: -16px -99px}
154 .flag.flag-mn {background-position: -32px -99px}
155 .flag.flag-mo {background-position: -48px -99px}
156 .flag.flag-mp {background-position: -64px -99px}
157 .flag.flag-mq {background-position: -80px -99px}
158 .flag.flag-mr {background-position: -96px -99px}
159 .flag.flag-ms {background-position: -112px -99px}
160 .flag.flag-mt {background-position: -128px -99px}
161 .flag.flag-mu {background-position: -144px -99px}
162 .flag.flag-mv {background-position: -160px -99px}
163 .flag.flag-mw {background-position: -176px -99px}
164 .flag.flag-mx {background-position: -192px -99px}
165 .flag.flag-my {background-position: -208px -99px}
166 .flag.flag-mz {background-position: -224px -99px}

```



```

167 .flag.flag-na {background-position: -240px -99px}
168 .flag.flag-nc {background-position: 0 -110px}
169 .flag.flag-ne {background-position: -16px -110px}
170 .flag.flag-nf {background-position: -32px -110px}
171 .flag.flag-ng {background-position: -48px -110px}
172 .flag.flag-ni {background-position: -64px -110px}
173 .flag.flag-nl {background-position: -80px -110px}
174 .flag.flag-no {background-position: -96px -110px}
175 .flag.flag-np {background-position: -112px -110px}
176 .flag.flag-nr {background-position: -128px -110px}
177 .flag.flag-nu {background-position: -144px -110px}
178 .flag.flag-nz {background-position: -160px -110px}
179 .flag.flag-om {background-position: -176px -110px}
180 .flag.flag-pa {background-position: -192px -110px}
181 .flag.flag-pe {background-position: -208px -110px}
182 .flag.flag-pf {background-position: -224px -110px}
183 .flag.flag-pg {background-position: -240px -110px}
184 .flag.flag-ph {background-position: 0 -121px}
185 .flag.flag-pk {background-position: -16px -121px}
186 .flag.flag-pl {background-position: -32px -121px}
187 .flag.flag-pm {background-position: -48px -121px}
188 .flag.flag-pn {background-position: -64px -121px}
189 .flag.flag-pr {background-position: -80px -121px}
190 .flag.flag-ps {background-position: -96px -121px}
191 .flag.flag-pt {background-position: -112px -121px}
192 .flag.flag-pw {background-position: -128px -121px}
193 .flag.flag-py {background-position: -144px -121px}
194 .flag.flag-qa {background-position: -160px -121px}
195 .flag.flag-re {background-position: -176px -121px}
196 .flag.flag-ro {background-position: -192px -121px}
197 .flag.flag-cs {background-position: -208px -121px}
198 .flag.flag-ru {background-position: -224px -121px}
199 .flag.flag-rw {background-position: -240px -121px}
200 .flag.flag-sa {background-position: 0 -132px}
201 .flag.flag-sb {background-position: -16px -132px}
202 .flag.flag-sc {background-position: -32px -132px}
203 .flag.flag-scotland {background-position: -48px -132px}
204 .flag.flag-sd {background-position: -64px -132px}
205 .flag.flag-se {background-position: -80px -132px}
206 .flag.flag-sg {background-position: -96px -132px}
207 .flag.flag-sh {background-position: -112px -132px}
208 .flag.flag-si {background-position: -128px -132px}
209 .flag.flag-sk {background-position: -144px -132px}
210 .flag.flag-sl {background-position: -160px -132px}
211 .flag.flag-sm {background-position: -176px -132px}
212 .flag.flag-sn {background-position: -192px -132px}
213 .flag.flag-so {background-position: -208px -132px}
214 .flag.flag-somaliland {background-position: -224px -132px}
215 .flag.flag-sr {background-position: -240px -132px}
216 .flag.flag-ss {background-position: 0 -143px}
217 .flag.flag-st {background-position: -16px -143px}
218 .flag.flag-sv {background-position: -32px -143px}
219 .flag.flag-sx {background-position: -48px -143px}
220 .flag.flag-sy {background-position: -64px -143px}
221 .flag.flag-sz {background-position: -80px -143px}
222 .flag.flag-tc {background-position: -96px -143px}
223 .flag.flag-td {background-position: -112px -143px}
224 .flag.flag-tf {background-position: -128px -143px}

```

```

225 .flag.flag-tg {background-position: -144px -143px}
226 .flag.flag-th {background-position: -160px -143px}
227 .flag.flag-tj {background-position: -176px -143px}
228 .flag.flag-tk {background-position: -192px -143px}
229 .flag.flag-tl {background-position: -208px -143px}
230 .flag.flag-tm {background-position: -224px -143px}
231 .flag.flag-tn {background-position: -240px -143px}
232 .flag.flag-to {background-position: 0 -154px}
233 .flag.flag-tr {background-position: -16px -154px}
234 .flag.flag-tt {background-position: -32px -154px}
235 .flag.flag-tv {background-position: -48px -154px}
236 .flag.flag-tw {background-position: -64px -154px}
237 .flag.flag-tz {background-position: -80px -154px}
238 .flag.flag-ua {background-position: -96px -154px}
239 .flag.flag-ug {background-position: -112px -154px}
240 .flag.flag-um {background-position: -128px -154px}
241 .flag.flag-us {background-position: -144px -154px}
242 .flag.flag-uy {background-position: -160px -154px}
243 .flag.flag-uz {background-position: -176px -154px}
244 .flag.flag-va {background-position: -192px -154px}
245 .flag.flag-vc {background-position: -208px -154px}
246 .flag.flag-ve {background-position: -224px -154px}
247 .flag.flag-vg {background-position: -240px -154px}
248 .flag.flag-vi {background-position: 0 -165px}
249 .flag.flag-vn {background-position: -16px -165px}
250 .flag.flag-vu {background-position: -32px -165px}
251 .flag.flag-wales {background-position: -48px -165px}
252 .flag.flag-wf {background-position: -64px -165px}
253 .flag.flag-ws {background-position: -80px -165px}
254 .flag.flag-xk {background-position: -96px -165px}
255 .flag.flag-ye {background-position: -112px -165px}
256 .flag.flag-yt {background-position: -128px -165px}
257 .flag.flag-za {background-position: -144px -165px}
258 .flag.flag-zanzibar {background-position: -160px -165px}
259 .flag.flag-zm {background-position: -176px -165px}
260 .flag.flag-zw {background-position: -192px -165px}

```

EasyAutocomplete-master/demo/resources/flags.png

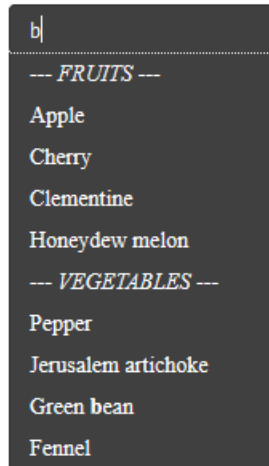


Fig. 3.2: EasyAutocomplete-master/demo/resources/flags.png

EasyAutocomplete categories example

file:///C:/projects_id3/django-test-autocomplete/doc_sphinx/annexes/EasyAutocomplete-master/demo/example_categories.html

EasyAutocomplete - categories example



EasyAutocomplete-master/demo/example_categories.html

Listing 3.9: EasyAutocomplete-master/demo/example_categories.html

```

1  <!DOCTYPE html>
2  <html lang="en">
3      <head>
4          <title>EasyAutocomplete categories</title>
5          <!-- STYLE -> CSS -->
6          <link href="../../dist/easy-autocomplete.min.css" rel="stylesheet" type=
7  ↪ "text/css">
8          <link href="../../dist/easy-autocomplete.themes.min.css" rel="stylesheet"
9  ↪ type="text/css">
10         <link href="//fonts.googleapis.com/css?family=Roboto:400,300,300italic,
11 ↪ 400italic,600,600italic&subset=latin,cyrillic-ext,greek-ext,latin-ext,cyrillic"
12 ↪ rel='stylesheet' type='text/css'>
13         <!-- end STYLE-->
14     </head>
15     <body>
16         <!-- STRUCTURE -> HTML5 elements -->
17         <h1>EasyAutocomplete - categories example</h1>
18         <input id="data-categories" placeholder="Fruits and vegetables" />
19         <!-- end STRUCTURE-->
20
21         <!-- BEHAVIOR -> Javascript scripts-->
22         <script src="../../lib/jquery-1.11.2.min.js"></script>
23         <script src="../../dist/jquery.easy-autocomplete.min.js" type="text/
24 ↪ javascript"></script>
25         <script>

```

```
21     var options_easy_autocomplete = {
22         url: "resources/categories.json",
23
24         categories: [{
25             listLocation: "fruits",
26             header: "--- FRUITS ---"
27         }, {
28             listLocation: "vegetables",
29             header: "--- VEGETABLES ---"
30         }],
31
32         list: {
33             match: {
34                 enabled: false
35             },
36             maxNumberOfElements: 10
37         },
38
39         theme: "dark"
40     };
41
42     $("#data-categories").easyAutocomplete(options_easy_autocomplete);
43
44     </script>
45     <!-- end BEHAVIOR -->
46 </body>
47
48 </html>
```

EasyAutocomplete-master/demo/resources/categories.json

Listing 3.10: EasyAutocomplete-master/demo/resources/categories.json

```
1 {
2     "fruits": ["Apple", "Cherry", "Clementine", "Honeydew melon", "Watermelon",
3     ↪ "Satsuma"],
4     "vegetables": ["Pepper", "Jerusalem artichoke", "Green bean", "Fennel",
5     ↪ "Courgette", "Yam"]
6 }
```

EasyAutocomplete static_link example

EasyAutocomplete-master/demo/example_static_link.html

Listing 3.11: EasyAutocomplete-master/demo/example_static_link.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3     <head>
4         <title>EasyAutocomplete static links</title>
5         <!-- STYLE -> CSS -->
6         <link href="../dist/easy-autocomplete.min.css" rel="stylesheet" type=
7     ↪ "text/css">
8         <!-- end STYLE-->
```



```

8      </head>
9      <body>
10
11          <!-- STRUCTURE -> HTML5 elements -->
12          <h1>EasyAutocomplete - static links</h1>
13          <input id="data-links" />
14          <!-- end STRUCTURE-->
15
16
17          <!--BEHAVIOR -> Javascript scripts-->
18          <script src="../../lib/jquery-1.11.2.min.js"></script>
19          <script src="../../dist/jquery.easy-autocomplete.min.js" type="text/
↪ javascript" ></script>
20          <script>
21              var options_easy_autocomplete = {
22                  url: "resources/site.json",
23
24                  getValue: "text",
25
26                  template: {
27                      type: "links",
28                      fields: {
29                          link: "site"
30                      }
31                  },
32
33                  list: {
34                      match: {
35                          enabled: false
36                      }
37                  },
38
39                  theme: "plate-dark"
40              };
41
42              $("#data-links").easyAutocomplete(options_easy_autocomplete);
43
44          </script>
45          <!-- end BEHAVIOR -->
46      </body>

```

```
47 </html>
```

EasyAutocomplete-master/demo/resources/site.json

Listing 3.12: EasyAutocomplete-master/demo/resources/site.json

```
1 [{
2   "text": "Home",
3   "site": "http://easyautocomplete.com"
4 },
5 {
6   "text": "Guide",
7   "site": "http://easyautocomplete.com/guide"
8 },
9 {
10  "text": "Examples",
11  "site": "http://easyautocomplete.com/examples"
12 },
13 {
14  "text": "Themes",
15  "site": "http://easyautocomplete.com/themes"
16 },
17 {
18  "text": "Download",
19  "site": "http://easyautocomplete.com/download"
20 }
21 ]
```

EasyAutocomplete email example

file:///C:/projects_id3/django-test-autocomplete/doc_sphinx/annexes/EasyAutocomplete-master/demo/example_email.html

EasyAutocomplete - email example

enim.Curabitur.massa@t

enim.Curabitur.massa@
convalliserat.edu - enim.
Curabitur.massa@convallis
erat.edu

Mauris.nulla@Curabitur
.net - Mauris.nulla@Cura
bitur.net

EasyAutocomplete-master/demo/example_email.html

Listing 3.13: EasyAutocomplete-master/demo/example_email.html

```

1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>EasyAutocomplete email example</title>
5     <!-- STYLE -> CSS -->
6     <link href="../../dist/easy-autocomplete.min.css" rel="stylesheet" type=
7     ↪ "text/css">
8     <!-- end STYLE-->
9   </head>
10   <body>
11     <!-- STRUCTURE -> HTML5 elements -->
12     <h1>EasyAutocomplete - email example</h1>
13     <input id="data-email" />
14     <!-- end STRUCTURE-->
15
16     <!-- BEHAVIOR -> Javascript scripts-->
17     <script src="../../lib/jquery-1.11.2.min.js"></script>
18     <script src="../../dist/jquery.easy-autocomplete.min.js" type="text/
19     ↪ javascript"></script>
20     <script>
21       var options_easy_autocomplete = {
22         url: "resources/people.json",
23
24         getValue: "email",
25
26         template: {
27           type: "description",
28           fields: {
29             description: "email"
30           }
31         },
32
33         list: {
34           match: {
35             enabled: true
36           }
37         },
38
39         theme: "plate-dark"
40       };
41
42       $("#data-email").easyAutocomplete(options_easy_autocomplete);
43     </script>
44     <!-- end BEHAVIOR -->
45
46   </body>
47 </html>

```

EasyAutocomplete-master/demo/resources/people.json

Listing 3.14: EasyAutocomplete-master/demo/resources/people.json

```
1  [ {
2      "name": "Declan Haley2 s",
3      "email": "Cr12as.lorem.lorem@nonquam.ca"
4  },
5  {
6      "name": "Francis Marsh",
7      "email": "ne2que@arcu.edu"
8  },
9  {
10     "name": "Gage Figueroa",
11     "email": "Sed.auctor.odio@magnis.ca"
12 },
13 {
14     "name": "Asher Gay",
15     "email": "Phasellus@nonsapien.ca"
16 },
17 {
18     "name": "Erasmus Roach",
19     "email": "amet@aptent.net"
20 },
21 {
22     "name": "Francis Johnston",
23     "email": "Ut.sagittis.lobortis@Donecsollicitudin.com"
24 },
25 {
26     "name": "Dustin Mckinney",
27     "email": "velit.Quisque@liberodui.co.uk"
28 },
29 {
30     "name": "Zane Camacho",
31     "email": "et.netus@Phasellusvitae.co.uk"
32 },
33 {
34     "name": "Murphy Larsen",
35     "email": "sit.amet@felisadipiscing.com"
36 },
37 {
38     "name": "Shad Kemp",
39     "email": "justo@laoreetlectus.co.uk"
40 },
41 {
42     "name": "Octavius Wolfe",
43     "email": "sociis@aptenttacitisociosqu.org"
44 },
45 {
46     "name": "Cooper Bell",
47     "email": "Quisque@apurus.ca"
48 },
49 {
50     "name": "Lev Mccarthy",
51     "email": "diam@amet.co.uk"
52 },
53 {
54     "name": "Arthur Mejia",
55     "email": "Nulla.dignissim@urna.ca"
56 },
```

```
57 {
58     "name": "Ferris Cameron",
59     "email": "vulputate.velit@In.net"
60 },
61 {
62     "name": "Ivor Greer",
63     "email": "nisi@Nuncmauriselit.ca"
64 },
65 {
66     "name": "Alan Church",
67     "email": "et.ipsuam@dolor.co.uk"
68 },
69 {
70     "name": "Walter Berry",
71     "email": "neque@lacus.edu"
72 },
73 {
74     "name": "Hyatt Bowman",
75     "email": "lobortis.ultrices@mollis.org"
76 },
77 {
78     "name": "Acton Bradley",
79     "email": "commodo.tincidunt@ut.com"
80 },
81 {
82     "name": "Duncan Hancock",
83     "email": "Donec@sem.net"
84 },
85 {
86     "name": "Macaulay Trujillo",
87     "email": "in.sodales.elit@Donec.ca"
88 },
89 {
90     "name": "Jack Estrada",
91     "email": "nibh@Quisquetinciduntpede.net"
92 },
93 {
94     "name": "Edward Wilkerson",
95     "email": "Quisque.fringilla.euismod@idliberoDonec.co.uk"
96 },
97 {
98     "name": "Clarke Mullen",
99     "email": "nunc.Quisque.ornare@leo.com"
100 },
101 {
102     "name": "Steel Rodriguez",
103     "email": "a@diam.ca"
104 },
105 {
106     "name": "Jerome Edwards",
107     "email": "tincidunt@consectetuer.net"
108 },
109 {
110     "name": "Quentin Blevins",
111     "email": "In.scelerisque@idrisus.co.uk"
112 },
113 {
114     "name": "Maxwell Baxter",
```

```
115     "email": "lacinia.mattis@erateget.ca"
116 },
117 {
118     "name": "Beau McClain",
119     "email": "dis.parturient@feugiatnon.co.uk"
120 },
121 {
122     "name": "Jarrod Valenzuela",
123     "email": "adipiscing@FuscefeugiatLorem.net"
124 },
125 {
126     "name": "Grant Rosario",
127     "email": "feugiat@rutrumFuscedolor.co.uk"
128 },
129 {
130     "name": "Wade Atkinson",
131     "email": "et@euerosNam.com"
132 },
133 {
134     "name": "Ryder Lindsey",
135     "email": "Quisque.fringilla.euismod@Suspendissealiquetmolestie.edu"
136 },
137 {
138     "name": "Upton Schmidt",
139     "email": "arcu.Nunc.mauris@quis.ca"
140 },
141 {
142     "name": "Seth Tate",
143     "email": "elit.a.feugiat@turpisnon.edu"
144 },
145 {
146     "name": "Emery Shields",
147     "email": "mi.Aliquam.gravida@vestibulummassarutrum.net"
148 },
149 {
150     "name": "Slade Bolton",
151     "email": "venenatis.a@fringillami.co.uk"
152 },
153 {
154     "name": "Lucas Yates",
155     "email": "purus.in@idnuncinterdum.ca"
156 },
157 {
158     "name": "Dean Rosa",
159     "email": "vulputate.ullamcorper@commodoipsum.ca"
160 },
161 {
162     "name": "Ahmed Daniel",
163     "email": "ligula@idsapien.ca"
164 },
165 {
166     "name": "Hector Hardin",
167     "email": "egestas@Nullam.edu"
168 },
169 {
170     "name": "Byron Dickerson",
171     "email": "sociis@Proinultrices.co.uk"
172 },
```



```
173 {
174     "name": "Myles Workman",
175     "email": "magnis.dis@odioPhasellusat.co.uk"
176 },
177 {
178     "name": "Jerry Howard",
179     "email": "tortor.nibh@sitamet.edu"
180 },
181 {
182     "name": "Igor Chase",
183     "email": "sagittis@gravida.edu"
184 },
185 {
186     "name": "Aquila Mcclure",
187     "email": "dictum@Phasellusinfelis.co.uk"
188 },
189 {
190     "name": "John Hansen",
191     "email": "ut@molestietellus.com"
192 },
193 {
194     "name": "Forrest Goodwin",
195     "email": "enim.nec.tempus@massalobortis.com"
196 },
197 {
198     "name": "Griffin Lott",
199     "email": "enim.Curabitur.massa@convalliserat.edu"
200 },
```

EasyAutocomplete Django integration

See also:

- <http://easyautocomplete.com/guide>

Include JS and CSS files from the distribution (static files)

See also:

- <http://easyautocomplete.com/download>
- <https://docs.djangoproject.com/en/dev/howto/static-files/>

First thing you need to do, is to download EasyAutocomplete plugin files.

More information about it can be found in the [Download section](#).

Javascript and css files(in the actual version) are located in *dist folder* of the project.

Note: If you want to make any changes, uncompiled javascript files and unprocessed sass files can be found in the *src* folder.

```
|
+---dist
|   |   easy-autocomplete.css
|   |   easy-autocomplete.min.css
|   |   easy-autocomplete.themes.css
|   |   easy-autocomplete.themes.min.css
|   |   jquery.easy-autocomplete.js
|   |   jquery.easy-autocomplete.min.js
|   |
|   \---maps
|       easy-autocomplete.css.map
|       easy-autocomplete.min.css.map
|       easy-autocomplete.themes.css.map
|       easy-autocomplete.themes.min.css.map
|
```

Exemples of static files in the Django World

The Django contrib module

```
C:\PROJECT\PYTHON_ENVS\DJANGO_TEST_AUTOCOMPLETE_35_64\LIB\SITE-
  ↳ PACKAGES\DJANGO\CONTRIB\ADMIN\STATIC
\---admin
+---css
|   base.css
|   changelists.css
|   dashboard.css
|   fonts.css
|   forms.css
|   login.css
|   rtl.css
|   widgets.css
|
+---fonts
|   LICENSE.txt
|   README.txt
|   Roboto-Bold-webfont.woff
|   Roboto-Light-webfont.woff
|   Roboto-Regular-webfont.woff
|
+---img
|   |   calendar-icons.svg
|   |   icon-addlink.svg
|   |   icon-alert.svg
|   |   icon-calendar.svg
|   |   icon-changelink.svg
|   |   icon-clock.svg
|   |   icon-deletelink.svg
|   |   icon-no.svg
|   |   icon-unknown-alt.svg
```

```

| | icon-unknown.svg
| | icon-yes.svg
| | inline-delete.svg
| | LICENSE
| | README.txt
| | search.svg
| | selector-icons.svg
| | sorting-icons.svg
| | tooltag-add.svg
| | tooltag-arrowright.svg
| |
| \---gis
|     move_vertex_off.svg
|     move_vertex_on.svg
|
| \---js
|     actions.js
|     actions.min.js
|     calendar.js
|     cancel.js
|     change_form.js
|     collapse.js
|     collapse.min.js
|     core.js
|     inlines.js
|     inlines.min.js
|     jquery.init.js
|     popup_response.js
|     prepopulate.js
|     prepopulate.min.js
|     prepopulate_init.js
|     SelectBox.js
|     SelectFilter2.js
|     timeparse.js
|     urlify.js
|
| +---admin
|     DateTimeShortcuts.js
|     RelatedObjectLookups.js
|
| \---vendor
|     +---jquery
|         |     jquery.js
|         |     jquery.min.js
|         |     LICENSE-JQUERY.txt
|         |
|         \---xregexp
|             LICENSE-XREGEXP.txt
|             xregexp.js
|             xregexp.min.js

```

Django_By_Example_Code/Chapter 1/mysite/blog

The static directory is in the blog **application directory**.

```
C:\TMP\DJANGO-BY-EXAMPLE-BOOK-MASTER\CHAP_1_BLOG\MYSITE\BLOG
|   admin.py
|   apps.py
|   models.py
|   tests.py
|   urls.py
|   views.py
|   __init__.py
|
+---migrations
|   0001_initial.py
|   __init__.py
|
+---static
|   \---css
|       blog.css
|
\---templates
|   pagination.html
|
\---blog
|   base.html
|   pagination.html
|
\---post
|   detail.html
|   list.html
|   pagination.html
```

Django_By_Example_Code/Chapter 8

There are static directories in:

- the courses and shop **applications directory**
- the **project directory**.

```
\---myshop
|   manage.py
|
+---cart
|   |   admin.py
|   |   cart.py
|
|
+---myshop
|   celery.py
|   db.sqlite3
|   settings.py
|   urls.py
|   wsgi.py
|   __init__.py
|
+---orders
|   |   admin.py
|   |   forms.py
|   |   models.py
```

```

| |
| | +---static
| | | \---css
| | |     admin.css
| | |     pdf.css
| |
|
+---shop
| | admin.py
| | models.py
| | tests.py
| | urls.py
|
| |
| | +---static
| | | +---css
| | | |     base.css
| | | |
| | | \---img
| | |     no_image.png
| |
|
\---static
    +---admin
    | +---css
    | |     base.css
    | |     changelists.css
    | |     dashboard.css
    | |     forms.css
    | |     ie.css
    | |     login.css
    | |     rtl.css
    | |     widgets.css
    | |
    | +---img
    | | changelist-bg.gif
    | | changelist-bg_rtl.gif
    | | default-bg-reverse.gif
    | | default-bg.gif
    | | deleted-overlay.gif
    | | icon-no.gif
    | | icon-unknown.gif
    | | icon-yes.gif
    | | icon_addlink.gif
    | | icon_alert.gif
    | | icon_calendar.gif
    | | icon_changelink.gif
    | | icon_clock.gif
    | | icon_deletelink.gif
    | | icon_error.gif
    | | icon_searchbox.png
    | | icon_success.gif
    | | inline-delete-8bit.png
    | | inline-delete.png
    | | inline-restore-8bit.png

```

```
| | | inline-restore.png
| | | inline-splitter-bg.gif
| | | nav-bg-grabber.gif
| | | nav-bg-reverse.gif
| | | nav-bg-selected.gif
| | | nav-bg.gif
| | | selector-icons.gif
| | | selector-search.gif
| | | sorting-icons.gif
| | | tooltag-add.png
| | | tooltag-arrowright.png
| | |
| | | \---gis
| | |     move_vertex_off.png
| | |     move_vertex_on.png
| | |
| | | \---js
| | |     actions.js
| | |     actions.min.js
| | |     calendar.js
| | |     collapse.js
| | |     collapse.min.js
| | |     core.js
| | |     inlines.js
| | |     inlines.min.js
| | |     jquery.init.js
| | |     jquery.js
| | |     jquery.min.js
| | |     LICENSE-JQUERY.txt
| | |     prepopulate.js
| | |     prepopulate.min.js
| | |     related-widget-wrapper.js
| | |     SelectBox.js
| | |     SelectFilter2.js
| | |     timeparse.js
| | |     urlify.js
| | |
| | | \---admin
| | |     DateTimeShortcuts.js
| | |     RelatedObjectLookups.js
| | |
| | | +---css
| | |     admin.css
| | |     base.css
| | |     pdf.css
| | |
| | | \---img
| | |     no_image.png
```

Django_By_Example_Code/Chapter 13

There are static directories in:

- the courses **application directory**
- the **project directory**.

```

\---educa
|   db.sqlite3
|   manage.py
|
+---config
|   nginx.conf
|   uwsgi.ini
|
+---courses
|   |   admin.py
|   |   fields.py
|   |   forms.py
|
|   |
|   +---static
|   |   \---css
|   |       base.css
|   |
|
|
+---educa
|   |   db.sqlite3
|   |   urls.py
|   |   wsgi.py
|   |   __init__.py
|   |
|   \---settings
|   |   base.py
|   |   local.py
|   |   pro.py
|   |   __init__.py
|   |
+---ssl
|   educa.crt
|   educa.key
|
+---static
|   +---admin
|   |   +---css
|   |   |   base.css
|   |   |   changelists.css
|   |   |   dashboard.css
|   |   |   forms.css
|   |   |   ie.css
|   |   |   login.css
|   |   |   rtl.css
|   |   |   widgets.css
|   |   |
|   |   +---img
|   |   |   changelist-bg.gif
|   |   |   changelist-bg_rtl.gif
|   |   |   default-bg-reverse.gif
|   |   |   default-bg.gif
|   |   |   deleted-overlay.gif
|   |   |   icon-no.gif
|   |   |   icon-unknown.gif
|   |   |   icon-yes.gif

```

```
| | | | icon_addlink.gif
| | | | icon_alert.gif
| | | | icon_calendar.gif
| | | | icon_changelink.gif
| | | | icon_clock.gif
| | | | icon_deletelink.gif
| | | | icon_error.gif
| | | | icon_searchbox.png
| | | | icon_success.gif
| | | | inline-delete-8bit.png
| | | | inline-delete.png
| | | | inline-restore-8bit.png
| | | | inline-restore.png
| | | | inline-splitter-bg.gif
| | | | nav-bg-grabber.gif
| | | | nav-bg-reverse.gif
| | | | nav-bg-selected.gif
| | | | nav-bg.gif
| | | | selector-icons.gif
| | | | selector-search.gif
| | | | sorting-icons.gif
| | | | tootlag-add.png
| | | | tootlag-arrowright.png
| | | |
| | | | \---gis
| | | |     move_vertex_off.png
| | | |     move_vertex_on.png
| | | |
| | | | \---js
| | | |     actions.js
| | | |     actions.min.js
| | | |     calendar.js
| | | |     collapse.js
| | | |     collapse.min.js
| | | |     core.js
| | | |     inlines.js
| | | |     inlines.min.js
| | | |     jquery.init.js
| | | |     jquery.js
| | | |     jquery.min.js
| | | |     LICENSE-JQUERY.txt
| | | |     prepopulate.js
| | | |     prepopulate.min.js
| | | |     related-widget-wrapper.js
| | | |     SelectBox.js
| | | |     SelectFilter2.js
| | | |     timeparse.js
| | | |     urlify.js
| | | |
| | | | \---admin
| | | |     DateTimeShortcuts.js
| | | |     RelatedObjectLookups.js
| | | |
| | | | +---css
| | | |     base.css
| | | |
| | | | \---rest_framework
| | | |     +---css
```



```

|         | bootstrap-tweaks.css
|         | bootstrap.min.css
|         | default.css
|         | prettify.css
|
| +---fonts
|         | glyphsicons-halflings-regular.eot
|         | glyphsicons-halflings-regular.svg
|         | glyphsicons-halflings-regular.ttf
|         | glyphsicons-halflings-regular.woff
|
| +---img
|         | glyphsicons-halflings-white.png
|         | glyphsicons-halflings.png
|         | grid.png
|
| \---js
|         | bootstrap.min.js
|         | default.js
|         | jquery-1.8.1-min.js
|         | prettify-min.js

```

Django settings example

See also:

- <https://docs.djangoproject.com/en/dev/ref/settings/#settings-staticfiles>

```

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

# https://docs.djangoproject.com/en/dev/howto/static-files/
STATIC_URL = '/static/'
STATIC_ROOT = os.path.join(BASE_DIR, 'static/')

```

Integration in our Django test project

```

+---static
|   +---css
|       | easy-autocomplete.min.css
|       | easy-autocomplete.themes.min.css
|       |
|   \---js
|       | jquery.easy-autocomplete.min.js

```

```

|   ajax_selects_singers_db
|   manage.py
|   run_local_dev_server.bat
|
+---projects
|   | admin.py
|   | apps.py
|   | forms.py
|   | models.py

```

```
| | tests.py
| | urls.py
| | views.py
| | __init__.py
| |
| +---migrations
| | | 0001_initial.py
| | | __init__.py
| | |
|
| +---templates
| | \---projects
| | | \---project
| | | | update.html
| | |
|
|
+---projet_ajax
| | settings.py
| | urls.py
| | wsgi.py
| | __init__.py
| |
|
+---singers
| | admin.py
| | apps.py
| | forms.py
| | lookups.py
| | models.py
| | tests.py
| | urls.py
| | views.py
| | __init__.py
| |
| +---migrations
| | | 0001_initial.py
| | | 0002_auto_20161017_1612.py
| | | 0003_auto_20161017_1632.py
| | | __init__.py
| | |
|
|
| +---templates
| | \---singers
| | | +---author
| | | | update.html
| | | |
| | | +---book
| | | | update.html
| | | |
| | | \---song
| | | | update.html
| | |
|
|
+---static
| +---css
```

```
| | easy-autocomplete.min.css
| | easy-autocomplete.themes.min.css
| |
| \---js
|     jquery.easy-autocomplete.min.js
|
| \---templates
|     base.html
|     search_form.html
```

Create a simple update_easy_simple.html

Le fichier projects/templates/projects/projet/update_easy_simple.html

```
{% load static %}
{% load staticfiles %}

<!DOCTYPE html>
<html lang="en">
  <head>
    <title>EasyAutocomplete simple example</title>

    <!-- STYLE -> CSS -->
    <link rel="stylesheet" href="{% static 'css/easy-autocomplete.css' %}"
→type="text/css">
    <link rel="stylesheet" href="{% static 'css/easy-autocomplete.themes.css'
→%}" type="text/css">

    <!-- end STYLE-->

  </head>
  <body>

    <!-- STRUCTURE -> HTML5 elements -->
    <h1>EasyAutocomplete - easy example</h1>
    <input id="simple" />
    <!-- end STRUCTURE-->

    <!--BEHAVIOR -> Javascript scripts-->
    <!-- Using jQuery with a CDN -->
    <script src="//code.jquery.com/jquery-1.11.2.js"></script>
    <script src="{% static 'js/jquery.easy-autocomplete.js' %}" type="text/
→javascript"></script>

    <script>

      var options_easy_autocomplete = {
        data: ["blue", "green", "pink", "red", "yellow"],

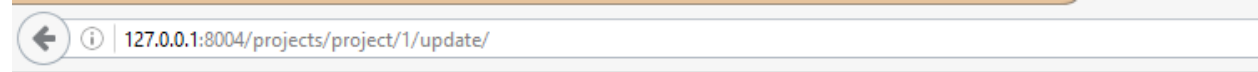
      };

      $("#simple").easyAutocomplete(options_easy_autocomplete);

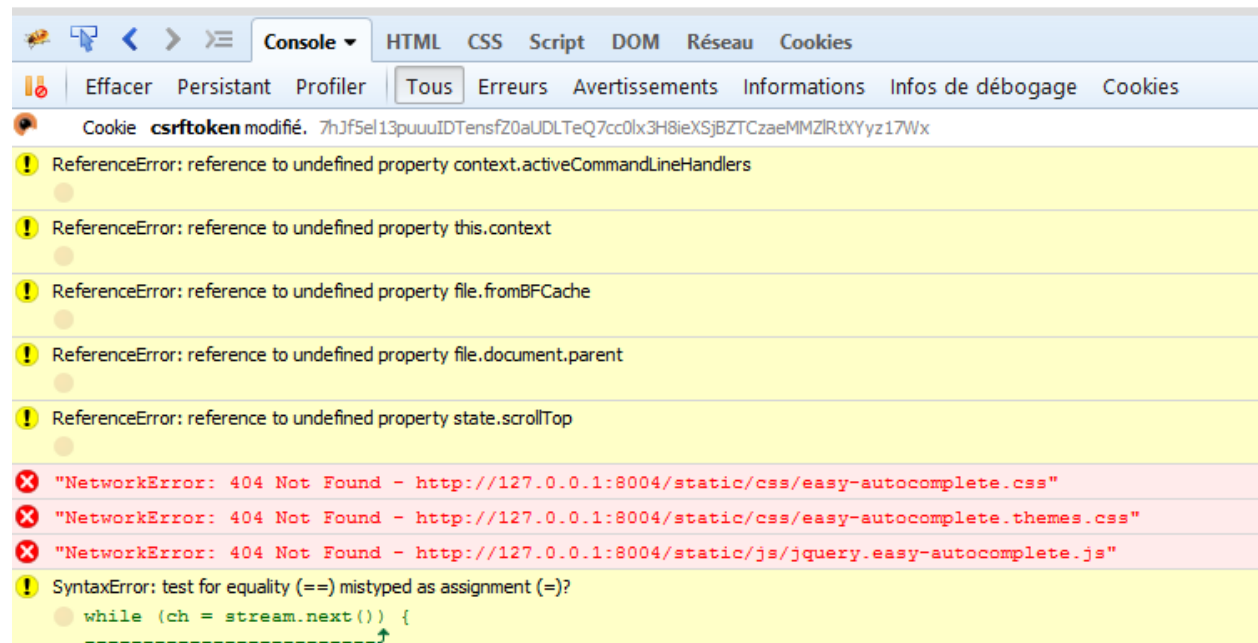
    </script>
    <!-- end BEHAVIOR -->
```

```
</body>
</html>
```

First problem : the libraries are not found



EasyAutocomplete - easy example



```
from os.path import (join,
                     basename,
                     dirname)

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
```

```

DJANGO_ROOT = BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
# Site name ('projet_ajax')
SITE_NAME = basename(DJANGO_ROOT)
# Absolute filesystem path to the top-level project folder:
PROJECT_ROOT=dirname(DJANGO_ROOT)

logging.info("DJANGO_ROOT={} SITE_NAME={} PROJECT_ROOT={}".format(
    DJANGO_ROOT,
    SITE_NAME,
    PROJECT_ROOT,
))

# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/dev/howto/static-files/
STATIC_ROOT = join(PROJECT_ROOT, 'staticfiles')
# STATIC_ROOT="C:/projects_id3/P5N001/XLOG5R372_id3_intranet/trunk/django-www/
# staticfiles"
# Static files (CSS, JavaScript, Images)
# http://www.marinamele.com/taskbuster-django-tutorial/create-home-page-with-tdd-
# staticfiles-templates-settings
# https://docs.djangoproject.com/en/dev/howto/static-files/
# http://whitenoise.evans.io/en/latest/django.html#runserver-nostatic
STATIC_URL = '/static/'
# https://docs.djangoproject.com/en/dev/howto/static-files/
# les répertoires ou sont sockés les fichiers statiques en developpement
STATICFILES_DIRS = [
    join(DJANGO_ROOT, 'static'),
]

```

collectstatic

+:

```
python manage.py collectstatic
```

New tree

```

+---doc_sphinx
|   +---actions
|   |   \---2016
|
+---projet_ajax
|   +---projects
|   |   +---migrations
|
\---staticfiles
    +---admin
    |   +---css
    |   +---fonts
    |   +---img
    |   |   \---gis
    |   \---js
    |       +---admin

```

```

|      \---vendor
|      +---jquery
|      \---xregexp
+---ajax_select
|   +---css
|   +---images
|   \---js
+---autocomplete_light
|   \---vendor
|       \---select2
|           +---dist
|               |   +---css
|               |   \---js
|               |       \---i18n
|           +---src
|               |   \---js
|               |       \---select2
|               |           +---compat
|               |           +---data
|               |           +---dropdown
|               |           +---i18n
|               |           \---selection
|           +---tests
|               |   +---ally
|               |   +---data
|               |   +---dropdown
|               |   +---integration
|               |   +---options
|               |   +---selection
|               |   +---utils
|               |   \---vendor
|               \---vendor
+---css
+---debug_toolbar
|   +---css
|   +---img
|   \---js
+---django_extensions
|   +---css
|   +---img
|   \---js
\---js

```

Create an easyautocomplete directory

```
python manage.py collectstatic
```

```

(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete\projet_ajax>python manage.py collectstatic
You have requested to collect static files at the destination
location as specified in your settings:
    C:\projects_id3\django-test-autocomplete\staticfiles
This will overwrite existing files!
Are you sure you want to do this?
Type 'yes' to continue, or 'no' to cancel: yes
Copying 'C:\projects_id3\django-test-autocomplete\projet_ajax\static\easyautocomplete\css\easy-autocomplete.css'
Copying 'C:\projects_id3\django-test-autocomplete\projet_ajax\static\easyautocomplete\css\easy-autocomplete.min.css'
Copying 'C:\projects_id3\django-test-autocomplete\projet_ajax\static\easyautocomplete\css\easy-autocomplete.themes.css'
Copying 'C:\projects_id3\django-test-autocomplete\projet_ajax\static\easyautocomplete\css\easy-autocomplete.themes.min.css'
Copying 'C:\projects_id3\django-test-autocomplete\projet_ajax\static\easyautocomplete\js\jquery.easy-autocomplete.js'
Copying 'C:\projects_id3\django-test-autocomplete\projet_ajax\static\easyautocomplete\js\jquery.easy-autocomplete.min.js'
Found another file with the destination path 'admin\js\jquery.init.js'. It will be ignored since only the first encountered file is collected. If this is not what you want, make sure every stati
c file has a unique path.
6 static files copied to 'C:\projects_id3\django-test-autocomplete\staticfiles', 281 unmodified.
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete\projet_ajax>

```

```
|
\---easyautocomplete
  +---css
  |     easy-autocomplete.css
  |     easy-autocomplete.min.css
  |     easy-autocomplete.themes.css
  |     easy-autocomplete.themes.min.css
  |
  \---js
        jquery.easy-autocomplete.js
        jquery.easy-autocomplete.min.js
```

Update the Django template file

```
{% load static %}
{% load staticfiles %}

<!DOCTYPE html>
<html lang="en">
  <head>
    <title>EasyAutocomplete simple example</title>

    <!-- STYLE -> CSS -->
    <link rel="stylesheet" href="{% static 'easyautocomplete/css/easy-
→autocomplete.css' %}" type="text/css">
    <link rel="stylesheet" href="{% static 'easyautocomplete/css/easy-
→autocomplete.themes.css' %}" type="text/css">

    <!-- end STYLE-->

  </head>
  <body>

    <!-- STRUCTURE -> HTML5 elements -->
    <h1>EasyAutocomplete - easy example</h1>
    <input id="simple" />
    <!-- end STRUCTURE-->

    <!-- BEHAVIOR -> Javascript scripts-->
    <!-- Using jQuery with a CDN -->
    <script src="//code.jquery.com/jquery-1.11.2.js"></script>
    <script src="{% static 'easyautocomplete/js/jquery.easy-autocomplete.js'
→%}" type="text/javascript"></script>

    <script>

      var options_easy_autocomplete = {
        data: ["blue", "green", "pink", "red", "yellow"],

      };

      $("#simple").easyAutocomplete(options_easy_autocomplete);

    </script>
    <!-- end BEHAVIOR -->
```

```
</body>

</html>
```

OK it works !

Add JSON files

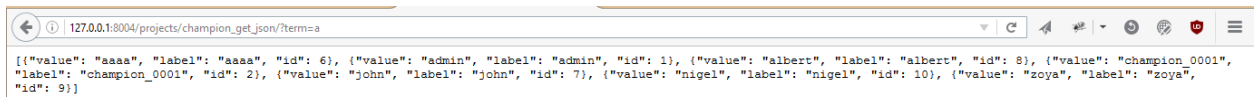
```
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete\projct_ajax>python manage.py collectstatic
You have requested to collect static files at the destination
location as specified in your settings:
    C:\projects_id3\django-test-autocomplete\staticfiles
This will overwrite existing files!
Are you sure you want to do this?
Type 'yes' to continue, or 'no' to cancel: yes
Copying 'C:\projects_id3\django-test-autocomplete\projct_ajax\static\easyautocomplete\data/categories.json'
Copying 'C:\projects_id3\django-test-autocomplete\projct_ajax\static\easyautocomplete\data/countries.json'
Copying 'C:\projects_id3\django-test-autocomplete\projct_ajax\static\easyautocomplete\data/countries.xml'
Copying 'C:\projects_id3\django-test-autocomplete\projct_ajax\static\easyautocomplete\data/flags.css'
Copying 'C:\projects_id3\django-test-autocomplete\projct_ajax\static\easyautocomplete\data/flags.png'
Copying 'C:\projects_id3\django-test-autocomplete\projct_ajax\static\easyautocomplete\data/icon_search.png'
Copying 'C:\projects_id3\django-test-autocomplete\projct_ajax\static\easyautocomplete\data/people.json'
Copying 'C:\projects_id3\django-test-autocomplete\projct_ajax\static\easyautocomplete\data/site.json'
Found another file with the destination path 'admin/js/jquery.init.js'. It will be ignored since only the first encountered file is collected. If this is not what you want, make sure every stati
c file has a unique path.
8 static files copied to 'C:\projects_id3\django-test-autocomplete\staticfiles', 287 unmodified.
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete\projct_ajax>
```

Adding the AJAX call in the Django template

See also:

<http://easyautocomplete.com/guide#sec-data-providers>

URL : http://127.0.0.1:8004/projects/champion_get_json/?term=a



```
[{"value": "aaaa", "label": "aaaa", "id": 6}, {"value": "admin", "label": "admin", "id": 1}, {"value": "albert", "label": "albert", "id": 8}, {"value": "champion_0001", "label": "champion_0001", "id": 2}, {"value": "john", "label": "john", "id": 7}, {"value": "nigel", "label": "nigel", "id": 10}, {"value": "zoza", "label": "zoza", "id": 9}]
```

pip install httpie (clihttp)

See also:

- <https://httpie.org/>
- <https://twitter.com/jkbrzt>
- <https://twitter.com/clihttp>
- <https://github.com/eliangcs/http-prompt>

HTTPIe—aitch-tee-tee-pie—is an open source CLI HTTP client that will make you smile: a user-friendly curl alternative that provides a simple http command designed for painless debugging and interaction with HTTP servers, RESTful APIs, and web services.

```
pip install httpie
```




```
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>pip install httpie
Collecting httpie
  Using cached httpie-0.9.6-py2.py3-none-any.whl
Requirement already satisfied (use --upgrade to upgrade): colorama>=0.2.4; sys_platform == "win32" in c:\project\python_envs\django_test_autocomplete_35_64\lib\
Requirement already satisfied (use --upgrade to upgrade): Pygments>=2.1.3 in c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from httpie)
Collecting requests>=2.11.0 (from httpie)
  Using cached requests-2.11.1-py2.py3-none-any.whl
Installing collected packages: requests, httpie
Successfully installed httpie-0.9.6 requests-2.11.1

(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>http http://127.0.0.1:8004/projects/champion_get_json/?term=a
HTTP/1.0 200 OK
Content-Type: application/json
Date: Fri, 21 Oct 2016 06:41:59 GMT
Server: WSGIServer/0.2 CPython/3.5.2
X-Frame-Options: SAMEORIGIN

[
  {
    "id": 6,
    "label": "aaaa",
    "value": "aaaa"
  },
  {
    "id": 1,
    "label": "admin",
    "value": "admin"
  },
  {
    "id": 8,
    "label": "albert",
    "value": "albert"
  },
  {
    "id": 2,
    "label": "champion_0001",
    "value": "champion_0001"
  },
  {
    "id": 7,
    "label": "john",
    "value": "john"
  },
  {
    "id": 10,
    "label": "nigel",
    "value": "nigel"
  },
  {
    "id": 9,
    "label": "zoya",
    "value": "zoya"
  }
]

(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>
```

http http://127.0.0.1:8004/projects/champion_get_json/?term=a

```
[
  {
    "id": 6,
    "label": "aaaa",
    "value": "aaaa"
  },
  {
    "id": 1,
    "label": "admin",
    "value": "admin"
  },
  {
    "id": 8,
    "label": "albert",
    "value": "albert"
  },
  {
    "id": 2,
    "label": "champion_0001",
    "value": "champion_0001"
  },
  {
    "id": 7,
    "label": "john",
    "value": "john"
  },
  {
    "id": 10,
    "label": "nigel",
    "value": "nigel"
  },
  {
    "id": 9,
    "label": "zoya",
    "value": "zoya"
  }
]
```

http <http://easyautocomplete.com/api/countrySearch.php?phrase=co>

```
[
  {
    "name": "COCOS (KEELING) ISLANDS"
  },
  {
    "name": "COLOMBIA"
  },
  {
    "name": "COMOROS"
  },
  {
    "name": "CONGO"
  },
  {

```

```
(sphinx_35) C:\projects_id3\django-test-autocomplete\doc_sphinx>http http://easyautocomplete.com/api/countrySearch.php?phrase=co
HTTP/1.1 200 OK
Connection: close
Content-Encoding: gzip
Content-Type: application/json
Date: Fri, 21 Oct 2016 06:45:16 GMT
Server: Apache
Transfer-Encoding: chunked
Vary: Accept-Encoding,User-Agent

[
  {
    "name": "COCOS (KEELING) ISLANDS"
  },
  {
    "name": "COLOMBIA"
  },
  {
    "name": "COMOROS"
  },
  {
    "name": "CONGO"
  },
  {
    "name": "CONGO, THE DEMOCRATIC REPUBLIC OF THE"
  },
  {
    "name": "COOK ISLANDS"
  },
  {
    "name": "COSTA RICA"
  },
  {
    "name": "COTE D IVOIRE"
  },
  {
    "name": "MEXICO"
  },
  {
    "name": "MONACO"
  },
  {
    "name": "MOROCCO"
  },
  {
    "name": "PUERTO RICO"
  },
  {
    "name": "TURKS AND CAICOS ISLANDS"
  }
]
```

```
    "name": "CONGO, THE DEMOCRATIC REPUBLIC OF THE"
  },
  {
    "name": "COOK ISLANDS"
  },
  {
    "name": "COSTA RICA"
  },
  {
    "name": "COTE D IVOIRE"
  },
  {
    "name": "MEXICO"
  },
  {
    "name": "MONACO"
  },
  {
    "name": "MOROCCO"
  },
  {
    "name": "PUERTO RICO"
  },
  {
    "name": "TURKS AND CAICOS ISLANDS"
  }
]
```

Avec countrySearch.php

```
<input id="provider-remote" />
```

```
var options = {
  url: function(phrase) {
    return "api/countrySearch.php?phrase=" + phrase + "&format=json";
  },

  getValue: "value"
};

$("#provider-remote").easyAutocomplete(options);
```

Avec Python/Django OK the first step is DONE

See also:

- <http://127.0.0.1:8004/projects/project/1/update/>
- http://127.0.0.1:8004/projects/champion_get_json/?term=a

The Template

```
{# -----#}
↪-----#}
{# simple AJAX call #}
{# http://easyautocomplete.com/guide#sec-data-providers #}
{# -----#}
↪-----#}
{% comment %}
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>http http://
↪127.0.0.1:8004/projects/champion_get_json/?term=z
HTTP/1.0 200 OK
Content-Type: application/json
Date: Fri, 21 Oct 2016 07:51:46 GMT
Server: WSGIServer/0.2 CPython/3.5.2
X-Frame-Options: SAMEORIGIN

[
  {
    "id": 9,
    "label": "zoya",
    "value": "zoya"
  }
]
{% endcomment %}
var options_easy_autocomplete_ajax = {
  {# we have to build this URL: http://127.0.0.1:8004/projects/champion_get_json/?
↪term=a #}
  url: function(term) {
    return "{% url 'projects:champion_get_json' %}" + "?term=" + term;
  },
  getValue: "value",
```

The screenshot displays a web browser window with the title 'EasyAutocomplete simple example' and the URL '127.0.0.1:8004/projects/project/1/update/'. The page content includes three sections: 'Input simple' with a text input field, 'Input data-json' with a text input field, and 'Input AJAX' with a text input field showing a dropdown menu of suggestions: 'a', 'aaaa', 'admin', 'albert', 'champion_0001', 'john', and 'nigel'. The terminal window on the left shows the Django development server running on http://127.0.0.1:8004/.

EasyAutocomplete examples

Input simple

Input data-json

Input AJAX

aaaa

admin

albert

champion_0001

john

nigel

on/?term="

Django version 1.10.2, using settings 'projct_ajax.settings'

Starting development server at http://127.0.0.1:8004/

quit the server with CTRL-BREAK.

21/oct/2016 09:43:29] "GET /projects/project/1/update/ HTTP/1.1" 200 11489

21/oct/2016 09:43:29] "GET /static/easyautocomplete/css/easy-autocomplete.css HTTP/1.1" 200 9277

21/oct/2016 09:43:29] "GET /static/easyautocomplete/css/easy-autocomplete.themes.css HTTP/1.1" 200 5989

21/oct/2016 09:43:29] "GET /static/easyautocomplete/js/jquery.easy-autocomplete.js HTTP/1.1" 200 34623

21/oct/2016 09:43:29] "GET /static/debug_toolbar/css/print.css HTTP/1.1" 200 29

21/oct/2016 09:43:29] "GET /static/debug_toolbar/css/toolbar.css HTTP/1.1" 200 20293

21/oct/2016 09:43:29] "GET /static/debug_toolbar/js/jquery_pre.js HTTP/1.1" 200 136

21/oct/2016 09:43:29] "GET /static/debug_toolbar/js/toolbar.js HTTP/1.1" 200 12552

21/oct/2016 09:43:29] "GET /static/debug_toolbar/js/jquery_post.js HTTP/1.1" 200 118

21/oct/2016 09:43:30] "GET /static/debug_toolbar/img/ajax-loader.gif HTTP/1.1" 200 404

21/oct/2016 09:43:37] "GET /projects/champion_get_json/?term=a HTTP/1.1" 200 342

21/oct/2016 09:43:49] "GET /projects/champion_get_json/?term=a HTTP/1.1" 200 342

21/oct/2016 09:44:18] "GET /projects/champion_get_json/?term=z HTTP/1.1" 200 45

21/oct/2016 09:44:29] "GET /projects/champion_get_json/?term=a HTTP/1.1" 200 342

```
};  
$("#data-ajax").easyAutocomplete(options_easy_autocomplete_ajax);
```

The View `ChampionAutoCompleteView`

```
class ChampionAutoCompleteView(FormView):  
    """  
    Documentation  
    =====  
  
    - https://ccbv.co.uk/projects/Django/1.9/django.views.generic.edit/FormView/  
  
    """  
    def get(self, request, *args, **kwargs):  
        """term is sent by the jquery-ui autocomplete widget.  
  
        The filter is on the username and the user email.  
  
        """  
        term = request.GET.get("term")  
        if term:  
            users = User.objects.filter(Q(username__icontains=term)  
                                       | Q(email__icontains=term)).order_by('username'  
→')  
        else:  
            users = User.objects.all()[:50]  
  
        results = []  
        for user in users:  
            user_json = {}  
            user_json['id'] = user.id  
            user_json['label'] = user.username  
            user_json['value'] = user.username  
            results.append(user_json)  
  
        data = json.dumps(results)  
        mimetype = 'application/json'  
        return HttpResponse(data, mimetype)
```

The projects `urls.py`

```
# calls by jquery-ui autocomplete (AJAX calls)  
# http://127.0.0.1:8004/projects/champion_get_json/?term=a  
url(r'^champion_get_json/$',  
    ChampionAutoCompleteView.as_view(),  
    name='champion_get_json'),
```

This is the happy end of the first step

2016-10-19 improve the Django templates + unobstrusive Javascript (step1)

Contents

- 2016-10-19 improve the Django templates + unobstrusive Javascript (step1)
 - The Javascript code must be in the the body element
 - Modify names
 - Test champion_get_json : http://127.0.0.1:8004/projects/champion_get_json/?term=a
 - Rendering fields manually
 - Adding django-debug-toolbar
 - First step for a better look

The Javascript code must be in the the body element

See also:

- https://www.amazon.fr/gp/product/1617292079/ref=oh_aui_detailpage_o00_s00?ie=UTF8&psc=1

1.2.1 Separating behavior from structure

For all the same reasons that it's desirable to segregate style from structure within an HTML document, it's just as beneficial (if not more so) to separate the *behavior* from the structure. Ideally, an HTML page should be structured as shown in figure 1.2, with structure, style, and behavior each partitioned nicely in its own niche.

This strategy, known as *unobtrusive JavaScript*, is now embraced by every major JavaScript library, helping page authors achieve this useful separation on their pages. As the library that popularized this movement, jQuery's core is well optimized for producing unobtrusive JavaScript easily. Unobtrusive JavaScript considers *any* JavaScript expressions or statements placed within or among HTML tags in the `<body>` of HTML pages, either as attributes of HTML elements (such as `onclick`) or in script blocks placed anywhere other than the very end of the body of the page, to be incorrect.

“But how can I instrument the button without the `onclick` attribute?” you might ask. Consider the following change to the button element:

```
<button id="test-button">Click Me</button>
```

Much simpler! But now, you'll note, the button doesn't do anything. You can click it all day long, and no behavior will result. Let's fix that.

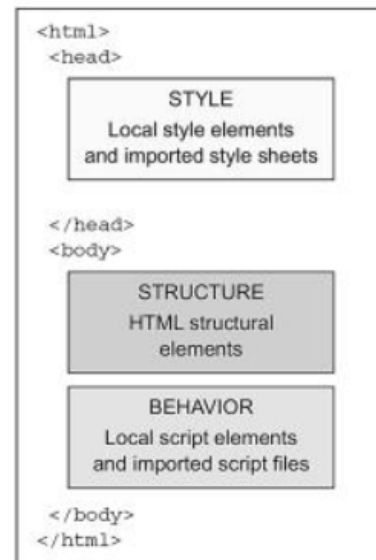


Figure 1.2 With structure, style, and behavior each neatly tucked away within a page, readability and maintainability are maximized.

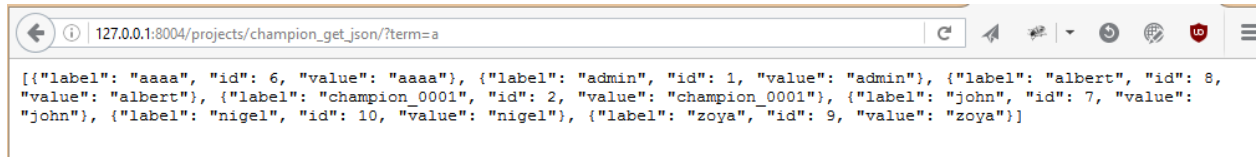
Modify names

- `champion_auto_complete` -> `champion_get_json`

Test `champion_get_json` : `http://127.0.0.1:8004/projects/champion_get_json/?term=a`

See also:

`http://127.0.0.1:8004/projects/champion_get_json/?term=a`



Rendering fields manually

See also:

- <https://docs.djangoproject.com/en/dev/topics/forms/#rendering-fields-manually>
- <https://docs.djangoproject.com/en/dev/topics/forms/#looping-over-hidden-and-visible-fields>

Adding django-debug-toolbar

See also:

- <https://django-debug-toolbar.readthedocs.io/en/stable/index.html>
- <https://django-debug-toolbar.readthedocs.io/en/stable/installation.html>

```
pip install django-debug-toolbar
```

```
# https://django-extensions.readthedocs.org/en/latest
'django_extensions',

# http://django-ajax-selects.readthedocs.io/en/latest/index.html
'ajax_select',
# http://django-crispy-forms.readthedocs.io/en/latest/install.html#installing-
→ django-crispy-forms
'crispy_forms',
# https://github.com/jazzband/django-debug-toolbar
'debug_toolbar',

# # https://docs.djangoproject.com/en/dev/ref/applications/#django.apps.AppConfig
'singers.apps.SingersConfig',
# http://guiqinqian.blogspot.fr/2012/01/using-jquery-auto-complete-in-django.html
'projects.apps.ProjectsConfig'
]
```

```
if settings.DEBUG:
    import debug_toolbar
    urlpatterns += [
```



```
url(r'^__debug__/', include(debug_toolbar.urls)),
]
```

```
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',

    'debug_toolbar.middleware.DebugToolbarMiddleware'
]
```


Warning: do not forget to set the new INTERNAL_IPS variable

INTERNAL_IPS = '127.0.0.1'

First step for a better look

See also:

<http://127.0.0.1:8004/projects/project/1/update/>

127.0.0.1:8004/projects/project/1/update/ | 

Update of the project '(title:project test champion:admin)

Title	<input type="text" value="project test"/>	
Champion	<input type="text"/>	admin

2016-10-18 improve the jquery-ui autocomplete look and feel

See also:

- <http://api.jqueryui.com/autocomplete/>
- <https://jqueryui.com/autocomplete/>
- <https://jqueryui.com/autocomplete/#combobox>

Contents

- *2016-10-18 improve the jquery-ui autocomplete look and feel*
 - *Last look*
 - *jquery-ui autocomplete options already used*

* Remote option

- Try <https://jqueryui.com/autocomplete/#combobox>
- Try <https://jqueryui.com/autocomplete/#maxheight>
- Try <https://jqueryui.com/autocomplete/#categories>
- Try <https://jqueryui.com/autocomplete/#remote-jsonp>

Last look

See also:

- <http://127.0.0.1:8004/projects/project/1/update/>

```
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete\projet_ajax>  
python manage.py runserver 127.0.0.1:8004 Performing system checks...
```

```
System check identified no issues (0 silenced).  
October 19, 2016 - 09:08:19  
Django version 1.10.2, using settings 'projet_ajax.settings'  
Starting development server at http://127.0.0.1:8004/  
Quit the server with CTRL-BREAK.
```

127.0.0.1:8004/projects/project/1/update/

Update of the project '(title:project_0001 champion:zoya)

Title: Update the champion:

type username or email

Update

zoya

```
1 {% extends "base.html" %}  
2 {% load static %}  
3 {% load staticfiles %}  
4  
5  
6 {# STYLE CSS #}  
7 {% block stylesheet %}  
8     {{ block.super }}  
9     {# https://www.bootstrapcdn.com/ #}  
10    <link rel="stylesheet" href="http://code.jquery.com/ui/1.12.1/themes/base/jquery-  
11    ui.css">  
12  
13    <style>  
14        {# https://jqueryui.com/autocomplete/#maxheight #}  
15        .ui-autocomplete {  
16            max-height: 100px;  
17            overflow-y: auto;  
18            /* prevent horizontal scrollbar */  
19            overflow-x: hidden;  
20        }  
21        * html .ui-autocomplete {  
22            height: 100px;
```

```

22     }
23     </style>
24
25 {% endblock %}
26
27
28 {% block content %}
29
30     <h1>Update of the project '(title:{{ project.title }} champion:{{ project.
↪champion }})' </h1>
31     <p></p>
32     <p></p>
33     {# https://docs.djangoproject.com/en/dev/topics/forms/ #}
34     <form id="id_form_project_update" action="{% url 'projects:project_update' _
↪project.id %}" method="post">
35         {% csrf_token %}
36         <div class="forms">
37             {{ form.id }}
38             {{ form.non_field_errors }}
39             {# Include the hidden fields #}
40             {% for hidden in form.hidden_fields %}
41                 {{ hidden }}
42             {% endfor %}
43             <table id="id_table" class="table table-hover table-bordered table-
↪condensed">
44                 <tbody>
45                     <tr>
46                         <td class="text-right">Title:</td>
47                         <td>{{ form.title }}</td>
48                     </tr>
49                     <tr>
50                         <td class="text-right">Champion:</td>
51                         <td><span id="id_champion_display">{{ project.champion }}
↪</span> {{ form.champion_term }} </td>
52                     </tr>
53                 </tbody>
54             </table>
55             <div>
56                 <input type="submit" name="btn_update" value="Update" class="btn btn-success _
↪btn-block" />
57             </div>
58
59
60         {% if messages %}
61         <ul class="messages">
62             {% comment %} https://getbootstrap.com/components/#alerts {% endcomment %}
63             {% for message in messages %}
64                 <!--li{% if message.tags %} class="{{ message.tags }}" {% endif %} -->
65                 {% if message.level == DEFAULT_MESSAGE_LEVELS.ERROR %}
66                     <div class="alert alert-danger" role="alert">{{ message }}</div>
67                 {% elif message.level == DEFAULT_MESSAGE_LEVELS.SUCCESS %}
68                     <div class="alert alert-success" role="alert">{{ message }}</div>
69                 {% else %}
70                     <div class="alert alert-info" role="alert">{{ message }}</div>
71                 {% endif %}
72                 <!-- /li -->
73             {% endfor %}
74         </ul>

```

```
75     {% endif %}
76
77
78 {% endblock content %}
79
80
81 {% BEHAVIOR Javascript %}
82 {% block javascript %}
83     {{ block.super }}
84
85     {% http://code.jquery.com/ui/ %}
86     <script src="http://code.jquery.com/ui/1.12.1/jquery-ui.min.js"></script>
87
88 {% endblock javascript %}
89
90
91 {% DOM ready %}
92 {% block domready %}
93
94     $( "#id_champion_term" ).autocomplete({
95         {% http://api.jqueryui.com/autocomplete/#option-source %}
96         source: "{% url 'projects:champion_get_json' %}",
97         {% http://api.jqueryui.com/autocomplete/#option-autoFocus %}
98         autoFocus:true,
99         {% http://api.jqueryui.com/autocomplete/#option-minLength %}
100        minLength:1,
101        {% http://api.jqueryui.com/autocomplete/#event-select %}
102        select:function(event,ui) {
103            $("#id_champion").val(ui.item.id);
104        }
105    });
106
107 {% endblock %}
```

```
1  #!/usr/bin/python
2  # -*- coding: utf8 -*-
3  """The project's forms.
4
5  """
6
7  from django import forms
8
9  from .models import Project
10
11 class ProjectChampionForm(forms.ModelForm):
12     """The champion project form"""
13     champions_choice_list = forms.CharField(max_length=100,
14                                             help_text='type username or email')
15
16     class Meta:
17         model = Project
18         fields = ('title',
19                 'champions_choice_list', 'champion',)
20
21     def __init__(self, *args, **kwargs):
22         super(ProjectChampionForm, self).__init__(*args, **kwargs)
23         self.fields['champions_choice_list'].label = "Update the champion"
24         self.fields['champion'].widget = forms.HiddenInput()
```

jquery-ui autocomplete options already used

Remote option

- <https://jqueryui.com/autocomplete/#remote>

The Autocomplete widget provides suggestions while you type into the field. Here the suggestions are bird names, displayed when at least two characters are entered into the field.

The datasource is a server-side script which returns JSON data, specified via a simple URL for the source-option. In addition, the minLength-option is set to 2 to avoid queries that would return too many results and the select-event is used to display some feedback.

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>jQuery UI Autocomplete - Remote datasource</title>
  <link rel="stylesheet" href="//code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css">
  <link rel="stylesheet" href="/resources/demos/style.css">
  <style>
    .ui-autocomplete-loading {
      background: white url("images/ui-anim_basic_16x16.gif") right center no-repeat;
    }
  </style>

  <script src="https://code.jquery.com/jquery-1.12.4.js"></script>
  <script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
  <script>
    $( function() {
      function log( message ) {
        $( "<div>" ).text( message ).prependTo( "#log" );
        $( "#log" ).scrollTop( 0 );
      }

      $( "#birds" ).autocomplete({
        source: "search.php",
        minLength: 2,
        select: function( event, ui ) {
          log( "Selected: " + ui.item.value + " aka " + ui.item.id );
        }
      });
    } );
  </script>
</head>
<body>

<div class="ui-widget">
  <label for="birds">Birds: </label>
  <input id="birds">
</div>

<div class="ui-widget" style="margin-top:2em; font-family:Arial">
  Result:
```

```
<div id="log" style="height: 200px; width: 300px; overflow: auto;" class="ui-widget-  
  content"></div>  
</div>  
  
</body>  
</html>
```

Try <https://jqueryui.com/autocomplete/#combobox>

See also:

- <http://www.learningjquery.com/2010/06/a-jquery-ui-combobox-under-the-hood>

Try <https://jqueryui.com/autocomplete/#maxheight>

See also:

- <https://jqueryui.com/autocomplete/#maxheight>

```
</doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>jQuery UI Autocomplete - Scrollable results</title>
  <link rel="stylesheet" href="//code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css">
  <link rel="stylesheet" href="/resources/demos/style.css">
  <style>
    .ui-autocomplete {
      max-height: 100px;
      overflow-y: auto;
      /* prevent horizontal scrollbar */
      overflow-x: hidden;
    }
    /* IE 6 doesn't support max-height
     * we use height instead, but this forces the menu to always be this tall
     */
    * html .ui-autocomplete {
      height: 100px;
    }
  </style>
  <script src="https://code.jquery.com/jquery-1.12.4.js"></script>
  <script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
  <script>
    $( function() {
      var availableTags = [
        "ActionScript",
        "AppleScript",
        "Asp",
        "BASIC",
        "C",
        "C++",
        "Clojure",
        "COBOL",
        "ColdFusion",

```

```

        "Erlang",
        "Fortran",
        "Groovy",
        "Haskell",
        "Java",
        "JavaScript",
        "Lisp",
        "Perl",
        "PHP",
        "Python",
        "Ruby",
        "Scala",
        "Scheme"
    ];
    $( "#tags" ).autocomplete({
        source: availableTags
    });
} );
</script>
</head>
<body>

<div class="ui-widget">
    <label for="tags">Tags: </label>
    <input id="tags">
</div>

</body>
</html>

```

Try <https://jqueryui.com/autocomplete/#categories>

See also:

- <https://jqueryui.com/autocomplete/#categories>

```

<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>jQuery UI Autocomplete - Categories</title>
    <link rel="stylesheet" href="//code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css">
    <link rel="stylesheet" href="/resources/demos/style.css">
    <style>
        .ui-autocomplete-category {
            font-weight: bold;
            padding: .2em .4em;
            margin: .8em 0 .2em;
            line-height: 1.5;
        }
    </style>
    <script src="https://code.jquery.com/jquery-1.12.4.js"></script>
    <script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
    <script>
        $( function() {

```

```
$.widget( "custom.catcomplete", $.ui.autocomplete, {
  _create: function() {
    this._super();
    this.widget().menu( "option", "items", "> :not(.ui-autocomplete-category)" );
  },
  _renderMenu: function( ul, items ) {
    var that = this,
        currentCategory = "";
    $.each( items, function( index, item ) {
      var li;
      if ( item.category != currentCategory ) {
        ul.append( "<li class='ui-autocomplete-category'>" + item.category + "</li>" );
        currentCategory = item.category;
      }
      li = that._renderItemData( ul, item );
      if ( item.category ) {
        li.attr( "aria-label", item.category + " : " + item.label );
      }
    });
  }
});

var data = [
  { label: "anders", category: "" },
  { label: "andreas", category: "" },
  { label: "antal", category: "" },
  { label: "annhxx10", category: "Products" },
  { label: "annk K12", category: "Products" },
  { label: "annttop C13", category: "Products" },
  { label: "anders andersson", category: "People" },
  { label: "andreas andersson", category: "People" },
  { label: "andreas johnson", category: "People" }
];

$( "#search" ).catcomplete({
  delay: 0,
  source: data
});
});
</script>
</head>
<body>

<label for="search">Search: </label>
<input id="search">

</body>
</html>
```

Try <https://jqueryui.com/autocomplete/#remote-jsonp>

See also:

- <https://jqueryui.com/autocomplete/#remote-jsonp>


```

<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>jQuery UI Autocomplete - Remote JSONP datasource</title>
  <link rel="stylesheet" href="//code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css">
  <link rel="stylesheet" href="/resources/demos/style.css">
  <style>
    .ui-autocomplete-loading {
      background: white url("images/ui-anim_basic_16x16.gif") right center no-repeat;
    }
  </style>
  <script src="https://code.jquery.com/jquery-1.12.4.js"></script>
  <script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
  <script>
    $( function() {
      function log( message ) {
        $( "<div>" ).text( message ).prependTo( "#log" );
        $( "#log" ).scrollTop( 0 );
      }

      $( "#birds" ).autocomplete({
        source: function( request, response ) {
          $.ajax( {
            url: "search.php",
            dataType: "jsonp",
            data: {
              term: request.term
            },
            success: function( data ) {
              response( data );
            }
          } );
        },
        minLength: 2,
        select: function( event, ui ) {
          log( "Selected: " + ui.item.value + " aka " + ui.item.id );
        }
      } );
    } );
  </script>
</head>
<body>

<div class="ui-widget">
  <label for="birds">Birds: </label>
  <input id="birds">
</div>

<div class="ui-widget" style="margin-top:2em; font-family:Arial">
  Result:
  <div id="log" style="height: 200px; width: 300px; overflow: auto;" class="ui-widget-
  ↪content"></div>
</div>

</body>

```

```
</html>
```

2016-10-18 try a more simple approach with jquery-ui autocomplete widget

See also:

- <https://jqueryui.com/autocomplete/>
- <http://guiqinqian.blogspot.fr/2012/01/using-jquery-auto-complete-in-django.html>
- <https://github.com/jquery/jquery-ui>

Autocomplete Widget

Categories: [Widgets](#)

Autocomplete Widgetversion added: 1.8

Description: *Autocomplete enables users to quickly find and select from a pre-populated list of values as they type, leveraging searching and filtering.*

QuickNav

Options

- [appendTo](#)
- [autoFocus](#)
- [classes](#)
- [delay](#)
- [disabled](#)
- [minLength](#)
- [position](#)
- [source](#)

Methods

- [close](#)
- [destroy](#)
- [disable](#)
- [enable](#)
- [instance](#)
- [option](#)
- [search](#)
- [widget](#)

Events

- [change](#)
- [close](#)
- [create](#)
- [focus](#)
- [open](#)
- [response](#)
- [search](#)
- [select](#)

Extension Points

- [renderItem](#)
- [renderMenu](#)
- [resizeMenu](#)

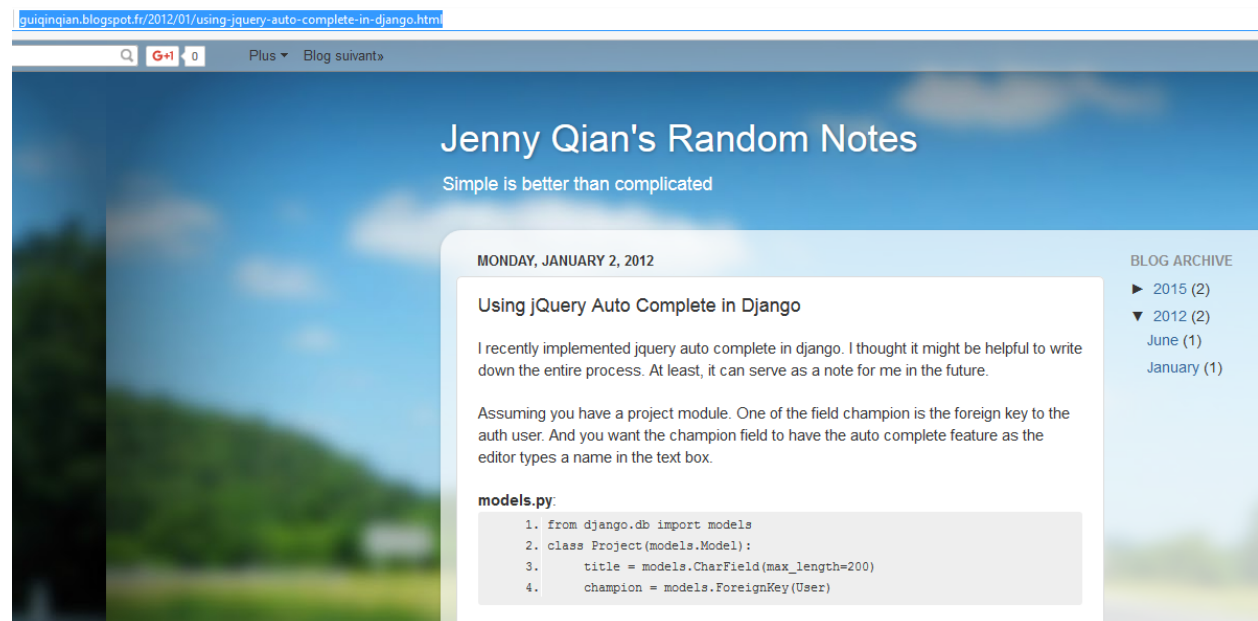
Examples

Contents

- 2016-10-18 try a more simple approach with jquery-ui autocomplete widget
 - Installation
 - Create a *projects* application
 - Add the Django model Project
 - Add the project in the *INSTALLED_APPS*

- *makemigrations*
- *sqlmigrate*
- *migrate*
- *Create the projects/forms.py file and add 2 forms*
- *Create the projects/urls.py file and add the champion auto complete URL*
- *Add the projects.urls file to the root urls.py*
- *Add the ProjectUpdateView*
- *Create the templates/projects/project/update.html django template*
- *Add the jquery-ui javascript code*
- *Add the admin interface*
- *Create some projects with the admin interface*
- *Reading Project record from the commande line*
- *Add some users with the admin interface*
- *Add some users with the command line interface*
- *Try the URL : `http://127.0.0.1:8000/projects/project/1/update`*

Installation



Other sources:

- <http://blog.appliedinformaticsinc.com/autocomplete-input-field-in-django-template-with-jquery-ui/>
- <https://openclassrooms.com/courses/decouvrez-la-puissance-de-jquery-ui/l-autocompletion-1>

Create a projects application

```
python manage.py startapp projects
```

```
| ajax_selects_singers_db
| manage.py
| tree.txt
|
+---projects
| | admin.py
| | apps.py
| | models.py
| | tests.py
| | views.py
| | __init__.py
| |
| \---migrations
|      __init__.py
```

Add the Django model Project

```
1  #!/usr/bin/python
2  # -*- coding: utf8 -*-
3  """The project's models.
4
5  """
6  from __future__ import unicode_literals
7  from django.utils.encoding import python_2_unicode_compatible
8
9  from django.db import models
10 from django.core.urlresolvers import reverse
11
12 # https://docs.djangoproject.com/en/dev/ref/contrib/auth/#user-model
13 from django.contrib.auth.models import User
14
15
16
17 @python_2_unicode_compatible
18 class Project(models.Model):
19     """A project with a title and a champion which is the foreign key to the
20     auth user.
21
22     Documentation
23     =====
24
25     - http://guiqinqian.blogspot.fr/2012/01/using-jquery-auto-complete-in-django.html
26
27     """
28     title = models.CharField(max_length=200)
29     champion = models.ForeignKey(User)
30
31     def __str__(self):
32         return "{} {}".format(self.title, self.champion)
33
34     def get_absolute_url(self):
35         """
```

```

36     https://docs.djangoproject.com/en/dev/ref/class-based-views/generic-editing/
37     """
38     return reverse('projects:project_update',
39                    kwargs={'pk': self.pk})

```

Add the project in the INSTALLED_APPS

```

# # https://docs.djangoproject.com/en/dev/ref/applications/#django.apps.AppConfig
'singers.apps.SingersConfig',
# http://guiqinqian.blogspot.fr/2012/01/using-jquery-auto-complete-in-django.html
'projects.apps.ProjectsConfig'
]

```

```

1  """
2  Django settings for projet_ajax project.
3
4  Generated by 'django-admin startproject' using Django 1.10.2.
5
6  For more information on this file, see
7  https://docs.djangoproject.com/en/1.10/topics/settings/
8
9  For the full list of settings and their values, see
10 https://docs.djangoproject.com/en/1.10/ref/settings/
11 """
12
13 import os
14 import logging
15
16 from os.path import (join,
17                      basename,
18                      dirname)
19
20 # Build paths inside the project like this: os.path.join(BASE_DIR, ...)
21 DJANGO_ROOT = BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
22 # Site name ('projet_ajax')
23 SITE_NAME = basename(DJANGO_ROOT)
24 # Absolute filesystem path to the top-level project folder:
25 PROJECT_ROOT=dirname(DJANGO_ROOT)
26
27 logging.info("DJANGO_ROOT={} SITE_NAME={} PROJECT_ROOT={}".format (
28     DJANGO_ROOT,
29     SITE_NAME,
30     PROJECT_ROOT,
31 ))
32
33 # Quick-start development settings - unsuitable for production
34 # See https://docs.djangoproject.com/en/1.10/howto/deployment/checklist/
35
36 # SECURITY WARNING: keep the secret key used in production secret!
37 SECRET_KEY = 'q#e(dtsmq9^!(%9*un6tuq5v)2!@00ftgk$!@h1*(e0iwo@=cc'
38
39 # SECURITY WARNING: don't run with debug turned on in production!
40 DEBUG = True
41
42 # for the debug toolbar
43 INTERNAL_IPS = '127.0.0.1'

```

```
44
45 DEBUG_TOOLBAR_PANELS = [
46     'debug_toolbar.panels.versions.VersionsPanel',
47     'debug_toolbar.panels.timer.TimerPanel',
48     'debug_toolbar.panels.settings.SettingsPanel',
49     'debug_toolbar.panels.headers.HeadersPanel',
50     'debug_toolbar.panels.request.RequestPanel',
51     'debug_toolbar.panels.sql.SQLPanel',
52     'debug_toolbar.panels.staticfiles.StaticFilesPanel',
53     'debug_toolbar.panels.templates.TemplatesPanel',
54     'debug_toolbar.panels.cache.CachePanel',
55     'debug_toolbar.panels.signals.SignalsPanel',
56     'debug_toolbar.panels.logging.LoggingPanel',
57     'debug_toolbar.panels.redirects.RedirectsPanel',
58 ]
59
60
61 ALLOWED_HOSTS = []
62
63
64 # Applications definition
65 INSTALLED_APPS = [
66     'dal',
67     'dal_select2',
68
69     'django.contrib.admin',
70     'django.contrib.auth',
71     'django.contrib.contenttypes',
72     'django.contrib.sessions',
73     'django.contrib.messages',
74     'django.contrib.staticfiles',
75
76     # https://django-extensions.readthedocs.org/en/latest
77     'django_extensions',
78
79     # http://django-ajax-selects.readthedocs.io/en/latest/index.html
80     'ajax_select',
81     # http://django-crispy-forms.readthedocs.io/en/latest/install.html#installing-
82     ↪ 'django-crispy-forms',
83     # https://github.com/jazzband/django-debug-toolbar
84     # 'debug_toolbar',
85
86     # # https://docs.djangoproject.com/en/dev/ref/applications/#django.apps.AppConfig
87     'singers.apps.SingersConfig',
88     # http://guiqinqian.blogspot.fr/2012/01/using-jquery-auto-complete-in-django.html
89     'projects.apps.ProjectsConfig'
90 ]
91
92 MIDDLEWARE = [
93     'django.middleware.security.SecurityMiddleware',
94     'django.contrib.sessions.middleware.SessionMiddleware',
95     'django.middleware.common.CommonMiddleware',
96     'django.middleware.csrf.CsrfViewMiddleware',
97     'django.contrib.auth.middleware.AuthenticationMiddleware',
98     'django.contrib.messages.middleware.MessageMiddleware',
99     'django.middleware.clickjacking.XFrameOptionsMiddleware',
100 ]
```

```

101     # 'debug_toolbar.middleware.DebugToolbarMiddleware'
102 ]
103
104
105 ROOT_URLCONF = 'projet_ajax.urls'
106
107 TEMPLATES = [
108     {
109         'BACKEND': 'django.template.backends.django.DjangoTemplates',
110         'DIRS': [join(BASE_DIR, 'templates')],
111         'APP_DIRS': True,
112         'OPTIONS': {
113             'context_processors': [
114                 'django.template.context_processors.debug',
115                 'django.template.context_processors.request',
116                 'django.contrib.auth.context_processors.auth',
117                 'django.contrib.messages.context_processors.messages',
118             ],
119         },
120     },
121 ]
122
123 WSGI_APPLICATION = 'projet_ajax.wsgi.application'
124
125
126 # Database
127 # https://docs.djangoproject.com/en/1.10/ref/settings/#databases
128 DATABASES = {
129     'default': {
130         'ENGINE': 'django.db.backends.sqlite3',
131         'NAME': os.path.join(BASE_DIR, 'ajax_selects_singers_db')
132     }
133 }
134
135 # Password validation
136 # https://docs.djangoproject.com/en/1.10/ref/settings/#auth-password-validators
137
138 AUTH_PASSWORD_VALIDATORS = [
139     {
140         'NAME': 'django.contrib.auth.password_validation.
141 ↪UserAttributeSimilarityValidator',
142     },
143     {
144         'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
145     },
146     {
147         'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
148     },
149     {
150         'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
151     },
152 ]
153
154 # Internationalization
155 # https://docs.djangoproject.com/en/1.10/topics/i18n/
156
157 LANGUAGE_CODE = 'en-us'

```

```
158 TIME_ZONE = 'UTC'
159
160
161 USE_I18N = True
162
163 USE_L10N = True
164
165 USE_TZ = True
166
167
168 # Static files (CSS, JavaScript, Images)
169 # https://docs.djangoproject.com/en/dev/howto/static-files/
170 STATIC_ROOT = join(PROJECT_ROOT, 'staticfiles')
171 # STATIC_ROOT="C:/projects_id3/P5N001/XLOG5R372_id3_intranet/trunk/django-www/
172 ↪staticfiles"
173 # Static files (CSS, JavaScript, Images)
174 # http://www.marinamele.com/taskbuster-django-tutorial/create-home-page-with-tdd-
175 ↪staticfiles-templates-settings
176 # https://docs.djangoproject.com/en/dev/howto/static-files/
177 # http://whitenoise.evans.io/en/latest/django.html#runserver-nostatic
178 STATIC_URL = '/static/'
179 # https://docs.djangoproject.com/en/dev/howto/static-files/
180 # les répertoires où sont stockés les fichiers statiques en développement
181 STATICFILES_DIRS = [
182     join(DJANGO_ROOT, 'static'),
183 ]
184
185 AJAX_LOOKUP_CHANNELS = {
186     # simplest way, automatically construct a search channel by passing a dict
187     'label': {'model': 'singers.label'
188              , 'search_field': 'name'},
189
190     # Custom channels are specified with a tuple
191     # channel: ( module.where_lookup_is, ClassNameOfLookup )
192     # 'person': ('singers.lookups', 'PersonLookup'),
193     # 'group': ('singers.lookups', 'GroupLookup'),
194     # 'song': ('singers.lookups', 'SongLookup'),
195 }
196
197 # voir https://github.com/jordij/bctt.nz/blob/master/ttwellington/settings/base.py
198 MEDIA_ROOT = join(BASE_DIR, 'media')
199 MEDIA_URL = '/media/'
```

makemigrations

```
python manage.py makemigrations projects
```

```
Migrations for 'projects':
  projects\migrations\0001_initial.py:
    - Create model Project
```

```
1 # -*- coding: utf-8 -*-
2 # Generated by Django 1.10.2 on 2016-10-18 10:45
3 from __future__ import unicode_literals
```



```

4
5 from django.conf import settings
6 from django.db import migrations, models
7 import django.db.models.deletion
8
9
10 class Migration(migrations.Migration):
11
12     initial = True
13
14     dependencies = [
15         migrations.swappable_dependency(settings.AUTH_USER_MODEL),
16     ]
17
18     operations = [
19         migrations.CreateModel(
20             name='Project',
21             fields=[
22                 ('id', models.AutoField(auto_created=True, primary_key=True,
23 ↪ serialize=False, verbose_name='ID')),
24                 ('title', models.CharField(max_length=200)),
25                 ('champion', models.ForeignKey(on_delete=django.db.models.deletion.
26 ↪ CASCADE, to=settings.AUTH_USER_MODEL)),
27     ],
28         ),
29     ],
30 ]

```

sqlmigrate

```
python manage.py sqlmigrate projects
```

```
(ajax_django_35) C:\projects_id3\django_ajax_select\projet_ajax>python manage.py
↪ sqlmigrate projects 0001
```

```

BEGIN;
--
-- Create model Project
--
CREATE TABLE "projects_project" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
↪ "title" varchar(200) NOT NULL, "champion_id" integer NOT NULL REFERENCES "auth_user
↪ ("id"));
CREATE INDEX "projects_project_68b9a77a" ON "projects_project" ("champion_id");
COMMIT;

```

migrate

```
python manage.py migrate projects
```

```

Operations to perform:
  Apply all migrations: projects
Running migrations:
  Applying projects.0001_initial... OK

```

Create the projects/forms.py file and add 2 forms

```
1  #!/usr/bin/python
2  # -*- coding: utf8 -*-
3  """The project's forms.
4
5  """
6
7  from django import forms
8
9  from .models import Project
10
11 class ProjectChampionForm(forms.ModelForm):
12     """The champion project form"""
13     champions_choice_list = forms.CharField(max_length=100,
14                                             help_text='type username or email')
15
16     class Meta:
17         model = Project
18         fields = ('title',
19                 'champions_choice_list', 'champion',)
20
21     def __init__(self, *args, **kwargs):
22         super(ProjectChampionForm, self).__init__(*args, **kwargs)
23         self.fields['champions_choice_list'].label = "Update the champion"
24         self.fields['champion'].widget = forms.HiddenInput()
```

Create the projects/urls.py file and add the champion auto complete URL

```
1  # -*- coding: utf-8 -*-
2
3  from django.conf.urls import url
4
5
6  from .views_json import (ApiEACGetProjectsJSONView,
7                          ApiEACGetChampionsJSONView)
8
9  from .views_django_autocomplete_light import (ApiUserDjangoAutocompleteLight,
10                                              ProjectDjangoAutoCompleteUpdateView,
11                                              ProjectAutocompleteView)
12
13  from .views import (ProjectUpdateView,
14                    ProjectUpdateViewEasyAutoComplete,
15                    ProjectUpdateViewjQueryUIAutoComplete)
16
17
18 urlpatterns = [
19     url(r'^project/(?P<pk>\d+)/update/$',
20         ProjectUpdateView.as_view(),
21         name='project_update'),
22
23     url(r'^project/(?P<pk>\d+)/updateeasy/$',
24         ProjectUpdateViewEasyAutoComplete.as_view(),
25         name='project_update_easy'),
26
27     url(r'^project/(?P<pk>\d+)/update_jquery_ui/$',
```

```

28     ProjectUpdateViewjQueryUIAutoComplete.as_view(),
29     name='project_update_jquery_ui'),
30
31     url(r'^project/(?P<pk>\d+)/update_django_autocomplete/$',
32         ProjectDjangoAutoCompleteUpdateView.as_view(),
33         name='project_update_django_autocomplete'),
34
35     # calls by jquery EasyAutocomplete
36     # http://127.0.0.1:8004/projects/api_get_champions/?term=a
37     url(r'^api_get_champions/$',
38         ApiEACGetchampionsJSONView.as_view(),
39         name='api_get_champions'),
40
41     # calls by jquery EasyAutocomplete (EAC)
42     # http://127.0.0.1:8004/projects/api_get_projects/?term=a
43     url(r'^apis_get_projects/$',
44         ApiEACGetProjectsJSONView.as_view(),
45         name='api_get_projects'),
46
47     # calling example:
48     # http://127.0.0.1:8004/projects/project_autocomplete/?q=a
49     url(
50         r'^project_autocomplete/$',
51         ProjectAutocompleteView.as_view(),
52         name='project_autocomplete',
53     ),
54
55     url(r'^api/get_users/$',
56         ApiUserDjangoAutocompleteLight.as_view(),
57         name='api_get_users'),
58 ]

```

Add the projects.urls file to the root urls.py

```
url(r'^projects/', include('projects.urls', namespace='projects')),
```

```

1  #!/usr/bin/python
2  # -*- coding: utf8 -*-
3  """projet_ajax URL Configuration
4
5  The `urlpatterns` list routes URLs to views. For more information please see:
6      https://docs.djangoproject.com/en/1.10/topics/http/urls/
7  Examples:
8  Function views
9      1. Add an import:  from my_app import views
10     2. Add a URL to urlpatterns:  url(r'^$', views.home, name='home')
11  Class-based views
12     1. Add an import:  from other_app.views import Home
13     2. Add a URL to urlpatterns:  url(r'^$', Home.as_view(), name='home')
14  Including another URLconf
15     1. Import the include() function: from django.conf.urls import url, include
16     2. Add a URL to urlpatterns:  url(r'^blog/', include('blog.urls'))
17  """
18
19  from django.conf import settings
20  from django.conf.urls import (include,

```

```
21         url)
22
23 from django.contrib import admin
24
25 from ajax_select import urls as ajax_select_urls
26 from singers import views
27
28
29 urlpatterns = [
30     url(r'^admin/', admin.site.urls),
31
32     url(r'^search',
33         view=views.search_form,
34         name='search_form'),
35     url(r'^admin/lookups/', include(ajax_select_urls)),
36
37     url(r'^projects/', include('projects.urls', namespace='projects')),
38     url(r'^singers/', include('singers.urls', namespace='singers')),
39 ]
40
41
42 if settings.DEBUG:
43     from django.contrib.staticfiles.urls import staticfiles_urlpatterns
44     from django.conf.urls.static import static
45     urlpatterns += staticfiles_urlpatterns()
46     # voir p.405 "Django By Example"
47     # https://docs.djangoproject.com/en/dev/howto/static-files/#serving-files-
48     ↪uploaded-by-a-user-during-development
49     urlpatterns += static(settings.MEDIA_URL ,
50                          document_root=settings.MEDIA_ROOT)
51
52     import debug_toolbar
53     urlpatterns += [
54         url(r'^__debug__/', include(debug_toolbar.urls)),
55     ]
```

Add the ProjectUpdateView

```
1
2
3 import logging
4
5 from django.core.urlresolvers import reverse
6
7 from django.views.generic.edit import UpdateView
8 from django.http import HttpResponseRedirect
9
10
11 from .models import Project
12 from .forms import ProjectChampionForm
13
14
15 # Get an instance of a logger
16 logger = logging.getLogger(__name__)
17
18
```

```

19 class ProjectUpdateView(UpdateView):
20     """
21         url(r'^projects/project/(?P<pk>\d+)/update/$', ProjctUpdate.as_view(),
↪ name='project_update'),
22
23     Documentation:
24
25     - http://ccbv.co.uk/projects/Django/1.10/django.views.generic.edit/UpdateView/
26
27     """
28     model = Project
29     form_class = ProjectChampionForm
30     context_object_name = 'project'
31     template_name = 'projects/project/update_easy_simple.html'
32
33     def get_object(self, queryset=None):
34         """Pour mémoriser self.demande_article"""
35         self.object = super(ProjectUpdateView, self).get_object(queryset)
36         return self.object
37
38     def post(self, request, *args, **kwargs):
39         logger.warning("Hello from ProjectUpdateView !")
40         if "cancel" in request.POST:
41             url = self.get_success_url()
42             return HttpResponseRedirect(url)
43         else:
44             return super(ProjectUpdateView, self).post(request, *args, **kwargs)
45
46
47
48
49 class ProjectUpdateViewEasyAutoComplete(UpdateView):
50     """Update thye view with the jQuery EasyAutocomplete plugin.
51
52     Documentation:
53
54     - http://ccbv.co.uk/projects/Django/1.10/django.views.generic.edit/UpdateView/
55
56     """
57     model = Project
58     form_class = ProjectChampionForm
59     context_object_name = 'project'
60     template_name = 'projects/project/update_easyautocomplete.html'
61
62     def get_object(self, queryset=None):
63         """Pour mémoriser self.demande_article"""
64         self.object = super(ProjectUpdateViewEasyAutoComplete, self).get_
↪ object(queryset)
65         return self.object
66
67     def post(self, request, *args, **kwargs):
68         logger.warning("Hello from ProjectUpdateViewEasyAutoComplete !")
69         if "cancel" in request.POST:
70             url = reverse('projects:project_update_easy', kwargs={'pk': self.pk})
71             return HttpResponseRedirect(url)
72         else:
73             return super(ProjectUpdateViewEasyAutoComplete, self).post(request, *args,
↪ **kwargs)

```

```
74
75
76 class ProjectUpdateViewjQueryUIAutoComplete(UpdateView):
77     """Update the view with the jQuery UI Autocomplete plugin.
78
79     Documentation:
80
81     - http://ccbv.co.uk/projects/Django/1.10/django.views.generic.edit/UpdateView/
82
83     """
84     model = Project
85     form_class = ProjectChampionForm
86     context_object_name = 'project'
87     template_name = 'projects/project/update_jquery_ui_autocomplete.html'
88
89     def get_object(self, queryset=None):
90         """Pour mémoriser self.demande_article"""
91         self.object = super(ProjectUpdateViewjQueryUIAutoComplete, self).get_
↪ object(queryset)
92         return self.object
93
94     def post(self, request, *args, **kwargs):
95         logger.warning("Hello from ProjectUpdateViewjQueryUIAutoComplete !")
96         return super(ProjectUpdateViewjQueryUIAutoComplete, self).post(request, *args,
↪ **kwargs)
```

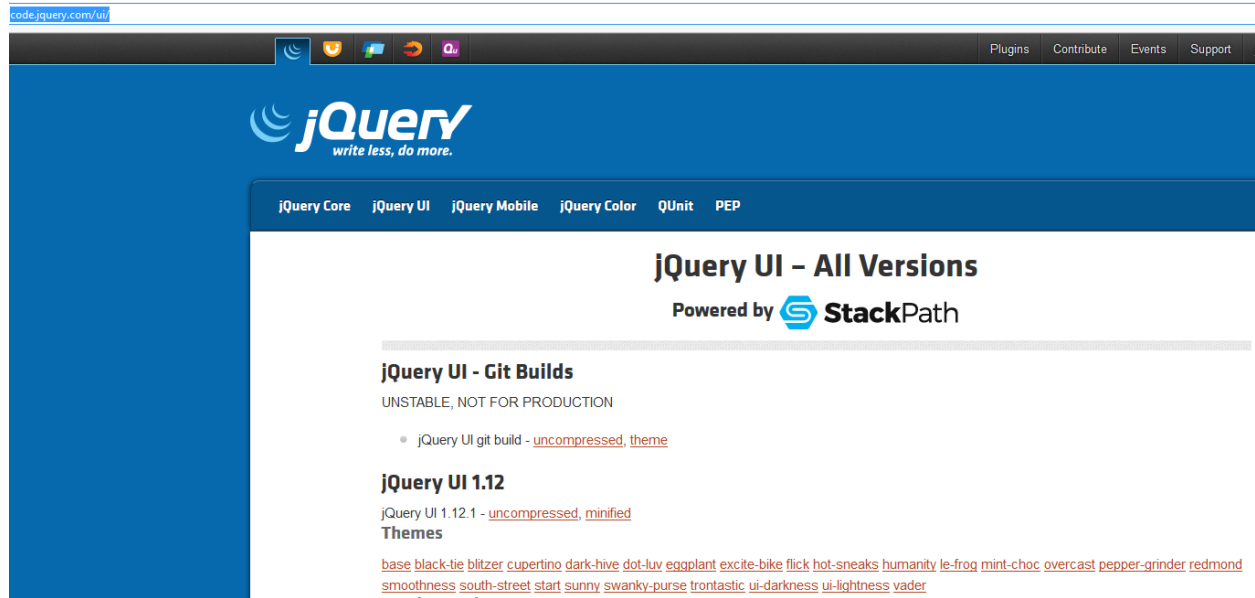
Create the templates/projects/project/update.html django template

See also:

- <http://code.jquery.com/ui/>

This template file is used by the ProjectUpdateView.

```
1  {% extends "base.html" %}
2  {% load static %}
3  {% load staticfiles %}
4
5
6  {% STYLE CSS %}
7  {% block stylesheet %}
8      {{ block.super }}
9      {% https://www.bootstrapcdn.com/ %}
10     <link rel="stylesheet" href="http://code.jquery.com/ui/1.12.1/themes/base/jquery-
↪ ui.css">
11
12     <style>
13         {% https://jqueryui.com/autocomplete/#maxheight %}
14         .ui-autocomplete {
15             max-height: 100px;
16             overflow-y: auto;
17             /* prevent horizontal scrollbar */
18             overflow-x: hidden;
```



```

19     }
20     * html .ui-autocomplete {
21         height: 100px;
22     }
23 </style>
24
25 {% endblock %}
26
27
28 {% block content %}
29
30     <h1>Update of the project '(title:{{ project.title }} champion:{{ project.
31     ↪champion }})' </h1>
32     <p></p>
33     <p></p>
34     {# https://docs.djangoproject.com/en/dev/topics/forms/ #}
35     <form id="id_form_project_update" action="{% url 'projects:project_update' _
36     ↪project.id %}" method="post">
37         {% csrf_token %}
38         <div class="forms">
39             {{ form.id }}
40             {{ form.non_field_errors }}
41             {# Include the hidden fields #}
42             {% for hidden in form.hidden_fields %}
43                 {{ hidden }}
44             {% endfor %}
45             <table id="id_table" class="table table-hover table-bordered table-
46     ↪condensed">
47                 <tbody>
48                     <tr>
49                         <td class="text-right">Title:</td>
50                         <td>{{ form.title }}</td>
51                     </tr>
52                     <tr>
53                         <td class="text-right">Champion:</td>
54                         <td> <span id="id_champion_display">{{ project.champion }}
55     ↪</span> {{ form.champion_term }} </td>

```

```

52         </tr>
53     </tbody>
54 </table>
55 </div>
56 <input type="submit" name="btn_update" value="Update" class="btn btn-success_
↪btn-block" />
57 </form>
58
59
60 {% if messages %}
61 <ul class="messages">
62     {% comment %} https://getbootstrap.com/components/#alerts {% endcomment %}
63     {% for message in messages %}
64 <!--li{% if message.tags %} class="{% message.tags %}"{% endif %} -->
65         {% if message.level == DEFAULT_MESSAGE_LEVELS.ERROR %}
66             <div class="alert alert-danger" role="alert">{{ message }}</div>
67         {% elif message.level == DEFAULT_MESSAGE_LEVELS.SUCCESS %}
68             <div class="alert alert-success" role="alert">{{ message }}</div>
69         {% else %}
70             <div class="alert alert-info" role="alert">{{ message }}</div>
71         {% endif %}
72     <!-- /li -->
73     {% endfor %}
74 </ul>
75 {% endif %}
76
77
78 {% endblock content %}
79
80
81 {% # BEHAVIOR Javascript #}
82 {% block javascript %}
83     {{ block.super }}
84
85     {% # http://code.jquery.com/ui/ #}
86     <script src="http://code.jquery.com/ui/1.12.1/jquery-ui.min.js"></script>
87
88 {% endblock javascript %}
89
90
91 {% # DOM ready #}
92 {% block domready %}
93
94     $( "#id_champion_term" ).autocomplete({
95         {% # http://api.jqueryui.com/autocomplete/#option-source #}
96         source: "{% url 'projects:champion_get_json' %}",
97         {% # http://api.jqueryui.com/autocomplete/#option-autoFocus #}
98         autoFocus:true,
99         {% # http://api.jqueryui.com/autocomplete/#option-minLength #}
100        minLength:1,
101        {% # http://api.jqueryui.com/autocomplete/#event-select #}
102        select:function(event,ui) {
103            $("#id_champion").val(ui.item.id);
104        }
105    });
106
107 {% endblock %}

```


Add the jquery-ui javascript code

See also:

- <http://api.jqueryui.com/autocomplete/>
- <http://api.jqueryui.com/autocomplete/#option-source>
- <http://api.jqueryui.com/autocomplete/#option-minLength>
- <http://api.jqueryui.com/autocomplete/#event-select>

```
$( "#id_champion_display" ).autocomplete({
    source: "{% url champion_auto_complete %}",
    minLength:2,
    select:function(event,ui) {
        $("#id_champion").val(ui.item.id)
    }
});
```



Add the admin interface

```
1  #!/usr/bin/python
2  # -*- coding: UTF-8 -*-
3  """Project Administration.
4
5  """
6
7  from django.contrib import admin
8
9  from .models import Project
10
11 from .forms_django_autocomplete_light import ProjectFormDjangoAutocomplete
12
13 @admin.register(Project)
14 class ProjectAdmin(admin.ModelAdmin):
15     """Project administration
16
17     Documentation
18     =====
19
20     - https://docs.djangoproject.com/en/dev/ref/contrib/admin/#modeladmin-objects
21
22     """
23     form = ProjectFormDjangoAutocomplete
24     list_display = ('title', 'champion')
25     search_fields = ('title', 'champion')
26     list_filter = ('title', 'champion')
```

Create some projects with the admin interface

Reading Project record from the commande line

```
python manage.py shell_plus
```

  127.0.0.1:8000/admin/

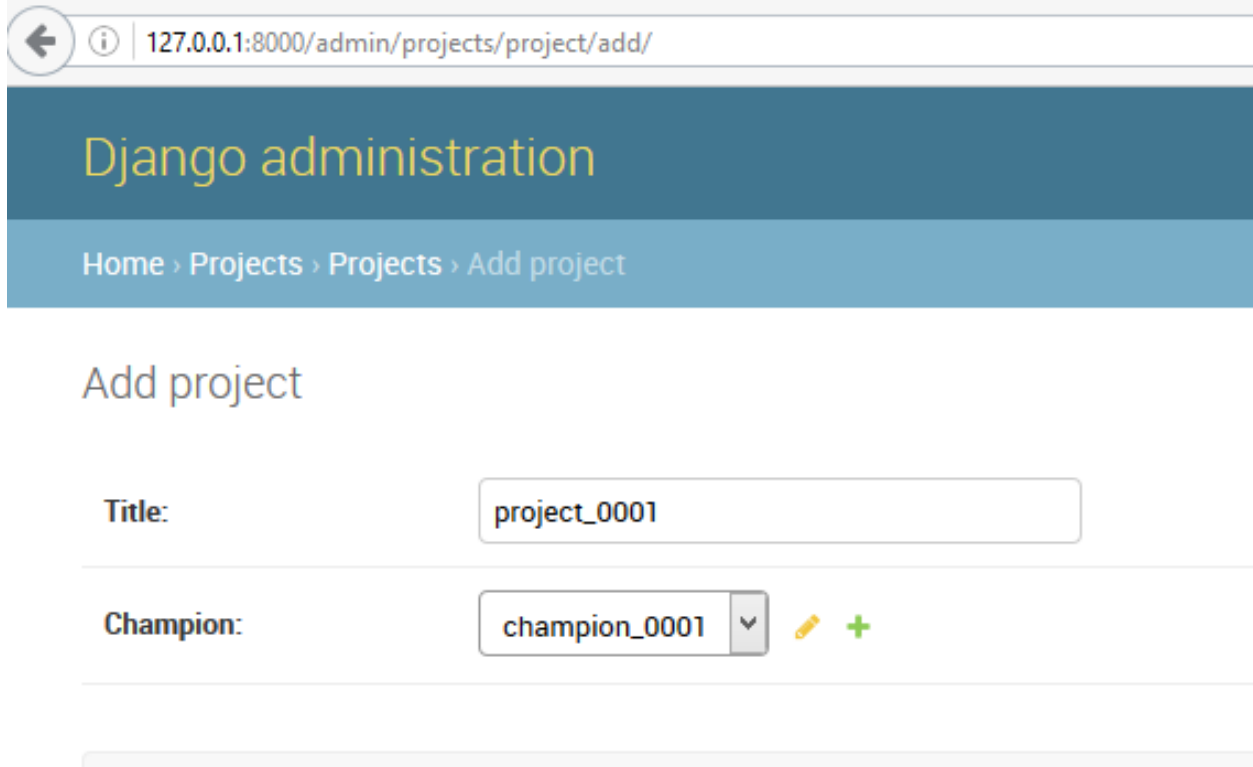
Django administration

Site administration

AUTHENTICATION AND AUTHORIZATION	
Groups	+ Add ✎ Change
Users	+ Add ✎ Change

PROJECTS	
Projects	+ Add ✎ Change

SINGERS	
Authors	+ Add ✎ Change
Groups	+ Add ✎ Change
Labels	+ Add ✎ Change
Persons	+ Add ✎ Change
Releases	+ Add ✎ Change
Songs	+ Add ✎ Change






127.0.0.1:8000/admin/projects/project/add/

Django administration

Home > Projects > Projects > Add project

Add project

Title:

Champion:   

```
(ajax_django_35) C:\projects_id3\django_ajax_select\projet_ajax>python manage.py_
↪shell_plus
# Shell Plus Model Imports
from django.contrib.admin.models import LogEntry
from django.contrib.auth.models import Group, Permission, User
from django.contrib.contenttypes.models import ContentType
from django.contrib.sessions.models import Session
from projects.models import Project
from singers.models import Author, Book, Group, Label, Person, Release, Song
# Shell Plus Django Imports
from django.db.models import Avg, Case, Count, F, Max, Min, Prefetch, Q, Sum, When
from django.db import transaction
from django.conf import settings
from django.utils import timezone
from django.urls import reverse
from django.core.cache import cache
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:18:55) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

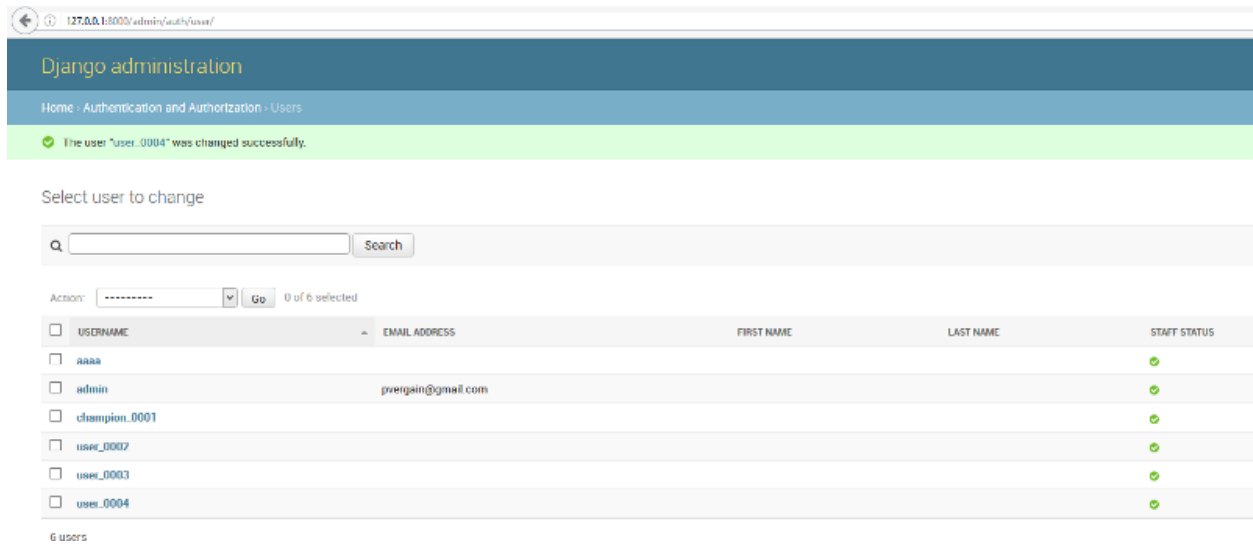
IPython 5.1.0 -- An enhanced Interactive Python.
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.

In [1]: p1=Project.objects.first()

In [2]: p1
Out[2]: <Project: project_0001 champion_0001>
```

```
In [3]: p1.id
Out[3]: 1
```

Add some users with the admin interface



Add some users with the command line interface

See also:

- <https://docs.djangoproject.com/en/dev/topics/auth/default/#user-objects>

The most direct way to create users is to use the included `create_user()` helper function:

```
In [4]: user = User.objects.create_user('john', 'lennon@thebeatles.com', 'johnpassword')
↪')

In [5]: user = User.objects.create_user('albert', 'albert@thebeatles.com',
↪'albertpassword')

In [6]: user = User.objects.create_user('zoya', 'zoya@thebeatles.com', 'zoyapassword')

In [7]: user = User.objects.create_user('nigel', 'nigel@thebeatles.com',
↪'nigelpassword')

In [8]: users=User.objects.all()

In [9]: users
Out[9]: <QuerySet [<User: admin>, <User: champion_0001>, <User: user_0002>, <User:
↪user_0003>, <User: user_0004>, <User: aaaa>, <User: john>, <User: albert>, <User:
↪zoya>, <User: nigel>]>
```

Try the URL : <http://127.0.0.1:8000/projects/project/1/update>

OK, this time this works fine !

Warning: La saisie est agréable, il faut améliorer le look.

127.0.0.1:8004/projects/project/1/update/

Update of the project '(title:project_0001 champion:zoya)

Title: Update the champion:

type username or email

zoya

Update

```
[18/Oct/2016 15:17:26] "GET /projects/project/1/update/ HTTP/1.1" 200 3423
[18/Oct/2016 15:17:44] "GET /projects/champion_auto_complete/?term=zo HTTP/1.1" 200 45
[18/Oct/2016 15:17:51] "POST /projects/project/1/update/ HTTP/1.1" 302 0
[18/Oct/2016 15:17:51] "GET /projects/project/1/update/ HTTP/1.1" 200 3418
[18/Oct/2016 15:18:47] "GET /projects/champion_auto_complete/?term=ni HTTP/1.1" 200 48
[18/Oct/2016 15:19:00] "POST /projects/project/1/update/ HTTP/1.1" 302 0
[18/Oct/2016 15:19:00] "GET /projects/project/1/update/ HTTP/1.1" 200 3420
```

2016-10-18 create some Django Templates : base.html

See also:

- <https://docs.djangoproject.com/en/dev/ref/templates/language/#>

Contents

- *2016-10-18 create some Django Templates : base.html*
 - *The Django Template templates/base.html*
 - *The Django Template singers/templates/song/update.html*
 - *Add the django-extensions and ipython modules*
 - * *New django commands*
 - *Get the first song*
 - *Try the URL : `http://127.0.0.1:8000/singers/song/1/update`*
 - *Compared to the django admin interface*
 - *Conclusion*

The Django Template templates/base.html

```
1 {% https://docs.djangoproject.com/en/dev/ref/templates/builtins/#std:templatetag-
   ↳ static %}
2 {% load static %}
3 {% https://docs.djangoproject.com/en/dev/ref/templates/builtins/#tz %}
4 {% load tz %}
5
6 <!DOCTYPE html>
```

```

7  {# https://www.w3.org/TR/2014/REC-html5-20141028/syntax.html#the-doctype #}
8  <!--[if lt IE 7]>      <html class="no-js lt-ie9 lt-ie8 lt-ie7"> <![endif]-->
9  <!--[if IE 7]>        <html class="no-js lt-ie9 lt-ie8"> <![endif]-->
10 <!--[if IE 8]>        <html class="no-js lt-ie9"> <![endif]-->
11 <!--[if gt IE 8]><!--> <html class="no-js"> <!--<![endif]-->
12 {# https://www.w3.org/TR/html5/semantics.html#the-html-element #}
13 <head>
14     {# https://www.w3.org/TR/html5/document-metadata.html#attr-meta-charset #}
15     <meta charset="utf-8">
16     {# https://n.survol.fr/n/x-ua-compatible-ieedge #}
17     <!-- meta http-equiv="X-UA-Compatible" content="IE=edge" -->
18     {# https://www.w3.org/TR/html5/dom.html#the-title-attribute #}
19     {# https://www.w3.org/TR/html5/document-metadata.html#the-title-element #}
20     <title>
21         {% block head_title %}
22         {% endblock %}
23     </title>
24
25     {# https://www.w3.org/TR/html5/document-metadata.html#attr-meta-name #}
26
27     {# https://developer.mozilla.org/fr/docs/Mozilla/Mobile/Balise_meta_viewport
28     {#}
29     <meta name="viewport" content="width=device-width, initial-scale=1" />
30
31     {# CSS #}
32     {% block stylesheet %}
33         {# https://docs.djangoproject.com/en/dev/ref/templates/language/#template-
34         {# inheritance #}
35         {# If you need to get the content of the block from the parent template,
36         {# the {{ block.super }} variable will do the trick #}
37         {# {{ block.super }} #}
38
39         {# https://blog.getbootstrap.com/2016/07/25/bootstrap-3-3-7-released_
40         {# (2016-07-25) Added support for jQuery 3 #}
41         {# https://www.bootstrapcdn.com/ #}
42         <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.
43         {# 3.7/css/bootstrap.min.css">
44         <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.
45         {# 3.7/css/bootstrap-theme.min.css">
46
47         {% endblock %}
48
49
50 </head>
51 <body class="{% block bodyclass %} {% endblock %}">
52
53     <div id="conteneur" class="container">
54         {% block content %}
55         {% endblock %}
56
57         {% block footer %}
58         {% endblock %}
59
60     </div> <!-- id conteneur -->

```

```

58     {% Javascript %}
59     {% block javascript %}
60
61         {# a employer dans les templates fils->{{ block.super }} #}
62         {# https://docs.djangoproject.com/en/dev/ref/templates/language/#template-
63         ↳inheritance #}
64         {# If you need to get the content of the block from the parent template,
65         ↳the {{ block.super }} variable will do the trick #}
66         <!-- Liaison aux scripts Javascript -->
67         <!-- CDN JQuery Placer jquery avant bootstrap, select2 -->
68         {# https://github.com/jquery/jquery-migrate/#README #}
69         {# https://blog.jquery.com/2016/09/22/jquery-3-1-1-released/ #}
70         {# https://code.jquery.com/ #}
71         <script src="https://code.jquery.com/jquery-3.1.1.min.js"></script>
72         <script src="https://code.jquery.com/jquery-migrate-3.0.0.min.js"></
73         ↳script>
74
75         <!-- Code javascript pour bootstrap 3.3.7 (2016-07-25) -->
76         <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.
77         ↳min.js"> </script>
78
79         {% endblock javascript %}
80
81         {# p.155 Adding AJAX actions with jQuery, Django By Example from Antonio Melé
82         ↳#}
83         {# https://www.amazon.com/Django-Example-Antonio-Mele/dp/1784391913/ #}
84         <script>
85             $(document).ready(function() {
86                 {% block domready %}
87                 {% endblock %}
88             });
89         </script>
90
91     </body>
92 </html>

```

```

|   ajax_selects_singers_db
|   manage.py
|
+---projet_ajax
|   |   settings.py
|   |   urls.py
|   |   wsgi.py
|   |   __init__.py
|
+---singers
|   |   admin.py
|   |   apps.py
|   |   forms.py
|   |   lookups.py
|   |   models.py
|   |   tests.py
|   |   urls.py
|   |   views.py
|   |   __init__.py
|   |
+---migrations

```

```
| | | 0001_initial.py
| | | 0002_auto_20161017_1612.py
| | | 0003_auto_20161017_1632.py
| | | __init__.py
| |
|
\---templates
    base.html
    search_form.html
```

The Django Template `singers/templates/song/update.html`

See also:

- <https://docs.djangoproject.com/en/dev/ref/templates/builtins/#std:templatetag-extends>

The template `templates/singers/song/update.html` extends the `templatesbase.html` **base template**.

```
1  {% extends "base.html" %}
2  {% load static %}
3  {% load staticfiles %}
4  {% load crispy_forms_tags %}
5
6  {% block content %}
7
8  <h1>Update of the song {{ song.title }} {{ song.group }} </h1>
9
10  {# https://docs.djangoproject.com/en/dev/topics/forms/ #}
11  <form id="song_update" method="post" action=".">
12      {% csrf_token %}
13      {{ form }}
14
15      {# https://django-ajax-selects.readthedocs.io/en/latest/Outside-of-Admin.html #}
16      {{ form.media }}
17      <input type="submit" name="btn_update" value="Update" class="btn btn-success btn-
18  ↪block" />
19  </form>
20
21  {% if messages %}
22      <ul class="messages">
23          {% comment %} https://getbootstrap.com/components/#alerts {% endcomment %}
24          {% for message in messages %}
25              <!--li{% if message.tags %} class="{{ message.tags }}" {% endif %} -->
26              {% if message.level == DEFAULT_MESSAGE_LEVELS.ERROR %}
27                  <div class="alert alert-danger" role="alert">{{ message }}</div>
28              {% elif message.level == DEFAULT_MESSAGE_LEVELS.SUCCESS %}
29                  <div class="alert alert-success" role="alert">{{ message }}</div>
30              {% else %}
31                  <div class="alert alert-info" role="alert">{{ message }}</div>
32              {% endif %}
33              <!-- /li -->
34              {% endfor %}
35      </ul>
36  {% endif %}
37
```



```
38 {% endblock content %}
```

Add the django-extensions and ipython modules

```
pip install django-extensions ipython
```

In the `projct_ajax/settings.py` file:

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
  
    # https://django-extensions.readthedocs.org/en/latest  
    'django_extensions',  
]
```

New django commands

```
[django_extensions]  
admin_generator  
clean_pyc  
clear_cache  
compile_pyc  
create_app  
create_command  
create_jobs  
create_template_tags  
describe_form  
drop_test_database  
dumpscrip  
export_emails  
find_template  
generate_secret_key  
graph_models  
mail_debug  
notes  
passwd  
pipchecker  
print_settings  
print_user_for_session  
reset_db  
runjob  
runjobs  
runprofiles  
runscript  
runserver_plus  
set_default_site  
set_fake_emails  
set_fake_passwords  
shell_plus
```

```
show_template_tags
show_templatetags
show_urls
sqlcreate
sqldiff
sqldsn
sync_s3
syncdata
unreferenced_files
update_permissions
validate_templates
```

The *shell_plus* command is very usefull.

```
(ajax_django_35) C:\projects_id3\django_ajax_select\projet_ajax>python manage.py shell_plus
# Shell Plus Model Imports
from django.contrib.admin.models import LogEntry
from django.contrib.auth.models import Group, Permission, User
from django.contrib.contenttypes.models import ContentType
from django.contrib.sessions.models import Session
from singers.models import Author, Book, Group, Label, Person, Release, Song
# Shell Plus Django Imports
from django.utils import timezone
from django.db.models import Avg, Case, Count, F, Max, Min, Prefetch, Q, Sum, When
from django.urls import reverse
from django.conf import settings
from django.db import transaction
from django.core.cache import cache
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:18:55) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license()" for more information.

IPython 5.1.0 -- An enhanced Interactive Python.
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.

In [1]: █
Ready
```

Get the first song

```
In [1]: first_song = Song.objects.all().first()

In [2]: first_song.id
Out[2]: 1
```

Try the URL : <http://127.0.0.1:8000/singers/song/1/update>

See also:

- <http://127.0.0.1:8000/singers/song/1/update>

Not very nice and not very easy to use.

Compared to the django admin interface

See also:

- <http://127.0.0.1:8000/admin/singers/song/1/change/>

More nice but not very easy to use.

127.0.0.1:8000/singers/song/1/update/

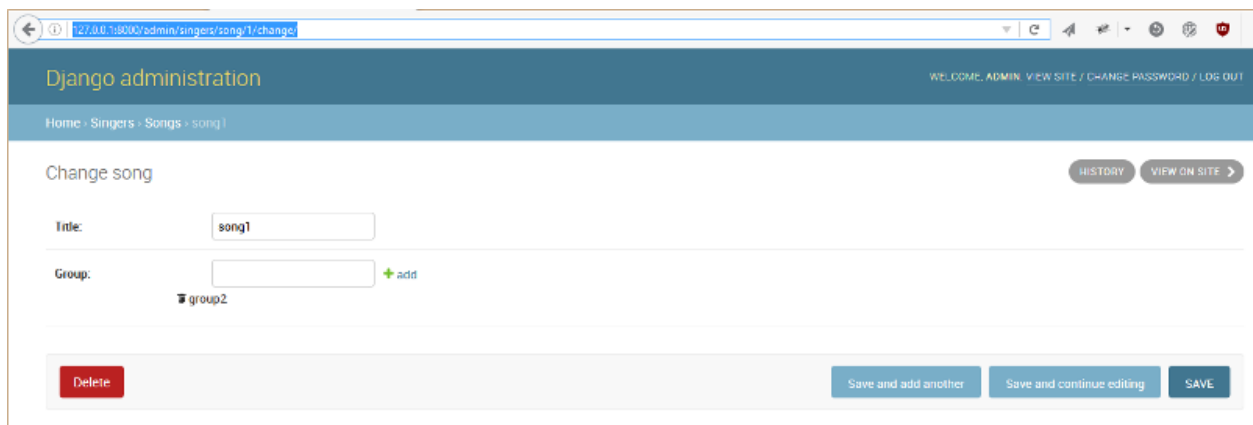
Update of the song song1

Title: Group:

 group2

Select the group

Update



The screenshot shows the Django administration interface in a web browser. The address bar displays the URL `127.0.0.1:8000/admin/singers/song/1/change/`. The page header includes the text "Django administration" and a welcome message for the admin user. The breadcrumb trail indicates the current location: "Home > Singers > Songs > song1". The main content area is titled "Change song" and contains a form with two fields: "Title" (with the value "song1") and "Group" (with a dropdown menu showing "group2" and a "+ add" link). At the bottom of the form, there are four buttons: "Delete" (in a red box), "Save and add another", "Save and continue editing", and "SAVE".

Conclusion

Try the django-autocomplete-light module.

2016-10-17 create a normal Django view

Contents

- *2016-10-17 create a normal Django view*
 - *Le fichier singers/models.py*
 - *Le fichier singers/forms.py*
 - *Le fichier singers/views.py*
 - *Le fichier singers/urls.py*

Le fichier singers/models.py

```
1  #!/usr/bin/python
2  # -*- coding: utf8 -*-
3  """The singers models.
4
5  """
6  from __future__ import unicode_literals
7  from django.utils.encoding import python_2_unicode_compatible
8
9  from django.db import models
10 from django.core.urlresolvers import reverse
11
12
13 @python_2_unicode_compatible
14 class Person(models.Model):
15     """ an actual singular human being """
16     name = models.CharField(blank=True, max_length=100)
17     email = models.EmailField()
18
19     def __str__(self):
20         return self.name
21
22
23 @python_2_unicode_compatible
24 class Group(models.Model):
25     """ a music group """
26     name = models.CharField(max_length=200,
27                             unique=True,
28                             help_text="Name of the group")
29     members = models.ManyToManyField(Person,
30                                     blank=True,
31                                     help_text="Enter text to search for and add each member of the group.")
32     url = models.URLField(blank=True)
33
34     def __str__(self):
35         return self.name
```

```

36
37
38 @python_2_unicode_compatible
39 class Label(models.Model):
40     """ a record label """
41     name = models.CharField(max_length=200,
42                             unique=True)
43
44     owner = models.ForeignKey(Person,
45                               blank=True,
46                               null=True)
47
48     url = models.URLField(blank=True)
49
50     def __str__(self):
51         return self.name
52
53 @python_2_unicode_compatible
54 class Song(models.Model):
55     """ a song """
56     title = models.CharField(blank=False, max_length=200)
57     group = models.ForeignKey(Group)
58
59     def __str__(self):
60         return self.title
61
62     def get_absolute_url(self):
63         """
64         https://docs.djangoproject.com/en/dev/ref/class-based-views/generic-editing/
65         """
66         return reverse('singers:song_update',
67                       kwargs={'pk': self.pk})
68
69
70 @python_2_unicode_compatible
71 class Release(models.Model):
72     """ a music release/product """
73     title = models.CharField(max_length=100)
74
75     catalog = models.CharField(blank=True,
76                                max_length=100)
77
78     group = models.ForeignKey(Group, blank=True, null=True,
79                               verbose_name=" (group) ")
80
81     label = models.ForeignKey(Label,
82                               blank=False,
83                               null=False)
84
85     songs = models.ManyToManyField(Song,
86                                    blank=True)
87
88     def __str__(self):
89         return self.title
90
91 @python_2_unicode_compatible
92 class Author(models.Model):
93     """ Author has multiple books,

```

```
94         via foreign keys
95         """
96         name = models.CharField(max_length=100)
97
98         def __str__(self):
99             return self.name
100
101
102 @python_2_unicode_compatible
103 class Book(models.Model):
104     """ Book has no admin, its an inline in the Author admin """
105
106     author = models.ForeignKey(Author)
107     title = models.CharField(max_length=100)
108     about_group = models.ForeignKey(Group)
109     mentions_persons = models.ManyToManyField(Person,
110                                                help_text="Person lookup renders html_
111 ↪in menu")
112
113     def __str__(self):
114         return self.title
```

Le fichier singers/forms.py

```
1  # -*- coding: utf-8 -*-
2
3  from __future__ import unicode_literals
4
5  from django.forms.models import ModelForm
6  from ajax_select import make_ajax_field
7  from dal import autocomplete
8
9  from .models import (Release,
10                      Song,
11                      Author,
12                      Book)
13
14  class ReleaseForm(ModelForm):
15
16      class Meta:
17          model = Release
18          exclude = []
19
20          # args: this model, fieldname on this model, lookup_channel_name
21          group = make_ajax_field(Release, 'group', 'group', show_help_text=True)
22
23          label = make_ajax_field(Release, 'label', 'label', help_text="Search for label by_
24 ↪name")
25
26          # any extra kwargs are passed onto the field, so you may pass a custom help_text_
27 ↪here
28          # songs = make_ajax_field(Release, 'songs', 'song', help_text=u"Search for song by_
29 ↪title")
30
31          # testing bug with no help text supplied
```

```

29     songs = make_ajax_field(Release, 'songs', 'song', help_text="", show_help_
↪text=True)
30
31     # these are from a fixed array defined in lookups.py
32     title = make_ajax_field(Release, 'title', 'cliche', help_text="Autocomplete will_
↪suggest clichés about cats.")
33
34
35 class SongForm(ModelForm):
36     """The Django form"""
37
38     class Meta:
39         model = Song
40         fields = ['title', 'group']
41
42     # args:  this model, fieldname on this model, lookup_channel_name
43     group = make_ajax_field(Song, # this model
44                             'group', # fieldname on this model
45                             'group', # lookup_channel_name
46                             help_text="Select the group")
47
48
49 class AuthorForm(ModelForm):
50     class Meta:
51         model = Author
52         fields = ('name',)
53         widgets = {
54             'name': autocomplete.Select(url='singers:author_autocomplete')
55         }
56
57 class BookForm(ModelForm):
58     class Meta:
59         model = Book
60         fields = ('title', 'author',)
61         widgets = {
62             'author': autocomplete.ModelSelect2(url='singers:author_autocomplete')
63         }
64
65

```

Le fichier singers/views.py

```

1  # -*- coding: utf-8 -*-
2
3  from __future__ import unicode_literals
4  from django import forms
5
6  from django.shortcuts import render_to_response
7
8  # https://docs.djangoproject.com/en/dev/ref/templates/api/
9  from django.template import RequestContext
10 from django.http import HttpResponseRedirect
11
12 from django.views.generic.edit import UpdateView
13
14 from ajax_select.fields import AutoCompleteField

```

```
15
16 # https://stackoverflow.com/questions/37915224/django-autocomplete-light-widgets-not-
    ↳ showing-up?rq=1
17 from dal import autocomplete
18
19 from .models import (Song,
20                      Author,
21                      Book)
22
23 from .forms import (SongForm,
24                     AuthorForm,
25                     BookForm)
26
27 class SearchForm(forms.Form):
28
29     q = AutoCompleteField('cliche',
30                           required=True,
31                           help_text="Autocomplete will suggest clichés about cats, but you can enter_
    ↳ anything you like.",
32                           label="Favorite Cliché",
33                           attrs={'size': 100})
34
35
36 def search_form(request):
37
38     dd = {}
39     if 'q' in request.GET:
40         dd['entered'] = request.GET.get('q')
41     initial = {'q': "\"This is an initial value,\" said O'Leary.\""}
42     form = SearchForm(initial=initial)
43     dd['form'] = form
44     return render_to_response('search_form.html',
45                               dd,
46                               context=RequestContext(request))
47
48
49
50 class SongUpdate(UpdateView):
51     """
52     url(r'^singers/song/(?P<pk>\d+)/update/$', SongUpdate.as_view(), name=
    ↳ 'song_update'),
53
54     Documentation:
55
56     - http://ccbv.co.uk/projects/Django/1.9/django.views.generic.edit/UpdateView/
57
58     """
59     model = Song
60     form_class = SongForm
61     context_object_name = 'song'
62     template_name = 'singers/song/update.html'
63
64     def get_object(self, queryset=None):
65         """Pour mémoriser self.demande_article"""
66         self.object = super(SongUpdate, self).get_object(queryset)
67         return self.object
68
69     def post(self, request, *args, **kwargs):
```



```

70         if "cancel" in request.POST:
71             url = self.get_success_url()
72             return HttpResponseRedirect(url)
73         else:
74             return super(SongUpdate, self).post(request, *args, **kwargs)
75
76
77 class AuthorAutocomplete(autocomplete.Select2QuerySetView):
78     def get_queryset(self):
79         # Don't forget to filter out results depending on the visitor !
80
81         qs = Author.objects.all()
82
83         if self.q:
84             qs = qs.filter(name__istartswith=self.q)
85
86         return qs
87
88
89
90 class AuthorUpdate(UpdateView):
91     """
92         url(r'^singers/song/(?P<pk>\d+)/update/$', SongUpdate.as_view(), name=
93         ↪ 'song_update'),
94
95         Documentation:
96
97         - http://ccbv.co.uk/projects/Django/1.9/django.views.generic.edit/UpdateView/
98
99     """
100     model = Author
101     form_class = AuthorForm
102     context_object_name = 'author'
103     template_name = 'singers/author/update.html'
104
105     def get_object(self, queryset=None):
106         """Pour mémoriser self.demande_article"""
107         self.object = super(AuthorUpdate, self).get_object(queryset)
108         return self.object
109
110     def post(self, request, *args, **kwargs):
111         if "cancel" in request.POST:
112             url = self.get_success_url()
113             return HttpResponseRedirect(url)
114         else:
115             return super(AuthorUpdate, self).post(request, *args, **kwargs)
116
117 class BookUpdate(UpdateView):
118     """
119         url(r'^singers/song/(?P<pk>\d+)/update/$', SongUpdate.as_view(), name=
120         ↪ 'song_update'),
121
122         Documentation:
123
124         - http://ccbv.co.uk/projects/Django/1.9/django.views.generic.edit/UpdateView/
125
126     """
127     model = Book

```

```
126 form_class = BookForm
127 context_object_name = 'author'
128 template_name = 'singers/author/update.html'
129
130 def get_object(self, queryset=None):
131     """Pour mémoriser self.demande_article"""
132     self.object = super(BookUpdate, self).get_object(queryset)
133     return self.object
134
135 def post(self, request, *args, **kwargs):
136     if "cancel" in request.POST:
137         url = self.get_success_url()
138         return HttpResponseRedirect(url)
139     else:
140         return super(BookUpdate, self).post(request, *args, **kwargs)
```

Le fichier singers/urls.py

```
1  #!/usr/bin/python
2  # -*- coding: utf8 -*-
3
4  from django.conf.urls import url
5  from .views import (SongUpdate,
6                     BookUpdate,
7                     AuthorAutocomplete)
8
9
10 urlpatterns = [
11     url(r'^song/(?P<pk>\d+)/update/$', SongUpdate.as_view(), name='song_update'),
12     url(r'^book/(?P<pk>\d+)/update/$', BookUpdate.as_view(), name='book_update'),
13     url(r'^author_autocomplete/$', AuthorAutocomplete.as_view(), name='author_
14     ↪autocomplete'),
15 ]
```

2016-10-17 create singers SQL tables

Contents

- 2016-10-17 create singers SQL tables
 - makemigrations singers
 - sqlmigrate singers 0001
 - migrate singers
 - show-migrations
 - createsuperuser
 - Connect to the admin interface

The

makemigrations singers

```
(ajax_django_35) projet_ajax>python manage.py makemigrations singers
```

```
Migrations for 'singers':
  singers\migrations\0001_initial.py:
    - Create model Author
    - Create model Book
    - Create model Group
    - Create model Label
    - Create model Person
    - Create model Release
    - Create model Song
    - Add field songs to release
    - Add field owner to label
    - Add field members to group
    - Add field about_group to book
    - Add field author to book
    - Add field mentions_persons to book
```

sqlmigrate singers 0001

```
(ajax_django_35) C:\projects_id3\django_ajax_select\projet_ajax>python manage.py
↪sqlmigrate singers 0001
```

```
BEGIN;
--
-- Create model Author
--
CREATE TABLE "singers_author" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name"
↪" varchar(100) NOT NULL);
--
-- Create model Book
--
CREATE TABLE "singers_book" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "title"
↪varchar(100) NOT NULL);
--
-- Create model Group
--
CREATE TABLE "singers_group" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name"
↪varchar(200) NOT NULL UNIQUE, "url" varchar(200) NOT NULL);
--
-- Create model Label
--
CREATE TABLE "singers_label" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name"
↪varchar(200) NOT NULL UNIQUE, "url" varchar(200) NOT NULL);
--
-- Create model Person
--
CREATE TABLE "singers_person" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name"
↪" varchar(100) NOT NULL, "email" varchar(254) NOT NULL);
--
-- Create model Release
--
CREATE TABLE "singers_release" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
↪"title" varchar(100) NOT NULL, "catalog" varchar(100) NOT NULL,
```

```
"group_id" integer NULL REFERENCES "singers_group" ("id"), "label_id" integer NOT_
↪NULL REFERENCES "singers_label" ("id"));
--
-- Create model Song
--
CREATE TABLE "singers_song" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "title"
↪varchar(200) NOT NULL, "group_id" integer NOT NULL REFERENCES "singers_group" ("id
↪"));
--
-- Add field songs to release
--
CREATE TABLE "singers_release_songs" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
↪ "release_id" integer NOT NULL REFERENCES "singers_release" ("id"), "song_id"
↪integer NOT NULL REFERENCES "singers_song" ("id"));
--
-- Add field owner to label
--
ALTER TABLE "singers_label" RENAME TO "singers_label__old";
CREATE TABLE "singers_label" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name"
↪varchar(200) NOT NULL UNIQUE, "url" varchar(200) NOT NULL,
"owner_id" integer NULL REFERENCES "singers_person" ("id"));
INSERT INTO "singers_label" ("url", "id", "owner_id", "name") SELECT "url", "id",
↪NULL, "name" FROM "singers_label__old";
DROP TABLE "singers_label__old";
CREATE INDEX "singers_release_0e939a4f" ON "singers_release" ("group_id");
CREATE INDEX "singers_release_abec2aca" ON "singers_release" ("label_id");
CREATE INDEX "singers_song_0e939a4f" ON "singers_song" ("group_id");
CREATE UNIQUE INDEX "singers_release_songs_release_id_be8581cc_uniq" ON "singers_
↪release_songs" ("release_id", "song_id");
CREATE INDEX "singers_release_songs_b07ccb57" ON "singers_release_songs" ("release_id
↪");
CREATE INDEX "singers_release_songs_e5c27d73" ON "singers_release_songs" ("song_id");
CREATE INDEX "singers_label_5e7b1936" ON "singers_label" ("owner_id");
--
-- Add field members to group
--
CREATE TABLE "singers_group_members" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
↪ "group_id" integer NOT NULL REFERENCES "singers_group" ("id"), "person_id" integer
↪NOT NULL REFERENCES "singers_person" ("id"));
--
-- Add field about_group to book
--
ALTER TABLE "singers_book" RENAME TO "singers_book__old";
CREATE TABLE "singers_book" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "title"
↪varchar(100) NOT NULL, "about_group_id" integer NOT NULL REFERENCES "singers_group"
↪("id"));
INSERT INTO "singers_book" ("title", "id", "about_group_id") SELECT "title", "id",
↪NULL FROM "singers_book__old";
DROP TABLE "singers_book__old";
CREATE UNIQUE INDEX "singers_group_members_group_id_6767fcfl_uniq" ON "singers_group_
↪members" ("group_id", "person_id");
CREATE INDEX "singers_group_members_0e939a4f" ON "singers_group_members" ("group_id");
CREATE INDEX "singers_group_members_a8452ca7" ON "singers_group_members" ("person_id
↪");
CREATE INDEX "singers_book_13ef9fae" ON "singers_book" ("about_group_id");
--
-- Add field author to book
--
```

```

ALTER TABLE "singers_book" RENAME TO "singers_book__old";
CREATE TABLE "singers_book" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "title"
↪ varchar(100) NOT NULL, "about_group_id" integer NOT NULL REFERENCES "singers_group"
↪ ("id"), "author_id" integer NOT NULL REFERENCES "singers_author" ("id"));
INSERT INTO "singers_book" ("author_id", "title", "id", "about_group_id") SELECT NULL,
↪ "title", "id", "about_group_id" FROM "singers_book__old";
DROP TABLE "singers_book__old";
CREATE INDEX "singers_book_13ef9fae" ON "singers_book" ("about_group_id");
CREATE INDEX "singers_book_4f331e2f" ON "singers_book" ("author_id");
--
-- Add field mentions_persons to book
--
CREATE TABLE "singers_book_mentions_persons" ("id" integer NOT NULL PRIMARY KEY
↪ AUTOINCREMENT, "book_id" integer NOT NULL REFERENCES "singers_book" ("id"), "person_
↪ id" integer NOT NULL REFERENCES "singers_person" ("id"));
CREATE UNIQUE INDEX "singers_book_mentions_persons_book_id_873aadf4_uniq" ON "singers_
↪ book_mentions_persons" ("book_id", "person_id");
CREATE INDEX "singers_book_mentions_persons_0a4572cc" ON "singers_book_mentions_
↪ persons" ("book_id");
CREATE INDEX "singers_book_mentions_persons_a8452ca7" ON "singers_book_mentions_
↪ persons" ("person_id");
COMMIT;

```

migrate singers

See also:

- <https://docs.djangoproject.com/en/dev/intro/tutorial02/>
- <https://docs.djangoproject.com/fr/1.10/intro/tutorial02/>

```

 ajax_django_35) C:\projects_id3\django_ajax_select\projet_ajax>python manage.py
↪ migrate singers

```

```

Operations to perform:
  Apply all migrations: singers
Running migrations:
  Applying singers.0001_initial... OK

```

show-migrations

```

 ajax_django_35) C:\projects_id3\django_ajax_select\projet_ajax>python manage.py
↪ showmigrations

```

```

admin
[X] 0001_initial
[X] 0002_logentry_remove_auto_add
auth
[X] 0001_initial
[X] 0002_alter_permission_name_max_length
[X] 0003_alter_user_email_max_length
[X] 0004_alter_user_username_opts
[X] 0005_alter_user_last_login_null
[X] 0006_require_contenttypes_0002

```

```
[X] 0007_alter_validators_add_error_messages
[X] 0008_alter_user_username_max_length
contenttypes
[X] 0001_initial
[X] 0002_remove_content_type_name
sessions
[X] 0001_initial
singers
[X] 0001_initial
```

createsuperuser

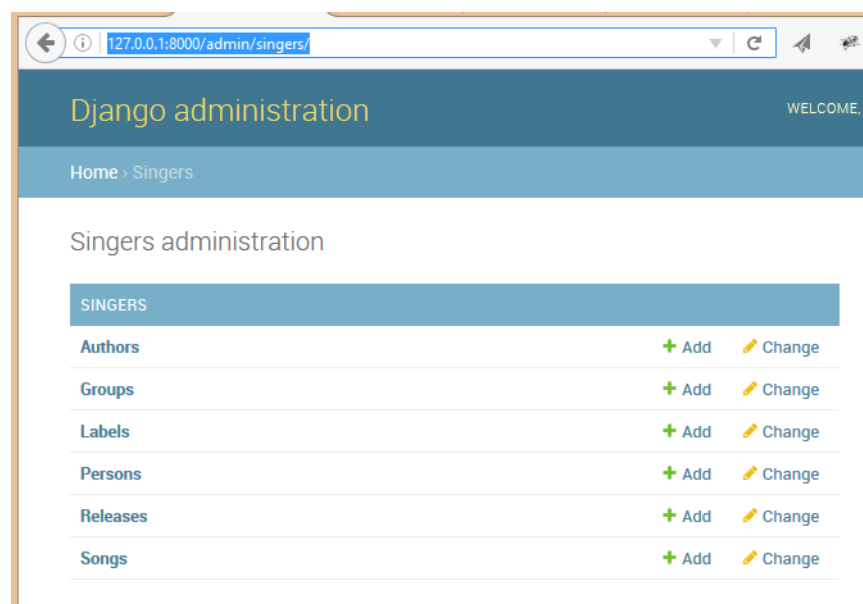
```
(ajax_django_35) C:\projects_id3\django_ajax_select\projet_ajax>python manage.py_
↪createsuperuser
```

```
Username (leave blank to use 'pvergain'): admin
Email address: pvergain@gmail.com
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
Password:
Password (again):
The password is too similar to the email address.
Password:
Password (again):
Superuser created successfully.
```

Connect to the admin interface

See also:

<http://127.0.0.1:8000/admin/>



Django test AJAX autocomplete development

Github Singers's development

See also:

- <https://swcarpentry.github.io/git-novice/>
- <https://github.com/github/gitignore>

Contents

- *Github Singers's development*
 - *Create new project*
 - *Create singers project*
 - *Git remote*

Create new project

Create singers project

Git remote

See also:

<https://swcarpentry.github.io/git-novice/07-github/>

https://github.com/new

Search GitHub Pull requests Issues Gist

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: pvergain / Repository name: singers ✓

Great repository names are short and memorable. Need inspiration? How about [bookish-waddle](#).

Description (optional): A simple Django application to test django-ajax-selects

☒ Public
Anyone can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

☐ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None Add a license: None ⓘ

Create repository

https://github.com/pvergain/singers

Error: Empty repository

This repository is empty.

pvergain / singers

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH <https://github.com/pvergain/singers.git>

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# singers" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/pvergain/singers.git
git push -u origin master
```

...or push an existing repository from the command line

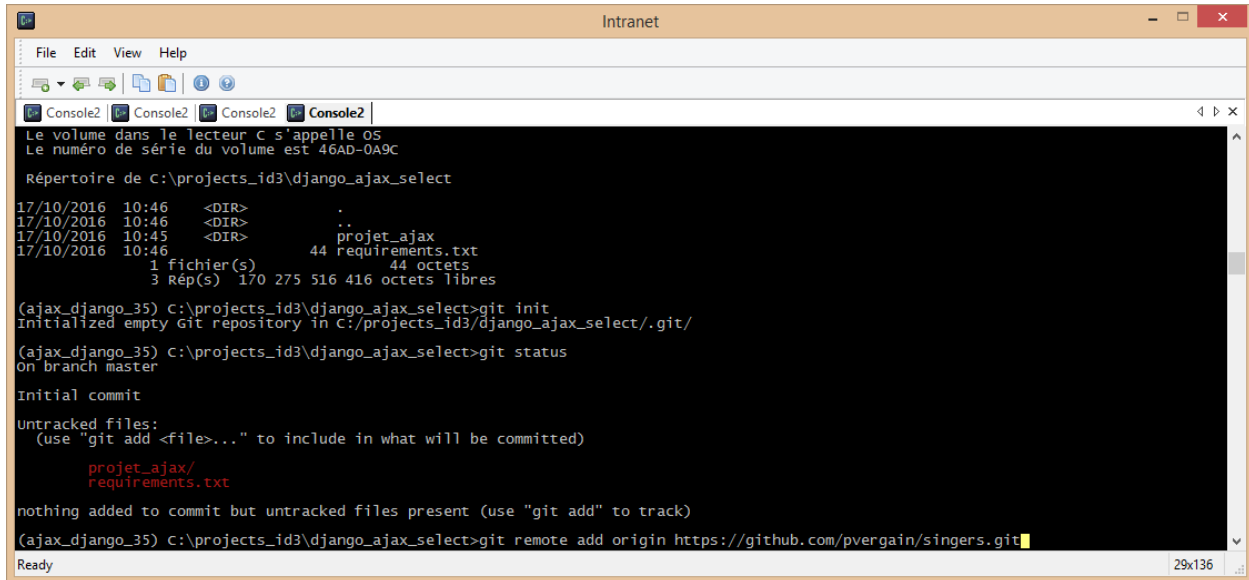
```
git remote add origin https://github.com/pvergain/singers.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

ProTip! Use the URL for this page when adding GitHub as a remote.



```

Le volume dans le lecteur C s'appelle OS
Le numéro de série du volume est 46AD-0A9C

Répertoire de C:\projects_id3\django_ajax_select
17/10/2016 10:46 <DIR>      .
17/10/2016 10:46 <DIR>      ..
17/10/2016 10:45 <DIR>      projet_ajax
17/10/2016 10:46          44 requirements.txt
1 fichier(s) 44 octets
3 Rép(s) 170 275 516 416 octets libres

(ajax_django_35) C:\projects_id3\django_ajax_select>git init
Initialized empty Git repository in C:/projects_id3/django_ajax_select/.git/

(ajax_django_35) C:\projects_id3\django_ajax_select>git status
on branch master

Initial commit
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    projet_ajax/
    requirements.txt

nothing added to commit but untracked files present (use "git add" to track)

(ajax_django_35) C:\projects_id3\django_ajax_select>git remote add origin https://github.com/pvergain/singers.git

```

Singers's Python virtualenv

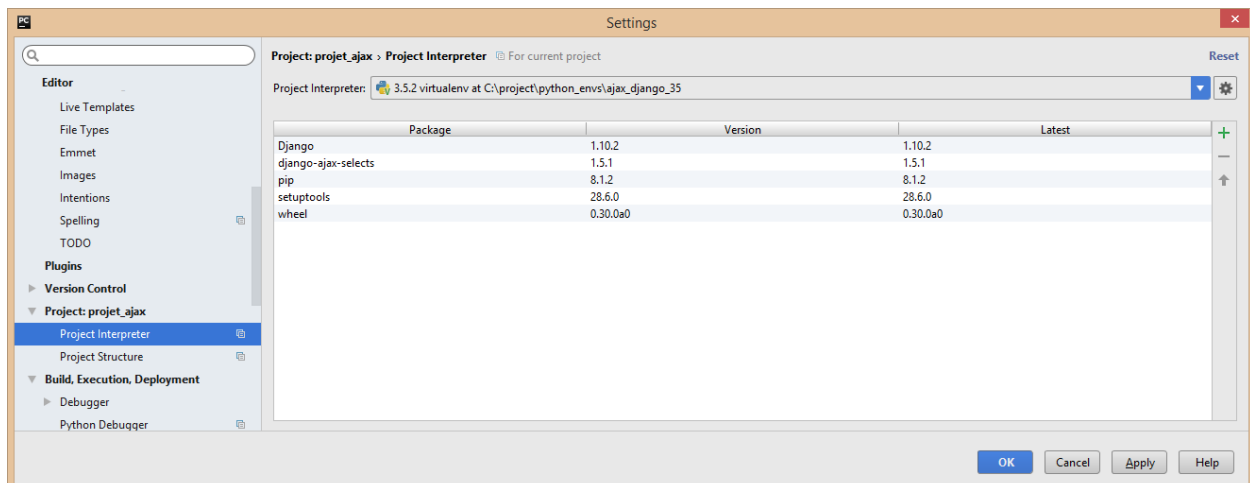
See also:

- <https://swcarpentry.github.io/git-novice/>

Contents

- *Singers's Python virtualenv*
 - *ajax_django35 Python 3.5 virtualenv*

ajax_django35 Python 3.5 virtualenv



Django

Django packages

Django tips

See also:

- <https://gist.github.com/kyle-eshares/0c30d81c4f8f201a41b916a2dd864b2c#file-values-py>
- <https://medium.com/@hansonkd/performance-problems-in-the-django-orm-1f62b3d04785#.qoy9oemyt>

```
>>> Book.objects.values('title', 'author__name')
<QuerySet [{'author__name': u'Nikolai Gogol', 'title': u'The Overcoat'}, {'author__
↪name': u'Leo Tolstoy', 'title': u'War and Peace'}]>

# Retrieve values as a tuple
>>> Book.objects.values_list('title', 'author__name')
<QuerySet [(u'The Overcoat', u'Nikolai Gogol'),
(u'War and Peace', u'Leo Tolstoy')]>
>>> Book.objects.values_list('title')
<QuerySet [(u'The Overcoat',), (u'War and Peace',)]>

# With one value, it is easier to flatten the list
>>> Book.objects.values_list('title', flat=True)
<QuerySet [u'The Overcoat', u'War and Peace']>
```

Solving Performance Problems in the Django ORM

Django is a wonderful tool which has helped thousands boost their productivity when writing web applications. Like any framework, when you first start out and the data model is simple, things are speedy. When you start adding real-world constraints and the data-model grows in complexity you'll probably find that the same strategies you used in the beginning won't be as effective. As you learn more about your problem domain, you need to adapt your code accordingly.

All frameworks require upfront knowledge about how the internals work in order to write high performance code. Django is fast, but sometimes it allows you to unwittingly write slow code.

What to look for

With something as complex like a web application, it is hard to know where to start.

A bottom-up technique to working with the data, starting from where the data lives to how it is displayed, gives a pragmatic approach to debugging performance problems.

The vast majority of performance problems in a web application are related to accessing the database. Unless you are dealing with large amounts of data and know what you are doing, don't approach the problem thinking about what the Big-O notation is for your View. The overhead of a database call will dwarf the overhead of loops and template rendering. Without first addressing how you are using the database, you can't move on to fixing other problems.

Django has a detailed summary of how to optimize working with the database. The ORM is large and strategies are needed to build efficient code from the beginning.

When approaching optimization, code can often become unclear. If faced with a choice between a small performance gain or clear code, understandable code should always come first. It takes practice to know where to place the threshold.

The first step to fixing a problem is being able to identify it. When dealing with the ORM, there are a few things you can do. Understand `django.db.connection`, which records the queries made with the current connection.

This can be cumbersome and as you make more and more queries, it can be hard to digest the information.

In the shell, use `django-extension`'s `shell_plus` command with the `--print-sql` flag turned on.

With the server, a middleware should run in the background of your `DEBUG` environment and log queries and point out duplicates.

`Django-debug-toolbar` provides this information in the page itself.

For our examples, we will use a classic author/book schema. Unexpected Queries

When checking the existence of an author for a book or grabbing the id of the author, it is tempting to want to use the author field directly.

It's tricky, but if the author object isn't needed, you potentially made an extra query for nothing. If you use the author value later, it doesn't have an impact. A good habit is to play it safe and always use the column name attribute. Size and Existence

Knowing when to use `count` and `exists` is tough. Django caches querysets, so when you are using the data of the queryset, use the built-in python operations. When not using the data, use the queryset methods.

The same holds true when you need the size of the queryset. If you are using the queryset, use `len`. If you only need the size use `count`.

Getting only what you need

By default, Django requests all the managed columns of the table and populates a Python object. When you only need a subset of columns from the table, consider using `values` and `values_list`. These methods skip the step of creating a complex python object and instead use dicts, tuples or even plain values. They can even handle getting columns through relationships straight forward.

Handling many rows

When you evaluate a queryset, Django caches the values. This makes sense if you will iterate over the queryset multiple times, but it doesn't make sense in an instance where you loop through once.

What Django actually does is load all books into memory and then iterate through each one. We want Django to hold the SQL connection open and read each row and call `do_stuff` before reading the next row. iterator to the rescue!

An added benefit of the iterator, is that it allows you to write linear data like table or a CSV as a stream. You can write a file or serve content to a user incrementally.

This is especially powerful when combined with `values` and `values_list`, because it keeps the minimum amount of information in memory as possible.

Using iterator also comes in handy during migrations where you need to mutate every row in a table. Just because the migration isn't client facing doesn't mean you should slack when it comes to efficiency. A long running migration could mean transaction locks or downtime.

Relationship Problems

Django's ORM allows you to interact with a relational database in a way that feels natural for the Object-Oriented Python programming language.

The code is precise code and semantically clear. Django uses lazy loading to only load the author if you need it. This is great, but can lead to an explosion of queries.

Django recognizes the problem and provides `select_related` and `prefetch_related` to solve it.

Using `prefetch_related` and `select_related` is critical when writing a Django app.

A caveat to `prefetch_related` is that if you plan on filtering the related queryset (`author.books.filter(...)`) the cache populated by `prefetch_related` won't be used and you will have to use a `Prefetch` object. Sometimes things can get complex and you might better off making 2 queries, one for the parents and one for the children, and then grouping

the children by the parent. If your prefetch plans get too complicated, evaluate how much you value the speed boost compared to making slightly less efficient but clearer code.

When `select_related` doesn't help

It is tempting to throw a `select_related` on everything, but there are certain circumstances that don't make sense. See the following result of the query. `id` in python gives you the unique id of an object instance. Objects with the same id value are the same instance.

You can be doing more work than needed. `select_related` creates a new instance for each row of the query, consuming memory. SQL also returns duplicate information for each row. If you are making a query where all the values of your `select_related` are the same, you'll want to use something else. Use related querysets or flip the query and use `prefetch_related`.

With the related queryset `author.books.all()`, Django caches the value of the author for each book using the same instance of the already queried value.

A hidden side affect is that if you use `select_related` and alter an author instance, that change won't propagate to the other authors in the queryset (even if they represent the same row) because they are different instances in python memory. With related querysets, the changes will propagate. Easier doesn't always mean better

Django makes following relationships too easy. This results in functions that cannot manage their own side-effects. When you pass in a model instance to a function and use a relationship, it is practically impossible to know if the relationship has already been fetched.

Will either `author_name_length` or `process_author_books` make a query? We can't tell. The relationship features of the ORM are so enticing that it's natural that we would want to write code this way. Using these functions without a `select_related` or `prefetch_related` in a loop can accidentally result in hundreds of queries. Django will happily make the queries without saying a peep. It is up to you to monitor your SQL logs and the callers of the function to figure out if it should pre-fetch or not.

We can rewrite our functions to be explicit by passing in a flattened data-structure that isn't a model (like a `namedtuple`), but we shouldn't have to think about it.

How do we fix it?

Knowing that we have this problem, how can we extend Django to be more explicit about resource consumption? Many database wrappers have solved this in different ways. In Ecto, the DB wrapper for Elixir, an unfetched relationship returns a `Ecto.Association.NotLoaded` struct instead of implicitly making the query.

Lets imagine a version of Django that implemented this behavior in a pythonic way.

An implementation can be relatively minimal.

There isn't a one size fits all answer to utilizing the ORM. Most of the time the performance gains for small apps won't make much of a difference. You should first seek to make your code clear and then work on optimizing it.

As your app grows, it is important to practice good hygiene when working with the ORM. Developing good habits now regarding consumption of resources will lead to big benefits later.

Optimization is a lot to handle, but a few simple rules can go a long way.

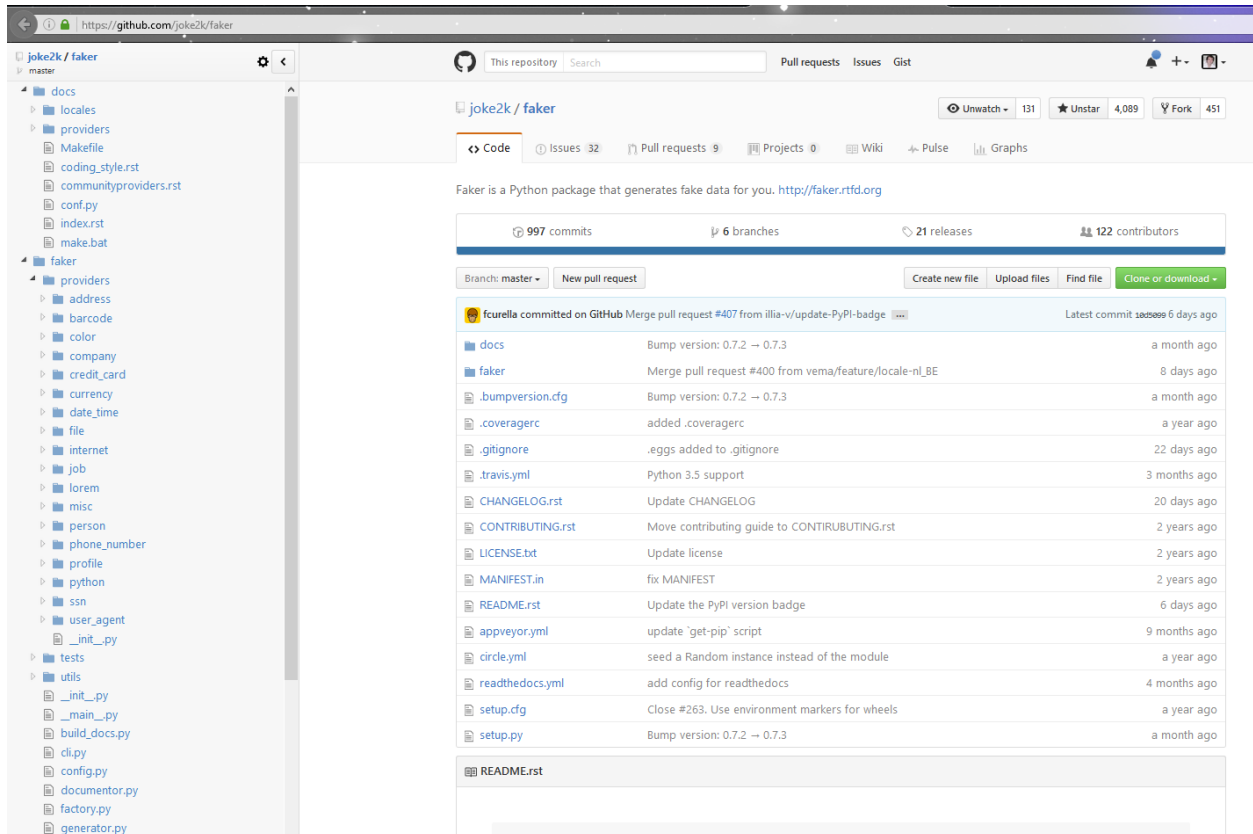
- Make a habit of isolating code and recording the queries it produces
- Queries should not be in loops
- Understand how the ORM caches data
- Know when Django will make a query
- Don't over-optimize at the expense of clarity

Python modules

Python faker module

See also:

- <https://github.com/joke2k/faker>
- <https://faker.readthedocs.io/en/master/>
- <http://blog.districtdatalabs.com/a-practical-guide-to-anonymizing-datasets-with-python-faker>



Description

Faker is a Python package that generates fake data for you. Whether you need to bootstrap your database, create good-looking XML documents, fill-in your persistence to stress test it, or anonymize data taken from a production service, Faker is for you.

install

```
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>pip install_
↵ faker
```

```
Collecting faker
  Downloading Faker-0.7.3-py2.py3-none-any.whl (529kB)
```

```
100% |#####| 532kB 1.2MB/s
Requirement already satisfied (use --upgrade to upgrade): six in c:\project\python_
↪envs\django_test_autocomplete_35_64\lib\site-packages (from faker)
Requirement already satisfied (use --upgrade to upgrade): python-dateutil>=2.4 in_
↪c:\project\python_envs\django_test_autocomplete_35_64\lib\site-packages (from faker)
Installing collected packages: faker
Successfully installed faker-0.7.3
```

Test

See also:

- <https://docs.djangoproject.com/en/dev/ref/contrib/auth/>
- <https://faker.readthedocs.io/en/master/providers/faker.providers.internet.html>
- <https://faker.readthedocs.io/en/master/providers/faker.providers.person.html>

Les attributs principaux de l'utilisateur par défaut sont :

- username
- password
- email
- first_name
- last_name

```
(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>faker_
↪address
```

```
042 Wilson Isle
Port Alan, MT 63548-2890
```

```
from faker import Factory

for lang in ['dk_DK', 'en_GB', 'fa_IR', 'it_IT', 'es_ES']:
    fake=Factory.create(lang)
    for _ in range(20):
        first_name = fake.first_name_female()
        last_name = fake.last_name()
        username = '{}_{}'.format(first_name, last_name)
        email = '{}_{}@{}'.format(first_name,
                                   last_name,
                                   fake.free_email_domain())
        female=User.objects.get_or_create(username=username,
                                           first_name=first_name,
                                           last_name=last_name,
                                           email=email,
                                           password='1234')
```

```
In [18]: User.objects.count()
Out[18]: 11
```

```
In [19]: fake=Factory.create('fr_FR')
```

```

...:     for _ in range(20):
...:         first_name = fake.first_name_female()
...:         last_name = fake.last_name()
...:         username = '{}_{}'.format(first_name, last_name)
...:         email = '{}_{}@{}'.format(first_name,
...:                                 last_name,
...:                                 fake.free_email_domain())
...:         female=User.objects.get_or_create(username=username,
...:                                         first_name=first_name,
...:                                         last_name=last_name,
...:                                         email=email,
...:                                         password='1234')
...:

```

In [20]: User.objects.count()

Out[20]: 31

```

for lang in ['dk_DK', 'en_GB', 'fa_IR', 'it_IT', 'es_ES']:
    fake=Factory.create(lang)
    for _ in range(20):
        first_name = fake.first_name_male()
        last_name = fake.last_name()
        username = '{}_{}'.format(first_name, last_name)
        email = '{}_{}@{}'.format(first_name,
                                last_name,
                                fake.free_email_domain())
        female=User.objects.get_or_create(username=username,
                                        first_name=first_name,
                                        last_name=last_name,
                                        email=email,
                                        password='1234')

```

```

In [24]:     for lang in ['dk_DK', 'en_GB', 'fa_IR', 'it_IT', 'es_ES']:
...:         fake=Factory.create(lang)
...:         for _ in range(20):
...:             first_name = fake.first_name_female()
...:             last_name = fake.last_name()
...:             username = '{}_{}'.format(first_name, last_name)
...:             email = '{}_{}@{}'.format(first_name,
...:                                     last_name,
...:                                     fake.free_email_domain())
...:             female=User.objects.get_or_create(username=username,
...:                                             first_name=first_name,
...:                                             last_name=last_name,
...:                                             email=email,
...:                                             password='1234')
...:

```

In [25]: User.objects.count()

Out[25]: 151

```

In [26]:     for lang in ['dk_DK', 'en_GB', 'fa_IR', 'it_IT', 'es_ES']:
...:         fake=Factory.create(lang)
...:         for _ in range(20):
...:             first_name = fake.first_name_male()
...:             last_name = fake.last_name()

```

```
...:         username = '{}_{}'.format(first_name, last_name)
...:         email = '{}_{}@{}'.format(first_name,
...:                                 last_name,
...:                                 fake.free_email_domain())
...:         female=User.objects.get_or_create(username=username,
...:                                          first_name=first_name,
...:                                          last_name=last_name,
...:                                          email=email,
...:                                          password='1234')
...:
```

In [27]: User.objects.count()

Out[27]: 251

```
from faker import Factory
for lang in ['nl_NL', 'cs_CZ', 'el_GR', 'fi_FI', 'sv_SE']:
    fake=Factory.create(lang)
    for _ in range(20):
        first_name = fake.first_name_female()
        last_name = fake.last_name()
        username = '{}_{}'.format(first_name, last_name)
        email = '{}_{}@{}'.format(first_name,
                                last_name,
                                fake.free_email_domain())
        female=User.objects.get_or_create(username=username,
                                          first_name=first_name,
                                          last_name=last_name,
                                          email=email,
                                          password='1234')
```

2016-10-21 Receive 2 new excellent books about jQuery: jQuery In Action and jquery UI in Action

See also:

- <https://twitter.com/tjvantoll>
- <https://twitter.com/AurelioDeRosa>
- <http://developer.telerik.com/featured/is-jquery-still-relevant/>
- <https://trends.builtwith.com/javascript/jQuery>

Contents

- *2016-10-21 Receive 2 new excellent books about jQuery: jQuery In Action and jquery UI in Action*
 - *If you were starting a new web app tomorrow, would you use jQuery ?*
 - * Aurelio De Rosa
 - * TJ VanToll
 - *jQuery in Action (third edition)*

* *clone*

- *jQuery UI in Action*
- *jquery plugins*

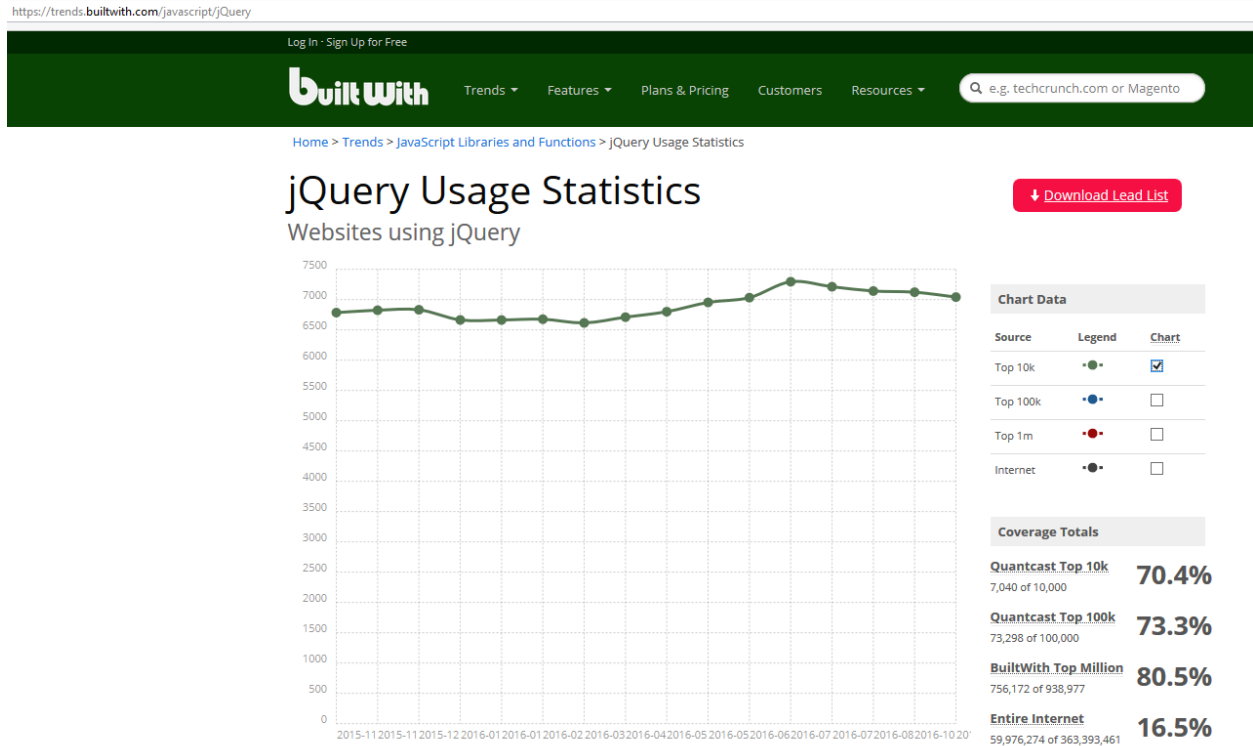


Fig. 4.1: <https://trends.builtwith.com/javascript/jquery>

If you were starting a new web app tomorrow, would you use jQuery ?

See also:

- <https://twitter.com/tjvantoll>
- <https://twitter.com/AurelioDeRosa>
- <http://developer.telerik.com/featured/is-jquery-still-relevant/>

Aurelio De Rosa

I don't use it by default anymore, but it also depends on the project and the browsers I have to support. These days I don't have to support IE8 anymore, but I'm still maintaining some code bases developed a couple of years ago that needed to support IE8. So, I still see jQuery around **from time** to time.

I think we can **all** agree that jQuery **is not** the tool **for** building SPAs, but to be honest, it was never conceived **for** that.

TJ VanToll

Exactly. I still find jQuery to provide the exact mix of functionality I need **for** little sites like that. I would probably **not**, however, use jQuery **if** I were starting on a large **complex** app nowadays. I think modern frameworks like Angular 2 just provide better paradigms **for** building apps at that scale.

jQuery in Action (third edition)

See also:

- <https://github.com/AurelioDeRosa/jquery-in-action>
- <https://twitter.com/AurelioDeRosa>
- <https://www.manning.com/books/jquery-in-action-third-edition>

clone

```
git clone https://github.com/AurelioDeRosa/jquery-in-action
```

jQuery UI in Action

See also:

- <https://github.com/tjvantoll/jquery-ui-in-action-demos>
- <https://twitter.com/tjvantoll>
- <https://twitter.com/jenlooper>
- <https://www.manning.com/books/jquery-ui-in-action>

jquery plugins

jquery plugins

jquery_ui_autocomplete_scroll plugin

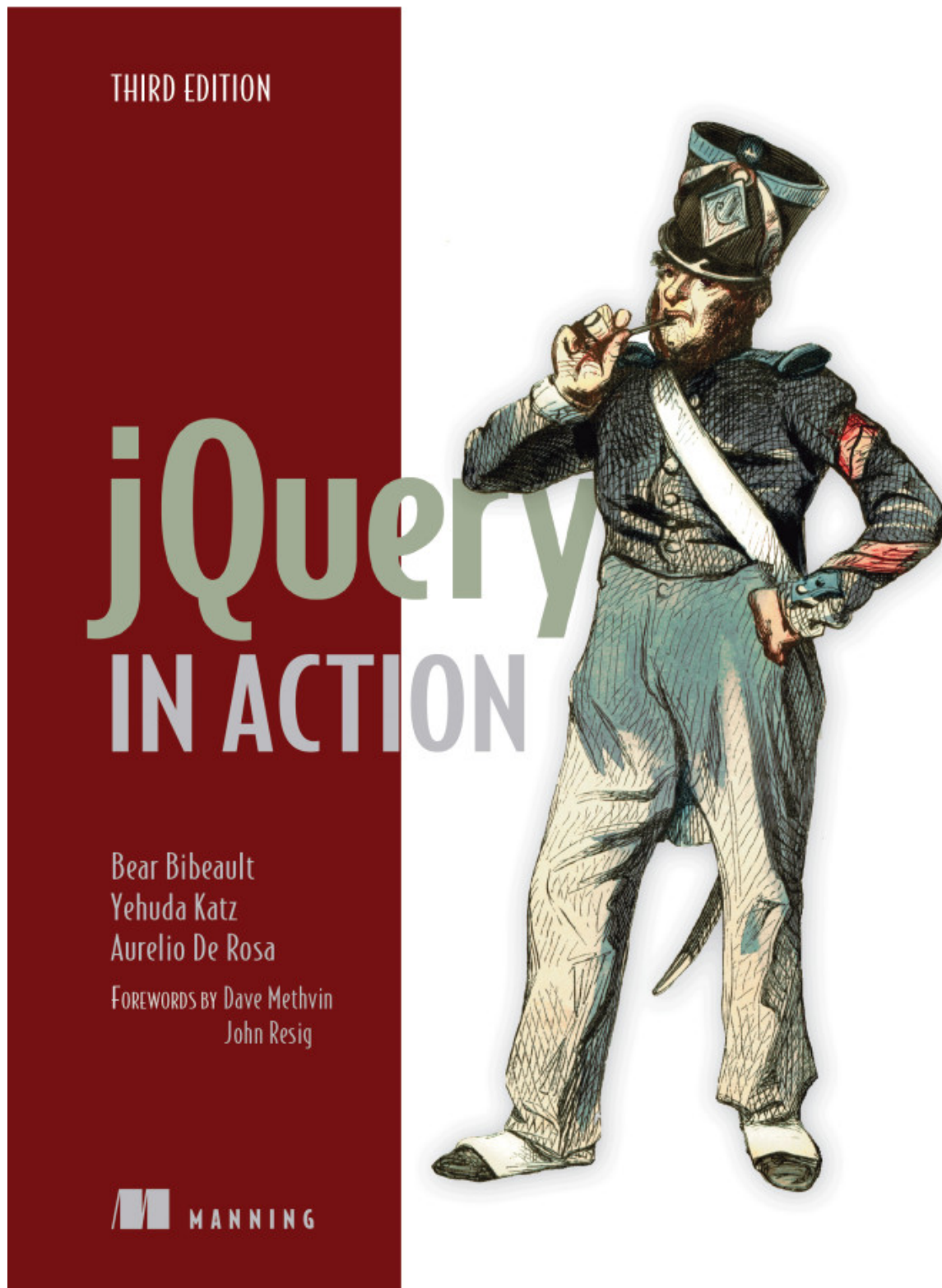
See also:

- <https://github.com/anseki>
- <https://anseki.github.io/jquery-ui-autocomplete-scroll/>
- <https://github.com/anseki/jquery-ui-autocomplete-scroll>

jquery select2 plugin

See also:

- <https://github.com/select2/select2>



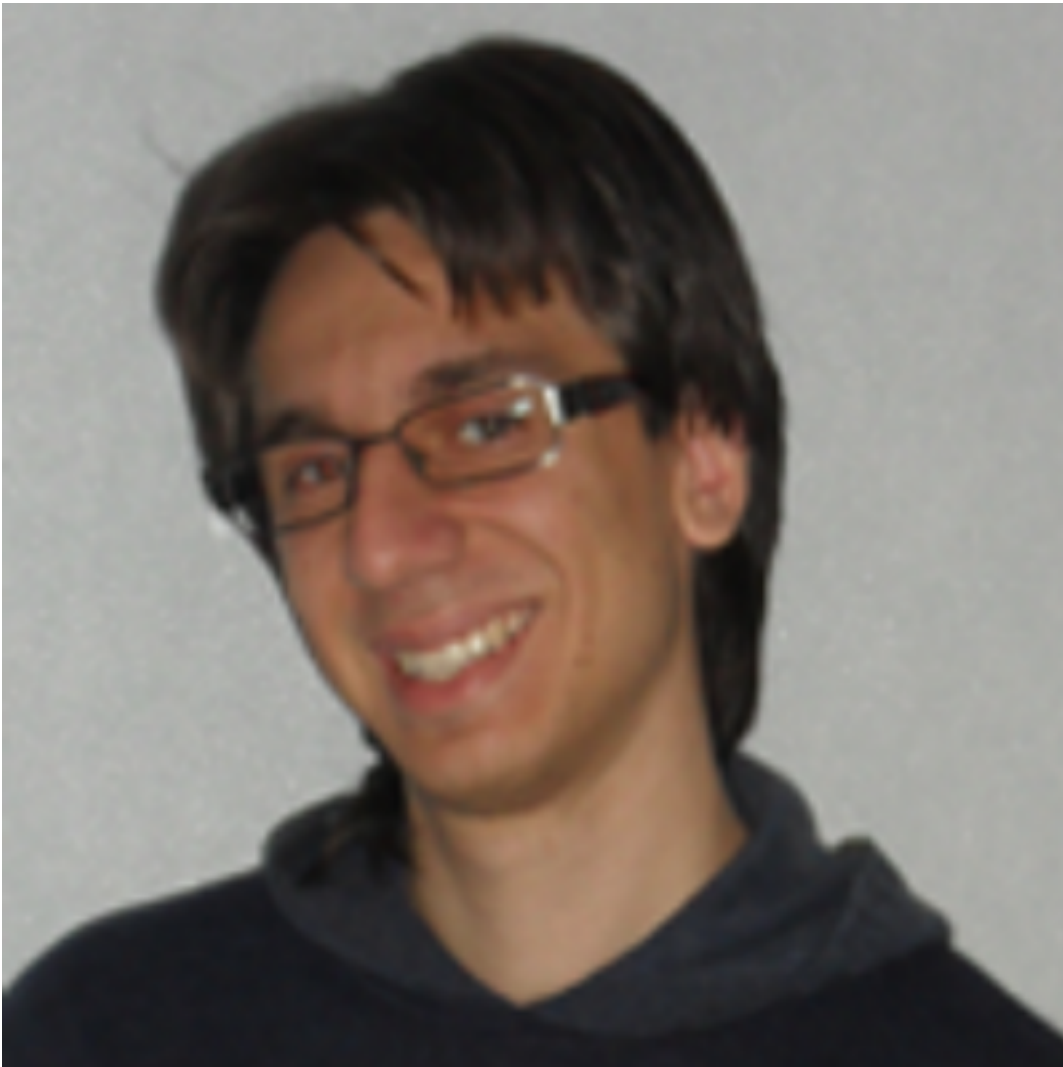
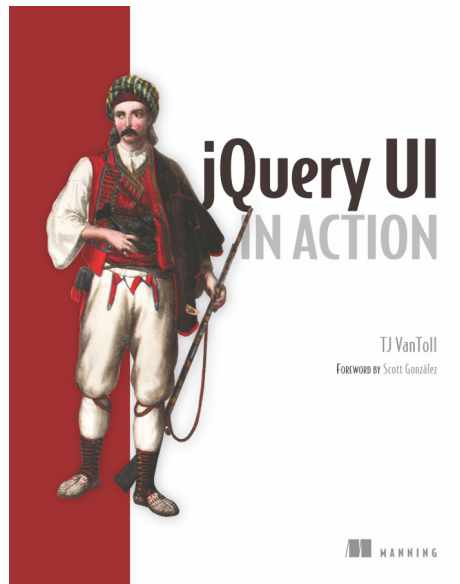


Fig. 4.2: Aurelio De Rosa (<https://twitter.com/AurelioDeRosa>)



- <https://select2.github.io/announcements-4.0.html>

Contents

- *jquery select2 plugin*
 - *Use cases*

Use cases

- Enhancing native selects with search.
- Enhancing native selects with a better multi-select interface.
- Loading data from JavaScript: easily load items via AJAX and have them searchable.
- Nesting optgroups: native selects only support one level of nesting. Select2 does not have this restriction.
- Tagging: ability to add new items on the fly.
- Working with large, remote datasets: ability to partially load a dataset based on the search term.
- Paging of large datasets: easy support for loading more pages when the results are scrolled to the end.
- Templating: support for custom rendering of results and selections.

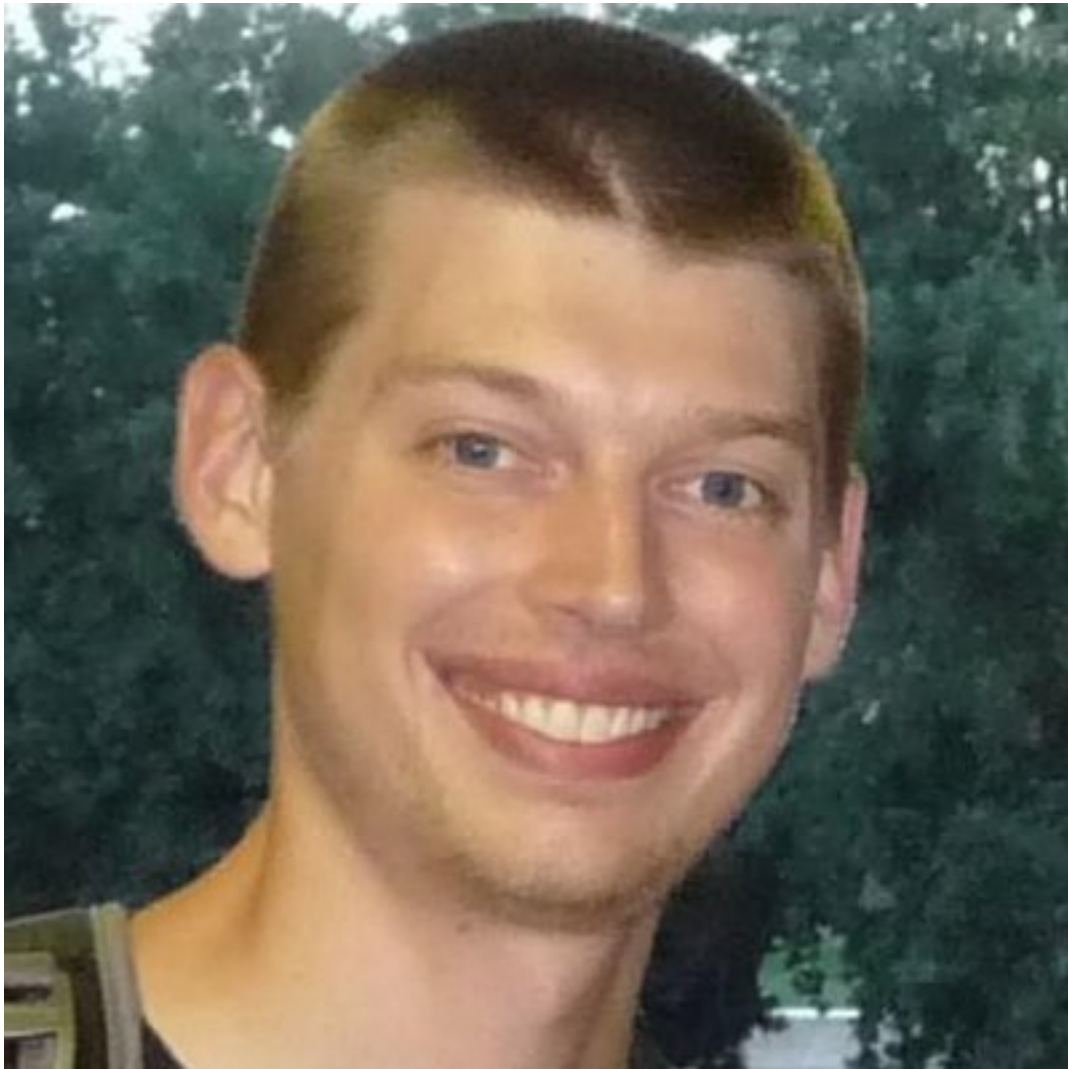


Fig. 4.3: TJ Vantoll (<https://twitter.com/tjvantoll>)

The screenshot shows the GitHub repository page for 'ansek / jquery-ui-autocomplete-scroll'. The repository has 23 commits, 2 branches, 8 releases, and 1 contributor. The latest commit is 'ansek Fix gh-pages URL in README' from 7 months ago. The repository contains files like LICENSE-MIT, README.md, bower.json, jquery.ui.autocomplete.scroll.js, jquery.ui.autocomplete.scroll.min.js, package.json, and ui.autocomplete.scroll.jquery.json. The README.md file is open, showing the title 'Scrollable jQuery UI Autocomplete' and a description: 'Scrollable jQuery UI Autocomplete Plugin for long items list. CSS max-height, overflow-y and padding-right can make scrollable list easy. But that is not beautiful. This plugin calculate with items and height of it. See DEMO'. The Usage section lists two points: 'Load the jquery.ui.autocomplete.scroll.min.js script file after loading jquery.ui.autocomplete.js etc.' and 'The maxShowItems is added to Autocomplete options. It accept the number of items which is max height of items list.'

The screenshot shows the GitHub repository page for 'select2 / select2'. The repository has a file tree on the left with folders like dist, docs, src, tests, and vendor. The main content area shows the README.md file, which includes sections for Integrations, Plugins, Themes, Internationalization (i18n), and Documentation. The Integrations section mentions third-party developers creating plugins for platforms like Django, Meteor, Ruby on Rails, Wicket, and Yii 2. The Plugins section lists several plugins: django-autocomplete-light, django-easy-select2, django-select2, Meteor - meteor-select2, Ruby on Rails - select2-rails, Wicket - wicketstuff-select2, and Yii 2 - yii2-widget-select2. The Themes section lists Bootstrap 3 - select2-bootstrap-theme, Flat UI - select2-flat-theme, and Metro UI - select2-metro. The Internationalization (i18n) section mentions that Select2 supports multiple languages by including the right language JS file. The Documentation section mentions that the documentation for Select2 is available through GitHub Pages and is located within this repository in the docs folder.

See also:

- <http://keepachangelog.com>
- <http://semver.org/>
- <https://github.com/pvergain/django-test-autocomplete.git>

Creating tags from the command line

To create a tag on your current branch, run this:

```
git tag <tagname>
```

This will create a local tag with the current state of the branch you are on. When pushing to your remote repo, tags are NOT included by default. You will need to explicitly say that you want to push your tags to your remote repo:

```
git push origin --tags
```

Versions

Version 0.2.0 (2016-10-27)

See also:

- <http://keepachangelog.com>
- <https://github.com/pvergain/django-test-autocomplete>

```
git tag 0.2.0  
git push origin --tags
```

in doc_sphinx/conf.py:

```
# The short X.Y version.
version = '0.2.0'
# The full version, including alpha/beta/rc tags.
release = '0.2.0'
```

Added

- the documentation is on read the docs (<https://django-test-autocomplete.readthedocs.io/en/latest/>)

Changed

The django-autocomplete-light is adopted.

Miscellaneous

Fixed

Test

Version 0.1.0 (2016-10-21)

See also:

- <http://keepachangelog.com>
- <https://github.com/pvergain/django-test-autocomplete/tree/0.1.0>

```
git tag 0.1.0

(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>git tag 0.1.
↪0

(django_test_autocomplete_35_64) C:\projects_id3\django-test-autocomplete>git push_
↪origin --tags
Fatal: ArgumentException encountered.
"C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-winstore.
↪exe" get: C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-
↪winstore.exe: No such file or directory
Username for 'https://github.com': pvergain
Password for 'https://pvergain@github.com':
Fatal: ArgumentException encountered.
"C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-winstore.
↪exe" store: C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-
↪credential-winstore.exe: No such file or directory
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/pvergain/django-test-autocomplete.git
 * [new tag]          0.1.0 -> 0.1.0
```

Added

See also:

EasyAutocomplete examples

Input simple

Input data-json

Input AJAX

aaaa
admin
albert
champion_0001
john
nigel

on/?term="

Django version 1.10.2, using settings 'projet_ajax.settings'
Starting development server at http://127.0.0.1:8004/
quit the server with CTRL-BREAK.

```
[21/oct/2016 09:43:29] "GET /projects/project/1/update/ HTTP/1.1" 200 11489
[21/oct/2016 09:43:29] "GET /static/easyautocomplete/css/easy-autocomplete.css HTTP/1.1" 200 9277
[21/oct/2016 09:43:29] "GET /static/easyautocomplete/css/easy-autocomplete.themes.css HTTP/1.1" 200 5989
[21/oct/2016 09:43:29] "GET /static/easyautocomplete/js/jquery.easy-autocomplete.js HTTP/1.1" 200 34623
[21/oct/2016 09:43:29] "GET /static/debug_toolbar/css/print.css HTTP/1.1" 200 29
[21/oct/2016 09:43:29] "GET /static/debug_toolbar/css/toolbar.css HTTP/1.1" 200 20293
[21/oct/2016 09:43:29] "GET /static/debug_toolbar/js/jquery_pre.js HTTP/1.1" 200 136
[21/oct/2016 09:43:29] "GET /static/debug_toolbar/js/toolbar.js HTTP/1.1" 200 12552
[21/oct/2016 09:43:29] "GET /static/debug_toolbar/js/jquery_post.js HTTP/1.1" 200 118
[21/oct/2016 09:43:30] "GET /static/debug_toolbar/img/ajax-loader.gif HTTP/1.1" 200 404
[21/oct/2016 09:43:37] "GET /projects/champion_get_json/?term=a HTTP/1.1" 200 342
[21/oct/2016 09:43:49] "GET /projects/champion_get_json/?term=a HTTP/1.1" 200 342
[21/oct/2016 09:44:18] "GET /projects/champion_get_json/?term=z HTTP/1.1" 200 45
[21/oct/2016 09:44:29] "GET /projects/champion_get_json/?term=a HTTP/1.1" 200 342
```

- *Adding the AJAX call in the Django template*

This is the First version !

The best autocomplete jquery plugin for the non-admin Django is the jquery EasyAutocomplete plugin

Changed

Miscellaneous

Fixed

Test

AJAX Ajax (also AJAX; /edæks/; short for asynchronous JavaScript and XML) is a set of web development techniques using many web technologies on the client-side to create asynchronous Web applications. With Ajax, web applications can send data to and retrieve from a server asynchronously (in the background) without interfering with the display and behavior of the existing page. By decoupling the data interchange layer from the presentation layer, Ajax allows for web pages, and by extension web applications, to change content dynamically without the need to reload the entire page.

In practice, modern implementations commonly substitute JSON for XML due to the advantages of being native to JavaScript.

Source: [https://en.wikipedia.org/wiki/Ajax_\(programming\)](https://en.wikipedia.org/wiki/Ajax_(programming))

JSON JSON (canonically pronounced /desn/ JAY-sn;[1] sometimes JavaScript Object Notation) is an open-standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. It is the most common data format used for asynchronous browser/server communication, largely replacing XML which is used by AJAX.

JSON is a language-independent data format. It derives from JavaScript, but as of 2016, code to generate and parse JSON-format data is available in many programming languages. The official Internet media type for JSON is application/json. The JSON filename extension is .json.

Douglas Crockford originally specified the JSON format; two competing standards, RFC 7159 and ECMA-404, define it. The ECMA standard describes only the allowed syntax, whereas the RFC also provides some semantic and security considerations.



Source: <https://en.wikipedia.org/wiki/JSON>

Symbols

- 0.1.0 (2016-10-21)
 - Version, [210](#)
- 0.2.0 (2016-10-27)
 - Version, [209](#)
- 2016-10-17
 - Actions, [180](#), [186](#)
- 2016-10-26
 - Actions, [37](#)

Numbers

- 101
 - jQuery, [37](#)
- 2016
 - Actions, [35](#)

A

- Actions
 - 2016-10-17, [180](#), [186](#)
 - 2016-10-26, [37](#)
 - 2016, [35](#)
 - Project, [34](#)
- AJAX, [213](#)
- Aurelio De Rosa
 - jQuery, [200](#), [202](#)
- Autocomplete
 - definitions, [1](#)
- autocomplete
 - jQuery-ui, [154](#)
 - Tests, [6](#)

B

- base.html
 - Django, [145](#), [154](#), [173](#)

C

- categories
 - EasyAutocomplete, [46](#)
- CSS

EasyAutocomplete, [46](#)

D

- definitions
 - Autocomplete, [1](#)
- Development
 - Django, [193](#)
 - Django test AJAX autocomplete, [190](#)
- Django
 - base.html, [145](#), [154](#), [173](#)
 - Development, [193](#)
 - django-autocomplete-light, [8](#)
 - django-extensions, [173](#)
 - makemigrations, [186](#)
 - normal Django Views, [180](#)
 - Packages, [194](#)
 - Template, [142](#)
 - Tips, [194](#)
- Django AJAX autocomplete
 - Project, [1](#)
- Django test AJAX autocomplete
 - Development, [190](#)
- Django test Autocomplete
 - Versions, [205](#)
- django-autocomplete-light
 - Django, [8](#)
- django-extensions
 - Django, [173](#)
- doc
 - read the docs, [45](#)
- Documentation
 - Read the docs, [1](#)

E

- EasyAutocomplete
 - categories, [46](#)
 - CSS, [46](#)
 - flags, [46](#)
 - Javascript, [46](#)
 - jQuery, [46](#)

- JSON, 46
- SCSS, 46
- easyautocomplete
 - Tests, 23

F

- Faker
 - Python, 37, 196, 197
- flags
 - EasyAutocomplete, 46

G

- git
 - gitsome, 35
- Github
 - singers, 191
 - swcarpentry, 191
- github
 - <https://github.com/anseki>, 35, 37
- gitsome
 - git, 35
- Glossary
 - Project, 212

H

- HTML5
 - placeholder, 39
- httpie
 - Python, 46
- <https://github.com/anseki>
 - github, 35, 37

I

- inheritance
 - Template, 173

J

- Javascript
 - EasyAutocomplete, 46
 - Unobstrusive, 142
- jQuery, **200, 202**
 - 101, 37
 - Aurelio De Rosa, 200, 202
 - EasyAutocomplete, 46
 - jquery_ui_autocomplete_scroll, 202
 - select2, 202
 - TJ VanToll, 200, 202
 - UI, 154
- jQuery EasyAutocomplete
 - plugin, 46
- jQuery UI autocomplete
 - scrollbar, 29
 - Tests, 29

- jQuery-ui
 - autocomplete, 154
- jquery_ui_autocomplete_scroll
 - jQuery, 202
- JSON, **213**
 - EasyAutocomplete, 46

M

- makemigrations
 - Django, 186

N

- normal Django Views
 - Django, 180

P

- Packages
 - Django, 194
- placeholder
 - HTML5, 39
- plugin
 - jQuery EasyAutocomplete, 46
- Project
 - Actions, 34
 - Django AJAX autocomplete, 1
 - Glossary, 212
- Python
 - Faker, 37, 196, 197
 - httpie, 46

R

- Read the docs
 - Documentation, 1
- read the docs
 - doc, 45

S

- scrollbar
 - jQuery UI autocomplete, 29
- SCSS
 - EasyAutocomplete, 46
- select2
 - jQuery, 202
- singers
 - Github, 191
 - virtualenv, 191
- swcarpentry
 - Github, 191

T

- Template
 - Django, 142
 - inheritance, 173

Tests

- autocomplete, [6](#)
- easyautocomplete, [23](#)
- jQuery UI autocomplete, [29](#)

Tips

- Django, [194](#)

TJ VanToll

- jQuery, [200](#), [202](#)

U

UI

- jQuery, [154](#)

Unobstrusive

- Javascript, [142](#)

V

Version

- 0.1.0 (2016-10-21), [210](#)
- 0.2.0 (2016-10-27), [209](#)

Versions, [205](#)

- Django test Autocomplete, [205](#)

virtualenv

- singers, [191](#)