[A Quick Guide to Django i18n – The Phrase Blog – Software Localization Experts](https://phrase.com/blog/posts/quick-guide-django-i18n/)

python -m django –version

List of language codes in <https://github.com/django/django/blob/master/django/conf/global_settings.py>

https://docs.djangoproject.com/en/dev/topics/i18n/translation/#set-language-redirect-view

1. Create a Django project. Python manage.py startproject mysite
2. Create a new Django app. Pyhton manage.py startapp polls
3. In settings.py, make sure you have activated internalizations:

LANGUAGE\_CODE = 'en-us'

#TIME\_ZONE = 'UTC'

TIME\_ZONE = 'US/Pacific'

USE\_I18N = True

USE\_L10N = True

USE\_TZ = True

LANGUAGES = [

    ('en','English'),

    ('zh-hans', u'简体中文'),

    ('zh-hant', u'繁體中文'),

]

1. Adding the middleware 'django.middleware.locale.LocaleMiddleware' **in settings.py**

MIDDLEWARE = [

    'django.middleware.security.SecurityMiddleware',

    'whitenoise.middleware.WhiteNoiseMiddleware',    # http://whitenoise.evans.io/en/stable/

    'django.contrib.sessions.middleware.SessionMiddleware',

    'django.middleware.locale.LocaleMiddleware',    # added for i18n, 2020.12.27

    'django.middleware.common.CommonMiddleware',

    'django.middleware.csrf.CsrfViewMiddleware',

    'django.contrib.auth.middleware.AuthenticationMiddleware',

    'django.contrib.messages.middleware.MessageMiddleware',

    'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

1. Then you have to mark all strings that will be translated. note: after Django 3.1, {% trans “str” %} has been changed to {% translate “str” %}

<!-- polls/templates/polls/index.html -->

<h1>Welcome to our site!</h1>

<p>Here you find polls.</p>

This file needs to be adapted to look like below:

<!-- polls/templates/polls/index.html -->

{% load i18n %}

<h1>{% trans 'Welcome Heading' %}</h1>

<p>{% trans 'Welcome Message' %}</p>

so we import the localization package and replace all texts with trans 'Some Translation Key Name'. Alternatively, you can use the default translation text as the translation key. This way, you will always have a good default text, if there is no [translation](https://phrase.com/blog/localization-and-translation-glossary/translation-t9n/) available for the particular key.

1. I18n inside python code:

When you want to localize strings within your python code (for example inside polls/views.py), you need to import the gettext function or gettext\_lazy() . It is normal to alias it to \_. So a simple localized view function would look like this

# polls/views.py

from django.http import HttpResponse

from django.utils.translation import ugettext as \_

def index(request):

output = \_('StatusMsg')

return HttpResponse(output)

1. Create translation file:

Now we have to create the translation files for every locale, we want to support. To do this create the directory polls/locale and inside the directory polls run

$django-admin makemessage -l de

$django-admin makemessage -l zh\_Hans (for “zh-hans”)

$django-admin makemessage -l zh\_Hant (for “zh-hant”)

\*\*Ensure folder name are zh\_Hans and zh\_Hant in Linux\*\*

Or to support all languages in LANGUAGES in settings.py:

* Django-admin makemessages -a (porcess

where you can replace de with the locale code of the language you like to add. In our example, this command will create the gettext file polls/locale/de/LC\_MESSAGES/django.po with the content

# polls/locale/de/LC\_MESSAGES/django.po

...

#: templates/polls/index.html:3

msgid "WelcomeHeading"

msgstr ""

#: templates/polls/index.html:4

msgid "WelcomeMessage"

msgstr ""

and you can fill in the translations:

# polls/locale/de/LC\_MESSAGES/django.po

...

#: templates/polls/index.html:3

msgid "WelcomeHeading"

msgstr "Willkommen auf unserer Seite!"

#: templates/polls/index.html:4

msgid "WelcomeMessage"

msgstr "Hier findet Ihr Umfragen."

When you have finished translating, you have to compile everything by running

$ django-admin compilemessages again inside the directory polls.

To quickly check that your translations work, you have to change the language code inside mysite/settings.py like so

# mysite/settings.py

LANGUAGE\_CODE = 'de'

When you open the polls application inside the browser, it should now be translated to German.

### Switching language in templates using the set\_language redirect view[¶](https://docs.djangoproject.com/en/dev/topics/i18n/translation/#switching-language-in-templates)

<https://docs.djangoproject.com/en/dev/topics/i18n/translation/#set-language-redirect-view>

https://docs.djangoproject.com/en/dev/topics/i18n/translation/#switching-language-in-templates

{% **load** i18n %}

<**form** action="{% **url** 'set\_language' %}" method="post">

{% **csrf\_token** %}

<**input** name="next" type="hidden" value="{{ redirect\_to }}">

<**select** name="language">

{% **get\_current\_language** **as** LANGUAGE\_CODE %}

{% **get\_available\_languages** **as** LANGUAGES %}

{% **get\_language\_info\_list** for LANGUAGES **as** languages %}

{% **for** language **in** languages %}

<**option** value="{{ language.code }}"{% **if** language.code == LANGUAGE\_CODE %} selected{% **endif** %}>

{{ language.name\_local }} ({{ language.code }})

</**option**>

{% **endfor** %}

</**select**>

<**input** type="submit" value="Go">

</**form**>

<**p**>{% **translate** "Welcome to our page" %}</**p**>

{% **endlanguage** %}

While the first occurrence of “Welcome to our page” uses the current language, the second will always be in English.

1. As a convenience, Django comes with a view, [**django.views.i18n.set\_language()**](https://docs.djangoproject.com/en/dev/topics/i18n/translation/#django.views.i18n.set_language), that sets a user’s language preference and redirects to a given URL or, by default, back to the previous page.

Activate this view by adding the following line to your URLconf:

path('i18n/', include('django.conf.urls.i18n')),