[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

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-cn', u'简体中文'),

    ('zh-tw', u'繁體中文'),

]

1. Then you have to mark all strings which have to be translated.

<!-- 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 'SomeTranslationKeyName'. 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 ugettext function. 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

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.