e-cidadania Documentation

Release a

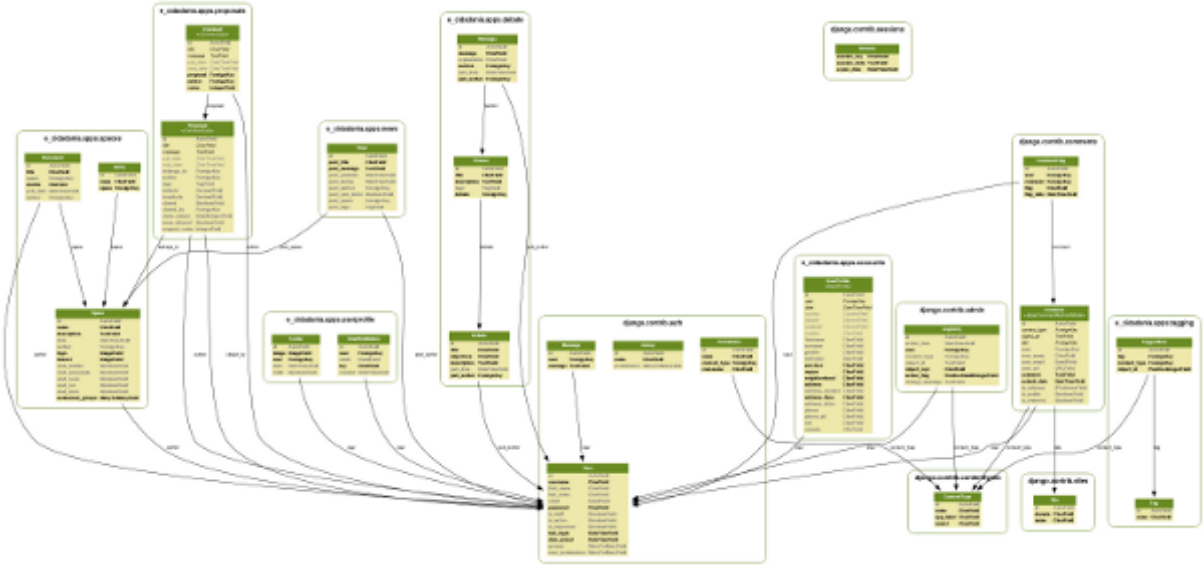
Cidadania S. Coop. Galega ewline Oscar Carballal Prego

26 de October de 2011

Índice general

e-cidadania es una plataforma web destinada a la participación ciudadana. Para ello cuenta con una serie de herramientas clave como son el sistema de propuestas, un revolucionario sistema de debate o un perfil avanzado de usuario con geolocalización y mensajería.

Este programa está basado en el framework django y en varias librerías externas.



Advertencia: e-cidadania está en una fase de desarrollo muy temprana y por ello algunas partes del programa así como los modelos de datos pueden sufrir grandes cambios hasta dentro de algunas versiones.

Documentación

CAPÍTULO 1

Instalación

La instalación de e-cidadania es muy sencilla y se cumple exactamente igual que cualquier otra plataforma Django.

1.1 Requisitos

- Apache, nginx, o cualquier otro servidor web que soporte CGI
- FastCGI, CGI, Passenger u otro intérprete CGI

Dependencias

- django 1.3
- PIL *(Python Imaging Library)*
- python-datetime *(versión 1.5)*
- django-tagging
- django-wysiwyg
- django-grappelli (para la administración)

Piedes instalar todas las dependiencias de forma automática mediante el siguiente comando:

```
# pip install -r requirements.txt
```

Advertencia: Esta sección no está terminada!

1.2 Descargar la plataforma

Hay diversas formas de descargar e-cidadania. La más sencilla de ellas es acceder a la página de Descargas de la web y bajarte la última versión estable, o la de desarrollo, ya empaquetadas y listas para usar.

1.2.1 Versión estable

Podrás encontrar la últiva versión estable en la página de descargas de ecidadania.org:

`http://ecidadania.org/downloads`

1.2.2 Versión de desarrollo

La versión de desarrollo está disponible desde diversos lugares:

Gitorious: *(repositorio oficial)*:

```
git clone git://gitorious.org/e-cidadania/mainline.git
```

GitHub *(repositorio secundario)*:

```
git clone git://github.com/oscarcp/e-cidadania.git
```

Repo.or.cz *(mirror oficial)*:

```
git clone git://repo.or.cz/e_cidadania.git
```

Y con eso ya tendriamos una copia del programa. Suponemos que si sabes manejar git, no necesitarás más ayuda, pero en cualquier caso puedes consultarnos en las listas de correo :-)

Configuración

La plataforma e-cidadania viene casi lista para funcionar, no hay que tocar nada salvo el fichero *settings.py* y la configuración específica de tu servidor web para servir el contendio estático.

2.1 settings.py

Configurar la base de datos:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': 'db/sqlite.db',
        'USER': '',
        'PASSWORD': '',
        'HOST': '',
        'PORT': '',
    }
}
```

Lo primero de todo será configurar la base de datos. Por defecto, e-cidadania viene configurado para utilizxar una base de datos local de SQLite3, que puede servirte para hacer pruebas de forma local si lo necesitas.

Un ejemplo de base de datos en un servidor compartido de DreamHost es este:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'ecidadania_database',
        'USER': 'databaseadmin',
        'PASSWORD': 'somepassword',
        'HOST': 'mysql.ecidadania.org',
        'PORT': '',
    }
}
```

Modo Debug

El modo debug viene activado por defecto y se recomienda encarecidamente desactivarlo para comenzar a utilizar e-cidadania en producción. Para ello hay que desactivarlo en el fichero *settings.py*:

```
DEBUG = False
```

Otras opciones de settings.py

***ACCOUNT_ACTIVATION_DAYS* (número)** Esta variable especifica cuándos días tiene el usuario para activar su cuenta desde que recibe el correo de confirmación.

***GOOGLE_MAPS_API_KEY* (hash)** Llave de la API de Google para poder utilizar la interfaz de mapas. Debes crearte una propia a pesar de que e-cidadania venga con una configurada, ya que sólo funcionará en el dominio que hayas especificado.

***EMAIL_HOST* (servidor)** Servidor de correo desde el cual se enviarán los correos a los usuarios.

DEFAULT_FROM_EMAIL Dirección por defecto desde la que se enviarán los correos si no se especifica otra.

Manual de usuario

3.1 Etiqueta

3.2 Cómo registrarse

El registro de usuarios se realiza de forma presencial en los lugares donde se anote para los procesos participativos. De momento el registro automático de usuarios no está permitido por motivos de seguridad.

3.3 Cómo enviar una propuesta

Enviar una propuesta es de lo más sencillo. Basta con acceder al espacio de participación correspondiente y hacer clic en “Añadir una propuesta” en la columna de la derecha.

3.4 Cómo participar en un debates

El sistema de debates es nuevo, así que procura prestar atención para no perderte, esto no es un foro de internet.

Se han trasladado diversos modelos de debate presencial hasta esta plataforma, y uno de ellos es el que te vamos a enseñar ahora. ¡Ya verás qué rápido te acostumbras!

3.5 Cómo consultar documentos

Basta con hacer clic en el documento que quieras leer para que comience a descargarse. Si el documento que quieres no figura en la columna de la derecha puedes darle al botón “Ver todos los documentos” y se cargará una página nueva que te mostrará todos los documentos que están almacenados en ese espacio.

3.6 Preguntas frecuentes

Si tienes alguna pregunta frecuente que no está aquí por favor dínoslo!

Manual de administración

Este es un pequeño manual introductorio que va a enseñarte a utilizar e-cidadania de forma correcta.

4.1 Registro de usuarios

El registro de usuarios en la versión *v0.1 alpha* se realiza de forma manual, debiado a que no hay ningún mecanismo seguro de autentificación, salvo el DNI-e que está poco extendido.

De todas formas, e-cidadania cuenta desde el principio con un sistema de registro automático de usuarios que el administrador podrá activar cuando lo crea conveniente quitando la marca de comentario (almohadilla).

apps/userprofile/urls/en.py:107:

```
# url(r'^register/$', register, name='signup'),
```

Si la plataforma está correctamente configurada el sistema de registro ya se debería encargar de todo.

4.2 Permisos

Los permisos en e-cidadania se heredan directamente del sistema de django. De esa manera tenemos permisos por usuario y por grupo. Para esta primera versión de e-cidadanía es suficiente, pero **sería extremadamente recomendable que no utilizases la aplicación si la seguridad es tu prioridad**.

e-cidadania 0.2 contará con un sistema de permisos por fila, mucho más detallados y seguros que los permisos actuales.

4.3 Grupos

Los grupos son una forma masiva de otorgar permisos a grupos de gente. En esta versión los grupos van a ser una forma de agrupar a la gente por espacios y otorgarles permisos en esos espacios determinados, salvo que por algún motivo se le otorgue un permiso diferente por alguna tarea que deba hacer.

4.4 Espacios

Los espacios son lugares donde se realizan procesos participativos.

4.5 Módulos

e-cidadania es una plataforma modular. Incluso sus características básicas (noticias, documentos, espacios) son meros módulos que pueden ser sustituídos en el momento que se preciso sin afectar a la estructura general de la aplicación.

4.5.1 Moderación

Las tareas de moderación de la plataforma son muy sencillas. Cada módulo consta de tres tareas básicas, que son: creación, edición y borrado.

Creación Dependiendo del grado de moderación que se te haya otorgado podrás agregar contenidos sencillos o más complejos. Los mayores niveles de moderación tienen un grado elevadísimo de detalle a la hora de agregar contenido.

Edición La tarea de edición es similar a la de creación, se presentará un formulario en base a las credenciales de las que disponga el moderador.

Borrado Por norma general en foros un moderador puede borrar las entradas de los usuarios. En e-cidadania ese no es el objetivo. Todo lo que diga la gente debe preservarse salvo que incurra en alguna falta grave. Sólo los moderadores de mayor nivel pueden borrar las entradas.

4.6 Errores frecuentes

Los errores más frecuentes son debidos al servidor o a una mala gestión del administrador en cuanto a los permisos y/o grupos.

4.7 Configuración avanzada

e-cidadania se ha diseñado para evitar tanto al usuario como al administrador el máximo trabajo posible, de forma que la configuración es muy sencilla.

Existen un número determinado de variables que hay que configurar, mientras que el resto se configurarán automáticamente.

Todas las variables de configuración se encuentran en el fichero settings.py

4.7.1 Usuarios

AUTH_PROFILE_MODULE El módulo que se va a utilizar para extender el modelo de usuario.

ACCOUNT_ACTIVATION_DAYS Número de días que se le permitirá al usuario para poder activar su cuenta tras registrarse.

LOGIN_REDIRECT_URL Dirección a la que será enviado el usuario tras un login correcto.

***GOOGLE_MAPS_API_KEY* (no disponible)** Llave para la API de Google Maps, esta se va a utilizar para el geoposicionamiento en diversas zonas de la plataforma.

4.7.2 Correo electrónico

EMAIL_HOST Servidor de correo desde el que se van a enviar los emails a los usuarios.

DEFAULT_FROM_EMAIL Dirección de correo bajo la que se enviarán los correos a los usuarios. Por defecto: *accounts@cidadania.coop*

4.7.3 Administración y BDD

ADMINS Lista de administradores del sitio. Esta parte es importante ya que cualquier error de la plataforma será enviado a las direcciones que se hayan escrito.

DATABASES Configuración de una o varias bases de datos de la plataforma. En un sistema concurrido recomendados dos bases de datos separadas en hosts que no sean el que almacena los datos.

4.7.4 Idioma

TIME_ZONE Zona horaria, si no sabes cuál es la tuya puedes consultarlo en aqui

LANGUAGE_CODE Código de idioma, habitualmente de dos letras (ES, EN, FR, etc.)

LANGUAGES Lista de idiomas admitidos en la plataforma. Si esta variable está vacía, se cargará con la lista por defecto de **todos** los idiomas soportados por Django.

Desarrollo

CAPÍTULO 5

Guía de estilo

La guía de estilo establece una serie de parámetros a seguir a la hora de programar código para e-cidadania. Estas normas son inquebrantables. La guía de estilo sigue bastante fielmente el PEP8, con algunas excepciones que vienen de la guía de estilo interna de Pocoo.

5.1 Python

Imports Todos los imports deben estar situados en la cabecera del fichero, por debajo de la cabecera de comentarios. Los imports de módulos de sistema o de python deben preceder a los demás, y los de las librerías externas deben preceder a los de la aplicación.

Ejemplo:

```
import os
import sys

from extlib import function

from myapp.module import function
```

Ancho de línea El código debe de ser de 80 columnas de ancho como máximo salvo en los casos de las plantillas.

Declacariones largas Si una línea de código no cabe en 80 columnas, intenta reducirla declarando variables previamente. Si aún así no se puede se deben dividir de las siguientes formas:

Parentesis:

```
website = models.URLField(_('Website'), verify_exists=True,
                          max_length=200, null=True, blank=True,
                          help_text=_('The URL will be checked'))
```

Declaraciones:

```
this_is_a_very_long(function_call, 'with many parameters') \
    .that_returns_an_object_with_an_attribute

MyModel.query.filter(MyModel.scalar > 120) \
             .order_by(MyModel.name.desc()) \
             .limit(10)
```

Listas, tuplas y diccionarios:

```
items = [
    'this is the first', 'set of items', 'with more items',
    'to come in this line', 'like this'
```

```
]

dict = {
    ('mobile': phone),
    ('car': key),
    ('another': thing),
}
```

Indentación La indentación debe ser de 4 espacios por nivel, sin excepciones. No se pueden utilizar tabulaciones para marcar los niveles de indentación.

Líneas en blanco Todas las funciones y clases deben estar separadas por dos líneas en blanco. El código dentro de una clase o método por una línea en blanco.

Ejemplo:

```
class ListDocs(ListView):

    """
    List all documents stored whithin a space.
    """
    paginate_by = 25
    context_object_name = 'document_list'

    def get_queryset(self):
        place = get_object_or_404(Space, url=self.kwargs['space_name'])
        objects = Document.objects.all().filter(space=place.id).order_by('pub_date')
        return objects

    def get_context_data(self, **kwargs):
        context = super(ListDocs, self).get_context_data(**kwargs)
        context['get_place'] = get_object_or_404(Space, url=self.kwargs['space_name'])
        return context


def whatever(args):

    """
    A comment.
    """
    this_is_something = 0
```

5.2 HTML

Columnas El código HTML no tiene límite de columnas, pero debe estar indentado de forma que se pueda localizar rápidamente cualquier elemento del documento. La disposición indentada en el desarrollo prevalece sobre el resultado renderizado de la aplicación.

Indentación El código X/HTML debe estar indentado con 2 espacios, sin excepción.

5.3 CSS

Indentación La indentación será de 4 espacios, siempre, igual que el código Python.

Ejemplo:

```
body {
    background: #FAFAFA;
    padding: 0;
    margin: 0;
```

```
    font-family: Verdana, "Lucida Sans", Arial;
    font-size: 1em;
    color: #000;
    cursor: default;
}
```

Colores Los colores siempre deberán estar escritos en su código hexadecimal. Se permiten las abreviaturas de tres dígitos.

Tamaños de letra Los tamaños de letra deben ser declarados siempre en **em's** y salvo una excepción muy casual no se deben declarar en píxels.

5.4 JavaScript

Estilo de código JavaScript.

Cuentas de usuario

El sistema de cuentas de usuario en e-cidadania está basado en el módulo *auth* de django, así como en django-registration y django-profile, creados por James Bennet.

6.1 Campos de datos

Las cuentas de usuario contienen los siguientes campos:

username ***(CharField, 200 caracteres)*** Este campo contiene el nombre de usuario. Es accesible como user.username

firstname ***(CharField, 50 caracteres)*** Este campo contiene el nombre *real* del usuario.

surname ***(CharField, 200 caracteres)*** Este campo contiene los apellidos *reales* del usuario.

gender ***(Choice)*** Lista de elecciones de género. Opciones válidas: F (Female) y M (Male)

birthdate ***(DateField)*** Fecha de nacimiento del usuario. Utilizada para calcular la edad.

province ***(CharField, 50 caracteres)*** Provincia de residencia del usuario.

municipality ***(CharField, 50 caracteres)*** Municipio o ciudad de residencia del usuario.

address ***(CharField)*** Dirección de residencia (calle) del usuario.

address_number ***(CharField, 3 caracteres)*** Número del edificio de residencia.

address_floor ***(CharField, 3 caracteres)*** Piso

address_letter ***(CharField, 2 caracteres)*** Letra del piso de residencia

phone ***(CharField, 9 caracteres)*** Teléfono de contacto del usuario.

phone_alt ***(CharField, 9 caracteres)*** Teléfono secundario de contacto

6.2 django-profile

django-userprofile se encarga de proveeder las vistas y funciones para extender el modelo de datos de usuario en django. Junto a un módulo creado para extender el modelo de datos todo va perfecto.

6.3 accounts

El módulo accounts es nuestro modelo extendido de usuario. En él se encuentran todos los campos extra de usuario que se necesitan y que serán incorporados de forma transparente a *django-userprofile*.

CAPÍTULO 7

Creando módulos

e-cidadania es extensible de forma sencilla a través de módulos, aunque la automatización de su instalación todavía no esté en funcionamiento.

7.1 Estructura

Un módulo es básicamente una aplicación de django que integraremos en e-cidadania. En principio abogamos por la estructura por defecto de django tanto en la distribución de los ficheros como en sus nombres.

Generando documentación

La documentación de e-cidadania se genera mediante Sphinx (1.0.7) en tres idiomas por defecto, que son:

- Inglés
- Español
- Gallego

8.1 Normas linguísticas

Estas son las normas linguisticas de la documentación de e-cidadania

8.2 Palabras confusas

Diccionario de palabras confusas

Traducciones

Para la traducción se pueden utilizar mayoritariamente dos herramientas:

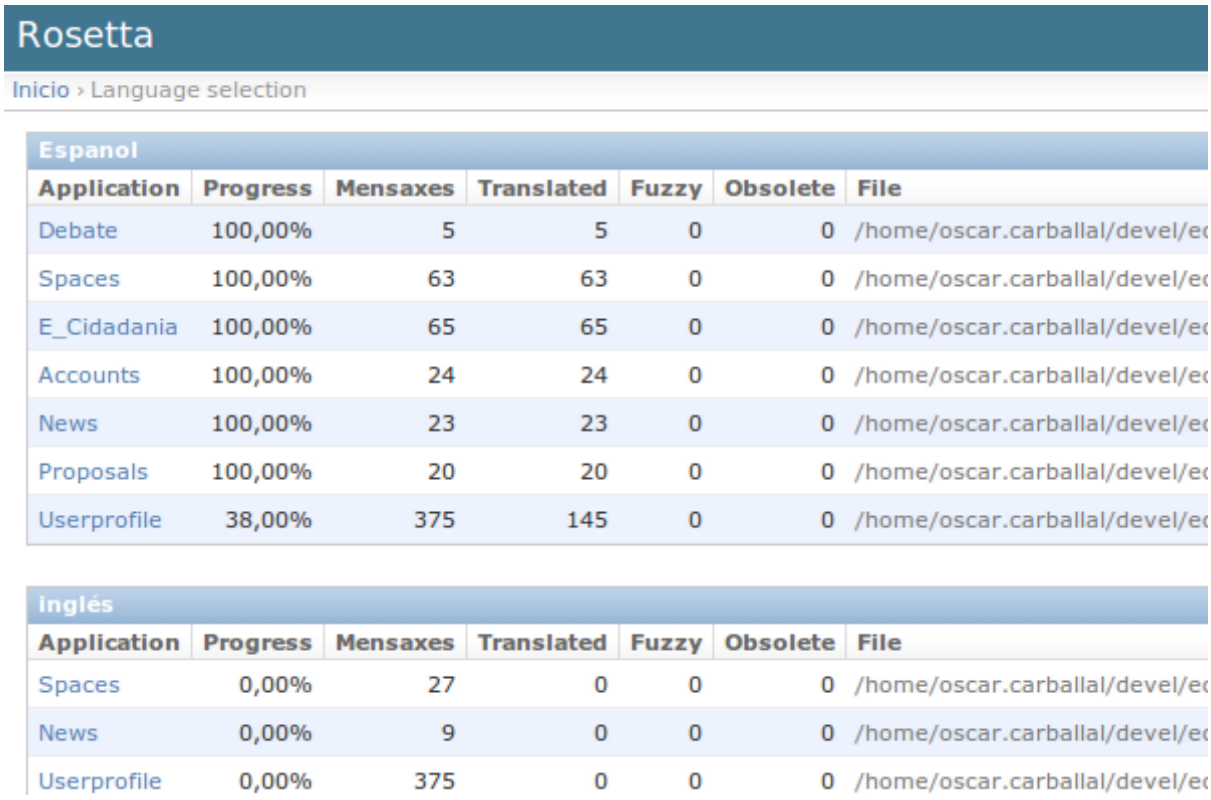
- django-rosetta
- gettext

Ambas formas de traducción son sencillas gracias al *middleware* de Django.

9.1 Traduciendo con rosetta

Para traducir con rosetta es necesario tener una cuenta en el sistema y pertenecer al grupo **'translators'**. Una vez hecho eso, el resto es sencillo.

Basta con acceder a la URL de traducción y lo primero que se verá será una lista de los idiomas disponibles para traducir.

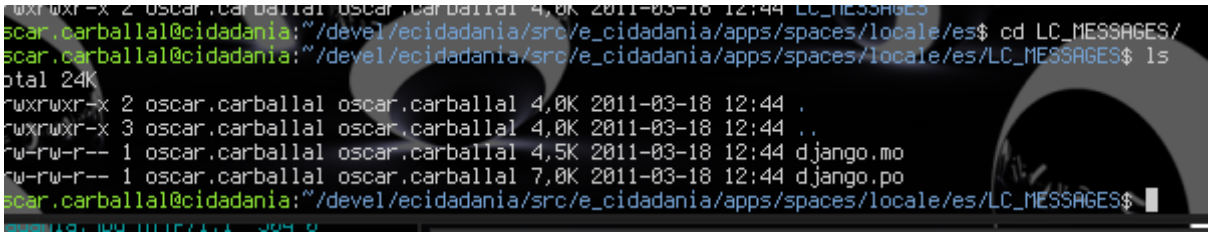


Basta con hacer clic en el componente que se desee traducir y comenzar la traducción (se realiza desde el inglés al resto de idiomas). Si te encuentras atascado puedes utilizar la opción “Sugerir” que consultará la base de datos de Google Translate y te dará el resultado que el crea correcto.

Advertencia: Nunca te fies del resultado del botón “sugerir” ya que en muchas ocasiones es incorrecto.

9.2 Traduciendo con gettext

Gettext es una herramienta de sobra conocida por todos los traductores del mundo. Es un estándar. Gracias al *middleware* de traducción que trae django de serie nuestro trabajao con gettext va a ser mínimo, tan sólo nos limitaremos a editar los ficheros .po del códgo fuente.



En vez de realizar una traducción global, hemos optado por diseñar una traducción esecífica para cada parte de la plataforma, de forma que las traducciones se perpetuen aunque los módulos se muevan.

La localización de las cadenas de texto habitualmente es en un directorio llamado **locale** dentro del módulo. Dentro del mismo, se encuentran directorios con el código de país (en, es, us, gl, fr, etc.) y dentro de éste, se encuentran los ficheros PO y MO.

Para traducir, debes editar el fichero PO, que es un fichero de texto plano ya hecho para ser posteriormente tratado.

El fichero MO es la traducción compilada a lenguaje máquina para que la plataforma pueda utilziarlo posteriormente.

Advertencia: Establecer un método de trabajo para traductores y explicarlo aqui.

Apariencias / Temas

Apariencias de e-cidadania

10.1 Otras apariencias

Este es un pequeño índice de apariencias de e-cidadania.

10.2 Crea tu propia apariencia para e-cidadania

Crear una nueva apariencia para e-cidadania puede llegar a ser algo difícil, pero no por ello imposible. Aquí te daremos las instrucciones precisas.

10.3 Referencia

Los temas de e-cidadania se localizan mayoritariamente en los propios módulos. Cada aplicación gestiona por separado sus apariencias, no en cuanto a estilo pero sí en cuanto a distribución.

Además de esto, existen algunas platillas generales que están situadas en el directorio *templates*.

10.3.1 General

asdasd

10.3.2 Perfil de usuario

asdasdas

10.3.3 Propuestas

sdfsdfs

10.3.4 Debates

sdfsdfsd

10.3.5 Noticias

sdfsdf

10.3.6 Documentos