Zinnia Drupal Documentation

Release 0.1.2

Branko Majic

September 16, 2013

CONTENTS

Zinnia Drupal is a utility application for importing Drupal blog into Django Blog Zinnia.

SUPPORT

In case of problems with the application, please do not hesitate to contact the author at **zinnia.drupal (at) majic.rs**. The application itself is hosted on Github:

- https://github.com/azaghal/zinnia-drupal

Contents:

1.1 About Zinnia Drupal

Zinnia Drupal is a small helper application for Django Blog Zinnia that provides a custom management command that can be used for importing content from Drupal into Zinnia.

1.1.1 Why was this application created?

This application was created as the means of migrating my current blog away from Drupal 6 into Django Blog Zinnia.

1.1.2 Features

Zinnia Drupal has the following features:

- Supports import from Drupal 6 database.

- Imports Drupal users. * All users or only a subset. * Change username during import.

- Imports Drupal vocabularies, converting them into Zinnia categories. * Support for both regular and tag-based vocabularies. * Hierarchy is preserved. * Vocabulary itself becomes a top-level category.

- Imports Drupal content as Zinnia blog entries.

- Imports Drupal comments. * Threaded comments supported (requires zinnia-thraded-comments).

- Import will skip existing data (re-runs should be safe).

There is a couple of limitations in the way the import is performed. Make sure that you are well aware of those limitations before starting the migration from Drupal:

- Only the MySQL database is supported at the moment. Despire this, it should be easy to extend the code other database since SQLAlchemy is used for accessing the Drupal database.

- Non-tag Drupal vocabularies must contain terms that have unique names. The import code will not distinguish between same-named terms coming from different vocabularies.

- Only username, password, and e-mail is imported for users. I.e. no other custom information from users will be imported.

- Users will not be granted any permissions in Zinnia beyond being marked as staff. I.e. the users won't be able to create new entries in Zinnia before being granted appropriate privileges.

- No formatting conversion is performed. Content is imported as-is.

- Revision history is not preserved (Django Blog Zinnia does not support revision history). Only the latest/current revision will be imported.

- Comment *titles* are not part of the import (Django Blog Zinnia does not support comment titles).

1.2 Installation

Zinnia Drupal can be installed through one of the following methods:

- Using *pip*, which is the easiest and recommended way for production websites.

1.2.1 Requirements

In order to use Zinnia Drupal it's necessary to install and configure Django Blog Zinnia itself. There's no special requirements for the import application, and standard Zinnia documentation should be followed.

Import of threaded comments is possible as well (preserving the hierarchy), but this requires Zinnia Threaded Comments. Otherwise the comments will be imported, but without information about threads.

1.2.2 Using pip

In order to install latest stable release of Zinnia Drupal using *pip*, run the following command:

```
pip install zinnia_drupal
```

In order to install the latest development version of Zinnia Drupal from Github, use the following command:

```
pip install -e git+https://github.com/azaghal/zinnia-drupal#egg=zinnia_drupal
```

> **Warning:** You will need to update the `pip` installation in your virtual environment if you get the following error while running the above command:
>
> ```
> AttributeError: 'NoneType' object has no attribute 'skip_requirements_regex'
> ```
>
> You can update `pip` to latest version with:
>
> ```
> pip install -U pip
> ```

After this you should proceed to *configure your Django installation*.

1.3 Configuring your Django installation

Once Zinnia Drupal has been installed, you need to perform the following steps in order to make it available inside of your Django Blog Zinnia project:

1. Edit your project's settings configuration file (`settings.py`), and update the `INSTALLED_APPS` to include application `zinnia_drupal`. Order does not matter.

After this you will have *additional management commands available* in your Django Blog Zinnia project that can be used for importing data from Drupal.

1.4 Usage

Zinnia Drupal provides custom management commands for importing blogs from Drupal database. The commands are:

- `drupal62zinnia`, for importing users, categories, content, and comments from Drupal 6 MySQL database.

1.4.1 `drupal62zinnia`

The command can be called via:

```
python manage.py drupal62zinnia [OPTIONS] database_name
```

The `database_name` should be the name of the Drupal database. By default this database is assumed to be hosted by a MySQL server on the same host where the command is run.

The following options are available (in addition to standard ones):

--database-hostname|-H DATABASE_HOSTNAME Hostname of database server providing the Drupal database. Default is `localhost`.

--database-port|-p DATABASE_PORT TCP port at which the database server is listening. Default is `3306`.

--database-username|-u DATABASE_USERNAME Username that should be used for connecting to database server. Default is `root`.

--database-password|-P DATABASE_PASSWORD_FILE Path to file containing the password for specified database username. If not set (default), the password will be read interactively.

--node-type|-n NODE_TYPE Drupal Node type that should be processed. Default is `blog`.

--user-mapping|-m USER_MAPPING Mapping of Drupal usernames to Zinnia usernames. Format is `duser1=zuser1:duser2=zuser2:...:dusern=zusern`. Default is to use same username as in Drupal.

--users|-U USERS Comma-separated list of Drupal users that should be imported, including user-created content. Default is to import content from all users.

--threaded-comments|-t Import comments while preserving threading information. Requires zinnia-threaded-comments application. Default is not to use threaded comments.

1.4.2 Examples

Import all blog entries for user `john`:

```
python manage.py drupal62zinnia -u dbuser -U john drupaldb
```

Import all blog entries for user `John Doe`, but map the user to user `john` in Zinnia:

```
python manage.py drupal62zinnia -u dbuser -U john -m 'John Doe=john' drupaldb
```

Import all blog entries for all users, but map user `John Doe` to user `john` in Zinnia:

```
python manage.py drupal62zinnia -u dbuser -m 'John Doe=john' drupaldb
```

Import all static pages for all users (treat them as blog entries), connecting to external database server listening on non-default port, and also use threaded comments:

```
python manage.py drupal62zinnia -H db.example.com -p 3307 -u dbuser -n page -t drupaldb
```

1.5 Release Notes

1.5.1 0.1

Initial relase of Zinnia Drupal. Contains full support for:

- Importing users.
- Importing categories (Drupal vocabularies).
- Importing content.
- Importing comments.

1.5.2 0.1.1

Patch release for fixing some naming in documentation. No changes in functionality.

1.5.3 0.1.2

Minor packaging changes in order to allow upload to PyPI.

TWO

INDICES AND TABLES

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