gwsql Documentation

Release 0.1.2

team useblocks

February 06, 2017

Contents

groundwork-database provides database management functions to applications based on the framework groundwork.

It provides a pattern for sql database support and a small console plugin to inspect those databases.

The main features are:

- Support of multiple database connections
- Support of various SQL based database like:
- PostgresSQL
- sqlite
- MySQL
- MariaDB
- and all other databases, which are supported by SQLAlchemy

Note: groundwork-database is based on SQLAlchemy. All functions from SQLAlchemy are available inside groundwork plugins, which are using groundwork-database as pattern.

CHAPTER 1

Quickstart

To use groundwork-database inside a groundwork plugin, simply integrate it as followed:

```
from groundwork import App
from groundwork_database.patterns import GwSqlPattern

class MyPlugin(GwSqlPattern):
    def _init_(self, app, *args, **kwargs):
        self.name = "My Plugin"
        super().__init__(app, *args, **kwargs)

    def activate(self):
        name = "my_db"
        database_url = "sqlite:///:memory:"
        description = "My personal test database"
        db = self.databases.register(name, database_url, description)

        User = _get_user_class(db.Base)
        my_user = User(name="Me")
        db.add(my_user)
        db.commit()

    def print_user(name):
        db = self.databases.get("my_db")
        user = db.query(User).filter_by(name=name).first()
        if user is not None:
            print(user.name)
        else:
            print("User %s not found." % name)

    def _get_user_class(base):
        class User(base):
            id = Column(Integer, primary_key=True)
            name = Column(String)
        return User


if __name__ == "__main__":
    my_app = App()
    my_plugin = MyPlugin(my_app)
    my_plugin.activate()
    my_plugin.print_user("me")
```