TurboGears

The WebFramework that scales with you

Contact Informations

Website: http://www.turbogears.org
G+: http://gplus.tg.gy
Mailing List: turbogears@googlegroups.com
Michael Pedersen: m.pedersen@turbogears.org
Alessandro Molina: amol@turbogears.org

About

TurboGears is a Web Framework focused on rapid prototyping while trying to keep best possible balance with custom needs.

TurboGears2 is a reinvention of the original TurboGears project to take advantage of new components to provide a fully customizable WSGI (Web Server Gateway Interface) stack. From the beginning TurboGears was designed to be a Full Stack framework built from bestof-breed components.

The framework is designed to provide easy to use, productive defaults, while still providing flexibility where it's useful:

- **Intuitive Dispatch System** and Controllers as natural as writing a method. Never face a regular expression again!
- **Designer Friendly Template Engine** with compile time HTML validation, if you don't like it feel free to use Jinja2, Mako or Kajiki
- Easy to make and reuse **Pluggable Applications**
- Transactional Declarative **SQLAlchemy** ORM with **built in master-slave** support ready to be used. The framework will revert transactions for you when an error happens.
- Builtin Support for **MongoDB** using the Ming ODM
- Powerful and **Reusable Widgets** with AJAX support, validation and error reporting.
- **Automatic Forms** generation from models both on SQLAlchemy and MongoDB.
- Automatic **CRUD** and **Admin** sections generation
- Built in pluggable **Authentication and Authorization**
- Database **Migrations** and versioning
- Templates and Controllers **caching**
- Application wide and Controller **hooks** permit to extend existing applications and libraries

The Present

The newly released 2.2 version introduced a great amount of new features and improvements:

- Greatly reduced number of dependencies on newly quickstarted applications
- New application quickstart based on Twitter Bootstrap for rapidly prototyping responsive applications.
- It is now possible to quickstart applications using the Jinja2 and Kajiki template engines.
- Faster Dispatch Engine (2x faster than before)
- New repoze.who v2 based Authentication Configuration makes easier adding OpenID or custom auth.
- New default widgets library, tw2, makes possible to add server side logic to widgets to create complex AJAX widgets.
- More flexible data validation: Feel free to use FormEncode, tw2.core or your own custom validators
- Built in support for master-slave replication on SQLAlchemy based applications, reads will be randomly dispatched to slaves and writes will go to master database. A bunch of decorators and with-statement context are provided to force controllers or single parts of code to run on master database.
- Decoration inheritance, whatever template a controller exposes or validators it uses it is now possible to inherit them when overriding the method in subclasses

2.2 is going to be the last Python 2.x only release and the last Pylons based one. While future releases will guarantee to be totally backward compatible and will provide a Pylons- Compatible mode, 2.2 is intended to be a safe place for all the people that relayed on internal Pylons features or behavior.

The Past

TurboGears has been maintained since 2005 by many skilled Python developers and has seen recognition by some great projects like SourceForge.net and Fedora.

While attracting new users due to its simplicity and intuitive feeling, the choice of being the glue around other Python libraries had the side effect of resulting in unclear and often changing API with scattered documentation. After having tried to be a better CherryPy and a better Pylons, TurboGears is now trying to be just TurboGears consolidating its foundations, focusing on its philosphies and trying to provide a reliable API for the years to come.

TurboGears was originally created by Kevin Dangoor as the framework behind the Zesty Newsproduct. When he released it as an open source framework in the end of September 2005, it received more than 30,000 screencast downloads in the first 3 months.

On January 2007 Kevin Dangoor retired as project leader and The TurboGears project got managed jointly by a group of about half a dozen core developers under the leadership of Mark Ramm (as the TurboGears 2 development lead) and Florent Aide (as the Turbogears 1.x release manager). The first stable release of the 2.x branch had seen the light in May 2009.

As of Fall 2008, TurboGears had a large and healthy community with over 3000 users on the TurboGears mailing list, a book from Prentice Hall published in Nov. '06, and a number of open source and proprietary TurboGears applications deployed to the real world.

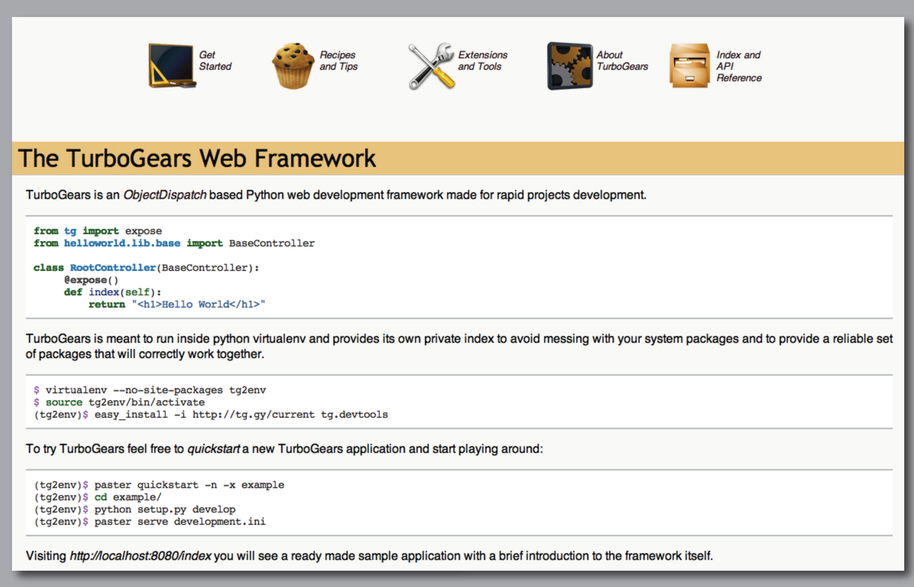
In 2010 the lead developers were called away due to real life issues and at the begin of 2011, the project had begun reorganizing under a new team. During 2011 TurboGears has managed to release new versions for the 2.1 branch and the new 2.2 major release.

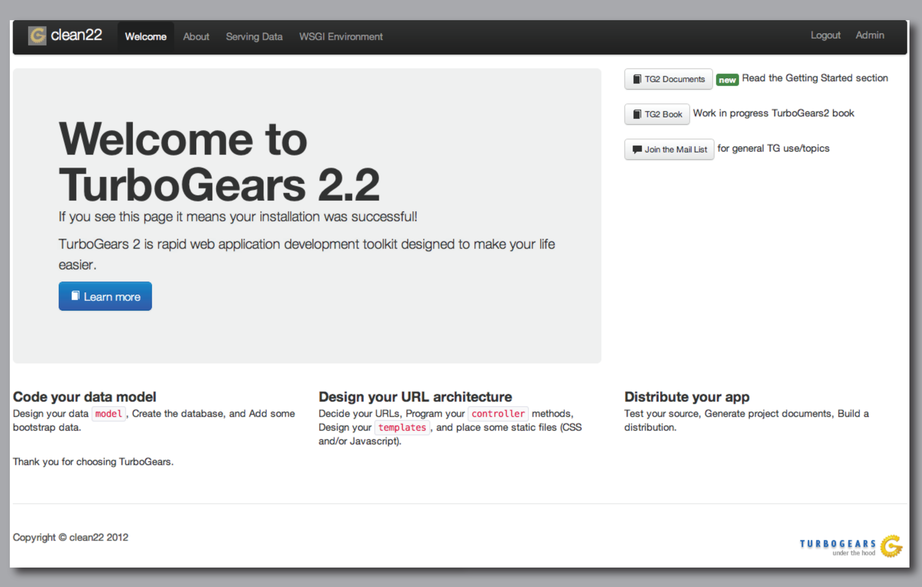
The Cogbin

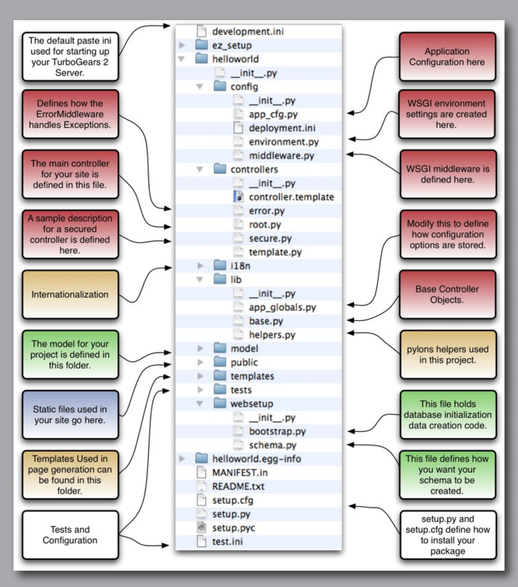
The TurboGears cogbin is a repository of extensions and pluggable applications that are ready to be used in new projects. The cogbin itself can be accessed at turbogears.org/cogbin, a short overview of the most used extensions is:

- **tgext.debugbar** provides the TurboGears debugbar with controller and template profiling and query inspection
- **tgext.pluggable** provides pluggable applications support for TurboGears
- **tgext.crud** autogenerate CRUD controllers based on SQLAlchemy or MongoDB models
- **tgext.admin** autogenerated administrative sections based on tgext.crud
- **tgext.menu** autogenerate menu bars from your application controllers
- **TurboMail** easy to use istant or asynchronous mail delivery
- **TGScheduler** schedule operations to be executed lazily or at scheduled times
- **tgext.datahelpers** advanced functions for models, attachments and data oriented caching
- **libacr** content management framework that permits to add content management functions to any TurboGears project, the ACRCms project is a web based CMS based on libacr
- **stroller** a pluggable, paypal based, ecommerce for TurboGears
- **tgapp-fbauth** Pluggable Facebook authentication for any TurboGears application
- **tgapp-registration** Pluggable registration for any TurboGears application
- **tgapp-smallpress** Pluggable blog system for any TurboGears application
- **tgapp-photos** Pluggable photo galleries for any TurboGears application
- **tgext.mobilemiddleware** expose different views when mobile devices access a controller
- **tgext.ajaxforms** make any form load and submit using ajax
- **tgext.asyncjob** queue of job to run asynchronously with advancement reporting
- **tgext.browserlimit** block older browsers based on the features they support
- **tgext.scss** support SCSS language for stylesheets
- **tgext.less** support LESS language for stylesheets
- **tgext.tagging** Tagging support and tag cloud widgets for any model
- **tgext.minify** JS and CSS automatic minification
- **tgext.geo** add GIS capabilities in a TurboGears2 application
- **tgext.command** add command line utilities that run in the context of your application

Framework Preview



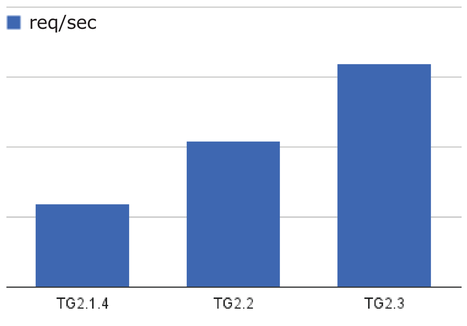




The Future

While on the short term the team is focusing on building up a great pool of reusable pluggable applications and refactoring the documentation, the 2.3 branch is where all the long term targets are being developed. The 2.3 branch will feature the first set of releases which won't be Pylons based anymore, this will open a wide set of opportunities permitting to improve speed, add new features and support Python 3. Development of 2.3 branch has already been in progess for the past 6 moths trying to focus on speed improvements and framework/documentation cleanup, adding the minimum set of required new features that made sense.

While performing refactoring on the codebase huge attention has been paid to backward compatibility, nothing should change for plain TurboGears applications and to provide the best possible backward compatibility with applications that made use of Pylons related code a **Pylons Compatible Mode** will be available. The pylons compatible mode, using monkeypatching, will make possible to keep Pylons related code around giving the time to migrate it. Main improvements in features will include the so called **Application Wrappers** which are some kind of middlewares that unlike WSGI middlewares run inside the TurboGears stack itself providing access to the TurboGears request, response, cache, session and so on.
They implement the same concept of the already available Controller Wrappers, but while controller wrappers run after dispatch has happened, the application wrappers run before the dispatcher permitting to also alter the request routing.



3x Faster than 2.1 series

```
app_config = AppConfig()
app_config['tg.root_controller'] = RootController()
app_config.init_config()

app = app_config.setup_tg_wsgi_app()(full_stack=False)
make_server('', 8080, app).serve_forever()
```

Single File Apps for Web Services and ClientSide World

python 3

Python3 Support