Tornado Server

Architecture Overview

Document Verison 1.0
17 July 2006

This document outlines the key Tornado Server architecture. It is not a full development guide; its purpose is to educate developers and administrators on the fundamentals of Tornado Server.

Contents

KEY COMPONENTS **3**

JAVA APPLICATION SERVER CODE 3
RELATIONAL DATABASE FOR SYSTEM TABLES 3
SYSTEM DATA 3

AUTHENTICATION AND AUTHORISATION **4**

DEVELOPING WEB APPLICATIONS **5**

COMPONENTS OF A TORNADO APPLICATION 5
Security *5*
Database Connections *5*
Keywords *5*
Resources *5*
Pages *5*
Actions *6*
Shared Code *6*
Scheduled Actions *6*
Business Widgets *6*
DESIGN PATTERNS 7
SELF-DOCUMENTING APPLICATIONS 7

WAS THIS DOCUMENT HELPFUL? **8**

Key Components

Java Application Server Code

The core of Tornado Server is very small. Without libraries it can be as small as about 800Kb. The files are organised into directories based on their functionality, for example all the core server files are in /bin, JDBC driver jar files are in /jdbc, server configuration files are in /config, temporary files in /temp, etc. Each directory has a readme.txt file in it to describe its use. Once you have the installed directory structure it can be copied easily between machines using a simple file copy. The server code is small and non-resource consuming, so it is easy to run a Tornado Server instance on your personal workstation without suffering any performance degradation.

The main file that powers the server is called puakma.jar (~500Kb in size). This is the only file required for a server upgrade. This makes it very easy to upgrade a Tornado instance either locally or remotely.

Relational Database for system tables

The unique feature of Tornado Server is that all application data is stored in a relational database. Once Tornado is started, it connects to the system database (usually a database named "puakma") and loads the custom web applications. Storing application data in a relational database has a number of advantages, including;

- Simple queries can find all applications using a specific method call or code fragment
- Each design element retains meta-data about its use and update history
- All applications can be easily altered by issuing an SQL update
- The design information is stored in a standard way for all applications
- Allows the Tornado Server core to be easily extended in the future to utilise other types of design elements
- Schema and meta-data can be extended for your own use

The table design can be viewed by checking the /config directory of your installation and opening a datadef.* file in a text editor.

System data

The main tables are APPLICATION and DESIGNBUCKET. The APPLICATION table holds the main entry point for an application, for example it has "Group" and "AppName" columns that determine a web applications URI, thus a "Group" of "system" and "AppName" of "admin" result in a web application being available at http://yourserver/system/admin.pma

Linking to the APPLICATION table is the DESIGNBUCKET. This table holds all design elements (eg Pages, Actions, Resources, etc). Each type of design element is identified by a "DesignType" INTEGER.

Storing the web applications in a database is fine, but there will be times where they need to be moved to other Tornado Server instances. To do this, Tornado has the ability to export the relational data to a compressed XML file (*.pmx, or PuakMa eXport). This *.pmx file can be easily copied, then imported into the destination Tornado installation. This activity is common when moving an application from a development server to a production server.

When Tornado is initially installed, three web applications will be automatically installed:

- /system/admin.pma : For administering the server instance using a web browser
- /system/webdesign.pma : For viewing and updating the design of web applications on that Tornado instance
- /puakma.pma : A simple entry point listing all the applications on the Tornado instance

Authentication and Authorisation

The standard Tornado Server installation uses a "pmaDefaultAuthenticator". This is a small piece of Java code that is called by the server to determine is a user can log in with the specified username/password credentials and if a particular user is in a group.

The power of authenticators is that new ones can be easily written to log in users against text files, SOAP calls, XMLRPC, almost anything is possible. This is the perfect way to hook Tornado into your existing directory for users. Tornado also includes an LDAPAuthenticator.

Authenticators may also be chained together. See Authenticators= in puakma.config

Developing Web Applications

Components of a Tornado Application

Tornado Server web applications have a definite structure and URL pattern. The well defined structure means that all developers can quickly understand how a web application is put together and the standard URL pattern means finding and resolving problems is very easy.

A Tornado Server web application is broken into the following parts.

Security

Defines the roles used by the developer and provides a mapping between those role names and users, groups and other roles. The “AllowAccess” role is used by Tornado to determine who can access the application, a value of “*” means anyone.

Database Connections

Database connections map a connection name to a physical database. Internally (in Action and Widget code) programmers simply refer to the connection name and Tornado will resolve that name to a physical database as specified by a database connection.

There can be any number of database connections in a web application.

Keywords

Keywords are a way of storing semi-volatile information in the application’s design. For example, a programmer may have a standard list of week names that are used in a number of combo box fields throughout the application. The combo box need only refer to that keyword, rather than recoding the list of week names again and again.

Resources

A resource is a static part of the application’s design, such as css, JavaScript or an image. Any type of file may be stored as a resource, with the resource’s content type instructing the browser how to process it.

The URI to reference a resource is: `/appname.pma/ResourceName?OpenResource`

Pages

Pages are HTML pages with a twist - they contain p-tags. A p-tag is a special tag recognised by Tornado as a “field”. At runtime, p-tags are converted automatically into their html equivalent by the framework. Consider the following:

```
<P@List name=”FavouriteDay” useKeyword=”WeekDays” @P>
```

This p-tag will generate a combo box with the list of choices coming from a keyword called "WeekDays". With one small change that can be changed to a set of radio buttons.
`<P@Radio name="FavouriteDay" useKeyword="WeekDays" @P>`

Each page has a "ContentType" piece of meta-data which is automatically sent to the browser with the page. It is usually "text/html", but can be easily changed to send plain text or xml, or any other type.

Pages can not contain any logic, they only contain a mix of text and p-tags. This cleanly separates visual layout from application logic.

The URI to reference a page is: `/appname.pma/PageName?OpenPage`
or: `/appname.pma/PageName?ReadPage`

Actions

Actions are small pieces of Java code which are executed automatically by the server. An action is loaded into the users context and run, once it has finished running it is discarded. An action does not behave like a J2EE servlet that is loaded once and shared by all users. For this reason it makes programming actions easier because the programmer can be certain the action is running only for the current user, thus does not have to worry to much about concurrency.

There are three ways actions can be run:
1. Directly called by the browser using an `?OpenAction` URI
2. Automatically when a page is opened
3. Automatically when a page is saved

Actions do the hard work of providing the interactivity, making a web application dynamic.

The URI to reference an action is: `/appname.pma/ActionName?OpenAction`

Shared Code

Shared Code is a place for storing addition Java classes used by actions and BusinessWidgets. Shared Code can be a simple class file or a precompiled jar. Shared Code is loaded automatically by Tornado and made available to all Actions and Business Widgets

Scheduled Actions

A scheduled action is code-wise exactly the same as a regular action. The only difference is that a Scheduled Action is run on a schedule (eg once per day) by the AGENDA server task. Scheduled Actions are useful for performing regular batch processing.

Business Widgets

Business Widgets are Java classes that have their public methods automatically exposed as SOAP web services.

Design Patterns

The standard Tornado application uses a design pattern called PRG, Post/Redirect/Get (see http://www.theserverside.com/common/printthread.tss?thread_id=20936).

The PRG pattern means that it is very easy to write Tornado web applications that all behave the same way. That way is: A user GETs a page and enters data in the page fields. Next they POST the data (eg submit button) to the Tornado server. Once the server receives and processes the POSTed data, it sends a Redirect to the browser causing the browser to GET the next page.

While Tornado makes it easy to use this pattern it does not enforce it strictly.

Self-Documenting Applications

For business grade applications, the bulk of the budget spend usually occurs in the maintenance phase of the application's life. Usually in this maintenance phase the original project team has long disappeared and the documentation produced for the project is a distant memory. Tornado serves to make web applications easier to maintain by encouraging programmers to develop in a structured way and storing the meta-data with the application forever. For example, Word documents and Visio diagrams can be attached to the application design, every time the application is copied to a new server (or a design is updated) its design information will be distributed with the application.

Further, Tornado applications have a data dictionary that serves two purposes. The first is a data dictionary that has descriptive information about the database, and secondly the database can be created based on the definition with one click.

Was this Document Helpful?

If you have any suggestions to improve this document, please send them to info@puakma.net