# 250px-Tglogo.jpg

# TinyG Web Server Version 0.00001

# Why wouldn’t you want to run a CNC machine from the web?

Table of Contents

Document Revision History 2

Introduction TingyG Web Server 3

TinyG Server Services 3

Tg Services 3

Supported commands 3

Server Service 4

Install instructions 5

To run: 5

Alertnatives. 6

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Author | Comment | Date |
| 1.0 | Jay Gill | A nice try | 8/14/2011 |
| 1.0.1 | Jay Gill | Added new features for load/save config. | 8/21/2011 |

# Introduction TingyG Web Server

TG server is an executable that provides a RESTful service layer on top of the Syntheos TinyG (TG) board. The server is built in Python (2.7) and leverages the Tornado Web Server.

At present the Server provides two top-level services tg and server. The tg service essentially acts as a pass through to send commands from the web to TG. However, having this service allows server side extensions to be made to the TG syntax. For example output from TG can be formatted in JSON or similar protocols.

The server service provides access to control the functions of the service itself. For example it provides calls to determine which serial port is connected to a TG board.

TG services are called in a restful manner. At present all services are invoked via http get commands. This was chosen to allow easy access via web browsers. Future versions will support more traditional REST leveraging GET, PUT etc.

# TinyG Server Services

## I [[1]](#footnote--1)Tg Services

TinyG services are accessed via the tg? Tag as follows:

http://<path to tg server:tg port>/tg?cmd1=args&cmd2=args…..

as an example

<http://localhost:8888/tg?g0> x10

### Supported commands

At present only one command is supported. That is a valid tg command. In addition an argument can be passed to define the return format (e.g. JSON). At present only JSON is supported.

# Server Service

The server services allow interaction with the server itself. The following commands are supported.

server?ListPorts – returns a list of ports available

Status: Not really tested!

server?OpenPort=<portname> opens the requested port to talk with TG.

Returns: a status string Port xxx is ??? where xxx is the port and ??? is Open or Error

Status : Tested on the Mac only

Server?SaveConfig=dummy DO NOT USE – In construction – will allow the current config to be saved. The principle use for this command is saving the open port.

Future versions will support baud rate etc.

Server?LoadConfig=dummy opens the file ServerConfig.JSON and sets the current port.

Examples

Get a list of ports on the current machine <should be moded to determine if a TG board is connected!>

Get a list of available ports.

<http://localhost:8888/server?ListPorts>

returns: Available Ports:/dev/tty.usbserial-A700fhWc (Does Not Work on PC???)

Open a port to talk to TG

http://localhost:8888/server?OpenPort=/dev/tty.usbserial-A700fhWc

returns: Port: /dev/tty.usbserial-A700fhWc is Open

Send a command to tg

<http://localhost:8888/tg?g0> x10

returns – the tg command prompt and moves x motor to 10 depending on the state of tg.

Send multiple commands to tg

<http://localhost:8888/tg?g0%20x10&?x=JSON&g0x20>

returns:

tinyg GCODE[mm] ok> {"Feed Rate": " 0.000 mm \\ min", "Motion mode": " G0 - linear traverse (seek)", "Position Z": " 11.000 mm", "Position Y": " 20.000 mm", "Position X": " 5.000 mm", "Feed rate mode": " G94 - units per minute", "Stop / end": " -- - running", "Offset I": " 0.000 mm", "Position A": " 0.000 degrees", "Seek Rate": " 0.000 mm \\ min", "Distance mode": " G90 - absolute distance", "Units": " G21 - millimeter mode", "Offset K": " 0.000 mm", "Offset J": " 0.000 mm", "Plane selection": " G17 - XY plane"} Units: G21 - millimeter mode Motion mode: G0 - linear traverse (seek) Plane selection: G17 - XY plane Distance mode: G90 - absolute distance Feed rate mode: G94 - units per minute Stop / end: -- - running Position X: 10.000 mm Position Y: 0.000 mm Position Z: 0.000 mm Position A: 0.000 degrees Offset I: 0.000 mm Offset J: 0.000 mm Offset K: 0.000 mm Seek Rate: 0.000 mm \ min Feed Rate: 0.000 mm \ min tinyg GCODE[mm] ok>

# Install instructions

<Future versions will have a simpler install (e.g. TGServer.exe)

1. Download and install Tornado
   1. Tornado can be downloaded from <http://www.tornadoweb.org/> which has full setup instructions
   2. To test try importing tornado.web into a python shell
2. Download and install Serial.py
   1. Serial.py can be downloaded from <http://pyserial.sourceforge.net/>
   2. To test try importing serial into a python shell
3. Once those are working Copy the following files and directories to your system
   1. simpleHTTP.py – main application
   2. serverconfig.json - the configuration file
   3. Static – the directory containing the static source for the web server
   4. TGJQ.html – the htmle file containing the TG terminal program
4. Edit the serverconfig.json file and set the portname to the port you will be using.

<Status – these instructions have not been tested>

## To run:

Start a python shell

Load simpleHTTP.py

Run the shell

The TG server will respond

TG Server Manager

Port xxx is open

Now you can list available ports

Go to a browser and type

http://<your host name :port>/server?ListPorts

this returns ports that are available

Now you can set up the port where your TinyG is located

http://<your host name:port?/server?OpenPort=<your port name>

example

<http://localhost:8888/server?OpenPort=/dev/tty.usbserial-A700fhWc>

# TG Web App

To start the web app go to

First start the SimpleHTTP.py app.

<http://localhost:8888/TGJQ.html>

In this application you to send commands to TG just type the gcode or TGcommand.

To send commands to the server enter server?<commands> at the prompt.

1. [↑](#footnote-ref--1)