tsung-gis Documentation

Release 1.1.0

Rodolphe Quiédeville

May 23, 2014

Contents

Contents:

CHAPTER 1

Modules

1.1 map.erl

1.1.1 Tsung Exports

Functions exported by the **map** module callable by Tsung.

- move_first/1

1.2 postgis.erl

Build SQL commend to manipulate geometric objects in a Postgis database.

The `postgis` module uses the `randomcoord` module to generate random datas.

1.2.1 Tsung Exports

Functions exported by the **postgis** module callable by Tsung. All functions which name that begins with **r_** return random data.

- r_box2d/1, r_box2d_srid/1
- r_point/1, r_point_srid/1

r_point/1

The function *r_point/1* returns a SQL command to build an ST_Point.

Paramaters : Tsung tuple {Pid, DynVars}

Return : string

```
12> postgis:r_point({Pid, DynVars}).
"ST_Point(69.896366, 63.997280)"
```

r_point_srid/1

The r_point_srid/1 functions works like *r_point/1* but enclose the `ST_Point()` result in `SetSRID()` postgis function.

Paramaters : Tsung tuple {Pid, DynVars}

Return : string

```
12> postgis:r_point_srid({Pid, DynVars}).
"ST_SetSRID(ST_Point(69.896366, 63.997280), 4326)"
```

r_box2d/1

The function *r_box2d/1* returns a SQL command to build a 2 dimension box with the ST_MakeBox2D.

Paramaters : Tsung tuple {Pid, DynVars}

Return : string

```
11> postgis:r_box2d_srid({Pid, Dynvars}).

"ST_SetSRID(ST_MakeBox2d(ST_Point(-72.170899, 51.890770),
ST_Point(30.764721, 75.803965))), 4326)"
```

r_box2d_srid/1

The function *r_box2d_srid/1* works like r_box2d/1 but enclose the result in `SetSRID()` postgis function.

Paramaters : Tsung tuple {Pid, DynVars}

Return : string

```
10> postgis:r_box2d({Pid, Dynvars}).
"ST_MakeBox2d(ST_Point(67.792555, -58.145776), ST_Point(163.686023,
88.730874))"
```

1.3 randomcoord.erl

1.3.1 Tsung Exports

Functions exported by the `randomcoord` module callable by Tsung

- rcoord_array/1
- url/1

rcoord_array/1

Function rcoord_array/1 return a tuple of coordinates {Lat, Lon}, Lat and Lon are float values.

Paramaters : classical tuple of tsung datas {Pid, DynVars}

Return : float tuple

```
10> randomcoord:rcoord_array({Pid, ts_dynvars:new()}).
["70.205974","33.121879"]
```

rcoord_array/4

Parameters : Left, Bottom, Right, Top

Return : array of float

Return an array of 4 value representing a bbox

1.3.2 Other exports

Functions exported by the `randomcoord` module not callable in a Tsung scenario

- rcoord/0
- rcoord/1
- rcoord/2
- rcoord/4

1.3.3 Geolocalized randoms

It is possible to specify country in DynVars to reduce the random size, this is done by define a DynVars called country in your scenario.

Countries already defined

- France
- Germany
- Portugal
- Spain

Define a new country

It's easy as create a new function *rcoord/1* with the country name as paramater, and return the value of rcoord/4 with the desired bbox.

In the following example the country name is *groland* which is defined by 7.14,42.84,14.07,56.84

```
rcoord("france")->
              rcoord(7.14,42.84,14.07,56.84);
```

1.4 reproj.erl

1.4.1 Tsung Exports

1.5 slippymap.erl

1.5.1 Tsung Exports

- deg2num/3
- num2deg/3
- num2bbox/3
- tmstowms/1
- tile2lat/2
- tile2lon/2

1.5.2 deg2num/3

```
3> slippymap:deg2num(48.84098,2.58741,18).
{132956,90202}
```

1.5.3 num2deg/3

```
5> slippymap:num2deg(132960,90200,18).
{2.59277344,48.84302835}
```

1.6 tms.erl

1.6.1 Tsung Exports

The following functions are directly callable in Tsung's scenario.

- move_first/1
- move_next/1
- move_north/1, move_south/1, move_west/1, move_east/1
- move_first_layers/1
- move_north_layers/1
- move_south_layers/1
- move_east_layers/1
- move_west_layers/1
- move_random_layers/1
- zoom_more_layers/1
- zoom_less_layers/1
- zoom_random_layers/1
- action_random_layers/1
- get_urlblock/1
- urlzxy/1

1.6.2 Other exports

The following functions are exported but are not callable directly on a Tsung scenario.

- get_urlfrom/2

1.6.3 Dynamic Variables

map_height

Define the map's height in pixel, integer value.

```
<setdynvars sourcetype="value" value="500">
    <var name="map_height" />
</setdynvars>
```

map_width

Define the map's width in pixel, integer value.

```
<setdynvars sourcetype="value" value="700">
    <var name="map_width" />
</setdynvars>
```

tms_layers

List of layer's names in comma separated value format.

```
<setdynvars sourcetype="value" value="roads,rivers,sea">
    <var name="tms_layers" />
</setdynvars>
```

1.6.4 Moving functions

move_first_layers/1

First move on all layers defined in DynVars,

Return a list of string representing tiles's urls

Required var : tms_layers, first_url

Sample usage :

```
<setdynvars sourcetype="erlang" callback="tms:move_first_layers">
  <var name="list_url" />
</setdynvars>

<foreach name="element" in="list_url">
  <request subst="true">
    <http url="/%%_element%%.png" method="GET" version="1.1"/>
  </request>
</foreach>
```

Result

```
["a/2/1/1","a/2/1/2","a/2/2/1","a/2/2/2",
 "b/2/1/1","b/2/1/2","b/2/2/1","b/2/2/2",
 "c/2/1/1","c/2/1/2","c/2/2/1","c/2/2/2"],
```

move_north_layers/1

As move_first_layers the function will return a list of string representing tiles's urls. Urls are compute

Required var : tms_layers, list_url

Sample usage :

1.7 wkb.erl

Erlang library to convert and generate datas in OGC Well Known Binary format defined by the Open Geospatial Consortium (OGC) and described in their Simple Feature Access.

Encodings are done in big indian.

1.7.1 Tsung Exports

Functions exported and callable direcxtly in a Tsung's scenario

wkb_point/1

Return a point with random coordinate in WKB format.

Paramaters : Tsung tuple {Pid, DynVars}

Return : string

```
24> wkb:wkb_point({Pid, DynVars}).
"0000000001C03FB4A8BE87B3804042B06F3D378054"
```

1.7.2 Other exports

float_to_wkb/1

Convert a float to OGC Well Known Binary format

Parameters : float

Return : string

```
79> wkb:float_to_wkb(42.14).
"404511EB851EB852"
```

wkb_point/2

wkb_linestring/1

Return a linestring encoded in WKB format.

Paramaters : Atrray of tuples {lat, lon}

Return : string

```
84> wkb:wkb_linestring([{42.5, -5.5},{135.2,5}]).
"00000000020000002C0160000000000004045400000000000401400000000000004060E66666666666"
```

1.8 xyz.erl

1.8.1 Tsung Exports

The following functions are directly callable in Tsung's scenarios.

- action_random/1
- move_first/1
- move_north/1, move_south/1, move_west/1, move_east/1
- move_first_layers/1
- move_north_layers/1
- move_south_layers/1
- move_east_layers/1
- move_west_layers/1

- move_random_layers/1
- zoom_more_layers/1
- zoom_less_layers/1
- zoom_random_layers/1
- action_random_layers/1
- get_urlblock/1
- urlzxy/1

1.8.2 Other exports

The following functions are exported but are not callable directly in Tsung's scenarios.

- get_urlfrom/2

1.8.3 Dynamic Variables

map_height

Define the map's height in pixel, integer value.

```
<setdynvars sourcetype="value" value="500">
    <var name="map_height" />
</setdynvars>
```

map_width

Define the map's width in pixel, integer value.

```
<setdynvars sourcetype="value" value="700">
    <var name="map_width" />
</setdynvars>
```

xyz_layers

List of layer's names in comma separated value format.

```
<setdynvars sourcetype="value" value="roads,rivers,sea">
    <var name="xyz_layers" />
</setdynvars>
```

1.8.4 Moving functions

move_first_layers/1

First move on all layers defined in DynVars,

Return a list of string representing tiles's urls

Required var : xyz_layers, first_url

Sample usage :

```
<setdynvars sourcetype="erlang" callback="xyz:move_first_layers">
  <var name="list_url" />
</setdynvars>

<foreach name="element" in="list_url">
  <request subst="true">
    <http url="/%%_element%%.png" method="GET" version="1.1"/>
  </request>
</foreach>
```

Result

```
["a/2/1/1","a/2/1/2","a/2/2/1","a/2/2/2",
 "b/2/1/1","b/2/1/2","b/2/2/1","b/2/2/2",
 "c/2/1/1","c/2/1/2","c/2/2/1","c/2/2/2"],
```

move_north_layers/1

As move_first_layers the function will return a list of string representing tiles's urls. Urls are compute

Required var : xyz_layers, list_url

Sample usage :

CHAPTER 2

Runnins tests

To run unit all tests you can do :

$ make dotest

Or for only one module, `tms` for example

$ make test $ /usr/bin/erl -noshell -pa ./ebin-test -s eunit test tms_tests -s init stop

CHAPTER 3

Continuous integration

A Jenkins job is set up for **tsung-gis** at http://jenkins.quiedeville.org/view/Tsung/job/TsungGIS/

CHAPTER 4

Indices and tables

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

m

p

r

s

t

w

x