User Inventory Format

Appendix A provides specifications on the format of the data files that must be supplied to customize the system to a user's facility inventory, fragilities and their user and notification information. An example of each file is provided with the V3 ShakeCast distribution in a folder found in the ShakeCast database directory, default at "/usr/local/shakecast/sc/db" A text editor is sufficient for customizing these files, though users with substantial inventories may consider keeping a database or spreadsheet for this purpose. The suggested strategy is to maintain or export CSV files for each of the required input files described below, and then use the drag-and-drop functionality within the ShakeCast web interface to upload (or update) any of the files.

Facility Data

The scope of facility data for ShakeCast V3 covers three main categories: 1) basic facility and associated simple fragility information; 2) probabilistic fragility information; and 3) supplemental feature information. Facility data can be prepared in the format of either Comma-Separated Values (CSV) or Extensible Markup Language (2003 Excel XML export format) to be imported into the ShakeCast database. Currently there are several import scripts for processing facility data of each category and to ensure backward compatibility with V2 facility data. Specifically, the format and requirement for basic facility and associated fragility information are identical for both V2 and V3 systems. Users can migrate from a V2 system to V3 using the same facility data file.

By default CSV fields are separated by commas; field values that include commas are protected by enclosing them in quotes, but these defaults can be modified if necessary. The first record in the input file must contain column headers allowing processing scripts to interpret the rest of the records. Each header field must specify a facility field, a facility metric field, or a group field. The header fields are case-insensitive; facility_name and FACILITY_NAME are equivalent. Fields can appear in any order.

Facility Fields

The following facility names are recognized. These fields correspond to tables and columns in the ShakeCast database. Please refer to the ShakeCast Database Description for a more detailed description of the structure of the ShakeCast Database.

external_facility_id (Text(32), required always)

This field identifies the facility. It must be unique for a facility type but the same external_facility_id may be used for different types of facilities.

facility_type (Text(10), required always)

This field identifies the type of facility. It must match one of the types in the facility_type table. Currently defined types are: BRIDGE, CAMPUS, CITY, COUNTY, DAM, DISTRICT, ENGINEERED, INDUSTRIAL, MULTIFAM, ROAD, SINGLEFAM, STRUCTURE, TANK, TUNNEL, UNKNOWN, and HAZUS building types. Refer the HAZUS Damage Level document for the 128 HAZUS building types and code era.

facility_name (Text(128), required for insert/replace)

The value of this field is what the user sees.

short_name (Text(10), optional)

The value of this field is used by ShakeCast when a shorter version of the name is needed due to space limitations in the output.

description (Text(255), optional)

You can use this field to include a short description of the facility.

lat (Float, required for insert/replace)

Specifies the latitude of the facility in degrees and fractional degrees.

lon (Float, required for insert/replace)

Specifies the longitude of the facility in degrees and fractional degrees.

Fragility Fields

Each field beginning with METRIC: is taken to be a facility fragility specifier. The format of a fragility specifier is:

METRIC: *metric-name* **:** *damage-level*

where metric-name is a valid ShakeMap metric (MMI, PGV, PGA, PSA03, PSA10, or PSA30) and alert-level is a valid damage level (GREEN, YELLOW, ORANGE, or RED). Examples of Facility Fragility column labels areMETRIC:MMI:REDandmetric:pga:yellow.

The metric-name values are defined by the ShakeMap system, and are generally not changed. The above values are current as of summer 2013. The damage-level values shown above are the default values shipped with ShakeCast. These values are defined in your local ShakeCast database, and you may use the administration web interface to change those values and the color-names that refer to them.

Attribute Fields

A facility can have attributes associated with it. These attributes can be used to group and filter facilities.

Each field beginning with ATTR:is taken to be a facility attribute specifier. The format of a facility attribute specifier is:

ATTR: *attribute-name* **:** *attribute-value*

where *attribute-name* is a string not more than 20 characters in length.

Examples of Facility Attribute column labels are `ATTR:COUNTY` and `ATTR:Construction`. Attribute values can be any string up to 30 characters long.

Example Facilities

Example 1 – Point Facilities

Assume we have a file named *ca_cities.csv* containing California cities that we want to load into the ShakeCast database. The file is in CSV format and includes the name of each city and the latitude/longitude of its city center or city hall. Records in the file are of the form

```
Rancho Cucamonga,34.1233,-117.5794
Pasadena,34.1561,-118.1318
```

The file is missing two required fields,external_facility_id and facility_type. Since the city name is unique we can add a new column that is a copy of the name column and use that as the external_facility_id. Another column containing the value CITY for each row is added for the facility_type. You can either make these changes using a spreadsheet program or with a simple script written in a text processing language like Perl.

After making these modifications the records look like

```
CITY,Rancho Cucamonga,Rancho Cucamonga,34.1233,-117.5794
CITY,Pasadena,Pasadena,34.1561,-118.1318
```

The input file also needs a header record; after adding one the input file looks like

```
FACILITY\_TYPE,EXTERNAL\_FACILITY\_ID,FACILITY\_NAME,LAT,LON
CITY,Rancho Cucamonga,Rancho Cucamonga,34.1233,-117.5794
CITY,Pasadena,Pasadena,34.1561,-118.1318
 ...
```

The facilities in this file can now be loaded into ShakeCast using the command

manage_facility.pl ca_cities.csv

Example 2 – Fragility Parameters

Building on the previous example, assume a simple model where Instrumental Intensity (MMI) above 7 corresponds to a high-level alert (RED), MMI between 5 and 7 corresponds to a medium-level alert (YELLOW), and MMI below 5 corresponds to a low alert level (GREEN). The lower threshold of each range (1, 5, 7) is appended to every record in the input file and the header record is changed to reflect the added fields:

```
FACILITY\_TYPE,EXTERNAL\_FACILITY\_ID,FACILITY\_NAME,LAT,LON, \
      METRIC:MMI:GREEN,METRIC:MMI:YELLOW,METRIC:MMI:RED
CITY,Rancho Cucamonga,Rancho Cucamonga,34.1233,-117.5794,1,5,7
CITY,Pasadena,Pasadena,34.1561,-118.1318,1,5,7
 ...
```

Example 3 – Multiple Attributes and Multiple Metrics

You can include multiple attributes, multiple metrics, or multiple attributes and multiple metrics for each row of an import file. For example,

```
FACILITY\_TYPE,EXTERNAL\_FACILITY\_ID,ATTR:COUNTY, ATTR:SIZE,\
      METRIC:MMI:GREEN, METRIC:MMI:YELLOW, METRIC:MMI:RED
CITY,Rancho Cucamonga,San Bernardino,Small,1,2,6
CITY,Pasadena,os Angeles,Medium,1,2,6
```

This file would be loaded using the command

```
manage\_facility.pl --update city\_county.csv
```

The above example updates the existing city locations to associate them with a county attribute and a size attribute, and defines the green, yellow, and red shaking alert thresholds.

Probabilistic Facility Fragility Data

ShakeCast V3 includes a generic processer for evaluating probabilities of exceedance of individual damage states and their likelihood as a combined set. The probability density function is modeled as log-normal distribution. To enable this optional function, users need to provide the mean (ALPHA) and the spread (BETA) value of a fragility curve for each potential damage state to be evaluated.

When preparing probability fragility data for ShakeCast import, the format requirements for facility data are applied to fragility data. Specifically, the **external_facility_id** and **facility_type** fields of the fragility data file must match the entry in the facility data file if they are imported separately. It is permitted to define more than one set of fragility curves targeting different aspects of facility performance. This function is implemented with an additional facility attribute field called " **component**." It is a user-defined field for describing facility-specific components to be modeled and evaluated for a given ground motion input. The user needs to specify the designated component of the facility for each fragility curve set. It is not required to define fragility curves for all potential damage states for a component. This usually applies to modeling secondary components or general stress indicators.

The user should be aware that the assessment of probabilistic analysis is considered as a secondary analysis. Results of the analysis will be stored on the ShakeCast system for inquiry by expert users but will not be sent out as the primary method for notification. Among defined components, the ALPHA value of the "SYSTEM" component will be translated as simple fragility information to be used as part of the basic facility information. This fragility information will be used for triggering notifications.

Facility Fields

The following facility names are recognized. These fields correspond to tables and columns in the ShakeCast database. Please refer to the ShakeCast Database Description for a more detailed description of the structure of the ShakeCast Database.

external_facility_id (Text(32), required always)

This field identifies the facility. It must match the one specified in the basic facility data file if the information is entered separately.

facility_type (Text(10), required always)

This field identifies the type of facility. It must match one of the types in the facility_type table. It must match the one specified in the basic facility data file if the information is entered separately.

class (Text(32), optional)

The value of this field is used by ShakeCast to categorize components.

component (Text(255), required always)

The value of this field is used by ShakeCast to specify the target component of the defined fragility curve.

Fragility Fields

Each field beginning with METRIC: is taken to be a facility fragility specifier. The format of a fragility specifier is:

METRIC: *metric-name***:[ALPHA,BETA]:***alert-level*

where *metric-name* is a valid Shakemap metric (MMI, PGV, PGA, PSA03, PSA10, or PSA30) and *alert-level* is a valid alert level (GREEN, YELLOW, ORANGE, or RED). Examples of Facility Fragility column labels are METRIC:MMI:ALPHA:RED and metric:pga:alpha:yellow.

The metric-name values are defined by the ShakeMap system, and are generally not changed. The above values are current as of summer 2013. The alert-level values shown above are the default values shipped with ShakeCast. These values are defined in your local ShakeCast database, and you may use the administration web interface to change those values and the color-names that refer to them.

Example Probabilistic Fragility Parameters

Assume a simple probability model where Instrumental Intensity (MMI) with ALPHA of 8 and BETA of 0.6 corresponds to a high-level alert (RED), MMI with ALPHA of 7 and BETA of 0.6 corresponds to a medium-level alert (YELLOW), and MMI with ALPHA of 5 and BETA of 0.6 corresponds to a low-level alert (GREEN). The input file and the header record is changed to reflect the added fields:

```
FACILITY\_TYPE,EXTERNAL\_FACILITY\_ID, \
     METRIC:MMI:ALPHA:GREEN,METRIC:MMI:BETA:GREEN, \
      METRIC:MMI:ALPHA:YELLOW,METRIC:MMI:BETA:YELLOW, \
      METRIC:MMI:ALPHA:RED,METRIC:MMI:BETA:RED
CITY,Rancho Cucamonga,Rancho Cucamonga,34.1233,-117.5794,5,0.6,7,0.6,8,0.6
CITY,Pasadena,Pasadena,34.1561,-118.1318,5,0.6,7,0.6,8,0.6
 ...
```

Facility Feature Data

ShakeCast facility feature data allows users to define geometric footprints and rich content descriptions of a facility. The optional description field is specifically designed to store a custom HTML snippet to be displayed in the ShakeCast web interface. Content of this field will mask the data from the description field of basic facility information. The optional facility geometry field is designed to allow ShakeCast to take in account the extent of facility footprints when assessing ground shaking and damage state. The ShakeMap data grid is usually produced at a resolution of ~2x2 km, depending on the producer. This feature is directly applicable to facilities with linear feature such as roadways, aqueducts, etc. For facilities with small footprints there is no added benefits to define complex geometry other than either point or rectangular type.

When preparing feature data for ShakeCast import, the format requirements for facility data are applied to fragility data. Specifically, the **external_facility_id** and **facility_type** fields of the feature data file must match the entry in the facility data file if they are imported separately. The user needs to specify the geometry type of the facility for the defined geometry coordinate set. The user should be aware that the assessment of probabilistic analysis is considered as a secondary analysis. In order to accommodate the unique nature HTML snippet for the facility description, the facility feature data needs to be in the format of XML with the CSV field definition translated to tagged format.

Facility Fields

The following facility names are recognized. These fields correspond to tables and columns in the ShakeCast database. Please refer to the ShakeCast Database Description for a more detailed description of the structure of the ShakeCast Database.

external_facility_id (Text(32), required always)

This field identifies the facility. It must match the one specified in the basic facility data file if the information is entered separately.

facility_type (Text(10), required always)

This field identifies the type of facility. It must match one of the types in the facility_type table. It must match the one specified in the basic facility data file if the information is entered separately.

geom_type (Text(32), required always)

The value of this field is used by ShakeCast to handle the geometry coordinates from the **geom** field. Currently defined types are: POINT, POLYLINE, POLYGON, RECTANGLE, and CIRCLE.

geom (Text(mediumtext), required always)

The value of this field is used by ShakeCast to specify the coordinates of the facility. The format of this field is in (longitude,latitude) pairs separating by a white space. The size limit of data is ~16MB.

description (Text(mediumtext), required always)

You can use this field to include a description of the facility. The size limit of data is ~16MB.

Example Probabilistic Fragility Parameters

Assume a simple probability model where Instrumental Intensity (MMI) with ALPHA of 8 and BETA of 0.6 corresponds to a high-level alert (RED), MMI with ALPHA of 7 and BETA of 0.6 corresponds to a medium-level alert (YELLOW), and MMI with ALPHA of 5 and BETA of 0.6 corresponds to low-level alert (GREEN). The input file and the header record is changed to reflect the added fields:

```
FACILITY\_TYPE,EXTERNAL\_FACILITY\_ID, \
      METRIC:MMI:ALPHA:GREEN,METRIC:MMI:BETA:GREEN, \
      METRIC:MMI:ALPHA:YELLOW,METRIC:MMI:BETA:YELLOW, \
      METRIC:MMI:ALPHA:RED,METRIC:MMI:BETA:RED
CITY,Rancho Cucamonga,Rancho Cucamonga,34.1233,-117.5794,5,0.6,7,0.6,8,0.6
CITY,Pasadena,Pasadena,34.1561,-118.1318,5,0.6,7,0.6,8,0.6
 ...
```

User Group Data

ShakeCast GROUP is a new user classification introduced in V3, to provide area-specific processing and notifications for geographic regions defined by the users. For user notifications, membership in a GROUP is a primary channel of notifications for users with the same need of earthquake information.

In ShakeCast V3, GROUP polygon data is used as a geospatial filter for incoming earthquakes and ShakeMaps. This function enables ShakeCast users to define their own monitoring regions beyond the existing ShakeMap region boundaries.

User GROUP data is given in Apache config format. Lines beginning with ‘#’ and empty lines will be ignored. Spaces at the beginning and the end of a line will also be ignored as well as tabulators. If you need spaces at the end or the beginning of a value you can use apostrophe **“**. An option line starts with its name followed by a value. An ‘=’ sign is optional. Some possible examples:

```
      user    max
      user  = max
      user            max
```

If there is more than one statement with the same name, it will create an array instead of a scalar.

Each group is defined as a **block** of options. A **block** looks much like a block in the apache config format. It starts with < **blockname** > and ends with </ **blockname** >. An example:

```
<CI>
        POLY         35.8000 -116.4000        \
                34.0815 -114.4717        \
                32.0000 -114.3333        \
                32.0000 -120.5000        \
                34.5000 -121.2500        \
                37.2167 -118.0167        \
                36.6847 -117.7930        \
                35.8000 -116.4000
        <NOTIFICATION>
                    NOTIFICATION\_TYPE        NEW\_EVENT
                    DELIVERY\_METHOD        EMAIL\_HTML
                EVENT\_TYPE                ALL
        </NOTIFICATION>
        <NOTIFICATION>
                    NOTIFICATION\_TYPE        NEW\_PROD
                    DELIVERY\_METHOD        EMAIL\_HTML
                    PRODUCT\_TYPE                GRID\_XML
                EVENT\_TYPE                ALL
        </NOTIFICATION>
 </CI>
```

Each group is defined as a **block** of options. A **block** looks much like a block in the well known apache config format. It starts with < **blockname** > and ends with </ **blockname** >. The above example defines the user group **CI**.

Group Tag Names

The following group tag names are recognized. These fields correspond to tables and columns in the ShakeCast database. Please refer to the ShakeCast Database Description for a more detailed description of the structure of the ShakeCast Database.

poly (float pairs, required always)

This field identifies the boundaries of the group geometry. It must contain at least three anchor points in order to define a polygon. The total number of anchor points should limit to less than 100, otherwise the administration interface may not be able to display the entire polygon during editing. The manage_profile.pl will however process the polygon definition.

notification (Text(32), optional)

One notification block represents one notification request associated with the group and applies to all facilities within the group polygon. Multiple notification blocks for a group are permitted.

facility_type (Text, optional)

One notification block represents facility types to be associated with the group and applies to all facilities of the specified types within the group polygon. Multiple type specifications must be separated by white spaces.

description (Text(255), optional)

One description block includes a simple description of the group.

Notification Tag Names

Each notification block defines one notification request. Tag names are corresponding to the field names of the table “notification_request.” Required tags for a notification block include NOTIFICATION_TYPE, DELIVERY_METHOD, and EVENT_TYPE. Valid notification types are CAN_EVENT, NEW_EVENT, UPD_EVENT, SHAKING, NEW_PROD, and DAMAGE.

can_event

This notification request is triggered when an event is cancelled by the seismic network in which the event was located and removed from the USGS web site. Require EVENT_TYPE and DELIVERY_METHOD tags.

new_event

This notification request is triggered when an event is located by a seismic network. A ShakeMap may or may not be produced for the earthquake depending on triggering criteria defined by the ShakeMap producers. Require EVENT_TYPE and DELIVERY_METHOD tags.

upd_event

This notification request is triggered when the source parameters of an event is updated with a new version by the seismic network. New versions of ShakeMaps for the event may or may not coincide with an updated event. Require EVENT_TYPE and DELIVERY_METHOD tags.

new_prod

This notification request is triggered when a specified ShakeMap product of an event is available on the USGS web site. Require EVENT_TYPE,DELIVERY_METHOD, and PRODUCT tags.

shaking

This notification request is triggered when the ground shaking parameter at the location of the facility exceeds the preset value. Require EVENT_TYPE,DELIVERY_METHOD, METRIC, and LIMIT_VALUE tags.

alert-level

This notification request is triggered when the ground shaking parameter at the location of the facility falls between the high and low values of facility fragility settings. Require EVENT_TYPE,DELIVERY_METHOD, and ALERT_LEVEL tags.

User Data

The scope of user data for ShakeCast V3 covers three user categories: 1) regular user; 2) group user; and 3) administrative user. Besides the additional group user type, there is little change to the requirements of user data and they can be prepared in the CSV format to be imported into the ShakeCast database.

Similar to facility CSV data, the first record of user data file must contain column headers. These headers tell manage_user.pl how to interpret the rest of the records. Each header field must specify a user name field and a user type field. The header fields are case-insensitive; username and USERNAME are equivalent. Fields can appear in any order.

User Fields

The following facility names are recognized. These fields correspond to tables and columns in the ShakeCast database. Please refer to the ShakeCast Database Description for a more detailed description of the structure of the ShakeCast Database.

username (Text(32), required always)

This field identifies the user. It must be unique for a user type.

user_type (Text(10), required always)

This field identifies the type of use. It must match one of the types in the user_type table. Currently defined types are: ADMIN, USER, and SYSTEM.

full_name (Text(32), optional)

The value of this field is the user's full name.

email_address (Text(255), optional)

The value of this field is the user's email address for receiving communication from the ShakeCast system.

password (Text(64), optional)

The value of this field is used by ShakeCast to generate password for accessing the ShakeCast interface and the web site if password protected. Internally the password is saved inside the database using a cryptographic hash function SHA-256.

phone_number (Text(32), optional)

You can use this field to include a user's phone number.

Delivery Method Fields

Each field beginning with DELIVERY: is taken to be a delivery method specifier. The format of a delivery method specifier is:

DELIVERY: *delivery-method*

where *delivery-method* is a valid message format (PAGER, EMAIL_HTML, or EMAIL_TEXT). Examples of Delivery Method column labels are DELIVERY:EMAIL_HTML and delivery:email_html.

The message format values are defined by the ShakeCast system, and are generally not changed. The alert-level values shown above are the default values shipped with ShakeCast. These values are defined in your local ShakeCast database, and you may use the administration web interface to change those values and the color-names that refer to them.

Group Fields

A user can have notification requests replicated from an existing group. Each field beginning with GROUP: is taken to be a group specifier. The format of a profile specifier is:

GROUP: *group-name[:goup-name…]*

where *group-name* is a valid group name.

User Fields

A user can have notification requests replicated from an existing user. Each field beginning with USER: is taken to be a user specifier. The format of a user specifier is:

USER: *shakecast-user*

where *shakecast-user* is a valid user id. In V3, use of this option is discouraged.

Example Users File

Assume we have a file named *test_user.csv* containing users that we want to load into the ShakeCast database. The file is in CSV format and includes the name of each user, user delivery method and group association. The input file with the header record looks like

```
USER\_TYPE,USERNAME,PASSWORD,FULL\_NAME,EMAIL\_ADDRESS,PHONE\_NUMBER,DELIVERY:PAGER,D
ELIVERY:EMAIL\_HTML,GROUP:GLOBAL
USER,test\_user,sc4all,Test User,testuser@usgs.gov,(123)456-7890, testuser@usgs.gov,
testuser@usgs.gov,GLOBAL
   ...
```

Plain-Text Product Template

ShakeCast V3 uses the same template engine (the Perl Template Toolkit) as V2 to generate plain-text products, such as csv, xml, and kml, etc. only available on the local ShakeCast system. A template also defined as a ShakeCast product can be included as an attachment in the ShakeCast notification to be delivered to the users directly. Please see the Template Manual manpage for the complete reference which goes into much greater details about the features and use of the Template Toolkit.

This section covers a brief summary of the template directives. ShakeCast specific identifiers include exposure, item, and type. Facility specific identifiers include name, latitude, longitude, alert_level, MMI, PGA, PGV, PSA03, PSA10, and PSA30.

GET

Evaluate and print a variable or value.

```
    [%   GET variable %]
    [%       variable %]
    [%       hash.key %]
    [%         list.n %]
    [%     code(args) %]
    [% obj.meth(args) %]
    [%  "value: $var" %]
```

CALL

As per GET but without printing result (e.g. call code)

```
[%  CALL variable %]
```

SET

Assign a values to variables.

```
[% SET variable = value %]    # 'SET' also optional
[%     variable = other\_variable
       variable = 'literal text @ $100'
       variable = "interpolated text: $var"
       list     = [val, val, val, val, ...]
       list     = [val..val]
       hash     = { var => val, var => val, ... }
%]
```

DEFAULT

Like SET above, but variables are only set if currently unset (i.e. have no true value).

```
[% DEFAULT variable = value %]
```

INSERT

Insert a file without any processing performed on the contents.

```
[% INSERT legalese.txt %]
```

INCLUDE

Process another template file or block and include the output. Variables are localised.

```
[% INCLUDE template %]
[% INCLUDE template  var = val, ... %]
```

PROCESS

As INCLUDE above, but without localising variables.

```
[% PROCESS template %]
[% PROCESS template  var = val, ... %]
```

WRAPPER

Process the enclosed block WRAPPER … END block then INCLUDE the named template, passing the block output in the ‘content’ variable.

```
[% WRAPPER template %]
   content...
[% END %]
```

BLOCK

Define a named template block for subsequent INCLUDE, PROCESS, etc.,

```
    [% BLOCK template %]
       content
    [% END %]
```

FOREACH

Repeat the enclosed FOREACH … END block for each value in the list.

```
    [% FOREACH variable = [val, val, val] %]    # either
    [% FOREACH variable = list %]                 # or
    [% FOREACH list %]                            # or
       content...
       [% variable %]
    [% END %]
```

WHILE

Enclosed WHILE … END block is processed while condition is true.

```
    [% WHILE condition %]
       content
    [% END %]
```

IF / UNLESS / ELSIF / ELSE

Enclosed block is processed if the condition is true / false.

```
    [% IF condition %]
       content
    [% ELSIF condition %]
         content
    [% ELSE %]
         content
    [% END %]
    [% UNLESS condition %]
       content
    [% # ELSIF/ELSE as per IF, above %]
       content
    [% END %]
```

SWITCH / CASE

Multi-way switch/case statement.

```
[% SWITCH variable %]
[% CASE val1 %]
   content
[% CASE [val2, val3] %]
   content
[% CASE %]         # or [% CASE DEFAULT %]
   content
[% END %]
```

MACRO

Define a named macro.

```
[% MACRO name <directive> %]
[% MACRO name(arg1, arg2) <directive> %]
...
[% name %]
[% name(val1, val2) %]
```

FILTER

Process enclosed FILTER … END block then pipe through a filter.

```
[% FILTER name %]                       # either
[% FILTER name( params ) %]             # or
[% FILTER alias = name( params ) %]     # or
   content
[% END %]
```

USE

Load a “plugin” module, or any regular Perl module if LOAD_PERL option is set.

```
[% USE name %]                          # either
[% USE name( params ) %]                # or
[% USE var = name( params ) %]          # or
...
[% name.method %]
[% var.method %]
```

PERL / RAWPERL

Evaluate enclosed blocks as Perl code (requires EVAL_PERL option to be set).

```
    [% PERL %]
         # perl code goes here
         $stash->set('foo', 10);
         print "set 'foo' to ", $stash->get('foo'), "\n";
         print $context->include('footer', { var => $val });
    [% END %]
    [% RAWPERL %]
       # raw perl code goes here, no magic but fast.
       $output .= 'some output';
    [% END %]
```

TRY / THROW / CATCH / FINAL

Exception handling.

```
    [% TRY %]
         content
       [% THROW type info %]
    [% CATCH type %]
         catch content
       [% error.type %] [% error.info %]
    [% CATCH %] # or [% CATCH DEFAULT %]
         content
    [% FINAL %]
       this block is always processed
    [% END %]
```

NEXT

Jump straight to the next item in a FOREACH/WHILE loop.

```
`[% NEXT %]`
```

LAST

Break out of FOREACH/WHILE loop.

```
`[% LAST %]`
```

RETURN

Stop processing current template and return to including templates.

```
`[% RETURN %]`
```

STOP

Stop processing all templates and return to caller.

```
`[% STOP %]`
```

TAGS

Define new tag style or characters (default: [% %]).

```
    [% TAGS html %]
    [% TAGS <!-- --> %]
```

COMMENTS

Ignored and deleted.

```
    [% # this is a comment to the end of line
       foo = 'bar'
    %]
    [%# placing the '#' immediately inside the directive
        tag comments out the entire directive
%]
```

Example Exposure Template

Assume we have a file named *exposure_csv.tt* containing template directives that we want to generate a local ShakeCast product "*exposure.csv*." The template file first includes a static header is in CSV format. The main body of the template file contains a directive that loops through exposure facilities and outputs selected fields, including basic facility information, shaking estimates and damage estimate. The template file with the header record looks like

```
FACILITY\_TYPE,FACILITY\_ID,FACILITY\_NAME,DIST,LATITUDE,LONGITUDE,\
DAMAGE\_LEVEL,MMI,PGA,PGV,PSA03,PSA10,PSA30,STDPGA,SVEL
[% FOREACH exposure = shakecast.exposure %]
[%- FOREACH item = exposure.item -%]
[% exposure.type %],"[% item.external\_facility\_id %]",\
"[% item.facility\_name %]",[% item.DIST %],[% item.latitude %],\
[% item.longitude %],[% item.damage\_level %],[% item.MMI %],\
[% item.PGA %],[% item.PGV %],[% item.PSA03 %],[% item.PSA10 %],\
[% item.PSA30 %],[% item.STDPGA %],[% item.SVEL %]
[%- END -%]
[% END %]
```

and the output exposure.csv looks like

```
FACILITY\_TYPE,FACILITY\_ID,FACILITY\_NAME,DIST,LATITUDE,LONGITUDE,DAMAGE\_LEVEL,MMI,
PGA,PGV,PSA03,PSA10,PSA30,STDPGA,SVEL
CITY,"101614","Warm Springs, NV (pop. 1K)",111.15,38.2,-116.4,GREEN,1,0.02,0.01,0.01,
0,0,,784
CITY,"100241","Caliente, NV (pop. 1.1K)",86.76,37.615,-114.511,GREEN,1.08,0.02,0.01,0
.02,0,0,,483.25
CITY,"100019","Alamo, NV (pop. < 1K)",32.61,37.365,-115.164,GREEN,3.08,0.15,0.04,0.15
,0.02,0,,460.5
   ...
```

Portable Document Format (PDF) Product Template

ShakeCast V3 introduces a new template engine to generate reports with flexible layouts in PDF format. Each PDF product template consists of one base PDF template and one configuration (or directive) file. Earthquake-specific PDF output will be saved into the earthquake-specific data directory under the same name as the PDF template. A template also defined as a ShakeCast product can be included as an attachment in the ShakeCast notification to be delivered to the users directly.

PDF directive file must be prepared in XML format. The PDF template engine runs as a middleware to translate directives to PDF layout commands. Thus although there are no ShakeCast-specific requirements, users need to refer to the Adobe PDF specifications regarding text, fonts, graphics, and other information needed to display it.

Each styled-content is defined as a **block** of options. A **block** looks much like a block in the well-known XML tag. It starts with < **blockname** > and ends with </ **blockname** >. An example:

```
        <image>
                <path>screenshot.jpg</path>
                <type>jpeg</type>
                <x>0</x>
                <y>0</y>
                <w>8.0</w>
                <h>4.0</h>
                <unit>inch</unit>
                <align>center</align>
                <valign>center</valign>
                <pad>0.1</pad>
        </image>
```

defines the content and layout of an image.

The following PDF tag names are recognized. These fields correspond to specific PDF format specification.

page

Insert a new page in the PDF document. The example below inserts a new page in the PDF document and imports a DYFI pdf into the page.

```
        <page>
                <pdf>
                        <path>eq\_product/[EVID]/\*\_ciim.pdf</path>
                </pdf>
        </page>
```

block

Insert a block of content inside a page at the specified location. The example below paints a gray rectangle with black borders and inserts a paragraph of text inside the block.

```
        <block fillcolor="lightgrey" strokecolor="black" >
                <action>rect</action>
                <style>fillstroke</style>
                <x>0.1</x>
                <y>8.3</y>
                <w>8.3</w>
                <h>0.7</h>
                <unit>inch</unit>
                <text>
                        <string size="12" >These results are from an automated system
 and users should consider the preliminary nature of this information when making dec
isions relating to public safety. ShakeCast results are often updated as additional o
r more accurate earthquake information is reported or derived.</string>
                        <x>0.15</x>
                        <y>8.8</y>
                        <w>8.2</w>
                        <h>1.0</h>
                        <lead>10</lead>
                        <align>justify</align>
                        <unit>inch</unit>
                </text>
        </block>
```

text

Insert a text block at the specified location.

```
        <text>
                <string size="22" type="Times-Bold" >Magnitude [MAG] - [LOCSTRING]</s
tring>
                <x>0.1</x>
                <y>9.9</y>
                <w>7.0</w>
                <h>0.5</h>
                <align>justify</align>
                <unit>inch</unit>
        </text>
```

image

Insert an image from an external file at the specified location with respect to the event directory. If width (w) and height (h) are specified, the image will be resized to the specified dimensions. Acceptable image types are jpeg, tiff, png, gif, and gd.

```
        <image>
                <path>screenshot.jpg</path>
                <type>jpeg</type>
                <x>0</x>
                <y>0</y>
                <w>8.0</w>
                <h>4.0</h>
                <unit>inch</unit>
                <align>center</align>
                <valign>center</valign>
                <pad>0.1</pad>
        </image>
```

table

Insert a table from an external CSV file to the current page. The example below inserts a table to the specified location of the current page. New pages will be inserted if the length of the table exceeds the page height.

```
        <table>
                <list>exposure.csv</list>
                <type>CITY USGS</type>
                <x>0.1</x>
                <w>8.3</w>
                <start\_y>3.9</start\_y>
                <next\_y>10.75</next\_y>
                <start\_h>3.0</start\_h>
                <next\_h>10.0</next\_h>
                <font\_size>8</font\_size>
                <padding>2</padding>
                <padding\_right>2</padding\_right>
                <background\_color\_even>snow</background\_color\_even>
                <background\_color\_odd>wheat</background\_color\_odd>
                <unit>inch</unit>
                <border>0.25</border>
                <border\_color>snow</border\_color>
        <field>FACILITY\_TYPE,FACILITY\_ID,FACILITY\_NAME,DIST,DAMAGE\_LEVEL,MMI,PGA,
PGV,PSA03,PSA10,PSA30,SDPGA,SVEL</field>
        </table>
```

[]:

Tags: troubleshooting (tag_troubleshooting) shakecast_v3 (tag_shakecast_v3)

Page last updated: July 3, 2016
Site last generated: Nov 15, 2016