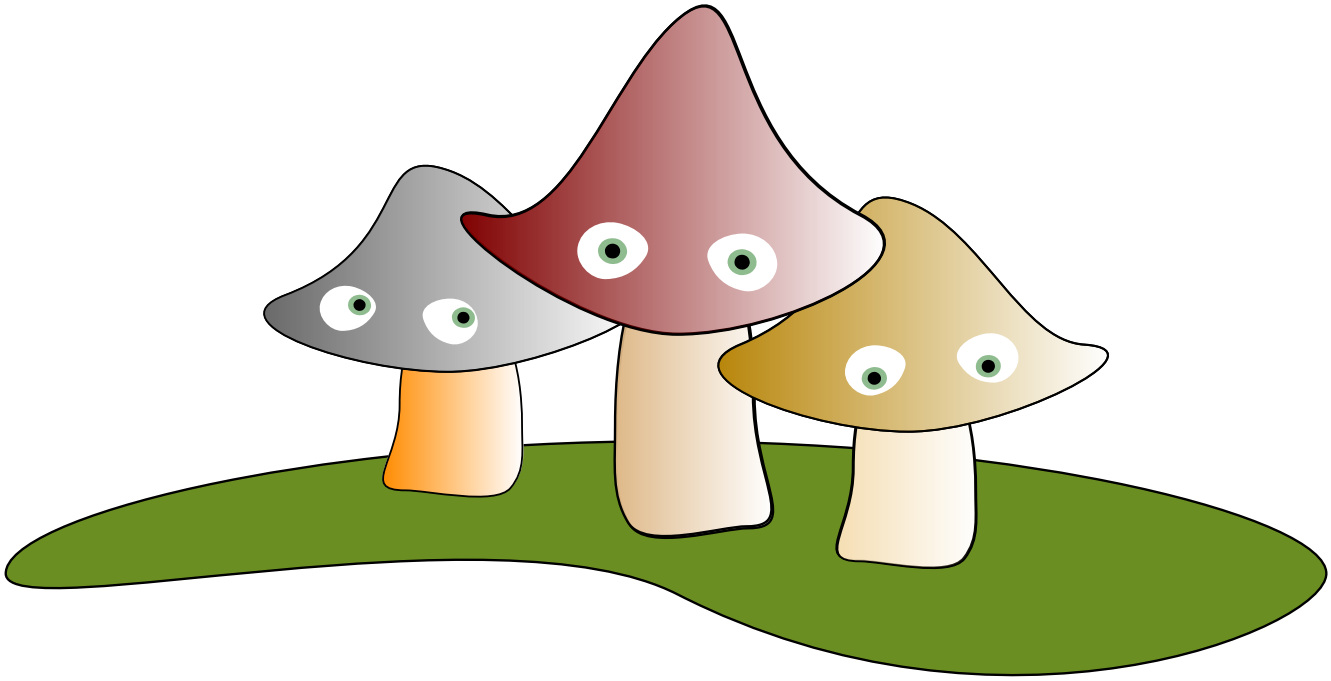
The High Cost of Mushroom Farming



Quality requires clear standards and strict enforcement. But these measures yield only limited benefit when key people lack reliable, timely feedback to alert them when something fails -- and how to fix it.

Real estate agents originate virtually all listings data consumed by search engines. But agents also receive virtually no *clear and immediate feedback* to tell them when and why some listings are rejected. Agents are treated like mushrooms, deep in the dark with no *actionable* information. It's not pretty.

A "mushroom farm" always looks like that, regardless of the specific process or industry. It's a useless and expensive drain on both people and profits.

Feedback systems drive out these hidden costs, allowing business processes to sustain **ever-rising service levels** at **ever-diminishing cost**. Everyone benefits from continuous improvement, end to end.

What follows is a simple feedback system to benefit property search engines -- and the agents who keep them well fed.

Version 1.20 of Dec 12, 2019

A Global Feedback System for Real Estate Postings

Overview

This document describes a standard method for "feedback" reporting by search engines to inform feed providers and their users of success or failure of listings submitted for posting. It supersedes any prior description of this method in any form.

The need has become more pressing with time for implementation of a single, flexible, non-proprietary reporting scheme. We describe hereafter a practical method which has been implemented "live" and has demonstrably achieved that fundamental goal:

Format Agnostic

We use XML as our base case for examples but the data encapsulation method is totally irrelevant. That is, JSON or any other format will achieve the same goal. We are defining only a *standardized coding method*, not a standard for data transfer since different engineers will likely have different preferences.

Defining "Success"

Search engines and their feed partners need a simple but flexible means to implement automated "feedback" reporting at whatever level of detail their respective managements may see fit. This proposed method allows search engines to commence reporting with virtually no detail offered beyond basic "pass/fail", but with full latitude to expand reporting detail as local priorities and resources may later permit. Ease of early implementation both by search engines and feed providers is a vital element of "success". Less than perfect is always better than nothing.

A truly successful reporting method must also be *globally adaptable* if it is to be *globally adopted.* It must accommodate the entire range of meaningful distinctions which domestic and international property search engines must manage, and it must also fully serve agents having differing levels of business and technical experience.

It is important to emphasize that *a listing failed under local rules of one search engine may very well be permitted under differing local rules of another search engine.* Success of this reporting method therefore depends on accommodating local definitions of "success" and "failure" with reasonable grace and economy at least cost of time and energy to implement.

"Stock" error messages provided by all reporting schemes (including this one) are limited to only brief explanations which may be enigmatic to some agents if not carefully phrased. This reporting method optionally provides for a link to "extended instructions" allowing agents to simply click on the local ("stock") error message in their browser to call up more detailed information provided directly from the individual search engine. [NB: we seriously doubt anyone will ever implement this feature but it's available if we prove wrong about that.]

Method of Reporting

The basic reporting method relies solely on a very simple "stack" of three-digit "Global" reference which lie in the nominal range '000' to '999'. These codes contain all information necessary to report whether and why listings failed or succeeded by "pairing" codes to report results in this form:

1. a *result code* showing whether the listing passed or failed, and...
2. a *comment code* providing a detailed explanation.

An example of how code-pairs might appear in a simple XML feedback file:

```
<!-- reporting a successful listing: -->
<ListingId>000234</ListingId><!-- required -->
<ListingResult>050</ListingResult><!-- required -->
<ListingComment>100</ListingComment><!-- required -->
<ImageStatus>060</ImageStatus><!-- shows "no status reporting" -->

<!-- reporting a failed listing: -->
<ListingId>000235</ListingId><!-- required -->
<ListingResult>059</ListingResult><!-- required -->
<ListingComment>247</ListingComment><!-- required -->
```

"Result" codes are defined within the lowest range of numbers, '000' up to '099', which also includes optional "status" codes to report status of the main image for each listing. The image status code is a "singleton" describing both the result and the description for the main image upload. If the image upload hasn't yet been attempted, the image status code may be reporting only a temporary pending status.

Any code ending in zero denotes success and any non-zero code denotes an *actionable error*. All zero-ending codes are numerically '100' or less. All "Result" codes lie below '100'.

"Comment" codes begin with '100' (which is the only *unqualified success* comment code). All other comment codes above '100' are non-zero-ending and serve only as failure or warning messages – depending on whether they have been paired to a result code which is non-zero-ending (failure) or zero-ending (success), respectively.

Code '699' is a default catch-all code in the public range which is effectively a null response showing no specific explanation has been provided to describe why a listing was failed. The public range reserved for public codes ends at '699', and codes '700' or greater are in a private range reserved codes locally defined for whatever private, internal purpose may be desired.

Feed providers may download a single XML file from the search engine which reports coded results for listings they have submitted, allowing them to stream the numerical codes directly onto a local database to generate automated agent reporting in whatever form and language may be locally preferred.

Because listing failures are easily distinguished from successes based solely on the last digit of the result code, straightforward SQL queries make it possible to easily collect associated results reported by any particular search engine or on a consolidated basis for all reporting search engines, regardless of how each search engine individually defines “success” or “failure” owing to differing local data requirements.

Moreover, this proposed method provides the means for each search engine response to include a custom URL pointer to a FAQ list of error descriptions and recommendations for correction -- a list which may also be provided in multiple languages when necessary. Search engines using this method may thereby provide extended instructions to agents without limitations of length or language.

We have also recently provided for supplementary *advisory codes* which individual search engines may wish to define as messages unique to their own local operations. When these codes are provided by the search engine, their text descriptions are appended to comment messages at the download location. These advisory codes therefore do not replace comment codes, but merely attach further details provided within a separate tag. We'll illustrate and further discuss these methods below.

Scope of Error Reporting is “One-Each”

There can be only a single response for each listing reported, and only a single comment code can be paired to a result code – that is to say, all possible causes of a listing failure cannot be reported when more than one cause of failure exists.

It is therefore possible that a listing might be failed even after having been previously “fixed” by the agent in response to a previous failure report for the same listing. Experience to date is that a failed listing is usually reviewed by the agent to look for other possible errors. As agents become more keenly aware of what's required for a listing to succeed, they also tend to become more vigilant about checking for data errors or omissions which cause failures.

“Re-fails” have proven to occur only infrequently in everyday practice, and failure reporting lacking multiple causes of failure is therefore a minor factor in light of obvious benefits this method offers for simplicity of implementation to achieve core standardization.

No Human Language Dependency

Ordinary users never see underlying numerical codes, but only the associated human language meanings which use plain language to assure **comprehension by non-technical persons**.

Human language explanations must be carefully phrased to convey unbiased meanings, whether presented in the context of a “failure” or a “warning”. If the current meaning of a particular code cannot be reasonably rephrased to cover some new and distinct cause of failure truly different at some particular search engine location, then a new code can always be provided to provide that vital distinction. In many cases, careful rephrasing of an existing explanation can avoid inflation of the code set so as to cover a variety of slightly different implications of basically similar conditions.

How to Use

The current, stable list of codes for this project is available in an on-line introduction at:

http://www.syndafeed.com/listcodes.php

The XML code list with English and Spanish language descriptions is available for production use at:

http://www.syndafeed.com/listcodes.xml

It's probably worth reiterating that even though this complete list of *available* error reporting codes is rather long – perhaps even intimidating at first glance -- each individual search engine will typically *use* fewer than a couple of dozen of these, on average.

This list is pretty much a Chinese menu from which coders need only select those few code pairs locally relevant. All others they can entirely ignore. In some cases codes may also be available which describe same/similar conditions with greater or lesser levels of precision, allowing some choice of codes according to local preference or necessity.

The current list also provides “Alpha” and “Mnemonic” (English-eccentric) codes to provide lookup for their numerical “Global” code equivalents, and are likewise unique. These alternate codes serve the sole purpose of convenience to generate reporting status by indirect reference as an alternative to hard-coding. It's sometimes easier for coders writing or reviewing software to recall human “word-like” representations than to exactly memorize arbitrary numeric codes. Alpha and mnemonic lookup codes in the public range must be lower case, and those in the private range must be upper case.

Example XML Feedback Report

This XML feedback file shows examples of alternative feedback reporting methods:

```
<?xml version="1.0" encoding="UTF-8"  ?>
<Feedback>
      <!-- reporting a successful listing: -->
      <ListingId>X000234</ListingId><!-- required -->
      <ListingResult>050</ListingResult><!-- required -->
      <ListingComment>100</ListingComment><!-- required -->
      <ImageStatus>090</ImageStatus><!-- shows "image posted" -->
      <Url>http://www.search.com/listings.php?prov=xyz.com&listing=000234</Url>

      <!-- reporting a failed listing: -->
      <ListingId>X000235</ListingId><!-- required -->
      <ListingResult>059</ListingResult><!-- required -->
      <ListingComment>247</ListingComment><!-- required -->
      <Url>http://www.search.com/error_faq.php?code=247</Url>

      <!-- reporting a similar failed listing with an advisory code -->
      <!-- rather than a URL reference to an instruction page: -->
      <ListingId>X000299</ListingId><!-- required -->
      <ListingResult>159</ListingResult><!-- required -->
      <ListingComment>247</ListingComment><!-- required -->
      <Advisory>Z51</Advisory><!-- search engine proprietary reference –>
</Feedback>
```

For every *successful* listing, a URL may be provided for the location of the listing on the search engine. This is highly recommended so agents can direcly confirm the listing is actually “there”. For every *failed* listing, a URL may be (optionally) provided to an *Error FAQ* page on the search engine to display comprehensive local explanations of error results and how to correct them.

An alternative method for extended instructions is for the search engine to provide “singleton” advisory codes referencing proprietary text comments which can be automatically appended to the standard “vanilla” comment. Search engine engineers may create and maintain their own master list of proprietary supplemental codes, or transmit their list to be centrally maintained. These codes, either way, must be available for download by providers using those codes for local reporting.

Simplicity Is Vital

While globally standardized tag names in XML would greatly simplify implementation across all locations, it remains the primary goal of this project to promote adoption of a functional set of simple reporting codes -- deliverable now by any reasonable means, and deliverable in the future in whatever standardized format (if any) for delivery may evolve *after* functional requirements are agreed and committed to general usage. Again, we're agnostic about whether that ever happens.

No tag naming conventions are therefore imposed or proposed by this method. Search engine operators are therefore free to pick whatever tag names they prefer, albeit it with the minor disadvantage to retrievers of feedback data that they must code individual parser variants to handle differing tag names preferred by individual search engines. (Tag names provided in our examples may be adopted if they pose no conflict with naming conventions already in common local use, but we provide these for illustration only.)

Even though this method emphasizes the *content* rather than the *container,* simplicity remains the main goal. Search engine engineers who design their own feedback schema may therefore wish to implement the least challenging tag structure possible -- a courtesy which will be appreciated by smaller feed providers blessed with only limited time, manpower, and technical skills.

For Further Information or Comments

This project is a cooperative venture among competitors and will hopefully advance the industry toward the rational goal of having a useful, fully automated, off-the-shelf feedback system which can be implemented at any sending or receiving location to achieve maximum available benefit, with minimum calendar time and man-hours expended. (Known cases to-date required 4-6 man hours.)

Participants whose staff may include native speakers of languages other than English will hopefully be generous enough to contribute additional language translations to further enhance the value of this reporting system as a community asset.

Some details and implications of this method may be (and probably are) less than fully expressed in these notes. Please feel free to contact us with any questions, comments, or suggestions for changes:

Bud Hovell, Project Manager
bud@syndafeed.com +1 503 676 6710

Appendix: Simple Code Snippets

```
mysqli_query($conn,"create table if not exists listcodes(
                global_ref          varchar(3) binary    not null,  /* global reference */
                alpha_local         varchar(3) binary    not null,  /* alternate short alpha reference */
                mnemonic_local      varchar(20)  binary  not null,  /* alternate long alpha  reference */
                en                  text                 not null,  /* English text description */
                es                  text                 not null,  /* Spanish text description */
                unique              idx_listcodes1(global_ref),
                unique              idx_listcodes2(alpha_local),
                unique              idx_listcodes3(mnemonic_local)
                )") || die("Couldn't create table 'listcodes'!\n");
// end

function global_convertcodes() {      // simple lookup for converting alpha and/or
                                      // mnemonic codes to global codes

        global  $lookups, $rptstatus, $rptcomments, $rptimage, $conn;

        if ( !is_array($lookups) ) {    // build array just once to look up global reference codes

                $resultc =      mysqli_query($conn,"select alpha_local, global_ref
                                        from listcodes
                                        union
                                        select mnemonic_local, global_ref
                                        from listcodes");

                while ( $crow = mysqli_fetch_row($resultc) ) {
                        $alphacode  = $crow[0];
                        $globalcode = $crow[1];
                        $lookups["$alphacode"] = "$globalcode";    // add to our lookup array
                }
        }

        // look up global reference from short alpha or mnemonic reference:
        if ( preg_match("/[A-z]+$/",$rptstatus) )      { $rptstatus       = $lookups["$rptstatus"]; }
        if ( preg_match("/[A-z]+$/",$rptcomments)){ $rptcomments = $lookups["$rptcomments"]; }
        if ( preg_match("/[A-z]+$/",$rptimage) )       { $rptimage       = $lookups["$rptimage"]; }
}

// endof
```