**PennShare ArcGIS Online (AGOL) Tagging – Standard Operating Procedures (Data)**

**Asset Management Team Review Draft v.1**

**Updated: 15 November 2013**

**Jim Knudson (Baker)**

**All PennShare data tags must include the following:**

* PA
* Pennsylvania
* Agency in {“USGS”, “USFW”, “NRCS”, “PennDOT”, “DCNR”, “DEP”, “PGC”, “PFBC”, “DVRPC”, “NCPRPDC”, …}
* Agency Type in {“Federal”, “Commonwealth”, “Planning Partner”, “County”, “Other”}
* Core Data Type in {“Transportation”, “Hydrography”, “Soils”, “Land Records”, “Other”, …}
* Data Descriptor in {“Roads”, “Bridges”, “Streams”, “Soils”, …}
* System Name in {“RMS”, “BMS2”, “MPMS”, …} – *If Applicable, otherwise not included*
* Feature name in source system {RMSADMIN, RMSSEG, BMSBridge, …} – *If Applicable, otherwise not included*
* Descriptive information tags about the data

**Sample Tag:** “PA, Pennsylvania, PennDOT, Commonwealth, Transportation, Roads, RMS, RSMADMIN, State Highway Administrative Segments, State Highway, Centerline”

**Online Testing:**

Jim added and shared a small shapefile of Baker offices where GIS staff are located to PennShare. Added the following tags for the shapefile:

“[tag](http://pennshare.maps.arcgis.com/home/search.html?t=content&q=tags:tag), [commonwealth-baker-test](http://pennshare.maps.arcgis.com/home/search.html?t=content&q=tags:commonwealth-baker-test), [rocks](http://pennshare.maps.arcgis.com/home/search.html?t=content&q=tags:rocks), [rebar](http://pennshare.maps.arcgis.com/home/search.html?t=content&q=tags:rebar), [federal, usfw, hydrography, fishing streams, blab, pa dep, relax](http://pennshare.maps.arcgis.com/home/search.html?t=content&q=tags:federal-usfw-hydrography-fishing%20streams-blab-padep-relax)”

Will return the shapefile in a search when any of the following search criteria are used:

* Tag
* Commonwealth
* Baker
* Test
* Rocks
* Rebar
* Federal
* Usfw
* Federal-usfw
* Hydrography
* Fishing
* Stream
* Fishing streams
* Blabs
* PA
* Dep
* Rebar AND hydrography
* Federal AND blab
* Fish
* relax

Will ***NOT*** return the shapefile when the following search criteria are used:

* Rebar and hydrography
* Federal and blab
* Hydro
* Us
* Common
* Shin

**AGOL TAGS Questions/Answers/Discussion Topics:**

* Should the words “PA”, “Pennsylvania” and “PennDOT” be required on all Tags?
  + The team discussed if PennShare data is only going to be data from PennDOT or not.  The consensus is that the data can indeed be from outside agencies, or partners.  Therefore, the term “DOT” should not be a standard tag, instead, “PA” and “Pennsylvania” should and then also the agency whom provided the data.  (It should also be noted here, that a map created in PennShare is not the same.  That is, a map created by PennDOT personnel, for a PennDOT application, even though it includes data obtained from outside sources, should be tagged with “PennDOT”.
* Why won’t “hydro”, “US” or “Common”  return the shapefile when “fish” does?  Is it because of the stemming algorithim?
  + Correct, the stemming algorithm (see below) reduces Fishing to Fish, but does not stem Commonwealth to Common.  *If* it used a wildcard search instead of a stemming algorithm, “Common” would match Commonwealth, “hydro” would match hydrography, and US would match USFW.  I didn’t check to see if the advanced search would allow wildcard searches or not.
  + However, Commonwealths or Commonwealthed or Commonwealthing should provide a stem match to Commonwealth.
* Is there a maximum number of tags?
  + No, but there are a maximum number (undocumented) of characters for tags.  The question has been asked but not answered officially by ESRI as to how many characters.  One user/developer indicated 250 characters, while other users/developers indicated they thought it was 180 characters. There may be different tag character limits for different AGOL objects (see next question).  I can test for the data tags if you think it would be helpful to know the answer.  However, the answer could change in the future, which is why Esri has been reluctant to document the limits.
* Group tags – limited to 180 text characters?  **What is meant by “group” here?**
  + Almost everything in AGOL uses tags.  In addition to the data tags, there are group tags, webmap tags, and web application tags that can be filled out and are used in the AGOL search criteria.  Search will return any AGOL object in PennShare with a matching tag unless you qualify the search criteria or use advanced search.
* At this time, Tags and descriptions from existing GIS Metadata for uploaded GIS data are ignored by AGOL.  **What is meant by this?**
  + We have been talking about creating metadata files for all PennDOT GIS data and other GIS data uploaded to PennShare or provided as services.  If a GIS metadata file is uploaded as part of an Esri shapefile, AGOL ignores the metadata instead of applying tags and descriptions from the uploaded metadata and generating the AGOL tags for the data automatically.  Tags must therefore be entered manually in AGOL.

# ArcGIS Online Tags Best Practices

Tags are words or short phrases that describe your item. Tags are important since they determine if your group shows up in the search results when someone enters a keyword that matches one of your tags. Separate tags with commas. Federal land is considered one tag, while Federal, land is considered two tags.

[Sharing Web Applications]

Type tags. Alternatively, you can click the **Choose from your tags** link to open the list of tags *you've* used.

**TAG LIST Box Selection Limitation:** The tag list is a ***personal*** tag list. Each AGOL subscriber has their own set of tags stored for future use/selection. **There is currently no way to set up a master list of tags for all PennShare subscribers to use to help standardize this practice.**

**Tag Limitations:** Different information I nArcGIS.com help – nothing definitive

* 250(?) is the maximum number of characters supported.
* Group tags – limited to 180 text characters?
* 180 character limit in summary field in groups and item details?

<http://forums.arcgis.com/threads/91351-Character-limits-with-Item-Details-and-Tag-Parameters?highlight=tags>

**ESRI** uses a [stemming model](#_Stemming_Model_Algorithm) for keyword searches, but results are weighted, effectively, by popularity.

At this time, Tags and descriptions from existing GIS Metadata for uploaded GIS data are ignored by AGOL.

# Using search

**AGOL Search uses a Stemming algorithm. *Not a wildcard search.* For example, “culture” would *not* return “agriculture”, but would return items with tags such as “culture”, “cultures”, “cultured”.**

Can use **“AND”** **(uppercase!)** between tag strings to search for AGOL items with multiple tags. Must match both to be returned.

Use double quotation marks to surround terms with multiple words; for example, "map services" returns items with the term map services in a field, whereas map services returns items with either maps or services in a field.

[Accessing content](http://resources.arcgis.com/en/help/arcgisonline/010q/010q0000000n000000.htm)

Use the keyword search to find maps, layer, app, tools, files, and groups in the website. Enter keywords in the search box and choose the type of items you're looking for from the search drop-down menu, for example, **Search for Apps**. A list of relevant results appears. If you don't see what you want, refine your keywords and search again. For example, if you want to find a street map, you could enterstreet and choose **Search for Maps**. You would see a list of all the maps related to streets. If the list is too long, you can filter the results for a category of maps—web maps or map files. You could also search again for streets AND europe and you would then see only street maps for Europe.

By default, search results show web content only. To include ArcGIS files such as layer packages in the results, check the box next to**Show ArcGIS Desktop Content**. For more information, see [Finding content for ArcGIS desktop products](http://resources.arcgis.com/en/help/arcgisonline/010q/010q00000004000000.htm).

**NoteNote:**

Your organization may be configured to only search items within your organization. One way to tell is if all the items in your results are owned by members of your organization. If you aren't sure, contact the administrator of your organization.

If you are an administrator, you will see an option on the search results page to search outside the organization, even if you've configured the organization to only allow members to search within the organization.

## Narrowing your search results

You can use advanced keyword searches to narrow your results by specifying how you want to search for an item. Below are descriptions for the different ways you can do this.

### Fields

When performing a search for content or groups, you can either specify a field or use the default fields. For [items](http://resources.arcgis.com/en/help/arcgisonline/010q/010q0000000n000000.htm#ESRI_SECTION3_3AC886850B4D43B99C112FCC5179E720), the default fields are title, tags, snippet, description, accessinformation, spatialreference, type, and typekeywords. For [groups](http://resources.arcgis.com/en/help/arcgisonline/010q/010q0000000n000000.htm#ESRI_SECTION3_AA263D5D46F344A88D56920CE0B24C14), the default fields are id, title, description, snippet, tags, and owner. The best match is always returned. See the tables below for descriptions of these fields.

You can search a specific field by typing the field name followed by a colon and the term you are looking for (for a term with multiple words, use double quotes, such as "washoe county"). If you do not use a field indicator, the default fields are searched.

#### Item fields

You can refine your item searches by using specific fields in your search string. These fields include the following:

|  |  |
| --- | --- |
| **Field** | **Details** |
| id | ID of the item; for example, id:4e770315ad9049e7950b552aa1e40869 returns the item for that ID. |
| owner | Owner of the item; for example, owner:esri returns all content published by Esri. Field and value are case sensitive. |
| uploaded | Uploaded is the date uploaded; for example, uploaded: [0000001249084800000 TO 0000001249548000000] finds all items published between August 1, 2009, 12:00 a.m., to August 6, 2009, 8:40 a.m. |
| title | Item title; for example, title:"Southern California" returns items with Southern California in the title. |
| type | Type returns the type of item and is a predefined field. For a list of supported item types, see [What can you add to ArcGIS Online?](http://resources.arcgis.com/en/help/arcgisonline/010q/010q0000000m000000.htm) For example, type:map returns items with map as the type, such as map documents and map services. |
| description | Item description; for example, description:California finds all items with the term California in the description. |
| tags | The tag field; for example, tags:"San Francisco" returns items tagged with the term San Francisco. |
| snippet | Summary; for example, snippet:"natural resources" returns items with natural resources in the summary. |
| spatialreference | The spatial reference; for example, spatialreference:102100 returns items in the Web Mercator auxiliary sphere projection. |
| access | The access field; for example, access:public returns public items. This field is predefined, and the options are public, private, or shared. You will only see private or shared items that you have access to. |
| group | The ID of the group; for example, group:1652a410f59c4d8f98fb87b25e0a2669 returns items within the given group. |
| numratings | Number of ratings; for example, numratings:6 returns items with six ratings. |
| numcomments | Number of comments; for example, numcomments:[1 TO 3] returns items that have one to three comments. |
| avgrating | Average rating; for example, avgrating:3.5 returns items with 3.5 as the average rating. |

#### Group fields

You can filter your searches on groups by using specific fields in your search string. Only public groups or groups that you have access to will be searched. These fields include the following:

|  |  |
| --- | --- |
| **Group field** | **Details** |
| id | Group ID; for example, id:1db70a32f5f84ea9a88f5f460f22557b returns the group for that ID. |
| title | Group title; for example, title:redlands returns groups with Redlands in the title. |
| owner | Group owner; for example, owner:esri returns groups owned by Esri. |
| description | Description; for example, description:"street maps" returns groups with street maps in the description field. |
| snippet | Summary; for example, snippet:transportation returns groups with transportation in the group summary. |
| tags | The tags field; for example, tags:"bike lanes" returns groups tagged with the term bike lanes. |
| phone | Contact information; for example, phone:jsmith33@esri.com returns groups with jsmith33@esri.com as the contact. |
| created | Created is the date created; for example, created:0000001247085176000 returns groups created on July 8, 2009. |
| access | The access level of the group. Values are private and public. Private is the default; for example,access:private returns private groups. |
| isinvitationonly | The isinvitationonly field returns groups that require an invitation to join. For example,isinvitationonly:false returns groups that do not require an invitation to join. This field is predefined with the options true or false. |

### Range searches

Range searches allow you to match on field values between the lower and upper bounds. Range queries can be inclusive or exclusive of the upper and lower bounds. Inclusive range queries are denoted by square brackets. Exclusive range queries are denoted by curly brackets.

For example, if you wanted to find all items uploaded between December 1, 2009, and December 9, 2009, use uploaded:[0000001259692864000 TO 0000001260384065000].

The uploaded field contains the date and time an item is uploaded in UNIX time. UNIX time is defined as the number of seconds that have elapsed since midnight January 1, 1970. The website stores time in milliseconds, so you need to add three zeros to the end of the UNIX time. Additionally, you need to pad six zeros on the front of the number. This is because the number is stored as a string in the database.

Range searches are not reserved for date fields. You could also use range queries with nondate fields, for example, owner:[arcgis\_explorer TO esri]. This will find all items from the owners between arcgis\_explorer and esri, including arcgis\_explorer and esri.

### Boosting a term

Boosting allows you to control the relevance of an item by boosting its term. To boost a term, use the caret symbol (^) with a boost factor (a number) at the end of the term you are searching. The higher the boost factor, the more relevant the term will be. For example, if you are searching for "recent fires" and want "fires" to be more relevant, create the expression recent fires^5.

### Boolean operators

Boolean operators allow terms to be combined through logic operators. The website supports AND, plus sign (+), OR, NOT, and minus sign (-) as Boolean operators. Boolean operators must be ALL CAPS.

|  |  |
| --- | --- |
| **Boolean operator** | **Details** |
| AND | The AND operator is the default conjunction. This means that if there is no Boolean operator between two terms, the AND operator is used. The AND operator performs matching where both terms exist in either the given field or the default fields. This is equivalent to an intersection using sets. |
| OR | The OR operator links two terms and finds a match if either of the terms exists. This is equivalent to a union using sets.  To search for an item that contains either the term "recent fires" or just "fires," use the query"recent fires" OR fires. |
| + | The plus sign, or the required operator, requires that the term after the symbol exist somewhere in the given field or the default fields.  To search for items that must contain "fires" and may contain "recent," use the query recent +fires. |
| NOT | The NOT operator excludes items that contain the term after NOT. This is equivalent to a difference using sets. To search for documents that contain "California" but not "imagery," use the queryCalifornia NOT Imagery. The NOT operator cannot be used with just one term. |
| - | The minus sign, or the prohibit operator, excludes items that contain the term after the symbol.  To search for documents that contain "California" but not "imagery," use the query California -Imagery. |

### Grouping

You can create subqueries using parentheses to group clauses. This can be very useful if you want to control the Boolean logic for a query.

To search for either "California" or "recent" and "fires," create the expression (California OR recent) AND fires.

### Field grouping

You can group multiple clauses to a single field using parentheses.

To search for a title that contains both the phrase "population change" and the word "recent," use the query title:(+"population change" +recent).

## Search tips

* When doing a field search, use a colon (:) after the field name, for example, owner:esri.
* Use double quotation marks to surround terms with multiple words; for example, "map services" returns items with the term map services in a field, whereas map services returns items with either maps or services in a field.
* You can build a search string by linking fields together in your search string with the AND operator, for example, owner:esri AND tags:streets.
* Use uppercase for search operators: AND, OR, and so forth.
* Sort your results with the available filters for most popular, highest rated, added today, and so forth.
* In addition to searching for content through keywords, you can also use the gallery to browse featured maps, web applications, and mobile applications.
* If you want to search for maps and data layers with a specific extent, use the ArcGIS.com map viewer. Open a new or existing web map, set the extent you want, and use the **Add** button to search for layers. For more information, see [Searching for layers](http://resources.arcgis.com/en/help/arcgisonline/010q/010q00000089000000.htm).

# Stemming Model Algorithm

In [linguistic morphology](http://en.wikipedia.org/wiki/Linguistic_morphology) and [information retrieval](http://en.wikipedia.org/wiki/Information_retrieval), **stemming** is the process for reducing inflected (or sometimes derived) words to their [stem](http://en.wikipedia.org/wiki/Word_stem), base or [root](http://en.wikipedia.org/wiki/Root_(linguistics)) form—generally a written word form. The stem need not be identical to the [morphological root](http://en.wikipedia.org/wiki/Morphological_root) of the word; it is usually sufficient that related words map to the same stem, even if this stem is not in itself a valid root. [Algorithms](http://en.wikipedia.org/wiki/Algorithm) for stemming have been studied in [computer science](http://en.wikipedia.org/wiki/Computer_science) since the 1960s. Many [search engines](http://en.wikipedia.org/wiki/Search_engine) treat words with the same stem as[synonyms](http://en.wikipedia.org/wiki/Synonym) as a kind of [query expansion](http://en.wikipedia.org/wiki/Query_expansion), a process called **conflation**.

Stemming programs are commonly referred to as **stemming algorithms** or **stemmers**.

**Contents**

* [1 Examples](http://en.wikipedia.org/wiki/Stemming#Examples)
* [2 History](http://en.wikipedia.org/wiki/Stemming#History)
* [3 Algorithms](http://en.wikipedia.org/wiki/Stemming#Algorithms)
  + [3.1 Lookup algorithms](http://en.wikipedia.org/wiki/Stemming#Lookup_algorithms)
    - [3.1.1 The production technique](http://en.wikipedia.org/wiki/Stemming#The_production_technique)
  + [3.2 Suffix-stripping algorithms](http://en.wikipedia.org/wiki/Stemming#Suffix-stripping_algorithms)
    - [3.2.1 Additional algorithm criteria](http://en.wikipedia.org/wiki/Stemming#Additional_algorithm_criteria)
  + [3.3 Lemmatisation algorithms](http://en.wikipedia.org/wiki/Stemming#Lemmatisation_algorithms)
  + [3.4 Stochastic algorithms](http://en.wikipedia.org/wiki/Stemming#Stochastic_algorithms)
  + [3.5 *n*-gram analysis](http://en.wikipedia.org/wiki/Stemming#n-gram_analysis)
  + [3.6 Hybrid approaches](http://en.wikipedia.org/wiki/Stemming#Hybrid_approaches)
  + [3.7 Affix stemmers](http://en.wikipedia.org/wiki/Stemming#Affix_stemmers)
  + [3.8 Matching algorithms](http://en.wikipedia.org/wiki/Stemming#Matching_algorithms)
* [4 Language challenges](http://en.wikipedia.org/wiki/Stemming#Language_challenges)
  + [4.1 Multilingual stemming](http://en.wikipedia.org/wiki/Stemming#Multilingual_stemming)
* [5 Error metrics](http://en.wikipedia.org/wiki/Stemming#Error_metrics)
* [6 Applications](http://en.wikipedia.org/wiki/Stemming#Applications)
  + [6.1 Information retrieval](http://en.wikipedia.org/wiki/Stemming#Information_retrieval)
  + [6.2 Domain Analysis](http://en.wikipedia.org/wiki/Stemming#Domain_Analysis)
  + [6.3 Use in commercial products](http://en.wikipedia.org/wiki/Stemming#Use_in_commercial_products)
* [7 See also](http://en.wikipedia.org/wiki/Stemming#See_also)
* [8 References](http://en.wikipedia.org/wiki/Stemming#References)
* [9 Further reading](http://en.wikipedia.org/wiki/Stemming#Further_reading)
* [10 External links](http://en.wikipedia.org/wiki/Stemming#External_links)

Examples[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=1" \o "Edit section: Examples)]

A stemmer for [English](http://en.wikipedia.org/wiki/English_language), for example, should identify the [string](http://en.wikipedia.org/wiki/String_literal) "cats" (and possibly "catlike", "catty" etc.) as based on the root "cat", and "stemmer", "stemming", "stemmed" as based on "stem". A stemming algorithm reduces the words "fishing", "fished", and "fisher" to the root word, "fish". On the other hand, "argue", "argued", "argues", "arguing", and "argus" reduce to the stem "argu" (illustrating the case where the stem is not itself a word or root) but "argument" and "arguments" reduce to the stem "argument".

History[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=2" \o "Edit section: History)]

The first published stemmer was written by [Julie Beth Lovins](http://en.wikipedia.org/wiki/Julie_Beth_Lovins) in 1968.[[1]](http://en.wikipedia.org/wiki/Stemming#cite_note-1) This paper was remarkable for its early date and had great influence on later work in this area.

A later stemmer was written by [Martin Porter](http://en.wikipedia.org/wiki/Martin_Porter) and was published in the July 1980 issue of the journal *Program*. This stemmer was very widely used and became the de facto standard algorithm used for English stemming. Dr. Porter received the [Tony Kent Strix award](http://en.wikipedia.org/wiki/Tony_Kent_Strix_award) in 2000 for his work on stemming and information retrieval.

Many implementations of the Porter stemming algorithm were written and freely distributed; however, many of these implementations contained subtle flaws. As a result, these stemmers did not match their potential. To eliminate this source of error, Martin Porter released an official [free-software implementation](http://tartarus.org/~martin/PorterStemmer/) of the algorithm around the year 2000. He extended this work over the next few years by building [Snowball](http://en.wikipedia.org/wiki/Snowball_programming_language), a framework for writing stemming algorithms, and implemented an improved English stemmer together with stemmers for several other languages.

Algorithms[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=3" \o "Edit section: Algorithms)]

There are several types of stemming algorithms which differ in respect to performance and accuracy and how certain stemming obstacles are overcome.

**Lookup algorithms**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=4" \o "Edit section: Lookup algorithms)]

A simple stemmer looks up the inflected form in a [lookup table](http://en.wikipedia.org/wiki/Lookup_table). The advantages of this approach is that it is simple, fast, and easily handles exceptions. The disadvantages are that all inflected forms must be explicitly listed in the table: new or unfamiliar words are not handled, even if they are perfectly regular (e.g. iPads ~ iPad), and the table may be large. For languages with simple morphology, like English, table sizes are modest, but highly inflected languages like Turkish may have hundreds of potential inflected forms for each root.

A lookup approach may use preliminary part-of-speech tagging to avoid overstemming.[[2]](http://en.wikipedia.org/wiki/Stemming#cite_note-2)

**The production technique**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=5" \o "Edit section: The production technique)]

The lookup table used by a stemmer is generally produced semi-automatically. For example, if the word is "run", then the inverted algorithm might automatically generate the forms "running", "runs", "runned", and "runly". The last two forms are valid constructions, but they are unlikely to appear in a normal English-language text.

**Suffix-stripping algorithms**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=6" \o "Edit section: Suffix-stripping algorithms)]

Suffix stripping algorithms do not rely on a lookup table that consists of inflected forms and root form relations. Instead, a typically smaller list of "rules" is stored which provides a path for the algorithm, given an input word form, to find its root form. Some examples of the rules include:

* if the word ends in 'ed', remove the 'ed'
* if the word ends in 'ing', remove the 'ing'
* if the word ends in 'ly', remove the 'ly'

Suffix stripping approaches enjoy the benefit of being much simpler to maintain than brute force algorithms, assuming the maintainer is sufficiently knowledgeable in the challenges of linguistics and morphology and encoding suffix stripping rules. Suffix stripping algorithms are sometimes regarded as crude given the poor performance when dealing with exceptional relations (like 'ran' and 'run'). The solutions produced by suffix stripping algorithms are limited to those [lexical categories](http://en.wikipedia.org/wiki/Lexical_category) which have well known suffixes with few exceptions. This, however, is a problem, as not all parts of speech have such a well formulated set of rules. Lemmatisation attempts to improve upon this challenge.

Prefix stripping may also be implemented. Of course, not all languages use prefixing or suffixing.

**Additional algorithm criteria**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=7" \o "Edit section: Additional algorithm criteria)]

Suffix stripping algorithms may differ in results for a variety of reasons. One such reason is whether the algorithm constrains whether the output word must be a real word in the given language. Some approaches do not require the word to actually exist in the language lexicon (the set of all words in the language). Alternatively, some suffix stripping approaches maintain a database (a large list) of all known morphological word roots that exist as real words. These approaches check the list for the existence of the term prior to making a decision. Typically, if the term does not exist, alternate action is taken. This alternate action may involve several other criteria. The non-existence of an output term may serve to cause the algorithm to try alternate suffix stripping rules.

It can be the case that two or more suffix stripping rules apply to the same input term, which creates an ambiguity as to which rule to apply. The algorithm may assign (by human hand or stochastically) a priority to one rule or another. Or the algorithm may reject one rule application because it results in a non-existent term whereas the other overlapping rule does not. For example, given the English term *friendlies*, the algorithm may identify the *ies* suffix and apply the appropriate rule and achieve the result of*friendl*. *friendl* is likely not found in the lexicon, and therefore the rule is rejected.

One improvement upon basic suffix stripping is the use of suffix substitution. Similar to a stripping rule, a substitution rule replaces a suffix with an alternate suffix. For example, there could exist a rule that replaces *ies* with *y*. How this affects the algorithm varies on the algorithm's design. To illustrate, the algorithm may identify that both the *ies* suffix stripping rule as well as the suffix substitution rule apply. Since the stripping rule results in a non-existent term in the lexicon, but the substitution rule does not, the substitution rule is applied instead. In this example, *friendlies* becomes *friendly* instead of *friendl*.

Diving further into the details, a common technique is to apply rules in a cyclical fashion (recursively, as computer scientists would say). After applying the suffix substitution rule in this example scenario, a second pass is made to identify matching rules on the term *friendly*, where the *ly* stripping rule is likely identified and accepted. In summary, *friendlies* becomes (via substitution)*friendly* which becomes (via stripping) *friend*.

This example also helps illustrate the difference between a rule-based approach and a brute force approach. In a brute force approach, the algorithm would search for *friendlies* in the set of hundreds of thousands of inflected word forms and ideally find the corresponding root form *friend*. In the rule-based approach, the three rules mentioned above would be applied in succession to converge on the same solution. Chances are that the rule-based approach would be faster.

**Lemmatisation algorithms**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=8" \o "Edit section: Lemmatisation algorithms)]

A more complex approach to the problem of determining a stem of a word is [lemmatisation](http://en.wikipedia.org/wiki/Lemmatisation" \o "Lemmatisation). This process involves first determining the [part of speech](http://en.wikipedia.org/wiki/Part_of_speech) of a word, and applying different normalization rules for each part of speech. The part of speech is first detected prior to attempting to find the root since for some languages, the stemming rules change depending on a word's part of speech.

This approach is highly conditional upon obtaining the correct lexical category (part of speech). While there is overlap between the normalization rules for certain categories, identifying the wrong category or being unable to produce the right category limits the added benefit of this approach over suffix stripping algorithms. The basic idea is that, if the stemmer is able to grasp more information about the word being stemmed, then it can apply more accurate normalization rules (which unlike suffix stripping rules cal also modify the stem.).

**Stochastic algorithms**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=9" \o "Edit section: Stochastic algorithms)]

[Stochastic](http://en.wikipedia.org/wiki/Stochastic) algorithms involve using probability to identify the root form of a word. Stochastic algorithms are trained (they "learn") on a table of root form to inflected form relations to develop a probabilistic model. This model is typically expressed in the form of complex linguistic rules, similar in nature to those in suffix stripping or lemmatisation. Stemming is performed by inputting an inflected form to the trained model and having the model produce the root form according to its internal ruleset, which again is similar to suffix stripping and lemmatisation, except that the decisions involved in applying the most appropriate rule, or whether or not to stem the word and just return the same word, or whether to apply two different rules sequentially, are applied on the grounds that the output word will have the highest probability of being correct (which is to say, the smallest probability of being incorrect, which is how it is typically measured).

Some lemmatisation algorithms are stochastic in that, given a word which may belong to multiple parts of speech, a probability is assigned to each possible part. This may take into account the surrounding words, called the context, or not. Context-free grammars do not take into account any additional information. In either case, after assigning the probabilities to each possible part of speech, the most likely part of speech is chosen, and from there the appropriate normalization rules are applied to the input word to produce the normalized (root) form.

***n*-gram analysis**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=10)]

Some stemming techniques use the [n-gram](http://en.wikipedia.org/wiki/N-gram) context of a word to choose the correct stem for a word.[*[citation needed](http://en.wikipedia.org/wiki/Wikipedia:Citation_needed" \o "Wikipedia:Citation needed)*]

**Hybrid approaches**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=11" \o "Edit section: Hybrid approaches)]

Hybrid approaches use two or more of the approaches described above in unison. A simple example is a [suffix tree](http://en.wikipedia.org/w/index.php?title=Probabilistic_suffix_tree&action=edit&redlink=1) algorithm which first consults a lookup table using brute force. However, instead of trying to store the entire set of relations between words in a given language, the lookup table is kept small and is only used to store a minute amount of "frequent exceptions" like "ran => run". If the word is not in the exception list, apply suffix stripping or lemmatisation and output the result.

**Affix stemmers**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=12" \o "Edit section: Affix stemmers)]

In [linguistics](http://en.wikipedia.org/wiki/Linguistics), the term [affix](http://en.wikipedia.org/wiki/Affix) refers to either a [prefix](http://en.wikipedia.org/wiki/Prefix) or a [suffix](http://en.wikipedia.org/wiki/Suffix). In addition to dealing with suffixes, several approaches also attempt to remove common prefixes. For example, given the word *indefinitely*, identify that the leading "in" is a prefix that can be removed. Many of the same approaches mentioned earlier apply, but go by the name **affix stripping**. A study of affix stemming for several European languages can be found here.[[3]](http://en.wikipedia.org/wiki/Stemming#cite_note-3)

**Matching algorithms**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=13" \o "Edit section: Matching algorithms)]

Such algorithms use a stem database (for example a set of documents that contain stem words). These stems, as mentioned above, are not necessarily valid words themselves (but rather common sub-strings, as the "brows" in "browse" and in "browsing"). In order to stem a word the algorithm tries to match it with stems from the database, applying various constraints, such as on the relative length of the candidate stem within the word (so that, for example, the short prefix "be", which is the stem of such words as "be", "been" and "being", would not be considered as the stem of the word "beside").

Language challenges[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=14" \o "Edit section: Language challenges)]

While much of the early academic work in this area was focused on the English language (with significant use of the Porter Stemmer algorithm), many other languages have been investigated.[[4]](http://en.wikipedia.org/wiki/Stemming#cite_note-4)[[5]](http://en.wikipedia.org/wiki/Stemming#cite_note-5)[[6]](http://en.wikipedia.org/wiki/Stemming#cite_note-6)[[7]](http://en.wikipedia.org/wiki/Stemming#cite_note-7)[[8]](http://en.wikipedia.org/wiki/Stemming#cite_note-8)

Hebrew and Arabic are still considered difficult research languages for stemming. English stemmers are fairly trivial (with only occasional problems, such as "dries" being the third-person singular present form of the verb "dry", "axes" being the plural of "axe" as well as "axis"); but stemmers become harder to design as the morphology, orthography, and character encoding of the target language becomes more complex. For example, an Italian stemmer is more complex than an English one (because of a greater number of verb inflections), a Russian one is more complex (more noun [declensions](http://en.wikipedia.org/wiki/Declension)), a Hebrew one is even more complex (due to non-catenative morphology, a writing system without vowels, and the requirement of prefix stripping: Hebrew stems can be two, three or four characters, but not more), and so on.

**Multilingual stemming**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=15" \o "Edit section: Multilingual stemming)]

Multilingual stemming applies morphological rules of two or more languages simultaneously instead of rules for only a single language when interpreting a search query. Commercial systems using multilingual stemming exist[*[citation needed](http://en.wikipedia.org/wiki/Wikipedia:Citation_needed" \o "Wikipedia:Citation needed)*].

Error metrics[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=16" \o "Edit section: Error metrics)]

There are two error measurements in stemming algorithms, **overstemming** and **understemming**. Overstemming is an error where two separate inflected words are stemmed to the same root, but should not have been—a [false positive](http://en.wikipedia.org/wiki/False_positive). Understemming is an error where two separate inflected words should be stemmed to the same root, but are not—a [false negative](http://en.wikipedia.org/wiki/False_negative). Stemming algorithms attempt to minimize each type of error, although reducing one type can lead to increasing the other.

For example, the widely-used Porter stemmer stems "universal", "university", and "universe" to "univers". This is a case of overstemming: though these three words are etymologically related, their modern meanings are in widely different domains, so treating them as synonyms in a search engine will likely reduce the relevance of the search results.

An example of understemming in the Porter stemmer is "alumnus" → "alumnu", "alumni" → "alumni", "alumna"/"alumnae" → "alumna". This English word keeps Latin morphology, and so these near-synonyms are not conflated.

Applications[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=17" \o "Edit section: Applications)]

Stemming is used as an approximate method for grouping words with a similar basic meaning together. For example, a text mentioning "daffodils" is probably closely related to a text mentioning "daffodil" (without the s). But in some cases, words with the same morphological stem have [idiomatic](http://en.wikipedia.org/wiki/Idiom) meanings which are not closely related: a user searching for "marketing" will not be satisfied by most documents mentioning "markets" but not "marketing".

**Information retrieval**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=18" \o "Edit section: Information retrieval)]

Stemmers are common elements in [query systems](http://en.wikipedia.org/wiki/Information_Retrieval) such as [Web](http://en.wikipedia.org/wiki/World_Wide_Web) [search engines](http://en.wikipedia.org/wiki/Search_engine). The effectiveness of stemming for English query systems were soon found to be rather limited, however, and this has led early [information retrieval](http://en.wikipedia.org/wiki/Information_retrieval) researchers to deem stemming irrelevant in general.[[9]](http://en.wikipedia.org/wiki/Stemming#cite_note-9) An alternative approach, based on searching for [n-grams](http://en.wikipedia.org/wiki/N-gram) rather than stems, may be used instead. Also, recent research has shown greater benefits for retrieval in other languages.[[10]](http://en.wikipedia.org/wiki/Stemming#cite_note-10)[[11]](http://en.wikipedia.org/wiki/Stemming#cite_note-11)

**Domain Analysis**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=19" \o "Edit section: Domain Analysis)]

Stemming is used to determine domain vocabularies in [domain analysis](http://en.wikipedia.org/wiki/Domain_analysis). [[12]](http://en.wikipedia.org/wiki/Stemming#cite_note-12)

**Use in commercial products**[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=20" \o "Edit section: Use in commercial products)]

Many commercial companies have been using stemming since at least the 1980s and have produced algorithmic and lexical stemmers in many languages.[[13]](http://en.wikipedia.org/wiki/Stemming#cite_note-13)[[14]](http://en.wikipedia.org/wiki/Stemming#cite_note-14)

The Snowball stemmers have been compared with commercial lexical stemmers with varying results.[[15]](http://en.wikipedia.org/wiki/Stemming#cite_note-15)[[16]](http://en.wikipedia.org/wiki/Stemming#cite_note-16)

[Google search](http://en.wikipedia.org/wiki/Google_search) adopted word stemming in 2003.[[17]](http://en.wikipedia.org/wiki/Stemming#cite_note-17) Previously a search for "fish" would not have returned "fishing". Other software search algorithms vary in their use of word stemming. Programs that simply search for substrings obviously will find "fish" in "fishing" but when searching for "fishes" will not find occurrences of the word "fish".

See also[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=21" \o "Edit section: See also)]

* [Root (linguistics)](http://en.wikipedia.org/wiki/Root_(linguistics)) - linguistic definition of the term "root"
* [Stem (linguistics)](http://en.wikipedia.org/wiki/Stem_(linguistics)) - linguistic definition of the term "stem"
* [Morphology (linguistics)](http://en.wikipedia.org/wiki/Morphology_(linguistics))
* [Lemma (morphology)](http://en.wikipedia.org/wiki/Lemma_(morphology)) - linguistic definition
* [Lemmatization](http://en.wikipedia.org/wiki/Lemmatization)
* [Lexeme](http://en.wikipedia.org/wiki/Lexeme)
* [Inflection](http://en.wikipedia.org/wiki/Inflection)
* [Derivation](http://en.wikipedia.org/wiki/Derivation_(linguistics)) - stemming is a form of reverse derivation
* [Natural language processing](http://en.wikipedia.org/wiki/Natural_language_processing) - stemming is generally regarded as a form of NLP
* [Text mining](http://en.wikipedia.org/wiki/Text_mining) - stemming algorithms play a major role in commercial NLP software
* [Computational linguistics](http://en.wikipedia.org/wiki/Computational_linguistics)

References[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=22" \o "Edit section: References)]

* 1. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-1) Lovins, Julie Beth (1968). "Development of a Stemming Algorithm". *Mechanical Translation and Computational Linguistics* **11**: 22–31.
  2. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-2) Yatsko, V. A.; [*Y-stemmer*](http://yatsko.zohosites.com/y-stemmer.html)
  3. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-3) Jongejan, B.; and Dalianis, H.; *Automatic Training of Lemmatization Rules that Handle Morphological Changes in pre-, in- and Suffixes Alike*, in the *Proceeding of the ACL-2009, Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Singapore, August 2–7, 2009*, pp. 145-153 [[1]](http://www.aclweb.org/anthology/P/P09/P09-1017.pdf)
  4. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-4) Dolamic, Ljiljana; and Savoy, Jacques; [*Stemming Approaches for East European Languages (CLEF 2007)*](http://clef.isti.cnr.it/2007/working_notes/DolamicCLEF2007.pdf)
  5. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-5) Savoy, Jacques; [*Light Stemming Approaches for the French, Portuguese, German and Hungarian Languages*](http://portal.acm.org/citation.cfm?doid=1141277.1141523), ACM Symposium on Applied Computing, SAC 2006, [ISBN 1-59593-108-2](http://en.wikipedia.org/wiki/Special:BookSources/1595931082)
  6. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-6) Popovič, Mirko; and Willett, Peter (1992); [*The Effectiveness of Stemming for Natural-Language Access to Slovene Textual Data*](http://onlinelibrary.wiley.com/doi/10.1002/%28SICI%291097-4571%28199206%2943:5%3C384::AID-ASI6%3E3.0.CO;2-L/abstract), Journal of the [American Society for Information Science](http://en.wikipedia.org/wiki/American_Society_for_Information_Science), Volume 43, Issue 5 (June), pp. 384–390
  7. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-7) [*Stemming in Hungarian at CLEF 2005*](http://staff.science.uva.nl/~mdr/Publications/Files/clef2005-proc-adhoc.pdf)
  8. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-8) Viera, A. F. G. & Virgil, J. (2007); [*Uma revisão dos algoritmos de radicalização em língua portuguesa*](http://informationr.net/ir/12-3/paper315.html), Information Research, 12(3), paper 315
  9. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-9) Baeza-Yates, Ricardo; and Ribeiro-Neto, Berthier (1999);*Modern Information Retrieval*, ACM Press/Addison Wesley
  10. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-10) Kamps, Jaap; Monz, Christof; de Rijke, Maarten; and Sigurbjörnsson, Börkur (2004); *Language-Dependent and Language-Independent Approaches to Cross-Lingual Text Retrieval*, in Peters, C.; Gonzalo, J.; Braschler, M.; and Kluck, M. (eds.); *Comparative Evaluation of Multilingual Information Access Systems*, Springer Verlag, pp. 152–165
  11. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-11) Airio, Eija (2006); *Word Normalization and Decompounding in Mono- and Bilingual IR*, Information Retrieval **9**:249–271
  12. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-12) Frakes, W.; Prieto-Diaz, R.; & Fox, C. (1998); *DARE: Domain Analysis and Reuse Environment*, Annals of Software Engineering (5), pp. 125-141
  13. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-13) [*Language Extension Packs*](http://www.dtsearch.co.uk/language.htm), dtSearch
  14. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-14) [*Building Multilingual Solutions by using Sharepoint Products and Technologies*](http://technet2.microsoft.com/Office/en-us/library/87065c9d-d39d-479d-909b-02160ec6d7791033.mspx?mfr=true), Microsoft Technet
  15. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-15) [CLEF 2003: Stephen Tomlinson compared the Snowball stemmers with the Hummingbird lexical stemming (lemmatization) system](http://clef.isti.cnr.it/2003/WN_web/19.pdf)
  16. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-16) [CLEF 2004: Stephen Tomlinson "Finnish, Portuguese and Russian Retrieval with Hummingbird SearchServer"](http://clef.isti.cnr.it/2004/working_notes/WorkingNotes2004/21.pdf)
  17. [**Jump up^**](http://en.wikipedia.org/wiki/Stemming#cite_ref-17) [*The Essentials of Google Search*](http://www.google.com/support/bin/static.py?page=searchguides.html&ctx=basics#stemming), Web Search Help Center, [Google Inc.](http://en.wikipedia.org/wiki/Google)

Further reading[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=23" \o "Edit section: Further reading)]

* Dawson, J. L. (1974); *Suffix Removal for Word Conflation*, Bulletin of the Association for Literary and Linguistic Computing, 2(3): 33–46
* Frakes, W. B. (1984); *Term Conflation for Information Retrieval*, Cambridge University Press
* Frakes, W. B. & Fox, C. J. (2003); *Strength and Similarity of Affix Removal Stemming Algorithms*, SIGIR Forum, 37: 26–30
* Frakes, W. B. (1992); *Stemming algorithms, Information retrieval: data structures and algorithms*, Upper Saddle River, NJ: Prentice-Hall, Inc.
* Hafer, M. A. & Weiss, S. F. (1974); *Word segmentation by letter successor varieties*, Information Processing & Management 10 (11/12), 371–386
* Harman, D. (1991); *How Effective is Suffixing?*, Journal of the American Society for Information Science 42 (1), 7–15
* Hull, D. A. (1996); *Stemming Algorithms – A Case Study for Detailed Evaluation*, JASIS, 47(1): 70–84
* Hull, D. A. & Grefenstette, G. (1996); *A Detailed Analysis of English Stemming Algorithms*, Xerox Technical Report
* Kraaij, W. & Pohlmann, R. (1996); *Viewing Stemming as Recall Enhancement*, in Frei, H.-P.; Harman, D.; Schauble, P.; and Wilkinson, R. (eds.);*Proceedings of the 17th ACM SIGIR conference held at Zurich, August 18–22*, pp. 40–48
* Krovetz, R. (1993); *Viewing Morphology as an Inference Process*, in *Proceedings of ACM-SIGIR93*, pp. 191–203
* Lennon, M.; Pierce, D. S.; Tarry, B. D.; & Willett, P. (1981); *An Evaluation of some Conflation Algorithms for Information Retrieval*, Journal of Information Science, 3: 177–183
* Lovins, J. (1971); [*Error Evaluation for Stemming Algorithms as Clustering Algorithms*](http://www.eric.ed.gov/sitemap/html_0900000b800c571a.html), JASIS, 22: 28–40
* Lovins, J. B. (1968); *Development of a Stemming Algorithm*, Mechanical Translation and Computational Linguistics, 11, 22—31
* Jenkins, Marie-Claire; and Smith, Dan (2005); [*Conservative Stemming for Search and Indexing*](http://www.uea.ac.uk/polopoly_fs/1.85493!stemmer25feb.pdf)
* Paice, C. D. (1990); [*Another Stemmer*](http://www.comp.lancs.ac.uk/computing/research/stemming/paice/article.htm), SIGIR Forum, 24: 56–61
* Paice, C. D. (1996) [*Method for Evaluation of Stemming Algorithms based on Error Counting*](http://www3.interscience.wiley.com/cgi-bin/abstract/57804/ABSTRACT), JASIS, 47(8): 632–649
* Popovič, Mirko; and Willett, Peter (1992); [*The Effectiveness of Stemming for Natural-Language Access to Slovene Textual Data*](http://onlinelibrary.wiley.com/doi/10.1002/%28SICI%291097-4571%28199206%2943:5%3C384::AID-ASI6%3E3.0.CO;2-L/abstract), Journal of the[American Society for Information Science](http://en.wikipedia.org/wiki/American_Society_for_Information_Science), Volume 43, Issue 5 (June), pp. 384–390
* Porter, Martin F. (1980); [*An Algorithm for Suffix Stripping*](http://telemat.det.unifi.it/book/2001/wchange/download/stem_porter.html), Program, 14(3): 130–137
* Savoy, J. (1993); [*Stemming of French Words Based on Grammatical Categories*](http://www3.interscience.wiley.com/cgi-bin/abstract/10049824/ABSTRACT?CRETRY=1&SRETRY=0) Journal of the American Society for Information Science, 44(1), 1–9
* Ulmschneider, John E.; & Doszkocs, Tamas (1983); [*A Practical Stemming Algorithm for Online Search Assistance*](http://www.eric.ed.gov/sitemap/html_0900000b8007ea83.html), Online Review, 7(4), 301–318
* Xu, J.; & Croft, W. B. (1998); [*Corpus-Based Stemming Using Coocurrence of Word Variants*](http://portal.acm.org/citation.cfm?doid=267954.267957), ACM Transactions on Information Systems, 16(1), 61–81

External links[[edit](http://en.wikipedia.org/w/index.php?title=Stemming&action=edit&section=24" \o "Edit section: External links)]

* [SMILE Stemmer](http://smile-stemmer.appspot.com/) - free online service, includes Porter and Paice/Husk' Lancaster stemmers (Java API)
* [Themis](http://code.google.com/p/ir-themis/) - open source IR framework, includes Porter stemmer implementation (PostgreSQL, Java API)
* [Snowball](http://snowball.tartarus.org/) - free stemming algorithms for many languages, includes source code, including stemmers for five romance languages
* [Snowball on C#](http://www.iveonik.com/blog/2011/08/snowball-stemmers-on-csharp-free-download/) - port of Snowball stemmers for C# (14 languages)
* [[2]](http://snowball.tartarus.org/wrappers/guide.html) - Python bindings to Snowball API
* [Ruby-Stemmer](http://locknet.ro/archive/2009-10-29-ann-ruby-stemmer.html) - Ruby extension to Snowball API
* [PECL](http://pecl.php.net/package/stem/) - PHP extension to the Snowball API
* [Oleander Porter's algorithm](http://www.oleandersolutions.com/stemming.html) - stemming library in C++ released under BSD
* [Unofficial home page of the Lovins stemming algorithm](http://www.cs.waikato.ac.nz/~eibe/stemmers/index.html) - with source code in a couple of languages
* [Official home page of the Porter stemming algorithm](http://www.tartarus.org/~martin/PorterStemmer/index.html) - including source code in several languages
* [Official home page of the Lancaster stemming algorithm](http://www.comp.lancs.ac.uk/computing/research/stemming/index.htm) - Lancaster University, UK
* [Official home page of the UEA-Lite Stemmer](http://www.cmp.uea.ac.uk/Research/stemmer/) - University of East Anglia, UK
* [Overview of stemming algorithms](http://www.comp.lancs.ac.uk/computing/research/stemming/general/index.htm)
* [PTStemmer](http://code.google.com/p/ptstemmer/) - A Java/Python/.Net stemming toolkit for the Portuguese language
* [jsSnowball](http://mazko.github.com/jssnowball/) - open source JavaScript implementation of Snowball stemming algorithms for many languages
* [Snowball Stemmer](http://trimc-nlp.blogspot.com/2013/08/snowball-stemmer-for-java.html) - implementation for Java
* [hindi\_stemmer](http://hlt.di.fct.unl.pt/luis/hindi_stemmer/) - open source stemmer for Hindi
* [czech\_stemmer](http://hlt.di.fct.unl.pt/luis/czech_stemmer/) - open source stemmer for Czech
* [Comparative Evaluation of Arabic Language Morphological Analysers and Stemmers](http://www.comp.leeds.ac.uk/eric/sawalha08coling.pdf)

This article is based on material taken from the [Free On-line Dictionary of Computing](http://en.wikipedia.org/wiki/Free_On-line_Dictionary_of_Computing) prior to 1 November 2008 and incorporated under the "relicensing" terms of the [GFDL](http://en.wikipedia.org/wiki/GNU_Free_Documentation_License), version 1.3 or later.

# Adding a PDF to the Gallery:

How are you adding the item? Here's how I do it.

1. Go to My Content and click Add Item
2. On the dialog, choose "the item is: an application"
3. Enter the URL to the PDF. The rest of the options are meaningless for the PDF.
4. Enter a tile, add tags and click ok.
5. Share the item with everyone and to the specific group you'll make the gallery from.
6. Go to the group and click the Share button.

# AGOL Security Issues

<http://forums.arcgis.com/threads/81200-Public-but-hidden?highlight=tags>

From Nathan: I've been using AGO Subscriptions since the beta. In my opinion, ESRI is going in the direction of making people buy ArcGIS Server to meet these kinds of needs. Right now whatever you post or host on AGO is either all public so every Tom Dick and Harry can view it, or you have to have users be part of your organization to keep things private, thus requiring logins. Even before last week's changes, you could share layers and maps with just a public group, but that still required people to have accounts and login. Now that that feature is gone, its either all public or all organization. The sharing is extremely messed up and complicated on AGO. I agree it would be nice to just make a map and send a link via email without having everyone in the world be able to access it. But that's what Server is for I guess....

From Jennifer:

Nathan -  
  
Yeah, that's what I was wondering. However, we have 3 GIS users and there is no way that our company is going to buy us a server. AGO just seemed so promising.  
  
The three potential solutions I see so far are and none are ideal, two allow the data to just be out there:  
  
1) Not searchable, but public. Accessible via AGO by login (Organization) or by specifically shared link.  
2) Not searchable, but public. Accessible via AGO by login (Organization) or by password protected link.  
3) Not public. Have some user level so that each client has their own organizational login to the products made for them but we would not be charged for each one of them as a separate user. No reason we'd buy a 200 user license if only 3 of us are actually using the product and the other 197 are just viewing products.  
  
It turns out that if you don't put important (i.e. logically searchable) tags into your tags your item will most likely not be found except by those with whom you've shared the link.  
  
However, any maps, applications etc you make will show up in that most recent maps section of the home page - meaning someone can scrape that page and get anything they want. It's not that I think that there is some malicious person out there scraping for important data but that it can be done. And our clients do not want their confidential information out there.