Land & Water

Summer Internship 2015/16



Developing Environmental Vocabularies

By Xavier Butcher – 29/01/16

Contents

[1 Heading 1 2](#_Toc394073019)

[1.1 Heading 2 2](#_Toc394073020)

[1.2 Formatting [style = Heading 2] 2](#_Toc394073021)

# Keywords

Database, Editors, Environment, Excel, Methods, Notepad, RDF, Registry, Semantics, SKOS, Soil, SPARQL, SQL, Taxonomy, TopBraid, URI, Vocabulary, Wiki.

# Introduction

The aim for this project was to create and collate a number of online vocabularies into a standard format and location. Vocabularies are often hosted online, and are used to support environmental applications. They contain labels, definitions and other related information for a particular concept or registry. This project mostly focussed on soil and land operations. Currently, a number of vocabularies are unique to their individual application, and are often in different formats (text, csv, pdf, or lists). Having a definitive vocabulary on a subject is critical, as the same term could have different definitions or meanings in different vocabularies.

It is also necessary to introduce semantic relationships, which essentially add another layer of detail to vocabulary. This means different terms can be placed in a hierarchical order (broader and narrower), as well as relating similar terms. These vocabularies follow the guidelines of SKOS, the Simple Knowledge Organization System, which is an industry standard of rules and guidelines across online vocabularies which aims to make publication and use of vocabularies an easy and standard process.

SPARQL the SPARQL Protocol and RDF Query Language is, as the acronym suggests, an RDF query language. It is tending to be the industry standard query language. SPARQL allows the user to query semantic data in databases, and retrieve and manipulate RDF data.

There were 4 main tasks relating to this project, which were completed successfully:

•          Source relevant existing vocabularies

•          Convert the vocabulary content using semantic web technologies (RDF/SKOS)

• Harmonising vocabulary content with existing vocabularies where applicable

•          Publish vocabularies to SPARQL triple stores where applicable

# Example

For those who have little or no knowledge of vocabularies, SKOS or semantics, this section will give a basic example of how these theories are used.

This example will centre on the ‘Concept’ of a “Margherita Pizza”. As a bare minimum, a Concept must have a Label (Margherita Pizza), a URI (Unique Resource Identifier – a unique web address where the concept can be located) and a Definition (Margherita Pizza is a flatbread generally topped with tomato sauce and cheese and baked in an oven). A Concept may also be part of a hierarchy of other Concepts. In this case, “Margherita Pizza” is a Narrower Concept of “Pizza”, which itself is a Narrower Concept of “Food”. A Concept can also be related to other concepts regardless of hierarchy, for instance “Margherita Pizza” is related to “Mushroom Pizza”. A Concept can also be part of a collection. For example “Margherita Pizza” can belong to the “Vegetarian Pizza” collection.

# Materials

* National Committee on Soil and Terrain (2009), ‘Australian soil and land survey field handbook (3rd edn).’ (CSIRO Publishing: Melbourne)
* ‘Soil Information Transfer and Evaluation System (SITES) – Database design and exchange protocols (version 2.0) (2012).’ Jacquier, D; Wilson, P; Griffin, T; Brough, D. (CSIRO Publishing: Canberra)
* ‘Soil Chemical Methods – Australasia’ (2011). Rayment, G; Lyons, D. (CSIRO Publishing: Melbourne).
* The CSIRO environmental informatics Wiki - <https://wiki.csiro.au/display/EI/Environmental+Informatics+Home>
* The CSIRO vocabularies and vocabulary services Wiki - <https://wiki.csiro.au/display/VOCAB/Home>
* The Linked Data Registry - <http://registry.it.csiro.au/sandbox/student/xavier>
* Microsoft office suite
* Notepad++
* TopBraid Composer
* RDF123
* Microsoft SQL Server Management Studio

# Method

This report will explain in depth the method used to create a vocabulary. Figure 1 illustrates the steps involved in generating a vocabulary. There are different programs that achieve the same outcome, however I recommend using the same programs Megan and myself used, as you will be able to follow our notes. What I felt was lacking was a clear step by step method with screenshots and annotations, which is what this method intends to fix. A clearer version of this can be found on my wiki.

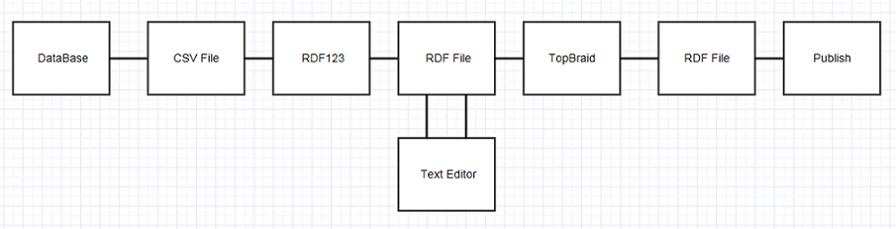


Figure 1: Process diagram

## Background Reading

Prior to starting the creation of a vocabulary, it was necessary to undertake an intensive few days of background reading. I had little-to-no knowledge of informatics before starting this internship. Except for Excel, I had not used any of the other programs necessary for this work, so I had to learn these from scratch. I also had minimal knowledge of the actual concept of vocabularies, especially including SKOS, SPARQL, RDF or Semantics, so much of my time was spent getting up to speed on these concepts.

There are a number of Wiki’s, both from previous interns who had worked in this area previously, as well as those managed by the Environmental Informatics team. I found that the most useful notes are in the Vacation Scholarship Readings, especially: Tom Baker's Presentation, the SKOS Primer, Megan’s AGIFT spreadsheet to RDF, and Jane Frazier's SKOS Manual. I found that this vocabulary work was easy to learn up to a certain degree, but very hard to master the more intricate workings of this process.

## Database

1. Download and Install Microsoft SQL Server Management Studio
2. When prompted to connect to server, input these fields (the password is ''password''):

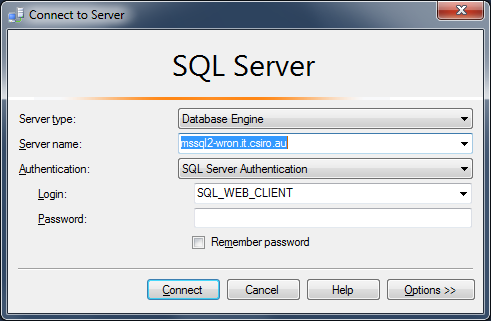


Figure 2: SQL Login

1. Find the Table folder in the Natsoil database:

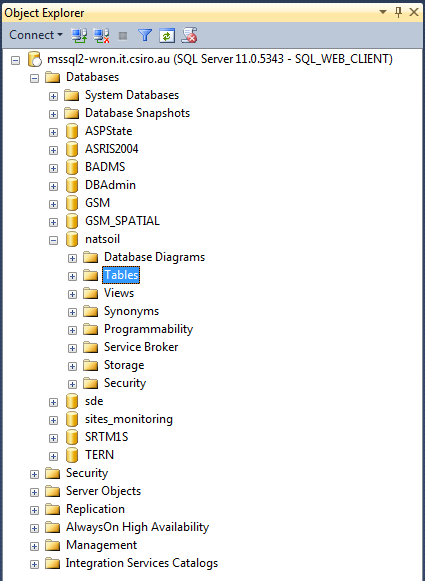


Figure 3: Menu Navigation

1. View the Top 1000 Rows of the table you're interested in (if there are more than 1000 entries, in the SQL code, just change ‘Select Top 1000’ to ‘Select’).

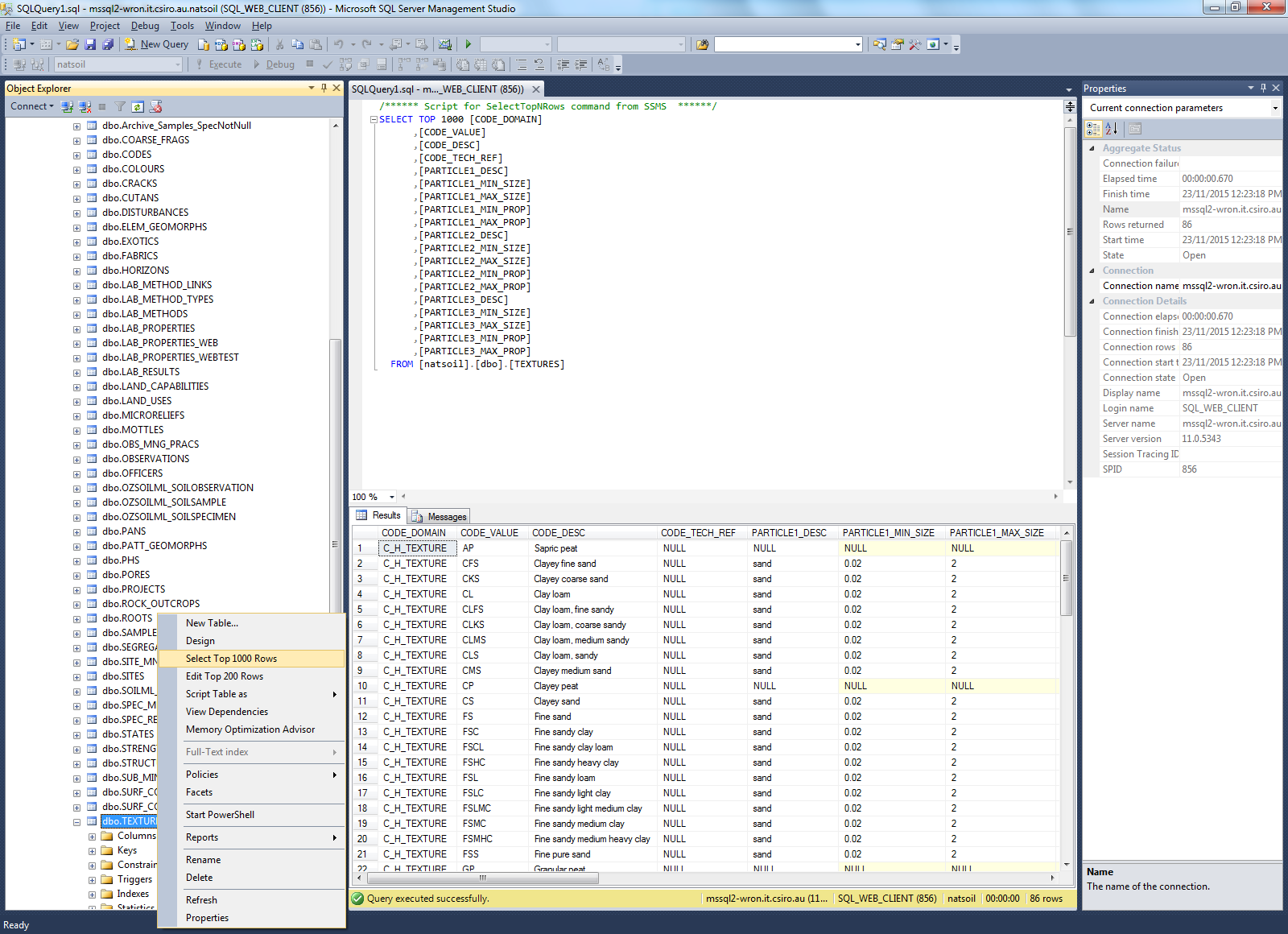


Figure 4: Top 1000 rows

1. Using the SQL code in the top half of the screen, eliminate the columns that are not needed. Then Save the Results as a .CSV:

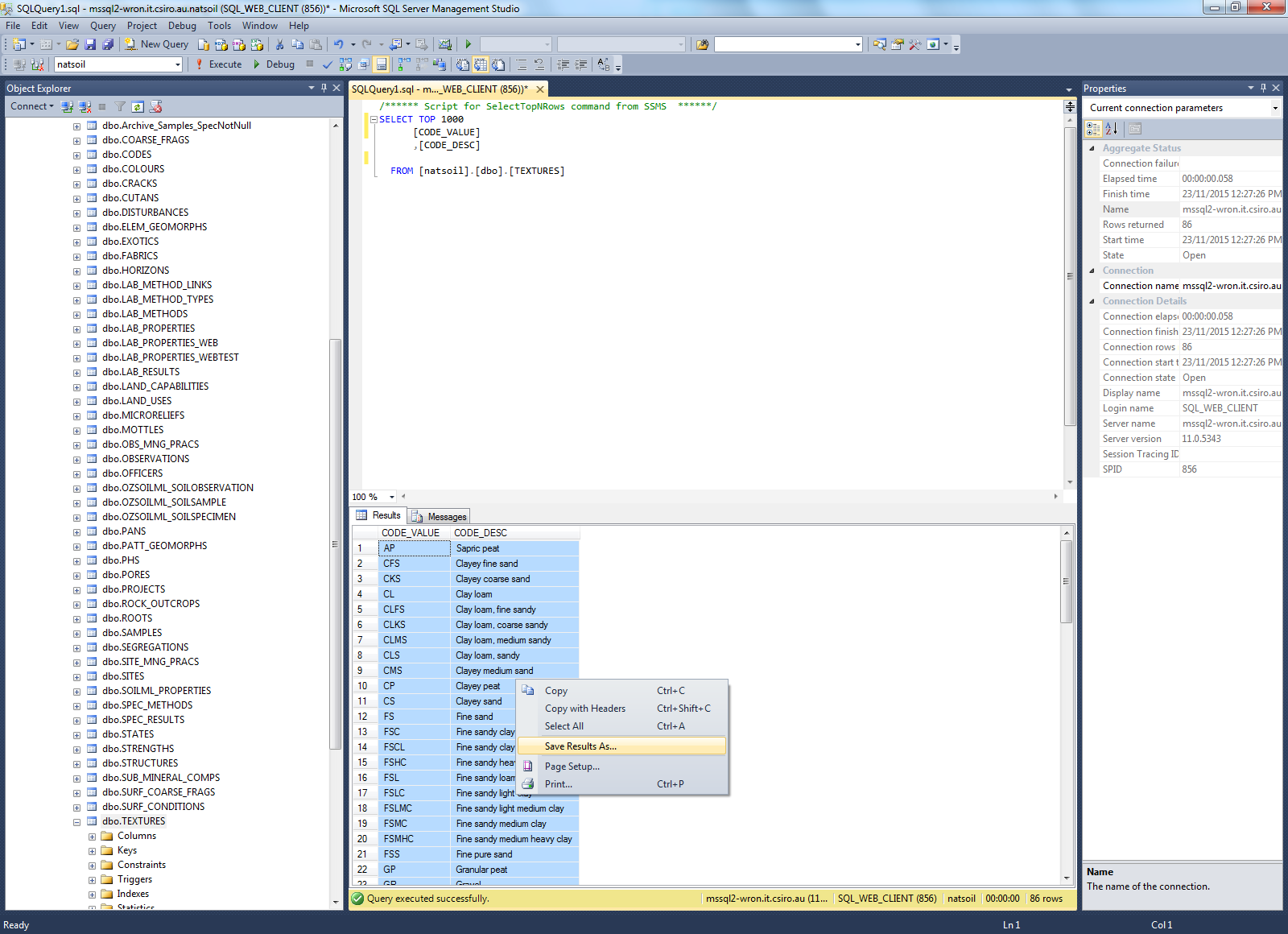


Figure 5: Save results

## Excel (CSV)

1. Establish best practice URI rules. In my case I used dashes for spaces, only used lowercase and used terms instead of codes
2. Decide on a base URI on which everything will be based off. eg: http://registry.it.csiro.au/def/soils/texture/ . Type it and Fill down.

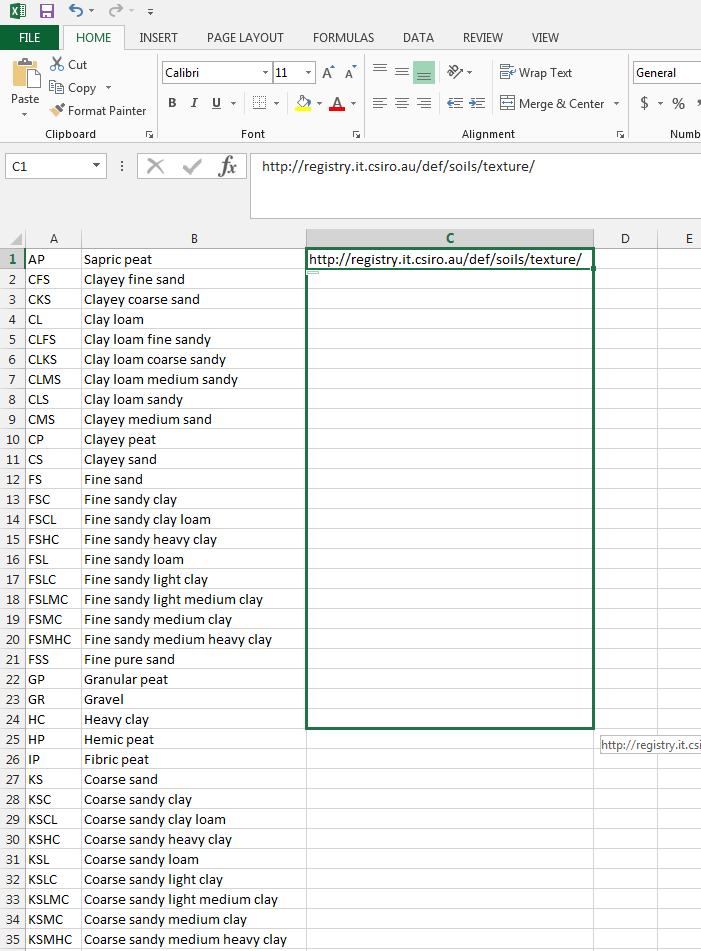


Figure 6: URI Filldown

1. Using a formula is the most efficient way to convert the terms in column B into a URI which follow the rules set out in point 1. The first function to use is the substitute formula, which replaces certain characters with something else. eg: =SUBSTITUTE(B1, " ", "-") . Fill down.

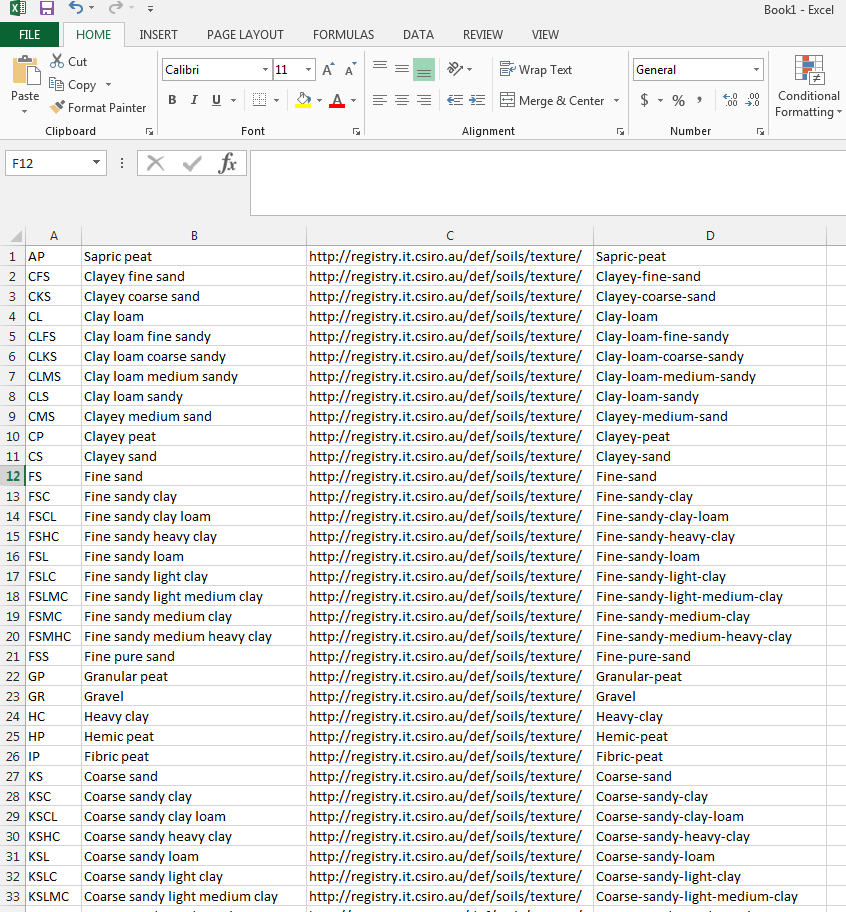


Figure 7: CSV Dashes

1. The next function will be the lower formula. This function changes everything in the cell to lower case. eg: =LOWER(D1) . Fill down.

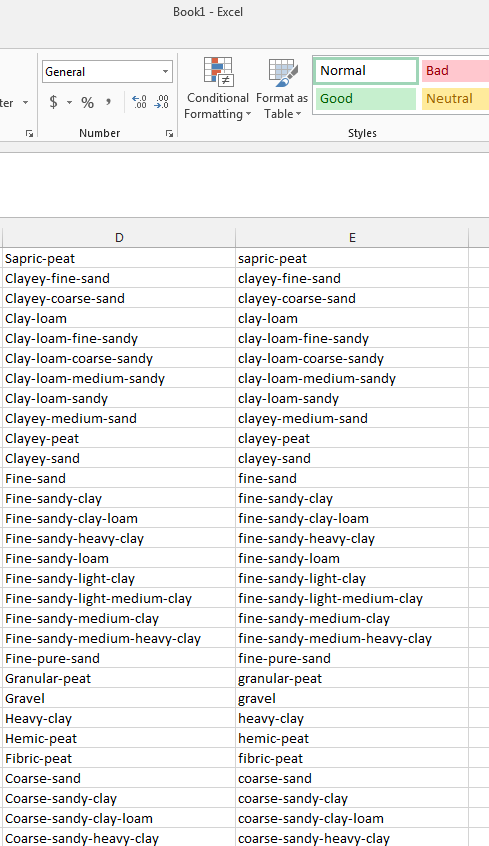


Figure 8: CSV Lower

1. To merge the base URI in column C with the unique URI prefix in column E, we need to use the concatenate function. This merges text from different cells and places it in a new one. eg: =CONCATENATE(C1, E1) . Fill down.

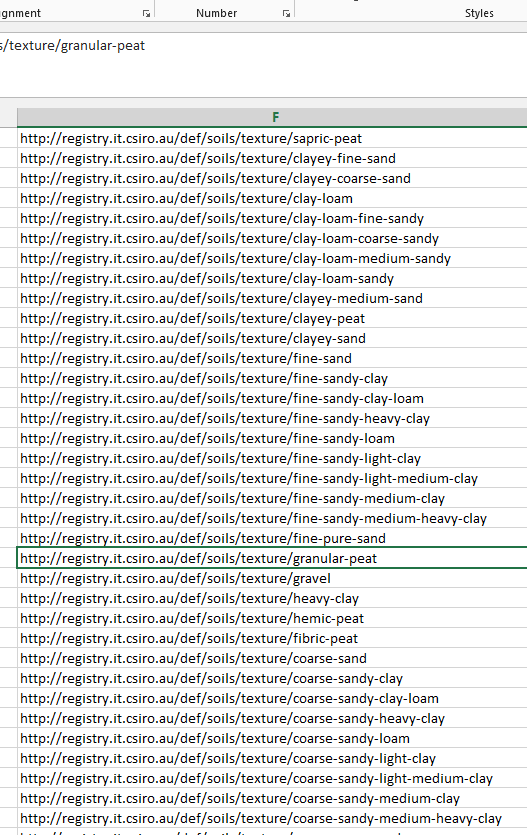


Figure 9: Concatenate

1. Next, we want to create the prefLabel. First, type 2 apostrophe's in a cell (Note: Not quotation marks). Fill down. Then type '@en' in the next cell. Fill down. We then want to concatenate these. eg: =CONCATENATE(G1, B1, G1, H1) . Fill down.

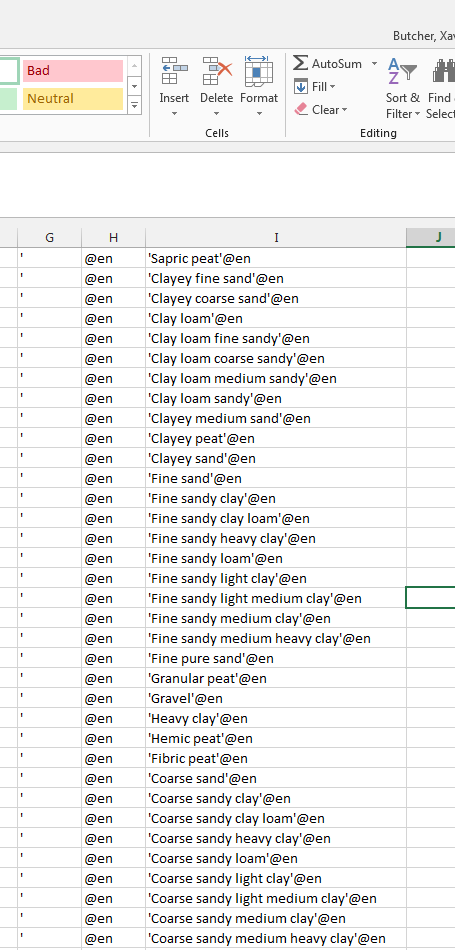


Figure 10: prefLabel

## RDF123

RDF123 is a powerful RDF editor. However, it is not user friendly at all. It may take some time to learn how to use the program properly.

1. In the prefix definition window, you will need to add or amend these definitions:

|  |  |
| --- | --- |
| **Title** | **URL** |
| Base | http://registry.it.csiro.au/def/soils/texture/ |
| mapBase | http://registry.it.csiro.au/def/soils/texture/ |
| Skos | http://www.w3.org/2004/02/skos/core# |
| Dct | http://purl.org/dc/terms/ |

1. The next step is to import the CSV file you created in excel. In the spreadsheet window, open your spreadsheet. Your screen should look like this so far:

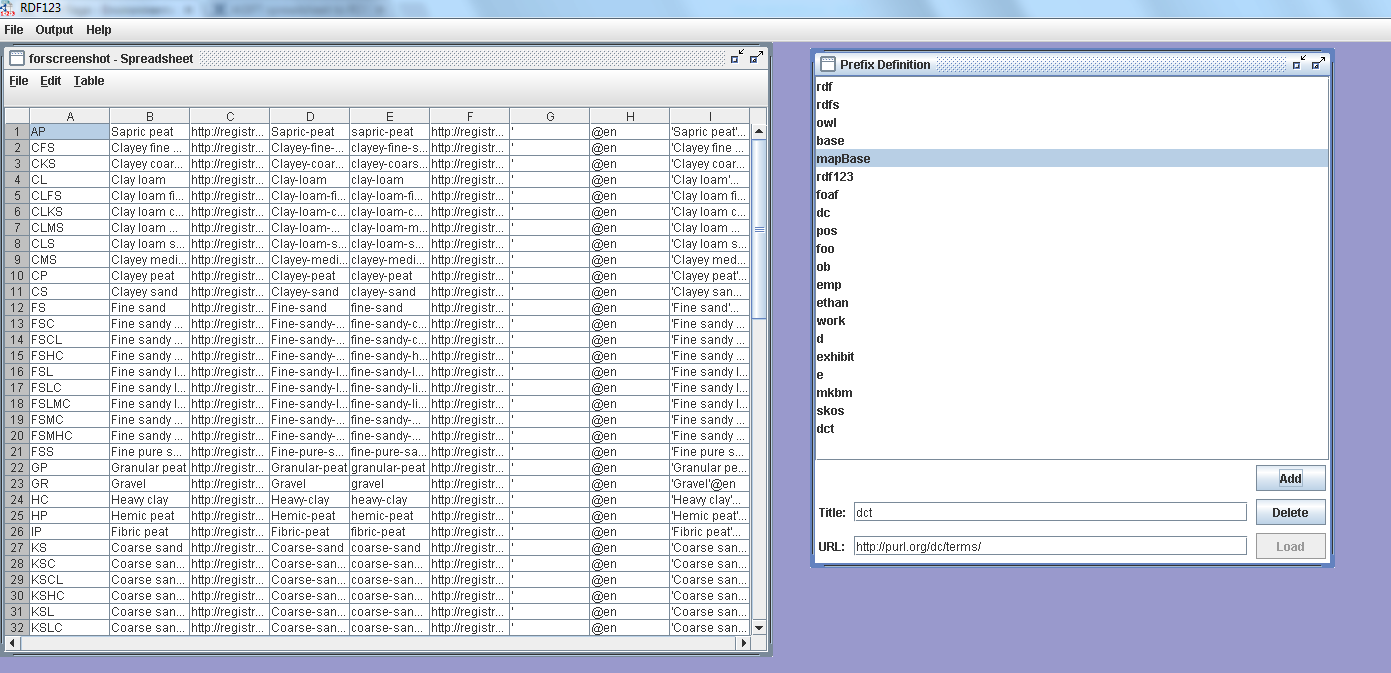


Figure 11: CSV Import

1. The next step is to use the 3rd window (map graph) to create a sort of mind map of what the data is trying to say. The first step is to create vertices and edges. If there is data already in the map area, just delete them, start with a blank canvas. In this case we will need 5 vertices and 4 edges. Don't worry about labels for now. The edges should radiate out of 1 central point to the other 4 vertices. It should look something like this:

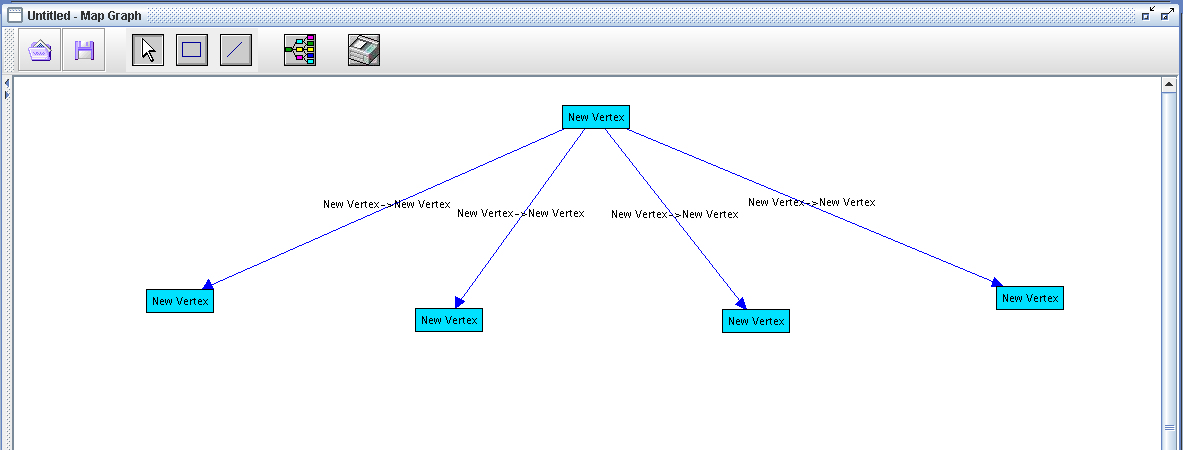


Figure 12: Blank Map

1. Now we need to start labeling the vertices. Firstly, the central vertex must be the unique URI. To tell the program that each individual URI is located in column F we need to use the notation "Ex:$6". The other columns you need are A, B and I (which are Ex:$1, 2 and 9 respectively). The remaining box must be called a "skos:Concept" to let the program know that each row is a separate concept.

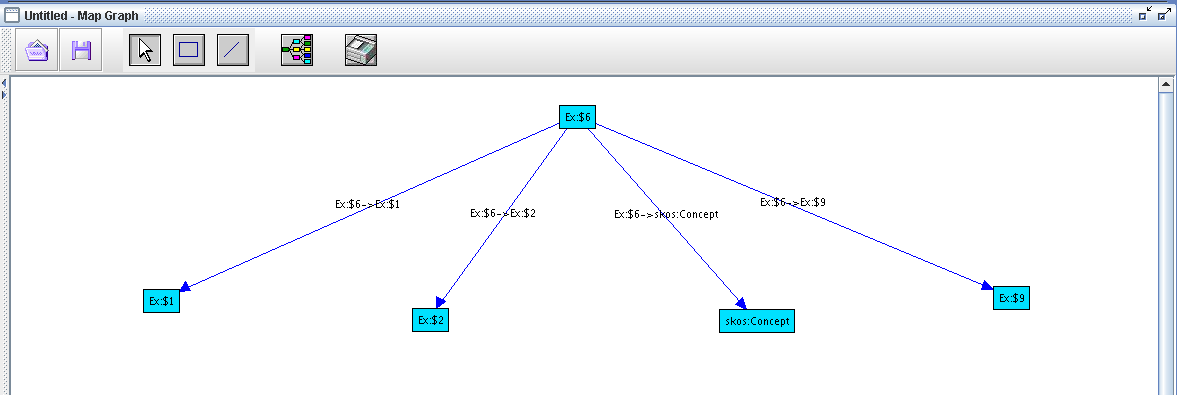


Figure 13: Map Labels

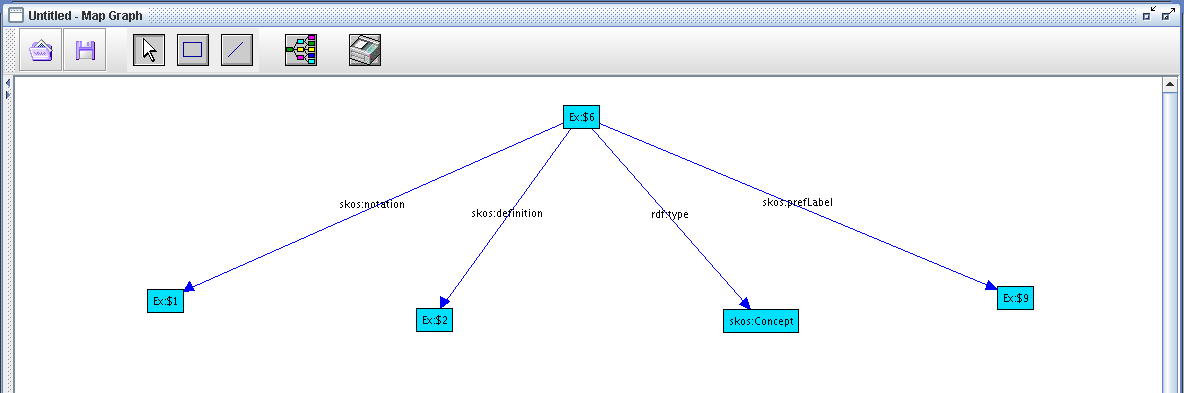
1. Next we must label the edges. Column 1 is a 'skos:notation', column 2 is a 'skos:definition', skos:Concept is a 'rdf:type' and column 9 is a 'skos:prefLabel'. Label the edges as such.
2. 

Figure 14: Map edges

1. You can now go to the overall window, and click the output menu at the top. Then click 'view spreadsheet in RDF'. Make the display type 'N3' as it’s easier to read. Copy this text output and paste it into notepad++.

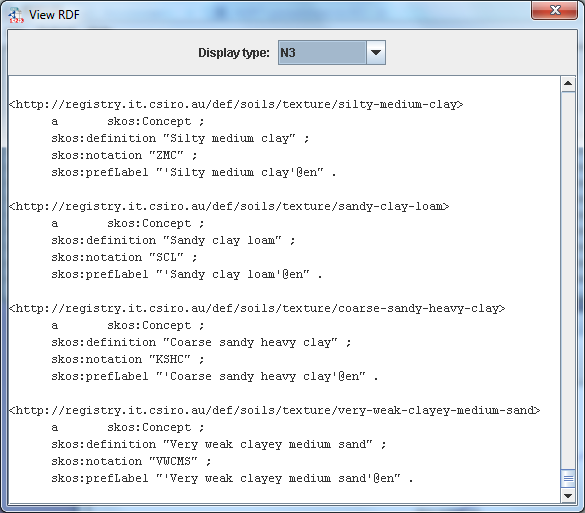


Figure 15: Output

## Text Editor (notepad++)

Unfortunately, the output from RDF123 is not usually error free. Some modifications to the text file will need to be made. As it is coding language, even the smallest discrepancy in the text will cause an error. Patience is key!

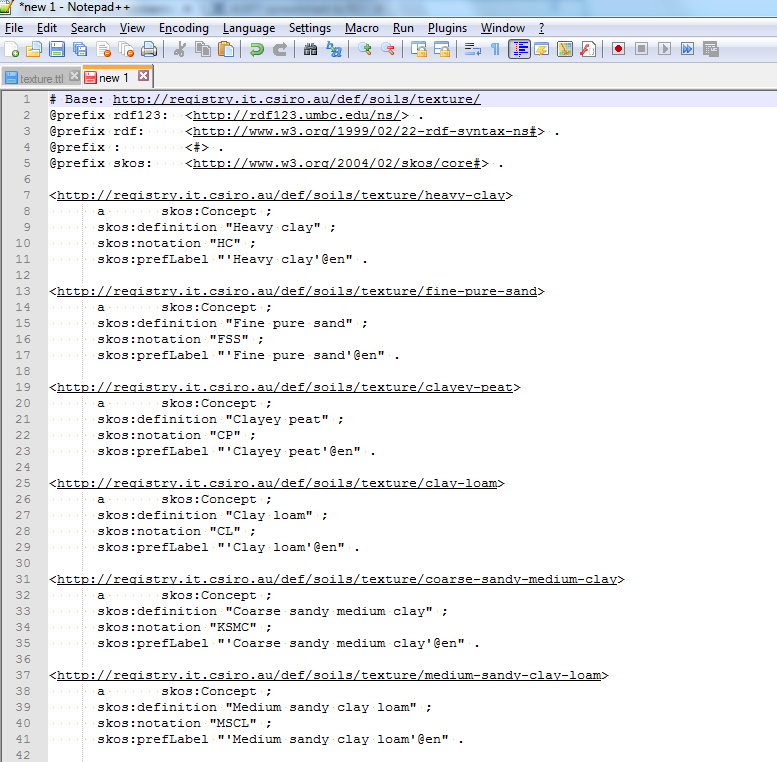
1. Initially it should look like this:  
   

Figure 16: N++ input

1. You're going to want to save it as a .TTL file so that it can be read by TopBraid.
2. The first thing you'll need to correct is in the first few lines:

|  |  |
| --- | --- |
| **Bad output from RDF123** | **Change to:** |
| # Base: <http://registry.it.csiro.au/def/soils/texture/> | #baseURI: <http://registry.it.csiro.au/def/soils/texture/> |
| @prefix : <#> . | DELETE |

1. The next thing that needs to be changed is that the prefLabel for each concept has not come across properly, due to RDF not recognising the '@en' as a separate suffix that isn't part of the concept name. To do this, use the Replace tool (in the Search menu). In this case for example, we will need to change <"'Heavy clay'@en"> to <"Heavy clay"@en>. This needs to be done throughout the whole document. The easiest way to do this is to replace all:

|  |  |
| --- | --- |
| **Find** | **Replace** |
| "'(quotation then apostrophe) | "(quotation) |
| '@en" | "@en |

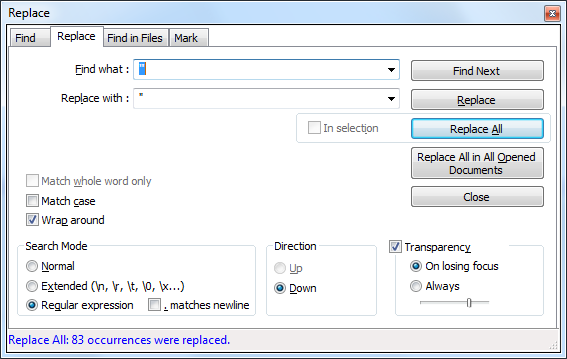


Figure 17: Replace text

1. This should be all that needs to be edited in notepad++, however there may be more depending on how it exported. It should look like this:

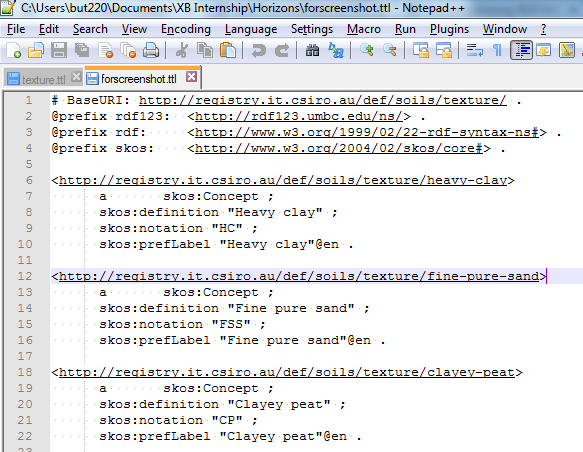


Figure 18: N++ Output

As a footnote, this table did not have a date created column. But if yours does, these are the steps needing to be completed, as dates come across poorly out of RDF:

**Editing [dct:created](http://dctcreated/) and [dct:modified](http://dctmodified/)**

Dates were not in the format required in RDF. There were a number of issues,   
1. Dates contained 0:00 at the end of the string  
2. RDF 123 picked up the text as a string  
3. dates were in the form dd/mm/yyyy, but need to be yyyy-mm-dd

**Regex to fix these issue**

|  |  |
| --- | --- |
| **1.** | **Remove the 0:00 and recognize number as a date** |
| FIND | "([/0-9]+) 0:00" |
| REPLACE | "$1"^^[xsd:date](http://xsddate/) |

|  |  |
| --- | --- |
| **2.** | **Re-format so that date reads yyyy-mm-dd** |
| FIND | "([0-9]+)/([0-9]+)/([0-9]+)"\^\^[xsd:date](http://xsddate/) |
| REPLACE | "$3-$2-$1"^^[xsd:date](http://xsddate/) |

|  |  |
| --- | --- |
| **3.** | **Ensure that single digits have a 0 in front ( eg 9 --> 09)** |
| FIND | -([1-9])"\^\^xsd |
| REPLACE | -0$1"^^xsd |

## TopBraid Composer

1. To import your .TTL file into TopBraid, you must add the containing folder to the navigation panel in the bottom left. Then just double click on the .TTL file and it should open.
2. You must also import the SKOS core .TTL file and the Dublin core .TTL file (dc-1.1.TTL). Go to the imports tab at the bottom, and import the local file. From the properties namespace in the top right, drag the dct:description into the pane at the bottom.

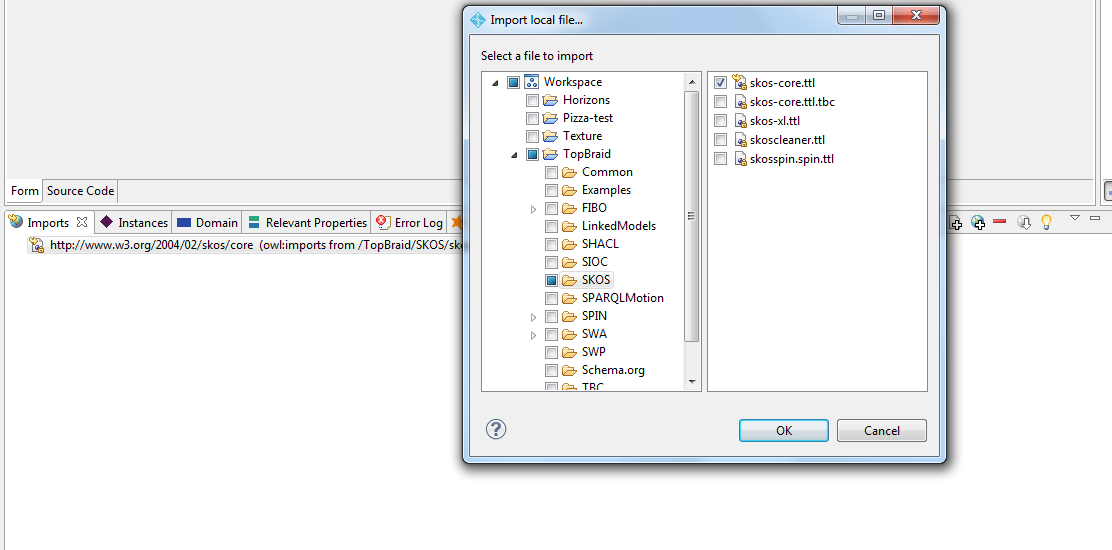


Figure 19: TopBraid Import

1. If the editing in notepad++ was done successfully, then each concept should appear in the Instances tab at the bottom, and in the skos:Concept tab in the Namespace at the top left.

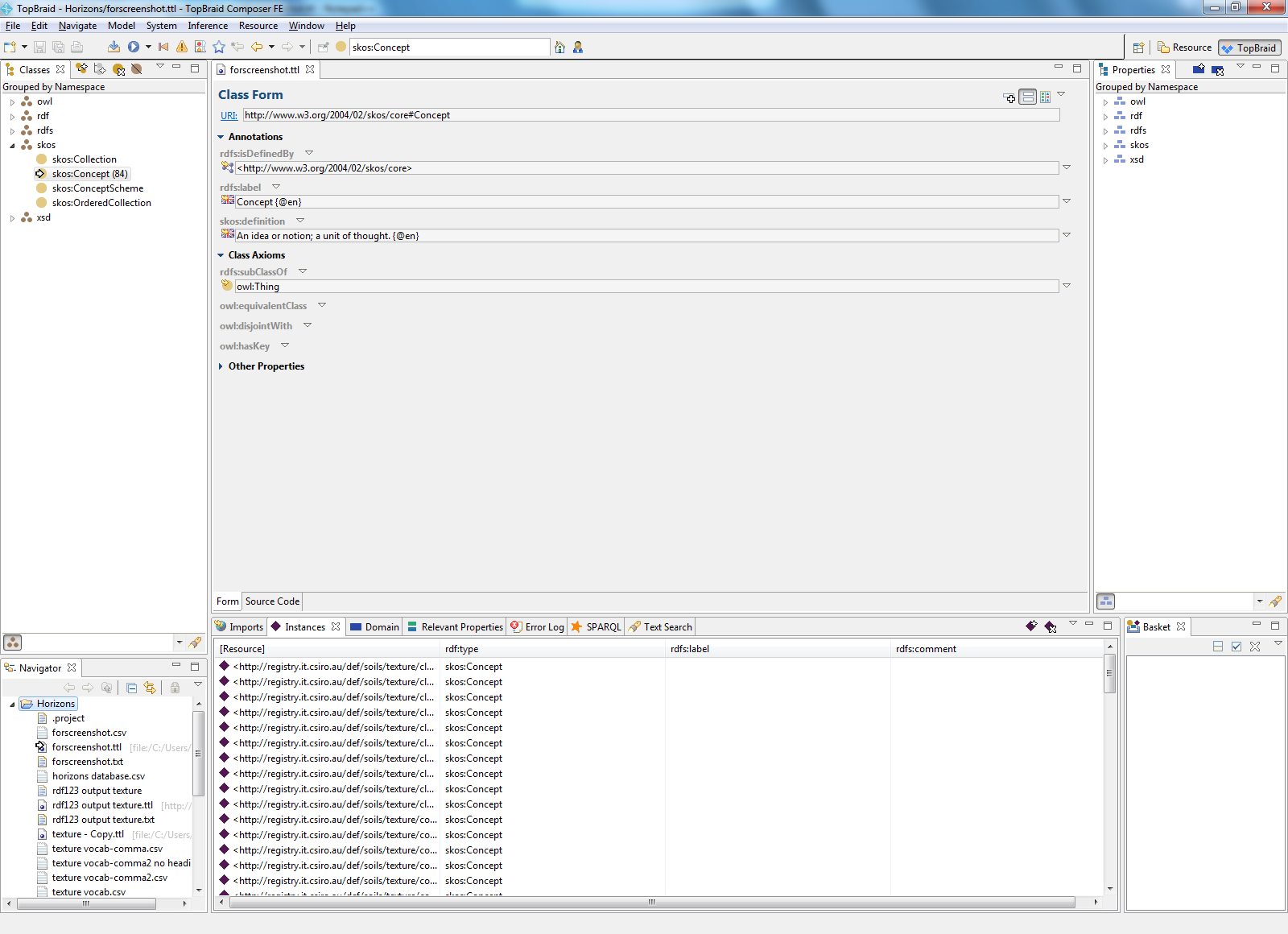


Figure 20: TopBraid Input

1. The next step is to add an rdfs:label to each concept. This is done by using SPARQL code found below:

INSERT{ ?s rdfs:label ?label }

WHERE {

{ ?s a skos:Concept . } UNION { ?s a skos:ConceptScheme . } UNION { ?s a skos:Collection . } UNION { ?s a owl:Ontology . }

NOT EXISTS { ?s rdfs:label ?l }

OPTIONAL { ?s skos:prefLabel ?pl . }

OPTIONAL { ?s dc:title ?t . }

OPTIONAL { ?s dct:title ?tt . }

BIND( REPLACE(str(?s), '^.\*(#|/)', "") AS ?localname)

BIND( REPLACE(str(?s), '[^/^#]+$', "") AS ?namespace)

BIND( REPLACE(str(?namespace), '(#|/)$', "") AS ?ns)

BIND( REPLACE(str(?ns), '^.\*(#|/)', "") AS ?nsfrag)

BIND ( STR ( COALESCE ( ?tt, ?t, ?pl, IF( STRLEN(?localname), ?localname, ?nsfrag) ) ) AS ?label )

}

1. The dct:description must also be added, using this SPARQL code.

INSERT { ?s dct:description ?desc }

WHERE {

{ ?s a skos:Concept . } UNION { ?s a skos:ConceptScheme . } UNION { ?s a skos:Collection . } UNION { ?s a owl:Ontology . }

NOT EXISTS { ?s dct:description ?d }

OPTIONAL { ?s skos:definition ?def }

OPTIONAL { ?s skos:scopeNote ?note }

OPTIONAL { ?s rdfs:comment ?com }

OPTIONAL { ?s skos:prefLabel ?plab }

?s rdfs:label ?lab .

BIND ( COALESCE ( ?def , ?note , ?com , ?plab , ?lab ) AS ?desc )

FILTER ( (datatype(?desc) = xsd:string) || ( lang(?desc) = "en" ) || (lang(?desc) = "EN") )

}

1. Once both of those search queries have been run, your instance panel should look like this:

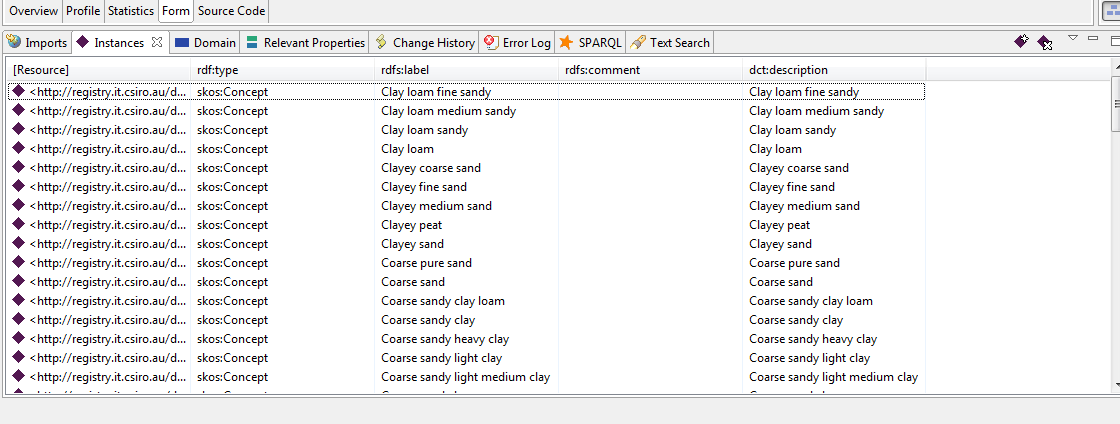


Figure 21: Description Added

1. The next step is to create a prefix for the concepts. Click the home button, and then click the overview tab. Add the prefix you would like to use, and then the base URI. I used texture in this case.

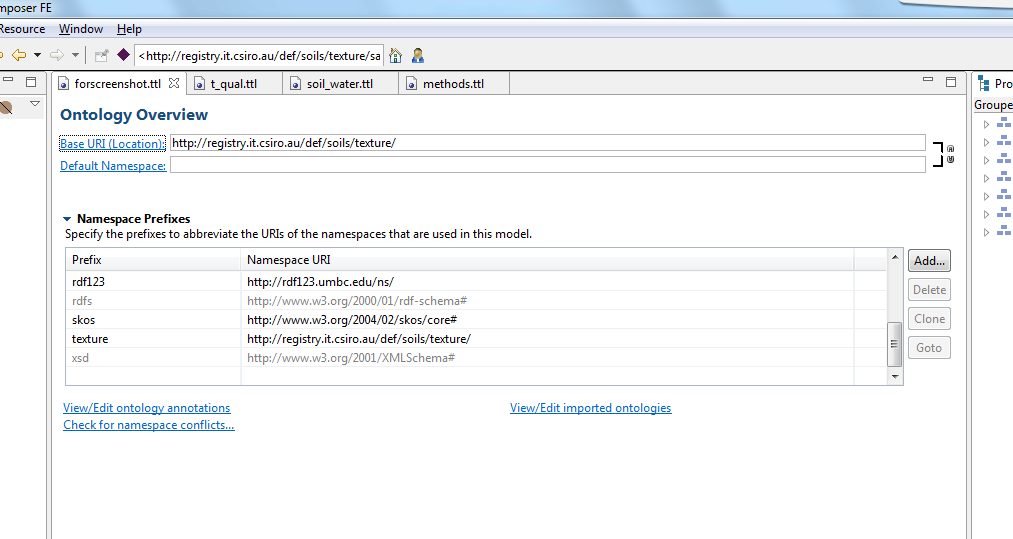


Figure 22: Prefix

1. Now is the time to do final quality control and checking of your concept data and make sure that all the properties you need have been filled. We must now do some final editing of the register in TopBraid before we can start uploading the vocabulary.
2. Click on the home screen, but this time navigate to the Form tab. Here we will need to add some necessary properties. The ones you'll need at a minimum are: dc:source, rdfs:comment, rdfs:label, dct:created, dct:creator, dct:description. Note: change the data type from string to date for date created (little white arrow at end of box). A purple diamond means the input should be a URI, whilst a blue square means the input should be a text string. If you do not have a URI for a particular field, you can get around this by firstly inserting a blank URI (<>) and then going to the source code and changing the <> to a " ". This will change the form to accept a text string instead. Your form should look like this after this has been completed:

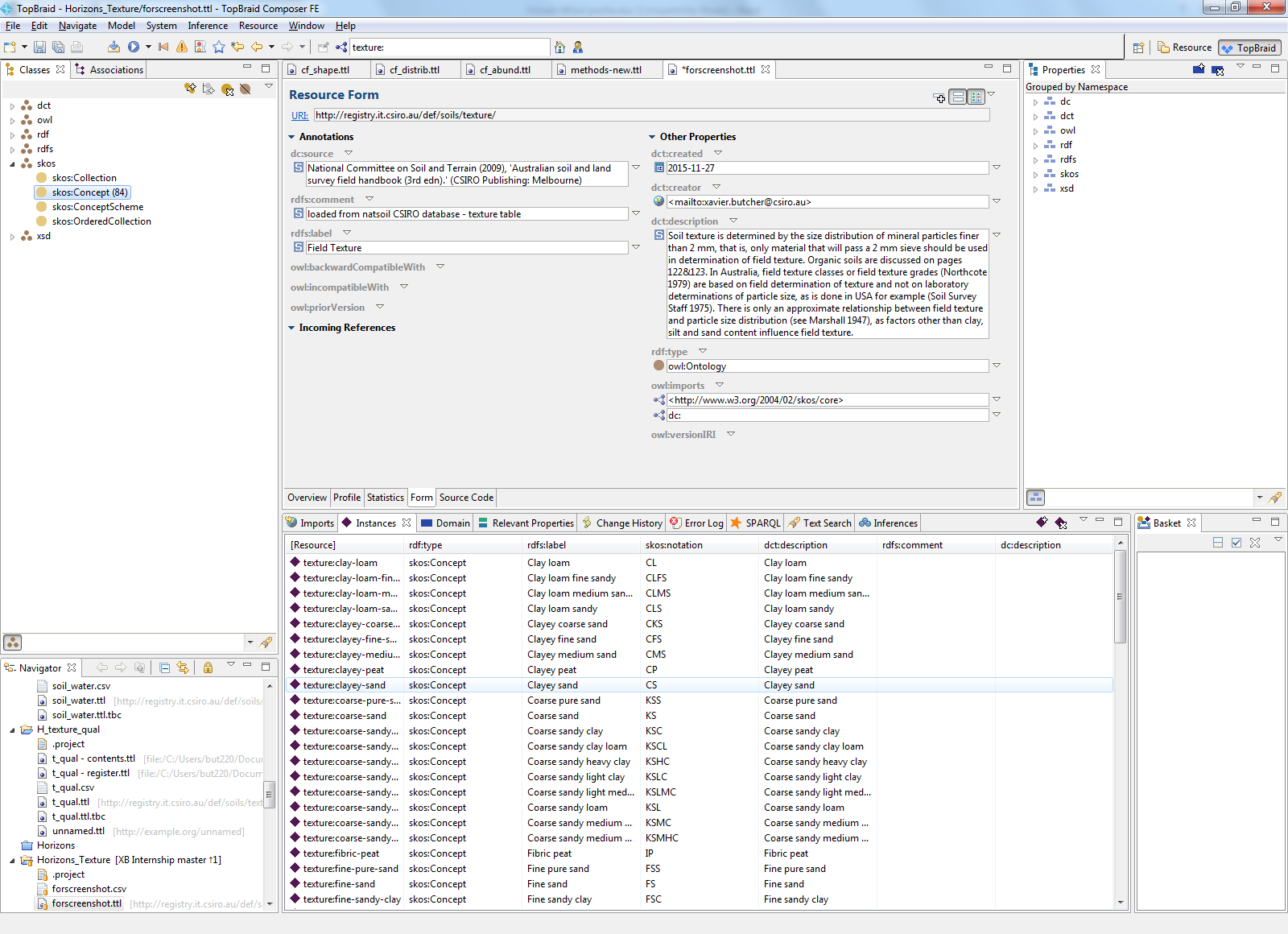


Figure 23: Registry Properties

1. This should be the end of the editing you need to do in TopBraid, however now we must use notepad++ again to do some final post processing.

## Post Processing (Text)

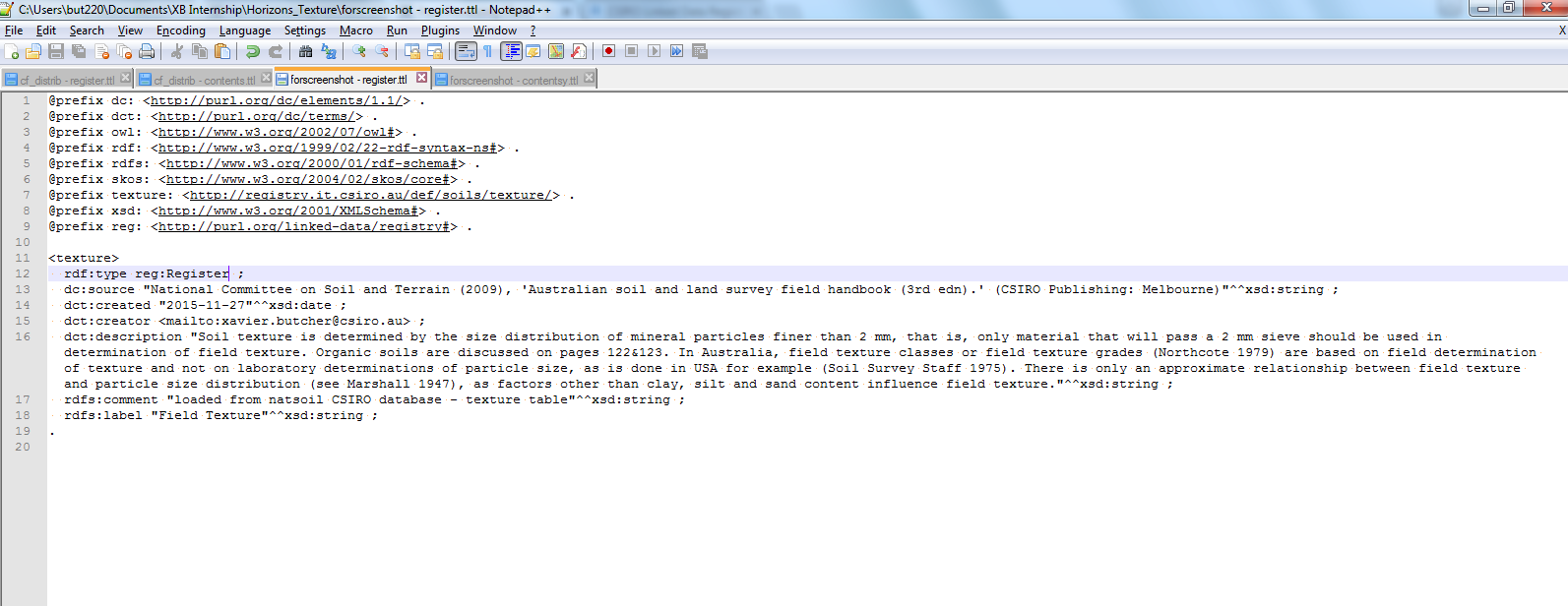
1. The first thing to do is create 2 copies of your .TTL file. One called registry, the other called contents.
2. We will start working with the register file. Firstly, delete the top 4 lines (baseURI, imports 1 and 2 and a blank line).
3. Next, we will delete all of the 'contents', leaving only the register information.
4. Following this, delete line 5 (the rdf123) prefix. Then add another prefix: @prefix reg: <[http://purl.org/linked-data/registry#](http://purl.org/linked-data/registry)> .This just tells the program that this particular file is part of the registry.
5. We will then change "owl:ontology" to "reg:register".
6. We will then change line 11 from "texture:" to "<texture>. This is just the end of your base URI.
7. Then we will delete lines 19 and 20, the 2 owl imports.
8. That is it for the registry file, we will now move onto the contents file. The final registry file should look like this:  
   

Figure 24: Registry

1. Open the contents file.
2. Much like the registry file, we will delete the top 4 lines. We will also delete the prefix rdf123 line. But this time we will delete the registry information, and keep the contents information.
3. We then need to do a bulk search and replace. We need to change "texture:clay-loam" to "<clay-loam>". The regular expression code you need for this is:

|  |  |
| --- | --- |
| **Find** | **Replace** |
| texture:([a-zA-Z0-9\_\-]+) | <$1> |

1. The text editing should now be done. The contents document should look like this:

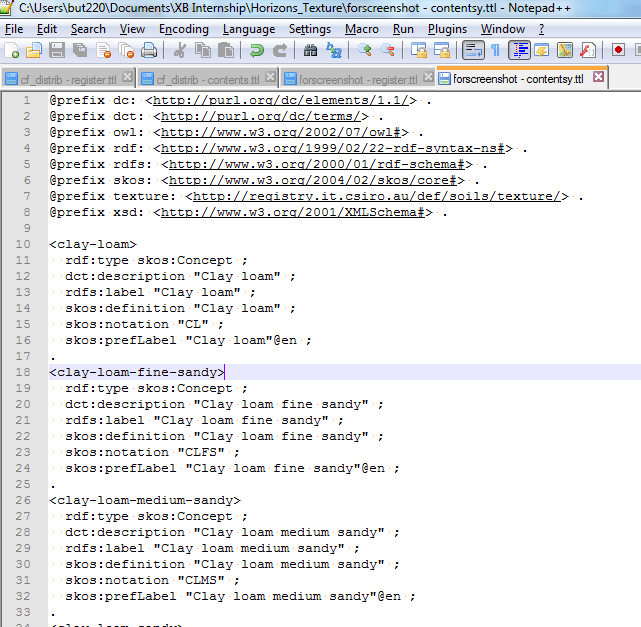


Figure 25: Contents File

## Publishing

1. Confirm your registry and contents files are correct, as once uploaded to the Linked Data Registry (LDR), it is hard/impossible to remove/add/modify.
2. Navigate to your registry location. You will need to create an account/login if you haven't already. Mine is: http://registry.it.csiro.au/sandbox/student/xavier
3. The page should look like:

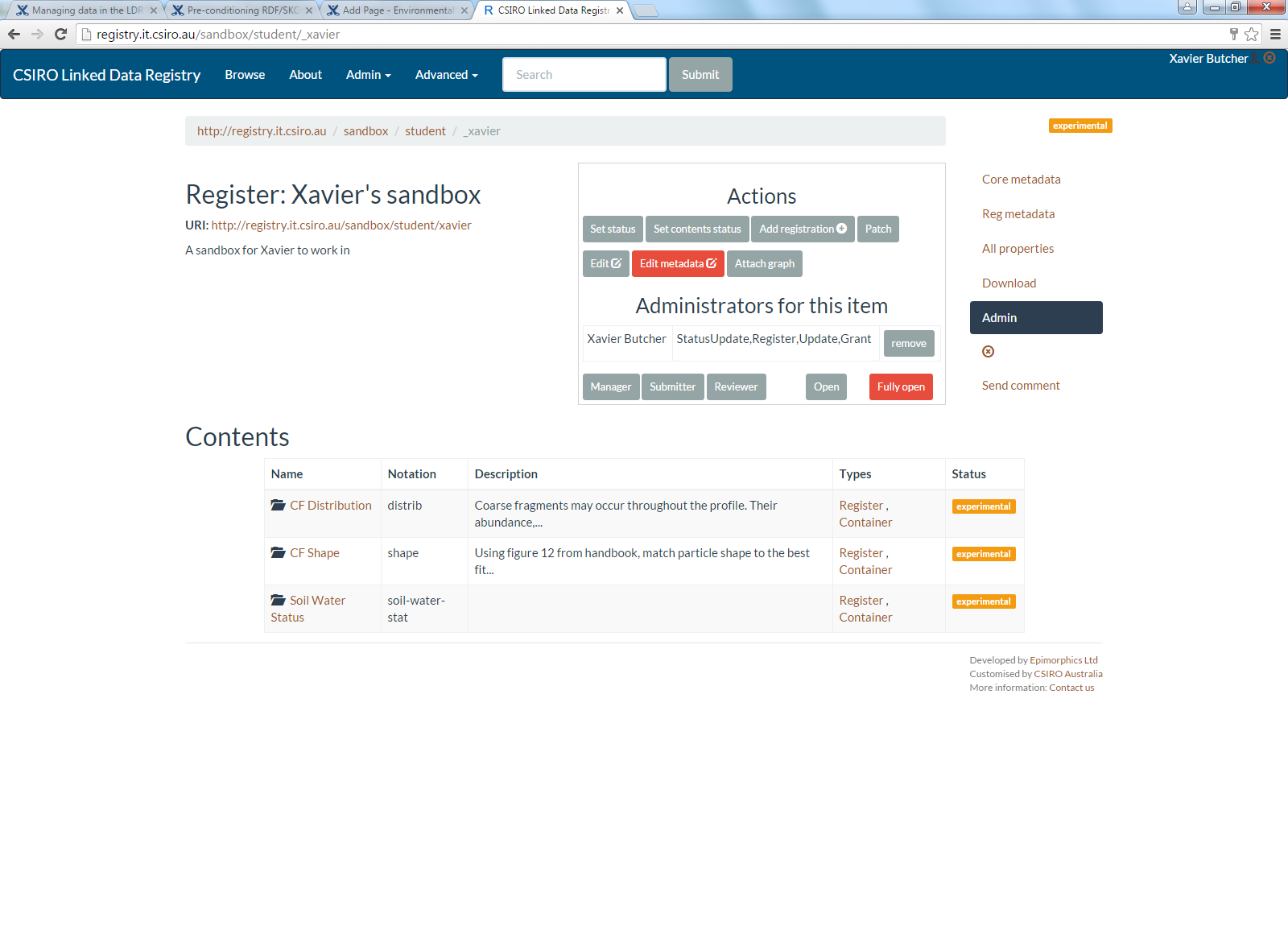


Figure 26: Navigation

1. Then click on admin, and select add registration.
2. Next, click upload, and choose your registry file.
3. It should look like this:

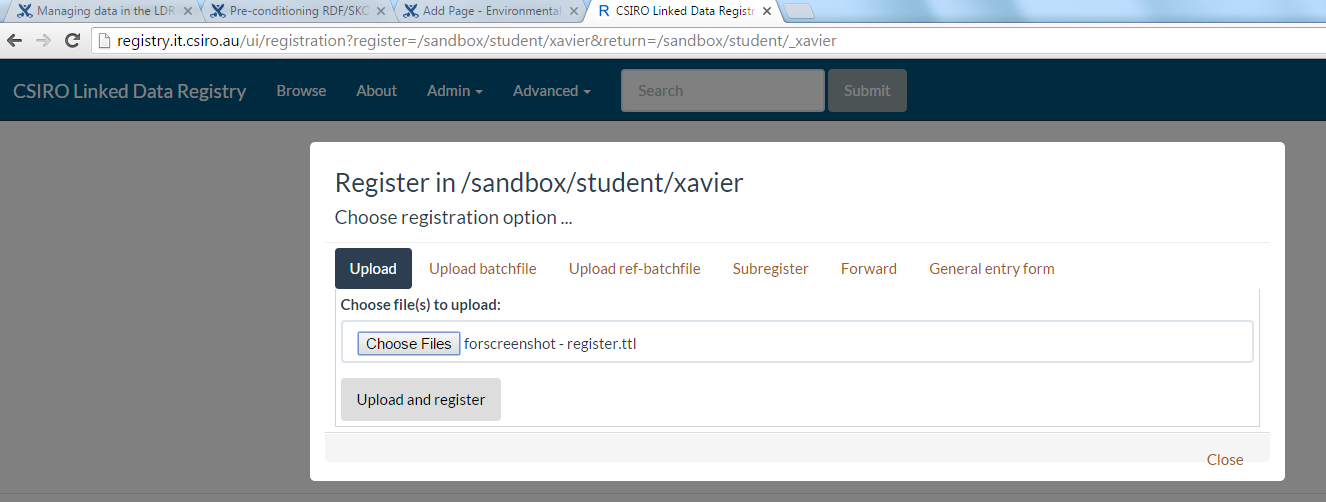


Figure 27: Register upload

1. Once uploaded, you should see the new register appear. Click on it, and you should see the register you uploaded, with labels and descriptions. However, there shouldn't be any contents - yet.
2. When inside the register, using the same method as before, upload the contents file.It should look like this:

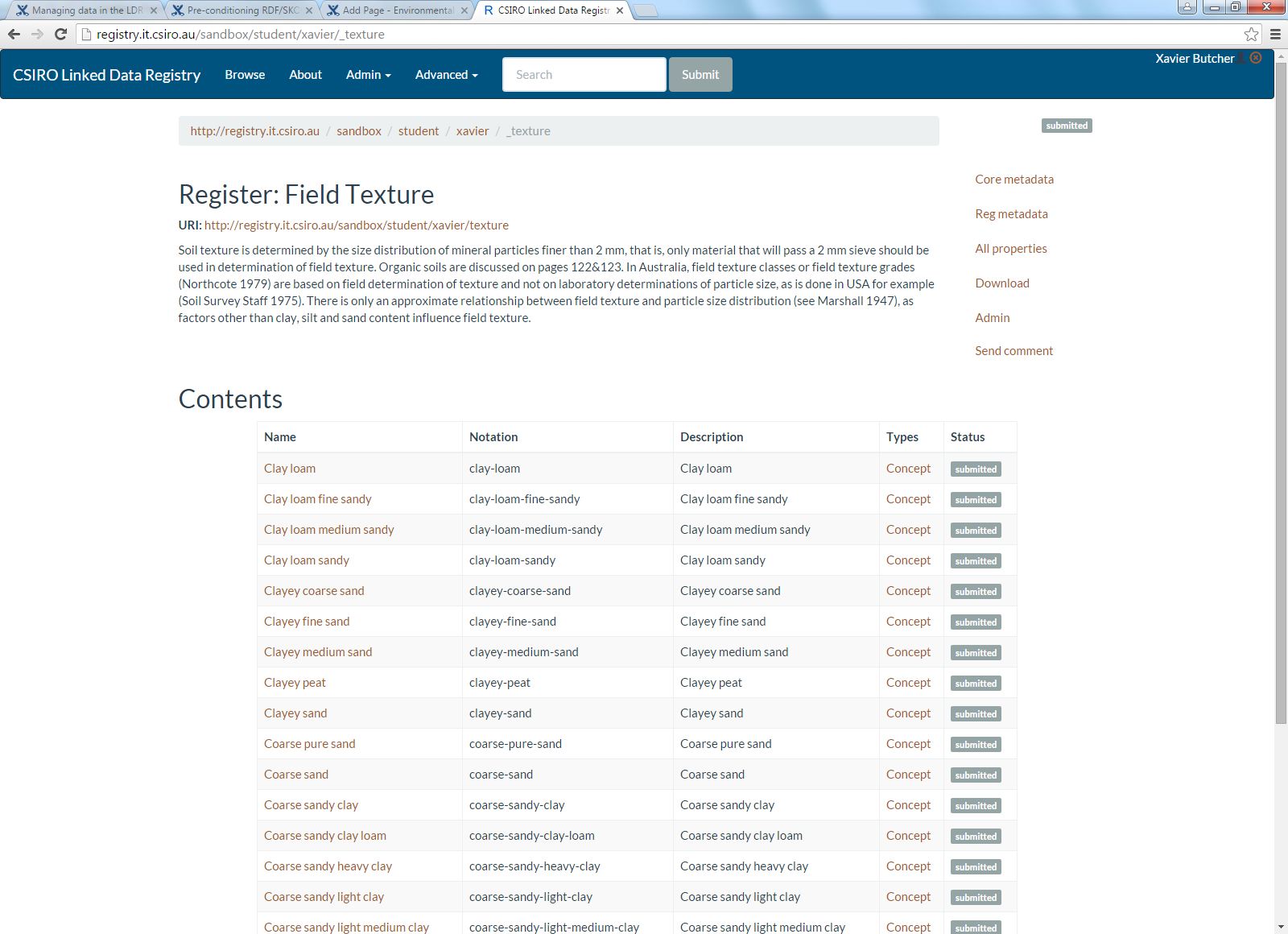


Figure 28: Contents Upload

1. You will notice the register and all the contents say submitted. This means only you can see them. To 'turn on' your vocabulary, you must make the status of both the register and the contents "Experimental".
2. This is done by clicking on admin, then 'set status'. A box should come up; click on the yellow experimental button. Do this process again for the 'set contents status' button. It should look like this:

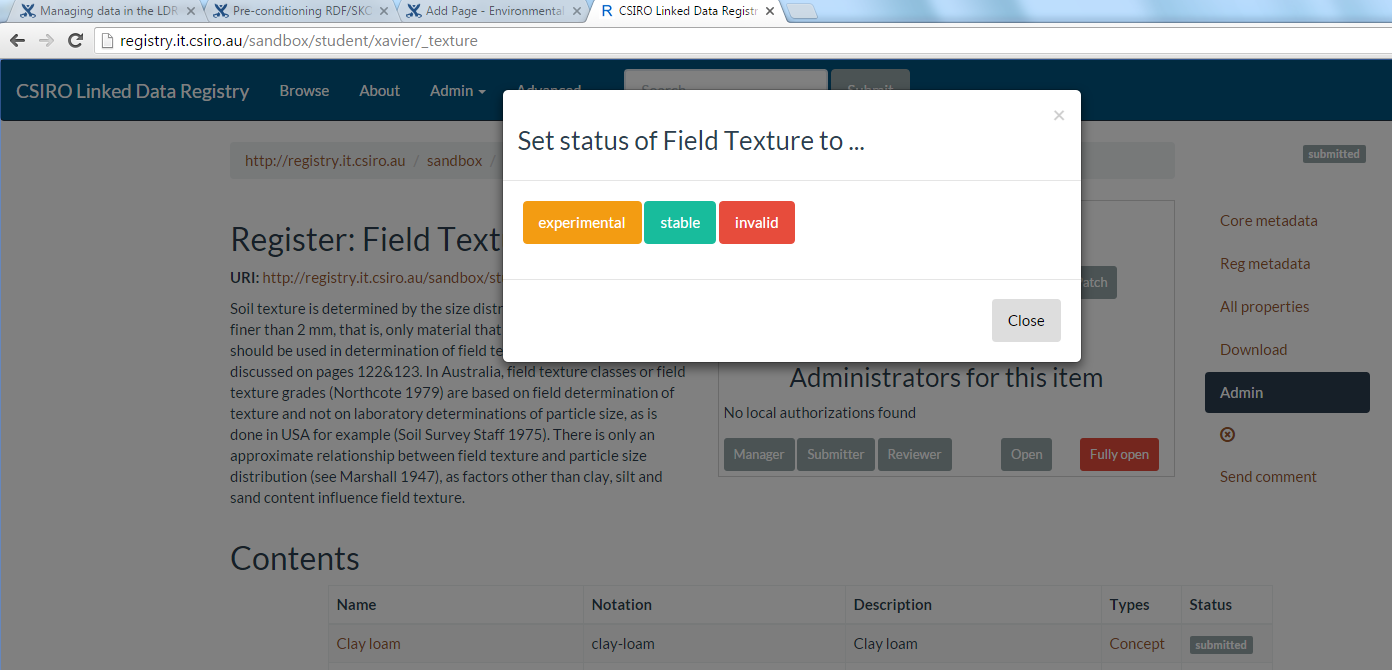


Figure 29: Set Status

1. Your vocabulary is now complete!

# Results

I have completed 27 vocabularies and counting. These are:

|  |  |  |
| --- | --- | --- |
| CF Abundance | Soil Taxonomy | Lab Methods |
| Plasticity Degree | Carbonate Effervescence | Moisture Status |
| Stickiness | CF Distribution | CF Size |
| ASC | CF Shape | Crack Width |
| ASC Confidence | Soil Water Status | Pore Abundance |
| ASC Family | Texture Qualifier | Root Abundance |
| Drainage | Texture | Root Size |
| Technical Reference | Water Repellence | Strg Class |
| Plasticity Type | Lab Units | OP Methods |

These can all be seen in full on the CSIRO Linked Data Registry.

# Discussion

## Key Issues

When exporting from CSV to RDF123, some CSV files inexplicably exported as Tab Separated Values instead of Comma Separated Values. This is means RDF123 won’t be able to read it. All that is needed to fix it is opening the CSV in notepad, and using the find and replace function to replace tabs with commas.

When trying to get TopBraid to prefix a URI, often times it would not prefix correctly because of an extra slash at the end of the URI. I don’t know how this error comes about, but it’s easy enough to fix – just delete the slash.

When adding the @en tag to a prefLabel, it is supposed to sit outside of the quotations of the label itself. However RDF123 recognises the whole thing as a label and puts it in quotations. There is no permanent fix I can see for this as that is how the software was designed. The temporary fix is just using the find and replace function in notepad to remove them.

Some vocabularies had information regarding the range of the field. For instance Crack Width. The issue was how to store that information in a vocabulary. I chose to use the xsd:minInclusive and xsd:maxExclusive properties, but it is possible there is a better solution.

The issue of copyright must also come into consideration. Many of the definitions of the concepts were taken verbatim from the soil handbook, so permission must first be granted by the publisher before this can proceed.

In the LDR, for some reason the system would take the arbitrary notation I had used and make that part of the URI instead of the proper name I had given it. Eg: /xavier/abund instead of /xavier/cf-abundance.

There were times when a missing definition needed to be added for a particular method or element or concept. Having done no work on soils, and only a mediocre chemistry and physics background, this was challenging – even if complex textbooks were available.

There were some instances where 2 tables had identical or very similar data. I chose to merge these tables where feasible as this not only creates less work for me, but is much more concise and causes less confusion.

There is also an issue with URI location. My vocabularies were initially stored in a sandbox registry – in case the intern stuffed something up. There was some confusion about whether the URI of an object should be pointing to the current sandbox location, or to the anticipated true location. The latter was chosen.

## Specific Issues

Blag

## Future Developments

Blaf

# Conclusion

Blah

# Acknowledgements

Blah

#### Heading 4

## Formatting [style = Heading 2]

The CSIRO font for all Office programs is Calibri. The styles of the headings vary in font size, weight and colour, and have numbered or not numbered options. Space before and after headings is included in the style, and no blank lines should be inserted between elements.

Main text is 12 pt Calibri, left justified [style = Body text]. Main text must flow in one column spreading over the full width of the page. Space between paragraphs is included in the style, and no blank lines should be inserted between paragraphs.

Two examples of bulleted lists follow. If the dot points are full sentences:

* Each should start with a capital letter. [style = List Bullet]
* Each should end in a full stop.

If the dot points are sentence fragments, follow these conventions:

* no capitals at beginning
  + second level bullet [style = List Bullet 2]
  + second level bullet [style = List Bullet 2]
* third level bullet [style = List Bullet 3]
* no ‘and’ after the second to last dot point
* no full stop at the end of each dot point (except for the last).

An example of a numbered list follows [style = List Number]. These are basic typographic rules for the reports as a whole.

1. In general, all text is aligned on the left, with ragged edges on the right. Do not justify or centre text.
2. second level numbered list [style = List Number 2]
3. second level numbered list [style = List Number 2]

* third level numbered list [style = List Number 3]

1. In general, use sentence case, unless otherwise specified (e.g. subheadings). Capitalise only the first letter of the sentence and proper nouns.
2. Between the full stop and the start of the next sentence, use only one space, not two.
3. Use single quotation marks, not double. Use single quotation marks for quoted speech, and to signal unusual usage.
4. Use an en dash to denote a range of values, e.g. ‘35%–50%’. Do not add spaces on either side.
5. Use an en dash with one space on either side to indicate a break of thought or a side note, e.g. ‘A mix of three of them – myristic, palmitic and palmitoleic acid – triggers a quick upsizing in heart muscle cells.’.
6. Scientific names and titles are set in italics of the same weight, e.g. ‘Homo sapiens first appeared …’. [style = Italics]

Example of a feature box heading [style = Boxed heading]

Example of a feature box body text [style = Boxed text]

### Hyperlinks [style = Heading 3]

All hyperlinks must include a ScreenTip. You can see the ScreenTip text if you hover over the hyperlink below. To enter the ScreenTip, right-click the hyperlink, click ‘Edit Hyperlink’, click the ‘ScreenTip’ button, and then enter ‘[Organisation Name] website’. For example: <[www.australia.gov.au](http://www.australia.gov.au/)> (ScreenTip text is ‘Australian Government website’) and <[www.anu.edu.au](http://www.anu.edu.au/)> (ScreenTip is ‘Australian National University website’). In reference lists, the ScreenTip for hyperlinks should be ‘Hyperlink to: ‘[Title of report]’’. For example: <[publications.csiro.au/rpr/download?pid=csiro:EP132686&dsid=DS5](https://publications.csiro.au/rpr/download?pid=csiro:EP132686&dsid=DS5)> (ScreenTip is ‘Hyperlink to: Water resource assessment for the Great Artesian Basin’).

### Equations

Equations should be numbered consecutively as they appear in the text with Arabic numerals, for example ‘(1)’. Equations must be typed, not hand printed. Go to ‘Insert’ > ‘Equation’ to generate equations within Word. The style includes spacing above and below [style = Equation]. Alternative text must be inserted for equations. This website should be used to generate alternative text for equations: <[www.wiris.net/demo/editor/demo/en/](http://www.wiris.net/demo/editor/demo/en/)>.

 (1)

### Figures and photographs

The resolution of figures should be minimum 300 dpi. The preferred file format is Windows enhanced metafile for figures. This can be achieved by:

* copy the figure in the source file (e.g. Excel)
* in Word, choose ‘paste special’, then click ‘Picture (Enhanced Metafile)’ and OK
* right-click on the image and select Text wrapping and choose ‘In Line with Text’.

Figure numbers and captions appear at the bottom of the figures. Figures should be numbered consecutively with Arabic numerals, in the order in which reference is made to them in the text, e.g. ‘Figure 1’, ‘Figure 2’, etc.

In the figure caption, the figure number is followed by a space. Captions do not end in a full stop, whether they are full sentences or sentence fragments. If required insert a source reference below the figure caption.

Resolution of photographs should be minimum 300 dpi. The preferred file formats are jpg or png for photographs. Do not use any picture styles (rounded corners, drop shadows, frames) on photographs.

All figures require alternative text. To insert a description, right-click the figure, click ‘Size’, and then click the ‘Alt Text’ tab. In instances where it is not possible to provide alternative text, please use the disclaimer: ‘For a description of this image please contact [email address]’.

To add alt text:

* Word 2007 – right-click on your figure, select Size > Alt Text *or* Format Picture > Alt Text
* Word 2010 – right-click on your figure, select Format Picture > Alt Text
* Word 2013 – right-click on your figure, select Format Picture > Layout & Properties tab > Alt Text

Line graph

Figure 30 Example figure, full width (17 cm) [use References/Insert Caption, label = Figure | style = Caption]

Use the caption note to add additional information under the caption, this doesn’t appear in the TOC [style = Caption Note]. To reduce the space between the caption and caption note, select the caption above, go to the Home tab - Paragraph - Line and Paragraph Spacing - in the drop-down click on ‘Remove Space After Paragraph’

1. [Insert text, style = Figure/Table Source]

(a) (b)

Bar graphBar graph

Figure 31 Example figures, half width (8.4 cm) [use References/Insert Caption, label = Figure | style = Caption]

Pie chart

Figure 32 Example figure, full width (17 cm) [use References/Insert Caption, label = Figure | style = Caption]

### Tables

[Table style = Table\_CSIRO]

Table numbers and captions appear at the top of the tables. Tables should be numbered consecutively with Arabic numerals, in the order in which reference is made to them in the text, e.g. ‘Table 1’, ‘Table 2’, etc.

The table itself is left aligned and should fit within the margins. Authors can choose how wide they wish their table(s) to be within the margin: either extending to the margins, or narrower if desired.

For accessibility purposes you must identify the column headers of tables: select the first row of the table, right-click and select Table properties > Row, tick the ‘Repeat as header row at the top of each page’ option. Also try to avoid complex tables, simple data tables without merged/nested cells are preferred and when possible separate the complex tables into simple tables.

In the table caption, the table number is followed by a space. Captions do not end in a full stop, whether they are full sentences or sentence fragments. If required insert a source reference below the table.

Table 1 Example table, full width (17 cm) [use References/Insert Caption, label = Table | style = Caption]

|  |  |  |  |
| --- | --- | --- | --- |
| column heading [style = Columnheading] | column heading | column heading | column heading |
| Text [style = TableText] | Text [style = TableText] | Text | Text |
|  | * Bullet [style = TableBullet] * Bullet | Text | Text |

1. [Insert text, style = Figure/Table Source]

Table 2 Example table, half width (8 cm) [use References/Insert Caption, label = Table | style = Caption]

|  |  |  |  |
| --- | --- | --- | --- |
| column heading [style = Columnheading] | column heading | column heading | column heading |
| Text [style = TableText] | 15.5 | 15.5 | 15.5 |
| Text [style = TableText] | 20.5 | 20.5 | 20.5 |
| Total | 36.0 | 36.0 | 36.0 |

1. [Insert text, style = Figure/Table Source]

Table 3 Example table, full width (17 cm) with coloured row headings [use References/Insert Caption, label = Table | style = Caption]

|  |  |  |
| --- | --- | --- |
| column heading [style = Columnheading] | column heading | column heading |
| Row heading [style = RowHeading] | Text [style = TableText] | Text [style = TableText] |
|  | Text [style = TableText] | Text [style = TableText] |
|  | Text [style = TableText] | Text [style = TableText] |
|  | Text [style = TableText] | Text [style = TableText] |
| Row heading [style = RowHeading] | Text [style = TableText] | Text [style = TableText] |
|  | Text [style = TableText] | Text [style = TableText] |
|  | Text [style = TableText] | Text [style = TableText] |
|  | Text [style = TableText] | Text [style = TableText] |
| Row heading [style = RowHeading] | Text [style = TableText] | Text [style = TableText] |
|  | Text [style = TableText] | Text [style = TableText] |
|  | Text [style = TableText] | Text [style = TableText] |

[Insert blank line if no source]

### Accessibility

Documents that are to be published on the web are required to be accessible to Web Content Accessibility Guidelines (WCAG) 2.0 Level AA. Documents should be provided in both Microsoft Word and Adobe PDF format.

To ensure documents are accessible, these requirements must be followed:

* Use the standard CSIRO report template.
* Use short headings throughout the document, including the title. If necessary, a subtitle could be used.
* Use styles for all text.
  + Use only the recommended styles that are already contained within this template.
  + Table styles are also included.
  + Do not modify the current styles.
* Include alternative text (alt text) for all images, tables, equations and objects.
  + For images, use this statement for the alt text where it is not possible to describe the image: ‘For a description of this image, please contact [email address].’
  + For equations, use this website to generate alt text: <[www.wiris.net/demo/editor/demo/en/](http://www.wiris.net/demo/editor/demo/en/)>.
* All images and objects should be in line with text and not floating.
* Avoid the use of text boxes and columns.
* Tables:
  + Use the recommended table styles contained within the template.
  + Keep tables simple – try to avoid merging and splitting cells and only use simple, single row headers.
  + If complex tables (with merged cells) are needed, add this statement to the table alt text: ‘For a description of this table, please contact [email address].’
  + Specify the table header row by selecting the first row, right-click the table, click ‘Table Properties’, ‘Row tab’ and then tick ‘Repeat as header row at the top of each page’. You must do this for all tables that have a header row, even if they do not flow onto another page. You do not need to do this if you have a table that uses a header column instead.
* All web hyperlinks must include a ScreenTip.
* Do not use empty paragraph returns.
* Do not use double spacing or use multiple spaces for alignment.
* Include a table of contents for large documents.

|  |  |  |
| --- | --- | --- |
|  | | |
| CONTACT US  t 1300 363 400  +61 3 9545 2176  e enquiries@csiro.au  w www.csiro.au  At CSIRO we shape the future  We do this by using science to solve real issues. Our research makes a difference to industry, people and the planet.  As Australia’s national science agency we’ve been pushing the edge of what’s possible for over 85 years. Today we have more than 5,000 talented people working out of 50-plus centres in Australia and internationally. Our people work closely with industry and communities to leave a lasting legacy. Collectively, our innovation and excellence places us in the top ten applied research agencies in the world.  WE ASK, WE SEEK AND WE SOLVE |  | For further information  Insert Flagship/Business Unit name  Insert contact name t +61 0 0000 0000  e first.last@csiro.au  w www.csiro.au/flagship-businessunit  Insert Flagship/Business Unit name  Insert contact name t +61 0 0000 0000  e first.last@csiro.au  w www.csiro.au/flagship-businessunit  Insert Flagship/Business Unit name  Insert contact name t +61 0 0000 0000  e first.last@csiro.au  w www.csiro.au/flagship-businessunit |