This version edited: 24th November 2016

Building a knowledge-base

Documentation for the Tom.bio ID Visualisation Framework

Dr Richard Burkmar
Biodiversity Project Officer
Field Studies Council
Head Office
Montford Bridge
Shrewsbury
SY4 1HW

richardb@field-studies-council.org
Tel: (01743) 852125

Tomorrow's Biodiversity Project
funded by the Esmée Fairbairn
Foundation

1 Contents

2 Introduction

This document is a reference for building knowledge-bases for use with the Tom.bio ID Visualisation framework (referred to in this document as 'the framework'). Most of the examples referred to are from the example 'biscuits' knowledge-base which is supplied with the framework ('biscuits.xlsm').

3 Building a knowledge-base

The framework uses a spreadsheet to represent taxonomic knowledge. The example spreadsheet provided with the framework is an Excel workbook, but you can use any type of spreadsheet you like to build a knowledge-base that can be used by the framework – the only requirement is that it can save to CSV (Comma Separated Value) files – since this is what the framework actually reads.

Each knowledge-base is represented by five CSV files with the following names:

- taxa.csv
- characters.csv
- values.csv
- media.csv
- config.csv

In the example Excel workbook, there is a separate worksheet for each of these. A sixth worksheet – called 'macros' – has a button on it which invokes a macro to save the current workbook and export each of the five worksheets – taxa, characters, values, media and config – to the CSV files that are read by the framework. This makes it easy to generate all the CSVs required by the framework at the push of a button (but you can save each of them manually if you don't want to use the macro).

Further sections in this document deal with each of the five CSV files (which will be referred to here as 'worksheets').

On all worksheets, placing a hash character ('#') as the first character of the first column in a row effectively 'comments out' the whole row. This is useful mechanism for knowledge-base authors to comment their knowledge-bases or to make rows invisible to the framework without removing them from the knowledge-base.

Note that, in general, text in the knowledge-base is treated case-sensitively by the framework, so, for example, Shape is not the same as shape. So be sure to have regard for case when creating knowledge-bases.

3.1 The taxa worksheet

The example below shows the taxa worksheet from the biscuits knowledge-base.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	**Taxon**	**Shape**	**BiscuitColour**	**Coating**	**Holes**	**Pattern**	**Words**	**SingleDouble**	**FillingColour**	**Width**	**Length**	**ChocTaste**	**TopTen**
2	Rich Highland Shortie	Round	Pale brown	None	Yes	Yes	No	Single	n/a	57	57	none	no
3	Romany Vanilla Cream	Round	Dark brown	None	No	Yes	No	Double	White	47	47	none	no
4	Shortcake	Rectangular	Pale brown	None	Yes	Yes	No	Single	n/a	37	61	none	no
5	Custard Cream	Rectangular	Pale brown	None	Yes	Yes	Yes	Double	White	34	47	none	yes
6	Bourbon Cream	Rectangular	Dark brown	None	Yes	No	Yes	Double	Dark brown	30	62	little	yes
7	Happy Face	Round	Pale brown	None	No	Yes	No	Double	White	47	47	none	no
8	Milk Chocolate Digestive	Round	Pale brown	Chocolate	No	No	No	Single	n/a	58	58	very	no
9	Milk Chocolate Finger	Cigar	Pale brown	Chocolate	No	No	No	Single	n/a	15	78	very	yes
10	Jam Sandwich Cream	Round	Pale brown	None	No	Yes	No	Double	White and red	47	47	none	no
11	Choc Chip Cookie	Round	Pale brown	None	No	No	No	Single	n/a	57	57	moderate	no
12	#Rich's fantasy biscuit 1	Triangle	Dark brown | Pale brown	Toffee	Yes	No	No	Single	n/a	[80-90]	[90-100]	none	no
13	Rich's fantasy biscuit 2	Hexagonal	Pink | Dark brown | Pale brown	Chocolate	Yes	No	?	Double	Dark brown	[80-90]	[90-100]	very	no

3.1.1 General rules for the taxa worksheet

These are the general rules to bear in mind when specifying knowledge on the taxa worksheet:

- There *must* be a column called 'Taxon'. This column holds the names of the taxa as they will appear to users of the visualisations.
- The order of the columns is unimportant (and the 'Taxon' column needn't be the first column).
- Further columns represent attributes of the taxa – e.g. morphological characters.
- No spaces are allowed in the column names.
- The column names are not what appears to users (see section on character worksheet) so you only need to use names that are meaningful to you as the knowledge-base author.
- Apart from the first row, which must contain the column names, each row represents a different taxon.
- A taxon can be included in the knowledge-base but excluded from the visualisations by being 'commented out' with the hash character '#' – as is the case for 'Rich's fantasy biscuit 1' in the biscuits knowledge-base.
- The values in the cells of a morphological character column contain the character state values for each of the taxa.
- There's no practical limit on the number of taxa (rows) or characters (columns) that can be specified.
- There are two broad types of character state values: text and numeric.

3.1.2 Text character state values

Text values are expressed with normal alphanumeric characters. In fact you can use most keyboard characters in text state values (but they have not all been tested exhaustively). However, one character – '|' – has a reserved meaning. It is interpreted as '**or**' by the framework so you should only use this to separate multiple alternatives for a given taxon/character. An example in the biscuits knowledge-base is the value for the 'Colour' character and 'Rich's fantasy biscuit 2' taxon which is specified as 'Pink | Dark brown | Pale brown'. That means that this type of biscuit can be either pink, dark brown or light brown and would score as a match for that character if the user specified either of those three colours with the frameworks character state input tools. (See also the section on 'special character state values'.)

There is no equivalent 'and' character. So you can't specify, for example, that a biscuit is both pink and dark brown in a way that would match a multiple selection by the user (see section on the character worksheet for an explanation of multiple state selection). Instead you would have to

specify ‘Pink and dark brown’ as the character state – the word ‘and’ would have no special meaning to the framework, but this character state would be considered by the framework to be completely different from both ‘Pink’ and ‘Dark brown’ and presented to the user as a separate possible value in its own right.

Normally, when constructing a knowledge-base, if you find yourself wishing to use ‘and’ a lot to combine character states, it is an indication that you might be trying to specify a character which would be better disaggregated into two or more different characters.

3.1.3 Numeric character state values

Numeric values are expressed in any of the commonly accepted ways (except scientific notation), so any of the following examples are valid: 1000, 1, 1.0, 0.01 and 123.123. It is frequently necessary to provide a numeric range and this is accomplished using the notation ‘[n1-n2]’ where n1 and n2 are numbers specified in a valid format. Examples from the biscuits knowledge-base include the values of the ‘Width’ and ‘Length’ characters for ‘Rich's fantasy biscuit 2’. That means that the size of ‘Rich's fantasy biscuit 2’ is variable and will match against any number specified by a user which is within the specified range (unusual for a biscuit but common for biological organisms!).

3.1.4 Special character state values

As well as the text and numeric state values indicated above, there are three other ‘special’ values that can be used in the character/taxon character state spreadsheet cells. These are:

- ‘’
- ‘?’
- ‘n/a’

The first one is an empty cell, i.e. no value at all. This is allowed in a knowledge-base and is treated as a missing value. Missing values neither score for or against a taxon when matching against user input. Obviously the performance of knowledge-bases with lots of missing values is poorer than those that are well-populated, but it is imperative that visualisations can operate ‘normally’ on knowledge-bases with missing values so that knowledge-base authors can see progress as their knowledge-bases mature.

The second – the question mark – also denotes missing data and is treated in exactly the same way as an empty cell by the framework. However it can be used by the knowledge-base author as a marker in the knowledge-base, for instance as a reminder that some work is needed on that character state.

The final one – ‘n/a’ – represents ‘not applicable’. This should be used for characters that are not applicable for a given taxon. For example in the biscuits knowledge-base the value ‘n/a’ is used for the character ‘FillingColour’ for those taxa that have a value of ‘single’ for the character ‘SingleDouble’, indicating that they do not have fillings. If the user specifies a value for the character ‘FillingColour’ then those taxa with a value of ‘n/a’ will score negatively for that character.

3.2 The characters worksheet

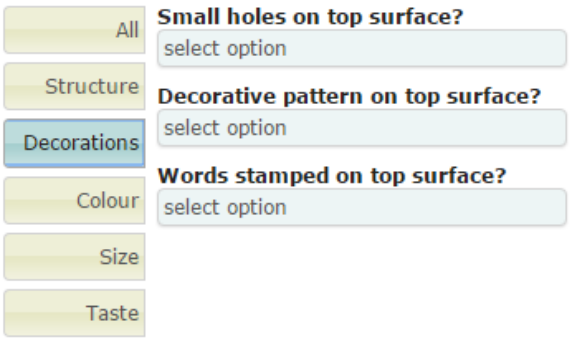
The example below shows the character worksheet from the biscuits knowledge-base.

	A	B	C	D	E	F	G	H	I	J	K	L
1	**Group**	**Character**	**Label**	**Help**	**Status**	**ValueType**	**ControlType**	**Params**	**Weight**	**Strictness**	**Source**	
2	None	Taxon	Taxon									
3	Structure	Shape	Shape	The actual shape of the bisc	key	text	single		10			
4	Structure	SingleDouble	Single or Double Biscuit	Is the biscuit a single layer,	key	text	single		10			
5	Structure	Coating	Coating	Does the biscuit have a choc	key	text	single		8			
6	Decorations	Holes	Small holes on top surface?	If the biscuit has small hole	key	text	single		8			
7	Decorations	Pattern	Decorative pattern on top surface?	Is there any sort of <i>delibe	key	text	single		8			
8	Decorations	Words	Words stamped on top surface?	As part of the decoration, so	key	text	single		10			
9	Colour	BiscuitColour	Biscuit Colour	The base colour of the biscu	key	text	multi		5			
10	Colour	FillingColour	Filling Colour	The colour of the filling whic	key	text	single		5			
11	Size	Width	Width	The width of the biscuit, mea	key	numeric	spin	1,100,1	8	1		
12	Size	Length	Length	The length of the biscuit, me	key	numeric	spin	1,100,1	8	1		
13	Taste	ChocTaste	How chocolatey?	Indicate how <i>chocolatey<	key	ordinal	single		3	1		
14	Other	TopTen	Top ten biscuit?		display						http://metro.co.ul	

Regardless of the columns specified on the taxa worksheet, they will only be considered as true characters by the framework if they have an entry on the characters worksheet – i.e. they appear under the 'Character' column. So every character that you want to use in the visualisations must have a line on this worksheet.

3.2.1 The Group column

The fist column – 'Group' – is used to group characters in the visualisations on the state input controls. The illustration on the right shows the appearance of the state input controls for the biscuits knowledge-base. Here the 'Decorations' tab is selected causing only those characters marked with this group on the characters worksheet to be shown.



If the value under 'Group' for every character is set to 'None', then the framework's state input controls do not group characters at all and the grouping tabs shown in the illustration here are completely hidden. Groups are useful when you are specifying a knowledge-base with lots of characters but can just be a hindrance if there are only a few, so it's useful to be able to specify 'None' in such cases.

3.2.2 The Label column

The 'Label' column is where you specify how the character is actually represented to users of the visualisations. So while the character name, as specified on the taxa worksheet (and in the 'Character' column of the character worksheet) is short and succinct and convenient for you as a knowledge-base author, it may not be that meaningful to a user of the visualisations. So in the 'Label' column you indicate the text that will appear to a user in place of the character name used in the knowledge-base. If you like, you can frame these labels as short questions – as in the example for the decoration characters in the biscuits knowledge-base. Alternatively you can use something simpler and leave the questions for the next column we will discuss – the 'Help' column.

3.2.3 The Help column

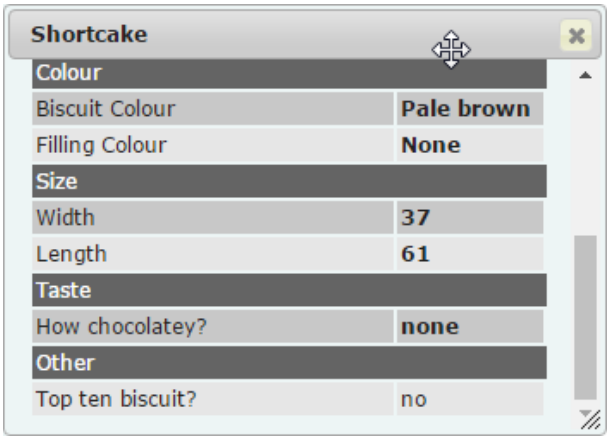
The 'Help' column is where you can specify a much lengthier explanation of the character – including instructions on how to use it – to users of the visualisations. When you specify text in this column it is presented to user when they click on the character's label in the state input controls as shown on the right. You can use plain text in this column, but you can also use simple HTML mark-up if you like, e.g. to embolden or italicise text.

Note that for numeric characters, you should indicate to the user what units the character is specified in. Sometimes you may want to do that in the 'Label' column, but more usually you would do it in the 'Help' column.

3.2.4 The Status column

The 'Status' column is where you can indicate how the character is to be treated by the visualisations. There are currently only two recognised values for this: 'key' and 'display'. A character with any other value in this column, or no value, will be ignored by the visualisations. (The exception is the 'Taxon' character which must appear on the characters worksheet, but which does not need to have a value specified here.)

If you specify a value of 'key' in the 'Status' column, the character against which you specify it will be included in the character state input controls for the visualisations which use them (e.g. multi-access keys). This is the usual state of affairs, but sometimes you might specify a character that you want users to be able to see – because it contains useful information – but not use for character state input. For such characters, use a value of 'display'. In the biscuits knowledge-base, the 'TopTen' character is specified with a status of 'display', so it does not appear in the state input controls, but users can see its value elsewhere in the visualisations, for example when clicking on a taxon name in either of the two multi-access key visualisations. Doing this invokes a dialog showing the knowledge-base values for the taxon as shown in the illustration on the right. Here you can see values for all characters including those that can be used for character state input (values in bold) and those which are display only (values in normal font weight).



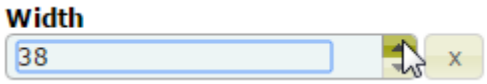
3.2.5 The ValueType column

Under the 'ValueType' column you specify the data type of the character. In the section on the taxa worksheet we noted that there were two broad types of character state values – numeric and text. In fact it can be a little more nuanced than this because text values can be treated as ordinal values by specifying a value of 'ordinal' in this column. So the three valid values for 'ValueType' are 'numeric', 'text' and 'ordinal'. The difference between text values and ordinal values is that the latter

can be ranked. So, for example, in the biscuits knowledge-base, the 'ChocTaste' character is specified as ordinal, because the values that can be taken by this character – 'none', 'little', 'moderate' and 'very' can be ranked, as here, in order of increasing chocolatiness! The values are still specified as text, but the framework will assign ranks to each of the values. (That part is specified on the values worksheet.)

3.2.6 The ControlType and Params columns

The 'ControlType' column is used to specify the type of input control used by the character state input controls on the visualisations. Currently this can only accept a one value – 'spin' – for characters with a 'ValueType' of 'numeric'. So any character with a 'ValueType' of 'numeric' must also have a 'ControlType' of 'spin'. This creates a 'spin' input control like that shown on the right. For each spin control you specify, you also need to provide a value in the '**Params**' column to indicate the minimum and maximum values that the spinner will travel between when you use the up and down arrows and an increment value that indicates how much the value with increase/decrease when you click the up/down arrows. The params value is specified in the form 'min,max,incr' – with no spaces. So the value for the character 'Width' in the biscuits knowledge-base – '1,100,1' – indicates that the control will spin between the values of 1 and 100 and increase/decrease by a value of 1. Note that fractional values are permissible for any of these values, e.g. '5.0,15.0, 0.25'. Currently, this is the only use of the 'Params' column.



The 'ControlType' column can accept one of two values – 'single' or 'multi' – for characters with a 'ValueType' of 'text' or 'ordinal'. Both 'single' and 'multi' control types are drop-down lists from which the user can select the appropriate value. The difference between them is that for 'single' controls, only one value can be specified at once whilst for 'multi' controls, more than one value can be specified. The illustration on the right shows the 'multi' type control specified in the biscuits knowledge-base for the character 'BiscuitColour'.

Normally it is best to specify 'single' for 'ControlType' – it makes for a clearer specification of knowledge and more transparent working of the visualisations. But it can be handy to allow the user to 'hedge their bets' for text characters that are particularly variable or subjective such as colour. (See separate doucment for details on the scoring/matching mechanism.)

3.2.7 The Weight column

Each character with a 'Status' of 'key' can be given a value in the 'Weight' column of between 1 and 10 to indicate its relative importance/reliability as an identification character. The matching scores for each character are adjusted by this value to reflect this relative importance. (See separate doucment for details on the scoring/matching mechanism.)

Your most reliable characters should be given a weighting of 10 and the weighting of all the others specified relative to this. In the biscuits knowledge-base, the characters with a weighting of 10 are 'Shape', 'SingleDouble' and 'Words' – all characters with are not variable and very unlikely to be misinterpreted by a user. (There's a good argument for giving the 'Width' and 'Length' characters of the biscuits a 'Weight' of 10 since they are likely to vary very little, however it is considered that a

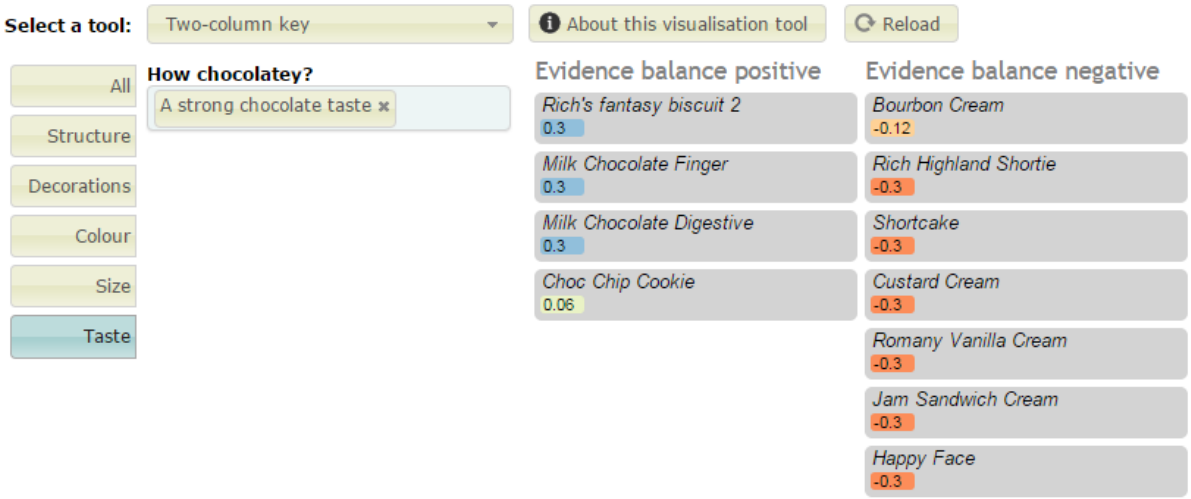
user is as likely to estimate these values as measure them, so they have been given a lower weighting.)

3.2.8 The Strictness column

The 'Strictness' column is only relevant for characters with a 'ValueType' of 'numeric' or 'ordinal'. You can use it to specify how much latitude to give to user-specified values that are not exact matches for a taxa but are relatively close. A strictness value of 10 – the strictest value – requires an exact match between character values specified by the user and those recorded in the knowledge-base for a taxon to score any sort of match for that character. But with a lower strictness value, values that are close but not an exact match can also score. The lower the strictness value, the more latitude there is for a match of some sort.

Imagine a character 'BodyLength' in a knowledge-base for Harvestemen (Opiliones). Suppose that you could populate this field from the literature, but the best data at your disposal was an average body length. You don't have the information required to specify a 'normal' range for the species, so you just use the average body length. Say for species A this was 5.5 mm and for species B 11.0 mm. If you specified a 'Strictness' value of 10 for this character, then a value of 10.5 mm specified by a user for the character would not match either taxon. But with scrictness value of 1, species B would score a partial match. Species A might also score a partial match but the score for species B would be much higher.

Similarly, for the ordinal character 'ChocTaste' in the biscuit knowledge-base, a user specified value of 'very' would only score a perfect match for taxa with a value of 'very'. And if the 'Strictness' value for this character was 10, then those taxa with a value of 'moderate' would score nothing – just like those with a value of 'none'. But with a 'Strictness' value of 1, those taxa with a value of 'moderate' would score less than those with a value of 'very' but more than those with a value of 'none'. (See separate doucment for details on the scoring/matching mechanism.)



The illustration above of the 'two-column key' visualisation working on the biscuits knowledge-base shows how each taxon (buiscuit) is scored when a single character – 'ChocTaste' – is specified by the user as 'strong' (the value 'strong' has been translated to 'A strong chocolate taste' – see the section on the values worksheet for an explanation of that).

The top three matches have the maximum score 0.3 (this is a low maximum because the character has a low weighting of 3). But because this is an ordinal character with a low strictness value of 1, even those values which are not strict matches can score positively, e.g. 'Choc Chip Cookie' which has a value of 'moderate' for this character state.

You can experiment with different values of the 'Strictness' column but, in practice, 10 and 1 are likely to be most useful.

3.3 The values worksheet

The image below shows part of the values worksheet from the biscuits knowledge-base.

	A	B	C	D	E	F	G
1	**Character**	**CharacterState**	**CharacterStateTranslation**	**StateHelp**			
2	ChocTaste	none	No chocolate taste				
3	ChocTaste	little	A slight chocolate taste				
4	ChocTaste	moderate	A moderate chocolate taste				
5	ChocTaste	very	A strong chocolate taste				
6	BiscuitColour	Dark brown		The overall colour of the biscuit is dark brown.			
7	BiscuitColour	Pale brown		The overall colour of the biscuit is pale brown.			
8	Coating	Chocolate		The biscuit has a chocolate coating.			
9	Coating	None		The biscuit does not have a coating.			
10	Shape	Round		The biscuit is round.			
11	Shape	Rectangular		The biscuit is rectangular.			
12	Shape	Hexagonal		The biscuit is hexagonal.			
13	Shape	Cigar	Cigar shaped	The biscuit is cigar shaped.			
14	FillingColour	Dark brown		The bisuit has a dark brown filling.			
15	FillingColour	White		The biscuit has a white filling.			
16	FillingColour	White and red		The biscuit has a filling of two layers - one red and one white.			

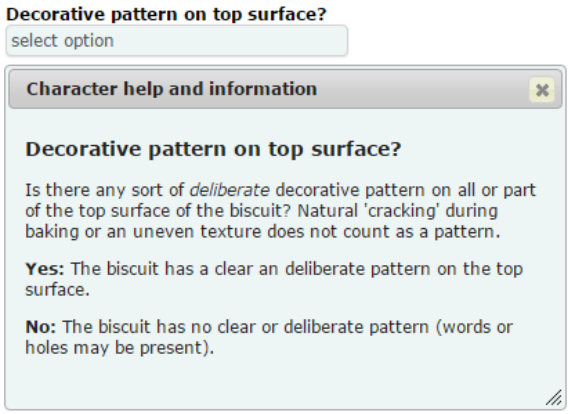
It should be noted right away that a knowledge-base will work without any rows in the values worksheet. You *do not* have to enter every character state into the values worksheet, but there are some circumstances under which doing so is necessary or just desirable. These are the reasons you might want to make entries in the values worksheet:

- To provide help to the user on a particular character state value.
- To translate the character state text that appears on the taxa worksheet to something more meaningful to users of the visualisations.
- To rank character state values for an ordinal character.
- To specify an order for text character state values in drop-down lists.

Each of these situations is dealt with below.

3.3.1 Providing help on character state values

To provide help on character state values, it is first necessary to ensure that the character itself has some help text specified on the characters worksheet. Providing help text for one or more character states gives maximum help to users of the visualisations who are trying to input character state values for a specimen. To provide help for a character state, enter the character and character state values (as they appear in the taxa worksheet) in the columns 'Character' and 'CharacterState' respectively and then enter the help text under 'StateHelp'.



The illustration on the right shows the effect of doing this for the character states 'Yes' and 'No' for the 'Pattern' characters.

Note that like help text for characters themselves, you can use either plain text in the 'StateHelp' column of the values worksheet, or simple HTML mark-up, e.g. to embolden or italicise text.

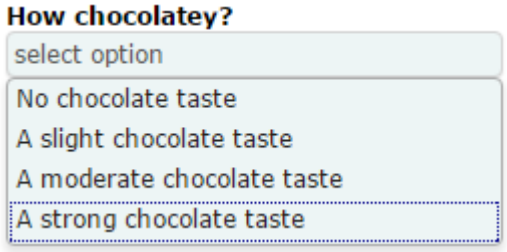
3.3.2 Translating character state values

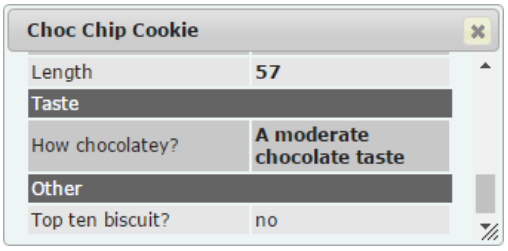
When entering characters state values into the taxa worksheet, wordy text character values can be unwieldy and really make it hard for the author to work with the knowledge-base. And yet, since users of the visualisations see and select character state values, they must be meaningful to them. To help with this there is a facility whereby succinct character state values specified on the taxa worksheet can be translated into more meaningful text that the users actually see.

To provide a translated value for a character state, enter the character and character state values (as they appear in the taxa worksheet) in the columns 'Character' and 'CharacterState' respectively and then enter the translated text for the character state under 'CharacterStateTranslation'.

In the buiscuits knowledge-base, the character states for the character 'ChocTaste' have been provided with translated values. So whilst it is the succinct values of 'none', 'little', 'moderate' and 'very' that appear in the knowledge-base on the taxon worksheet, these are translated, by means of the values worksheet, to the values 'No chocolate taste', 'A slight chocolate taste', 'A moderate chocolate taste' and 'A strong chocolate taste' respectively.

The images on the right show the effects of this translation in the character state selection control and the knowledge-base values display dialog.





3.3.3 Ranking ordinal character state values

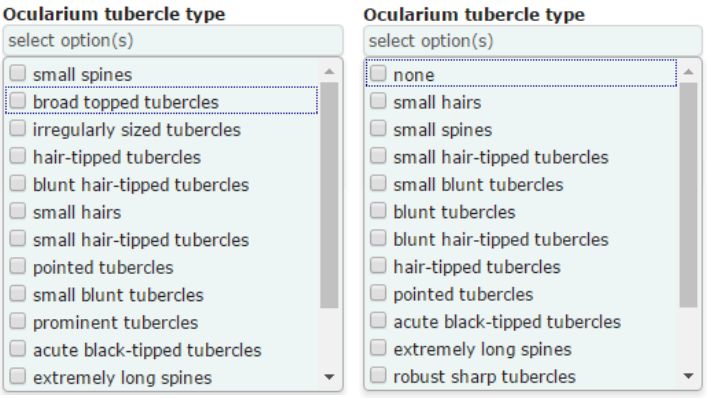
Ordinal characters have been described under the relevant parts of the section on the characters worksheet. But ordinal characters can only operate if a ranking is assigned to each possible character state value and this is done on the values worksheet. A number is not explicitly assigned to define rank; instead, rank is implicitly assigned on the basis of the order in which the values are listed on the values worksheet.

	A	B
1	**Character**	**CharacterState**
2	ChocTaste	none
3	ChocTaste	little
4	ChocTaste	moderate
5	ChocTaste	very

The relevant part of the worksheet is shown on the right for the 'ChocTaste' character in the biscuits knowledge-base. It does not matter whether the values are ranked in 'ascending' or 'descending' order - the effect on scoring will be the same. The important thing is that they appear in rank order. (However the exact order does dictate the order in which they are presented to the user in drop-down lists - see below.)

3.3.4 Specifying order for text character values in drop-down lists

When the framework creates drop-down lists of character states for use in the visualisation's state input controls, the lists are created, by default, simply by adding state values in the order that they are encountered as the taxon worksheet is read from the first row to the last. Therefore the order in which the character states appear in the lists is, essentially, random. Sometimes this is okay – especially if there are very few possible character state values. But sometimes this can lead to ugly and unintuitive lists, as in the case of the immediate right which shows the drop-down list generated by default from a harvestman knowledge-base.



The list on the far right was generated after all the possible character state values for this character were specified on the values worksheet in the order in which the knowledge-base author wanted them to be presented to the user. As you can see, this makes for a more intuitively navigable list.

Remember that unless this is an 'ordinal' character, the order has no effect whatsoever on the scoring of the character, only the order in which the state values appear to visualisation users.

3.4 The media worksheet

The image below shows part of the media worksheet from the biscuits knowledge-base.

	A	B	C	D	E	F	G	H	I
1	**URI**	**ImageWidth**	**Type**	**Priority**	**Caption**	**Taxon**	**Character**	**State**	
2	**# 'Words' character help images**								
3	resources/BourbonTop.jpg	100%	image-local		Words on the top of a Bourbon		Words		
4	**# 'FillingColour' character state help**								
5	resources/CusCreamFilling.jpg	100%	image-local		The filling of a Custard Cream		FillingColour	White	
6	resources/CusCreamXSection.jpg	100%	image-local		White filling in a Custard Cream		FillingColour	White	
7	resources/BourbonFilling.jpg	100%	image-local		The filling inside a Bourbon		FillingColour	Dark brown	
8	resources/BourbonXSection.jpg	100%	image-local		Dark brown filling in a Bourbon		FillingColour	Dark brown	
9	resources/JSCreamXSection.jpg	100%	image-local	2	White and red filling in a Jam Sandwich Cream		FillingColour	White and red	
10	resources/HappyFaceFilling.jpg	100%	image-local	1	White and red filling inside a Happy Face		FillingColour	White and red	
11	**# Taxon images...**								
12	resources/MCC1.jpg	100%	image-local		A photo showing a top down view of a Chc	Choc Chip Cookie			
13	resources/MCC2.jpg	100%	image-local		A photo showing an oblique view of a Chc	Choc Chip Cookie			
14	**# Taxon HTML files...**								
15	resources/DigestivesHist.htm		html-local	2	History of the Digestive	Milk Chocolate Digestive			
16	resources/DigestivesRecipe.htm		html-local	1	Recipe for a Digestive	Milk Chocolate Digestive			
17	resources/Bourbon.htm		html-local		About the Bourbon biscuit	Bourbon Cream			

The media worksheet is where media files are associated with taxa, characters or character states. Framework visualisations can then display these files in appropriate situations. Situations where you might want to specify media files with the current visualisation tools include the following:

- Images to supplement help text for characters for visualisations that use the state input controls (currently the two key visualisations).
- Images to supplement help text for character state values for visualisations that use the state input controls (currently the two key visualisations).
- Images to illustrate taxa for visualisations capable of displaying images (currently all the default visualisations).
- HTML files to provide further information on taxa (currently used by the 'Full taxon details' visualisation).

Each of these situations is dealt with below.

3.4.1 Images to supplement help text

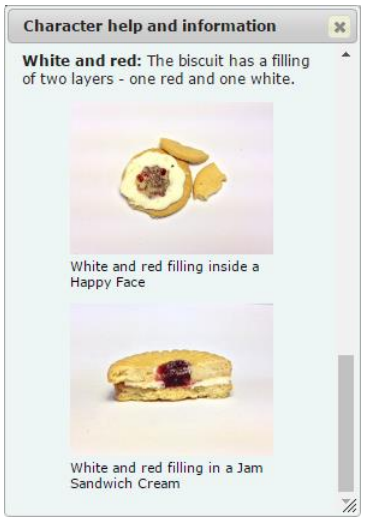
We have seen how to use the characters worksheet and the values worksheet to specify help text for characters and character state values respectively. You can also provide images to be displayed in the appropriate places on the same help dialog by specifying the images on the media tab.

Images should be stored in sub-folders of the folder which contains your knowledge-base (and therefore CSV files). The 'URI' column is used to specify the relative path, from this folder, to the image file.

An image is associated with a character by specifying it in the 'Character' column (the 'State' column must be blank to associate an image with the character itself).

To associate an image with a specific character state value, then values must be supplied for both the 'Character' and the 'State' columns.

It is normally best to use the value of '100%' in the 'ImageWidth' column. This will ensure that the image is resized to fit the help dialog. Without this value, the image is shown at its natural size.



Currently you must specify a value of 'image-local' for the 'Type' column.

The 'Priority' column can be used to specify the order in which images are presented when more than one is specified for a given character or character state value. Lower numbers have the highest priority. Fractional numbers are permitted.

The 'Caption' column is used to specify a caption for an image. It's a useful place to credit copyright holders. You can use plain text or basic HTML mark-up in captions.

The top image on the right shows an image specified in the biscuits knowledge-base for the 'Words' character with no images specified for either of the valid character states.

The bottom image on the right shows the two images specified in the biscuits knowledge-base for the 'White and red' state value for the 'FillingColour' character. Although the Happy Face image is listed under the Sandwich Cream image on the media worksheet, it has a higher priority and is therefore displayed first.

3.4.2 Images to illustrate taxa

Images to illustrate taxa are specified in exactly the same way as those to illustrate character and state values, except that instead of specifying a value for the columns 'State' and/or 'Character' (which must both be left blank), you specify the name of the associated taxon in the 'Taxon' column.

The 'Priority' column is used in the same way. Make sure that if you have more than one image associated with a taxon the highest priority (lowest number) is associated with an image that is suitable to illustrate the taxon under most circumstances since, if only one image is required by a visualisation (e.g. the 'two-column key' visualisation), this is the one it will use.

The image on the right shows an image viewer (included in a number of the visualisations) for a taxon (Rich Highland Shortie) for which three images have been specified in the biscuits knowledge-base. The first image displayed is either that with the highest priority (lowest priority number) or, if no priorities are specified, that listed first in the knowledge-base.

3.4.3 HTML files to provide further information on taxa

If the very phrase 'HTML file' makes you feel queasy, fear not! Think instead of a Microsoft Word file containing detailed information about a taxon. You can use Word's 'Save As' function to save this file with a file type of 'Web Page (*.htm; *.html)' – then you will be able to display it using the framework.

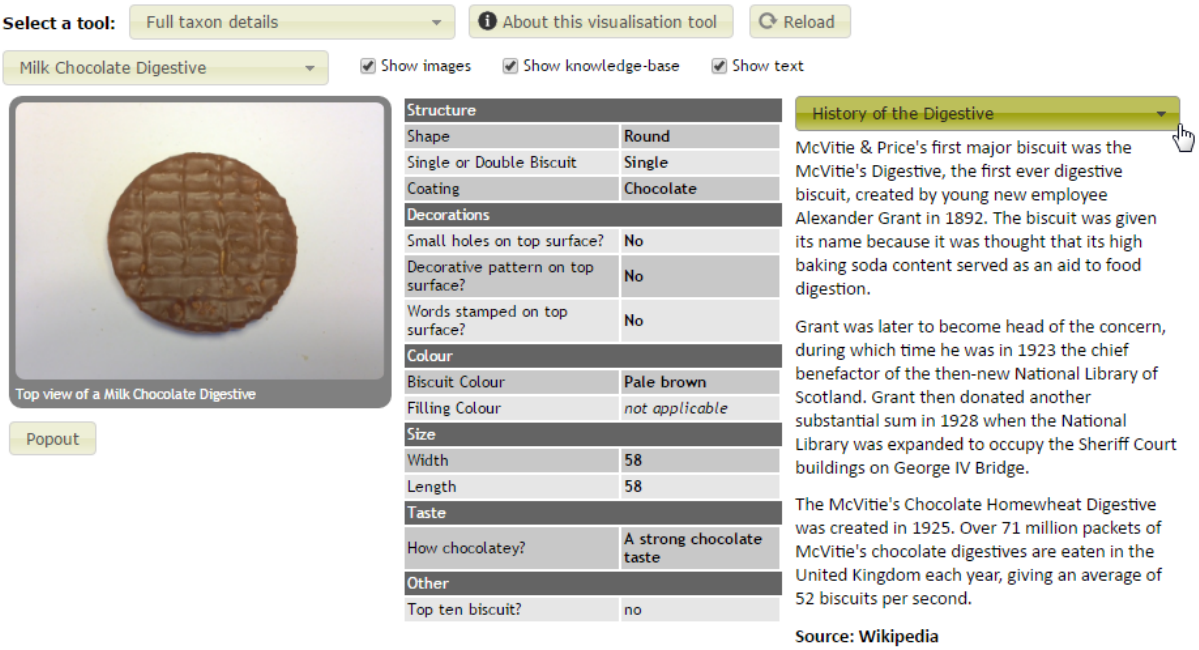
HTML files should be stored in sub-folders of the folder which contains your knowledge-base (and therefore CSV files). The 'URI' column is used to specify the relative path, from this folder, to the HTML file.

If you have saved an HTML file that just contains text, all is very simple. But if you want your HTML file to reference external resources like images, you will need to be careful about how you specify the image URLs. Unless you are very competent with dealing with HTML and websites, it is best not to include images in your HTML files.

Using the media worksheet to specify HTML files associated with taxa is very similar to specifying images associated with taxa. Follow the same steps as for images, but use a value of 'html-local' in the 'Type' column. The value of the 'Priority' column is used by the framework to decide which file to show first if more than one is specified for a taxon.

A few simple examples have been included with the biscuits knowledge-base. Two files are specified for the 'Milk Chocolate Digestive' taxon and one for the 'Bourbon' taxon.

The image below is from the 'Full Taxon Details' visualisation and shows one of the two text files specified for Chocolate Digestive. The other can be displayed by selecting it from the drop-down list (currently showing 'History of the Digestive').



3.4.4 General comments on the media worksheet

Blank rows and those that start with a hash symbol ('#') in the first column – to denote a comment – can be used to organise the information on the media worksheet to help you as a knowledge-bae author keep track of what's on there.

Where a single image is used for several purposes – e.g. illustrate a taxon *and* illustrate a character state value – there must be a separate entry on the media worksheet to specify each way in which the image is used.

3.5 The config worksheet

The image below shows the config worksheet from the biscuits knowledge-base.

	A	B	C	D	E	F
1	**Key**	**Type**	**Mandatory**	**Value**	**Date**	**Notes**
2	**#Tools - those default tools to exclude and non-default tools to include**					
3	excludedDefaultTools	config	no			
4	otherIncludedTools	config	no			
5	selectedTool	config	no			
6	checkValidity	config	no	yes		If value set to 'yes' (lower case) validity checks and reporting will be carried out by the visualisation software.
7	**#Main metadata about the key (e.g. for citing the key)**					
8	title	metadata	yes	Family Circle Biscuits		
9	year	metadata	yes	2016		
10	authors	metadata	yes	Bell, C.		
11	publisher	metadata	no	Field Studies Council		
12	location	metadata	no	Preston Montford, Shrewsbury		
13	contact	metadata	no			
14	version	metadata	yes	1.0		
15	**#Releases - keep track of the release history below this line. Most recent releases at the top. The version code in the 'value' column below this line should match that of the current release above it.**					
16	release	history	no	1.0	22/11/2016	Modified by RB to illustrate some of the different features of knowledge representation.
17	release	history	no	0.1	11/11/2016	CB published an initial test version

The config worksheet is used to specify some configuration options for the visualisation framework and metadata about the knowledge-base itself.

The basic format of entries in on this worksheet is simple key/value pairs corresponding to the columns ‘Key’ and ‘Value’. Any keys with the value of the ‘Mandatory’ column set to ‘yes’ must be specified in order for the framework to function correctly – those set to ‘no’ are optional. The ‘Type’ column indicates the type of key/value pair corresponding to the three sections below.

3.5.1 Framework configuration (type ‘config’)

The key ‘**excludedDefaultTools**’ can be used to remove one or more of the default visualisations from the framework when used with your knowledge-base. There are currently four default tools:

- Two-column key (vis1)
- Single-column key (vis2)
- Side by side comparison (vis3)
- Full taxon details (vis4)

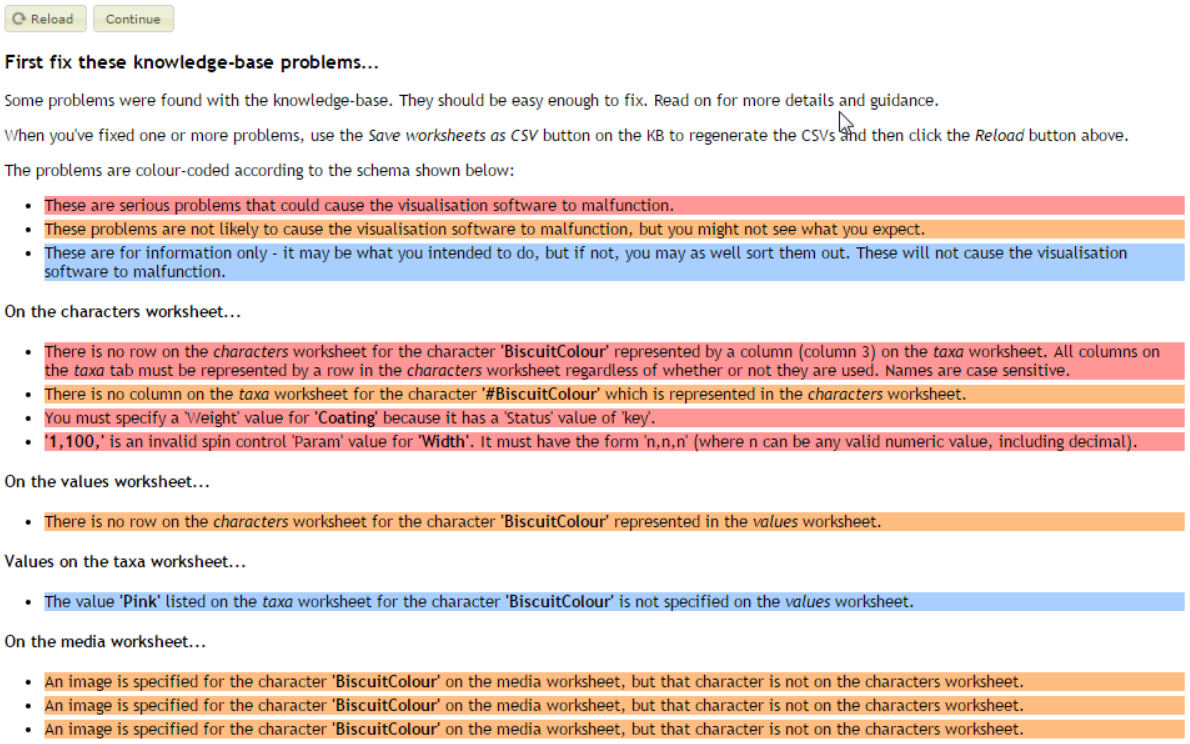
The codes in parentheses can be used in the ‘Values’ column for the ‘excludedDefaultTools’ key to exclude the visualisation when the framework starts. For example set the value to 'vis1, vis3' to remove the 'Two-column key' and 'Side-by-side comparison' visualisations from the ‘Select a tool’ drop-down list in the framework.

The key ‘**otherIncludedTools**’ can be used to specify non-default visualisation tools that you have access to. You are unlikely to need to use this unless you have access to a non-default visualisation, e.g. you are a programmer who has created a new visualisation for the framework.

The key ‘**selectedTool**’ can be used to specify the visualisation that you want the framework to show upon initialisation. By default this is the first included visualisation – ‘Two-column key (vis1)’ if you haven’t excluded it. For example to instruct the framework to initialise with the ‘Single-column key’ visualisation, set the value of this key to ‘vis2’.

The key ‘**checkValidity**’ is a very important one if you are creating a new knowledge-base. Setting the value of this to ‘yes’ instructs the framework to carry out and report on some validity checks on

your knowledge base when it is initialising. The image below shows some example output for these checks.



The validity check is very valuable whilst you are developing a knowledge-base and should always be enabled. However once you have developed your knowledge-base, you may want to set the value of this key to 'no' in order to speed up the initialisation of the framework.

3.5.2 Knowledge-base metadata

Knowledge-base metadata provides your users some important information about your knowledge base. It is used to build a citation for users of your knowledge-base.

The citation is available from the 'Get citation text' option of the 'Select a tool' drop-down list in the framework.

#Main metadata about the key (e.g. for citing the key)			
title	metadata	yes	Family Circle Biscuits
year	metadata	yes	2016
authors	metadata	yes	Bell, C.
publisher	metadata	no	Field Studies Council
location	metadata	no	Preston Montford, Shrewsbury
contact	metadata	no	
version	metadata	yes	1.0

The metadata key values for the biscuits knowledge-base shown in the illustration here in the following citation:

result

Bell, C. (2016) *Family Circle Biscuits* (Version 1.0) [Knowledge-base] (for Tom.bio Framework for ID visualisations). Field Studies Council. Preston Montford, Shrewsbury. Accessed Wed Nov 23 2016. http://localhost:54236/general.html

Citation text is important because it allows people who make determinations based on your knowledge-base to reference it when required (e.g. for the purposes of verification).

The keys 'title', 'year', 'authors' and 'version' are mandatory whilst 'publisher', 'location' and 'contact' are optional. The value of the 'author' key will be used 'as is' in the citation so it is best to use a format, e.g. Surname, initial., that is most frequently used in citations.

3.5.3 Release history

You can use 'release' keys to keep track of the release history of your knowledge-base. You can include any number of release keys – the value of each of them should be the version number of a milestone version of the knowledge-base. There are a couple of extra columns associated with 'release' keys that you should provide values for: 'Date' and 'Notes'. The date should correspond to a release date and you can use the notes to give a very brief description of the release. Unlike the other keys on the config worksheet, the 'Date' and 'Notes' column values are reported by the framework: a release history is displayed to users when then select the 'About the knowledge-base' option from the 'Select a tool' dropdown list (see below).

Knowledge-base revision history

Current version: 1.0

Date	Version	Notes
22/11/2016	1.0	Modified by RB to illustrate some of the different features of knowledge representation.
11/11/2016	0.1	CB published an initial test version

The release history is simply presented in the order the keys are entered into the knowledge-base. It makes sense to have the most recent versions at the top of the list and the oldest at the bottom. Note that the 'current version' in the above report is taken from the value of the 'version' key – not the most recent 'release' key – so you should make sure that the value of the 'version' key always matches the value of the most recent 'release' key.

3.6 Including an information file about your knowledge-base

When a user selects the 'About the knowledge-base' option from the 'Select a tool' dropdown list the framework looks for an HTML file called 'info.html' in the same folder that the knowledge-base CSV files are found. If it finds the file, it is rendered before the release history table generated from the config worksheet (see previous section) as shown below for the biscuits knowledge-base.

The Biscuits knowledge-base

This knowledge-base was developed by Charlie Bell. It is a 'template' knowledge-base designed to demonstrate some of the features of the Tom.bio Framework for ID Visualisations. Most of the examples in the documentation describing how to build a knowledge-base refer to this knowledge-base. If you are building a completely new knowledge-base, consider making a copy of this template as a starting point for your work.

Knowledge-base revision history

Current version: 1.0

Date	Version	Notes
22/11/2016	1.0	Modified by RB to illustrate some of the different features of knowledge representation.
11/11/2016	0.1	CB published an initial test version

If you want to include external resources in your 'info.html' file, they need to be included in sub-folders of the folder containing your knowledge-base. Furthermore, you need to reference them in a particular way in your knowledge-base as shown below for the Tom.bio logo:

```
<img src="##tombiokbpath##resources/tombio-logo.png" style="float: right; padding: 0 0 10px 10px" />
```

An important part in the above HTML image tag is the '##tombiokbpath##' token. The framework replaces this token with the URL of the knowledge-base folder. That means that the knowledge-base can be moved around – e.g. from local computer to a web-server – without having to edit this file every time.

3.7 Dealing with sexual dimorphism

You have a couple of options for representing knowledge for sexually dimorphic organisms. The first option is to create a separate line on the taxa worksheet of your knowledge base for each sex – sometimes referred to as a 'taxon-sex'. So, for example, if biscuits were sexually dimorphic, you would have one line on the taxa workbook for 'Shortcake male' and one line for 'Shortcake female'. An advantage of this approach is its simplicity and a disadvantage is that you need to maintain two lines in your knowledge base for each sexually dimorphic species, repeating much of the information that is common to both sexes.

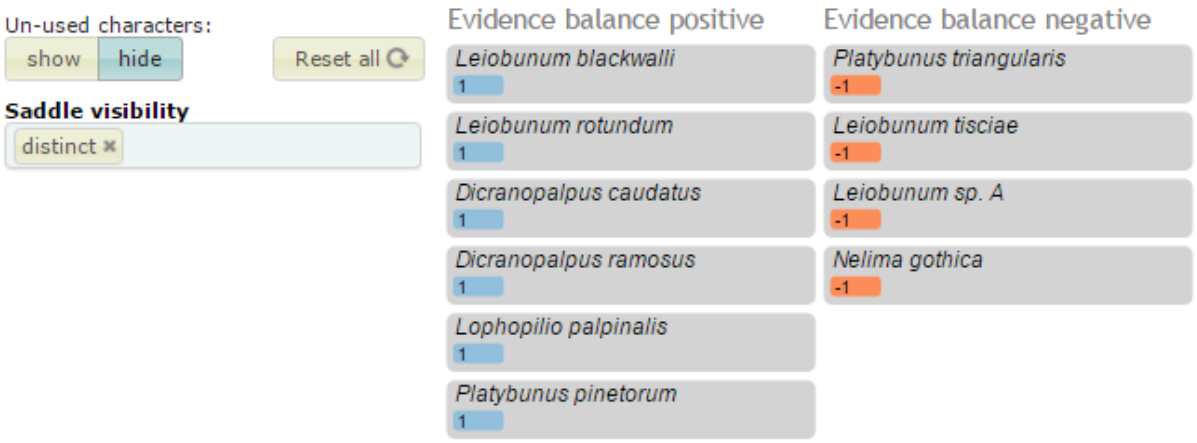
	A	M
1	Taxon	Saddle
22	Platybunus triangularis	indistinct
23	Platybunus pinetorum	indistinct| distinct
24	Lophopilio palpinalis	indistinct| distinct
25	Dicranopalpus ramosus	no (m)| distinct (f)
26	Dicranopalpus caudatus	no (m)| distinct (f)
27	Leiobunum rotundum	no (m) | indistinct (m)| distinct (f)
28	Leiobunum blackwalli	no (m)| distinct (f)
29	Leiobunum tisciae	no| indistinct
30	Leiobunum sp. A	no| indistinct
31	Nelima gothica	no (m)| indistinct (f)

The framework also supports a second approach which is to tag character state values on the taxa worksheet with sex where the value is specific to one sex or another. The example right shows part of the taxa worksheet for a knowledge-base on harvestmen showing how saddle visibility is represented for a number of different species and sexes within a species.

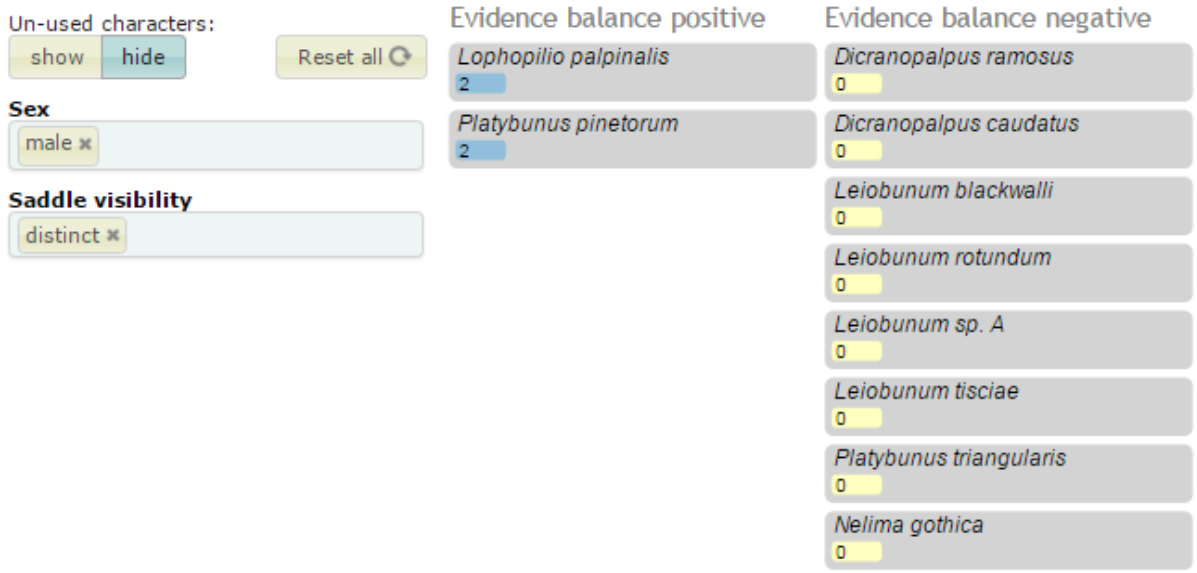
If you represent sexually dimorphic characters in this way, then you need to have a character called 'Sex' (with an uppercase 'S') and the possible values must be 'male' and 'female' (all lower case). This text-type character must have a 'ControlType' of 'single' on specified on the characters worksheet. Users will be able to specify sex just like any other morphological character.

To understand how this all works, consider the following scenarios for the harvestman knowledge-base shown above.

Scenario 1: user specifies a saddle visibility as 'distinct' and does *not specify a value for sex*. The result (for the 'Two-column visualisation') is shown below:



Scenario 2: user specifies a saddle visibility as 'distinct' *and specifies sex as 'male'*. The result (for the 'Two-column visualisation') is shown below:



If no sex is specified, then it is as if no sex tags are present in the knowledge-base - *the values are still there, but without the sex tags*. They only come into play if a user specifies a sex.

3.8 The macros worksheet

At the start of this document the 'Save worksheets as CSV' macro was described. There is another convenience tool on the macro worksheets that you might find useful – the 'Get unique values' macro. This macro was developed to ease the process of extracting unique values for a character from the taxa worksheet for use on the values worksheet.

To get the unique values for a character, select the cells from which you want to extract the values on the taxa worksheet and then go to the macros worksheet and click the 'Get unique values' button – the unique values are copied to the computer's buffer and you can simply paste them anywhere you like. (Don't select an entire column by clicking on the header – the macro won't work if you do that.)

The macro takes care of sorting out cells where multiple values are specified with the 'or' character ('|') and it also strips out any '(m)' and '(f)' sex suffixes. Special values (e.g. 'n/a' and '?') are ignored.

To use any of the macros in the knowledge-base, you need to ensure that your Excel security settings enable you to do so.

3.9 Using other spreadsheet features

You are free to use whatever features of your particular spreadsheet provider (e.g. Excel) that are available to you when constructing a knowledge-base – e.g. formulas, data ranges, comments, etc. None of these should interfere with the production of the CSV files and it is the CSV files that the framework actually reads.

In Excel, for example, you can create as many extra worksheets as you like containing supplementary information to help you as a knowledge-base author. For example if you are converting an existing spreadsheet of knowledge to the framework's format, it is convenient to copy the existing spreadsheet, as is, into extra worksheets. They won't interfere with the framework.

That's an advantage of making use of a spreadsheet format for representing knowledge – it leverages all your existing experience of using spreadsheets and managing information within them.