

Watermark Identification in Rembrandt's Etchings (WIRE) Project:

Porting Instructions Manual

Fall 2017

By: Katrina Ferreira and Amanda House

The purpose of this Porting Manual is to instruct WIRE students and others who wish to use it on the proper procedure (as of this date) through which to assist the porting of old branches and their files to the new WIRE website, and to streamline the coding of new branches. Porting, as a process, refers to migrating the old files and their respective coding, as well as new branch files and data, to the new coding system of the WIRE website. Doing so will replace the previously cumbersome system of branch coding, which required excessive nesting of folders, non-standardized file-naming procedures, and manual editing of each file in case of a change or mistake. In its place will be a new system, with only one folder for each branch (instead of multitudes), a standardized file-naming procedure, and no manual editing of each individual file's code in the case of changes or mistakes. This Manual is meant to be a comprehensive guide to the porting process and its numerous components, especially in addressing usability and knowledge gaps for non-coders.

Table of Contents:

- [Step 0 – Pre-Porting](#)
- [Step 0.1 – Understanding and Using CSV Files](#)
- [Step 1 – Building Blocks of Porting: Short Names](#)
- [Step 2 – Building Blocks of Porting: Branch Tree](#)
- [Step 3 – Building Blocks of Porting: File Folders](#)
- [Step 4 – Porting: Questions Files](#)
- [Step 5 – Porting: Watermarks Files](#)
- [Step 6 – Porting: Further Utilization in Code](#)

Step 0 – Pre-Porting

Before porting, you must first ensure that your watermark branch is ready. The porting process requires the use of a branch with a branch diagram. If you do not already have a branch diagram for your watermark branch, please complete one before beginning the porting process. The branch diagram will become the jumping-off point for porting the watermark branch into the new WIRE website.

In order to create your branch diagram, you must have clearly and specifically worded questions, with terms that are understandable and defined where needed (and/or defined in the WIRE glossary). For use in porting, the questions you use in your branch diagram will be ported into the website, and so it is vitally important that they are as clear and concise as possible, and communicate their meaning in a manner which allows the reader a high degree of certainty in not only what the question refers to, but in what their correct answer should be (relative to the specific watermark subvariant they are using or identifying).

It is important to also take care with proper notation of subvariants, which require the use of periods as opposed to dashes. For example, the correct subvariant notation of “B.a.b.” must be used, rather than the incorrect notation of “B-a-b”.

If you have not already created a branch diagram, you can create one using a number of diagram design tools. Two such tools are Draw.io and Adobe InDesign. These tools will allow you to create a branch diagram with standard shapes and formatting that is also editable if changes will be needed later. In your branch diagram format, try to keep the use of certain shapes, texts, and other design details uniform. For example, if you use a rectangular box shape to indicate a watermark subvariant, use this same shape for all subvariants in your diagram. If you use a circle to indicate questions in the diagram, use circles for all of the questions. And so on.

You should also have already created image files (preferably saved as PNG files) for your watermark subvariants, as well as the illustrative images you will use in your branch such as composition images (yes/no images, full/magnified images, etc.). If you have not yet added annotations (markings or shapes overlaid on the image to point to a specific feature), do so now in any image editing program. When creating your annotations, make sure that they are consistent across all the images you use (e.g. If you use red markings in one image, use red markings in all of your images for that watermark). These image files will later be saved in an “Images Folder” within the overall watermark type folder.

The porting process requires multiple parts in order to work properly. These parts include a standard naming scheme, a list of questions and their answers, and a list of the resultant watermark subvariants. These lists will be entered into CSV form files and used in the website’s code. The following steps will provide you with an understanding of CSV files and how to use them, and how to compile the required parts of porting together to prepare for coding.

Step 0.1 – Understanding and Using CSV Files

CSV “comma separated values” files allow you to save and store data in a table format, which provides useful structure for a larger database such as the ones that we use for coding the WIRE website. CSV files are saved with the file extension “.csv” or “*.csv”.

CSV files are “flat-form” and plain text files, which make them easier for coders and website developers to create and use, and makes the data easy to import across programs, databases, and software. Its data field values are separated or “delimited” by commas. For coders or developers, CSV files will usually be created and edited in a text editor. The current coding/development platform used by WIRE students is GitHub, an online repository hosting service (like the “cloud” for code), which can host and track changes to source code projects across different programming languages. Popular text editors include TextEdit, Notepad, Vim, and Atom. For non-coders, CSV files can most easily be opened and exported in Microsoft Excel.

Creating a CSV File in Plain Text:

A CSV file can be created in any text editor. Non-coders will most likely not be using text editors to create CSV files directly, but it is still essential to understand the basic format of plain text CSV files and the rules of creating them.

After opening your chosen text editor, you will open a new file. The basic format for creating a correct CSV file is to enter the text data and separate each value with a comma, and separate each row, or “record”, with a new line (created by hitting the enter/return button for a line break). Essentially, the use of commas to separate each value acts as vertical columns in a table, and new lines act as horizontal rows in a table.

Example:

Plain Text Format:

Watermark Type , Subvariant
Basel Crosier , B.a.b.
Seven Provinces , A.a.a.

The above plain text CSV file format would result in a table which would look similar to the one below.

Tabulated/Spreadsheet Format:

Watermark Type	Subvariant
Basel Crosier	B.a.b.
Seven Provinces	A.a.a.

Rules to Follow for Creating a CSV File in Text Editor:

There are a number of rules to follow when creating plain text CSV files in a text editor.

- Each row of data (record) must exist on a separate line from the other data rows, and must be separated by a line break.
- Each field value must be separated by a comma (,). Spaces in between field values are considered part of the value and should be taken into account. (In the above example, spaces are left between commas in order to illustrate format more clearly, but would normally be removed if not necessary).
- Within each row there may be multiple fields, but each row must contain the same number of fields. (i.e. If there are two fields in the first row, there must be two fields in every row in the file). If you do not have a value to input for a certain field, you must enter a placeholder, such as an additional comma, space, or other. Most formats will simply use multiple commas (,,) or (, ,) to denote empty fields. Be sure to note how many commas you use and their placement in relation to other field values, as “Basel Crosier, ,” would denote one empty field, while “ , ,” (with nothing preceding it) denotes two empty fields. Multiple commas for empty fields can be utilized at the start of the record, the end of the record, and at any point in between when needed.
- The last field in a record/row must NOT be followed by a comma. If the last field value of a record is an empty field (and therefore denoted by a comma), it should be followed with a space as a placeholder for that value. When you do have a real value for the last field in a record, your record should NEVER look like the following: “Watermark Type, Subvariant,”. The comma at the end suggests a new field will follow, which is incorrect.
- If your record/row is long or consists of many fields, it may take up multiple lines of text, but it is still for all intents and purposes considered one record or row. The ending of a row must be explicit, with a line break to start a new line beginning the next row.
- If there are commas within a field value (ex. When entering “Your watermark is Basel Crosier, E.a.” as a value), the content of the field value should be enclosed in double quotation marks. Take care to always insert both opening marks (“) and ending marks (”) following appropriate punctuation. Without quotation marks, the example used would appear as two separate fields, “Your watermark is Basel Crosier” [,] and “E.a.”.
- Headers or column names, which are strongly suggested, must be the first row of the file. In the example shown, our first row is our header row, with the field values of “Watermark Type” and “Subvariant”, properly delimited with commas.

CSV File in Microsoft Excel:

In order to create a CSV file in Microsoft Excel, you must first launch Excel and create a new Excel file or open a previous one.

If you are creating a new file, enter your data in to the Excel worksheet as you would normally, making sure to input headers if needed and to maintain row consistency. If you will be using a previously-created Excel file, simply open the file now.

Once your desired Excel file is open, click “File” (which should be either at the top of your screen or the top of the Excel window), choose the “Save as” option from the dropdown menu, and select

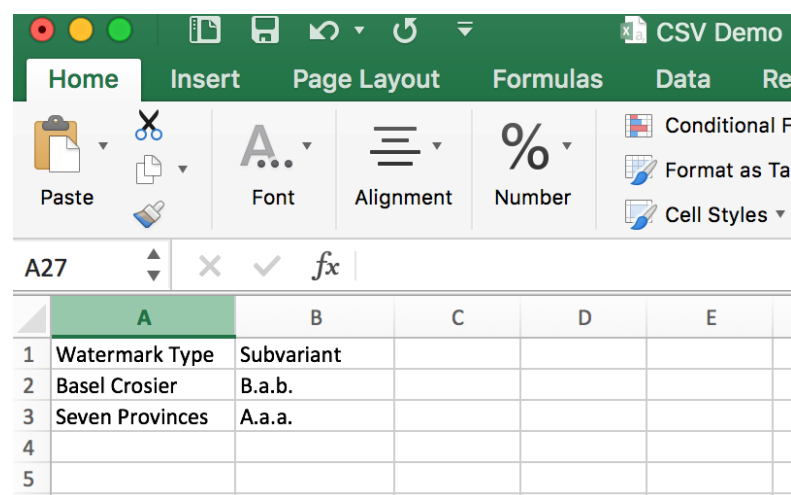
the option which corresponds with a CSV file. Depending on your version of Excel, this will most likely be “Comma Separated Values (.csv)” or an option with the extension (*.csv), or some variation of this.

After saving your file as a CSV file through Excel, you may continue to edit it in Excel.

You may also now open the CSV file in its plain text format, through a text editor. To do so, find the file in your computer files (in the folder to which it was saved, typically “Documents”, “Desktop”, etc.). After locating the file, right-click it to view the options menu, select “Open With”, and select an appropriate text editor with which to open the CSV file.

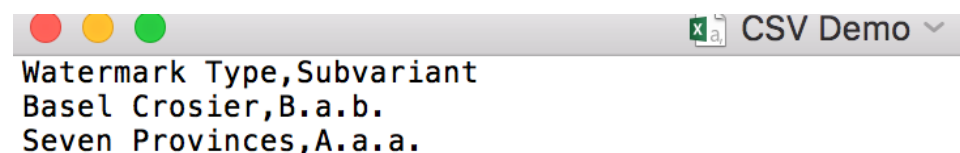
The CSV file in Microsoft Excel and the same CSV file in plain text opened in a text editor (TextEdit), for the sample provided above, are both displayed in screenshots below.

CSV file displayed in Microsoft Excel:



	A	B	C	D	E
1	Watermark Type	Subvariant			
2	Basel Crosier	B.a.b.			
3	Seven Provinces	A.a.a.			
4					
5					

CSV file displayed in plain text in TextEdit:



```
Watermark Type,Subvariant
Basel Crosier,B.a.b.
Seven Provinces,A.a.a.
```

Opening a CSV File:

If you already have a CSV file to work with, it should open automatically in Microsoft Excel when double-clicked. If you are prompted with a choice of programs to open the CSV file, choose Microsoft Excel.

To open a CSV file in a text editor, follow the instructions provided in the preceding paragraphs.

Step 1 – Building Blocks of Porting: Short Names

The first step in obtaining the required parts of porting is to establish a standard naming format. This applies to short names for questions, watermark subvariants, CSV files, and images.

“Short names”, are (as the name implies) an abbreviation of the full subvariant name or full question, for file naming and organizational purposes during coding. The general procedure of creating a short name is to take each question and distill it into the most significant and identifiable parts or key words. Words should be separated with an underscore (_). It is important to make sure that short names are each only applicable and identifiable to one question, to prevent confusion and mistakes.

Example:

You have two questions which involve aspects of the “crown band” on your watermark.

Question 1: Does the band of the crown contain diamonds in it?

Question 2: Is there one side of the crown band which is not vertically straight?

It is unwise to use just “crown_band” as the short name for one of these questions, as it could be mistaken for the other question instead. In this situation, both questions could be more immediately identifiable by the specific features they refer to. A possible short name for Question 1 could be “crown_band_diamonds”. A possible short name for Question 2 could be “crown_band_straight”.

Make sure that each short name you choose for a question is unique, identifiable, clear, and as succinct as possible while retaining accuracy. Others should be able to deduce which full question you are referring to just by reading the short name, so using key terms or words is preferable.

For short names of the watermark subvariants, the naming procedure is much more straightforward. Simply take the name of the watermark (or shortened watermark name) and the subvariant and input it in the following structure: [watermark name]_[Subvariant]. Note that the watermark name will be lowercase normally. For longer watermark names, the “watermark name” component can be shortened (i.e. Basel Crosier could become “basel”). The watermark name and subvariant will be separated by an underscore, without spaces. The subvariant follows after the underscore, with the first letter capitalized and additional letters lowercase and without separation. The typical format of subvariant notation (B.a.b.) will have excess characters (periods) removed in short name notation. “B.a.b.” in typical subvariant notation will become “Bab” in short name format.

Example:

Long Format: Basel Crosier, Subvariant B.a.b. (or Basel Crosier, B.a.b.)

Short Name Format: basel_Bab

As you can see above, the long format of watermark and subvariant notation becomes shortened and standardized in short name format.

It is important to also note that “prime” watermarks, (i.e. B’.a.a., B’.b., etc.) will retain “prime” in their short name format. “Basel Crosier B’.a.a.” would become “basel_B’aa” in short name format

The last considerations of short name format are their use in the CSV file name and image name. The CSV file name should be the SAME as the short name of the question (or watermark) the file pertains to. The image name should also be the same as the short name of the question or watermark it pertains to if there is only one image used for that page. If there are multiple images compiled into one file (such as the four-image comparison with full yes, full no, magnified yes, and magnified no), then this image can either be named with the question’s short name or as a part of an image process, with the format of “watermark name_Step 1”. If you choose to follow this format, each subsequent image will be named in consequential order (i.e. “watermark name_Step 2”, “watermark name_Step 3”, etc.). If there are multiple images used as multiple files (not compiled into one), the image short name for each should use the short name of the question it pertains to and an identifier, either numbers used in consequential order, or an additional word which may more properly identify that specific image. The proper format should therefore be either “shortname_1” in consequential order, or “shortname_imageword”.

Step 2 – Building Blocks of Porting: Branch Tree

After establishing all of your short names, you will be ready to compile the parts of your branch into a branch tree CSV file. In the [Pre-Porting Step](#), you should have also completed your branch diagram. Consulting both your branch diagram and your short names, you will now construct the CSV file of your branch tree.

Begin by creating a CSV file either in a text editor or Microsoft Excel (following the guidelines of [Step 0.1 – Understanding and Using CSV Files](#)).

Insert your data with the following format of rows in sequential order:

1. The first row is your Header, “Branch Name”.
2. The second row is the full name of your branch. (ex. Single-Headed Eagle).
3. The third row consists of your categories: “Parent” (referring to the preceding or “parent” question); “Yes”; and “No”. Yes and No refer to the answers given to the parent question.
4. The fourth row, and each row after, uses short names. Under the parent column (first field in the fourth row of your CSV file) you will enter the short name of your first question. In the second field of this row (under the “yes” column) you will enter the short name of the question that results from answering yes to the first question. In the third field of this row (under the “no” column) you will enter the short name of the question that results from answering no to the first question.
5. In the fifth row, you will use either of the questions stemming from the first/parent question. This means that, if you use the question from answering yes to the parent question, the short name which you entered in the second field of the fourth row will now become the short name of the first field of the fifth row.
6. Continue for each subsequent row in this manner. The exact order of the rows of questions is not exceedingly important, as choosing whether to use the “yes” question or “no” question from the previous question, for any given row is largely arbitrary.

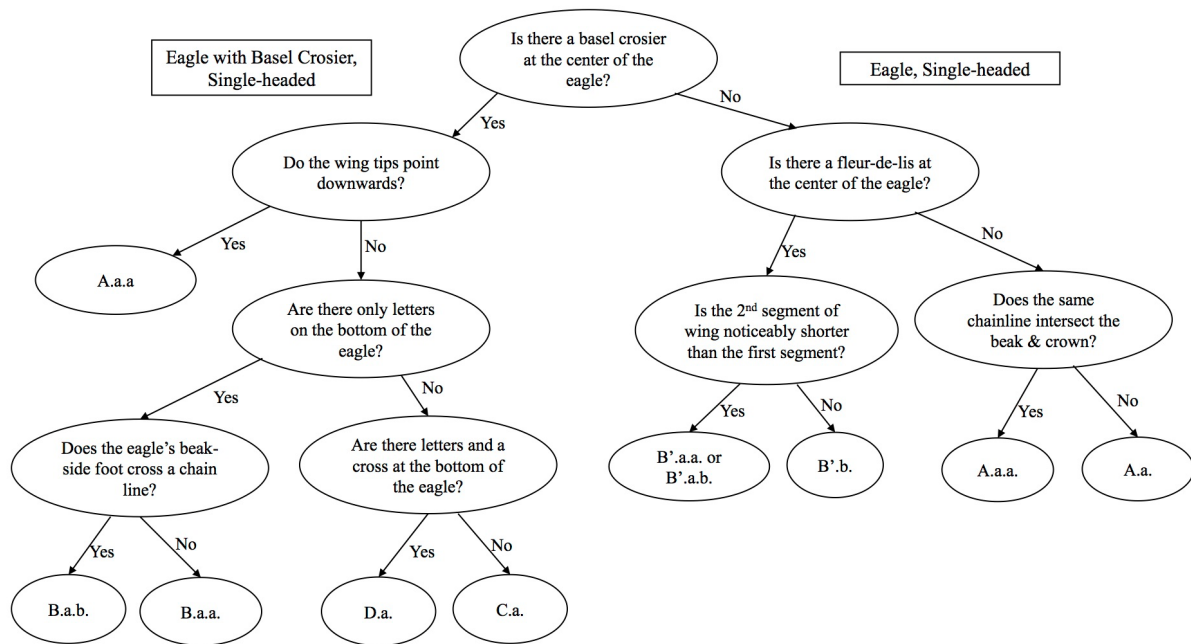
The purpose of creating a branch tree in the CSV file is to have an easy to access record of the short names and paths to each subvariant. It is important to know which questions lead to which answers, and while this information is more intuitive when presented in a branch diagram, the same sequences must be translated into code.

The following page shows an example of the formation of the branch tree. The example includes the branch diagram and branch tree CSV file (in GitHub) for the Single-Headed Eagle watermark branch.

By following this example, you should be able to understand and utilize the format of the branch tree CSV file in order to create your own branch tree CSV file.

Example:

Single-Headed Eagle Branch Diagram:



Single-Headed Eagle Branch Tree CSV File:

12 lines (11 sloc) | 340 Bytes

RawBlameHistory

Q Search this file...

1	BRANCH NAME		
2	Single-Headed Eagle		
3	PARENT	YES	NO
4	basel_eagle	wings_downward	fleur_eagle
5	wings_downward	basel_Aaa	only_letters_eagle
6	only_letters_eagle	wings_chain_line	letters_cross_eagle
7	wings_chain_line	basel_Baa	basel_Bab
8	letters_cross_eagle	basel_Da	basel_Ca
9	fleur_eagle	wing_shorter	beak_crown
10	wing_shorter	B'aa	B'b
11	beak_crown	Aaa	Aa

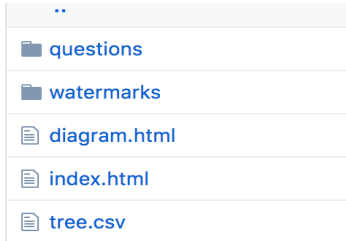
Step 3 – Building Blocks of Porting: File Folders

Now that you have created your Branch Tree CSV File, you are prepared to create the files for your branch's questions and watermarks. In order to do this, however, you must understand the structure and format of file folders to be used. One of the goals of the porting process is to reduce the clutter and chaos of the previous coding structure which required folders nested inside of folders inside of folders ad infinitum, was confusing and difficult to work with, and even more difficult to edit. The new porting and coding process is meant to use just one folder for each watermark branch.

The file folder format is as follows:

- The overarching folder or “mother” folder, which comprises all files relevant to this watermark. This folder will usually be named with the full name of the watermark, with underscores used instead of spaces, and in lowercase. (e.g. single_headed_eagle).
Inside of the mother folder are generally five components important for our purposes- the questions folder, watermarks folder, images folder, diagram file, and branch tree file. There is also an index.html file in most folders.
 - Questions Folder- The questions folder is made up of all of the CSV files for questions used in the branch (one file for each question).
 - Question File(s)- Each question will have its own file within the folder.
 - Watermarks Folder- The watermarks folder is made up of all of the CSV files for watermark subvariants in the branch (one file for each subvariant).
 - Watermark File(s)- Each subvariant will have its own file within the folder.
 - Images Folder- The images folder is made up of all of the PNG files for the images used in the branch (one file for each image or image composite).
 - Image File(s)- Each image will have its own file within the folder.
 - Diagram File- This file should represent your branch diagram, either in PDF form or similar, or as an html file.

- Branch Tree File- This file is the same CSV file of your branch tree created in [Step 2](#).
- Index.html File- The default file for the website's home page, usually created automatically, which can link to other main pages if named appropriately.



The image on the left shows an incomplete “mother” folder, consisting of the questions folder, watermarks folder, diagram file, index.html file, and branch tree file.

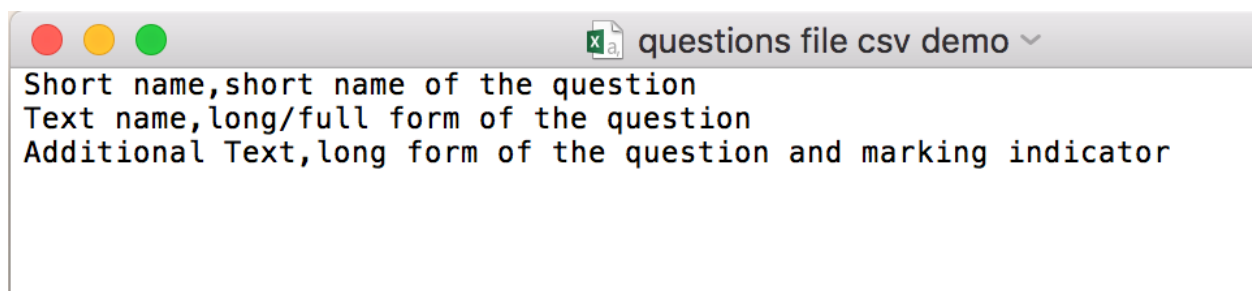
The files which non-coders will be able to contribute to (the questions folder, watermarks folder, and images folder, after having done the diagram and branch tree files in previous steps) will be demonstrated in greater detail in the next steps.

Step 4 – Porting: Questions Files

The Questions Folder of each watermark branch consists of all the individual files of the questions used in the branch. The short names for this folder will already have been determined, and should also be represented in your Branch Tree.

The components of each question file are the question short name, the “text name”, and “additional text”. The “text name” is the long format of the question (which is what will appear on the website). The “additional text” refers to the extra text (outside of the question) which appears on the question page. For our purposes, the additional text of the question file will almost always be the descriptor of the image. This is typically the same question as in “text name”, but with the addition of a marking indicator, such as “(Marked with red)”.

The **format** of the CSV file for each question file in the questions folder is as follows:



The following screenshot is an example of a question file in the Single-Headed Eagle branch's Questions Folder.

Example:

1	Short name	basel_eagle
2	Text name	Is there a crosier of the Basel type, with three lobes at the bottom, on the eagle's breast?
3	Additional text	Is there a crosier of the Basel type, with three lobes at the bottom, on the eagle's breast? (circled in grey)

This standard structure is used for every question in the branch. Each question gets its own CSV file. It is important to remember that the question file should be named the same as the question's short name. In the screenshot above, the file would be named "basel_eagle.csv".

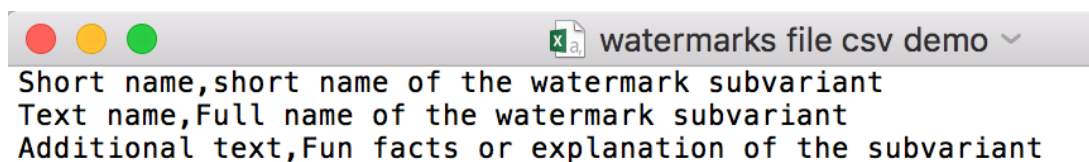
When porting your own watermark branch, follow the format provided above for each question file (saved/exported as a CSV file).

Step 5 – Porting: Watermarks Files

The Watermarks folder is very similar to the Questions folder both in format and purpose. The Watermarks folder houses a CSV file for each watermark subvariant in the branch. In coding and porting, the subvariant (and subsequently the watermark CSV file in the watermarks folder) uses, as you recall, short name structure without punctuation (no periods), except for prime subvariants.

The components of a watermark subvariant file in the watermarks folder are the watermark subvariant short name, the "text name", and the "additional text". In these files, the "text name" is the full name of the watermark subvariant in proper notation, which is what a WIRE website user would see if they went through the branch. The "additional text" is the fun facts or explanation of the subvariant.

The **format** of each watermark subvariant file in the watermarks folder is as follows:



The following screenshot is an example of a watermark subvariant file in the Single-Headed Eagle branch's Watermarks Folder.

Example:

1	Short name	B'b
2	Text name	Eagle, Single-Headed, B'b.
3	Additional text	Watermark B'b. is distinguishable by these three features: a fleur-de-lis at the center of the eagle, three long claws extending

This standard structure is used for every watermark subvariant in the branch. Each subvariant gets its own CSV file. It is important to remember that the subvariant file should be named the same as the watermark subvariant's short name. In the screenshot above, the file would be named "B'b.csv" (if subvariant naming is only within one watermark- otherwise watermark short name identifiers would be added, such as "single_eagle_B'b.csv").

When porting your own watermark branch, follow the format provided above for each question file (saved/exported as a CSV file).

Step 6 – Porting: Further Utilization in Code

Now that you have finished all of the components required for porting- a folder for your watermark, short names for all of your questions, subvariants, etc., a branch diagram, a branch tree CSV file, image files, and CSV files for all of your questions and watermark subvariants- you are ready for the final step of this Porting Manual, further utilization of your components in actual code.

This portion of the porting process is the most difficult to understand and use, especially for non-coders, and so this step will most likely be performed by WIRE coders.

This step basically consists of taking the folders, files, and data provided from the previous steps, and inserting them into the website code in order to display this data onto the website. The coding will allow the website to display your questions, "yes" and "no" choices, images, and subvariant results in a functional and user-friendly format.

Essentially, there will be "template" code files which will just require a placeholder file name to be switched for the actual desired file name (the name of one of the CSV files in your folder). In the future, we hope to make these code templates accessible to all WIRE students along with more detailed instructions on the placement of file names and any other components needed.

In order to use the code templates, it is important to utilize your short names, question files, watermark subvariant files, and branch tree.

To demonstrate the process, examples of code template files for question files and watermark subvariant files are provided below.

Question File Code Template:

7 lines (6 sloc) | 289 Bytes

```
1 <h3>replace_question_textname</h3>
2 <a class="button" href="replace_left_shortcode.html">Yes</a>
3 <a class="button" href="replace_right_shortcode.html">No</a>
4 <h3>Example</h3>
5 <p>replace_question_additional_text</p>
6 
```

In the image above, you can see that there are several spaces with “replace name” in them. You will use the code template and replace those placeholders with the actual file names, as determined in [Step 4](#). Using our example from the Questions Files process (question about basel crosier on single-headed eagle), the “text name” (full question text) is, “Is there a crosier of the Basel type, with three lobes at the bottom, on the eagle’s breast?”. If you insert the text name into line 1, it will then become “<h3>Is there a crosier of the Basel type, with three lobes at the bottom, on the eagle’s breast?</h3>”.

Lines 2 and 3 refer to the short names of the yes and no questions to the parent question. You should be able to find these in your Branch Tree from [Step 2](#). Continuing with our basel crosier eagle example, the short name of the “yes” question would be “fleur_de_lis”. The short name of the “no” question would be “wing_tips_down”. Inserting into the template, line 2 would now read “Yes”. Line 3 would now read “No”.

Line 5 refers to “additional text” (found in your question file). The “additional text” (question with annotation for use with images) for our example is “Is there a crosier of the Basel type, with three lobes at the bottom, on the eagle’s breast? (circled in grey)”. When inserted into the template, line 5 will now read “<p>Is there a crosier of the Basel type, with three lobes at the bottom, on the eagle’s breast? (circled in grey)</p>”.

Line 6 refers to the image file used in conjunction with your question. For our purposes, we will say that the short name of the image file is “basel_eagle_image1”. When inputted into the template, line 6 will now read “”.

Watermark File Code Template:

4 lines (3 sloc) | 170 Bytes

```
1 <h3>Your watermark is replace_watermark_textname</h3>
2 <p>replace_watermark_additional_text</p>
3 
```

The procedure for the watermark file code template is generally the same as that for the question file code template- simply replacing the placeholder file names in the code. The “text name” and “additional text” can be found in your watermark file from [Step 5](#). We will continue with the example used in that step of Single-Headed Eagle subvariant B’.b.

Line 1 refers to the watermark “text name” of the watermark subvariant, with the format of “Your watermark is” followed by the text name. With our example, that would be “Your watermark is Eagle, Single-Headed, B’.b.” When inserted into line 1, it would read “<h3>Your watermark is Eagle, Single-Headed, B’.b.</h3>”.

Line 2 refers to the “additional text” of the watermark subvariant. In our example the additional text would be “Watermark B’.b. can be distinguished by three features: the eagle has a fleur-de-lis at its center, three long claws extending from its foot, and segmented wings (of which the first two segments are equally sized). (Marked with red).”. Inserted into line 2, the code becomes “<p>Watermark B’.b. can be distinguished by three features: the eagle has a fleur-de-lis at its center, three long claws extending from its foot, and segmented wings (of which the first two segments are equally sized). (Marked with red).</p>”.

Line 3 refers to the image associated with the watermark subvariant. Let’s assume that the short name of the image for B’.b. is “single_eagle_B’b”. When inserted into line 3, it will read “”.

Now that you have reached the end of this Porting Manual, you have learned how to create and compile all of the required parts of porting, as well as how to insert file names into the code templates, you are ready to complete your own porting processes!