Boston Streets: Early Directory Tagging Guidelines

Created: 9/22/03 Revised: 1/14/04

Author: Jessica M. Branco, DCA

These guidelines are to be used for directories which retain the typographical and punctuation-based field delimitations, common in earlier 19th century city directories (1845-1905), see the Modern Directory Guidelines for information on how to tag directories without punctuation and stylistic delimiters.

It is critical to tagging success to follow these guidelines in the order they are outlined, since certain regular expressions are dependent on previous expressions having generated certain character patterns.

Part I: Basic Structural Markup (Steps 1-7):

1. Open tab-delimited text file for directory and save as the directoryyear(YYYY)_mmddyy.yourinitials..xml. (Example: 2004_010104.jmb.xml). Each time you work with a document, be sure to save the file as a new document, adjusting the date and initials as needed.

2. At the top of the document, insert the Boston Streets XML DTD and P4 TEI Header. (This information can be copied and pasted from Q: Boston Streets\XML Markup\XML markup resources\DTDandTEIheader.txt) Some information is already included in the header and may need to be adjusted to match the specific document on which you are working.
    a. Note: Be sure to paste in the appropriate DTD for the text editor you are using – both are available in this file. The TEI header statement worksin all text editors.

3. After the last line of the header, insert open text and body tags by inserting:
    a. <text><body>

4. Scrolling to the end of the document, close the body, text and TEI tags with by inserting the following
    a. </body></text></TEI.2>

5. Next, page numbers need to be converted to xml format:
    a. Find: <pb n=([0-9]*)>
    b. Replace with: <pb n="\1" id="p.\1"/>

6. Fractions and Ampersands in the text need to be converted to Unicode
    a. It is easiest to convert all fractions global to an "&fracxx;" format, and then convert each fraction to its Unicode equivalent
        i. Find: <F>([0-9])/([0-9])</F>
        ii. Replace with: \&frac\1\2;
        iii. Find: </F>([0-9])/([0-9])</F>

iv. Replace with: \&frac\1\2;
v. Find: <F>([0-9])/([0-9])<F>
vi. Replace with: \&frac\1\2;
vii. Find: ½
viii. Replace with: 00BD
ix. Find: ¼
x. Replace with: 00BC
xi. Find: ¾
xii. Replace with: 00BE
    1. Other fraction-based parsing errors may occur that are not accounted for here. It may be necessary to adjust some fractions by hand, or with a temporary regular expression.

b. All Ampersands (&) in the BODY of the text need to be converted to Unicode. Do Not Convert Ampersands in the DTD.
    i. Select all text in the Body of the document.
    ii. Open the Search and Replace tool
    iii. Under "Scope", check "Selected Text"
    iv. Under "Conditions," deselect "Regular expression." This procedure does not work if regular expression is selected.
    v. Find: &
    vi. Replace with: 0026
        1. Note: 0026=zero-zero-two-six

7. Next, insert the three levels of divisions into the document. Divisions within the directories are as follows:
    a. Sections are divided up by the letter using <div1>. Occasionally, the first <div1> in a document will consist of an "Additions, Omissions, etc." section. See 7.4.b. below for information concerning these occurances.

        i. Use the following tag, with attributes to mark letter divisions: <div1 type="section" n="div title" id="d.year.div title">
        ii. Run the following series of regular expressions to insert the tag with attribute values filled in for letter sections. (Example: <div1 type="section" n="A" id="d.1865.A">) It is necessary to "find next" and "replace next" for <div1>. There may be other instances of multiple uppercase letters (AEtna, etc.) this regular expression needs to be done with careful review before "find next"-ing for the whole document. It is best to not automatically "replace all" when using this expression
            1. Find:
            ^([A-Z])([A-Z])
            2. Replace with:
            <div1 type="section" n="\1" id="d.YYYY.\1">\n<div2 type="entry unit" id="d.YYYY.u.">\n\1\2

a. Replace YYYY with the appropriate year for the directory (1865, 1925, etc.) before running the expression.

3. Find: <div1
4. Replace with: </div1>\n&
    a. Note: Do not replace for first div1 in document, and close the last div1 by hand.
    b. If an Additions or other section begins the document, adjust the inserted tags to reflect the accurate information in the n label and id attribute values. Use the full section title in the n label and an abbreviated form for the id. (Example <div1 type="section" n="Additions and Omissions" id="d.1865.AdOm">
    c. If entries are segregated, so that there are, in fact, two A sections, two B sections, etc., then adjust the second appearance of the letter's div id value to "AA", "BB", and so on.

b. Entries are isolated and identified by their placement in the entire directory. Each entry is encased in a <div3> tag. It is necessary to identify the entries BEFORE grouping them into deliverable chunks.

i. Use the following tag, with attributes to mark entry divisions:
<div3 type="entry" id="d.1865.e.entrynumer">
(Example: <div3 type="entry" id="d.1865.e.1">)

ii. Use the following Regular Expressions to automatically insert the tag with attribute values filled in

1. Find: ^([A-Z][A-Za-z]*)
2. Replace with:<div3 type="entry" id="d.YYYY.e.">\1
    a. Note: Replace YYYY with the appropriate year before running regular expression.
3. Find: ^([a-z])
4. Replace with:<div3 type="entry" id="d.YYYY.e.">\u\1
5. Find: <div3 type="entry" id="d.YYYY.e.">
6. Replace with:<div3 type="entry" id="d.YYYY.e.\i">
    a. \i will automatically number the entries in consecutive order. Move the cursor to the absolute top of the document before running this expression.
    b. If it becomes clear that <div3>s were missed or omitted in the tagging and number, this value can be removed and replaced easily, with no need to panic. See "More Useful Expressions" below for more information.
7. Find: ([.)A-Za-z0-9])$
8. Replace with: \1</div3>

9. Find: ([-])$
10. Replace with :\1</div3>
11. Find: ([^>^])$
12. Replace with: \1</div3>
13. Find: ([^>^])</div3>
14. Replace with: \1 </div3>

c. Some entries are actually redirect or "see also" lines for use within the directory. These entries continue to be div3s, but have a different type value (see.also). The format of these tags is as follows:
   i. <div3 type="see.also" n="see also information"> see also information </div3>
   (Example: <div3 type="see.also" n="Abot see Abbot, Abbott, and Abott"> Abot see Abbot, Abbott, and Abott </div3>

d. Groups of 100 entries, or the remainder in an alphabetic section are grouped into "entry units," with each marked as <div2>. This grouping is for structural purposes allowing for manageable delivery and organization of the document's information.
   i. Use the following tag, with attributes, to mark entry units: <div2 type="entry unit" n="alpha-range" id="d.year.u.batch"> (Complete Example: <div2 type="entry unit" n="AAA-Abb" id="d.YYYY.u.1">)
   ii. Use the following Regular Expressions to automatically insert the

Until further notice ignore this section.
(1/13/04)

e year

tep 8b.

6. Replace with:<div2 type="entry unit" id="d.YYYY.u.\i">
   a. \i will automatically number the entries in consecutive order. Move the cursor to the absolute top of the document before running this expression.
   b. If it becomes clear that <div2>s were missed or omitted in the tagging and number, this value can be removed and replaced easily, with no need to panic. See "More Useful Expressions" below for more information.

e. More Useful Expressions: If you have numbered the <div2> or <div3> tags in a document and it becomes clear that some entry units or entries

were missed, you will need to renumber the div ids. The following regular expressions will remove the existing id and re-run the numbering process. It is very important to move the cursor to the start of the document before running the number generating expressions.

i. Remove and renumber <div2> ids: (remember to change YYYY to the appropriate year)
    1. Find: <div2 type="entry unit" id="d.YYYY.u.*">
    2. Replace with: <div2 type="entry unit" id="d.YYYY.u.">
    3. Find: <div2 type="entry unit" id="d.YYYY.u.">
    4. Replace with:<div2 type="entry unit" id="d.YYYY.u.\i">

ii. Remove and renumber <div3> ids: (remember to change YYYY to the appropriate year)
    1. Find: <div3 type="entry" id="d.YYYY.e.*">
    2. Replace with: <div3 type="entry" id="d.YYYY.e.">
    3. Find: <div3 type="entry" id="d.YYYY.e.">
    4. Replace with:<div3 type="entry" id="d.YYYY.e.\i">

f. Each entry is encased in a paragraph tag which should have been inserted in a previous step. However, to make sure all … tags are in place, it may be helpful to run the following:
    i. Find: <div3 type="entry" id="d.1865.e.[0-9]*">
    ii. Replace with:&
    iii. Find: </div3>
    iv. Replace with: &

At this point in the document, it is best to "parse" the document. (check the format of the tags in the document in order to make sure there are no typos, errors, etc.) To parse the document in TextPad, select "Tools" from the top menu bar. "Validate XML" or "Validate TEI XML" should be an option. Select this option and review the "Command results." Any errors in the mark up will be displayed here.

If the message you see says "Tool completed Successfully" you can proceed to the next stage of Markup. If there are errors, clicking on the error reported in the "Command Results" will jump the cursor to that point, or a nearby point in the document. After you fix the error, you can parse the document again to see if the problem has been resolved. Multiple errors can be reviewed and fixed between parsings, but it is best to work from the top of the list down, and correcting one mistake often reduces the cascading number of errors.

Part II: Content-Relevant Markup

8. Within each entry, a number of tags are used to identify and encase components, such as names, occupations, and addresses. The following elements(tags) with

attributes and defined value sources are used within the entries in the early directories:

Name:

<persName n="last name, first name, generational data, role/title"> (all elements listed may not be used for each entry)
<surname n="last name">...</surname>
<foreName n="first name/initials, and all other names/initials">...</foreName>
<genName n="generational data">...</genName>
<roleName n="role/title">...</roleName>
</persName>

or

<orgName n="orgnization's name">organization's name</orgName>
Note: Entries may contain multiple occurances of persName and orgName elements.

Occupation/Business focus:

<rs type="occupation" n="type of occupation listed here">occupation</rs>
(<rs>may encase orgName, persName, or placeName tags; or it may not be present)

Addresses:

<address n="commercial">
<street n="street number and name"><num>street number</num> street name</street>
</address>

and/or

<address n="commercial.nonBoston">
<street n="street number and name"><num>street number</num> street name</street>
<name><placeName n="settlement name">settlement name</placeName></name>
</address>

and/or

<address n="residential">
<name type="residence.type" key="type of residence listed here">type of residing (boards, house, etc)</name>
<street n="street number and name"><num>street number</num> street name</street>
</address>

and/or

<address n="residential.nonBoston">
<name type="residence.type" key="type of residence listed here">type of residing (boards, house, etc)</name>
<street n="street number and name"><num>street number</num> street name</street>
<name><placeName n="settlement name">settlement name</placeName></name>
</address>

a. Once <p> tags are in place, generate the "3 character (AAA) – 3 character (Aab)" alpha-range with for the <div2> units, using the following series of regular expressions:
    i. Find: (<div2 type="entry unit")(id="d.1870.u.[0-9]*">)\n(<div3 type="entry" id="d.1870.e.[0-9]*"><p>)([A-Z][A-Za-z'][A-Za-z])
    ii. Replace with: \1n="\4-"\2\n\3\4
    iii. Find: (</div3>$)\n(^<[a-z])
    iv. Replace with: \1\t\2
    v. Find: (</div3>$)\n(^</div2)
    vi. Replace with: \1\t\2

Until further notice ignore this section.
(1/13/04)

    xi. Find: n="[A-Z][a-zA-Z']*[-]"
    xii. Replace with: fix by hand
    xiii. Find: </div3>\t<pb
    xiv. Replace with: </div3>\n<pb
    xv. Find: />\t<div
    xvi. Replace with: />\n<div
    xvii. Find: ^\t<pb
    xviii. Replace with: <pb

b. As defined above, each entry contains either a personal name or organizational/business name, identified with <orgName> or <persName> tags and subtags. In the directories <surname>,<foreName>, <genName>, and <roleName> are used to identify components of the <persName> field.

i. You MUST identify <roleName> and <genName> components prior to running any other tagging expression, as Mrs., Miss, Misses, Rev., jr., and sr. may accidentally be identified as <foreName> components.

1. To find a <surname><roleName> only string, use the following expressions:
    a. Find: <p>([A-Z][a-z]+)\t([MR][ire][sv][s.][e.,][s]),
    b. Replace with: <p><persName n="\1 \2"><surname n="\1">\1</surname> <roleName n="\2">\2</roleName></persName>,
        i. When finding and encasing <roleName>s, "Messenger" may be caught do to its similarity to "Misses." Run the following regular expression to correct the error.
            1. Find: <roleName n="Mess">Mess</roleName></persName>enger
            2. Replace with: Messenger

    c. Find: <p>([A-Z][a-z]+)\t([MR][ire][sv][s.])\t
    d. Replace with: <p><persName n="\1, \2"><surname key="\1">\1</surname> <roleName n="\2">\2</roleName></persName>\t
        i. Note: Maintaining exiting tabs, spaces and punctuation if very important. Be careful when running a search and replace expression, as similar replacement expressions may appear to be the same. It is best to test one or two replacements before replacing all occurances.

2. Use these regular expressions to find other occurances of Mrs., Miss, and Rev. and generational signifiers in the document:
    a. Find: [ ]([MR][ire][sv][s.])
    b. Replace with: <roleName n="\1">\1</roleName></persName>
    c. Find: ([ ])([js]r[.])\t
    d. Replace with: \1<genName n="\2">\2</genName></persName>\t

ii. Begin to find and tag other personal names using the following search expressions, which may all be replaced with the "Replace with" expression:

1. Find: <p>([A-Z][a-z]+)\t([-]),\t
2. Find: <p>([A-Z][a-z]+)\t([A-Z][a-z]+[ ][A-Z][.]),\t
3. Find: <p>([A-Z][a-z]+)\t([A-Z][.][ ][A-Z][.]),\t
4. Find: <p>([A-Z][a-z]+)\t([A-Z][a-z][.]),\t
5. Find: <p>(O'[A-Z][a-z]+)\t([A-Z][a-z]+)[,]\t
6. Find: <p>(O'[A-Z][a-z]+)\t([A-Z][a-z]+[ ][A-Z][.])\t
7. Find: <p>([A-Z][a-z]+)\t([A-Z][a-z]+)[,]\t
8. Replace with: <p><persName n="\1, \2"><surname n="\1">\1</surname>\t<forename n="\2">\2</foreName></persName>,\t

iii. Continue to find and tag personal names using the following series of find and replace expressions. (Scroll to the end of each "grouping" to find the "Replace with:" expression) A number or already identified combinations of Regular Expressions for PersName identification and tagging are available in the following text file: Q:Boston Streets\XML Markup\XML Markup resources\PersNameRegEx.txt

1. Note: It may be necessary to adapt and/or develop your own regular expressions. Textpad's Help feature has a good introduction to Regular and Replacement expressions.

c. Any entity other than a personal listing (i.e. Business name, organizational name, federal office, school, hospital, society…) is encased in <orgName n="entity name">…</orgName>. Potential <orgName> entries are usually passed over by the <persName> regular expressions when there are two words beginning with uppercase letters separated by a space ([ ]) instead of a tab (\t). Variations of the following regular expression, built along similar lines to the <persName> expressions, with space substitution sfor <persName>'s tabs will find and tag most orgNames

i. Find: <p>([A-Z][a-z]+[ ]0026[ ][A-Z][a-z]*)([,.]\t)
ii. Replace With: <p><orgName n="\1">\1</orgName>\2
1. This

d. Occupations or a business type often follows the <persName> or <orgName> fields and are identifiable by the tab preceeding a lower case word pattern. Encase occupations with the following <rs> tag:

i. Find: (\t)([a-z]{1,200)([,]\t)
ii. Replace with: \1<rs type="occupation" n="\2">\2</rs>\3

1. Note: This expression will not find all variations of occupations, as it excludes uppercase letters and punctuation found in some entries.

e. Commercial addresses found within the city of Boston are listed beginning with the street number, street name and room number if available, in the following format:

<address n="commercial"><street n="street number and name"><num>street number</num> street name</street></address>

or

<address n="commercial"><street n="street number and name room/flr number"><num>street number</num> street name <num>room/flr number</num></street></address>

An address without a floor or room number can be found using the following set of regular expressions. These may be adapted to accommodate additional suite, room, and floor information:

i. Find: \t([0-9]*)[ ]([A-Z][a-z]+)[,]
ii. Find: \t([0-9]*)[ ]([A-Z][a-z]+[.])[,]
iii. Find: \t([0-9]*)[ ]([A-Z][a-z]+[ ][A-Za-z][a-z]+)[,]
iv. Find: \t([0-9]*)[ ]([A-Z][a-z]+[ ][A-Za-z][a-z]+[ ][A-Za-z][a-z]+)[,]
v. Replace with: \t<address n="commercial"><street n="\1 \2"><num>\1</num> \2</street></address>,

f. Residential addresses begin with an indicator of the type of residence (boards, h., res., etc.) and then follow a pattern similar to commercial addresses. Residential address are often irregular, have no street number, additional information or a non-Boston location (house at Melrose).

i. To find standard residential addresses in Boston in entries that have commercial addresses as well, use the following regular expression:

1. Find: </address>,\t([bhr][eo.].*)[ ]([0-9]*)[ ]([A-Z][a-z]+)</p></div3>

2. Replace with: <address n="residential"><name type="residence.type" key="\1">\1</name> <street n="\2 \3"><num>\2</num> \3</street></address></p></div3>

9. Additional Regular Expressions for directories

i. Commercial and residential <address> open tags can be used to further identify preceding occupations that have irregular typography.

1. Use the following regular expression to find irregular occupations that are followed by an address field:
    a. Find: (\t)([a-z .',]{1,200}[A-Za-z.',]{0,200}[,.])(\t<address)
    b. Replace with: \1<rs type="occupation" n="\2">\2</rs>\3