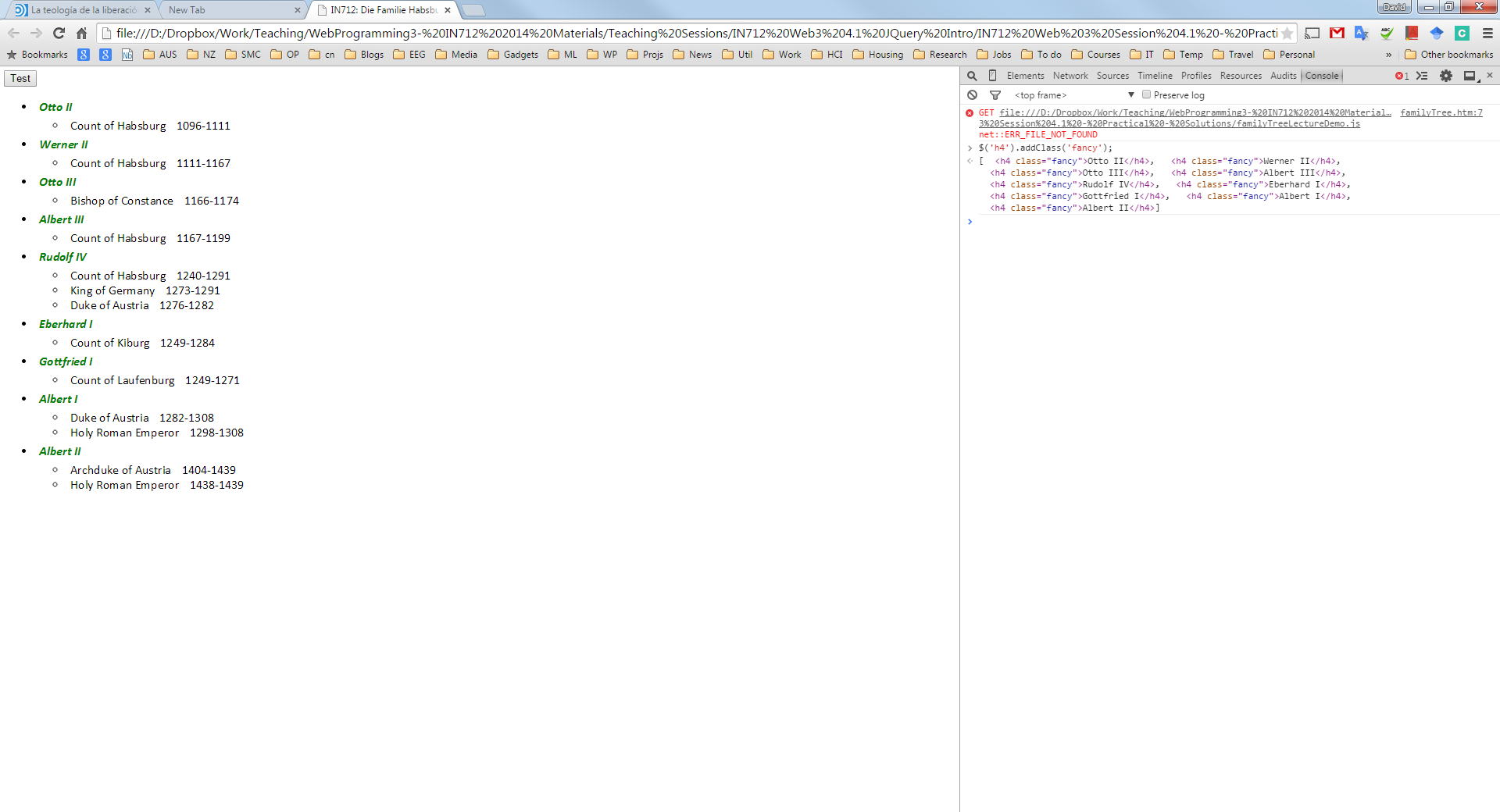
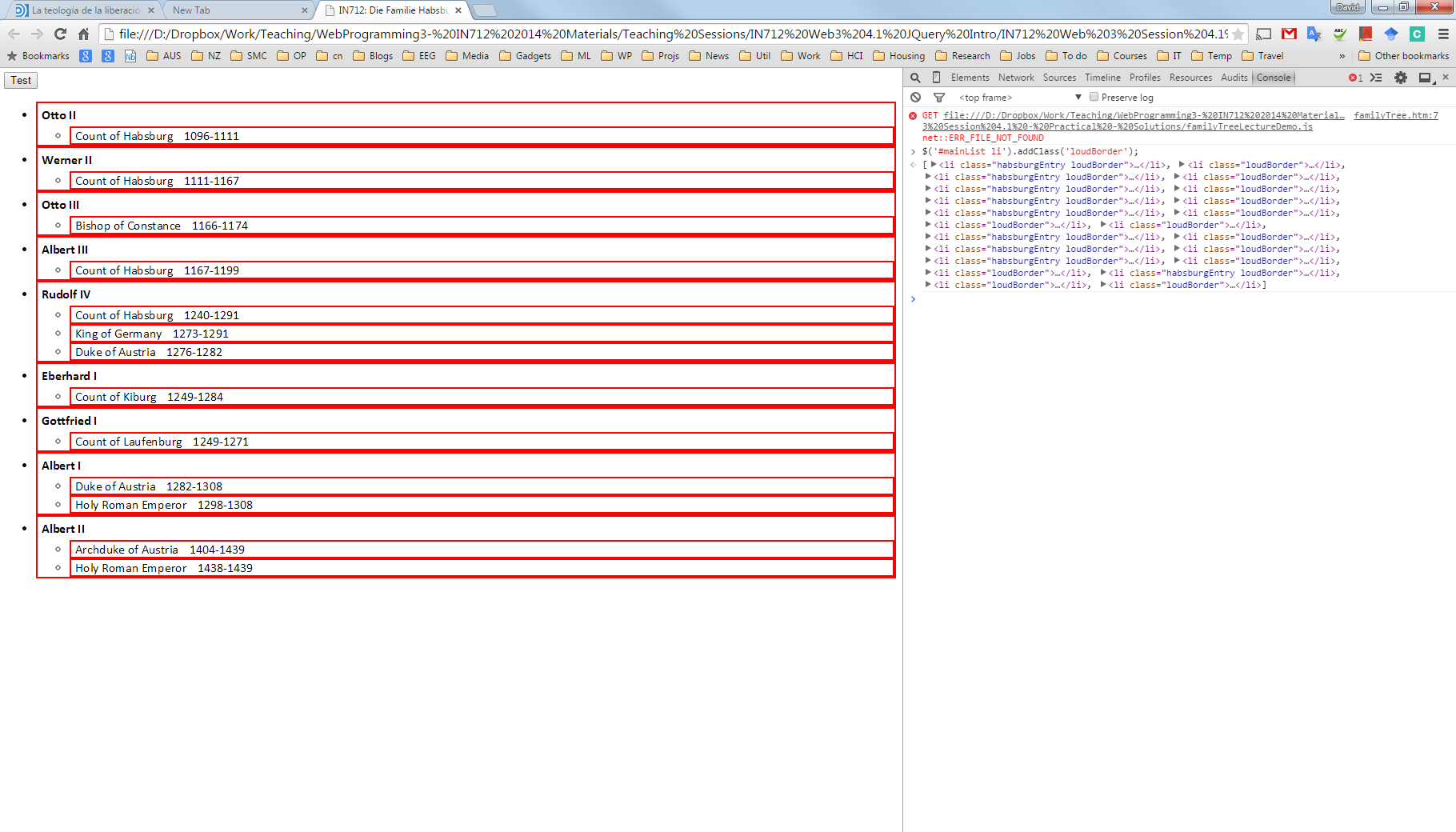
# IN712 jQuery Practical

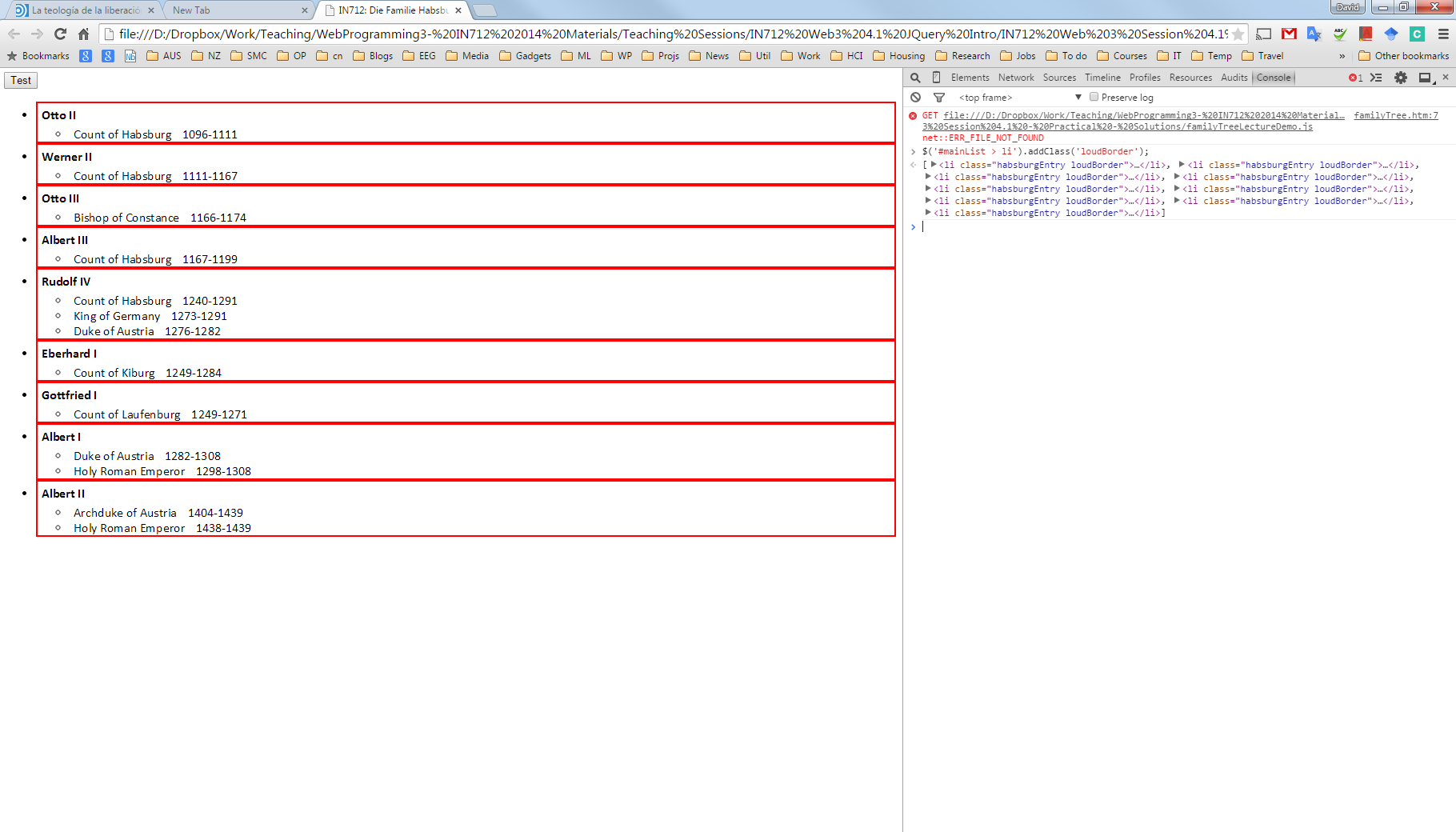
1. Pull down and open up FamilyTree.html and FamilyTree.css. Link the jquery library. . Then for each of the exercises below either:
   1. Bind the test button to a .js file with your jquery functionality for each exercise
   2. Use the javascript console (Ctrl+shift+j in Chrome) for input jquery one-liners for each exercise
2. Select all the names (Otto II, Werner II, etc). Two ways to do this, either by class (royalName) or element (h4). Add the class ‘fancy’ to the selection. Two ways to do this, either by class (royalName) or element (h4). Here is a display of what you should get:



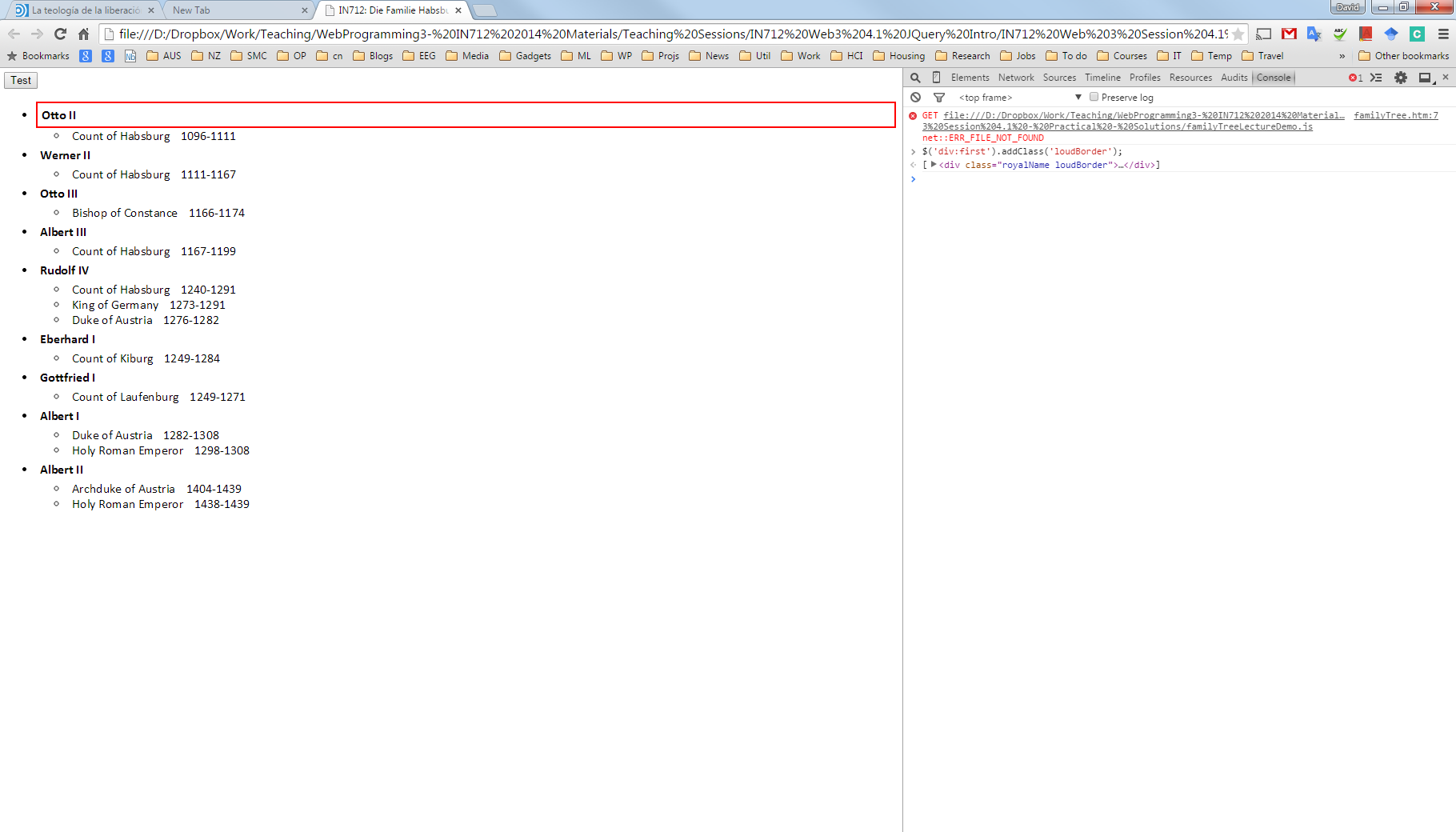
1. Select every li that is contained within the object with ID ‘mainList’ and draw a border around it by adding the class ‘loudBorder’. This selection includes the children and grandchildren, so we should get a border around each entry, plus an additional border around each item in the titleList ul’s. Note: from now on, use the class loudBorder to highlight your selections.



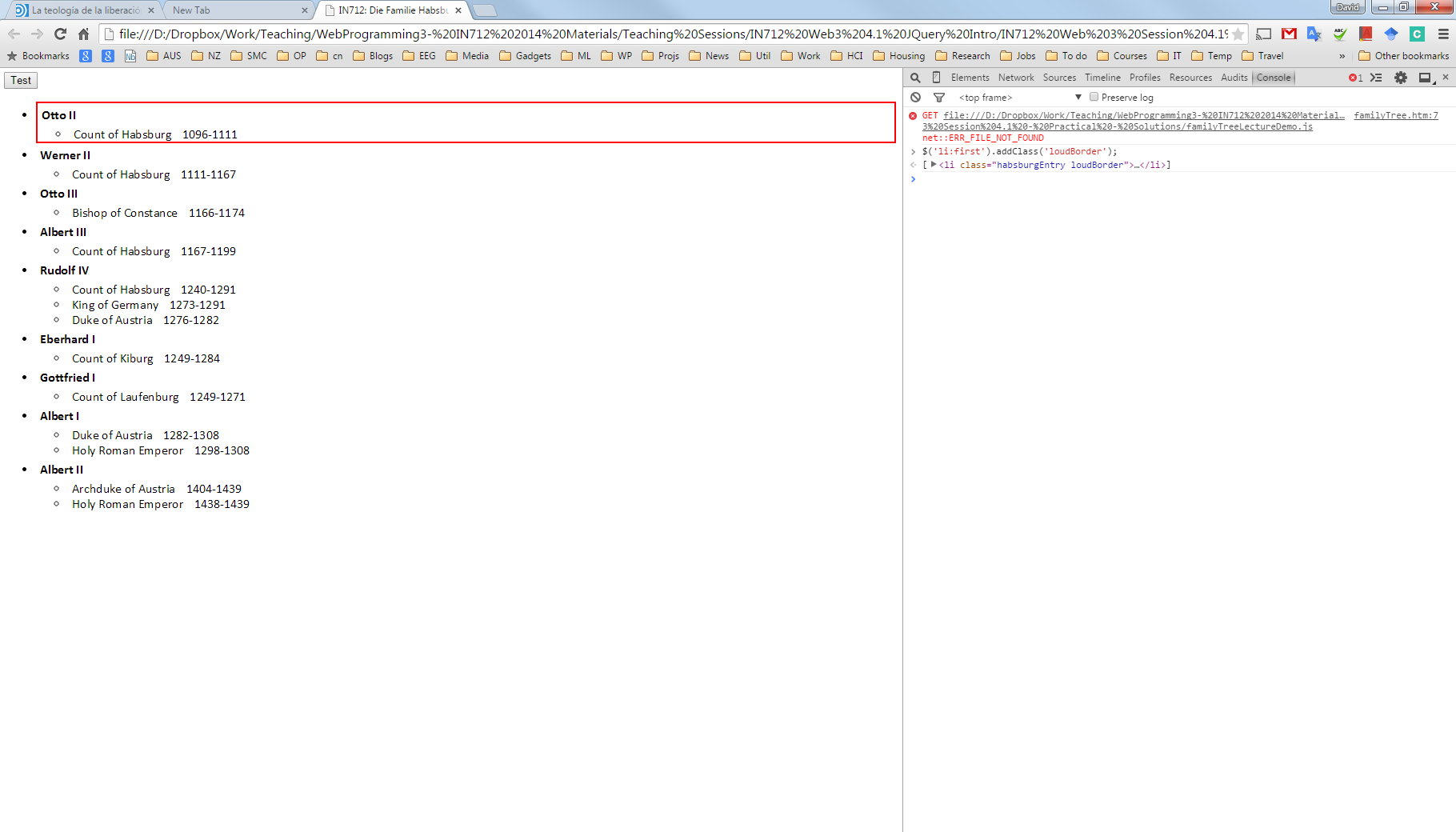
1. As above but select “direct child only”, so get children but not grandchildren. So the individual entries are bordered, but not the nested title List entries.



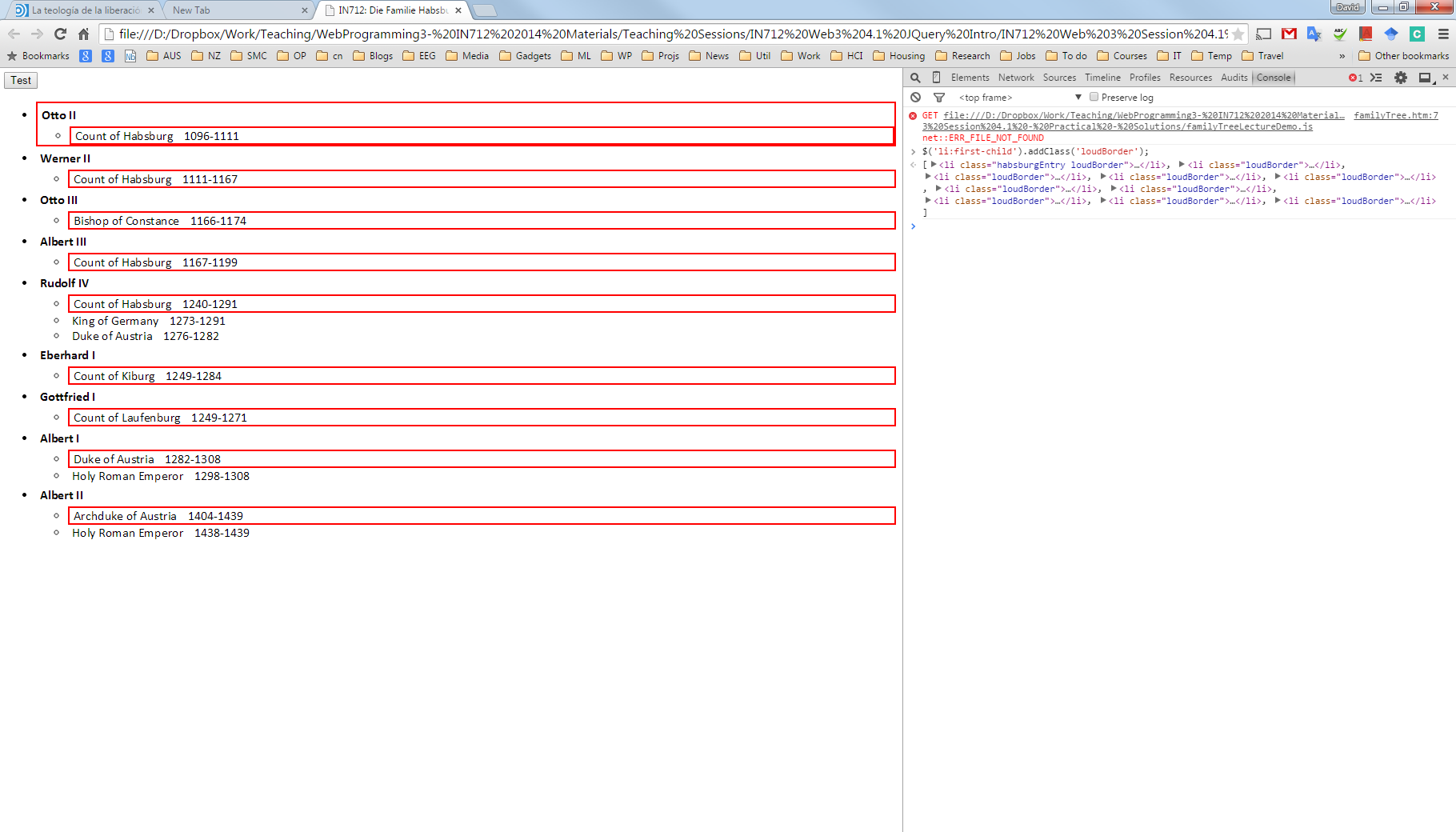
1. Select the first div on the page, and nothing else. So only the name Otto II should get the border.



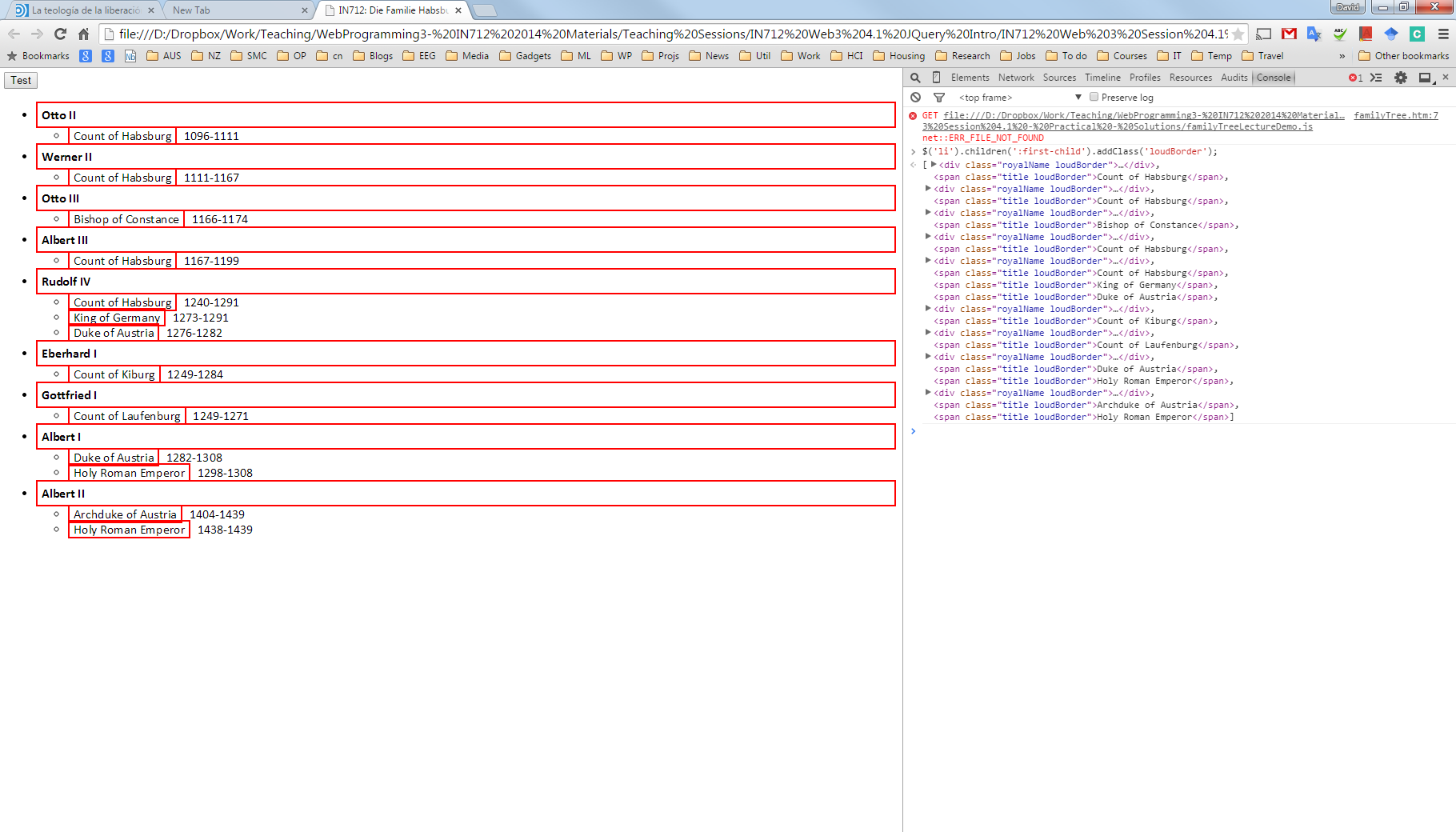
1. Select only the very first li on the page, so the first habsburgEntry item should gets the border (all of Otto II’s entry).



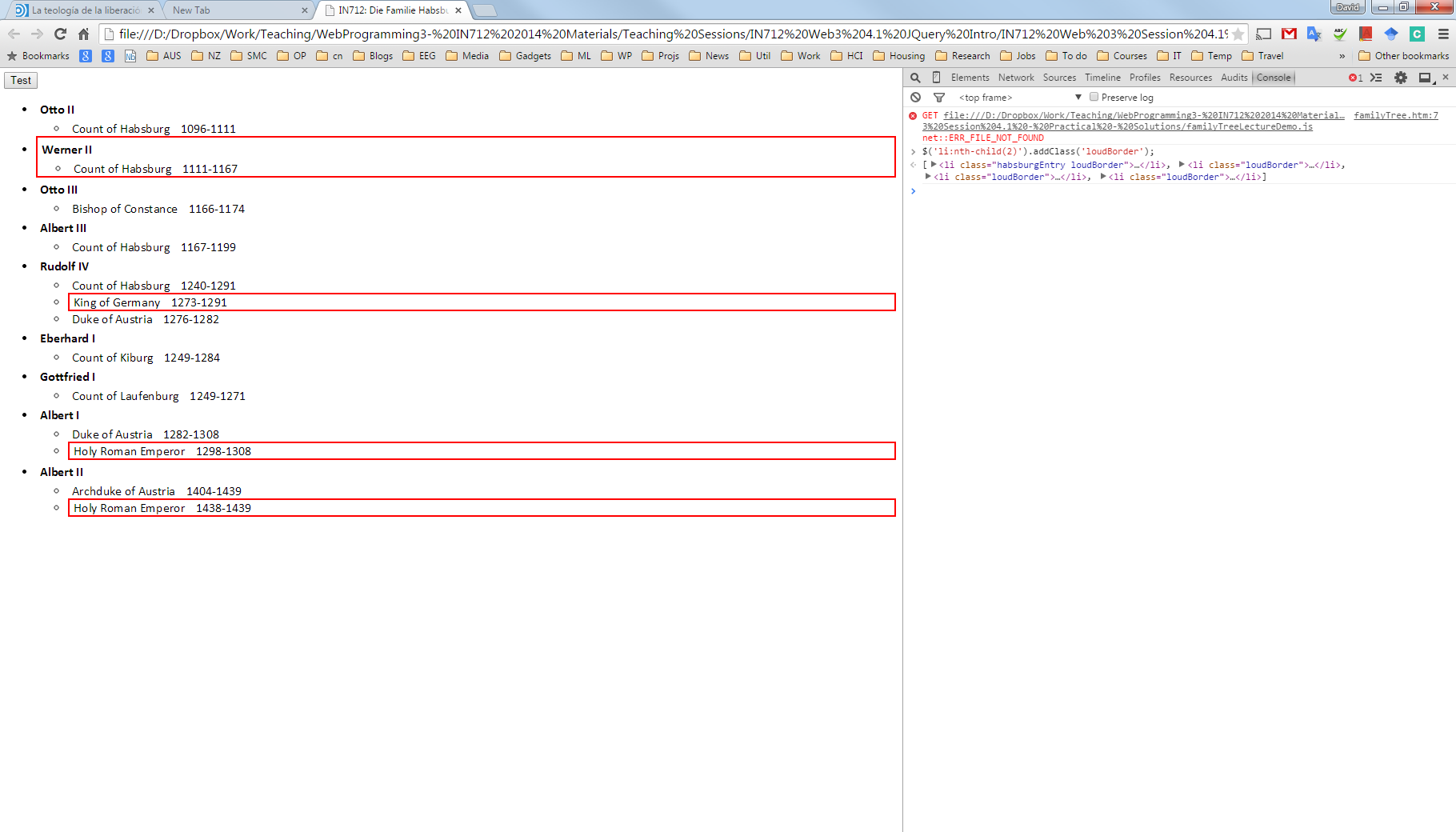
1. Select all the <li>. Then select all those that are first children of their parents. So you should actually get Otto II’s whole entry (first child of his <ul>)and the first titleList item (the two spans) for each guy, because those are the first children of their containing <ul>s.



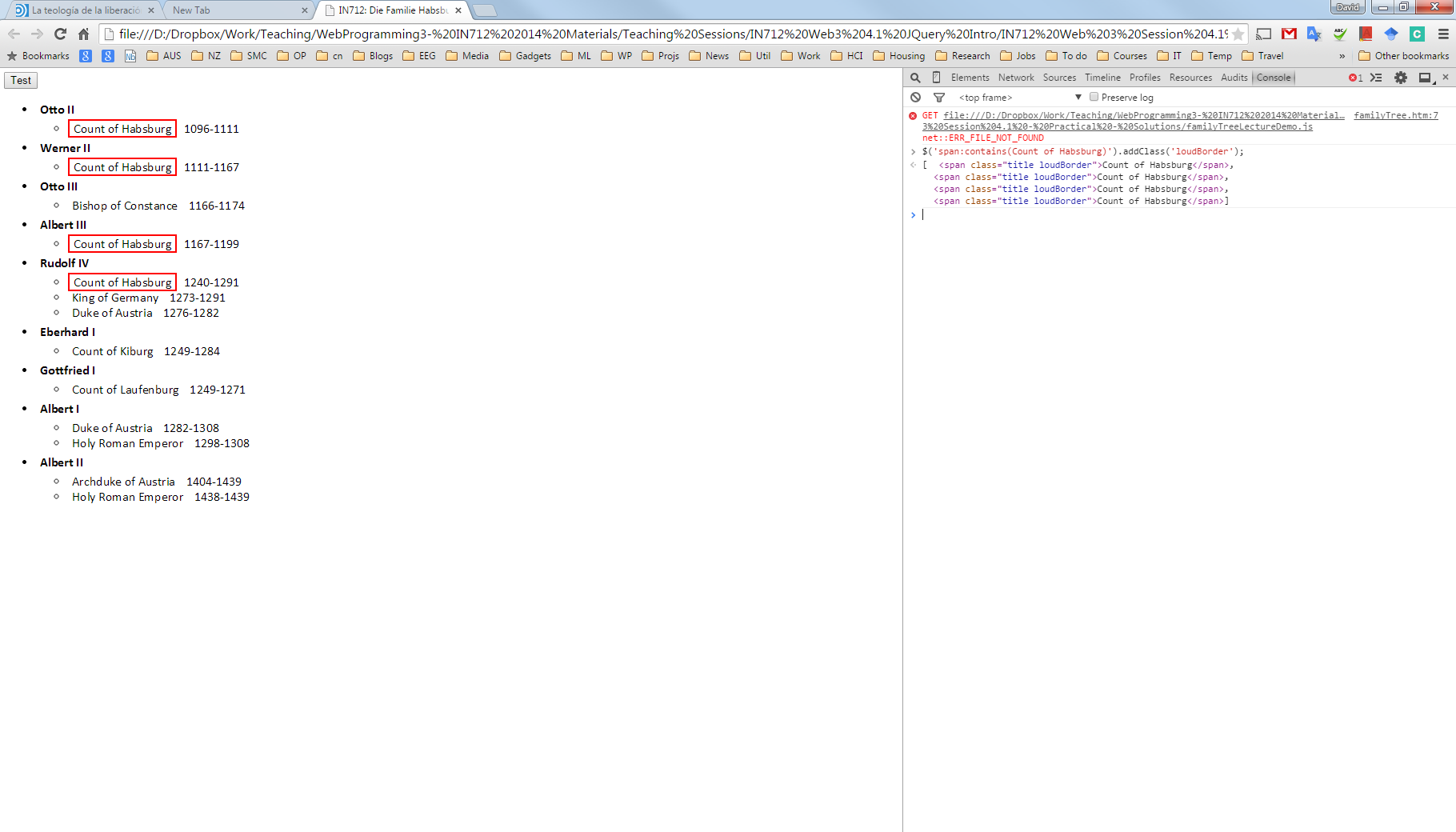
1. Select “all first children of <li> elements”? First get all the <li>, then get all the children of elements in that set, then filter any elements of ***that set*** who are themselves first-children.



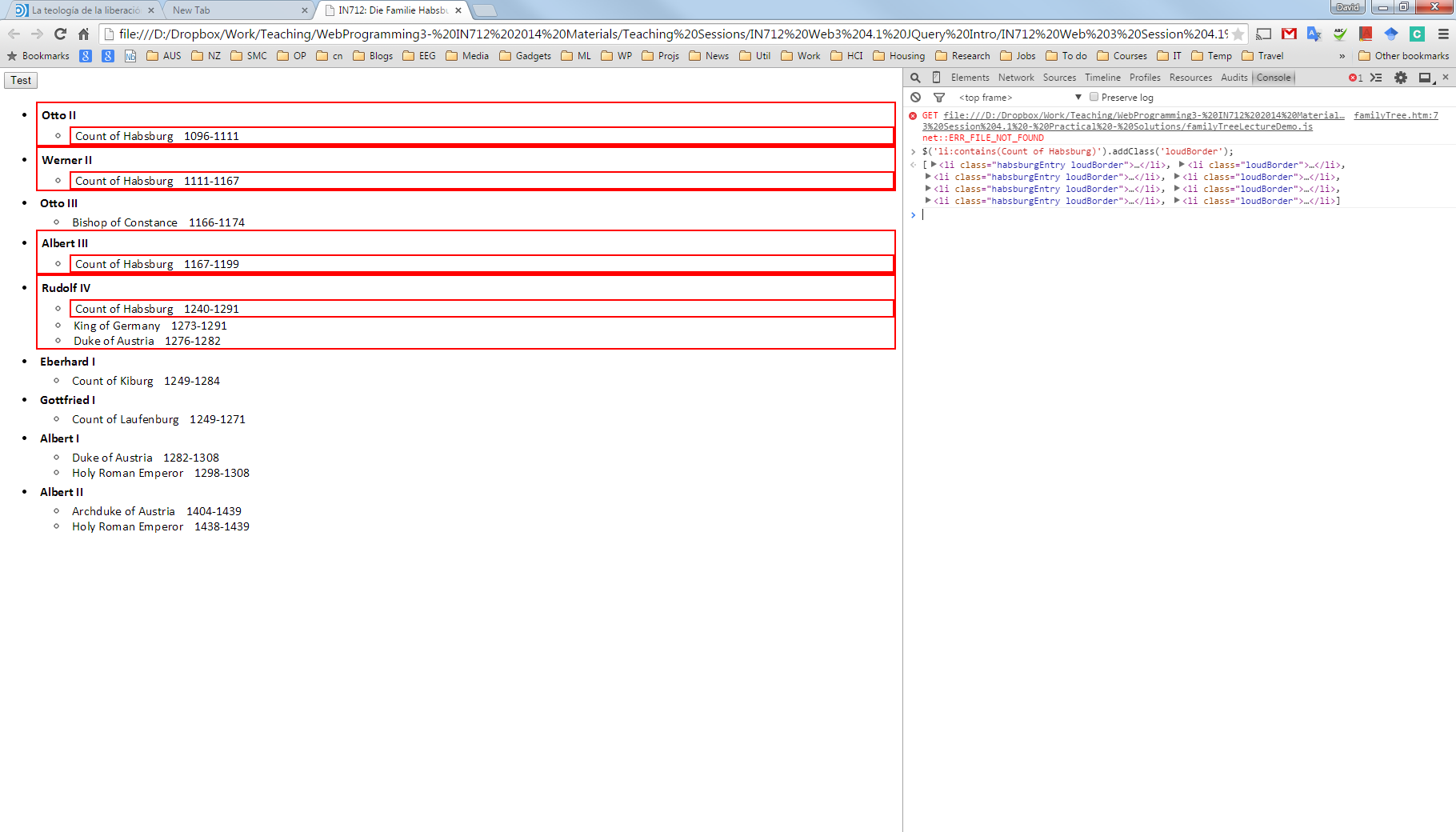
1. Grab all the li, then grab from that set, all those who are child number 2 (index start at 1). All the <li> are each entry (top level) and each titleList (nested). So this selector gets the second whole entry (Werner II), and the second title for everyone who has at least two.



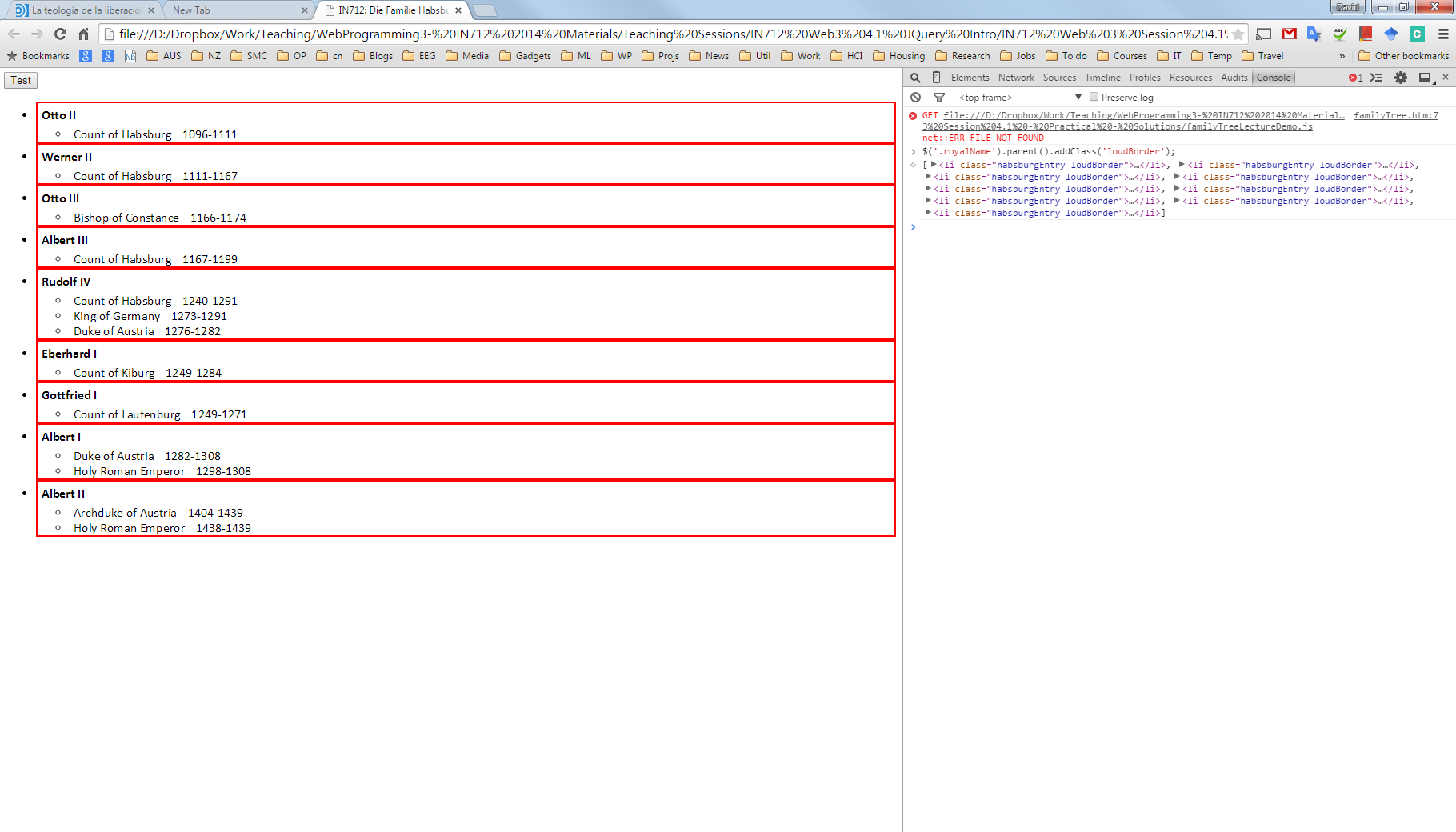
1. Use the :contains filter to fetch all the titles (just the span) for each guy who was Count of Habsburg? Here you can’t go by class, element, or position/relative position in the DOM. You really need to match the contents. Note that you don’t need additional quotes around the search string.



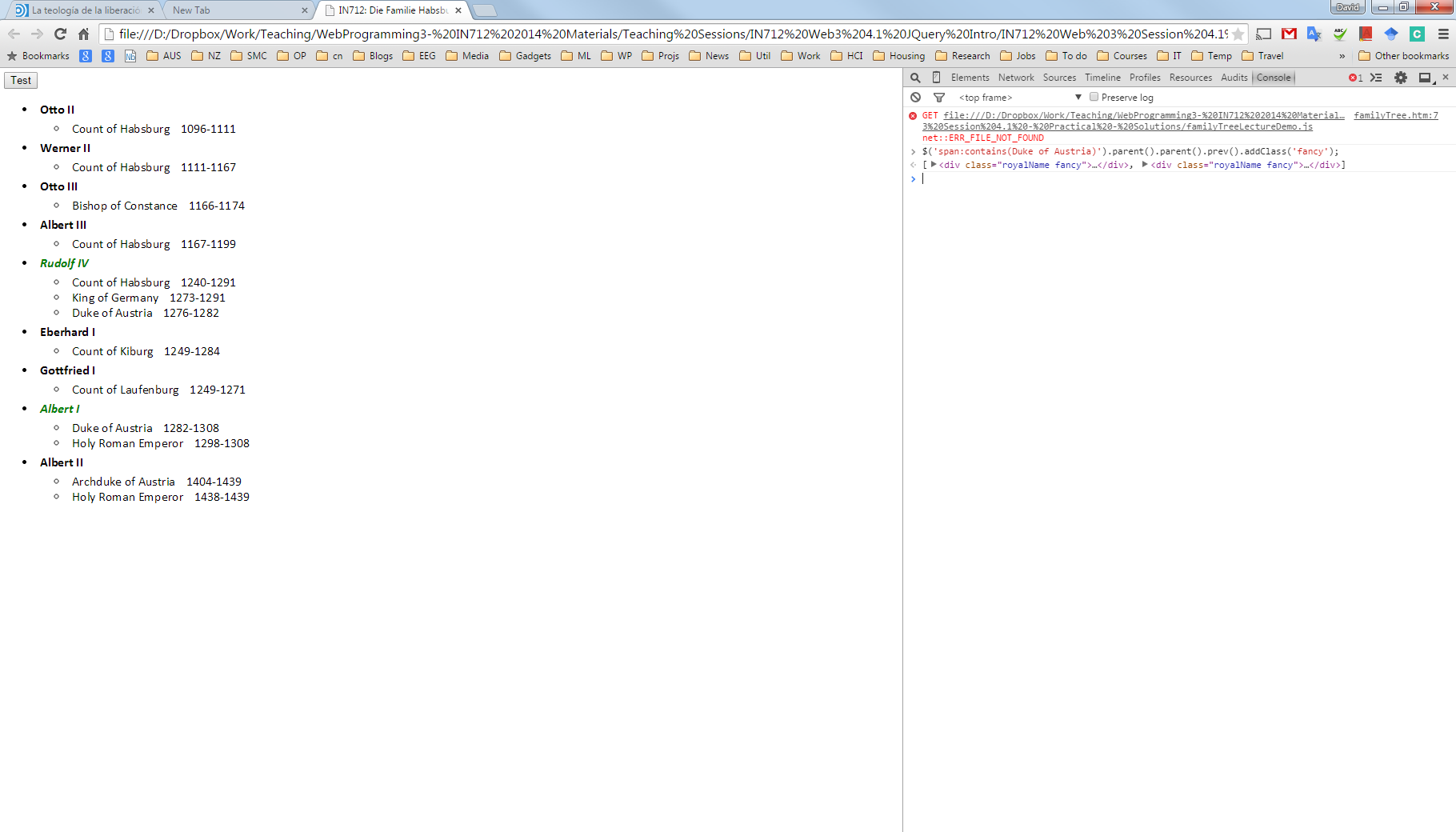
1. Select all <li> (top level and nested), then selects all those containing the string, and borders them. So you get double borders: both the spans that contain the string directly, and the entries that contain a span that contains the string.



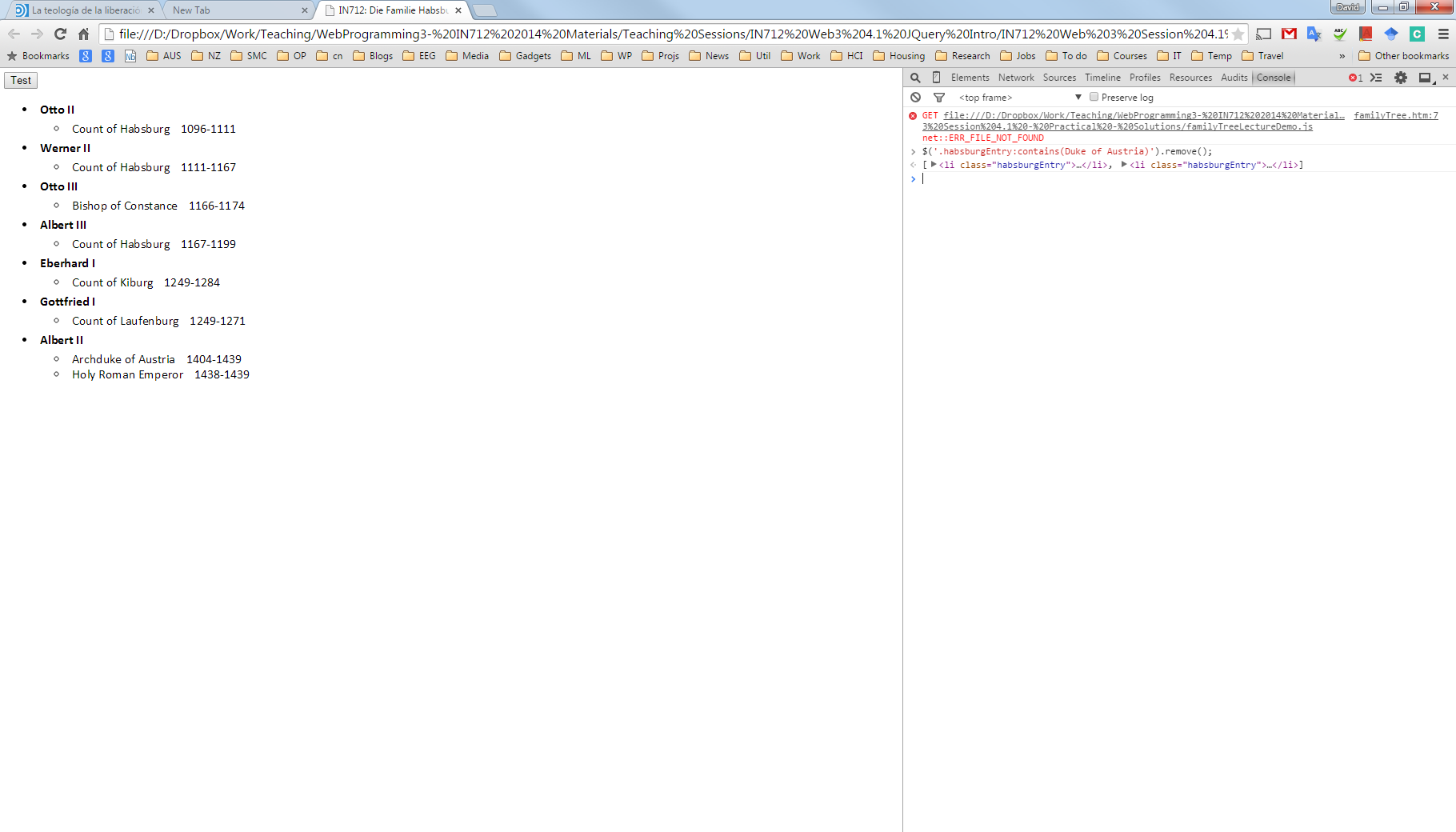
1. Get everyone of class royalName. Then select the parent node ***of each guy in that set***. Border them. These are the habsburgEntry <li>’s, and they each should get a border.



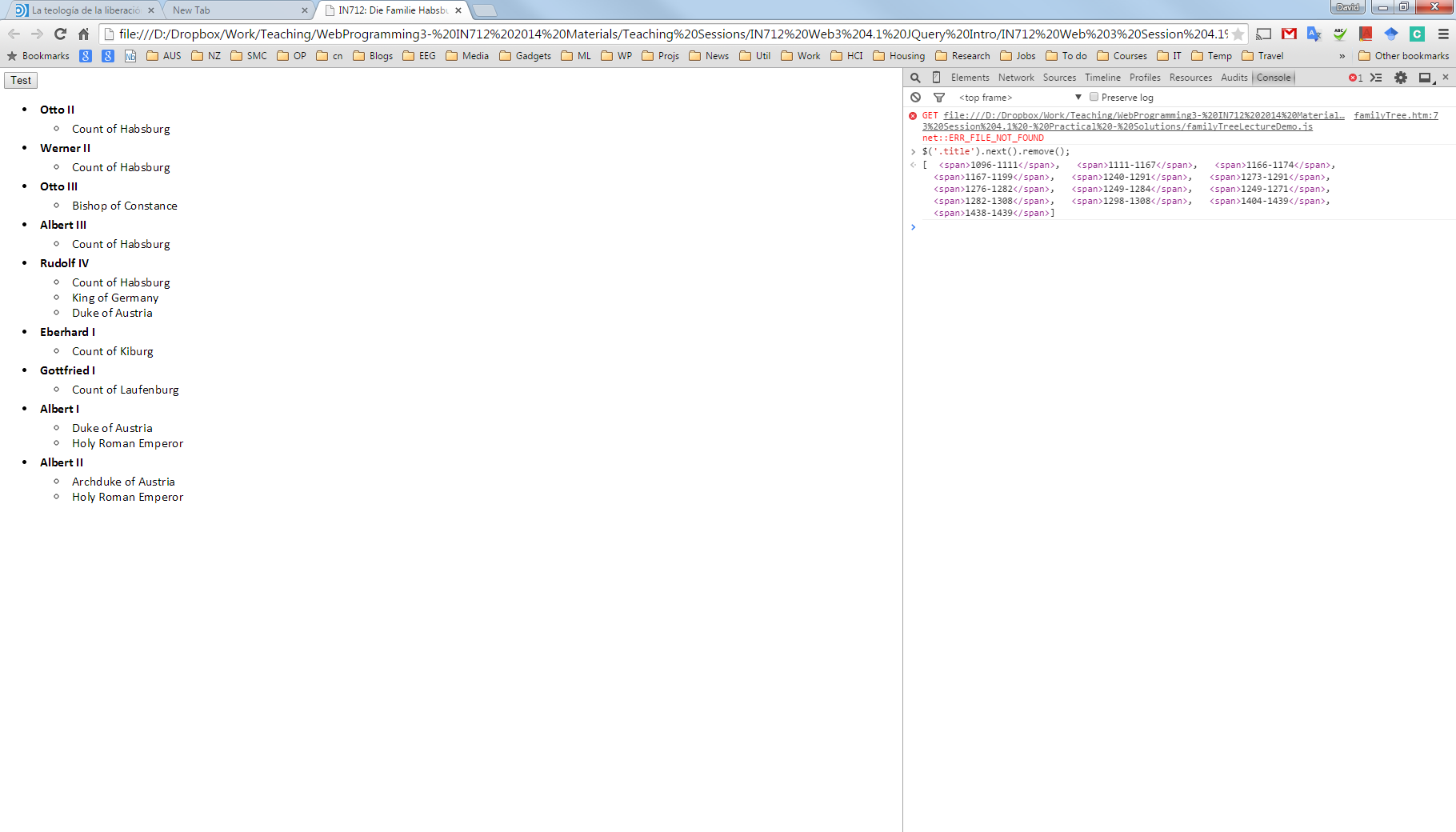
1. Select all the name (the royalName div) of guys who have been Duke of Austria and add the class ‘fancy’. How would you do it?



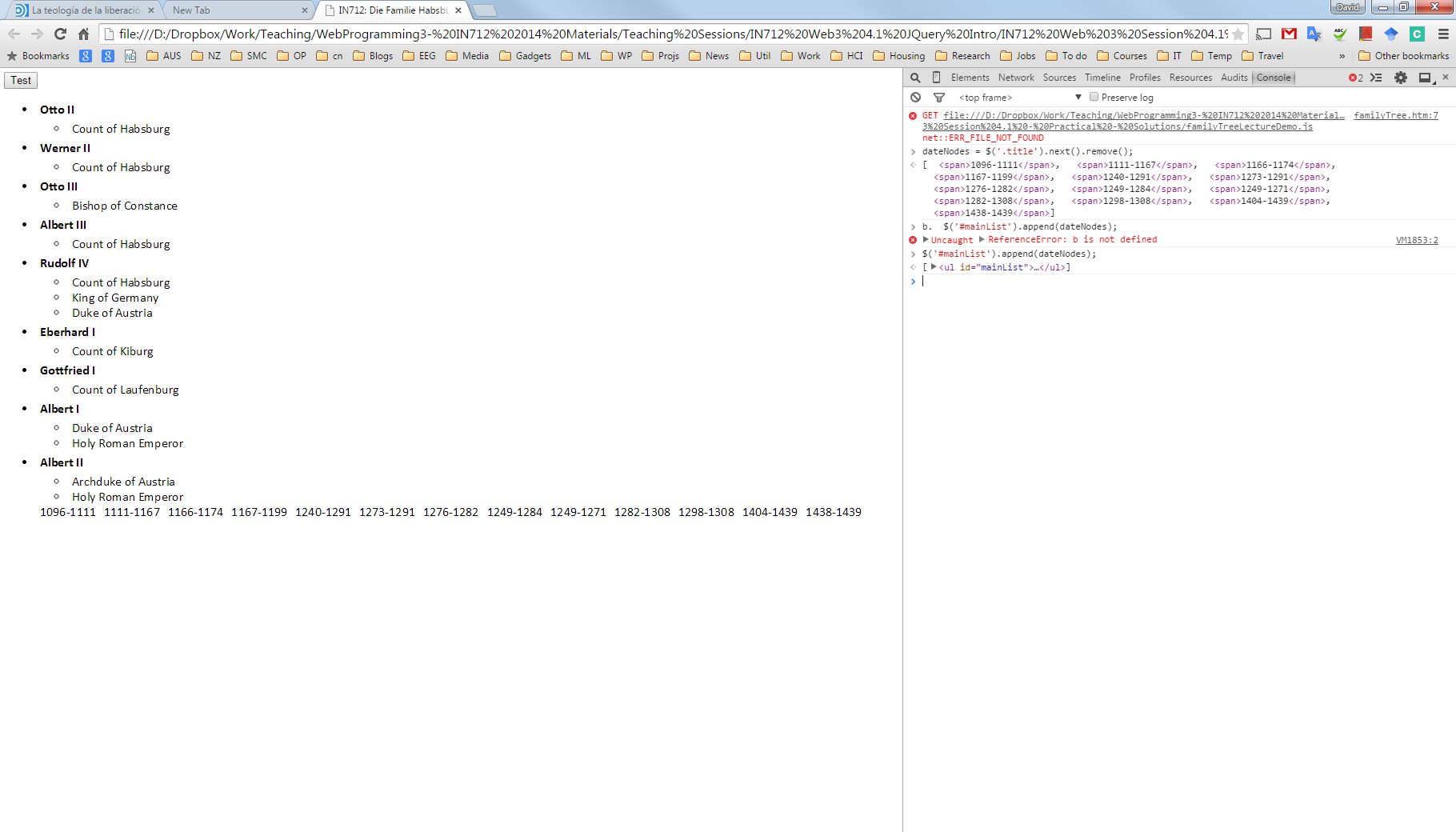
1. So far, we have just been changing the style, but once you have the set of elements, you can do lots of things to them. For example, you can remove them from the DOM. Delete the whole entry for Rudolf IV and Albert I



1. Remove all the dates and store them in a variable ‘dateNodes’



1. Append all the dates deleted at the bottom of the HTML document using the variable ‘dateNodes’.



1. Grab everyone who was Holy Roman Emperor and move their entries to the head of the list?

