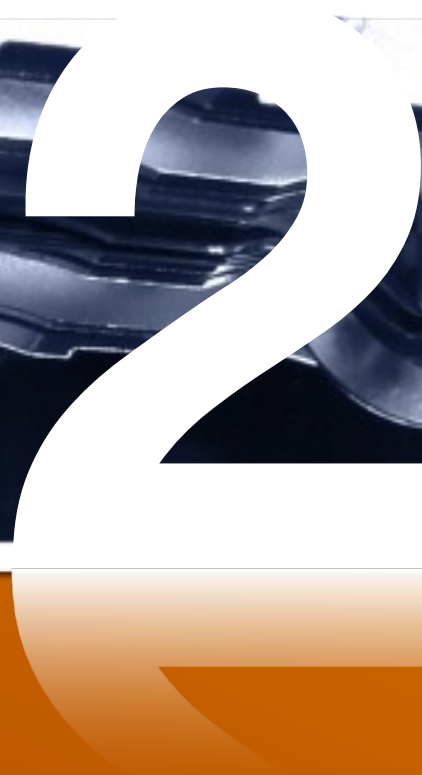




FULL SAIL
UNIVERSITY

programming for web applications



jQuery. **Lists. Trees. Tables**

jQuery.**Lists**

jQuery.Lists

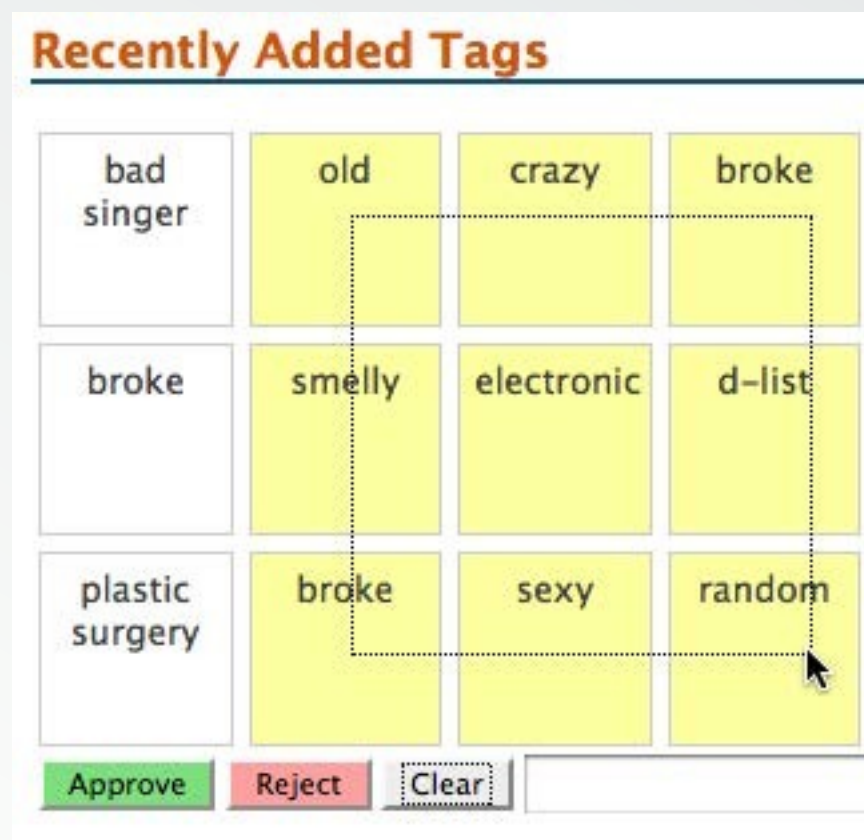
UI Selectables

- ▶ Lists are heroes of the post table-based layout period of the web.
- ▶ UI Selectable helps when selecting information based from tags
- ▶ Ex: The client wants a way to easily see tags, select them (and any duplicates), and click a button to approve or reject them.
- ▶ Making an element selectable gives the user the ability to lasso any of the element's children to select them: if you click on one element and then drag over subsequent elements, they are all highlighted.

jQuery.Lists

UI Selectables

- ▶ Selectable behavior also lets you add non-sequential items to the list using the Ctrl key (as you can do in most desktop applications).



Keep Users in the Loop

Making selections in this manner is a nonstandard form of interaction on the Web, so you'll need to provide instructions to your users about how to use your new functionality.

jQuery.Lists

Utility Methods: \$.map and \$.inArray

- ▶ \$.map method allows you to take each element in the array, process it in some manner, and return the results as a new array.
- ▶ You use it when you want to transform every element in the same way
- ▶ Can also be used directly on a selection where we could rewrite the name-grabbing code.

```
var names = $('.ui-selected, this').map(function(i, element){  
    return $(element).text();  
});
```

jQuery.Lists

Utility Methods: \$.map and \$.inArray

- ▶ \$.inArray method searches an array (but only plain JavaScript arrays, no jQuery selections) for a specific value.
- ▶ \$.inArray(value, array) , it will return the value's index in the array.
- ▶ -1 if the value is not found in the array

jQuery.Lists

Sorting Lists

- ▶ Ex: A client asks you to build some sorting capabilities into all the lists in the admin section, so that they can be sorted in ascending or descending alphabetical order with the click of a button.
- ▶ jQuery objects lack any built-in sorting functionality.
- ▶ We will fallback on some JavaScript array methods.
- ▶ Let's build a reusable list-sorting widget.

```
var SORTER = {};  
SORTER.sort = function(which, dir) {  
    //Check to see if desc was passed in as the dir parameter  
    SORTER.dir = (dir == "desc") ? -1 : 1;  
    $(which).each(function() {  
        // Find the list items and sort them  
        var sorted = $(this).find("> li").sort(function(a, b) {  
            return $(a).text().toLowerCase() > $(b).text().toLowerCase() ?  
            ↪SORTER.dir : -SORTER.dir;  
        });  
        $(this).append(sorted);  
    });  
};
```

jQuery.Lists

The sort Function

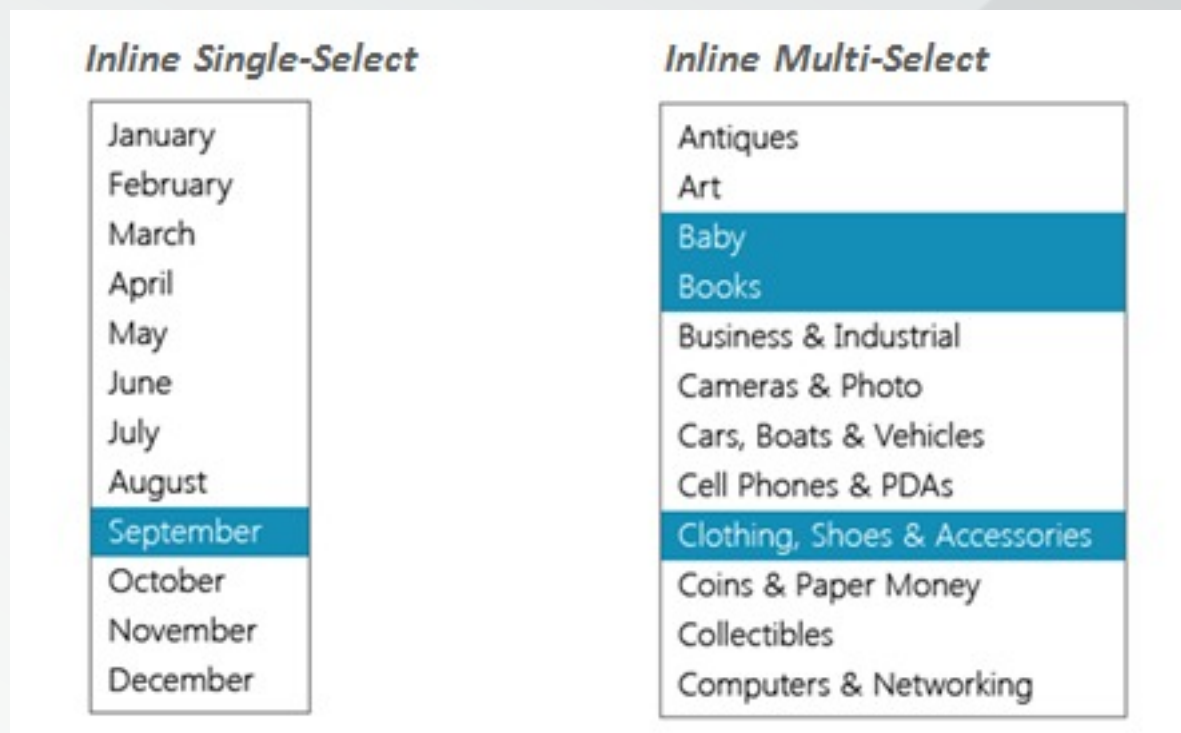
- ▶ The sort function is plain old JavaScript.
- ▶ It sorts an array based on either default rules such as sorting numerically for numbers and alphabetically for strings, or by the results of the function you pass to it.
- ▶ sort will go over the contents of the array and pass them to your function in pairs.
- ▶ If your function returns 1, sort will swap the items and place the second one first.
- ▶ If your function returns -1, JavaScript will put the first item first.
- ▶ Finally, if your function returns 0, sort will consider that both items are equal and no sorting will take place.

```
$( '#ascending' ).click(function() {  
    SORTER.sort('.sortable');  
});  
$( '#descending' ).click(function() {  
    SORTER.sort('.sortable', 'desc');  
});
```


jQuery.Lists

Manipulating Select Box List/Swapping List Elements

- ▶ Ex: Client wants to improve the admin functionality for assigning top clients to a specific list.
- ▶ Functionality consists of two select elements: top clients and all other clients
- ▶ Client wants to be able to easily swap top clients between list, if they no longer become top clients.
- ▶ We need to build a SWAPLIST object that will contain all the functionality we'll build.



jQuery.Lists

Inverting a Selection

- ▶ Ex: Client requests to add a button that invert the current selection, to make it easier for the staff when dealing with large selections.
- ▶ When this link is clicked, all currently selected items in the target list become deselected, and vice versa.
- ▶ To do this: we retrieve every list item and swap its selected property using the prop action
- ▶ prop is similar to attr, except is is used to se properties instead of attributes.
- ▶ if boolean (True or False) use prop

jQuery.Lists

Searching through Lists

- ▶ A quick search feature lets the user type some characters and then automatically select any matching elements
- ▶ First, grab all list items then clear any previous selections (by setting selected to false)
- ▶ Use the filter action accepts a function as a parameter and runs that function against every jQuery object in the selection.
- ▶ If the function is true the element stays in selection, otherwise false, selection is gone
- ▶ To check to see if the text they contain has the text we're looking for we use the text action
- ▶ Attach a keyup handler to the input itself, so it selects as you type

```
$('#search').keyup(function() {  
    SWAPLIST.search("#a-listers, #candidates", $(this).val());  
});
```

jQuery.Trees

jQuery.Trees

Expandable Tree

- ▶ Trees can be thought of as categories
- ▶ They are hard to do well
- ▶ There are few situations where they make sense.
- ▶ Just nested lists
- ▶ Trees can nest as far as is needed



```
graph TD
    A["- A-list Celebrities"] --> B["+ In a successful band"]
    A --> C["- In an indie band"]
    A --> D["+ In film or television"]
    A --> E["- Famous because they're rich"]
    B --> B1["+ In film or television"]
    B --> B2["+ Famous parents"]
    B --> B3["+ Dot-Com millionaires"]
    C --> C1["☐ Computadors"]
    C --> C2["☐ The Great Apes"]
    C --> C3["☐ Bosom"]
    E --> E1["+ B-list Celebrities"]
    E --> E2["+ C-list Celebrities"]
    E --> E3["+ Up and Comers"]
```

- A-list Celebrities
+ In a successful band
- In an indie band
 ☐ [Computadors](#)
 ☐ [The Great Apes](#)
 ☐ [Bosom](#)
+ In film or television
- Famous because they're rich
 + Famous parents
 + Dot-Com millionaires
+ B-list Celebrities
+ C-list Celebrities
+ Up and Comers



jQuery.Trees

Advanced toggleClass

- ▶ The toggleClass accepts a second parameter: a Boolean value that specifies whether the class should be added or removed. It's a nice shortcut.

```
if(x==1){  
    $(this).addClass('opened');  
} else {  
    $(this).removeClass('opened');  
}
```

With the toggleClass(class, switch) syntax, we can replace the if statement with the following concise syntax:

```
$(this).toggleClass('opened', x == 1);
```


jQuery.Trees

Event Delegation

- ▶ Dealing with large trees
- ▶ If we added a click event handler to every list item — `$('#picker li').click(...)` — we could end up with hundreds of handlers
- ▶ With event delegation, we add our lone event handler to the parent of the list, and then access the target of the event to determine the element that was actually clicked on
- ▶ The event will bubble up until the parent handler catches them (Event Propagation)
- ▶ If the event is stopped on its way up, event delegation will fail.

```
$('#picker').click(function(e) {  
    $('#current').text($(e.target).text());  
});
```



jQuery.Tables

jQuery.Tables

Fixed Table Headers

- ▶ Keeping the header row fixed at the top of the table is an effective way to keep track of what our columns are about
- ▶ As the user scrolls the table to reveal new data, the header follows along.

Id	Name	Occupation	Approx. Location	Price
2031	Mo'Nique	Producer	New York	\$19.95
007F	Kellie Kelly	Singer	Omaha	\$11.95
8A05	Darth Vader	DJ	London	\$19.95

jQuery.Tables

Fixed Table Headers

- ▶ Append as Infrequently as Possible
- ▶ the method chosen executes much more quickly in the browser
- ▶ Every time you insert a new element into the DOM, the browser needs to recalculate the position of every element on the page.
- ▶ If you do this a lot your script can become very slow
- ▶ Storing new elements in a variable, processing them as necessary, appending them all at one time ensures optimal performance.

```
var $list = $('<ul class="faux-head"></ul>');
$ths.each(function(i) {
    _th = $(this);
    $list.append($("<li></li>")
        .addClass(_th.attr("class"))
        .html(_th.html())
        .width(_th.width())
        .click(function() {
            _th.click()
        })
    ).hide().css({left: $table.data('left'),top:
    $table.data('top')});
});
$('body').append($list);
```

jQuery.Tables

Repeating Header

- ▶ Another Approach: repeat the header at regular intervals
- ▶ Good if the intention is for the data to be printed out
- ▶ Goal: take the first row of the table and copy it every ten rows.(aprox.)

Id	Name	Occupation	Approx. Location	Price
203A	Johny Stardust	Front-man	Los Angeles	\$39.95
6636	Glendatronix	Keytarist	London	\$39.95
141B	Beau Dandy	Singer	New York	\$39.95
2031	Mo' Fat	Producer	New York	\$19.95
007F	Kellie Kelly	Singer	Omaha	\$11.95
8A05	Darth Fader	DJ	London	\$19.95
203A	Johny Stardust	Front-man	Los Angeles	\$39.95
6636	Glendatronix	Keytarist	London	\$39.95
141B	Beau Dandy	Singer	New York	\$39.95
2031	Mo' Fat	Producer	New York	\$19.95
Id	Name	Occupation	Approx. Location	Price
007F	Kellie Kelly	Singer	Omaha	\$11.95
8A05	Darth Fader	DJ	London	\$19.95

```
$('#repeatheader')  
  .find('tr:first')  
  .clone()  
  .insertAfter('#reapeatheader tr:nth-child(10n)');
```

You can use the letter “n” to indicate repetition; for example, :nth-child(10n) will select every tenth row. A plus or minus to offset which rows are selected.

:nth-child(10n+3) or
:nth-child(10n-7)

jQuery.Tables

Data Grids

- ▶ Ex: Client wants to replace the old desktop application that the marketing manager uses. It hooks into the same database, but lets her sort and move around the data, and edit the different cells all on the one page. Can you do it
- ▶ There's no set definition for what constitutes a data grid, but some of its common features include sorting, filtering, searching, paging, column resizing, and row editing

My CD Collections							
	Album	Artist	Year	Origin	With Poster?	Price	
1	Dearest	Theresa Fu	2009	Hong Kong ↕	<input checked="" type="checkbox"/>	168.9	↶ ☒ ↗ ↘
2	To be Free	Arashi	2010	Japan ↕	<input checked="" type="checkbox"/>	152.6	↶ ☒ ↗ ↘
3	Count On Me	Show Luo	2012	Taiwan ↕	<input type="checkbox"/>	306.8	↶ ☒ ↗ ↘
4	Wonder Party	Wonder Girls	2012	Korea ↕	<input checked="" type="checkbox"/>	108.6	↶ ☒ ↗ ↘
5	Reflection	Kelly Chen	2013	Hong Kong ↕	<input type="checkbox"/>	138.2	↶ ☒ ↗ ↘
+ ×							

jQuery.Tables

Pagination

- ▶ Adding paging to a table lets us display a small subset of the data at any one time, while allowing the user to move through it easily with navigation buttons.
- ▶ If Pagination is often handled by the server.
- ▶ The user can request a specific page of data and the server will return it.

Celebs				
<div>< 1 of 3 ></div>				
Id	Name	Occupation	Approx. Location	Price
203A	Johny Stardust	Front-man	Los Angeles	\$39.95
6636	Glendatronix	Keytarist	London	\$39.95
141B	Beau Dandy	Singer	New York	\$39.95

jQuery.Tables

Adding the Controls Dynamically

- ▶ An option would be to build the navigation controls completely in jQuery
- ▶ It can be easily applied it to any table, and the controls would be added automatically
- ▶ Note: Such an approach is often favored by plugin authors
- ▶ It's a good idea to hide the controls container in CSS and then show it with jQuery when the page loads
- ▶ Reason: Anyone without JavaScript enabled can avoid seeing the redundant controls

jQuery.Tables

Editing a Row

- Add editing functionality to the data grid by inserting an edit button at the end of each row, turning the entire row of cells into input elements

Celebs					
Id	Name	Occupation	Approx. Location	Price	
203A	Johnny Stardust	Front-man	Los Angeles	\$39.95	Edit
<input type="text" value="66"/>	<input type="text" value="Glendatronix"/>	<input type="text" value="Keytarist"/>	<input \""="" http:="" type="text" value="	<input type="text" value="\$39."/>	<input type="button" value="save"/>
141B	Beau Dandy	Singer	New York	\$39.95	Edit

jQuery.Tables

Data Grid Plugins

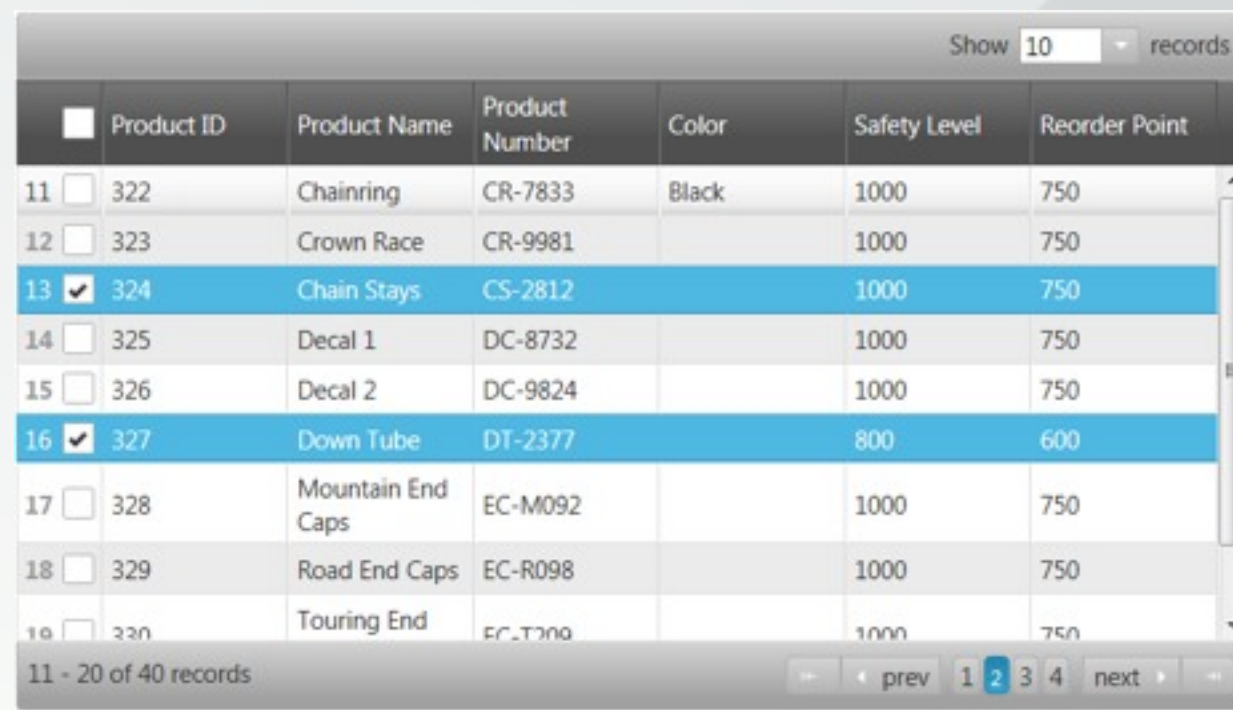
- ▶ DataTables is a truly impressive plugin for turning your HTML tables into fully functional data grids, complete with pagination, column sorting, searching, ThemeRoller support, Ajax loading, and more.
- ▶ As always, the decision to use a plugin or build custom functionality comes down to how much time you have, how many features you need, and how big a file you're willing to serve to your visitors.

My CD Collections							
	Album	Artist	Year	Origin	With Poster?	Price	
1	Dearest	Theresa Fu	2009	Hong Kong ▾	<input checked="" type="checkbox"/>	168.9	   
2	To be Free	Arashi	2010	Japan ▾	<input checked="" type="checkbox"/>	152.6	   
3	Count On Me	Show Luo	2012	Taiwan ▾	<input type="checkbox"/>	306.8	   
4	Wonder Party	Wonder Girls	2012	Korea ▾	<input checked="" type="checkbox"/>	108.6	   
5	Reflection	Kelly Chen	2013	Hong Kong ▾	<input type="checkbox"/>	138.2	   
 							

jQuery.Tables

Selecting Rows with Checkboxes

- ▶ Ability to select rows of a table (or any kind of list) by checking a checkbox associated with the row.
- ▶ As selections being able to be made on an individual basis, there are often shortcuts for selecting all items, selecting none, or inverting the current selection of checkboxes.
- ▶ Some users also want to be able to select continuous rows by using the Shift key



The screenshot shows a web interface for a table. At the top right, there is a 'Show 10 records' dropdown. The table has 7 columns: a checkbox column, Product ID, Product Name, Product Number, Color, Safety Level, and Reorder Point. Rows 13 and 16 are selected, indicated by blue highlighting and checked checkboxes. The bottom of the interface shows pagination: '11 - 20 of 40 records' and a set of navigation buttons including 'prev', '1', '2' (highlighted), '3', '4', 'next', and arrows.

<input type="checkbox"/>	Product ID	Product Name	Product Number	Color	Safety Level	Reorder Point
11 <input type="checkbox"/>	322	Chainring	CR-7833	Black	1000	750
12 <input type="checkbox"/>	323	Crown Race	CR-9981		1000	750
13 <input checked="" type="checkbox"/>	324	Chain Stays	CS-2812		1000	750
14 <input type="checkbox"/>	325	Decal 1	DC-8732		1000	750
15 <input type="checkbox"/>	326	Decal 2	DC-9824		1000	750
16 <input checked="" type="checkbox"/>	327	Down Tube	DT-2377		800	600
17 <input type="checkbox"/>	328	Mountain End Caps	EC-M092		1000	750
18 <input type="checkbox"/>	329	Road End Caps	EC-R098		1000	750
19 <input type="checkbox"/>	330	Touring End	EC-T200		1000	750

jQuery.Tables

Selecting a Column of Checkboxes

- ▶ Dealing with columns of data is much trickier than dealing with rows of data.
- ▶ Grab the table
- ▶ Try to find the checkboxes in the first column of each row
- ▶ Retrieve the table rows, `#celebs tr` , and then the first columns of each row, `#celebs tr td:nth-child(1)` .
- ▶ Only selecting checkboxes: `#celebs tr td:nth-child(1) :checkbox` .
- ▶ Add another checkbox (with an id of checker) in the thead of our table
- ▶ Attach a click handler to turn on/off all the checkboxes by setting all the checkboxes' checked property to that of the top checkbox
- ▶ The selector is stored as a string (chkSelector) because we want to reuse it later.

```
var chkSelector = 'tr td:nth-child(1) :checkbox';

$('#checker').click(function() {
    $('#celebs ' + chkSelector)
        .prop('checked', $(this).prop('checked'));
});
```


jQuery.Tables

Shift-selecting Checkboxes

- ▶ The event state of the Shift key can check the Boolean property by the e.shiftKey. Its contained within the click event itself
- ▶ We can also check the ctrlKey , altKey , and metaKey in the same manner.

	Genre	Artist	Year	Album
<input type="checkbox"/>	Easy Listening	Bette Midler	2003	Bette Midler Sings the Rosemary Clooney Songbook
<input type="checkbox"/>	Classic Rock	Jimi Hendrix	1993	Are You Experienced
<input type="checkbox"/>	Jazz	Andy Narell	1992	Down the Road
<input type="checkbox"/>	Progressive Rock	Emerson, Lake & Palmer	1992	The Atlantic Years
<input type="checkbox"/>	Rock	Blood, Sweat & Tears	1968	Child Is Father To The Man
<input type="checkbox"/>	Jazz	Andy Narell	1989	Little Secrets

week3.Assignment9

assignment.0309

Today's Assignment: List, Trees, Table **Due:** By End of Lab

- **Make 1 list (must validate), 1Tree and 1 Table**
- **ALL** html/css/js completed, only 1 stylesheet file for the entire project
- Turn into GitHub Repo: **0309_list.zip**

This Week's Milestone: Ajax **Due:** Tuesday (By the End of Lab)

- **ALL** html/css markup completed, ajax a must in deliverable
- filler content (**NO** *lorem ipsum*) must be used inside html to test your design
- **ALL** components of your app as HTML (i.e. landing.html, registration.html)
- only 1 stylesheet file for the entire project
- *each html page should **look** like it would when live.*
- Turn into GitHub Repo: **milestone3_ajax_lastname_firstname.zip**