The **Page Manager** is a tool that users of BrandingPays web tools will have access to. There will be a Page Manager button on every page that, when clicked, will open up either a modal Dialog or a movable <div> that appears over everything else. This tool will act similarly to file managers/file browsers such as Windows Explorer or a Mac’s Finder in that it will provide a list of all the “pages” (a “page” being a single Positioning Statement, Elevator Pitch, or Brand Strategy) the user has created and the pages’ properties in a table format.

*What the Page Manager should (sort of) look like (see also svn🡪brandingpays🡪trunk🡪app🡪images🡪Designs🡪(CS4)PageManager.ai)*

PageManager.tiff

The Page Manager will use the dojox.grid.EnhancedGrid to display the list of pages the user is working on.

**Requirements for the Page Manager grid:**

The grid you see in the above picture was produced by the code in *svn🡪brandingpays🡪trunk🡪exampleDojoEnhancedGrid3.html.* The code for this grid is far from complete, however.

Here some of are the features the grid currently has:

* Ability to select rows
* Ability to select multiple rows by clicking on the left-most column of boxes and dragging the mouse up or down, or by selecting one row and then selecting another by pressing shift
* The correct column headings

What features need to be added:

1. Code that processes server data (JSON-formatted) so that it can be inserted into the grid dynamically

2. A way for the Date Created and Date Modified columns to be sorted properly

3. Currently dates that would populate the Date Created and Date Modified columns are in this format: “YYYY-MM-DD”. Is there a way to format them like “January/Jan 1, 2011” and still have the column sort properly? If not that’s ok.

4. The grid should initially be rendered sorted by Type, or more specifically in the order of: 1. Type = “Positioning Statement” 2. Type = “Elevator Pitch” 3. Type = “Brand Strategy” (the order in which each are filled out by the user). If possible, after the grid is sorted by type, it also has “nested sorting” in which it is sorted by Date Created as well.

5. When the Select All button is clicked, all the rows get selected.

6. Once one or multiple rows are selected, the functions Edit, Delete, Print, and Email need to apply to the selected pages.

7. When the Search button is clicked (it should actually by labeled “Search By” as opposed to what is shown in the above image), a drop down menu should appear that has as options all of the column headings: Title, Type, Tags, Date Created, and Date Modified. When the user selects an option, a pop up should appear with a field into which the user can enter his/her search terms. Maybe if the user choose Date Created or Modifid, a date picker should replace the text field. Once the user clicks some buttons that submits ther search terms (probably a “Search” button), the grid is then filtered to show only search results.

**Detailed Notes and Explanations about the Requirements:**

1. “Code that processes server data (JSON-formatted) so that it can be inserted into the grid dynamically”

The example file *exampleDojoEnhancedGrid3.html* populates the grid statically. The grid is created programmatically rather than declaritively:

// create a new grid (line 103):

var grid = new dojox.grid.EnhancedGrid({

query: {

Title: '\*'

},

store: store,

rowSelector: '20px',

structure: layout,

clientSort: true,

sortFields: {attribute: 'Type'},

plugins: {

nestedSorting: true,

dnd: true,

indirectSelection: {

name: "Selection",

width: "70px",

styles: "text-align: center;"

}

}

},

document.createElement('div'));

// append the new grid to the div "gridContainer4":

dojo.byId("gridDiv").appendChild(grid.domNode);

The line in orange is how the grid gets its data: it gets it from its *store* variable, which in turn gets the data from another variable named *store* that I defined:

var store = new dojo.data.ItemFileReadStore({

data : {items:

[

{'Title': 'My Positioning Statement',

'Type': 'Positioning Statement',

'Tags': 'One\_Piece',

'date\_created': 'Jan 24, 2011',

'date\_modified': 'Feb 15, 2011'},

{'Title': 'My Positioning Statement2',

'Type': 'Positioning Statement',

'Tags': 'Luffy',

'date\_created': 'Jan 5, 2011',

'date\_modified': 'Feb 10, 2011'},

{'Title': 'My Elevator Pitch',

'Type': 'Elevator Pitch',

'Tags': 'Nami',

'date\_created': 'Feb 1, 2011',

'date\_modified': 'Feb 10, 2011'},

{'Title': 'My Elevator Pitch2',

'Type': 'Elevator Pitch',

'Tags': 'Zoro',

'date\_created': 'Feb 2, 2011',

'date\_modified': 'Feb 10, 2011'},

{'Title': 'My Brand Strategy',

'Type': 'Brand Strategy',

'Tags': 'Sanji',

'date\_created': 'Feb 1, 2011',

'date\_modified': 'Feb 10, 2011'}

]

}

});

As you can see, I just plugged in some static data to get it working. [[Get tags working on all 3 pages so can show how they should go about reworking data to be in right format]]

{"PosnStmt":{"UArray":[{"Title":"The Grand Line","TargetAudience":"a","ProblemStatement":"b","TitleAndFunction":"c","ValueProposition":"d","Differentiation":"e","Evidence":"f","Tags":"Luffy Nami One\_Piece"}],"UIndex":0},"ElevatorPitch":{"UArray":[{"Title":"BrandingPays","5second":"g","15second":"h","30second":"i","Evidence":"j","Tags":"Naruto Sasuke Sakura"}],"UIndex":0},"Strategy":{"UArray":[{"Title":"Brand Strategy2001","CoreValue1":"k","CoreValue2":"l","CoreValue3":"m","CoreValue4":"n","HardSkills":"o","PersonalityAttributes":"p","BrandAssociations":"q","RationalValue":"r","WhatILove":"s","SoftSkills":"t","TypeOfLeader":"u","ExternalImage":"v","EmotionalValue":"w","CareerDreams":"x","Expertise":"y","BrandMetaphor":"z","RelationshipImage":{},"BrandExperience":{},"KeyDescriptors":{"1":{},"2":{},"3":{}},"Tags":"Guts Casca Griffith"}],"UIndex":0},"ActionPlan":{"UArray":[{"Title":{}}],"UIndex":0},"\_date":"2011-02-17","\_time":"13:47:45","\_timezoneoffset":8,"\_version":"0.2","\_updateType":"full","UI":{"NavTab":"brand\_strategy\_div"}}