

[illegible]

[illegible]

[illegible]

[illegible]

```

        j--;
        i--;
    }
}
return theList;
}

- (List *)query:(const char *)theQuery field:(const char *) theField {
    id          theDocInfo;
    HashTable   *docInfoTable;
    long        key;
    NXHashState state;
    static List*theList=nil;

    if (!theList) theList = [[List alloc] init];
    [theList empty];
    docInfoTable = [eTDocInfo docInfoTable];
    state = [docInfoTable initState];
    while ([docInfoTable nextState:&state key:(void **)&key
           value:(void**) &theDocInfo])
        if ([theDocInfo searchFor:NXUniqueString(theQuery)
            in:NXUniqueString(theField)])
            [theList addObject:theDocInfo];
    return theList;
}

- (List *)queryByPanel {
    id          theDocInfo;
    HashTable   *docInfoTable;
    long        key;
    NXHashState state;
    static List*theList=nil;
    const char *iqA, *iqD, *iqK, *iqC;

    if (!theList) theList = [[List alloc] init];
    [theList empty];
    docInfoTable = [eTDocInfo docInfoTable];
    state = [docInfoTable initState];
    iqA = [iqAuthor stringValue];
    iqD = [iqDocTitle stringValue];
    iqK = [iqKeyword stringValue];
    iqC = [iqComments stringValue];
    while ([docInfoTable nextState:&state key:(void **)&key
           value:(void**) &theDocInfo]) {
        if ((*iqA && [theDocInfo searchFor:iqA in:AUTHOR]) ||
            (*iqD && [theDocInfo searchFor:iqD in:DOCTITLE]) ||
            (*iqK && [theDocInfo searchFor:iqK in:KEYWORDS]) ||

```

```
( *iqC && [theDocInfo searchFor:iqC in:COMMENTS]))
```

```
[theList addObject:theDocInfo];
```

```
}  
return theList;  
}
```

```
//DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD  
// AppKit Delegate  
//
```

```
static int Navigator_doccompare(eTDocInfo **x, eTDocInfo **y)  
{  
    return strcmp([*(eTDocInfo **)x docTitle],[*(eTDocInfo **)y docTitle]);  
}  
  
- (int) browser:theBrowser fillMatrix:theMatrix inColumn:(int) col {  
    int rows,i,count;  
    id cell;  
    List * result;  
    BOOL allLeaves=NO;  
  
    if (theBrowser == btBrowser) {  
        result = [self btList];  
    } else if (col == 0) {  
        if (shouldUseQP)  
            result = [self queryByPanel];  
        else result = [self query:"^$" field:PARENT];  
    } else {  
        if ([[theBrowser matrixInColumn:col-1 selectedCell] docID]) {  
            result = [self query:[[[theBrowser matrixInColumn:col-1]  
                selectedCell] docIDStr] field: PARENT];  
        } else {  
            result = [self query:"*" field: "*"];  
            allLeaves = YES;  
        }  
    }  
  
    if (theBrowser == navBrowser) {  
        // Sort the list alphabetically  
        qsort(result->dataPtr, result->numElements,sizeof(id),  
            Navigator_doccompare);  
    }  
  
    count = [result count];  
    if ((theBrowser == navBrowser) && (col == 0)){  
        rows = count+1;  
        [theMatrix renewRows:rows cols:1];  
        cell = [theMatrix cellAt:count :0];  
        [cell setListAllMode]; // add the "All Known Documents" entry
```

```

        [cell setLeaf:NO];
    } else {
        rows = count;
        [theMatrix renewRows:rows cols:1];
    }

    for (i=0; i <count; i++) {
        cell = [theMatrix cellAt:i :0];
        [cell setDocInfo:[result objectAt:i]];
        if ((theBrowser == btBrowser) || allLeaves) [cell setLeaf:YES];
        else [cell setLeaf:![self isParent:[result objectAt:i]]];
    }
    return rows;
}

- windowDidUpdate:sender {
    static char navPath[2*MAXPATHLEN]; // I suspect the previous declaration of 3k
    on the stack (non-static) was smashing structures randomly.

    [navPanel disableFlushWindow];
    if (navDirty) {
        [navBrowser getPath:navPath toColumn:([navBrowser lastColumn]+1)];
        [navBrowser loadColumnZero];
        [navBrowser setPath:navPath];
        navDirty = NO;
    }
    if (btDirty) {
        [btBrowser loadColumnZero];
        btDirty = NO;
    }
    [[navPanel reenableFlushWindow] flushWindow];
    return self;
}

@end

```