Task List

NOTICE: This task list has been superceded by the author with the Interactive Task List. While you are still free to use this one the Interactive Task List should give all the same capabilities and more.

I have been taking advantage of the various resources and examples listed on this site as well and the forums so I figured it was about time I gave something back. Below are some instructions on how I created a task list for our company intranet using templates within MindTouch.

Goal: To have the ability to insert a standard tasklist within any page as simply as possible. The tasks should be easy to read, sortable, as well as have the ability to be searchable for all tasks assigned to a user so they dont need to go looking around for them. Multiple users should also be able to be assigned to a single task item.

Result: All my users have a task list page of their own and I have also embeded the task list template within other templates for example: Our meeting minutes template has a task list section so that tasks that are assigned to people in the meeting can be tracked. The item in this task list are then aggregated under thier personal task list within thier user pages. Also management has a special page that they can visit that will aggregate ALL open items for every user so that they can see the current workload of the company. This works great for our company of around 50 people, but if your company is much larger you may need to modify my examples to fit your needs.

You can see an example of the script in action HERE.

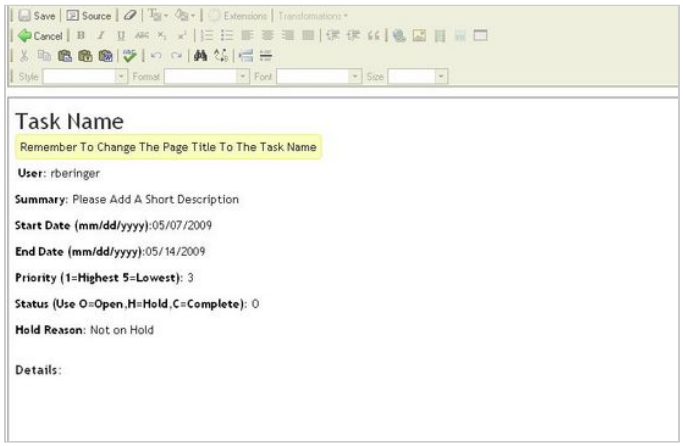
Requirements:

1. Lyons 9.02 or greater
2. Template:TaskList
3. Template:TaskItem
4. Tempate:TaskListMaster
5. Optional: Sortable Tables Script From HERE (Gives it more functionality)

I have attached the required templates (just paste into sort mode) and a couple of examples of what the Task List would look like.

NOTICE: It should be noted that I am new to dekiscript and I am confident that others could make the code more readable and more efficient, but it does work for my needs and possibly will yours.

Template:TaskItem

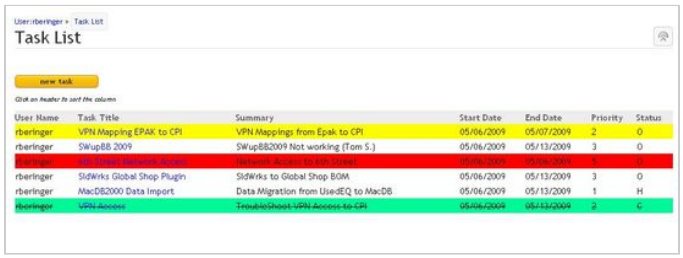


The first template that I will show is the TaskItem. This is the indivitual task that is assigned. Notice that each field of interest is wrapped in a field and has a class assigned to it. This allowed me to target the specific pieces of information needed for my task list. You can assign multiple users to a task by listing thier username seperated by a comma or semi-colon. Below is the raw HTML of my TaskItem Template.

```
<h1>Template:TaskItem</h1>
<p><span class="comment">Remember To Change The Page Title To The Task Name</span></p>
<p> <strong>User:</strong><span class="username"> {{ edit: user.name}} <br />
</span></p>
<p><strong>Summary:</strong><span class="summary"> Please Add A Short
Description</span></p>
<p><strong>Start Date (mm/dd/yyyy):</strong><span class="startdate">{{ edit:
date.format(date.changetimezone(date.now, user.timezone), <span style="font-size:
small;">"MM/dd/yyyy"</span>) }}</span></p>
<p><strong>End Date (mm/dd/yyyy):</strong><span class="enddate">{{ edit:
date.format(date.changetimezone(date.adddays(date.now, 7), user.timezone), <span
style="font-size: small;">"MM/dd/yyyy"</span>) }}</span></p>
<p><strong>Priority (1=Highest 5=Lowest):</strong><span class="priority"> 3<br />
</span></p>
<p><strong>Status (Use O=Open,H=Hold,C=Complete):</strong><span class="status"> O<br />
</span></p>
<p><strong>Hold Reason:</strong><span class="holdreason"> Not on Hold<br />
</span></p>
<h5>Details: </h5>
<p> </p>
```

Template:TaskList

The next template is the TaskList. This list will only show TaskItems attached as a Sub-Page of the current page. So in essence this would be the TaskList for a specific meeting, project, etc. This templage also provided the button to create a new Task Item. As you can see from the screen shot below Items are color Coded: Green is a closed item, Yellow due today, Red Overdue, White still in the future.




```
{{ template.tasklist() }}
```

Below is the DekiScript of the TaskList Template.

{{ wiki.create{label: "New Task", path: page.path ..'/Assigned_Tasks', template: "TaskItem", title: 'Task Name'} }}

Click on header to sort the column

```
{{
 var IsLate='false';
 var BackColor = 'white';
 var todaysdate = date.format(date.changetimezone(date.now, user.timezone), "yyyy/MM/dd");
    if(wiki.pageexists(page.path ..'/Assigned_Tasks')) {
        if(#wiki.getpage(page.path ..'/Assigned_Tasks').subpages > 0) {
 <table width="100%" class="sortable"><tbody>
    <th>
        <b>'User Name'</b>
    </th>
    <th>
        <b>'Task Title'</b>
    </th>
    <th>
        <b>'Summary'</b>
    </th>
    <th>
        <b>'Start Date'</b>
    </th>
```

```
    <th>
        <b>'End Date'</b>
    </th>
    <th>
        <b>'Priority'</b>
    </th>
    <th>
        <b>'Status'</b>
    </th>

  foreach(var p in wiki.getpage(page.path ..'/Assigned_Tasks').subpages) {
            var utext = xml.text(wiki.page(p.path),"//*[@class='username']");
            var stext = xml.text(wiki.page(p.path),"//*[@class='summary']");
            var sdtext = xml.text(wiki.page(p.path),"//*[@class='startdate']");
            var edtext = xml.text(wiki.page(p.path),"//*[@class='enddate']");
            var prtext = xml.text(wiki.page(p.path),"//*[@class='priority']");
            var sttext = xml.text(wiki.page(p.path),"//*[@class='status']");
            if(date.IsAfter(todaysdate,edtext)) {
                let IsLate = 'true';
                let BackColor = 'red';
            } else {
                if(date.IsSameDay(todaysdate,edtext)) {
                    let IsLate = 'false';
                    let BackColor = 'yellow';
                } else {
                    let IsLate = 'false';
                    let BackColor = 'white';
                }
            }
if(string.toupper(string.trim(sttext)) == 'C') {
    <tr BGCOLOR="Mediumspringgreen">
        <td>
        <strike>
            utext;
        </strike>
        </td>
        <td>
        <strike>
            web.link(p.uri, p.title);
        </strike>
        </td>
        <td>
        <strike>
```

```
            stext;
        </strike>
        </td>
        <td>
        <strike>
            sdtext;
        </strike>
        </td>
        <td>
        <strike>
            edtext;
        </strike>
        </td>
        <td>
        <strike>
            prtext;
        </strike>
        </td>
        <td>
        <strike>
            sttext;
        </strike>
        </td>
    </tr>
} else {
    <tr BGCOLOR=(BackColor)>
        <td>
            utext;
        </td>
        <td>
            wiki.edit{label: p.title, path: p.path, button: 'false'};
        </td>
        <td>
            stext;
        </td>
        <td>
            sdtext;
        </td>
        <td>
            edtext;
        </td>
        <td>
            prtext;
```

```
        </td>
        <td>
            sttext;
        </td>
    </tr>
  }
}
</tbody></table>
} else { 'There are currently no tasks assigned' }
} else { 'There are currently no tasks assigned' }
}}
```

Template:TaskListMaster

The final template needed is the TaskMListaster. This is the template that I used to create our management section as well as the one the users have for thier personal task list under thier user page. The only real downfall of this template is that the results are relient on the Lucine engine and therefor results must wait to be indexed prior to showing up on the list.

The TaskListMaster is the only template that has parameters to be passed to it, I have listed a brief summary of this template and parameters below:

Parameters:

- showcomplete (Default = True)
    - Excludes any task item marked as complete.
    - EXAMPLE: showcomplete: 'false'
        - The above example will include all task items marked as complete
- onlymine (Default = True)
    - Will only show the items that are assigned to you
    - EXAMPLE: onlymine: 'false'
        - The above example will show task items from everybody found
- path (Default = Entire Wiki)
    - Will limit the search to a specific path (subpath included)
    - EXAMPLE: path: 'User:rberinger'
        - The above example will search for task items listed in rberingers user page.
        - Omit the path parameter completely to search entire site.
- exclude[] (Default = [""] or none)
    - Excludes certain paths from being included
    - EXAMPLE: exclude: ["Resources/How_Do_I/Insert_a_Task_List/Assigned_Tasks","User:rberinger/SandBox/Task_List/Assigned_Tasks"];
        - The above example will not include tasks from the listed paths

The example below will list every task item found on the entire site wheter complete or not that are assigned to the logged in user.

```
{{ template.tasklistmaster{showcomplete: 'true', onlymine: 'true'}; }}
```

Once again below is the DekiScript of the TemplateListMaster

```
{{
/* User Variables */
var srcpath = ($0 ?? args.path ..'/*');
var showcomplete = ($1 ?? args.showcomplete ?? 'true');
var onlymine = ($2 ?? args.onlymine ?? 'true');
var excludepages = ($3 ?? args.exclude ??  [""]);

/* System Variables DO NOT EDIT below this line! */

if(srcpath == '/*') {let srcpath = 'type: wiki AND title: "Assigned
Tasks"'} else { let srcpath = 'type: wiki AND title: "Assigned Tasks"
AND path: ' ..string.replace(srcpath, 'User:','User\\:')}

var IsLate='false';
var BackColor = 'white';
var todaysdate = date.format(date.changetimezone(date.now, user.timezone), "yyyy/MM/
dd");
var listpage = 'true';
var globalexcludepages = ["Resources/How_Do_I/Insert_a_Task_List/Assigned_Tasks",
                                "User:rberinger/SandBox/Task_List/Assigned_Tasks"];

<table width="100%" class="sortable"><tbody>
    <th>
        <b>'Parent Page'</b>
    </th>
    <th>
        <b>'User Name'</b>
    </th>
    <th>
        <b>'Task Title'</b>
    </th>
    <th>
        <b>'Summary'</b>
    </th>
    <th>
        <b>'Start Date'</b>
    </th>
    <th>
        <b>'End Date'</b>
    </th>
    <th>
        <b>'Priority'</b>
```

```
    </th>
    <th>
        <b>'Status'</b>
    </th>

    foreach(var t in wiki.getsearch('Assigned_Tasks', 100, 'title', srcpath )) {
         let listpage = 'true';

         foreach(var e in excludepages) {
             if(e == t.path) {
                 let listpage = 'false';
             }
         }
         foreach(var e in globalexcludepages) {
             if(e == t.path) {
                 let listpage = 'false';
             }
         }

if(listpage == 'true') {
        if(#t.subpages > 0) {
  foreach(var p in t.subpages) {
            var utext = xml.text(wiki.page(p.path),"//*[@class='username']");
            var stext = xml.text(wiki.page(p.path),"//*[@class='summary']");
            var sdtext = xml.text(wiki.page(p.path),"//*[@class='startdate']");
            var edtext = xml.text(wiki.page(p.path),"//*[@class='enddate']");
            var prtext = xml.text(wiki.page(p.path),"//*[@class='priority']");
            var sttext = xml.text(wiki.page(p.path),"//*[@class='status']");

            if(date.IsAfter(todaysdate,edtext)) {
               let IsLate = 'true';
                let BackColor = 'red';
            } else {
                if(date.IsSameDay(todaysdate,edtext)) {
                     let IsLate = 'false';
                     let BackColor = 'yellow';
                } else {
                     let IsLate = 'false';
                     let BackColor = 'white';
                }
            }

let listitem = 'true';
if(onlymine == 'true') {
    if(!string.contains(string.tolower(utext),user.name)) {
        let listitem = 'false'
    }
}

if(listitem == 'true') {
if(string.toupper(string.trim(sttext)) == 'C') {
  if(showcomplete == 'true') {
   <tr BGCOLOR="Mediumspringgreen">
        <td>
```

```
        <strike>
            web.link(p.parent.parent.uri,p.parent.parent.title);
        </strike>
        </td>
        <td>
        <strike>
            utext;
        </strike>
        </td>
        <td>
        <strike>
            web.link(p.uri, p.title);
        </strike>
        </td>
        <td>
        <strike>
            stext;
        </strike>
        </td>
        <td>
        <strike>
            sdtext;
        </strike>
        </td>
        <td>
        <strike>
            edtext;
        </strike>
        </td>
        <td>
        <strike>
            prtext;
        </strike>
        </td>
        <td>
        <strike>
            sttext;
        </strike>
        </td>
    </tr>
}
} else {
    <tr BGCOLOR=(BackColor)>
        <td>
            web.link(p.parent.parent.uri,p.parent.parent.title);
        </td>
        <td>
            utext;
        </td>
        <td>
            wiki.edit{label: p.title, path: p.path, button: 'false'};
        </td>
        <td>
            stext;
```

```
        </td>
        <td>
            sdtext;
        </td>
        <td>
            edtext;
        </td>
        <td>
            prtext;
        </td>
        <td>
            sttext;
        </td>
    </tr>
  }
}
}
        }
    }
}
</tbody></table>
}}
```

Special thanks to the original author of the DekiCon Notebook example (which I modeled this after) and also everyone that got the sortable tables working in Deki, as well as the wonderful people of the forum that helped me thru a few ignorance problems.