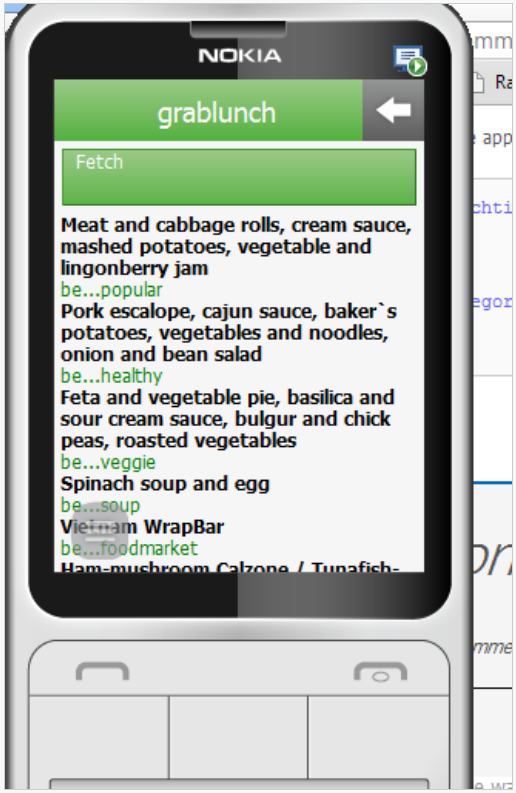
# Lunch list RESTful service (Nokia Asha Web App tutorial)

This tutorial shows how to call a RESTful API from a Nokia Asha web app

## Introduction

The example creates a front-end for the "lunch list" RESTful service using Nokia Asha (Series 40) web apps.




Grab Lunch web app

## Create new project

1. **File | New | Series 40 web app**, Select "multi view project".
2. Give name: "grablunch", fill author name and other metadata.
3. Do "local preview" (**Ctrl + R**)
4. Select various subviews, press back button to get back to front page.
5. Right click on Item-3, select **inspect** to browse DOM at that node.
6. We will look at mwl commands (generated by the multi view project) in detail later on.
7. We will delete `subview-3` (there are only two lunch places we care about). Delete the last of `.ui-list-item` nodes in `.ui-list`, and `#content3`.
8. **Ctrl-R**, ensure the example still runs.
9. Replace some "lorem ipsum" text to `#content1` and `#content2`.
10. Give id to first `ui-list-item` ("listitem1")

## MWL analysis

In the generated template, mwl code is embedded as `onclick` attribute in the div, like this:

```
<div id="listitem1" class="ui-list-item"  onclick="mwl.switchClass('#subContent', 'ui-hide', 'ui-showmv'); mwl.setGroupTarget('#subContent', '#content1', 'ui-showmv', 'ui-hide'); mwl.switchClass('#sliderArea', 'slideLeft', 'slideRight'); mwl.switchClass('#homeBtn', 'ui-hide', 'ui-showmv'); return false;">Item-1</div>
```


(C) Copyright Nokia 2014. All rights reserved.

Or line-by-line:

```
mwl.switchClass('#subContent', 'ui-hide', 'ui-showmv');
mwl.setGroupTarget('#subContent', '#content1', 'ui-showmv', 'ui-hide');
mwl.switchClass('#sliderArea', 'slideLeft', 'slideRight');
mwl.switchClass('#homeBtn', 'ui-hide', 'ui-showmv');
```

The (simplified) layout of DOM is like this:

```
#sliderarea (width 700 px)

// left pane
#mainContent (width 240px)
.ui-list
.ui-list-item
.ui-list-item

// right pane
#subContent (width 240px)
.ui-hide#content1
.ui-hide#content2
```

What MWL does, line by line:

```
mwl.switchClass('#subContent', 'ui-hide', 'ui-showmv');
```

Remove `ui-hide` from `#subcontent`, add `ui-showmv`. This makes `subcontent` visible.

```
mwl.setGroupTarget('#subContent', '#content1', 'ui-showmv', 'ui-hide');
```

add `ui-showmv` to `#content1`; for other children of `#subContent`, remove `ui-showmv` and add `ui-hide`

```
mwl.switchClass('#sliderArea', 'slideLeft', 'slideRight');
```

Remove slideLeft, add slideRight from the whole content.

This causes the right pane to scroll into viewport as per this css:

```
#sliderArea{
 width: 480px;
 -webkit-transition: margin-left 0.5s linear;
}

.slideRight{
 margin-left: -240px;
}

.slideLeft{
 margin-left: 0px;
}
```


(C) Copyright Nokia 2014. All rights reserved.

# Fetching the data

Now, actually fetch some lunch lists!

Add button from toolbar (under .ui-content), with `onclick` handler `getList()`:

```
<div class="ui-button" onclick="getList();">Fetch</div>
```

Add **sodexo.js** as a new file under *s40-theme/js*

Create `getList()` there, add reference to the script in index.html.

We execute the request with jquery's $.ajax():

```
function getList() {
 d = new Date();
 var dstring = "" + (d.getFullYear()) + "/" + (d.getMonth() + 1) + "/" + (d.getDate());
 var urlForPlace = function(placeid) {
     return "http://www.sodexo.fi/ruokalistat/output/daily_json/" + placeid + "/" +
dstring + "/fi?mobileRedirect=false";
 };
 var url = urlForPlace(731)
 console.log("Will fetch from " + url)
 $.ajax( {
  url: url,
  datatype: "json",
  async: false,
  crossDomain: false,
  success: function(data) {
   console.log("Got data ");
   console.log(data)
   respToHtml(data)
  }
 });

}
```

Try to run it, note that jquery is missing, add it.

Run again, browse the response object in debugger.

Responses from the service are like this:

```
{
 courses: Array (7)
 0: Object
 category: "be...popular"
 price: "5,90 / 8,55"
 properties: "VL"
 title_en: "Chrispy saithe filee, mushroom sauce, cream potatoes and vegetables"
 title_fi: "Crispy seiti filettä, herkkusienikastiketta, kermaperunoita ja kasviksia"
 __proto__: Object

 1: Object
 2: Object
 3: Object
 4: Object
```


(C) Copyright Nokia 2014. All rights reserved.

```
 5: Object
 6: Object
}
```

On request completion, we render it on screen using `respToHtml()`:

```
function respToHtml(resp) {
 var courses = resp.courses
 var frags = []

 for (var i in resp.courses ) {
  var item = resp.courses[i]
  var frag = htmlFragment(item)
  frags.push(frag)
 }

 var content = '<div class="lunchlist">' + frags.join("\n") + '</div>'
 $("#content1").html(content)
 $("#listitem1").text("Hermia 3")

}
```

`htmlFragment()` above creates the html for one lunch item. We annotate it with classes:

```
function htmlFragment(obj) {
 return '<div class="lunchtitle">' + obj.title_en + '</div> <div class="category">' +
  obj.category + "</div>"
}
```

Now the application works properly. To spice it up, add some css rules to **s40-theme.css**.

```
.lunchtitle {
 font-weight: bold;
}

.category {
 color: green;
}
```

The app is now completed - as shown in the Introduction


(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.