Matrix navigation solution for small multi-language web

If you have a small web site that contains information on many languages, you need some special navigation solution for it.

A navigation solution must support “matrix” or “table” metaphor: supported languages are the rows and rubrics of your site are columns of this imaginary matrix.

There are following requirements for these sites:

1. The start page determinates the default language of user browser and redirect to main page on this language. When default language not supported, the site shows page with default site language.
2. Pages of site are ready for mobile devices (support responsive design).
3. A user can move to main page for each rubric anytime.
4. A user can change language and move to main page of selected language.
5. When you want in the future add or remove rubrics or languages, you can made it easy, without changing of existing pages.

What it means in details you can see on my home page www.sirotin.eu

My solution contents only 60 lines of JavaScript code. It use famous JavaScript framework Bootstrap (see tutorial in http://www.w3schools.com/bootstrap/).

The problem of determining the browse default language has many solutions in a different GitHub projects. Here my mix of these solutions for page index.shtml in the main folder of the project:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Redirection</title>
</head>
<body>
<div class="container">
    <p>You will be redirected to your language variant. If you see this
page 3 seconds please click <a
href="rub_main/lang_en/index.shtml">here</a>.</p>
</div>
<script>
    const DEFAULT_LANG_FOLDER = "lang_en";
    const SUPPORTED_LANGS =["en", "de", "ru"];
    var folder = "lang_en";
    var lang = window.navigator.userLanguage || window.navigator.language;
    if(lang){
        var pos_undl = lang.indexOf("-");
        if(pos_undl != -1){
            lang = lang.substring(0, pos_undl);
        }
     };
     if(SUPPORTED_LANGS.indexOf(lang) >=0){
        folder = "lang_" + lang;
     }
    window.location = "rub_main/" + folder + "/index.shtml";
</script>
</body>
```

The idea is simple. First, we try to determine the country code that is specified in the browser and to write it into a variable `lang`.

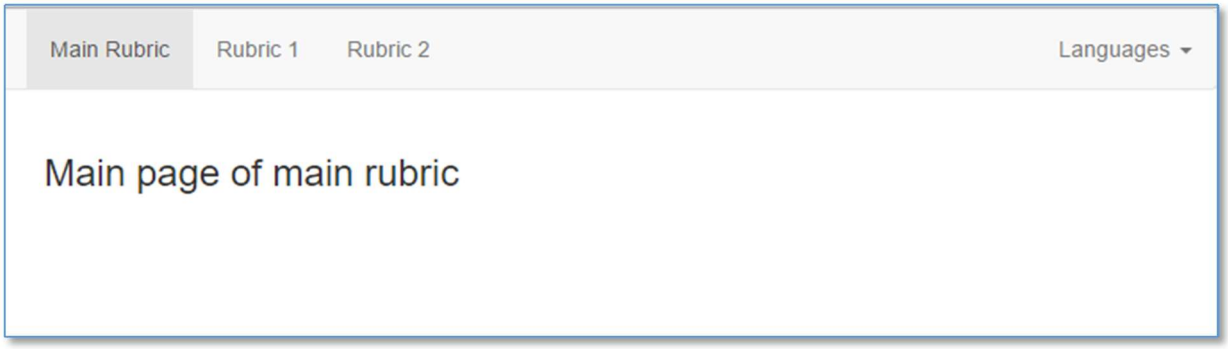
If this operation was successful, we compare the code with the language codes supported by our site. By positive comparing, we use the country code for the creation of a folder name (variable `folder`). If the browser language does not supported by the site, we move the user to the English main page.

As mentioned above, the site have matrix-like structure. Columns in this matrix are site rubrics. Its rows present languages. Cells of the matrix table are main pages of rubric in the appropriate language. All “cell-pages” have the same name: index.shtml

Now we map this concept into a structure of site folders. Since the file folder structure can only be hierarchical, we map our matrix into two-level tree:

To load navigation menu I chose a mix of JavaScript and SSI (Server Side Include). The essence of the SSI - technology that we use here is: the server depends on settings in `htacess` file and URL of called page replaces the placeholder of page with static content from other file.

The figure below shows some page from my GitHub project:



Here the HTML code of this page.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Path and parameter using example</title>
<!--#include virtual="../../navbar/navbar1.htm" -->
</head>
<body>
<!--#include virtual="../../navbar/navbar2.htm" -->
<div class="container">
    <h3>Main page of main rubric</h3>
</div>
</body>
```

In this small script we see two placeholders with names of included files, which are the same for all pages of the site:

```
<!--#include virtual="../../navbar/navbar1.htm" -->
```
and
```
<!--#include virtual="../../navbar/navbar2.htm" -->
```

The first included file is very small and contains links to JavaScript libraries and styles.

The second file is larger. Its code is:

```
<nav class="navbar navbar-default" id="nav_bar">
    <div class="container-fluid">
        <ul class="nav navbar-nav" >
            <li id="rub_main"><a
href="../../rub_main/lang_en/index.shtml">Main Rubric</a></li>
        </ul>
        <ul class="nav navbar-nav">
            <li  id="rub_1" ><a
href=href="../../rub_1/lang_en/index.shtml">Rubric 1</a></li>
        </ul>
        <ul class="nav navbar-nav">
            <li id="rub_2" ><a
href=href="../../rub_2/lang_en/index.shtml">Rubric 2</a></li>
        </ul>
        <ul class="nav navbar-nav navbar-right" >
            <li class="dropdown"><a class="dropdown-toggle" data-
toggle="dropdown" href="#">Languages <span class="caret"></span></a>
                <ul class="dropdown-menu" >
                    <li id="lang_en"><a
href="../../rub_main/lang_en/index.shtml">English</a></li>
                    <li id="lang_de"><a
href="../../rub_main/lang_de/index.shtml">Deutsch</a></li>
```

```
                    <li id="lang_ru"><a
href="../../rub_main/lang_ru/index.shtml">Русский</a></li>
                </ul>
            </li>
        </ul>
    </div>
</nav>

<script>
    const RUBRIC_NAMES_TABLE = '{' +
            '"en" : {"rub_main" : "Main Rubric",     "rub_1" : "Rubric 1",
"rub_2" : "Rubric 2"},' +
            '"de" : {"rub_main" : "Hauptrubrik",     "rub_1" : "Rubrik 1",
"rub_2" : "Rubrik 2"},' +
            '"ru" : {"rub_main" : "Главная рубрика", "rub_1" : "Рубрика 1",
"rub_2" : "Рубрика 2"}' +
             '}';

    const RUBRICS =["rub_main", "rub_1", "rub_2"];

    $(document).ready(process_href());
    function process_href(){
        var href =  window.location.href;
        var posLang = href.indexOf("/lang_");
        if(posLang > 0){
            var lang = href.substring(posLang + 6, posLang + 8);
            var names = JSON.parse(RUBRIC_NAMES_TABLE)[lang];
            var posRub = href.indexOf("/rub_");
            if(posRub > 0){
                 var activeRubric = href.substring(posRub + 1, posLang);
                 resetRubrics(lang, names, activeRubric);
            }
        }
    }
    function resetRubrics(lang, names, activeRubric){
          for(i = 0; i < RUBRICS.length; i++){
            var rub  =RUBRICS[i];
            resetRubric(rub, names, lang, activeRubric);
          }
     }

     function resetRubric(rub, names, lang, activeRubric){
        var el = $( "#" + rub);
        var className = "";
        if(activeRubric == rub){
            className = "active";
        }
        el.toggleClass(className);
        var inHTML = "<a href='../../" + rub + "/lang_" + lang +
"/index.shtml'>" + names[rub] + "</a>"
        el.html(inHTML);
    }
</script>
```

HTML script inside the element <nav ... </ nav> specifies the rubrics of menu. By click on menu item, the appropriated page will be load. After loading starts JavaScript inside of<script> ... </ script>.

The main work process the function process_href (). It parses the URL of the page and adjusts the menu items for the new language of the user.

Well, that is all.

Please use my solution if you want. The page of GitHub project: https://github.com/vsirotin/Matrix-Navigator