

HLVC Website

DEVELOPMENT AND DESIGN GUIDE

Konstantin Shapoval

Table of Contents

OVERVIEW of the website's STRUCTURE and DESIGN	3
STRUCTURAL COMPONENTS of every page	3
DESIGN FEATURES of every page	6
Dynamic FUNCTIONALITY of some pages.....	6
GRAPHICAL and AUDIO content of pages.....	7
ADDING, EDITING, DELETING and MOVING pages.....	9
ADDING a new PAGE to the website.....	9
DELETING a PAGE from the website	12
EDITING the CONTENT of a page	13
MOVING a page TO the NEW CATEGORY	13
Working with the MAP COMPONENT	15
ADDING a new PARTICIPANT to the MAP COMPONENT	15
Keeping RESEARCH ASSISTANTS pages UP TO DATE	18
ADDING a NEW research assistant (RA).....	18
MOVING a research assistant TO the FORMER RA's page.....	21
MIGRATING the WEBSITE to a different location.....	22
LINKING to the website FROM OTHER WEBSITES	25

OVERVIEW OF THE WEBSITE'S STRUCTURE AND DESIGN

STRUCTURAL COMPONENTS OF EVERY PAGE

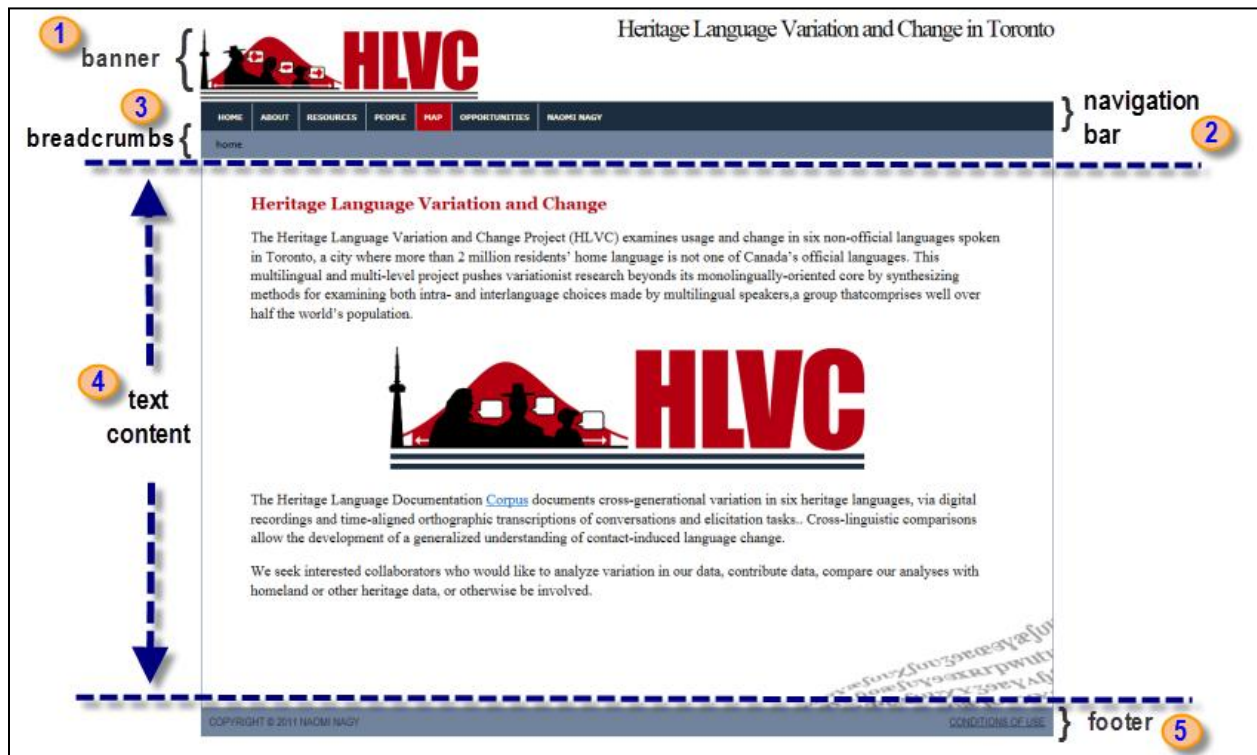


FIGURE 1.0

The website has five visible components: banner, navigation bar, breadcrumbs, text, footer (Figure 1.0). While the breadcrumbs and the text components are different for each individual page, the banner, navigation bar and the footer are the same for all pages.

You can edit banner, navigation bar and footer by accessing the following files:

<code>/includes/parts/banner_include.html</code>	(banner)
<code>/includes/parts/footer_include.html</code>	(footer)
<code>/includes/parts/nav_include.html</code>	(navigation bar)

How to access text component and breadcrumbs will be explained later on in this chapter.

In addition to the five visible components, the website also has one invisible component: header. The header contains links to the cascading style sheets, which are responsible for the design-related definitions such as font size and colour.

The header is the same for all pages and can be accessed at the following location:

`/includes/parts/head_include.html` (header)

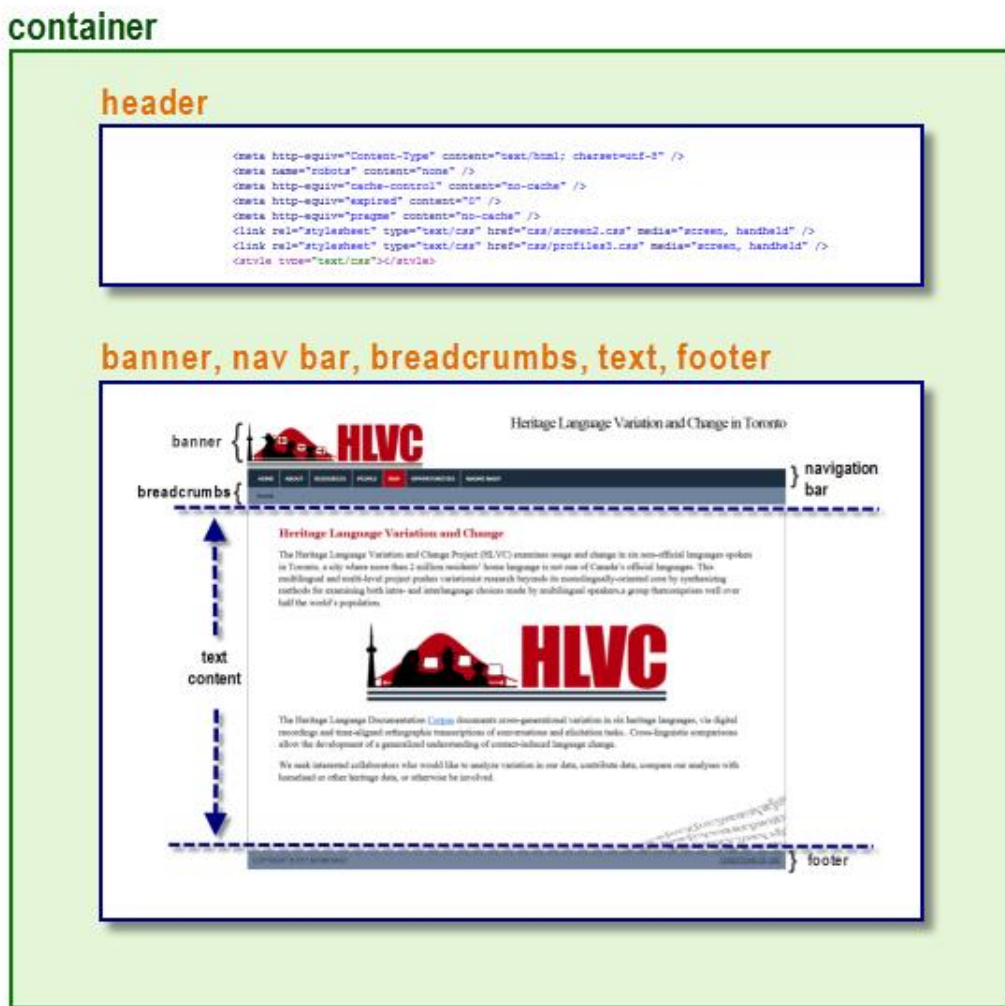


FIGURE 1.1

The visible and invisible components get assembled into a web-page by the container (**Figure 1.1**). While visible and invisible components are written in HTML, the container is written in PHP.

```
1 <?php include 'settings.php'; ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w
3 <html xmlns="http://www.w3.org/1999/xhtml">
4
5 <head>
6 <title>Resources</title>
7 <!-- CSS, JavaScript, Meta Tags -->
8 <?php include $BASE_URL.'/includes/parts/head_include.html'; ?>
9 <!-- CSS, JavaScript, Meta Tags -->
10 </head>
11
12 <body>
13
14 <div id="container">
15
16 <!-- banner -->
17 <?php include $BASE_URL.'/includes/parts/banner_include.html'; ?>
18 <!-- /banner -->
19
20 <!-- navigation bar -->
21 <?php include $BASE_URL.'/includes/parts/nav_include.html'; ?>
22 <!-- /navigation bar -->
23
24 <!-- breadcrumbs -->
25 <div id="breadcrumbs">
26 <div id="crumbs">home &#187; resources &#187; for researchers</div>
27 </div>
28 <!-- /breadcrumbs -->
29
30 <!-- main content -->
31 <div id="main-content">
32 <?php include $BASE_URL.'/includes/content/2_2_linguists.html'; ?>
33 </div>
34 <!-- /main content -->
35
36 <?php include $BASE_URL.'/includes/parts/footer_include.html'; ?>
37
38 </div><!-- /container -->
39
40
41 </body>
42
43 </html>
44
```



FIGURE 1.2

There is one container for each web-page of the web-site. A breadcrumb definition is written in the code of each container (**Figure 1.2**).

All containers are located in the main folder of the website project. All text components are located in **/include/content** folder. The filename of the container and the filename of the text component must correspond. However the filename of the container and the filename of the text component differ from each other in that they have different extensions. Being written in PHP, the container ends with **.php**, while the text component ends with **.html** as it is written in HTML.

For example:

<code>/1_1_objectives.php</code>	(container)
<code>/includes/content/1_1_objectives.html</code>	(text component)

Besides containers, there are two more files in the main folder.

The file, **index.php**, serves to redirect the user to the homepage in case the user does not know the explicit path to the homepage.

The file, **settings.php**, sets the global variables for all web-pages of the web-site. You will need to modify those variables if you are moving the entire web-site from one host provider to another, or if you change the domain name.

DESIGN FEATURES OF EVERY PAGE

Such features as the font colour, table width, border's colour are all defined in cascading style sheet files. These files are located at:

`/css`

The link to the global cascading style sheet is provided within the **header** component. However, in addition to the global style, each individual page can have its own specific set of styles. For example, the Map web-page has the same border width as every page of the web-site, but in addition it has special styles that define the design features of the Google Map application.

Page-specific styles are encoded within the **container** file.

Global style definitions → header component
Page-specific style definitions → container file

DYNAMIC FUNCTIONALITY OF SOME PAGES

For this website, the dynamic functionality is achieved through the use of JavaScript. All JavaScript code resides in

`/js`

JavaScript code is added to the web-page in its container file (**Figure 1.3**).

```

1 <?php include 'settings.php'; ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/1999/xhtml">
3 <html xmlns="http://www.w3.org/1999/xhtml">
4
5 <head>
6   <title>HLVC - Map</title>
7   <!-- CSS, JavaScript, Meta Tags -->
8   <?php include $BASE_URL.'/includes/parts/head_include.html'; ?>
9   <!-- /CSS, JavaScript, Meta Tags -->
10
11   <!-- Special Map Components -->
12   <link type="text/css" href="skin/map_oplayer2.css" rel="stylesheet" />
13   <link type="text/css" href="css/map_portrait.css" rel="stylesheet" />
14   <script type="text/javascript" src="http://ajax.googleapis.com/ajax/lib:
15   <script type="text/javascript" src="http://maps.google.com/maps/api/js?:
16   <script type="text/javascript" src="js/jquery.jplayer.min.js"></script>
17   <script type="text/javascript" src="js/infobox.js"></script>
18   <script type="text/javascript" src="js/table_of_samples.js"></script>
19   <script type="text/javascript" src="js/map_component.js"></script>
20   <!-- /Special Map Components -->
21
22   <script type="text/javascript">
23     /* You can add page-specific JavaScript here. */
24   </script>
25 </head>

```

JavaScript is added to the page in the container file.

FIGURE 1.3

GRAPHICAL AND AUDIO CONTENT OF PAGES

In terms of multimedia, the website makes use of Flash **.swf** files, graphical images and Google Maps applications.

Flash **.swf** files are found in

/audio

They are used in the Google Map application to allow the user to listen to the interview samples. Each folder within the **audio** folder corresponds to the speaker. In another chapter I will show you how to make use of these folders.

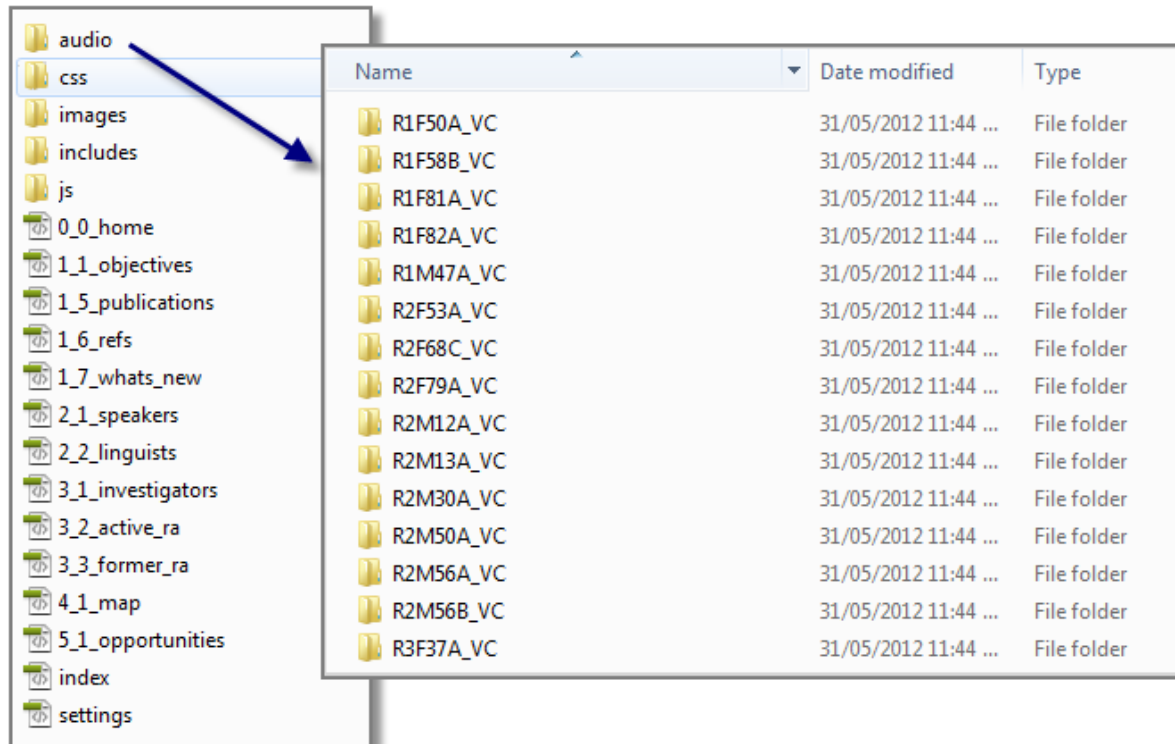


FIGURE 1.4

All graphical content is located at

/images

There isn't much more I can say about the graphical contents.

ADDING, EDITING, DELETING AND MOVING PAGES

ADDING A NEW PAGE TO THE WEBSITE

In this chapter we will go through an example of adding a *What's New* page to the website (Figure 2.0).



FIGURE 2.0

Suppose you want to add your new page to go after the *References* page. This will require the following modifications:

1. Create container page
2. Create text page
3. Edit navigation bar
4. Upload the three modified pages to the hosting server

First, create a new container for the page. The easiest way to create a container is to copy the existing one and modify that copy.

Take the container for the *Reference* page, which is called `1_6_refs.php` and make a copy of it. Notice how the name of each file starts with a number. The number corresponds to the location on the navigation bar. Since *What's New* page comes after the *Reference* page increase the number by one: `1_7_whats_new.php`.

Next, open your Notepad or a web-page editor and make three modifications to the code of the file (**Figure 2.1**).

```
1 <?php include 'settings.php'; ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
3 <html xmlns="http://www.w3.org/1999/xhtml">
4
5 <head>
6   <title>References</title>
7   <!-- CSS, JavaScript, Meta Tags -->
8   <?php include $BASE_URL.'/includes/parts/head_include.html'; ?>
9   <!-- CSS, JavaScript, Meta Tags -->
10 </head>
11
12 <body>
13
14 <div id="container">
15
16   <!-- banner -->
17   <?php include $BASE_URL.'/includes/parts/banner_include.html'; ?>
18   <!-- /banner -->
19
20   <!-- navigation bar -->
21   <?php include $BASE_URL.'/includes/parts/nav_include.html'; ?>
22   <!-- /navigation bar -->
23
24   <!-- breadcrumbs -->
25   <div id="breadcrumbs">
26     <div id="crumbs">home &#187; about &#187; references</div>
27   </div>
28   <!-- /breadcrumbs -->
29
30   <!-- main content -->
31   <div id="main-content">
32     <?php include $BASE_URL.'/includes/content/1_6_refs.html'; ?>
33   </div>
34   <!-- /main content -->
35
36   <?php include $BASE_URL.'/includes/parts/footer_include.html'; ?>
37
38 </div><!-- /container -->
39
40 </body>
41
42 </html>
```

Change the title of the page to *What's New*. This is the text between `<title>` and `</title>` keywords. The new code should look like this:
`<title>What's New</title>`

Change the breadcrumbs. The new code should have *what's new* after the last `»` and before `</div>`

Change the link to the text-content of the new web-page. The filename of the text-content must be the same as that of its container, except at the end it will have **.html** instead of **.php**.

In the given example the container file is `1_7_whats_new.php` Therefore the text-content file should be `1_7_whats_new.html`

FIGURE 2.1

Then, go to the folder `/includes/content` and in that folder create a new HTML file called `1_7_whats_new.html`, which will serve as the text-content file for the new web-page.

Once again, the easiest way to do that is simply to copy an existing HTML file.

Notice that at this point we are **not** creating an HTML web-page file. We are creating an HTML file. The difference is that the file we create will contain very little HTML code – it will only contain HTML code related to representing text and graphics on screen and nothing more (**Figure 2.2**).

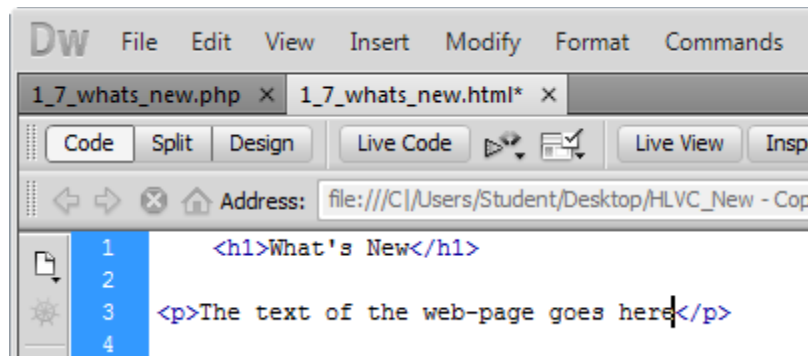


FIGURE 2.2

Once you have created the text-content file you are not required to program its code. Instead, you can switch to the designer view and edit text as you would in Microsoft Word (**Figure 2.3**).

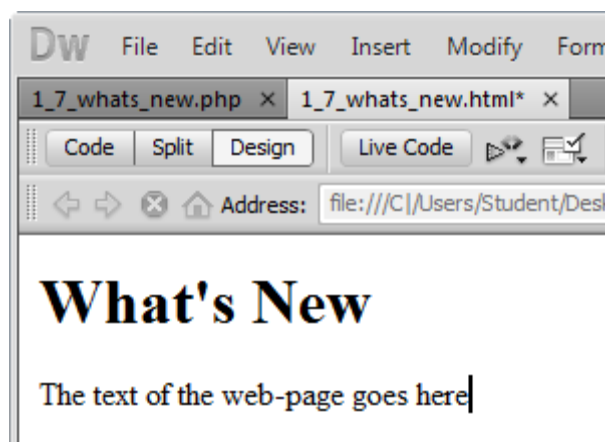


FIGURE 2.3

Next go to the folder `/navigation/parts` and open up the file called `nav_include.html`, which is responsible for producing the navigation bar for the website.

```

<!-- navigation menu -->
<div id="nav-menu">
  <ul id="navbar">
    <li><a href="0_0_home.php" class="navbar_first">HOME</a></li>
    <li><a href="#">ABOUT</a>
      <ul id="navbar-vertical">
        <li><a href="1_1_objectives.php">Objectives</a></li>
        <li><a href="1_1_objectives.php">Description</a></li>
        <li><a href="1_1_objectives.php">Target Languages</a></li>
        <li><a href="1_5_publications.php">Publications</a></li>
        <li><a href="1_6_refs.php">References</a></li>
        <li><a href="1_7_whats_new.php">What's New</a></li>
      </ul>
    </li>

    <li><a href="#">RESOURCES</a>
      <ul id="navbar-vertical">
        <li><a href="2_1_speakers.php">For Speakers</a></li>
        <li><a href="2_2_linguists.php">For Researchers</a></li>
        <li><a href="http://individual.utoronto.ca/ngn/Resources/resources.htm">For Linguists</a></li>
        <!-- li><a href="#">Miscelanous</a></li -->
      </ul>
    </li>

    <li><a href="#">PEOPLE</a>
      <ul id="navbar-vertical">
        <li><a href="3_1_investigators.php">Investigators</a></li>
        <li><a href="3_2_active_ra.php">Active RAs</a></li>
        <li><a href="3_3_former_ra.php">Former RAs</a></li>
      </ul>
    </li>

    <li><a href="4_1_map.php" class="navbar_red">MAP</a></li>
    <li><a href="5_1_opportunities.php">OPPORTUNITIES</a></li>
    <li><a href="http://individual.utoronto.ca/ngn/">NAOMI NAGY</a></li>

  </ul>
</div><!-- /navigation menu -->

```



FIGURE 2.4

Add the link that will point to the container file (**Figure 2.4**). For the *What's New* page, which comes after the *References* page, the line of code would be:

```
<li><a href="1_7_whats_new.php">What's New</a></li>
```

As shown on the image, it should be inserted right after the link to the *Reference* page.

Once this is done, you are ready to upload the three files to the server.

DELETING A PAGE FROM THE WEBSITE

To delete the you need to delete the following:

- Page container (.php)
- Text-content file (.html)

Also you will need to remove the link to the page from the navigation bar. Therefore, you would need to modify `/include/parts/nav_include.html` page.

Lastly, you will need to upload the updated version of the `nav_include.html` file to your server.

It is a good idea to remove the page container (`.php`) and the text-content file (`.html`) from the server as well as from your development environment.

EDITING THE CONTENT OF A PAGE

Go to `/include/content` folder and open up the file which you would like to modify. The text of each page is written in the most basic HTML code. However, to simplify things you can use a visual editor application such as Dreamweaver to work on the content of the file. The application will allow you to edit the file in much the same way you would edit a Word file.

Notice that to modify the content of the page you do not need to work on the container file (`.php`).

MOVING A PAGE TO THE NEW CATEGORY

Suppose you want to move the *What's New* page from About category to People category.

To do that go to `/include/content` folder and open up the `nav_include.html` file.

```

<!-- navigation menu -->
<div id="nav-menu">
  <ul id="navbar">
    <li><a href="0_0_home.php" class="navbar_first">HOME</a></li>
    <li><a href="#">ABOUT</a>
      <ul id="navbar-vertical">
        <li><a href="1_1_objectives.php">Objectives</a></li>
        <li><a href="1_1_objectives.php">Description</a></li>
        <li><a href="1_1_objectives.php">Target Languages</a></li>
        <li><a href="1_5_publications.php">Publications</a></li>
        <li><a href="1_6_refs.php">References</a></li>
      </ul>
    </li>
    <li><a href="#">RESOURCES</a>
      <ul id="navbar-vertical">
        <li><a href="2_1_speakers.php">For Speakers</a></li>
        <li><a href="2_2_linguists.php">For Researchers</a></li>
        <li><a href="http://individual.utoronto.ca/ngn/Resources/resources.htm">For Linguists</a></li>
        <!-- li><a href="#">Miscellaneous</a></li -->
      </ul>
    </li>
    <li><a href="#">PEOPLE</a>
      <ul id="navbar-vertical">
        <li><a href="3_1_investigators.php">Investigators</a></li>
        <li><a href="3_2_active_ra.php">Active RAs</a></li>
        <li><a href="3_3_former_ra.php">Former RAs</a></li>
        <li><a href="1_7_what's_new.php">What's New</a></li>
      </ul>
    </li>
    <li><a href="4_1_map.php" class="navbar_red">MAP</a></li>
    <li><a href="5_1_opportunities.php">OPPORTUNITIES</a></li>
    <li><a href="http://individual.utoronto.ca/ngn/">NAOMI NAGY</a></li>
  </ul>
</div><!-- /navigation menu -->

```

FIGURE 2.5

Move the link to the container file from ABOUT category to the PEOPLE category (**Figure 2.5**).

Upload the updated version of the `nav_include.html` to the server.

WORKING WITH THE MAP COMPONENT

ADDING A NEW PARTICIPANT TO THE MAP COMPONENT

Suppose you want to add a new marker for the participant to the map component. To do that go to `/js` folder and open up the `table_of_samples.js` file. Although the file is written in JavaScript, it actually contains virtually no code.

```
// An array that holds all key locations, with accompanying relevant information.
var arrayAllRows = [
  ['R1F50A', 43.78987, -79.45776, 'ADDRESS AVAILA ', 'R1F50A_VC', 'RU', 1, 50],
  ['R1F58B', 43.63682, -79.55647, 'Not found in IV', 'R1F58B_VC', 'RU', 1, 58],
  ['R1F81A', 43.72192, -79.23597, 'ADDRESS AVAILA ', 'R1F81A_VC', 'RU', 1, 81],
  ['R1F82A', 43.72217, -79.41562, 'ADDRESS AVAILA ', 'R1F82A_VC', 'RU', 1, 82],
  ['R1M47A', 43.66228, -79.39699, 'ADDRESS AVAILA ', 'R1M47A_VC', 'RU', 1, 47],
  ['R2F53A', 43.65620, -79.37932, 'ADDRESS AVAILA ', 'R2F53A_VC', 'RU', 2, 53],
  ['R2F68C', 43.76762, -79.18971, 'ADDRESS AVAILA ', 'R2F68C_VC', 'RU', 2, 68],
  ['R2F79A', 43.67898, -79.34489, 'ADDRESS AVAILA ', 'R2F79A_VC', 'RU', 2, 79],
  ['R2M12A', 43.88493, -79.43039, 'ADDRESS AVAILA ', 'R2M12A_VC', 'RU', 2, 12],
  ['R2M13A', 43.62887, -79.49038, 'ADDRESS AVAILA ', 'R2M13A_VC', 'RU', 2, 13],
  ['R2M30A', 43.77989, -79.41570, 'ADDRESS AVAILA ', 'R2M30A_VC', 'RU', 2, 30],
  ['R2M50A', 43.65794, -79.40000, 'ADDRESS AVAILA ', 'R2M50A_VC', 'RU', 2, 50],
  ['R2M56A', 43.65514, -79.39167, 'ADDRESS AVAILA ', 'R2M56A_VC', 'RU', 2, 56],
  ['R2M56B', 43.58905, -79.64412, 'ADDRESS AVAILA ', 'R2M56B_VC', 'RU', 2, 56],
  ['R3F37A', 43.76824, -79.18628, 'ADDRESS AVAILA ', 'R3F37A_VC', 'RU', 3, 37]
]; // arrayAllGroupKeyLocatons[][]
```

FIGURE 3.0

As the picture above illustrates, the file contains a table of values (**Figure 3.0**). Each row represents a participant (**Figure 3.1**).

```
// An array that holds all key locations, with accompanying relevant information.
var arrayAllRows = [
  ['R1F50A', 43.78987, -79.45776, 'ADDRESS AVAILA ', 'R1F50A_VC', 'RU', 1, 50],
  ['R1F58B', 43.63682, -79.55647, 'Not found in IV', 'R1F58B_VC', 'RU', 1, 58],
  ['R1F81A', 43.72192, -79.23597, 'ADDRESS AVAILA ', 'R1F81A_VC', 'RU', 1, 81],
  ['R1F82A', 43.72217, -79.41562, 'ADDRESS AVAILA ', 'R1F82A_VC', 'RU', 1, 82],
  ['R1M47A', 43.66228, -79.39699, 'ADDRESS AVAILA ', 'R1M47A_VC', 'RU', 1, 47],
  ['R2F53A', 43.65620, -79.37932, 'ADDRESS AVAILA ', 'R2F53A_VC', 'RU', 2, 53],
  ['R2F68C', 43.76762, -79.18971, 'ADDRESS AVAILA ', 'R2F68C_VC', 'RU', 2, 68],
  ['R2F79A', 43.67898, -79.34489, 'ADDRESS AVAILA ', 'R2F79A_VC', 'RU', 2, 79],
  ['R2M12A', 43.88493, -79.43039, 'ADDRESS AVAILA ', 'R2M12A_VC', 'RU', 2, 12],
  ['R2M13A', 43.62887, -79.49038, 'ADDRESS AVAILA ', 'R2M13A_VC', 'RU', 2, 13],
  ['R2M30A', 43.77989, -79.41570, 'ADDRESS AVAILA ', 'R2M30A_VC', 'RU', 2, 30],
  ['R2M50A', 43.65794, -79.40000, 'ADDRESS AVAILA ', 'R2M50A_VC', 'RU', 2, 50],
  ['R2M56A', 43.65514, -79.39167, 'ADDRESS AVAILA ', 'R2M56A_VC', 'RU', 2, 56],
  ['R2M56B', 43.58905, -79.64412, 'ADDRESS AVAILA ', 'R2M56B_VC', 'RU', 2, 56],
  ['R3F37A', 43.76824, -79.18628, 'ADDRESS AVAILA ', 'R3F37A_VC', 'RU', 3, 37]
];
// arrayAllGroupKeyLocatons[][][]
```

You will be working with this portion of the code. When you add a new line, add it from the top.

FIGURE 3.1

In order to add a participant you will need the following seven pieces of information ready:

- Participant's code id
- Latitude (Google.maps)
- Longitude (Google.maps)
- Directory of the corresponding audio sample
- Language
- Generation
- Age

I will describe how to determine the latitude and the longitude for a participant in another chapter. For now, let's just assume that you already have those.

In addition to the seven mandatory pieces of information, you can also include a description, but this is optional. The description is for the developer; it will not be displayed to the end-user.

Each row starts must start with “[” and must end with “]”. If you look at the very last line of the value of tables code you will notice that it ends on “[” – no comma at the end (**Figure 3.1**). This is because it marks the end of the table. Insert new values from the top in order to avoid accidentally leaving the table open. If you leave the table open, it will generate an error.

Every column is delimited by a comma. There are eight columns (**Figure 3.2**).

1	2	3	4	5	6	7	8
['R1F50A',	43.78987,	-79.45776,	'ADDRESS AVAILA '	'R1F50A_VC',	'RU',	1,	50],
['R1F58B',	43.63682,	-79.55647,	'Not found in IV',	'R1F58B_VC',	'RU',	1,	58],
['R1F81A',	43.72192,	-79.23597,	'ADDRESS AVAILA '	'R1F81A_VC',	'RU',	1,	81],
['R1F82A',	43.72217,	-79.41562,	'ADDRESS AVAILA '	'R1F82A_VC',	'RU',	1,	82],
['R1M47A',	43.66228,	-79.39699,	'ADDRESS AVAILA '	'R1M47A_VC',	'RU',	1,	47],
['R2F53A',	43.65620,	-79.37932,	'ADDRESS AVAILA '	'R2F53A_VC',	'RU',	2,	53],
['R2F68C',	43.76762,	-79.18971,	'ADDRESS AVAILA '	'R2F68C_VC',	'RU',	2,	68],
['R2F79A',	43.67898,	-79.34489,	'ADDRESS AVAILA '	'R2F79A_VC',	'RU',	2,	79],
['R2M12A',	43.88493,	-79.43039,	'ADDRESS AVAILA '	'R2M12A_VC',	'RU',	2,	12],
['R2M13A',	43.62887,	-79.49038,	'ADDRESS AVAILA '	'R2M13A_VC',	'RU',	2,	13],
['R2M30A',	43.77989,	-79.41570,	'ADDRESS AVAILA '	'R2M30A_VC',	'RU',	2,	30],
['R2M50A',	43.65794,	-79.40000,	'ADDRESS AVAILA '	'R2M50A_VC',	'RU',	2,	50],
['R2M56A',	43.65514,	-79.39167,	'ADDRESS AVAILA '	'R2M56A_VC',	'RU',	2,	56],
['R2M56B',	43.58905,	-79.64412,	'ADDRESS AVAILA '	'R2M56B_VC',	'RU',	2,	56],
['R3F37A',	43.76824,	-79.18628,	'ADDRESS AVAILA '	'R3F37A_VC',	'RU',	3,	37]

FIGURE 3.2

Column 1: Participant's unique id

Column 2: Latitude

Column 3: Longitude

Column 4: (optional) your comments, descriptions, etc.

Column 5: Audio sample directory (do not put filename extension)

Column 6: Language

Column 7: Generation

Column 8: Age

Language is abbreviated as follows:

CA – Cantonese

FA – Faetar

IT – Italian

KO – Korean

RU – Russian

UK – Ukrainian

After adding the participant's information to the table of values, save and close the file.

Next, make sure to add the directory containing audio sample files to the **/audio** folder.

Next, upload the **table_of_samples.js** file and the audio sample directory to the server.

KEEPING RESEARCH ASSISTANTS PAGES UP TO DATE

ADDING A NEW RESEARCH ASSISTANT (RA)

The project relies on colour-coding to represent languages that are being covered in the research.

Information about each RA is encapsulated in a box with the border color that matches the language the RA is working on (**Figure 4.0**).

The screenshot shows the HLVC website with a navigation bar and a main content area. The main content area is titled "Active Research Assistants" and lists several RAs grouped by language. Each RA is represented by a colored box with their name, photo, and contact information. A green arrow points to one of the boxes.

Language Group	Research Assistant Name	Contact Information
Cantonese	Sita Fung (Team Leader)	Site.fung@utoronto.ca
	Sarah Truong	sarah.truong@utoronto.ca
	Olivia Yu	olivia.yu@utoronto.ca
Foster	Tasha Digenic	tasha.digenic@utoronto.ca
	Rich Grissom	rich.grissom@utoronto.ca
Italian	Vanessa Bertone (Team Leader)	vanessa.bertone@utoronto.ca
	Silvia Iulietta	silvia.iulietta@utoronto.ca
	Courtney Chisno	courtney.chisno@utoronto.ca
Korean	Shella Chung (Team Leader)	shella.chung@utoronto.ca

Each box represents a Research Assistant.

Therefore do add a new RA you need to duplicate one of the boxes and change the name on the box.

FIGURE 4.0

To add a new RA you need to duplicate the box and add change the name, the email address. If the new RA supplied you with their photograph and personal bio text, you can also add those to the newly created box.

This is best done by editing the HTML code of the page. To do that, go to `/include/content` and open `3_2_active_ra.html` in your HTML editor.

Browse through the file until you find a box from the language group that you wish to add the new RA to. For example, if your new RA is working on Faetar language, then you would look for an RA who is already in Faetar group and duplicate that RA's box.

The code will look like this. Notice that every box begin with the person's name.

Copy the duplicate the box by doing Copy and Paste.

Change the name that marks the beginning of the box (**Figure 4.1: Step 1**).

Change the name displayed to the end user (**Figure 4.1: Step 2**).

Change the email address displayed to the end-user (**Figure 4.1: Step 3**). Notice the format.

```
<!-- Rick_Grimm -->
<div class="profile_wrapper" language="faetar">
<table class="profile_table">
  <tr>
    <td class="td_photo"></td>
    <td class="td_bio">
      <span class="profile_name">Rick Grimm</span><span>&nbsp;</span><span class="mark_leader"></span><br />
      <span class="profile_email">rgrimm at yorku dot ca</span><br />
      <span class="show_bio_button">
        <a id="link_Rick_Grimm" href="javascript:toggleBio('bio_Rick_Grimm', 'link_Rick_Grimm');">more information</a>
      </span>
      <div id="bio_Rick_Grimm" style="display: none">
        <span class="profile_bio">No additional information is available.</span>
      </div>
    </td>
  </tr>
</table>
</div>
```

FIGURE 4.1

If you have the photograph of the person, save it to `/images/photo/id` and update the code:

```
<!-- Rick_Grimm -->
<div class="profile_wrapper" language="faetar">
<table class="profile_table">
  <tr>
    <td class="td_photo"></td>
    <td class="td_bio">
      <span class="profile_name">Rick Grimm</span><span>&nbsp;</span><span class="mark_leader"></span><br />
      <span class="profile_email">rgrimm at yorku dot ca</span><br />
      <span class="show_bio_button">
        <a id="link_Rick_Grimm" href="javascript:toggleBio('bio_Rick_Grimm', 'link_Rick_Grimm');">more information</a>
      </span>
      <div id="bio_Rick_Grimm" style="display: none">
        <span class="profile_bio">No additional information is available.</span>
      </div>
    </td>
  </tr>
</table>
</div>
```

FIGURE 4.2

Change the filename of the image (**Figure 4.2: Step 4**).

Change the `class` keyword to equal to “profile_photo” instead of “blank_photo” (**Figure 4.2: Step 5**).

If you have the any other information supplied by the person you can add it to be displayed when the end-user pressed on the “more information” link.

```
<!-- Rick_Grimm -->
<div class="profile_wrapper" language="faetar">
<table class="profile_table">
  <tr>
    <td class="td_photo"></td>
    <td class="td_bio">
      <span class="profile_name">Rick Grimm</span>&nbsp;<span class="mark_leader"></span><br />
      <span class="profile_email">rgrimm at yorku dot ca</span><br />
      <span class="show_bio_button">
        <a id="link_Rick_Grimm" href="javascript:toggleBio('bio_Rick_Grimm', 'link_Rick_Grimm');">more information</a>
      </span>
      <div id="bio_Rick_Grimm" style="display: none">
        <span class="profile_bio">No additional information is available.</span>
      </div>
    </td>
  </tr>
</table>
</div>
```



FIGURE 4.3

Change the name in the four different places highlighted in the illustration above (**Figure 4.3: Steps 6-9**).

```
<!-- Rick_Grimm -->
<div class="profile_wrapper" language="faetar">
<table class="profile_table">
  <tr>
    <td class="td_photo"></td>
    <td class="td_bio">
      <span class="profile_name">Rick Grimm</span>&nbsp;<span class="mark_leader"></span><br />
      <span class="profile_email">rgrimm at yorku dot ca</span><br />
      <span class="show_bio_button">
        <a id="link_Rick_Grimm" href="javascript:toggleBio('bio_Rick_Grimm', 'link_Rick_Grimm');">more information</a>
      </span>
      <div id="bio_Rick_Grimm" style="display: none">
        <span class="profile_bio">No additional information is available.</span>
      </div>
    </td>
  </tr>
</table>
</div>
```



FIGURE 4.4

Next, add the information supplied by the RA (**Figure 4.4: Step 10**).

Finally, you need to declare whether the person is a team leader or not (**Figure 4.5**).

```

<!-- Rick_Grimm -->
<div class="profile_wrapper" language="faetar">
<table class="profile_table">
  <tr>
    <td class="td_photo"></td>
    <td class="td_bio">
      <span class="profile_name">Rick Grimm</span>&nbsp;<span class="mark_leader"></span><br />
      <span class="profile_email">rgrimm at yorku dot ca</span><br />
      <span class="show_bio_button">
        <a id="link_Rick_Grimm" href="javascript:toggleBio('bio_Rick_Grimm', 'link_Rick_Grimm');">more information</a>
      </span>
      <div id="bio_Rick_Grimm" style="display: none">
        <span class="profile_bio">No additional information is available.</span>
      </div>
    </td>
  </tr>
</table>
</div>

```



FIGURE 4.5

If the newly added RA is a team leader the highlighted line should read:

```
<span class="mark_leader">(Team Leader)</span>
```

Otherwise, the line should read:

```
<span class="mark_leader"></span>
```

At this point you are done editing the file. Save and upload it to the server.

MOVING A RESEARCH ASSISTANT TO THE FORMER RA'S PAGE

When a research assistant leaves the project we need to update the `3_2_active_ra.html` page in order to reflect the change. In this case, the person in question stops being an active RA and becomes a former RA. Therefore, we remove the person from the `3_2_active_ra.html` page and list him or her on the `3_3_former_ra.html` page.

Go to `/include/content` and open `3_2_active_ra.html` in your HTML editor. Find the box that contains RA's information. Copy the box and then delete it.

Now, open `3_3_former_ra.html` in your editor and find the appropriate language group. Paste the box with RA's information.

Save `3_3_former_ra.html` and close the file. Save `3_2_active_ra.html` and close the file.

Upload both files to the server.

MIGRATING THE WEBSITE TO A DIFFERENT LOCATION

Suppose you need to move the website from your current server to a new server. In this case you will need to modify the following files:

```
/settings.php  
/index.php  
/js/map_components.js
```

Open up `settings.php` and modify the line of code that specifies the base domain name (**Figure 5.0**).

In order to make the modification you need to know the folder structure of the destination server. Notice that this folder structure may or may not be equivalent to the URL that you see in the web browser. For example, you may type in `www.somedomain.com/hlvc/index.php` to access the website on the new server, however the folder structure may be something like this: `/user/production/v1.2/hlvc/index.php`. Therefore, to perform this task correctly, make sure you know the correct folder structure. Sometime you are able to find this information just by browsing through directories via an FTP client, however often FTP client won't be enough. In such cases you will need to obtain such information from your host provider or from the server administrator.

```
<?php  
$BASE_URL = '/home/konstant/public_html/k/484c5643';  
?>
```



FIGURE 5.0

It is a good idea to leave the original code and add your new code right below. Make sure to comment out the old code, so that the server won't execute it. In PHP, you can comment the code by using `//` (**Figure 5.1**).

```
<?php  
//$BASE_URL = '/home/konstant/public_html/k/484c5643';  
$BASE_URL = '/new_server/p/v1.2/hlvc';  
?>
```

FIGURE 5.1

Save your work and close **settings.php** file.

If it is the case that the URL with which you are accessing the website will change as a result of the migration, then you will need to modify the **index.php** and the **map_components.js** files as well.

Open **index.php** file in the editor and change the URL address to point to a new web address (**Figure 5.2**).

```
<?php

if (!headers_sent()) {
    header( 'Location: http://k.shapoval.ca/484c5643/0_0_home.php' );
    exit;
} else {
    echo "sent";
}

/*
If you use this code, but get a report that the header has already been sent, it may be due to the
BOM (Byte-order mark). Make sure to save it without the BOM and this should take care of it.
*/
?>
```

A screenshot of a code editor showing PHP code. The code is as follows: <?php, followed by a block: if (!headers_sent()) { header('Location: http://k.shapoval.ca/484c5643/0_0_home.php'); exit; } else { echo "sent"; }. Below this is a comment block: /* If you use this code, but get a report that the header has already been sent, it may be due to the BOM (Byte-order mark). Make sure to save it without the BOM and this should take care of it. */ followed by ?>. A green oval highlights the URL 'http://k.shapoval.ca/484c5643/0_0_home.php' in the header function call. A green arrow points from the right towards the oval.

FIGURE 5.2

You only need to modify the highlighted portion of the web address (**Figure 5.3**).

```
if (!headers_sent()) {
    header( 'Location: http://k.shapoval.ca/484c5643/0_0_home.php' );
    exit;
} else {
    echo "sent";
}
```

A screenshot of a code editor showing the same PHP code as in Figure 5.2. A green rectangular box highlights the path portion of the URL: '/484c5643/0_0_home.php'. A green arrow points from the right towards the box.

FIGURE 5.3

Save and close the file.

Once again, notice the difference between the **settings.php** and the **index.php** files. In **settings.php** you are updating the folder structure. In **index.php** you are updating the URL with which you are accessing the website.

Next, go to `/js` folder and open `map_components.js` file. The 8th line of the file contains a link to the folder, which contains audio samples (**Figure 5.4**).

```
1 // Declare and initialize the Global Variables
2 var initLatLng = new google.maps.LatLng(43.74233,-79.43802); // initial Lat/Lng = Toronto
3 var initZoom = 10; // initial zoom level
4 var allMarkersArray = []; // array of markers
5 var map; // main map
6 var image = 'images/marker2.png'; // target location market image
7 var ib;
8 var pathToAudioFiles = "http://k.shapoval.ca/484c5643/audio"; // path to audio files
9 // on the iMAC: iMac2/hlvc/RUSSIAN/Voice Clips/R3F37A_VC
10
11
12 // Set array's columns names
13 var COL_NAME = 0;
14 var COL_LAT = 1;
15 var COL_LONG = 2;
16 var COL_DESC = 3;
```

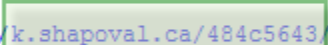


FIGURE 5.4

Update the URL so that it points to the new location (**Figure 5.5**).

```
// Declare and initialize the Global Variables
var initLatLng = new google.maps.LatLng(43.74233,-79.43802); // initial Lat/
var initZoom = 10; // initial zoom
var allMarkersArray = []; // array of mar
var map; // main map
var image = 'images/marker2.png'; // target locat
var ib;

var pathToAudioFiles = "http://k.shapoval.ca/484c5643/audio";
```



Update this portion



FIGURE 5.5

Upload the modified files to your server.

LINKING TO THE WEBSITE FROM OTHER WEBSITES

Always link to the container page (**.php**). Never link to the text content components or any other components.

Below is an example of an HTML link code:

```
<a href="http://www.domainname.com/index.php">HLVC Website</a>
```

Of course, this is only an example. In reality you will need to substitute the "http://www.domainname.com/index.php" for the real URL address.