CCE Kiosk Technical Overview

# Purpose

The purpose of this document is to give a technical overview of the CCE kiosk. This will allow anyone who wants to work on the kiosk to quickly get started or for bugs in the kiosk to be quickly fixed. This guide will also help with some common trouble shooting topics as well as give an overview of how the Kiosk works.

# Architecture

The CCE Kiosk has no real architecture as it’s a simple web app. This project is HTML files running statically in a directory system. These files use Javascript and CSS but there is no need for a back end as they are served strait form the directory. Additionally the photos being displayed from the kiosk are also just in static directories and are not served by any sort of backend.

# Frame Works or Third Party Modules

Again because this is a simple web app served from a file system there are no frame works to work with or Third Part Modules. The Whole application consists of raw HTML, JavaScript and CSS. When manipulations to the DOM are made they are made through raw JavaScript.

# Overview of the Application

The application starts at the main menu (contained in the start.html along with sub menus) from the main menu you can go to the various sub menus and from the sub menus to the various slide shows in each of the sub menus.

# How the Menu System works.

The menu system is contained within a div in the start.html page. The menus are created in the kiosk.js file. At the top of this file is a case statement that determines which menu needs to be display through the query parameter in the URL. The default option is the main menu. In this case statement the various menu items to be rendered on the page are made. This is why only the menu items for the CEM page are rendered when we click on the CEM sub menu. The menu items are an object that contains three strings. The first string is the caption that will be displayed below the menu item. The second string is a path to the photo that will be displayed for each menu item. The third string represents a function that will be run when the menu items is clicked. Later when parsing out the raw HTML for these items the source image will go in an image tag the caption will go below the image in a div and the function will be placed in the mousedown attribute.

The menu items will call one of two functions either change menu which will navigate you to a new menu or changePage which will render a page in the iframe in the html page. In this section we will only talk about the menu system so we will restrict discussion to the changeMenu function. The change Menu function is rather simple in that all it does is change the location of the page by adding a query parameter to the current URL. This works because we are choosing which menu to render using a switch at the top of the file which uses the quarry parameter to choose which menu is rendered.

Finally the final part of the menu system is that each sub menu has a home button. This button navigates the user back to the home page by changing the URL the page is at back to the base URL with no query parameters.

# How the Slide Shows Work

In this section we will discuss how the slide shows are being rendered and displayed.

For each slide show there is a separate HTML file. This is because each slide show has its own preloading JavaScript file (we will get into that later). Also there is an iframe in the start.html file that renders the slide show. All of the slide show files are simple html pages rendered in an iframe. For each slide show we want to include different photos to do this we are calling a different preloading script for each slide show, to load up the various different photos. This is why you will notice in the various different html files for the slide show there is a prealoading\_\_\_\_\_.js script (where the blank is filled in with some descriptive name).

The preloading script takes all of the photos and loads them into an array. This array contains an object that contains two strings, The path to the photo and the caption for the photo. When it comes time to display the photo the path is place in the src attribute in an image tag and the caption is place in a div below the photo. This allows us to navigate through the array with relative ease by moving up and down the array. In the slideShow.js file there a several functions related to the slide show. Most notably the start function (which starts the slide show) The nextImage Function which moves to the forward in the array by n elements and perviousImage function which moves backward through the array by n elements.

# Common Troubleshooting topics

Generally there are only a few things that can go wrong with the CE kiosk. Four things can happen, either a menu item photo is not rendered, a submenu doesn’t work, a menu item does not do what is supposed to or a photo in a slide show does not render.

If a submenu does not work you need to check the code for the menu item that directs you to that submenu. Ensure that the correct function is being called as follows: “changeMenu(‘yourMenu’);”. Ensure that the function call is surrounded by double quotes and that the name of the menu is in single quotes. Also ensure that the menu name is spelled correctly in both the function call and the case statement.

If a menu item photo is not rendered you need to check that the path is spelled correctly. You will also need to ensure that the photo you are using is no larger than 300 px on its longest side. If the photo is too large us MS paint to resize it. Also you will need to ensure that the photo is either a png or a jpg.

If a menu item does not do what is supposed to you will want to make sure that the function associated with that menu item is correct. Make sure that the function call is contained in double quotes while the argument for the function is in single quotes. You will also need to make sure that the argument for the function call is spelled correctly. If it is a menu item for a slide show you will also need to ensure that the path for the html page is correct.

Finally if a photo in a slide show does not render there are several things you might try. First ensure that the path for the photo in the preloading script is correct. Next check the photo size, if the photo is rather large try resizing it. If a photo is to large this sometimes causes it not to render. If this still doesn’t work try using MS paint to turn the photo into a different format. Sometimes changing a jpg to to png or vise versa will do the trick. If you change the type or name of the photo make sure you also make the same changes in the preloading.js file for the page you are changing.

If you have further questions email Thomas Noelcke @ noelcket@oregonstate.edu.