EWTK

Efficient Web: Toolkit

Project Goals and synopsis

**General Idea**

Create set of .net and JavaScript libraries, template html, css and JavaScript files and programming methodologies that facilitate fast load times, fast code, responsiveness across environments and good user experience overall. Central to achieving this goal will be a JavaScript-.NET library for compressing files on the server side and loading them into a page on the client side using RESTful queries.

**Features**

1. FCC
   1. (First Class Compressor) Compression algorithm written in C++ and implemented in .NET. It will compress html, css and javascript while acknowledging first class objects. Resulting code will not need to be decompressed on the client side, this will improve execution time. It will also improve parse time in most browsers.
   2. Key Features
      1. Multiple files can be treated as part of the same context.
         1. This will allow files that are dependent on each other to be compressed without references between them being broken.
      2. Will create the smallest possible code that is still capable of executing.
   3. Goals
      1. Improve load time
2. RDC
   1. (Raw Data Compressor) Compression algorithm written in C++, implemented in .NET and a decompression algorithm written and implemented in JavaScript. This will perform lossless compression of files before they are sent to the client by eliminating redundancy. The client will then decompress the file and load it into the DOM.
   2. Key Features
      1. High speed compression before sending
      2. Support for combining multiple files into a single compressed file for higher send speed.
   3. Goals
      1. Improve load time
3. DJSL
   1. (Dynamic Javascript Load) A library that provides a function for loading a page into the DOM. It will take a set of parameters for finding the style, content and script to load. It will support but not require the RDC library.
   2. Key Features
      1. Supports but does not require RDC
      2. Allows multiple files or one file
      3. Allows a target element to be specified for loading the content into
      4. Template JavaScript file containing load functions will be included.
   3. Goals
      1. Better user experience due to lack of redundant library loading
      2. Better user experience due to lack of interruption while loading new content
      3. Faster load time, due to support for RDC
4. PLE
   1. (Page Load Eliminator) A library that wraps DJSL and allows the user to specify pages that will be loaded in response to different additions to the url. The user can specify a mapping in JSON. From then, they can load other pages by calling a special function. When the user types a new sub-url into the search bar it will cause PLE to call some DJSL code.
   2. Key Features
      1. Supports but does not require DJSL
      2. Handles changes to the url in the search bar
      3. Allows for page name aliases to be mapped to files that can be loaded
   3. Goals
      1. Make web development easier
      2. Improve user experience
      3. Improve load time
5. Notify.js
   1. Extremely simple library for giving the user notifications.
   2. Key Features
      1. Exposes a function for generating a non-obtrusive notification at the bottom right of the screen
      2. Exposes a function for generating an obtrusive notification that covers the screen and requires the user to press a button to get rid of it.
6. PF-DD
   1. (Persistent Forms - Dynamic Data) Wraps DJSL and allows a parameter for the server to request data from. It will receive this data as JSON and load the data into forms in a specified DIV.
   2. Key Features
      1. Reads special attributes of DOM elements in a specified div in order to load data from a server
      2. Provides a persistent object that holds the data and changes when the user makes changes
      3. Allows the form to be hidden and shown without any reloading.
      4. Allows the data within it to be refreshed with minimal load time due to nothing but the data being changed.
   3. Goals
      1. Make development much easier
      2. Improve user experience by dramatically reducing load time
7. Base.css
   1. A template CSS file that contains media queries that select Large screens, Desktop/laptop screens, tablet screens and phone screens.
   2. Goals
      1. Improve responsiveness by making responsive design easier
8. Base.html
   1. A template html file that is highly compliant with web standards and contains the proper meta-data for rendering on phone screens.
   2. Goals
      1. Improve responsiveness and user experience