Weather Stack Net Core

Sample .NET Core Application for consuming Weather Stack API

Table of Contents

1 Symbol Reference

1.1 Variables

The following table lists variables in this documentation.

Variables

Name	Description
Lobibox (see page 1)	This is variable Lobibox.
MyBookShelfWeb (see page 1)	This is variable MyBookShelfWeb.

1.1.1 Lobibox Variable

JavaScript

```
var Lobibox = Lobibox || {};
```

File

notifications.js (see page 5)

Description

This is variable Lobibox.

1.1.2 MyBookShelfWeb Variable

JavaScript

```
var MyBookShelfWeb = MyBookShelfWeb || {};
```

File

notificationManager.js (see page 4)

Description

This is variable MyBookShelfWeb.

1.2 Files

The following table lists files in this documentation.

Files

Name	Description
AutoComplete.cs (see page 2)	This is file AutoComplete.cs.
AutoCompleteController.cs (see page 2)	This is file AutoCompleteController.cs.
AutoCompleteViewModel.cs (see page 2)	This is file AutoCompleteViewModel.cs.
Current.cs (see page 3)	This is file Current.cs.
CurrentWeather.cs (see page 3)	This is file CurrentWeather.cs.
CurrentWeatherController.cs (see page 3)	This is file CurrentWeatherController.cs.
CurrentWeatherViewModel.cs (see page 3)	This is file CurrentWeatherViewModel.cs.
Error.cs (see page 3)	This is file Error.cs.
ErrorResult.cs (see page 3)	This is file ErrorResult.cs.
ErrorViewModel.cs (see page 3)	This is file ErrorViewModel.cs.
GeneralRequest.cs (see page 3)	This is file GeneralRequest.cs.
HomeController.cs (see page 4)	This is file HomeController.cs.
ItemList.cs (see page 4)	This is file ItemList.cs.
Location.cs (see page 4)	This is file Location.cs.
LocationRequest.cs (see page 4)	This is file LocationRequest.cs.
notificationManager.js (see page 4)	Look documentation for more detail http://lobianijs.com/site/lobibox#lobibox-notification-usage
notifications.js (see page 5)	
site.js (see page 12)	Please see documentation at https://docs.microsoft.com/aspnet/core/client-side/bundling-and-minification for details on configuring this project to bundle and minify static web assets.
WeatherStackNetCore.csproj (see page 12)	This is file WeatherStackNetCore.csproj.
WeatherStackNetCore.sln (see page 13)	This is file WeatherStackNetCore.sln.

1.2.1 AutoComplete.cs

This is file AutoComplete.cs.

1.2.2 AutoCompleteController.cs

This is file AutoCompleteController.cs.

1.2.3 AutoCompleteViewModel.cs

This is file AutoCompleteViewModel.cs.

1.2.4 Current.cs

This is file Current.cs.

1.2.5 CurrentWeather.cs

This is file CurrentWeather.cs.

1.2.6 CurrentWeatherController.cs

This is file CurrentWeatherController.cs.

1.2.7 CurrentWeatherViewModel.cs

This is file CurrentWeatherViewModel.cs.

1.2.8 Error.cs

This is file Error.cs.

1.2.9 ErrorResult.cs

This is file ErrorResult.cs.

1.2.10 ErrorViewModel.cs

This is file ErrorViewModel.cs.

1.2.11 GeneralRequest.cs

This is file GeneralRequest.cs.

1.2.12 HomeController.cs

This is file HomeController.cs.

1.2.13 ItemList.cs

This is file ItemList.cs.

1.2.14 Location.cs

This is file Location.cs.

1.2.15 LocationRequest.cs

This is file LocationRequest.cs.

1.2.16 notificationManager.js

Look documentation for more detail http://lobianijs.com/site/lobibox#lobibox-notification-usage

Body Source

```
?//Look documentation for more detail
http://lobianijs.com/site/lobibox#lobibox-notification-usage
var MyBookShelfWeb = MyBookShelfWeb || {};


MyBookShelfWeb.notification = {

    ShowErrorMessage: function (title,message) {
        Lobibox.notify('error', {
            title:title,
            msg: message
        });
    },

    ShowOKMessage: function (title, message) {
        Lobibox.notify('success', {
            title: title,
            msg: message
        });
    },

    ShowInfoMessage: function (title, message) {
        Lobibox.notify('info', {
            title: title,
            msg: message
        });
    },

    ShowWarningMessage: function (title, message) {
        Lobibox.notify('warning', {
            title: title,
            msg: message
        });
    }
}
```

Variables

Name	Description
MyBookShelfWeb (see page 1)	This is variable MyBookShelfWeb.

1.2.17 notifications.js

@arboshiki *

- Generates random string of n length.
- String contains only letters and numbers

*

- @param {int} n
- @returns {String}

Body Source

```
//Author       : @arboshiki
/**
 * Generates random string of n length.
 * String contains only letters and numbers
 *
 * @param {int} n
 * @returns {String}
 */
Math.randomString = function (n) {
    var text = "";
    var possible = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";

    for (var i = 0; i < n; i++)
        text += possible.charAt(Math.floor(Math.random() * possible.length));

    return text;
};
var Lobibox = Lobibox || {};
(function () {

    var LobiboxNotify = function (type, options) {
//------------------------------------------------------------------------------
//----------------PROTOTYPE VARIABLES-------------------------------------------
//------------------------------------------------------------------------------
        this.$type = null;
        this.$options = null;
        this.$el = null;
//------------------------------------------------------------------------------
//-----------------PRIVATE VARIABLES--------------------------------------------
//------------------------------------------------------------------------------
        var me = this;
//------------------------------------------------------------------------------
//-----------------PRIVATE FUNCTIONS--------------------------------------------
//------------------------------------------------------------------------------
```

```
        var _processInput = function (options) {

            if (options.size === 'mini' || options.size === 'large' || options.size ===
'awesome') {
                options = $.extend({}, Lobibox.notify.OPTIONS[options.size], options);
            }
            options = $.extend({}, Lobibox.notify.OPTIONS[me.$type],
Lobibox.notify.DEFAULTS, options);

            if (options.size !== 'mini' && options.title === true) {
                options.title = Lobibox.notify.OPTIONS[me.$type].title;
            } else if (options.size === 'mini' && options.title === true) {
                options.title = false;
            }
            if (options.icon === true) {
                options.icon = Lobibox.notify.OPTIONS.icons[options.iconSource][me.$type];
            }
            if (options.sound === true) {
                options.sound = Lobibox.notify.OPTIONS[me.$type].sound;
            }
            if (options.sound) {
                options.sound = options.soundPath + options.sound + options.soundExt;
            }
            return options;
        };

        var _appendInWrapper = function ($el, $wrapper) {
            if (me.$options.size === 'normal') {
                if ($wrapper.hasClass('bottom')) {
                    $wrapper.prepend($el);
                } else {
                    $wrapper.append($el);
                }

            } else if (me.$options.size === 'mini') {
                if ($wrapper.hasClass('bottom')) {
                    $wrapper.prepend($el);
                } else {
                    $wrapper.append($el);
                }
            } else if (me.$options.size === 'large') {
                var tabPane = _createTabPane().append($el);
                var $li = _createTabControl(tabPane.attr('id'));
                $wrapper.find('.lb-notify-wrapper').append(tabPane);
                $wrapper.find('.lb-notify-tabs').append($li);
                _activateTab($li);
                $li.find('>a').click(function () {
                    _activateTab($li);
                });
            }
            else if (me.$options.size === 'awesome') {
                var tabPane = _createTabPane().append($el);
                var $li = _createTabControl(tabPane.attr('id'));
                $wrapper.find('.lb-notify-wrapper').append(tabPane);
                $wrapper.find('.lb-notify-tabs').append($li);
                _activateTab($li);
                $li.find('>a').click(function () {
                    _activateTab($li);
                });
            }
        };
        var _activateTab = function ($li) {
            $li.closest('.lb-notify-tabs').find('>li').removeClass('active');
            $li.addClass('active');
            var $current = $($li.find('>a').attr('href'));
            $current.closest('.lb-notify-wrapper').find('>.lb-tab-pane').removeClass('active
');
            $current.addClass('active');
        };
        var _createTabControl = function (tabPaneId) {
            var $li = $('<li></li>', {
                'class': Lobibox.notify.OPTIONS[me.$type]['class']
            });
            $('<a></a>', {
                'href': '#' + tabPaneId
            }).append('<i class="tab-control-icon ' + me.$options.icon + '"></i>')
                .appendTo($li);
            return $li;
        };
        var _createTabPane = function () {
            return $('<div></div>',
                {
                    'class': 'lb-tab-pane',
                    'id': Math.randomString(10)
                });
        };
        var _createNotifyWrapper = function () {
            var selector = (me.$options.size === 'large' ? '.lobibox-notify-wrapper-large'
 : '.lobibox-notify-wrapper')
                    + "." + me.$options.position.replace(/\s/gi, '.'),
                $wrapper;

            //var classes = me.$options.position.split(" ");
            $wrapper = $(selector);
            if ($wrapper.length === 0) {
                $wrapper = $('<div></div>')
                    .addClass(selector.replace(/\./g, ' ').trim())
                    .appendTo($('body'));
                if (me.$options.size === 'large') {
                    $wrapper.append($('<ul class="lb-notify-tabs"></ul>'))
                        .append($('<div class="lb-notify-wrapper"></div>'));
                }
                if (me.$options.size === 'awesome') {
                    $wrapper.append($('<ul class="lb-notify-tabs"></ul>'))
                        .append($('<div class="lb-notify-wrapper"></div>'));
                }
            }
            return $wrapper;
        };
        var _createNotify = function () {
            var OPTS = Lobibox.notify.OPTIONS,
                $iconEl,
                $innerIconEl,
                $iconWrapper,
                $body,
                $msg,
                $notify = $('<div></div>', {
                    'class': 'lobibox-notify ' + OPTS[me.$type]['class'] + ' ' +
OPTS['class'] + ' ' + me.$options.showClass
                });

            $iconWrapper = $('<div
class="lobibox-notify-icon-wrapper"></div>').appendTo($notify);
            $iconEl = $('<div class="lobibox-notify-icon"></div>').appendTo($iconWrapper);
            $innerIconEl = $('<div></div>').appendTo($iconEl);

            // Add image or icon depending on given parameters
            if (me.$options.img) {
                $innerIconEl.append('<img src="' + me.$options.img + '"/>');
            } else if (me.$options.icon) {
                $innerIconEl.append('<div class="icon-el"><i class="' + me.$options.icon +
'"></i></div>');
            } else {
                $notify.addClass('without-icon');
            }
            // Create body, append title and message in body and append body in notification
            $msg = $('<div class="lobibox-notify-msg">' + me.$options.msg + '</div>');
```

1

```
            if (me.$options.messageHeight !== false) {
                $msg.css('max-height', me.$options.messageHeight);
            }

            $body = $('<div></div>', {
                'class': 'lobibox-notify-body'
            }).append($msg).appendTo($notify);

            if (me.$options.title) {
                $body.prepend('<div class="lobibox-notify-title">' + me.$options.title +
'<div>');
            }
            _addCloseButton($notify);
            // burada tï¿½klandï¿½ï¿½ï¿½nda kapanmasï¿½ iï¿½in gerekli dï¿½zenlemeler
yapï¿½lï¿½yor
            if (me.$options.size === 'normal' || me.$options.size === 'mini' ||
me.$options.size === 'large' || me.$options.size === 'awesome') {
                _addCloseOnClick($notify);
                _addDelay($notify);
            }

            // Give width to notification
            if (me.$options.width) {
                $notify.css('width', _calculateWidth(me.$options.width));
            }

            return $notify;
        };
        var _addCloseButton = function ($el) {
            if (!me.$options.closable) {
                return;
            }
            $('<span class="lobibox-close">&times;</span>').click(function () {
                me.remove();
            }).appendTo($el);
        };
        var _addCloseOnClick = function ($el) {
            if (!me.$options.closeOnClick) {
                return;
            }
            $el.click(function () {
                me.remove();
            });
        };
        var _addDelay = function ($el) {
            if (!me.$options.delay) {
                return;
            }
            if (me.$options.delayIndicator) {
                var delay = $('<div class="lobibox-delay-indicator"><div></div></div>');
                $el.append(delay);
            }
            var time = 0;
            var interval = 1000 / 30;
            var currentTime = new Date().getTime();
            var timer = setInterval(function () {
                if (me.$options.continueDelayOnInactiveTab){
                    time = new Date().getTime() - currentTime;
                } else {
                    time += interval;
                }

                var width = 100 * time / me.$options.delay;
                if (width >= 100) {
                    width = 100;
                    me.remove();
                    timer = clearInterval(timer);
                }
                if (me.$options.delayIndicator) {
                    delay.find('div').css('width', width + "%");
                }

            }, interval);

            if (me.$options.pauseDelayOnHover) {
                $el.on('mouseenter.lobibox', function () {
                    interval = 0;
                }).on('mouseleave.lobibox', function () {
                    interval = 1000 / 30;
                });
            }
        };
        var _findTabToActivate = function ($li) {
            var $itemToActivate = $li.prev();
            if ($itemToActivate.length === 0) {
                $itemToActivate = $li.next();
            }
            if ($itemToActivate.length === 0) {
                return null;
            }
            return $itemToActivate;
        };
        var _calculateWidth = function (width) {
            width = Math.min($(window).outerWidth(), width);
            return width;
        };
//------------------------------------------------------------------------------
//----------------PROTOTYPE FUNCTIONS-------------------------------------------
//------------------------------------------------------------------------------
        /**
         * Delete the notification
         *
         * @returns {LobiboxNotify}
         */
        this.remove = function () {
            me.$el.removeClass(me.$options.showClass)
                .addClass(me.$options.hideClass);
            var parent = me.$el.parent();
            var wrapper = parent.closest('.lobibox-notify-wrapper-large');

            var href = '#' + parent.attr('id');

            var $li = wrapper.find('>.lb-notify-tabs>li:has(a[href="' + href + '"])');
            $li.addClass(Lobibox.notify.OPTIONS['class'])
                .addClass(me.$options.hideClass);
            setTimeout(function () {
                if (me.$options.size === 'normal' || me.$options.size === 'mini') {
                    me.$el.remove();
                } else if (me.$options.size === 'large') {

                    var $newLi = _findTabToActivate($li);
                    if ($newLi) {
                        _activateTab($newLi);
                    }
                    $li.remove();
                    parent.remove();
                }
                else if (me.$options.size === 'awesome') {

                    var $newLi = _findTabToActivate($li);
                    if ($newLi) {
                        _activateTab($newLi);
                    }
                    $li.remove();
                    parent.remove();
                }
                var list = Lobibox.notify.list;
                var ind = list.indexOf(me);
```

```
                  list.splice(ind, 1);
                  var next = list[ind];
                  if (next && next.$options.showAfterPrevious){
                      next._init();
                  }
              }, 500);
              return me;
          };
          me._init = function () {
              // Create notification
              var $notify = _createNotify();
              if (me.$options.size === 'mini') {
                  $notify.addClass('notify-mini');
              }

              if (typeof me.$options.position === 'string') {
                  var $wrapper = _createNotifyWrapper();
                  _appendInWrapper($notify, $wrapper);
                  if ($wrapper.hasClass('center')) {
                      $wrapper.css('margin-left', '-' + ($wrapper.width() / 2) + "px");
                  }
              } else {
                  $('body').append($notify);
                  $notify.css({
                      'position': 'fixed',
                      left: me.$options.position.left,
                      top: me.$options.position.top
                  });
              }

              me.$el = $notify;
              if (me.$options.sound) {
                  var snd = new Audio(me.$options.sound); // buffers automatically when
created
                  snd.play();
              }
              if (me.$options.rounded) {
                  me.$el.addClass('rounded');
              }
              me.$el.on('click.lobibox', function(ev){
                  if (me.$options.onClickUrl){
                      window.location.href = me.$options.onClickUrl;
                  }
                  if (me.$options.onClick && typeof me.$options.onClick === 'function'){
                      me.$options.onClick.call(me, ev);
                  }
              });
              me.$el.data('lobibox', me);
          };
//------------------------------------------------------------------------------
//------------------------------------------------------------------------------
//------------------------------------------------------------------------------
          this.$type = type;
          this.$options = _processInput(options);
          if (!me.$options.showAfterPrevious || Lobibox.notify.list.length === 0){
              this._init();
          }

      };

      Lobibox.notify = function (type, options) {
          if (["default", "info", "warning", "error", "success"].indexOf(type) > -1) {
              var lobibox = new LobiboxNotify(type, options);
              Lobibox.notify.list.push(lobibox);
              return lobibox;
          }
      };
      Lobibox.notify.list = [];
      Lobibox.notify.closeAll = function () {
          var list = Lobibox.notify.list;
          for (var i in list){
              list[i].remove();
          }
      };
      //User can set default options to this variable
      Lobibox.notify.DEFAULTS = {
          title: true,                // Title of notification. If you do not include the
  title in options it will automatically takes its value
          //from Lobibox.notify.OPTIONS object depending of the type of the notifications or
  set custom string. Set this false to disable title
          size: 'normal',                                            // normal, mini, large
          soundPath: '/Content/tools/otomasyon-notification/sounds/',       // The folder
  path where sounds are located
          soundExt: '.ogg',           // Default extension for all sounds
          showClass: 'fadeInDown',    // Show animation class.
          hideClass: 'zoomOut',       // Hide animation class.
          icon: true,                 // Icon of notification. Leave as is for default icon
  or set custom string
          msg: '',                    // Message of notification
          img: null,                  // Image source string
          closable: true,             // Make notifications closable
          hideCloseButton: false,     // Notification may be closable but you can hide close
  button and it will be closed by clicking on notification itsef
          delay: 6000,                // Hide notification after this time (in miliseconds)
          delayIndicator: true,       // Show timer indicator
          closeOnClick: true,         // Close notifications by clicking on them
          width: 500,                 // Width of notification box
          sound: true,                // Sound of notification. Set this false to disable
  sound. Leave as is for default sound or set custom soud path
          // Place to show notification. Available options: "top left", "top right", "bottom
  left", "bottom right", "center top", "center bottom"
          // It can also be object {left: number, top: number} to position notification at
  any place
          position: "bottom right",
          iconSource: 'bootstrap',    // "bootstrap" or "fontAwesome" the library which will
  be used for icons
          rounded: false,             // Whether to make notification corners rounded
          messageHeight: 60,          // Notification message maximum height. This is not for
  notification itself, this is for <code>.lobibox-notify-msg</code>
          pauseDelayOnHover: true,    // When you mouse over on notification delay (if it is
  enabled) will be paused.
          onClickUrl: null,           // The url which will be opened when notification is
  clicked
          showAfterPrevious: false,   // Set this to true if you want notification not to be
  shown until previous notification is closed. This is useful for notification queues
          continueDelayOnInactiveTab: true, // Continue delay when browser tab is inactive

          // Events
          onClick: null
      };
      //This variable is necessary.
      // large boyutunda sadece hata mesajlarï¿½ gï¿½sterildiï¿½inden
      // title ï¿½zelliï¿½i gï¿½ncellendi
      Lobibox.notify.OPTIONS = {
          'class': 'animated-fast',
          large: {
              width: 500,
              messageHeight: 96,
              'title': 'Error'
          },
          awesome: {
              width: 500,
              messageHeight: 96,
              'title': 'Situation'
          },
          mini: {
              'class': 'notify-mini',
              messageHeight: 32
          },
          default: {
              'class': 'lobibox-notify-default',
              'title': 'Default',
              sound: false
          },
          success: {
              'class': 'lobibox-notify-success',
              'title': 'Success',
              sound: 'sound2'
          },
          error: {
              'class': 'lobibox-notify-error',
              'title': 'Error',
              sound: 'sound4'
          },
          warning: {
              'class': 'lobibox-notify-warning',
              'title': 'Warning',
              sound: 'sound5'
          },
          info: {
              'class': 'lobibox-notify-info',
              'title': 'Info',
              sound: 'sound6'
          },
          icons: {
              bootstrap: {
                  success: 'glyphicon glyphicon-ok-sign',
                  error: 'glyphicon glyphicon-remove-sign',
                  warning: 'glyphicon glyphicon-exclamation-sign',
                  info: 'glyphicon glyphicon-info-sign'
              },
              fontAwesome: {
                  success: 'fa fa-check-circle',
                  error: 'fa fa-times-circle',
                  warning: 'fa fa-exclamation-circle',
                  info: 'fa fa-info-circle'
              }
          }
      };
  })();
```

Variables

Name	Description
Lobibox (see page 1)	This is variable Lobibox.

1.2.18 site.js

Please see documentation at https://docs.microsoft.com/aspnet/core/client-side/bundling-and-minification for details on configuring this project to bundle and minify static web assets.

1.2.19 WeatherStackNetCore.csproj

This is file WeatherStackNetCore.csproj.

1.2.20 WeatherStackNetCore.sln

This is file WeatherStackNetCore.sln.

Index