(function (global, factory) {

typeof exports === 'object' && typeof module !== 'undefined' ? factory(exports, require('jquery')) :

typeof define === 'function' && define.amd ? define(['exports', 'jquery'], factory) :

(global = typeof globalThis !== 'undefined' ? globalThis : global || self, factory(global.adminlte = {}, global.jQuery));

})(this, (function (exports, $) { 'use strict';

function \_interopDefaultLegacy (e) { return e && typeof e === 'object' && 'default' in e ? e : { 'default': e }; }

var $\_\_default = /\*#\_\_PURE\_\_\*/\_interopDefaultLegacy($);

var NAME$e = 'CardRefresh';

var DATA\_KEY$e = 'lte.cardrefresh';

var EVENT\_KEY$7 = "." + DATA\_KEY$e;

var JQUERY\_NO\_CONFLICT$e = $\_\_default["default"].fn[NAME$e];

var EVENT\_LOADED = "loaded" + EVENT\_KEY$7;

var EVENT\_OVERLAY\_ADDED = "overlay.added" + EVENT\_KEY$7;

var EVENT\_OVERLAY\_REMOVED = "overlay.removed" + EVENT\_KEY$7;

var CLASS\_NAME\_CARD$1 = 'card';

var SELECTOR\_CARD$1 = "." + CLASS\_NAME\_CARD$1;

var SELECTOR\_DATA\_REFRESH = '[data-card-widget="card-refresh"]';

var Default$c = {

source: '',

sourceSelector: '',

params: {},

trigger: SELECTOR\_DATA\_REFRESH,

content: '.card-body',

loadInContent: true,

loadOnInit: true,

loadErrorTemplate: true,

responseType: '',

overlayTemplate: '<div class="overlay"><i class="fas fa-2x fa-sync-alt fa-spin"></i></div>',

errorTemplate: '<span class="text-danger"></span>',

onLoadStart: function onLoadStart() {},

onLoadDone: function onLoadDone(response) {

return response;

},

onLoadFail: function onLoadFail(\_jqXHR, \_textStatus, \_errorThrown) {}

};

var CardRefresh = /\*#\_\_PURE\_\_\*/function () {

function CardRefresh(element, settings) {

this.\_element = element;

this.\_parent = element.parents(SELECTOR\_CARD$1).first();

this.\_settings = $\_\_default["default"].extend({}, Default$c, settings);

this.\_overlay = $\_\_default["default"](this.\_settings.overlayTemplate);

if (element.hasClass(CLASS\_NAME\_CARD$1)) {

this.\_parent = element;

}

if (this.\_settings.source === '') {

throw new Error('Source url was not defined. Please specify a url in your CardRefresh source option.');

}

}

var \_proto = CardRefresh.prototype;

\_proto.load = function load() {

var \_this = this;

this.\_addOverlay();

this.\_settings.onLoadStart.call($\_\_default["default"](this));

$\_\_default["default"].get(this.\_settings.source, this.\_settings.params, function (response) {

if (\_this.\_settings.loadInContent) {

if (\_this.\_settings.sourceSelector !== '') {

response = $\_\_default["default"](response).find(\_this.\_settings.sourceSelector).html();

}

\_this.\_parent.find(\_this.\_settings.content).html(response);

}

\_this.\_settings.onLoadDone.call($\_\_default["default"](\_this), response);

\_this.\_removeOverlay();

}, this.\_settings.responseType !== '' && this.\_settings.responseType).fail(function (jqXHR, textStatus, errorThrown) {

\_this.\_removeOverlay();

if (\_this.\_settings.loadErrorTemplate) {

var msg = $\_\_default["default"](\_this.\_settings.errorTemplate).text(errorThrown);

\_this.\_parent.find(\_this.\_settings.content).empty().append(msg);

}

\_this.\_settings.onLoadFail.call($\_\_default["default"](\_this), jqXHR, textStatus, errorThrown);

});

$\_\_default["default"](this.\_element).trigger($\_\_default["default"].Event(EVENT\_LOADED));

};

\_proto.\_addOverlay = function \_addOverlay() {

this.\_parent.append(this.\_overlay);

$\_\_default["default"](this.\_element).trigger($\_\_default["default"].Event(EVENT\_OVERLAY\_ADDED));

};

\_proto.\_removeOverlay = function \_removeOverlay() {

this.\_parent.find(this.\_overlay).remove();

$\_\_default["default"](this.\_element).trigger($\_\_default["default"].Event(EVENT\_OVERLAY\_REMOVED));

} // Private

;

\_proto.\_init = function \_init() {

var \_this2 = this;

$\_\_default["default"](this).find(this.\_settings.trigger).on('click', function () {

\_this2.load();

});

if (this.\_settings.loadOnInit) {

this.load();

}

} // Static

;

CardRefresh.\_jQueryInterface = function \_jQueryInterface(config) {

var data = $\_\_default["default"](this).data(DATA\_KEY$e);

var \_options = $\_\_default["default"].extend({}, Default$c, $\_\_default["default"](this).data());

if (!data) {

data = new CardRefresh($\_\_default["default"](this), \_options);

$\_\_default["default"](this).data(DATA\_KEY$e, typeof config === 'string' ? data : config);

}

if (typeof config === 'string' && /load/.test(config)) {

data[config]();

} else {

data.\_init($\_\_default["default"](this));

}

};

return CardRefresh;

}();

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_REFRESH, function (event) {

if (event) {

event.preventDefault();

}

CardRefresh.\_jQueryInterface.call($\_\_default["default"](this), 'load');

});

$\_\_default["default"](function () {

$\_\_default["default"](SELECTOR\_DATA\_REFRESH).each(function () {

CardRefresh.\_jQueryInterface.call($\_\_default["default"](this));

});

});

$\_\_default["default"].fn[NAME$e] = CardRefresh.\_jQueryInterface;

$\_\_default["default"].fn[NAME$e].Constructor = CardRefresh;

$\_\_default["default"].fn[NAME$e].noConflict = function () {

$\_\_default["default"].fn[NAME$e] = JQUERY\_NO\_CONFLICT$e;

return CardRefresh.\_jQueryInterface;

};

var NAME$d = 'CardWidget';

var DATA\_KEY$d = 'lte.cardwidget';

var EVENT\_KEY$6 = "." + DATA\_KEY$d;

var JQUERY\_NO\_CONFLICT$d = $\_\_default["default"].fn[NAME$d];

var EVENT\_EXPANDED$3 = "expanded" + EVENT\_KEY$6;

var EVENT\_COLLAPSED$4 = "collapsed" + EVENT\_KEY$6;

var EVENT\_MAXIMIZED = "maximized" + EVENT\_KEY$6;

var EVENT\_MINIMIZED = "minimized" + EVENT\_KEY$6;

var EVENT\_REMOVED$1 = "removed" + EVENT\_KEY$6;

var CLASS\_NAME\_CARD = 'card';

var CLASS\_NAME\_COLLAPSED$1 = 'collapsed-card';

var CLASS\_NAME\_COLLAPSING = 'collapsing-card';

var CLASS\_NAME\_EXPANDING = 'expanding-card';

var CLASS\_NAME\_WAS\_COLLAPSED = 'was-collapsed';

var CLASS\_NAME\_MAXIMIZED = 'maximized-card';

var SELECTOR\_DATA\_REMOVE = '[data-card-widget="remove"]';

var SELECTOR\_DATA\_COLLAPSE = '[data-card-widget="collapse"]';

var SELECTOR\_DATA\_MAXIMIZE = '[data-card-widget="maximize"]';

var SELECTOR\_CARD = "." + CLASS\_NAME\_CARD;

var SELECTOR\_CARD\_HEADER = '.card-header';

var SELECTOR\_CARD\_BODY = '.card-body';

var SELECTOR\_CARD\_FOOTER = '.card-footer';

var Default$b = {

animationSpeed: 'normal',

collapseTrigger: SELECTOR\_DATA\_COLLAPSE,

removeTrigger: SELECTOR\_DATA\_REMOVE,

maximizeTrigger: SELECTOR\_DATA\_MAXIMIZE,

collapseIcon: 'fa-minus',

expandIcon: 'fa-plus',

maximizeIcon: 'fa-expand',

minimizeIcon: 'fa-compress'

};

var CardWidget = /\*#\_\_PURE\_\_\*/function () {

function CardWidget(element, settings) {

this.\_element = element;

this.\_parent = element.parents(SELECTOR\_CARD).first();

if (element.hasClass(CLASS\_NAME\_CARD)) {

this.\_parent = element;

}

this.\_settings = $\_\_default["default"].extend({}, Default$b, settings);

}

var \_proto = CardWidget.prototype;

\_proto.collapse = function collapse() {

var \_this = this;

this.\_parent.addClass(CLASS\_NAME\_COLLAPSING).children(SELECTOR\_CARD\_BODY + ", " + SELECTOR\_CARD\_FOOTER).slideUp(this.\_settings.animationSpeed, function () {

\_this.\_parent.addClass(CLASS\_NAME\_COLLAPSED$1).removeClass(CLASS\_NAME\_COLLAPSING);

});

this.\_parent.find("> " + SELECTOR\_CARD\_HEADER + " " + this.\_settings.collapseTrigger + " ." + this.\_settings.collapseIcon).addClass(this.\_settings.expandIcon).removeClass(this.\_settings.collapseIcon);

this.\_element.trigger($\_\_default["default"].Event(EVENT\_COLLAPSED$4), this.\_parent);

};

\_proto.expand = function expand() {

var \_this2 = this;

this.\_parent.addClass(CLASS\_NAME\_EXPANDING).children(SELECTOR\_CARD\_BODY + ", " + SELECTOR\_CARD\_FOOTER).slideDown(this.\_settings.animationSpeed, function () {

\_this2.\_parent.removeClass(CLASS\_NAME\_COLLAPSED$1).removeClass(CLASS\_NAME\_EXPANDING);

});

this.\_parent.find("> " + SELECTOR\_CARD\_HEADER + " " + this.\_settings.collapseTrigger + " ." + this.\_settings.expandIcon).addClass(this.\_settings.collapseIcon).removeClass(this.\_settings.expandIcon);

this.\_element.trigger($\_\_default["default"].Event(EVENT\_EXPANDED$3), this.\_parent);

};

\_proto.remove = function remove() {

this.\_parent.slideUp();

this.\_element.trigger($\_\_default["default"].Event(EVENT\_REMOVED$1), this.\_parent);

};

\_proto.toggle = function toggle() {

if (this.\_parent.hasClass(CLASS\_NAME\_COLLAPSED$1)) {

this.expand();

return;

}

this.collapse();

};

\_proto.maximize = function maximize() {

this.\_parent.find(this.\_settings.maximizeTrigger + " ." + this.\_settings.maximizeIcon).addClass(this.\_settings.minimizeIcon).removeClass(this.\_settings.maximizeIcon);

this.\_parent.css({

height: this.\_parent.height(),

width: this.\_parent.width(),

transition: 'all .15s'

}).delay(150).queue(function () {

var $element = $\_\_default["default"](this);

$element.addClass(CLASS\_NAME\_MAXIMIZED);

$\_\_default["default"]('html').addClass(CLASS\_NAME\_MAXIMIZED);

if ($element.hasClass(CLASS\_NAME\_COLLAPSED$1)) {

$element.addClass(CLASS\_NAME\_WAS\_COLLAPSED);

}

$element.dequeue();

});

this.\_element.trigger($\_\_default["default"].Event(EVENT\_MAXIMIZED), this.\_parent);

};

\_proto.minimize = function minimize() {

this.\_parent.find(this.\_settings.maximizeTrigger + " ." + this.\_settings.minimizeIcon).addClass(this.\_settings.maximizeIcon).removeClass(this.\_settings.minimizeIcon);

this.\_parent.css('cssText', "height: " + this.\_parent[0].style.height + " !important; width: " + this.\_parent[0].style.width + " !important; transition: all .15s;").delay(10).queue(function () {

var $element = $\_\_default["default"](this);

$element.removeClass(CLASS\_NAME\_MAXIMIZED);

$\_\_default["default"]('html').removeClass(CLASS\_NAME\_MAXIMIZED);

$element.css({

height: 'inherit',

width: 'inherit'

});

if ($element.hasClass(CLASS\_NAME\_WAS\_COLLAPSED)) {

$element.removeClass(CLASS\_NAME\_WAS\_COLLAPSED);

}

$element.dequeue();

});

this.\_element.trigger($\_\_default["default"].Event(EVENT\_MINIMIZED), this.\_parent);

};

\_proto.toggleMaximize = function toggleMaximize() {

if (this.\_parent.hasClass(CLASS\_NAME\_MAXIMIZED)) {

this.minimize();

return;

}

this.maximize();

} // Private

;

\_proto.\_init = function \_init(card) {

var \_this3 = this;

this.\_parent = card;

$\_\_default["default"](this).find(this.\_settings.collapseTrigger).click(function () {

\_this3.toggle();

});

$\_\_default["default"](this).find(this.\_settings.maximizeTrigger).click(function () {

\_this3.toggleMaximize();

});

$\_\_default["default"](this).find(this.\_settings.removeTrigger).click(function () {

\_this3.remove();

});

} // Static

;

CardWidget.\_jQueryInterface = function \_jQueryInterface(config) {

var data = $\_\_default["default"](this).data(DATA\_KEY$d);

var \_options = $\_\_default["default"].extend({}, Default$b, $\_\_default["default"](this).data());

if (!data) {

data = new CardWidget($\_\_default["default"](this), \_options);

$\_\_default["default"](this).data(DATA\_KEY$d, typeof config === 'string' ? data : config);

}

if (typeof config === 'string' && /collapse|expand|remove|toggle|maximize|minimize|toggleMaximize/.test(config)) {

data[config]();

} else if (typeof config === 'object') {

data.\_init($\_\_default["default"](this));

}

};

return CardWidget;

}();

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_COLLAPSE, function (event) {

if (event) {

event.preventDefault();

}

CardWidget.\_jQueryInterface.call($\_\_default["default"](this), 'toggle');

});

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_REMOVE, function (event) {

if (event) {

event.preventDefault();

}

CardWidget.\_jQueryInterface.call($\_\_default["default"](this), 'remove');

});

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_MAXIMIZE, function (event) {

if (event) {

event.preventDefault();

}

CardWidget.\_jQueryInterface.call($\_\_default["default"](this), 'toggleMaximize');

});

$\_\_default["default"].fn[NAME$d] = CardWidget.\_jQueryInterface;

$\_\_default["default"].fn[NAME$d].Constructor = CardWidget;

$\_\_default["default"].fn[NAME$d].noConflict = function () {

$\_\_default["default"].fn[NAME$d] = JQUERY\_NO\_CONFLICT$d;

return CardWidget.\_jQueryInterface;

};

var NAME$c = 'ControlSidebar';

var DATA\_KEY$c = 'lte.controlsidebar';

var EVENT\_KEY$5 = "." + DATA\_KEY$c;

var JQUERY\_NO\_CONFLICT$c = $\_\_default["default"].fn[NAME$c];

var EVENT\_COLLAPSED$3 = "collapsed" + EVENT\_KEY$5;

var EVENT\_COLLAPSED\_DONE$1 = "collapsed-done" + EVENT\_KEY$5;

var EVENT\_EXPANDED$2 = "expanded" + EVENT\_KEY$5;

var SELECTOR\_CONTROL\_SIDEBAR = '.control-sidebar';

var SELECTOR\_CONTROL\_SIDEBAR\_CONTENT$1 = '.control-sidebar-content';

var SELECTOR\_DATA\_TOGGLE$4 = '[data-widget="control-sidebar"]';

var SELECTOR\_HEADER$1 = '.main-header';

var SELECTOR\_FOOTER$1 = '.main-footer';

var CLASS\_NAME\_CONTROL\_SIDEBAR\_ANIMATE = 'control-sidebar-animate';

var CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN$1 = 'control-sidebar-open';

var CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE = 'control-sidebar-slide-open';

var CLASS\_NAME\_LAYOUT\_FIXED$1 = 'layout-fixed';

var CLASS\_NAME\_NAVBAR\_FIXED = 'layout-navbar-fixed';

var CLASS\_NAME\_NAVBAR\_SM\_FIXED = 'layout-sm-navbar-fixed';

var CLASS\_NAME\_NAVBAR\_MD\_FIXED = 'layout-md-navbar-fixed';

var CLASS\_NAME\_NAVBAR\_LG\_FIXED = 'layout-lg-navbar-fixed';

var CLASS\_NAME\_NAVBAR\_XL\_FIXED = 'layout-xl-navbar-fixed';

var CLASS\_NAME\_FOOTER\_FIXED = 'layout-footer-fixed';

var CLASS\_NAME\_FOOTER\_SM\_FIXED = 'layout-sm-footer-fixed';

var CLASS\_NAME\_FOOTER\_MD\_FIXED = 'layout-md-footer-fixed';

var CLASS\_NAME\_FOOTER\_LG\_FIXED = 'layout-lg-footer-fixed';

var CLASS\_NAME\_FOOTER\_XL\_FIXED = 'layout-xl-footer-fixed';

var Default$a = {

controlsidebarSlide: true,

scrollbarTheme: 'os-theme-light',

scrollbarAutoHide: 'l',

target: SELECTOR\_CONTROL\_SIDEBAR,

animationSpeed: 300

};

var ControlSidebar = /\*#\_\_PURE\_\_\*/function () {

function ControlSidebar(element, config) {

this.\_element = element;

this.\_config = config;

} // Public

var \_proto = ControlSidebar.prototype;

\_proto.collapse = function collapse() {

var \_this = this;

var $body = $\_\_default["default"]('body');

var $html = $\_\_default["default"]('html'); // Show the control sidebar

if (this.\_config.controlsidebarSlide) {

$html.addClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_ANIMATE);

$body.removeClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE).delay(300).queue(function () {

$\_\_default["default"](SELECTOR\_CONTROL\_SIDEBAR).hide();

$html.removeClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_ANIMATE);

$\_\_default["default"](this).dequeue();

});

} else {

$body.removeClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN$1);

}

$\_\_default["default"](this.\_element).trigger($\_\_default["default"].Event(EVENT\_COLLAPSED$3));

setTimeout(function () {

$\_\_default["default"](\_this.\_element).trigger($\_\_default["default"].Event(EVENT\_COLLAPSED\_DONE$1));

}, this.\_config.animationSpeed);

};

\_proto.show = function show(toggle) {

if (toggle === void 0) {

toggle = false;

}

var $body = $\_\_default["default"]('body');

var $html = $\_\_default["default"]('html');

if (toggle) {

$\_\_default["default"](SELECTOR\_CONTROL\_SIDEBAR).hide();

} // Collapse the control sidebar

if (this.\_config.controlsidebarSlide) {

$html.addClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_ANIMATE);

$\_\_default["default"](this.\_config.target).show().delay(10).queue(function () {

$body.addClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE).delay(300).queue(function () {

$html.removeClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_ANIMATE);

$\_\_default["default"](this).dequeue();

});

$\_\_default["default"](this).dequeue();

});

} else {

$body.addClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN$1);

}

this.\_fixHeight();

this.\_fixScrollHeight();

$\_\_default["default"](this.\_element).trigger($\_\_default["default"].Event(EVENT\_EXPANDED$2));

};

\_proto.toggle = function toggle() {

var $body = $\_\_default["default"]('body');

var target = this.\_config.target;

var notVisible = !$\_\_default["default"](target).is(':visible');

var shouldClose = $body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN$1) || $body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE);

var shouldToggle = notVisible && ($body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN$1) || $body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE));

if (notVisible || shouldToggle) {

// Open the control sidebar

this.show(notVisible);

} else if (shouldClose) {

// Close the control sidebar

this.collapse();

}

} // Private

;

\_proto.\_init = function \_init() {

var \_this2 = this;

var $body = $\_\_default["default"]('body');

var shouldNotHideAll = $body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN$1) || $body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE);

if (shouldNotHideAll) {

$\_\_default["default"](SELECTOR\_CONTROL\_SIDEBAR).not(this.\_config.target).hide();

$\_\_default["default"](this.\_config.target).css('display', 'block');

} else {

$\_\_default["default"](SELECTOR\_CONTROL\_SIDEBAR).hide();

}

this.\_fixHeight();

this.\_fixScrollHeight();

$\_\_default["default"](window).resize(function () {

\_this2.\_fixHeight();

\_this2.\_fixScrollHeight();

});

$\_\_default["default"](window).scroll(function () {

var $body = $\_\_default["default"]('body');

var shouldFixHeight = $body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN$1) || $body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE);

if (shouldFixHeight) {

\_this2.\_fixScrollHeight();

}

});

};

\_proto.\_isNavbarFixed = function \_isNavbarFixed() {

var $body = $\_\_default["default"]('body');

return $body.hasClass(CLASS\_NAME\_NAVBAR\_FIXED) || $body.hasClass(CLASS\_NAME\_NAVBAR\_SM\_FIXED) || $body.hasClass(CLASS\_NAME\_NAVBAR\_MD\_FIXED) || $body.hasClass(CLASS\_NAME\_NAVBAR\_LG\_FIXED) || $body.hasClass(CLASS\_NAME\_NAVBAR\_XL\_FIXED);

};

\_proto.\_isFooterFixed = function \_isFooterFixed() {

var $body = $\_\_default["default"]('body');

return $body.hasClass(CLASS\_NAME\_FOOTER\_FIXED) || $body.hasClass(CLASS\_NAME\_FOOTER\_SM\_FIXED) || $body.hasClass(CLASS\_NAME\_FOOTER\_MD\_FIXED) || $body.hasClass(CLASS\_NAME\_FOOTER\_LG\_FIXED) || $body.hasClass(CLASS\_NAME\_FOOTER\_XL\_FIXED);

};

\_proto.\_fixScrollHeight = function \_fixScrollHeight() {

var $body = $\_\_default["default"]('body');

var $controlSidebar = $\_\_default["default"](this.\_config.target);

if (!$body.hasClass(CLASS\_NAME\_LAYOUT\_FIXED$1)) {

return;

}

var heights = {

scroll: $\_\_default["default"](document).height(),

window: $\_\_default["default"](window).height(),

header: $\_\_default["default"](SELECTOR\_HEADER$1).outerHeight(),

footer: $\_\_default["default"](SELECTOR\_FOOTER$1).outerHeight()

};

var positions = {

bottom: Math.abs(heights.window + $\_\_default["default"](window).scrollTop() - heights.scroll),

top: $\_\_default["default"](window).scrollTop()

};

var navbarFixed = this.\_isNavbarFixed() && $\_\_default["default"](SELECTOR\_HEADER$1).css('position') === 'fixed';

var footerFixed = this.\_isFooterFixed() && $\_\_default["default"](SELECTOR\_FOOTER$1).css('position') === 'fixed';

var $controlsidebarContent = $\_\_default["default"](this.\_config.target + ", " + this.\_config.target + " " + SELECTOR\_CONTROL\_SIDEBAR\_CONTENT$1);

if (positions.top === 0 && positions.bottom === 0) {

$controlSidebar.css({

bottom: heights.footer,

top: heights.header

});

$controlsidebarContent.css('height', heights.window - (heights.header + heights.footer));

} else if (positions.bottom <= heights.footer) {

if (footerFixed === false) {

var top = heights.header - positions.top;

$controlSidebar.css('bottom', heights.footer - positions.bottom).css('top', top >= 0 ? top : 0);

$controlsidebarContent.css('height', heights.window - (heights.footer - positions.bottom));

} else {

$controlSidebar.css('bottom', heights.footer);

}

} else if (positions.top <= heights.header) {

if (navbarFixed === false) {

$controlSidebar.css('top', heights.header - positions.top);

$controlsidebarContent.css('height', heights.window - (heights.header - positions.top));

} else {

$controlSidebar.css('top', heights.header);

}

} else if (navbarFixed === false) {

$controlSidebar.css('top', 0);

$controlsidebarContent.css('height', heights.window);

} else {

$controlSidebar.css('top', heights.header);

}

if (footerFixed && navbarFixed) {

$controlsidebarContent.css('height', '100%');

$controlSidebar.css('height', '');

} else if (footerFixed || navbarFixed) {

$controlsidebarContent.css('height', '100%');

$controlsidebarContent.css('height', '');

}

};

\_proto.\_fixHeight = function \_fixHeight() {

var $body = $\_\_default["default"]('body');

var $controlSidebar = $\_\_default["default"](this.\_config.target + " " + SELECTOR\_CONTROL\_SIDEBAR\_CONTENT$1);

if (!$body.hasClass(CLASS\_NAME\_LAYOUT\_FIXED$1)) {

$controlSidebar.attr('style', '');

return;

}

var heights = {

window: $\_\_default["default"](window).height(),

header: $\_\_default["default"](SELECTOR\_HEADER$1).outerHeight(),

footer: $\_\_default["default"](SELECTOR\_FOOTER$1).outerHeight()

};

var sidebarHeight = heights.window - heights.header;

if (this.\_isFooterFixed() && $\_\_default["default"](SELECTOR\_FOOTER$1).css('position') === 'fixed') {

sidebarHeight = heights.window - heights.header - heights.footer;

}

$controlSidebar.css('height', sidebarHeight);

if (typeof $\_\_default["default"].fn.overlayScrollbars !== 'undefined') {

$controlSidebar.overlayScrollbars({

className: this.\_config.scrollbarTheme,

sizeAutoCapable: true,

scrollbars: {

autoHide: this.\_config.scrollbarAutoHide,

clickScrolling: true

}

});

}

} // Static

;

ControlSidebar.\_jQueryInterface = function \_jQueryInterface(operation) {

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY$c);

var \_options = $\_\_default["default"].extend({}, Default$a, $\_\_default["default"](this).data());

if (!data) {

data = new ControlSidebar(this, \_options);

$\_\_default["default"](this).data(DATA\_KEY$c, data);

}

if (data[operation] === 'undefined') {

throw new Error(operation + " is not a function");

}

data[operation]();

});

};

return ControlSidebar;

}();

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_TOGGLE$4, function (event) {

event.preventDefault();

ControlSidebar.\_jQueryInterface.call($\_\_default["default"](this), 'toggle');

});

$\_\_default["default"](document).ready(function () {

ControlSidebar.\_jQueryInterface.call($\_\_default["default"](SELECTOR\_DATA\_TOGGLE$4), '\_init');

});

$\_\_default["default"].fn[NAME$c] = ControlSidebar.\_jQueryInterface;

$\_\_default["default"].fn[NAME$c].Constructor = ControlSidebar;

$\_\_default["default"].fn[NAME$c].noConflict = function () {

$\_\_default["default"].fn[NAME$c] = JQUERY\_NO\_CONFLICT$c;

return ControlSidebar.\_jQueryInterface;

};

var NAME$b = 'DirectChat';

var DATA\_KEY$b = 'lte.directchat';

var EVENT\_KEY$4 = "." + DATA\_KEY$b;

var JQUERY\_NO\_CONFLICT$b = $\_\_default["default"].fn[NAME$b];

var EVENT\_TOGGLED = "toggled" + EVENT\_KEY$4;

var SELECTOR\_DATA\_TOGGLE$3 = '[data-widget="chat-pane-toggle"]';

var SELECTOR\_DIRECT\_CHAT = '.direct-chat';

var CLASS\_NAME\_DIRECT\_CHAT\_OPEN = 'direct-chat-contacts-open';

var DirectChat = /\*#\_\_PURE\_\_\*/function () {

function DirectChat(element) {

this.\_element = element;

}

var \_proto = DirectChat.prototype;

\_proto.toggle = function toggle() {

$\_\_default["default"](this.\_element).parents(SELECTOR\_DIRECT\_CHAT).first().toggleClass(CLASS\_NAME\_DIRECT\_CHAT\_OPEN);

$\_\_default["default"](this.\_element).trigger($\_\_default["default"].Event(EVENT\_TOGGLED));

} // Static

;

DirectChat.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY$b);

if (!data) {

data = new DirectChat($\_\_default["default"](this));

$\_\_default["default"](this).data(DATA\_KEY$b, data);

}

data[config]();

});

};

return DirectChat;

}();

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_TOGGLE$3, function (event) {

if (event) {

event.preventDefault();

}

DirectChat.\_jQueryInterface.call($\_\_default["default"](this), 'toggle');

});

$\_\_default["default"].fn[NAME$b] = DirectChat.\_jQueryInterface;

$\_\_default["default"].fn[NAME$b].Constructor = DirectChat;

$\_\_default["default"].fn[NAME$b].noConflict = function () {

$\_\_default["default"].fn[NAME$b] = JQUERY\_NO\_CONFLICT$b;

return DirectChat.\_jQueryInterface;

};

var NAME$a = 'Dropdown';

var DATA\_KEY$a = 'lte.dropdown';

var JQUERY\_NO\_CONFLICT$a = $\_\_default["default"].fn[NAME$a];

var SELECTOR\_NAVBAR = '.navbar';

var SELECTOR\_DROPDOWN\_MENU = '.dropdown-menu';

var SELECTOR\_DROPDOWN\_MENU\_ACTIVE = '.dropdown-menu.show';

var SELECTOR\_DROPDOWN\_TOGGLE = '[data-toggle="dropdown"]';

var CLASS\_NAME\_DROPDOWN\_RIGHT = 'dropdown-menu-right';

var CLASS\_NAME\_DROPDOWN\_SUBMENU = 'dropdown-submenu'; // TODO: this is unused; should be removed along with the extend?

var Default$9 = {};

var Dropdown = /\*#\_\_PURE\_\_\*/function () {

function Dropdown(element, config) {

this.\_config = config;

this.\_element = element;

} // Public

var \_proto = Dropdown.prototype;

\_proto.toggleSubmenu = function toggleSubmenu() {

this.\_element.siblings().show().toggleClass('show');

if (!this.\_element.next().hasClass('show')) {

this.\_element.parents(SELECTOR\_DROPDOWN\_MENU).first().find('.show').removeClass('show').hide();

}

this.\_element.parents('li.nav-item.dropdown.show').on('hidden.bs.dropdown', function () {

$\_\_default["default"]('.dropdown-submenu .show').removeClass('show').hide();

});

};

\_proto.fixPosition = function fixPosition() {

var $element = $\_\_default["default"](SELECTOR\_DROPDOWN\_MENU\_ACTIVE);

if ($element.length === 0) {

return;

}

if ($element.hasClass(CLASS\_NAME\_DROPDOWN\_RIGHT)) {

$element.css({

left: 'inherit',

right: 0

});

} else {

$element.css({

left: 0,

right: 'inherit'

});

}

var offset = $element.offset();

var width = $element.width();

var visiblePart = $\_\_default["default"](window).width() - offset.left;

if (offset.left < 0) {

$element.css({

left: 'inherit',

right: offset.left - 5

});

} else if (visiblePart < width) {

$element.css({

left: 'inherit',

right: 0

});

}

} // Static

;

Dropdown.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY$a);

var \_config = $\_\_default["default"].extend({}, Default$9, $\_\_default["default"](this).data());

if (!data) {

data = new Dropdown($\_\_default["default"](this), \_config);

$\_\_default["default"](this).data(DATA\_KEY$a, data);

}

if (config === 'toggleSubmenu' || config === 'fixPosition') {

data[config]();

}

});

};

return Dropdown;

}();

$\_\_default["default"](SELECTOR\_DROPDOWN\_MENU + " " + SELECTOR\_DROPDOWN\_TOGGLE).on('click', function (event) {

event.preventDefault();

event.stopPropagation();

Dropdown.\_jQueryInterface.call($\_\_default["default"](this), 'toggleSubmenu');

});

$\_\_default["default"](SELECTOR\_NAVBAR + " " + SELECTOR\_DROPDOWN\_TOGGLE).on('click', function (event) {

event.preventDefault();

if ($\_\_default["default"](event.target).parent().hasClass(CLASS\_NAME\_DROPDOWN\_SUBMENU)) {

return;

}

setTimeout(function () {

Dropdown.\_jQueryInterface.call($\_\_default["default"](this), 'fixPosition');

}, 1);

});

$\_\_default["default"].fn[NAME$a] = Dropdown.\_jQueryInterface;

$\_\_default["default"].fn[NAME$a].Constructor = Dropdown;

$\_\_default["default"].fn[NAME$a].noConflict = function () {

$\_\_default["default"].fn[NAME$a] = JQUERY\_NO\_CONFLICT$a;

return Dropdown.\_jQueryInterface;

};

var NAME$9 = 'ExpandableTable';

var DATA\_KEY$9 = 'lte.expandableTable';

var EVENT\_KEY$3 = "." + DATA\_KEY$9;

var JQUERY\_NO\_CONFLICT$9 = $\_\_default["default"].fn[NAME$9];

var EVENT\_EXPANDED$1 = "expanded" + EVENT\_KEY$3;

var EVENT\_COLLAPSED$2 = "collapsed" + EVENT\_KEY$3;

var SELECTOR\_TABLE = '.expandable-table';

var SELECTOR\_EXPANDABLE\_BODY = '.expandable-body';

var SELECTOR\_DATA\_TOGGLE$2 = '[data-widget="expandable-table"]';

var SELECTOR\_ARIA\_ATTR = 'aria-expanded';

var ExpandableTable = /\*#\_\_PURE\_\_\*/function () {

function ExpandableTable(element, options) {

this.\_options = options;

this.\_element = element;

} // Public

var \_proto = ExpandableTable.prototype;

\_proto.init = function init() {

$\_\_default["default"](SELECTOR\_DATA\_TOGGLE$2).each(function (\_, $header) {

var $type = $\_\_default["default"]($header).attr(SELECTOR\_ARIA\_ATTR);

var $body = $\_\_default["default"]($header).next(SELECTOR\_EXPANDABLE\_BODY).children().first().children();

if ($type === 'true') {

$body.show();

} else if ($type === 'false') {

$body.hide();

$body.parent().parent().addClass('d-none');

}

});

};

\_proto.toggleRow = function toggleRow() {

var $element = this.\_element;

if ($element[0].nodeName !== 'TR') {

$element = $element.parent();

if ($element[0].nodeName !== 'TR') {

$element = $element.parent();

}

}

var time = 500;

var $type = $element.attr(SELECTOR\_ARIA\_ATTR);

var $body = $element.next(SELECTOR\_EXPANDABLE\_BODY).children().first().children();

$body.stop();

if ($type === 'true') {

$body.slideUp(time, function () {

$element.next(SELECTOR\_EXPANDABLE\_BODY).addClass('d-none');

});

$element.attr(SELECTOR\_ARIA\_ATTR, 'false');

$element.trigger($\_\_default["default"].Event(EVENT\_COLLAPSED$2));

} else if ($type === 'false') {

$element.next(SELECTOR\_EXPANDABLE\_BODY).removeClass('d-none');

$body.slideDown(time);

$element.attr(SELECTOR\_ARIA\_ATTR, 'true');

$element.trigger($\_\_default["default"].Event(EVENT\_EXPANDED$1));

}

} // Static

;

ExpandableTable.\_jQueryInterface = function \_jQueryInterface(operation) {

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY$9);

if (!data) {

data = new ExpandableTable($\_\_default["default"](this));

$\_\_default["default"](this).data(DATA\_KEY$9, data);

}

if (typeof operation === 'string' && /init|toggleRow/.test(operation)) {

data[operation]();

}

});

};

return ExpandableTable;

}();

$\_\_default["default"](SELECTOR\_TABLE).ready(function () {

ExpandableTable.\_jQueryInterface.call($\_\_default["default"](this), 'init');

});

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_TOGGLE$2, function () {

ExpandableTable.\_jQueryInterface.call($\_\_default["default"](this), 'toggleRow');

});

$\_\_default["default"].fn[NAME$9] = ExpandableTable.\_jQueryInterface;

$\_\_default["default"].fn[NAME$9].Constructor = ExpandableTable;

$\_\_default["default"].fn[NAME$9].noConflict = function () {

$\_\_default["default"].fn[NAME$9] = JQUERY\_NO\_CONFLICT$9;

return ExpandableTable.\_jQueryInterface;

};

var NAME$8 = 'Fullscreen';

var DATA\_KEY$8 = 'lte.fullscreen';

var JQUERY\_NO\_CONFLICT$8 = $\_\_default["default"].fn[NAME$8];

var SELECTOR\_DATA\_WIDGET$2 = '[data-widget="fullscreen"]';

var SELECTOR\_ICON = SELECTOR\_DATA\_WIDGET$2 + " i";

var EVENT\_FULLSCREEN\_CHANGE = 'webkitfullscreenchange mozfullscreenchange fullscreenchange MSFullscreenChange';

var Default$8 = {

minimizeIcon: 'fa-compress-arrows-alt',

maximizeIcon: 'fa-expand-arrows-alt'

};

/\*\*

\* Class Definition

\* ====================================================

\*/

var Fullscreen = /\*#\_\_PURE\_\_\*/function () {

function Fullscreen(\_element, \_options) {

this.element = \_element;

this.options = $\_\_default["default"].extend({}, Default$8, \_options);

} // Public

var \_proto = Fullscreen.prototype;

\_proto.toggle = function toggle() {

if (document.fullscreenElement || document.mozFullScreenElement || document.webkitFullscreenElement || document.msFullscreenElement) {

this.windowed();

} else {

this.fullscreen();

}

};

\_proto.toggleIcon = function toggleIcon() {

if (document.fullscreenElement || document.mozFullScreenElement || document.webkitFullscreenElement || document.msFullscreenElement) {

$\_\_default["default"](SELECTOR\_ICON).removeClass(this.options.maximizeIcon).addClass(this.options.minimizeIcon);

} else {

$\_\_default["default"](SELECTOR\_ICON).removeClass(this.options.minimizeIcon).addClass(this.options.maximizeIcon);

}

};

\_proto.fullscreen = function fullscreen() {

if (document.documentElement.requestFullscreen) {

document.documentElement.requestFullscreen();

} else if (document.documentElement.webkitRequestFullscreen) {

document.documentElement.webkitRequestFullscreen();

} else if (document.documentElement.msRequestFullscreen) {

document.documentElement.msRequestFullscreen();

}

};

\_proto.windowed = function windowed() {

if (document.exitFullscreen) {

document.exitFullscreen();

} else if (document.webkitExitFullscreen) {

document.webkitExitFullscreen();

} else if (document.msExitFullscreen) {

document.msExitFullscreen();

}

} // Static

;

Fullscreen.\_jQueryInterface = function \_jQueryInterface(config) {

var data = $\_\_default["default"](this).data(DATA\_KEY$8);

if (!data) {

data = $\_\_default["default"](this).data();

}

var \_options = $\_\_default["default"].extend({}, Default$8, typeof config === 'object' ? config : data);

var plugin = new Fullscreen($\_\_default["default"](this), \_options);

$\_\_default["default"](this).data(DATA\_KEY$8, typeof config === 'object' ? config : data);

if (typeof config === 'string' && /toggle|toggleIcon|fullscreen|windowed/.test(config)) {

plugin[config]();

} else {

plugin.init();

}

};

return Fullscreen;

}();

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_WIDGET$2, function () {

Fullscreen.\_jQueryInterface.call($\_\_default["default"](this), 'toggle');

});

$\_\_default["default"](document).on(EVENT\_FULLSCREEN\_CHANGE, function () {

Fullscreen.\_jQueryInterface.call($\_\_default["default"](SELECTOR\_DATA\_WIDGET$2), 'toggleIcon');

});

/\*\*

\* jQuery API

\* ====================================================

\*/

$\_\_default["default"].fn[NAME$8] = Fullscreen.\_jQueryInterface;

$\_\_default["default"].fn[NAME$8].Constructor = Fullscreen;

$\_\_default["default"].fn[NAME$8].noConflict = function () {

$\_\_default["default"].fn[NAME$8] = JQUERY\_NO\_CONFLICT$8;

return Fullscreen.\_jQueryInterface;

};

/\*\*

\* --------------------------------------------

\* AdminLTE IFrame.js

\* License MIT

\* --------------------------------------------

\*/

/\*\*

\* Constants

\* ====================================================

\*/

var NAME$7 = 'IFrame';

var DATA\_KEY$7 = 'lte.iframe';

var JQUERY\_NO\_CONFLICT$7 = $\_\_default["default"].fn[NAME$7];

var SELECTOR\_DATA\_TOGGLE$1 = '[data-widget="iframe"]';

var SELECTOR\_DATA\_TOGGLE\_CLOSE = '[data-widget="iframe-close"]';

var SELECTOR\_DATA\_TOGGLE\_SCROLL\_LEFT = '[data-widget="iframe-scrollleft"]';

var SELECTOR\_DATA\_TOGGLE\_SCROLL\_RIGHT = '[data-widget="iframe-scrollright"]';

var SELECTOR\_DATA\_TOGGLE\_FULLSCREEN = '[data-widget="iframe-fullscreen"]';

var SELECTOR\_CONTENT\_WRAPPER = '.content-wrapper';

var SELECTOR\_CONTENT\_IFRAME = SELECTOR\_CONTENT\_WRAPPER + " iframe";

var SELECTOR\_TAB\_NAV = SELECTOR\_CONTENT\_WRAPPER + ".iframe-mode .nav";

var SELECTOR\_TAB\_NAVBAR\_NAV = SELECTOR\_CONTENT\_WRAPPER + ".iframe-mode .navbar-nav";

var SELECTOR\_TAB\_NAVBAR\_NAV\_ITEM = SELECTOR\_TAB\_NAVBAR\_NAV + " .nav-item";

var SELECTOR\_TAB\_NAVBAR\_NAV\_LINK = SELECTOR\_TAB\_NAVBAR\_NAV + " .nav-link";

var SELECTOR\_TAB\_CONTENT = SELECTOR\_CONTENT\_WRAPPER + ".iframe-mode .tab-content";

var SELECTOR\_TAB\_EMPTY = SELECTOR\_TAB\_CONTENT + " .tab-empty";

var SELECTOR\_TAB\_LOADING = SELECTOR\_TAB\_CONTENT + " .tab-loading";

var SELECTOR\_TAB\_PANE = SELECTOR\_TAB\_CONTENT + " .tab-pane";

var SELECTOR\_SIDEBAR\_MENU\_ITEM = '.main-sidebar .nav-item > a.nav-link';

var SELECTOR\_SIDEBAR\_SEARCH\_ITEM = '.sidebar-search-results .list-group-item';

var SELECTOR\_HEADER\_MENU\_ITEM = '.main-header .nav-item a.nav-link';

var SELECTOR\_HEADER\_DROPDOWN\_ITEM = '.main-header a.dropdown-item';

var CLASS\_NAME\_IFRAME\_MODE$1 = 'iframe-mode';

var CLASS\_NAME\_FULLSCREEN\_MODE = 'iframe-mode-fullscreen';

var Default$7 = {

onTabClick: function onTabClick(item) {

return item;

},

onTabChanged: function onTabChanged(item) {

return item;

},

onTabCreated: function onTabCreated(item) {

return item;

},

autoIframeMode: true,

autoItemActive: true,

autoShowNewTab: true,

autoDarkMode: false,

allowDuplicates: false,

allowReload: true,

loadingScreen: true,

useNavbarItems: true,

scrollOffset: 40,

scrollBehaviorSwap: false,

iconMaximize: 'fa-expand',

iconMinimize: 'fa-compress'

};

/\*\*

\* Class Definition

\* ====================================================

\*/

var IFrame = /\*#\_\_PURE\_\_\*/function () {

function IFrame(element, config) {

this.\_config = config;

this.\_element = element;

this.\_init();

} // Public

var \_proto = IFrame.prototype;

\_proto.onTabClick = function onTabClick(item) {

this.\_config.onTabClick(item);

};

\_proto.onTabChanged = function onTabChanged(item) {

this.\_config.onTabChanged(item);

};

\_proto.onTabCreated = function onTabCreated(item) {

this.\_config.onTabCreated(item);

};

\_proto.createTab = function createTab(title, link, uniqueName, autoOpen) {

var \_this = this;

var tabId = "panel-" + uniqueName;

var navId = "tab-" + uniqueName;

if (this.\_config.allowDuplicates) {

tabId += "-" + Math.floor(Math.random() \* 1000);

navId += "-" + Math.floor(Math.random() \* 1000);

}

var newNavItem = "<li class=\"nav-item\" role=\"presentation\"><a href=\"#\" class=\"btn-iframe-close\" data-widget=\"iframe-close\" data-type=\"only-this\"><i class=\"fas fa-times\"></i></a><a class=\"nav-link\" data-toggle=\"row\" id=\"" + navId + "\" href=\"#" + tabId + "\" role=\"tab\" aria-controls=\"" + tabId + "\" aria-selected=\"false\">" + title + "</a></li>";

$\_\_default["default"](SELECTOR\_TAB\_NAVBAR\_NAV).append(unescape(escape(newNavItem)));

var newTabItem = "<div class=\"tab-pane fade\" id=\"" + tabId + "\" role=\"tabpanel\" aria-labelledby=\"" + navId + "\"><iframe src=\"" + link + "\"></iframe></div>";

$\_\_default["default"](SELECTOR\_TAB\_CONTENT).append(unescape(escape(newTabItem)));

if (autoOpen) {

if (this.\_config.loadingScreen) {

var $loadingScreen = $\_\_default["default"](SELECTOR\_TAB\_LOADING);

$loadingScreen.fadeIn();

$\_\_default["default"](tabId + " iframe").ready(function () {

if (typeof \_this.\_config.loadingScreen === 'number') {

\_this.switchTab("#" + navId);

setTimeout(function () {

$loadingScreen.fadeOut();

}, \_this.\_config.loadingScreen);

} else {

\_this.switchTab("#" + navId);

$loadingScreen.fadeOut();

}

});

} else {

this.switchTab("#" + navId);

}

}

this.onTabCreated($\_\_default["default"]("#" + navId));

};

\_proto.openTabSidebar = function openTabSidebar(item, autoOpen) {

if (autoOpen === void 0) {

autoOpen = this.\_config.autoShowNewTab;

}

var $item = $\_\_default["default"](item).clone();

if ($item.attr('href') === undefined) {

$item = $\_\_default["default"](item).parent('a').clone();

}

$item.find('.right, .search-path').remove();

var title = $item.find('p').text();

if (title === '') {

title = $item.text();

}

var link = $item.attr('href');

if (link === '#' || link === '' || link === undefined) {

return;

}

var uniqueName = unescape(link).replace('./', '').replace(/["#&'./:=?[\]]/gi, '-').replace(/(--)/gi, '');

var navId = "tab-" + uniqueName;

if (!this.\_config.allowDuplicates && $\_\_default["default"]("#" + navId).length > 0) {

return this.switchTab("#" + navId, this.\_config.allowReload);

}

if (!this.\_config.allowDuplicates && $\_\_default["default"]("#" + navId).length === 0 || this.\_config.allowDuplicates) {

this.createTab(title, link, uniqueName, autoOpen);

}

};

\_proto.switchTab = function switchTab(item, reload) {

var \_this2 = this;

if (reload === void 0) {

reload = false;

}

var $item = $\_\_default["default"](item);

var tabId = $item.attr('href');

$\_\_default["default"](SELECTOR\_TAB\_EMPTY).hide();

if (reload) {

var $loadingScreen = $\_\_default["default"](SELECTOR\_TAB\_LOADING);

if (this.\_config.loadingScreen) {

$loadingScreen.show(0, function () {

$\_\_default["default"](tabId + " iframe").attr('src', $\_\_default["default"](tabId + " iframe").attr('src')).ready(function () {

if (\_this2.\_config.loadingScreen) {

if (typeof \_this2.\_config.loadingScreen === 'number') {

setTimeout(function () {

$loadingScreen.fadeOut();

}, \_this2.\_config.loadingScreen);

} else {

$loadingScreen.fadeOut();

}

}

});

});

} else {

$\_\_default["default"](tabId + " iframe").attr('src', $\_\_default["default"](tabId + " iframe").attr('src'));

}

}

$\_\_default["default"](SELECTOR\_TAB\_NAVBAR\_NAV + " .active").tab('dispose').removeClass('active');

this.\_fixHeight();

$item.tab('show');

$item.parents('li').addClass('active');

this.onTabChanged($item);

if (this.\_config.autoItemActive) {

this.\_setItemActive($\_\_default["default"](tabId + " iframe").attr('src'));

}

};

\_proto.removeActiveTab = function removeActiveTab(type, element) {

if (type == 'all') {

$\_\_default["default"](SELECTOR\_TAB\_NAVBAR\_NAV\_ITEM).remove();

$\_\_default["default"](SELECTOR\_TAB\_PANE).remove();

$\_\_default["default"](SELECTOR\_TAB\_EMPTY).show();

} else if (type == 'all-other') {

$\_\_default["default"](SELECTOR\_TAB\_NAVBAR\_NAV\_ITEM + ":not(.active)").remove();

$\_\_default["default"](SELECTOR\_TAB\_PANE + ":not(.active)").remove();

} else if (type == 'only-this') {

var $navClose = $\_\_default["default"](element);

var $navItem = $navClose.parent('.nav-item');

var $navItemParent = $navItem.parent();

var navItemIndex = $navItem.index();

var tabId = $navClose.siblings('.nav-link').attr('aria-controls');

$navItem.remove();

$\_\_default["default"]("#" + tabId).remove();

if ($\_\_default["default"](SELECTOR\_TAB\_CONTENT).children().length == $\_\_default["default"](SELECTOR\_TAB\_EMPTY + ", " + SELECTOR\_TAB\_LOADING).length) {

$\_\_default["default"](SELECTOR\_TAB\_EMPTY).show();

} else {

var prevNavItemIndex = navItemIndex - 1;

this.switchTab($navItemParent.children().eq(prevNavItemIndex).find('a.nav-link'));

}

} else {

var \_$navItem = $\_\_default["default"](SELECTOR\_TAB\_NAVBAR\_NAV\_ITEM + ".active");

var \_$navItemParent = \_$navItem.parent();

var \_navItemIndex = \_$navItem.index();

\_$navItem.remove();

$\_\_default["default"](SELECTOR\_TAB\_PANE + ".active").remove();

if ($\_\_default["default"](SELECTOR\_TAB\_CONTENT).children().length == $\_\_default["default"](SELECTOR\_TAB\_EMPTY + ", " + SELECTOR\_TAB\_LOADING).length) {

$\_\_default["default"](SELECTOR\_TAB\_EMPTY).show();

} else {

var \_prevNavItemIndex = \_navItemIndex - 1;

this.switchTab(\_$navItemParent.children().eq(\_prevNavItemIndex).find('a.nav-link'));

}

}

};

\_proto.toggleFullscreen = function toggleFullscreen() {

if ($\_\_default["default"]('body').hasClass(CLASS\_NAME\_FULLSCREEN\_MODE)) {

$\_\_default["default"](SELECTOR\_DATA\_TOGGLE\_FULLSCREEN + " i").removeClass(this.\_config.iconMinimize).addClass(this.\_config.iconMaximize);

$\_\_default["default"]('body').removeClass(CLASS\_NAME\_FULLSCREEN\_MODE);

$\_\_default["default"](SELECTOR\_TAB\_EMPTY + ", " + SELECTOR\_TAB\_LOADING).height('100%');

$\_\_default["default"](SELECTOR\_CONTENT\_WRAPPER).height('100%');

$\_\_default["default"](SELECTOR\_CONTENT\_IFRAME).height('100%');

} else {

$\_\_default["default"](SELECTOR\_DATA\_TOGGLE\_FULLSCREEN + " i").removeClass(this.\_config.iconMaximize).addClass(this.\_config.iconMinimize);

$\_\_default["default"]('body').addClass(CLASS\_NAME\_FULLSCREEN\_MODE);

}

$\_\_default["default"](window).trigger('resize');

this.\_fixHeight(true);

} // Private

;

\_proto.\_init = function \_init() {

var usingDefTab = $\_\_default["default"](SELECTOR\_TAB\_CONTENT).children().length > 2;

this.\_setupListeners();

this.\_fixHeight(true);

if (usingDefTab) {

var $el = $\_\_default["default"]("" + SELECTOR\_TAB\_PANE).first(); // eslint-disable-next-line no-console

console.log($el);

var uniqueName = $el.attr('id').replace('panel-', '');

var navId = "#tab-" + uniqueName;

this.switchTab(navId, true);

}

};

\_proto.\_initFrameElement = function \_initFrameElement() {

if (window.frameElement && this.\_config.autoIframeMode) {

var $body = $\_\_default["default"]('body');

$body.addClass(CLASS\_NAME\_IFRAME\_MODE$1);

if (this.\_config.autoDarkMode) {

$body.addClass('dark-mode');

}

}

};

\_proto.\_navScroll = function \_navScroll(offset) {

var leftPos = $\_\_default["default"](SELECTOR\_TAB\_NAVBAR\_NAV).scrollLeft();

$\_\_default["default"](SELECTOR\_TAB\_NAVBAR\_NAV).animate({

scrollLeft: leftPos + offset

}, 250, 'linear');

};

\_proto.\_setupListeners = function \_setupListeners() {

var \_this3 = this;

$\_\_default["default"](window).on('resize', function () {

setTimeout(function () {

\_this3.\_fixHeight();

}, 1);

});

if ($\_\_default["default"](SELECTOR\_CONTENT\_WRAPPER).hasClass(CLASS\_NAME\_IFRAME\_MODE$1)) {

$\_\_default["default"](document).on('click', SELECTOR\_SIDEBAR\_MENU\_ITEM + ", " + SELECTOR\_SIDEBAR\_SEARCH\_ITEM, function (e) {

e.preventDefault();

\_this3.openTabSidebar(e.target);

});

if (this.\_config.useNavbarItems) {

$\_\_default["default"](document).on('click', SELECTOR\_HEADER\_MENU\_ITEM + ", " + SELECTOR\_HEADER\_DROPDOWN\_ITEM, function (e) {

e.preventDefault();

\_this3.openTabSidebar(e.target);

});

}

}

$\_\_default["default"](document).on('click', SELECTOR\_TAB\_NAVBAR\_NAV\_LINK, function (e) {

e.preventDefault();

\_this3.onTabClick(e.target);

\_this3.switchTab(e.target);

});

$\_\_default["default"](document).on('click', SELECTOR\_TAB\_NAVBAR\_NAV\_LINK, function (e) {

e.preventDefault();

\_this3.onTabClick(e.target);

\_this3.switchTab(e.target);

});

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_TOGGLE\_CLOSE, function (e) {

e.preventDefault();

var target = e.target;

if (target.nodeName == 'I') {

target = e.target.offsetParent;

}

\_this3.removeActiveTab(target.attributes['data-type'] ? target.attributes['data-type'].nodeValue : null, target);

});

$\_\_default["default"](document).on('click', SELECTOR\_DATA\_TOGGLE\_FULLSCREEN, function (e) {

e.preventDefault();

\_this3.toggleFullscreen();

});

var mousedown = false;

var mousedownInterval = null;

$\_\_default["default"](document).on('mousedown', SELECTOR\_DATA\_TOGGLE\_SCROLL\_LEFT, function (e) {

e.preventDefault();

clearInterval(mousedownInterval);

var scrollOffset = \_this3.\_config.scrollOffset;

if (!\_this3.\_config.scrollBehaviorSwap) {

scrollOffset = -scrollOffset;

}

mousedown = true;

\_this3.\_navScroll(scrollOffset);

mousedownInterval = setInterval(function () {

\_this3.\_navScroll(scrollOffset);

}, 250);

});

$\_\_default["default"](document).on('mousedown', SELECTOR\_DATA\_TOGGLE\_SCROLL\_RIGHT, function (e) {

e.preventDefault();

clearInterval(mousedownInterval);

var scrollOffset = \_this3.\_config.scrollOffset;

if (\_this3.\_config.scrollBehaviorSwap) {

scrollOffset = -scrollOffset;

}

mousedown = true;

\_this3.\_navScroll(scrollOffset);

mousedownInterval = setInterval(function () {

\_this3.\_navScroll(scrollOffset);

}, 250);

});

$\_\_default["default"](document).on('mouseup', function () {

if (mousedown) {

mousedown = false;

clearInterval(mousedownInterval);

mousedownInterval = null;

}

});

};

\_proto.\_setItemActive = function \_setItemActive(href) {

$\_\_default["default"](SELECTOR\_SIDEBAR\_MENU\_ITEM + ", " + SELECTOR\_HEADER\_DROPDOWN\_ITEM).removeClass('active');

$\_\_default["default"](SELECTOR\_HEADER\_MENU\_ITEM).parent().removeClass('active');

var $headerMenuItem = $\_\_default["default"](SELECTOR\_HEADER\_MENU\_ITEM + "[href$=\"" + href + "\"]");

var $headerDropdownItem = $\_\_default["default"](SELECTOR\_HEADER\_DROPDOWN\_ITEM + "[href$=\"" + href + "\"]");

var $sidebarMenuItem = $\_\_default["default"](SELECTOR\_SIDEBAR\_MENU\_ITEM + "[href$=\"" + href + "\"]");

$headerMenuItem.each(function (i, e) {

$\_\_default["default"](e).parent().addClass('active');

});

$headerDropdownItem.each(function (i, e) {

$\_\_default["default"](e).addClass('active');

});

$sidebarMenuItem.each(function (i, e) {

$\_\_default["default"](e).addClass('active');

$\_\_default["default"](e).parents('.nav-treeview').prevAll('.nav-link').addClass('active');

});

};

\_proto.\_fixHeight = function \_fixHeight(tabEmpty) {

if (tabEmpty === void 0) {

tabEmpty = false;

}

if ($\_\_default["default"]('body').hasClass(CLASS\_NAME\_FULLSCREEN\_MODE)) {

var windowHeight = $\_\_default["default"](window).height();

var navbarHeight = $\_\_default["default"](SELECTOR\_TAB\_NAV).outerHeight();

$\_\_default["default"](SELECTOR\_TAB\_EMPTY + ", " + SELECTOR\_TAB\_LOADING + ", " + SELECTOR\_CONTENT\_IFRAME).height(windowHeight - navbarHeight);

$\_\_default["default"](SELECTOR\_CONTENT\_WRAPPER).height(windowHeight);

} else {

var contentWrapperHeight = parseFloat($\_\_default["default"](SELECTOR\_CONTENT\_WRAPPER).css('height'));

var \_navbarHeight = $\_\_default["default"](SELECTOR\_TAB\_NAV).outerHeight();

if (tabEmpty == true) {

setTimeout(function () {

$\_\_default["default"](SELECTOR\_TAB\_EMPTY + ", " + SELECTOR\_TAB\_LOADING).height(contentWrapperHeight - \_navbarHeight);

}, 50);

} else {

$\_\_default["default"](SELECTOR\_CONTENT\_IFRAME).height(contentWrapperHeight - \_navbarHeight);

}

}

} // Static

;

IFrame.\_jQueryInterface = function \_jQueryInterface(config) {

if ($\_\_default["default"](SELECTOR\_DATA\_TOGGLE$1).length > 0) {

var data = $\_\_default["default"](this).data(DATA\_KEY$7);

if (!data) {

data = $\_\_default["default"](this).data();

}

var \_options = $\_\_default["default"].extend({}, Default$7, typeof config === 'object' ? config : data);

localStorage.setItem('AdminLTE:IFrame:Options', JSON.stringify(\_options));

var plugin = new IFrame($\_\_default["default"](this), \_options);

$\_\_default["default"](this).data(DATA\_KEY$7, typeof config === 'object' ? config : data);

if (typeof config === 'string' && /createTab|openTabSidebar|switchTab|removeActiveTab/.test(config)) {

plugin[config]();

}

} else {

new IFrame($\_\_default["default"](this), JSON.parse(localStorage.getItem('AdminLTE:IFrame:Options'))).\_initFrameElement();

}

};

return IFrame;

}();

$\_\_default["default"](window).on('load', function () {

IFrame.\_jQueryInterface.call($\_\_default["default"](SELECTOR\_DATA\_TOGGLE$1));

});

$\_\_default["default"].fn[NAME$7] = IFrame.\_jQueryInterface;

$\_\_default["default"].fn[NAME$7].Constructor = IFrame;

$\_\_default["default"].fn[NAME$7].noConflict = function () {

$\_\_default["default"].fn[NAME$7] = JQUERY\_NO\_CONFLICT$7;

return IFrame.\_jQueryInterface;

};

var NAME$6 = 'Layout';

var DATA\_KEY$6 = 'lte.layout';

var JQUERY\_NO\_CONFLICT$6 = $\_\_default["default"].fn[NAME$6];

var SELECTOR\_HEADER = '.main-header';

var SELECTOR\_MAIN\_SIDEBAR = '.main-sidebar';

var SELECTOR\_SIDEBAR$1 = '.main-sidebar .sidebar';

var SELECTOR\_CONTENT = '.content-wrapper';

var SELECTOR\_CONTROL\_SIDEBAR\_CONTENT = '.control-sidebar-content';

var SELECTOR\_CONTROL\_SIDEBAR\_BTN = '[data-widget="control-sidebar"]';

var SELECTOR\_FOOTER = '.main-footer';

var SELECTOR\_PUSHMENU\_BTN = '[data-widget="pushmenu"]';

var SELECTOR\_LOGIN\_BOX = '.login-box';

var SELECTOR\_REGISTER\_BOX = '.register-box';

var SELECTOR\_PRELOADER = '.preloader';

var CLASS\_NAME\_SIDEBAR\_COLLAPSED$1 = 'sidebar-collapse';

var CLASS\_NAME\_SIDEBAR\_FOCUSED = 'sidebar-focused';

var CLASS\_NAME\_LAYOUT\_FIXED = 'layout-fixed';

var CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE\_OPEN = 'control-sidebar-slide-open';

var CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN = 'control-sidebar-open';

var CLASS\_NAME\_IFRAME\_MODE = 'iframe-mode';

var Default$6 = {

scrollbarTheme: 'os-theme-light',

scrollbarAutoHide: 'l',

panelAutoHeight: true,

panelAutoHeightMode: 'min-height',

preloadDuration: 200,

loginRegisterAutoHeight: true

};

var Layout = /\*#\_\_PURE\_\_\*/function () {

function Layout(element, config) {

this.\_config = config;

this.\_element = element;

} // Public

var \_proto = Layout.prototype;

\_proto.fixLayoutHeight = function fixLayoutHeight(extra) {

if (extra === void 0) {

extra = null;

}

var $body = $\_\_default["default"]('body');

var controlSidebar = 0;

if ($body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_SLIDE\_OPEN) || $body.hasClass(CLASS\_NAME\_CONTROL\_SIDEBAR\_OPEN) || extra === 'control\_sidebar') {

controlSidebar = $\_\_default["default"](SELECTOR\_CONTROL\_SIDEBAR\_CONTENT).outerHeight();

}

var heights = {

window: $\_\_default["default"](window).height(),

header: $\_\_default["default"](SELECTOR\_HEADER).length > 0 ? $\_\_default["default"](SELECTOR\_HEADER).outerHeight() : 0,

footer: $\_\_default["default"](SELECTOR\_FOOTER).length > 0 ? $\_\_default["default"](SELECTOR\_FOOTER).outerHeight() : 0,

sidebar: $\_\_default["default"](SELECTOR\_SIDEBAR$1).length > 0 ? $\_\_default["default"](SELECTOR\_SIDEBAR$1).height() : 0,

controlSidebar: controlSidebar

};

var max = this.\_max(heights);

var offset = this.\_config.panelAutoHeight;

if (offset === true) {

offset = 0;

}

var $contentSelector = $\_\_default["default"](SELECTOR\_CONTENT);

if (offset !== false) {

if (max === heights.controlSidebar) {

$contentSelector.css(this.\_config.panelAutoHeightMode, max + offset);

} else if (max === heights.window) {

$contentSelector.css(this.\_config.panelAutoHeightMode, max + offset - heights.header - heights.footer);

} else {

$contentSelector.css(this.\_config.panelAutoHeightMode, max + offset - heights.header);

}

if (this.\_isFooterFixed()) {

$contentSelector.css(this.\_config.panelAutoHeightMode, parseFloat($contentSelector.css(this.\_config.panelAutoHeightMode)) + heights.footer);

}

}

if (!$body.hasClass(CLASS\_NAME\_LAYOUT\_FIXED)) {

return;

}

if (typeof $\_\_default["default"].fn.overlayScrollbars !== 'undefined') {

$\_\_default["default"](SELECTOR\_SIDEBAR$1).overlayScrollbars({

className: this.\_config.scrollbarTheme,

sizeAutoCapable: true,

scrollbars: {

autoHide: this.\_config.scrollbarAutoHide,

clickScrolling: true

}

});

} else {

$\_\_default["default"](SELECTOR\_SIDEBAR$1).css('overflow-y', 'auto');

}

};

\_proto.fixLoginRegisterHeight = function fixLoginRegisterHeight() {

var $body = $\_\_default["default"]('body');

var $selector = $\_\_default["default"](SELECTOR\_LOGIN\_BOX + ", " + SELECTOR\_REGISTER\_BOX);

if ($body.hasClass(CLASS\_NAME\_IFRAME\_MODE)) {

$body.css('height', '100%');

$\_\_default["default"]('.wrapper').css('height', '100%');

$\_\_default["default"]('html').css('height', '100%');

} else if ($selector.length === 0) {

$body.css('height', 'auto');

$\_\_default["default"]('html').css('height', 'auto');

} else {

var boxHeight = $selector.height();

if ($body.css(this.\_config.panelAutoHeightMode) !== boxHeight) {

$body.css(this.\_config.panelAutoHeightMode, boxHeight);

}

}

} // Private

;

\_proto.\_init = function \_init() {

var \_this = this;

// Activate layout height watcher

this.fixLayoutHeight();

if (this.\_config.loginRegisterAutoHeight === true) {

this.fixLoginRegisterHeight();

} else if (this.\_config.loginRegisterAutoHeight === parseInt(this.\_config.loginRegisterAutoHeight, 10)) {

setInterval(this.fixLoginRegisterHeight, this.\_config.loginRegisterAutoHeight);

}

$\_\_default["default"](SELECTOR\_SIDEBAR$1).on('collapsed.lte.treeview expanded.lte.treeview', function () {

\_this.fixLayoutHeight();

});

$\_\_default["default"](SELECTOR\_MAIN\_SIDEBAR).on('mouseenter mouseleave', function () {

if ($\_\_default["default"]('body').hasClass(CLASS\_NAME\_SIDEBAR\_COLLAPSED$1)) {

\_this.fixLayoutHeight();

}

});

$\_\_default["default"](SELECTOR\_PUSHMENU\_BTN).on('collapsed.lte.pushmenu shown.lte.pushmenu', function () {

setTimeout(function () {

\_this.fixLayoutHeight();

}, 300);

});

$\_\_default["default"](SELECTOR\_CONTROL\_SIDEBAR\_BTN).on('collapsed.lte.controlsidebar', function () {

\_this.fixLayoutHeight();

}).on('expanded.lte.controlsidebar', function () {

\_this.fixLayoutHeight('control\_sidebar');

});

$\_\_default["default"](window).resize(function () {

\_this.fixLayoutHeight();

});

setTimeout(function () {

$\_\_default["default"]('body.hold-transition').removeClass('hold-transition');

}, 50);

setTimeout(function () {

var $preloader = $\_\_default["default"](SELECTOR\_PRELOADER);

if ($preloader) {

$preloader.css('height', 0);

setTimeout(function () {

$preloader.children().hide();

}, 200);

}

}, this.\_config.preloadDuration);

};

\_proto.\_max = function \_max(numbers) {

// Calculate the maximum number in a list

var max = 0;

Object.keys(numbers).forEach(function (key) {

if (numbers[key] > max) {

max = numbers[key];

}

});

return max;

};

\_proto.\_isFooterFixed = function \_isFooterFixed() {

return $\_\_default["default"](SELECTOR\_FOOTER).css('position') === 'fixed';

} // Static

;

Layout.\_jQueryInterface = function \_jQueryInterface(config) {

if (config === void 0) {

config = '';

}

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY$6);

var \_options = $\_\_default["default"].extend({}, Default$6, $\_\_default["default"](this).data());

if (!data) {

data = new Layout($\_\_default["default"](this), \_options);

$\_\_default["default"](this).data(DATA\_KEY$6, data);

}

if (config === 'init' || config === '') {

data.\_init();

} else if (config === 'fixLayoutHeight' || config === 'fixLoginRegisterHeight') {

data[config]();

}

});

};

return Layout;

}();

$\_\_default["default"](window).on('load', function () {

Layout.\_jQueryInterface.call($\_\_default["default"]('body'));

});

$\_\_default["default"](SELECTOR\_SIDEBAR$1 + " a").on('focusin', function () {

$\_\_default["default"](SELECTOR\_MAIN\_SIDEBAR).addClass(CLASS\_NAME\_SIDEBAR\_FOCUSED);

}).on('focusout', function () {

$\_\_default["default"](SELECTOR\_MAIN\_SIDEBAR).removeClass(CLASS\_NAME\_SIDEBAR\_FOCUSED);

});

$\_\_default["default"].fn[NAME$6] = Layout.\_jQueryInterface;

$\_\_default["default"].fn[NAME$6].Constructor = Layout;

$\_\_default["default"].fn[NAME$6].noConflict = function () {

$\_\_default["default"].fn[NAME$6] = JQUERY\_NO\_CONFLICT$6;

return Layout.\_jQueryInterface;

};

var NAME$5 = 'PushMenu';

var DATA\_KEY$5 = 'lte.pushmenu';

var EVENT\_KEY$2 = "." + DATA\_KEY$5;

var JQUERY\_NO\_CONFLICT$5 = $\_\_default["default"].fn[NAME$5];

var EVENT\_COLLAPSED$1 = "collapsed" + EVENT\_KEY$2;

var EVENT\_COLLAPSED\_DONE = "collapsed-done" + EVENT\_KEY$2;

var EVENT\_SHOWN = "shown" + EVENT\_KEY$2;

var SELECTOR\_TOGGLE\_BUTTON$1 = '[data-widget="pushmenu"]';

var SELECTOR\_BODY = 'body';

var SELECTOR\_OVERLAY = '#sidebar-overlay';

var SELECTOR\_WRAPPER = '.wrapper';

var CLASS\_NAME\_COLLAPSED = 'sidebar-collapse';

var CLASS\_NAME\_OPEN$3 = 'sidebar-open';

var CLASS\_NAME\_IS\_OPENING$1 = 'sidebar-is-opening';

var CLASS\_NAME\_CLOSED = 'sidebar-closed';

var Default$5 = {

autoCollapseSize: 992,

enableRemember: false,

noTransitionAfterReload: true,

animationSpeed: 300

};

/\*\*

\* Class Definition

\* ====================================================

\*/

var PushMenu = /\*#\_\_PURE\_\_\*/function () {

function PushMenu(element, options) {

this.\_element = element;

this.\_options = $\_\_default["default"].extend({}, Default$5, options);

if ($\_\_default["default"](SELECTOR\_OVERLAY).length === 0) {

this.\_addOverlay();

}

this.\_init();

} // Public

var \_proto = PushMenu.prototype;

\_proto.expand = function expand() {

var $bodySelector = $\_\_default["default"](SELECTOR\_BODY);

if (this.\_options.autoCollapseSize && $\_\_default["default"](window).width() <= this.\_options.autoCollapseSize) {

$bodySelector.addClass(CLASS\_NAME\_OPEN$3);

}

$bodySelector.addClass(CLASS\_NAME\_IS\_OPENING$1).removeClass(CLASS\_NAME\_COLLAPSED + " " + CLASS\_NAME\_CLOSED).delay(50).queue(function () {

$bodySelector.removeClass(CLASS\_NAME\_IS\_OPENING$1);

$\_\_default["default"](this).dequeue();

});

if (this.\_options.enableRemember) {

localStorage.setItem("remember" + EVENT\_KEY$2, CLASS\_NAME\_OPEN$3);

}

$\_\_default["default"](this.\_element).trigger($\_\_default["default"].Event(EVENT\_SHOWN));

};

\_proto.collapse = function collapse() {

var \_this = this;

var $bodySelector = $\_\_default["default"](SELECTOR\_BODY);

if (this.\_options.autoCollapseSize && $\_\_default["default"](window).width() <= this.\_options.autoCollapseSize) {

$bodySelector.removeClass(CLASS\_NAME\_OPEN$3).addClass(CLASS\_NAME\_CLOSED);

}

$bodySelector.addClass(CLASS\_NAME\_COLLAPSED);

if (this.\_options.enableRemember) {

localStorage.setItem("remember" + EVENT\_KEY$2, CLASS\_NAME\_COLLAPSED);

}

$\_\_default["default"](this.\_element).trigger($\_\_default["default"].Event(EVENT\_COLLAPSED$1));

setTimeout(function () {

$\_\_default["default"](\_this.\_element).trigger($\_\_default["default"].Event(EVENT\_COLLAPSED\_DONE));

}, this.\_options.animationSpeed);

};

\_proto.toggle = function toggle() {

if ($\_\_default["default"](SELECTOR\_BODY).hasClass(CLASS\_NAME\_COLLAPSED)) {

this.expand();

} else {

this.collapse();

}

};

\_proto.autoCollapse = function autoCollapse(resize) {

if (resize === void 0) {

resize = false;

}

if (!this.\_options.autoCollapseSize) {

return;

}

var $bodySelector = $\_\_default["default"](SELECTOR\_BODY);

if ($\_\_default["default"](window).width() <= this.\_options.autoCollapseSize) {

if (!$bodySelector.hasClass(CLASS\_NAME\_OPEN$3)) {

this.collapse();

}

} else if (resize === true) {

if ($bodySelector.hasClass(CLASS\_NAME\_OPEN$3)) {

$bodySelector.removeClass(CLASS\_NAME\_OPEN$3);

} else if ($bodySelector.hasClass(CLASS\_NAME\_CLOSED)) {

this.expand();

}

}

};

\_proto.remember = function remember() {

if (!this.\_options.enableRemember) {

return;

}

var $body = $\_\_default["default"]('body');

var toggleState = localStorage.getItem("remember" + EVENT\_KEY$2);

if (toggleState === CLASS\_NAME\_COLLAPSED) {

if (this.\_options.noTransitionAfterReload) {

$body.addClass('hold-transition').addClass(CLASS\_NAME\_COLLAPSED).delay(50).queue(function () {

$\_\_default["default"](this).removeClass('hold-transition');

$\_\_default["default"](this).dequeue();

});

} else {

$body.addClass(CLASS\_NAME\_COLLAPSED);

}

} else if (this.\_options.noTransitionAfterReload) {

$body.addClass('hold-transition').removeClass(CLASS\_NAME\_COLLAPSED).delay(50).queue(function () {

$\_\_default["default"](this).removeClass('hold-transition');

$\_\_default["default"](this).dequeue();

});

} else {

$body.removeClass(CLASS\_NAME\_COLLAPSED);

}

} // Private

;

\_proto.\_init = function \_init() {

var \_this2 = this;

this.remember();

this.autoCollapse();

$\_\_default["default"](window).resize(function () {

\_this2.autoCollapse(true);

});

};

\_proto.\_addOverlay = function \_addOverlay() {

var \_this3 = this;

var overlay = $\_\_default["default"]('<div />', {

id: 'sidebar-overlay'

});

overlay.on('click', function () {

\_this3.collapse();

});

$\_\_default["default"](SELECTOR\_WRAPPER).append(overlay);

} // Static

;

PushMenu.\_jQueryInterface = function \_jQueryInterface(operation) {

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY$5);

var \_options = $\_\_default["default"].extend({}, Default$5, $\_\_default["default"](this).data());

if (!data) {

data = new PushMenu(this, \_options);

$\_\_default["default"](this).data(DATA\_KEY$5, data);

}

if (typeof operation === 'string' && /collapse|expand|toggle/.test(operation)) {

data[operation]();

}

});

};

return PushMenu;

}();

$\_\_default["default"](document).on('click', SELECTOR\_TOGGLE\_BUTTON$1, function (event) {

event.preventDefault();

var button = event.currentTarget;

if ($\_\_default["default"](button).data('widget') !== 'pushmenu') {

button = $\_\_default["default"](button).closest(SELECTOR\_TOGGLE\_BUTTON$1);

}

PushMenu.\_jQueryInterface.call($\_\_default["default"](button), 'toggle');

});

$\_\_default["default"](window).on('load', function () {

PushMenu.\_jQueryInterface.call($\_\_default["default"](SELECTOR\_TOGGLE\_BUTTON$1));

});

$\_\_default["default"].fn[NAME$5] = PushMenu.\_jQueryInterface;

$\_\_default["default"].fn[NAME$5].Constructor = PushMenu;

$\_\_default["default"].fn[NAME$5].noConflict = function () {

$\_\_default["default"].fn[NAME$5] = JQUERY\_NO\_CONFLICT$5;

return PushMenu.\_jQueryInterface;

};

var NAME$4 = 'SidebarSearch';

var DATA\_KEY$4 = 'lte.sidebar-search';

var JQUERY\_NO\_CONFLICT$4 = $\_\_default["default"].fn[NAME$4];

var CLASS\_NAME\_OPEN$2 = 'sidebar-search-open';

var CLASS\_NAME\_ICON\_SEARCH = 'fa-search';

var CLASS\_NAME\_ICON\_CLOSE = 'fa-times';

var CLASS\_NAME\_HEADER = 'nav-header';

var CLASS\_NAME\_SEARCH\_RESULTS = 'sidebar-search-results';

var CLASS\_NAME\_LIST\_GROUP = 'list-group';

var SELECTOR\_DATA\_WIDGET$1 = '[data-widget="sidebar-search"]';

var SELECTOR\_SIDEBAR = '.main-sidebar .nav-sidebar';

var SELECTOR\_NAV\_LINK = '.nav-link';

var SELECTOR\_NAV\_TREEVIEW = '.nav-treeview';

var SELECTOR\_SEARCH\_INPUT$1 = SELECTOR\_DATA\_WIDGET$1 + " .form-control";

var SELECTOR\_SEARCH\_BUTTON = SELECTOR\_DATA\_WIDGET$1 + " .btn";

var SELECTOR\_SEARCH\_ICON = SELECTOR\_SEARCH\_BUTTON + " i";

var SELECTOR\_SEARCH\_LIST\_GROUP = "." + CLASS\_NAME\_LIST\_GROUP;

var SELECTOR\_SEARCH\_RESULTS = "." + CLASS\_NAME\_SEARCH\_RESULTS;

var SELECTOR\_SEARCH\_RESULTS\_GROUP = SELECTOR\_SEARCH\_RESULTS + " ." + CLASS\_NAME\_LIST\_GROUP;

var Default$4 = {

arrowSign: '->',

minLength: 3,

maxResults: 7,

highlightName: true,

highlightPath: false,

highlightClass: 'text-light',

notFoundText: 'No element found!'

};

var SearchItems = [];

var SidebarSearch = /\*#\_\_PURE\_\_\*/function () {

function SidebarSearch(\_element, \_options) {

this.element = \_element;

this.options = $\_\_default["default"].extend({}, Default$4, \_options);

this.items = [];

} // Public

var \_proto = SidebarSearch.prototype;

\_proto.init = function init() {

var \_this = this;

if ($\_\_default["default"](SELECTOR\_DATA\_WIDGET$1).length === 0) {

return;

}

if ($\_\_default["default"](SELECTOR\_DATA\_WIDGET$1).next(SELECTOR\_SEARCH\_RESULTS).length === 0) {

$\_\_default["default"](SELECTOR\_DATA\_WIDGET$1).after($\_\_default["default"]('<div />', {

class: CLASS\_NAME\_SEARCH\_RESULTS

}));

}

if ($\_\_default["default"](SELECTOR\_SEARCH\_RESULTS).children(SELECTOR\_SEARCH\_LIST\_GROUP).length === 0) {

$\_\_default["default"](SELECTOR\_SEARCH\_RESULTS).append($\_\_default["default"]('<div />', {

class: CLASS\_NAME\_LIST\_GROUP

}));

}

this.\_addNotFound();

$\_\_default["default"](SELECTOR\_SIDEBAR).children().each(function (i, child) {

\_this.\_parseItem(child);

});

};

\_proto.search = function search() {

var \_this2 = this;

var searchValue = $\_\_default["default"](SELECTOR\_SEARCH\_INPUT$1).val().toLowerCase();

if (searchValue.length < this.options.minLength) {

$\_\_default["default"](SELECTOR\_SEARCH\_RESULTS\_GROUP).empty();

this.\_addNotFound();

this.close();

return;

}

var searchResults = SearchItems.filter(function (item) {

return item.name.toLowerCase().includes(searchValue);

});

var endResults = $\_\_default["default"](searchResults.slice(0, this.options.maxResults));

$\_\_default["default"](SELECTOR\_SEARCH\_RESULTS\_GROUP).empty();

if (endResults.length === 0) {

this.\_addNotFound();

} else {

endResults.each(function (i, result) {

$\_\_default["default"](SELECTOR\_SEARCH\_RESULTS\_GROUP).append(\_this2.\_renderItem(escape(result.name), encodeURI(result.link), result.path));

});

}

this.open();

};

\_proto.open = function open() {

$\_\_default["default"](SELECTOR\_DATA\_WIDGET$1).parent().addClass(CLASS\_NAME\_OPEN$2);

$\_\_default["default"](SELECTOR\_SEARCH\_ICON).removeClass(CLASS\_NAME\_ICON\_SEARCH).addClass(CLASS\_NAME\_ICON\_CLOSE);

};

\_proto.close = function close() {

$\_\_default["default"](SELECTOR\_DATA\_WIDGET$1).parent().removeClass(CLASS\_NAME\_OPEN$2);

$\_\_default["default"](SELECTOR\_SEARCH\_ICON).removeClass(CLASS\_NAME\_ICON\_CLOSE).addClass(CLASS\_NAME\_ICON\_SEARCH);

};

\_proto.toggle = function toggle() {

if ($\_\_default["default"](SELECTOR\_DATA\_WIDGET$1).parent().hasClass(CLASS\_NAME\_OPEN$2)) {

this.close();

} else {

this.open();

}

} // Private

;

\_proto.\_parseItem = function \_parseItem(item, path) {

var \_this3 = this;

if (path === void 0) {

path = [];

}

if ($\_\_default["default"](item).hasClass(CLASS\_NAME\_HEADER)) {

return;

}

var itemObject = {};

var navLink = $\_\_default["default"](item).clone().find("> " + SELECTOR\_NAV\_LINK);

var navTreeview = $\_\_default["default"](item).clone().find("> " + SELECTOR\_NAV\_TREEVIEW);

var link = navLink.attr('href');

var name = navLink.find('p').children().remove().end().text();

itemObject.name = this.\_trimText(name);

itemObject.link = link;

itemObject.path = path;

if (navTreeview.length === 0) {

SearchItems.push(itemObject);

} else {

var newPath = itemObject.path.concat([itemObject.name]);

navTreeview.children().each(function (i, child) {

\_this3.\_parseItem(child, newPath);

});

}

};

\_proto.\_trimText = function \_trimText(text) {

return $.trim(text.replace(/(\r\n|\n|\r)/gm, ' '));

};

\_proto.\_renderItem = function \_renderItem(name, link, path) {

var \_this4 = this;

path = path.join(" " + this.options.arrowSign + " ");

name = unescape(name);

link = decodeURI(link);

if (this.options.highlightName || this.options.highlightPath) {

var searchValue = $\_\_default["default"](SELECTOR\_SEARCH\_INPUT$1).val().toLowerCase();

var regExp = new RegExp(searchValue, 'gi');

if (this.options.highlightName) {

name = name.replace(regExp, function (str) {

return "<strong class=\"" + \_this4.options.highlightClass + "\">" + str + "</strong>";

});

}

if (this.options.highlightPath) {

path = path.replace(regExp, function (str) {

return "<strong class=\"" + \_this4.options.highlightClass + "\">" + str + "</strong>";

});

}

}

var groupItemElement = $\_\_default["default"]('<a/>', {

href: decodeURIComponent(link),

class: 'list-group-item'

});

var searchTitleElement = $\_\_default["default"]('<div/>', {

class: 'search-title'

}).html(name);

var searchPathElement = $\_\_default["default"]('<div/>', {

class: 'search-path'

}).html(path);

groupItemElement.append(searchTitleElement).append(searchPathElement);

return groupItemElement;

};

\_proto.\_addNotFound = function \_addNotFound() {

$\_\_default["default"](SELECTOR\_SEARCH\_RESULTS\_GROUP).append(this.\_renderItem(this.options.notFoundText, '#', []));

} // Static

;

SidebarSearch.\_jQueryInterface = function \_jQueryInterface(config) {

var data = $\_\_default["default"](this).data(DATA\_KEY$4);

if (!data) {

data = $\_\_default["default"](this).data();

}

var \_options = $\_\_default["default"].extend({}, Default$4, typeof config === 'object' ? config : data);

var plugin = new SidebarSearch($\_\_default["default"](this), \_options);

$\_\_default["default"](this).data(DATA\_KEY$4, typeof config === 'object' ? config : data);

if (typeof config === 'string' && /init|toggle|close|open|search/.test(config)) {

plugin[config]();

} else {

plugin.init();

}

};

return SidebarSearch;

}();

$\_\_default["default"](document).on('click', SELECTOR\_SEARCH\_BUTTON, function (event) {

event.preventDefault();

SidebarSearch.\_jQueryInterface.call($\_\_default["default"](SELECTOR\_DATA\_WIDGET$1), 'toggle');

});

$\_\_default["default"](document).on('keyup', SELECTOR\_SEARCH\_INPUT$1, function (event) {

if (event.keyCode == 38) {

event.preventDefault();

$\_\_default["default"](SELECTOR\_SEARCH\_RESULTS\_GROUP).children().last().focus();

return;

}

if (event.keyCode == 40) {

event.preventDefault();

$\_\_default["default"](SELECTOR\_SEARCH\_RESULTS\_GROUP).children().first().focus();

return;

}

setTimeout(function () {

SidebarSearch.\_jQueryInterface.call($\_\_default["default"](SELECTOR\_DATA\_WIDGET$1), 'search');

}, 100);

});

$\_\_default["default"](document).on('keydown', SELECTOR\_SEARCH\_RESULTS\_GROUP, function (event) {

var $focused = $\_\_default["default"](':focus');

if (event.keyCode == 38) {

event.preventDefault();

if ($focused.is(':first-child')) {

$focused.siblings().last().focus();

} else {

$focused.prev().focus();

}

}

if (event.keyCode == 40) {

event.preventDefault();

if ($focused.is(':last-child')) {

$focused.siblings().first().focus();

} else {

$focused.next().focus();

}

}

});

$\_\_default["default"](window).on('load', function () {

SidebarSearch.\_jQueryInterface.call($\_\_default["default"](SELECTOR\_DATA\_WIDGET$1), 'init');

});

$\_\_default["default"].fn[NAME$4] = SidebarSearch.\_jQueryInterface;

$\_\_default["default"].fn[NAME$4].Constructor = SidebarSearch;

$\_\_default["default"].fn[NAME$4].noConflict = function () {

$\_\_default["default"].fn[NAME$4] = JQUERY\_NO\_CONFLICT$4;

return SidebarSearch.\_jQueryInterface;

};

var NAME$3 = 'NavbarSearch';

var DATA\_KEY$3 = 'lte.navbar-search';

var JQUERY\_NO\_CONFLICT$3 = $\_\_default["default"].fn[NAME$3];

var SELECTOR\_TOGGLE\_BUTTON = '[data-widget="navbar-search"]';

var SELECTOR\_SEARCH\_BLOCK = '.navbar-search-block';

var SELECTOR\_SEARCH\_INPUT = '.form-control';

var CLASS\_NAME\_OPEN$1 = 'navbar-search-open';

var Default$3 = {

resetOnClose: true,

target: SELECTOR\_SEARCH\_BLOCK

};

var NavbarSearch = /\*#\_\_PURE\_\_\*/function () {

function NavbarSearch(\_element, \_options) {

this.\_element = \_element;

this.\_config = $\_\_default["default"].extend({}, Default$3, \_options);

} // Public

var \_proto = NavbarSearch.prototype;

\_proto.open = function open() {

$\_\_default["default"](this.\_config.target).css('display', 'flex').hide().fadeIn().addClass(CLASS\_NAME\_OPEN$1);

$\_\_default["default"](this.\_config.target + " " + SELECTOR\_SEARCH\_INPUT).focus();

};

\_proto.close = function close() {

$\_\_default["default"](this.\_config.target).fadeOut().removeClass(CLASS\_NAME\_OPEN$1);

if (this.\_config.resetOnClose) {

$\_\_default["default"](this.\_config.target + " " + SELECTOR\_SEARCH\_INPUT).val('');

}

};

\_proto.toggle = function toggle() {

if ($\_\_default["default"](this.\_config.target).hasClass(CLASS\_NAME\_OPEN$1)) {

this.close();

} else {

this.open();

}

} // Static

;

NavbarSearch.\_jQueryInterface = function \_jQueryInterface(options) {

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY$3);

var \_options = $\_\_default["default"].extend({}, Default$3, $\_\_default["default"](this).data());

if (!data) {

data = new NavbarSearch(this, \_options);

$\_\_default["default"](this).data(DATA\_KEY$3, data);

}

if (!/toggle|close|open/.test(options)) {

throw new Error("Undefined method " + options);

}

data[options]();

});

};

return NavbarSearch;

}();

$\_\_default["default"](document).on('click', SELECTOR\_TOGGLE\_BUTTON, function (event) {

event.preventDefault();

var button = $\_\_default["default"](event.currentTarget);

if (button.data('widget') !== 'navbar-search') {

button = button.closest(SELECTOR\_TOGGLE\_BUTTON);

}

NavbarSearch.\_jQueryInterface.call(button, 'toggle');

});

$\_\_default["default"].fn[NAME$3] = NavbarSearch.\_jQueryInterface;

$\_\_default["default"].fn[NAME$3].Constructor = NavbarSearch;

$\_\_default["default"].fn[NAME$3].noConflict = function () {

$\_\_default["default"].fn[NAME$3] = JQUERY\_NO\_CONFLICT$3;

return NavbarSearch.\_jQueryInterface;

};

var NAME$2 = 'Toasts';

var DATA\_KEY$2 = 'lte.toasts';

var EVENT\_KEY$1 = "." + DATA\_KEY$2;

var JQUERY\_NO\_CONFLICT$2 = $\_\_default["default"].fn[NAME$2];

var EVENT\_INIT = "init" + EVENT\_KEY$1;

var EVENT\_CREATED = "created" + EVENT\_KEY$1;

var EVENT\_REMOVED = "removed" + EVENT\_KEY$1;

var SELECTOR\_CONTAINER\_TOP\_RIGHT = '#toastsContainerTopRight';

var SELECTOR\_CONTAINER\_TOP\_LEFT = '#toastsContainerTopLeft';

var SELECTOR\_CONTAINER\_BOTTOM\_RIGHT = '#toastsContainerBottomRight';

var SELECTOR\_CONTAINER\_BOTTOM\_LEFT = '#toastsContainerBottomLeft';

var CLASS\_NAME\_TOP\_RIGHT = 'toasts-top-right';

var CLASS\_NAME\_TOP\_LEFT = 'toasts-top-left';

var CLASS\_NAME\_BOTTOM\_RIGHT = 'toasts-bottom-right';

var CLASS\_NAME\_BOTTOM\_LEFT = 'toasts-bottom-left';

var POSITION\_TOP\_RIGHT = 'topRight';

var POSITION\_TOP\_LEFT = 'topLeft';

var POSITION\_BOTTOM\_RIGHT = 'bottomRight';

var POSITION\_BOTTOM\_LEFT = 'bottomLeft';

var Default$2 = {

position: POSITION\_TOP\_RIGHT,

fixed: true,

autohide: false,

autoremove: true,

delay: 1000,

fade: true,

icon: null,

image: null,

imageAlt: null,

imageHeight: '25px',

title: null,

subtitle: null,

close: true,

body: null,

class: null

};

var Toasts = /\*#\_\_PURE\_\_\*/function () {

function Toasts(element, config) {

this.\_config = config;

this.\_prepareContainer();

$\_\_default["default"]('body').trigger($\_\_default["default"].Event(EVENT\_INIT));

} // Public

var \_proto = Toasts.prototype;

\_proto.create = function create() {

var toast = $\_\_default["default"]('<div class="toast" role="alert" aria-live="assertive" aria-atomic="true"/>');

toast.data('autohide', this.\_config.autohide);

toast.data('animation', this.\_config.fade);

if (this.\_config.class) {

toast.addClass(this.\_config.class);

}

if (this.\_config.delay && this.\_config.delay != 500) {

toast.data('delay', this.\_config.delay);

}

var toastHeader = $\_\_default["default"]('<div class="toast-header">');

if (this.\_config.image != null) {

var toastImage = $\_\_default["default"]('<img />').addClass('rounded mr-2').attr('src', this.\_config.image).attr('alt', this.\_config.imageAlt);

if (this.\_config.imageHeight != null) {

toastImage.height(this.\_config.imageHeight).width('auto');

}

toastHeader.append(toastImage);

}

if (this.\_config.icon != null) {

toastHeader.append($\_\_default["default"]('<i />').addClass('mr-2').addClass(this.\_config.icon));

}

if (this.\_config.title != null) {

toastHeader.append($\_\_default["default"]('<strong />').addClass('mr-auto').html(this.\_config.title));

}

if (this.\_config.subtitle != null) {

toastHeader.append($\_\_default["default"]('<small />').html(this.\_config.subtitle));

}

if (this.\_config.close == true) {

var toastClose = $\_\_default["default"]('<button data-dismiss="toast" />').attr('type', 'button').addClass('ml-2 mb-1 close').attr('aria-label', 'Close').append('<span aria-hidden="true">&times;</span>');

if (this.\_config.title == null) {

toastClose.toggleClass('ml-2 ml-auto');

}

toastHeader.append(toastClose);

}

toast.append(toastHeader);

if (this.\_config.body != null) {

toast.append($\_\_default["default"]('<div class="toast-body" />').html(this.\_config.body));

}

$\_\_default["default"](this.\_getContainerId()).prepend(toast);

var $body = $\_\_default["default"]('body');

$body.trigger($\_\_default["default"].Event(EVENT\_CREATED));

toast.toast('show');

if (this.\_config.autoremove) {

toast.on('hidden.bs.toast', function () {

$\_\_default["default"](this).delay(200).remove();

$body.trigger($\_\_default["default"].Event(EVENT\_REMOVED));

});

}

} // Static

;

\_proto.\_getContainerId = function \_getContainerId() {

if (this.\_config.position == POSITION\_TOP\_RIGHT) {

return SELECTOR\_CONTAINER\_TOP\_RIGHT;

}

if (this.\_config.position == POSITION\_TOP\_LEFT) {

return SELECTOR\_CONTAINER\_TOP\_LEFT;

}

if (this.\_config.position == POSITION\_BOTTOM\_RIGHT) {

return SELECTOR\_CONTAINER\_BOTTOM\_RIGHT;

}

if (this.\_config.position == POSITION\_BOTTOM\_LEFT) {

return SELECTOR\_CONTAINER\_BOTTOM\_LEFT;

}

};

\_proto.\_prepareContainer = function \_prepareContainer() {

if ($\_\_default["default"](this.\_getContainerId()).length === 0) {

var container = $\_\_default["default"]('<div />').attr('id', this.\_getContainerId().replace('#', ''));

if (this.\_config.position == POSITION\_TOP\_RIGHT) {

container.addClass(CLASS\_NAME\_TOP\_RIGHT);

} else if (this.\_config.position == POSITION\_TOP\_LEFT) {

container.addClass(CLASS\_NAME\_TOP\_LEFT);

} else if (this.\_config.position == POSITION\_BOTTOM\_RIGHT) {

container.addClass(CLASS\_NAME\_BOTTOM\_RIGHT);

} else if (this.\_config.position == POSITION\_BOTTOM\_LEFT) {

container.addClass(CLASS\_NAME\_BOTTOM\_LEFT);

}

$\_\_default["default"]('body').append(container);

}

if (this.\_config.fixed) {

$\_\_default["default"](this.\_getContainerId()).addClass('fixed');

} else {

$\_\_default["default"](this.\_getContainerId()).removeClass('fixed');

}

} // Static

;

Toasts.\_jQueryInterface = function \_jQueryInterface(option, config) {

return this.each(function () {

var \_options = $\_\_default["default"].extend({}, Default$2, config);

var toast = new Toasts($\_\_default["default"](this), \_options);

if (option === 'create') {

toast[option]();

}

});

};

return Toasts;

$\_\_default["default"].fn[NAME$2] = Toasts.\_jQueryInterface;

$\_\_default["default"].fn[NAME$2].Constructor = Toasts;

$\_\_default["default"].fn[NAME$2].noConflict = function () {

$\_\_default["default"].fn[NAME$2] = JQUERY\_NO\_CONFLICT$2;

return Toasts.\_jQueryInterface;

};

var NAME$1 = 'TodoList';

var DATA\_KEY$1 = 'lte.todolist';

var JQUERY\_NO\_CONFLICT$1 = $\_\_default["default"].fn[NAME$1];

var SELECTOR\_DATA\_TOGGLE = '[data-widget="todo-list"]';

var CLASS\_NAME\_TODO\_LIST\_DONE = 'done';

var Default$1 = {

onCheck: function onCheck(item) {

return item;

},

onUnCheck: function onUnCheck(item) {

return item;

}

};

var TodoList = /\*#\_\_PURE\_\_\*/function () {

function TodoList(element, config) {

this.\_config = config;

this.\_element = element;

this.\_init();

} // Public

var \_proto = TodoList.prototype;

\_proto.toggle = function toggle(item) {

item.parents('li').toggleClass(CLASS\_NAME\_TODO\_LIST\_DONE);

if (!$\_\_default["default"](item).prop('checked')) {

this.unCheck($\_\_default["default"](item));

return;

}

this.check(item);

};

\_proto.check = function check(item) {

this.\_config.onCheck.call(item);

};

\_proto.unCheck = function unCheck(item) {

this.\_config.onUnCheck.call(item);

} // Private

;

\_proto.\_init = function \_init() {

var \_this = this;

var $toggleSelector = this.\_element;

$toggleSelector.find('input:checkbox:checked').parents('li').toggleClass(CLASS\_NAME\_TODO\_LIST\_DONE);

$toggleSelector.on('change', 'input:checkbox', function (event) {

\_this.toggle($\_\_default["default"](event.target));

});

} // Static

;

TodoList.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY$1);

if (!data) {

data = $\_\_default["default"](this).data();

}

var \_options = $\_\_default["default"].extend({}, Default$1, typeof config === 'object' ? config : data);

var plugin = new TodoList($\_\_default["default"](this), \_options);

$\_\_default["default"](this).data(DATA\_KEY$1, typeof config === 'object' ? config : data);

if (config === 'init') {

plugin[config]();

}

});

};

return TodoList;

}();

$\_\_default["default"](window).on('load', function () {

TodoList.\_jQueryInterface.call($\_\_default["default"](SELECTOR\_DATA\_TOGGLE));

});

$\_\_default["default"].fn[NAME$1] = TodoList.\_jQueryInterface;

$\_\_default["default"].fn[NAME$1].Constructor = TodoList;

$\_\_default["default"].fn[NAME$1].noConflict = function () {

$\_\_default["default"].fn[NAME$1] = JQUERY\_NO\_CONFLICT$1;

return TodoList.\_jQueryInterface;

};

var NAME = 'Treeview';

var DATA\_KEY = 'lte.treeview';

var EVENT\_KEY = "." + DATA\_KEY;

var JQUERY\_NO\_CONFLICT = $\_\_default["default"].fn[NAME];

var EVENT\_EXPANDED = "expanded" + EVENT\_KEY;

var EVENT\_COLLAPSED = "collapsed" + EVENT\_KEY;

var EVENT\_LOAD\_DATA\_API = "load" + EVENT\_KEY;

var SELECTOR\_LI = '.nav-item';

var SELECTOR\_LINK = '.nav-link';

var SELECTOR\_TREEVIEW\_MENU = '.nav-treeview';

var SELECTOR\_OPEN = '.menu-open';

var SELECTOR\_DATA\_WIDGET = '[data-widget="treeview"]';

var CLASS\_NAME\_OPEN = 'menu-open';

var CLASS\_NAME\_IS\_OPENING = 'menu-is-opening';

var CLASS\_NAME\_SIDEBAR\_COLLAPSED = 'sidebar-collapse';

var Default = {

trigger: SELECTOR\_DATA\_WIDGET + " " + SELECTOR\_LINK,

animationSpeed: 300,

accordion: true,

expandSidebar: false,

sidebarButtonSelector: '[data-widget="pushmenu"]'

};

var Treeview = /\*#\_\_PURE\_\_\*/function () {

function Treeview(element, config) {

this.\_config = config;

this.\_element = element;

} // Public

var \_proto = Treeview.prototype;

\_proto.init = function init() {

$\_\_default["default"]("" + SELECTOR\_LI + SELECTOR\_OPEN + " " + SELECTOR\_TREEVIEW\_MENU + SELECTOR\_OPEN).css('display', 'block');

this.\_setupListeners();

};

\_proto.expand = function expand(treeviewMenu, parentLi) {

var \_this = this;

var expandedEvent = $\_\_default["default"].Event(EVENT\_EXPANDED);

if (this.\_config.accordion) {

var openMenuLi = parentLi.siblings(SELECTOR\_OPEN).first();

var openTreeview = openMenuLi.find(SELECTOR\_TREEVIEW\_MENU).first();

this.collapse(openTreeview, openMenuLi);

}

parentLi.addClass(CLASS\_NAME\_IS\_OPENING);

treeviewMenu.stop().slideDown(this.\_config.animationSpeed, function () {

parentLi.addClass(CLASS\_NAME\_OPEN);

$\_\_default["default"](\_this.\_element).trigger(expandedEvent);

});

if (this.\_config.expandSidebar) {

this.\_expandSidebar();

}

};

\_proto.collapse = function collapse(treeviewMenu, parentLi) {

var \_this2 = this;

var collapsedEvent = $\_\_default["default"].Event(EVENT\_COLLAPSED);

parentLi.removeClass(CLASS\_NAME\_IS\_OPENING + " " + CLASS\_NAME\_OPEN);

treeviewMenu.stop().slideUp(this.\_config.animationSpeed, function () {

$\_\_default["default"](\_this2.\_element).trigger(collapsedEvent);

treeviewMenu.find(SELECTOR\_OPEN + " > " + SELECTOR\_TREEVIEW\_MENU).slideUp();

treeviewMenu.find(SELECTOR\_OPEN).removeClass(CLASS\_NAME\_IS\_OPENING + " " + CLASS\_NAME\_OPEN);

});

};

\_proto.toggle = function toggle(event) {

var $relativeTarget = $\_\_default["default"](event.currentTarget);

var $parent = $relativeTarget.parent();

var treeviewMenu = $parent.find("> " + SELECTOR\_TREEVIEW\_MENU);

if (!treeviewMenu.is(SELECTOR\_TREEVIEW\_MENU)) {

if (!$parent.is(SELECTOR\_LI)) {

treeviewMenu = $parent.parent().find("> " + SELECTOR\_TREEVIEW\_MENU);

}

if (!treeviewMenu.is(SELECTOR\_TREEVIEW\_MENU)) {

return;

}

}

event.preventDefault();

var parentLi = $relativeTarget.parents(SELECTOR\_LI).first();

var isOpen = parentLi.hasClass(CLASS\_NAME\_OPEN);

if (isOpen) {

this.collapse($\_\_default["default"](treeviewMenu), parentLi);

} else {

this.expand($\_\_default["default"](treeviewMenu), parentLi);

}

} // Private

;

\_proto.\_setupListeners = function \_setupListeners() {

var \_this3 = this;

var elementId = this.\_element.attr('id') !== undefined ? "#" + this.\_element.attr('id') : '';

$\_\_default["default"](document).on('click', "" + elementId + this.\_config.trigger, function (event) {

\_this3.toggle(event);

});

};

\_proto.\_expandSidebar = function \_expandSidebar() {

if ($\_\_default["default"]('body').hasClass(CLASS\_NAME\_SIDEBAR\_COLLAPSED)) {

$\_\_default["default"](this.\_config.sidebarButtonSelector).PushMenu('expand');

}

} // Static

;

Treeview.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $\_\_default["default"](this).data(DATA\_KEY);

var \_options = $\_\_default["default"].extend({}, Default, $\_\_default["default"](this).data());

if (!data) {

data = new Treeview($\_\_default["default"](this), \_options);

$\_\_default["default"](this).data(DATA\_KEY, data);

}

if (config === 'init') {

data[config]();

}

});

};

return Treeview;

}();

$\_\_default["default"](window).on(EVENT\_LOAD\_DATA\_API, function () {

$\_\_default["default"](SELECTOR\_DATA\_WIDGET).each(function () {

Treeview.\_jQueryInterface.call($\_\_default["default"](this), 'init');

});

});

$\_\_default["default"].fn[NAME] = Treeview.\_jQueryInterface;

$\_\_default["default"].fn[NAME].Constructor = Treeview;

$\_\_default["default"].fn[NAME].noConflict = function () {

$\_\_default["default"].fn[NAME] = JQUERY\_NO\_CONFLICT;

return Treeview.\_jQueryInterface;

};

exports.CardRefresh = CardRefresh;

exports.CardWidget = CardWidget;

exports.ControlSidebar = ControlSidebar;

exports.DirectChat = DirectChat;

exports.Dropdown = Dropdown;

exports.ExpandableTable = ExpandableTable;

exports.Fullscreen = Fullscreen;

exports.IFrame = IFrame;

exports.Layout = Layout;

exports.NavbarSearch = NavbarSearch;

exports.PushMenu = PushMenu;

exports.SidebarSearch = SidebarSearch;

exports.Toasts = Toasts;

exports.TodoList = TodoList;

exports.Treeview = Treeview;

Object.defineProperty(exports, '\_\_esModule', { value: true });

}));

//# sourceMappingURL=adminlte.js.map