import $ from 'jquery'

/\*\*

\* Constants

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

\*/

const NAME = 'CardWidget'

const DATA\_KEY = 'lte.cardwidget'

const EVENT\_KEY = `.${DATA\_KEY}`

const JQUERY\_NO\_CONFLICT = $.fn[NAME]

const EVENT\_EXPANDED = `expanded${EVENT\_KEY}`

const EVENT\_COLLAPSED = `collapsed${EVENT\_KEY}`

const EVENT\_MAXIMIZED = `maximized${EVENT\_KEY}`

const EVENT\_MINIMIZED = `minimized${EVENT\_KEY}`

const EVENT\_REMOVED = `removed${EVENT\_KEY}`

const CLASS\_NAME\_CARD = 'card'

const CLASS\_NAME\_COLLAPSED = 'collapsed-card'

const CLASS\_NAME\_COLLAPSING = 'collapsing-card'

const CLASS\_NAME\_EXPANDING = 'expanding-card'

const CLASS\_NAME\_WAS\_COLLAPSED = 'was-collapsed'

const CLASS\_NAME\_MAXIMIZED = 'maximized-card'

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

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

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

const SELECTOR\_CARD = `.${CLASS\_NAME\_CARD}`

const SELECTOR\_CARD\_HEADER = '.card-header'

const SELECTOR\_CARD\_BODY = '.card-body'

const SELECTOR\_CARD\_FOOTER = '.card-footer'

const Default = {

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'

}

class CardWidget {

constructor(element, settings) {

this.\_element = element

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

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

this.\_parent = element

}

this.\_settings = $.extend({}, Default, settings)

}

collapse() {

this.\_parent.addClass(CLASS\_NAME\_COLLAPSING).children(`${SELECTOR\_CARD\_BODY}, ${SELECTOR\_CARD\_FOOTER}`)

.slideUp(this.\_settings.animationSpeed, () => {

this.\_parent.addClass(CLASS\_NAME\_COLLAPSED).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($.Event(EVENT\_COLLAPSED), this.\_parent)

}

expand() {

this.\_parent.addClass(CLASS\_NAME\_EXPANDING).children(`${SELECTOR\_CARD\_BODY}, ${SELECTOR\_CARD\_FOOTER}`)

.slideDown(this.\_settings.animationSpeed, () => {

this.\_parent.removeClass(CLASS\_NAME\_COLLAPSED).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($.Event(EVENT\_EXPANDED), this.\_parent)

}

remove() {

this.\_parent.slideUp()

this.\_element.trigger($.Event(EVENT\_REMOVED), this.\_parent)

}

toggle() {

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

this.expand()

return

}

this.collapse()

}

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(),

position: 'fixed',

transition: 'all .15s'

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

const $element = $(this)

$element.addClass(CLASS\_NAME\_MAXIMIZED)

$('html').addClass(CLASS\_NAME\_MAXIMIZED)

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

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

}

$element.dequeue()

})

this.\_element.trigger($.Event(EVENT\_MAXIMIZED), this.\_parent)

}

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 () {

const $element = $(this)

$element.removeClass(CLASS\_NAME\_MAXIMIZED)

$('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($.Event(EVENT\_MINIMIZED), this.\_parent)

}

toggleMaximize() {

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

this.minimize()

return

}

this.maximize()

}

// Private

\_init(card) {

this.\_parent = card

$(this).find(this.\_settings.collapseTrigger).click(() => {

this.toggle()

})

$(this).find(this.\_settings.maximizeTrigger).click(() => {

this.toggleMaximize()

})

$(this).find(this.\_settings.removeTrigger).click(() => {

this.remove()

})

}

// Static

static \_jQueryInterface(config) {

let data = $(this).data(DATA\_KEY)

const \_config = $.extend({}, Default, $(this).data())

if (!data) {

data = new CardWidget($(this), \_config)

$(this).data(DATA\_KEY, 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($(this))

}

}

}

/\*\*

\* Data API

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

\*/

$(document).on('click', SELECTOR\_DATA\_COLLAPSE, function (event) {

if (event) {

event.preventDefault()

}

CardWidget.\_jQueryInterface.call($(this), 'toggle')

})

$(document).on('click', SELECTOR\_DATA\_REMOVE, function (event) {

if (event) {

event.preventDefault()

}

CardWidget.\_jQueryInterface.call($(this), 'remove')

})

$(document).on('click', SELECTOR\_DATA\_MAXIMIZE, function (event) {

if (event) {

event.preventDefault()

}

CardWidget.\_jQueryInterface.call($(this), 'toggleMaximize')

})

/\*\*

\* jQuery API

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

\*/

$.fn[NAME] = CardWidget.\_jQueryInterface

$.fn[NAME].Constructor = CardWidget

$.fn[NAME].noConflict = function () {

$.fn[NAME] = JQUERY\_NO\_CONFLICT

return CardWidget.\_jQueryInterface

}

export default CardWidget

Footer