import $ from 'jquery'

const NAME = 'TodoList'

const DATA\_KEY = 'lte.todolist'

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

const SELECTOR\_DATA\_TOGGLE = '[data-widget="todo-list"]'

const CLASS\_NAME\_TODO\_LIST\_DONE = 'done'

const Default = {

onCheck(item) {

return item

},

onUnCheck(item) {

return item

}

}

class TodoList {

constructor(element, config) {

this.\_config = config

this.\_element = element

this.\_init()

}

toggle(item) {

item.parents('li').toggleClass(CLASS\_NAME\_TODO\_LIST\_DONE)

if (!$(item).prop('checked')) {

this.unCheck(item)

return

}

this.check(item)

}

check(item) {

this.\_config.onCheck(item)

}

unCheck(item) {

this.\_config.onUnCheck(item)

}

\_init() {

const $toggleSelector = this.\_element

$toggleSelector.find('input:checkbox:checked').parents('li').toggleClass(CLASS\_NAME\_TODO\_LIST\_DONE)

$toggleSelector.on('change', 'input:checkbox', event => {

this.toggle($(event.target))

})

}

static \_jQueryInterface(config) {

return this.each(function () {

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

const \_config = $.extend({}, Default, typeof config === 'object' ? config : $(this).data())

if (!data) {

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

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

data.\_init()

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

if (typeof data[config] === 'undefined') {

throw new TypeError(`No method named "${config}"`)

}

data[config]()

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

data.\_init()

}

})

}

}

$(window).on('load', () => {

TodoList.\_jQueryInterface.call($(SELECTOR\_DATA\_TOGGLE))

})

$.fn[NAME] = TodoList.\_jQueryInterface

$.fn[NAME].Constructor = TodoList

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

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

return TodoList.\_jQueryInterface

}

export default TodoList

Footer