در تمام فایل های less , کلاس های dropdown را به .as-dropdown تبدیل می کنیم

در فایل js تمام دوسری Classes ها از ‘dropdown به ‘as-dropdown تبدیل می کنیم , یک سری در defaults در Custom Classes قرار دارند مثل as-dropdown-icon-toggle

در js کد rtl: true به defaults اضافه می شود

در js داخل تابع \_mergeClasses کدهای زیر اضافه شد

// Check for user class

if (user[i]) {

if (self.options.rtl) {

if (i === "iconNext")

classStr = 'as-dropdown-icon-rtl'

if (i === "toggleIcon")

classStr = 'as-dropdown-icon-rtl'

}

classStr += ' ';

classStr += user[i];

}

if (self.options.rtl) {

if (i === "menuItem") {

classStr += ' ';

classStr += 'as-dropdown-item-rtl'

}

else if (i === "toggleText") {

classStr += ' ';

classStr += 'as-dropdown-text-rtl'

}

}

برای اضافه کردن Value به حالت لیست ابزارک در تابع زیر

/\*\*

\*

\* Populate from ordered or unordered list

\*

\* ================================================================ \*/

\_populateList: function( $parent ) {

کد زیر را در خط 2511 اضافه می کنیم پایین Check For Text

// Check for value

if ($this.data('dropdown-value')) {

item.value = $this.data('dropdown-value');

}

و ایجاد حالت لینک کد زیر اضافه شده است :

if (opt.link === true) {

// Check for href

if ($this.data('dropdown-href')) {

item.href = $this.data('dropdown-href');

}

}

در تابع زیر به صورت زیر تغییر داده شد تا حالت انتخاب parent تنها از طریق option باشد و نه بر اساس href یا value

/\*\*

\*

\* Add an item

\*

\* ================================================================ \*/

addItem: function( item, menu ) {

تغییر :

// Add parent

//if ( item.value || item.href || opt.selectParents ) {

if (opt.selectParents ) {

برای deselect شدن ایتم انتخاب شده در حالت single ما یک آپشن به اسم enableDeselectOnSingleMode اضافه کردیم که در متد زیر چک می شود

/\*\*

\*

\* Select an item

\*

\* ================================================================ \*/

selectItem: function( item ) {

نحوه چک کردن :

if ( opt.multiple ) {

if ( !inst.selected )

inst.selected = [];

if ( item.selected ) {

// Deselect

self.deselect( item );

} else {

// Select

item.selected = true;

item.elem.addClass( cls.selected );

inst.selected.push( item.uid );

// Add value

if ( item.value != null ) {

if ( inst.value == null )

inst.value = [];

inst.value.push( item.value );

}

}

} else {

if (opt.enableDeselectOnSingleMode == true && item.selected) {

// Deselect

self.deselect(item);

} else {

// Select

inst.selected = item.uid;

item.selected = true;

item.elem.addClass(cls.selected);

// Update value

inst.value = item.value;

}

}

سپس در متد deselect برای حالت single در قسمت else تیکه کد else زیر را قرار دادیم

if ( inst.selected && opt.multiple ) {

var index = $.inArray( item.uid, inst.selected );

if ( index )

inst.selected.splice( index, 1 );

// Remove value(s)

if ( item.value != null ) {

if ( inst.value == null )

inst.value = [];

inst.value = jQuery.grep( inst.value, function(value) {

return value != item.value;

});

}

} else {

if ( item.value == inst.value )

inst.value = null;

inst.selected = false

self.toggleText(opt.titleText);

}

و در نهایت برای اینکه بعد از deselect متن دارپدون به حالت اول برگردد , کد

self.toggleText(opt.titleText); را افزوده اییم

توابع اضافه شده :

/\*\*

\*

\* Get an item by Value

\*

\* ================================================================ \*/

getItemByValue: function (value) {

var self = this

var inst = self.instance

// No item, bail

if (!value)

return false;

return $.grep($.asObjectToArray(inst.items), function (e) { return e.value == value; })

}

توابع مورد استفاده

$drpService.asDropdown('selectValue', [8,7]);

$drpService.asDropdown('getItemByValue',8);

$drpService.asDropdown('selected');

Event ها :

$drpService.on("change", function (event, item) {

//console.dir(event)

console.dir(item)

//console.dir(self)

})

قطعه کد $drpService.on("change", در توابع زیر استفاده می شود :

/\*\*

\*

\* Select item(s) by value(s)

\*

\* ================================================================ \*/

selectValue: function( values, clear ) {

var self = this;

var inst = self.instance,

opt = self.options,

elem = self.elements,

cls = self.classes;

if ( !values )

values = [];

if ( ! ( values instanceof Array ) )

values = [ values ];

// Deselect all

if ( clear ) {

for ( var uid in inst.items )

self.deselect( uid );

}

// Multiple

if ( opt.multiple ) {

// Deselect all

if ( clear ) {

if ( opt.autoToggle )

self.toggleTextMulti();

}

if ( !values.length )

return;

$.each( values, function( i, value ) {

for ( var uid in inst.items ) {

if ( self.value( uid ) == value ) {

self.selectItem( uid );

if ( opt.autoToggle )

self.toggleTextMulti( self.text( uid ) );

if ('SELECT' != self.$elem.prop('tagName')) {

// Event

self.$elem.parent().trigger('change', [uid, self]);

}

}

}

});

return;

}

$.each( values, function( i, value ) {

for ( var uid in inst.items ) {

if ( self.value( uid ) == value ) {

self.selectItem( uid );

if ( opt.autoToggle )

self.toggleText( self.text( uid ) );

if ('SELECT' != self.$elem.prop('tagName')) {

// Event

self.$elem.parent().trigger('change', [uid, self]);

}

}

}

});

},

و تابع

/\*\*

\*

\* Select an item or menu

\*

\* ================================================================ \*/

select: function( item ) {

var self = this;

var inst = self.instance,

opt = self.options,

elem = self.elements,

cls = self.classes;

// Get the item

item = self.getItem( item );

// Get a menu

if ( !item ) {

var menu = self.getMenu( item );

// Open menu

if ( menu )

self.openMenu( menu );

return;

}

// Parent, open menu

if ( opt.nested && item.selectable && item.children.menu ) {

self.openMenu( item.children.menu );

return;

}

// Not selectable

if ( !item.selectable ) {

// Link?

if ( item.href )

window.location.href = item.href;

return;

}

// Get currently selected item

var previous = false;

if ( !opt.multiple )

previous = self.getItem( inst.selected );

if ( previous.uid == item.uid )

previous = false;

// Callback

self.\_beforeSelect( item, previous );

// Select an item

self.selectItem( item, previous );

// Select/deselect previous

if ( previous ) {

if ( !opt.multiple ) {

previous.selected = false;

previous.elem.removeClass( cls.selected );

}

// Select/deselect previous parent

self.selectParent( previous );

}

// Update toggle text

if ( opt.autoToggle ) {

if ( !inst.selected || !inst.selected.length ) {

if ( opt.multiple )

self.toggleTextMulti();

else

self.toggleText();

} else {

if ( opt.multiple )

self.toggleTextMulti( item.text );

else

self.toggleText( item.text );

}

}

// Close the dropdown

if ( !opt.multiple ) {

self.close();

}

// Callback

self.\_afterSelect(item, previous);

if ('SELECT' != self.$elem.prop('tagName')) {

// Event

self.$elem.parent().trigger('change', [item, self]);

}

}

به منظور عدم قابلیت انتخاب پدر ها قطعه کد زیر اصلاح گردید

// Add parent

//if ( item.value || item.href || opt.selectParents ) {

if ( item.href || opt.selectParents ) {

این قطعه کد در تابع زیر استفاده می شود

/\*\*

\*

\* Add an item

\*

\* ================================================================ \*/

addItem: function( item, menu ) {

در متد AfterClose , self.rset را کامنت کردیم تا منو در باز شدن مججد سر جایش باشد .