```
(function ($) {
  Drupal.behaviors.improved_multi_select = {
    attach: function(context, settings) {
      if (settings.improved_multi_select && settings.improved_multi_select.selectors) {
        var options = settings.improved_multi_select;

        for (var key in options.selectors) {
          var selector = options.selectors[key];
          $(selector, context).once('improvedselect', function() {
            var $select = $(this),
              moveButtons = '',
              improvedselect_id = $select.attr('id'),
              $cloned_select = null,
              cloned_select_id = '';
            if (options.orderable) {
              // If the select is orderable then we clone the original select
              // so that we have the original ordering to use later.
              $cloned_select = $select.clone();
              cloned_select_id = $cloned_select.attr('id');
              cloned_select_id += '-cloned';
              $cloned_select.attr('id', cloned_select_id);
              $cloned_select.appendTo($select.parent()).hide();
              // Move button markup to add to the widget.
              moveButtons = '<span class="move_up" sid="' + $select.attr('id') + '">' + Drupal.checkPlain(options.buttontext_moveup) + '</span>' +
                            '<span class="move_down" sid="' + $select.attr('id') + '">' + Drupal.checkPlain(options.buttontext_movedown) + '</span>';
            }
            $select.parent().append(
              '<div class="improvedselect" sid="' + $select.attr('id') + '" id="improvedselect-' + $select.attr('id') + '">' +
                '<div class="improvedselect-text-wrapper">' +
                  '<input type="text" class="improvedselect_filter" sid="' + $select.attr('id') + '" prev="" />' +
                '</div>' +
                '<ul class="improvedselect_sel"></ul>' +
                '<ul class="improvedselect_all"></ul>' +
                '<div class="improvedselect_control">' +
                  '<span class="add" sid="' + $select.attr('id') + '">' + Drupal.checkPlain(options.buttontext_add) + '</span>' +
                  '<span class="del" sid="' + $select.attr('id') + '">' + Drupal.checkPlain(options.buttontext_del) + '</span>' +
                  '<span class="add_all" sid="' + $select.attr('id') + '">' + Drupal.checkPlain(options.buttontext_addall) + '</span>' +
                  '<span class="del_all" sid="' + $select.attr('id') + '">' + Drupal.checkPlain(options.buttontext_delall) + '</span>' +
                  moveButtons +
                '</div>' +
                '<div class="clear"></div>' +
              '</div>');
            if ($select.find('optgroup').has('option').length > 0) {
              $select.parent().find('.improvedselect').addClass('has_group');
              // Build groups.
              $('#improvedselect-' + improvedselect_id + ' .improvedselect-text-wrapper', context)
                .after('<div class="improvedselect_tabs-wrapper" sid="' + $select.attr('id') + '"><ul class="improvedselect_tabs"></ul></div>');
              $select.find('optgroup').has('option').each(function() {
                $('#improvedselect-' + improvedselect_id + ' .improvedselect_tabs', context)
                  .append('<li><a href="">' + $(this).attr('label') + '</a></li>');
              });
              // Show all groups option.
              $('#improvedselect-' + improvedselect_id + ' .improvedselect_tabs', context)
                .prepend('<li class="all"><a href="">' + Drupal.t('All') + '</a></li>');
              // Select group.
              $('#improvedselect-' + improvedselect_id + ' .improvedselect_tabs li a', context).click(function(e) {
                var $group = $(this),
                  sid = $group.parent().parent().parent().attr('sid');
                $('#improvedselect-' + improvedselect_id + ' .improvedselect_tabs li.selected', context).removeClass('selected').find('a').unwrap();
                $group.wrap('<div>').parents('li').first().addClass('selected');

                // Any existing selections in the all list need to be unselected
                // if they aren't part of the newly selected group.
                if (!$group.hasClass('all')) {
                  $('#improvedselect-' + improvedselect_id + ' .improvedselect_all li.selected[group!=' + $group.text() + ']', context).removeClass('selected');
                }

                // Clear the filter if we have to.
                if (options.grouppresetfilter) {
                  // Clear filter box.
                  $('#improvedselect-' + improvedselect_id + ' .improvedselect_filter', context).val('');
                }
                // Force re-filtering.
                $('#improvedselect-' + improvedselect_id + ' .improvedselect_filter', context).attr('prev', '');
                // Re-filtering will handle the rest.
                improvedselectFilter(sid, options, context);
                e.preventDefault();
              });
              // Select all to begin.
              $('#improvedselect-' + improvedselect_id + ' .improvedselect_tabs li.all a', context).click();
            }

            $select.find('option, optgroup').each(function() {
              var $opt = $(this),
                group = '';
              if ($opt.attr('tagName') == 'OPTGROUP') {
                if ($opt.has('option').length) {
                  $('#improvedselect-'+ improvedselect_id +' .improvedselect_all', context)
```

```
                    .append('<li isgroup="isgroup" so="---' + $opt.attr('label') + '---">--- '+ $opt.attr('label') +' ---</li>');
                }
              }
              else {
                group = $opt.parent("optgroup").attr("label");
                if (group) {
                  group = ' group="' + group + '"';
                }
                if ($opt.attr('selected')) {
                  $('#improvedselect-' + improvedselect_id + ' .improvedselect_sel', context)
                    .append('<li so="' + $opt.attr('value') + '"' + group + '>' + $opt.text() + '</li>');
                }
                else {
                  $('#improvedselect-' + improvedselect_id + ' .improvedselect_all', context)
                    .append('<li so="' + $opt.attr('value') + '"' + group + '>' + $opt.text() + '</li>');
                }
              }
            });
            $('#improvedselect-'+ improvedselect_id + ' .improvedselect_sel li, #improvedselect-' + improvedselect_id + ' .improvedselect_all li[isgroup!="isgroup"]', context).click(function() {
              $(this).toggleClass('selected');
            });
            $select.hide();
            // Double click feature request.
            $('#improvedselect-'+ improvedselect_id + ' .improvedselect_sel li, #improvedselect-' + improvedselect_id + ' .improvedselect_all li[isgroup!="isgroup"]', context).dblclick(function() {
              // Store selected items.
              var selected = $(this).parent().find('li.selected'),
                current_class = $(this).parent().attr('class');
              // Add item.
              if (current_class == 'improvedselect_all') {
                $(this).parent().find('li.selected').removeClass('selected');
                $(this).addClass('selected');
                $(this).parent().parent().find('.add').click();
              }
              // Remove item.
              else {
                $(this).parent().find('li.selected').removeClass('selected');
                $(this).addClass('selected');
                $(this).parent().parent().find('.del').click();
              }
              // Restore selected items.
              if (selected.length) {
                for (var k = 0; k < selected.length; k++) {
                  if ($(selected[k]).parent().attr('class') == current_class) {
                    $(selected[k]).addClass('selected');
                  }
                }
              }
            });
```

```

      // Set the height of the select fields based on the height of the
      // parent, otherwise it can end up with a lot of wasted space.
      $('.improvedselect_sel, .improvedselect_all').each(function() {
       $(this).height($(this).parent().height() - 35);
      });
     });
    }

    $('.improvedselect_filter', context).keyup(function() {
     improvedselectFilter($(this).attr('sid'), options, context);
    });

    // Add selected items.
    $('.improvedselect .add', context).click(function() {
     var sid = $(this).attr('sid');
     $('#improvedselect-' + sid + ' .improvedselect_all .selected', context).each(function() {
      $opt = $(this);
      $opt.removeClass('selected');
      improvedselectUpdateGroupVisibility($opt, 1);
      $('#improvedselect-' + sid + ' .improvedselect_sel', context).append($opt);
     });
     improvedselectUpdate(sid, context);
    });

    // Remove selected items.
    $('.improvedselect .del', context).click(function() {
     var sid = $(this).attr('sid');
     $('#improvedselect-' + sid + ' .improvedselect_sel .selected', context).each(function() {
      $opt = $(this);
      $opt.removeClass('selected');
      $('#improvedselect-' + sid + ' .improvedselect_all', context).append($opt);
      improvedselectUpdateGroupVisibility($opt, 0);
     });
     // Force re-filtering.
     $('#improvedselect-' + sid + ' .improvedselect_filter', context).attr('prev', '');
     // Re-filtering will handle the rest.
     improvedselectFilter(sid, options, context);
     improvedselectUpdate(sid, context);
    });

    // Add all items.
    $('.improvedselect .add_all', context).click(function() {
     var sid = $(this).attr('sid');
     $('#improvedselect-' + sid + ' .improvedselect_all li[isgroup!=isgroup]', context).each(function() {
      $opt = $(this);
      if ($opt.css('display') != 'none') {
       $opt.removeClass('selected');
       improvedselectUpdateGroupVisibility($opt, 1);
       $('#improvedselect-' + sid + ' .improvedselect_sel', context).append($opt);
```

```
          }
        });
        improvedselectUpdate(sid, context);
      });

      // Remove all items.
      $('.improvedselect .del_all', context).click(function() {
        var sid = $(this).attr('sid');
        $('#improvedselect-' + sid + ' .improvedselect_sel li', context).each(function() {
          $opt = $(this);
          $opt.removeClass('selected');
          $('#improvedselect-' + sid + ' .improvedselect_all', context).append($opt);
          improvedselectUpdateGroupVisibility($opt, 0);
        });
        // Force re-filtering.
        $('#improvedselect-' + sid + ' .improvedselect_filter', context).attr('prev', '');
        // Re-filtering will handle the rest.
        improvedselectFilter(sid, options, context);
        improvedselectUpdate(sid, context);
      });

      // Move selected items up.
      $('.improvedselect .move_up', context).click(function() {
        var sid = $(this).attr('sid');
        $('#improvedselect-' + sid + ' .improvedselect_sel .selected', context).each(function() {
          var $selected = $(this);
          // Don't move selected items past other selected items or there will
          // be problems when moving multiple items at once.
          $selected.prev(':not(.selected)').before($selected);
        });
        improvedselectUpdate(sid, context);
      });

      // Move selected items down.
      $('.improvedselect .move_down', context).click(function() {
        var sid = $(this).attr('sid');
        // Run through the selections in reverse, otherwise problems occur
        // when moving multiple items at once.
        $($('#improvedselect-' + sid + ' .improvedselect_sel .selected', context).get().reverse()).each(function() {
          var $selected = $(this);
          // Don't move selected items past other selected items or there will
          // be problems when moving multiple items at once.
          $selected.next(':not(.selected)').after($selected);
        });
        improvedselectUpdate(sid, context);
      });
    }
  }
};

/**
 * Filter the all options list.
 */
function improvedselectFilter(sid, options, context) {
  $filter = $('.improvedselect_filter', context);
  // Get current selected group.
  var $selectedGroup = $('#improvedselect-' + sid + ' .improvedselect_tabs li.selected:not(.all) a', context),
    text = $filter.val(),
    pattern,
    regex,
    words;

  if (text.length && text != $filter.attr('prev')) {
    $filter.attr('prev', text);
    switch (options.filtertype) {
      case 'partial':
      default:
        pattern = text;
        break;
      case 'exact':
        pattern = '^' + text + '$';
        break;
      case 'anywords':
        words = text.split(' ');
        pattern = '';
        for (var i = 0; i < words.length; i++) {
          if (words[i]) {
            pattern += (pattern) ? '|\\b' + words[i] + '\\b' : '\\b' + words[i] + '\\b';
          }
        }
        break;
      case 'anywords_partial':
        words = text.split(' ');
        pattern = '';
        for (var i = 0; i < words.length; i++) {
          if (words[i]) {
            pattern += (pattern) ? '|' + words[i] + '' : words[i];
          }
        }
        break;
      case 'allwords':
        words = text.split(' ');
        pattern = '^';
        // Add a lookahead for each individual word.
        // Lookahead is used because the words can match in any order
        // so this makes it simpler and faster.
        for (var i = 0; i < words.length; i++) {
          if (words[i]) {
            pattern += '(?=.*?\\b' + words[i] + '\\b)';
          }
        }
        break;
      case 'allwords_partial':
        words = text.split(' ');
        pattern = '^';
        // Add a lookahead for each individual word.
        // Lookahead is used because the words can match in any order
```

```
          // so this makes it simpler and faster.
          for (var i = 0; i < words.length; i++) {
            if (words[i]) {
              pattern += '(?=.*?' + words[i] + ')';
            }
          }
          break;
      }

      regex = new RegExp(pattern,'i');
      $('#improvedselect-' + sid + ' .improvedselect_all li', context).each(function() {
        $opt = $(this);
        if ($opt.attr('isgroup') != 'isgroup') {
          var str = $opt.text();
          if (str.match(regex) && (!$selectedGroup.length || $selectedGroup.text() == $opt.attr('group'))) {
            $opt.show();
            if ($opt.attr('group')) {
              // If a group is selected we don't need to show groups.
              if (!$selectedGroup.length) {
                $opt.siblings('li[isgroup="isgroup"][so="---' + $opt.attr('group') + '---"]').show();
              }
              else {
                $opt.siblings('li[isgroup="isgroup"][so="---' + $opt.attr('group') + '---"]').hide();
              }
            }
          }
          else {
            $opt.hide();
            if ($opt.attr('group')) {
              if ($opt.siblings('li[isgroup!="isgroup"][group="' + $opt.attr('group') + '"]:visible').length == 0) {
                $opt.siblings('li[isgroup="isgroup"][so="---' + $opt.attr('group') + '---"]').hide();
              }
            }
          }
        }
      });
    }
    else {
      if (!text.length) {
        $filter.attr('prev', '');
      }
      $('#improvedselect-' + sid + ' .improvedselect_all li', context).each(function() {
        var $opt = $(this);
        if ($opt.attr('isgroup') != 'isgroup') {
          if (!$selectedGroup.length || $selectedGroup.text() == $opt.attr('group')) {
            $opt.show();
          }
          else {
            $opt.hide();
          }
          improvedselectUpdateGroupVisibility($opt, 0);
        }
      });
    }
  }

  /**
   * Update the visibility of an option's group.
   *
   * @param $opt
   *   A jQuery object of a select option.
   * @param numItems
   *   How many items should be considered an empty group. Generally zero or one
   *   depending on if an item has been or is going to be removed or added.
   */
  function improvedselectUpdateGroupVisibility($opt, numItems) {
    var $selectedGroup = $opt.parents('.improvedselect').first().find('.improvedselect_tabs li.selected:not(.all) a');

    // Don't show groups if a group is selected.
    if ($opt.parent().children('li[isgroup!="isgroup"][group="' + $opt.attr('group') + '"]:visible').length <= numItems || $selectedGroup.length) {
      $opt.siblings('li[isgroup="isgroup"][so="---' + $opt.attr('group') + '---"]').hide();
    }
    else {
      $opt.siblings('li[isgroup="isgroup"][so="---' + $opt.attr('group') + '---"]').show();
    }
  }

  function improvedselectUpdate(sid, context) {
    // If we have sorting enabled, make sure we have the results sorted.
    var $select = $('#' + sid),
      $cloned_select = $('#' + sid + '-cloned');

    if ($cloned_select.length) {
      $select.find('option, optgroup').remove();
      $('#improvedselect-' + sid + ' .improvedselect_sel li', context).each(function() {
        var $li = $(this);
        $select.append($('<option></option').attr('value', $li.attr('so')).attr('selected', 'selected').text($li.text()));
      });
      // Now that the select has the options in the correct order, use the
      // cloned select for resetting the ul values.
      $select = $cloned_select;
    }
    else {
      $select.find('option:selected').attr("selected", "");
      $('#improvedselect-' + sid + ' .improvedselect_sel li', context).each(function() {
        $('#' + sid + ' [value="' + $(this).attr('so') + '"]', context).attr("selected", "selected");
      });
    }
```

```

    $select.find('option, optgroup').each(function() {
      $opt = $(this);
      if ($opt.attr('tagName') == 'OPTGROUP') {
        if ($opt.has('option').length) {
          $('#improvedselect-' + sid + ' .improvedselect_all', context).append($('#improvedselect-' + sid + ' .improvedselect_all [so="---' + $opt.attr('label') + '---"]', context));
        }
      }
      else {
        // When using reordering, the options will be from the cloned select,
        // meaning that there will be none selected, which means that items
        // in the selected list will not be reordered, which is what we want.
        if ($opt.attr("selected")) {
          $('#improvedselect-' + sid + ' .improvedselect_sel', context).append($('#improvedselect-' + sid + ' .improvedselect_sel [so="' + $opt.attr('value') + '"]', context));
        }
        else {
          $('#improvedselect-' + sid + ' .improvedselect_all', context).append($('#improvedselect-' + sid + ' .improvedselect_all [so="' + $opt.attr('value') + '"]', context));
        }
      }
    });
    // Don't use the $select variable here as it might be the clone.
    // Tell the ajax system the select has changed.
    $('#'+ sid, context).trigger('change');
  }

})(jQuery, Drupal);

```