【Bootstrap-table】bs_table_common.js

```javascript
var bootstrapTableBaseConfig = {
    height: getHeight(),
    idField:"dv_no",
    toolbar:"#toolbar",
    minimumCountColumns:2, //最小顯示欄位數量
    pagination:true,
    pageSize:'10000000', //預設row筆數
    pageList:['ALL'],
    queryParams:getQueryParams,
    search:false,
    showColumns:true,
    showFooter:false,
    showPaginationSwitch:true,
    showRefresh:true,
    showExport:true, //匯出按鈕
    showToggle:false,
    detailView:false,
    detailFormatter:detailFormatter,
    // responseHandler:responseHandler,
    virtualScroll:false,
    sidePagination:"server",
    contextMenu: '#bootstraptable-context-menu',
    contextMenuButton: '.bootstraptable-button',
    classes:'table table-bordered table-hover table-sm table-condensed nowrap',
}


/** common function begin */

function ajaxUpdata(config){

    var url = config.url || "";
    var update_type = config.update_type || "更新";
    var update_data = config.update_data || {};
    var ajax_type = config.ajax_type || "POST";
    var ajax_data_type = config.ajax_data_type || "json";
    var after_success_func = config.after_success_func || function(){};
    var after_error_func = config.after_success_func || function(){};
    // var bs_table = config.bs_table || $table;

    console.log('ajaxUpdata_config',config);

    var success_func = config.success_func || function(response){
        console.log("response",response);
        if (response.msg != undefined && (response.msg == "success")) {
            if(config.bs_table !== undefined) config.bs_table.bootstrapTable('refresh');
            alert(update_type+"資料成功!!");
        } else {
            alert((response.msg == undefined)?update_type+"失敗請洽系統管理員":response.msg);
        }
        after_success_func();
    }

    var error_func = success_func || function(response){
        var err_msg = response.msg || "";
        alert(err_msg+","+update_type+"失敗請洽系統管理員!!");
        // dialog.close();  //關閉對話視窗
        after_error_func();
    }
```

```javascript
        var ajax_config = {
                url:  url,
                type: ajax_type,
                data: update_data,
                error: error_func,
                success: success_func,
                dataType: ajax_data_type
                };
        console.log(ajax_config);
        //寫入資料庫
        $.ajax(ajax_config);
}


function getEditDataStr(beforDate,afterData){
    var resultTextArr = [];
    for(var key in beforDate){
        if (afterData[key] !== undefined && beforDate[key] != afterData[key]
        ){
            if (beforDate[key] == null && afterData[key] == '') continue;
            resultTextArr.push(key+":"+beforDate[key]+"=>"+afterData[key]);
        }
    }
    if(resultTextArr.length > 0){
        return resultTextArr.join(",");
    }else{
        return "";
    }
}

function getDataByClass(class_name,perfix){
    var data = {};
    _perfix = perfix || "";
    $('.'+class_name).each(function(){
        var el = this;
        console.log(el.id);
        var id = (el.id).replace(_perfix,'');
        var value = $(el).val();
        data[id] = value;

    });
    return data;
}


function initTypeahead(config){

    console.log('initTypeahead typeahead',typeof jQuery().typeahead);
    console.log('initTypeahead Bloodhound',typeof Bloodhound);
    //check typeahead exist
    if(typeof jQuery().typeahead !== 'function') return;
    if(typeof Bloodhound !== 'function') return;

    console.log("initTypeahead");

    var selectFunc = (config && config.selectFunc) || function(event, data){
        $("#USER_ID").val(data['USER_ID']);
        $("#USER_NAME").val(data['USER_NAME']);
    }
    var userInfo = new Bloodhound({
        datumTokenizer: Bloodhound.tokenizers.obj.whitespace('value'),
        queryTokenizer: Bloodhound.tokenizers.whitespace,
        remote: {
            url: '/db/php/getOisUser.php?search=%QUERY',
            wildcard: '%QUERY'
        }
    });

    $('#typeahead_input').typeahead({
```

```javascript
      // classNames: {
      //   input: 'Typeahead-input',
      //   hint: 'Typeahead-hint',
      //   selectable: 'Typeahead-selectable'
      // }
      // },
      hint: true,
      highlight: true,
      minLength: 1
      },
      {
      name: 'best-pictures',
      display: function(data){
        return  data['USER_ID']  ;
        // return  data['USER_ID']  + '-' + data['AD_ID']  + '-' + data['USER_NAME'];
      },
      source: userInfo,
      templates: {
        //查無資料顯示
        empty: [
        '<div class="empty-message">',
          '查無使用者',
        '</div>'
        ].join('\n'),

        suggestion: function(data) {
          return '<p><strong>' + data['USER_ID']  + '-' + data['AD_ID']  + '-' + data['USER_NAME'] + ' #' + data['EXT'] + '</strong>
</p>';
        }
      }
    });

    $('.typeahead').on('typeahead:select', function(event, data) {
      selectFunc(event, data);
    });

    $('.typeahead').on('typeahead:change ', function(event, data) {
      // console.log('change : ' , data);
    });

    $('.js_typeahead_reset_btn').on('click',function(){
      $('.js_typeahead_reset').each(function(index,el){
        $(el).val('');
      });
    });

  // for(var i = 1;i<=5;i++){
  //    if($('.js_typeahead_reset_btn_'+i).length > 0){
  //        $('.js_typeahead_reset_btn_'+i).on('click',function(){
  //            $('.js_typeahead_reset_'+i).each(function(index,el){
  //                $(el).val('');
  //            });
  //        });
  //    }
  // }
}

function isEmpty(str){
  if(str === undefined || str.trim() === "" || str === null){
    return true;
  }
  return false;
}


function getEditDataStr(beforDate,afterData){
  var resultTextArr = [];
```

```
      for(var key in beforDate){
        if (afterData[key] !== undefined && beforDate[key] != afterData[key]
        ){
          if (beforDate[key] == null && afterData[key] == '') continue;
          resultTextArr.push(key+":"+beforDate[key]+"=>"+afterData[key]);
        }
      }
      if(resultTextArr.length > 0){
        return resultTextArr.join(",");
      }else{
        return "";
      }
  }
  /*
  ================================================================================

  function responseHandler(res) {
      $.each(res.rows, function (i, row) {
        row.state = $.inArray(row.dv_no, selections) !== -1;
      });
      return res;
  }

  function getHeight(_height) {
      var _height = _height || 110;
      return $(window).height() - $(".panel-heading").height() - _height;
  }

  // function getQueryParams(params){

  //    var queryParams = {};
  //    $(".query").each(function(i,e){
  //       var id = (e.id).replace("q_","");
  //       queryParams[id]=$(e).val();
  //    });
  //    $.extend(queryParams,params);
  //       console.log("getQueryParams",queryParams);

  //    return queryParams;
  // }

  function getQueryParams(params){
      var queryParams = {};
      $(".query").each(function(i,e){
        var id = (e.id).replace("q_","");
        queryParams[id]=$(e).val();
      });
      $.extend(queryParams,params);
      return queryParams;
  }

  /* bootsrap table formatter */
  function detailFormatter(index, row) {
      var html = [];
      var trans = [];
      $.each(row, function (key, value) {
        if (key != 'total' && key != 'check'){
          var name=(trans[key] === undefined) ? key: trans[key];
          var value=(value === null || value === undefined ) ? '': value;
          html.push('<p><b>' + name + ':</b> ' + value + '</p>');
        }
      });
      return html.join('');
  }
```

```javascript
function operateFormatter(value, row, index) {

    return [
      '<a class="add" href="javascript:void(0)" title="add">',
      '<i class="glyphicon glyphicon-plus text-success"></i>',
      '</a> ',
      '<a class="edit" href="javascript:void(0)" title="edit">',
      '<i class="glyphicon glyphicon-edit"></i>',
      '</a> ',
      '<a class="remove" href="javascript:void(0)" title="remove">',
      '<i class="glyphicon glyphicon-remove text-danger"></i>',
      '</a>'
    ].join('');
}

function editFormatter(value, row, index){
    return [
    '<a class="edit" href="javascript:void(0)" title="edit">',
    '<i class="glyphicon glyphicon-edit"></i>',
    '</a>'
    ].join('');
}

function yesNoFormatter(value,row,index){
    return (value == 1)?'是':'否';
}

function snFormatter(value,row,index) {
    return index+1;
}

function scanNumberFormatter(data) {
    return (data == 1)?'是':'否';
}

function totalTextFormatter(data) {
    return 'Total';
}

function totalNameFormatter(data) {
    return data.length;
}

function nowrapFormatter(){
    return { css:{"whiie-space":"nowrap"}} ;
}

function totalPriceFormatter(data) {
    var total = 0;
    $.each(data, function (i, row) {
      total += +(row.price.substring(1));
    });
    return '$' + total;
}


// function buildTable($el) {
    //     var i, j, row,
    //           columns = [],
    //           data = [];
    //     var opt = $el.bootstrapTable('getOptions');
    //     var fixedColumnsOption = { fixedColumns: true,fixedNumber: +$('#fixedNumber').val()};
    //     $.extend(opt,fixedColumnsOption);
    //     console.log("bootstrap_option",opt);
    //     $el.bootstrapTable('destroy').bootstrapTable(opt);
    // }
```

```
$(window).resize(function () {

    var $table = $table || null;
    if($table !== null){

        $table.bootstrapTable('resetView', {
            height: getHeight()
        });
    }
});
```

選項設定預設值

```
var DEFAULTS = {
    height: undefined,
    classes: 'table table-bordered table-hover',
    buttons: {},
    theadClasses: '',
    headerStyle: function headerStyle(column) {
        return {};
    },
    rowStyle: function rowStyle(row, index) {
        return {};
    },
    rowAttributes: function rowAttributes(row, index) {
        return {};
    },
    undefinedText: '-',
    locale: undefined,
    virtualScroll: false,
    virtualScrollItemHeight: undefined,
    sortable: true,
    sortClass: undefined,
    silentSort: true,
    sortName: undefined,
    sortOrder: undefined,
    sortReset: false,
    sortStable: false,
    rememberOrder: false,
    serverSort: true,
    customSort: undefined,
    columns: [[]],
    data: [],
    url: undefined,
    method: 'get',
    cache: true,
    contentType: 'application/json',
    dataType: 'json',
    ajax: undefined,
    ajaxOptions: {},
    queryParams: function queryParams(params) {
        return params;
    },
    queryParamsType: 'limit',
    // 'limit', undefined
    responseHandler: function responseHandler(res) {
        return res;
    },
    totalField: 'total',
    totalNotFilteredField: 'totalNotFiltered',
    dataField: 'rows',
    footerField: 'footer',
    pagination: false,
    paginationParts: ['pageInfo', 'pageSize', 'pageList'],
    showExtendedPagination: false,
    paginationLoop: true,
    sidePagination: 'client',
    // client or server
```

```
totalRows: 0,
totalNotFiltered: 0,
pageNumber: 1,
pageSize: 10,
pageList: [10, 25, 50, 100],
paginationHAlign: 'right',
// right, left
paginationVAlign: 'bottom',
// bottom, top, both
paginationDetailHAlign: 'left',
// right, left
paginationPreText: '&lsaquo;',
paginationNextText: '&rsaquo;',
paginationSuccessivelySize: 5,
// Maximum successively number of pages in a row
paginationPagesBySide: 1,
// Number of pages on each side (right, left) of the current page.
paginationUseIntermediate: false,
// Calculate intermediate pages for quick access
search: false,
searchHighlight: false,
searchOnEnterKey: false,
strictSearch: false,
regexSearch: false,
searchSelector: false,
visibleSearch: false,
showButtonIcons: true,
showButtonText: false,
showSearchButton: false,
showSearchClearButton: false,
trimOnSearch: true,
searchAlign: 'right',
searchTimeOut: 500,
searchText: '',
customSearch: undefined,
showHeader: true,
showFooter: false,
footerStyle: function footerStyle(column) {
  return {};
},
searchAccentNeutralise: false,
showColumns: false,
showColumnsToggleAll: false,
showColumnsSearch: false,
minimumCountColumns: 1,
showPaginationSwitch: false,
showRefresh: false,
showToggle: false,
showFullscreen: false,
smartDisplay: true,
escape: false,
filterOptions: {
  filterAlgorithm: 'and'
},
idField: undefined,
selectItemName: 'btSelectItem',
clickToSelect: false,
ignoreClickToSelectOn: function ignoreClickToSelectOn(_ref) {
  var tagName = _ref.tagName;
  return ['A', 'BUTTON'].includes(tagName);
},
singleSelect: false,
checkboxHeader: true,
maintainMetaData: false,
multipleSelectRow: false,
uniqueId: undefined,
cardView: false,
detailView: false,
detailViewIcon: true,
```

```
        detailViewByClick: false,
        detailViewAlign: 'left',
        detailFormatter: function detailFormatter(index, row) {
          return '';
        },
        detailFilter: function detailFilter(index, row) {
          return true;
        },
        toolbar: undefined,
        toolbarAlign: 'left',
        buttonsToolbar: undefined,
        buttonsAlign: 'right',
        buttonsOrder: ['paginationSwitch', 'refresh', 'toggle', 'fullscreen', 'columns'],
        buttonsPrefix: CONSTANTS.classes.buttonsPrefix,
        buttonsClass: CONSTANTS.classes.buttons,
        icons: CONSTANTS.icons,
        iconSize: undefined,
        iconsPrefix: CONSTANTS.iconsPrefix,
        // glyphicon or fa(font-awesome)
        loadingFontSize: 'auto',
        loadingTemplate: function loadingTemplate(loadingMessage) {
          return "<span class=\"loading-wrap\">\n      <span class=\"loading-text\">".concat(loadingMessage, "</span>\n      <span
class=\"animation-wrap\"><span class=\"animation-dot\"></span></span>\n      </span>\n    ");
        },
        onAll: function onAll(name, args) {
          return false;
        },
        onClickCell: function onClickCell(field, value, row, $element) {
          return false;
        },
        onDblClickCell: function onDblClickCell(field, value, row, $element) {
          return false;
        },
        onClickRow: function onClickRow(item, $element) {
          return false;
        },
        onDblClickRow: function onDblClickRow(item, $element) {
          return false;
        },
        onSort: function onSort(name, order) {
          return false;
        },
        onCheck: function onCheck(row) {
          return false;
        },
        onUncheck: function onUncheck(row) {
          return false;
        },
        onCheckAll: function onCheckAll(rows) {
          return false;
        },
        onUncheckAll: function onUncheckAll(rows) {
          return false;
        },
        onCheckSome: function onCheckSome(rows) {
          return false;
        },
        onUncheckSome: function onUncheckSome(rows) {
          return false;
        },
        onLoadSuccess: function onLoadSuccess(data) {
          return false;
        },
        onLoadError: function onLoadError(status) {
          return false;
        },
        onColumnSwitch: function onColumnSwitch(field, checked) {
          return false;
```

```
    },
    onPageChange: function onPageChange(number, size) {
      return false;
    },
    onSearch: function onSearch(text) {
      return false;
    },
    onToggle: function onToggle(cardView) {
      return false;
    },
    onPreBody: function onPreBody(data) {
      return false;
    },
    onPostBody: function onPostBody() {
      return false;
    },
    onPostHeader: function onPostHeader() {
      return false;
    },
    onPostFooter: function onPostFooter() {
      return false;
    },
    onExpandRow: function onExpandRow(index, row, $detail) {
      return false;
    },
    onCollapseRow: function onCollapseRow(index, row) {
      return false;
    },
    onRefreshOptions: function onRefreshOptions(options) {
      return false;
    },
    onRefresh: function onRefresh(params) {
      return false;
    },
    onResetView: function onResetView() {
      return false;
    },
    onScrollBody: function onScrollBody() {
      return false;
    },
    onTogglePagination: function onTogglePagination(newState) {
      return false;
    }
  };
```