Apertura scheda lista classe (gwClassList)

La lista di dettaglio di classe (gwClassList) è una griglia evoluta contenente i record che fanno riferimento ad una data gwClass, con capacità di ordinare, filtrare, etc..

openGwClassListTab()

La lista viene sempre aperta in un tab di tipo tipo gwClassList descritto qui. Per fare ciò è esposta la function openGwClassListTab().

==== Parametri: ====

- **gwClassName** String, required, name of gwClass
- **title** String, optional, default ''
- **options** Object, optional, default null. Contains all Additional parameters

=== Parametri opzionali contenuti in *options*: ===

- **filters** Object [], optional, default null. Allow to show only elements matching filters. This filter List, unlikely *staticFilters*, can be removed by the user through the gwClassList UI. Example:

```
[
        {
          condition: 'AND',
          columnName: 'id_item',
          attributeGwid: 1,
          operator: 'IN',
          filterType: 'INTEGER',
          value: [1, 2, 3]
        }
]
```

- **staticFilters** Object [], optional, default null. Allow to show only elements matching filters. This filter List, unlikely *filters*, can be removed by the user through the gwClassList UI. Example:

```
[
        {
          condition: 'AND',
          columnName: 'status',
          attributeGwid: 1,
          operator: '=',
          filterType: 'STRING',
          value: 'OPN'
        }
]
```

- **hideToolbar** Boolean, optional, default false. When true Toolbar is hidden
- **hideNewIgnorePermissions** Boolean, optional, default false. When true the 'New' button is

hidden
- **hideDeleteIgnorePermissions** Boolean, optional, default false. When true the 'Delete' button is hidden
- **hideActionsDropDownButtonColumn** Boolean, optional, default false. When true the 'More Options' (AKA '3 dots') button is hidden. It overrides locally the eventually cnfigured global gwProject setting (configured inside gwProject xml <gwClassSettings> tag)
- **hideIndirectSelectionColumn** Boolean, optional, default false . When true Indirect Selection Column in grid is hidden. It overrides locally the eventually cnfigured global gwProject setting (configured inside gwProject xml <gwClassSettings> tag)
- **forceListedAttributes** Boolean, optional, default false. It works with 'listedAttributesNames' parameter. When true in grid are shown only the gwAttribute which name is inside 'listedAttributesNames' parameter
- **listedAttributesNames** String, optional, default null. Evaluated only when 'forceListedAttributes' is true. Is a list, comma separated, of gwAttribute name, which we want to force to compare in list (bypassing the set in gwAttribute List section in gwAdmin)
- **hiddenAttributesNamesList** String, optional, default null. Is a list, comma separated, of gwAttribute name, which we want to hide in list (the shown list is managed in gwAttribute List section in gwAdmin)
- **hiddenActionsNamesList** String, optional, default null. Is a list, comma separated, of gwAction name, which we want to hide in gwClassList Tab. This is useful when configuring many gwClassList tabs with different static filters, and it's needed to hide some actions, usually List Action, that works on incompatible (with current static filter) elements, and for this reason they have no sense in that context. This problematic can't be solved using standard DACL mechanism, and the only alternative without this parameter, will be duplicating the gwClass (and removing permissions for the specified gwActions)
- **customBeforeColumns** ArrayList<HashMap<String, String», optional, default null. Object [], each one containing all necessary infos to render a column. This columns are added BEFORE the base column set in gwAttribute List section in gwAdmin for the gwClass. Each column definition is made like this: { formatter: 'column_formatter', label: 'Header Label', width: 'auto' || 50, headerStyles: 'color: red;', styles: 'color: red;' } where 'formatter' is the name of the js function that generates the content (invoked with this arguments: 'item', 'rowIndex', 'cell'), 'label' is the column header text, 'width' can be an integer (px) or 'auto', 'headerStyles' e 'styles' are css rules applied respectively to header and cells.
- **customAfterColumns** ArrayList<HashMap<String, String», optional, default null. Object [], each one containing all necessary infos to render a column. This columns are added AFTER the base column set in gwAttribute List section in gwAdmin for the gwClass. Each column definition is made like this: { formatter: 'column_formatter', label: 'Header Label', width: 'auto' || 50, headerStyles: 'color: red;', styles: 'color: red;' }
- **onRowClickGwActionName** String, opzionale, default null. Se presente indica il nome di una azione Geoweb la quale verrà invocata (passando questi paramenti [/*Integer*/ rowIndex, /*Object*/ grid]) al click della riga della griglia. Ha priorità su onRowClickFunctionName e onRowClickFunction
- **onRowClickFunctionName** String, opzionale, default null. Se presente indica il nome di una funzione js la quale verrà invocata (passando questi paramenti [/*Integer*/ rowIndex, /*Object*/ grid]) al click della riga della griglia. Ha priorita su onRowClickFunction
- **onRowClickFunctionString** String, opzionale, default null. Se presente indica direttamente il codice js che verrà invocato (passando questi paramenti [/*Integer*/ rowIndex, /*Object*/ grid]) al click della riga della griglia.
- **insertIndex** Integer, optional, default null. If absent the new tab will be opened at the last

position

=== Parametri opzionali contenuti in options [**Deprecati**]: === * ***hideAttributesNamesList*** *String, opzionale, default null. @Deprecated, replaced by hiddenAttributesNamesList ==== Esempi ==== Esempi con codice minimale: <code javascript> var gwClassName = 'gwd_resource'; openGwClassListTab(gwClassName); </code> <code javascript> var gwClassName = 'gwd_resource'; var title = 'Risorse Operative';* optional openGwClassListTab(gwClassName, title); </code>

Esempio generale:

```
var gwClassName = 'gw_class_name';
var title = 'title';
var options = {
    //..
};
var tabWidget = openGwClassListTab(gwClassName, title , options );
```

Esempio generale con utilizzo di tutti i parametri. Utile per copiare velocemente uno scheletro con la struttura di base per poi eventualmente commentare i parametri non utilizzati:

```
var gwClassName = 'gwd_resource';
var title = 'Risorse Operative'; //optional
var options = { //optional
    filters: [
        {condition: 'AND', columnName: 'id_contract', operator: 'IN',
filterType: 'INTEGER', value: ['123', '456']}
    ],
    staticFilters: [
        {condition: 'AND', columnName: 'type_contract', operator: '=',
filterType: 'STRING', value: 'Forfait'}
    ]
    hideToolbar: false,
    hideNewIgnorePermissions: false,
    hideDeleteIgnorePermissions: false,
    hideActionsDropDownButtonColumn: false,
    hideIndirectSelectionColumn: false,
    forceListedAttributes: true,
    listedAttributesNames: 'attr_name_1,attr_name_2,attr_name_3',
    hiddenAttributesNamesList: 'attr_name_4,attr_name_5',
    hiddenActionsNamesList: 'action_name_1,action_name_2',
    customBeforeColumns: [
        { formatter: 'column_formatter', label: 'Header Label', width:
'auto' || 50, headerStyles: 'color: red;', styles: 'color: red;' }
    ],
    customAfterColumns: [
        { formatter: 'column_formatter', label: 'Header Label', width:
'auto' || 50, headerStyles: 'color: red;', styles: 'color: red;' }
    ],
    onRowClickGwActionName: 'gwAction_name',
    onRowClickFunctionName: 'name_in_js_global_name_space',
    onRowClickFunctionString: 'if(rowIndex==0)alert(\'click on row 0\');'
```

```
};
var tabWidget = openGwClassListTab(gwClassName, title , options);
```

Esempio con imposizione di un singolo filtro statico:

```
var gwClassName = 'gwd_resource';
var title = 'Risorse Operative'; //optional
var options = { //optional
    forceReplaceTab: true,
    staticFilters: [
        {condition: 'AND', columnName: 'type_contract', operator: '=',
filterType: 'STRING', value: ['Forfait']}
    ]
};
openGwClassListTab(gwClassName, title , options);
```

Esempio che utilizza la piu generare API javascript openTab(). Modalità d'uso supportata, ma @Deprecated. Utilizzare una delle modalità più specifiche descritte sopra.

```
var tabWidgetType = 'gwClassList';
var identifier = 'resources_list';
var tabWidgetId = createTabId(tabWidgetType, identifier);
var tabWidgetTitle = 'Risorse Operative';

var parametersMap = {
    forceReplaceTab: true,
    className: 'gwd_resource',
    staticFilters: [
        {condition:'AND', columnName:'type_contract',
        operator:'=', filterType:'STRING', value: ['Forfait']}
    ]
};
var tabWidget = openTab(tabWidgetId, tabWidgetType, tabWidgetTitle,
parametersMap);
```

Requisisti

Per funzionare il componente ha bisogno di questi vincoli:

- esistenza della classe con name uguale al parametro 'className' nei metadati di Geoweb
- gli attributi che devono comparire in lista devono essere correttamente configurati: anche il non corretto funzionamento di uno di essi può causare, il non funzionamento della griglia (in genere sono problematici gli attributi che hanno nella loro configurazione una query, o comunque hanno il concetto di campo in visualizzazione, fieldToShow, e campo da salvare, fieldToStore)

Note

In caso la stessa scheda sia apribile sia da menu di terzo livello (leafItem su xml di progetto) che, mettiamo, da un link di una gwHtmlReport, è bene che il name del leaftItem, che viene in automatico passato alla createTabId() sia uguale al gwClassName. Cosi tutti i comandi di apertura scheda cofluiranno di fatto sulla stessa scheda e non si avranno conflitti di id dojo. Se nei vari punti da cui viene aperta la gwClassList possono venire applicati filtri diversi alla lista è bene specificare il parametro forceReplaceTab a true.