Schede

In generale una componente scheda apre in Geoweb un tab specializzato nella visualizzazione/gestione di uno specifica set di dati. A seconda che il Progetto Geoweb dove ci troviamo sia di tipo 'GeoManager' o di tipo 'GeoExplorer', le schede potrebbero venire visualizzate con modalità diverse. Con il GeoManager vengono tutte aperte come tab nel tab container principale, mentre invece nel GeoExplorer, a seconda del tipo, potrebbero venire aperte in un apposito spazio, che lascia sempre in primo piano le schede del tab container principale (schede che in genere sono di tipo gwMap, ma non necessariamente). In genere tutte le schede hanno una barra header in alto, del colore del tema del framework. A seconda del tipo di scheda e del progetto di dove viene visualizzata, la barra header potrebbe essere più o meno spessa, potrebbe contenere o meno un titolo e potrebbe ospitare un apposito tasto per visualizzare la scheda a tutto schermo.

Una scheda può essere aperta in più maniere, che tipicamente sono:

- Azione JavaScript
- LeafItem (Menù)

Azione JavaScript

L'azione JavaScript può essere lanciata da più parti (vedi sezione [TODO MAKE LINK]). A seconda di dove viene lanciata un'azione si avrà un diverso set di argomenti disponibili in ingresso.

LeafItem

Un leafItem è una tipologia di menu di terzo livello che si può inserire nell'interfaccia utente di un progetto Geoweb.
Per inserire questa voce di menu bisogna aggiungere nell'xml di Progetto [TODO MAKE LINK] un tag leafItem [TODO MAKE LINK].
Un tag leafItem è sempre figlio di un tag accordionPaneItem [TODO MAKE LINK] di tipo leafItemContainer.
A determinare il tipo di scheda aperta sarà l'attributo type del leafItem.
Un leafItem tipicamente può avere come tag figli un set di tag parameter [TODO MAKE LINK]che ne vanno a personalizzare la modalità grafiche ed di funzionamento della scheda.

```
<accordionPaneItem name="accordionPaneItem_name" label="Label"
image="image.png" type="leafItemContainer">
    <leafItem name="test_gw_gantt" label="Test gwGantt"
image="test_gw_gantt.png" type="gwLeafItemType">
        <!-- inizio lista dei paramteri  -->
        ...
        <parameter name="parameter_name" value="parameter_value"
hideToClient="false"></parameter>
        ...
        <!-- fine lista dei paramteri  -->
    </leafItem>
</accordionPaneItem>
```

Elenco schede

Nome	Descrizione breve
gwAction	esegue il codice di un'azione (es. apertura scheda di dettaglio)
gwAttachmentsSheet	
gwClassDetail	
gwClassList	visualizzazione dei record di una classe di Geoweb in una lista «standard»
gwHtml	visualizzzazione del contenuto di un qualsiasi file html raggiungbile
gwHtmlReport	visualizzazione di una report in formato html
gwIndicatorsSheet	visualizzazione della pagina di cruscotto degli indicatori
gwMap	visualizzazione della «mappa cartografica»
gwPrefilteredList	visualizzazione dei record di una classe di Geoweb in una lista «prefiltrata» attraverso l'applicazione di una clausola «where»
gwReportsSheet	visualizzazione della pagina di cruscotto delle report (pdf, xls, ecc.)
gwHistoryTransactionSheet	
gwCalendar	visualizzazione del «calendario»
gwDynamicTimeLine	Il componente permette di visualizzare e gestire elementi appartenenti ad una generica classe
gwGantt	
gwTimeTable	(ex **gwTimeLine**) Agenda delle Risorse. Visualizza un calendario di 15 gg (durata configurabile) dove sono visualizzate le attività assegnate alle Risorse Operative. Il componente permette di accedere al dettaglio e procedere a modifica degli attributi di programmazione delle singole Attività
gwArchivedProcessList	Scheda per la visualizzazione delle informazioni delle istanze di processo terminate relative al processo
gwClassProcesses	visualizzazione della pagina di gestione del processo (work-flow) associato ad una classe di Geoweb
gwRunningProcessList	Scheda per la visualizzazione delle informazioni delle istanze di processo attive relative al processo
gwStartProcessInstanceAction	
gwStartProcessInstanceByMessageAction	
gwSuspendedProcessList	Scheda per la visualizzazione delle informazioni delle istanze di processo sospese relative al processo
gwTaskHierarchicalFilterMenu	
gwTaskDetail	Questa scheda permette di aprire una form contenente le informazioni relative ad un task
gwTaskList	Scheda per la visualizzazione dei task relativi alle istanze di processo
gwMnemonicCodeSheet	Scheda per la visualizzazione dei record organizzati gerarchicamente sulla base di un codice parlante
gwPhoto360	Scheda per la visualizzazione di una scheda gwPhoto360

Nome	Descrizione breve
gwOrgChart	Scheda per la visualizzazione/esportazione di grafici gerarchici
gwXlsxViewer	Scheda/floatingPane per la preview di file xlsx

Configurazione schede

gwAction

(torna a elenco schede)

Utilizzato per l'esecuzione del codice di una azione, presenta il seguente XML di configurazione (che dovrà essere incluso all'interno di un tag di tipo «accordion».

```
<leafItem name=" leafitem_name" label=" leafitem_label" image="
leafitem_image.png" type="gwAction">
      <parameter name="gwClassName" value="class_name"
hideToClient="false"></parameter>
      <parameter name="gwActionName" value="action_name"
hideToClient="false"></parameter>
</leafItem>
```

---

gwAttachmentsSheet

(torna a elenco schede)

---

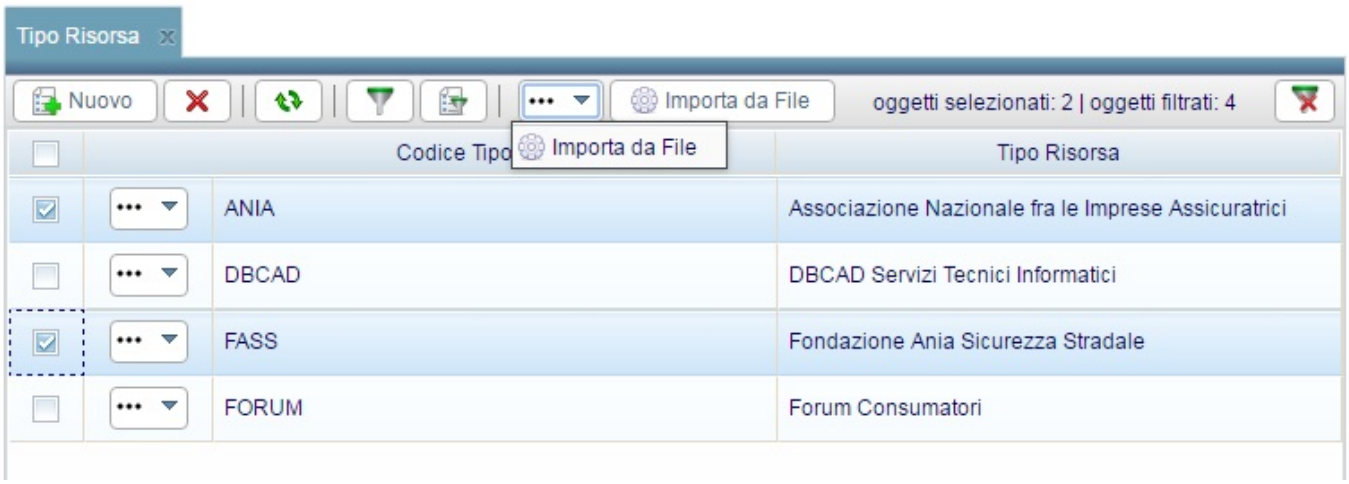
gwClassDetail

(torna a elenco schede)

---

gwClassList

(torna a elenco schede)



Con la scheda di tipo gwClassList viene aperta una griglia evoluta, che permette di visualizzare e gestire il contenuto dei record di una classe Geoweb. Termini equivalenti con i quali si fa riferimento alla griglia di Geoweb in questa guida sono: lista, grid, gwClassList.

PARAMETRI

Parametro	**Descrizione**
className	String, richiesto, il nome della classe Geoweb della quale si vuole aprire la lista dei record

filters	Object [], opzionale, default null, permette di imporre criteri di filtro per poter mostrare solo i record di dati aderenti. A differenza di *staticFilters*, questa lista di filtri PUO'essere rimossa dall'utente agendo sulla UI della scheda gwClassList. Esempio:
	`{`
	`condition: 'AND',`
	`columnName: 'id_item',`
	`attributeGwid: 1,`
	`operator: 'IN',`
	`filterType: 'INTEGER',`
	`value: [1, 2, 3]`
	`}`

staticFilters	Object [], opzionale, default null, permette di imporre criteri di filtro per poter mostrare solo i record di dati aderenti. A differenza di *filters*, questa lista di filtri NON PUO' essere rimossa dall'utente agendo sulla UI della scheda gwClassList. Esempio:
	`{`
	`condition: 'AND',`
	`columnName: 'status',`
	`attributeGwid: 1,`
	`operator: '=',`
	`filterType: 'STRING',`
	`value: 'OPN'`
	`}`

hideToolbar	Boolean , opzionale, default *false*, se true nasconde la Toolbar
hideNewIgnorePermissions	Boolean , opzionale, default *false*, se true nasconde dalla Toolbar il tasto Nuovo a prescindere dai permessi che ha il gruppo a cui appartiene l'utente su quella classe

Parametro	Descrizione
hideDeleteIgnorePermissions	Boolean , opzionale, default *false*, se true nasconde dalla Toolbar il tasto Cancella a prescindere dai permessi che ha il gruppo a cui appartiene l'utente su quella classe
hideActionsDropDownButtonColumn	Boolean, opzionale, default false. Determina se il button 'More Options' (AKA 'Tre puntini') sarà visibile o meno. Quando è valorizzato sovrascrive, localmente, l'eventuale configurazione globale per la specifica gwClass (o per tutte) che può essere impostata a livello di Progetto, nell'apposita sezione con tag <gwClassSettings>.
hideIndirectSelectionColumn	Boolean, opzionale, default false. Determina se la colonna della griglia adibita alla selezione delle singole righe sarà visibile o meno. Quando è valorizzato sovrascrive, localmente, l'eventuale configurazione globale per la specifica gwClass (o per tutte) che può essere impostata a livello di Progetto, nell'apposita sezione con tag <gwClassSettings>.
forceListedAttributes	Boolean , opzionale, default *false*, lavora insieme al parametro 'listedAttributesNames' se true mostra in griglia solo le colonne degli attributi il cui name è specificato nella lista listedAttributesNames
listedAttributesNames	ArrayList<String>, opzionale, default *null*, valutato solo se 'forceListedAttributes' vale true. Lista, separata da virgola, dei nomi degli attributi le cui colonne vogliamo compaiano in griglia
hideAttributesNamesList	ArrayList<String>, opzionale, default *null*,. Lista, separata da virgola, dei nomi degli attributi le cui colonne vogliamo non compaiano in griglia: vengono rimosse dagli attributi in lista impostati sulla classe. Viene valutato solo con hideAttributesNamesList diverso da true

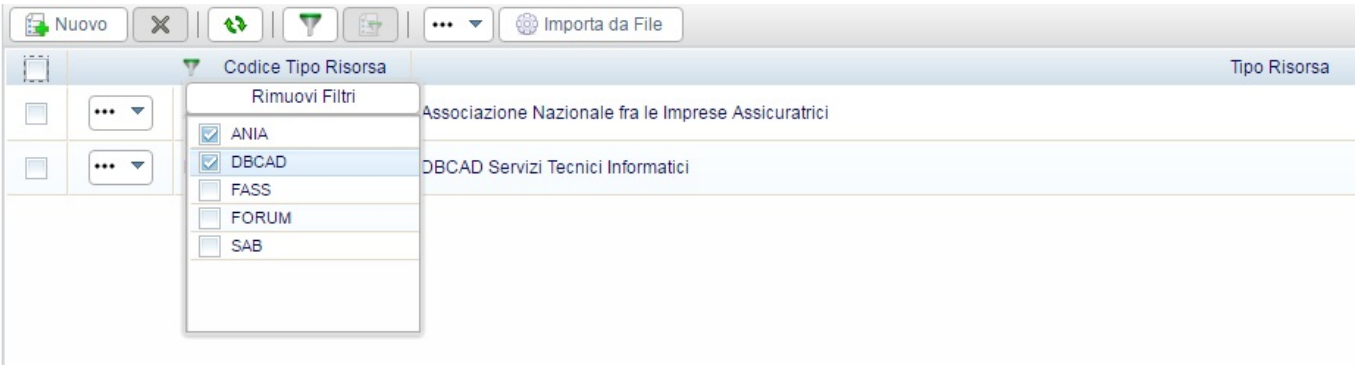
customBeforeColumns	ArrayList<HashMap<String, String», opzionale, default *null*. Lista (Array) di oggetti, ognuno dei quali contiene tutte le informazioni necessarie alla renderizzazione di tutte le celle di una colonna. Queste colonne vengono aggiunte PRIMA delle colonne degli attributi base in lista della classe. Ogni colonna è definita da oggetti del tipo:
	`{`
	`formatter: 'column_formatter', //nome della funzione js che genera il contenuto della cella (invocata con questi argomenti: 'item', 'rowIndex', 'cell')`
	`label: 'Header Label', //intestazione della colonna`
	`width: 50, //può essere un numero intero (px) o 'auto'`
	`headerStyles: 'color: red;', //sono regole css applicate all'intestazione della colonna`
	`styles: 'color: red;' //sono regole css applicate alle celle`
	`}`

Parametro	**Descrizione**

customAfterColumns	ArrayList<HashMap<String, String», opzionale, default *null*. Lista (Array) di oggetti, ognuno dei quali contiene tutte le informazioni necessarie alla renderizzazione di tutte le celle di una colonna. Queste colonne vengono aggiunte DOPO delle colonne degli attributi base in lista della classe. Ogni colonna è definita da oggetti del tipo:
	`{`
	`formatter: 'column_formatter', //nome della funzione js che genera il contenuto della cella (invocata con questi argomenti: 'item', 'rowIndex', 'cell')`
	`label: 'Header Label', //intestazione della colonna`
	`width: 50, //può essere un numero intero (px) o 'auto'`
	`headerStyles: 'color: red;', //sono regole css applicate all'intestazione della colonna`
	`styles: 'color: red;' //sono regole css applicate alle celle`
	`}`

onRowClickGwActionName	String, opzionale, default *null*. Se presente indica il nome di una gwAction appartenenete alla medesima gwClass, la quale verrà invocata (passando questi paramenti [/*Integer/ rowIndex, /*Object/ grid, /*Object/ item]) al click della riga della griglia. Ha priorità su onRowClickFunctionName
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, /*Object/ item]) al click della riga della griglia. Ha priorità su onRowClickFunctionString
onRowClickFunctionString	String, opzionale, default *null*. Se presente indica direttamente il codice js che verrà invocato (passando questi paramenti [/*Integer/ rowIndex, /*Object/ grid, /*Object/ item]) al click della riga della griglia
onRowClickFunction	String, opzionale, default *null*. @Deprecated. Rimpiazzato da onRowClickFunctionString. Se presente indica direttamente il codice js che verrà invocato (passando questi paramenti [/*Integer/ rowIndex, /*Object/ grid, /*Object/ item]) al click della riga della griglia
avoidColumnWidthAuto	Boolean, opzionale, default *false*. Avoid the 'auto' width setting upon column when related attributes are without and esplicit 'listWidth' (grid equally divide the remaining space among all with 'auto'. So grid is no allowed to increase it's width, and to show an horizontal scrollbar. This is problematic with many attributes in list, that are simpli not shown if the window isn't big enough) (disponibile dalla versione **4.6.10** issue #1140)

Nella parte superiore della scheda è presente una toolbar che, a seconda del set di permessi (statici e dinamici) impostati nel Geoweb Admin, mostra tutta una serie di bottoni, in ordine da sinistra a destra:



- **Nuovo**: visibile solo l'utente ha il permesso di inserimento, al click apre un dettaglio (su tab o floatingPane, a seconda della configurazione) con una form dove inserire i dati del nuovo record.
- **Cancella**: visibile solo l'utente ha il permesso di eliminazione, attivo solo con uno o più record selezionati, al click apre un dialog modale dove viene richiesta una conferma: in caso di risposta affermativa il record verrà elimato dal DB e rimosso dalla lista
- **Ricarica**: sempre mostrato, al click esegue il refresh del set di dati della lista.
- **Filtra**: sempre mostrato, al click apre un dialogo, non modale, che permette di scegliere criteri di filtro (che si aggiungono ad eventuali filtri statici impostati col parametro staticFilters) che andranno a filtrare i record mostrati in griglia. L'eventuale filtro impostato è rimovibile dal tasto 'Rimuovi Filtro' (vedi sotto) che compare a destra nella toolbar se un filtro è impostato.
- **Raffina**: sempre mostrato, attivo solo con uno o più record selezionati, al click crea un filtro a partire dalle righe selezionate (tramite l'apposito check a sinistra), mostrando solo quelle. L'eventuale filtro impostato è rimovibile dal tasto 'Rimuovi Filtro' (vedi sotto) che compare a destra nella toolbar se un filtro è impostato.
- **Azioni Utente**: bottone che al click apre un menu a tendina contenente tanti menuItem, i quali al click eseguono specifiche azioni, o aprono reportistiche. I menuItem report sono mostrati se esistone le relative report di classe. I menuItem di tipo azione sono mostrati se esistono le relative azioni e se sono attivi i relativi permessi di esecuzione per l'utente che sta visualizzando la griglia. In caso di nessuna azione e nessuna report configurate, il bottone non viene visualizzato.
- **Azioni in Toolbar**: numero variabile di bottoni, che al click eseguono specifiche azioni javascript, mostrati solo in presenza di azioni configurate e con l'utente in possesso dei relativi permessi di esecuzione.
- **Rimuovi Filtro**: mostrato solo in presenza di un filtro, non statico, impostato sulla lista. al click rimuove il filtro ed il tasto di conseguenza scompare.

Oltre ai bottoni relativi al filtro presenti sulla toolbar, se configurati possono esserci dei tastini detti '**filtri di colonna**' [TODO MAKE LINK]. Al click viene aperto un menu a tendina, con una serie di check che attivano/disattivano i relativi filtri (messi in AND ad eventuali filtri di qualunque tipo già presenti), ed un tasto 'Rimuovi Filtri' (rimuove tutti e solo i filtri impostati dal quel particolare filtro di colonna).



Su ogni riga della griglia, nella prima colonna a sinistra, è disponibile un check, che serve per selezionare la riga, ed attivare una serie di funzioni (come affina, elimina). Nella seconda colonna è disponibile un bottone con le Azioni Utente, che permette di eseguire le azioni e lanciare le report esplicitamente configurate per operare sul singolo record. Anche qui, se non sono configurate ne azioni ne report per il record singolo, l'intera colonna con i tasti Azioni Utente non viene mostrata.

Le colonne successive sono generate in automatico, a partire dagli attributi configurati nel Geoweb

Admin. Si può impostare l'ordine e la larghezza della singola colonna. Lo stile degli header di colonna e delle celle possono essere anche esso ampiamente customizzato dall'xml di configurazione del tipo di controllo (widget) associato all'attributo. Nella sezione specifica sono esposti ulteriori dettagli [TODO MAKE LINK].

Al click sui record della griglia si apre il dettaglio della classe. La modalità di apertura di default è floatingPane. questa modalità può essere sovrascritta dall'xml dei Progetto Geoweb, impostandola su tab [TODO MAKE LINK]

Requisiti
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.

JavaScript
Consulta l'apposita sezione di javascrip API apertura scheda lista classe (gwClassList)

LeafItem

```
<leafItem name="resources_list" label="Resources" image="resources.png"
type="gwClassList">
    <parameter name="className" value="gwd_resource"
hideToClient="false"></parameter>
    <parameter
        name="staticFilters"
        value="[{ condition: 'AND', columnName: 'cod_resource', operator:
'IN', filterType: 'INTEGER', value: [1, 2, 3] }]"
        hideToClient="false">
    </parameter>
</leafItem>
```

Personalizzazione Stile degli ATTRIBUTI in LISTA

E' possibile personalizzare lo stile grafico dei controlli (widget) in lista. Si può intervenire sia sullo stile di visualizzazione delle celle che contengono i controlli, sia sugli header di colonna.



Task Name	Project	Start Time	Duration	Percentage (%)	Previous Task	Task Owner	Parent Task
Analysis	Development Project	18-10-2012	40	80	Requirement	Michael	
Deliver	Development Project	18-11-2012	100	30	Implementation	Michael;Jason	
Design	Development Project	18-10-2012	60	80	Requirement	Jason	
Detail design	Development Project	18-10-2012	30	50	Requirement	Michael	
Detail Implement	Development Project	17-11-2012	120	50	Prototype	Jason	
Development doc	Development Project	20-10-2012	20	10	Requirement	Rock;Jack	
Implementation	Development Project	06-11-2012	30	80	Sketch	Jason	
Prototype	Development Project	06-11-2012	60	80	Sketch	Rock	
Requirement	Development Project	11-10-2012	50	50		Jack	

Questi parametri sono trasversali a tutti i tipi di controllo (widget) in Geoweb, e sono configurabili dall'xml del widget dell'attributo.

Parametri:

- “**listCellTextAlign**”, String, valori ammessi: 'center', 'left', 'right'. Imposta in modo rapido l'allineamento del valore del widget dentro la cella. Sovrascrive le eventuali impostazioni di base che avrebbe il widget (es: 'center' è il default dei widget 'Date' in lista)
- “**listCellStyleRules**”, String, valori ammessi: qualsiasi regola css. Le regole css vengono applicate alle celle che ospitano il widget
- “**listCellHeaderStyleRules**”, String, valori ammessi: qualsiasi regola css. Queste regole di stile vengono applicate all'header della colonna del widget
- “**listCellClass**”, String, valori ammessi: qualsiasi nome di classe css. Le regole css definite dalla classe vengono applicate alle celle. La regola css deve essere già pre-esistente fra quelle base (od essere stata in qualche modo aggiunta tramite i meccanismi dei plugin)

Esempio di configurazione su un xml di un widget WidgetBase (Normal):

```
<WidgetBase>

    ...

    <listCellTextAlign>

        right

    </listCellTextAlign>

    <listCellStyleRules>

        color: red; background: #AABBAA;

    </listCellStyleRules>
```

```
    <listCellClass>

        cssClassName1 cssClassName2

    </listCellClass>

    <listCellHeaderStyleRules>

        color: green; background: #CCCCAA;

    </listCellHeaderStyleRules>

    ...

</WidgetBase>
```

---

gwHtml

(torna a elenco schede)

Con la scheda di tipo gwHtml è possibile mostrare in Geoweb il contenuto di un qualsiasi file html.

- Parametri
- Requisiti
- JavaScript
- LeafItem

Parametri

- **htmlPath**: obbligatorio, il path del file .html, comprensivo di estensione, a partire dalla cartella WEB/html.

Requisiti

Per funzionare il componente ha bisogno di questi vincoli:

- esistenza cartella base WEB/html nei contenuti statici
- esistenza file .html sotto la cartella base

JavaScript

```
var tabWidgetType = 'gwHtml';
var identifier = 'cruscotto_svg';
var tabWidgetId = createTabId(tabWidgetType, identifier);
var tabWidgetTitle = 'Cruscotto';

var parametersMap = {
    forceReplaceTab: true,
    htmlPath: 'cruscotto_svg.html'
};
```

```
openTab(tabWidgetId, tabWidgetType, tabWidgetTitle, parametersMap);
```

LeafItem

```
<leafItem name="cruscotto" label="Cruscotto" image="cruscotto.png"
type="gwHtml">
    <parameter name="htmlPath" value="cruscotto_svg.html"
hideToClient="false"></parameter>
</leafItem>
```

---

gwHtmlReport

(torna a elenco schede)

Con la scheda di tipo gwHtmlReport è possibile mostrare in Geoweb il contenuto di di un file html generato tramite le API java di *JasperReport*, a partire da un file **.jasper** e posizionato in maniera opportuna.

PARAMETRI

Parametro	Descrizione
reportUrl	String, Richiesto, path relativo, comprensivo di estensione **.jasper**, a partire dalla cartella **WEB/template**.
parametersMap	Object, opzionale, contiene ulteriori parametri di configurazione
title	String, opzionale, Titolo per la scheda
insertIndex	Integer, opzionale. Indice della posizione dove verrà aperta la scheda. Se omesso la scheda si apre in ultima posizione a destra

PARAMETRI parametersMap

Parametro	Descrizione
tabWidgetId	String, optional, default null. When provided is used as tabWidget id. Configuring it allow to open report on the already opened tab when needed, basing on tab id check
reportParameters	Object, optional, default null. This map contains the parameter that will be available inside the report
gwClassName from 4.4.18	String, optional, default null. If omitted the eventual gwClass scope will not to be applied
className pre 4.4.18	String, optional, default null. Like gwClassName
updateOnChangeInGwClassNameListString	String, optional, default null. Comma separated (',') gwClassNames. This parameter decides to which gwClass publish events will cause a gwHtmlReport tab refresh (insert/update/delete). So when user modifies a gwClassDetail of a specific gwClass, if updateOnChangeInGwClassNameListString contains the specific gwClassName, the tab will automatically refresh itself.

REQUISITI Per funzionare il componente ha bisogno di questi vincoli:

- esistenza cartella base WEB/template nei contenuti statici
- esistenza file **.jasper** individuato dal parametro *reportUrl* sotto la cartella base

JAVASCRIPT Il jascript può essere inglobato in gwAction, o dentro le le report stesse per creare un vero e proprio menu di navigazione.

La UI *gwHtmlReport* può essere aperta in una scheda tramite la js api **openGwHtmlReportTab()** descritta qui.

La UI *gwHtmlReport* può anche essere aperta in un *floatingPane* tramite la js api **openGwHtmlReportFloatingPane()** descritta qui.

N.B. reportUrl: percorso a partire da «WEB/template/» dei contenuti statici.

Codice minimale **dalla 4.4.18**

example.js

```
var reportUrl = 'path/To/file_name.jasper';
openGwHtmlReportTab(reportUrl);
```

Codice minimale **pre 4.4.17**

example.js

```
var reportUrl = 'path/To/file_name.jasper';
openHtmlReport(reportUrl);
```

Esempio con utilizzo di tutti i parametri, anche opzionali **dalla 4.4.18**:

example.js

```
var reportUrl = 'path/To/file_name.jasper';
var parametersMap = {
    //tabWidgetId: 'custom_tab_id_'+item.cod_entity,
    reportParameters: {
        date: new Date(),
        project: 'P01'
    },
    gwClassName: 'scoped_class_name',
    updateOnChangeInGwClassNameList: 'class_name_1'
};
var title = 'title';
var gwHtmlReportTab = openGwHtmlReportTab(reportUrl, parametersMap,
title);
```

Esempio con utilizzo di tutti i parametri, anche opzionali **pre 4.4.17**:

example.js

```
var reportUrl = 'path/To/file_name.jasper';
var parametersMap = {
    //tabWidgetId: 'custom_tab_id_'+item.cod_entity,
    reportParameters: {
        date: new Date(),
        project: 'P01'
    },
    gwClassName: 'scoped_class_name',
    updateOnChangeInGwClassNameList: 'class_name_1'
};
var title = 'title';
var gwHtmlReportTab = openHtmlReport(reportUrl, parametersMap, title);
```

LEAFITEM

```
<leafItem name="cruscotto_html_report" label="HTML Report"
image="html_report.png" type="gwHtmlReport">
    <parameter name="reportUrl" value="path/To/file_name.jasper"
hideToClient="false"></parameter>
    <parameter name="reportParameters" value="{param1: 'param1', param2: 2}"
hideToClient="false"></parameter>
    <parameter name="gwClassName" value="class_name_1"
hideToClient="false"></parameter>
    <parameter name="updateOnChangeInGwClassNameList"
value="class_name_1,class_name_2" hideToClient="false"></parameter>
</leafItem>
```

---

gwIndicatorsSheet

(torna a elenco schede)

Utilizzato per la visualizzazione della pagina di cruscotto degli indicatori, presenta il seguente XML di configurazione (che dovrà essere incluso all'interno di un tag di tipo «accordion».

```
<leafItem name="leafitem_name" label="leafitem_label"
image="leafitem_image.png" type="gwIndicatorsSheet">
     <parameter name="NAME" value="Indicatori"></parameter>
     <parameter name="VISUALIZATION_GROUPS"
value="groupV1,...,groupVn"></parameter>
     <parameter name="MANAGER_GROUPS"
value="groupM1,...,groupMn"></parameter
</leafItem>
```

gwMap

(torna a elenco schede)

Utilizzato per la visualizzazione di una mappa cartografica, presenta il seguente XML di configurazione (che dovrà essere incluso all'interno di un tag di tipo «accordion».

```
<leafItem name="leafitem_name" label="leafitem_label"
image="leafitem_image.png" type="gwMap">
     <parameter name="mgMapName" value="mappa_name_mgos"
hideToClient="false"></parameter>
</leafItem>
```

gwPrefilteredList

(torna a elenco schede)

Utilizzato per la visualizzazione dei record di una classe di Geoweb pre-filtrati attraverso l'applicazione di una clausola «where», presenta il seguente XML di configurazione (che dovrà essere incluso all'interno di un tag di tipo «accordion»).

```
 <leafItem name= "leafitem_name" label= "leafitem_label"
image="leafitem_image.png" type="gwPrefilteredList">
     <parameter name="className" value="class_name"></parameter>
     <parameter name="MENU_TITOLO" value="title_value"></parameter>
     <parameter name="DETAIL_CLASS_NAME" value=""></parameter>
     <parameter name="QUERY" value="select field1,…,fieldN from tablel_name
where fieldX='vsl1'"></parameter>
     <parameter name="OrderBy" value="field_orderby"></parameter>
     <parameter name="DETAIL_CLASS_NAME" value=""></parameter>
     <parameter name="PK_RECORD_FIELD" value="pk_class"></parameter>
     <parameter name="DATASOURCE" value="datasource_name"></parameter>
     <parameter name="PX_HIGHT" value=""></parameter>
     <parameter name="USER_GROUPS" value="group1,…,groupN"></parameter>
     <parameter name="ColumnsViews" value="field1,…,fieldN"></parameter>
     <parameter name="ColumnsLabel" value="label1,…,labelN"></parameter>
     <parameter name="ColumnsWidth" value="auto,…,auto"></parameter>
</leafItem>
```

Attrbiuti del *leafitam* nell'xml:

- **name**: nome del leaf item;
- **label**: etichetta del leaf item visualizzata nel menù di II° livello;
- **image**: nome dell'immagine visualizzata nel menù di II° livello (../WEB/images);
- **type**: tipo di leafitem (*gwPrefilteredList* per l'esempio corrente).

Attributi dei *parameter* nell'xml:

- **className** : nome della classe di Geoweb;
- **MENU_TITOLO**: valore da visualizzare come titolo della lista;
- **QUERY**: query per la selezione dei record da visualizzare nella lista;
- **OrderBy**: nome del campo per cui effettuare l'ordinamento della lista;
- **PK_RECORD_FIELD**: nome del campo “chiave primaria”;
- **DATASOURCE**: nome del datasource (schema dati);
- **USER_GROUPS**: elenco dei gruppi utenti che possono visualizzare la lista;
- **ColumnsViews**: elenco dei campi da visualizzare come colonne della lista;
- **ColumnsLabel**: elenco delle etichette da visualizzare come intestazione delle colonne della lista;
- **ColumnsWidth**: larghezza (px) delle colonne della lista (default :auto).

---

gwReportsSheet

(torna a elenco schede)

Utilizzato per la visualizzazione della pagina di cruscotto delle report (formato PDF,XLS), presenta il seguente XML di configurazione (che dovrà essere incluso all’interno di un tag di tipo «accordion».

```
<leafItem name="report_name" label="leafitem_label"
image="leafitem_image.png" type="gwReportsSheet">
      <parameter name="NAME" value="report_name"></parameter>
      <parameter name="VISUALIZATION_GROUPS"
value="groupV1,...,groupVn"></parameter>
      <parameter name="MANAGER_GROUPS"
value="groupM1,...,groupMn"></parameter>
</leafItem>
```

N.B. il valore assegnato a leafitem name deve coincidere con il valore del parametro NAME

---

gwHistoryTransactionSheet

(torna a elenco schede)

---

gwCalendar

(torna a elenco schede)

Utilizzato per la visualizzazione della pagina del calendario

Parametri

- **calendarCodes** String, array: comma separated values

- **eventClassName** String, array: comma separated values. parallel array to calendarCodes (each n position refers to nth calendarCodes's one)
- **filters** Object or Object[], optional removable filters. Inside filters columnName property are available some predefined placeholders: '#id_event#', '#cod_event#', '#summary#', '#place#', '#start_date#', '#end_date#', '#notes#', '#notes#', '#cod_calendar#', '#cod_event_status#', '#cod_priority#', '#cod_event_type#'

This can be used to make filters shared with different gwClasses implementing the event interface (each one that may overrides some field through a specific mapping) Simple filter based on a specific (=gwClass related) columnName, for example evt_id_event instead of the default id_event still will works, but gwClass cross compatibility may not be granted. Ex:

```
{condition: 'AND', columnName: '#cod_event_status#', operator: '=',
filterType: 'STRING', value: 'CLO'}
```

- **staticFilters** Object or Object[], optional fixed filters. Inside filters columnName property are available some predefined placeholders: '#id_event#', '#cod_event#', '#summary#', '#place#', '#start_date#', '#end_date#', '#notes#', '#notes#', '#cod_calendar#', '#cod_event_status#', '#cod_priority#', '#cod_event_type#'

This can be used to make filters shared with different gwClasses implementing the event interface (each one that may overrides some field through a specific mapping) Simple filter based on a specific (=gwClass related) columnName, for example evt_id_event instead of the default id_event still will works, but gwClass cross compatibility may not be granted. Ex:

```
{condition: 'AND', columnName: '#cod_event_type#', operator: '=',
filterType: 'STRING', value:  'ETP3'}
```

- **minHours** Integer, optional, default 0. min hours shown is day view
- **maxHours** Integer, optional, default 23. Max hours shown is day view
- **date** Date, optional: ex: value="new Date(2015, 26, 3)". If not set today is used. new Date() takes 3 parameters, in order: year, month, day
- **dateInterval** String, optional, default 'week'. possible options: 'day', 'week', 'month'
- **dateIntervalSteps** Integer, optional, default 5. n days from the reference date

Gli ultimi 5 parametri sono ripresi direttamente fra quelli esposti da dojo, ed influiscono sulla grafica della scheda Calendario. Sono opzionali, e in caso di omissione sono utilizzati dei valori di default. Essi agiscono come descritto nella documentazione ufficiale visualizzabile qui.

Requisisti

Tabelle necessarie al funzionamento (nello schema dei dati):

- **evt_calendar**
- **evt_class_columns_map** Contiene, tra le altre, le seguenti colonne utilizzate dalla scheda gwCalendar:
  - class_name character varying(30)
  - id_event character varying(30)
  - summary character varying(30)
  - place character varying(30)
  - start_date character varying(30)
  - end_date character varying(30)

- notes character varying(30)

Nella tabella **evt_class_columns_map** vanno dichiarati tanti record quanti sono le classi che vogliamo siano visualizzabili nel calendario (e gestite dall'event memo scheduler). Queste classi devono implementare l'interfaccia definita dalla gwClass evento base di geoweb: **evt_event**. La classe evt_event è gestita in automatico e quindi non va ci va creato un record sopra. evt_event ha i seguenti campi fissi:

- **id_event** character varying(30)
- **summary** character varying(30)
- **place** character varying(30)
- **start_date** character varying(30)
- **end_date** character varying(30)
- **notes** character varying(30)

Nel popolare i record di evt_class_columns_map i campi fissi della evt_event possono essere rimappati nei nomi specifici della tabella che ospita i record con le connotazioni di evento. In caso i nomi siano i medesimi, non c'è bisogno di rimappare nulla. La unica informazione da specificare, obbligatoria, è il campo **class_name**.

JavaScript
Consulta l'apposita sezione di javascrip API apertura scheda gwCalendar

LeafItem
Presenta il seguente XML di configurazione (che dovrà essere incluso all'interno di un tag di tipo «accordion».

```
<leafItem name="calendar_0" label="Calendario" image="asm_configuration.png"
type="gwCalendar">

     <!--lista codici calendario, separati da virgola-->
     <parameter name="calendarCodes" value="ECL1,ECL2,ECL3,ECL4"
hideToClient="false"></parameter>

     <!--lista nomi classi, separati da virgola l ordine è importante, e va
correlato a quello dei valori di calendarCodes I dettagli degli eventi con
un certo codice calendario, che hanno un indice nell array calendarCodes,
verranno aperti con il dettaglio della classe con className di stesso indice
nell array eventClassName -->
     <parameter name="eventClassName"
value="eventi_ECL1,eventi_ECL2,eventi_ECL3,eventi_ECL4"
hideToClient="false"></parameter>

     <!-- new Date() takes 3 parmameter, in order: year, month, day.
Optional: if not set today is used. -->
     <parameter name="date" value="new Date(2015, 3, 26)"
hideToClient="false"></parameter>

     <!--possible options: day, week, month. Default value is day-->
     <parameter name="dateInterval" value="day"
hideToClient="false"></parameter>
```

```
     <!-- if dateInterval= day - n days from the reference date, 7 default-
->
     <!-- if dateInterval= week - n week from the first day of week that
contains the reference date, 5 default-->
     <!-- if dateInterval= month - n months from the first day of the month
that contains the reference date, 1 default-->
     <parameter name="dateIntervalSteps" value="2"
hideToClient="false"></parameter>

     <parameter name="hourSize" value="60" hideToClient="false"></parameter>
     <!-- The desired size in pixels of one hour, 100 default -->

     <!-- The duration of minutes of the time slot (must be a divisor of
60), 15 default -->
     <parameter name="timeSlotDuration" value="15"
hideToClient="false"></parameter>

     <!-- in range [0, 23], 8 default-->
     <parameter name="minHours" value="8" hideToClient="false"></parameter>

     <!-- in range [1, 36], 16 default-->
     <parameter name="maxHours" value="18" hideToClient="false"></parameter>

</leafItem>
```

Codice minimale:

```
<leafItem name="calendar_0" label="Calendario" image="asm_configuration.png"
type="gwCalendar">
     <parameter name="calendarCodes" value="ECL1,ECL2,ECL3,ECL4"
hideToClient="false"></parameter>
     <parameter name="eventClassName"
value="eventi_ECL1,eventi_ECL2,eventi_ECL3,eventi_ECL4"
hideToClient="false"></parameter>
</leafItem>
```

---

gwDynamicTimeLine

(torna a elenco schede)

Il componente permette di visualizzare e gestire elementi appartenenti ad una generica classe (di seguito **baseGwClass**, per esempio la classe *intervento*).

La visualizzazione avviene con una linea del tempo in sottofondo, mentre la gestione permette di modificare, oltre i parametri temporali, qualsiasi caratteristica degli elementi (sempre al netto delle policy sui permessi di Geoweb).

Gli elementi verranno visualizzati nel piano della timeline, con nell’ascissa il tempo e nell’ordinata una

o più criteri di raggruppamento (massimo 2) derivanti da altre classi, che determineranno parzializzazioni dei dati. Un terza classe collegata può essere specificata e venire visualizzata nel piano della timeline arricchendo le informazioni visualizzate

Queste altre classi saranno collegate alla *baseGwClass* da apposite chiavi esterne, i cui attributi coinvolti saranno specificati nella configurazione (per esempio le classi collegate potrebbero essere *risorsa*, *oggetto*, *attività*).

Per la *baseGwClass* si dovranno specificare anche i nomi degli attributi di geoweb (uno o due) che referenziano i campi sul DB contenenti l'informazione temporale dell'entità (uno per entità puntuali, due per entità che vivono in un certo arco temporale).

Una delle classi collegate potrà essere eletta a principale (**mainGwClass**): le sue informazioni verranno visualizzate sugli item del piano della timeline.

Una delle classi collegate potrà essere eletta come quella da usare come primo criterio di raggruppamento per le ordinate (**groupGwClass**).

Una delle classi collegate potrà essere eletta come quella da usare come secondo criterio di raggruppamento per le ordinate (**nestedGroupGwClass**), innestato nel criterio di gerarchizzazione principale.

C'è la possibilità di cambiare direttamente nell'interfaccia, tramite drag&drop, il ruolo delle classi collegate alla classe base, scambiando a piacimento i ruoli (**mainGwClass**, **groupGwClass**, **nestedGroupGwClass**).

Al doppio click su un qualsiasi elemento della nel piano della timeLine verrà aperto il corrispettivo dettaglio di classe: *baseGwClass*.

È possibile aprire la timeline con dei filtri sia statici che dinamici, su ognuna delle classi coinvolte.

I filtri ambito e DACL sono impostati in automatico per ognuna delle entità rappresentate.

C'è la possibilità di abilitare l'aggregamento automatico dei dati sul piano della timeline (**aggregationEnabled**=*true*), potendo scegliere il numero di elementi massimo oltre il quale i dati vengono presentati in maniera aggregata (**aggregationThreshold**). La soglia è applicata a porzioni di timeline computate dinamicamente (se ad una certa scala vedo come arco temporale un mese, i dati verranno aggregati per giorni). Al doppio click sui dati aggregati verrà aperta una gwClassList con applicato un filtro per i soli record interessati.

Esiste la possibilità di impostare al momento dei filtri per tutte le classi coinvolte.

PARAMETRI

- **tabWidgetId**: String, optional, default null. Se fornito è l'id che verrà assegnato al tab (e con il quale si potrà in seguito fare riferimento). Se omesso ne verrà generato uno in automatico del tipo 'gwDynamicTimeLine_tab_1576518200437'
- **title**: String, optional, default "Exported Data". Dovrebbe essere uguale al title del tab. E' usato come nome del file xlsx esportato
- **baseGwClass**: Object, required. Object javascript contenente le informazioni della gwClass che detiene i campi temporali per la rappresentazione sulla timeline.
  Parametri:
  - **gwClassName**, String, required. Name della baseGwClass;

    - **startAttrName**, String, required. Name dell'attributo che contiene la data di inizio dell'evento della classe base;
    - **endAttrName**, String, optional. Name dell'attributo che contiene la data di fine dell'evento della classe base. Se omesso gli eventi si intendono puntuali e sono comunque visualizzati sulla timeline;
    - **codAttrName**: String, required. Name dell'attributo della baseGwClass che contiene il codice;
    - **codToShowAttrName**: String, opzionale. Name dell'attributo della baseGwClass che contiene il codice da mostrare. Se omesso al suo posto viene visualizzato il valore di *codAttrName*;
    - **visualizationAttrName**: String, optional. Name dell'attributo della baseGwClass che contiene la descrizione;
    - **staticFilters**: Object[], optional;
    - **filters**: Object[], optional;
    - **order**: String o function, optional. Quando String è il name dell'attributo della *gwClass che per cui va eseguito l'ordinamento. Si può specificare una modalità di ordinamento ascendente o discendente, anteponendo il prefisso '+' [ASC] od il prefisso '-' [DESC]. Il default per l'ordinamento è '+'.
    - **applyOrderInStackedItems**: Boolean, default false. Questo verrà valutato solo in presenza di order non nullo, ed influirà sull'ordinamento dello stack degli item solo quando la gwClass di competenza si trovi sul piano della timeline (in quanto temporaneamente maiGwClass od, in assenza, baseGwClass)
- **groupGwClass**: Object, optional. Object javascript contenente le informazioni della gwClass da usare come primo criterio di raggruppamento nelle ordinate.
  Parametri:
    - **gwClassName**: String, required. Name della baseGwClass;
    - **referencingCodAttrName**: String, required. Name dell'attributo della baseGwClass che referenzia la groupGwClass;
    - **codAttrName**: String, required. Name dell'attributo della groupGwClass referenziato dalla referencingCodAttrName;
    - **codToShowAttrName**: String, opzionale. Name dell'attributo della groupGwClass che contiene il codice da mostrare. Se omesso al suo posto viene visualizzato il valore di *codAttrName*;
    - **visualizationAttrName**: String, required. Name dell'attributo della groupGwClass che contiene le informazioni da mostrare;
    - **staticFilters**: Object[], optional, default [];
    - **filters**: Object[], optional, default [];
    - **nestedGroupFilters**: Object, optional, default *null*. Contiene i filtri statici che verranno applicati in automatico qualora questa gwClass si trovasse (fin dall'inizio, o tramite swap) in posizione di *nestedGrupoGwClass*. L'oggetto contiene una o due chiavi, con nomi uguali alla(e) altra(e) gwClass, che (fin dall'inizio, o tramite swap) potrebbe(ro) trovarsi ad essere in posizione di *groupGwClass*. I valori corrispondenti alle chiavi saranno la lista dei filtri che verranno applicati quando per ogni elemento group verrà fatta la query per ottenere gli elementi del nestedGroup. Nei filtri si potrà far riferimento ai valori correnti del record group padre tramite la notazione: *#{map.cod_campo_group_gw_class}*. Potranno essere inoltre impostati filtri di tipo **CUSTOM**, per poter inserire operatori di I**N/NOT IN** che operano su risultati di query **SELECT** innestate.
      Esempio:

```
nestedGroupFilters: {
    res_resource: [],
```

```
    oma_activity_to_plan: [
        {
            condition: 'AND',
            columnName: 'cod_object',
            filterType: 'CUSTOM',
            value: '(select
oma_activity_to_plan.cod_object_program from oma_activity_to_plan
where cod_object_program=#{map.cod_object_program})'
        }
    ]
}
```

  - **order**: String o function, optional. Quando String è il name dell'attributo della *gwClass che per cui va eseguito l'ordinamento. Si può specificare una modalità di ordinamento ascendente o discendente, anteponendo il prefisso '+' [ASC] od il prefisso '-' [DESC]. Il default per l'ordinamento è '+'.
  - **applyOrderInStackedItems**: Boolean, default false. Questo verrà valutato solo in presenza di order non nullo, ed influirà sull'ordinamento dello stack degli item solo quando la gwClass di competenza si trovi sul piano della timeline (in quanto temporaneamente maiGwClass od, in assenza, baseGwClass)
- **nestedGroupGwClass**: Object, optional. Object javascript contenente le informazioni della gwClass da usare come secondo criterio di raggruppamento nelle ordinate. Valuto solo in presenza del parametro *groupGwClass*.
  Parametri:
  - **gwClassName**: String, required. Name della *nestedGroupGwClass*;
  - **referencingCodAttrName**: String, required. Name dell'attributo della baseGwClass che referenzia la *nestedGroupGwClass*;
  - **codAttrName**: String, required. Name dell'attributo della *nestedGroupGwClass* referenziato dalla *referencingCodAttrName*;
  - **codToShowAttrName**: String, opzionale. Name dell'attributo della *nestedGroupGwClass* che contiene il codice da mostrare. Se omesso al suo posto viene visualizzato il valore di *codAttrName*;
  - **visualizationAttrName**: String, required. Name dell'attributo della *nestedGroupGwClass* che contiene le informazioni da mostrare;
  - **staticFilters**: Object[], optional, default [];
  - **filters**: Object[], optional, default [];
  - **nestedGroupFilters**: Object, optional, default null. Contiene i filtri statici che verranno applicati in automatico qualora questa gwClass si trovasse (fin dall'inizio, o tramite swap) in posizione di *nestedGrupoGwClass*. L'oggetto contiene una o due chiavi, con nomi uguali alla(e) altra(e) gwClass, che (fin dall'inizio, o tramite swap) potrebbe(ro) trovarsi ad essere in posizione di groupGwClass. I valori corrispondenti alle chiavi saranno la lista dei filtri che verranno applicati quando per ogni elementi group verrà fatto la query per ottenere gli elementi del nestedGroup. Nei filtri si potrà far riferimento ai valori correnti del record group padre tramite la notazione: *#{map.cod_campo_group_gw_class}*. Potranno essere inoltre impostati filtri di tipo **CUSTOM**, per poter inserire operatori di **IN/NOT IN** che operano su risultati di query **SELECT** innestate.
    Esempio:

```
nestedGroupFilters: {
    res_resource: [],
    oma_activity_to_plan: [
```

```
        {
            condition: 'AND',
            columnName: 'cod_object',
            operator: 'IN',
            filterType: 'CUSTOM',
            value: '(select
oma_activity_to_plan.cod_object_program from oma_activity_to_plan
where cod_object_program=#{map.cod_object_program})'
        }
    ]
}
```

  - **order**: String o function, optional. Quando String è il name dell'attributo della *gwClass che per cui va eseguito l'ordinamento. Si può specificare una modalità di ordinamento ascendente o discendente, anteponendo il prefisso '+' [ASC] od il prefisso '-' [DESC]. Il default per l'ordinamento è '+'.
  - **applyOrderInStackedItems**: Boolean, default false. Questo verrà valutato solo in presenza di order non nullo, ed influirà sull'ordinamento dello stack degli item solo quando la gwClass di competenza si trovi sul piano della timeline (in quanto temporaneamente maiGwClass od, in assenza, baseGwClass)
- **mainGwClass**: Object, optional. Object javascript contenente le informazioni della gwClass da usare per essere visualizzata nel piano nella timeline, recuperando i parametri temporali dalla baseGwClass.
  Parametri:
  - **gwClassName**: String, required. Name della baseGwClass;
  - **referencingCodAttrName**: String, required. Name dell'attributo della baseGwClass che referenzia la *mainGwClass*;
  - **codAttrName**: String, required. Name dell'attributo della *mainGwClass* referenziato dalla *referencingCodAttrName*;
  - **codToShowAttrName**: String, opzionale. Name dell'attributo della *mainGwClass* che contiene il codice da mostrare. Se omesso al suo posto viene visualizzato il valore di *codAttrName*;
  - **visualizationAttrName**: String, required. Name dell'attributo della *mainGwClass* che contiene le informazioni da mostrare;
  - **staticFilters**: Object[], optional, default [];
  - **filters**: Object[], optional, default [];
  - **nestedGroupFilters**: Object, optional, default null. Contiene i filtri che statici che verranno applicati in automatico qualora questa gwClass si trovasse (fin dall'inizio, o tramite swap) in posizione di *nestedGrupoGwClass*. L'oggetto contiene una o due chiavi, con nomi uguali alla(e) altra(e) gwClass, che (fin dall'inizio, o tramite swap) potrebbe(ro) trovarsi ad essere in posizione di *groupGwClass*. I valori corrispondenti alle chiavi saranno la lista dei filtri che verranno applicati quando per ogni elementi group verrà fatto la query per ottenere gli elementi del *nestedGroup*. Nei filtri si potrà far riferimento ai valori correnti del record group padre tramite la notazione:
    #{map.cod_campo_group_gw_class}.
    Potranno essere inoltre impostati filtri di tipo **CUSTOM**, per poter inserire operatori di **IN**/**NOT IN** che operano su risultati di query **SELECT** innestate.
    Esempio:

```
nestedGroupFilters: {
    res_resource: [],
```

```
    oma_activity_to_plan: [
        {
            condition: 'AND',
            columnName: 'cod_object',
            operator: 'IN',
            filterType: 'CUSTOM',
            value: '(select
oma_activity_to_plan.cod_object_program from oma_activity_to_plan
where cod_object_program=#{map.cod_object_program})'
        }
    ]
}
```

  - **order**: String o function, optional. Quando String è il name dell'attributo della *gwClass che per cui va eseguito l'ordinamento. Si può specificare una modalità di ordinamento ascendente o discendente, anteponendo il prefisso '+' [ASC] od il prefisso '-' [DESC]. Il default per l'ordinamento è '+'.
  - **applyOrderInStackedItems**: Boolean, default false. Questo verrà valutato solo in presenza di order non nullo, ed influirà sull'ordinamento dello stack degli item solo quando la gwClass di competenza si trovi sul piano della timeline (in quanto temporaneamente maiGwClass od, in assenza, baseGwClass)
- **showOnlyGroupsAlreadyInUse**: Boolean, default *true*. Valutata solo se è configurata una *nestedGroupGwClass* (e quindi una groupGwClass). Se posta a *true*, vengono visualizzati solo gruppi e sottogruppi per cui esiste almeno un elemento della baseGwClass che fa riferimento a d essi. Se posta a *false*, vengono visualizzati tutti i gruppi e sottogruppi, al netto dei filtri statici e dinamici e degli ambiti. In genere, ma non necessariamente, si imposta a *false* quando si prevede di rendere editabile la timeline;
- **editable**: Boolean o Object, default *false*. Quando il parametro è di tipo Object le sue sotto sezioni posso essere a loro volta Boolean o Object.
  L'oggetto foglia standard per il tipo Object è il seguente:

```
{allowedCases: [{groupGwClassName: '', nestedGroupGwClassName: '',
mainGwClassName: ''}, ..]}
```

  Alcune sotto sezioni hanno uno specifico formato (vedi sotto), ma il loro elemento foglia può essere Boolean o un Object standard *allowedCases*.

```
{
    canAdd{
        addInGroup: false,
        addInNestedGroup: {
            allowedCases: [
                {
                    groupClassName: 'msc_object',
                    nestedGroupClassName: 'res_resource',
                    mainClassName: 'oma_activity_to_plan'
                }
            ]
        }
    }
    canRemove: true,
```

```
    canResize: {
        allowedCases: [
            {
                groupClassName: 'msc_object',
                nestedGroupClassName: 'oma_activity_to_plan',
                mainClassName: 'res_resource'
            }
        ]
    },
    canDrag: {
        allowedCases: [
            {
                groupClassName: 'msc_object',
                nestedGroupClassName: 'oma_activity_to_plan',
                mainClassName: 'res_resource'
            }
        ]
    },
    canChangeGroup: {
        group2group: true,
        nestedGroup2group: true,
        nestedGroup2groupParent: true,
        nestedGroup2nestedGroup: false,
        nestedGroup2nestedGroupSibling: {
            allowedCases: [
                {
                    groupClassName: 'msc_object',
                    nestedGroupClassName: 'res_resource',
                    mainClassName: 'oma_activity_to_plan'
                }
            ]
        }
    }
}
```

- **quickEdit**: Boolean, optional, default *false*. Valutato solo quando *editable* è *true* o si manifestano sotto condizioni per le quali si può editare. Di base le operazioni di ridimensionamento, spostamento e cancellazione non vengono persistite immediatamente su DB, ma i cambiamenti vengono mantenuti in locale finché l'utente non ne effettua il commit o annulla il set di modifiche (si abilitano a tale scopo due tasti nella toolbar). Ponendo a *true* questo flag i meccanismi di persistenza dei cambiamenti vengono eseguiti immediatamente (eventualmente invocando le azioni di validazione e di consolidamento);
- **showConfirmationRequestDialog**: Boolean, optional, default *true*. Valutato solo quando *editable* è *true* o si manifestano sotto condizioni per le quali si può editare. Se posto a *true* per ogni operazione di modifica e cancellazione, sia in modalità quickEdit o meno, vengono mostrati dialog di richiesta conferma. Si può disabilitare sia per far fare all'utente meno click sia per far visualizzare solo l' eventuale dialog mostrabile tramite *validationGwActionName* o *consolidationGwActionName*;
- **validationGwActionName**: String, optional, default *null*. Nome della gwAction (libera), presente nella baseGwClass, che esegue una validazione sul set di modifiche che si stanno effettuando. Se omessa si dà per scontato che la validazione sia superata.

In qualsiasi caso in ingresso viene fornito un unico argomento, chiamato 'params', cosi fatto:

```
{
    insertedItems: [],
    modifiedItems: [],
    oldItems: [],
    removedItems: [],
    callback
}
```

Di volta in volta dentro i vari array saranno popolati con gli opportuni elementi se presenti. In caso di assenza di modifiche per una data categoria l'array sarà passato vuoto, ma mai *null*. Per comunicare la correttezza della validazione l'azione dovrà esplicitamente eseguire la callback dentro params, con argomento *true*, cosi:

```
params.callback(true);
```

L'azione è soggetta al controllo delle DACL, pertanto vanno abilitati i permessi necessari per l'utente che sta utilizzando la gwDynamicTimeline;
L'azione viene invocata su ogni tipo di modifica:

- insert puntuale;
- update puntuale;
- cancellazione puntuale;
- modifiche multiple;

- **consolidationGwActionName**: String, optional, default *null*. Nome della gwAction (libera), presente nella baseGwClass, che esegue le operazioni di consolidamento in seguito al salvataggio del set di modifiche. Se omessa nessun consolidamento verrà effettuato.
In qualsiasi caso in ingresso viene fornito un unico argomento, chiamato 'params', cosi fatto:

```
{
    insertedItems: [],
    modifiedItems: [],
    oldItems: [],
    removedItems: [],
    callback
}
```

Di volta in volta dentro i vari array saranno popolati con gli opportuni elementi, se presenti. In caso di assenza di modifiche per una data categoria l'array sarà passato vuoto, ma mai null.
L'azione è soggetta al controllo delle DACL, pertanto vanno abilitati i permessi necessari per l'utente che sta utilizzando la gwDynamicTimeline;
L'azione viene invocata su ogni tipo di modifica:

- insert puntuale;
- update puntuale;
- cancellazione puntuale;
- modifiche multiple;

- **inToolbarActionNameList**: String, optional, default *null*. Lista di nomi di gwAction (della baseClass, di tipo 'Libera'), separate da virgola, che determina quali azioni saranno lanciabili

dalla toolbar della timeline. Verranno valutati i permessi statici. Le azioni verranno invocate con I seguenti parametri in ingresso:
    - gwDynamicTimeline, Object;
- **showNoGroupSection**: Boolean or Object, default *false*. Quando è di tipo Object il formato è il seguente:

```
{
    allowedCase: [
        {
            groupClassName: 'msc_object',
        nestedGroupClassName: 'res_resource',
        mainClassName: 'oma_activity_to_plan'
        }
    ]
}
```

- **showContentOverflow**: Boolean, optional, default *false*. Quando a true, evita che il testo dentro gli item nel piano della timeline venga troncato nel caso oltrepassi il confine dimensionale dell'item (che deriva direttamente dai parametri temporali sottostanti in caso di start ed end);
- **autoExpandFirstGroup**: Boolean, optional, default true. In presenza di gruppi con sottogruppi, quando a true fa si che il primo nodo dei gruppi si espanda in automatico
- **swapAllowed**: Boolean, optional, default *true*. Se posto a false il pulsante di swap viene nascosto
- **aggregationEnabled**: Boolean, optional, default *true*. Abilita l'aggregazione automatica dei dati;
- **aggregationThreshold**: Integer, optional, default *10*. Valutato solo se *aggregationEnable* vale *true*. Determina il numero massimo di elementi per sezione oltre il quale il dato viene presentato in modo aggregato;
- **styles**: Object[], optional, default *null*. Permette di impostare una lista di regole che in caso di match applicano delle regole css al nodo dom dell'item nel piano della timeline. La lista è un array composto da elementi come nell'esempio qui sotto:

```
[
    {
        property: 'is_scheduled',
        operator: '=',
        value: 0,
        styleCssRule: 'background-color: rgba(255, 0, 0, 0.2); color:
red;'
    }
]
```

Le regole applicate sovrascrivono anche le proprietà modificate al momento dalla timeline quando l'utente seleziona un item, quindi vanno scelte con cura.
La regole contenute in styleCssRule vengono applicate solo se la proprietà dell'item property matcha value secondo il criterio definito in operator.

- **property** è il nome di uno dei campi della baseGwClass;
- **operator** può assumere i seguenti valori: '=', '!=', '<>', '⇐', '<', '>', '>=', 'IN', 'NOT IN', 'IS NULL', 'IS NOT NULL';

    - **value** può essere di tipo Integer, String, Boolean, Integer[], String[] o Boolean[];
    - **styleCssRule** è un set di regole css ognuna terminante con il separatore ';' .
- **openingDetailMode**: String, optional, default '*floatingPane*'. Determina con quale modalità verranno aperti i dettagli delle gwClass coinvolte;
- **start**: Date or Number or String or Moment, optional, default *null*. Imposta la data di inizio della finestra temporale della timeline. Inoltre ha la priorità sul parametro *initialWindow*. Se non viene impostato viene usata la data più vecchia presente negli eventi. Formati ammessi: new Date(), 1696515867321, '1696515867321', '2013-04-20', '2013-04-20 15:30:40', moment();
- **end**: Date or Number or String or Moment, optional. Imposta la data di fine della finestra temporale della timeline. Inoltre ha la priorità sul parametro *initialWindow*. Se non viene impostato viene usata la data più recente presente negli eventi. Formati ammessi: new Date(), 1696515867321, '1696515867321', '2013-04-20', '2013-04-20 15:30:40', moment();
- **initialWindow**: String, optional, default *null*, valori possibili: '*year*', '*month*', '*week*', '*day*', '*hour*', '*second*'. Imposta la data di inizio e fine per gli assi della timeline, in modo da mostrare gli oggetti appartenenti allo specifico intervallo. Viene valutato solo quando i parametri di inizio e fine data non vengono specificati. Se il parametro viene omesso la timeline proverà a calcolare in automatico le date di inizio e fine;
- **max**: Date or Number or String or Moment, optional, default *null*. Imposta la data massima di fine range, oltre la quale non è possibile spostarsi. Formati ammessi: new Date(), 1696515867321, '1696515867321', '2013-04-20', '2013-04-20 15:30:40', moment();
- **min**: Date or Number or String or Moment, optional. Imposta la data minima di inizio range, oltre la quale non è possibile spostarsi. Formati ammessi: new Date(), 1696515867321, '1696515867321', '2013-04-20', '2013-04-20 15:30:40', moment();
- **moveable**: Boolean, optional, default *true*. Specifica se la timeline può essere zoomata e spostata trascinando la finestra. Guarda anche il parametro *zoomable*. When true some specific buttons appear (move left, move right);
- **showMoveToNowButton**, Boolean, optional, default false. When true a specific button that move visualization to now appears. Evaluated only if *moveable* is *true*;
- **zoomable**: Boolean, optional, default true. Specifica se la timeline può essere zoomata con lo scrolling della finestra, applicabile solo se il parametro *moveable* è impostato a *true*. When true some specific buttons appear (zoom in, zoom out);
- **zoomMax**: number, optional, default *315360000000000*. Imposta il limite massimo per lo zoom in millisecondi, corrisponde a 10000 anni;
- **zoomMin**: number, optional, default *10*. Imposta il limite minimo dello zoom in millisecondi.
- **backgroundItems**: (**from 4.6.10, issue #1126**) Object[], optional. Contiene stilizzazioni delle finestre temporali specificate. La stilizzazione è di default o può essere esplicitata. array di objects, fatti come da esempio, con le seguenti proprietà:
    - start: Date(), String ('yyyy-MM-dd'), moment, required
    - end: Date(), String ('yyyy-MM-dd'), moment, required
    - content, string, optional. una descrizione per il periodo
    - bgcolor, string, optional mutualmente esclusivo con className (previsto default)
    - className, string, optional mutualmente esclusivo con bgcolor (needs somewhere a css class: .vis-item.vis-background.negative { background-color: rgba(255, 0, 0, 0.2); }) (previsto default)

      example.js

```
backgroundItems: [
 {content: 'Forbidden Period 1', start: '2023-01-01', end:
'2023-10-01', type: 'background'},
 {content: 'Forbidden Period 2', start: '2024-01-01', end:
```

```
'2024-10-01', type: 'background', bgcolor: 'rgba(255, 0, 0, 0.2)'},
 {content: 'Forbidden Period 3', start: '2025-01-01', end: '2025-10-01', type: 'background', className: 'negative'}
]
```

JavaScript

Consulta l'apposita sezione di javascrip API apertura scheda gwDynamicTime.

LeafItem TODO

---

gwGantt

(torna a elenco schede)

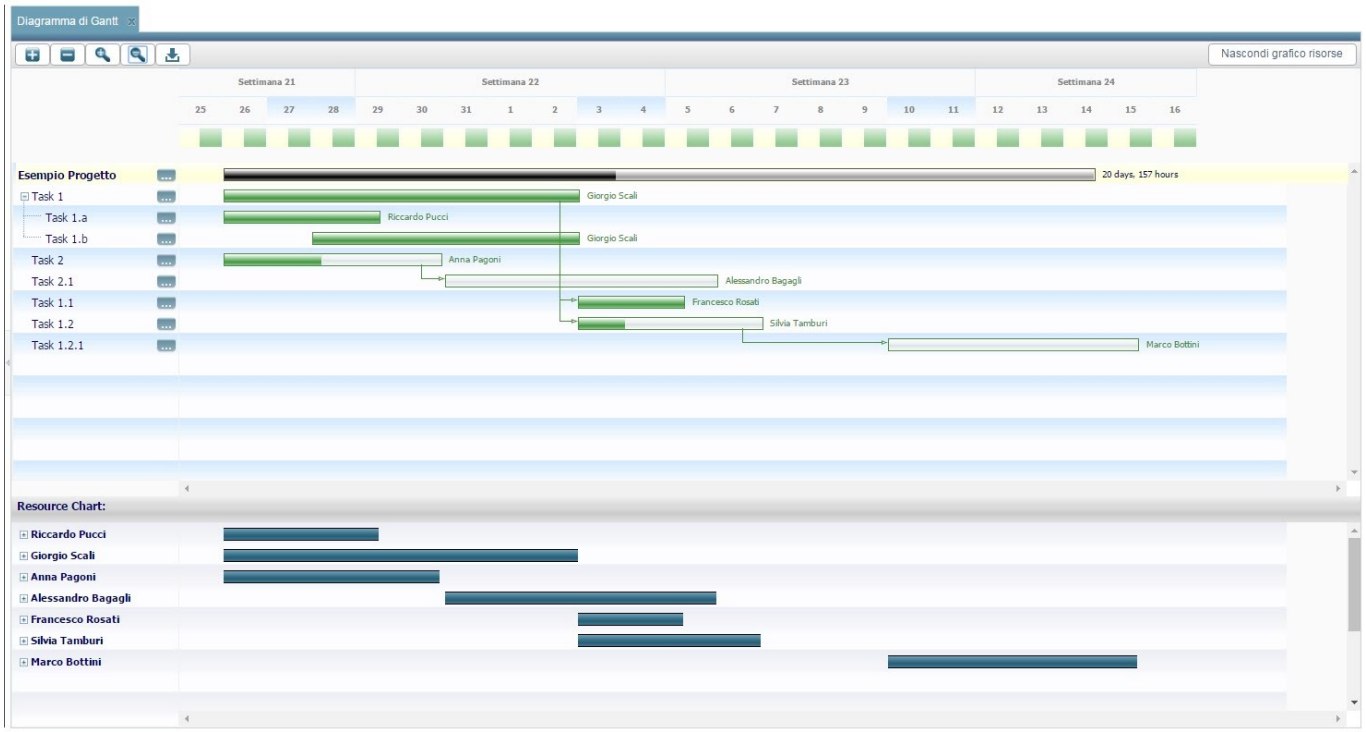
Con la scheda di tipo gwGantt Geoweb si dota di un visualizzatore/gestore di modelli gantt.

- Gantt in Generale
- Gantt in Geoweb
- Parametri
- Funzionalità
- Requisiti
- JavaScript
- LeafItem

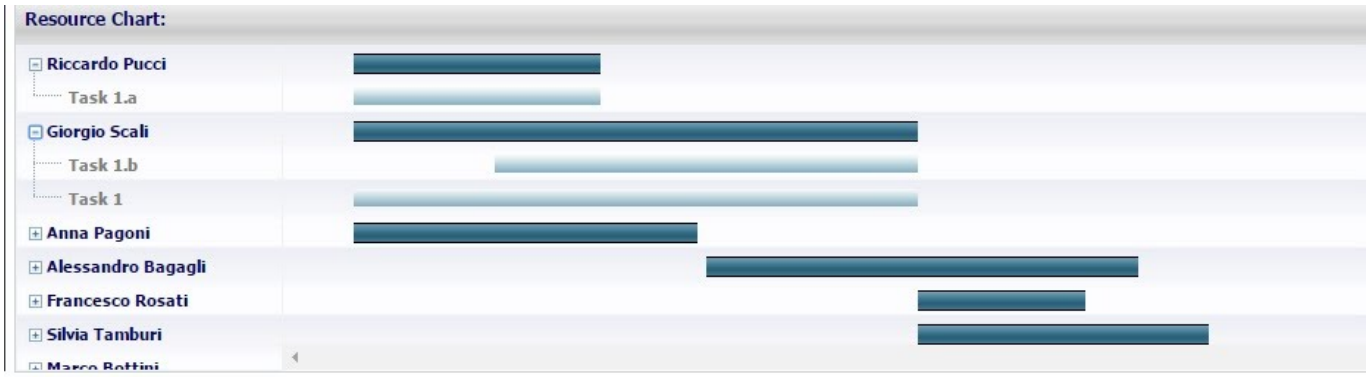
Gantt in Generale

Il diagramma di Gantt, usato principalmente nelle attività di project management, è costruito partendo da un asse orizzontale - a rappresentazione dell'arco temporale totale del progetto,suddiviso in fasi incrementali (ad esempio, giorni, settimane, mesi) - e da un asse verticale - a rappresentazione delle mansioni o attività che costituiscono il progetto.
Delle barre orizzontali di lunghezza variabile rappresentano le sequenze, la durata e l'arco temporale di ogni singola attività del progetto (l'insieme di tutte le attività del progetto ne costituisce la work breakdown structure). Queste barre possono sovrapporsi durante il medesimo arco temporale ad indicare la possibilità dello svolgimento in parallelo di alcune delle attività.
Man mano che il progetto progredisce, delle barre secondarie, delle frecce o delle barre colorate possono essere aggiunte al diagramma, per indicare le attività sottostanti completate o una porzione completata di queste. Una linea verticale è utilizzata per indicare la data di riferimento.
Un diagramma di Gantt permette dunque la rappresentazione grafica di un calendario di attività, utile al fine di pianificare, coordinare e tracciare specifiche attività in un progetto dando una chiara illustrazione dello stato d'avanzamento del progetto rappresentato; di contro, uno degli aspetti non tenuti in considerazione in questo tipo di diagrammazione è l'interdipendenza delle attività, caratteristica invece della programmazione reticolare, cioè del diagramma PERT[1]. Ad ogni attività possono essere in generale associati una serie di attributi: durata (o data di inizio e fine), predecessori, risorsa, costo.
Gantt In Geoweb

La scheda gwGantt di Geoweb si basa sul componente dojox.gantt.GwGantt



La scheda Gantt in Geoweb è costituita da due sezioni. La principale, in alto, è il diagramma vero e proprio. La seconda, in basso, denominata 'Grafico Risorse', mostra sulla linea del tempo il carico delle attività raggruppandole per risorsa. Questo grafico può essere nascosto/mostrato alla bisogna agendo sul pulsante in altro a destra della toolbar.



Essa permette, a seconda del set dei permessi, di aggiungere/rimuovere/modificare Progetti e relativi Task. Si possono aggiungere anche Task non direttamente figli di un Progetto, ma vincolati al altri Task.
In particolare abbiamo 2 tipi di vincoli: * vincolo precedente/successivo: due Task sono in questa situazione se l'esecuzione del secondo task non può avvenire prima che sia terminata l'esecuzione del primo (Es: Task 1.1 è successivo al Task 1)
* vincolo padre/figlio: un Task impostato come 'figlio' di un altro quando è in pratica una sotto attività dello stesso (Es: Task 1.a è figlio del Task 1). Deve essere eseguito nell'intervallo temporale occupato dal task padre, anche se senza un particolare ordine rispetto ad altri eventuali sub-Task 'fratelli'.
Il gwGantt preserva questi vincoli facendo validazioni quando l'utente prova a cambiare i parametri temporali dei task. In caso di validazione negativa l'operazione viene negata mostrando un messaggio di errore specifico. Si possono innestare e concatenare i vincoli fra i task a piacimento. Per esempio si può avere che un task figlio (di un altro task) abbia dei suoi task successivi. E tutte le altre possibili

combinazioni.
Il gwGantt permette la modifica di tutti i parametri di sua competenza dalla propria interfaccia, ma in generale non permette la modifica contemporanea dei parametri che determinano vincoli temporali: la modifica è permessa per un solo parametro temporale alla volta. Questa scelta viene fatta per semplificare i controlli: dipendendo alcuni parametri temporali da altri, la modifica in contemporanea di più parametri potrebbe generare dati inconsistenti.

Per esempio non posso impostare al Task 1 una durata tale per cui, sommata alla sua data di inizio, mi determini una fine prevista che vada oltre la data di inizio del task 1.1. Nel caso volessi mantenere come vincolo quella durata dovrei prima o anticipare la data di inizio del Task 1 o posticipare la data di inizio del Task 1.1.

Inoltre, per esempio, non posso far iniziare/terminare il Task 1.a, figlio del Task 1, prima/oltre dell'inizio/della fine del padre Task 1.

Proprio per preservare la consistenza dei parametri temporali di progetti e task, nelle implementazioni che usano il gwGantt è molto importante che gli attributi con name start_date (per i progetti), start_time, duration e duration_in_days (per i task), siano tutti rigorosamente posti a readonly. Questo per evitare che l'utente possa creare inconsistenze modificando tali valori dalle form in edit dei dettagli di Classe (gwClassDetail). Sarà il gwGantt stesso che internamente (e in modo trasparente a Geoweb) andrà a sovrascrivere di volta in volta le proprietà required, hidden e readonly dei vari attributi.

Parametri

- **tabWidgetId**: String, optional, default null. Se fornito è l'id che verrà assegnato al tab (e con il quale si potrà in seguito fare riferimento). Se omesso ne verrà generato uno in automatico del tipo 'gwGantt_tab_1576518200437'
- **projects**, richiesto, alternativo a projectsFilters, determina quali progetti saranno visibili nel gwGantt. Può essere un singolo valore String, multipli valori String uniti insieme e separati da virgola, o array String [] contenenti valori String.

Esempi di configurazione validi: 'project_name' 'project_name_1,project_name_2' ['project_name_1', 'project_name_2']

- **projectsFilters**, richiesto, alternativo a projects, determina quali progetti saranno visibili nel gwGantt. Array di oggetti JSON filtro [TODO MAKE LINK] del tipo:

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

- **gwOuterContainerId**, richiesto, String, è l'id del contenitore dojo che racchiude il widget gwGantt. E' usato per lanciare la function di refresh del widget gwGantt.

- **projectGwClassName**, opzionale, String, default 'ppm_gantt_project', nome della gwClass dei Progetti Gantt dove lavora il gantt. Può essere utilizzato per settare il nome di gwClass eventualmente costruite su viste basate sulla tabella 'ppm_gantt_project'
- **taskGwClassName**, opzionale, String, default 'ppm_gantt_task', nome della gwClass dei Task Gantt dove lavora il gantt. Può essere utilizzato per settare il nome di gwClass eventualmente costruite su viste basate sulla tabella 'ppm_gantt_task'
- **taskDurationChooseMode**, opzionale, String, valori permessi: 'duration' (default), 'duration_in_days', 'end_time' . I valori permessi non sono latro che i name degli attributi (che devono tutti esistere) dei widget che permettono la modifica della durata temporale dei task. Questo parametro controlla la modalità con la quale l'utente potra modificare nella form di dettaglio del gwGantt la durata dei singoli task. duration, valore di default, mostra un attributo che permette di impostare la durata del task in ore. duration_in_days, mostra un attributo che permette di impostare la durata del task in giorni (sul DB verrà persistito il valore della durata in ore, considerando il vincolo 8 ore/giorno). end_time, mostra un attributo che permette di impostare la data di fine del task in giorni (sul DB verrà persistito il valore della durata in ore, calcolando la differenza in giorni tra la data di fine e quella di inizio e considerando il vincolo di 8 ore/giorno). Questo parametro influisce anche su quale attributo (relativo alla durata del task) viene mostrato nelle form di dettaglio del gwGantt, anche quando si modificano altri parametri: per esempio il nome del gwGantt
- **withResourceChart**, opzionale, Boolean, default true. Flag che abilita o meno la visualizzazione del Resource Chart
- **limitEditing**, opzionale, Boolean, default false. Se true restringe il set di funzioni disponibili per l'editing di Project e Task rendendo modificabili solamente name, percentage e taskOwner. E' necessario perché attualmente il gwGantt non ha un modello dati elaborato come quello di MS Project. Accade che, modificando in GW i parametri temporali di progetto importato da file MSPDI, e successivamente riesportandolo, si generino inconsistenze. Questo perché il sistema di editing in GW non considera ne i 'Calendar', ne altri aspetti del modello. Limitando l'editing solo ad un set di funzioni sicure, è possibile in seguito esportare il progetto garantendone l'originale consistenza dei dati temporali.

I prossimi 7 parametri, opzionali, riguardano l'eventuale sovrascrittura dei permessi di Geoweb, rispettivamente per le classi dei Progetti Gantt e dei Task Gantt. Sono utilizzati solo se presenti. In loro mancanza valgono i permessi base (statici/dinamici) impostati in Geoweb per le classi dei Progetti e dei Task. Ciò è utile perchè spesso si vuole rendere i parametri temporali di Progetti e Task editabili solo dal componente gwGantt, per prevenire possibili modifiche inconsistenti da parte degli utenti.

- **allowInsertProjects** ,opzionale, Boolean, se assente valgono i permessi base (statici/dinamici)
- **allowModifyProjects**,opzionale, Boolean, se assente valgono i permessi base (statici/dinamici)
- **allowModifyProjectsPercentage**, Boolean, opzionale, default a true, valutato solo se allowModifyProjects vale true o sono presenti i permessi di modifica base (statici/dinamici) sulla classe dei progetti. Se assente o è posto a true, il gwGantt permette di settare la percentuale del progetto. Il meccanismo che, alla modifica della % di completamento di un task, ricalcola la anche la % globale del progetto (pesando le componenti delle % di completamento dei singoli task) continua a funzionare. Cioè settando la % del progetto, questa verrà poi sovrascritta alla prima modifica della % effettuata sul singolo task
- **allowDeleteProjects**,opzionale, Boolean, se assente valgono i permessi base (statici/dinamici)
- **allowInsertTasks**,opzionale, Boolean, se assente valgono i permessi base (statici/dinamici)
- **allowModifyTasks**,opzionale, Boolean, se assente valgono i permessi base (statici/dinamici)
- **allowDeleteTasks**, opzionale, Boolean, se assente valgono i permessi base (statici/dinamici)

Funzionalità

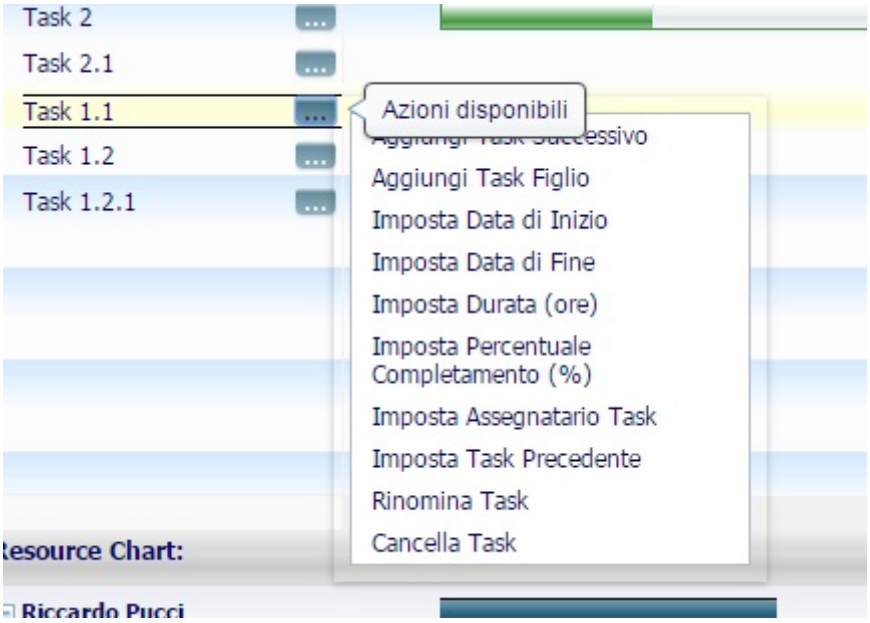
Sulla toolbar sono disponibili i seguenti comandi: Gw Gantt Toolbar Allarga sequenza temporale Contrai sequenza temporale Zoom In Zoom Out Esporta dati su file MSPDI (.xml) Importa dati da file MSPDI (.xml) In caso per qualche ragione ci siano attualmente più progetti mostrati nel grafico gwGantt, in fase di export ed import verrà richiesto all'utente quale progetto esportare o quale progetto sovrascrivere. In fase di esportazione, il file xml originariamente caricato (e dotato di tutti i vincoli del modello MSProject), se presente, viene usato come base, e gli vengono applicate tutte le modifiche man mano effettuate sui task. In caso di assenza del file xml, esso viene generato al volo a partire dai task presenti sul DB. Durante l'import viene sovrascritto il nome del progetto con quello del file importato. Anche qui si cerca di preservare l'univocità del name del progetto: viene aggiunta la data odierna in caso di nome già in uso, e se non dovesse bastare un ulteriore marker. Gli elementi di tipo Progetto e di tipo Task, a seconda del set di permessi impostato (derivante dai permessi base delle classi o dalle sovrascritture), espongono le seguenti azioni nel menu a tendina. Il menu a tendina compare al click sull'elemento.

Progetto:

- Aggiungi Progetto (permessi necessari: inserimento progetti)
- Rinomina Progetto (permessi necessari: modifica progetti)
- Imposta Percentuale Completamento (%) (permessi necessari: modifica progetti, pemesso % progetto assente o true)
- Imposta Data di Inizio (permessi necessari: modifica progetti)
- Aggiungi Task (permessi necessari: inserimento task, modifica progetti)

Task:

- Aggiungi Task Successivo (permessi necessari: inserimento task, modifica progetti)
- Aggiungi Task Figlio (permessi necessari: inserimento task, modifica progetti)
- Imposta Data di Inizio (permessi necessari: modifica task, modifica progetti) mantiene inalterata la durata, ricalcolando (con vincolo 8 ore per giorno) la data di fine (non persistita su DB)
- Imposta Data di Fine (permessi necessari: modifica task, modifica progetti) mantiene inalterata la data di inizio, ricalcolando (con vincolo 8 ore per giorno) la durata
- Imposta Durata (ore) (permessi necessari: modifica task, modifica progetti) mantiene inalterata la data di inizio, ricalcolando (con vincolo 8 ore per giorno) la data di fine (non persistita su DB)
- Imposta Durata (giorni) (permessi necessari: modifica task, modifica progetti) mantiene inalterata la data di inizio, ricalcolando (con vincolo 8 ore per giorno) la durata in ore (persistita su DB) e la data di fine (non persistita su DB)
- Imposta Percentuale Completamento (%) (permessi necessari: modifica task) verrà ricalcolata di conseguenza anche la percentuale di completamento del progetto (ma solo in visualizzazione sul gantt, lato client: non previsto campo sul DB)
- Imposta Assegnatario Task (permessi necessari: modifica task)
- Imposta Task Precedente (permessi necessari: modifica task, modifica progetti)
- Rinomina Task (permessi necessari: modifica task)
- Cancella Task (permessi necessari: modifica task, modifica progetti)



Requisiti

Per funzionare il componente gwGantt ha bisogno di questi vincoli:

- esistenza gwClass con nome ppm_gantt_project (di default basata sull'omonima tabella), o sovrascrivendo 'projectGwClassName' con altro nome classe, magari costruita su una vista di ppm_gantt_project, ma sempre con i seguenti campi base (Il nome degli Attributi deve corrispondere al campo della tabella):

```
 (
id_project INTEGER NOT NULL, -- chiave primaria (gantt)
project_name CHARACTER VARYING(250), -- nome del progetto (gantt)
start_date TIMESTAMP WITHOUT TIME zone, -- data di avvio del progetto
(gantt)
percentage INTEGER, -- percentuale di completamento del progetto (gantt)
expected_end_date TIMESTAMP WITHOUT TIME zone, -- data di fine progetto
prevista (da task collegati)
xml CHARACTER VARYING, -- usato per salvare l'xml durante un import tramite
MSPDI .xml, in genere usato quando il gantt è settato con limitEditing a
true
 )
```

- esistenza gwClass con nome ppm_gantt_task (di default basata sull'omonima tabella), o sovrascrivendo 'taskGwClassName' con altro nome classe, magari costruita su una vista di ppm_gantt_task, ma sempre con i seguenti campi base (Il nome degli Attributi deve corrispondere al campo della tabella):

```
 (
  id_task INTEGER NOT NULL, -- chiave primaria (gantt)
  task_name CHARACTER VARYING(250), -- denominazione del task (gantt)
  start_time TIMESTAMP WITHOUT TIME zone, -- data inizio del task (gantt)
  duration INTEGER NOT NULL, -- durata (hh) del task (gantt)
  percentage INTEGER, -- percentuale di completamento (gantt)
  previous_task_id INTEGER, -- id del task precedente (gantt)
```

```
  task_owner CHARACTER VARYING(100), -- assegnatario (nome e cognome) del
task (gantt)
  id_project INTEGER NOT NULL, -- id del progetto di appartenenza (gantt)
  parent_task_id INTEGER, -- id del task padre (gantt)
  is_milestone INTEGER DEFAULT 0 -- flag per identificare particolari task
  unique_id INTEGER campo di supporto usato usato durante le PROCEDURE di
import
 )
```

- esistenza di un attributo di tipo Date con nome 'end_time' sulla classe dei task (anche non mappato su un campo della tabella). Infatti dall'interfaccia del gwGantt è possibile modificare la durata temporale di un Task specificandone una data di fine. A venire persistito sul DB non è pero la data di fine, ma la durata, che viene calcolata a partire dalla data di fine del Task, mantenendo inalterata la data di inizio dello stesso. L'attributo serve alla scheda gwGantt solo per poterlo mostrare della scheda di dettaglio relativa al cambio data di fine del Task.
- esistenza di un attributo di tipo Integer con nome 'duration_in_days' sulla classe dei task (anche non mappato su un campo della tabella). Infatti dall'interfaccia del gwGantt è possibile modificare la durata temporale di un Task specificandone i giorni. A venire persistito (e rieltto dal Gantt) sul DB non è pero mai la durata in giorni, ma sempre il campo che contiene la durata in ore.
- Gli unici attributi del gwGantt che supportano widget che usano valori codificati (salvando il valueToStore, ma mostrando il valueToShow) sono il task_name, il task_owner.

JavaScript

Consulta l'apposita sezione di javascrip API apertura scheda gwGantt.

LeafItem

E' un leafItem che al click apre la scheda di tipo gwGantt. Il type che caratterizza il leafItem è *gwGantt*. Codice da inserire dentro un accordionPaneItem nell'xml di progetto:

```
<leafItem name="test_gw_gantt" label="Test gwGantt"
image="test_gw_gantt.png" type="gwGantt">
    <parameter name="projects" value="nome_progetto"
hideToClient="false"></parameter>
    <parameter name="projectGwClassName" value="ppm_gantt_project"
hideToClient="false"></parameter>
    <parameter name="taskGwClassName" value="ppm_gantt_task"
hideToClient="false"></parameter>
    <parameter name="withResourceChart" value="true"
hideToClient="false"></parameter>
    <parameter name="allowInsertProjects" value="true"
hideToClient="false"></parameter>
    <parameter name="allowModifyProjects" value="true"
hideToClient="false"></parameter>
    <parameter name="allowDeleteProjects" value="true"
hideToClient="false"></parameter>
    <parameter name="allowInsertTasks" value="true"
hideToClient="false"></parameter>
    <parameter name="allowModifyTasks" value="true"
```

```
hideToClient="false"></parameter>
    <parameter name="allowDeleteTasks" value="true"
hideToClient="false"></parameter>
    <parameter name="allowModifyProjectsPercentage" value="true"
hideToClient="false"></parameter>
    <parameter name="taskDurationChooseMode" value="duration"
hideToClient="false"></parameter>
    <parameter name="limitEditing" value="true"
hideToClient="false"></parameter>
    <parameter name="gwOuterContainerId" value="container_id"
hideToClient="false"></parameter>
</leafItem>
```

---

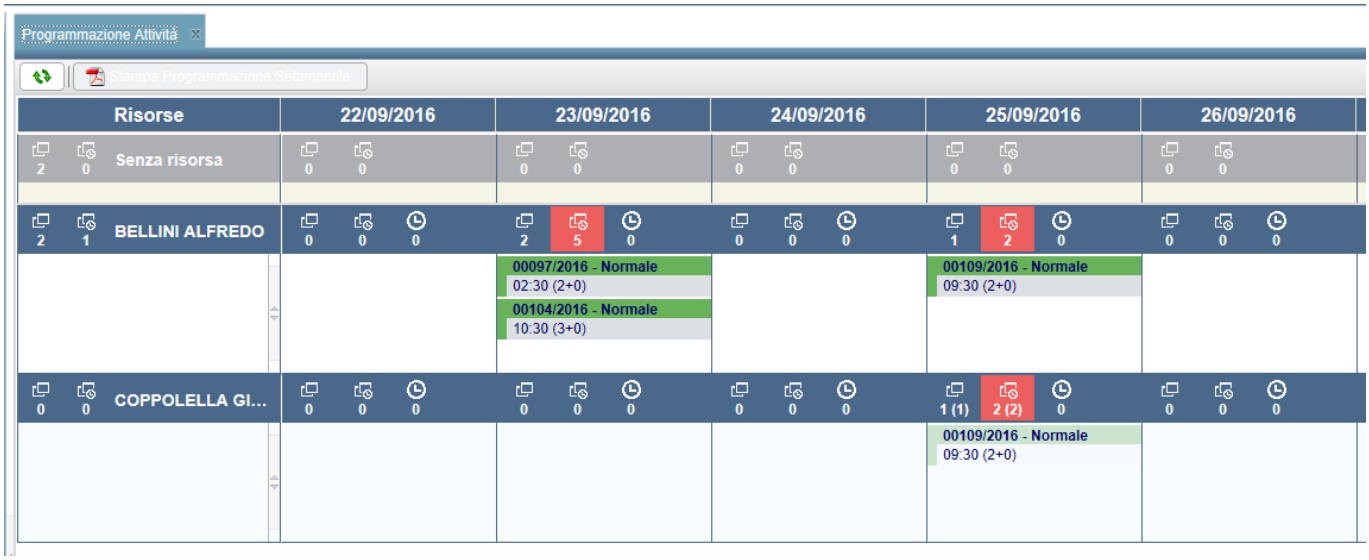
gwTimeTable

(torna a elenco schede)

- Parametri
- Requisiti
- JavaScript
- LeafItem

Descrizione

Con la scheda di tipo gwTimeTable (ex **gwTimeLine**) è possibile gestire la pianificazione delle attività, relazionandola ad una risorsa ed una data. Il widget può opzionalmente operare in sinergia con il più generale modulo dei turni delle risorse, se presente.



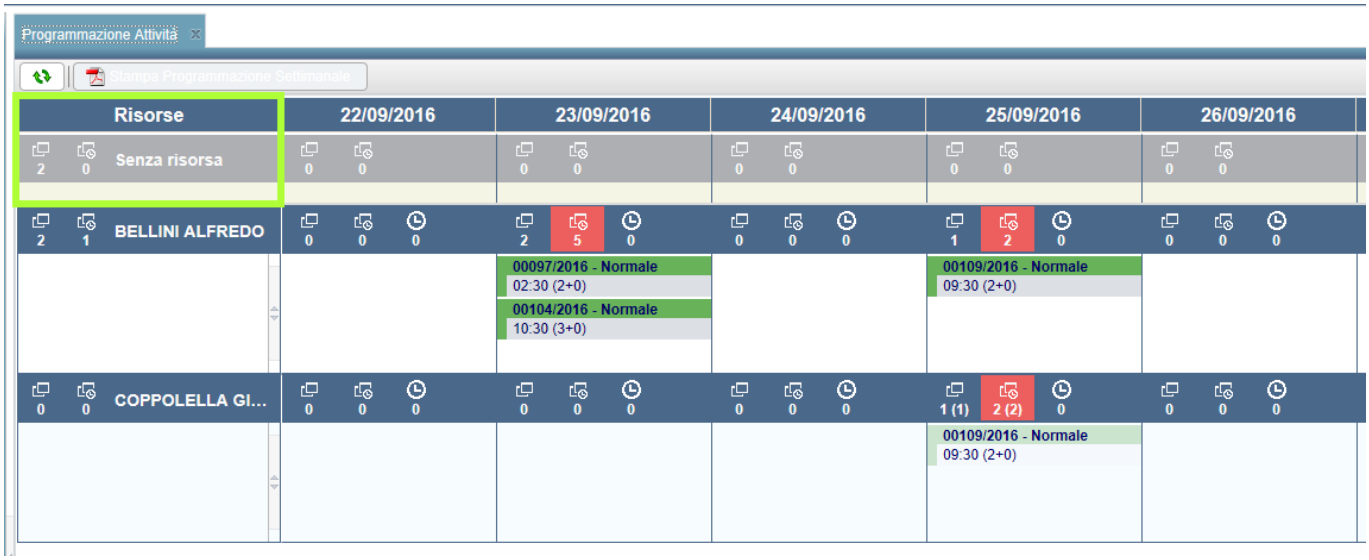
Nomenclatura:

- per attività assegnata si intende un'attività vincolata ad una risorsa
- per attività programmata si intende un'attività vincolata ad una data

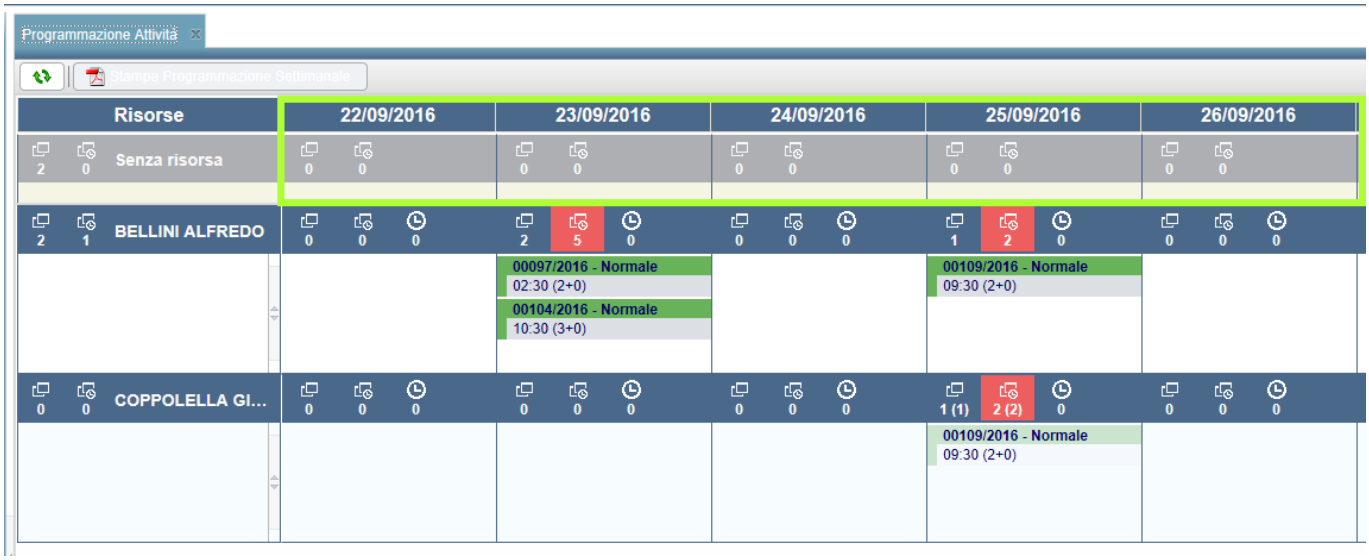
La scheda è sostanzialmente una tabella.

Essa ha delle date per intestazioni di colonna, tante quanto indicato dal parametro nDays. La prima intestazione di colonna è comunque sia sempre 'Risorse'.
Essa ha delle risorse per intestazione di riga, tante quanto indicato dal parametro gwdResources (o gwdResourcesFilters). La prima intestazione di riga è comunque sia sempre 'Senza Risorse'.
Le due tipologie di intestazioni di colonna incrociate con le due tipologie di intestazioni di riga, determinano 4 sezioni (una matrice 2×2) nella tabella, che contengono oggetti fra loro omogenei:

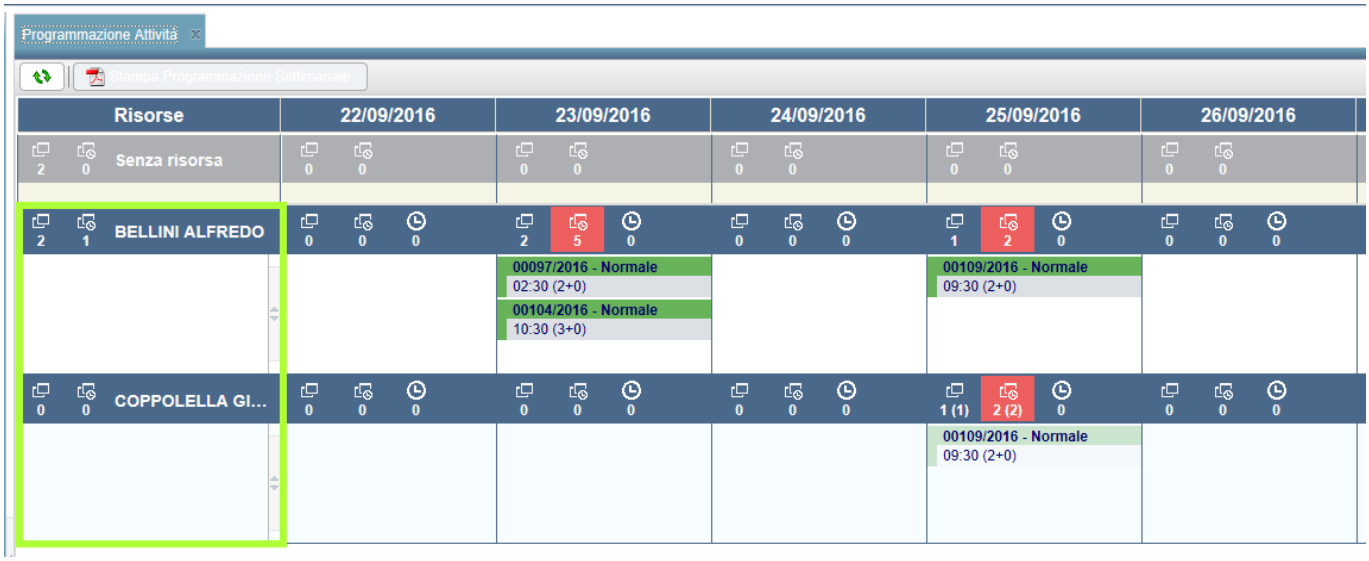
- La sezione con indici 0,0 nella matrice, cella all'incrocio fra la colonna 'Risorse' e la riga 'Senza Risorse', contiene le attività **ne assegnate ne programmate** (sezione formata da una sola cella, quella di indice 0,0).



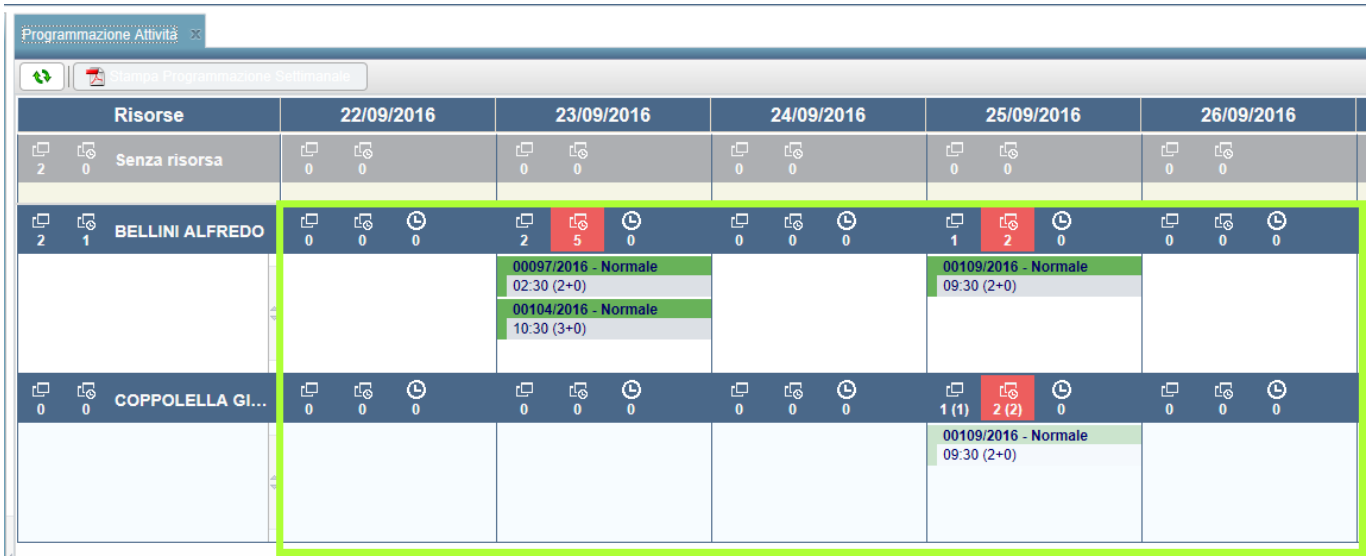
- La sezione con indici 0,1 nella matrice, contenente celle all'incrocio fra una colonna data e la riga 'Senza Risorse', contiene le attività **programmate ma non assegnate** (sezione formata dalla prima colonna di celle, esclusa la prima cella).



- La sezione con indici 1,0 nella matrice, contenente celle all'incrocio fra la colonna 'Risorse' e una riga risorsa, contiene le attività **assegnate ma non programmate** (sezione formata dalla prima riga di celle, esclusa la prima cella).



- La sezione con indici 1,1 nella matrice, contenente celle all'incrocio fra una colonna data e una riga risorsa, contiene le attività **assegnate e programmate** (sezione formata da tutte le celle ad esclusione di quelle appartenenti alla prima riga e alla prima colonna).



Sulle celle di ogni sezione è presente una toolbar sommario che può indicare dove applicabile:

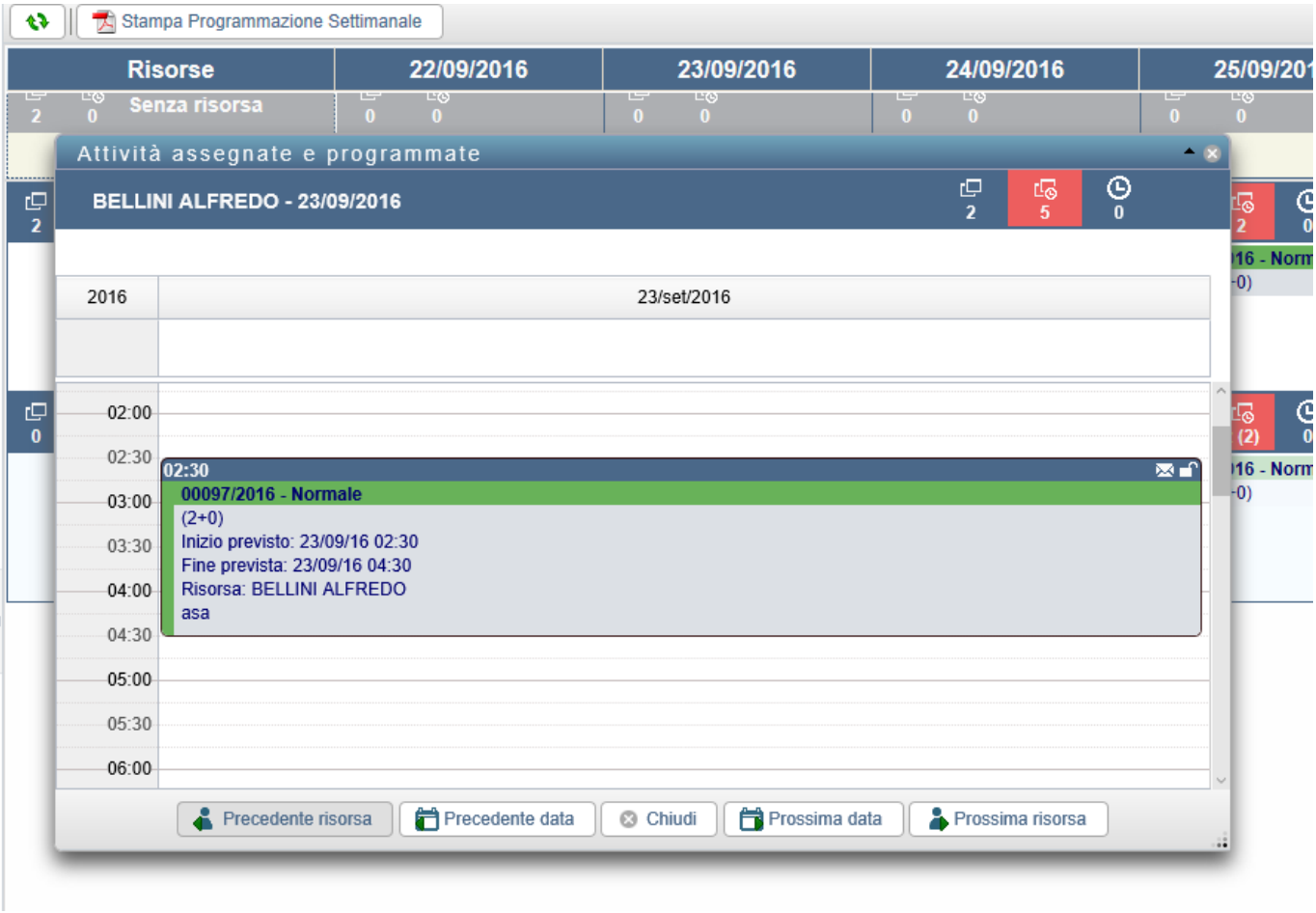
- numero delle attività della risorsa per quel giorno (tra parentesi le eventuali a supporto).
- numero delle ore assegnate alla risorsa per quel giorno (tra parentesi le eventuali a supporto) (diventa con sfondo rosso se le ore assegnate superano quelle lavorative).
- numero delle ore lavorative della risorsa per quel giorno.
- disponibilità della risorsa per un dato giorno (compare in verde solo in caso di disponibilità, con info sugli orari di disponibilità).

Sulla sezione delle attività assegnate e programmate compare inoltre una preview delle attività.
Tale preview presenta per ogni attività informazioni come codice, criticità, ora di inizio e durata. Essa si limita a mostrare le prime due attività del giorno, eventualmente evidenziando la presenza di altre (tramite '...').
In corrispondenza della prima colonna, sulle celle di ogni risorsa, è presente un tastino che permette di espandere/collassare l'intera riga relativa alla risorsa, cosi da mostrare in preview un numero maggiore di attività.

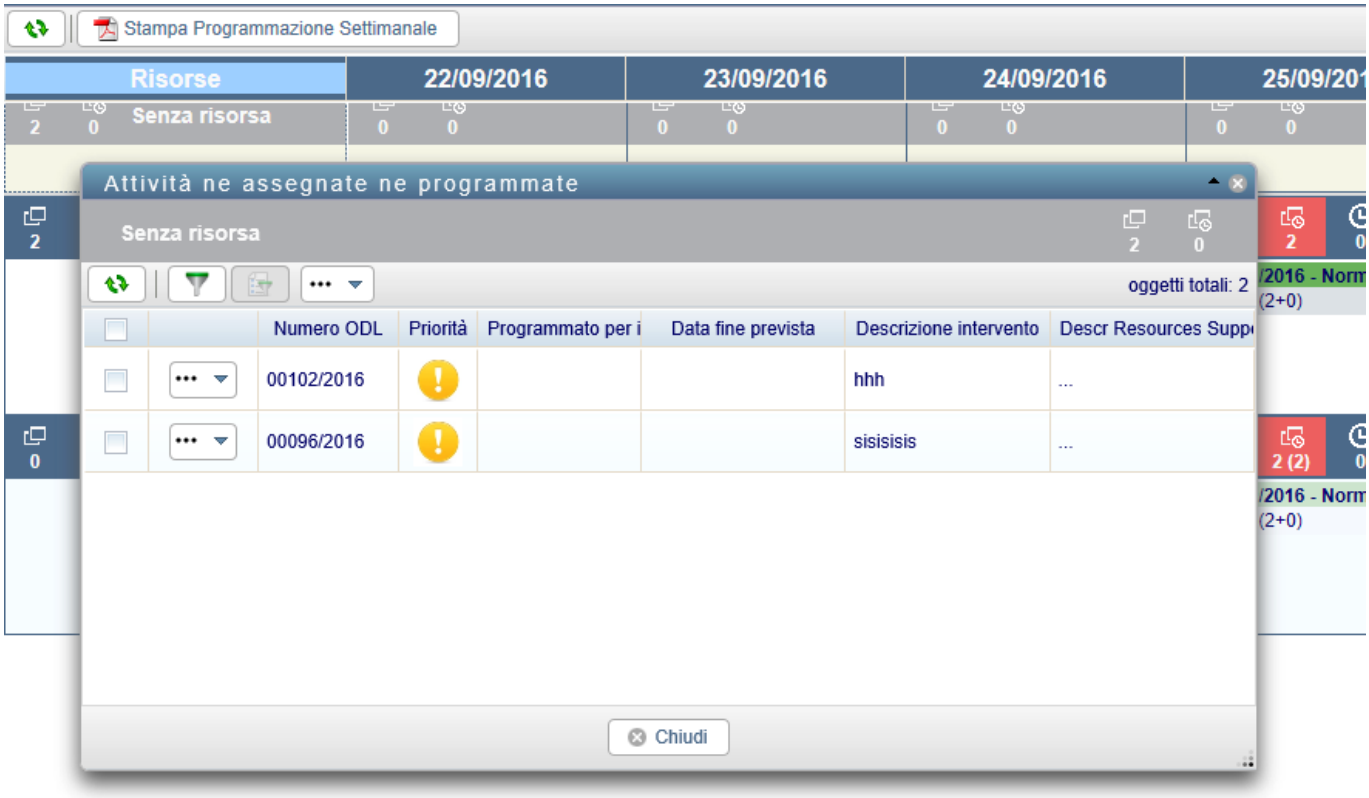
Dove applicabile sono presenti numerosi tooltip contenenti tutte le informazioni necessarie, molto utili in contesti dove la mole delle attività potrebbe nascondere le info delle stesse (es: scheda dettaglio attività programmate ed assegnate)

Scheda di dettaglio gruppo attività
Al click su una cella una qualsiasi sezione viene aperto un **dettaglio del gruppo di attività** ivi presenti.

Per la sezione delle attività assegnate e programmate il dettaglio è rappresentato tramite un Calendario (dojo),

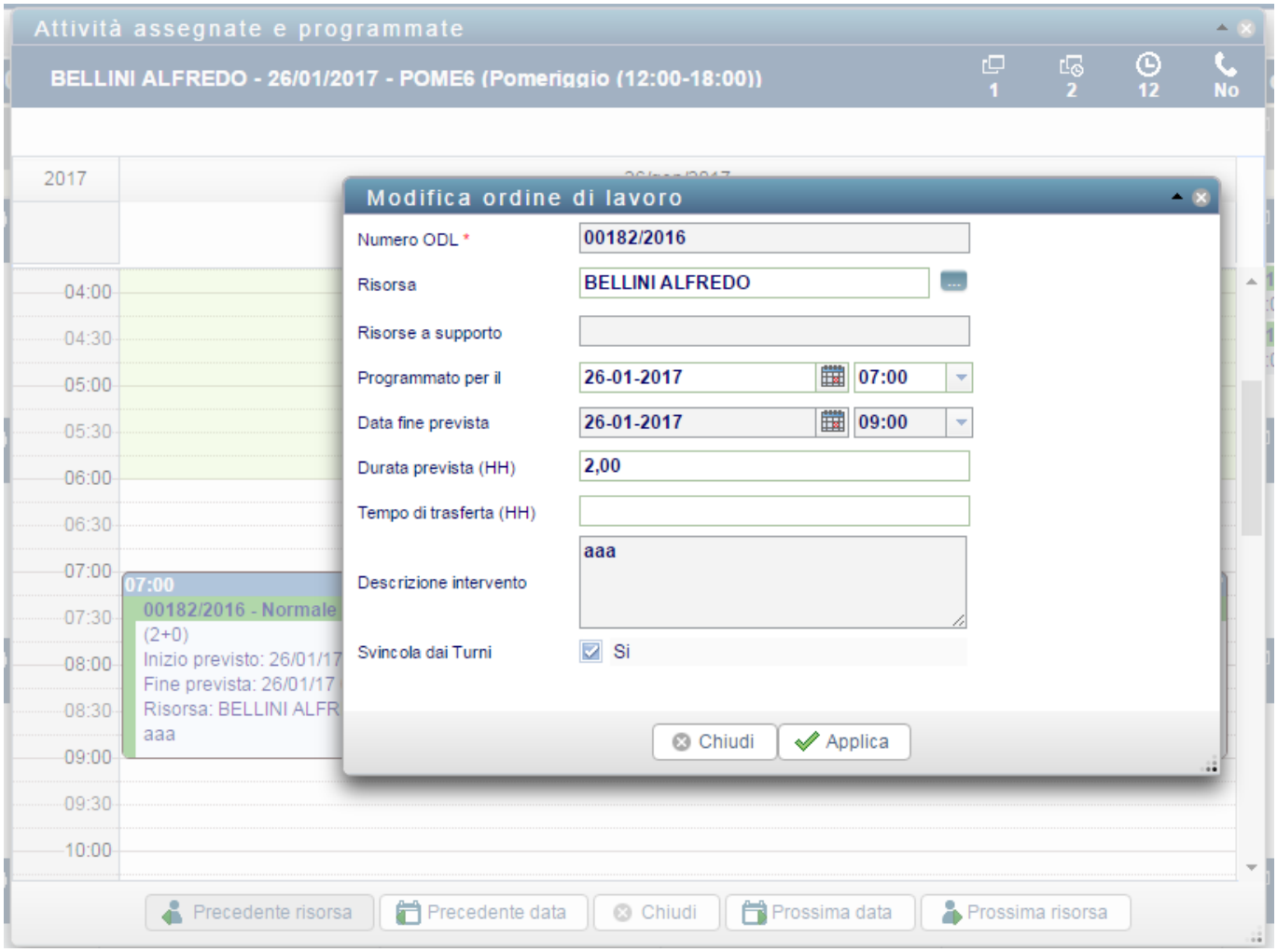


Mentre per le altre tre sezioni viene usata una lista.



Scheda di dettaglio della singola attività
All'interno dei dettagli di gruppi di attività è possibile gestire la pianificazione delle singole attività, riprogrammando e/o riassegnando le stesse.

Ciò viene fatto dentro il **dettaglio della singola attività**. Questo è apribile con singolo click sulle righe, per i dettagli (di gruppo attività) delle celle che usano liste come forma di rappresentazione delle attività, o doppio click per i dettagli (di gruppo attività) delle celle contenenti attività programmate e assegnate.

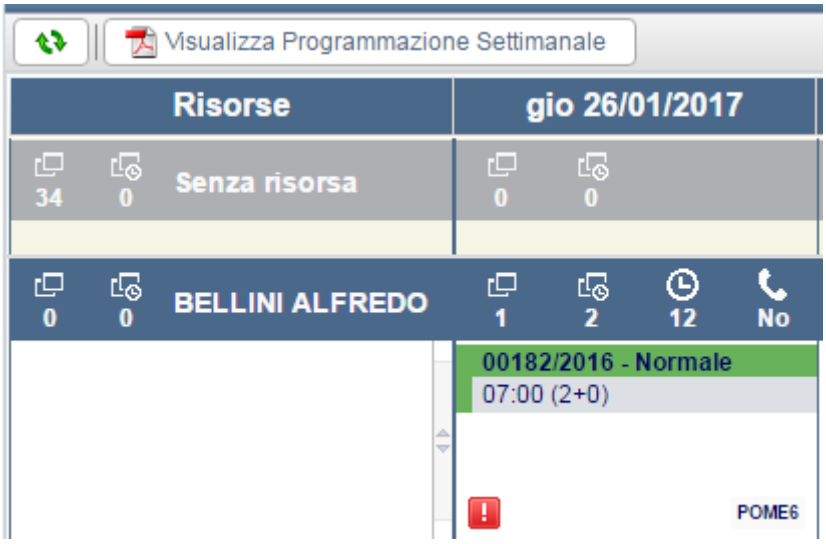


La scheda di dettaglio della singola attivià permette di cambiare a piacimento sia la risorsa assegnataria che la data di programmazione (e durata intervento). A seconda dei valori assunti dalla coppia risorsa/data, al click su 'Applica' l'attività potrà andare in una specifica sezione delle quattro viste precedentemente.
Nel caso in cui si scelga sia la risorsa che la data di programmazione, il sistema provvederà a calcolare in automatico la data di fine. In presenza del modulo dei turni, di default questa operazione viene fatta distribuendo sequenzialmente (fino ad esaurimento) le ore della durata dell'attività sulle fasce orarie dei turni programmati per una data risorsa (da una certa data in poi).
Questo viene fatto fino a quando non vengono esaurite le ore 'residue' del'attività o non esistono turni programmati, o comunque non coprono l'entità della durata.
Se la durata è grande in misura per cui non è possibile distribuirla totalmente sulle fasce orarie dei torni già programmati l'utente viene notificato con un messaggio. L'utente viene anche notificato quando sceglie una data di inizio che sta fuori da tutte le fasce orarie dei turni relative alla risorsa scelta.
Per escludere questo comportamento è disponibile il check 'Svincola dai turni'. Quando attivo esso escluderà il meccanismo di calcolo della data di fine sopra descritto, e la data di fine verrà calcolata semplicemente sommando la durata alla data di inizio. Inoltre ogni data di inizio sarà considerata valida, anche se fuori dai turni.

Da notare che a prescindere da come sia stata calcolata la data di fine, in caso di emerse incompatibilità con la programmazione sottostante essa viene 'preservata', anche eventualmente a danno della consistenza del valore dalla durata. Per esempio, se si effettua una programmazione di un'attività, non svincolandola dai turni, basandosi su una programmazione dei turni relativi ad una certa risorsa, indirettamente si sta fissando una data di fine (secondo le modalità descritte sopra). Se

ora, tramite la gestione della programmazione dei turni, si va a modificare i turni della stessa risorsa, nella stessa settimana in cui si era programmato l'attività, si puo generare un'inconsistenza.
Tale incosistenza sarà segnalata all'utente o tramite il meccanismo di segnalazione delle ore fuori dai turni (vedi sotto) e/o tramite l'ammontare di ore assegnate per la giornata, che sarà in genere piu alto del giusto (ed eventualmente segnalato in rosso, se eccede il monte ore lavorative).
Infatti preservando le variabili data inizio/data fine, la durata dell'attività sarà la mera differenza delle due, anche se originariamente la programmazione era stata fatta non svincolandola dai turni. Ne consegue che se, per esempio, l'attività aveva una durata per cui era stata originariamente spalmata sui turni di piu giorni adesso, dopo che la programmazione sottostante è stata modificata, essa verrà mostrata nella scala temporale nel dettaglio dell'attività come un unicum che va dalla sua data di inizio e termina sulla sua data di fine, mostrando una durata potenzialmente superiore a quella presente sul DB.

Ore assegnate fuori dai turni
Nelle celle contenti attività assegnate e programmate, il widget segnala eventualmente la presenza di ore (appartenenti ad una qualche attività) al di fuori delle fasce orarie individuate dai turni della risorsa.
Ciò viene ovviamente fatto solo in presenza del modulo dei turni.
Tale segnalazione è effettuata applicando un marker (punto esclamatvo su sfondo rosso) nell'angolo in basso a sinistra della cella. Basta anche solo la presenza di una singola attività con ore fuori dai turni per quel giorno per far comparire il segnale. Se l'attività coinvolge piu giorni, e presenta ore fuori dai turni in altri giorni, cioè verrà segnalato solo nelle celle di competenza dei giorni interessati.

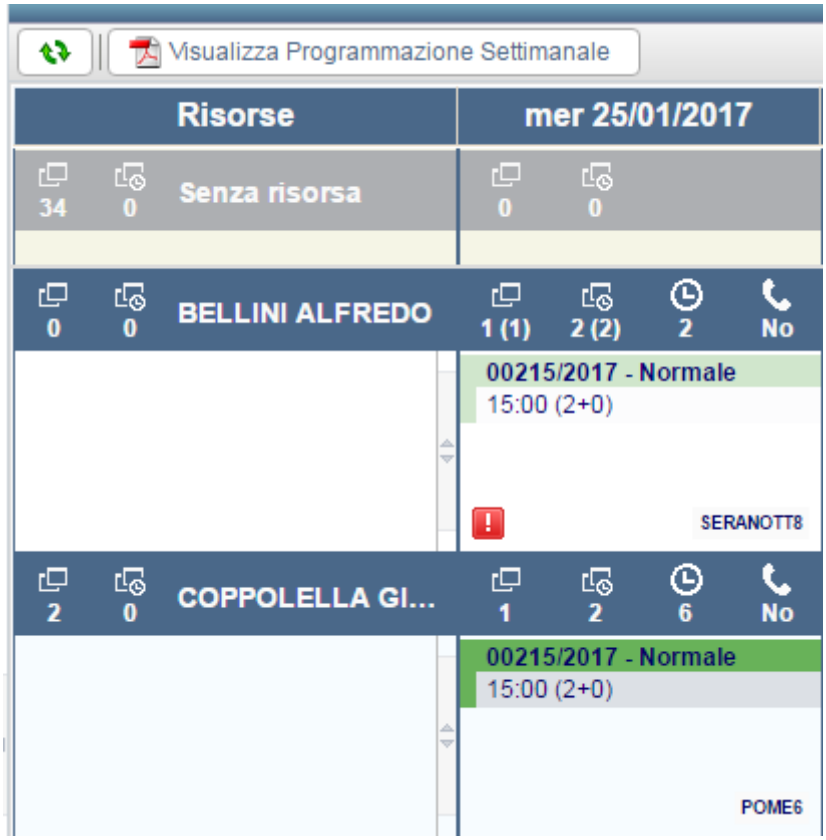


Questa segnalazione in genere si manifesta:

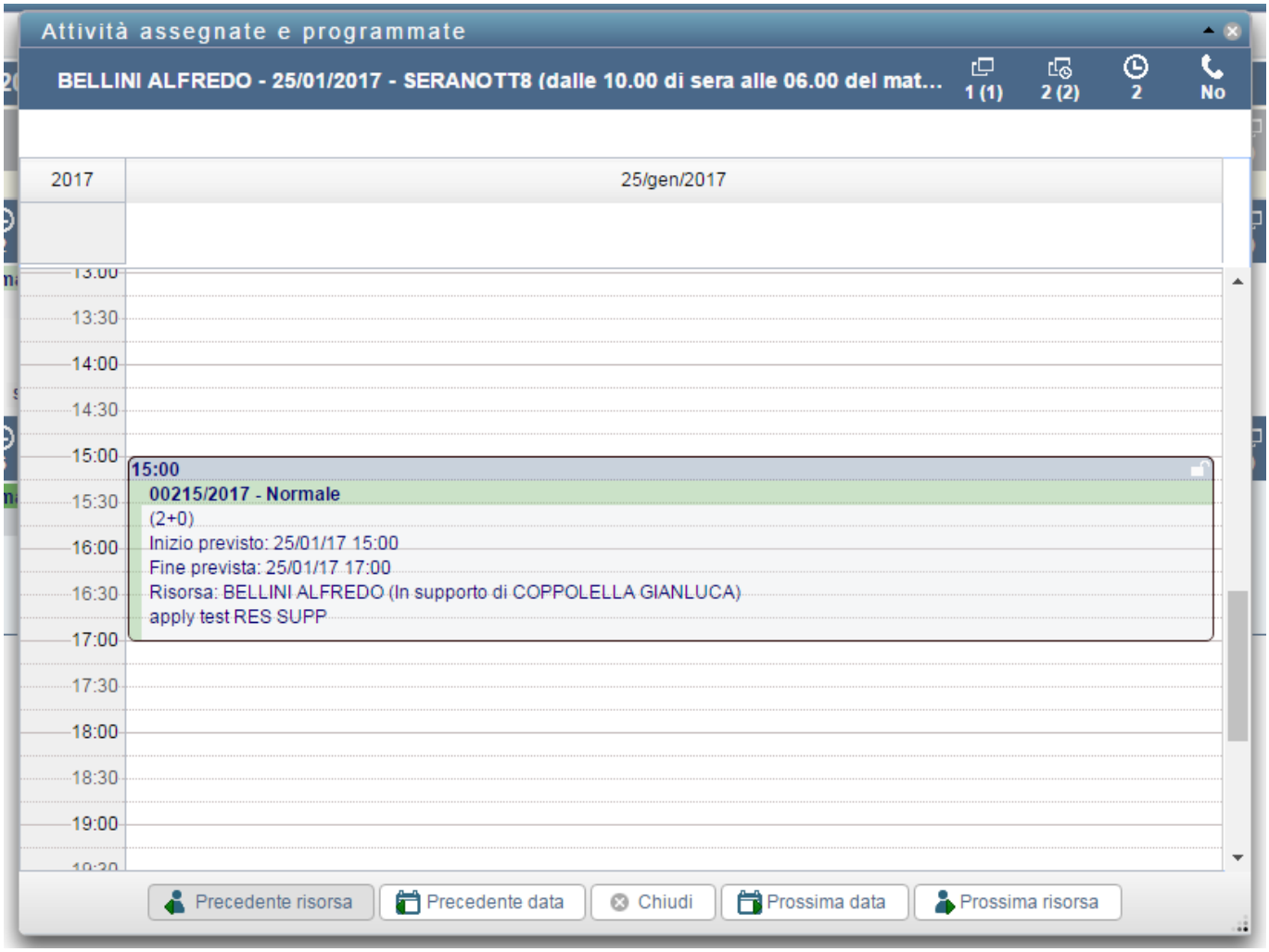
- Quando l'attività e segnata come 'svincolata dai turni' e le sue ore ricadono fuori dalle fasce orarie dei turni di quella risorsa
- In seguito ad una riprogrammazione dei turni per una data risorsa in una data settimana. Ciò puo aver invalidato la programmazione delle attività (non svincolate dai turni) che era stata fatta interpolando i dati dell'attività (data di inizio, durata e data di fine risultante) con quelli dei turni programmati (fasce orarie ora inizio/ora fine)

Attività in cui la risorsa è a supporto
Geoweb permette, esternamente al widget, di definire risorse a supporto per una data attività assegnata ad una risorsa principale.
Nel widget vengono visualizzate e gestite anche le attività programmate per cui una risorsa è a

supporto di un'altra. Queste vengono considerate nel computo del numero delle attività e delle ore assegnate ad una data risorsa in una data giornata.
In generale nel widget le attività in cui la risorsa è a supporto vengono visualizzate con una minore opacità dei colori di sfondo.
Per esempio, nell'immagine qui sotto, l'attività '00215/2017' risulta assegnata alla risorsa 'COPPOLELLA GIANLUCA', in data '25/01/2017'. Per questa attività la risorsa 'BELLINI ALFREDO' risulta a supporto. Infatti la stessa attività '00215/2017' compare anche nella sua cella per lo stesso giorno. Ciò viene denotato grazie alla differente gradazione dei colori, e al fatto che nel riepilogo vengono segnalate, tra parentesi, le attività e le ore a supporto (sul totale) di quella risorsa in quella giornata.



Qui un esempio di come viene renderizzata un'attività programmata in un dato giorno ed assegnata ad una certa risorsa, quando la si visualizza dal dettaglio calendario (risorsa/giorno) di un'altra risorsa, che è a supporto per quell'attività.



Parametri

- **params** Object, required. Contiene:
  - **className** String, required, gwClass name where the gwTimeTable works
  - **listedAttributesNames** String, optional, default null. gwAttributes name, comma separated (','). Allow to control which columns to show inside the gwTimetable list. If not provided the default for the gwClass will be used
  - **detailLayoutName** String, optional. detailLayout name used to open the gwClassDetail on the gwTimeTable's cells click. If omitted the gwClass's default detailLayout is used
  - **title** String, optional. A title used for the tab.
  - **gwdResources** String, optional, default null, alternative to gwdResourcesFilters. String, comma separated (',') list containing the cod_resource inside gwd_resource relation. It can be used the notation #{gw_activeUser} that a will be replaced with the session gwUser.
  - **gwdResourcesFilters** Object[], optional, alternative to gwdResources. Filter[] that impose selection criteria on gwd_resource records.
  - **nDays** Integer, optional, default 7. è un intero che rappresenta l'orizzonte temporale di programmazione attività che verrà mostrato all'utente quando aprirà la scheda gwTimeTable (parte sempre dal giorno risultante dall'impostazione del parametro fromTodayOffset).
  - **fromTodayOffset** Integer, opzionale, valore di default 1, è un intero che rappresenta l'offset (in giorni) della prima data mostrata nel widget, rispetto ad oggi. Può essere anche negativo. Es: se vale 0 la prima data mostrata sara oggi; se vale 1, domani.
  - **dayColumnWidth** Integer, optional, valore di default 120, è un intero che rappresenta la

dimensione in px della larghezza della colonna giorno della griglia della gwTimeTable. Può essere impostato anche con il valore 'auto', in tal caso lo spazio a disposizione in orizzontale viene equamente diviso fra il numero delle colonne individuato da **nDays**, senza aggiungere una scrollbar orizzontale
- **useWorkshift** Integer, optional, valore di default false, flag che consente di far interagire la gwTimeTable con il modulo dei turni. Ciò comporta che alla modifica dei parametri temporali (inizio e durata o inizio e fine) di un'attività assegnata (ad una risorsa) comporti un' interpolazione con i turni della risorsa per quel giorno. In caso di incongruenze la modifica verrà negata segnalando la cosa con un messaggio di errore. In sostanza l'effetto con useWorkshift posto a true è che non sarà possibile assegnare un attività ad una risorsa in un certo giorno fuori dal suo turno di lavoro. Inoltre se posto a true, farà comparire un controllo di tipo 'check' nella scheda di modifica dell'attività (con label 'Svincola da turni'), che permetterà di escludere quell'attività dall'incrocio con i moduli dei turni, rendendola di fatto programmabile per qualsiasi giorno ed ora.
- **readonly** Boolean, optional, default false, flag che consente di aprire la scheda in modalità di sola lettura. Ciò comporta che le schede di modifica delle attività saranno aperte in modalità visualizzazione, indipendentemente da tutti i permessi statici e dinamici impostati per la classe con nome className.
- **reportPath** String, optional, default null, se presente sara disponibile sulla toolbar un bottone che al click apre la report come da path, passando come parametro 'gwd_resources', String contenente una clausola fra parentesi tonde di valori (valori gwd_resource in ingresso al widget) String separati dal carattere ','. Esempio: gwd_resources: ('001','002').
- **openReportButtonLabel** String, optional, default null, valutato solo se reportPath è presente. E' la label (e tooltip) applicata al bottone che apre la report. Se omessa viene mostrato un bottone senza label e con un generico 'Apri' localizzato come tooltip.
- **hourSize** Integer, optional, default 100. The desired size (height) in pixels of one hour
- **timeSlotDuration** Integer, optional, default 15. The duration of minutes of the time slot (must be a divisor of 60)
- **minHours** Integer, optional, in range [0, 23], 8 default
- **maxHours** Integer, optional, in range [1, 36], 16 default

Gli ultimi 4 parametri influiscono sulla visualizzazione grafica della scheda dettaglio delle attività assegnate e/o programmate in un dato giorno. Questa scheda è basata principalmente sul componente **dojox/calendar/Calendar**. Sono tutti opzionali, e in caso di omissione sono utilizzati dei valori di default. Essi agiscono come descritto nella documentazione ufficiale visualizzabile qui.

Requisiti

- gwClass
  - jor_workshift_resource_wsinst (solo con modulo dei turni presente)
  - jor_odl (o un'altra gwClass appositamente costituita)
- groovy
  - jor_odl_trigger.groovy

JavaScript

Consulta l'apposita sezione di javascrip API apertura scheda gwTimeTable.

LeafItem

E' un leafItem che al click apre la scheda di tipo gwTimeTable. Il type che caratterizza il leafItem è *gwTimeTable*. Il type *gwTimeLine* è da considerarsi *deprecato*, ma ancora supportato.

Codice da inserire dentro un accordionPaneItem nell'xml di progetto:

```
<leafItem name="test_gw_time_table" label="Test gwTimeTable"
image="evt_avvisi_scadenza.png" type="gwTimeTable">
    <parameter name="className" value="jor_job_to_manage"
hideToClient="false"></parameter>
    <parameter name="detailLayoutName" value="who_n_when"
hideToClient="false"></parameter>
    <parameter name="listedAttributesNames"
value="job_order_code,pr_type,pr_expected_start_date,pr_expected_end_date,pr
_expected_timespan,job_order_description,address" hideToClient="false"
    ></parameter>
    <parameter name="gwdResources" value="001,002"
hideToClient="false"></parameter>
    <parameter name="readonly" value="false"
hideToClient="false"></parameter>
    <parameter name="nDays" value="15" hideToClient="false"></parameter>
    <parameter name="fromTodayOffset" value="1"
hideToClient="false"></parameter>
    <parameter name="dayColumnWidth" value="120"
hideToClient="false"></parameter>

    <parameter name="useWorkshift" value="true"
hideToClient="false"></parameter>

    <!--DAY DETAIL GHRAPHIC PARAMS STARTS-->
    <parameter name="hourSize" value="60"
hideToClient="false"></parameter><!-- The desired size in pixels of one
hour, 100 default -->
    <parameter name="timeSlotDuration" value="15"
hideToClient="false"></parameter><!-- The duration of minutes of the time
slot (must be a divisor of 60), 15 default -->
    <parameter name="minHours" value="8"
hideToClient="false"></parameter><!-- in range [0, 23], 8 default-->
    <parameter name="maxHours" value="18"
hideToClient="false"></parameter><!-- in range [1, 36], 16 default-->

    <parameter name="reportPath" value="path/to/report.jasper"
hideToClient="false"></parameter>
    <parameter name="openReportButtonLabel" value="Stampa Programmazione
Settimanale" hideToClient="false"></parameter>
</leafItem>
```

---

gwArchivedProcessList

(torna a elenco schede)

Scheda per la visualizzazione delle informazioni delle istanze di processo terminate relative al processo con processDefinitionKey (uguale al parametro gwProcessName). Al click sulle righe della

lista verranno aperte le classiche form del gwClassDetail. In caso di gwProcess di Tipo PROCEDURE , verrà invece aperta l'apposita interfaccia riepilogativa. Al click sull'ultima colonna si aprirà il diagramma del flusso seguito dalla particolare istanza di processo.

Documentazione completa

---

gwClassProcesses

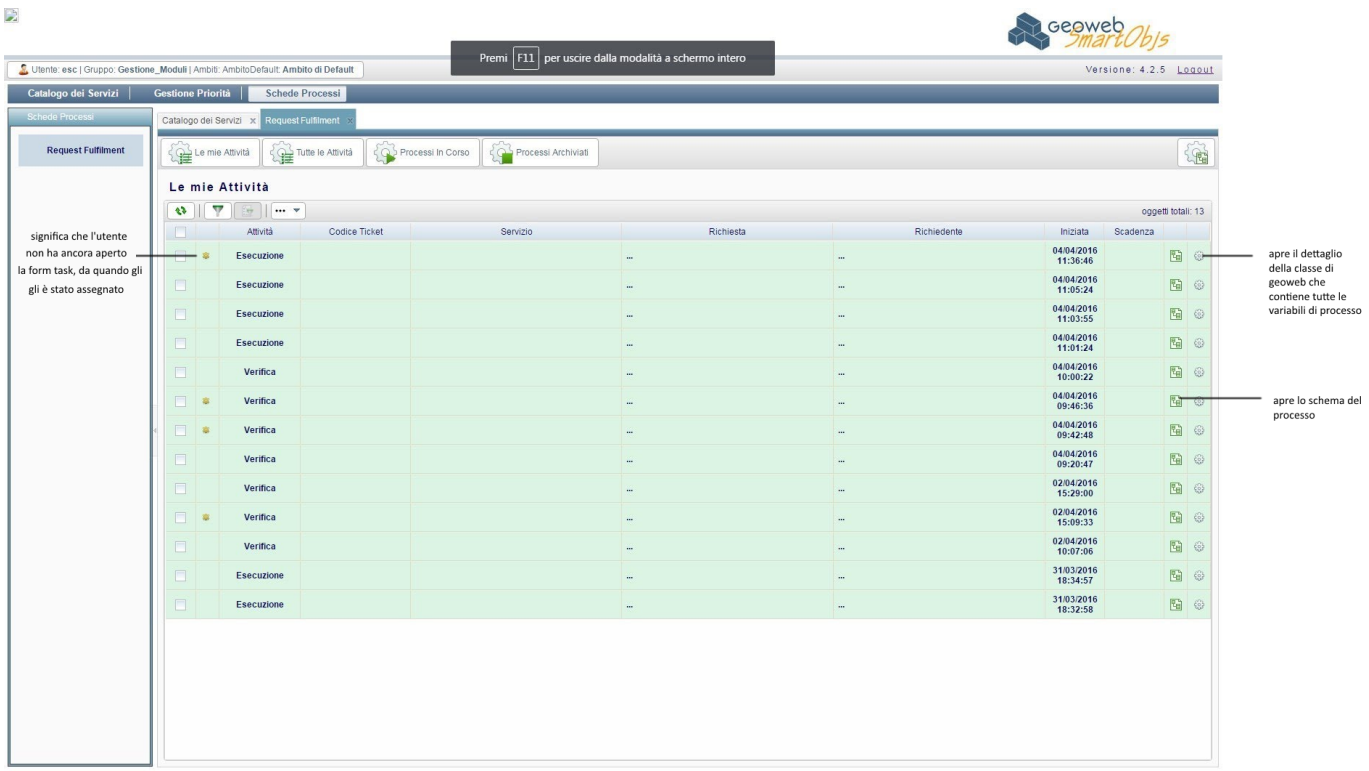
(torna a elenco schede)

In Geoweb è presente una scheda per la visualizzazione e la gestione dei processi, dove fra le altre cose, è possibile:

- avviare una nuova istanza di processo
- vedere tutte le proprie attività
- completare le proprie attività
- vedere le attività degli altri utenti coinvolti nel processo (in sola lettura)
- richiedere per se (Claim) lo svolgimento di tutti quei Task che non sono stati ancora assegnati ma prevedono una lista di utenti candidati od una lista di gruppi candidati a svolgere quella attività (fra i quale si è presenti).
- consultare la lista dei processi in corso
- consultare la lista dei processi terminati (in sola lettura)
- consultare un immagine grafica del modello .bpmn del processo
- consultare un immagine grafica del modello .bpmn per i singoli task, processi in corso, e processi terminati

Liste Attività ('*Mie*' e '*Tutte*')
I record di tali liste hanno:

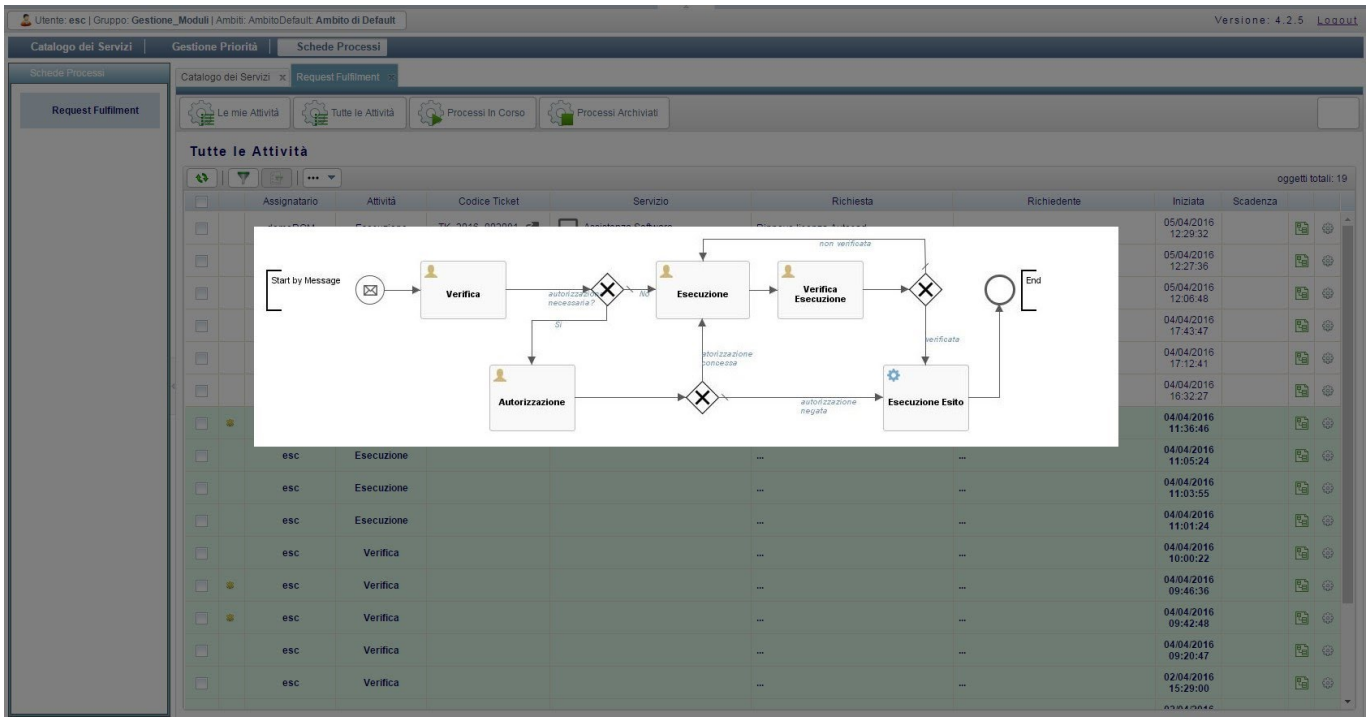
- Sfondo verde: se il task è assegnato da me.
- Sfondo Giallo: se il task è assegnato a una specifica lista di utenti, o ad una specifica lista di gruppi, nei quali è invcluso l'utente corrente, od il gruppo dell'utente corrente.
- Sfondo Bianco: se il task è assegnato ad un altro utente oppure non l'utente corrente non risulta nella lista degli utenti candidati all'esecuzione.



Se è presente il marker nella prima colonna significa che la form del task non è mai stata aperta dall'utente dopo che gli è stata assegnata tale attività dal motore di workflow.

In queste schede ci sono colonne fisse (nome attività, assegnatario, data inizio, scadenza, tasto apertura diagramma, tasto apertura record istanza processo) e colonne variabili (individuate in base agli attributi della classe di processo messi in lista)

Qui sotto un esempio di diagramma di processo.



Liste Istanze di processo ('*In Corso*' e '*Archiviate*')

Questo è la parte di codice da aggiungere all'xml di Progetto Geoweb per far comparire una voce di menu che al click apra la scheda Processo:

```
<leafItem name="request_fulfilment" label="Request Fulfilment"
image="process.png" type="gwClassProcesses">
    <parameter name="className" value="rqm_wpi_request_fulfilment"
hideToClient="false"></parameter>
    <parameter name="gwProcessName" value="rqm_wpi_request_fulfilment"
hideToClient="false"></parameter>
</leafItem>
```

---

gwRunningProcessList

(torna a elenco schede)

Scheda per la visualizzazione delle informazioni delle istanze di processo attive relative al processo con processDefinitionKey (uguale al parametro gwProcessName). Al click sulle righe della lista verranno aperte le classiche form del gwClassDetail. In caso di gwProcess di Tipo PROCEDURE verrà invece aperta l'apposita interfaccia riepilogativa. Al click sull'ultima colonna si aprirà il diagramma del flusso seguito dalla particolare istanza di processo.

Documentazione completa

---

gwStartProcessInstanceAction

leafItem che apre la form di avvio di processo (openGwProcessStartTaskForm())

```
<leafItem name="eqm_wpi_cespiti_new" label="Nuovo Processo" image="fas fa-
play-circle icon32x32 iconMainColor" type="gwStartProcessInstanceAction">
          <parameter name="gwProcessName" value="eqm_wpi_cespiti"
hideToClient="false"></parameter>
          <parameter name="openingDetailMode" value="tab"
hideToClient="false"></parameter>
          <parameter name="startFormKey" value="wpi_new"
hideToClient="false"></parameter>
 </leafItem>
```

(torna a elenco schede)

---

gwStartProcessInstanceByMessageAction

leafItem che avvia il processo by message (sotto usa l'api js startProcessInstanceByMessage())

```
<leafItem name="eqm_wpi_cespiti_new" label="Nuovo Processo" image="fas fa-
play-circle icon32x32 iconMainColor"
type="gwStartProcessInstanceByMessageAction">
          <parameter name="message"
value="start_by_message_eqm_wpi_cespiti" hideToClient="false"></parameter>
          <parameter name="openingDetailMode" value="tab"
hideToClient="false"></parameter>
          <parameter name="processVariables" value="{a: 1, b: 'c'}"
hideToClient="false"></parameter>
 </leafItem>
```

(torna a elenco schede)

---

gwSuspendedProcessList

(torna a elenco schede)

---

gwTaskHierarchicalFilterMenu

(torna a elenco schede)

---

gwTaskDetail

(torna a elenco schede)

Questa scheda permette di aprire una form contenente le informazioni relative ad un task.

Documentazione completa

---

gwTaskList

(torna a elenco schede)

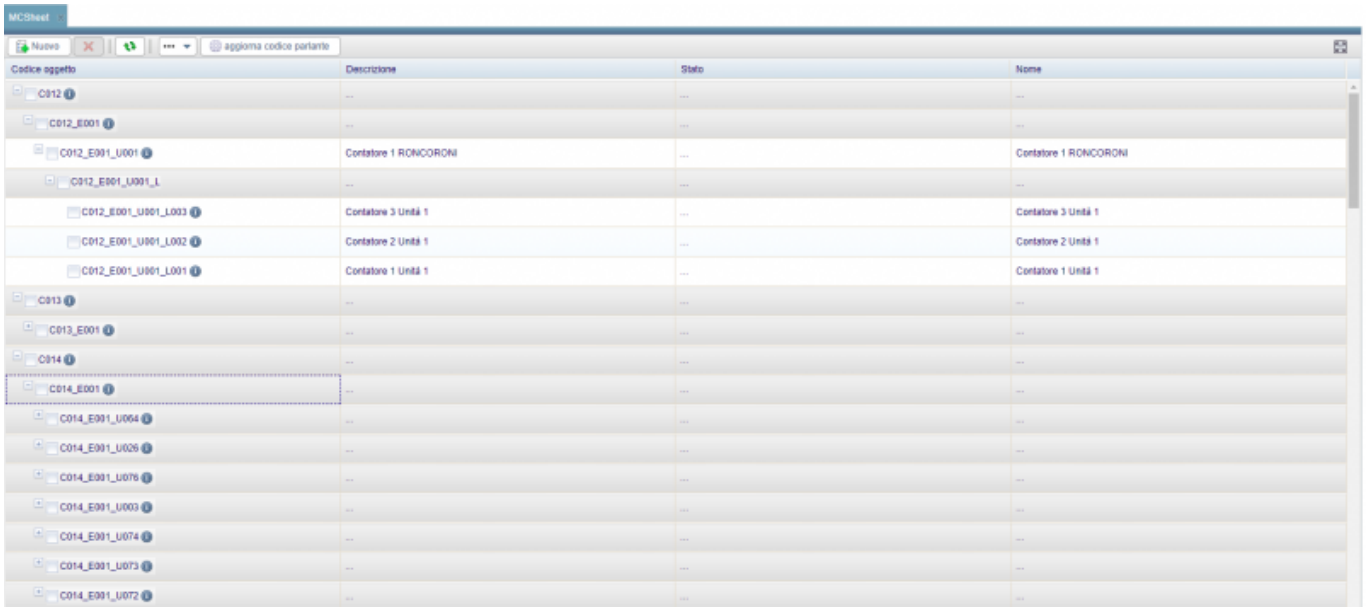
Scheda per la visualizzazione dei task relativi alle istanze di processo generate dalla definizione di processo con processDefinitionKey (uguale al parametro gwProcessName).

Documentazione completa

---

gwMnemonicCodeSheet

(torna a elenco schede)



Con la scheda di tipo gwMnemonicCodeSheet viene aperta una griglia gerarchica, che permette di visualizzare e gestire il contenuto dei record di una classe Geoweb tramite una visualizzazione ad albero. Termini equivalenti con i quali si fa riferimento alla griglia di Geoweb in questa guida sono: lista, grid, gwMnemonicCodeSheet.

PARAMETRI

Parametro	Descrizione
gwClassName	String, richiesto, il nome della classe Geoweb della quale si vuole aprire la lista dei record

staticFilters	Object [], opzionale, default null, permette di imporre criteri di filtro per poter mostrare solo i record di dati aderenti. Array di oggetti JSON filtro [TODO MAKE LINK] del tipo:
	`{`
	`condition: 'AND',`
	`columnName: 'cod_item',`
	`operator: 'START',`
	`filterType: 'STRING',`
	`value: ['C001']`
	`}`

mc_name	String , richiesto, il nome del MnemonicCode di riferimento per questa scheda, configurato nella classe espressa nel parametro *gwClassName*
ColumnsViews	String , opzionale, lista dei nomi degli attributi da mostrare nella griglia dei record. Se lasciato vuoto mostrerà gli attributi in lista della classe. **ATTENZIONE**: questo parametro diventa obbligatorio nel caso in cui si intenda definire anche i parametri *ColumnsLabels* e/o *ColumnsWidths*
ColumnsLabels	String , opzionale, lista delle etichette da mostrare nella griglia dei record, rispettivamente corrispondenti agli attributi definiti nel parametro *ColumnsViews*. **ATTENZIONE**: la lista di etichette definita in questo parametro deve avere lo stesso numero di elementi della lista definita nel parametro *ColumnsViews*

Parametro	Descrizione
ColumnsWidths	String , opzionale, lista delle misure d'ampiezza delle colonne della griglia dei record, rispettivamente corrispondenti agli attributi definiti nel parametro *ColumnsViews*. **ATTENZIONE**: la lista di ampiezze definita in questo parametro deve avere lo stesso numero di elementi della lista definita nel parametro *ColumnsViews*
consistentHierarchy	Boolean, opzionale, default *false*, indica se la gerarchia del codice parlante utilizzato per costruire l'albero dei record è di tipo consistente o non consistente. Una gerarchia definita in modo consistente non permette la presenza nell'albero di elementi non 'esistenti', ovvero elementi legati ad un codice che non rappresenta un record realmente esistente. **ATTENZIONE**: questo parametro è ancora in fase di sviluppo, pertanto va lasciato *false* al momento attuale
hiddenActionsNamesList	ArrayList<String>, opzionale, default *null*, lista separata da virgola dei nomi delle azioni che si desidera non compaiano in griglia: nonostante siano state configurate correttamente nella classe (definita nel parametro *gwClassName*), non verranno comunque mostrate nella scheda

Nella parte superiore della scheda è presente una toolbar che, a seconda del set di permessi (statici e dinamici) impostati nel Geoweb Admin, mostra tutta una serie di bottoni, in ordine da sinistra a destra:

- **Nuovo**: visibile solo per un utente con il permesso di inserimento, al click apre un dettaglio (su tab o floatingPane, a seconda della configurazione) con una form dove inserire i dati del nuovo record. Se, prima del click, è stato selezionato (tramite click sulla riga) un record dalla griglia, la form da compilare verrà pre-popolata con il codice del record selezionato, in modo da creare un “figlio” (ovvero un record gerarchicamente dipendente) del suddetto record.
- **Cancella**: visibile solo per un utente con il permesso di eliminazione, attivo solo con uno o più record selezionati (tramite caselle al lato della riga), al click apre un dialog modale dove viene richiesta una conferma: in caso di risposta affermativa il record verrà elimato dal DB e rimosso dall'albero.
- **Ricarica**: sempre mostrato, al click esegue il refresh del set di dati dell'albero.
- **Azioni Utente**: bottone che al click apre un menu a tendina contenente tanti menuItem, i quali al click eseguono specifiche azioni, o aprono reportistiche. I menuItem report sono mostrati se esistone le relative report di classe. I menuItem di tipo azione sono mostrati se esistono le relative azioni e se sono attivi i relativi permessi di esecuzione per l'utente che sta visualizzando la griglia. In caso di nessuna azione e nessuna report configurate, il bottone non viene visualizzato.
- **Azioni in Toolbar**: numero variabile di bottoni, che al click eseguono specifiche azioni javascript, mostrati solo in presenza di azioni configurate e con l'utente in possesso dei relativi permessi di esecuzione.

Su ogni riga della griglia, nella prima colonna a sinistra, è disponibile un check, che serve per selezionare la riga ed attivare una serie di funzioni (come elimina o esporta), oltre che il pulsante + utilizzabile per espandere il nodo di quel record e mostrarne i *figli*. È infine presente un pulsante *info*, il quale permette di visualizzare le informazioni relative al codice mnemonico del rispettivo record in due modi, in base alla configurazione del MnemonicCode a cui questo componente fa riferimento:

- sotto forma di dettaglio del MnemonicCodeWidget, se il record contiene un attributo codice di tipo MnemonicCodeWidget.
- sotto forma di tooltip, se il record contiene un differente tipo di attributo che ne rappresenta comunque il codice.

ATTENZIONE: il pulsante *info* non compare in quelle righe che rappresentano solo un raggruppatore, e che quindi non corrispondono ad un vero e proprio record esistente.

Le colonne successive sono generate in automatico, a partire dagli attributi configurati nel parametro *ColumnsViews* di questo componente. Si può impostare l'ordine della singola colonna, oltre che la larghezza (tramite il parametro *ColumnsWidths*).

Al click su un record della griglia si evidenzia tale record, mentre cliccando di nuovo si rimuove tale selezione. Questo permette, nel caso in cui si cliccasse successivamente sul pulsante *Nuovo*, di creare un nuovo record il cui codice mnemonico sarebbe precompilato con quello del record evidenziato. Se il click avviene su una riga corrispondente ad un record esistente nella classe su cui questo componente è configurato, si aprirà il dettaglio di classe relativo a quest'ultimo. La modalità di apertura di default è floatingPane. questa modalità può essere sovrascritta dall'xml dei Progetto Geoweb, impostandola su tab. [TODO MAKE LINK]

Requisiti

Per funzionare il componente ha bisogno di questi vincoli:

- esistenza della classe con nome uguale al parametro 'gwClassName'
- esistenza del MnemonicCode con nome uguale al parametro 'mc_name' correttamente configurato nella classe di cui sopra
- 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*)

JavaScript

```
var gwClassName = 'gwd_item';
var mcName = 'mc_cod_item';
var tabWidgetTitle = 'Scheda gerarchica'; //optional
var parametersMap = { //optional
    staticFilters: [
        {condition:'AND', columnName:'cod_wbs', operator:'START',
filterType:'STRING', value: ['R3']}
    ],
    ColumnsViews: 'id_item,name_item',
    ColumnsLabels: 'ID,Nome',
    ColumnsWidths: 'auto, auto',
    consistentHierarchy: false,
    hiddenActionsNamesList: 'importCSV, exportSelected'
};
var insertIndex = 0; //optional

openGwMnemonicCodeSheetTab(gwClassName, mcName, tabWidgetTitle,
parametersMap, insertIndex);
```

O, con la notazione più generale:

```
var tabWidgetId = createTabId(tabWidgetType, identifier);
var tabWidgetType = 'gwMnemonicCodeSheet';
```

```
var tabWidgetTitle = 'Scheda gerarchica';
var parametersMap = {
    staticFilters: [
        {condition:'AND', columnName:'cod_wbs', operator:'START',
filterType:'STRING', value: ['R3']}
    ],
    ColumnsViews: 'id_item,name_item',
    ColumnsLabels: 'ID,Nome',
    ColumnsWidths: 'auto, auto',
    consistentHierarchy: false,
    hiddenActionsNamesList: 'importCSV, exportSelected'
};
var insertIndex = 0; //optional

openTab(tabWidgetId, tabWidgetType, tabWidgetTitle, parametersMap,
insertIndex);
```

LeafItem
Esempio:

```
<leafItem name="menu_name" label="MCSheet" image="classificazione.png"
type="gwMnemonicCodeSheet">
  <parameter name="gwClassName" value="gwd_item"
hideToClient="false"></parameter>
  <parameter name="staticFilters"
       value="[{condition:'AND', columnName:'cod_wbs', operator:'START',
filterType:'STRING', value: ['R3']}]"
       hideToClient="false"></parameter>
  <parameter name="mc_name" value="mc_cod_item"
hideToClient="false"></parameter>
  <parameter name="ColumnsViews" value="id_item,name_item"
hideToClient="false"></parameter>
  <parameter name="ColumnsLabels" value="ID,Nome"
hideToClient="false"></parameter>
  <parameter name="ColumnsWidths" value="auto,auto"
hideToClient="false"></parameter>
  <parameter name="consistentHierarchy" value="true"
hideToClient="false"></parameter>
  <parameter name="hiddenActionsNamesList" value=""
hideToClient="false"></parameter>
</leafItem>
```

---

gwPhoto360

Requisisti
Esistenza gwClass **gwd_phototag**, od equivalente. La classe deve avere un field **url** (string) contenente il path relativo (alla cartella dei contenuti statici) del file immagine 360. Se la classe contiene un field **name**, ed il record lo ha valorizzato, questo valore viene usato come titolo della

scheda, se non esplicitamente forzato. (stesso discorso per il field **description**, in assenza di **name**).

JavaScript
Consulta l'apposita sezione di javascrip API apertura scheda gwPhoto360.

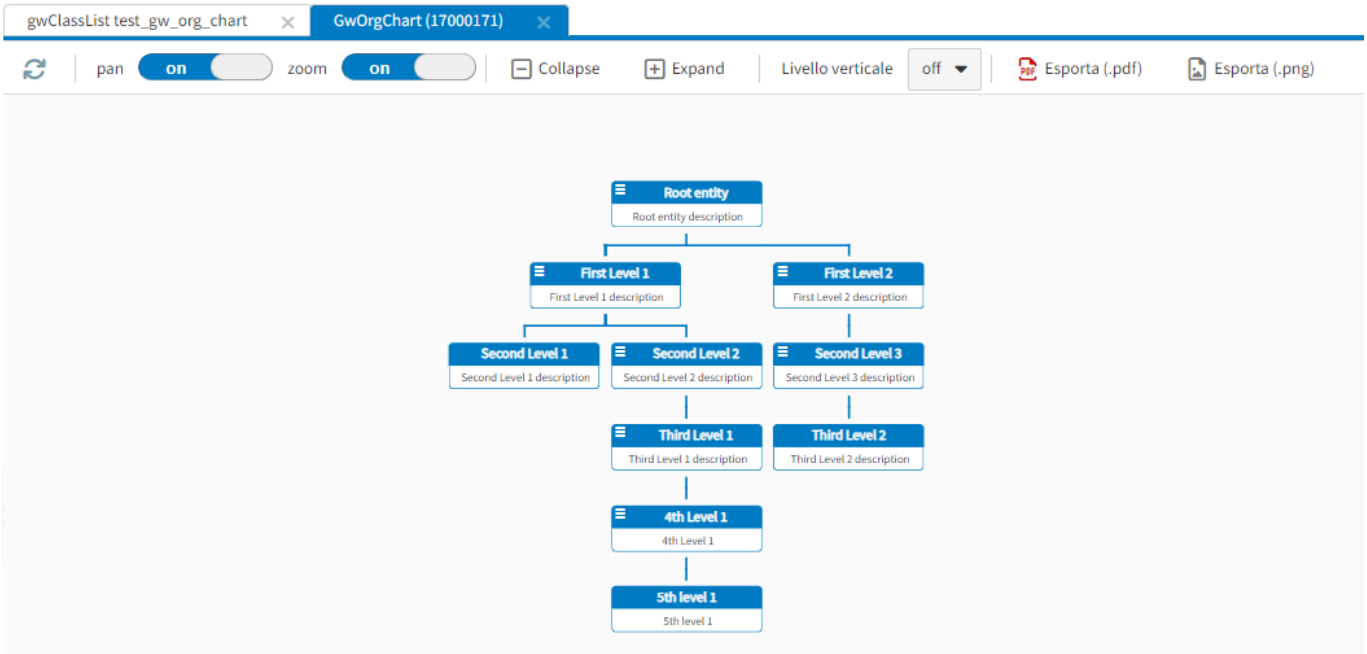
LeafItem
Dato che la scheda è legata ad uno specifico record di classe che ospita l'immagine, è inusuale che venga aperta da leafItem nel menu di progetto.

```
<leafItem name="menu_name" label="Photo 360" image="gwPhoto360.png"
type="gwPhoto360">
    <parameter name="itemId" value="123" hideToClient="false"></parameter>
</leafItem>
```

```
<leafItem name="menu_name" label="Photo 360" image="gwPhoto360.png"
type="gwPhoto360">
    <parameter name="itemId" value="123" hideToClient="false"></parameter>
    <parameter name="gwClassName" value="gwd_phototag_2"
hideToClient="false"></parameter>
</leafItem>
```

gwOrgChart

La scheda gwOrgChart permette di visualizzare ed esportare in forma di grafo le relazioni gerarchiche fra varie entità (omogenee e non).

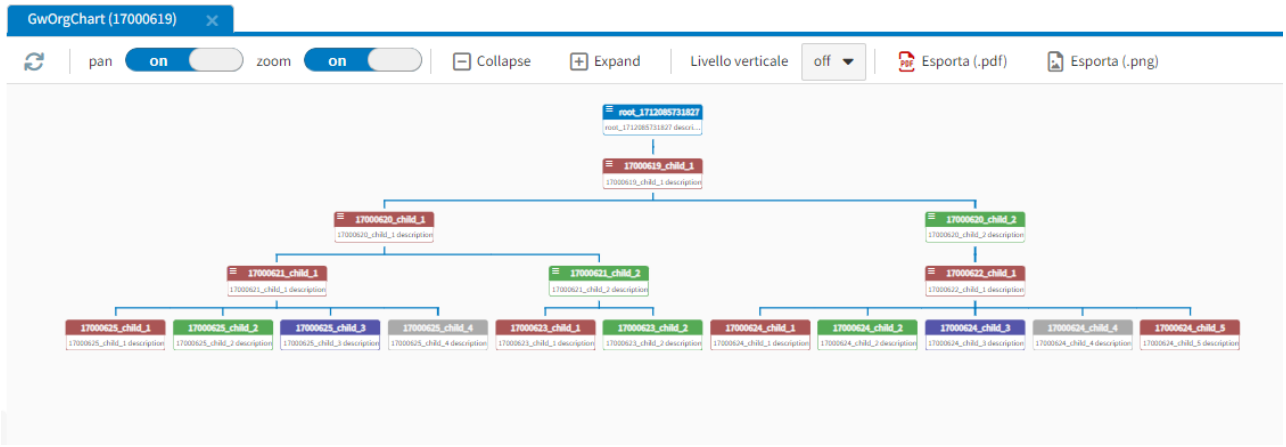


Caratteristiche

- permette di navigare fra la gerarchia espandendo/collassando entità children, parent/siblings (da grafo)



- permette di abilitare il pan per navigare fra gerarchie estese (parametro iniziale e poi modificabile da toolbar)
- permette di abilitare lo zoom tramite rotellina del mouse (parametro iniziale e poi modificabile da toolbar)
- espone funzionalità per espandere/collassare il grafo (da toolbar)
- permette di configurare l'orientamento con il quale si sviluppa il grafo (parametro iniziale)
- permette di stilizzare le entità omogenee (ma anche singolarmente) (configurazione iniziale)



- permette l'esportazione del grafo come immagine .png (da toolbar)
- permette l'esportazione del grafo come file .pdf (da toolbar)
- permette di impostare un livello che rappresenta la profondità massima del grafo. Gli elementi di livello superiore ad esso NON verranno visualizzati (parametro iniziale)
- permette di impostare un livello rappresentante la profondità iniziale del grafo. Gli elementi di livello superiore saranno visualizzati, se presenti, solo agendo sulle frecce di espansione degli elementi figlio oppure tramite il button in toolbar 'Exapand All' (parametro iniziale)
- permette di impostare un livello rappresentante la profondità del grafo, a partire dalla quale, in ottica di migliorarne la leggibilità, gli elementi vengono visualizzati in verticale. (parametro iniziale e poi modificabile da toolbar)

Requisisti

Esistenza di una tabella/vista di supporto, opportunamente preparata con questi i seguenti **nomi dei campi fissi**:

- **entity_id** integer NOT NULL chiave dell'entità corrente
- **entity_parent_id** integer chiave che lega l'entità corrente dell'entità padre
- **entity_name** String NOT NULL titolo del nodo, che compare nella banda colorata
- **entity_descr** String descrizione del nodo che comparirà nel corpo dello stesso (E' supportata tutta la **notazione HTML per la stilizzazione** compresi i ritorni a capo <*br*>)
- **gw_class_name** String nome della classe, da usare congiuntamente a item_id per aprire eventualmente il dettaglio di classe
- **item_id** String chiave del record, da usare congiuntamente a gw_class_name per aprire eventualmente il dettaglio di classe
- **hex_color** String se popolato, con una stringa rappresentante un colore esadecimale (ES: 'AA0000'), il valore verrà usato per stilizzare il nodo dell'entità corrente

Questa tabella/vista può essere preparata anche a partire da tabelle diverse, afferenti ad entità non omogenee. L'omogeneità per la scheda gwOrgChart verrà garantita dal sistema di campi fissi.

script Postgres

gw_org_chart_postgres.sql

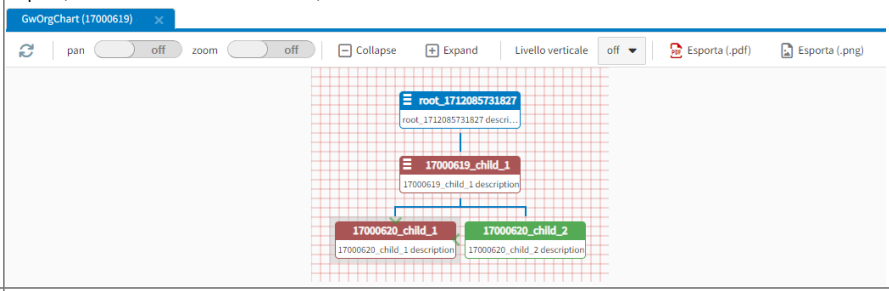
```
-- Table: test_gw46data.test_gw_org_chart

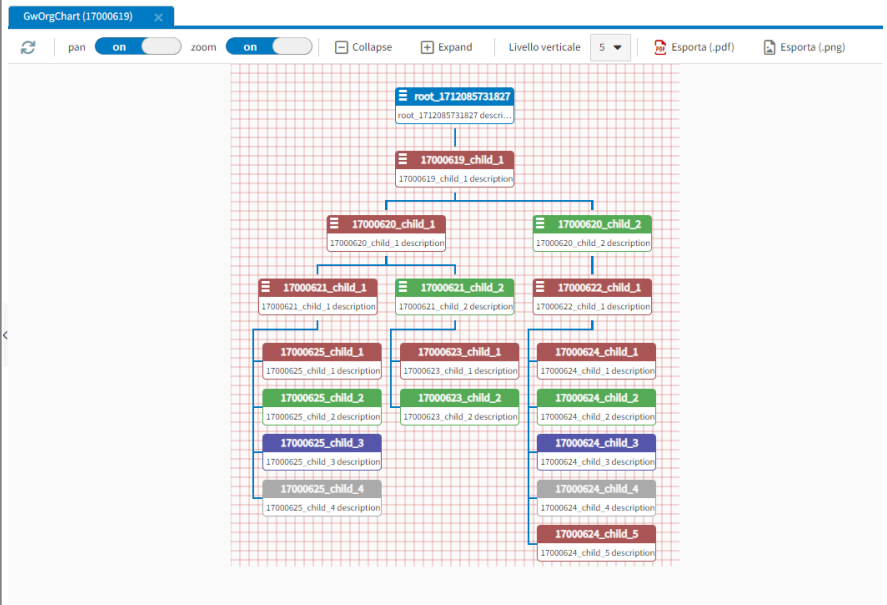
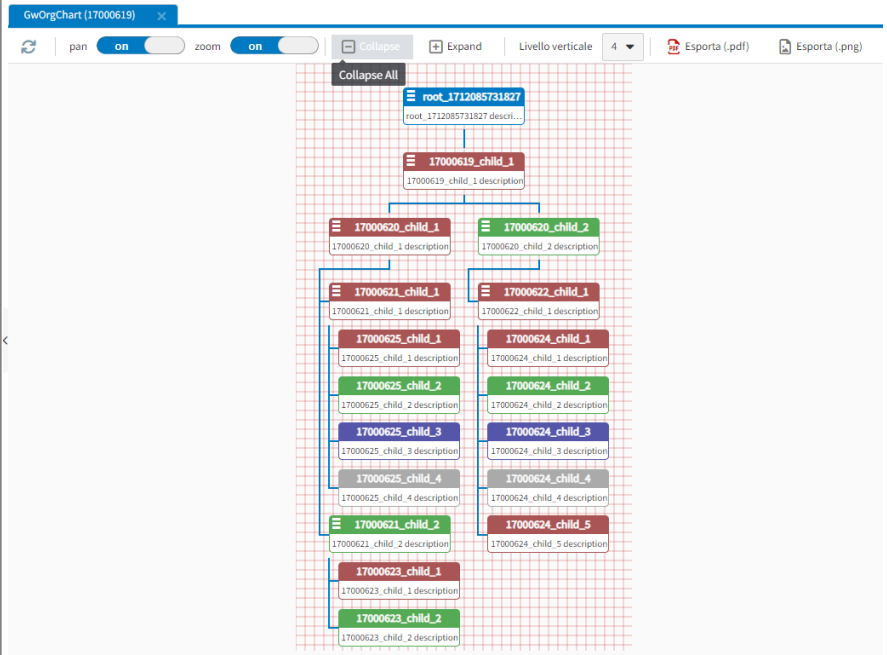
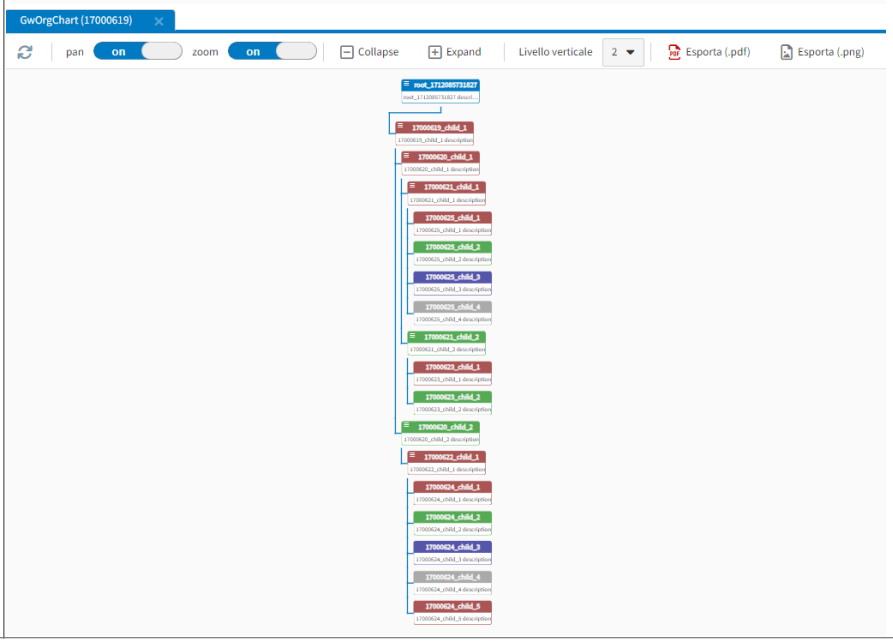
-- DROP TABLE test_gw46data.test_gw_org_chart;

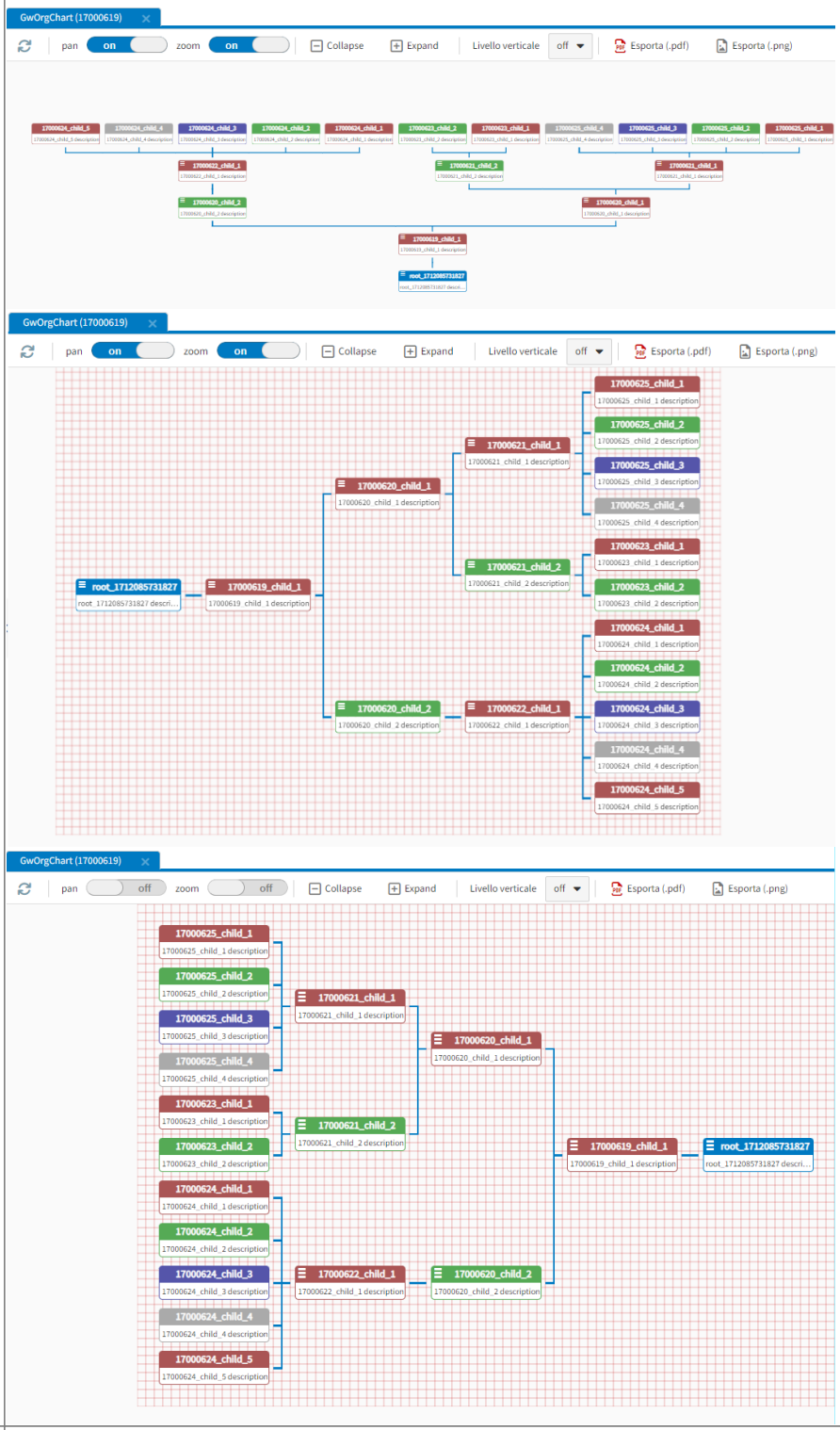
CREATE TABLE IF NOT EXISTS test_gw46data.test_gw_org_chart
(
    entity_id INTEGER NOT NULL,
    entity_parent_id INTEGER,
    entity_name CHARACTER VARYING COLLATE pg_catalog."default" NOT
NULL,
    entity_descr CHARACTER VARYING COLLATE pg_catalog."default",
    gw_class_name CHARACTER VARYING COLLATE pg_catalog."default",
    item_id CHARACTER VARYING COLLATE pg_catalog."default",
    hex_color CHARACTER VARYING COLLATE pg_catalog."default",
    CONSTRAINT test_gw_org_chart_pkey PRIMARY KEY (entity_id)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE test_gw46data.test_gw_org_chart
    OWNER TO test_gw46data;
```

PARAMETRI

Parametro	Descrizione
relationName	String, richiesto, è il nome della tabella/vista appositamente preparata per erogare i dati della gwOrgChart.
entityid	String, richiesto, default null, Chiave dell'elemento root dal quale si vuole fra partire il grafo (può anche essere un elemento a sua volta discendente da altre entità, le quali verranno escluse)
pan	Boolean, optional, default false. flag che abilita fin dall'inizio la funzionalità di pan (che può essere abilitata anche in seguito da comando in toolbar)
zoom	Boolean, optional, default false. flag che abilita fin dall'inizio la funzionalità di zoom (che può essere abilitata anche in seguito da comando in toolbar)
depth	Integer, optional, default 999. Intero rappresentante la profondità massima del grafo. Gli elementi di livello superiore a depth (con elemento root = livello 1) NON verranno visualizzati 
visibleLevel	Integer, optional, default 999. Intero rappresentante la profondità iniziale del grafo. Gli elementi di livello superiore a visibleLevel (con elemento root = livello 1) saranno visualizzati, se presenti, solo agendo sulle frecce di espansione degli elementi figlio oppure tramite il button in toolbar 'Exapand All'

Parametro	Descrizione
verticalLevel	Integer, optional, default null. Intero rappresentante la profondità del grafo, a partire dalla quale, in ottica di migliorarne la leggibilità, gli elementi vengono visualizzati in verticale.   

Parametro	Descrizione
direction	String, optional, default 't2b'. Possibili valori: **['t2b', 'b2t', 'l2r', 'r2l']**. Determina la direzione nella quale si sviluppa il grafo. Rispettivamente: dall'alto verso il basso (default), dal basso verso l'alto, da sinistra verso destra e da destra verso sinistra 
updateOnChangeInGwClassNameList	String, optional, default null. Lista di gwClassName, separati da virgola (',') tali per cui la scheda si aggiorna in automatico di fronte ad un evento di insert/update/delete su quelle classi
openDetailOnDClick	Boolean, optional, default false. Quando è a true e sono presenti gw_class_name e item_id sul DB, viene aperto il relativo dettaglio di classe al doppio click
nodeWidth	Integer, optional, default 130 (px). Quando impostato rappresenta la width, in pixel, applicata a tutti i nodi dell'orgChart
nodeHeight	Integer, optional, default 20 (px). Quando impostato rappresenta la height (della sola parte descrittiva del nodo), in pixel, applicata a tutti i nodi dell'orgChart

JavaScript

Consulta l'apposita sezione di javascrip API apertura scheda gwOrgChart .

LeafItem

Dato che la scheda è legata ad uno specifico record di classe che ospita l'immagine, è inusuale che venga aperta da leafItem nel menu di progetto. E' tuttavia possibile valorizzare entityId tramite espressioni ${} che fanno riferimento a variabili in sessione.

Set minimale

```
<leafItem name="li_1" label="all defaults"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
</leafItem>
```

Refresh scheda su eventi insert/update/delete di tre classi

```
                <leafItem name="li_2" label="pan"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
                    <parameter name="pan" value="true"
hideToClient="false"></parameter>
                    <parameter name="updateOnChangeInGwClassNameList"
value="test_gw_org_chart,test_gw_org_chart_2,test_gw_org_chart_3"
hideToClient="false"></parameter>
                </leafItem>
```

zoom, pan , verticalLevel 3 abilitati (e comunque sia modificabili dalla toolbar in seguito all'apertura)

```
                <leafItem name="li_3" label="zoom"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
                    <parameter name="zoom" value="true"
hideToClient="false"></parameter>
                    <parameter name="pan" value="true"
hideToClient="false"></parameter>
                    <parameter name="verticalLevel " value="3"
hideToClient="false"></parameter>
                </leafItem>
```

Il massimo numero di livelli visualizzati è 3

```
                <leafItem name="li_4" label="depth 3"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
                    <parameter name="depth" value="3"
hideToClient="false"></parameter>
                </leafItem>
```

All'inizio vengono visualizzati aperti solo i primi 3 livelli. Nel caso ce ne fossero altri è comunque sia possibile espanderli tramite le apposite frecce che compaiono su on hover del mouse sul nodo

```
                <leafItem name="li_5" label="visibleLevel 2"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
                    <parameter name="visibleLevel" value="3"
hideToClient="false"></parameter>
                </leafItem>
```

grafo da sinistra a destra

```
                <leafItem name="li_6" label="verticalLevel 3"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
                    <parameter name="direction" value="l2r"
hideToClient="false"></parameter>
                </leafItem>
```

apertura dettaglio con doppio click

```
                <leafItem name="li_6" label="verticalLevel 3"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
                    <parameter name="openDetailOnDClick" value="true"
```

```
hideToClient="false"></parameter>
                </leafItem>
```

width nodi impostata a 200px

```
                <leafItem name="li_7" label="nodeWidth 200"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
                    <parameter name="nodeWidth" value="200"
hideToClient="false"></parameter>
                </leafItem>
```

height parte descrittiva nodi impostata a 80px

```
                <leafItem name="li_7" label="nodeWidth 200"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000619"
hideToClient="false"></parameter>
                   <parameter name="nodeHeight" value="80"
hideToClient="false"></parameter>
                </leafItem>
```

esempio di utilizzo di tutti i parametri

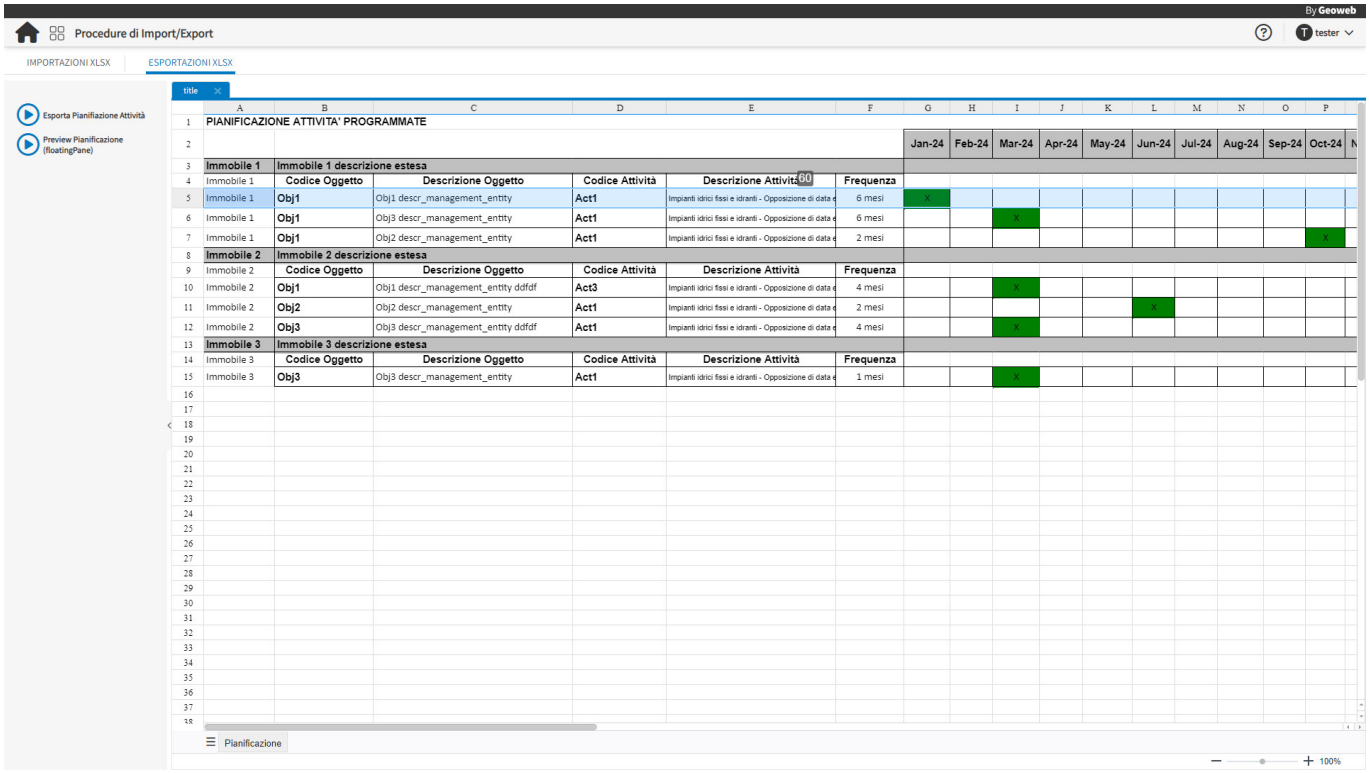
```
                <leafItem name="li_8" label="direction left to right"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwOrgChart">
                    <parameter name="relationName" value="test_gw_org_chart"
hideToClient="false"></parameter>
                    <parameter name="entityid" value="17000171"
hideToClient="false"></parameter>
                    <parameter name="direction" value="l2r"
hideToClient="false"></parameter>
                    <parameter name="depth" value="5"
hideToClient="false"></parameter>
                    <parameter name="visibleLevel" value="3"
hideToClient="false"></parameter>
                    <parameter name="zoom" value="true"
hideToClient="false"></parameter>
                    <parameter name="pan" value="true"
hideToClient="false"></parameter>
```

```
                    <parameter name="verticalLevel " value="4"
hideToClient="false"></parameter>
                    <parameter name="openDetailOnDClick" value="true"
hideToClient="false"></parameter>
                    <parameter name="updateOnChangeInGwClassNameList"
value="test_gw_org_chart" hideToClient="false"></parameter>
                    <parameter name="nodeWidth" value="200"
hideToClient="false"></parameter>
                    <parameter name="nodeHeight" value="80"
hideToClient="false"></parameter>
                </leafItem>
```

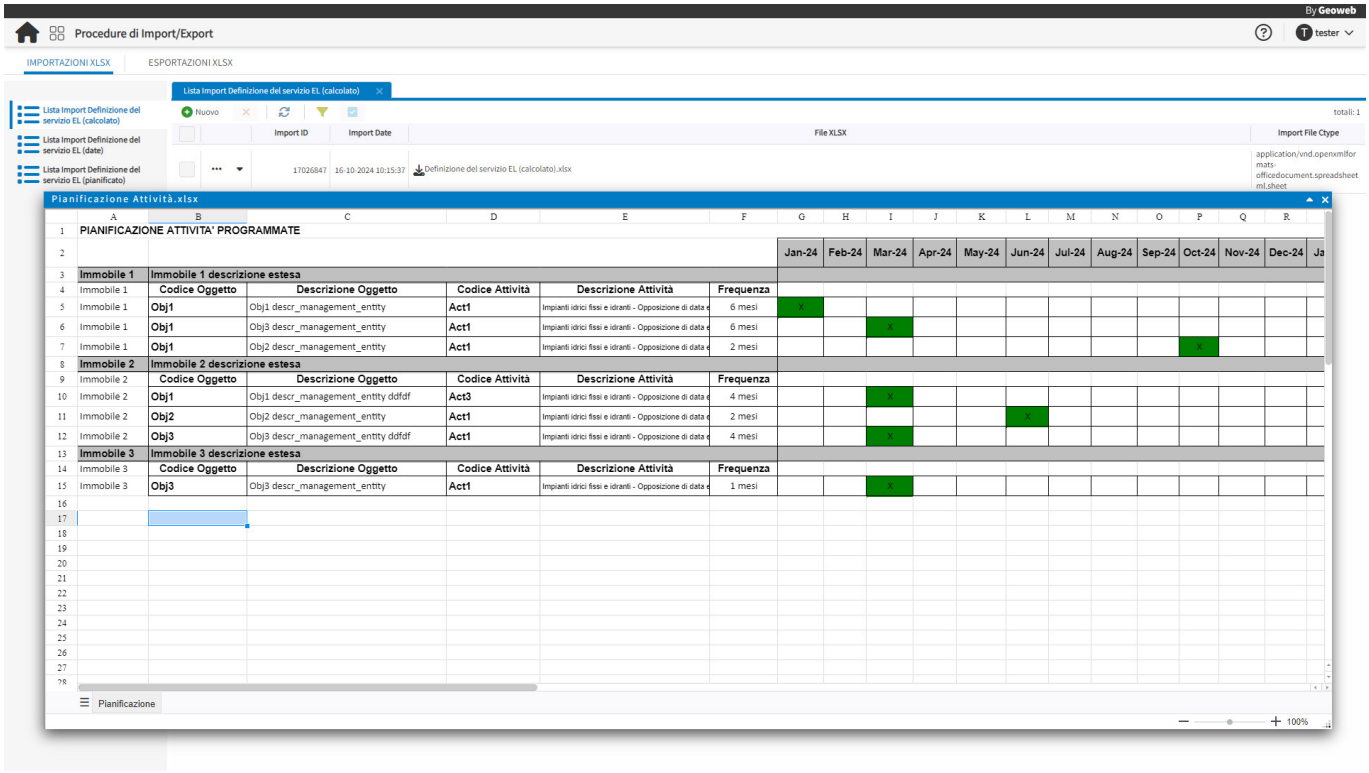
gwXlsxViewer

La scheda/floatingPane gwXlsxViewer rende disponibile una preview di file .xlsx all'interno del browser.

Come tab



Come floatingPane



Introdotta in **4.7.3** (issue #1417)

Il visualizzatore è basato su LuckySheet

PARAMETRI

Parametro	Descrizione
href	String, required, default gwContextPath+'/rest/exportByGroovyExecution'. url del controller che genera i bytes
method	String, required, default 'POST' method associato ad href
fileName	String, required, nome del file che verrà generato
data	Object, optional, Dati da postare in acciazione a href e method Es: { groovy: 'export_pianificazione.groovy', fileName: fileName, otherStuff: 1 }
viewerOptions	Object, optional, default null. Opzioni passabili al visualizzatore basato su LuckySheet per sovrascrivere il comportamento di default (full readonly). Parametri https://dream-num.github.io/LuckysheetDocs/guide/config.html#configuration-item

JavaScript

Consulta l'apposita sezione di javascrip API Apertura gwXlsxViewer .

LeafItem

Non dovrebbe essere mai integrato come leafItem, ma è supportato. Set minimale, usando i vari default

```
<leafItem name="li_1" label="all defaults"
image="eyJjc3NDbGFzcyI6ImZhLXNvbGlkIGZhLXNpdGVtYXAiLCJ3aWR0aCI6IjMycHgiLCJoZ
WlnaHQiOiIzMnB4IiwiY29sb3IiOiIjMDA3QUMyIn0=" type="gwXlsxViewer ">
    <parameter name="data" value="{ groovy: 'export_pianificazione.groovy',
fileName: 'preview.xlsx'}" hideToClient="false"></parameter>
</leafItem>
```

From:
https://wiki.geowebframework.com/ - **GeowebFramework**

Permanent link:
https://wiki.geowebframework.com/doku.php?id=gwusermanual:interface:schede

Last update: **2026/01/30 10:41**