
ng-gentelella Documentation

Release

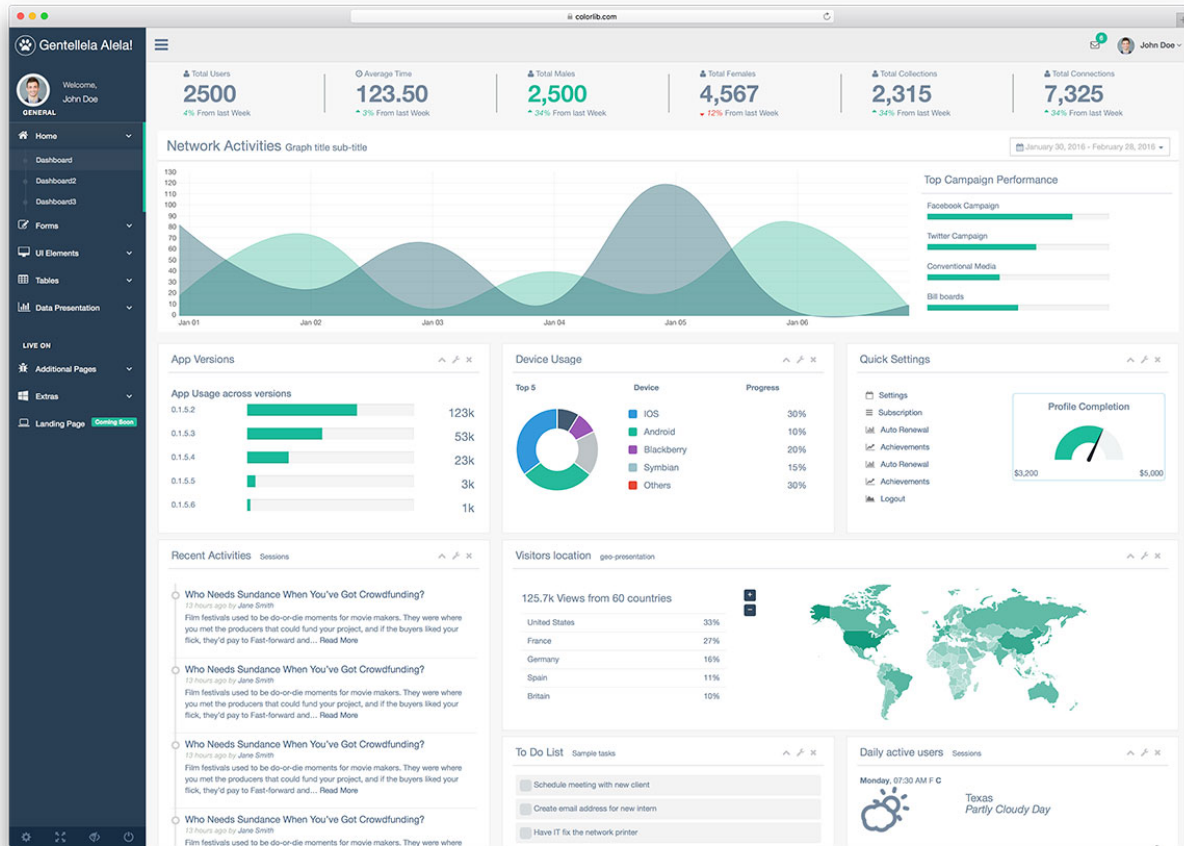
George Karakostas

Sep 30, 2017

Contents

1	Installation	3
2	ga-panel	5
3	ga-panel-table-form	9
4	ga-panel-actions	13
5	ga-resource	15
6	ga-paginate	19
7	ga-progress	21
8	ga-dashboard-counter	23
9	ga-dashboard-graph-flot	25
10	ga-dashboard-graph-bars	27
11	ga-dashboard-graph-chart	29
12	form-field-text	31
13	form-field-select	33
14	form-field-checkbox	35
15	form-field-image	37
16	Animations	39
17	Features and limitations	41
18	Alternatives	43

Easily create an administration interface using Angular components that are based on the markup by [Gentelella](#) bootstrap template.



CHAPTER 1

Installation

Use:

```
npm install -S ng-gentelella
```

Static files

Built files are conveniently provided under the `build/` directory. These include all gentelella and ng-gentelella css and js files.

Otherwise, include the `node_modules/ng-gentelella/gentelella` js files in your html or build system (eg gulp, [example gulpfile](#)). It is recommended that you include the templates path `node_modules/ng-gentelella/gentelella/**/*.html` using [some html2js module](#).

Alternatively you can expose the template files as `/ng-gentelella` with `app.use('/ng-gentelella', express.static(path.join(__dirname, 'node_modules', 'ng-gentelella')));`

Develop

In your application, render a gentelella default index page as you would.

Replace the main page content markup with an angular `ng-view` as in [this example](#): `<div ng-view class="view-frame"></div>`

Then develop proper Angular dashboard, list and detail components as you would [normally do](#), and use the above components in their templates to automate development.

CHAPTER 2

ga-panel

This is a main component that generates a simple gentelella panel.

SKUs			
<input type="text" value="Search"/>			<div>+ ^ x</div>
#	Name	Product	Status
1	F102	Funky	✓
2	F101	Funky	✓
3	F103	Funky	✓
4	F104	Funky	✓
5	F105	Funky	✓
6	F106	Funky	✓
7	F107	Funky	✓
8	F108	Funky	✓
9	F109	Funky	✓
10	F110	Funky	✓
More...			

Binding reference

- `panel-title`: The panel title (string)
- `panel-subtitle`: The panel subtitle displayed in smaller font next to the title (string)
- `panel-query`: Whether to show a small input text box usually for filtering (boolean)
- `panel-query-string`: A controller variable to hold the `panel-query` input (variable)
- `panel-add-record-url`: A url to direct for adding a record. If provided a + icon will be available (string)
- `panel-query-model-options`: Additional `ng-model-options` to pass to the query field (object)
- `on-query-change`: A callback function to call if query value changes (function)

Transclude

The component will present any content transcluded.

It also allows the optional transclude element `panel-toolbar` for presenting additional buttons (v0.2.5, see examples below).

Controller

The component will initiate the necessary jquery required by gentelella as well.

Code sample

Using simple filter

Template:

```
<ga-panel panel-title="Products"
  panel-query="true"
  panel-query-string="$ctrl.queryValue"
  panel-add-record-url="#!/products/add">
  <panel-toolbar>
    <a href="" title="Some button"><i class="fa fa-modx"></i></a>
    <a href="" title="And another"><i class="fa fa-random"></i></a>
  </panel-toolbar>
  <table class="table table-hover dataTable">
    <thead>
      <tr>
        <th>#</th>
        <th>Name</th>
      </tr>
    </thead>
    <tbody>
      <tr ng-repeat="product in $ctrl.products | filter:$ctrl.queryValue">
        <th scope="row"><a href="#!/products/{{ product._id }}">{{ $index + 1 }}</a></
→th>
        <td><a href="#!/products/{{ product._id }}">{{ product.name }}</a></td>
```

```

    </tr>
  </tbody>
</table>
</ga-panel>

```

Reference

Using callback function

Template:

```

<ga-panel panel-title="Products"
  panel-query="true"
  panel-query-string="$ctrl.queryValue"
  panel-query-model-options="{debounce: 1000}"
  on-query-change="$ctrl.filter(queryValue)"
  panel-add-record-url="#!/products/add">
  <table class="table table-hover dataTable">
    <thead>
      <tr>
        <th>#</th>
        <th>Name</th>
      </tr>
    </thead>
    <tbody>
      <tr ng-repeat="product in $ctrl.products">
        <th scope="row"><a href="#!/products/{{ product._id }}">{{ $index + 1 }}</a></
→th>
        <td><a href="#!/products/{{ product._id }}">{{ product.name }}</a></td>
      </tr>
    </tbody>
  </table>
</ga-panel>

```


CHAPTER 3

ga-panel-table-form

`ga-panel-table-form` together with `ga-panel-table-form-body` offer a more advanced *ga-panel* that can be used to display a table of records. Upon clicking on a record, an accordion with a form for that record opens.

Images

Title	Group	Image
Funky	collection	collection1.jpg
Funky	slideshow	1.jpg


Title

Funky

Group

slideshow

Image



Επιλογή αρχείου

Δεν επιλέχθηκε κανένα αρχείο.

Delete

Funky	slideshow	2.jpg
--	slideshow	3.jpg
--	slideshow	4.jpg
--	slideshow	5.jpg
--	slideshow	6.jpg

Binding reference

ga-panel-table-form bindings

- `panel-title`: The panel title (string)
- `panel-subtitle`: The panel subtitle displayed in smaller font next to the title (string)
- `panel-add-record`: Whether to allow add record. Usually evaluated by an expression, eg. `new vs edit op` (boolean)
- `panel-values`: A controller variable holding an array of the records (variable)

ga-panel-table-form-body bindings

- `body-id`: A unique id for each table row that will be used by bootstrap accordion (string)
- `body-value`: The variable from the `ng-repeat` that will be used to render the table (variable)

Transclude

ga-panel-table-form transclude

The component allows for two elements for transclude:

- `<panel-table-form-head>`: Used to display the table headers. Refrain from using `<table>` elements or CSS styles, and use the bootstrap grid instead because the table rendering might break in some browsers.
- `<panel-table-form-body>`: This is where the `ga-panel-table-form-body` component will be used with `ng-repeat` to present a table row with data and a form row with data detail.

ga-panel-table-form-body transclude

The component allows to transclude. Add in here the table row and the form row.

Controller

The `ga-panel-table-form` controller handles the add/delete operations of the table.

Code sample

Template:

```
<ga-panel-table-form panel-title="Specifications"
                    panel-add-record="$ctrl.skuId"
                    panel-values="$ctrl.sku.specs">
  <panel-table-form-head>
    <div class="col-md-6">Spec</div>
    <div class="col-md-6">Value</div>
```

```
</panel-table-form-head>

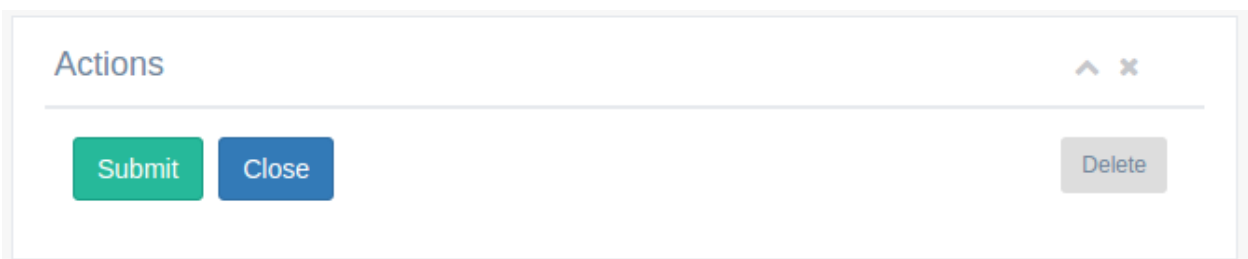
<panel-table-form-body>
  <ga-panel-table-form-body ng-repeat="spec in $ctrl.sku.specs"
    body-id="spec-{{ $index }}"
    body-value="spec">

    <body-row>
      <div class="col-md-6">{{ (spec.spec || '--' )}}</div>
      <div class="col-md-6">{{ ((spec.value || '--' )}}</div>
    </body-row>

    <body-form>
      <form-field-text>...</form-field-text>
      ...
    </body-form>
  </ga-panel-table-form-body>
</panel-table-form-body>
</ga-panel-table-form>
```

Reference

This offers a small panel with basic form buttons: Save, Close/cancel, delete.



Binding reference

- `action-form`: The parent form controller variable to use in order to check if form is clean (variable)
- `action-close-url`: A url to redirect to if user hits close (string)
- `action-allow-delete`: True to allow delete (boolean)
- `on-delete`: A callback function to call if user hits delete (function)

Requires

Requires a parent *ga-panel*.

Transclude

The controller allows transclude in order to present additional markup such as more buttons next to Delete.

Code sample

```
<ga-panel panel-title="Actions">
  <ga-panel-actions action-form="$ctrl.<%= objectName %>Edit"
    action-close-url="#!/<%= objectUrl %>s"
    action-allow-delete="$ctrl.<%= objectName %>Id"
    on-delete="$ctrl.delete<%= objectTitle %>()"></ga-panel-actions>
</ga-panel>
```

Reference

`ga-resource` is an [Angular service factory](#) wrapper. Automates the CRUD operations using gentelella's `PNotify` and default REST responses.

Methods

`resource.getAndNotify(options)`

Retrieve an entity with `$resource.get` and notify on error.

```
* @param options.getId: the entity id to retrieve, in object
* @param options.url: the entity url
*
* @param options.error404.title: not found error title
* @param options.error404.body: not found error body
*
* @param options.callbacks.err: additional error callback
* @param options.callbacks.next: additional success callback
*
* @returns {*}: the entity from $resource.get
```

`resource.submitAndNotify(options)`

Save an entity with `$resource.save` and notify.

```
* @param options.id: the entity id
* @param options.entity: the entity to save
* @param options.form: the ng-form to set pristine
* @param options.url: the entity url
*
* @param options.success.title: submit success title
```

```
* @param options.success.body: submit success body
* @param options.error.title: submit fail title
* @param options.error.conflict409: duplicate id fail body
*
* @param options.callbacks.err: additional error callback
* @param options.callbacks.next: additional success callback
*
* @returns {*}: the saved entity
```

resource.deleteAndNotify(options)

Delete an entity and notify.

```
* @param options.getId: the entity id to delete, in object
* @param options.url: the entity url
*
* @param options.success.title: success title
* @param options.success.body: success body
* @param options.error.title: fail title
*
* @param options.callbacks.err: additional error callback
* @param options.callbacks.next: additional success callback
```

Code sample

Example in factory

```
angular
  .module('core.products')
  .factory('Product', ['gaResource',
    function ($resource) {
      return $resource('api/products');
    }
  ]);
```

Example of factory use inside a controller

```
self.product = Product.getAndNotify({
  getId: {productId: self.productId},
  url: '/products',
  error404: {
    title: 'Product not found',
    body: 'The product cannot be found.'
  }
});

self.submitProduct = function() {
  self.product = Product.submitAndNotify({
    id: self.productId,
    entity: self.product,
    form: self.productEdit,
    url: '/products/',
    success: {
      title: 'Product saved',
```

```
        body: 'Product saved successfully.'
      },
      error: {
        title: 'Product not saved',
        conflict409: 'Product already exists'
      },
      callbacks: {next: self.getProduct}
    });
  };

self.deleteProduct = function() {
  Product.deleteAndNotify({
    getId: {productId: self.productId},
    url: '/products',
    success: {
      title: 'Product deleted',
      body: 'Product deleted successfully.'
    },
    error: {title: 'Product not deleted'}
  });
};
```


Provide a list paginator.

1 to 10 of 39 | 10 ▼ | more

First	Previous	1	2	3	4	Next	Last
-------	----------	---	---	---	---	------	------

Binding reference

- `paginate-id`: A unique HTML id for the page size combo box (string)
- `paginate-page`: A controller variable in which the current page number will be returned (variable)
- `paginate-size`: A controller variable in which the current page size will be returned (variable)
- `paginate-initial-size`: The initial page size (integer)
- `paginate-sizes`: An array of page sizes, default `[10, 25, 50, 100]` (array)
- `paginate-count`: The total number of records (integer)
- `paginate-ellipsis`: The maximum number of page numbers to show if too many, default 5. For more than that, ellipsis `...` are presented (integer)
- `on-paginate`: A callback function to call when a new page is selected. Used to fetch new data (function)

Transclude

The controller presents additional text right of the page size combo box such as a link.

Controller

The controller handles the operation of the paginator.

Code sample

```
<ga-paginate paginate-id="paginate-subscriptions"
  paginate-page="$ctrl.page"
  paginate-size="$ctrl.pageSize"
  paginate-initial-size="$ctrl.viewLimit"
  paginate-count="$ctrl.count"
  on-paginate="$ctrl.getList(paginator)">
  <a href="#!{{ $ctrl.viewLink }}">| more</a>
</ga-paginate>
```


Render a progress bar.

Subscriptions



Binding reference

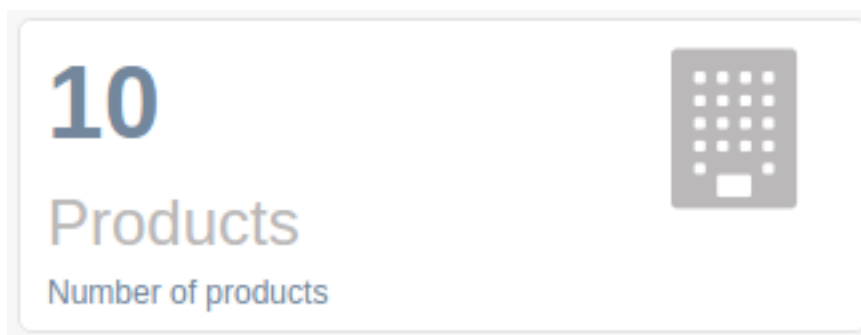
- `progress-size`: The progress bar size `sm` or `md`, default `sm` (string)
- `progress-value`: The progress bar percent value (number)

Notice: `progress-value` binding since v0.2.4 has changed to `@` requiring to be passed with `{{ }}`. This allows calculations in template.

Code sample

```
<ga-progress progress-value="{{ $ctrl.progress }}"></ga-progress>
<ga-progress progress-value="{{ (item.count / $ctrl.total * 100) | number: 0 }}"></ga-
↪progress>
```


Provide a large counter panel for dashboard as in [Gentelella index2](#).



Binding reference

- `counter-icon`: A [font awesome](#) icon name, eg building (string)
- `counter-var`: The number to present (integer)
- `counter-title`: The title to present (string)

Transclude

The component allows transclude to present additional text.

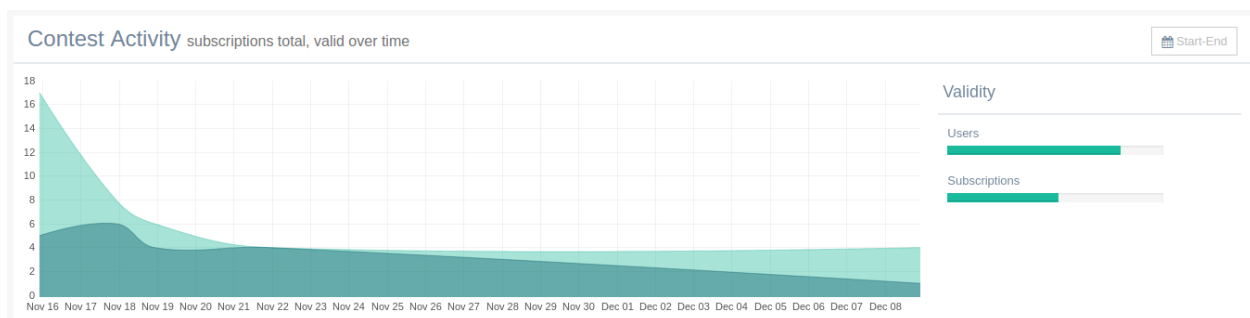
Code sample

```
<ga-dashboard-counter counter-icon="users"
                      counter-var="$ctrl.dashboard.users"
                      counter-title="Users">
  Number of total users
</ga-dashboard-counter>
```

Reference

ga-dashboard-graph-flot

Render a line graph in panel as in [Gentelella index](#).



Binding reference

- `graph-title`: The graph panel title (string)
- `graph-sub-title`: The subtitle presented next to title in smaller font size (string)
- `graph-range`: The date range to present (string)
- `graph-id`: A unique HTML id for jquery reference, default `main-graph` (string)
- `graph-legend-title`: The title of the legend column (string)
- `graph-colours`: An array of strings with colours for the series, defaults to gentelella colours (array)
- `graph-data`: The main graph data (array)

Regarding the graph data. Gentelella uses the [Flot](#) graph library that requires that data is sorted by date. Otherwise the data looks bizarre. The graph data should follow the format:

```
[
  {
    _id: {year: 2016, month: 12, day: 19},
```

```
    count: 10
  },
  ...
]
```

Transclude

The component allows the transclude of markup for the legend column body.

Controller

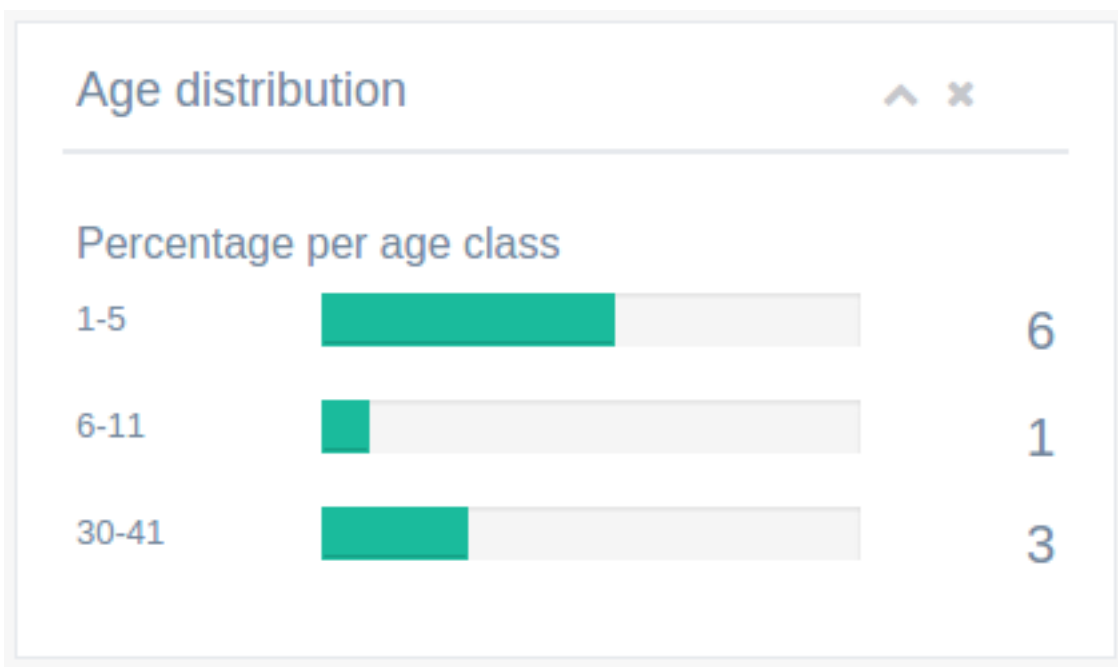
The controller:

- Transforms the data from the above more common JSON format to Flot format.
- Initializes the Flot graph appropriately.

Code sample

```
<ga-dashboard-graph-flot graph-title="Contest Activity"
    graph-sub-title="subscriptions total, valid over time"
    graph-legend-title="Validity"
    graph-range="Start-End"
    graph-data="$ctrl.dashboard.data">
  <div class="col-md-12 col-sm-12 col-xs-6">
    <p>Users</p>
    <ga-progress progress-value="$ctrl.userValidity"></ga-progress>
  </div>
  <div class="col-md-12 col-sm-12 col-xs-6">
    <p>Subscriptions</p>
    <ga-progress progress-value="$ctrl.subscriptionValidity"></ga-progress>
  </div>
</ga-dashboard-graph-flot>
```

Render a bars graph in panel as in [Gentelella index](#). The graph consists of *ga-progress* bars.



Binding reference

- `graph-title`: The graph panel title (string)
- `graph-sub-title`: The subtitle presented next to title in smaller font size (string)
- `graph-heading`: The heading in the content (string)

- `graph-data`: The main graph data (array)

The graph data should follow the format:

```
[{lower: 1, upper: 5, percentage: 15, count: 6}, ...]
```

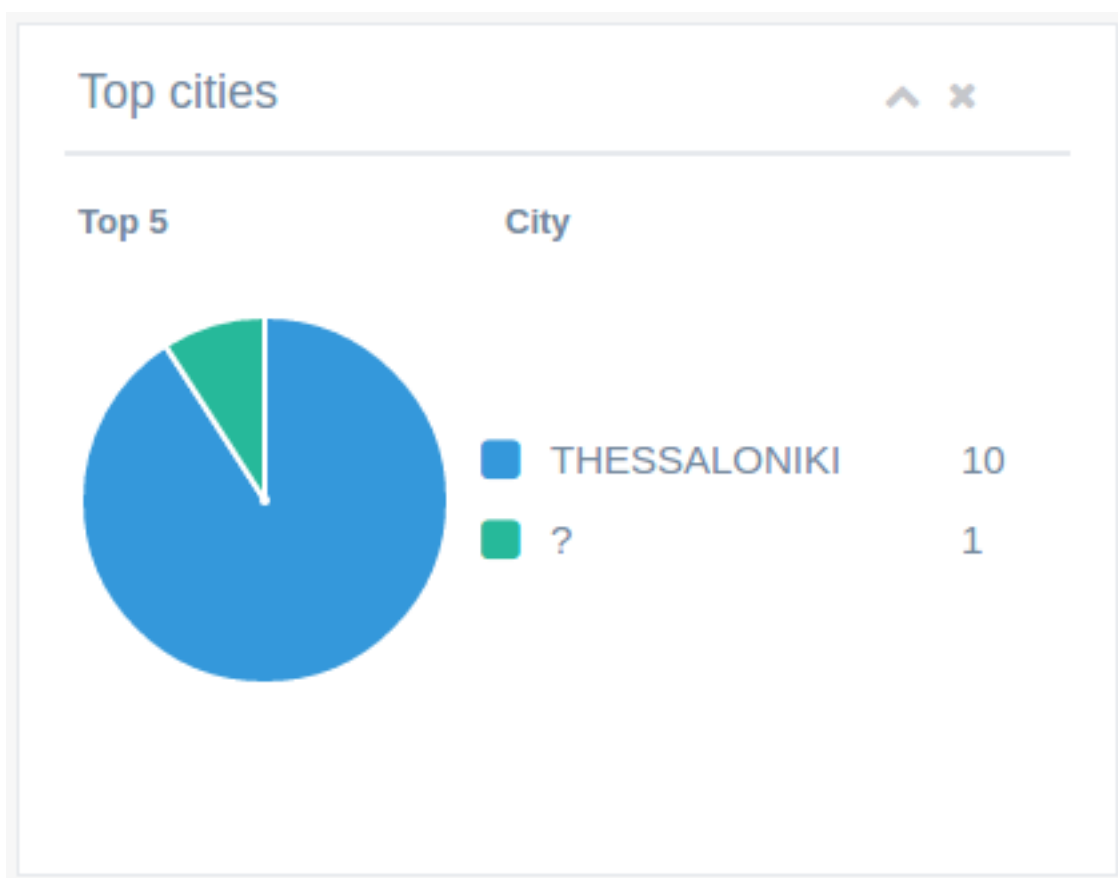
Transclude

The component allows the transclude of further content below.

Code sample

```
<ga-dashboard-graph-bars graph-title="Age distribution"  
  graph-heading="Percentage per age class"  
  graph-data="$ctrl.dashboard.ages"></ga-dashboard-graph-bars>
```


Render a `chart.js` graph in panel as in [Gentelella index](#).



Binding reference

- `graph-title`: The graph panel title (string)
- `graph-sub-title`: The subtitle presented next to title in smaller font size (string)
- `graph-heading`: The heading or name of the series (string)
- `graph-id`: A unique HTML id for jquery reference, default `main-graph` (string)
- `graph-type`: The Chart.js type: line, bar, radar, pie, doughnut, bubble (string)
- `graph-max-values`: If the values provided are more than this number, the remaining will be added to the last. For example, if the chart shows top 5 cities, and 6 are provided, the 6th value will be added to the 5th. Default 5 (integer)
- `graph-max-ellipsis`: If the above parameter is used, the text to replace the last label with. Default 'All others' (string)
- `graph-colours`: An array of strings with colours for the series, defaults to gentelella colours (array)
- `graph-data`: The main graph data (array)

The graph data should follow the format:

```
[{label: '', value: 0}, ...]
```

Transclude

The component allows the transclude of further content below.

Controller

The controller:

- Groups the data to the specified maximum (if any).
- Transforms the data from the above more common JSON format to Chart.js format.
- Initializes the Chart.js graph appropriately.

Code sample

```
<ga-dashboard-graph-chart graph-title="Top cities"
  graph-heading="City"
  graph-id="canvas2"
  graph-type="pie"
  graph-data="$ctrl.dashboard.cities"></ga-dashboard-graph-
↵chart>
```

Render a standard gentelella form textbox.

Tolerance

Binding reference

- `field-id`: A unique HTML id to associate label and input (string)
- `field-type`: The [HTML input type](#), default `text` (string)
- `field-label`: The label text (string)
- `field-placeholder`: The input placeholder, default empty (string)
- `field-width`: The width of the field in bootstrap columns (1-12), default 6 (integer)
- `field-label-width`: The width of the label in bootstrap columns (1-12), default 3 (integer)
- `field-required`: Whether the field is required, default `false` (boolean)
- `field-form`: The controller form variable to update if validation is required (variable)
- `field-name`: The field's name in the [Angular form](#). Requires `field-form` (string)
- `field-pattern`: A regular expression **without surrounding slashes** to test the input validity against. Combine with `field-form` and `field-name` to allow Angular to validate (string)
- `field-alert`: The text to display if the field is invalid. Requires `field-form` and `field-name` (string)
- `field-model-options`: Additional [ng-model-options](#) to pass to the field (object)
- `field-value`: A controller variable to return the `ng-model` input value (variable)
- `on-change`: A callback function to call if value changes (function)

Controller

The controller initializes the default values and regex pattern.

Code sample

```
<form-field-text field-id="product-name"
  field-label="Name"
  field-placeholder="Product full name"
  field-required="true"
  field-value="$ctrl.product.name"></form-field-text>

<form-field-text field-id="product-alias"
  field-label="Alias"
  field-placeholder="Unique machine name"
  field-required="true"
  field-width="5"
  field-name="productAlias"
  field-form="$ctrl.productEdit"
  field-pattern="^[a-z0-9-]+$"
  field-alert="Example: 'my-product'"
  field-value="$ctrl.product.alias"></form-field-text>
```

Reference

Render a gentelella form select box. Depending on the options provided this can be extended to multiple selection.

A screenshot of a Gentelella form select box. It consists of a label "Family" in blue text, followed by a white input box with a light gray border containing the text "Evolution". To the right of the input box is a small downward-pointing triangle icon, and further right is a small blue icon with a white arrow pointing up and to the right.

Binding reference

- `field-id`: A unique HTML id to associate label and input (string)
- `field-label`: The label text (string)
- `field-placeholder`: The input placeholder, default empty (string)
- `field-width`: The width of the field in bootstrap columns (1-12), default 6 (integer)
- `field-label-width`: The width of the label in bootstrap columns (1-12), default 3 (integer)
- `field-required`: Whether the field is required, default false (boolean)
- `field-multiple`: Allow multiple selection, default false (boolean)
- `field-link`: Provide a URL to present a link next to the combo box, default empty (string)
- `field-link-text`: A description of the above link to present on hover (string)
- `field-value`: A controller variable to return the `ng-model` input value (variable)
- `on-change`: A callback function to call if value changes (function)

Controller

The controller initializes the gentelella select2 script (currently disabled due to [issue #11](#)).

Code sample

```
<form-field-select field-id="product-family"
                  field-label="Family"
                  field-placeholder="Product Family"
                  field-link="#!/product-families"
                  field-link-text="Open family"
                  field-value="$ctrl.product.family">
  <option></option>
  <option ng-repeat="item in $ctrl.productFamilies | orderBy:''"
          value="{{ item._id }}">{{ item.name }}</option>
</form-field-select>
```

Reference

Render a checkbox.

Active ☒ Product status

Binding reference

- `field-id`: A unique HTML id to associate label and input (string)
- `field-label`: The label text (string)
- `field-placeholder`: The input placeholder, default empty (string)
- `field-width`: The width of the field in bootstrap columns (1-12), default 6 (integer)
- `field-label-width`: The width of the label in bootstrap columns (1-12), default 3 (integer)
- `field-value`: A controller variable to return the `ng-model` input value (variable)
- `on-change`: A callback function to call if value changes (function)

Transclude

The component allows transclude to provide additional markup.

Code sample

```
<form-field-checkbox field-id="product-status"
  field-label="Active"
```

```
field-placeholder="Product status"  
field-value="$ctrl.product.status"></form-field-checkbox>
```

Reference

CHAPTER 15

form-field-image

Provide an image upload field. This relies on `ng-file-upload`.

Image



Επιλογή αρχείου Δεν επιλέχθηκε κανένα αρχείο.

Binding reference

- `field-id`: A unique HTML id to associate label and input (string)
- `field-label`: The label text (string)
- `field-width`: The width of the field in bootstrap columns (1-12), default 9 (integer)
- `field-label-width`: The width of the label in bootstrap columns (1-12), default 3 (integer)

- `field-required`: Whether the field is required, default false (boolean)
- `field-media-url`: The url where the image filename value resides (string)
- `field-title`: The `` (string)
- `field-data`: A relevant entity data to send with the upload, optional (object)
- `field-value`: A controller variable to return the `ng-model` input value (variable)

Transclude

The controller allows transclude to replace the thumbnail markup.

Controller

The controller uploads the file to `api/uploads` using `ng-file-upload`.

Code sample

```
<form-field-image field-id="image-{{ $index }}-image"
  field-media-url="/media/product"
  field-title="{{ image.title || 'Product image' }}"
  field-data="{_id: $ctrl.productId, entity: 'product'}"
  field-value="image.image"></form-field-image>
```

Reference

The following animations are currently supported.

Table updates

A 500msec fade occurs when table rows change. This is useful when a list data is updated or filtered.

View change

A 700 msec fade occurs when the angular view changes.

Helper classes

Additionally, the following classes can be applied in any element.

Spinning

Add the `spinning` class to make an element spin. Useful for bootstrap or fa icons:

```
<span class="glyphicon glyphicon-refresh spinning">
```


CHAPTER 17

Features and limitations

The project offers a small number of components that aim to directly reduce the development time of an admin interface. It currently not yet offers a wide number of components to fully automate the development of a gentelella interface.

CHAPTER 18

Alternatives

- [angular2-webpack-starter-gentelella](#)
- [commercial angular templates](#)
- [inspinia commercial template](#)
- [ng-admin](#): one of the most well developed angular admin with a configuration system that is too advanced for my taste.