Libreria itaCharts

La libreria itaCharts è una libreria wrapper per ChartJS che permette di creare diversi tipi di grafico.

Concettualmente la libreria è divisa in 3 classi principali:

- **itaCharts**: è la classe principale e rappresenta il grafico nel suo insieme.
- **itaChartsDataSet**: la classe contiene un dataset. La classe itaCharts conterrà uno o più datasets.
- **itaChartsOptions**: la classe contiene le opzioni generali del grafico.

Oltre a queste tre classi è presente una quarta classe di supporto su cui non dovrebbe mai essere necessario intervenire:

- **itaChartsColor**: la classe funziona da traduttore per passare da una descrizione testuale di un colore ad una rappresentazione come oggetto.

Per usare la libreria è sufficiente fare l'include della classe base, quindi:

```
require_once ITA_LIB_PATH . '/itaCharts/itaCharts.class.php';
```

---

itaCharts

La classe rappresenta un grafico nel suo insieme ed è formata dai seguenti metodi:

__construct()

Il costruttore della classe prende in input un parametro:

- **$container**: E' l'id dell'oggetto html che andrà a contenere il grafico, si consiglia di usare come contenitore un div vuoto, ma in realtà qualsiasi elemento in grado di contenere un canvas dovrebbe essere in grado di funzionare correttamente.

setLabels()

Il metodo permette di impostare le etichette che poi andranno sull'asse delle ascisse (per i grafici cartesiani). Questo metodo prende i seguenti parametri:

- **$labels**: il parametro deve essere un array contenente delle stringhe.

NB: *Il grafico dovrà avere tanti label quanti sono gli elementi presenti nei singoli dataset (tutti i dataset devono avere lo stesso numero di elementui).*

addDataSet()

Il metodo permette di aggiungere al grafico un dataset. Questo metodo prende i seguenti parametri:

- **$dataSet**: il parametro deve essere un oggetto della classe itaChartsDataSet, rappresenta un dataset del grafico, ogni grafico può avere un numero illimitato di datasets.

resetData()

Il metodo permette di resettare tutti i dati del grafico, annullando label e dataset.

setChartType()

Il metodo permette di definire quale tipo di grafico si va a creare. Questo metodo prende i seguenti parametri:

- **$chartType**: rappresenta la tipologia di grafico, accetta in input una delle costanti definite come **itaCharts::ITA_CHARTS_TYPE_***

setOptions()

Permette di impostare le opzioni del grafico. Questo metodo prende i seguenti parametri:

- **$options**: il parametro deve essere un oggetto della classe itaChartsOptions, rappresenta le opzioni del grafico.

setCustomParameters()

Permette di impostare parametri addizionali liberi. Questo metodo prende il seguente parametro:

- **$parameters**: Il parametro deve essere un oggetto di tipo stdClass avente la struttura dei parametri aggiuntivi che si vuole impostare.

setAdditionalCode()

Permette di aggiungere del codice al termine della creazione del chart, in modo da poter applicare plugin o fare altre operazioni avanzate. Questo metodo prende il seguente parametro:

- **$code**: Stringa contenente il codice che si vuole aggiungere. La variabile contenente l'oggetto ChartJS si chiama '<*container*>_Chart', laddove <*container*> è l'id del div che contiene il grafico.

—-

itaChartsDataSet

La classe rappresenta un dataset del grafico ed include oltre ai dati in quanto tale anche le opzioni relative al dataset. La classe è formata dai seguenti metodi:

__construct()

Al momento dell'inizializzazione dell'oggetto verranno valorizzati dei parametri di base per le opzioni del dataset.

setLabel()

Il metodo permette di impostare il nome del dataset. Questo metodo prende i seguenti parametri:

- **$label**: Stringa contenente il nome del dataset (verrà visualizzato nella legenda.

setBackgroundColor()

Il metodo permette di impostare il colore di riempimento dei dati del dataset. Questo metodo prende i seguenti parametri:

- **$color**: Stringa o Array di stringhe. Rappresenta il colore del dataset, se viene passato un array questo dovrà avere tanti elementi quanti sono gli elementi del dataset, così facendo ad ogni elemento del dataset verrà assegnato un colore differente. I colori sono rappresentabili in diverse forme: **#RRGGBB**, **#RGB**, **rgba(r,g,b,a)**, **rgb(r,g,b)**, **testuale** (es: red, green, etc..)
- **$alpha**: decimale o Array di decimali. Rappresenta il grado di trasparenza del colore di riempimento. Il valore va da 0 (completamente trasparente) a 1 (completamente opaco). Se si passa un array questo dovrà avere tanti elementi quanti sono gli elementi del dataset, così facendo ad ogni elemento del dataset verrà assegnato un valore di trasparenza differente.

setHoverBackgroundColor()

Metodo identico a *setBackgroundColor()*, setta il colore nel caso di mouse hover.

setBorderColor()

Il metodo permette di impostare il colore delle linee esterne del dataset. Ricalca esattamente la stessa logica di funzionamento del metodo setBackgroundColor.

N.B: *Se solo uno dei due metodi fra setBorderColor e setBackgroundColor viene richiamato il valore dell'altro viene assegnato automaticamente. Nel dettaglio i colori vengono ripresi dall'elemento che è stato settato mentre la trasparenza viene fissata a 0.8 per il bordo e 0.5 per il colore di riempimento.*

setHoverBorderColor()

Metodo identico a *setBorderColor()*, setta il colore del bordo in caso di mouse hover

setBorderWidth()

Il metodo permette di impostare la larghezza del bordo del dataset. Questo metodo prende i seguenti parametri:

- **$width**: valore intero che rappresenta la larghezza in pixel del bordo.

setHoverBorderWidth()

Metodo identico a *setBorderWidth()*, setta la larghezza del bordo in caso di mouse hover

setFill()

Il metodo permette di definire se l'area sottesa ad una curva nei grafici a linee viene colorata o meno. Questo metodo prende i seguenti parametri:

- **$fill**: valore booleano che indica se l'area viene colorata o meno.

setData()

Il metodo permette di caricare i dati veri e propri. Questo metodo prende i seguenti parametri:

- **$data**: array che contiene i dati del dataset.

setInterpolation()

Il metodo permette di settare che tipo di intepolazione usare nei grafici a linea. Questo metodo prende i seguenti parametri:

- **$interpolation**: deve assumere come valore una delle costanti definite come **itaChartsDataSet::ITA_CHARTS_INTERPOLATION_***

__toString()

Il metodo non dovrebbe venir richiamato direttamente dall'utente, restituisce una rappresentazione json del dataset e delle sue opzioni che verrà poi usata da ChartJS.

---

itaChartsOptions

La classe rappresenta le opzioni del grafico. Ogni grafico può avere un solo oggetto itaChartsOptions abbinato, questo è facoltativo e nel caso non venisse passato verranno prese le opzioni di default. L'oggetto è formato dai seguenti metodi:

__construct()

Per l'oggetto al momento della creazione vengono valorizzate tutte le opzioni disponibili dal wrapper impostate con i valori di default.

setResponsive()

Il metodo permette di impostare se il grafico sarà responsive e andrà quindi a modificare le proprie dimensioni in accordo con la dimensione dell'elemento padre. Questo metodo prende i seguenti parametri:

- **$responsive**: valore booleano, se true l'oggetto viene impostato come responsive, altrimenti no. Di default è impostato a true.

setPadding()

Il metodo permette di impostare i valori di padding del grafico, questi rappresentano essenzialmente la cornice interna che viene lasciata fra il grafico e gli altri elementi della pagina. Questo metodo prende i seguenti parametri:

- **$left**: valore intero, è lo spazio in pixel che viene lasciato verso sinistra
- **$right**: valore intero, è lo spazio in pixel che viene lasciato verso destra
- **$top**: valore intero, è lo spazio in pixel che viene lasciato verso l'alto
- **$bottom**: valore intero, è lo spazio in pixel che viene lasciato verso il basso

Nota: ChartJS lavora con i canvas, il padding è effettivamente uno spazio trasparente all'interno dell'immagine generata dal canvas e non il classico padding html.

setLegendDisplay()

Il metodo permette di definire se deve venir mostrata o meno la legenda del grafico. Questo metodo prende i seguenti parametri:

- **$display**: valore booleano, se true viene mostrata la legenda.

setLegendLabel()

Il metodo permette di definire tutta una serie di aspetti grafici della legenda. Questo metodo prende i

seguenti parametri:

- **$boxWidth**: valore intero, rappresenta la larghezza in pixel dei quadratini della legenda.
- **$fontSize**: valore intero, rappresenta la dimensione in punti del font
- **$fontStyle**: stringa, rappresenta lo stile del testo, prende in input un valore fra quelli di **itaChartsOptions::ITA_CHARTS_FONT_***
- **$fontColor**: stringa, rappresenta il colore del testo
- **$fontFamily**: stringa, rappresenta la famiglia di font da usare, come da css
- **$padding**: valore intero, rappresenta il padding in pixel fra la legenda e gli altri elementi del grafico

setTitleDisplay()

Il metodo permette di definire se deve venir mostrato o meno il titolo del grafico. Questo metodo prende i seguenti parametri:

- **$title**: stringa, è il titolo del grafico.

setTitle()

Il metodo permette di impostare il titolo del grafico. Questo metodo prende i seguenti parametri:

- **$display**: valore booleano, se true viene mostrato il titolo.

setTitlePosition()

Il metodo permette di definire dove verrà mostrato il titolo del grafico. Questo metodo prende i seguenti parametri:

- **$position**: stringa, indica dove viene posizionato il titolo, accetta un valore fra quelli di **itaChartsOptions::ITA_CHARTS_***

setTitleOptions()

Permette di impostare tutta una serie di opzioni relative al titolo del grafico. Questo metodo prende i seguenti parametri:

- **$fontSize**: valore intero, rappresenta la dimensione in punti del font
- **$fontStyle**: stringa, rappresenta lo stile del testo, prende in input un valore fra quelli di **itaChartsOptions::ITA_CHARTS_FONT_***
- **$fontColor**: stringa, rappresenta il colore del testo
- **$fontFamily**: stringa, rappresenta la famiglia di font da usare, come da css
- **$padding**: valore intero, rappresenta il padding in pixel fra la legenda e gli altri elementi del grafico

setXAxis()

Permette di impostare tutta una serie di parametri relativi all'asse X del grafico. Questo metodo prende i seguenti parametri:

- **$display**: valore booleano, indica se mostrare o meno, graficamente, l'asse
- **$type**: stringa, indica se l'asse ha una rappresentazione lineare o logaritmica (quest'ultima solo per gli assi con etichette numeriche). Accetta in input un valore fra quelli di **itaChartsOptions::ITA_CHARTS_AXIS_***
- **$position**: stringa, indica dove posizionare l'asse. Accetta in input un valore fra quelli di **itaChartsOptions::ITA_CHARTS_***
- **$stacked**: valore booleano, indica se, nel caso di dataset multipli, le barre devono venir sovrapposte o meno.

setYAxis()

Il metodo è speculare a setXAxis() ma relativamente all'asse X, si rimanda alla documentazione di quest'ultimo.

setCustomOptions()

Permette di aggiungere ulteriori opzioni in modo libero. Questo metodo prende il seguente parametro:

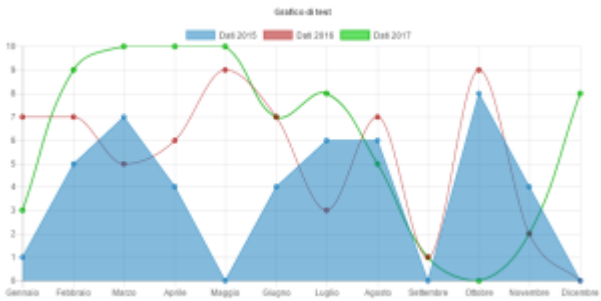
- **$options** oggetto di tipo stdClass, contiene la struttura dati delle opzioni aggiuntive che si vogliono inserire, ad esempio:

```
$options = new stdClass;
$options->rotation = $pi;
$options->circumference = 15;
```

Esempi d'uso

Per un esempio d'uso si può fare riferimento **/apps/CityBase/cwbZzzChartTest.php** Di seguito si riportano alcuni esempi estratti da tale file

Creazione di un grafico a linee



```
$months =
array('Gennaio','Febbraio','Marzo','Aprile','Maggio','Giugno','Luglio','Agos
to','Settembre','Ottobre','Novembre','Dicembre');

//CREAZIONE DI 3 DATASET RANDOM
$dataSet0 = new itaChartsDataSet();
$dataSet0->setData(array(rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,
10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0
,10)));
$dataSet0->setLabel("Dati 2015");
$dataSet0->setInterpolation(itaChartsDataSet::ITA_CHARTS_INTERPOLATION_NONE)
;

$dataSet1 = new itaChartsDataSet();
$dataSet1->setBorderColor('#AA0000', '0.8');
$dataSet1->setData(array(rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,
10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0
,10)));
$dataSet1->setFill(false);
$dataSet1->setLabel("Dati 2016");
$dataSet1->setInterpolation(itaChartsDataSet::ITA_CHARTS_INTERPOLATION_MONOT
ONE);

$dataSet2 = new itaChartsDataSet();
$dataSet2->setBorderColor('#00AA00', '0.8');
$dataSet2->setBackgroundColor('#00CC00', '0.6');
$dataSet2->setBorderWidth(2);
$dataSet2->setData(array(rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,
10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0
,10)));
$dataSet2->setFill(false);
$dataSet2->setLabel("Dati 2017");
$dataSet2->setInterpolation(itaChartsDataSet::ITA_CHARTS_INTERPOLATION_CUBIC
);

//CREAZIONE DI UN GRAFICO A LINEA A PARITRE DAI TRE DATASET CREATI E CON
OPZIONI CUSTOM
$options = new itaChartsOptions();
$options->setTitle("Grafico di test");

$lineChart = new itaCharts('cwbZzzChartTest_chart');
$lineChart->setLabels($months);
$lineChart->addDataSet($dataSet0);
$lineChart->addDataSet($dataSet1);
$lineChart->addDataSet($dataSet2);
$lineChart->setChartType(itaCharts::ITA_CHARTS_TYPE_LINE);
$lineChart->setOptions($options);
$lineChart->render();
```

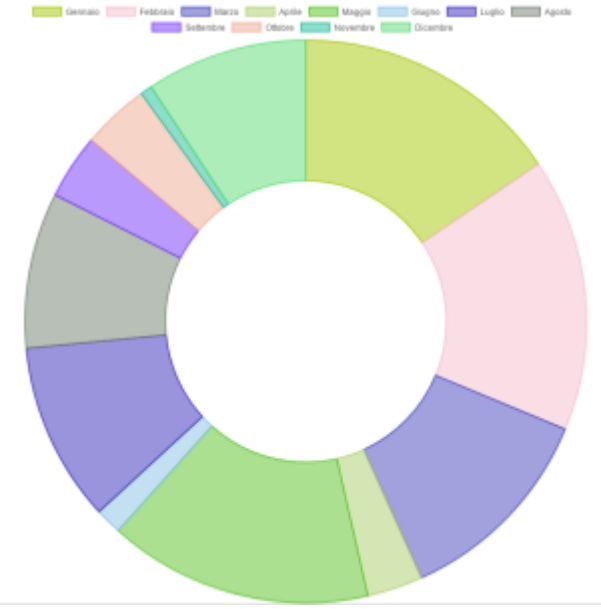
Creazione di un istogramma

Per questo esempio si andranno ad usare i dataset creati nell'esempio precedente.

```
//CREAZIONE DI UN ISTOGRAMMA A PARTIRE DAI TRE DATASET CREATI
$barChart = new itaCharts('cwbZzzChartTest_chart2');
$barChart->setLabels($months);
$barChart->addDataSet($dataSet0);
$barChart->addDataSet($dataSet1);
$barChart->addDataSet($dataSet2);
$barChart->setChartType(itaCharts::ITA_CHARTS_TYPE_BAR);
$barChart->render();
```

Creazione di un grafico a torta

Per questo esempio si creerà un dataset ex-novo. Notare che in questo dataset viene assegnato un colore differente per ogni valore, in caso contrario il grafico a torta sarebbe risultato tutto dello stesso colore.



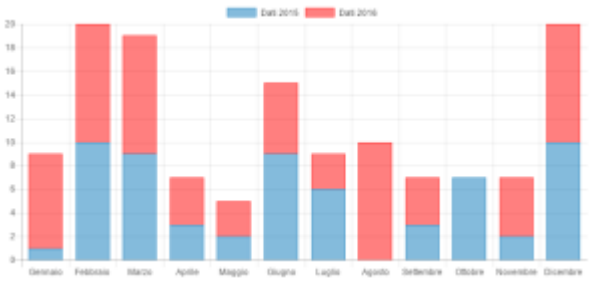
```
//CREAZIONE DI UN SINGOLO DATASET PER UN GRAFICO A TORTA
$dataSetPie = new itaChartsDataSet();
$colors = array();
for($i=0;$i<12;$i++){
    $colors[] = 'rgba('.rand(0,255).','.rand(0,255).','.rand(0,255).',0.5)';
}
$dataSetPie->setBackgroundColor($colors);
$dataSetPie->setData(array(rand(0,100),rand(0,100),rand(0,100),rand(0,100),r
and(0,100),rand(0,100),rand(0,100),rand(0,100),rand(0,100),rand(0,100),rand(
0,100),rand(0,100)));
```

```
$dataSetPie->setBorderWidth(2);
$dataSetPie->setLabel("Dati 2017");

//CREAZIONE DI UN GRAFICO A TORTA
$pieChart = new itaCharts('cwbZzzChartTest_chart3');
$pieChart->setLabels($months);
$pieChart->addDataSet($dataSetPie);
$pieChart->setChartType(itaCharts::ITA_CHARTS_TYPE_DOUGHNUT);
$pieChart->render();
```

Creazione di un istogramma a barre impilate

In questo esempio viene mostrato un istogramma con due dataset, in questo caso le barre però non vengono affiancate far loro ma vengono impilate una sull'altra. Questo effetto è ottenuto impostando stacked=true sia per l'asse x che per l'asse y sulle opzioni del grafico.



```
//CREAZIONE DELLE OPZIONI PER UN ISTOGRAMMA CON DATASET IMPILATI
$optionsStacked = new itaChartsOptions();
$optionsStacked->setXAxis(true, null, null, true); //L'ULTIMO PARAMETRO
IMPOSTATO A TRUE PERMETTE DI SOVRAPPORRE LE BARRE
$optionsStacked->setYAxis(true, null, null, true); //L'ULTIMO PARAMETRO
IMPOSTATO A TRUE PERMETTE DI IMPILARE LE BARRE
//CREAZIONE DEI DATASET PER UN ISTOGRAMMA CON DATASET IMPILATI
$dataSetStacked = new itaChartsDataSet();
$dataSetStacked->setData(array(rand(0,10),rand(0,10),rand(0,10),rand(0,10),r
and(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),
rand(0,10)));
$dataSetStacked->setLabel("Dati 2015");
$dataSetStacked2 = new itaChartsDataSet();
$dataSetStacked2->setBorderColor("red");
$dataSetStacked2->setData(array(rand(0,10),rand(0,10),rand(0,10),rand(0,10),
rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10),rand(0,10)
,rand(0,10)));
$dataSetStacked2->setLabel("Dati 2016");

//CREAZIONE DI UN ISTOGRAMMA CON DATASET IMPILATI
$stackedChart = new itaCharts('cwbZzzChartTest_chart4');
$stackedChart->setLabels($months);
$stackedChart->addDataSet($dataSetStacked);
$stackedChart->addDataSet($dataSetStacked2);
$stackedChart->setChartType(itaCharts::ITA_CHARTS_TYPE_BAR);
$stackedChart->setOptions($optionsStacked);
$stackedChart->render();
```

24 visualizzazioni.