Grafici per App BlackBerry 10

Aggiungere grafici nelle proprie app è sempre stato un grattacapo per molti sviluppatori. Non esistono, infatti, soluzioni che consentano una semplice integrazioni di grafici di vario tipo.  
Per risolvere questo problema è stato sviluppato un progetto guida per una semplice e rapida integrazione di grafici impostando i dati da rappresentare, le opzioni di visualizzazione ed il tipo di grafico desiderato.  
I tipi di grafici disponibili sono i grafici a barre, a linee, a radar, ad area polare, a torta e ciambella e a bolle.

Il sistema utilizzato si basa sull’integrazione di una pagina web nella pagina QML dell’app attraverso un componente WebView. Si costruiscono i datasets e le opzioni del grafico in una funzione Javascript della pagina QML che poi li invia alla pagina HTML che li elabora generando il grafico finale. La pagina HTML può restituire un messaggio di risposta alla WebView che può essere utilizzato per diversi scopi.

C:\Users\Jacopo\AppData\Local\Microsoft\Windows\INetCacheContent.Word\chart4BBlogic.png

La libreria javascript utilizzata per generare il grafico nella pagina HTML si chiama ChartJS. È una libreria di facile integrazione e di semplice utilizzo. ([www.chartjs.org](http://www.chartjs.org))

Nella pagina HTML includere la libreria javascript

<script src=**"chart.min.js"**></script>

all’interno dei tag *head* e aggiungere la semplice struttura HTML di seguito

<div id=**"container"** style=**"width: 95%;"**><canvas id=**"myChart"**></canvas></div>

con all’interno il *canvas* che verrà richiamato dal codice javascript.  
Aggiungere il seguente codice per la ricezione della stringa JSON e la generazione del grafico nella pagina HTML

navigator.cascades.onmessage **=** ***function*** onmessage**(**message**)** **{** /\*\*here the creation of the chart\*\*/  
**}**

Nella pratica si vedrà che la pagina HTML rimarrà sempre la stessa, escluse modifiche per particolari usi.

Nella pagina QML inserire un componente WebView con il percorso del file HMTL assegnato alla proprietà *url*. (è necessario aggiungere *local* al percorso. Es: *"local:///assets/html/crazyCharts.html"*)  
Creare una funzione javascript (nel codice chiamata *createChart*) il cui compito è quello di generare i datasets dei dati che si vogliono rappresentare, le etichette e le opzioni che si vogliono assegnare al grafico.

All’evento *onLoadingChanged* richiamare la funzione appena creata per generare il grafico al caricamento della pagina HTML.

Riassumendo:

1. la funzione *createChart* genera tutto il necessario per la creazione del grafico (datasets, etichette e opzioni), converte gli oggetti javascript in una stringa JSON e li invia alla pagina HTML con la funzione *postMessage* della WebView.
2. La pagina HTML cattura l’evento con la funzione assegnata a *navigator.cascades.onmessage* e crea il grafico con i parametri decodificati dalla stringa JSON ricevuta.

In alcuni casi è necessario ottenere l’immagine del grafico. In questi casi ci viene in aiuto il metodo toBase64Image**(),** messo a disposizione dalla libreria ChartJs,e codifica l’immagine in una stringa base 64.

Quindi, la funzione javascript nella pagina HTML, dopo la creazione del grafico lo converte in stringa e lo invia alla WebView come risposta utilizzando la funzione *navigator.cascades.postMessage(base64string);*La WebView riceve la risposta con l’evento *onMessageReceived* e la passa ad un metodo scritto in C++ per la conversione ed il salvataggio dell’immagine. In particolare la funzione accetta la stringa base64 ed il percorso di salvataggio come parametri di ingresso e restituisce true o false in funzione dell’esito dell’operazione.

In questo modo è possibile utilizzare l’immagine in formato PNG all’interno della propria app oppure condividere il grafico creato.