Telegram Road
Manuale di amministratore

Rel. 1.3.2

Introduzione

Telegram Road è una piattaforma per lo sviluppo di **chatbots**, cioè applicazioni interattive in cui l'utente colloquia con un server (un *bot*) tramite un' applicazione di messaggistica, generalmente su uno smartphone. L' applicazione di messaggistica in questo caso è Telegram Messenger (d'ora in avanti, **Telegram**), che è una app largamente diffusa su molte piattaforme.

La descrizione di Telegram esula dalle finalità di questo documento, e si può facilmente trovare sulla Rete ad esempio in https://telegram.org/

Scopo di questo documento

Questo documento descrive come creare e gestire una chatbot (o nel prosieguo, *app*), tramite la piattaforma Telegram Road.

E' possibile che l'utente abbia già creato una sua app. Ciò è descritto nel documento :

Telegram Road – Guida d'utente

E' assai probabile che questa nuova app sia stata creata tramite uno smartphone, operando su due bot specifici:

- **BotFather**, l'app standard Telegram per creare e gestire altri bot per Telegram
- **Telegram Road** , un bot che attiva, registra e collega il nuovo bot all'applicazione stessa. L'applicazione, che gira sul server Telegram Road, è ciò che d'ora in avanti andremo a descrivere.

Data l'applicazione, l'utente opera su di essa solo tramite Telegram. Ecco perché Telegram Road è un mezzo potente per sviluppare applicazioni mobili senza che l'utente finale debba installare nulla di nuovo sul proprio dispositivo. L'unica applicazione richiesta è Telegram, niente di più.

Ambiente di sviluppo

Lo strumento di sviluppo suggerito per il gestore (a cui ci stiamo rivolgendo) NON è una chatbot, ma un'applicazione tradizionale basata suu form, ed eseguita su un browser (*web-based*). La ragione di questo è che un'applicazione basata su chatbot risulterebbe troppo **lenta,** dal punto di vista dell'esperienza di utente, a causa del

fatto che una chatbot richiede di spezzare ogni richiesta di informazioni in più domande semplici. Man mano che le informazioni da gestire crescono, il processo globale per un'applicazione basata su chat diventerebbe troppo snervante.

Invece, un browser è uno strumento che si trova sia su un desktop, sia su smartphone, o qualsiasi dispositivo mobile, come ad esempio un tablet. La scelta di far uso di un browser su un desktop oppure su tablet o smartphonne sta al gestore, e dipende da diverse considerazioni:

- più grande è lo schermo, maggiore è il numero delle informazioni che vi possono stare sopra;
- un browser basato su desktop, come Chrome, Firefox, Edge, Internet Explorer (Windows) o Chrome, Safari (Mac) è più ricco di funzioni del suo corrispondente su mobile ;
- l'uso di un mouse (su desktop) permette ulteriori possibilità di quanto offerto dai dispositivi "touch screen"

Ciò detto, sarà sempre possibile usare ciascuno dei due ambienti, magari mescolati. Cioè, si potrà cominciare lo sviluppo dell'app su uno smartphone, portare il lavoro pesante su desktop, e magari dare gli ultimi ritocchi ancora su mobile. La stessa app sarà infatti sempre disponibile su ciascuna piattaforma, a piacere.

Ciclo di vita di una App

Prima di addentrarci nei passaggi di uno sviluppo, occorre definire un paio di termini importanti nel ciclo di vita di una App :

- **fase di sviluppo (*design time*)** è la fase in cui il gestore dell'app la progetta e la configura. Ricordiamo che il gestore, cioè la persona a cui questo documento si indirizza, è chi definisce i gli elemeti (nodi, archi) ed il loro comportamento, e che l'utente finale adopererà in *run time*
- **fase di esecuzione (*run time*)** è quando l'utente apre, usa e sfrutta l'app. Un buon gestore dovrebbe sempre tenere conto della facilità di uso dell'app per l'utente finale (*end user*).

Ci si riferirà a questi due termini molte volte nel prosieguo di questo documento.

Tipi di applicazione

In Telegram Road sono possibili diversi tipi di applicazione. Si possono differenziare per diversi livelli di funzioni e complessità.

App statiche

Le app statiche sono completamente definite in fase di progettazione: il loro comportamento è completamente determinato dalla struttura della app progettata dal manager, sia in termini di domande poste all'utente, sia per le risposte che sarà permesso di dare.

Il prossimo capitolo sarà dedicato esclusivamente allo sviluppo di app statiche. Le app statiche sono le uniche disponibili nella versione gratuita (free) df Telegram Road. Per lo sviluppo di una app dinamica, il manager dovrà sottoscrivere l'opzione Premium di questa piattaforma.

App dinamiche, database statico

A differenza di una app statica, che ha sempre lo stesso comportamento, dato che l'utente ha la libertà di muoversi in dati fissi, una app dinamica permette all'utente di introdurre, cercare e gestire dati di sua proprietà, in modo da costruire un suo ambiente dati (database) privato. Con una app dinamica, l'utente è come se avesse la sua versione privata della app, gestendo suoi dati personali.

Per esempio, un utente potrebbe voler costruire un database di contatti in cui memorizzare gli amici ed i loro dati . I dati salvati potrebbero essere nome, cognome, indirizzo (testo), ma anche data di nascita (data) o numero di telefono (numero). La struttura è decisa dal manager in fase di sviluppo , ma i veri e propri dati sono scelti dall'utente in fase di esecuzione. Anche la tipologia dei dati è predefinita: l'utente finale non sarà in grado di sceglierla.

Attualmente, sono disponibili due app di questo tipo, una Rubrica e gli Appunti, cioè brevi testi da salvare e recuperare velocemente (come un post-it).

Un capitolo che segue sarà dedicato ai nodi Q ed allo sviluppo di applicazioni dinamiche.

App dinamiche, database dinamico

Una App dinamica con un database dinamico permette all'utente finale di scegliere non solo i suoi dati, ma anche la loro natura e struttura. Per esempio si potrebbe desiderare di avere un archivio di libri, o dischi, o magari eventi, e così via.

E dato che tutte queste cose differiscono, si deve essere liberi di fare scelte in fase di esecuzione (**run time)** sulla struttura dei dati , cioè degli attributi che li caratterizzano.

L'operatività diventa naturalmente più complessa, ed occorre che anche l'utente finale (e non solo il gestore) abbia consapevolezza della struttura dei dati

Generalmente verrà trattato un solo tipo di dati per ogni app, anche se il gestore potrà avere a che fare con tipi dati eterogenei.

Una descrizione più approfondita di questo tipo di applicazione verrà data nel prossimo capitolo che tratta i nodi X.

Grafo dell'applicazione

La struttura di una generica applicazione Telegram Road è ben descritta attraverso un **grafo.**

I nodi del grafo rappresentano i passi attraverso i quali si sviluppa l'applicazione. In ogni momento della sua vita, un'applicazione si trova in uno e un solo nodo del grafo..

La migliore rappresentazione per un grafo di Telegram Road graph è un **flowchart** (diagramma di flusso), che è ereditata da un approccio informatico del problema, anche se non ne sfrutta tutte le caratteristiche . Le caratteristiche di un flowchart di Telegram Road verranno descritte nel dettaglio nel prossimo capitolo.

Nodi

I nodi sono gli oggetti basilari dell'architettura di Telegram Road . Per come è concepito il grafo, un nodo può essere associato a una **domanda** che il sistema pone ed a cui l'utende dovrà rispondere. Ogni nodo porta con sé un certo numero di caratteristiche, che lo caratterizzano e il cui uso è descritto qui avanti.

Proprietà dei nodi

Ecco i campi (proprietà di un nodo generico:

- Nome della domanda : un nome breve per il nodo. Meglio usare un nome “parlante”, che chiarisca la funzione del nodo.
- Tipo : può essere alternativamente “S”, “I”, “Q” o “X” (vedi oltre)
- Presentazione: una breve introduzione, che viene presentata all'utente prima della domanda vera e propria.
- Domanda: questa è la domanda vera e propria che viene posta all'utente. A questa l'utente dovrebbe essere in grado con una scelta tra le risposte offerte.
- Immagine: una figura (opzionale) offerta insieme alla domanda, per aiutare la comprensione e rendere più piacevole la navigazione.
- Risposte: una lista di possibili risposte, che possono essere chiuse (scelta fissa) ma anche aperte (un testo da introdurre a piacere)

- Campo: Questo valore può essere usato per contenere una variabile chiesta all'utente e da usare poi nello sviluppo dell'applicazione. (su domande di tipo S, I o Q).
- Punteggio : un valore numerico per la domanda, da aggiungere indipendentemente da analoghi valori attribuiti alle risposte
- NCol: il numero di colonne in cui i bottoni delle risposte saranno presentati
- Iniziale: questa opzione identifica questo nodo come un nodo di partenza. Solo un nodo dovrebbe avere questo attributo vero.
- Finale: una risposta finale è un punto di arrivo per un'applicazione. A differenza dell'iniziale, un'applicazione può contenere più domande finali, che sono caratterizzate dal non portare una domanda, essendo nodi "di commiato".
- Prossima : puntatore alla domanda cui si va quando la risposta data non si adatta a nessuna di quelle disponibili

Tipi di nodo

I nodi possono essere di uno tra quattro possibili tipi:

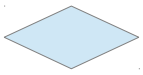
Nodo S

Un nodo di tipo "S", dove "S" sta per "selezione", è il tipo base per i nodi di Telegram Road. **Un'applicazione statica è composta di soli nodi S.**

Dato che un'applicazione statica ha la struttura di una sequenza di domande e risposte, un nodo di tipo S ha la funzione di una **domanda** , laddove gli archi che connettono i nodi tra loro sono le **risposte** che portano da una domanda all'altra.

Ecco perché in questo documento il termine "nodo" ed il termine "domanda" sono usati indifferentemente. Domande vengono poste anche per altri tipi di nodo, comunque.

Il simbolo grafico che verrà usato per rappresentare \un nodo S è il classico box di controllo dei flowchart:



con un ingresso e più uscite , tante quante le risposte offerte alla domanda. Il nome della domanda è riportato a fianco al box.

Il testo di ogni risposta è scritto invece sull'arco che ne esce.

Nodo I

Un nodo di tipo "I", dove I sta per "ingresso", permette di chiedere all'utente un valore, che viene salvato e memorizzato per passaggi successivi.

Il valore dato, che è sempre un **testo**, viene memorizzato in una variabile, il cui nome deve essere indicato nello spazio "campo" del nodo.

Dato che la risposta dell'utente in un nodo I non può avere altra funzione che riempire questa variabile, **un nodo I non può avere risposte associate**. Allora il nodo a cui si viene diretti in uscita a un nodo I è il nome puntato dal campo "Prossimo" del nodo I stesso,che ricordiamo essere l'uscita di "default" di un nodo.

Il simbolo grafico usato per un nodo di tipo I è il box di input usato nei flowchart (parallelogramma):

con un solo arco di ingresso ed uno di uscita.

Nota: così come per i nodi I, anche i nodi S possono memorizzare variabili. Il valore memorizzato sarà dato dall'opzione (risposta) data dall'utente, mentre il nome della variabile è sempre dato dallo spazio "campo" del nodo.

Nodo Q

Un nodo di tipo Q permette al manager dell'app di realizzare una query SQL. La query, generalmente di tipo SELECT, potrà fare riferimento ad una o più variabili precedentemente inserite da nodi "I" o "S".

La query verrà eseguita su un database (o per meglio dire datasource) che è puntata dal campo "datasource" della App. Dato che questo valore è unico, è possibile effettuare query su un solo datasource alla volta per ogni App.

La struttura (o "schema") del database deve essere ben conosciuta dal manager della App, in termini di nomi di tabelle e di campi, e le loro proprietà. Di fatto , un nodo Q è un punto fondamentale di un'applicazione di tipo dinamico.

Lo scopo principale di un nodo Q è quello di eseguire una query SELECT SQL , il cui obiettivo è quello di estrarre valori dal database, valori che saranno offerti all'utente come altrettante risposte da scegliere in uscita al nodo Q. In effetti, il nodo Q differisce da un nodo I in quanto le risposte in uscita di un nodo S sono determinate staticamente in fase di progettazione, mentre per un nodo Q sono date dinamicamente dall'estrazione di valori da un database.

Così come per i nodi S ed I, anche per un nodo Q il valore scelto dall'utente può essere a sua volta memorizzato in una variabile, il cui nome è da indicare al manager nell'area "campo" del nodo Q.

Insieme ai valori estratti dinamicamente, il manager può anche prevedere delle risposte statiche, come ad esempio "aiuto" o "esci", da usare come vie di uscita, o

valori di default (nel caso che nessun valore sia estratto dal database) . Queste risposte statiche vengono valutate in runtime prima di ogni altra opzione.

NOTE

1. si può estrarre un solo valore
2. Per il testo della query SQL va utilizzato il campo "Domanda" di un oggetto domanda. Ciò comporta che per un nodo Q viene presentato all'utente la sola Presentazione, non una domanda
3. poiché un campo generico di un database può avere qualsiasi nome, è mandatorio dare ai valori estratti un alias che deve avere nome **"val"**. Così, se il manager vuole presentare tra le possibili risposte il campo "Jobname" di una certa tabella "Jobs", la giusta query potrebbe essere:

   SELECT Jobname AS val FROM Jobs ORDER BY Jobname ;

Notare l'alias " AS val " . Dimenticare l'uso dell'alias produrrebbe un errore del tipo:

The column "Val" could not be found in table "Jobs".

Un nodo Q può contenere non solo istruzioni SELECT , ma anche INSERT, UPDATE o DELETE. Di fatto, viene rispettato l'intero set dell' SQL ANSI . Questo dà al manager un potere enorme nello sviluppo di applicazioni basate su database.

Il motore MySQL 5 , per esempio supporta

- JOINS a livello multiplo
- Inner queries

La query SQL può fare riferimento a valori precedentemente memorizzati attraverso nodi "I" , "S" e persino "Q". Questi valori sono salvati in una struttura apposita, denominata "CurrentQuery", che verrà descritta più avanti.

Se una variabile è stata memorizzata con un certo nome (ad es. var), essa dovrà essere referenziata dalla query circondandola da caratteri $, cioè var. Ad esempio, se si era memorizzato un valore "nome", una query che lo utilizza potrebbe essere :

SELECT indirizzo AS val FROM amici WHERE amici.nome = '$nome$'

Il simbolo grafico usato per il nodo Q è il seguente:

Nodo X

La combinazione di nodi **“S”**, **“I”** e **“Q”** da al manager dell'App grandi possibilità per costruire un'applicazione statica e dinamica.

Il limite di un nodo Q è la capacità di eseguire una sola query e non più di una. Ogni operazione su database che possa essere risolta in una sola istruzione SQL può essere risolta attraverso un nodo Q.

Purtroppo non tutte le situazioni sono di questo tipo. Se più operazioni su database sono necessarie, si potrebbero, è vero, far susseguire diversi nodi Q, ma l'effetto per l'utente finale sarebbe un insieme di **“**domande**”**, che avrebbe un effetto noioso e magari anche incomprensibile.

Il nodo di tipo X si rivolge a quest'esigenza, dando la possibilità di eseguire (*eXecute*) una procedura esterna. Il nome di tale procedura deve essere fornito (dal manager) nell'area **“**campo**”** del nodo stesso.

Tutti i valori di ingresso da fornire alla procedura devono essere stati precedentemente salvati nella struttura CurrentQuery (vedi più avanti 11) attraverso un uso appropriato di nodi S, I e Q .

Valori in uscita, se necessari, possono essere altresì salvati nella medesima struttura CurrentQuery .

Il nodo X dà al programmatore la potenza di un linguaggio di programmazione. L'effetto collaterale di questo nodo così potente è la necessità per il manager dell'App di avere capacità di programmazione, oltre alla conoscenza del linguaggio SQL richiesta dal nodo Q.

Questo è il motivo per cui i nodi X vengono usati in un limitato insieme di applicazioni, per le quali sono state scritte alcune funzioni di libreria in linguaggio Coldfusion [1].

Tutte queste applicazioni appartengono alla categoria denominata **“**Dynamic app, dynamic database**”**. L'uso di tali funzioni ed App è descritto in un altro documento

1Coldfusion è il linguaggio in cui è stata scritta l'applicazione Telegram Road

separato.

Il simbolo grafico usato per un nodo di tipo X è il seguente:

Riassunto dei tipi di nodo

Ecco per ricapitolare i tipi di nodo ed il loro comportamento:

Tipo	Output	Input	Prossimo	Action
S	Risposte predefinite	Una risposta scelta	Domanda puntata	Può salvare la scelta in campo
I	“Inserisci un valore”	Valore inserito	Prossimo	Salva la scelta in campo
Q	Selezione da query + risposte	Selezione singola	Valore puntato	Esegue una query, salva risultato in campo
X	Selezione da query + risposte	Selezione singola	Valore puntato dalla risposta	Esegue procedura esterna

Proprietà di una APP

Ogni App, una volta generata e registrata su Telegram Road, porta con sé le seguenti proprietà:

Nome App : un nome generico , che può essere lo stesso con cui l'App è stata registrata in Telegram. Ricordiamo che il bot Telegram (creato sotto Telegram attraverso BotFather) e la App di Telegram Road (che gira sui server di Telegram Road) sono due cose **differenti,** e sono solo collegate dai parametri Token e Template desctritti sotto. L'utente finale interagisce con L'App Telegram Road attraverso il bot Telegram (la sola cosa che vede sul proprio dispositivo)

Proprietario : un numero, associato con il dispositivo dell'utente finale, e caricato automaticamente all'atto della registrazione. NON DEVE ESSERE MODIFICATO.

Template : il nome della funzione (“Webhook”) associata con la App, creata all'atto della registrazione. NON DEVE ESSERE MODIFICATO.

Token : una stringa unica e segreta assegnata dal Botfather all'atto della creazione del bot. NON DEVE ESSERE MODIFICATO.

Datasource: il nome del database dell'applicazione dinamica . Non viene usato per applicazioni statiche.

Questi campi possono essere acceduti dal gestore dell' App manager attraverso

l'interfaccia web della console:

http://telegramroad.com/cfusion/interroga/index.cfm?chat_id=Owner

dove “Owner” è il summenzionato codice Proprietario associato al dispositivo da parte di Telegram.

Questo indirizzo (“url”) è accessibile sia da una piattaforma desktop (da preferirsi) , o anche da un dispositivo mobile (con un po' meno comfort).

CurrentQuery

CurrentQuery è una struttura dati incorporata nell' applicazione Telegram Road. E' dedicata all'immagazzinamento di parametri di ingresso e uscita usati nella vita dell'applicazione, e che devono essere mantenuti durante i vari passaggi (le “domande”) di cui la App è formata.

La natura ed il numero di tali parameri sono lasciati al manager dell' App . Il loro nome è definito nell'area “campo” di un nodo “S” ,”Q” o “I” (vedi “Proprietà dei Nodi”)

I parametri memorizzati in CurrentQuery possono essere usati nei seguenti modi:

possono essere richiamati da domande, in modo che l'end user vedrà domande che li contengono. Il modo per il manager di richiamarli è includere il loro nome attraverso due caratteri $. Per esempio, se nodi di tipo “I” hanno richiesto parametri chiamati “Nome” e “Cognome”, una ulteriore domanda in un nodo “S” potrebbe suonare come:

Salve, $Nome$ $Cognome$! Come vorresti proseguire?

offrendo così domande personalizzate.

NOTA: solo la parte “Domanda” di un nodo può tradurre i nomi di parametri in questo modo. La parte “presentazione” non può.

2. Possono essere usati in query, che è il modo di costruire interrogazioni in applicazioni dinamiche. Abbiamo visto sopra che le query sono una parte fondamentale dei nodi di tipo “Q”, che contengono la query nel loro campo Domanda. In questo modo, è possibile fare riferimento a loro, sempre con la notazione $variabile$.

Per esempio, se un precedente nodo “S” ha fatto scegliere all'utente una tabella su cui lavorare, e l'ha salvata nella variabile “myTable” , allora un nodo “Q” che segue può fare riferimento ad essa ponendo nel campo Domanda una query come:

SELECT names AS val FROM $MyTable$ LIMIT 10;

per estrarre nomi dalla tabella precedentemente scelta.

****Si prega di notare l'uso dell'alias val, come descritto nelle regole d'uso dei nodi Q.*

Oltre a tali valori espliciti, caricati in CurrentQuery esplicitamente attraverso nodi **"S"**, **"Q"** o **"I"**, le seguenti variabili sono caricate implicitamente e disponibili:

- **chat_id** : l'identificativo **"Proprietario"** passato da Telegram e legato al dispositivo;
- **ds**: il datasource assegnato all' App, come configurato in fase di progetto dal manager
- **app**: l'applicazione corrente

Geo-localizzazione

E' possibile costruire con Telegram Road applicazioni che siano consapevoli della posizione del terminale su cui stanno girando. Associando tale informazione alla conoscenza di dove alcuni servizi sono disponibili permette allo sviluppatore di un'applicazione una nuova importante possibilità .

Nota 1: La posizione (localizzazione) di un terminale deve essere quella di un dispositivo reale. I servizi di geolocalizzazione di Telegram NON sono disponibili su piattaforme di emulazione, com,e ad es. Telegram Desktop o Telegram Web.

Nota 2: Per essere localizzato, un terminale deve avere abilitate le sue funzionalità di localizzazione, come il Wi-Fi o (meglio) il GPS. Tali servizi sono offerti dai sistemi operativi degli smartphone (sia iOS che Android) e vengono attivati su richiesta se necessario.

La Geo-localizzazione si ottiene in Telegram Road tramite due differenti passaggi, che adesso andiamo a illustrare:

Richiamo della Localizzazione

Il richiamo delle informazioni di geo-localizzazione si ottiene usando in una delle risposte di un nodo un testo particolare. Al momento attuale , si può utilizzare indifferentemente una delle seguenti parole chiave:

geo

location

whereami

localize

Se la risposta consiste esattamente di uno di tali testi, allora l'applicazione:

- chiede all'utente conferma se vuole fornire le informazioni sulla propria

localizzazione

- salva tali info nella struttura CurrentQuery , sotto **location.longitude e location.latitude,** sotto forma di coordinate (numeriche) del terminale
- procede quindi con il prossimo nodo. ATTENZIONE: sarà seguito solo il nodo marcato come default nel nodo corrente, e non quello puntato eventualmente dalla risposta geolocalizzata, che in questo caso è ininfluente.

Uso della Geolocalizzazione

Dopo la localizzazione, l'applicazione può andare avanti recuperando le informazioni contenute in **location.longitude** and **location.latitude** dentro CurrentQuery**,** e confrontandole con altre localizzazioni di elementi contenuti nell'applicazione stessa. Ciò naturalmente dipenderà dall'applicazione stessa.

Certamente si dovrà fare uso di nodi di tipo Q (query) o X (Execute) per eseguire operazioni di confronto ed elaborazione di tali dati , come misura di distanze e recupero di risorse vicine.

Punteggi

Quello dei punteggi è un modo utile ed economico per costruire applicazioni che sommano un punteggio man mano che l'utente segue un percorso, in funzione della strada che ha seguito.

Esempi di questo potrebbero essere test, quiz culturali, e così via.

I punteggi possono essere assegnati a:

nodi : la visita di un nodo provoca l'assegnazione di un punteggio

risposte: scegliere una risposta invece che un'altra porta all'assegnazione di un punteggio legato a quella risposta.

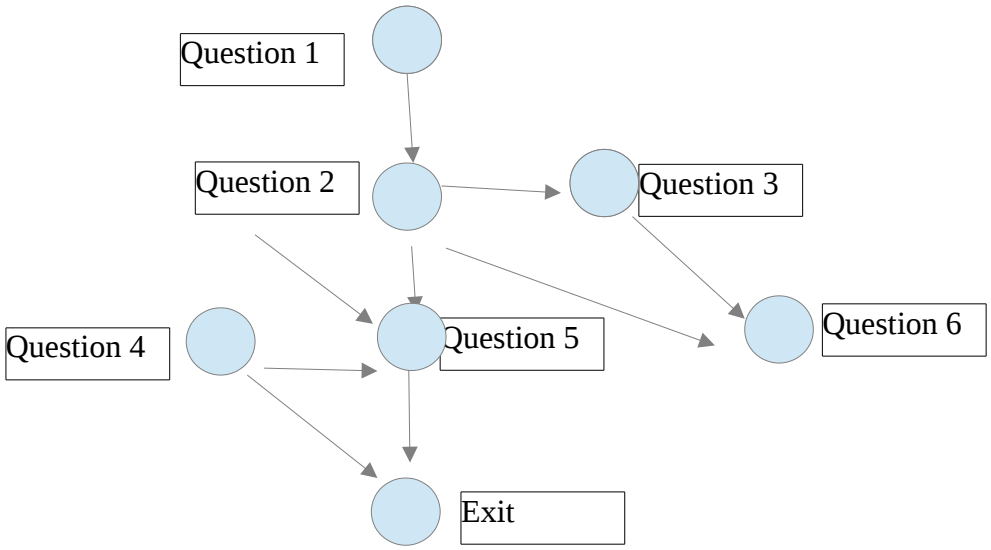
All'inizio dell'applicazione il punteggio cumulativo viene posto naturalmente a zero.

La somma finale di tutti i punti acquisiti può essere presentata in un nodo finale, basta includere la parola “score” (letteralmente) nel corpo della domanda

Tutto qui.

Creazione e registrazione di un'Applicazione

1 – Innanzitutto è bene avere carta e matita. Su un foglio, si possono disegnare le domande sotto forma di blocchi (magari "palle") che sono connesse tra loro tramite archi, rappresentanti le risposte, e che portano ad altri blocchi. Si può in questo modo disegnare un **grafo** che rappresenta la Applicazione, come ad es. questo



Questo grafo verrà sottoposto con ogni probabilità a molte modifiche e aggiornamenti, man mano che la App prende forma, così non preoccupatevi troppo di trovare la forma corretta subito.

2 – Il passaggio successivo è tradurre il tutto in un bot su Telegram. Occorre avere Telegram sul proprio dispositivo mobile , oppure sul proprio PC (*desktop*). In ogni caso, l'indirizzo dea visitare è :

http://telegram.org/dl

Nota: dal momento che la registrazione è legata al numero del proprio telefono cellulare, ci si può registrare una volta e poi entrare da diversi ambienti o postazioni, avendo la garanzia di ritrovare il proprio profilo (chat, bot, preferenze, …), perfettamente replicato ed allineato. Il numero di telefono cellulare è richiesto solo in questa fase (registrazione) , e non verrà utilizzato o rilasciato a chicchessia in nessuna occasione.

3 – Dopo l'installazione di Telegram, si deve cercare un bot particolare, chiamato **BotFather**, che serve come "padrone di casa" per la creazione di nuovi bot. La creazione di un bot, così come tutte le altre funzioni di Telegram, è **gratuita**.

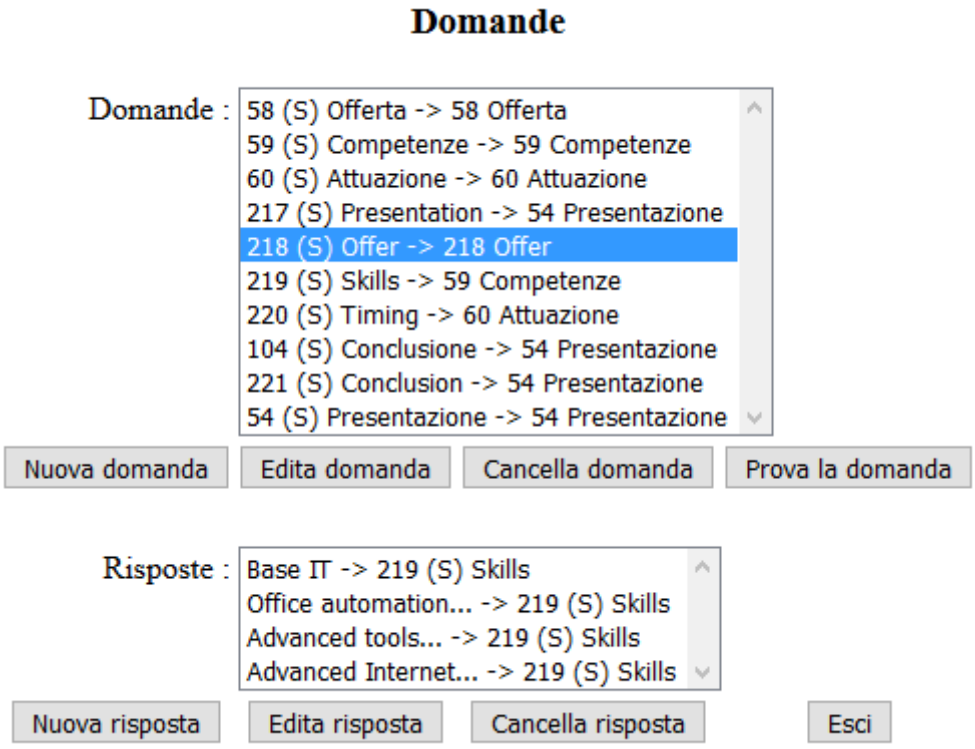
4 – Con il comando **/newbot**, dopo pochi semplici passaggi, possiamo dare al nostro bot un nome ed ottenere quindi un codice unico e segreto (*token*) che ci darà l'accesso al nuovo bot .

5 – Il passaggio successivo, sempre all'interno di Telegram, è cercare un altro bot, **Troadbot**. Questo bot, specifico della piattaforma Telegram Road, ci chiederà il nome dato al nuovo bot (quello scelto al punto 4) ed il token che BotFather gli ha assegnato. (La cosa migliore è senz'altro fare copia/incolla).

6 – Dopo aver registrato il nuovo bot, si hanno due possibilità. La prima prevede di lavorare da un computer su un'applicazione , il cosiddetto *backstage* di Telegram Road, dove avremo la possibilità di introdurre il grafo disegnato al passaggio 1.

Tutte queste attività sono demandate all'amministratore del bot ("App manager"), che possiede le credenziali (password) che gli danno il controllo dell'App.

7 – A questo punto l' App è pronta per essere testata su una piattaforma desktop o tablet (via web), e probabilmente essere sottoposta ad una fase di messa a punto (debug). Per esempio, questo è come potrebbe apparire al manager l'App nel backstage di Telegram Road:



8 – Potremo inserire qui i blocchi, corrispondenti alle domande, e gli archi, relativi alle rispettive risposte, di cui è composta l'App.

9 – In ogni momento, l'App può essere testata in tempo reale, sia su PC fisso che su un qualsiasi dispositivo mobile, nel suo naturale ambiente Telegram. E' disponibile altresì un emulatore, più adatto alle funzioni di debug.

10 – Una strada alternativa per inserire il grafo è descritta nel capitolo "Creazione interattiva di un'App".

Redirezione su chat

E' possibile dall'interno del bot redirigere l'utente su un'altra chat, sia privata (1 utente) che di gruppo. La cosa può avere un senso se il bot costituisce una guida automatizzata, che possa ad un certo punto indirizzare l'utente ad una interazione con umani che rispondano. Ha quindi senso che la redirezione avvenga verso un **gruppo**, e tale gruppo sia **privato** (ad es. un help desk).

Vediamo qui le azioni da fare.

- Costruire in Telegram un gruppo.
- Definirlo privato : Manage type > Group type > private
- copiare il link di invito
- Definire gli utenti del gruppo, a piacere (ad es. gli appartenenti all'help

desk)

- Entrare nel backstage di Telegram Road visto sopra , e creare un nodo di tipo **“S”.**
- Nella presentazione o nella domanda di tale nodo, inserire un testo che contenga il link di invito che punta al gruppo. Ad esempio, se il link di invito era
  - https://t.me/joinchat/B7MRxBg5hH_yNBIrOpD3X7
- allora la presentazione potrebbe essere :
  - Clicca qui per chattare con un nostro consulente: https://t.me/joinchat/B7MRxBg5hH_yNBIrOpD3X7

Creazione interattiva di un'App

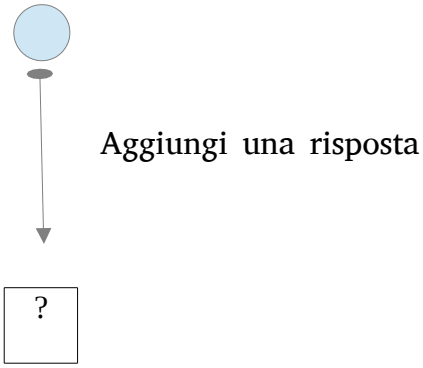
Dalla release **1.3**, si ha la possibilità di introdurre e creare un'App non solo per un manager , ma anche per l'utente finale. Questo dà la possibilità di creare un'App in modo collaborativo , dato che ciascuno può aggiungere nodi a un'App esistente, portando nuove opzioni (nodi, archi) al grafico che rappresenta l'App.

Questo può dare vita ad esempio a :

- **manuali** di manutenzione e di utilizzo, in cui ciascuno che conosca il modo giusto di risolvere un problema può portare il suo contributo con la sua conoscenza (*wiki*)
- **guide turistiche**, in cui notizie, informazioni e consigli possono essere aggiunti man mano dai turisti stessi
- **giochi**, nei quali lo sviluppo di una storia può essere costruito dai giocatori stessi, in un modo collaborativo.

Questa lista può crescere in funzione della fantasia dell'utente e degli sviluppatori.

Quando una nuova App nasce, la prima domanda porta con sé sempre un'opzione **"aggiungi una risposta"**.



Questo è un punto cruciale di un'applicazione Telegram Road , perché se un utente sceglie tale opzione:

- gli sarà domandata la risposta (testo) che sente che manca; questo testo si aggiungerà all'opzione "aggiungi una risposta"
- gli sarà chiesto se conosce un seguito a tale risposta; a questo si può pensare come un modo di risolvere un problema, puntato dalla risposta appena aggiunta.
- Se l'utente non conosce la soluzione, l'opzione appena aggiunta rimarrà

“appesa” , in attesa di qualcun altro che voglia partecipare espandendo il grafo. Aggiungere una risposta appesa permetterà ad altri utenti di sapere che un caso è ancora aperto ed attende un suo sviluppo.

- Se un altro utente (o magari l'utente stesso) rivisita la App e segue la nuova risposta, gli viene domandato se conosce una nuova soluzione.
- Accettando, gli viene richiesta una nuova domanda, cioè un nodo nel grafo a cui la risposta porterà, con le sue caratteristiche, che sono la introduzione, la domanda vera e propria e magari un'immagine di accompagnamento (vedi le caratteristiche dei nodi in 4)
- La nuova domanda ha automaticamente le caratteristiche di “aperta” ma non “pubblica”. Ciò permetterà l'ulteriore crescita del grafo, pur inibendo la pubblicazione del nodo fino a quando un amministratore lo abbia autorizzato. Solo il creatore e l'amministratore la potranno vedere fino a quel momento.
- Si potranno aggiungere nuovi archi (risposte) alla nuova domanda, ripetendo iterativamente i passaggi detti sopra. Ogni nuovo nodo “aperto” porterà con sé l'opzione “aggiungi una risposta” per permettere al grafo di crescere ulteriormente.

Questa possibilità di aggiungere opzioni da parte dell'utente finale dà a Telegram Road una grande potenzialità di costruire applicazioni innovative e collaborative senza aver bisogno di alcuno strumento di sviluppo (linguaggi di programmazione, conoscenze informatiche).

Il manager dell'App mantiene naturalmente la possibilità di editare ogni singolo nodo ed arco del grafo, sempre facendo uso del backstage di Telegram Road, che gli dà tutti gli strumenti per farlo (14)

NOTA: i nodi introdotti in questo modo sono solo di tipo “S” (vedi 5) . Non è possibile introdurre altri tipi di nodo (I, Q o X) se non con un controllo manuale. Ciò sta a significare che le applicazioni nate in tale modo sono solo di tipo **statico** (vedi 3).

Come posso ottenere informazioni?

E-mail : telegramroad@gmail.com

Web : http://telegramroad.com