

## Il Framework di IG

Uno degli strumenti più interessanti del progetto di IG è senza dubbio il suo Framework. Sviluppando per anni applicazioni di diverso tipo ma che insistevano sempre sugli stessi strumenti, vale a dire Database, Ambiente Web, Operazioni comuni su date su stringhe etc si è arrivati a sviluppare un framework che via via nel tempo ha creato uno pseudo linguaggio che implementa a quanto offerto dal Perl altre funzioni più specifiche.

Il Framework si deve posizionare tra gli strumenti il linguaggio che occorre per gestirli e l'applicazione che vogliamo realizzare. Questa astrazione permette di liberare le varie applicazioni da problemi di logica che possono essere affrontati semplicemente operando sul framework. Infatti una delle potenzialità sta nel fatto che solo ottimizzando il framework di IG si ottimizzano tutte le sue applicazioni.

Inoltre utilizzare un Framework offre l'enorme vantaggio di adeguare l'intero progetto a nuovi strumenti senza dover modificare nessuna delle applicazioni. Pensate ad esempio all'implementazione di un nuovo Database, il framework di IG infatti mette a disposizione dello sviluppatore funzioni comuni per l'invio di query e il recupero dei dati indipendentemente dal database che si sta utilizzando. Se volessimo quindi oltre a MySQL e Postgres implementare ad esempio l'uso di SQLite (prossimamente verrà veramente fatto) basterebbe agire esclusivamente all'interno del framework e fare in modo che le varie funzioni specifiche per l'uso di database operino le giuste chiamate a SQLite.

### Un primo approccio

Un modo veloce per approcciare al framework di IG e capire come funziona, è quello di visualizzare e provare le applicazioni "demo" presenti nel pacchetto. Ad esempio questa piccola applicazione <http://www.igsuite.org/cgi-bin/demo1> è un insieme di features di IG rappresentate da un form che ogni volta che viene inviato richiama se stesso dimostrando la gestione delle sessioni e dei dati via web. Una volta visualizzata e provata l'applicazione se ne può studiare il codice che la genera prendendolo da <http://www.igsuite.org/live/3.2.2/demo1>

### L'ambiente definito dal Framework

Il Framework di IG ha come obiettivo principale quello di offrire allo sviluppatore un ambiente di sviluppo che contenga funzioni e informazioni necessarie allo sviluppo di applicazioni web-based.

Utilizzare il modulo di IG significa definire preliminarmente una serie di variabili "globali" che impostano l'ambiente in cui opererà l'applicazione. Ad esempio ci saranno variabili globali che definiscono la localizzazione o altre che definiscono lo skin grafico, ogni aspetto che riguarda le impostazioni grafiche, lo stile, i permessi etc.

Per comprendere completamente il modo in cui opera il Framework elencheremo passo per passo le operazioni che compie al momento in cui viene utilizzato da parte dell'applicazione.

#### Step 1 - Impostazione di variabili Globali

Molte delle variabili globali che definiscono l'ambiente del framework sono esportate dal modulo IG.pm nel namespace "main".

[Variabili che definiscono la sessione.](#)

#### **\$session\_timeout**

Definisce il tempo massimo di ogni sessione autenticata

**\$cgi\_dir**

Contiene il percorso dove sono presenti gli script di IG

**\$logs\_dir**

Contiene il percorso della directory dove sono contenute le sessioni attive (generalmente in "\$cgi\_dir/log").

**\$conf\_dir**

Contiene il percorso della directory che contiene i file di configurazione di IG

**\$user\_dir**

Contiene il percorso della directory "Home" dell'utente autenticato (generalmente in "\$cgi\_dir/data/users").

**\$htdocs\_dir**

Contiene il percorso della Document Root di IG. In genere coincide con quella configurata in Apache

**\$temp\_dir**

E' una directory temporanea utilizzabile e nella quale automaticamente IG cancella i file più vecchi di 15gg (generalmente in "\$cgi\_dir/data/temp").

### Variabili per l'input/output da e verso i form

**%in**

Questo è l'hash nel quale IG salva tutti i parametri provenienti dai form inviati dall'utente e quotati cioè pronti per essere inseriti nel database.

**%on**

Questo hash è il gemello del precedente, con la differenza che in esso i valori non vengono quotati ma lasciate esattamente allo stato originale.

**%cookie**

Questo hash è popolato da tutti i cookies passati dal browser del client.

**%set\_cookie**

Attraverso questo cookie è possibile per lo sviluppatore creare in modo molto veloce nuovi cookie. Infatti al momento dell'invio della pagina di risposta IG controlla il contenuto di questo hash e invia nuovi cookie al browser del client.

### Variabili che definiscono l'utente autenticato

**%users**

Contiene tutti gli utenti registrati in IG con alcuni dati principali.

**\$auth\_user**

Contiene la login dell'utente autenticato

**\$login\_admin**

Contiene la login dell'amministratore (impostata in igsuite.conf)

**\$remote\_host**

Contiene l' Ip/host dell'utente autenticato

## Step 2 - Ottiene ulteriori informazioni dai form e dai file di configurazione

La lettura dei dati inviati al server attraverso i form è affidata al modulo CGI Simple.pm e più precisamente ad un suo wrapper che è la funzione IG::MkCgiEnv(). Tale funzione oltre a leggere i parametri da CGI Simple.pm ricava anche i valori hidden da una cache su database che salva i valori in funzione dell'id di sessione dell'utente.

A questo punto IG legge il file 'igsuite.conf' importando tutti i valori specificati per le seguenti funzioni:

- Imposta l'accesso al db;
- Imposta l'accesso a Hylafax;
- Imposta il percorso per accedere ad applicazioni esterne

## Step 3 - Autenticazione dell'utente

L'autenticazione di un utente da parte di IG avviene attraverso la creazione di una sessione e il passaggio di un cookie di sessione al client dell'utente. Per la creazione della sessione viene creato un file in \$logs\_dir che mantiene l'id di sessione e registra mano a mano il tempo trascorso dall'ultimo utilizzo della sessione da parte dell'utente.

Nel caso IG non riesce ad autenticare l'utente definisce quest'ultimo come utente "guest". L'utente guest può accedere ad alcune limitate applicazioni all'interno dell'ambiente di IG (igchats, igwiki, igfile).

## Step 4 - Imposta l'ambiente specifico per l'utente

Nell'ambiente definito dal Framework vi è la possibilità di specificare per ogni utente molti parametri e preferenze personalizzate. Tali impostazioni vengono salvate in un file all'interno della home dell'utente che in genere è del tipo ( Esempio: /cgi-bin/data/user/lucas/lucas.cf ). Ad ogni chiamata le impostazioni personalizzate sovrascrivono quelle di default del sistema.

## Step 5 - Imposta valori di default per i parametri non ancora definiti

Alcuni parametri che in genere potrebbero non essere definiti e che hanno quindi bisogno di valori di default sono per esempio: la localizzazione del linguaggio; lo skin; il formato della data e molti altri.

## Step 6 - Imposta alcuni valori temporali

La gestione del tempo è importantissima all'interno di IG è per questo che vengono impostati dei valori utili all'interno delle applicazioni in relazione all'ora locale al fuso orario e alla localizzazione del formato della data.

## Le funzioni del Framework

Nonostante qualcuno non approverà tale scelta, abbiamo preferito utilizzare per le funzioni del framework la convenzione di mettere le iniziali maiuscole per differenziarle da funzioni core o funzioni importate da altri moduli.

### Funzioni generiche

#### MkUrl(\$TextToUrlize)

Codifica una stringa di caratteri per essere utilizzata all'interno di un URL.

### **MkEntities(\$TextToEscape)**

Codifica una stringa di caratteri sostituendo e definendo entità Html (Wrapper a HTML::Entities).

### **ParseLink(\$TextToLink)**

Genera dei collegamenti automatici ai documenti gestiti dal protocollo di IG in relazione anche ai privilegi dell'utente autenticato.

### **MkLink(\$TextToLink)**

Elabora una stringa di caratteri per sostituire alcuni tag di IGWiki (solo un piccolo subset). All'interno di MkLink viene anche chiamato ParseLink.

### **SendIsms()**

Invia un messaggio Isms. E' possibile passare come parametri un hash con le seguenti chiavi: sender receiver body msgto reply delrepliedmsg type.

### **TrashDoc(\$protocol\_id)**

Cestina un documento di IG. Accetta come parametro il numero di protocollo (Es. 200123.02)

### **GetDayByDate(\$day,\$month,\$year)**

Ritorna il giorno della settimana di una data.

### **GetValuesByDate(\$date)**

Ritorna una lista di 3 valori corrispondenti al giorno mese e anno di una data passata come stringa. Elabora la data in funzione della localizzazione del formato della data.

### **GetDateByFormat(\$day,\$month,\$year)**

Ritorna una data formattata in base alla localizzazione del formato della data.

### **SumDate(\$day,\$month,\$year,-4)**

Ritorna una data formattata in base alla localizzazione del formato della data, ottenuta dalla somma o sottrazione di valori espressi in giorni mesi o anni su una data passata come parametro  
Es. +1y (add 1 year) +3m (add 3 months) -50 (less 50 days)

### **CkDate(\$mydate,\$flag)**

Controlla il formato di una data in base alla localizzazione del formato corrente completando se necessario la data con i valori della data attuale.

31.12.2004	Valore corretto
31122004	Valore corretto
31.12.04	Valore corretto
311204	Valore corretto
31.12	Valore corretto, IG aggiunge l'anno corrente.
3112	Valore corretto. Come sopra.
31	IG aggiungerà il mese e l'anno corrente.

Se nessun valore è specificato o è specificata una data non valida la funzione ritorna '0'.

### **MkByte(\$file\_size)**

Converte un valore numerico della dimensione di un file in una forma più leggibile.

### **CheckSecur(\$my\_procedure)**

Controlla se un utente in base ai suoi permessi può eseguire una determinata operazione. Nel caso l'utente ha i permessi richiesti la funzione ritorna il valore 1 altrimenti blocca l'esecuzione dell'applicazione e visualizza un messaggio che avvisa dell'impossibilità a procedere per via dei suoi privilegi.

Se non viene passato nessun parametro la funzione ritorna 1 quando si ha un utente con permessi di Amministratore altrimenti blocca l'esecuzione.

### **CheckPrivilege(\$my\_procedure)**

Come nella funzione CheckSecur controlla i privilegi dell'utente ma ritorna '1' se l'utente ha i privilegi o '0' in caso contrario senza bloccare l'esecuzione dell'applicazione.

Se non si specifica nessun parametro la funzione controlla se l'utente autenticato è un utente "guest" ritornando '0' o '1' nel caso è un utente di IG.

## **Funzioni che definiscono la Gui**

### **MkMenu()**

Genera il menù che elenca le features di IG installate.

### **HtmlHead()**

Genera una valida e completa intestazione Html.

### **HtmlFoot()**

Chiude la pagina Html disattivando nello stesso tempo il buffer di output.

### **TaskHead(options)**

Il 'Task' all'interno del Framework di IG è un elemento della GUI. Esso è il contenitore di form, liste di valori e tanti altri elementi della Gui. Si potrebbe pensare al 'Task' di IG come ad una finestra di un comune window manager.

### **TaskMsg(\$my\_msg,\$my\_icon,\$my\_width)**

Visualizza un box contenente informazioni del tipo Warning o Info. Ne esistono vari tipi a seconda del secondo parametro passato che può assumere i seguenti valori:

ico=0 Warning Message

ico=1 Info Message

ico=2 Other Info Message

ico=3 Black info message

ico=4 Very very nice task message

### **QuickHelp()**

Genera un quick help visualizzabile attraverso un evento 'onmouseover'.

### **TaskFoot()**

Chiude un elemento TaskHead();

## **TaskListMenu()**

Un 'TaskList' è un'altro elemento della GUI di IG che in genere visualizza il risultato di query inviate a database. Attraverso questa funzione si definisce il menù (la prima riga) della tabella.

## **TaskListItem()**

E' la funzione che genera le righe del TaskList.

## **TaskListFoot(\$empty\_rows\_showen\_if\_any)**

Chiude il TaskList.

## **MkCalendar**

Genera un calendario.

## **MkTaskPage(\$show\_many\_pages,\$base\_url)**

Offre la possibilità di dividere elenchi di risultati in più pagine con un comodo widget che ne permette di sfogliare le pagine.

## **Button(\$my\_text,\$my\_link,\$my\_alt,\$procedure\_permissions)**

Genera un Bottone da inserire nella ButtonBar del Task.

## **MkMap()**

Genera un link automatico a [www.mapporama.com](http://www.mapporama.com) per la generazione di una mappa prendendo i dati dall'indirizzo di un contatto.

## **MkGraph()**

Genera un istogramma attraverso le serie di valori numerici passate.

## **GetTableVal()**

Estrae valori da una tabella base.

## **Gestione dei Form**

### **ReadParam()**

Fa il parsing dei valori passati attraverso un form.

### **FormHead()**

Genera l'header del form e crea uno spazio indicizzato dall'id di sessione per salvare i valori dei campi hidden.

### **Input()**

Inserisce un campo all'interno del Form che si sta generando. Questa funzione è unica per ogni tipo di capo (text select submit etc) e cambia il suo output in base ai valori passati.

Text Box	Select Box	Advc Select	Button/Radio box	Buttons
text hidden	select combo	logins template	checkbox radio	button submit

Text Box	Select Box	Advc Select	Button/Radio box	Buttons
textarea password date file	multiselect	findable sendmode basictable contact		reset image

Parameters	Text Box	Select Box	Advc Select	Button/Radio box	Buttons
show	yes	yes	yes	yes	yes
name	yes	yes	yes	yes	yes
style	yes	yes	yes	yes	yes
class	yes	yes	yes	yes	yes
size	text file password	yes	yes	no	no
rows	textarea	no	no	no	no
cols	textarea	no	no	no	no
wrap	textarea	no	no	no	no
data	no	yes	no	no	no
dir	no	no	template	no	no
alt	no	no	no	no	image
src	no	no	no	no	image
maxlen	yes	no	no	no	no
value	yes	yes	yes	yes	yes
order	no	yes	yes	no	no
override	yes	yes	yes	yes	yes
intable	yes	yes	yes	yes	yes
pattern	text	no	no	no	no
allvalue	no	yes	yes	no	no
zerovalue	no	yes	yes	no	no
multiple	no	yes	no	no	no
quickhelp	yes	yes	yes	yes	yes
table	no	basic table	no	no	no
onchange	yes	yes	yes	yes	yes
onfocus	yes	yes	yes	yes	yes
onblur	yes	yes	yes	yes	yes
onclick	yes	yes	yes	yes	yes
onselect	yes	yes	yes	yes	yes
onmouseover	yes	yes	yes	yes	yes
onmouseout	yes	yes	yes	yes	yes

### **HLayer()**

Permette di impaginare un serie di elementi grafici.

### **Br(n)**

E' solo un modo veloce per dire <br />" x n;

### **FormFoot()**

Chiude il form corrente.

## Funzioni generiche

### LogD()

Logga operazioni all'interno del database

### RemoveOldSession()

Rimuove sessioni scadute.

### Logout()

Disconnette un utente specifico.

### FileCopy(\$old\_file,\$new\_file,\$flag)

Effettua la copia di un file da \$old\_file a \$new\_file e cancella \$old\_file se \$flag è uguale a '1'

### SysExec()

Esegue un applicazione esterna. E' un wrap a system() di Perl con l'aggiunta che esegue una ricerca attraverso la variabile PATH per individuare l'applicazione da eseguire nel caso non la trova.

### MkLastNum()

Genera un protocollo univoco per un determinato tipo di documenti.

### MkId()

Genera un id numerico composto dal risultato della funzione perl "time" più 10 numeri generati in modo random.

## Funzioni per l'accesso al Database

La gestione dei dati in IG avviene completamente attraverso l'uso di RDBM e i relativi moduli Perl a disposizione. Attualmente IG può lavorare con Postgres e con Mysql questo grazie alle funzioni comuni messe a disposizione dal framework. Si fa presente che il modulo utilizzato per Postgres non è quello DBI ma quello nativo per PG questo dovrebbe far capire al lettore quanto l'interfaccia offerta dal framework attraverso e accomuni differenti sistemi di approccio al database.

Per lanciare query verso database, IG si avvale di sole tre funzioni: DbQuote, DbQuery, FetchRow.

### DbQuote()

Questa funzione serve molto semplicemente a 'quotare' i valore da passare all'interno delle query.

### DbQuery()

E' la funzione con la quale si lancia la query al database

### FetchRow()

E' la funzione con la quale si recupera una array contenente ad ogni chiamata i valori dei campi di un singolo record.



## I parametri CGI standard

Ogni risultato, ogni form ogni cosa generata da IG è generata in virtù di una chiamata ad un CGI utilizzando quindi metodi comuni a tale interfaccia. Ovviamente IG supporta sia il metodo GET che POST è da sottolineare però che l'url che compone le chiamate ai vari cgi puo' contenere parametri 'standard' cioè validi per ogni applicazione di IG. Tramite tali parametri si puo' influenzare il comportamento del framework di IG indipendentemente dall'applicazione.

Parametro	Descrizione
action	L'action è il parametro che definisce quale azione deve essere eseguita all'interno dell'applicazione. L'action di default è 'lista' che in genere visualizza la pagina iniziale dell'applicazione.
tema	E' possibile forzare il rendering delle pagine al tema specificato con questo parametro sovrascrivendo le impostazioni definite dall'utente autenticato. Se il parametro espresso è errato o inesistente viene utilizzato quello di default.
lang	E' possibile forzare il linguaggio di IG con questo parametro sovrascrivendo le impostazioni definite dall'utente autenticato. Se il parametro espresso è errato o inesistente viene utilizzato quello di default.
print	Se valorizzato (quindi uguale a 1 o on o qualsiasi cosa) viene forzato il rendering delle pagine in forma stampabile.
attach	Se valorizzato (quindi uguale a 1 o on o qualsiasi cosa) viene visualizzata la pagina ma il suo url viene aggiunto automaticamente ai 'preferiti' dell'utente autenticato.

## Direttive per lo sviluppo

Per coordinare il lavoro di sviluppo sul progetto occorre assolutamente definire qualche direttiva o netiquette o forse meglio dire 'orientamento'. Infatti da sempre nello sviluppo del progetto sono state rispettate delle regole a volte anche a svantaggio delle features.

1. IG è un progetto OpenSource, è a disposizione di tutti, è aperto a tutti. La sua apertura deve essere estesa anche alla portabilità di IG e degli strumenti di cui IG stesso fa uso.
2. Qualsiasi features implementata nel progetto deve poter funzionare perfettamente sia su piattaforma Windows che su piattaforma Linux.
3. Se si scrive una features che fa uso di un'applicazione esterna, tale applicazione deve essere anch'essa un progetto OpenSource e deve poter girare sulle stesse piattaforme su cui gira IG.
4. IG è scritto in Perl. Per coerenza ma anche per non rendere più difficile l'installazione si è scelto di non utilizzare parti di codice ad esempio scritte in php o python.
5. IG non deve aver bisogno di nessun modulo Perl non Core per funzionare. Qualora sia necessario sviluppare applicazioni che hanno bisogno di tali moduli occorrerà inserirli unbundle nel progetto, e chiaramente questa direttiva vale solo per moduli scritti in Pure Perl portabili sia su windows che su Linux e che supportino mod\_perl.
6. Il codice deve essere sviluppato in inglese, vale a dire con commenti o nomi di variabili/funzioni/tabelle in inglese. Questa regola non è stata sempre applicata, ma da tempo si sta procedendo a tradurre tutto il codice.
7. Il Linguaggio SQL utilizzato nelle query deve essere scritto in modo che possa funzionare su tutti i database supportati da IG.
8. Qualsiasi nuova applicazione creata utilizzando il Framework di IG va approvata dal team di sviluppo per poter essere inserita nel progetto.
9. Qualsiasi modifica ad un'applicazione esistente va approvata dall'autore dell'applicazione stessa prima di essere inserita nel CVS.
10. Qualsiasi modifica al Framework va approvata da tutti gli sviluppatori del progetto in modo da mantenere la compatibilità con tutte le applicazioni che fanno uso del framework.