Interscratchet - guida all’installazione e all’utilizzo

# Cos’è

Interscratchnet (ISN) è una collezione di script php autoconsistenti, che permettono a N scratch di scambiarsi variabili di tipo stringa e numero sulla rete.

ISN, accompagnato da una opportuna estensione permette quindi di realizzare giochi in rete locale o Internet, ove il componente denominato Central Server fosse installato su una LAMP esposta su internet.

# Componenti

### 

### ScratchLocalServer. (SLS)

Installato in locale.

Servizio lanciato senza parametri dalla linea di comando di tipo PHP CLI (Php5). Resta in ascolto su una porta definita in testa, dei comandi Http provenienti da Scratch (Quindi implementa il protocollo delle estensioni http di Scratch 2.0).

E’ costituito dai seguenti files:

* httpserver-master/examples/scratchLocalServer.php
* httpserver-master/\*.\*

ScratchLocalserver implementa la gestione di due code (output e input) per ogni destinatario di connessione. In sostanza Scratch consuma le code di input e riempie le code di output

SLS gestisce anche una coda di comunicazioni (busyqueue) in modo che la comunicazione con un utente possa avvenire solo dopo che Scratch (con il comando “Lascia libero per altri” - /releas) abbia finito di comunicare con un altro utente. Questo permette la gestione asincrona da Scratch delle comunicazioni di diversi sprite o diversi blocchi dello stesso sprite senza confondere comandi e variabili tra diversi destinatari

Per lanciare Scratchlocalserver è sufficiente il comando ***php ScratchLocalServer.php***

### Asyncronous Variable Exchanger (AsyncVE)

installato in locale su processo differente da **SLS**

Ha il compito di implementare il protocollo esposto da SLS in modo da consumare la coda di output di SLS (/dumpcommands) e spedire i comandi (POST) al Central Server. Inoltre ha il compito di scaricare i comandi destinati all’utente dichiarato in Scratch (con il comando “Io sono” - user) dal Central server e spedirli a Scratch stesso con la chiamata http GET /pushVariables.

AsyncVE non ascolta su porte in quanto esegue chiamate

* E’ costituito dal solo file **httpserver-master/examples/AsyncVE.php**

Il comando per lanciare AsyncVE è**:**

***php asyncVe.php* *lp:<porta\_locale> rp:<porta\_remota> ct:<test\_path> is:<input\_script> os:<output\_script>***

dove:

* **porta\_locale:** porta sulla quale risponde SLS
* **porta\_remota:** porta sulla quale risponde il server centrale (il nome/indirizzo viene impostato da Scratch col comando “Il server è” - server)
* **test\_path:** script di test che AsyncVE invoca (GET) sul Server centrale per testare al connessione. Semplicemente deve ritornare **OK**
* **input\_script:** path (senza il nome del server - e.g. *scratchnet/sender.php*) dello Script dal quale vengono richeste le variabili destinate a essere mandate verso SLS e quindi prese da Scratch
* **Output\_script:** path (senza il nome del server - e.g. *scratchnet/receiver.php*) delloscript AL QUALE vengono postati i comandi che da Scratch sono stati inseriti nella coda di output di SLS e successivamente scaricati da AsyncVE (/dumpcpmmands)

### Central Server (CS)

Installato su un server in rete locale o Internet

Collezione di tre script php da installare sotto un qualsiasi Web Server (e.g. Apache). Fa uso di Mysql per adesso con uteste root (cablata) senza password.

Il DB si chiama “**scartchnet**” nome cablato negli script e contiene una sola tabella denominata **commands**

i tre script:

* **httpserver-master/CentralServer/test.php**: restituisce OK
* **httpserver-master/CentralServer/receiver.php**: riceve (via POST http) i comandi da AsyncVE, li parsa e inserisce nella tabella commands
* **httpserver-master/CentralServer/sender.php**: seleziona i comandi destinati all’utente sul quale risiede il AsyncVE che sta invocandolo (“WHERE destination = ‘...’ ), li formatta e li passa nella response.

### Estensione di prova (primaestensione.json)

file **httpserver-master/extensions/primaestensione.json**

E’ il file che viene importato in Scratch e permette di visualizzare e eseguire i blocchi di codice.

di seguito si riporta il testo attuale:

*{ "extensionName": "InterScratchNet",*

*"extensionPort": 8000,*

*"blockSpecs": [*

***["r", "nome del server", "servername"],***

***["r", "nome della persona con cui dialogo", "counterpart"],***

*["r", "frase1", "frase1"],*

*["r", "frase2", "frase2"],*

*["r", "frase3", "frase3"],*

*["r", "frase4", "frase4"],*

*["r", "frase5", "frase5"],*

*["r", "numero1", "numero1"],*

*["r", "numero2", "numero2"],*

*["r", "numero3", "numero3"],*

*["r", "numero4", "numero4"],*

*["r", "numero5", "number5"],*

***[" ","Il server e %s", "server"],***

***[" ","Io sono %s", "user"],***

***["w","Comunica e ascolta da %s e aspetta", "dest"],***

***[" ","Spedisci sul numero %m.numeri il valore %n", "numericvalue"],***

***[" ","Spedisci sulla frase %m.frasi il valore %s", "stringvalue"],***

***[" ","lascia libero per altri", "release"],***

*],*

*"menus": {*

*"numeri": ["numero1", "numero2", "numero3", "numero4", "numero5"],*

*"frasi": ["frase1", "frase2", "frase3", "frase4", "frase5"],*

*},*

*}*

Per importare l’estensione si tenga premuto il tasto “SHIFT” e si clicchi sul Menu file. Apparirà una voce in più in fondo “Import Experimental http extension” che permetterà di scegliere il file.

L’estensione può essere modificata a piacimento con le seguenti condizioni che permettono il corretto funzionamento con i componenti sopra. Ovviamente le label (e.g. **“*nome del server”)***sono libere.

* Gli elementi in grassetto devono permanere
* Gli elementi in blu rappresentano variabili che si ricevono
* I comandi in verde sono quelli per spedire variabili alla controparte scelta con il comando *["w","Comunica e ascolta da %s e aspetta", "dest"],*
* Le label delle variabili in blu , di tipo **“r”, devono essere uguali al alias interno** (e.g.*["r", "frase2", "frase2"],* sarebbe errato *["r", "frase2", "string2"],)*
* I menu (in rosso) possono essere sostititi da testi liberi, ma in questo caso il programmatore sarà costretto a scrivere personalmente il nome della variabile da spedire invece di sceglierlo da menu. potrebbe essere utile nel caso si voglia usare per spedire comandi una variabile - col relativo contenuto - algoritmicamente scelta tra quelle disponibili.
* Le voci dei menu in rosso possono essere differenti dalle label delle variabili che si ascoltano, ma in questo caso lo scratch che potrà ricevere il comando Spedisci sulla frase … o “spedisci sul numero …” dovrà avere una variabile dal nome corrispondente alla voce di menu. Ad esempio se nel menu “frasi” scegliamo sostituiamo “frase1” con “string1” potrà ricevere una variabile spedita col nome “string1” solo uno Scratch che abbia importato una estensione che abbia una riga *["r", "string1", "string1"]*, (quindi NON la stessa estensione sopra riportata).

**ATTENZIONE**:

L’unico gruppo di file che deve permanere con la struttura di directory indicata è quello relativo a SLS. Gli altri file possono risiedere in qulsiasi percorso, anche se naturalmente si consiglia di tenere raggruppati nella stessa Dir. i tre script del CS

### Interscratchnet\_demo.sb2

**IMPORTANTISSIMO: con le vecchissime versioni di Scratch NON FUNZIONA. Ieri provato con l’ultima installata funziona benissimo. bisogna aggiornare quello sulla chiavetta Scratch**

File di scratch che importa l’estensione json sopra riportata e implementa una semplice chat a due.

Si noti il particolare del comando “lascia libero per altri” all’inizio dello script che rappresenta la persona (il gatto). In genere si immagina che questo comando venga posto alla fine di un ciclo di comunicazioni, ma in questo caso, non dovendo switchare con altri” serve solo a eviutare che una precedente esecuzione del progetto abbia lasciato il destinatario impostato e quindi quello di una nuova esecuzione (lo stesso) venga messo in coda.

## TODO

### Riparare il bug costituito dal fatto che non vengono passate frasi con apostrofo (solito bug nella INSERT in una tabella, visto che l’apostrofo è il delimitatore di stringhe SQL

* ~~implementare il comando “\_reset\_all” che viene lanciato da scratch allo stop dell’esecuzione. L’implementazione consiste nello svuotare le code di input, di output, le variabili globali, e la coda di busy di SLS~~
* Implementare il “Broadcast”, cioè mettere nei comandi di spedizione la voce *TUTTI ,* che provoca in SLS il fatto che il comando venga replicato su tutte le code di output di tutte le controparti al momento esistenti