Java CGI HOWTO

di David H. Silber *dhs@orbits.com* v0.4, 18 Novembre 1996

Questo HOWTO spiega come configurare il vostro server per accogliere programmi CGI scritti in Java e come utilizzare Java per scrivere programmi. Nonostante gli HOWTO abbiano come obiettivo l'utilizzo di Linux come sistema operativo, questo documento si rivolge a tutti gli utilizzatori indipendentemente dalla versione di unix usata. La traduzione italiana è stata curata da Luca Rossetti (*lukaros@tin.it*).

Contents

1 Introduzione

A causa del metodo con cui Java è stato scritto, il programmatore non ha un accesso molto semplice alle variabili d'ambiente del sistema. Inoltre, per le modalità con cui il Java Development Kit (JDK) è stato costruito, per lanciare un programma è necessario utilizzare segnali multipli, cosa che non si addice molto alla metodologia di operazioni dello standard HTML forms/CGI. Esistono vari metodi per eliminare queste limitazioni, ed io ho implementato uno di questi. Il seguito di questo documento ne spiega tutti i dettagli.

1.1 Conoscenze necessarie

Si assume che abbiate una conoscenza generale del linguaggio HTML e dei concetti legati al CGI e almeno una minima conoscenza del vostro server HTTP. Dovreste anche conoscere la programmazione in Java o molto di quanto leggerete non avrà granché senso.

1.2 Dove trovare questo Documento

La versione più recente di questo documento si può trovare presso *http://www.orbits.com/software/Java_CGI.html* .

1.3 Il Pacchetto Software

La versione più recente del pacchetto software descritto può essere scaricata mediante FTP anonimo presso *ftp://ftp.orbits.com/pub/software/java_cgi-0.4.tgz* .La distribuzione del pacchetto include anche un sorgente SGML di questo documento.

Se utilizzate questo software, rendete disponibile un collegamento alla pagina *http://www.orbits.com/software/Java_CGI.html* , cosicché altre persone possano trovare le classi CGI Java.

1.4 Inserzione senza Pudore

Questo documento vi proposto viene grazie a **Stellar Orbits Technology Services**. (Visitate il nostro sito *http://www.orbits.com/* per vedere di cosa ci occupiamo.)

2 Come Configurare il Vostro Server per lanciare Programmi Java CGI (con Spiegazioni)

Questa sezione vi guiderà nella installazione dei miei pacchetti *Java CGI* con spiegazioni approfondite per farvi capire quali siano gli effetti di tutte le vostre azioni. Se volete solo installare i programmi e non vi interessano tutti i motivi ed i perché, saltate alla sezione 3 (Come Configurare il Vostro Server per lanciare Programmi Java CGI (forma abbreviata)).

2.1 Requisiti di Sistema

Questo software dovrebbe funzionare su un qualsiasi web server unix-like che ha il Java Development Kit installato. Io lo utilizzo su un sistema con *Debian Linux* che usa *apache* come demone HTTP. Se non dovesse funzionare sul vostro server, contattatemi presso `dhs@orbits.com` .

Sfortunatamente, l'interprete run-time di Java sembra essere veramente ingordo di memoria – potreste avere la necessità di installare qualche altro megabyte di RAM nel vostro server se vorrete usare estensivamente le capacità dei programmi Java CGI.

2.2 Il Software Add-On di Java CGI

Il software di cui ho scritto il codice è chiamato *Java CGI*. Potete scaricarlo dal sito ftp: *ftp://www.orbits.com/pub/software/java_cgi-0.4.tgz* . (Il numero della versione potrebbe essere cambiato).

2.3 Come decomprimere il Pacchetto Sorgente

Scegliete opportunamente una directory per decomprimere il pacchetto. (Se non avete ancora fissato un posto standard in cui installare i pacchetti software, suggerisco di utilizzare `/usr/local/src`.) Per decomprimere il file contenente la distribuzione si utilizza il seguente comando:

```
gzip -dc java_cgi-0.4.tgz | tar -xvf -
```

Questo comando creerà una directory chiamata `java_cgi-0.4`. All'interno di questa directory troverete tutti i file a cui farò riferimento nel resto del documento. (Se il numero della versione è cambiato, consultate le istruzioni contenute nella distribuzione per continuare l'installazione).

2.4 Le Directory di Installazione

A questo punto dovete decidere dove volete che i vostri programmi Java CGI risiedano. Generalmente si mettono in una directory in parallelo con la directory `cgi-bin`. Il mio server *apache* è configurato per usare `/var/web/cgi-bin` come directory `cgi-bin`, per cui uso `/var/web/javacgi` come la directory in cui inserire programmi Java. Probabilmente è meglio non mettere i vostri programmi Java CGI nelle directory esistenti nel `CLASSPATH`. Modificate il Makefile per fare in modo che esso che si adatti alla configurazion del vostro sistema. Come root lanciate `make install`. Questo compilerà i programmi Java, modificate lo script `java.cgi` per fare in modo che si adatti al vostro sistema ed installi i programmi nei posti giusti. Se invece volete avere la versione HTML di questo documento e una pagina HTML di test in aggiunta, lanciate `make all`.

2.5 Come Testare l'installazione.

Assieme alla distribuzione trovere dei documenti HTML chiamati `javacgitest.html`, `javaemailtest.html` e `javahtmltest.html`. Se avete installato tutto con `make all` come menzionato nella sezione precedente, i documenti saranno nella directory che avete specificato come WEBDIR nel `Makefile`. Se non lo avete fatto, potrete lanciare `make test` per costruirli da `javacgitest.html-dist`, `javaemailtest.html-dist` e `javahtmltest.html-dist`.

Quando siete sicuri che la vostra installazione sia andata a buon fine, potrete decidere se rimuovere `CGI_Test.class`, `Email_Test.class` e `HTML_Test.class` dalla vostra directory JAVACGI e `javacgitest.html`, `javaemailtest.html` e `javahtmltest.html` dalla vostra directory WEBDIR visto che essi mostrano alcune informazioni relative all'utente che sono normalmente disponibili solo al server.

3 Come Configurare il Vostro Server per lanciare Programmi Java CGI (forma abbreviata)

- Scaricate il pacchetto *Java CGI* dal sito ftp *ftp://www.orbits.com/pub/software/java_cgi-0.4.tgz* . (Il numero della versione potrebbe essere cambiato).
- Decomprimete il file contenente la distribuzione utilizzando il comando:

  ```
  gzip -dc java_cgi-0.4.tgz | tar -xvf -
  ```

  (Se il numero della versione è cambiato, consultate le istruzioni contenute nella distribuzione per continuare l'installazione).
- Controllate il contenuto del `Makefile` che troverete all'interno della directory `java_cgi-0.4` appena creata e modificatelo opportunamente per il vostro sistema.
- Come root, lanciate `make install`. Questo comando compilerà i programmi Java, applicherà le informazioni che avete specificato per il vostro sistema ed installerà tutti i file. Se invece desiderate consultare la documentazione in formato HTML e visionare un documento HTML di test, lanciate `make all`.
- Giunti a questo punto dovreste essere pronti per iniziare.

4 Come Eseguire un Programma Java CGI

4.1 Ostacoli quando si lanciano Programmi Java sotto il Modello CGI

Esistono due problemi principali nel lanciare un programma Java da un server Web:

4.1.1 Non si possono lanciare programmi Java come normali eseguibili.

Bisogna lanciare l'interprete run-time di Java e fornire la classe iniziale (programma da eseguire) in linea di comando. Con un form HTML, non esiste nessuna precauzione nell'inviare una linea di comando al server web.

4.1.2 Con Java non si ha un accesso generale alle variabili d'ambiente.

Ogni variabile d'ambiente necessaria al programma Java deve essere passata esplicitamente. Non esiste un metodo simile alla funzione **C** `getenv()`.

4.2 Superare i Problemi lanciando Programmi Java CGI

Per eliminare questi ostacoli, ho scritto un programma CGI di shell che fornisce le informazioni necessarie all'interprete Java.

4.2.1 Lo script java.cgi.

Questo script di shell si occupa dell'interazione tra il demone HTTP daemon e il programma Java CGI che dovete usare. Estrae il nome del programma che volete lanciare dai dati forniti al server e raccoglie tutti i dati d'ambiente in un file temporaneo. Quindi lancia l'interprete run-time di Java con il nome del file di informazioni d'ambiente e il nome del programma aggiunto alla linea di comando.

La spiegazione di come viene configurato ed installato lo script `java.cgi` si trova nella sezione 2.4 (Le Directory di Installazione).

4.2.2 Come Utilizzare il java.cgi da un form HTML.

I forms che usano i programmi Java CGI specificano delle azioni come di seguito formalizzato:

```
<form action="/cgi-bin/java.cgi/CGI_Test" method="POST">
```

Dove `/cgi-bin/` è la directory locale che avete specificato per i file binari CGI, `java.cgi` è il front-end di Java che permette di lanciare programmi Java sul Web e `CGI_Test` è un esempio del nome del programma Java da lanciare.

5 Utilizzo delle Classi Java.

Attualmente sono supportate tre classi principali definite nelle sezioni 5.1 (CGI), 5.3 (Email) e 5.5 (HTML). Sto considerando di aggiungere le classi per trattare l'input e l'output MIME-formatted rispettivamente con MIMEin & MIMEout.

Esistono anche classi di supporto e di test definite nelle sezioni 5.2 (CGI_Test), 5.4 (Email_Test) e 5.4 (HTML_Test). L'utilizzo di queste classi è stato concepito con lo scopo di testare la vostra installazione. Ciononostante possono essere utilizzate come punto di partenza per i vostri programmi in Java che fanno uso di questa libreria di classi. La classe descritta nella sezione 5.7 (Text) è la superclasse per entrambe le classi `Email` e `HTML`.

5.1 CGI

5.1.1 Sintassi della Classe

```
public class CGI
```

5.1.2 Descrizione della Classe

La classe CGI mantiene la variabile d'ambiente "CGI Information" impostata dal server web e il nome/valore inviato da un form quando si decide di selezionare l'azione **submit**. Tutte le informazioni sono memorizzate in un oggetto della classe `Properties`.

Questa classe è nel pacchetto "Orbits.net".

5.1.3 Sommario dei membri della Classe

```
CGI()         //  Costruttore.
getNames()    //  Prende la lista di nomi.
getValue()    //  Prende il valore del form specificando il nome.
```

5.1.4 Vedi Anche

`CGI_Test`.

5.1.5 CGI()

Obiettivo

Costruisce un oggetto che contiene i dati CGI disponibili.

Sintassi

```
public CGI()
```

Descrizione

Quando viene costruito un oggetto CGI, tutte le informazioni disponibili vengono assorbite ed immagazzinate localmente al nuovo oggetto.

5.1.6 getNames()

Obiettivo

Lista i nomi che sono definiti per avere valori corrispondenti.

Sintassi

```
public Enumeration getKeys ()
```

Descrizione

Fornisce la lista intera di nome per i quali sono definiti valori corrispondenti.

Restituisce

Una `Enumerazione` di tutti i nomi definiti.

5.1.7 getValue()

Obiettivo

Ricerca i **valori** associati al **nome** specificato.

Sintassi

```
public String getValue ( String name )
```

Descrizione

Questo metodo fornisce una corrispondenza tra `nomi` e `valori` inviati da un form HTML.

Parametri

name

La chiave con la quale i valori vengono selezionati.

Restituisce

Una `Stringa` che contiene il valore.

5.2 CGI_Test

Questa classe fornisce un esempio di come usare la classe `CGI` ed anche un programma di test che può essere usato per avere la conferma che il pacchetto *Java CGI* funzioni correttamente.

5.2.1 Sommario dei membri della Classe

```
main()      //  Programma main().
```

5.2.2 Vedi Anche

`CGI`.

5.2.3 main()

Obiettivo

Fornire un metodo `main()`.

Sintassi

```
public static void main( String argv[] )
```

Descrizione

Questo è il punto di ingresso del programma CGI che restituisce una lista delle coppie nomi/valori disponibili e il loro valore attuale.

Parametri

argv[]

Argomenti passati al programma dallo script `java.cgi`. Attualmente inutilizzato.

5.3 Email

5.3.1 Sintassi della Classe

```
public class Email extends Text
```

5.3.2 Descrizione della Classe

I Messaggi sono costruiti con la classe `Text` e i metodi `add*()` e i metodi specifici e-mail-specific aggiunti a da questa classe. Quando completo, il messaggio viene inviato ai suoi destinatari.

Questa classe è contenuta nel pacchetto “Orbits.net”.

5.3.3 Sommario dei membri della Classe

```
Email()      //  Costruttore.
send()       //  Invia un messaggio e-mail.
sendTo()     //  Aggiunge un destinatario al messaggio.
subject()    //  Imposta il Subject del messaggio.
```

5.3.4 Vedi Anche

`Email_Test`, `Text`.

5.3.5 Email()

Obiettivo

Costruisce un oggetto che conterrà un messaggio email.

Sintassi

```
public Email()
```

Descrizione

Costruisce un messaggio vuoto da completare da parte dei metodi Email.

Vedi Anche

`Text`.

5.3.6 send()

Obiettivo

Invia un messaggio e-mail.

Sintassi

```
public void send ()
```

Descrizione

Formatta ed invia il messaggio. Se non viene specificato un indirizzo di destinazione, non viene intrapresa alcuna azione.

5.3.7 sendTo()

Obiettivo

Aggiunge una destinazione per questo messaggio.

Sintassi

```
public String sendTo ( String address )
```

Descrizione

Aggiunge `address` alla lista delle destinazioni per questo metodo. Non esiste limite al numero di destinatari che un messaggio e-mail può avere. Sono sicuro che costruendo una lista abbastanza grande, potreste fare in modo da superare la memoria oppure eccedere la dimensione della lista di parametri che il vostro *Agente di Trasporto della Posta* è in grado di accettare.

Parametri

address
Una destinazione a cui inviare il messaggio.

5.3.8 subject()

Obiettivo

Imposta l'oggetto/subject per questo messaggio.

Sintassi

```
public void subject ( String subject )
```

Descrizione

Questo metodo imposta il testo per la riga `Subject:` di un messaggio email. Se vengono chiamati più di una volta, viene utilizzato il subject più recente.

Parametri

subject
Il testo della riga `Subject:` del messaggio.

5.4 Email_Test

Questa classe fornisce sia un esempio di come utilizzare la classe `Email` sia un programma di test per verificare che il pacchetto *Java CGI* funzioni correttamente.

5.4.1 Sommario dei membri della Classe

```
main()      //  Programma main().
```

5.4.2 Vedi Anche

`Email`.

5.4.3 main()

Obiettivo

Fornisce un metodo `main()`.

Sintassi

```
public static void main( String argv[] )
```

Descrizione

Questo è il punto di ingresso del programma CGI che restituisce una lista delle coppie nomi/valori disponibili e il loro valore attuale. Invierà anche questa lista agli indirizzi specificati nella variabile `Email`.

Parametri

argv[]
Argomenti passati al programma dallo script `java.cgi`. Attualmente inutilizzato.

5.5 HTML

5.5.1 Sintassi della Classe

```
public class HTML extends Text
```

5.5.2 Descrizione della Classe

I Messaggi vengono costruiti con i metodi `Text` class `add*()` e dai metodi specifici-HTML aggiunti da questa classe. Quando completo, il messaggio viene inviato alle sue destinazioni.

Attualmente non esiste un controllo di errore per avere la conferma che i metodi di costruzione delle liste siano usati nell'ordine corretto: il programmatore deve prestare attenzione a non violare la sintassi HTML.

Questa classe è nel pacchetto "Orbits.net".

5.5.3 Sommario dei membri della Classe

```
HTML()                 //  Costruttore.
author()               //  Imposta il nome dell'autore del documento.
definitionList()       //  Avvia una definition list.
definitionListTerm()   //  Aggiunge un termine ad una definition list.
endList()              //  Termina una lista.
listItem()             //  Aggiunge una entrata ad una lista.
send()                 //  Invia un messaggio HTML.
title()                //  Imposta il testo per il titolo del documento.
```

5.5.4 Vedi Anche

`HTML_Test`, `Text`.

5.5.5 HTML()

Obiettivo

Costruisce un oggetto che contiene un messaggio HTML.

Sintassi

```
public HTML()
```

Descrizione

Costruisce un messaggio vuoto da completare con i metodi HTML.

Vedi Anche

`Text`.

5.5.6 author()

Obiettivo

Imposta il nome dell'autore del documento.

Sintassi

```
public void author ( String author )
```

Descrizione

Imposta il nome dell'autore del documento in `author`.

Parametri

author

Il testo da usare come autore di questo messaggio.

Vedi Anche

`title()`.

5.5.7 definitionList()

Obiettivo

Avvia una definition list.

Sintassi

```
public void definitionList ()
```

Descrizione

Avvia una definition list. Una *definition list* è una particolare lista per la quale ogni elemento di ingresso della lista è *term* seguito dalla definizione *text* per quel termine. L'avvio di una definition list dovrebbe essere seguito dalla creazione di almeno un termine/testo e una chiamata al metodo `endList()`. *Da notare che, attualmente, le liste non possono essere nidificate*

Vedi Anche

`definitionListTerm()`, `endList()`, `listItem()`.

5.5.8 definitionListTerm()

Obiettivo

Aggiunge un termine alla definition list.

Sintassi

```
public void definitionListTerm ()
```

Descrizione

Aggiunge un termine alla definition list. Il testo dell'elemento lista corrente deve essere appeso al messaggio dopo che questo metodo viene chiamato e prima di un corrispondente metodo `listItem` venga chiamato.

Vedi Anche

`definitionList()`, `listItem()`.

5.5.9 endList()

Obiettivo

Termina una lista.

Sintassi

```
public void endList ()
```

Descrizione

Termina una lista. Questo metodo elimina una lista. *Da notare che al momento attuale, le liste non possono essere nidificate.*

Vedi Anche

`definitionList()`.

5.5.10 listItem()

Obiettivo

Aggiunge una voce nella lista.

Sintassi

```
public void listItem ()


public void listItem ( String item )


public boolean listItem ( String term, String item )
```

Descrizione

Aggiunge una voce alla lista. Se viene usato il primo form, il testo dell'elemento lista corrente deve essere accodato al messaggio dopo che questo metodo viene chiamato e prima che venga chiamata ogni altra lista di metodi. Nel secondo e terzo form, il testo `item` è specificato come un parametro al metodo invece di (o in aggiunta) essere accodato al messaggio. Il terzo form è specifico per le definition lists e fornisce sia i termini che una definizione dell'ingresso della lista.

Parametri

item

Il testo di questo ingresso di lista.

term

Il testo di questa parte del termine di ingresso della definition list.

Vedi Anche

`definitionList()`, `definitionListTerm()`, `endList()`.

5.5.11 send()

Obiettivo

Invia un messaggio HTML.

Sintassi

```
public void send ()
```

Descrizione

Invia un messaggio HTML.

5.5.12 title()

Obiettivo

Imposta il testo per il titolo del documento.

Sintassi

```
public void title ( String title )
```

Descrizione

Imposta il testo per il titolo del documento.

Parametri

title

Il testo del titolo del documento.

Vedi Anche

`author()`.

5.6 HTML_Test

Questa classe fornisce sia un esempio di come usare la classe `HTML` sia un programma di test che può essere usato per avere la conferma che il pacchetto *Java CGI* funzioni correttamente.

5.6.1 Sommario dei membri della Classe

```
main()      //  Programma main().
```

5.6.2 Vedi Anche

`HTML`.

5.6.3 main()

Obiettivo

Fornire un metodo `main()`.

Sintassi

```
public static void main( String argv[] )
```

Descrizione

Questo è il punto di ingresso del programma CGI che restituisce una lista delle coppie nomi/valori disponibili e il loro valore attuale.

Parametri

argv[]

Argomenti passati al programma dallo `java.cgi`. Attualmente inutilizzato.

5.7 Text

5.7.1 Sintassi delle Classi

```
public abstract class Text
```

5.7.2 Descrizione delle Classi

Questa classe è la superclasse delle classi `Email` e `HTML`. I Messaggi sono costruiti con i metodi di questa classe poi completati e formattati con i metodi delle sottoclassi.

Questa classe è contenuta nel pacchetto "Orbits.text".

5.7.3 Member Summary

```
Text()            //  Costruttore.
add()             //  Aggiunge testo a questo oggetto.
addLineBreak()    //  Aggiunge un terminatore di riga.
addParagraph()    //  Aggiunge un terminatore di paragrafo.
```

5.7.4 Vedi Anche

`Email`, `HTML`.

5.7.5 add()

Obiettivo

Aggiunge del testo a questo elemento.

Sintassi

```
public void add ( char addition )
```

```
public void add ( String addition )
```

```
public void add ( StringBuffer addition )
```

Descrizione

Aggiunge `addition` ai contenuti di questo elemento di testo.

Parameter

addition

Testo da aggiungere all'elemento di testo.

Vedi Anche

`addLineBreak()`, `addParagraph()`.

5.7.6 addLineBreak()

Obiettivo

Forza una terminazione di riga nel punto esatto del testo.

Sintassi

```
public void addLineBreak ()
```

Descrizione

Aggiunge una terminazione di riga al testo nel punto attuale.

Vedi Anche

`add()`, `addParagraph()`.

5.7.7 addParagraph()

Obiettivo

Comincia un nuovo paragrafo.

Sintassi

```
public void add ()
```

Descrizione

Cominicia un nuovo paragrafo in questo punto del testo.

Vedi Anche

`add()`, `addLineBreak()`.

6 Piani Futuri

- Aggiungere alla classe Email:

  Email(int capacity)
  Usato quando conosciamo quanto spazio necessita al messaggio per essere allocato.

  sendTo(String [] address)
  Aggiunge una lista di destinazioni primarie al messaggio e-mail.

  sendCc(String address)
  Aggiunge una destinazione Carbon-Copy al messaggio e-mail.

  sendCc(String [] address)
  Aggiunge una lista di destinazioni Carbon-Copy al messaggio e-mail.

  sendBcc(String address)
  Aggiunge una destinazione Blind Carbon-Copy al messaggio e-mail.

  sendBcc(String [] address)
  Aggiunge una lista di destinazioni Blind Carbon-Copy al messaggio e-mail.

- Aggiungere alla classe HTML:

  HTML(int capacity)
  Usato quando conosciamo quanto spazio necessita al messaggio per essere allocato.

  public void unorderedList()
  Avvia una lista non-ordinata.

  public void orderedList()
  Avvia una lista ordinata.

  public void directoryList()
  Avvia una lista di directory.

  public void menuList()
  Avvia una lista di menu.

  void anchor(String anchorName)
  Specifica un anchor.

  void link(String url, String text)
  Specifica un link.

  void applet(String url, String altText)
  Specifica un link ad un applet.

- Permettere la nidificazione delle liste HTML.
- Aggiungere un codice a controllo di errore per rinforzare l'ordinamento corretto dei codici di formattazione delle liste HTML.
- Il posizionamento del file dei dati d'ambiente deve essere configurabile tramite la modifica del `Makefile`.
- Eliminare le coppie spurie di nomi/valori vuoti che appaiono nella lista quando si tratta il metodo di trasferimento di dati GET.
- Considerare di fare in modo che CGI implementi l'interfaccia java.util.Enumeration per fornire successivamente i nomi delle variabili.

- Aggiungere una classe `Test`, con la quale utilizzare ogni metodo di questo pacchetto.
- Documentare come `CGI_Test`, `Email_Test` e `HTML_Test` facciano affidamento l'uno sull'altro per fornire tests incrementali per obiettivi di debugging.
- Documentare come Test utilizzi ogni caratteristica disponibile in questo pacchetto.

7 Varianti

7.1 Varianti da 0.3 a 0.4

- Buttata fuori la classe HTML per fornire una minima funzionalità
- Scritta la classe HTML_Test e javahtmltest.html-dist.
- Aggiunti i metodi HTML per trattare con la definition list.

7.2 Varianti da 0.2 a 0.3

- Aggiunte le classi Text e Email. Viene aggiunto anche l'HTML, ma è solo un inizio.
- Raccolte le varie classi in pacchetti. Le classi principali sono in `Orbits.net.*`, la classe di supporto `Text` è in `Orbits.text.Text`.
- Cambiato `CGItest` in `CGI_Test`.
- Aggiunta la classe `Email_Test`.

7.3 Varianti da 0.1 a 0.2

- Le variabili d'ambiente vengono messe in un file temporaneo invece di essere stipate in linea di comando nell'interprete Java. Le classi `CGI` e `java.cgi` sono state modificate.
- Il documento `javacgitest.html` diventa parte della distribuzione.
- I file di testo che sono modificati da `make` durante l'installazione vengono forniti con nomi che terminano con *-dist*.