La guida al deploy serverless di file statici in locale con http-server, in remoto con surge e per dispositivi reali con ngrok + ottimizzare l'uso di programmi da linea di comando con cmder

In questa puntata vedremo gli strumenti relativi al deploy di file statici. I casi d'uso comprendono:

- server locale per i file statici nella cartella corrente
- fare ovunque una demo di un progetto locale
- rendere disponibile un server locale per testare un progetto su un dispositivo mobile
- rendere disponibili delle API all'esterno quando si usano sistemi di test come crossbrowsertesting
- deploy con un solo comando su un dominio esterno

Autore

Ciao, sono Salvatore e trovi tutto su di me sul sito https://linkedin.com/in/salvatoreromeo. Ho scritto questo mini libro principalmente perché mi piace scrivere e condividere esperienza.

Nella vita mi occupo di sviluppo software fin da quando avevo 12 anni. Oggi è il mio lavoro principale. Insegno Ingegneria del Software all'Università degli Studi di Perugia e un corso di 3 giorni su Angular per le aziende insieme al Codemotion. Il corso è da poco stato trasformato in un libro che puoi trovare su Amazon.

Se vuoi maggiori informazioni su di me contattami pure tramite linkedin o tramite il sito https://devexp.io. Lì potrai anche trovare articoli sul front-end, altri libri del gruppo devexp.io e i codici sorgenti pubblicati su Github.

Sommario

Introduzione

Questo articolo fa parte di una serie di articoli dedicati ai migliori strumenti per velocizzare il processo di sviluppo front-end.
Abbiamo già visto due tool potentissimi: Postman per verificare e gestire richieste HTTP e AutoHotkey per avere un assistente nello sviluppo front-end attraverso l'automazione di task noiosi e ripetitivi.

In questa sezione ci concentreremo su alcuni tool da linea di comando. Si tratta di strumenti che utilizzo quotidianamente per svolgere determinate funzioni in maniera rapida e semplice.

Vedremo di seguito *http-server*, per creare un server nella cartella corrente e servire i file statici presenti in questa cartella; *ngrok* per creare un tunnel e rendere disponibili anche all'esterno i nostri server locali; *surge* per fare il deploy su web di un progetto front-end eseguendo un unico comando.

Nell'ultima parte di questa puntata vedremo invece uno strumento per coloro che usano Windows. Si chiama *cmder* e possiamo usarlo al posto del prompt di Windows creando un ambiente a linea di comando moderno, con funzioni extra rispetto al classico prompt, tra cui multi-tab, autocompletamento e storico dei comandi.

Gli altri tool che andremo a conoscere nelle prossime puntate

I tool che andremo a vedere nelle varie puntate sono:

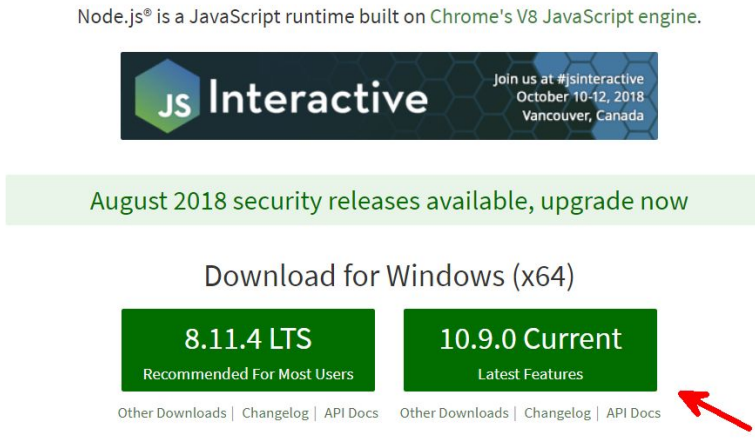
- Postman per lavorare con le richieste al server
- AutoHotkey per automatizzare qualsiasi task ripetitivo
- **Cmder** per ottimizzare l'uso della linea di comando
- **Http-server** per creare un server velocemente nella cartella corrente
- **Surge** per pubblicare la cartella corrente online con un comando
- **Ngrok** per rendere disponibile un server locale all'esterno
- **ColorPicker** per catturare i colori
- **Chrome** con i suoi strumenti base e avanzati
- **JSON Viewer** per visualizzare oggetti JSON

Esigenza

Creare un server non è mai un'operazione banalissima. Se sei uno sviluppatore con anni di esperienza nel back-end, sicuramente sai quanto tempo ci vuole per configurare un server Apache o Java. Quando sviluppiamo per il front-end, spesso abbiamo bisogno di avviare velocemente un server per pochi minuti, giusto il tempo di vedere i risultati di un deploy.
Fortunatamente oggi esistono degli strumenti con Node che ci permettono di ottenere un server digitando un comando da linea di comando.

Premessa

Per utilizzare i tool che vedremo di seguito è necessario installare Node sul proprio PC. Basta recarsi sul sito https://nodejs.org/en/ e scaricare la versione “current” per il nostro sistema operativo:



Un server per i file statici della cartella corrente: http-server

Anche se per back-end serverless si intende un server che fornisce delle API, qui useremo un server che non va configurato, non fornisce API extra, ma ci velocizza di molto il serving di file statici.
Lo scenario è il seguente: abbiamo dei file statici HTML in una cartella e vogliamo vedere il risultato sul browser come se fossero serviti da un server web.
Trascinando direttamente il file index.html su chrome il risultato non è infatti lo stesso perché Chrome vede il file come un file locale di sistema e disabilita alcune funzioni.

Possiamo allora usare *http-server*, un server velocissimo che si avvia con un solo comando.

Installazione

Per installare http-server, da linea di comando eseguiamo il comando

npm install -g http-server

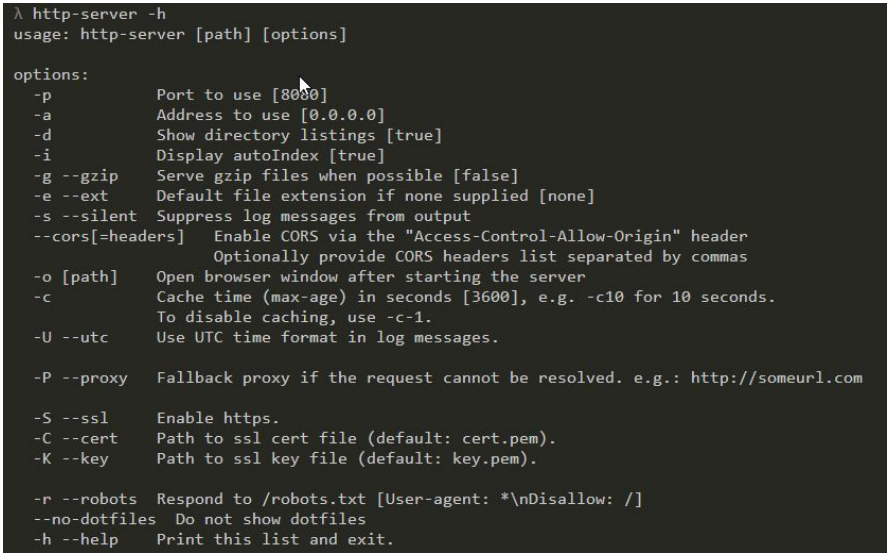
Usiamo il parametro *-g* per installare il programma globalmente nel sistema.
Dopo aver eseguito questo comando avremo disponibile il programma *http-server* su tutto il sistema.

http-server: il comando più utile

Dopo aver installato *http-server* possiamo esplorare tutte le funzioni con il comando

http-server -h

Il risultato sarà il seguente:



Le funzioni sono molte, ma la funzione principale che useremo più spesso è quella di servire i file statici nella cartella corrente. Basta eseguire il comando

http-server

e saranno serviti i file nella cartella corrente del prompt.

Eventualmente è possibile specificare la porta

http-server --p 4001

Ovviamente, per vedere i file serviti, basta andare sul browser all'indirizzo mostrato nel prompt con la porta specificata.

Un proxy per i nostri server locali: ngrok

Nella sezione precedente abbiamo visto come creare un server nella cartella corrente. Il server è però disponibile solo sul nostro PC. E se volessimo rendere disponibile questo server locale all'esterno, magari per farlo vedere ad un nostro collega?
Ci viene in aiuto *ngrok*.
ngrok è una sorta di proxy che crea un tunnel di un qualsiasi server locale verso l'esterno. I casi d'uso sono diversi, tra cui:

- fare una demo di un progetto locale
- rendere disponibile un server locale per testare un progetto su un dispositivo mobile reale
- rendere disponibili delle API all'esterno quando si usano sistemi di test come crossbrowsertesting

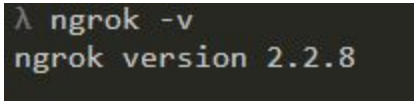
Il suo funzionamento è semplicissimo.

Installazione

Come nel caso precedente, l'installazione richiede un solo comando

npm install -g ngrok

Se tutto è andato a buon fine, stampiamo a video la versione corrente:

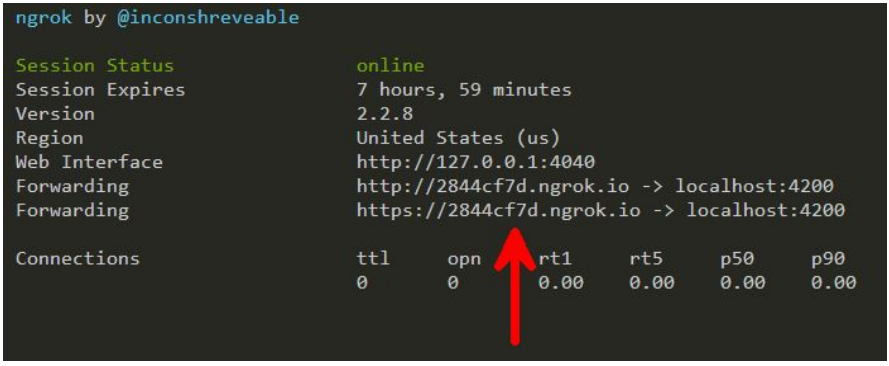


ngrok: il comando principale

ngrok ha un comando semplice per fare il tunnel di un server http:

ngrok http 4200

Dove 4200 è la porta del server che vogliamo rendere disponibile all'esterno.
Dopo aver eseguito il comando vedremo una serie di informazioni:



La più importante è quella indicata con la freccia in figura: l'URL del server esterno.
Quando nel browser inseriremo l'indirizzo http://2844cf7d.ngrok.*io* ci risponderà il server del nostro PC.
Questo indirizzo funziona anche da un PC esterno, ma attenzione: se stoppiamo il tunnel (con CTRL+C) e lo riavviamo l'indirizzo cambia.
Per usare un indirizzo custom dobbiamo registrarci.
Ovviamente non aspettiamoci grandi velocità, poiché:

1. la connessione locale non avrà performance da server di produzione

2. Il tunnel potrebbe trovarsi all'estero

In ogni caso per i casi d'uso visti sopra è l'ideale.

Deploy di file statici con un unico comando: surge

L'ultimo tool che vedremo in questa puntata ci serve per il seguente caso d'uso: fare il deploy di file statici su un dominio esterno.
La combinazione *http-server* + *ngrok* è sicuramente utile in molti scenari, ma se vogliamo rendere disponibili versioni parziali o temporanee di un'applicazione ad un cliente o ad un collega, è meglio fare il deploy su un server performante e il cui dominio è mnemonico. *surge* ci consente di coprire entrambe queste esigenze.

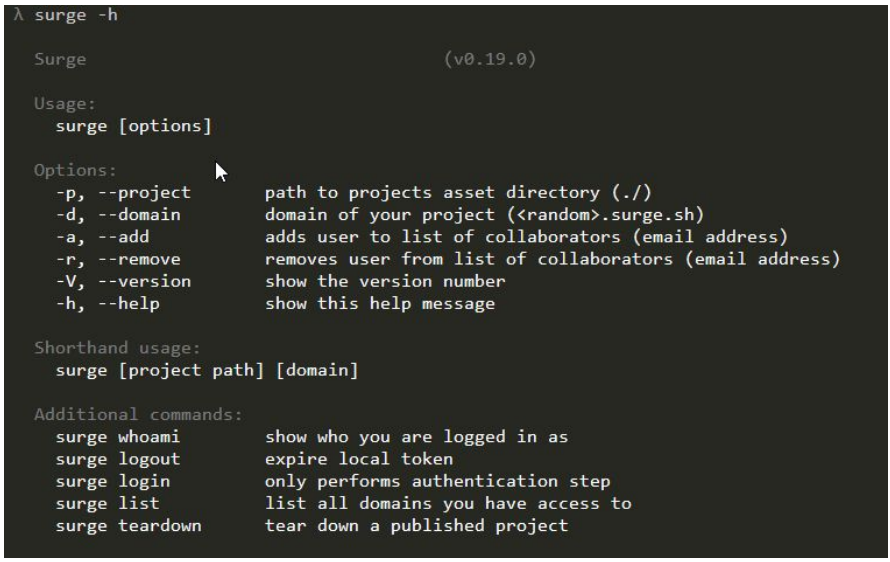
L'installazione avviene come nel caso precedente

npm install -g surge

ed eseguendo il comando

surge -h

avremo una panoramica delle funzioni:



Al primo avvio ci verrà chiesto di registrarci, sempre da linea di comando, specificando un'email, ma non è richiesta conferma ;-)

Il comando principale ci permette di fare l'upload dei file nella cartella corrente in un dominio a nostra scelta, che finisce con *.surge.sh*

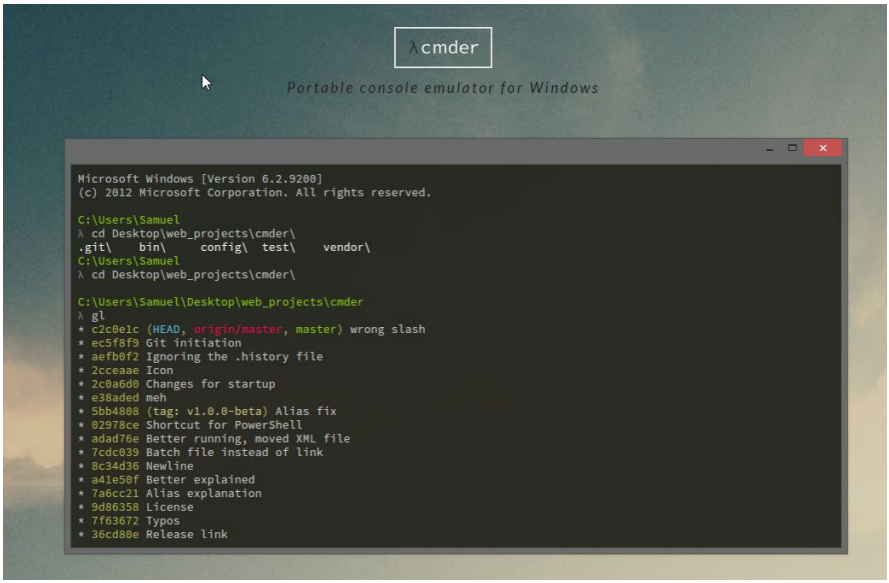
surge --domain mio-dominio.surge.sh

Possiamo quindi accedere al nostro progetto digitando l'indirizzo completo su qualsiasi browser:
http://mio-dominio.surge.sh/

Un prompt moderno e avanzato per eseguire comandi da linea di comando: cmder

Concludiamo questa puntata con un tool potentissimo per chi usa spesso la linea di comando. E per uno sviluppatore front-end la linea di comando è il pane quotidiano.
Mentre sotto linux ci sono molti strumenti che agevolano l'uso del terminale, sotto windows di default c'è solo *cmd* e ha molti limiti.

Per essere efficienti da linea di comando suggerisco di usare [cmder]



Si installa come un normale programma, ma esiste anche la versione portable.

I casi d'uso e le funzioni principali sono:

- sostituire completamente il prompt
- avere tab nel terminale (CTRL+T per un nuovo tab)
- CTRL+V o tasto destro del mouse per incollare nel prompt
- semplice selezione di un testo per copiare nella clipboard
- supportare comandi linux che altrimenti non ci sarebbero sotto windows, come *ls* per stampare la lista dei file
- avere una history molto lunga accessibile con la freccia in alto
- autocompletamento
- ricerca di comandi passati con CTRL+R

Con *cmder* avrete a disposizione anche *git* e *ssh* di default.

Ho reso disponibile inoltre un menu custom che usa AutoHotkey per avere comandi extra quando si preme con il tasto centrale, come andare nelle proprie cartelle preferite o eseguire comandi:

Lo trovate qui:

https://github.com/devexp-io/codici-articoli/tree/master/cmder

e per utilizzarlo serve AutoHotkey.

Conclusioni

Abbiamo visto in questa puntata diversi tool a linea di comando legati al deploy. Nell'ecosistema *npm* ne esistono in realtà molti altri e tutti si installano facilmente con il comando *npm install*.
La prossima settimana vedremo invece Chrome con molti trucchi che probabilmente non conosci ancora e altri tool legati sempre al front-end.
Buon lavoro e come sempre per chiarimenti o feedback non esitare a contattarmi.