Node.js: estrarre i link interni da una pagina remota

In questo tutorial vedremo come estrarre i link da una pagina HTML di un sito con Node.js.

Useremo i moduli NPM cheerio e got rispettivamente per il parsing del codice HTML e per effettuare le richieste HTTP. Il package got verrà usato nella versione 11, in quanto le versioni successive non utilizzano più il tradizionale modo di includere i moduli e per tale motivo non sono compatibili con tutte le installazioni di Node.js.

Si tratta sostanzialmente di individuare tutti gli elementi HTML a il cui attributo `href` faccia riferimento a pagine interne al sito scelto come punto di partenza. Dobbiamo anche fare in modo che l'array restituito non contenga URL duplicati e per farlo ricorreremo all'operatore spread e all'oggetto `Set` trattandosi di valori primitivi (stringhe).

La nostra implementazione potrebbe essere la seguente:

```
'use strict';

const got = require('got');
const cheerio = require('cheerio');

class Crawler {
    constructor(url) {
        this.url = url.endsWith( '/') ? url : url +
'/';
        this.links = [];
    }
```

```
    async fetch() {
        try {
            await this.getLinks();
            this.links = [...new Set(this.links)];
        } catch(err) {
            throw err;
        }
    }

    async getLinks() {
        try {
            const response = await this.getURL();
            const $ = cheerio.load(response);
            const links = $('a[href^="' + this.url +
'"]');
            const hrefs = this.parseLinks($, links);

            if(hrefs.length > 0) {
                hrefs.forEach(href => {
                    this.links.push(href);
                });
            }
        } catch(err) {
            throw new Error('Unable to get links.');
        }
    }

    async getURL() {
        try {
            const response = await got(this.url);
            return response.body;
        } catch(err) {
            throw err;
        }
```

```
    }

    parseLinks( $dom, $links ) {
        const hrefs = [];

        $links.each(function() {
            hrefs.push($dom(this).attr('href'));
        });

        return hrefs;
    }
}

(async () => {
    try {
        const crawler = new
Crawler('https://gabrieleromanato.com');
        await crawler.fetch();

        console.log(crawler.links);

    } catch(err) {
        console.log(err);
    }
})();
```

L'aspetto importante da ricordare è che la gestione degli errori non può mancare. Le versioni più recenti di Node terminano immediatamente il processo in esecuzione se un'eccezione nelle Promise non viene gestita.