// Definitions by: Carlos Ballesteros Velasco <https://github.com/soywiz>

// Leon Yu <https://github.com/leonyu>

// BendingBender <https://github.com/BendingBender>

// Maple Miao <https://github.com/mapleeit>

/// <reference types="node" />

import \* as stream from 'stream';

import \* as http from 'http';

export = FormData;

// Extracted because @types/node doesn't export interfaces.

interface ReadableOptions {

highWaterMark?: number;

encoding?: string;

objectMode?: boolean;

read?(this: stream.Readable, size: number): void;

destroy?(this: stream.Readable, error: Error | null, callback: (error: Error | null) => void): void;

autoDestroy?: boolean;

}

interface Options extends ReadableOptions {

writable?: boolean;

readable?: boolean;

dataSize?: number;

maxDataSize?: number;

pauseStreams?: boolean;

}

declare class FormData extends stream.Readable {

constructor(options?: Options);

append(key: string, value: any, options?: FormData.AppendOptions | string): void;

getHeaders(userHeaders?: FormData.Headers): FormData.Headers;

submit(

params: string | FormData.SubmitOptions,

callback?: (error: Error | null, response: http.IncomingMessage) => void

): http.ClientRequest;

getBuffer(): Buffer;

getBoundary(): string;

getLength(callback: (err: Error | null, length: number) => void): void;

getLengthSync(): number;

hasKnownLength(): boolean;

}

declare namespace FormData {

interface Headers {

[key: string]: any;

}

interface AppendOptions {

header?: string | Headers;

knownLength?: number;

filename?: string;

filepath?: string;

contentType?: string;

}

interface SubmitOptions extends http.RequestOptions {

protocol?: 'https:' | 'http:';

}

}