declare module 'source-map-js' {

export interface StartOfSourceMap {

file?: string;

sourceRoot?: string;

}

export interface RawSourceMap extends StartOfSourceMap {

version: string;

sources: string[];

names: string[];

sourcesContent?: string[];

mappings: string;

}

export interface Position {

line: number;

column: number;

}

export interface LineRange extends Position {

lastColumn: number;

}

export interface FindPosition extends Position {

// SourceMapConsumer.GREATEST\_LOWER\_BOUND or SourceMapConsumer.LEAST\_UPPER\_BOUND

bias?: number;

}

export interface SourceFindPosition extends FindPosition {

source: string;

}

export interface MappedPosition extends Position {

source: string;

name?: string;

}

export interface MappingItem {

source: string;

generatedLine: number;

generatedColumn: number;

originalLine: number;

originalColumn: number;

name: string;

}

export class SourceMapConsumer {

static GENERATED\_ORDER: number;

static ORIGINAL\_ORDER: number;

static GREATEST\_LOWER\_BOUND: number;

static LEAST\_UPPER\_BOUND: number;

constructor(rawSourceMap: RawSourceMap);

computeColumnSpans(): void;

originalPositionFor(generatedPosition: FindPosition): MappedPosition;

generatedPositionFor(originalPosition: SourceFindPosition): LineRange;

allGeneratedPositionsFor(originalPosition: MappedPosition): Position[];

hasContentsOfAllSources(): boolean;

sourceContentFor(source: string, returnNullOnMissing?: boolean): string;

eachMapping(callback: (mapping: MappingItem) => void, context?: any, order?: number): void;

}

export interface Mapping {

generated: Position;

original: Position;

source: string;

name?: string;

}

export class SourceMapGenerator {

constructor(startOfSourceMap?: StartOfSourceMap);

static fromSourceMap(sourceMapConsumer: SourceMapConsumer): SourceMapGenerator;

addMapping(mapping: Mapping): void;

setSourceContent(sourceFile: string, sourceContent: string): void;

applySourceMap(sourceMapConsumer: SourceMapConsumer, sourceFile?: string, sourceMapPath?: string): void;

toString(): string;

}

export interface CodeWithSourceMap {

code: string;

map: SourceMapGenerator;

}

export class SourceNode {

constructor();

constructor(line: number, column: number, source: string);

constructor(line: number, column: number, source: string, chunk?: string, name?: string);

static fromStringWithSourceMap(code: string, sourceMapConsumer: SourceMapConsumer, relativePath?: string): SourceNode;

add(chunk: string): void;

prepend(chunk: string): void;

setSourceContent(sourceFile: string, sourceContent: string): void;

walk(fn: (chunk: string, mapping: MappedPosition) => void): void;

walkSourceContents(fn: (file: string, content: string) => void): void;

join(sep: string): SourceNode;

replaceRight(pattern: string, replacement: string): SourceNode;

toString(): string;

toStringWithSourceMap(startOfSourceMap?: StartOfSourceMap): CodeWithSourceMap;

}

}

declare module 'source-map-js/lib/source-map-generator' {

import { SourceMapGenerator } from 'source-map-js'

export { SourceMapGenerator }

}

declare module 'source-map-js/lib/source-map-consumer' {

import { SourceMapConsumer } from 'source-map-js'

export { SourceMapConsumer }

}

declare module 'source-map-js/lib/source-node' {

import { SourceNode } from 'source-map-js'

export { SourceNode }

}