import {LegacyException} from './exception';

import {LegacyOptions} from './options';

/\*\*

\* The object returned by [[render]] and [[renderSync]] after a successful

\* compilation.

\*

\* @category Legacy

\* @deprecated This is only used by the legacy [[render]] and [[renderSync]]

\* APIs. Use [[compile]], [[compileString]], [[compileAsync]], and

\* [[compileStringAsync]] instead.

\*/

export interface LegacyResult {

/\*\*

\* The compiled CSS. This can be converted to a string by calling

\* [Buffer.toString](https://nodejs.org/api/buffer.html#buffer\_buf\_tostring\_encoding\_start\_end).

\*

\* @example

\*

\* ```js

\* const result = sass.renderSync({file: "style.scss"});

\*

\* console.log(result.css.toString());

\* ```

\*/

css: Buffer;

/\*\*

\* The source map that maps the compiled CSS to the source files from which it

\* was generated. This can be converted to a string by calling

\* [Buffer.toString](https://nodejs.org/api/buffer.html#buffer\_buf\_tostring\_encoding\_start\_end).

\*

\* This is `undefined` unless either

\*

\* \* [[LegacySharedOptions.sourceMap]] is a string; or

\* \* [[LegacySharedOptions.sourceMap]] is `true` and

\* [[LegacySharedOptions.outFile]] is set.

\*

\* The source map uses absolute [`file:`

\* URLs](https://en.wikipedia.org/wiki/File\_URI\_scheme) to link to the Sass

\* source files, except if the source file comes from

\* [[LegacyStringOptions.data]] in which case it lists its URL as `"stdin"`.

\*

\* @example

\*

\* ```js

\* const result = sass.renderSync({

\* file: "style.scss",

\* sourceMap: true,

\* outFile: "style.css"

\* })

\*

\* console.log(result.map.toString());

\* ```

\*/

map?: Buffer;

/\*\* Additional information about the compilation. \*/

stats: {

/\*\*

\* The absolute path of [[LegacyFileOptions.file]] or

\* [[LegacyStringOptions.file]], or `"data"` if [[LegacyStringOptions.file]]

\* wasn't set.

\*/

entry: string;

/\*\*

\* The number of milliseconds between 1 January 1970 at 00:00:00 UTC and the

\* time at which Sass compilation began.

\*/

start: number;

/\*\*

\* The number of milliseconds between 1 January 1970 at 00:00:00 UTC and the

\* time at which Sass compilation ended.

\*/

end: number;

/\*\*

\* The number of milliseconds it took to compile the Sass file. This is

\* always equal to `start` minus `end`.

\*/

duration: number;

/\*\*

\* An array of the absolute paths of all Sass files loaded during

\* compilation. If a stylesheet was loaded from a [[LegacyImporter]] that

\* returned the stylesheet’s contents, the raw string of the `@use` or

\* `@import` that loaded that stylesheet included in this array.

\*/

includedFiles: string[];

};

}

/\*\*

\* This function synchronously compiles a Sass file to CSS. If it succeeds, it

\* returns the result, and if it fails it throws an error.

\*

\* @example

\*

\* ```js

\* const sass = require('sass'); // or require('node-sass');

\*

\* const result = sass.renderSync({file: "style.scss"});

\* // ...

\* ```

\*

\* @category Legacy

\* @deprecated Use [[compile]] or [[compileString]] instead.

\*/

export function renderSync(options: LegacyOptions<'sync'>): LegacyResult;

/\*\*

\* This function asynchronously compiles a Sass file to CSS, and calls

\* `callback` with a [[LegacyResult]] if compilation succeeds or

\* [[LegacyException]] if it fails.

\*

\* \*\*Heads up!\*\* When using Dart Sass, \*\*[[renderSync]] is almost twice as fast

\* as [[render]]\*\* by default, due to the overhead of making the entire

\* evaluation process asynchronous.

\*

\* ```js

\* const sass = require('sass'); // or require('node-sass');

\*

\* sass.render({

\* file: "style.scss"

\* }, function(err, result) {

\* // ...

\* });

\* ```

\*

\* @category Legacy

\* @deprecated Use [[compileAsync]] or [[compileStringAsync]] instead.

\*/

export function render(

options: LegacyOptions<'async'>,

callback: (exception?: LegacyException, result?: LegacyResult) => void

): void;