import {LegacyPluginThis} from './plugin\_this';

/\*\*

\* The value of `this` in the context of a [[LegacyImporter]] function.

\*

\* @category Legacy

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

\* APIs. Use [[Importer]] with [[compile]], [[compileString]], [[compileAsync]],

\* and [[compileStringAsync]] instead.

\*/

interface LegacyImporterThis extends LegacyPluginThis {

/\*\*

\* Whether the importer is being invoked because of a Sass `@import` rule, as

\* opposed to a `@use` or `@forward` rule.

\*

\* This should \*only\* be used for determining whether or not to load

\* [import-only files](https://sass-lang.com/documentation/at-rules/import#import-only-files).

\*

\* @compatibility dart: "1.33.0", node: false

\*/

fromImport: boolean;

}

/\*\*

\* The result of running a [[LegacyImporter]]. It must be one of the following

\* types:

\*

\* \* An object with the key `contents` whose value is the contents of a stylesheet

\* (in SCSS syntax). This causes Sass to load that stylesheet’s contents.

\*

\* \* An object with the key `file` whose value is a path on disk. This causes Sass

\* to load that file as though it had been imported directly.

\*

\* \* `null`, which indicates that it doesn’t recognize the URL and another

\* importer should be tried instead.

\*

\* \* An [Error](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\_Objects/Error)

\* object, indicating that importing failed.

\*

\* @category Legacy

\* @deprecated This only works with the legacy [[render]] and [[renderSync]]

\* APIs. Use [[ImporterResult]] with [[compile]], [[compileString]],

\* [[compileAsync]], and [[compileStringAsync]] instead.

\*/

export type LegacyImporterResult =

| {file: string}

| {contents: string}

| Error

| null;

/\*\*

\* A synchronous callback that implements custom Sass loading logic for

\* [`@import` rules](https://sass-lang.com/documentation/at-rules/import) and

\* [`@use` rules](https://sass-lang.com/documentation/at-rules/use). This can be

\* passed to [[LegacySharedOptions.importer]] for either [[render]] or

\* [[renderSync]].

\*

\* See [[LegacySharedOptions.importer]] for more detailed documentation.

\*

\* ```js

\* sass.renderSync({

\* file: "style.scss",

\* importer: [

\* function(url, prev) {

\* if (url != "big-headers") return null;

\*

\* return {

\* contents: 'h1 { font-size: 40px; }'

\* };

\* }

\* ]

\* });

\* ```

\*

\* @param url - The `@use` or `@import` rule’s URL as a string, exactly as it

\* appears in the stylesheet.

\*

\* @param prev - A string identifying the stylesheet that contained the `@use`

\* or `@import`. This string’s format depends on how that stylesheet was loaded:

\*

\* \* If the stylesheet was loaded from the filesystem, it’s the absolute path of

\* its file.

\* \* If the stylesheet was loaded from an importer that returned its contents,

\* it’s the URL of the `@use` or `@import` rule that loaded it.

\* \* If the stylesheet came from the data option, it’s the string "stdin".

\*

\* @category Legacy

\* @deprecated This only works with the legacy [[render]] and [[renderSync]]

\* APIs. Use [[Importer]] with [[compile]], [[compileString]], [[compileAsync]],

\* and [[compileStringAsync]] instead.

\*/

type LegacySyncImporter = (

this: LegacyImporterThis,

url: string,

prev: string

) => LegacyImporterResult;

/\*\*

\* An asynchronous callback that implements custom Sass loading logic for

\* [`@import` rules](https://sass-lang.com/documentation/at-rules/import) and

\* [`@use` rules](https://sass-lang.com/documentation/at-rules/use). This can be

\* passed to [[LegacySharedOptions.importer]] for either [[render]] or

\* [[renderSync]].

\*

\* An asynchronous importer must return `undefined`, and then call `done` with

\* the result of its [[LegacyImporterResult]] once it's done running.

\*

\* See [[LegacySharedOptions.importer]] for more detailed documentation.

\*

\* ```js

\* sass.render({

\* file: "style.scss",

\* importer: [

\* function(url, prev, done) {

\* if (url != "big-headers") done(null);

\*

\* done({

\* contents: 'h1 { font-size: 40px; }'

\* });

\* }

\* ]

\* });

\* ```

\*

\* @param url - The `@use` or `@import` rule’s URL as a string, exactly as it

\* appears in the stylesheet.

\*

\* @param prev - A string identifying the stylesheet that contained the `@use`

\* or `@import`. This string’s format depends on how that stylesheet was loaded:

\*

\* \* If the stylesheet was loaded from the filesystem, it’s the absolute path of

\* its file.

\* \* If the stylesheet was loaded from an importer that returned its contents,

\* it’s the URL of the `@use` or `@import` rule that loaded it.

\* \* If the stylesheet came from the data option, it’s the string "stdin".

\*

\* @param done - The callback to call once the importer has finished running.

\*

\* @category Legacy

\* @deprecated This only works with the legacy [[render]] and [[renderSync]]

\* APIs. Use [[Importer]] with [[compile]], [[compileString]], [[compileAsync]],

\* and [[compileStringAsync]] instead.

\*/

type LegacyAsyncImporter = (

this: LegacyImporterThis,

url: string,

prev: string,

done: (result: LegacyImporterResult) => void

) => void;

/\*\*

\* A callback that implements custom Sass loading logic for [`@import`

\* rules](https://sass-lang.com/documentation/at-rules/import) and [`@use`

\* rules](https://sass-lang.com/documentation/at-rules/use). For [[renderSync]],

\* this must be a [[LegacySyncImporter]] which returns its result directly; for

\* [[render]], it may be either a [[LegacySyncImporter]] or a

\* [[LegacyAsyncImporter]] which calls a callback with its result.

\*

\* See [[LegacySharedOptions.importer]] for more details.

\*

\* @category Legacy

\* @deprecated This only works with the legacy [[render]] and [[renderSync]]

\* APIs. Use [[Importer]] with [[compile]], [[compileString]], [[compileAsync]],

\* and [[compileStringAsync]] instead.

\*/

export type LegacyImporter<sync = 'sync' | 'async'> = sync extends 'async'

? LegacySyncImporter | LegacyAsyncImporter

: LegacySyncImporter;