API Report File for "@angular/localize_tools"

> *Do not edit this file. It is a report generated by [API Extractor](https://api-extractor.com/).*

```ts
import { AbsoluteFsPath } from '@angular/compiler-cli/private/localize';
import * as _babelNamespace from '@babel/core';
import { Element as Element_2 } from '@angular/compiler';
import { Logger } from '@angular/compiler-cli/private/localize';
import { NodePath } from '@babel/traverse';
import { ParseError } from '@angular/compiler';
import { PathManipulation } from '@angular/compiler-cli/private/localize';
import { ReadonlyFileSystem } from '@angular/compiler-cli/private/localize';
import * as t from '@babel/types';
import { ɵMessageId } from '@angular/localize';
import { ɵParsedMessage } from '@angular/localize';
import { ɵParsedTranslation } from '@angular/localize';
import { ɵSourceLocation } from '@angular/localize';
import { ɵSourceMessage } from '@angular/localize';

// @public
export class ArbTranslationParser implements TranslationParser<ArbJsonObject> {
    // (undocumented)
    analyze(_filePath: string, contents: string): ParseAnalysis<ArbJsonObject>;
    // @deprecated (undocumented)
    canParse(filePath: string, contents: string): ArbJsonObject | false;
    // (undocumented)
    parse(_filePath: string, contents: string, arb?: ArbJsonObject):
ParsedTranslationBundle;
}

// @public
export class ArbTranslationSerializer implements TranslationSerializer {
    constructor(sourceLocale: string, basePath: AbsoluteFsPath, fs:
PathManipulation);
    // (undocumented)
    serialize(messages: ɵParsedMessage[]): string;
}

// @public
export function buildLocalizeReplacement(messageParts: TemplateStringsArray,
substitutions: readonly t.Expression[]): t.Expression;

// @public
export function checkDuplicateMessages(fs: PathManipulation, messages:
ɵParsedMessage[], duplicateMessageHandling: DiagnosticHandlingStrategy, basePath:
AbsoluteFsPath): Diagnostics;

// @public
export type DiagnosticHandlingStrategy = 'error' | 'warning' | 'ignore';
```

```
// @public
export class Diagnostics {
    // (undocumented)
    add(type: DiagnosticHandlingStrategy, message: string): void;
    // (undocumented)
    error(message: string): void;
    // (undocumented)
    formatDiagnostics(message: string): string;
    // (undocumented)
    get hasErrors(): boolean;
    // (undocumented)
    merge(other: Diagnostics): void;
    // (undocumented)
    readonly messages: {
        type: 'warning' | 'error';
        message: string;
    }[];
    // (undocumented)
    warn(message: string): void;
}

// @public
export function isGlobalIdentifier(identifier: NodePath<t.Identifier>): boolean;

// @public
export class LegacyMessageIdMigrationSerializer implements TranslationSerializer {
    constructor(_diagnostics: Diagnostics);
    // (undocumented)
    serialize(messages: ɵParsedMessage[]): string;
}

// @public
export function makeEs2015TranslatePlugin(diagnostics: Diagnostics, translations:
Record<string, ɵParsedTranslation>, { missingTranslation, localizeName }?:
TranslatePluginOptions, fs?: PathManipulation): PluginObj;

// @public
export function makeEs5TranslatePlugin(diagnostics: Diagnostics, translations:
Record<string, ɵParsedTranslation>, { missingTranslation, localizeName }?:
TranslatePluginOptions, fs?: PathManipulation): PluginObj;

// @public
export function makeLocalePlugin(locale: string, { localizeName }?:
TranslatePluginOptions): PluginObj;

// @public
export class MessageExtractor {
    constructor(fs: ReadonlyFileSystem, logger: Logger, { basePath, useSourceMaps,
localizeName }: ExtractionOptions);
    // (undocumented)
    extractMessages(filename: string): ɵParsedMessage[];
}
```

```
// @public
export class SimpleJsonTranslationParser implements
TranslationParser<SimpleJsonFile> {
    // (undocumented)
    analyze(filePath: string, contents: string): ParseAnalysis<SimpleJsonFile>;
    // @deprecated (undocumented)
    canParse(filePath: string, contents: string): SimpleJsonFile | false;
    // (undocumented)
    parse(_filePath: string, contents: string, json?: SimpleJsonFile):
ParsedTranslationBundle;
}

// @public
export class SimpleJsonTranslationSerializer implements TranslationSerializer {
    constructor(sourceLocale: string);
    // (undocumented)
    serialize(messages: ɵParsedMessage[]): string;
}

// @public
export function translate(diagnostics: Diagnostics, translations: Record<string,
ɵParsedTranslation>, messageParts: TemplateStringsArray, substitutions: readonly
any[], missingTranslation: DiagnosticHandlingStrategy): [TemplateStringsArray,
readonly any[]];

// @public
export function unwrapExpressionsFromTemplateLiteral(quasi:
NodePath<t.TemplateLiteral>, fs?: PathManipulation): [t.Expression[],
(ɵSourceLocation | undefined)[]];

// @public
export function unwrapMessagePartsFromLocalizeCall(call: NodePath<t.CallExpression>,
fs?: PathManipulation): [TemplateStringsArray, (ɵSourceLocation | undefined)[]];

// @public
export function unwrapMessagePartsFromTemplateLiteral(elements:
NodePath<t.TemplateElement>[], fs?: PathManipulation): [
TemplateStringsArray,
(ɵSourceLocation | undefined)[]
];

// @public
export function unwrapSubstitutionsFromLocalizeCall(call:
NodePath<t.CallExpression>, fs?: PathManipulation): [t.Expression[],
(ɵSourceLocation | undefined)[]];

// @public
export class Xliff1TranslationParser implements
TranslationParser<XmlTranslationParserHint> {
    // (undocumented)
    analyze(filePath: string, contents: string):
```

```
ParseAnalysis<XmlTranslationParserHint>;
    // @deprecated (undocumented)
    canParse(filePath: string, contents: string): XmlTranslationParserHint | false;
    // (undocumented)
    parse(filePath: string, contents: string, hint?: XmlTranslationParserHint):
ParsedTranslationBundle;
}

// @public
export class Xliff1TranslationSerializer implements TranslationSerializer {
    constructor(sourceLocale: string, basePath: AbsoluteFsPath, useLegacyIds:
boolean, formatOptions?: FormatOptions, fs?: PathManipulation);
    // (undocumented)
    serialize(messages: ɵParsedMessage[]): string;
}

// @public
export class Xliff2TranslationParser implements
TranslationParser<XmlTranslationParserHint> {
    // (undocumented)
    analyze(filePath: string, contents: string):
ParseAnalysis<XmlTranslationParserHint>;
    // @deprecated (undocumented)
    canParse(filePath: string, contents: string): XmlTranslationParserHint | false;
    // (undocumented)
    parse(filePath: string, contents: string, hint?: XmlTranslationParserHint):
ParsedTranslationBundle;
}

// @public
export class Xliff2TranslationSerializer implements TranslationSerializer {
    constructor(sourceLocale: string, basePath: AbsoluteFsPath, useLegacyIds:
boolean, formatOptions?: FormatOptions, fs?: PathManipulation);
    // (undocumented)
    serialize(messages: ɵParsedMessage[]): string;
}

// @public
export class XmbTranslationSerializer implements TranslationSerializer {
    constructor(basePath: AbsoluteFsPath, useLegacyIds: boolean, fs?:
PathManipulation);
    // (undocumented)
    serialize(messages: ɵParsedMessage[]): string;
}

// @public
export class XtbTranslationParser implements
TranslationParser<XmlTranslationParserHint> {
    // (undocumented)
    analyze(filePath: string, contents: string):
ParseAnalysis<XmlTranslationParserHint>;
    // @deprecated (undocumented)
    canParse(filePath: string, contents: string): XmlTranslationParserHint | false;
    // (undocumented)
    parse(filePath: string, contents: string, hint?: XmlTranslationParserHint):
ParsedTranslationBundle;
}

// (No @packageDocumentation comment for this package)
```