ローカリゼーションモジュール

`Desktop.i18n` モジュールは、lit-element ベースのウィジェット用のローカリゼーションバンドルを作成し、維持します（Starter Widget をベースとして考慮した場合）。

ローカリゼーションモジュールは、i18next パッケージをベースとして作成されており、ウィジェット開発者がAgent Desktopの国際化メカニズムを利用し、追加のローカリゼーションバンドルを読み込むことができます。i18next の詳細については、次のリソースを参照してください。

- `Desktop.i18n` インスタンス化オブジェクトについては、*https://www.i18next.com/overview/api#instance-creation* を参照してください。
- i18n インスタンスのバックエンド構成については、*https://github.com/i18next/i18next-http-backend* を参照してください。
- i18n インスタンスの languageDetector 構成については、*https://github.com/i18next/i18next-browser-languageDetector* を参照してください。
- i18n インスタンスの初期化オプションについては、*https://www.i18next.com/overview/configuration-options* を参照してください。

このモジュールを使用すると、ウィジェットに lit-element および lit-html ライブラリが作成されます。

```
import {
    Desktop
} from "@wxcc-desktop/sdk";

...
// All CreateOptions for i18n are optional
type CreateOptions = {
    backend ? : Backend // import Backend from "i18next-http-backend";
    languageDetector ? : LanguageDetector // import LanguageDetector from
"i18next-browser-languagedetector";
};

const i18n = Desktop.i18n.createInstance(createOptions ? : CreateOptions) // returns
instance described in https://www.i18next.com/overview/api#instance-creation
const i18nMixin = Desktop.i18n.createMixin({
    i18n /*Injecting i18n service instance into lit-element mixin */
})
```

```
// FYI you can see default options like so
console.log(Desktop.i18n.DEFAULT_INIT_OPTIONS); // => i18n.init options that are using
by Desktop by default

// To get started, Init i18n with options to be able call "t" function translations
if (!i18n.isInitialized) {
    // Here, you are adding (merging) your localization package with the Agent Desktop
existing set of packages
   const initOptions = Desktop.i18n.getMergedInitOptions(Desktop.i18n.DEFAULT_INIT_OPTIONS
 || {}, {
       defaultNS: "my-ns", // "ns" here stands for the default JSON file name containing
 the localization
        ns: ["my-ns"],
        fallbackLng: "en",
        backend: {
            loadPath: "/.../path-to-locales/.../{{lng}}/{{ns}}.json"
        }
    });

    i18n.init(initOptions).catch(err => console.log(err));
}
```

サービスが初期化されたら、事前に作成した混合でコンポーネントを作成します。

```
import {
    customElement,
    LitElement
} from "lit-element";
import {
    html
} from "lit-html";

@customElement("my-awesome-component")
export class MyAwesomeComponent extends i18nMixin(LitElement) {
    render() {
        return html `
       <!-- i18nMixin will subscribe component tree updates on languages load & language
 change -->
       <!-- Component wrapped by i18nMixin can access t funcation via this.t(...) -->
       <p>${this.t("my-ns:key1")}</p>` <
            p > $ {
                this.t("my-ns:key2")
            } < /p>`
    }
}
```

• メソッド （2 ページ）

メソッド

init(initOptions)

ローカリゼーションモジュールを開始します。

例

```
i18n.init(initOptions)
```

パラメータ

名前	タイプ	説明	必須
initOptions	文字列	Desktopでデフォルトで使用される init オプション。 `console.log(Desktop.i18n.DEFAULT_INIT_OPTIONS)`	はい

createInstance(createOptions)

lit-element ベースのウィジェット用にローカリゼーションバンドルを作成します。

例

```
const i18n = Desktop.i18n.createInstance(createOptions ? : CreateOptions)
// returns instance described in https://www.i18next.com/overview/api#instance-creation
```

パラメータ

名前	タイプ	説明	必須
createOptions	文字列	ローカリゼーション（i18n）モジュールのオプションを作成します。	はい

createMixin()

ローカリゼーション（i18n）サービスインスタンスを lit-element との混合として作成します。

例

```
const i18nMixin = Desktop.i18n.createMixin({
    i18n /*Injecting i18n service instance into lit-element mixin */
})
```

getMergedInitOptions()

ローカリゼーションパッケージを、Agent Desktopの既存のパッケージセットに追加または統合します。

例

```
const initOptions = Desktop.i18n.getMergedInitOptions(Desktop.i18n.DEFAULT_INIT_OPTIONS
 || {}, {
    defaultNS: "my-ns", // "ns" here stands for the default JSON file name containing
the localization
    ns: ["my-ns"],
    fallbackLng: "en",
    backend: {
        loadPath: "/.../path-to-locales/.../{{lng}}/{{ns}}.json"
    }
});
```

次の表に、ペイロードの詳細を示します。

名前	タイプ	説明	必須
DEFAULT_INIT_OPTIONS	文字列	Desktopでデフォルトで使用されるinitオプション。	はい
defaultNS	文字列	デフォルトのローカリゼーションJSONファイル名。	はい
ns	文字列	ロケールのJSONファイル名。	はい
fallbackLng	文字列	構成されたパスにファイルが見つからない場合、またはユーザーの優先言語を提供できない場合に使用する言語。	はい
backend	文字列	ロードパスが含まれます。	はい
-->loadPath	文字列	ファイルパスの場所。	はい

cleanup()

初期化サービスが未定義の応答を返すと、クリーンアップをトリガーします。

例

```
cleanup() {
    this.SERVICE = undefined;

    this.logger.info("Cleaned");
}
```