Finesse ガジェットの移行

Finesse 埋め込み Web アプリケーションガジェットの移行

Finesse 埋め込み Web アプリケーションのガジェットを、Webex Contact Center エージェントデスクトップの iFrame ベースのウィジェットに移行できます。

（注） Finesse ガジェットの変換は、Web アプリケーションの構成、つまり iFrame に読み込めるかどうかによってによって異なります。

Finesse：埋め込み Web アプリケーションのサンプルガジェット

埋め込み Web アプリケーションのサンプルガジェットは、ガジェット内の iFrame に Web ページを表示します。これは、外部 Web アプリケーションを Finesse ガジェットに配置する例として示すことを目的としています。

埋め込み Web アプリケーションのサンプルガジェットの詳細については、https://github.com/CiscoDevNet/finesse-sample-code/tree/master/EmbeddedWebAppSampleGadget を参照してください。

Webex Contact Center エージェントデスクトップ：agentx-wc-iframe

デスクトップには、iFrame 内に Web アプリケーションを読み込むための固有の機能（Web コンポーネント）があります。デスクトップレイアウト内でコンポーネント `agentx-wc-iframe` を使用して、水平ヘッダー、ナビゲーションバー、補助情報ペインに Web アプリケーションを読み込むことができます。

Webex Contact Center エージェントデスクトップのウィジェットの詳細については、https://github.com/CiscoDevNet/webex-contact-center-widget-starter/tree/master/Examples を参照してください。

Finesse 埋め込みガジェットファイルは次のとおりです。

- `EmbeddedWebApp.css`
- `EmbeddedWebApp.js`
- `EmbeddedWebApp.xml`

`EmbeddedWebApp.xml` は、Web アプリケーションの URL を含むメインファイルです。

始める前に

- Finesse 埋め込みガジェットファイル `EmbeddedWebApp.xml` からのスクリプトタイプの詳細を理解します。

```
<script type="text/javascript">
                // initialize the gadget running the init handler defined in
EmbeddedWebApp.js
                gadgets.HubSettings.onConnect = function () {
                 finesse.modules.EmbeddedWebAppGadget.init("https://www.bing.com");

                };
            </script>
```

- JSON レイアウトフォーマットを理解します。JSON レイアウトの詳細については、『*Cisco Webex Contact Center Setup and Administration Guide*（*Cisco Webex Contact Center* 設定およびアドミニストレーションガイド）』の「プロビジョニング（Provisioning）」の章の「デスクトップレイアウト（Desktop Layout）」セクションを参照してください。

ステップ 1 Webex Contact Center 管理ポータルから、JSON レイアウトファイルにアクセスします。

ステップ 2 **comp** プロパティタグ内に、コンポーネント値として **agentx-wc-iframe** を入力します。

```
{
    "comp": "agentx-wc-iframe"
},
```

ステップ 3 **src attributes** プロパティタグ内に、Web アプリケーションの URL を入力します。

```
{
    "comp": "agentx-wc-iframe",
    "attributes": {
        "src": "https://www.bing.com"
    },
```

ステップ 4 **wrapper** プロパティタグ内に、タイトルと最大化エリア名を入力します。

```
{
    "comp": "agentx-wc-iframe",
    "attributes": {
        "src": "https://www.bing.com"
    },
    "wrapper": {
        "title": "AgentX iFrame",
        "maximizeAreaName": "app-maximize-area"
    },
```

ステップ 5 iFrame のサイズを変更するには、**style** プロパティタグを使用します。

（注） デスクトップ内の iFrame は、ネイティブの iFrame プロパティをサポートしています。

```
{
    "comp": "agentx-wc-iframe",
    "attributes": {
        "src": "https://www.bing.com"
    },
    "wrapper": {
        "title": "AgentX iFrame",
        "maximizeAreaName": "app-maximize-area"
    },
    "style": {
        "height": 504 px,
        "width": 520 px,
        "display": "inline-block",
        "align-items": "center"
    },
```

ステップ 6 iFrame 属性を追加するには、**attributes** プロパティタグを使用します。

```
{
    "comp": "agentx-wc-iframe",
    "attributes": {
        "src": "https://www.bing.com"
        "sandbox": "allow-scripts allow popups"
    },
    "wrapper": {
        "title": "AgentX iFrame",
        "maximizeAreaName": "app-maximize-area"
    },
    "style": {
        "height": 504 px,
        "width": 520 px,
        "display": "inline-block",
        "align-items": "center"
    },
```

JavaScript ベースの Finesse ガジェットの移行

Shindig ベースのガジェットから wxcc-desktop ベースのウィジェットに直接移行する方法はありません。Shindig ガジェットと wxcc-desktop ウィジェットはテクノロジを活用しているため、既存のソースコードをリファクタリングして wxcc-desktop ベースのウィジェットを開発することができます。Finesse ガジェットは、Apache Shindig ガジェットの仕様に基づいています。Apache Shindig の詳細については、https://shindig.apache.org/ を参照してください。任意の JavaScript ライブラリ、フレームワーク、通常の JavaScript を使用して、UI ベースのガジェットを構築できます。最終的なコンテンツは、ガジェットの XML または HTML 内に埋め込む必要があります。

Finesse ガジェット

静的リソース：Finesse ガジェットは、XML、HTML、CSS、JavaScript、画像などの構成要素で構成されています。これらのガジェットは Finesse 3rdpartygadget Web アプリケーションに展開されます。

要素：ガジェットの重要な要素は次のとおりです。

- <Module> タグは、ルートレベルの要素であるガジェットが XML ファイルに含まれていることを示します。
- <ModulePrefs> タグには、ガジェットに関する情報が含まれます。このタグには、タイトル、説明、作成者、その他のオプション機能と、Finesse デスクトップコンテナからのガジェット関連の構成が含まれます。
- <Content type="html"> ヘッダーは、ガジェットのコンテンツタイプが HTML であることを示します。このタグには、コンテンツ要素内に埋め込まれているすべてのマークアップコンテンツ（HTML）と静的リソースが含まれます。

例：サンプルガジェット

```
<Module>
    <ModulePrefs title="X-Counter" description="X-Counter Example"></ModulePrefs>
    <Content type="html">
        <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"/>
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.29.1/moment.min.js"></script>
        <![CDATA[
    <style>
        button, p {
          display: inline-block;
        }
    </style><button aria-label="decrement">-</button><p>0</p><button
aria-label="increment">+</button><script>
        var valueElement = document.querySelector('p');
        function increment() {
            const value = valueElement.innerText;
            valueElement.innerText = parseInt(value) + 1;
        }
        function decrement() {
            const value = valueElement.innerText;
            valueElement.innerText = parseInt(value) - 1;
        }
        var incrementButton = document.querySelectorAll('button')[1];
        var decrementButton = document.querySelectorAll('button')[0];

        incrementButton.addEventListener('click', e => increment);

        decrementButton.addEventListener('click', e => decrement);
    </script>
    ]]>
    </Content>
</Module>
```

Cisco Finesse ガジェットの詳細については、https://developer.cisco.com/docs/finesse/#!finesse-overview/cisco-finesse-gadgets-finesse-javascript-library-api を参照してください。

Webex Contact Center エージェントデスクトップのウィジェット

デスクトップのウィジェットは、ブラウザのネイティブ Web コンポーネントモジュールを活用するマイクロフロントエンドの方法に基づいています。

静的リソース：カスタムウィジェットでは、バンドルライブラリ（Webpack、Rollup、Parcel など）を使用して、すべての静的リソース（CSS、JavaScript、マークアップ、画像）を 1 つの

JavaScript ファイルにバンドルする必要があります。このバンドルは、コンテンツ配信ネットワーク（CDN）にホストする必要があります。カスタムウィジェットでは、Web コンポーネントを開発するのに任意の JavaScript フレームワークを使用できます。

デスクトップからの共有データ（ModulePrefs など）：カスタムウィジェットでは、デスクトップからの共有データまたは構成を Web コンポーネントのプロパティまたは属性として使用できます。

（注） デスクトップの共有データまたは構成は、Finesse ガジェットの共有データまたは構成とは異なります。開発者は、`wxcc-desktop` で利用可能な、関連する共有データまたは構成について理解しておく必要があります。詳細については、データプロバイダー：ウィジェットのプロパティと属性を参照してください。

JavaScript バンドルには、Web コンポーネントとしてのルート要素が必要であり、その要素名はデスクトップレイアウト構成内に記述する必要があります。カスタムウィジェットは、任意の Web ベースのライブラリまたはフレームワーク（(例：React、Angular、Web コンポーネント）を使用して開発できます。カスタムウィジェットでは、CSS、JavaScript、マークアップ、画像などの追加の静的リソースを、Ajax またはマークアップタグ（script、link、img など）を使用して内部に読み込むことができます。

以下に、バンドルライブラリ（helloworld.js）を使用せずに通常の JavaScript で構築されたサンプルの Hello World ウィジェットを例として示します。この例では、ウィジェット名は `<x-counter>` です。

例：バンドルライブラリを使用しないサンプルのウィジェット

```
const template = document.createElement('template');
template.innerHTML = `
  <style>
    button, p {
      display: inline-block;
    }
  </style>
  <button aria-label="decrement">-</button>
    <p>0</p>
  <button aria-label="increment">+</button>
`;

class XCounter extends HTMLElement {


    increment() {
        const value = this.valueElement.innerText;
        this.valueElement.innerText = parseInt(value) + 1;
    }
    decrement() {
        const value = valueElement.innerText;
        valueElement.innerText = parseInt(value) - 1;
    }
    constructor() {
        super();

        this.attachShadow({
            mode: 'open'
```

```
        });
        this.shadowRoot.appendChild(template.content.cloneNode(true));

        this.valueElement = this.shadowRoot.querySelector('p');
        this.incrementButton = this.shadowRoot.querySelectorAll('button')[1];
        this.decrementButton = this.shadowRoot.querySelectorAll('button')[0];

        this.incrementButton.addEventListener('click', e => this.increment);

        this.decrementButton.addEventListener('click', e => this.decrement);
    }
}

customElements.define('x-counter', XCounter);
```