Live Agent 開発者ガイド

バージョン 37.0, Summer '16

@salesforcedocs

最終更新日: 2016/04/22

本書の英語版と翻訳版で相違がある場合は英語版を優先するものとします。

目次

第 1 章: このガイドの内容 1

第 2 章: 前提条件 2

第 3 章: API バージョン 3

第 4 章: リリース API を使用したリリースのカスタマイズ 4

リリースの作成 4

リリース API を使用したリリースアクティビティのログ記録 5

enableLogging 5

リリース API を使用したチャットウィンドウのカスタマイズ 10

setChatWindowHeight 10

setChatWindowWidth 11

リリース API を使用したチャットボタンのカスタマイズ 11

showWhenOnline 12

showWhenOffline 13

addButtonEventHandler 14

startChat 16

startChatWithWindow 17

チャットボタンの対応するコール 17

リリース API を使用したレコードの自動検索および自動作成 18

addCustomDetail 18

findOrCreate 19

setName 23

リリース API を使用したナレッジ記事の検索 23

レコード検索および作成のリリース API のコードサンプル 24

リリース API を使用した自動チャット招待のカスタマイズ 25

setCustomVariable 26

rejectChat 26

addButtonEventHandler 27

自動チャット招待のコードサンプル 28

開発 API のコードサンプル 30

第 5 章: 事前チャットを使用した訪問者情報の収集とエージェントのコンテキストの設定 34

事前チャット API を使用したレコードの自動検索および自動作成 34

findOrCreate.map 35

findOrCreate.saveToTranscript 39

findOrCreate.showOnCreate 40

findOrCreate.linkToEntity 41

findOrCreate.displayToAgent 42
レコード検索および作成の事前チャット API のコードサンプル 43
事前チャット API を使用したチャットの詳細へのアクセス 44
preChatInit 44
事前チャットフォームのコードサンプル 45

第 6 章: Visualforce を使用したカスタムチャットウィンドウの実装 49

Live Agent Visualforce コンポーネント 49
Live Agent Visualforce コンポーネントのコードサンプル 51

第 7 章: 事後チャットを使用した顧客とのチャットのやりとりのまとめ 55

事後チャットのコードサンプル 55

第 8 章: リリース API を使用したエージェントへのチャットの直接転送の設定 59

リリース API を使用したエージェントへの直接転送 59
エージェントへの直接転送のコードサンプル 59
事前チャットフォームの代替転送 61
代替転送のコードサンプル 61

第 1 章　　このガイドの内容

会社のニーズに合わせて Live Agent をカスタマイズします。このガイドでは、カスタマイズされたチャットウィンドウ、ボタン、フォーム、およびページの理解と作成に役立つくつかの例を示します。

Live Agent により、サービス組織は、Web ベースのテキストのみの Live Chat を使用して顧客または Web サイトの訪問者とリアルタイムに接続できます。カスタムコードを使用して Live Agent をカスタマイズすると、カスタマーサービスエージェントとその顧客のためにパーソナライズされたチャット環境を作成できます。このガイドでは、次の操作方法を示します。

- リリース API を使用してリリースをカスタマイズする。
- Visualforce ページおよびコンポーネントを使用して、顧客向けチャットウィンドウの外観をカスタマイズする。
- 事前チャットフォームを作成し、顧客がエージェントとのチャットを開始する前に顧客情報を収集できるようにする。
- チャット終了後に顧客に表示される、事後チャットページを作成する。

また、Salesforce 設定を使用して、上記の操作および他の Live Agent コンポーネントをカスタマイズできます。詳細は、Salesforce ヘルプの「Live Agent 実装のカスタマイズ」を参照してください。

第 2 章　前提条件

Live Agent をカスタマイズする前に、次のことを確認してください。

- 組織で Live Agent が有効になっている。
- システム管理者から Live Agent 機能ライセンスが付与されている。機能ライセンスがなくても製品をカスタマイズできますが、機能ライセンスがあるとカスタマイズにアクセスし、テストすることができます。
- Force.com サイトを作成し、チャットボタンおよびウィンドウの画像を静的リソースとしてアップロードした。Force.com サイトを使用せずに Live Agent をカスタマイズする場合は、このステップを省略します。

メモ: Live Agent カスタムチャットページに Force.com サイトを使用する場合、URL にパス「/liveagent」を使用しないでください。このパスを使用すると、送受信チャット通知音でエラーが発生し、エージェントがチャットの更新を聞き取れなくなることがあります。

第 3 章　API バージョン

Live Agent の API の異なるバージョンでは、使用できるメソッドとパラメータが異なります。リリース API または事前チャット API を使用して開発を開始する前に、コードで使用している API のバージョン番号が正しいことを確認します。

リリース API バージョン

リリース作成後に生成されたリリースコードから、組織で使用するリリースAPIのバージョンを確認できます。

Summer '13 以前のリリースでは、バージョン 28.0 の開発 API をサポートしています。API バージョン 28.0 の URL は、`https://hostname.salesforceliveagent.com/content/g/deployment.js` のようになります。

Winter '14 では、バージョン 29.0 の開発 API をサポートしています。API バージョン 29.0 の URL には、`https://hostname.salesforceliveagent.com/content/g/js/29.0/deployment.js` のようにバージョン番号が含まれます。

メモ: リリースで新しいメソッドとパラメータを使用するには、各 Web ページでリリースコードを更新して、リリース API のバージョン 29.0 の URL を使用する必要があります。

事前チャット情報 API バージョン

Winter '14 では、バージョン 29.0 の事前チャット API をサポートしています。API バージョン 29.0 の URL には、`https://hostname.salesforceliveagent.com/content/g/js/29.0/prechat.js` のようにバージョン番号が含まれます。

リリース作成後に生成されたリリースコードから、組織のホスト名を確認できます。

第 4 章　リリース API を使用したリリースのカスタマイズ

Live Agent リリース API を使用してリリースをカスタマイズします。

リリースとは、会社の Web サイト上の、Live Agent を有効化する場所です。リリースは、Web ページに追加する数行の JavaScript で構成されます。組織は、Live Agent リリースを 1 つまたは複数持つことができます。たとえば、1 つのサービスセンターで複数の Web サイトをサポートする場合、サイトごとに別々のリリースを作成すると、訪問者に複数のチャットウィンドウを表示できます。各リリースには、訪問者がサポートエージェントとチャットするために使用するチャットウィンドウが含まれます。

リリース API は、JavaScript ベースの API であり、リリースをカスタマイズしてバックエンド機能を指定できます。

リリースの作成
Web サイトで Live Agent をホストするリリースを作成します。各リリースには、訪問者がサポートエージェントとチャットするために使用するチャットウィンドウが含まれます。

リリース API を使用したリリースアクティビティのログ記録
特定のリリースで発生するアクティビティのログを記録します。

リリース API を使用したチャットウィンドウのカスタマイズ
顧客向けチャットウィンドウのサイズをカスタマイズします。これは、モバイルベースのブラウザには適用されません。

リリース API を使用したチャットボタンのカスタマイズ
チャットボタンをカスタマイズして、顧客のチャットの開始方法を設定します。

リリース API を使用したレコードの自動検索および自動作成
リリース API を使用して、エージェントが顧客とチャットを開始したときにケース、取引先責任者、取引先、リードなどの Salesforce レコードを自動的に検索または作成します。

リリース API を使用した自動チャット招待のカスタマイズ
Web サイトで顧客に表示される自動チャット招待をカスタマイズします。

開発 API のコードサンプル
リリース API がリリースのカスタマイズにどのように役立つかをテストおよびプレビューします。

リリースの作成

Web サイトで Live Agent をホストするリリースを作成します。各リリースには、訪問者がサポートエージェントとチャットするために使用するチャットウィンドウが含まれます。

リリース API で会社のニーズに合わせて Live Agent リリースをカスタマイズできます。次の手順を実行すると、リリースコードが生成されて、チャットおよび追跡で有効にするページに配置されます。リリースコードのあるページは、訪問者のチャットセッションの一部として自動的に追跡され、訪問者がチャットを要求したとき

にエージェントのコンソールに表示されます。また、この追跡により、自動招待を顧客に提供できるようになります。

リリースを作成する手順は、次のとおりです。

1. [設定] から、［クイック検索］ボックスに「*リリース*」と入力し、[リリース] を選択します。
2. [新規] をクリックします。
3. リリースの名前を入力します。このリリースの名前、つまりバージョンは自動的に ［API 参照名］ になります。
4. チャットウィンドウのタイトルを入力します。
5. 訪問者がチャットセッションのコピーをダウンロードできるようにするには、［訪問者にトランスクリプトの保存を許可する］ を選択します。
6. Force.com サイトでホストしたブランド画像を使用する場合、リリースに関連付けるサイトを選択します。
7. (省略可能) ［［チャット］ ウィンドウのブランド画像］ で、チャットウィンドウに表示される画像を選択します。
8. (省略可能) ［［モバイルチャット］ ウィンドウのブランド画像］ で、モバイルデバイスを使用する訪問者のチャットウィンドウに表示される画像を選択します。
9. [保存] をクリックします。Salesforce は、リリースコードを生成します。
10. リリースコードをコピーして、Live Agent をリリースする各 Web ページに貼り付けます。最高のパフォーマンスを得るには、このコードを body タグの直前に貼り付けます。

例: リリースの作成についての詳細は、「Live Agent リリースの作成」を参照してください。

リリース API を使用したリリースアクティビティのログ記録

特定のリリースで発生するアクティビティのログを記録します。

ログ記録では、特定のリリースを介して顧客がエージェントとチャットするときに、顧客の Web ブラウザで発生するアクティビティに関する情報を保存できます。これは、自動招待を実装していて、送信ルールをテストまたはトラブルシューティングする場合に特に役立ちます。これらのメソッドは、リリースの作成時に自動生成されるコード内で、追加スクリプトとして追加できます。

次のリリースメソッドは、特定のリリースでログ記録を有効化するために使用します。

enableLogging

`enableLogging` リリースメソッドは、特定のリリースでログ記録を有効化するために使用します。API バージョン 28.0 以降で使用できます。

enableLogging

`enableLogging` リリースメソッドは、特定のリリースでログ記録を有効化するために使用します。API バージョン 28.0 以降で使用できます。

使用方法

特定のリリースでのログ記録を有効化し、Web ブラウザの JavaScript コンソールで、リリース内で発生したアクティビティに関する情報を保存できるようにします。ブラウザの開発者コンソールから情報を取得できます。そのため、情報の検索方法がわからない場合はブラウザのヘルプを確認してください。

構文

```
liveagent.enableLogging();
```

パラメータ

なし

記録されたイベントのメッセージ

メッセージ	トリガされるタイミング	意味
System initialized. Waiting for the DOM to be ready. (システムが初期化されました。DOM が準備できるまで待機しています。)	liveagent.init() がコールされたとき (通常はページの読み込み時)	Live Agent のエンドポイント URL、組織 ID およびリリース ID が設定されていて、現在 DOM が準備できるまで続行を待機しています。
No available event model. Exiting. (使用可能なイベントモデルがありません。終了しています。)	liveagent.init() 実行中 (エラーがある場合)	これは、ほとんど発生することはありませんが、DOM イベントリスナーが検出されなかったことを意味します。この時点で続行できないため、強制終了されます。
DOM is ready. Setting up environment. (DOM の準備ができました。環境を設定しています。)	ページの DOM の準備ができたとき	ページが完全に読み込まれ、DOM の準備ができたため、サーバへの最初の「ping」を実行し、指定されたリリース ID に関する設定/情報を取得します。
Setting state for button {Button ID} to online (ボタン {Button ID} の状態をオンラインに設定しています)	ボタンの状態がオンラインに変更されたとき	ボタンを使用してチャット要求を実行できます。
Setting state for button {Button ID} to offline (ボタン {Button ID} の状態をオフラインに設定しています)	ボタンの状態がオフラインに変更されたとき	ボタンを使用してチャット要求を実行できません。
Requesting new session (新しいセッションを要求しています)	サーバへの最初の ping 実行中	セッション ID Cookie が検出されなかったため、新たに生成する必要があります。これは、この参照セッションのこのリリースコードのあ

メッセージ	トリガされるタイミング	意味
		るサイトに初めてアクセスしたことを意味します。
Reusing existing session (既存のセッションを再利用しています)	サーバへの最初の ping 実行中	セッション Cookie が存在しているため、再利用されます。これは、この参照セッション (ページ間の移動など) 中に訪問者がすでにこのサイトにアクセスしていることを意味します。
Received new session ID (新しいセッション ID を受信しました)	最初の ping に対する応答として	新しいセッション ID がサーバで生成され、「liveagent_sid」という名前のセッション Cookie として保存されています。
Ping rate set to {Rate}ms (Ping の頻度が {Rate}ms に設定されました)	最初の ping に対する応答として	ページで Live Agent サーバを ping する頻度 (ミリ秒) を示します。デフォルトは 50000 (50 秒) です。これは、実質的にボタンの更新のタイミングを示します。
Pinging server to keep presence (プレゼンスを維持するためにサーバを ping しています)	サーバへの ping が実行されたとき	訪問者がまだ Live Agent サーバに接続されていて、ping していることを示します。つまり、エラーや切断は発生していません。
Disconnecting from Live Agent (Live Agent から切断されています)	エラーが発生したとき	サーバからの応答またはネットワーク接続の問題が原因でエラーが発生しました。訪問者がこのページの読み込みで Live Agent を ping していないことを示します (つまり、更新が必要になる)。
Received updated LiveAgent server url: {URL}! (更新された LiveAgent サーバ URL {URL} を受信しました。) Consider updating this site's deployment code. (サイトのリリースコードの更新を検討してください。)	組織が新しいコアインスタンスに移動したとき	リリースコードで指定された Live Agent インスタンスがこの組織で無効になったため、新しい URL が提供されました。パフォーマンス向上のため、この URL を受信している場合はリリースコードを更新することをお勧めします。
Server Warning: {Message} (サーバの警告: {Message})	致命的でない例外が発生したとき	警告状態になりましたが、処理は続行できます。メッセージには、詳細が記載されています。

メッセージ	トリガされるタイミング	意味
Server sent an anonymous warning (サーバから匿名の警告が送信されました)	致命的でない例外が発生したとき	警告状態になりましたが、処理は続行できます。メッセージはありません。
Server Error: {Message} (サーバエラー: {Message})	致命的な例外が発生したとき	エラー状態になったため、処理は続行できません。メッセージには、詳細が記載されています。
Server responded with an error (サーバがエラーで応答しました)	致命的な例外が発生したとき	エラー状態になったため、処理は続行できません。メッセージはありません。
Group Start: Invite {Button ID} Rule Evaluation (グループの先頭: 招待 {Button ID} のルール評価)	ルール評価がトリガされたとき	指定された招待ボタン ID の検索条件ロジックの評価が開始されました。これは、ボタンがオンラインでチャットに使用でき、表示/提供すべきかどうかを決定するために検索条件ロジックが使用されることを意味します。
Filter Logic: {Filter Logic} (検索条件ロジック: {Filter Logic})	ルール評価がトリガされたとき	情報ログには、管理設定領域で指定した招待ルールの検索条件ロジックの文字列表現が含まれます。ルールの評価方法を理解するのに便利です。
Evaluating StandardInviteRule (StandardInviteRule を評価しています)	標準ルールが評価されているとき	標準ルールは、「ページ表示数」および「URL の一致」です。これらは、管理設定領域で提供される標準ルールの一部です。
Evaluating TimerInviteRule (TimerInviteRule を評価しています)	タイマーベースルールが評価されているとき	タイマーベースルールは、「ページでの秒数」および「サイトでの秒数」です。これらも標準ルールの一部ですが、これらのルールは、初回時(ページの読み込み時など)に条件が一致しなかった場合、その後に必要な秒数が経過してから再評価されます。
Evaluating CustomInviteRule (CustomInviteRule を評価しています)	カスタムルールが評価されているとき	「カスタム変数」ルールでは、これらのルールを評価するときに比較される変数名を指定できます。「setCustomVariable」API とこれらのルールを併用して、管理設定領域

メッセージ	トリガされるタイミング	意味
		で指定された値と比較される値を指定します。
CustomInviteRule evaluation failed due to missing custom variable (カスタム変数が欠落していたため、CustomInviteRule の評価に失敗しました)	カスタムルールが評価されているとき	「カスタム変数」ルールが設定されましたが、この変数名が指定された「setCustomVariable」API がコールされなかったため、ルールを評価できません。
Evaluate: {From Value} == {To Value} (評価: {From Value} == {To Value})	「次の文字列と一致する」比較子を含むルールが評価されているとき	2つの値が完全に一致するかどうかを比較することでルールが評価されています。
Not Equals - Evaluate: {From Value} != {To Value} (次の文字列と一致しない - 評価: {From Value} != {To Value})	「次の文字列と一致しない」比較子を含むルールが評価されているとき	2つの値が一致しないかどうかを比較することでルールが評価されています。
Starts With - Evaluate: {From Value} indexOf {To Value} == 0 (次の文字列で始まる - 評価: {From Value} indexOf {To Value} == 0)	「次の文字列で始まる」比較子を含むルールが評価されているとき	最初の値が2番目の値で始まるかどうかを比較することでルールが評価されています。
Contains - Evaluate: {From Value} indexOf {To Value} != -1 (次の文字列を含む - 評価: {From Value} indexOf {To Value} != -1)	「次の文字列を含む」比較子を含むルールが評価されているとき	最初の値に2番目の値が含まれているかどうかを比較することでルールが評価されています。
Does Not Contain - Evaluate: {From Value} indexOf {To Value} == -1 (次の文字列を含まない - 評価: {From Value} indexOf {To Value} == -1)	「次の文字列を含まない」比較子を含むルールが評価されているとき	最初の値に2番目の値が含まれていないかどうかを比較することでルールが評価されています。
Less Than - Evaluate: {From Value} < {To Value} (＜ - 評価: {From Value} < {To Value})	「＜」比較子を含むルールが評価されているとき	最初の値が2番目の値より小さいかどうかを比較することでルールが評価されています。
Greater Than - Evaluate: {From Value} > {To Value} (＞ - 評価: {From Value} > {To Value})	「＞」比較子を含むルールが評価されているとき	最初の値が2番目の値より大きいかどうかを比較することでルールが評価されています。
Less or Equal - Evaluate: {From Value} <= {To Value} (＜＝ - 評価: {From Value} <= {To Value})	「＜＝」比較子を含むルールが評価されているとき	最初の値が2番目の値以下かどうかを比較することでルールが評価されています。
Greater or Equal - Evaluate: {From Value} >= {To Value} (＞＝ - 評価: {From Value} >= {To Value})	「＞＝」比較子を含むルールが評価されているとき	最初の値が2番目の値以上かどうかを比較することでルールが評価されています。
Evaluating Atom Node: {Rule ID} (Atom ノードを評価しています: {Rule ID})	ルールが評価されているとき	実際のルールが評価されていることを示します。

メッセージ	トリガされるタイミング	意味
Group Start: Evaluating And Node (グループの先頭: AND ノードを評価しています)	2 つのルールが「AND」句で評価されているとき	複数のルールが使用されている場合、これは、ルールのペアの条件が両方とも「true」になっている必要があることを示します。
Group Start: Evaluating Or Node (グループの先頭: OR ノードを評価しています)	2つのルールが「OR」句で評価されているとき	複数のルールが使用されている場合、これは、ルールのペアの条件のいずれかが「true」になっている必要があることを示します。
Group Start: Evaluating Not Node (グループの先頭: NOT ノードを評価しています)	2 つのルールが「NOT」句で評価されているとき	これは、条件の評価と反対になっていることを確認することを示します。
Setting invite delay to: {Invite Delay} (招待の遅延を {Invite Delay} に設定しています)	タイマーベースルールの条件がまだ満たされていないとき	タイマーベースルールの条件が満たされていない場合、条件が一致したときに後でルールを再評価するための遅延が設定されます。

リリース API を使用したチャットウィンドウのカスタマイズ

顧客向けチャットウィンドウのサイズをカスタマイズします。これは、モバイルベースのブラウザには適用されません。

次のリリースメソッドを使用して、チャットウィンドウの高さと幅をカスタマイズします。いずれのメソッドも、リリースの作成時に自動生成されるコード内で、追加スクリプトとして追加できます。

メモ: これは、チャットがページ全体で開くモバイルブラウザには適用されません。

setChatWindowHeight

`setChatWindowHeight` メソッドは、チャットウィンドウの高さをカスタマイズするために使用します。

setChatWindowWidth

`setChatWindowWidth` メソッドは、チャットウィンドウの幅をカスタマイズするために使用します。

`setChatWindowHeight`

`setChatWindowHeight` メソッドは、チャットウィンドウの高さをカスタマイズするために使用します。

使用方法

顧客に表示されるチャットウィンドウの高さ (ピクセル単位) を設定します。API バージョン 28.0 以降で使用できます。

構文

```
void setChatWindowHeight(Number height)
```

パラメータ

名前	型	説明	使用可能なバージョン
height	数値	カスタムチャットウィンドウの高さ (ピクセル単位)。	API バージョン 28.0 以降で使用できます。

setChatWindowWidth

setChatWindowWidth メソッドは、チャットウィンドウの幅をカスタマイズするために使用します。

使用方法

顧客に表示されるチャットウィンドウの幅 (ピクセル単位) を設定します。API バージョン 28.0 以降で使用できます。

構文

```
void setChatWindowWidth(Number width)
```

パラメータ

名前	型	説明	使用可能なバージョン
width	数値	カスタムチャットウィンドウの幅 (ピクセル単位)。	API バージョン 28.0 以降で使用できます。

リリース API を使用したチャットボタンのカスタマイズ

チャットボタンをカスタマイズして、顧客のチャットの開始方法を設定します。

各チャットボタンには、Web サイトに配置して顧客がチャットを開始できるようにするコードが含まれます。Live Agent は、エージェントの対応可能状況や組織の設定に基づいてボタンを使用できるかどうかを自動的に処理します。また、ボタンからのチャット要求の開始も処理します。

次のリリースメソッドを使用して、チャットボタンとチャットの開始をカスタマイズします。どのメソッドも、リリースの作成時に自動生成されるコード内で、追加スクリプトとして追加できます。

showWhenOnline

showWhenOnline メソッドは、特定のボタンがオンラインの場合に顧客に表示される内容を指定するために使用します。

showWhenOffline

showWhenOffline メソッドは、特定のボタンがオフラインの場合に顧客に表示される内容を指定するために使用します。

addButtonEventHandler

addButtonEventHandler メソッドは、特定のイベントが発生したときのチャットボタンの動作を定義するために使用します。API バージョン 28.0 以降で使用できます。

startChat

startChat メソッドは、新しいウィンドウのボタンからチャットを要求するために使用します。

startChatWithWindow

startChatWithWindow メソッドは、ウィンドウの名前を使用してボタンからチャットを要求するために使用します。

チャットボタンの対応するコール

ボタン、直接転送エージェント、ボタンへの代替があるエージェントを使用する場合は、コールを調整してチャットが正常に開始されることを確認します。

showWhenOnline

showWhenOnline メソッドは、特定のボタンがオンラインの場合に顧客に表示される内容を指定するために使用します。

使用方法

指定したボタン、エージェント、またはボタンへの代替があるエージェントがオンラインの場合に、特定の要素を表示します。API バージョン 28.0 以降で利用できます。

構文

ボタンの場合、userId は省略可能です。`void showWhenOnline(String buttonId, Object element, (optional) String userId)`

エージェントの場合、buttonId の代わりに userId を使用します。`void showWhenOnline(String userId, Object element)`

ボタンへの代替があるエージェントの場合、両方の ID を使用します (要素はエージェントまたはボタンがオンラインの場合に表示されます)。`void showWhenOnline(String buttonId, Object element, String userId)`

メモ: buttonId と userId の両方を使用する場合は、buttonId を最初に指定する必要があります。

パラメータ

名前	型	説明	使用可能なバージョン
`buttonId`	String	ボタンに関連付けられているエージェントがチャットに対応できる場合に、指定した `element` オブジェクトを表示するチャットボタンの ID。	API バージョン 28.0 以降で使用できます。
`element`	Object	指定したボタンがオンラインの場合に表示する要素。	API バージョン 28.0 以降で使用できます。
`userId`	String	ボタンに関連付けるエージェントの ID。エージェントが対応できる場合に、`element` オブジェクトが表示されます。	API バージョン 29.0 以降で使用できます。

パラメータにボタン ID を指定してユーザ ID は指定しない場合、要素はボタンがオンラインの場合にのみ表示されます。

ユーザ ID を指定してボタン ID は指定しない場合、要素はエージェントがオンラインの場合にのみ表示されます。たとえば、次の構文は、エージェントのオンライン状況を追跡し、エージェントが対応できる場合はボタンをオンラインに設定して、対応できない場合はオフラインに設定します。

```
liveagent.showWhenOnline('005xx000001Sv1m',
document.getElementById('liveagent_button_toAgent_online'));
```

ボタン ID とエージェント ID を指定する場合、要素はボタンとエージェントのいずれかがオンラインの場合に表示されます。たとえば、次の構文は、エージェントとボタンの状況を追跡して、スキルのある 1 人以上のエージェントが対応できる場合に要素を表示します。

```
liveagent.showWhenOnline('573xx0000000006',
document.getElementById('liveagent_button_online_573xx0000000006_USER1'), '005xx000001Sv1m');
```

showWhenOffline

`showWhenOffline` メソッドは、特定のボタンがオフラインの場合に顧客に表示される内容を指定するために使用します。

使用方法

指定したボタン、エージェント、またはボタンへの代替があるエージェントがオフラインの場合に、特定の要素を表示します。API バージョン 28.0 以降で使用できます。

構文

ボタンの場合、`userId` は省略可能です。`void showWhenOffline(String buttonId, Object element, (optional) String userId)`

エージェントの場合、buttonId の代わりに userId を使用します。**void** showWhenOffline(**String** userId, **Object** element)

ボタンへの代替があるエージェントの場合、両方の ID を使用します (要素はエージェントまたはボタンがオフラインの場合に表示されます)。**void** showWhenOffline(**String** buttonId, **Object** element, **String** userId)

メモ: buttonId と userId の両方を使用する場合は、buttonId を最初に指定する必要があります。

パラメータ

名前	型	説明	使用可能なバージョン
buttonId	String	エージェントがチャットに対応できない場合に、指定した element オブジェクトを表示するチャットボタンの ID。	API バージョン 28.0 以降で使用できます。
element	Object	指定したボタンがオフラインの場合に表示する要素。	API バージョン 28.0 以降で使用できます。
userId	String	ボタンに関連付けるエージェントの ID。このエージェントが対応できない場合に、element オブジェクトが表示されます。	API バージョン 29.0 以降で使用できます。

パラメータにボタン ID を指定してユーザ ID は指定しない場合、要素はボタンがオフラインの場合にのみ表示されます。

ユーザ ID を指定してボタン ID は指定しない場合、要素はエージェントがオフラインの場合にのみ表示されます。たとえば、次の構文は、エージェントのオンライン状況を追跡し、エージェントが対応できない場合はボタンをオフラインに設定します。

```
liveagent.showWhenOffline('005xx000001Sv1m',
document.getElementById('liveagent_button_toAgent_offline'));
```

ボタン ID とエージェント ID を指定する場合、要素はボタンとエージェントのいずれも対応できない場合に表示されます。たとえば、次の構文は、エージェントとボタンの状況を追跡して、いずれも対応できない場合に要素を表示します。

```
liveagent.showWhenOffline('573xx0000000006',
document.getElementById('liveagent_button_offline_573xx0000000006_USER1'),
'005xx000001Sv1m');
```

addButtonEventHandler

addButtonEventHandler メソッドは、特定のイベントが発生したときのチャットボタンの動作を定義するために使用します。API バージョン 28.0 以降で使用できます。

使用方法

次のイベントが発生したときのチャットボタンの動作を定義します。

- チャットに対応できるエージェントがいる。
- チャットに対応できるエージェントがいない。

「チャットに対応できるエージェントがいない」というイベントは、設定したチャットボタンを使用してチャットをエージェントに届けることができない場合に発生します。このイベントは次の場合に発生します。

- エージェントがオンラインになっていない。
- ボタンに関連付けられているスキルに割り当てられたエージェントがオンラインになっていない。
- オンラインのエージェントの状況が [不在] になっている。
- オンラインのエージェントがフル (Live Agent 設定またはオムニチャネルを使用したプレゼンス設定で指定) になっている。
- オンラインのエージェントがオムニチャネルを使用しているが、他のサービスチャネルにのみ対応できる。

構文

```
void addButtonEventHandler(String buttonId, Function callback)
```

パラメータ

名前	型	説明	使用可能なバージョン
`buttonId`	String	特定のイベントが発生したときの動作を定義するチャットボタンの ID。	API バージョン 28.0 以降で使用できます。
`callback`	function	特定のイベントが発生したときにコールする関数。必要なイベントタイプ (ページ 15)それぞれにボタンの動作を指定する必要があります。	API バージョン 28.0 以降で使用できます。

イベントタイプ

次のイベントタイプを `callback` 関数に取り込み、特定のイベントが発生したときのボタンの動作をカスタマイズします。次の各イベントタイプにボタンの動作を指定する必要があります。

関数	イベントタイプ	構文	説明
`callback`	BUTTON_AVAILABLE	`liveagent.BUTTON_EVENT.BUTTON_AVAILABLE`	適切なスキルを持つエージェントがチャットに対応できるときなどの、顧客がエージェントとチャットできる

関数	イベントタイプ	構文	説明
			条件が満たされたときのボタンの動作を指定します。
	BUTTON_UNAVAILABLE	`liveagent.BUTTON_EVENT.BUTTON_UNAVAILABLE`	エージェントがチャットに対応できない場合のボタンの動作を指定します。

`startChat`

`startChat` メソッドは、新しいウィンドウのボタンからチャットを要求するために使用します。

使用方法

新しいウィンドウに表示されたボタンからチャットを要求します。

必要に応じて、特定のボタンから指定の `userId` を持つエージェントにチャットを直接転送できます。エージェントが対応できない場合、ボタンの転送ルールに戻すか(`true`)否か(`false`)を指定して、チャットを他のエージェントに転送できます。

構文

```
void startChat(String buttonId, (optional) String userId, (optional) Boolean fallback)
```

パラメータ

名前	型	説明	使用可能なバージョン
`buttonId`	String	新しいウィンドウでチャットを要求するために使用するチャットボタンの ID。	API バージョン 28.0 以降で使用できます。
(省略可能) `userId`	String	ボタンからチャットを直接転送するエージェントの Salesforce.com ユーザ ID。	API バージョン 29.0 以降で使用できます。
(省略可能) `fallback`	Boolean	指定された `sfdcUserId` のエージェントが対応できない場合、ボタンの転送ルールに戻すか(`true`)否か(`false`)を指定します。	API バージョン 29.0 以降で使用できます。

`startChatWithWindow`

`startChatWithWindow` メソッドは、ウィンドウの名前を使用してボタンからチャットを要求するために使用します。

使用方法

表示されたウィンドウ名を使用して、表示されたボタンからチャットを要求します。API バージョン 28.0 以降で使用できます。

構文

```
void startChatWithWindow(String buttonId, String windowName, (optional) String userId,
(optional) Boolean fallback)
```

パラメータ

名前	型	説明	使用可能なバージョン
`buttonId`	String	新しいウィンドウでチャットを要求するために使用するチャットボタンの ID。	API バージョン 28.0 以降で使用できます。
`windowName`	String	ウィンドウの名前。	API バージョン 28.0 以降で使用できます。
(省略可能) `userId`	String	ボタンからチャットを直接転送するエージェントの Salesforce ユーザ ID。	API バージョン 29.0 以降で使用できます。
(省略可能) `fallback`	Boolean	指定された `sfdcUserId` のエージェントが対応できない場合、ボタンの転送ルールに戻すか (`true`) 否か (`false`) を指定します。	API バージョン 29.0 以降で使用できます。

チャットボタンの対応するコール

ボタン、直接転送エージェント、ボタンへの代替があるエージェントを使用する場合は、コールを調整してチャットが正常に開始されることを確認します。

`startChat` の構文は `startChatWithWindow` にも適用され、`showWhenOnline` の構文は `showWhenOffline` にも適用されます。

ボタン、直接転送エージェント、ボタンへの代替があるエージェントを使用してチャットを作成する場合、次の対応するコールを使用します。

シナリオ	**startChat** (または **startChatWithWindow**) へのコール	**showWhenOnline** (または **showWhenOffline**) へのコール	**addButtonEventHandler** へのコール
ボタン	`startChat(String buttonId)`	`showWhenOnline(String buttonId, Object element, (optional) String userId)`	`addButtonEventHandler(String buttonId, Function callback)`
エージェント (代替なし)	`startChat(String buttonId, String userId, false)`	`showWhenOnline(String userId, Object element)`	`addButtonEventHandler(String userId, Function callback)`
エージェント (ボタンへの代替あり)	`startChat(String buttonId, String userId, true)`	`showWhenOnline(String buttonId, Object element, String userId)`	複数のハンドラを使用します。

リリース API を使用したレコードの自動検索および自動作成

リリース API を使用して、エージェントが顧客とチャットを開始したときにケース、取引先責任者、取引先、リードなどの Salesforce レコードを自動的に検索または作成します。

どのメソッドも、リリースの作成時に自動生成されるコード内で、追加スクリプトとして追加できます。

addCustomDetail

`addCustomDetail` メソッドは、各チャット訪問者のカスタム詳細を追加するために使用します。

findOrCreate

`findOrCreate` メソッドは、特定の条件に基づいて、既存のレコードを検索するか、新しいレコードを作成するために使用します。

setName

`setName` メソッドは、Live Agent コンソールまたは Salesforce コンソールに表示される訪問者名を設定するために使用します。

リリース API を使用したナレッジ記事の検索

リリース API は、顧客が事前チャットフォームに入力した情報に基づいてナレッジ記事を検索するために使用します。

レコード検索および作成のリリース API のコードサンプル

このコードサンプルを使用して、レコードの自動作成が Live Agent リリースでどのように動作するかをテストおよびプレビューします。

addCustomDetail

`addCustomDetail` メソッドは、各チャット訪問者のカスタム詳細を追加するために使用します。

使用方法

チャット訪問者の新しいカスタム詳細を追加します。カスタム詳細は、チャットが有効になっている間、エージェントの Salesforce コンソールのフッターウィジェットおよびチャットの詳細ページに表示されます。API バージョン 28.0 以降で使用できます。

構文

```
addCustomDetail(String label, String value, (optional) Boolean displayToAgent)
```

パラメータ

名前	型	説明	使用可能なバージョン
`label`	String	カスタム詳細の表示ラベル。例: `"Name"`。	API バージョン 28.0 以降で利用できます。
`value`	String	カスタム詳細の値。例: `"John Doe"`.	API バージョン 28.0 以降で利用できます。
(省略可能) `displayToAgent`	Boolean	顧客が事前チャットフォームに入力したカスタム詳細をエージェントに対して表示するか (`true`) 否か (`false`) を指定します。	API バージョン 29.0 以降で利用できます。

findOrCreate

`findOrCreate` メソッドは、特定の条件に基づいて、既存のレコードを検索するか、新しいレコードを作成するために使用します。

使用方法

エージェントがチャット要求を受け入れたときに、指定されたタイプのレコードを検索または作成します。

メモ: `findOrCreate` メソッドは、エージェントが顧客とチャットを開始したときに、既存のレコードの検索または新規レコードの作成を行う API コールを開始します。リリース API を使用してレコードを検索または作成するには、他の `findOrCreate` サブメソッドをコールする前にこのメソッドを使用する必要があります。

API バージョン 29.0 以降で使用できます。

構文

```
liveagent.findOrCreate(String EntityName)
```

パラメータ

名前	型	説明	使用可能なバージョン
`EntityName`	String	エージェントが顧客とのチャットを受け入れたときに、検索または作成するレコードのタイプ。例: 取引先責任者レコード。	API バージョン 29.0 以降で使用できます。

findOrCreate.map

`findOrCreate.map` メソッドは、特定の顧客の詳細を含むレコードを検索または作成するために使用します。

findOrCreate.saveToTranscript

`findOrCreate.saveToTranscript` メソッドは、検索または作成したレコードを、チャットに関連付けられたチャットトランスクリプトに保存するために使用します。

findOrCreate.showOnCreate

`findOrCreate.showOnCreate` メソッドは、作成したレコードを Salesforce コンソールのサブタブで自動的に開くために使用します。

findOrCreate.linkToEntity

`findOrCreate.linkToEntity` メソッドは、検索または作成したレコードを別のレコードタイプにリンクするために使用します。

`findOrCreate.map`

`findOrCreate.map` メソッドは、特定の顧客の詳細を含むレコードを検索または作成するために使用します。

使用方法

`addCustomDetail` リリース API メソッドで指定された顧客データを含むレコードを検索または作成します。このメソッドは、カスタム詳細の値を、Salesforce コンソールの指定レコードの項目に対応付けます。

`findOrCreate.map` メソッドは、適切なレコードを検索するために何度でもコールできます。検索対象の各項目および対応するカスタム詳細値に対してメソッドを 1 回コールします。

メモ: 標準オブジェクトの項目の API 名を検索する方法については、API ドキュメントを参照してください。標準オブジェクト以外の場合、[設定] でオブジェクトの項目詳細を確認します。

API バージョン 29.0 以降で使用できます。

構文

```
liveagent.findOrCreate(Object EntityName).map(String FieldName, String DetailName,
Boolean doFind, Boolean isExactMatch, Boolean doCreate)
```

パラメータ

名前	型	説明	使用可能なバージョン
`FieldName`	String	対応するカスタム詳細 `DetailName` に対応付けるレコード `EntityName` の項目の API 名。	API バージョン 29.0 以降で使用できます。
`DetailName`	String	対応する項目 `FieldName` に対応付けるカスタム詳細の値。	API バージョン 29.0 以降で使用できます。
`doFind`	Boolean	項目 `FieldName` のカスタム詳細 `DetailName` を含むレコードを検索するか(`true`)、否か(`false`)を指定します。	API バージョン 29.0 以降で使用できます。
`isExactMatch`	Boolean	項目 `FieldName` で指定したカスタム詳細 `DetailName` の正確な値を含むレコードを検索するか (`true`)、否か (`false`) を指定します。	API バージョン 29.0 以降で使用できます。
`doCreate`	Boolean	項目 `FieldName` で検出されなかった場合に、カスタム詳細 `DetailName` を持つ新規レコードを作成するか (`true`)、否か (`false`) を指定します。	API バージョン 29.0 以降で使用できます。

findOrCreate.saveToTranscript

`findOrCreate.saveToTranscript` メソッドは、検索または作成したレコードを、チャットに関連付けられたチャットトランスクリプトに保存するために使用します。

使用方法

`findOrCreate` および `findOrCreate.map` リリース API メソッドを使用して検索または作成したレコードを、チャットに関連付けられたチャットトランスクリプトに保存します。

API バージョン 29.0 以降で使用できます。

構文

```
liveagent.findOrCreate(String EntityName).saveToTranscript(String TranscriptFieldName)
```

パラメータ

名前	型	説明	使用可能なバージョン
`TranscriptFieldName`	String	検索または作成したレコードの ID の保存先のチャットトランスクリプトレコードの項目の名前。	API バージョン 29.0 以降で使用できます。

findOrCreate.showOnCreate

`findOrCreate.showOnCreate` メソッドは、作成したレコードを Salesforce コンソールのサブタブで自動的に開くために使用します。

使用方法

`findOrCreate` および `findOrCreate.map` リリース API メソッドを使用して作成したレコードを、Salesforce コンソールのサブタブで自動的に開きます。

API バージョン 29.0 以降で使用できます。

構文

```
liveagent.findOrCreate(String EntityName).showOnCreate()
```

findOrCreate.linkToEntity

`findOrCreate.linkToEntity` メソッドは、検索または作成したレコードを別のレコードタイプにリンクするために使用します。

使用方法

`findOrCreate` および `findOrCreate.map` リリース API メソッドを使用して検索または作成したレコードを、別の `findOrCreate` API コールを使用して作成した、レコードタイプが異なる別のレコードにリンクします。たとえば、組織内で検出したケースレコードを、作成した取引先責任者レコードにリンクできます。

メモ: レコードは、`findOrCreate` API コールを使用して親レコードが作成されている場合にのみリンクできます。子レコードは、`findOrCreate.linkToEntity` メソッドを使用して検出したレコードにリンクできません。

API バージョン 29.0 以降で使用できます。

構文

```
liveagent.findOrCreate(String EntityName).linkToEntity(String EntityName, String FieldName)
```

パラメータ

名前	型	説明	使用可能なバージョン
`EntityName`	String	検索または作成した子レコードのリンク先のレコードタイプ。	API バージョン 29.0 以降で使用できます。
`FieldName`	String	検索または作成した子レコードの ID の保存先のレコード `EntityName` の API 項目名。	API バージョン 29.0 以降で使用できます。

メモ: 標準オブジェクトの項目の API 名を検索する方法については、API ドキュメントを参照してください。標準オブジェクト以外の場合、[設定] でオブジェクトの項目詳細を確認します。

`setName`

`setName` メソッドは、Live Agent コンソールまたは Salesforce コンソールに表示される訪問者名を設定するために使用します。

使用方法

Salesforce コンソールに表示される訪問者名を設定します。この名前は、チャットの主タブ、チャットトランスクリプトが含まれるエージェントのチャットログ、および Live Agent のスーパーバイザパネルに表示されます。API バージョン 28.0 以降で使用できます。

構文

```
setName(String name)
```

パラメータ

名前	型	説明	使用可能なバージョン
`name`	String	Live Agent コンソールまたは Salesforce コンソールに表示する訪問者名。	API バージョン 28.0 以降で使用できます。

リリース API を使用したナレッジ記事の検索

リリース API は、顧客が事前チャットフォームに入力した情報に基づいてナレッジ記事を検索するために使用します。

addCustomDetail.doKnowledgeSearch

`knowledgeSearch` メソッドは、事前チャットフォームの条件に基づいてナレッジ記事を自動的に検索するために使用します。

addCustomDetail.doKnowledgeSearch

knowledgeSearch メソッドは、事前チャットフォームの条件に基づいてナレッジ記事を自動的に検索するために使用します。

使用方法

顧客がエージェントとのチャットを要求すると、事前チャットフォームからカスタム詳細値が取得されます。エージェントがチャット要求を受け入れると、この値は Knowledge One ウィジェットの記事を検索するための検索キーワードとして使用されます。doKnowledgeSearch() メソッドは、addCustomDetail メソッドの value パラメータを使用して検索を実行します。API バージョン 31.0 以降で使用できます。

構文

```
liveagent.addCustomDetail(String label, String value, (optional) Boolean
displayToAgent).doKnowledgeSearch()
```

レコード検索および作成のリリース API のコードサンプル

このコードサンプルを使用して、レコードの自動作成が Live Agent リリースでどのように動作するかをテストおよびプレビューします。

次のコードは、以下のメソッドを使用してエージェントが顧客とチャットを開始したときにレコードを検索および作成します。

- addCustomDetail
- findOrCreate
- findOrCreate.map
- findOrCreate.saveToTranscript
- findOrCreate.linkToEntity
- findOrCreate.showOnCreate

```
<script type='text/javascript'>
/* Creates a custom detail called First Name and sets the value to "Jane" */
liveagent.addCustomDetail("First Name", "Jane");

/* Creates a custom detail called Last Name and sets the value to "Doe" */
liveagent.addCustomDetail("Last Name", "Doe");

/* Creates a custom detail called Phone Number and sets the value to "555-1212" */
liveagent.addCustomDetail("Phone Number", "415-555-1212");

/* Creates a custom detail called Case Subject and sets the value to "Best snowboard for
a beginner" and will perform a knowledge search when the chat is accepted for the agent
*/

liveagent.addCustomDetail("Case Subject", "Best snowboard for a
beginner").doKnowledgeSearch();

/* Creates a custom detail called Case Status and sets the value to "New" */
```

```
liveagent.addCustomDetail("Case Status", "New");

/* This does a non-exact search on cases by the value of the "Case Subject" custom detail.

 If no results are found, it will create a case and set the case's subject and status
 The case that's found or created will be associated to the chat and the case will open
in
 the Console for the agent when the chat is accepted */
liveagent.findOrCreate("Case").map("Subject","Case
Subject",true,false,true).map("Status","Case
Status",false,false,true).saveToTranscript("CaseId").showOnCreate();

/* This searches for a contact whose first and last name exactly match the values in the
custom details for First and Last Name
 If no results are found, it will create a new contact and set it's first name, last name,
 and phone number to the values in the custom details */
liveagent.findOrCreate("Contact").map("FirstName","First
Name",true,true,true).map("LastName","Last Name",true,true,true).map("Phone","Phone
Number",false,false,true);

/* The contact that's found or created will be saved or associated to the chat transcript.
The contact will be opened for the agent in the Console and the case is linked to the
contact record */
liveagent.findOrCreate("Contact").saveToTranscript("ContactId").showOnCreate().linkToEntity("Case","ContactId");
</script>
```

リリース API を使用した自動チャット招待のカスタマイズ

Web サイトで顧客に表示される自動チャット招待をカスタマイズします。

次のリリースメソッドを使用して、自動チャット招待をカスタマイズします。

setCustomVariable

`setCustomVariable` メソッドは、自動招待が顧客に送信されるために満たす必要がある送信ルールに、カスタマイズされた条件を作成するために使用します。

rejectChat

`rejectChat` メソッドは、顧客に送信された招待を却下して撤回するために使用します。

addButtonEventHandler

`addButtonEventHandler` メソッドは、特定のイベントが発生したときの自動招待の動作を定義するために使用します。

自動チャット招待のコードサンプル

このコードサンプルを使用して、Web サイトで自動チャット招待がどのように動作するかをテストおよびプレビューします。

setCustomVariable

`setCustomVariable` メソッドは、自動招待が顧客に送信されるために満たす必要がある送信ルールに、カスタマイズされた条件を作成するために使用します。

使用方法

自動招待が顧客に送信されるために満たす必要がある送信ルールに、カスタマイズされた条件を作成します。送信ルールの条件に使用されるカスタム変数の比較値を指定します。API バージョン 28.0 以降で使用できます。

構文

```
void setCustomVariable(String variableName, Object value)
```

パラメータ

名前	型	説明	使用可能なバージョン
`variableName`	String	カスタム送信ルールのカスタマイズされた条件の名前。	API バージョン 28.0 以降で使用できます。
`value`	Object	カスタム送信ルールの比較値。	API バージョン 28.0 以降で使用できます。

rejectChat

`rejectChat` メソッドは、顧客に送信された招待を却下して撤回するために使用します。

使用方法

招待を却下して撤回します。

API バージョン 28.0 以降で使用できます。

構文

```
void rejectChat(String buttonId)
```

パラメータ

名前	型	説明	使用可能なバージョン
`buttonId`	String	却下するチャットのチャットボタンの ID。	API バージョン 28.0 以降で使用できます。

addButtonEventHandler

addButtonEventHandler メソッドは、特定のイベントが発生したときの自動招待の動作を定義するために使用します。

使用方法

次のイベントが発生したときの招待の動作を定義します。

- 招待が画面に表示されるための条件が満たされる。
- 招待が画面に表示されるための条件が満たされない。
- 顧客がチャットの招待を受諾する。
- 顧客がチャットの招待を却下する。

「招待が画面に表示されるための条件が満たされない」というイベントは、設定したチャットボタンまたは自動招待を使用してチャットをエージェントに届けることができない場合に発生します。このイベントは次の場合に発生します。

- エージェントがオンラインになっていない。
- ボタンに関連付けられているスキルに割り当てられたエージェントがオンラインになっていない。
- オンラインのエージェントの状況が [不在] になっている。
- オンラインのエージェントがフル (Live Agent 設定またはオムニチャネルを使用したプレゼンス設定で指定) になっている。
- オンラインのエージェントがオムニチャネルを使用しているが、他のサービスチャネルにのみ対応できる。

API バージョン 28.0 以降で使用できます。

構文

```
void addButtonEventHandler(String buttonId, Function callback)
```

パラメータ

名前	型	説明	使用可能なバージョン
buttonId	String	特定のイベントが発生したときの動作を定義する自動招待に関連付けられたチャットボタンの ID。	API バージョン 28.0 以降で使用できます。
callback	function	特定のイベントが発生したときにコールする関数。必要なイベントタイプ (ページ 28)それぞれに招待の動作を指定する必要があります。	API バージョン 28.0 以降で使用できます。

イベントタイプ

次のイベントタイプを `callback` 関数に取り込み、特定のイベントが発生したときの招待の動作をカスタマイズします。次のイベントタイプそれぞれに招待の動作を指定する必要があります。

関数	イベントタイプ	構文	説明
`callback`	BUTTON_AVAILABLE	`liveagent.BUTTON_EVENT.BUTTON_AVAILABLE`	招待が画面に表示されるための条件が満たされた場合の自動招待の動作を指定します。
	BUTTON_UNAVAILABLE	`liveagent.BUTTON_EVENT.BUTTON_UNAVAILABLE`	招待が画面に表示されるための条件が満たされていない場合の自動招待の動作を指定します。
	BUTTON_ACCEPTED	`liveagent.BUTTON_EVENT.BUTTON_ACCEPTED`	顧客が招待を受諾した場合の自動招待の動作を指定します。このイベント種別は、自動チャット招待でのみ使用できます。
	BUTTON_REJECTED	`liveagent.BUTTON_EVENT.BUTTON_REJECTED`	顧客が招待を却下した場合の自動招待の動作を指定します。このイベント種別は、自動チャット招待でのみ使用できます。

自動チャット招待のコードサンプル

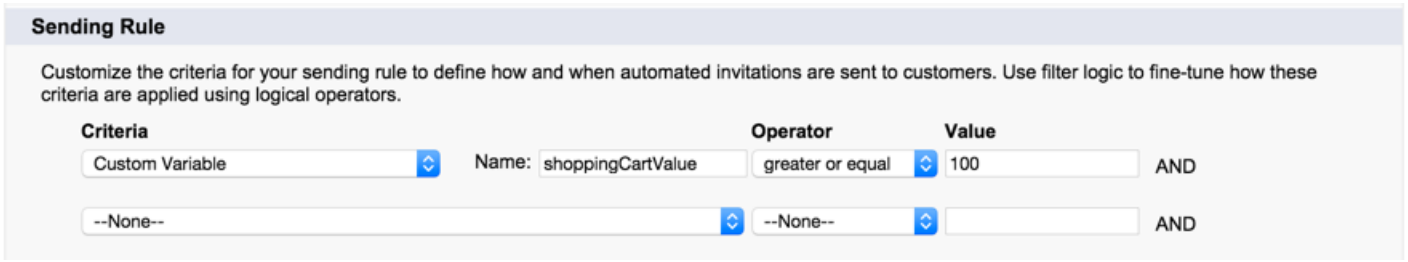
このコードサンプルを使用して、Webサイトで自動チャット招待がどのように動作するかをテストおよびプレビューします。

カスタマイズされた招待を Web サイトに表示するために `addButtonEventHandler()` メソッドを使用する自動チャット招待のコードを次に示します。この招待では、適切なスキルを持つエージェントがチャットに対応できる場合に、顧客がエージェントとチャットを開始できます。

```
<apex:page>

<!-- This section creates the div with the UI for chat invitation whose id is 573D01234567890
 -->
<!-- For this usage, the "Animation" type of this invitation should be set to "Custom",
otherwise two invitations will appear (the Salesforce-provided one and this custom one).
-->
<div id="liveagent_invite_button_573D01234567890" style="display: none; position: fixed;
border: 2px solid darkblue; border-radius: 5px; background-color: lightblue; height: 100px;
 width: 200px;">
 <!-- Creates an "X" option to reject or close the invitation if it's offered  -->
 <div style="cursor: pointer; padding: 5px; right: 0px; position: absolute; color: darkred;
 font-weight: bold;" onclick="liveagent.rejectChat('573D01234567890')">X</div>
<!-- Provides the Start Chat option for the customer to accept or start the chat for the
invitation  -->
<div style="cursor: pointer; top: 42px; left: 65px; position: absolute;font-weight: bold;
 font-size: 16px;" onclick="liveagent.startChat('573D01234567890')">Start Chat</div>
</div>

<!-- Live Agent Deployment Code that makes chat available -->
<script type='text/javascript'
src='https://c.la1s1.saleforceliveagent.com/content/g/js/36.0/deployment.js'></script>

<script type='text/javascript'>
 <!-- Creates the callback function used for the Live Agent chat invitation to present it
 or not based on availability and the customer's interaction with it -->
 function buttonCallback(e) {

  <!-- When the chat invitation is online (i.e. at least one available, skilled agent),
display it at position top 200px and left 300px -->
  if (e == liveagent.BUTTON_EVENT.BUTTON_AVAILABLE) {
  document.getElementById('liveagent_invite_button_573D01234567890').style.display = '';
  document.getElementById('liveagent_invite_button_573D01234567890').style.left = '300px';

  document.getElementById('liveagent_invite_button_573D01234567890').style.top = '200px';

  }

  <!-- When the chat invitation is offline, don't display it -->
  if (e == liveagent.BUTTON_EVENT.BUTTON_UNAVAILABLE) {
  document.getElementById('liveagent_invite_button_573D01234567890').style.display = 'none';

  }

  <!-- When the chat invitation is accepted, stop displaying it -->
  if (e == liveagent.BUTTON_EVENT.BUTTON_ACCEPTED) {
  document.getElementById('liveagent_invite_button_573D01234567890').style.display = 'none';

  }
```

```
  <!-- When the chat invitation is rejected, stop displaying it -->
  if (e == liveagent.BUTTON_EVENT.BUTTON_REJECTED) {
  document.getElementById('liveagent_invite_button_573D01234567890').style.display = 'none';

  }
 }

 <!-- Registers the function buttonCallback() above as the callback for the chat invitation
 whose id is 573D01234567890 -->
 liveagent.addButtonEventHandler('573D01234567890', buttonCallback);

 // Let's say there is data available in JavaScript that you want to use in a custom sending
 rule.
 var shoppingCartValue = 123.45;
 // To pass this data so it can be used in Sending Rules in Salesforce setup, call
setCustomVariable.
 liveagent.setCustomVariable('shoppingCartValue', shoppingCartValue);

 <!-- Live Agent deployment code that initializes chat for the deployment whose id is
572D01234567890 and org is 00DD01234567890 -->
 liveagent.init('https://d.la1s1.salesforceliveagent.com/chat', '572D01234567890',
'00DD01234567890');

 <!-- Enable Live Agent advanced logging to be available through the Browser's Developer
Console -->
 liveagent.enableLogging();
</script>

</apex:page>
```

このコードによって、[設定] の招待の送信ルールで使用できるように JavaScript で使用可能なデータを渡すことができます。設定の例を次に示します。



開発 API のコードサンプル

リリース API がリリースのカスタマイズにどのように役立つかをテストおよびプレビューします。

このコードサンプルでは、次のリリース API メソッドを使用するチャットウィンドウが作成されます。

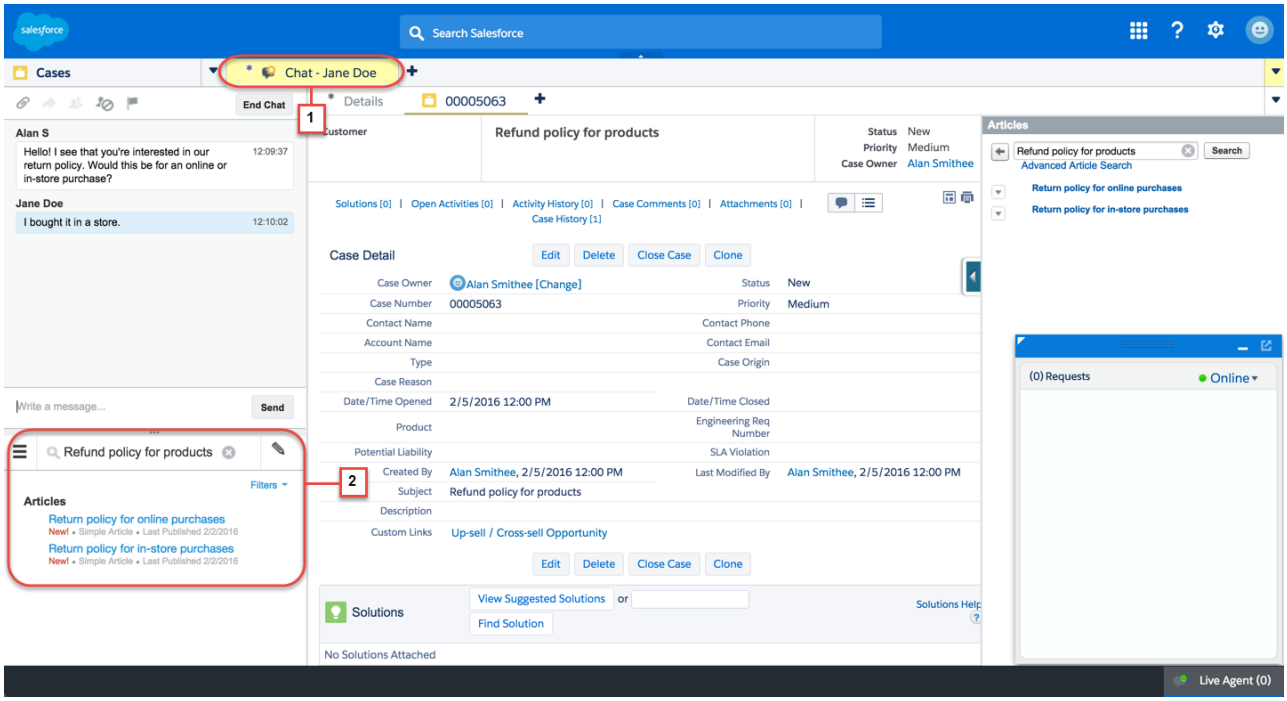
- `startChat`
- `showWhenOnline`
- `showWhenOffline`
- `addCustomDetail`

- setName
- map
- setChatWindowWidth
- setChatWindowHeight
- doKnowledgeSearch

```
<apex:page showHeader="false">
<style> body { margin: 25px 0 0 25px; } </style>

<h1>Welcome to Support</h1>
<br />
Thank you for your interest.
<br /><br />

<!-- START Button code, Replace this section with your Live Agent button code snippet -->
<a id="liveagent_button_online_573B0000000033Y" href="javascript://Chat" style="display:
none;" onclick="liveagent.startChat('573B0000000033Y')">Chat Now</a>
<div id="liveagent_button_offline_573B0000000033Y" style="display: none;">Live Chat is
currently unavailable</div>
<script type="text/javascript">
    if (!window._laq) { window._laq = []; }
    window._laq.push(function(){
        liveagent.showWhenOnline('573B0000000033Y',
document.getElementById('liveagent_button_online_573B0000000033Y'));
        liveagent.showWhenOffline('573B0000000033Y',
document.getElementById('liveagent_button_offline_573B0000000033Y'));
});</script>

<!-- END Button code -->

<!-- Live Agent Deployment Code, replace with your org's values -->
<script type='text/javascript'
src='https://c.la.gus.salesforce.com/content/g/js/36.0/deployment.js'></script>

<!-- Deployment Code API examples -->
<script type='text/javascript'>
/* Adds a custom detail called Contact Email and sets it value to jane@doe.com */
liveagent.addCustomDetail('Contact E-mail', 'jane@doe.com');

/* Creates a custom detail called First Name and sets the value to Jane */
liveagent.addCustomDetail('First Name', 'Jane');

/* Creates a custom detail called Last Name and sets the value to Doe */
liveagent.addCustomDetail('Last Name', 'Doe');

/* Creates a custom detail called Phone Number and sets the value to 415-555-1212 */
liveagent.addCustomDetail('Phone Number', '415-555-1212');

/* An auto-query that searches Contacts whose Email field exactly matches 'jane@doe.com'.
 If no result is found, it will create a Contact record with the email, first name, last
name, and phone number fields set to the custom detail values. */
liveagent.findOrCreate('Contact').map('Email','Contact
E-mail',true,true,true).map('FirstName','First Name',false,false,true).map('LastName','Last
```

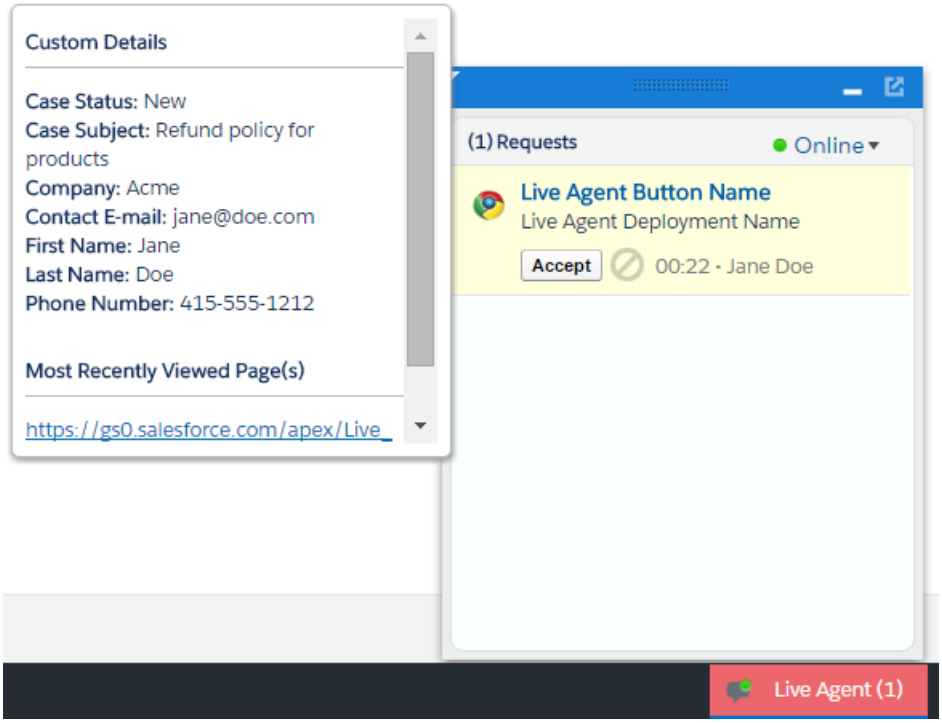
```
 Name',false,false,true).map('Phone','Phone Number',false,false,true);

/* The contact that's found or created will be saved or associated to the chat transcript.
 The contact will be opened for the agent in the Console and the case is linked to the
contact record */
liveagent.findOrCreate('Contact').saveToTranscript('ContactId').showOnCreate().linkToEntity('Case','ContactId');

/* Creates a custom detail called Case Subject and sets the value to 'Refund policy for
products' and will perform a knowledge search when the chat is accepted for the agent */
liveagent.addCustomDetail('Case Subject','Refund policy for products').doKnowledgeSearch();

/* Creates a custom detail called Case Status and sets the value to 'New' */
liveagent.addCustomDetail('Case Status','New');

/* This does a non-exact search on cases by the value of the 'Case Subject' custom detail
 If no results are found, it will create a case and set the case's subject and status.
The case that's found or created will be associated to the chat and the case will open in
 the Console for the agent when the chat is accepted */
liveagent.findOrCreate('Case').map('Subject','Case
Subject',true,false,true).map('Status','Case
Status',false,false,true).saveToTranscript('CaseId').showOnCreate();

/* Saves the custom detail to a custom field on LiveChatTranscript at the end of a chat.
 Assumes a custom field called Company__c was added to the Live Chat Transcript object */
liveagent.addCustomDetail('Company', 'Acme').saveToTranscript('Company__c');

/* For internal or technical details that don't concern the agent, set showToAgent to false
 to hide them from the display. */
liveagent.addCustomDetail('VisitorHash', 'c6f440178d478e4326a1', false);

/* Sets the display name of the visitor in the agent console when engaged in a chat */
liveagent.setName('Jane Doe');

/* Sets the width of the chat window to 500px */
liveagent.setChatWindowWidth(500);

/* Sets the height of the chat window to 500px */
liveagent.setChatWindowHeight(500);

<!-- Live Agent Deployment Code to initialize, replace with your org's values -->
liveagent.init('https://d.la.gus.salesforce.com/chat', '572B000000003KL', '00DB00000000Rr8');
</script>
</apex:page>
```

このコードでは、コンソールを使用するエージェントに次のビューが表示されます。



`setName` (1) に基づいて、コンソールに顧客の名前 (この例では Jane Doe) が表示されます。`addCustomDetail.doKnowledgeSearch` をコールすると、ナレッジウィジェット (2) に検索が自動的に表示されます。

エージェントがチャットを受信すると、設定されたカスタム詳細がフロート表示ウィンドウに表示されます。



第 5 章　事前チャットを使用した訪問者情報の収集とエージェントのコンテキストの設定

訪問者から情報を収集して事前チャット環境をカスタマイズするには、事前チャットフォームを使用します。

事前チャットフォームでは、顧客名、メールアドレス、カスタマーサポートへの問い合わせ理由などの情報を収集できます。この情報を利用すると、チャット要求の転送を効率化し、エージェントの情報収集時間を短縮できます。この情報を使用して、チャットウィンドウに名前を表示するなど、エージェントとチャットするときの顧客の操作性をカスタマイズすることもできます。

事前チャットフォームをホストする Visualforce ページを作成したり、フォームを独自に開発したりできます。このガイドでは、Visualforce の使用に焦点を絞っています。

事前チャット API を使用したレコードの自動検索および自動作成
事前チャット API を使用して、顧客が事前チャットフォームを完了したときに顧客レコードを自動的に検索または作成します。

事前チャット API を使用したチャットの詳細へのアクセス
事前チャット API を使用して、リリース API からカスタム詳細にアクセスし、事前チャットに取り込みます。

事前チャットフォームのコードサンプル
エージェントと顧客で事前チャットフォームがどのように動作するかをテストおよびプレビューします。

事前チャット API を使用したレコードの自動検索および自動作成

事前チャット API を使用して、顧客が事前チャットフォームを完了したときに顧客レコードを自動的に検索または作成します。

findOrCreate.map

`findOrCreate.map` メソッドは、特定の顧客の詳細を含むレコードを検索または作成するために使用します。

findOrCreate.saveToTranscript

`findOrCreate.saveToTranscript` メソッドは、レコードを検索または作成して、チャットに関連付けられたチャットトランスクリプトに保存するために使用します。

findOrCreate.showOnCreate

`findOrCreate.showOnCreate` メソッドは、レコードを検索または作成して、Salesforce コンソールのサブタブで自動的に開くために使用します。

findOrCreate.linkToEntity

`findOrCreate.linkToEntity` メソッドは、検索または作成したレコードを別のレコードタイプにリンクするために使用します。

findOrCreate.displayToAgent

`findOrCreate.displayToAgent` メソッドは、エージェントがチャット要求を受信したときに、受信チャットのエージェントのウィジェットおよび [詳細] タブにどの事前チャット詳細を表示するか指定するために使用します。

レコード検索および作成の事前チャット API のコードサンプル

このコードサンプルを使用して、顧客が事前チャットフォームを完了したときにレコードがどのように自動作成されるかをテストおよびプレビューします。

findOrCreate.map

`findOrCreate.map` メソッドは、特定の顧客の詳細を含むレコードを検索または作成するために使用します。

使用方法

顧客が完了した事前チャットフォームで指定された顧客データを含むレコードを検索または作成します。このメソッドは、カスタム詳細の値を、Salesforce コンソールの指定レコードの項目に対応付けます。

`findOrCreate.map` メソッドは、適切なレコードを検索するために何度でもコールできます。複数の項目とそれに対応する詳細をリストして、レコード内の適切な項目に詳細の値を対応付けることができます。

API バージョン 29.0 以降で使用できます。

構文

```
<input type= "hidden" name= "liveagent.prechat.findorcreate.map: String entityName"
value= "String fieldName, String detailName;" />
```

パラメータ

名前	型	説明	使用可能なバージョン
`entityName`	String	エージェントが顧客とのチャットを受け入れたときに、検索または作成するレコードのタイプ。例: 取引先責任者レコード。	API バージョン 29.0 以降で利用できます。
`fieldName`	String	対応するカスタム詳細 `value` に対応付けるレコード `EntityName` の項目の名前。	API バージョン 29.0 以降で利用できます。
`detailName`	String	対応する項目 `fieldName` に対応付けるカスタム詳細の値。	API バージョン 29.0 以降で利用できます。

findOrCreate.map.doFind

`findOrCreate.map.doFind` メソッドは、顧客が事前チャットフォームを完了したときに既存の顧客レコードに対して検索する項目を指定するために使用します。

findOrCreate.map.isExactMatch

`findOrCreate.map.isExactMatch` メソッドは、`findOrCreate.map` メソッドを使用して検索を実行するときに、項目値が既存のレコードの項目値と完全に一致する必要があるかどうかを指定するために使用します。

findOrCreate.map.doCreate

`findOrCreate.map.doCreate` メソッドは、既存のレコードが見つからない場合にレコードを新規作成するために使用する、`findOrCreate.map` メソッドの項目を指定するために使用します。

findOrCreate.map.doFind

`findOrCreate.map.doFind` メソッドは、顧客が事前チャットフォームを完了したときに既存の顧客レコードに対して検索する項目を指定するために使用します。

使用方法

既存のレコードを検索するために使用する、`findOrCreate.map` メソッドの項目を指定します。レコード内の1つ以上の項目を検索できますが、複数の項目を指定すると論理関係がANDになります。つまり、指定されたすべての項目が既存のレコードと一致することが検出される条件になります。

カスタム項目を使用する場合は、次のガイドラインに従います。

- 関連カスタム項目を検索または作成する場合、チェックボックスには `true` と `false` の有効な値が設定されます。
- 日付形式はYYYY-MM-DDです。
- [通貨]項目の数値には通貨記号を含めません。
- [パーセント]項目の数値にはパーセント記号を含めません。

API バージョン 29.0 以降で使用できます。

構文

```
<input type= "hidden" name= "liveagent.prechat.findorcreate.map.doFind: String
entityName" value= "String fieldName, Boolean find;"/>
```

パラメータ

名前	型	説明	使用可能なバージョン
`entityName`	String	エージェントが顧客とのチャットを受け入れたときに、検索または作成するレコードのタイプ。例: 取引先責任者レコード。	API バージョン 29.0 以降で利用できます。

名前	型	説明	使用可能なバージョン
fieldName	String	既存のレコードで検索する API 項目の名前。 メモ: 標準オブジェクトの項目の API 名を検索する方法については、API ドキュメントを参照してください。標準オブジェクト以外の場合、[設定] でオブジェクトの項目詳細を確認します。	API バージョン 29.0 以降で利用できます。
find	Boolean	項目 fieldName を含む既存のレコードを検索するか (true)、否か (false) を指定します。 メモ: find が true に一致する項目のみを指定できます。メソッドは、find が false に一致する項目を含むレコードは検索しません。	API バージョン 29.0 以降で利用できます。

findOrCreate.map.isExactMatch

findOrCreate.map.isExactMatch メソッドは、findOrCreate.map メソッドを使用して検索を実行するときに、項目値が既存のレコードの項目値と完全に一致する必要があるかどうかを指定するために使用します。

使用方法

既存のレコードを検索するときに項目値が完全に一致する必要がある findOrCreate.map メソッドの項目を指定します。レコード内の 1 つ以上の項目に対して指定できます。

API バージョン 29.0 以降で使用できます。

構文

```
<input type= "hidden" name= "liveagent.prechat.findorcreate.map.isExactMatch: String entityName" value= "String fieldName, Boolean exactMatch;" />
```

パラメータ

名前	型	説明	使用可能なバージョン
`entityName`	String	エージェントが顧客とのチャットを受け入れたときに、検索または作成するレコードのタイプ。例: 取引先責任者レコード。	API バージョン 29.0 以降で利用できます。
`fieldName`	String	既存のレコードで検索する項目の API 名。 メモ: 標準オブジェクトの項目の API 名を検索する方法については、API ドキュメントを参照してください。標準オブジェクト以外の場合、[設定] でオブジェクトの項目詳細を確認します。	API バージョン 29.0 以降で利用できます。
`find`	Boolean	完全に一致する項目 `fieldName` を含む既存のレコードを検索するか (`true`)、否か (`false`) を指定します。 メモ: 指定する必要があるのは、`exactMatch` が `true` に一致する項目のみです。メソッドは、`exactMatch` が `false` に一致する項目を含むレコードは検索しません。	API バージョン 29.0 以降で利用できます。

findOrCreate.map.doCreate

`findOrCreate.map.doCreate` メソッドは、既存のレコードが見つからない場合にレコードを新規作成するために使用する、`findOrCreate.map` メソッドの項目を指定するために使用します。

使用方法

既存のレコードが見つからない場合にレコードを新規作成するために使用する、`findOrCreate.map` メソッドの項目を指定します。レコードを新規作成するには 1 つ以上の項目を指定できます。

API バージョン 29.0 以降で使用できます。

構文

```
<input type= "hidden" name= "liveagent.prechat.findorcreate.map.doCreate: String entityName" value= "String fieldName, Boolean create;" />
```

パラメータ

名前	型	説明	使用可能なバージョン
`entityName`	String	エージェントが顧客とのチャットを受け入れたとき、既存のレコードが見つからない場合に作成するレコードのタイプ。例: 取引先責任者レコード。	API バージョン 29.0 以降で利用できます。
`fieldName`	String	新規レコードに含める項目の API 名。 メモ: `findOrCreate` メソッドは、エージェントが顧客とチャットを開始したときに、既存のレコードの検索または新規レコードの作成を行う API コールを開始します。リリース API を使用してレコードを検索または作成するには、他の `findOrCreate` サブメソッドをコールする前にこのメソッドを使用する必要があります。	API バージョン 29.0 以降で利用できます。
`create`	Boolean	項目 `fieldName` を含む新規レコードを作成するか (`true`)、否か (`false`) を指定します。 メモ: 指定する必要があるのは、`create` が `true` に一致する項目のみです。このメソッドは、`create` が `false` に一致する項目は作成しません。	API バージョン 29.0 以降で利用できます。

`findOrCreate.saveToTranscript`

`findOrCreate.saveToTranscript` メソッドは、レコードを検索または作成して、チャットに関連付けられたチャットトランスクリプトに保存するために使用します。

使用方法

`findOrCreate.map.doCreate` または `findOrCreate.map.doFind` 事前チャット API メソッドを使用して検索または作成したレコードを、チャット終了時に、チャットに関連付けられたチャットトランスクリプトに保存します。

API バージョン 29.0 以降で使用できます。

構文

```
<input type="hidden" name= "liveagent.prechat.findorcreate.saveToTranscript: String entityName" value= "String transcriptFieldName" />
```

パラメータ

名前	型	説明	使用可能なバージョン
entityName	String	エージェントが顧客とのチャットを受け入れたときに、検索または作成するレコードのタイプ。例:取引先責任者レコード。	API バージョン 29.0 以降で利用できます。
transcriptFieldName	String	検索または作成したレコードの ID の保存先のチャットトランスクリプトレコードの項目の API 名。	API バージョン 29.0 以降で利用できます。

メモ: 標準オブジェクトの項目の API 名を検索する方法については、API ドキュメントを参照してください。標準オブジェクト以外の場合、[設定] でオブジェクトの項目詳細を確認します。

findOrCreate.showOnCreate

findOrCreate.showOnCreate メソッドは、レコードを検索または作成して、Salesforce コンソールのサブタブで自動的に開くために使用します。

使用方法

findOrCreate.map.doCreate および findOrCreate.map.doFind 事前チャット API メソッドを使用して検索または作成したレコードを、Salesforce コンソールのサブタブで自動的に開きます。

API バージョン 29.0 以降で使用できます。

構文

```
<input type= "hidden" name= "liveagent.prechat.findorcreate.showOnCreate: String entityName" value= "Boolean show" />
```

パラメータ

名前	型	説明	使用可能なバージョン
`entityName`	String	エージェントが顧客とのチャットを受け入れたときに、検索または作成するレコードのタイプ。例: 取引先責任者レコード。	API バージョン 29.0 以降で利用できます。
`show`	Boolean	作成したレコードを Salesforce コンソールのサブタブで表示するか (`true`) 否か (`false`) を指定します。	API バージョン 29.0 以降で利用できます。

`findOrCreate.linkToEntity`

`findOrCreate.linkToEntity` メソッドは、検索または作成したレコードを別のレコードタイプにリンクするために使用します。

使用方法

`findOrCreate.map.doFind` および `findOrCreate.map.doCreate` メソッドを使用して検索または作成したレコードを、別の `findOrCreate.map` API コールを使用して作成した、レコードタイプが異なる別のレコードにリンクします。たとえば、組織内で検出したケースレコードを、作成した取引先責任者レコードにリンクできます。

`findOrCreate` API コールを使用して作成したレコードの項目は、`findOrCreate.linkToEntity` メソッドを使用して入力することができません。代わりに、`findOrCreate.map` メソッドを使用して、レコードの項目値を更新します。

メモ: レコードは、`findOrCreate` API コールを使用して親レコードが作成されている場合にのみリンクできます。子レコードは、`findOrCreate.linkToEntity` メソッドを使用して検出したレコードにリンクできません。

API バージョン 29.0 以降で使用できます。

構文

```
<input type= "hidden" name= "liveagent.prechat.findorcreate.linkToEntity: String entityName" value= "String parentEntityName, String fieldName" />
```

パラメータ

名前	型	説明	使用可能なバージョン
`entityName`	String	検索または作成した親レコードにリンクされるレコードタイプ。	API バージョン 29.0 以降で利用できます。

名前	型	説明	使用可能なバージョン
parentEntityName	String	検索または作成した子レコードにリンクする親レコードタイプ。	API バージョン 29.0 以降で利用できます。
fieldName	String	検索または作成した子レコードの ID が保存されるレコード parentEntityName の項目名。	API バージョン 29.0 以降で利用できます。

findOrCreate.displayToAgent

findOrCreate.displayToAgent メソッドは、エージェントがチャット要求を受信したときに、受信チャットのエージェントのウィジェットおよび[詳細]タブにどの事前チャット詳細を表示するか指定するために使用します。

使用方法

エージェントがチャット要求を受信したときに、エージェントの Salesforce コンソールの[詳細]タブにどの事前チャット詳細を表示するか指定します。通常、この方法は、値を false に設定することにより、特定のカスタム詳細をエージェントに表示しないように使用されます。

API バージョン 29.0 以降で使用できます。

構文

```
<input type= "hidden" name= "liveagent.prechat.findorcreate.displayToAgent: String detailName" value= "Boolean display" />
```

パラメータ

名前	型	説明	使用可能なバージョン
detailName	String	エージェントがチャット要求を受信したときにエージェントに対して表示される詳細の名前。	API バージョン 29.0 以降で利用できます。
display	Boolean	カスタム詳細をエージェントのチャット通知および[詳細]タブに表示するか (true)、否か (false) を指定します。 メモ: 指定する必要があるのは、display が false に一致する詳細のみです。このメソッドは、display が false に一致する詳細は表示しません。display パラメータの値を指定	API バージョン 29.0 以降で利用できます。

名前	型	説明	使用可能なバージョン
		しない場合、デフォルト値は `true` に設定されます。	

レコード検索および作成の事前チャット API のコードサンプル

このコードサンプルを使用して、顧客が事前チャットフォームを完了したときにレコードがどのように自動作成されるかをテストおよびプレビューします。

次のコードは、以下のメソッドを使用して顧客が事前チャットフォームを完了したときにレコードを検索および作成します。

- `findOrCreate.map`
- `findOrCreate.map.doFind`
- `findOrCreate.map.isExactMatch`
- `findOrCreate.map.doCreate`
- `findOrCreate.saveToTranscript`
- `findOrCreate.showOnCreate`
- `findOrCreate.linkToEntity`

```
<form method="post" action="#">
<label>First Name: </label> <input type='text' name='liveagent.prechat:ContactFirstName'
/><br />
<label>Last Name: </label> <input type='text' name='liveagent.prechat:ContactLastName'
/><br />
<label>Subject: </label> <input type='text' name='liveagent.prechat:CaseSubject' /><br />
<input type="hidden" name="liveagent.prechat:CaseStatus" value="New" /><br />
<input type="hidden" name="liveagent.prechat.findorcreate.map:Contact"
value="FirstName,ContactFirstName;LastName,ContactLastName" />
<input type="hidden" name="liveagent.prechat.findorcreate.map.doFind:Contact"
value="FirstName,true;LastName,true" />
<input type="hidden" name="liveagent.prechat.findorcreate.map.isExactMatch:Contact"
value="FirstName,true;LastName,true" />
<input type="hidden" name="liveagent.prechat.findorcreate.map.doCreate:Contact"
value="FirstName,true;LastName,true" />
<input type="hidden" name="liveagent.prechat.findorcreate.saveToTranscript:Contact"
value="ContactId" />
<input type="hidden" name="liveagent.prechat.findorcreate.showOnCreate:Contact" value="true"
 />
<input type="hidden" name="liveagent.prechat.findorcreate.linkToEntity:Contact"
value="Case,ContactId" />
<input type="hidden" name="liveagent.prechat.findorcreate.map:Case"
value="Subject,CaseSubject;Status,CaseStatus" />
<input type="hidden" name="liveagent.prechat.findorcreate.map.doCreate:Case"
value="Subject,true;Status,true" />
<input type="submit" value="Submit" />
</form>
```

事前チャット API を使用したチャットの詳細へのアクセス

事前チャット API を使用して、リリース API からカスタム詳細にアクセスし、事前チャットに取り込みます。

preChatInit
preChatInit メソッドは、addCustomDetail リリース API メソッドを使用してチャットに渡されたリリース情報にアクセスするために使用します。

preChatInit

preChatInit メソッドは、addCustomDetail リリース API メソッドを使用してチャットに渡されたリリース情報にアクセスするために使用します。

使用方法

事前チャットで使用するカスタム詳細などのチャットリリース情報を抽出します。preChatInit を使用する場合、prechat.js ファイルを deployment.js ファイルと同じパスの同じ Apex ページに含めます。

API バージョン 29.0 以降で使用できます。

構文

```
liveagent.details.preChatInit(String chatUrl, function detailCallback, (optional) String chatFormName)
```

パラメータ

名前	型	説明	使用可能なバージョン
chatUrl	String	カスタム詳細を取得するチャットの URL。	API バージョン 29.0 以降で使用できます。
detailCallback	String	メソッドの完了時にコールされる JavaScript 関数の名前。	API バージョン 29.0 以降で使用できます。
(省略可能) chatFormName	String	カスタム詳細を取り込む事前チャットフォームの HTML form タグの名前。	API バージョン 29.0 以降で使用できます。

応答

名前	型	説明	使用可能なバージョン
details	Object	preChatInit メソッドを使用して事前チャットフォームに格納されたリリース情報を含むオブジェクト。	API バージョン 29.0 以降で使用できます。

detailCallback

detailCallback メソッドは、preChatInit メソッドが details オブジェクトを返した後に発生する動作を指定します。

構文	パラメータ	説明	使用可能なバージョン
function myCallBack(details) { // Customer specific code }	details	preChatInit メソッドを使用してカスタム詳細が取得された後に実行されるアクションを指定します。	API バージョン 29.0 以降で使用できます。

事前チャットフォームのコードサンプル

エージェントと顧客で事前チャットフォームがどのように動作するかをテストおよびプレビューします。

次の操作を行う事前チャットフォームのコードを次に示します。

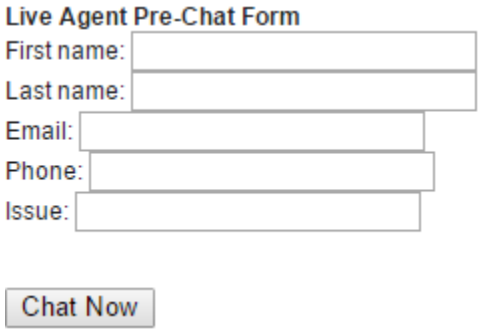
- 訪問者の名と姓、メールアドレス、電話番号、問題を収集し、値をカスタム詳細に設定する。
- メールが顧客がフォームで入力した値と完全一致する取引先責任者を検索する。一致するものがない場合は、取引先責任者レコードが作成されて、顧客が入力した値がレコードの項目に対応付けられます。
- ケースレコードを作成してチャットに添付し、検出された詳細をレコードの項目に対応付ける。
- 検出または作成された取引先責任者レコードをケースの取引先責任者として設定する。
- レコードの取引先責任者およびケースを Live Chat トランスクリプトレコードに関連付け、エージェントのコンソールで開く。
- テキストに基づいてナレッジ記事を検索する (ナレッジが設定されている場合)。
- エージェントのコンソールに訪問者の名と姓を表示する。

```
<apex:page showHeader="false">

<!-- This script takes the endpoint URL parameter passed from the deployment page and makes
 it the action for the form -->
<script type='text/javascript'>
(function() {
function handlePageLoad() {
var endpointMatcher = new RegExp("[\\?\\&]endpoint=([^&#]*)");
```

```
document.getElementById('prechatForm').setAttribute('action',
decodeURIComponent(endpointMatcher.exec(document.location.search)[1]));
} if (window.addEventListener) {
window.addEventListener('load', handlePageLoad, false);
} else { window.attachEvent('onload', handlePageLoad, false);
}})();
</script>

<h1>Live Agent Pre-Chat Form</h1>

<!-- Form that gathers information from the chat visitor and sets the values to Live Agent
 Custom Details used later in the example -->
<form method='post' id='prechatForm'>
    First name: <input type='text' name='liveagent.prechat:ContactFirstName' id='firstName'
 /><br />
    Last name: <input type='text' name='liveagent.prechat:ContactLastName' id='lastName'
/><br />
    Email: <input type='text' name='liveagent.prechat:ContactEmail' id='email' /><br />
    Phone: <input type='text' name='liveagent.prechat:ContactPhone' id='phone' /><br />
    Issue: <input type='text' name='liveagent.prechat:CaseSubject' id='subject' /><br />

<!-- Hidden fields used to set additional custom details -->
    <input type="hidden" name="liveagent.prechat:CaseStatus" value="New" /><br />

    <!-- This example assumes that "Chat" was added as picklist value to the Case Origin
field -->
    <input type="hidden" name="liveagent.prechat:CaseOrigin" value="Chat" /><br />

    <!-- This example will set the Case Record Type to a specific value for the record
type configured on the org. Lookup the case record type's id on your org and set it here
-->
    <input type="hidden" name="liveagent.prechat:CaseRecordType" value="012D0000000Eyni"
/>

    <!-- Used to set the visitor's name for the agent in the Console -->
    <input type="hidden" name="liveagent.prechat.name" id="prechat_field_name" />

<!-- map: Use the data from prechat form to map it to the Salesforce record's fields -->
<input type="hidden" name="liveagent.prechat.findorcreate.map:Contact"
value="FirstName,ContactFirstName;LastName,ContactLastName;Email,ContactEmail;Phone,ContactPhone"
 />

<input type="hidden" name="liveagent.prechat.findorcreate.map:Case"
value="Subject,CaseSubject;Status,CaseStatus;Origin,CaseOrigin" />

<!-- doFind, doCreate and isExactMatch example for a Contact:
    Find a contact whose Email exactly matches the value provided by the customer in the
form
    If there's no match, then create a Contact record and set it's First Name, Last Name,
 Email, and Phone to the values provided by the customer -->
<input type="hidden" name="liveagent.prechat.findorcreate.map.doFind:Contact"
value="Email,true" />
<input type="hidden" name="liveagent.prechat.findorcreate.map.isExactMatch:Contact"
value="Email,true" />
```

```
<input type="hidden" name="liveagent.prechat.findorcreate.map.doCreate:Contact"
value="FirstName,true;LastName,true;Email,true;Phone,true" />

<!-- doCreate example for a Case: create a case to attach to the chat, set the Case Subject
 to the value provided by the customer and set the case's Status and Origin fields -->
<input type="hidden" name="liveagent.prechat.findorcreate.map.doCreate:Case"
value="Subject,true;Status,true;Origin,true" />

<!-- linkToEntity: Set the record Contact record, found/created above, as the Contact on
the Case that's created -->
<input type="hidden" name="liveagent.prechat.findorcreate.linkToEntity:Contact"
value="Case,ContactId" />

<!-- showOnCreate: Open the Contact and Case records as sub-tabs to the chat for the agent
 in the Console -->
<input type="hidden" name="liveagent.prechat.findorcreate.showOnCreate:Contact" value="true"
 />
<input type="hidden" name="liveagent.prechat.findorcreate.showOnCreate:Case" value="true"
 />

<!-- saveToTranscript: Associates the records found / created, i.e. Contact and Case, to
the Live Chat Transcript record. -->
<input type="hidden" name="liveagent.prechat.findorcreate.saveToTranscript:Contact"
value="ContactId" />
<input type="hidden" name="liveagent.prechat.findorcreate.saveToTranscript:Case"
value="CaseId" />

<!-- displayToAgent: Hides the case record type from the agent -->
<input type="hidden" name="liveagent.prechat.findorcreate.displayToAgent:CaseRecordType"
value="false" />

<!-- searchKnowledge: Searches knowledge article based on the text, this assumes that
Knowledge is setup -->
<input type="hidden" name="liveagent.prechat.knowledgeSearch:CaseSubject" value="true" />

<input type='submit' value='Chat Now' id='prechat_submit' onclick="setName()"/>

<!-- Set the visitor's name for the agent in the Console to first and last name provided
by the customer -->
<script type="text/javascript">
   function setName() {
    document.getElementById("prechat_field_name").value =
        document.getElementById("firstName").value + " " +
document.getElementById("lastName").value;
    }
</script>

<style type="text/css">
p {font-weight: bolder }
</style>

</form>
</apex:page>
```

このコードの結果は、次のような事前チャットフォームになります。


Live Agent Pre-Chat Form
First name:
Last name:
Email:
Phone:
Issue:
Chat Now

第 6 章　Visualforce を使用したカスタムチャットウィンドウの実装

チャットウィンドウは、訪問者がサポートエージェントとメッセージを交換するために使用されます。各 Live Agent リリースにはチャットウィンドウが含まれます。Visualforce を使用してカスタマイズしたチャットウィンドウを作成し、HTML、CSS、および JavaScript を使用してスタイルや機能を追加できます。

チャットウィンドウをカスタマイズする場合、Salesforce CSS スタイルシートにリンクしないでください。これらはバージョン設定されていないため、通知なしで変更できます。代わりに、スタイルシートを直接参照せずに Salesforce スタイルを模倣する Visualforce コンポーネントを使用することをお勧めします。こうすることで、チャットウィンドウの外観を常に制御できます。スタイルシートを無効にする方法については、*「Salesforce スタイルシートのスタイルの使用」*を参照してください。

Visualforce の使用についての詳細は、*『Visualforce 開発者ガイド』*を参照してください。

Live Agent Visualforce コンポーネント
Visualforce コンポーネントを使用して、チャットウィンドウの外観と動作をカスタマイズします。

Live Agent Visualforce コンポーネントのコードサンプル
このコードサンプルを使用して、Visualforce コンポーネントがチャットウィンドウのカスタマイズにどのように役立つかをテストおよびプレビューします。

Live Agent Visualforce コンポーネント

Visualforce コンポーネントを使用して、チャットウィンドウの外観と動作をカスタマイズします。

Live Agent には、カスタマイズ可能な次の Visualforce コンポーネントが含まれます。これらのコンポーネントを Visualforce フォームに配置して、特定の機能を使用できるようにしたり、チャットウィンドウの外観をカスタマイズしたりします。

チャットウィンドウには、チャット訪問者にチャットの進行状況を通知するさまざまな状態が表示されます。次のコンポーネントでは、これらの状態のチャットウィンドウの外観と動作をカスタマイズして、訪問者の最適なサポートを実現できます。

コンポーネント名	説明
`liveAgent:clientChat`	Live Agent のチャットウィンドウの主要な親要素です。Live Agent の追加のカスタマイズを行うには、この要素が必要になります。このコンポーネントは Live Agent リリースで一度のみ使用できます。
`liveAgent:clientChatAlertMessage`	システムアラートメッセージ（「You have been disconnected」（切断されました）など）を表示する Live Agent のチャットウィンドウ内の領域です。
`liveAgent:clientChatMessages`	システム状況メッセージ（「Chat session has been disconnected」（チャットセッションが切断されました）など）を表示する Live Agent のチャッ

コンポーネント名	説明
	トウィンドウ内の領域です。`liveAgent:clientChat` 内で使用する必要があります。各チャットウィンドウに作成できるメッセージ領域は 1 つのみです。
`liveAgent:clientChatStatusMessage`	システム状況メッセージ (「You are being reconnected」(再接続されます) など) を表示する Live Agent のチャットウィンドウ内の領域です。
`liveAgent:clientChatQueuePosition`	プッシュ転送を使用するボタンで開始されるチャットセッションのキュー内の訪問者の位置を示すテキストラベルです (プル転送を使用するボタンでは、このコンポーネントは無効です)。`liveAgent:clientChat` 内で使用する必要があります。このコンポーネントについての詳細は、「`liveAgent:clientChatQueuePosition` の使用」を参照してください。
`liveAgent:clientChatCancelButton`	チャットが待機状態になったときに訪問者がチャットをキャンセルできるようにする Live Agent のチャットウィンドウ内のボタンです。`liveAgent:clientChat` 内で使用する必要があります。
`liveAgent:clientChatSaveButton`	訪問者がチャットトランスクリプトをローカルファイルとして保存するためにクリックする Live Agent のチャットウィンドウ内のボタンです。`liveAgent:clientChat` 内で使用する必要があります。各チャットウィンドウに複数の保存ボタンを作成できます。
`liveAgent:clientChatEndButton`	訪問者がチャットセッションを終了するためにクリックする Live Agent のチャットウィンドウ内のボタンです。`liveAgent:clientChat` 内で使用する必要があります。
`liveAgent:clientChatLog`	訪問者にチャットの会話を表示する Live Agent のチャットウィンドウ内の領域です。`liveAgent:clientChat` 内で使用する必要があります。各チャットウィンドウに作成できるチャットログは 1 つのみです。
`liveAgent:clientChatInput`	訪問者がサポートエージェントへのメッセージを入力する Live Agent のチャットウィンドウ内のテキストボックスです。`liveAgent:clientChat` 内で使用する必要があります。各チャットウィンドウに作成できる入力ボックスは 1 つのみです。
`liveAgent:clientChatSendButton`	訪問者がエージェントにチャットメッセージを送信するためにクリックする Live Agent のチャットウィンドウ内のボタンです。`liveAgent:clientChat` 内で使用する必要があります。各チャットウィンドウに複数の送信ボタンを作成できます。
`liveAgent:clientChatLogAlertMessage`	訪問者にアイドルタイムアウトアラート (顧客警告) を表示する Live Agent のチャットウィンドウ内の領域です。

コンポーネント名	説明
liveAgent:clientChatFileTransfer	訪問者がエージェントにファイルを送信できるLive Agentのチャットウィンドウ内のファイルアップロード領域です。liveAgent:clientChat 内で使用する必要があります。

各コンポーネントについての詳細は、「標準のコンポーネントの参照」を参照してください。

liveAgent:clientChatQueuePosition の使用

liveAgent:clientChatQueuePosition コンポーネントは、チャットキュー内の訪問者の位置を示します。チャットがキューに入る条件は次のとおりです。

- チャットの要求元のボタンでキューが有効になっている。
- (該当する場合は関連するスキルを持つ) すべてのオンラインのエージェントが対応中であるため、キューが形成されている。
- チャットがキューに入り、エージェントに割り当てられていない。

これらの3つの条件が満たされていない場合、liveAgent:clientChatQueuePosition は値を示しません。

Live Agent Visualforce コンポーネントのコードサンプル

このコードサンプルを使用して、Visualforceコンポーネントがチャットウィンドウのカスタマイズにどのように役立つかをテストおよびプレビューします。

次のコードサンプルでは、次のコンポーネントを使用するチャットウィンドウを示します。

- liveAgent:clientChat
- liveAgent:clientChatMessages
- liveAgent:clientChatEndButton
- liveAgent:clientChatLog
- liveAgent:clientChatInput

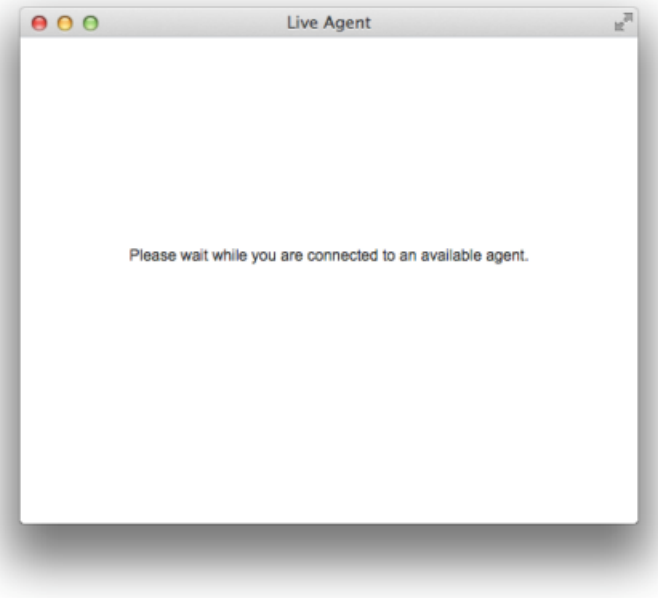
```
<apex:page showHeader="false">
<style>
#liveAgentClientChat.liveAgentStateWaiting {
// The CSS class that is applied when the chat request is waiting to be accepted
// See "Waiting State" screenshot below
}
#liveAgentClientChat {
// The CSS class that is applied when the chat is currently engaged
// See "Engaged State" screenshot below
}
#liveAgentClientChat.liveAgentStateEnded {
// The CSS class that is applied when the chat has ended
// See "Ended State" screenshot below
}
body { overflow: hidden; width: 100%; height: 100%; padding: 0; margin: 0 }
```

```
#waitingMessage { height: 100%; width: 100%; vertical-align: middle; text-align: center;
display: none; }
#liveAgentClientChat.liveAgentStateWaiting #waitingMessage { display: table; }
#liveAgentSaveButton, #liveAgentEndButton { z-index: 2; }
.liveAgentChatInput {
height: 25px;
border-width: 1px;
border-style: solid;
border-color: #000;
padding: 2px 0 2px 4px;
background: #fff;
display: block;
width: 99%;
}
.liveAgentSendButton {
display: block;
width: 60px;
height: 31px;
padding: 0 0 3px;
position: absolute;
top: 0;
right: -67px;
}
#liveAgentChatLog {
width: auto;
height: auto;
top: 0px;
position: absolute;
overflow-y: auto;
left: 0;
right: 0;
bottom: 0;
}
</style>
<div style="top: 0; left: 0; right: 0; bottom: 0; position: absolute;">
43
<liveAgent:clientChatSaveButton />
<liveAgent:clientChatEndButton />
<div style="top: 25px; left: 5px; right: 5px; bottom: 5px; position: absolute; z-index:
0;">
<liveAgent:clientChatAlertMessage />
<liveAgent:clientChatStatusMessage />
<table id="waitingMessage" cellpadding="0" cellspacing="0">
<tr>
<td>Please wait while you are connected to an available agent.</td>
</tr>
</table>
<div style="top: 0; right: 0; bottom: 41px; left: 0; padding: 0; position: absolute;
word-wrap: break-word; z-index: 0;">
<liveAgent:clientChatLog />
</div>
<div style="position: absolute; height: auto; right: 0; bottom: 0; left: 0; margin-right:
67px;">
<liveagent:clientChatInput /><liveAgent:clientChatSendButton />
```

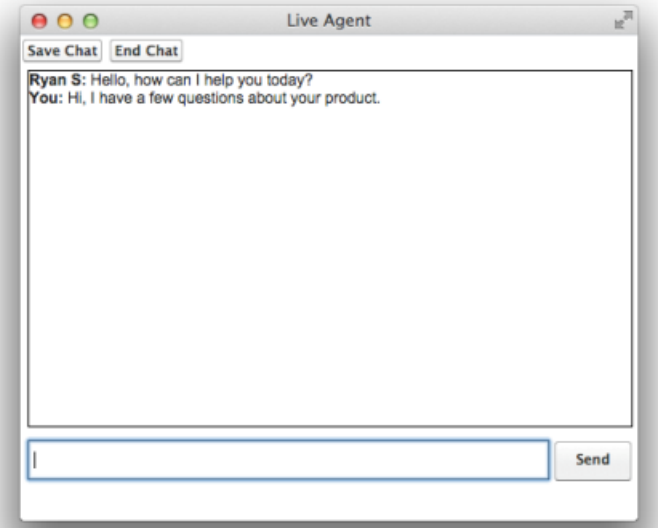
```
</div>
</div>
</liveAgent:clientchat>
</div>
</apex:page>
```

有効なチャットの場合、このコードでは次のようなチャットウィンドウの状態になります。

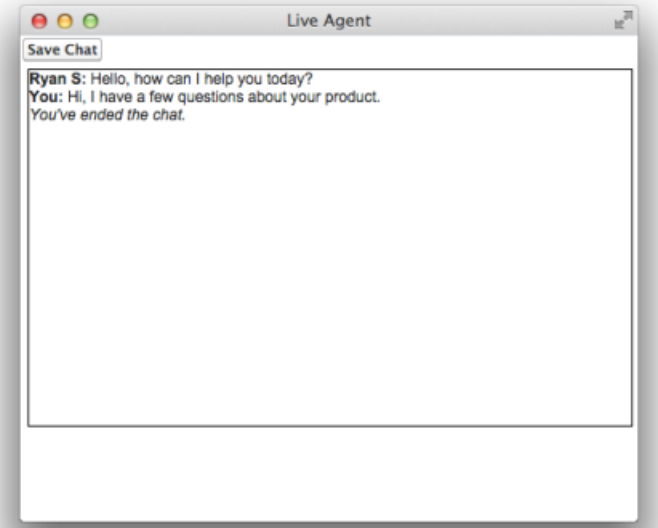
待機中のチャット



進行中のチャット



終了したチャット



第 7 章　事後チャットを使用した顧客とのチャットのやりとりのまとめ

事後チャットページを利用すると、チャットセッションの終了時に顧客と情報を共有できます。たとえば、エージェントとのチャットを終了した顧客に別の Web ページや、チャットの操作性に関するアンケートを表示することができます。

事後チャットページをホストする Visualforce ページを作成したり、ページを独自に開発して URL をチャットボタン設定に追加したりできます。このガイドでは、Visualforce の使用に焦点を絞っています。

事後チャットのコードサンプル
このコードサンプルを使用して、エージェントと顧客で事後チャットページがどのように動作するかをテストおよびプレビューします。

事後チャットのコードサンプル

このコードサンプルを使用して、エージェントと顧客で事後チャットページがどのように動作するかをテストおよびプレビューします。

表示する変数を含めて、事後チャットページをカスタマイズできます。

使用可能な変数	説明
`requestTime`	システムでチャット要求を受信したときのタイムスタンプ。
`startTime`	エージェントがチャットを受け入れたときのタイムスタンプ。
`deploymentId`	リリースの ID。
`buttonId`	チャットを開始したボタンの ID。
`chatKey`	一意のチャットキー。
`lastVisitedPage`	エージェントに送信される前回アクセスしたページの値。
`originalReferrer`	リリースコードが含まれる、顧客がアクセスした最初のページ。
`latitude`	チャット訪問者の地理的位置 (緯度)。
`longitude`	チャット訪問者の地理的位置 (経度)。
`city`	チャット訪問者の地理的位置 (市区郡)。
`region`	チャット訪問者の地理的位置 (地域)。

使用可能な変数	説明
country	チャット訪問者の地理的位置 (国)。
organization	チャットをホストした Salesforce 組織 ID。
disconnectedBy	チャットを終了した理由。使用可能な値: • agent - エージェントがチャットを終了した • client - チャット訪問者がチャットを終了した • error - システムでエラーが発生してチャットが切断された • clientIdleTimeout - 割り当てられた時間内にチャット訪問者が応答しなかった(アイドルタイムアウトが設定されている必要がある) • agentsUnavailable - チャットを受信できるエージェントがいないか、キューに空きがない
windowLanguage	チャットボタンに設定されたウィンドウの言語。
chatDetails	チャットデータの JSON 表現。
transcript	トランスクリプトのプレーンテキストコピー。
attachedRecords	JSON 配列形式のチャットセッションに割り当てられた ID のリスト。
error	チャット中に発生したエラーの説明。

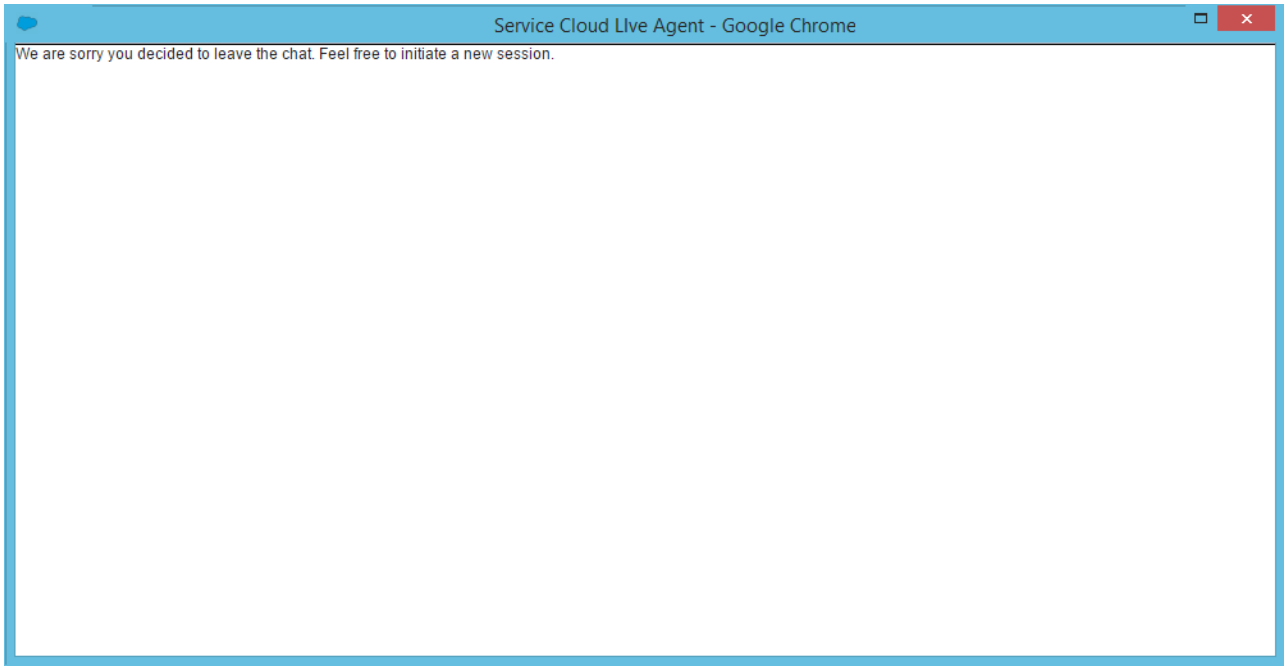
次のコードサンプルでは、チャットに関する基本情報を含む事後チャットページが作成されます。

```
<apex:page showHeader='false'>
      <div id='details'>
      <!-- This will present all the post chat parameters available to this page -->
            <h1>Post Chat Page</h1>
            <p>
            <!-- These variables are passed to the post-chat page and can be used to
customize your post-chat experience -->
                  Request Time:  <apex:outputText id='c_rt'
value='{!$CurrentPage.parameters.requestTime}' /><br/>
                  Start Time:  <apex:outputText id='c_st'
value='{!$CurrentPage.parameters.startTime}' /><br/>
                  Deployment Id: <apex:outputText
value='{!$CurrentPage.parameters.deploymentId}' /><br/>
                  Button Id: <apex:outputText value='{!$CurrentPage.parameters.buttonId}'
 /><br/>
                  Chat Key: <apex:outputText value='{!$CurrentPage.parameters.chatKey}'
/><br />
                  Last Visited Page: <apex:outputText
value='{!$CurrentPage.parameters.lastVisitedPage}' /><br/>
                  Original Referrer: <apex:outputText
```

```
value='{!$CurrentPage.parameters.originalReferrer}' /><br/>
                    <!-- When the GeoLocation is not available this will appear as Unknown
-->
                    Latitude: <apex:outputText value='{!$CurrentPage.parameters.latitude}'
/><br/>
                   Longitude: <apex:outputText value='{!$CurrentPage.parameters.longitude}'
 /><br/>
                    City: <apex:outputText value='{!$CurrentPage.parameters.city}' /><br/>
                  Region: <apex:outputText value='{!$CurrentPage.parameters.region}' /><br/>

                    Country: <apex:outputText value='{!$CurrentPage.parameters.country}'
/><br/>
                    <!-- End of GeoLocation information -->
                    Organization: <apex:outputText
value='{!$CurrentPage.parameters.organization}' /><br/>
                    Disconnected By: <apex:outputText
value='{!$CurrentPage.parameters.disconnectedBy}' /><br/>
                    Window Language: <apex:outputText
value='{!$CurrentPage.parameters.windowLanguage}' /><br/>
                    Chat Details: <apex:outputText
value='{!$CurrentPage.parameters.chatDetails}' /><br />
                  Transcript: <apex:outputText value='{!$CurrentPage.parameters.transcript}'
 /><br/>
                    Attached Records : <apex:outputText
value='{!$CurrentPage.parameters.attachedRecords}' /><br />
                    Error: <apex:outputText value='{!$CurrentPage.parameters.error}' /><br
/>
            </p>
      </div>
      <hr/>
      <!-- Message to show if chat is abandoned -->
      <div id='abandoned' style='display: none;'>
          We are sorry you decided to leave the chat. Feel free to initiate a new session.

      </div>
      <!-- Code to decide if we show the abandoned block or the full data -->
      <script type='text/javascript'>
            var requestTime = '{!$CurrentPage.parameters.requestTime}';
            var startTime = '{!$CurrentPage.parameters.startTime}';
            // when startTime doesn't have a value, it means the chat never started
            if (!startTime) {
                  document.getElementById('details').style.display = 'none';
                  document.getElementById('abandoned').style.display = 'block';
            }
      </script>

</apex:page>
```

このコードでは、次の事後チャットページがエージェントに表示されます。

チャットが中止されると、次のメッセージが表示されます。



第 8 章　リリース API を使用したエージェントへのチャットの直接転送の設定

Live Agent のリリース API のいくつかのパラメータを編集することで、特定のボタンからのチャットを特定のエージェントに転送または招待できます。指定したエージェントが対応できない場合に別のボタンまたはキューに代替するようにチャットを設定できます。

エージェントへの直接転送を使用すると、訪問者はエージェントと直接連絡できるようになります。エージェントは、エージェントへの直接転送リンクを提供して訪問者が新しいエージェントと最初からやり直す必要がないようにできるため、フォローアップの会話が必要な訪問者の問題の場合に役立ちます。

リリース API を使用したエージェントへの直接転送
特定のボタンからのチャットを 1 人のエージェントに転送できるエージェントへの直接転送を設定できます。

事前チャットフォームの代替転送
チャットを受信したときにエージェントへの直接転送に指定したエージェントが対応できない場合に備えて、事前チャットフォームに代替転送ルールを設定します。

リリース API を使用したエージェントへの直接転送

特定のボタンからのチャットを1人のエージェントに転送できるエージェントへの直接転送を設定できます。

次のリリース API メソッドを使用して、エージェントへの直接転送を設定します。このとき、エージェントがチャットに対応できない場合に備えて代替転送を有効にします。

- `startChat`
- `startChatWithWindow`
- `showWhenOnline`
- `showWhenOffline`

エージェントへの直接転送のコードサンプル
リリース API を使用してエージェントへの直接転送を実装します。

エージェントへの直接転送のコードサンプル

リリース API を使用してエージェントへの直接転送を実装します。

次のコードサンプルは、エージェントが訪問者に[私とチャット]リンクを送信できるようにエージェントへの直接転送を設定する方法を示します。

```
<apex:page standardController="User" showHeader="false">
  <h1>Direct-to-Agent Chat with {!user.name}</h1>

  <!-- dta_online is displayed whenever the specific agent is available to chat. -->
```

```
  <div id="dta_online" style="display: none;">

    <!-- A valid button is required here even though it's direct-to-agent - some button
settings still apply. -->
    <!-- {!left(user.id,15)} is needed to truncate an 18-char ID to the 15-char version
that Live Agent uses. -->
    <a href="javascript://Chat" onclick="liveagent.startChat('573D01234567890',
'{!left(user.id,15)}')">Chat with {!user.name}!</a>

  </div>

  <!-- dta_offline is displayed if the specific agent is unavailable. -->
  <div id="dta_offline" style="display: none;">

    <!-- button_online is displayed if any agents are available to chat for the button.
-->
    <div id="button_online" style="display: none;">Sorry, {!user.name} is not available.
If you&rsquo;d like, you can
      <a href="javascript://Chat" onclick="liveagent.startChat('573D01234567890')">start
a chat with another agent</a>.
    </div>

    <!-- button_offline is displayed if no agents are available to chat for the button.
-->
    <div id="button_offline" style="display: none;">Sorry, all agents (including
{!user.name}) appear to be unavailable.</div>

  </div>

  <!-- Change the domain name to the correct one for your org. -->
  <script type='text/javascript'
src='https://c.la1s1.saleforceliveagent.com/content/g/deployment.js'></script>

  <script type='text/javascript'>
    /* The following calls pass the user ID as the first argument and show whether the
agent is online.*/
    liveagent.showWhenOnline('{!left(user.id,15)}', document.getElementById('dta_online'));

    liveagent.showWhenOffline('{!left(user.id,15)}', document.getElementById('dta_offline'));


    /* The following calls pass the button ID as the first argument and show whether
    any agents are available to handle chats from the button. */
    liveagent.showWhenOnline('573D01234567890', document.getElementById('button_online'));

    liveagent.showWhenOffline('573D01234567890', document.getElementById('button_offline'));


    /* This domain and the IDs are specific to your org, so replace these with your own.
*/
    liveagent.init('https://d.la1s1.salesforceliveagent.com/chat', '572D01234567890',
'00DD01234567890');
  </script>
</apex:page>
```

組織でこのコードサンプル(ChatWithMe)を使用すると、エージェントはチャット要求を直接送信するリンクを作成できます。

http://your.website/ChatWithMe?id=005D01234567890

どのエージェントも使用できるクイックテキストメッセージを作成することで、エージェントはより簡単に[私とチャット]リンクを作成できます。

http://your.website/ChatWithMe?id={!User_ID}

リンクのUserIDの部分には、クイックテキストを使用するエージェントのユーザIDが自動的に入力されます。

事前チャットフォームの代替転送

チャットを受信したときにエージェントへの直接転送に指定したエージェントが対応できない場合に備えて、事前チャットフォームに代替転送ルールを設定します。

エージェントへの直接転送を設定していても、チャットを受信するように指定したエージェントが対応できなかったらどうなるでしょうか。エージェントがオフラインだと、チャットが成立しなくなります。

組織で顧客情報を収集するために事前チャットフォームを使用している場合は、エージェントへの直接転送に使用するボタンに代替転送オプションを設定できます。

次のコードサンプルは、ボタンに割り当てられたエージェントが対応できない場合に、別のボタンのデフォルトの転送ルールを使用してチャットを転送する方法を示します。コードのこのセクションを見ていきましょう。

```
<h1>Pre-chat Form</h1>
<form method='post' id='prechatForm'>
      Name: <input type='text' name='liveagent.prechat.name' id='prechat_field' /><br />

      Email Address: <input type='text' name='liveagent.prechat:Email' /><br />
      Department: <select name="liveagent.prechat.buttons">
          <!-- Values are LiveChatButton and/or User IDs. -->
          <option value="573D01234567890">Route through button 573D01234567890</option>
          <option value="005D01234567890">Route to agent 005D01234567890</option>
          <option value="005D01234567890_573D01234567890">Route to agent 005D01234567890

          with Fallback to button 573D01234567890</option>
```

このセクションは、このボタンから発生したチャットがエージェントID 005xx000001Sv1mを持つエージェントに転送されるよう指定します。そのエージェントが対応できない場合、受信チャットはボタンID 573xx0000000001を持つボタンのデフォルトの転送ルールに基づいて転送されます。

代替転送のコードサンプル
指定されたエージェントが対応できない場合に訪問者が別のエージェントとチャットできるように、代替転送ルールを事前チャットフォームに実装します。

代替転送のコードサンプル

指定されたエージェントが対応できない場合に訪問者が別のエージェントとチャットできるように、代替転送ルールを事前チャットフォームに実装します。

このサンプルでは、代替転送ルールが有効にされた事前チャットフォームが作成されます。このフォームでは、次の操作が行われます。

- 訪問者の名前とメールアドレスを要求する。
- チャットログ内およびチャット要求ウィンドウ内の情報を表示する。
- 顧客の情報が含まれる新しいまたは既存の取引先責任者レコードをSalesforceコンソールの新しいタブに表示する。既存のレコードの検索には、顧客の名前とメールアドレスが使用されます。既存のレコードが見つからない場合は、新しいレコードが作成され、顧客の情報が入力されます。
- チャット要求を転送するために訪問者が異なるLive Agentボタンを選択できるようにするドロップダウンリストを表示する。
- 特定のエージェントにチャットを直接転送し、そのエージェントが対応できない場合は、ボタンのデフォルトの転送ルールに基づいてチャットを転送する。

```
<apex:page showHeader="false">
<!-- This script takes the endpoint URL parameter passed from the deployment page
    and makes it the action for the form -->
<script type="text/javascript">
     (function() {
     function handlePageLoad() {
       var endpointMatcher = new RegExp("[\\?\\&]endpoint=([^&#]*)");
       document.getElementById('prechatForm').setAttribute('action',
       decodeURIComponent(endpointMatcher.exec(document.location.search)[1]));
     }

      if (window.addEventListener) {
              window.addEventListener('load', handlePageLoad, false);
      } else { window.attachEvent('onload', handlePageLoad, false);
              }})();
</script>
<h1>Pre-chat Form</h1>
<form method='post' id='prechatForm'>
      Name: <input type='text' name='liveagent.prechat.name' id='prechat_field' /><br />

      Email Address: <input type='text' name='liveagent.prechat:Email' /><br />
      Department: <select name="liveagent.prechat.buttons">
          <!-- Values are LiveChatButton and/or User IDs. -->
          <option value="573D01234567890">Route through button 573D01234567890</option>
          <option value="005D01234567890">Route to agent 005D01234567890</option>
          <option value="005D01234567890_573D01234567890">Route to agent 005D01234567890

          with Fallback to button 573D01234567890</option>
      </select><br />
      <input type='submit' value='Request Chat' id='prechat_submit'/>
</form>

</apex:page>
```