Introduction to the Custom Element for Account Setup
# Table of Contents

About this document.................................................................1  
Changes in this version ............................................................1  
Custom Element for Account Setup.............................................1  
Known limitations.................................................................1  
User types..............................................................................1  
Account Setup behavior .........................................................2  
Timeout handling .....................................................................2  
Browser support........................................................................2  
Authentication .........................................................................2  
Download and install the custom element.....................................3  
Incorporate the custom element into your parent page ..................3  
Minimun example.....................................................................4  
Example setting all attributes ..................................................4  
Attributes................................................................................5  
Events.....................................................................................7  
Managing events in the workflow ..............................................9  
userExit .................................................................................9  
badAuthentication ......................................................................9  
internalError .........................................................................9  
Customizations to the user interface .........................................10  
Appendix A.  Cumulative changes to the Custom Element ............. A-1
About this document
This document provides a preliminary overview of the new custom element for Account Setup which is currently under development for Morningstar® ByAllAccountsSM (BAA) aggregation service and undergoing beta testing with clients.

Changes in this version
This document is evolving along with the custom element which is currently at version 1.4.0-beta.1.

This update is required
The user-visible change in this release is:

▪ Support for financial institutions that do not support account discovery has been enhanced to allow editing of the credential information and the manual addition of accounts during the edit workflow. There remain limitations on editing of account numbers for accounts at financial institutions that do not support account discovery.

Refer to Appendix A Cumulative changes to the Custom Element for important information about other recent changes.

Custom Element for Account Setup
Morningstar® ByAllAccountsSM aggregation service provides a W3C custom HTML element that implements the ByAllAccounts Consumer User Interface (CUI) account setup functionality for aggregation within another application without the use of iframes or browser tabs.

The custom element (mstar-aggregation-consumer-accountsetup) enables account setup. The main features are selecting financial institutions (FI), entering FI credentials, discovering accounts, and aggregating the account data. The aggregated account data, which can include balance, holding, and transaction information from the FI, are available via other Morningstar ByAllAccounts applications and APIs.

The custom element can be fully customized to integrate into your parent page, with customizations to terminology, styles, and features.

This document describes the custom element, steps for embedding it in a web application, methods of authentication, and ways to customize it.

Known limitations
Because the Account Setup custom element is currently in beta testing, the capabilities, interfaces, and behaviors are subject to change as we make refinements based on test results and customer feedback.

This section describes the known limitations for the Account Setup custom element.

User types
The Account Setup custom element can only be used with BAA Investor users and those users must:

▪ have full Read-Write permission to their own data
▪ be Single Sign On (SSO) users (either non-SAML-SSO or SAML-SSO)
▪ be in a BAA Firm configured to use ConsumerUI V2.0. Note that a BAA Firm can be configured to use only one version of ConsumerUI.
Account Setup behavior

The following are true for the Account Setup custom element:

- the Add account ‘progress step’ only shows the most-recently added accounts. Each time a user invokes the 'Add Another' feature, the display shows the current set being added, not any that were added previously
- it does not yet support FI Request
- it does not yet support adding accounts at FIs that do not support Account Discovery

Timeout handling

Following a timeout, the custom element does not automatically restart the customer’s action when the parent page provides the new authentication. This limitation will continue until resume is implemented.

Browser support

- **Chrome** – fully supported
- **Firefox** – fully supported
- **Internet Explorer and Microsoft Edge** – Partially supported. All functionality works but the CSS styling is not encapsulated so the styling becomes global and affects the parent page. In particular, the header elements (h1, h2...) use the Morningstar Design System (MDS) fonts.
- **Safari** – not yet tested

Authentication

Authentication is handled by the parent page before it invokes the custom element; authentication can be accomplished via SAML Single Sign On (SSO) or non-SAML SSO (via DataConnect).

- **SAML SSO** – The SAML interchange establishes a SAML session with a cookie that identifies that session. The cookie must be available to the custom element. For this case, use the auth-context attribute to indicate the use of SAML.

  The custom element makes service calls to www.byallaccounts.net. To allow these calls within the user’s browser, the parent page’s domain(s) must be configured in the custom element’s CORS whitelist within the ByAllAccounts service. Your Morningstar Implementation Manager will assist you with getting this configuration established.

- **Non-SAML SSO** – The parent page acquires values for jsessionid and csrftoken and must pass them to the custom element using the auth-context attribute.

Although it provides many details not relevant to the custom element, the guide for AccountView SSO may provide some helpful information about non-SAML SSO:

http://www.byallaccounts.net/Manuals/Accountview/AccountView_SingleSignOn.pdf

Likewise, this guide for SAML connectivity may be helpful:

Download and install the custom element

**Note:** Ultimately the custom element will be distributed via the npm repository as a versioned npm package named mstar-aggregation-consumer-accountsetup. However, for now it is available as a packaged file (with a .tgz extension) from Morningstar ByAllAccounts. In the installation instructions below, use the name of the .tgz file you receive from ByAllAccounts.

The `mstar-aggregation-consumer-accountsetup` custom element is built with Angular and can be used in frameworks such as React, and Vue, or in vanilla JS applications.

Install the package by running the following command in your development environment:

```
npm install <xxxx>.tgz
```

After the npm package is installed, there is a `mstar-aggregation-consumer-accountsetup` folder under the `node_modules` directory. The folder contains:

- **`mstar-aggregation-consumer-accountsetup.js`** – Main JavaScript bundle for the Account Setup custom element.
- **`assets`** – Folder containing various configuration files and images for the custom element.
- **`package.json`** – File that describes the npm package.

Depending on your build system/framework, there may be different steps to take to get Web Components loaded. In any case, the build system should ensure that the assets folder and the main JS file (`mstar-aggregation-consumer-accountsetup.js`) of the custom element are placed under the application’s distribution (/dist) folder from where it is being served.

Incorporate the custom element into your parent page

Import the necessary JavaScript files into your HTML page and create an instance of the `mstar-aggregation-consumer-accountsetup` tag element. Then add code to:

- Reference the font faces.
- Load the custom element.
- Instantiate the element in the page.
- Set attributes. Minimally set `auth-context`, which is required. If one or more route attributes are used, set the `auth-context` before the first route. That is, the user authentication must be given before the element can attempt the operation.
- Make the custom element visible in the page.

The following sections include:

- A simple code example showing the minimum requirements
- A more detailed example showing all the attributes

Additionally, consider how to handle the events described in [Events](#) on page 7.
Minimum example

At a minimum you must define the auth-context attribute for authentication, as shown here. By default, the custom element starts with add accounts. To begin with editing credentials, use the route attribute.

```html
<!-- Reference the required font faces -->
<link rel="stylesheet" href="./assets/css/fonts.css">
<!-- Load the custom element main JavaScript bundle file -->
<script src="/mstar-aggregation-consumer-accountsetup.js"></script>

<script>
    mstarWebComp = document.createElement("mstar-aggregation-consumer-accountsetup");
    mstarWebComp.setAttribute("auth-context", JSON.stringify(
        { jsessionid: "212764322EB9DBEBED880425DE3216EB",
          csrfToken: "DBC63BDE361C192B3CEF641B4C551DD8AC27B4A0276E91"
        }));
    content.appendChild(mstarWebComp);
</script>

Example setting all attributes

This example uses all available attributes for the custom element. Because the attributes that customize the user interface (ui-config-file, translate-file-path, and override-css-file) typically do not change during the lifetime of a parent page, for simplicity they are defined first. The others (auth-context, route) may be reset any number of times during the life of the parent page. For more information about the attributes, refer to Attributes, page 5.

```html
<!-- Reference the required font faces -->
<link rel="stylesheet" href="./assets/css/fonts.css">
<!-- Load the custom element main JavaScript bundle file -->
<script src="/mstar-aggregation-consumer-accountsetup.js"></script>

<script>
    mstarWebComp = document.createElement("mstar-aggregation-consumer-accountsetup");
    mstarWebComp.setAttribute("ui-config-file", "./customassets/config/ui-config.json");
    mstarWebComp.setAttribute("translate-file-path", "./customassets/i18n/");
    mstarWebComp.setAttribute("override-css-file", "./customassets/css/corporatestyle.css");
    mstarWebComp.setAttribute("auth-context", JSON.stringify(
        { jsessionId: "212764322EB9DBEBED880425DE3216EB",
          csrfToken: "DBC63BDE361C192B3CEF641B4C551DD8AC27B4A0276E91"
        }));
    mstarWebComp.setAttribute("route", "/edit-credential;credentialId=21911");
    content.appendChild(mstarWebComp);
</script>
Attributes

The following table describes the attributes for the custom element. The only required attribute is auth-context. The only requirement for order is that the auth-context must be specified before the first route, if there is one, because user authentication must be given before the element can attempt the operation. Otherwise, a badAuthentication event will occur.

Any of these attributes can be set (and reset) at any time but ui-config-file, translate-file-path, and override-css-file typically do not change during the lifetime of a parent page. An example of when auth-context and route might be reset during the lifetime of a parent would be handling of an error event. For information about events, see Events page 7.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Req’d?</th>
<th>Possible Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>auth-context</td>
<td>Yes</td>
<td>Must contain one of:</td>
<td>The auth-context attribute is used for user authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• { jsessionId: &lt;jsessionid value&gt;, csrfToken: &lt;csrf token value&gt; }</td>
<td>It must contain one of the possible parameters: the jsessionId/csrfToken pair, the SAML indicator, or an empty object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• {saml: “true”}</td>
<td>If the user was authenticated via non-SAML single sign-on then the jsessionId and csrfToken pair must be included, using the form:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• {}</td>
<td>{ jsessionId: &lt;jsessionid value&gt;, csrfToken: &lt;token&gt; }</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If the user was authenticated via SAML then the SAML cookie must exist in the browser and only the SAML flag is included in the attribute:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{saml: “true”}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>To prevent further use of the element by the user, use the empty value to clear the authentication data from the element:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{}</td>
</tr>
</tbody>
</table>
## Introduction to the Custom Element for Account Setup

### Attribute | Req’d? | Possible Parameters | Description
--- | --- | --- | ---
route | No | • /add-account  
• /edit-credential;credentialId=<id>  

To suppress the progress step, set the optional showProgressStep parameter to false, as shown:

• /add-account;showProgressStep=false  
• /edit-credential;credentialId=<id>;showProgressStep=false

Specifies the operation to be initiated, so:

• /add-account will cause the first step (FI selection) to be displayed. This value is the default.

• /edit-credential;credentialId=<BAAID> where <BAAID> is the ByAllAccounts unique internal identifier for the Account Credential to be edited. This route will display the form containing the current login and password.

Use the optional parameter showProgressStep=false with each route for which you want to suppress the final progress step that appears after a successful account add or edit.

The custom element defaults to the Add account page when it is first made visible in the parent page. If you want to suppress the progress step for that, you must explicitly set route and use the optional parameter.

**Note:** auth-context must precede the first route, otherwise a badAuthentication event will occur.

### translate-file-path

<table>
<thead>
<tr>
<th>Req’d?</th>
<th>Possible Parameters</th>
<th>Description</th>
</tr>
</thead>
</table>
| No | The path to the custom en.json file. | Identifies the path to the en.json file that you created to provide custom terminology for certain text and labels displayed by the custom element. The path is relative to the base href of the deployed application. The default path is:  
./assets/i18n/ |
## Attribute

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Req’d?</th>
<th>Possible Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ui-config-file</td>
<td>No</td>
<td>The path and file name to your custom user interface (UI) configuration file.</td>
<td>Identifies the path to and the name of the file that contains options for turning on/off UI sections in the custom element. The path is relative to the base href of the deployed application. The default is: ./assets/ui-config.json</td>
</tr>
<tr>
<td>override-css-file</td>
<td>No</td>
<td>The path and file name for your custom cascading style sheet (CSS) file.</td>
<td>Identifies the path to and file name of a custom CSS file containing overrides to the default Morningstar Design System (MDS) styles.</td>
</tr>
</tbody>
</table>

### Events

The following table describes the events that the mstar-aggregation-consumer-accountsetup custom element may communicate to the parent page and the actions that trigger them. These events signal that the user has completed their work with the custom element or there is an error that the custom element cannot resolve and that must be handled by the parent page.

Other events that occur within the custom element are handled by the custom element itself. For example, any problems that occur when a user interacts with the custom element to enter credentials are handled by the custom element.

Managing events in the workflow on page 9 describes ways in which the parent page may need to handle these events.
<table>
<thead>
<tr>
<th>Trigger</th>
<th>Name</th>
<th>Event Detail String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User completes action and/or exits the operation</td>
<td>userExit</td>
<td>“User exit”</td>
<td>User has 1) completed the account add, 2) completed the account edit, or 3) canceled out of the workflow.</td>
</tr>
<tr>
<td>A request to server finds user’s authentication context is not valid (resulted in a 401 Unauthorized)</td>
<td>badAuthentication</td>
<td>“Unauthorized access”</td>
<td>User’s authentication context is either not valid or has timed out. The user will no longer be able to interact with the element after a badAuthentication (401) event until the element receives changes to the auth-context attribute that provide valid authentication data. The user’s authentication could end at any time. Whenever the parent page receives a badAuthentication event it must get new tokens (or a new cookie in the case of SAML) and invoke a setAttribute auth-context with the new tokens before the user can proceed with the operation they had started.</td>
</tr>
<tr>
<td>A request to the server resulted in an error other than 401 or an internal code error occurred</td>
<td>internalError</td>
<td>One of: ▪ “Forbidden access” ▪ “Service unavailable” ▪ “Unknown server error” ▪ “Unknown error” ▪ “Resource not found”</td>
<td>An unexpected error occurred during user’s interaction with the element. The parent page must remedy the problem and possibly display a message to the user. Some causes include: ▪ Forbidden access: User does not have the necessary permissions; user may have read only permissions. ▪ Service unavailable: Possibly an intermittent service interruption. ▪ Resource not found: Could be a misnamed path or file.</td>
</tr>
</tbody>
</table>
Managing events in the workflow

This section describes managing the trigger events listed in Events on page 7.

**userExit**
The custom element sends the userExit event when the user exits the custom element, such as by completing adding an account, completing editing an account, or by cancelling out of the workflow. The userExit event can also occur when an account add or edit credential successfully completes and the progress step is suppressed.

Upon userExit, the parent page could hide the custom element or activate it again. To activate it again, if the authorization context is still valid, the parent page can set the route attribute to reopen the custom element at set point in the workflow.

**badAuthentication**
A badAuthentication event can be caused by:

- **timeout:** the user stops interacting with the custom element for a period of time and the session for the auth-context times out.
- **authentication ended:** the authorized connection ended. For SAML SSO users, that could mean their cookie timed out.

For badAuthentication events, the parent page must reauthenticate the user and set a new auth-context. Until the reauthentication happens, the custom element is in a blocked state and is greyed out.

When the reauthentication is complete, the user’s experience depends on where they were in the interface when the badAuthentication occurred. If the user was in the middle of selecting an FI when they walked away, they may be able to proceed as soon as the authentication is reestablished. A user who left the FI login screen sitting for too long when entering their credentials may have to re-submit the request when the authentication is reestablished.

Note: Until resume is fully functional, the user will need to re-do the Connect action.

**internalError**
The internalError event can happen for several reasons. For example, a “Resource not found” error would occur if the parent page provided an invalid credential identifier when requesting a route to edit-credential.

When an internalError occurs, the user interface for the custom element remains static. The parent page needs to recognize that error and remedy the problem. The parent page can pass a message to the user, describing the type of error.
Customizations to the user interface

There are multiple types of customization you can provide for the custom element. You can apply these customizations using the files provided in the assets folder.

- **Terminology** – the i18n/en.json translation file contains all the static text displayed on pages, dialogs, and buttons within the interface. Edit this file to customize text and terminology.

- **Styles** – define an optional CSS file to override the default Morningstar Design System (MDS) styles and adjust fonts, colors, and backgrounds for elements in the display. Configure the element to use your custom CSS file by setting the override-css-file attribute.

  To use fonts other than the MDS Univers font, you can add a global entry like this to your CSS file:
  ```
  * {
    font-family: YourFont, Verdana, sans-serif !important;
  }
  ```

- **Icons** - the icons/mds.svg file contains icon definitions in the scalable vector graphics format. Edit this file to modify the icons used by the custom element.

- **Optional features** – the config/ui-config.json contains options that enable you to turn on/off UI features in the custom element. Only a subset of the options currently applies to the custom element. These options are:

  ```
  "cui-fi-select": {
    "fiLogosVisible": false,
    "includeURLinSearch": false
  }
  ```

  Note that:
  - "fiLogosVisible" - controls whether the popular institution logos are available in the FI selection step
  - "includeURLinSearch" - controls whether the FI URLs are included in the search criteria when the user is searching for an institution
Appendix A.  Cumulative changes to the Custom Element

This appendix keeps a running list of released versions and a summary of the changes in each version starting with 1.1.0-beta.1.

**WARNING:** When a release is required, you **must** use the latest version of the custom element. If you do not, any project you have that uses the custom element will fail because the new version has changes to the underlying API.

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Required?</th>
<th>Summary of Changes</th>
</tr>
</thead>
</table>
| April 12, 2019   | 1.1.0-beta.1 | Yes       | ▪ Changed attribute names  
▪ Authorization is now handled using a complex attribute  
▪ Documentation changes reflect those code changes, plus expanded information about events and revised code example. |
| May 2, 2019      | 1.2.0-beta.1 | Yes       | ▪ Initial support for financial institutions that require In-Session Activation Codes (ISAC) during authentication. Timely, successful responses are supported but more work is required to handle timeouts and error cases.  
▪ Documentation changes include enhancements to the code examples and clarifications to some descriptions. |
| May 23, 2019     | 1.3.0-beta.1 | Yes       | ▪ Support is complete for financial institutions that require In-Session Activation Codes (ISAC) during authentication, including handling timeouts and error cases.  
▪ An optional parameter can be set on the route attribute so that the progress step does not show after a successful account add or credential edit.  
▪ Documentation changes support the code changes and provide additional details. |
| June 13, 2019    | 1.4.0-beta.1 | Yes       | ▪ Support for financial institutions that do not support account discovery has been enhanced to allow editing of the credential information and the manual addition of accounts during the edit workflow. There remain limitations on editing of account numbers for accounts at financial institutions that do not support account discovery. |