MORNINGSTAR Wealth

ByAllAccounts Connect Suite Integration Guide - Investor Aggregation

Account Summary	1.6
Account Linking	3.10
Document:	BAA_Suite_Int_Inv-1.7-202411212
Document Issue Date:	December 12, 2024

Technical Support:	(866) 856-4951
Telephone:	(781) 376-0801
Fax:	(781) 376-8040
Web:	byallaccounts.morningstar.com

Table of Contents

Overview

This guide gives the step-by-step information you need to develop a proof-of-concept implementation embedding the ByAllAccounts Connect Suite components in your web application. It includes instructions to create and authenticate investor users, embed and interact with the components in your web application, manage critical component events, and collect aggregated data.

The ByAllAccounts Connect Suite is comprised of a pair of custom elements developed by ByAllAccounts℠. The components are:

- Account Summary component
- Account Linking component

You will implement the ByAllAccounts Connect suite using:

- ByAllAccounts REST API to register new investor users and as the primary method of data extraction
- ByAllAccounts DataConnect API for Single Sign On (SSO) authentication and (optionally) batch data extraction

The API calls to get started with ByAllAccounts for Investors are provided in a Postman collection.

About ByAllAccounts Connect Suite

ByAllAccounts Connect Suite is comprised of two components that are each independently implemented in the main page of the parent application, which is referred to in the rest of this guide as the parent page.

- Account Summary component - shows the investor's account summary information and facilitates user maintenance of the accounts. It provides a summary of the linked accounts, balances, and connectivity status and allows investors to manage their accounts on an ongoing basis.
- Account Linking component - is used by Account Summary to add new accounts and maintain account credential information. The component can be used by investors to link accounts across 15,000 financial institutions.

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

How the components work together

The components of ByAllAccounts Connect Suite are used together, in cooperation, to provide a complete solution. In a typical usage scenario, a user action in the Account Summary component will trigger an event that requires the Account Linking component to be in the parent page. Conversely, as the user performs certain actions in the Account Linking component that may change the user data displayed in Account Summary view, the Account Summary component needs to react to refresh its data. The parent page provides the coordination, or "glue", to enable this experience. The parent page controls when each component is visible and when they call each other. Typically Account Summary is implemented to always show in the parent page. Account Linking may be invoked as a popup by the parent page as needed to complete an action called for by an event in the Account Summary component.

To fully integrate Account Summary with Account Linking, you will need to work with a ByAllAccounts Implementation Manager who will help with the internal ByAllAccounts setups required including creating a Firm for you in our system, providing an administrator login and password to perform API calls, etc.

About the parent page

ByAllAccounts Connect Suite components are each independently implemented in their own iframe in the main page of your parent application, which is referred as the parent page. You independently size and position the components in the parent page. The parent page manages authentication and instantiation of each component, monitors for and handles critical events in each component, and manages hiding and revealing the Account Linking component.

About Account Summary

The Account Summary component provides access to account data for the ByAllAccounts℠ aggregation service. It is a W3C custom HTML element that implements the account summary functionality within a parent page of your application. The Account Summary component is used in conjunction with the Account Linking component. The Account Summary component provides a page of summary credential and account information for an authenticated investor, in groups such as:

- In Progress – financial institution credentials for which newly added accounts are currently being aggregated
- Needs Attention – financial institution credentials that need user action such as providing additional authentication information
- Main – all credentials and accounts for the investor
- Pending Requests – requests for new financial institution Financial Institution connections

About Account Linking

The Account Linking component enables you to embed end-client access to the ByAllAccounts℠ aggregation service in your investor portal. It is a W3C custom HTML element that implements the account setup functionality for aggregation within a parent page of your application.

Account Linking supports adding and maintaining accounts within the aggregation system. The aggregated account data, including balance, holding, and transaction information, are available via ByAllAccounts APIs.

Investors use Account Linking to select a financial institution (FI), submit their credentials, and aggregate their held-away investment and non-investment account data. The investors can also use Account Linking to update information for their accounts on an ongoing basis.

Account Linking expedites the onboarding experience for your end-client investors and eliminates manual entry of hard-to-get account/asset data.

ByAllAccounts' aggregated data is produced using best in class normalization and security matching technology. It is used for holistic performance reporting and investment analysis as well as to drive budgeting and savings tools for personal financial management. Our tools enable you to get a unified view of portfolios by outsourcing aggregation of securities and alternative investments from multiple custodians.

Account Linking provides the following:

- selecting financial institutions (FIs or custodians) from our set of 15,000 FIs; this list can be customized for your firm
- entering FI credentials
- discovering available accounts and adding them to the system
- aggregating the account data

Investor users access the components using single sign-on (SSO).

Technical Details

The components are W3C custom HTML elements. They are built with Angular and can be used in frameworks such as React and Vue, or in vanilla JS applications.

All communication between the components and the ByAllAccounts service is performed using HTTPS. Additional techniques are employed to ensure secure treatment of the data.

User types

The components 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
- be in a BAA Firm that is licensed to use Account Linking and Account Summary

Browser support

- Chrome (including incognito mode), Firefox, Microsoft Edge, and Safari are fully supported

Postman collection of API calls

The DataConnect and BAA REST API calls to get started with ByAllAccounts for Investors are provided in a Postman collection.

Integrating ByAllAccounts Connect Suite

To integrate the ByAllAccounts Connect Suite with your application, you will do the following:

- Get set up for access by your implementation manager
- Register investor users
- Authenticate investor users
- Set permissions to allow framing
- Download and install the Account Linking and Account Summary component files
- Embed Account Linking and Account Summary into your parent page
- Collect aggregated data
- Optionally, customize the user interface of the components

Get set up for access

1. Contact your implementation manager to discuss getting set up for ByAllAccounts Connect Suite.
2. Your implementation manager will:
    - Review your statement of work (SOW) to confirm use case details which will inform how we configure your access to our system.
    - Create a definition for a Firm in our system, configured for your needs.

- Provide you with an Administrator login and password to perform DataConnect API calls for registering users, establishing user sessions, and collecting aggregated data.
- Provide you with a "technical support administrator" login for AccountView, which can be used as an effective troubleshooting tool for data and connectivity errors.
- Ask for your parent page's domain(s). The ByAllAccounts Connect Suite components use a Content Security Policy (CSP) that controls which domains are allowed to frame them. ByAllAccounts must configure the domain(s) for your parent pages into the CSP to allow your pages to frame the ByAllAccounts Connect Suite components.

Register Investor users

Investor users are the account holders; these are the users who will use Account Summary for reviewing summary account information and Account Linking for aggregation. Create Investor users with the BAA REST API endpoint POST /api/v1/persons. Note that you will need the "Person Id" from the response to authenticate the investor user in the future.

Link to Postman for REST API endpoint for creating investor user.

For details about the REST API, refer to
http://www.byallaccounts.net/Manuals/DataConnect/BAA_REST_API.pdf.

Authenticate Investor users

Authentication is handled by the parent page before it invokes the Account Linking and Account Summary components. There are two methods of SSO authentication.

- Invoking a DataConnect API <SESSIONAUTHRQ> request, providing your Administrator login and password, plus the "Person Id" you stored from the response to the POST /api/v1/persons endpoint. The response will provide you with values for *jsessionId* and *csrfToken*, which must be passed to the components using the *auth-context* attribute.
  Link to Postman for DataConnect API operation for authenticating an investor using <SESSIONAUTHRQ>.
- Invoking a DataConnect API <SAMLAUTHRQ> request, providing a base64-encoded SAML assertion from the identity provider (IP). The response will provide you with values for *jsessionId* and *csrfToken*, which must be passed to the components using the *auth-context* attribute.
  Link to Postman for DataConnect API operation for authenticating an investor using <SAMLAUTHRQ>

Note: For DataConnect operations in Postman, use "Send and Download" and save the result to a .zip file. Inside that .zip file find the XML response which contains the *jsessionId* and *csrfToken* values.

For details about how DataConnect handles authentication, you can refer to the Single Sign-on Operations section in the DataConnect V4 Ultra User Guide.

Permissions for framing

The Account Linking and Account Summary components use a Content Security Policy (CSP) that controls which domains are allowed to frame them. The parent page domain(s) must be configured as allowed in this

CSP to allow this framing. Your ByAllAccounts Implementation Manager will assist you with this configuration.

Download and install ByAllAccounts Connect Suite components

Each component must be downloaded and installed. They are packaged files with a .tgz extension.

Step 1

- Download the packaged file for Account Summary *mstar-aggregation-consumer-accountsummary* component from https://www.byallaccounts.net/WebPortfolio/mstar-aggregation-consumer-accountsummary.tgz.
- Download the packaged file for Account Linking *mstar-aggregation-consumer-accountsetup* component from https://www.byallaccounts.net/WebPortfolio/mstar-aggregation-consumer-accountsetup.tgz.

Step 2

Install the packages by running the following commands in your development environment:

```
npm install mstar-aggregation-consumer-accountsummary.tgz
```

```
npm install mstar-aggregation-consumer-accountsetup.tgz
```

Step 3

Confirm these files are under the *node_modules* directory*:*

- *mstar-aggregation-consumer-accountsummary* folder. The folder contains:
  - *mstar-aggregation-consumer-accountsummary.js*, which is the JavaScript bundle for the Account Summary component.
  - *as_assets* folder, which contains various configuration files and images for the component.
  - *package.json* file that describes the npm package.
- *mstar-aggregation-consumer-accountsetup* folder. The folder contains:
  - *mstar-aggregation-consumer-accountsetup.js*, which is the JavaScript bundle for the Account Linking component.
  - *assets* folder, which contains various configuration files and images for the component.
  - *package.json* file that describes the npm package.

The build system should ensure that the *as_assets* and *assets* folders and the main JavaScript files (*mstar-aggregation-consumer-accountsummary.js* and *mstar-aggregation-consumer-accountsetup.js*) are placed under the application's distribution (*/dist*) folder from where it is being served.

Step 4

Import the necessary JavaScript files into your HTML parent page and create an instance of the *mstar-aggregation-consumer-accountsummary* tag element and of the *mstar-aggregation-consumer-accountsetup* tag element.

Step 5

Incorporate the components into your parent page. See Incorporate the components into your parent page, on page 6.

Step 6

Optionally customize the user interface of the components. See Customize the user interface on page on page 19.

Incorporate the components into your parent page

When the ByAllAccounts Connect Suite components are incorporated into a parent page, the Account Summary is visible and acts as the primary controlling entity for the Suite. The parent page must monitor for the *accountSetup* event issued by the Account Summary component and relay that to the Account Linking component to tell it which workflow to start when it exposes the component. That's the only information the parent page needs to relay between the two components, but the parent page must also listen for and manage critical events issued from both components. Additionally, the required attribute (*auth-context*) must be set for each component to manage authorization. Optionally, other attributes may be set for customizations.

This section includes:

- Descriptions of all attributes for the components
- Code for instantiating the components in your parent page
- Code samples for each component, showing all applicable attributes
- Descriptions of the critical and standard events, how to monitor for them, and how to manage them.

About Attributes

The following table describes the attributes for the Account Linking component and the Account Summary component. The *auth-context* is the only required attribute for the components.

Any of these attributes can be set (and reset) at any time but *translate-file-path*, *override-css-file*, and *ui-config-file* (for Account Linking) typically do not change during the lifetime of a parent page. An example of when *auth-context* might be reset during the lifetime of a parent page would be handling of an authorization expiring.

Attribute	Required?	Parameters and Description
auth-context	Yes	**Possible Parameters** Must contain one of: *{ jsessionId: <jsessionIdValue>, csrfToken: <csrfTokenValue> }* *{}* **Description** The *auth-context* attribute is used for user authentication. It is required and may be reset any number of times during the lifetime of the parent page. For example, it would need to be reset after a timeout. It must contain the parameter pair *jsessionId/csrfToken* or an empty object. When the user is authenticated via SSO, the jsessionId and csrfToken pair must be included, using the form: *{ jsessionId: <jsessionIdValue>, csrfToken: <csrfTokenValue>}*

Attribute	Required?	Parameters and Description
		To prevent further use of the component by the user, use the empty value to clear the authentication data from the component: *{}*
route	No	*Only applies to Account Linking*. **Possible Parameters** The event detail strings that come from the Account Summary *accountSetup* event described in About Events on page 11. **Description** When any one of certain user actions is performed in the Account Summary component, a specific response is expected in a particular workflow of the Account Linking component. To effect the response, the parent page must be listening for the *accountSetup* event from the Account Summary component and pass its event detail string as the parameter on the *route* attribute. The parameter specifies the workflow to initiate in Account Linking. When a workflow completes, the Account Linking component tells the parent page that it is done. The *route* attribute must be set after the *auth-context* attribute because user authentication must be set before the component can attempt the operation. If the *auth-context* is not set before the *route* attribute, a badAuthentication event will occur.
translate-file-path	No	**Possible Parameters** The path to the custom en.json file. **Description** The *translate-file-path* attribute identifies the URL that is the path to the *en.json* files that you can create to provide custom terminology for certain text and labels. The path is relative to the base *href* of the deployed application. Note: Each component has its own en.json file with different terminology.
ui-config-file	No	*Only applies to Account Linking.* **Possible Parameters** Path and file name to your custom user interface (UI) configuration file. **Description** The *ui-config-file* attribute identifies the URL that is the path to and the name of the file that contains options for turning on/off UI sections in the Account Linking component. The path is relative to the base *href* of the deployed application.
override-css-file	No	**Possible Parameters** The path and file name for your custom cascading style sheet (CSS) file. **Description**

Attribute	Required?	Parameters and Description
		The *override-css-file* attribute identifies the URL that is the path to and file name of a custom CSS file containing overrides to the default Morningstar Design System (MDS) styles. Note: Each component has its own CSS file.
custom-fonts	No	**Possible Parameters** The path and name for your font definitions. **Description** The *custom-fonts* attribute identifies the URL that is the path to and file name of your custom font definition file that will override to the default Morningstar Design System (MDS) fonts. Note: Each component has its own fonts file.
iframe-style	No	**Possible Parameters** Standard CSS inline style values. For example, iframe-style="height:50vh; border:dashed;" **Description** The *iframe-style* attribute applies CSS inline styles to the iframe of the component. Clearing this attribute after it has been set reverts the iframe style of the component back to its original state.

Instantiating ByAllAccounts Connect Suite into your parent page

In a typical usage scenario of the ByAllAccounts Connect Suite, the Account Summary component is displayed first and occupies a section on the parent page.

When the user performs an action in the Account Summary component, the Account Linking component is launched in a modal dialog where the user steps through the wizard flow to perform the associated operation.

When the user completes the operation in the Account Linking component, the user exits that component, the dialog is dismissed, and control is returned back to the Account Summary component.

These interactions between the Account Summary and Account Linking components can repeat multiple times during a user session, and at each time the Account Linking component needs to be launched in the dialog and then dismissed.

This launching of the Account Linking component can potentially be an expensive and lengthy operation as it involves network activity to load the component's JS bundles and to establish the session. To make this behavior as performant and smooth as possible, and to provide a good user experience to the user, it is recommended that both the Account Summary and the Account Linking components be preloaded and instantiated at the start of the parent application session, when the user logs in. The preloaded Account Linking component should then be placed in the modal dialog, ready to be used but kept hidden until it is needed.

The following code sample shows how to instantiate the components as recommended, including setting the *auth-context* attribute for each component for authentication purposes.

Note: Your parent page ***must*** additionally listen for and manage the critical events which are described in About Events on page 11.

```
<!-- Load the main JavaScript bundle file of the Account Summary component -->
<script src="./mstar-aggregation-consumer-accountsummary.js"></script>
<!—Instantiate Account Summary (summaryCE) component -->
<script>
  summaryCE = document.createElement(“mstar-aggregation-consumer-accountsummary”);
  summaryCE.setAttribute(“auth-context”, JSON.stringify(
  { jsessionId: “<jsession id>”,
    csrfToken: “<csrf token>”
  }));

// Add an event listener that listens to accountSetup events fired by the Account
Summary (summaryCE) component, and when fired, display the Account Linking (setupCE)
component
  summaryCE.addEventListener('accountSetup', (e) => {
  // Set the route attribute of the Account Linking (setupCE) component using the
value passed in the event
  setupCE.settAttribute(‘route’, e.detail);
  // Show the Account Linking (setupCE) modal dialog
  Document.getElementById(‘dialog’).showModal();
  });

 content.appendChild(summaryCE);
</script>

<!—Preload the main JavaScript bundle file of the Account Linking (setupCE) component
-->
<script src="./mstar-aggregation-consumer-accountsetup.js"></script>
<!—Instantiate Account Linking (setupCE) component and keep it hidden-->
<script>
  setupCE = document.createElement(“mstar-aggregation-consumer-accountsetup”);
  setupCE.setAttribute(“auth-context”, JSON.stringify(
  { jsessionId: “<jsession id>”,
    csrfToken: “<csrf token>”
  }));


  // Add an event listener that listens to userExit events fired by the Account
Linking (setupCE) component, and when fired, hide the modal dialog
  setupCE.addEventListener(‘userExit’, (e) => {
    // Hide the Account Linking (setupCE) modal dialog
    Document.getElementById(‘dialog’).close();
    });

  // Add the Account Linking (setupCE) component to the dialog element and keep it
hidden
  content = document.getElementById(‘dialog’);
  content.appendChild(setupCE);
</script>
```

Example launching Account Summary

This section shows a code example for launching Account Summary with all applicable attributes. This sample integration code:

- Loads the Account Summary component.
- Instantiates the component in the page.
- Sets all common attributes including the required *auth-context* attribute to provide authentication.
- Makes Account Summary visible in the page.

Attributes are available to:

- Customize the user interface: *translate-file-path*, *override-css-file*, *custom-fonts*

  These attributes typically do not change during the lifetime of the parent page. We recommend setting these attributes first.
- Provide authentication: *auth-context*

  Authentication can change during the life of the parent page. When authentication expires or a new user is authenticated, you need to reset the *auth-context* attribute.

See About Attributes on page 6 for detailed information about attributes.

In this example, replace *hostname* with your host name for those files.

```
<!-- Load the main JavaScript bundle file of the Account Summary component -->
<script src="./mstar-aggregation-consumer-accountsummary.js"></script>
<script>
  mstarWebComp = document.createElement("mstar-aggregation-consumer-accountsummary");
  mstarWebComp.setAttribute("ui-config-
file",  "https://hostname/customassets/config/ui-config.json");
  mstarWebComp.setAttribute("translate-file-
path",  "https://hostname/customassets/i18n/");
  mstarWebComp.setAttribute("override-css-
file",  "https://hostname/customassets/css/corporatestyle.css");
  mstarWebComp.setAttribute("custom-
fonts",  "https://fonts.googleapis.com/css?family=Lobster");
  mstarWebComp.setAttribute("auth-context", JSON.stringify(          { jsessionId:
"212764322EB9DBEBED880425DE3216EB.s1a",            csrfToken:
"DBC63BDE361C192B3CEF641B4C551DD8AC27B4A0276E91"          }));
  content.appendChild(mstarWebComp);
</script>
```

Example launching Account Linking

This section shows a code example for launching Account Linking with all applicable required attributes. This sample integration code:

- Loads the Account Linking component.
- Instantiates the component in the page.
- Sets all common attributes including the required *auth-context* attribute to provide authentication.
- Makes Account Linking visible in the page.

Attributes are available to:

- Customize the user interface: *ui-config-file*, *translate-file-pat*h, *override-css-file*, *custom-fonts*

  These attributes typically do not change during the lifetime of the parent page. We recommend setting these attributes first.

- Provide authentication: *auth-context*

  Authentication can change during the life of the parent page when authentication expires or a new user is authenticated. If you reset the *auth-context* attribute, you must then reset the *route* attribute.

See About Attributes on page 6 for detailed information about attributes.

In this example, replace *hostname* with your host name for those files.

```
<!-- Load the main JavaScript bundle file of the Account Linking component -->
<script src="./mstar-aggregation-consumer-accountsetup.js"></script>
<script>{
// This event listener listens to accountSetup events fired by the Account Summary
component, and when fired it launches the Account Linking component
summaryCE.addEventListener('accountSetup', (e) => {
  mstarWebComp = document.createElement("mstar-aggregation-consumer-accountsetup");
// Set attributes
  mstarWebComp.setAttribute("ui-config-
file",  "https://hostname/customassets/config/ui-config.json");
  mstarWebComp.setAttribute("translate-file-
path",  "https://hostname/customassets/i18n/");
  mstarWebComp.setAttribute("override-css-
file",  "https://hostname/customassets/css/corporatestyle.css");
  mstarWebComp.setAttribute("custom-
fonts",  "https://fonts.googleapis.com/css?family=Lobster");
// Pass in the auth context of the logged in user to the mstar component
  mstarWebComp.setAttribute("auth-context", JSON.stringify(
      { jsessionId: "212764322EB9DBEBED880425DE3216EB.s1a",
        csrfToken: "DBC63BDE361C192B3CEF641B4C551DD8AC27B4A0276E91"
  }));
  content.appendChild(mstarWebComp);
  });
</script>
```

About Events

The Account Summary component (*mstar-aggregation-consumer-accountsummary*) and the Account Linking component (*mstar-aggregation-consumer-accountsetup*) can issue two types of events, which are caused by a trigger action.

How an event is resolved depends on the type of event:

- Critical events must be monitored for and managed by the parent page. Typically, these events signal that there is an error that the component cannot resolve or that the user has completed their work. Additionally, the Account Summary component can issue an event that signals there is a parameter value that the parent page must pass to Account Linking.
- Standard (informational) events occur in Account Linking only and provide information for the parent page. Informational events may be monitored, but are managed by the component. For example, Account Linking handles any problems that occur when the credentials entered for a financial institution cannot be used to successfully authenticate at that institution.

Monitoring for events on page 14 provides a code sample for monitoring for a specific event. Managing critical events on page 15 describes ways in which the parent page may need to handle the critical events. Managing standard events in Account Linking on page 16 describes managing the informational events.

Critical event only in Account Summary

The following critical event only occurs in Account Summary.

Trigger	Name	Event Detail String	Description
A user action in the Account Summary component requires the parent page to provide *route* information to the Account Linking component	accountSetup	A string to pass as a parameter value to Account Linking on the *route* attribute.	This event is triggered by a user action in Account Summary. The parent page must pass the resulting event detail string as the parameter on the *route* parameter for Account Linking to tell Account Linking what workflow to open. Examples that can cause the event to fire include when user clicks: ▪ "Link More Accounts" button. ▪ "Connect Accounts" menu option and selects "Enter New Credential" button in the dialog. ▪ "Connect Accounts" menu option and selects "Link More Accounts" button in the dialog. ▪ "Re-Authenticate" menu action. ▪ The edit pending request action. ▪ The account name. For information about managing this event, refer to accountSetup on page 15.

Critical events in both components

The following critical events must be monitored in both components.

Trigger	Name	Event Detail String	Description
A request to the server finds user's authentication context is not valid (resulted in a 401 Unauthorized)	badAuthentication	"Unauthorized access"	User's authentication context is either not valid, the session timed out, or their authentication expired. The user will no longer be able to interact with the component

Trigger	Name	Event Detail String	Description
			after a *badAuthentication* (401) event until the component 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 re-authenticate and invoke a *setAttribute auth-context* with the refreshed header information before the user can proceed with the operation they had started. Note: When authentication fails for either component, it is recommended that authentication is reestablished for both. For information about managing this event, refer to badAuthentication on page15.
Component is not licensed	badConfiguration	"User not licensed"	If the component is not licensed at initiation, there will be a blank area in the parent page where the component is expected. If the configuration changes while the component is in use, the component will be in a blocked state and greyed out on the parent page. For information about managing this event, refer to badConfiguration on page 16.
A request to the server resulted in an error other than 401 or an internal code error occurred	internalError	One of: "Forbidden access" "Service unavailable" "Unknown server error" "Unknown error" "Resource not found"	An unexpected error occurred during the user's interaction with the component. The parent page must remedy the problem and possibly display a message to the user. Some causes include:

Trigger	Name	Event Detail String	Description
			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. For information about managing this event, refer to internalError on page 16.

Critical event only in Account Linking

The following critical event must be monitored in Account Linking.

Trigger	Name	Event Detail String	Description
User completes action and/or exits the operation	userExit	"User exit"	User has done one of the following: completed the account add, completed the account edit, canceled out of the workflow, or declined the user agreement. For information about managing this event, refer to userExit on page 16.

Standard events in Account Linking

The following standard events can occur in Account Linking. These events that may be monitored by the parent page but are handled by the Account Linking component.

Trigger	Name	Event Detail String	Description
User makes a change to credential or account	dataChanged	"Data changed"	User makes a change to a credential or account that may result in synchronous changes to any or all of the following: authentication status, accounts linked to the credential, and/or financial data for those accounts.
User initiates an aggregation operation	dataChangedAsync	"Async Data change"	User initiates the service to run the asynchronous process of composite "aggregate" which can be discover/add/aggregate, add/aggregate, or just aggregate.
User initiates an authenticate operation	dataChangedAsync Authenticate	"Async Authenticate Data change"	User initiates an authentication operation, which causes the service to run the asynchronous process of attempting to authenticate.

Trigger	Name	Event Detail String	Description
User declines user agreement	userDeclinedAgreement	"User declined user agreement"	User does not accept the terms of the latest user agreement when prompted.

Monitoring for events

This code sample shows how to listen for a specific event. Both components must be monitored for critical events.

```
mstarWebcomp.addEventListener('userExit', function (event) {
   console.log(' @@@@ userExit event is called. detail: ' + event.detail);
});
```

Managing critical events

This section describes managing the critical trigger events listed in About Events on page 11. These critical events must be monitored and managed by the parent page.

Standard events, such as dataChanged and userDeclinedAgreement, are resolved by Account Linking. For details about standard events refer to Managing standard events in Account Linking on page 16.

accountSetup

Applies to Account Summary *only.*

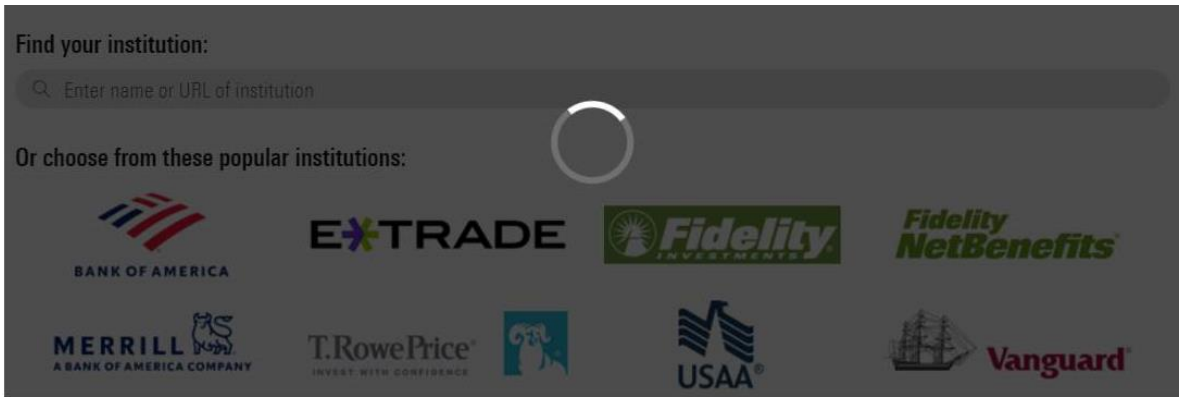
Some user actions in the Account Summary component cause it to fire its *accountSetup* event which the Account Linking component should respond to. To effect that response, the parent page must monitor for that event and provide the glue logic to handle it. The parent page must pass event detail string of the *accountSetup* event as the parameter on the *route* attribute when invoking the Account Linking component to tell the Account Linking component where in its workflow to open.

badAuthentication

A badAuthentication event can be caused by:

- timeout: the user stops interacting with a component for a period of time and the session for the *auth-context* times out.
- authentication ended: the authorized connection ended.

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



When the authentication is reestablished, the user may proceed from where they left off.

Note: When authentication fails for either component, reestablish it for both.

badConfiguration

A badConfiguration event is caused by an unlicensed component.

If the component is not licensed at initiation, there will be a blank area in the parent page where the component is expected. If the configuration changes while the component is in use, the component will be in a blocked state and greyed out as it is for a badAuthentication event. The parent page needs to recognize that error and remedy the problem.

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 event occurs, the user interface for the component 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.

userExit

Applies to Account Linking only.

A component sends the userExit event when the user exits it, such as by completing adding an account, completing editing an account, canceling out of the workflow, or declining the user agreement.

Upon an userExit event, the parent page could hide Account Linking 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 Account Linking at a set point in the workflow.

Managing standard events in Account Linking

These events apply to the Account Linking component only. This section describes managing the standard (informational) trigger events listed in About Events on page 11. These events that may be monitored by the parent page but are handled by the Account Linking component. You may want to monitor standard events to update your own back end.

dataChanged

The dataChanged event lets the parent page know something has changed in Account Linking. The parent page should refresh any display of account, credential, position, and transaction data. The event is emitted each time one of these data changes occurs, so the parent page could receive this event multiple times during a single session of the Account Linking component.

dataChangedAsync

The dataChangedAsync event lets the parent page know the user submitted a composite "aggregate" in Account Linking that triggered the service to run the complex and long-running process asynchronously. The composite "aggregate" can be discover/add/aggregate or just aggregate. When the parent page receives this event it should poll for asynchronous activity completion and then query the API to obtain the changed data.

dataChangedAsyncAuthenticate

The dataChangedAsyncAuthenticate event lets the parent page know that an authentication was initiated in Account Linking. The asynchronous process can be complex and long-running. When the parent page receives this event, it can poll for asynchronous activity completion and then query the API to obtain the authentication status.

userDeclinedAgreement

A userDeclinedAgreement event occurs when a user does not accept the terms of a user agreement when it is presented in Account Linking.

Link accounts with test financial institutions (FIs)

We have test financial institutions (FIs) that enable you to simulate the account setup workflow within Account Linking. These test FIs can help you develop to manage the Account Linking events, and also help you develop retrieval operations of the aggregated data.

The available test FIs include those shown here. Each has data suited for testing different scenarios.

FI Name	Login Name	Password	Multi-factor authentication (MFA) Type	Required MFA value	Comments
Data Test 1 - 401(k) (Investment)	*Any Value*	*Any Value*			This test FI has complete positional data and historical transactional data. There are other data tests (Data Test 2, 3, 4) which can be used as well that support different account types.
TEST: TechFirst Investments (Demo)	*This must be a non-blank value. Must be a new string to*	*A non-blank value*	Activation Code	1234	Useful for testing activation code workflow. For quality positional data use

	receive a prompt for more information. We recommend using 'yourname_number'.	*Use 'badpassword' to cause a 1007 error*			Data Test 1 - 401(k) (Investment).
TEST: ByAllAccounts Demonstration FI #2 (SQA) (Investment)	username	*Any value*	Security Question	Boston	Login and security answer are case sensitive. Incorrect values will cause a login error. This test FI enables you to see the multi-factor authentication (MFA) workflow and you can break it by using an incorrect username or security answer.

Collect aggregated data for an investor

Our service automatically aggregates the data for an account when that account is first set up and then each night after that. An account must have valid credentials to the institution to aggregate. When you retrieve aggregated data from our service you are getting the data from the most recent successful aggregation.

To receive position and transaction data for an individual investor, use these REST API endpoints:

- GET positions GET /api/v1/positions
  Link to Postman for REST API endpoint for retrieving positions for an investor
- GET transactions GET /api/v1/transactions
  Link to Postman for REST API endpoint for retrieving transactions for an investor

For a guide to the BAA REST API and response details, refer to
http://www.byallaccounts.net/Manuals/DataConnect/BAA_REST_API.pdf

Collect aggregated data for multiple investors (batch)

Use the DataConnect Get Data (asynchronous) <DATAGETRQ_A> operation to retrieve a wider scope of data, including personal profile, position, and transaction information, for one or more Investor users in XML format. Additionally, use the DataConnect <DATACLAIMRQ> operation to poll for completion and collect the resulting data.

- Link to Postman for DataConnect operation to retrieve data for all investors - Step 1, featuring <DATAGETRQ_A>. This example uses a firm-level data scope, not an individual investor. It includes tags for user, account, holding, and transaction. Other tags can be used to include additional information.
- Link to Postman for DataConnect operation to retrieve data for all investors - Step 2, featuring <DATACLAIMRQ>.

Note: For DataConnect operations in Postman, use "Send and Download" then save the response as a .zip file. The response XML is inside that .zip file.

For more about these operations, refer to the DataConnect V4 Ultra User Guide. For <DATAGETRQ_A> refer to the "Get Data (Asynchronous)" section. For details about <DATACLAIMRQ>, refer to the "Asynchronous Helper Operations" section.

Customize the user interface

After you integrate the components, you may consider modifications such as customizing the user interface by setting attributes, including those that refer to customized .css and font files.

There are multiple types of interface customizations you can optionally provide for the components to seamlessly blend their appearance into your parent page. Customizations must be explicitly set individually on each component. Apply these customizations using the files provided in the *assets* folder for Account Linking and the *as_assets* folder for Account Summary.

- **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. Configure the component to use your custom terminology file by setting the *translate-file-path* attribute. Note that the translation file for each component has a different set of terms.
- **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 component 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;
  }
```

- **Fonts** – define an optional fonts file to override the default Morningstar Design System (MDS) fonts for elements in the display. Configure the components to use your custom fonts by setting the *custom-fonts* attribute. These new fonts are then available to be set using the CSS file specified using the *override-css-file* attribute.
- **Optional features** – *For Account Linking only,* the config/ui-config.json file contains options that enable you to turn on/off UI features in Account Linking. Configure Account Linking to user your custom

configuration file by setting the *ui-config-file* attribute. Only a subset of the options currently applies to the component. These options are:

- "fiLogosVisible" - controls whether a page of popular financial institution (FI) logos is shown in the FI selection step. If true, the logos are shown; if false, then only a search control and FI list are shown.
- "includeURLinSearch" - controls whether the FI URLs are included in the search criteria when the user is searching for an institution.

  For example:

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

Applying the user interface customizations

Use these instructions to customize terminology, styles, fonts, and optional features.

1. Edit the files provided in the *assets* and *as_assets* folders. These files and edits are described in Customize the user interface on page 19.
2. Modify your web server to allow our origin (www.byallaccounts.net) to access those files. We suggest the following:
   - Header set Access-Control-Allow-Origin "www.byallaccounts.net"
   - Header set Access-Control-Allow-Headers "Cache-Control,Content-Type,Accept,Referer,User-Agent,Sec-Fetch-Dest"
   - Header set Access-Control-Allow-Methods "GET,OPTIONS"

   If this access is not allowed, then the components will not be able to access the customization files and each will have to use its own version of those files.
3. Set the appropriate attribute to identity each custom file. For attribute information see About Attributes on page 6.

Using AccountView for troubleshooting

AccountView can be used as an effective troubleshooting tool for data and connectivity errors.

Best practices for "running as" Investor

- Access this URL https://www.byallaccounts.net/BAAWebApp/BAALogin.html.
- Use the "technical support administrator" login you received from your implementation manager.
- Choose to run as Investor.
- Use the internal ID (Person ID) to identify the investor to run as. Alternately, use “Search for User” and leave fields blank and the interface will show all investors you can choose to run as.

AccountView for troubleshooting guidance

Refer to these resources to help use AccountView for troubleshooting aggregation and data errors.

- Key Areas in AccountView -- Reference Guide
- Troubleshooting Aggregation Errors -- Video, Transcript

- Data Troubleshooting – Video, Transcript
- Troubleshooting Aggregation and Data Issues FAQ – FAQ

These resources and more are available on the AccountView Training page. Focus on the section "For Support and Administrative Roles".

Getting help and more information

Your implementation manager is ready to help you.

Related documentation

To access all the power of ByAllAccounts Connect Suite, you will want to review information about single sign-on (SSO), our DataConnect API, and our REST API, etc. Here are some helpful guides:

- For more details about our REST API, refer to http://www.byallaccounts.net/Manuals/DataConnect/BAA_REST_API.pdf.
- Our *DataConnect V4 Ultra User Guide* a provides additional detail about operations described in this guide as well as additional operations you may wish to use depending on your needs http://www.byallaccounts.net/Manuals/DataConnect/DataConnect_V4_Ultra_User_Guide.PDF.
- Although it provides many details not relevant to ByAllAccounts Connect Suite, the guide for AccountView SSO may provide some helpful information about SSO http://www.byallaccounts.net/Manuals/Accountview/AccountView_SingleSignOn.pdf.
- Postman Collection of DataConnect and BAA REST API calls to get started with ByAllAccounts for Investors.

Cumulative Changes

This section keeps a running list of released versions and a summary of the changes integral to the implementation.

Date	Change	Description
November 30, 2023	Postman update	DataConnect base URL was incorrect in a few places in Postman. This error has been corrected.
October 17, 2024	Documentation change	Product names have changed and are now prefixed with ByAllAccounts.
December 12, 2024	Documentation change	Added content for SAMLAUTHRQ here and in Postman Collection. Updated links to Postman collection.