Build an App Workshop

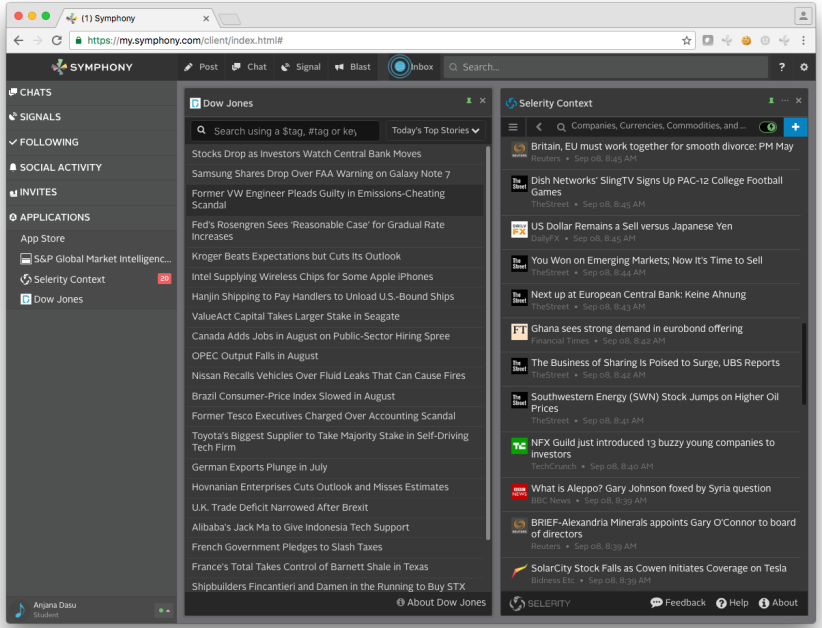
Symphony Platform

Apps Overview

Use Cases and Management

Overview

- Developers can use Symphony's **Client Extension API** to build standalone applications that are embedded within Symphony's user interface.
- Dow Jones and Selerity are examples of Symphony apps that provide feeds of the latest business news.



Who should build an app?

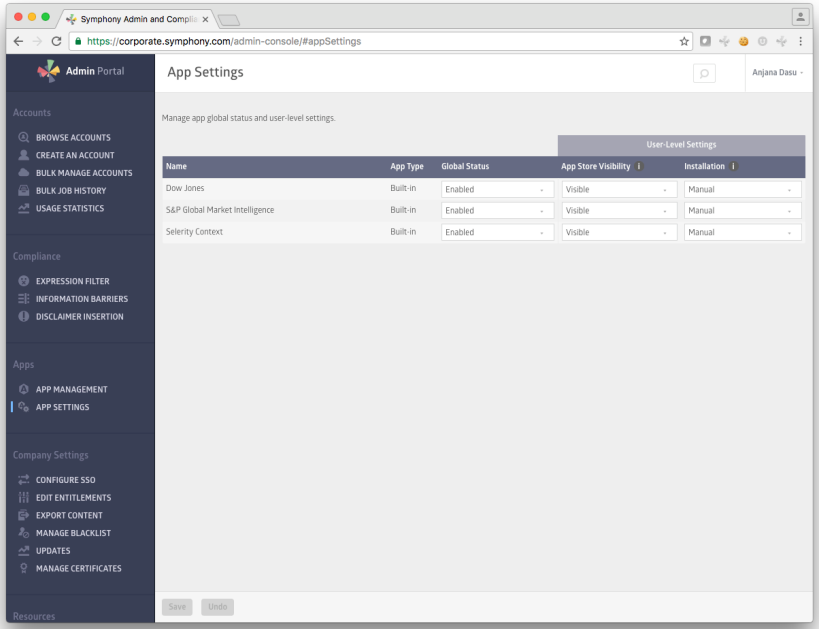
- **Partners / app developers** who want to provide value to Symphony's customers through new functionality
  - Free apps
  - Freemium/Premium apps
  - Ex. News, market data, charting
- **Enterprise organizations** who want to provide tools to their end-users that often tie into existing internal systems
  - Ex. Research portal, CRM, task management

Distribution and Management of Partner Apps

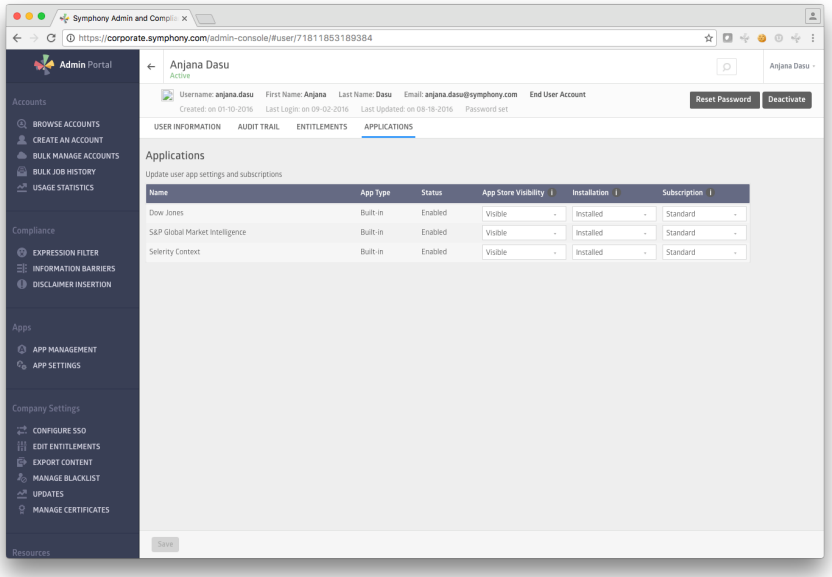
- Partners work closely with Symphony's Partnerships and Platform teams to release apps on Symphony
- **Symphony deploys partner apps** to all of Symphony's customers
- Symphony administrators can choose whether to **enable** them for their organization
- Apps can generally be accessed from Symphony's app store
- Once enabled, admins choose how apps are made available to end-users
  - **App Store Visibility: Visible or Hidden** – whether the app is displayed in the app store for the user to install/uninstall
  - **Installation: Automatic or Manual** – whether the app is automatically installed for new users or must be manually installed from the app store
- Admins can also manage app settings on a per-user basis

Enterprise Admin App Management

Company-Level App Settings

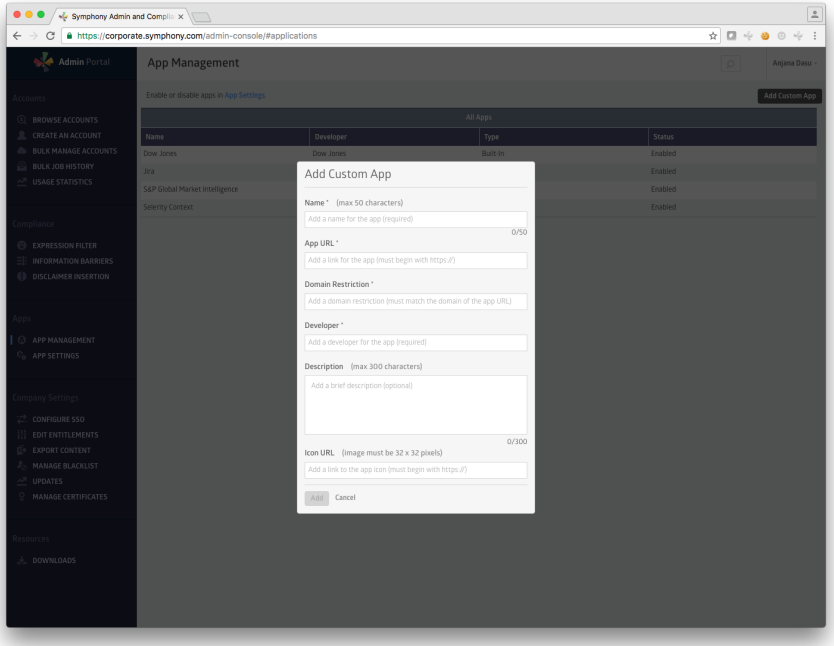


User-Level App Settings



Distribution and Management of Enterprise-Custom Apps

- **Enterprises deploy their own custom-built apps to their organization** from the admin portal
- Admins manage these apps – at the company and user level – just as they do partner apps



Building an App

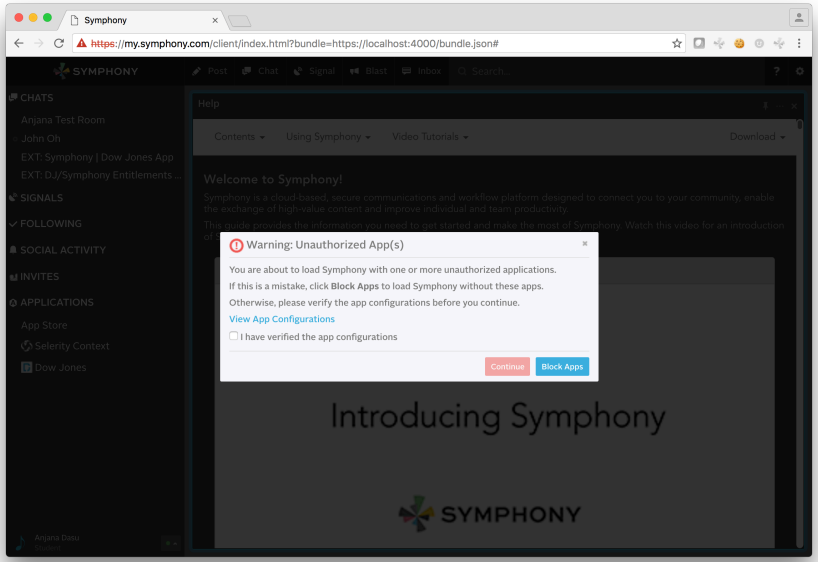
Technologies and Walk-through

Building an app using the Client Extensions API

- The **Client Extension API** is a JavaScript API that consists of several **remote** services for interacting with different parts of Symphony's UI – referred to as services
- Applications can be built using any web development framework of your choice
- Some Symphony service methods will require you to implement your own **local** services with methods to handle events
- Services are used for communication between your app and the Symphony client.
- Apps created with the Client Extension API run in **iframes** inside the Symphony client
- The main application iframe is called the **application controller** – it has no visible UI but uses the Client Extension API services to extend Symphony's user interface by adding new elements like modules, left navigation items, module buttons, and hovercard links
- In most cases, the contents of the application are created in a separate iframe called the **application view** which is rendered inside a Symphony module
- Developers can also implement **remote** services that are shared between application controller and view

Developer Mode

- Symphony provides a mechanism for loading applications in a development environment
- Create a **bundle.json** file in your application that specifies your application: id, name, description, url, domain, icon, description, etc.
- Enable the #allow-insecure-localhost flag in Chrome by enabling the after navigating to **“Allow invalid certificates for resources loaded from localhost”** chrome://flags
- Load your app on your pod by going to https://your-pod.symphony.com/client/index.html?bundle=https://path/to/bundle.json
- Accept the dialogue asking you whether you would like to load Symphony with unauthorized apps



Code Walk-through

1. Setting up your application
   a. Include Symphony's JavaScript API
   b. Create local service(s) for handling events
   c. Register your controller – subscribe to remote services and register local services
   d. Connect any other application services – subscribe to remote services and register local services
2. Obtain a userReferenceId for saving settings
3. Add a left navigation item that opens an app module when clicked
   a. Focus the app module
4. Add theming, pinning, and menu items to your module
   a. Respond to theme changes
5. Register extensions (buttons and links) on various parts of the Symphony UI
6. Open a new module in the context of a cashtag extension
7. Share articles from your app and link back into your app

Resources

- Client Extension API Documentation: https://extension-api.symphony.com/
- Client Extension API Examples on Symphony Foundation GitHub: https://github.com/symphonyoss/extension-api-examples
- Email questions to Symphony Foundation Community Support: dev@symphony.foundation
- Email questions to Symphony developer support: developers@symphony.com

Thank You

Symphony Platform