Welcome to your SAP Learning Class

SAP Build App

SAP

Public

THE BEST RUN SAP

Learning Objectives & content

Class Objectives

- Describe the characteristics of LCNC development platforms
- Open the project and preview app
- Use components to build the UI
- Build the app logic flow
- Bind data variables to UI components
- Work with bindings and formulas
- Make an HTTP request with an API URL
- Set up the API data resource
- Explain variables
- Check that your weather app works

Course content

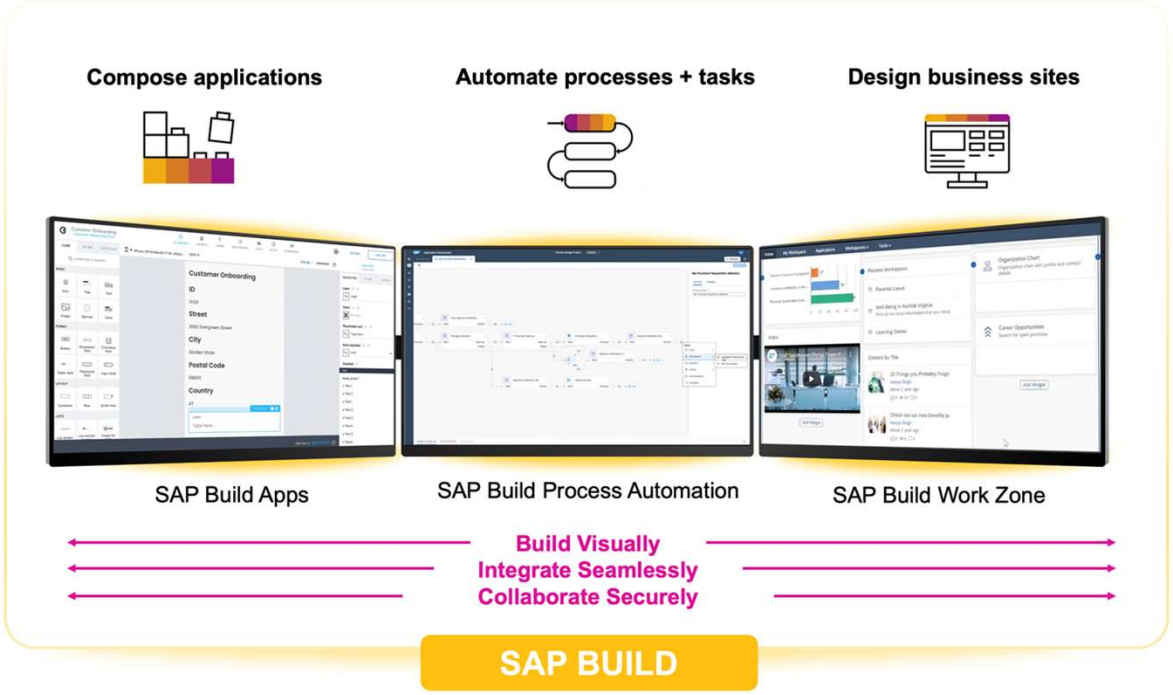
- Overview on SAP Build APP

Logon Details and Information

Specific information for this class

- Sign up for SAP Build Apps Sandbox: SAP Build Apps Lobby (ondemand.com).
- Download SAP Build Apps application on your mobile devices.
  - For iOS: SAP Build Apps Preview on the App Store (apple.com)
  - For Andriod: SAP Build Apps Preview - Apps on Google Play

Introduction – SAP Build for Citizen Developers



Citizen developers are business users with little to no coding experience that build applications with IT-approved technology.

SAP Build brings together SAP Build Apps (formerly SAP AppGyver), SAP Build Process Automation (formerly SAP Process Automation), and SAP Build Work Zone (formerly SAP Work Zone) into a unified development experience with new innovations to rapidly build apps, automate processes and create business websites. With SAP Build, developers can integrate smoothly with SAP and non-SAP applications and leverage hundreds of prebuilt processes, bots and UX components to unify business data and processes.

Introduction – Low-code and no-code (LCNC)

Low-code and no-code (LCNC) refer to a new style of visual programming that makes it possible to develop applications without the use of coding languages. To differentiate between the two aspects:

- **Low-code** uses both a traditional programming language-based environment combined with no-code platforms and is used by developers with at least basic technical knowledge (examples of low code are framework such as SAP Cloud Application Programming Model). Low-code tools enable developers to create apps quickly and with a minimal amount of manual programming.
- **No-code** is simpler, and it fully replaces the traditional programming language-based tooling with a suite of visual development tools (ex. drag-and-drop components) and can be used by technical and non-technical people alike, such as citizen developers. No-code tools, which are aimed toward users who are unfamiliar with any programming languages but still want to develop an app for a specific use case

Low-code and no-code modular approaches let professional developers quickly build applications by relieving them of the need to write code line by line. They also enable business analysts, office administrators, small-business owners and others who are not software developers to build and test applications. These people can create applications with little to no knowledge of traditional programming languages, machine code or the development work behind the platform's configurable components.

Introduction – What is SAP Build Apps

SAP Build Apps (former SAP AppGyver) is a visual development environment that allows citizen developers and professional developers to use drag and drop capabilities and minimal coding to create web or mobile applications with **user interfaces, business logic** and **data models**.

- You can build and customize enterprise applications visually using pre-built drag-and-drop components. And, despite the fact that they are pre-built, you can add custom functionality to those components depending on your requirements. Design beautiful user interfaces without coding. Create data models and business logic visually.

- Connect to SAP and non-SAP applications; direct integration with SAP systems; integrate with applications, processes and data across diverse systems. Integrate modern APIs in minutes with the REST integration wizard. Secure access to SAP and 3rd party data using SAP BTP Authentication for login and access control.

- Foster collaboration between fusion teams.

Documentation available at https://docs.appgyver.com/docs/home

Logon Details and Information

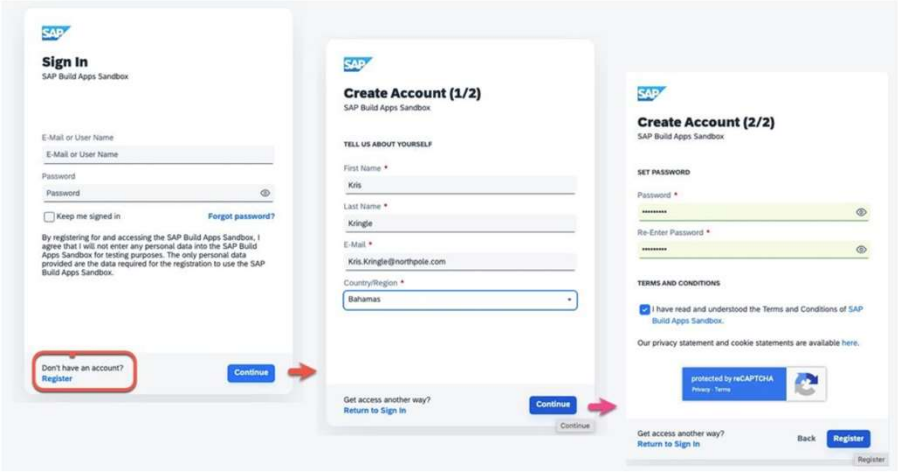
Sign-up and Use SAP Build Apps Sandbox

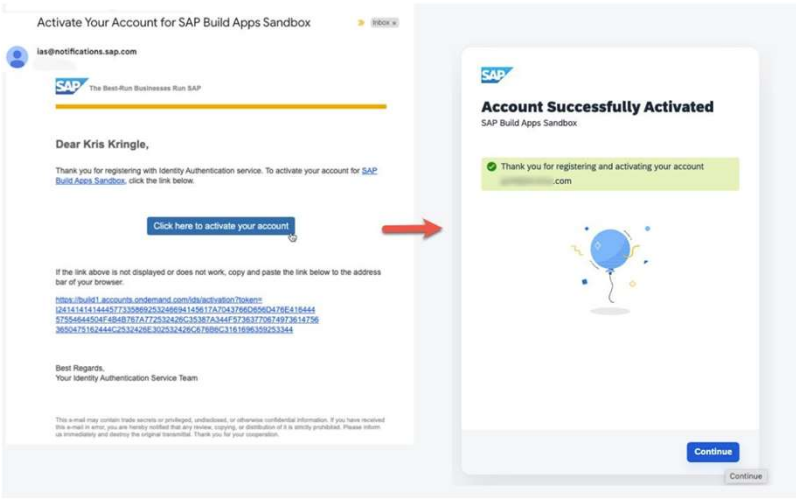
1. The entry point for the sandbox is the SAP Build lobby. (https://build-sb1.eu10.build.cloud.sap/lobby)

2. As a new user, you will need to register for an account. Only a name, email address, and country are required.

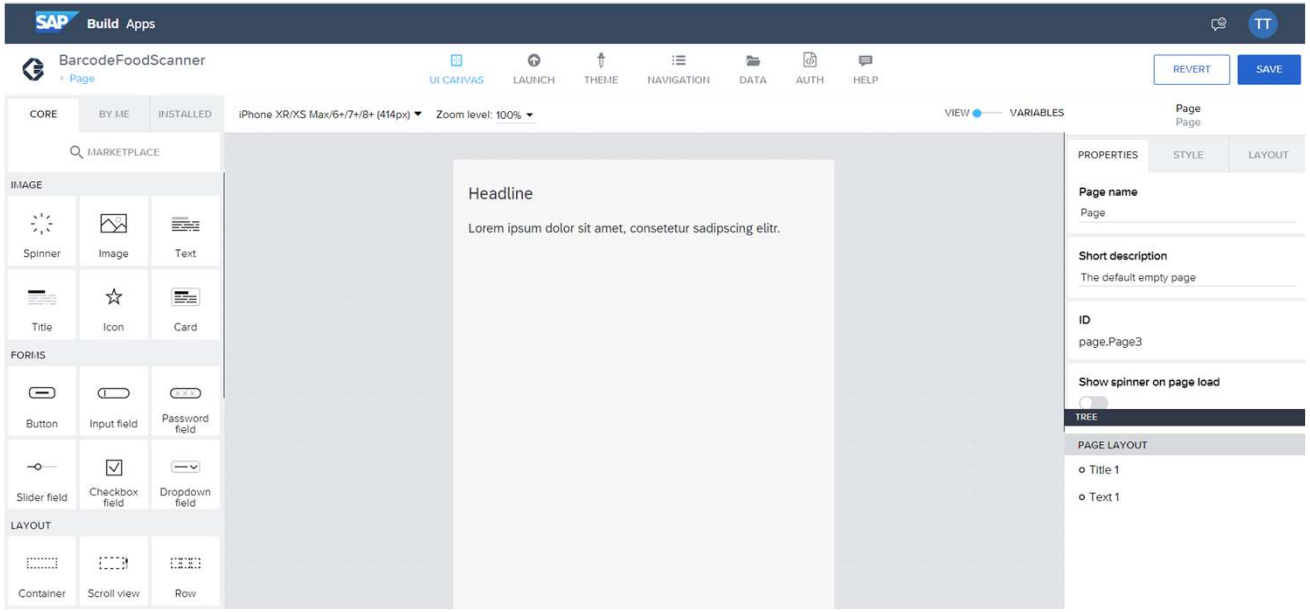
3. Wait for an email and then activate your account.

4. You can start building apps immediately. Web & Mobile Applications as well as Application Backends are available.





SAP Build APP - UI Canvas – View components



A **view component** is a UI control and can be *primitive* or *composite*.
A *primitive component* can be a simple UI control such as an input field or a button; a *composite component* is basically a combination of a container and one or more primitive component. A *container* can hold multiple child components (primitive or/and composite)

The component library panel gives you easy access to the various view components you can add to your app. It takes up the top half of the left sidebar. The panel is divided into three tabs.

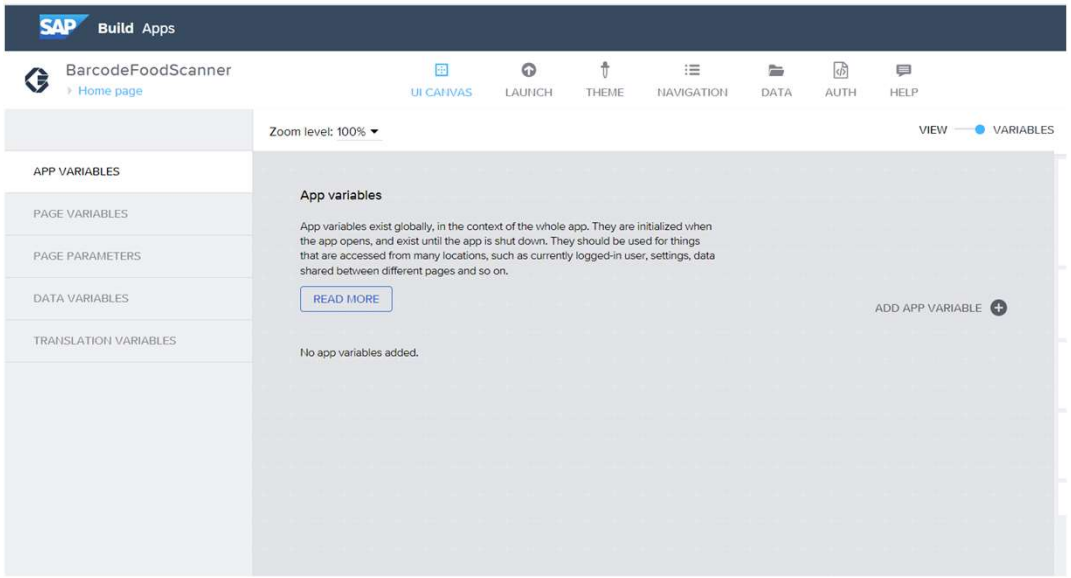
- **Core** – core components by SAP Build, common in most apps
- **By me** – components created by you for this app
- **Added** – components added from the Marketplace for this app

To add a component into the app page there are two ways:

- Drag-and-drop onto the view canvas
- Drag-and-drop onto the layout tree

Both methods work the same way regardless of the component library panel tab you are on.

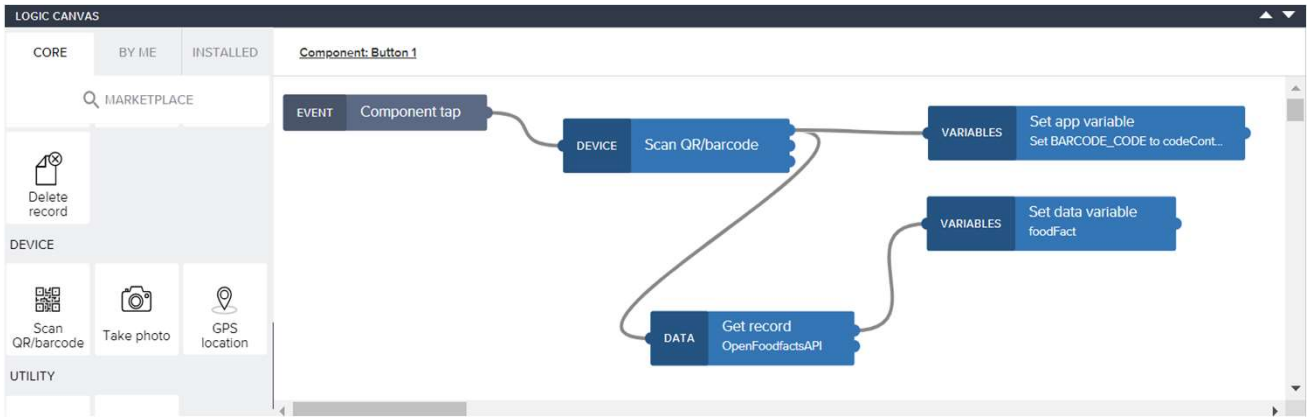
SAP Build APP - UI Canvas – Binding: Variables



A **variable** is just some value we're interested in storing, which will change over time.
Variables can contain anything from numbers to text to much more complex data. Variables are typically only stored while the application remains active, and are reset when the application is shut down.

- **App variables:** they serve as global variables which means that the variables remains accessible across the app (ex. Site id, plant code, material code)
- **Page variables:** are local to individual pages, they are instantiated when the page is created and expires when the page is closed.
- **Page parameters:** suitable for storing read only text data, expecially for passing data from one page to another
- **Data variables:** are generated as a result of reading data from a data resource, like a data source or an external API.
- **Translation variables:** these variables are dedicated to taking care of the internationalization of the UI by translating the text bound to component properties at runtime based on the current language.
- **Sensor variables:** these special variables can help you access native device data such as geolocation capabilities, compass, accelerometer etc.

SAP Build APP – Logic canvas – Application logic

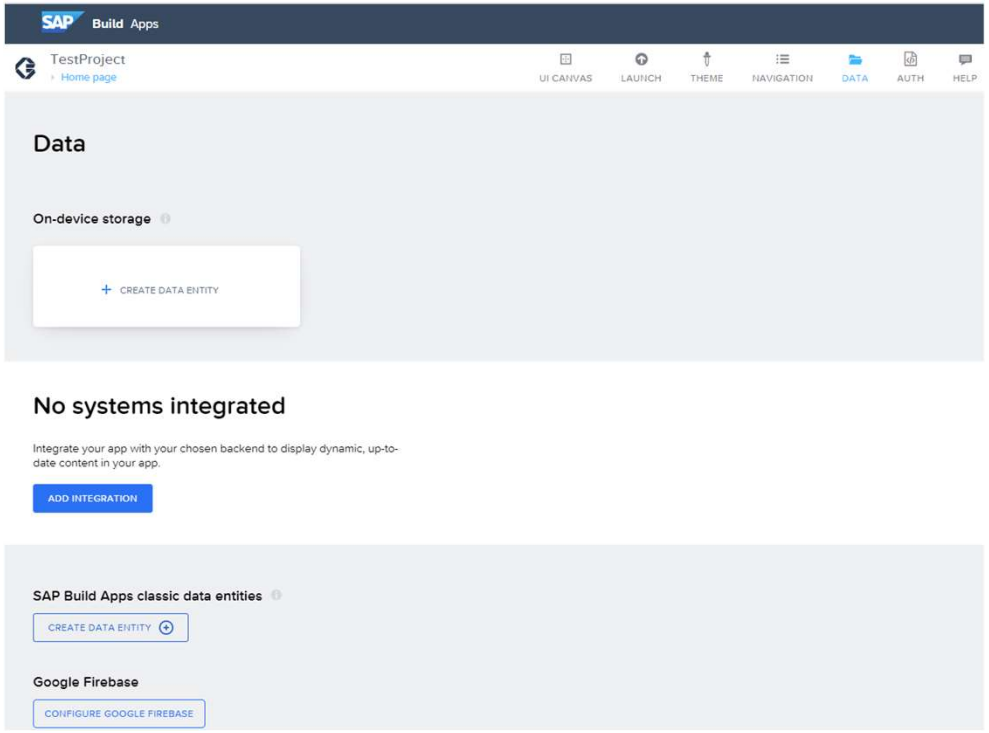


Application logic is what makes application do things. Logic is fundamentally based on business rules that are set to produce an outcome, and the requirements for those should be well understood during the planning process.

To ensure that business rules are appropriately met by the application, the supporting logic needs to be defined as a flow of events, actions, and functions. Events are triggers, actions are responses, and functions produce a result.

- **Events:** anything that happens (page loaded, button being clicked etc)
- **Flow Functions:** core functions forming the no-code based business logic. They address almost every technical aspect of app development like variables setting, data retrival, navigation, access to device native feature, notification, custom scripting and media content.
- **Formula Function:** Formulas are used in bindings of components, and enable you to manipulate data before it is displayed in a component or before it is sent to a data source.
- **Binding:** Is needed to connect variables values to a UI component

SAP Build APP – Data management

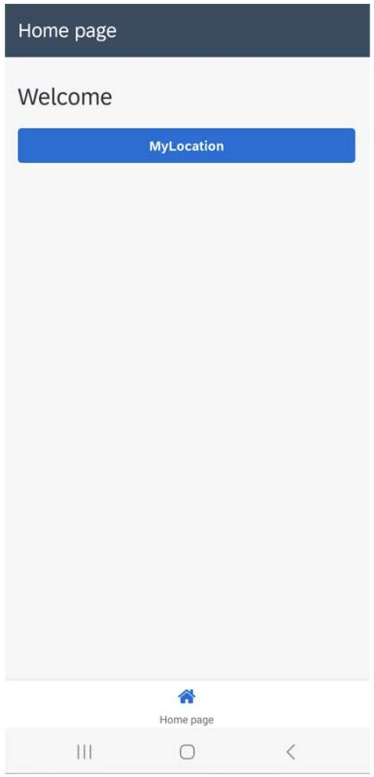


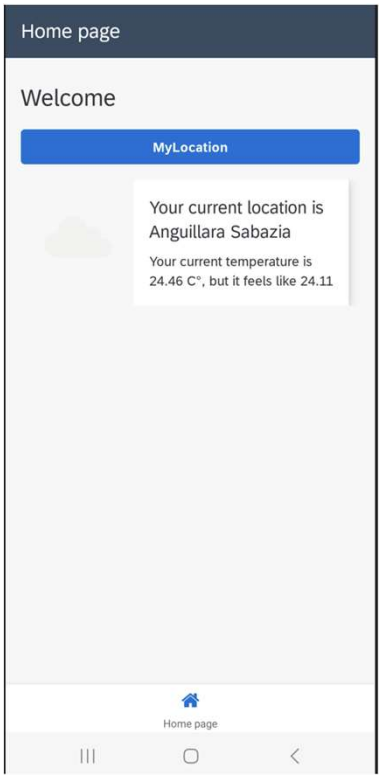
Data in the context of SAP BUILD App, means all those data that are stored outside the context of the running app. Typically, this means that the data is available online via an API, but it could also be stored on the device itself as a client-side storage. A **data resource** defines access to such data (ex. client-side storage resource or REST API integration).

Available data resources:

- On-device storage – data stored on the device/web browser itself; cannot be shared between different devices
- REST API direct integration – direct integration to an external JSON-based API
- OData Integration
- Google Firebase
- Custom resources found on the marketplace

Exercise – Weather App





Documentation on Weather API:

https://openweathermap.org/current

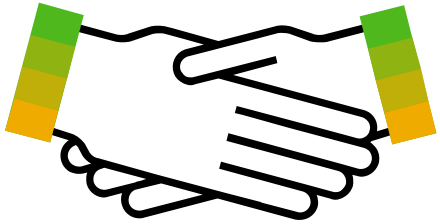
Ex:

GET
https://api.openweathermap.org/data/2.5/weather?lat=41.90&lon=12.49&appid=ba007f58593d774bb4bea4865969f0cf&units=imperial

Icons API description

https://openweathermap.org/weather-conditions

Thank you.



Thank you for your attention, enjoy your SAP Learning Class