



appli

Documentation

TABLE OF CONTENTS

1. The Login Screen
2. The Project Selection Screen
3. Overview of Appli IDE Interface
4. Tools palette
5. Keyboard Shortcuts
6. Screen Management
7. Property Inspector
8. Responsive Design
9. Low-Code & Action Scripts
10. Data Management
11. Tutorial: Cool Coffee Shops
12. Elements
13. Element: Browser
14. Element: Button
15. Element: Camera
16. Element: Create Account
17. Element: Dropdown
18. Element: Field
19. Element: Graphic
20. Element: Image
21. Element: Layout
22. Element: Login
23. Element: Map
24. Element: Radio Group
25. Element: Search Field
26. Element: Switch
27. Element: Tab Menu
28. Element: Table
29. Element: Text
30. Element: Media

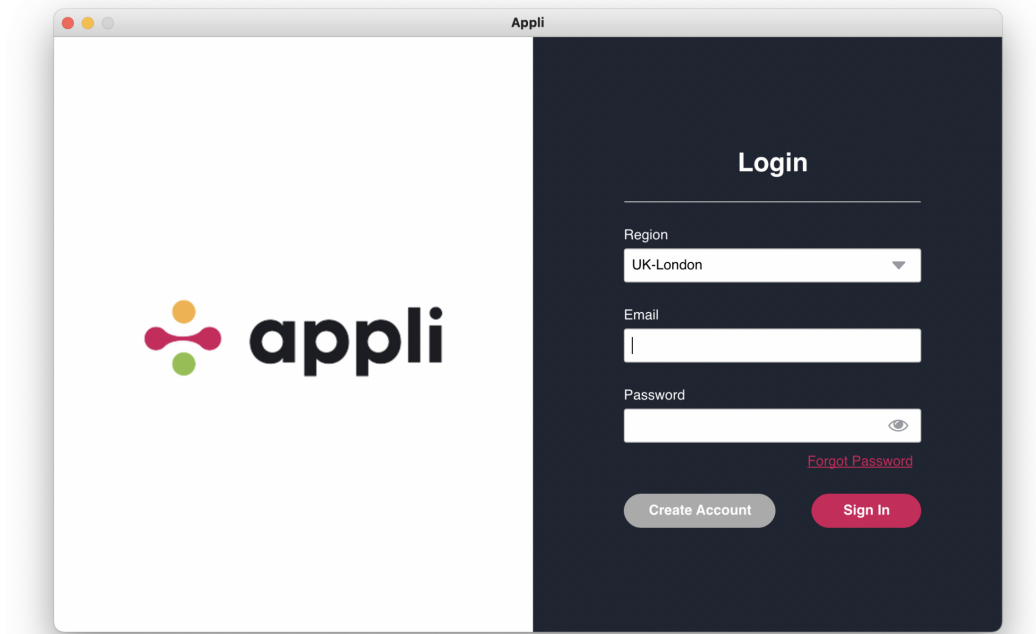
31. Element: Form
32. Element: LineGraph
33. Image Credits

DISCLAIMER

This document is a draft version of the Appli documentation. The content is not complete. It is a beta version of the material we are building. We see value in sharing this in this early form because it is already useful for our early users, and also because it helps us gather feedback as we're building it.

Please, send your comments, feedback, and issues to our editorial team.

THE LOGIN SCREEN



Login Screen

The first screen you see when you launch Appli IDE is the login screen. It can be used to create new accounts or to sign into an existing account. To use Appli IDE, fill in your login information, making sure you selected the correct region for your account and click *Sign In*. Once you sign in, you'll be shown the Projects Screen.

ACCOUNT CREATION

Accounts are unique to each region, so make sure you have selected your correct region first before filling in the login information.

Appli

Create Account

🌐 Region: UK-London ▼

first name

last name

email

phone number (optional)

password

confirm password

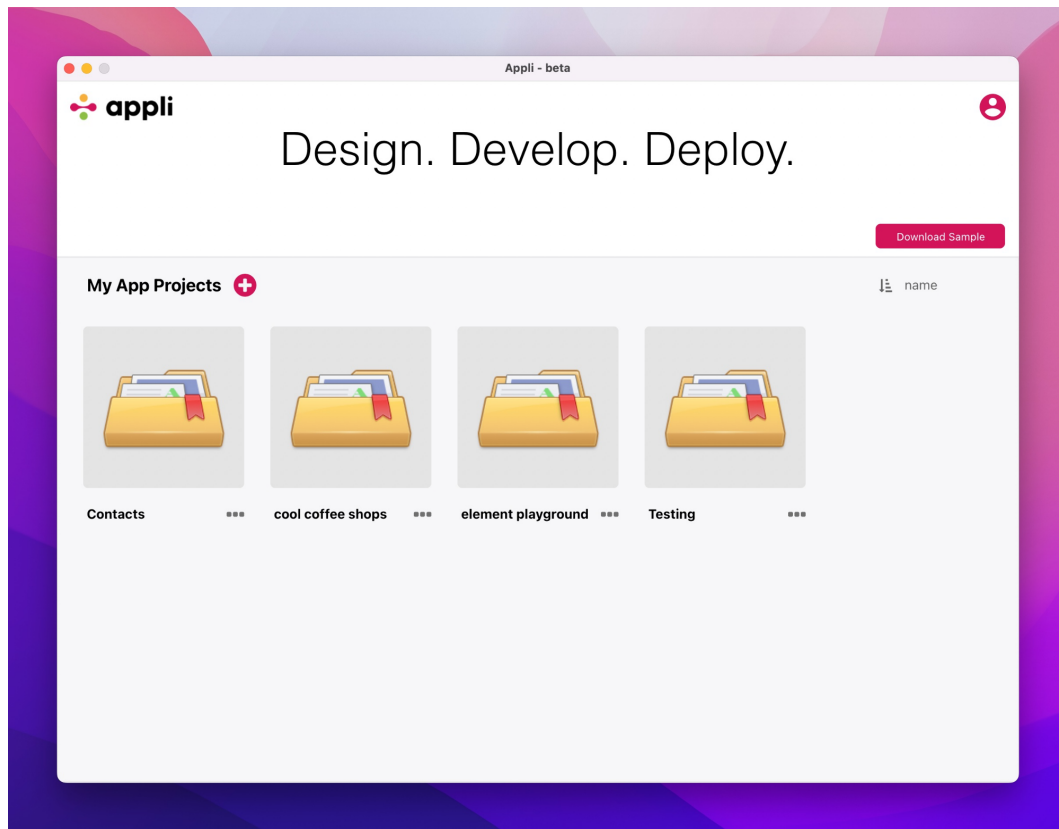
☐ Show passwords

Create Account

Already have an account? [Back to Login](#)

Create Account Screen

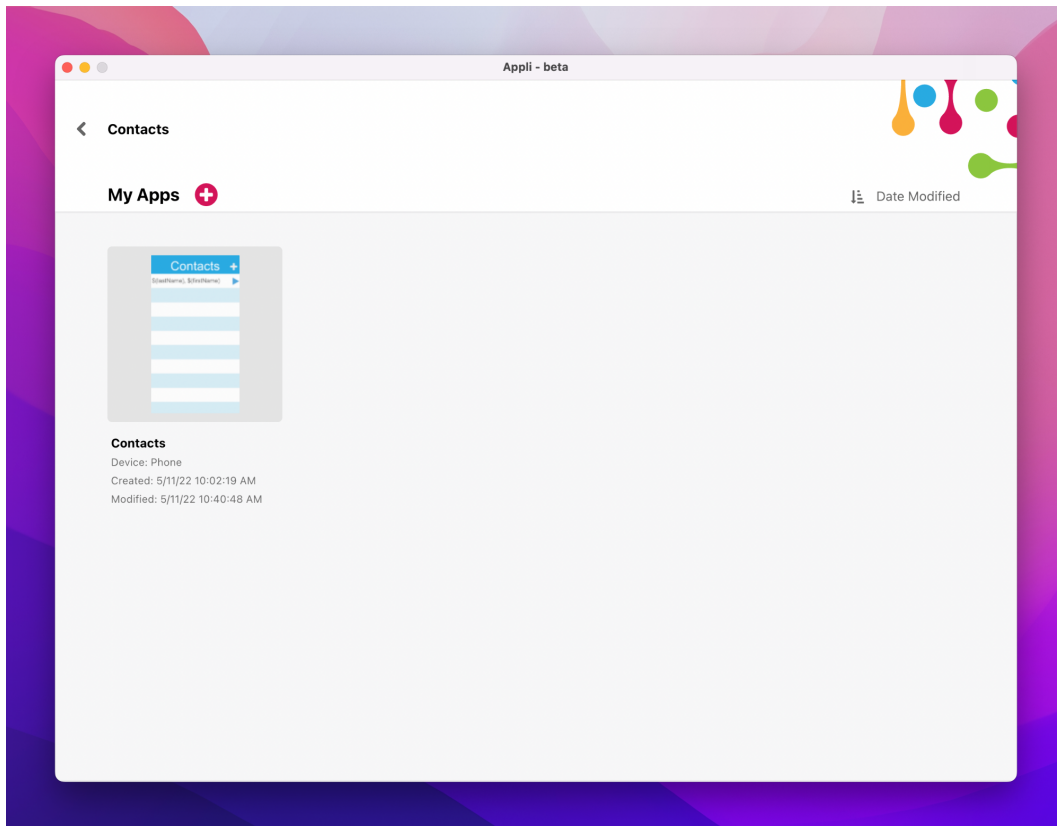
THE PROJECT SELECTION SCREEN



Project Selection Screen

This screen is used to create or open projects. Projects can contain multiple applications. The bottom half of the screen shows your current projects.

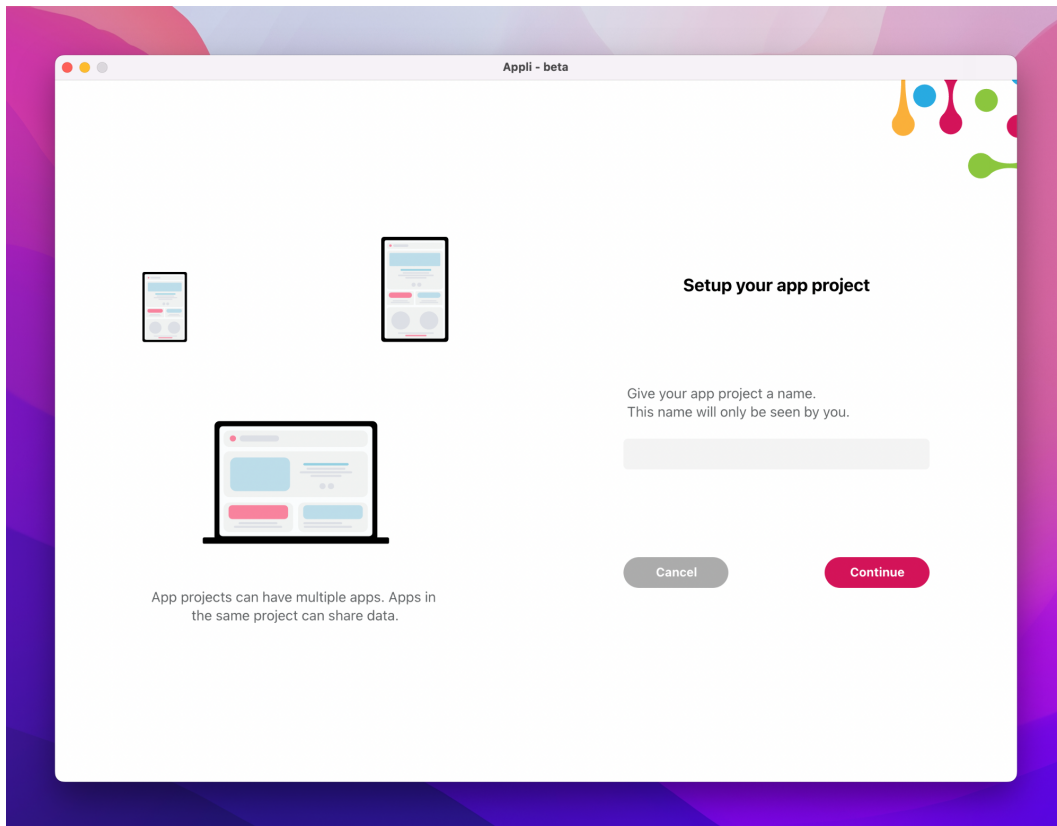
Clicking on an existing project let you select which application from that project you want to open in the Playground. To change how your projects are listed, use the *Sort* dropdown on the right side of the project selection area.



Application Selection Screen

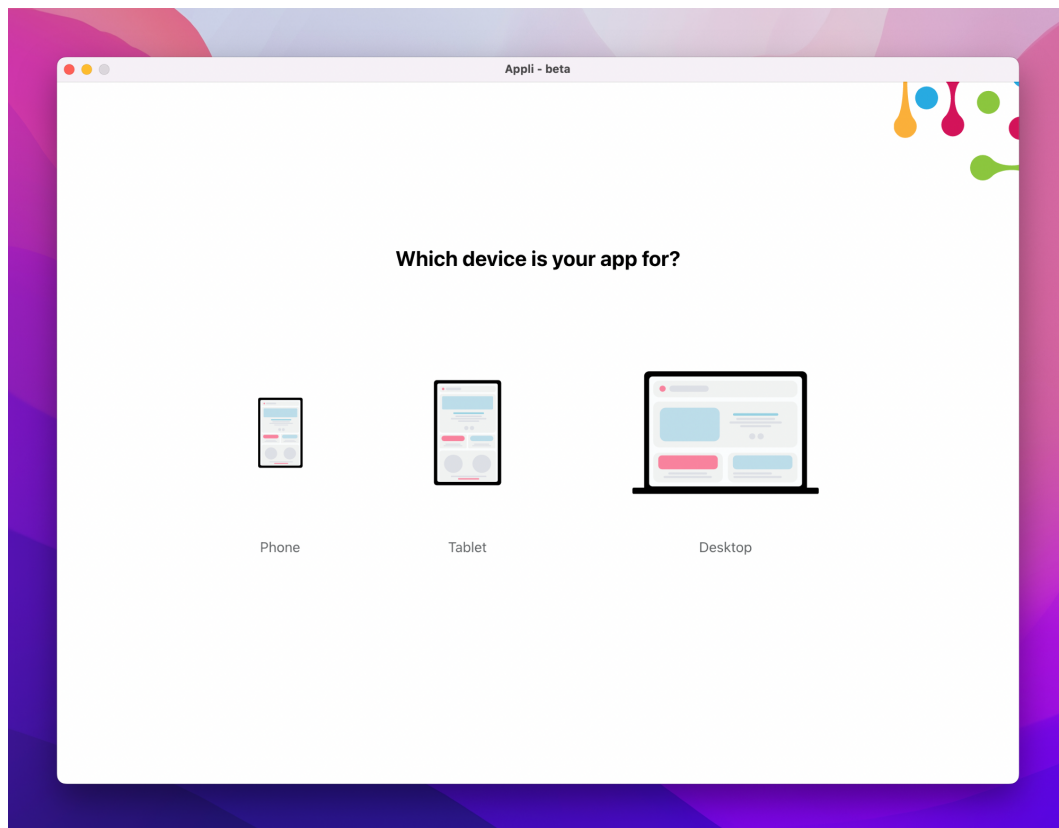
PROJECT CREATION

Use the *plus* button next to *My App Projects* label to create a new project.



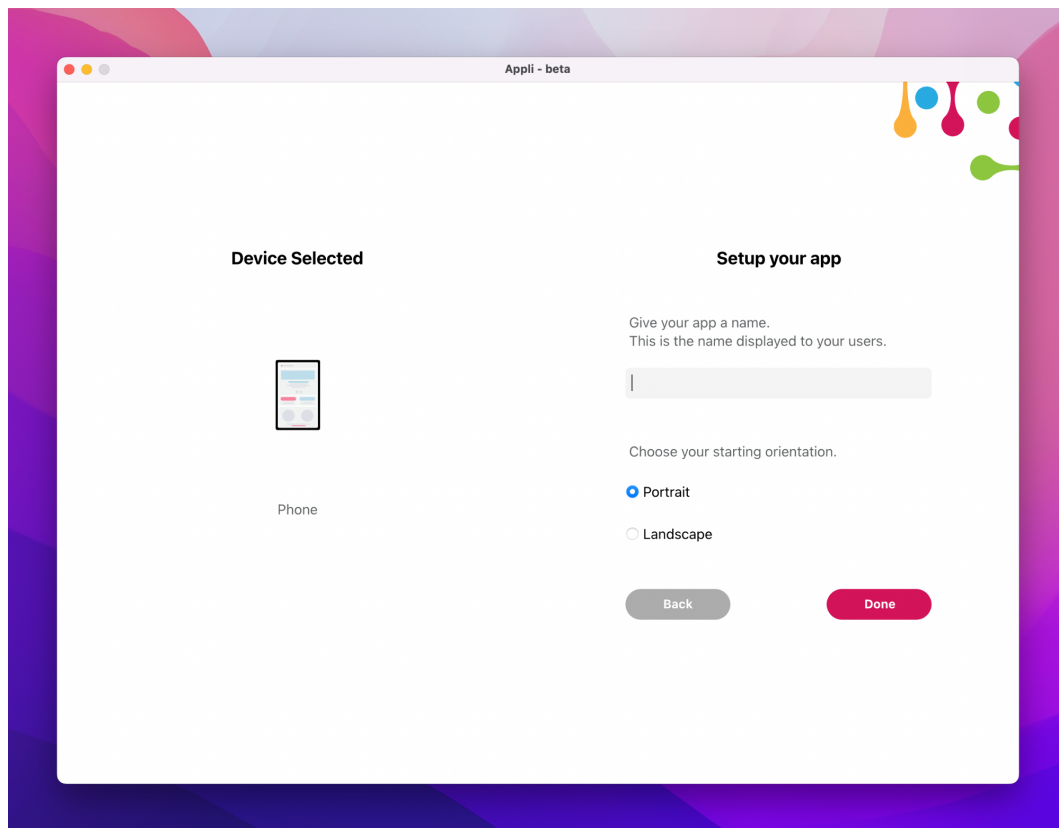
Project creation dialog

Once you give the new project a name, choose a platform for your first application in that project. You can have many applications per project. This allows you to support many platforms. You can add more platforms later.



Platform selection

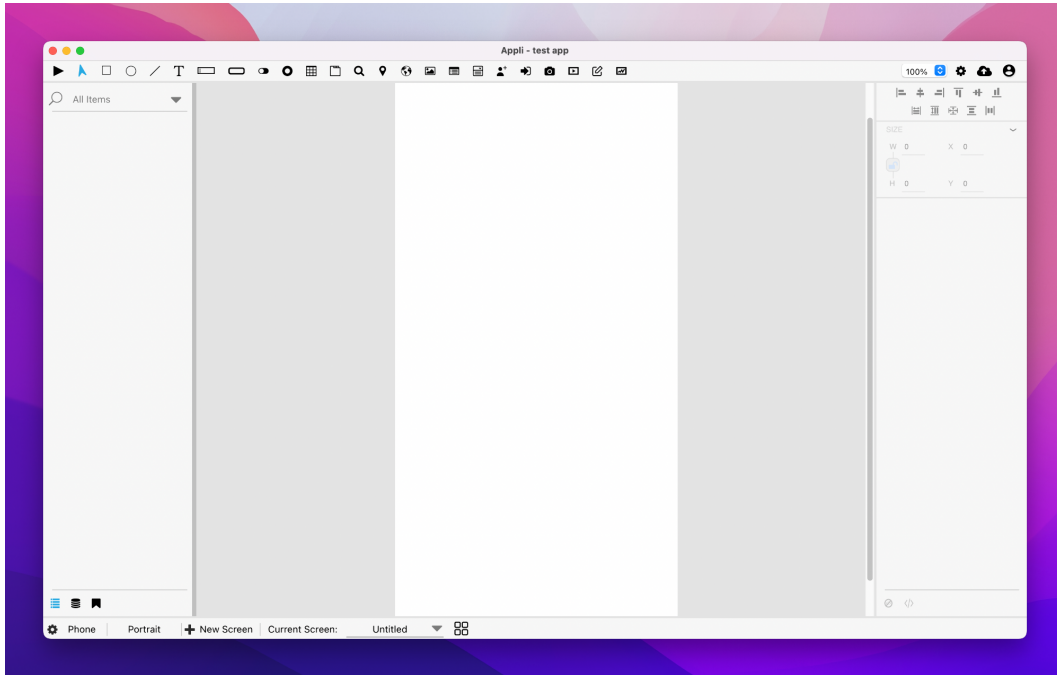
After clicking on a platform, Appli IDE displays a dialog allowing you to set the name of the new project, a description, and the initial orientation.



App creation dialog

Once the project is created, Appli IDE will load the new project into the Playground.

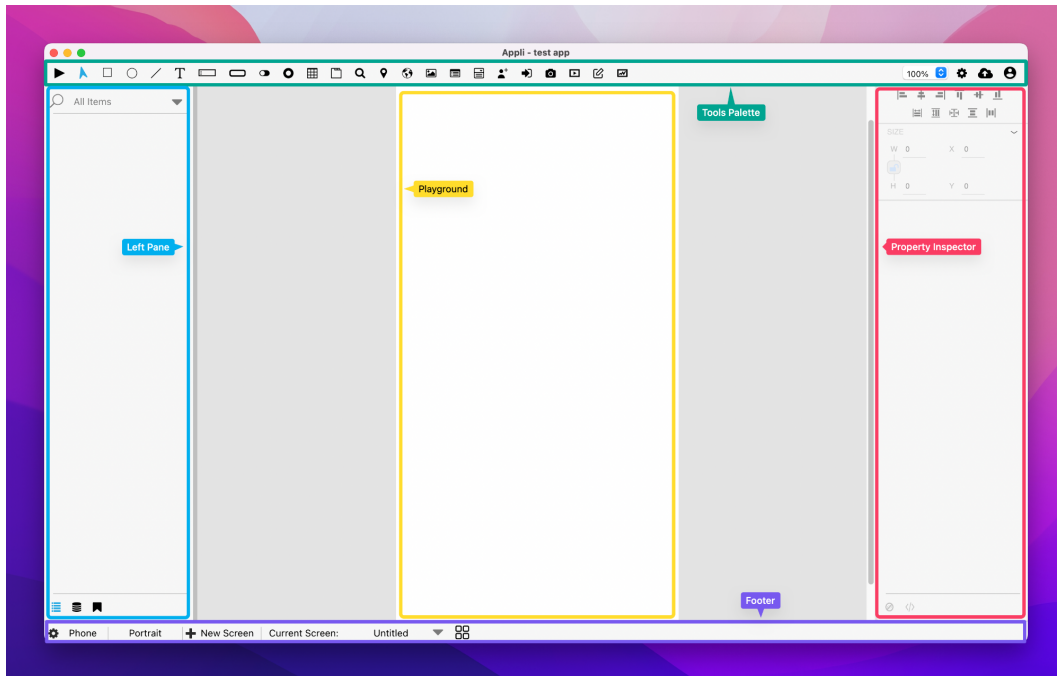
OVERVIEW OF APPLI IDE INTERFACE



Playground Screen

Welcome to the primary interface of Appli IDE. This is where you'll spend most of your time as a developer working on your next awesome project. This section will help you become familiar with the various features and workflows inside the playground.

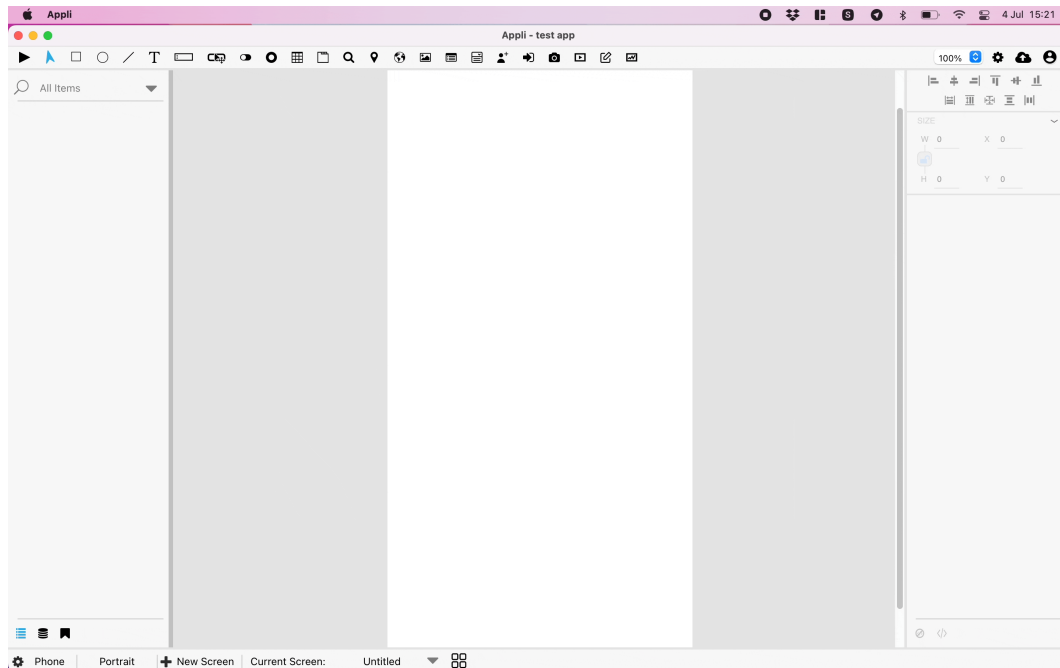
The playground screen is divided into five regions: tools palette, project browser, playground, property inspector, and footer. In the next sections, we'll provide a high-level overview of each of these regions with links to dive deeper into each of them.



Playground Regions

TOOLS PALETTE

The tools palette on the top of the interface holds the *elements* that the developer uses to build their own application. To add an element to the playground, first select the element, and then use the mouse to create it by clicking and dragging a rectangle that represents its dimensions on the playground.



Adding an element to the playground

On the right side of the tools palette is a collection of buttons to enable to you to:

- Save your project to the cloud.
- Change application-wide settings.
- Handle workflow related tasks such as signing out, going back to the home screen,.

To learn more about which elements are available for you, check out the [elements gallery](#).

Check out the [Tools Palette documentation](#) to learn more about the tools palette itself.

PLAYGROUND

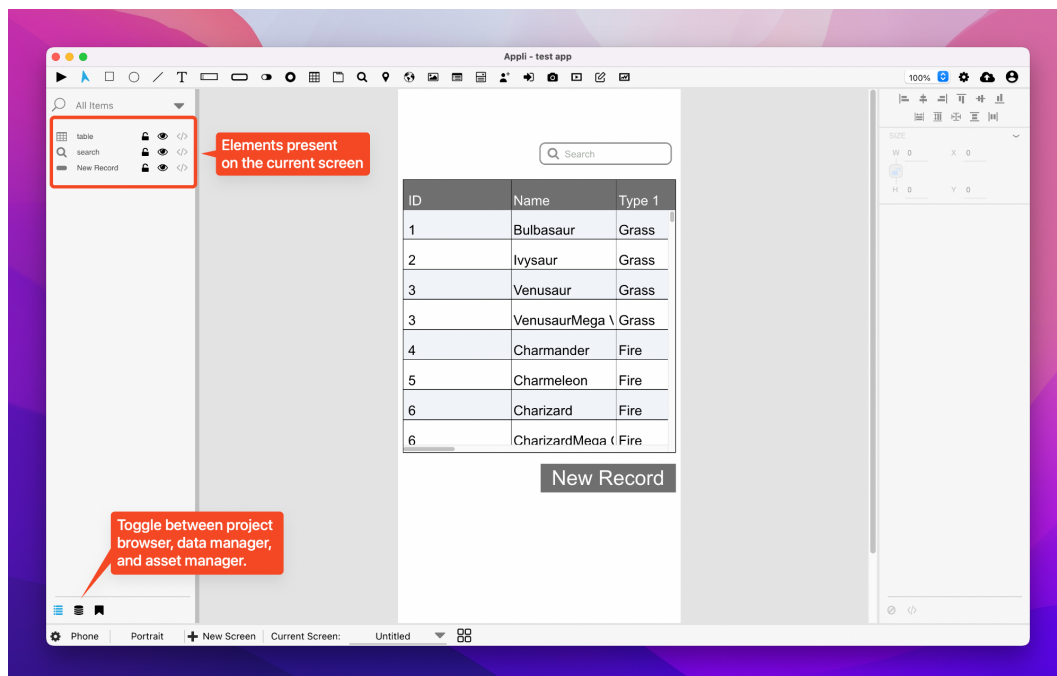
The playground is the most important region in the Appli IDE interface, which is why we call this screen *the playground screen*. This is the canvas in which the developer will build their application.

THE LEFT PANE

The left side of the IDE hosts three important tools: the project browser, the data manager, and the asset manager. To switch between them, use the buttons at the bottom of the pane. The first button switches to the project browser, the second selects the data manager, and the last one activates the asset manager.

Project Browser

The project browser lists all the elements on the current screen as a hierarchical list. Some elements are containers. In such cases, the elements they contain will appear under them on the list.



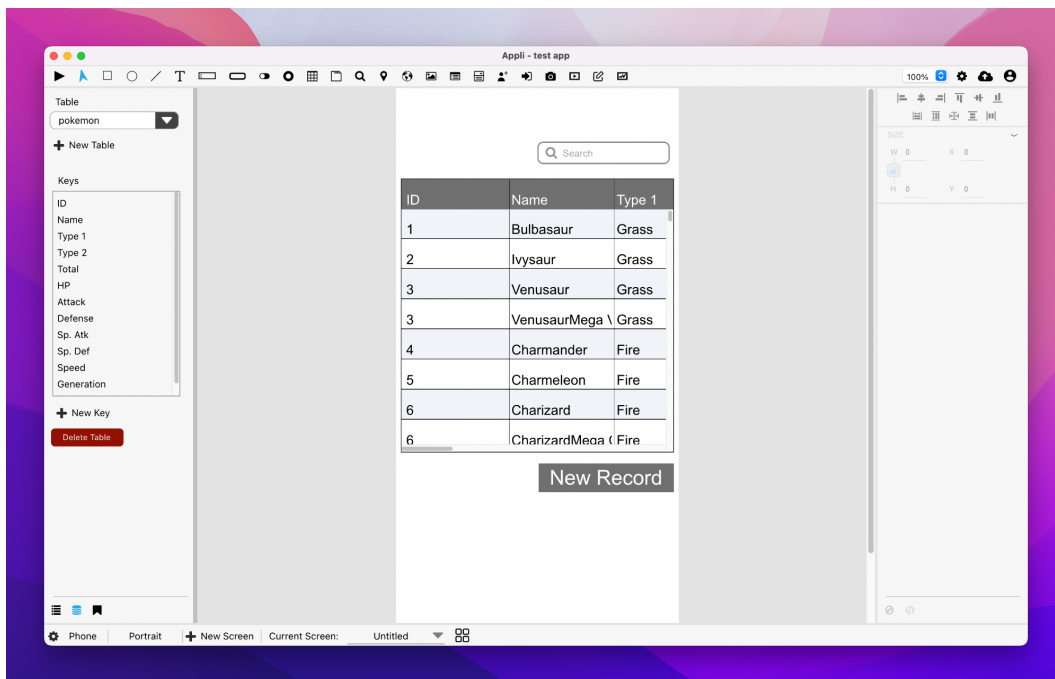
Project browser

A search box is provided in case you need to filter the list of elements. For each element on the list, you can:

- Use the *padlock button* to lock the element and prevent accidental changes to its properties.
- Use the *eye button* to toggle the visibility of the element.
- Use the *brackets button* to change the low-code/no-code flow for the element.

Data Manager

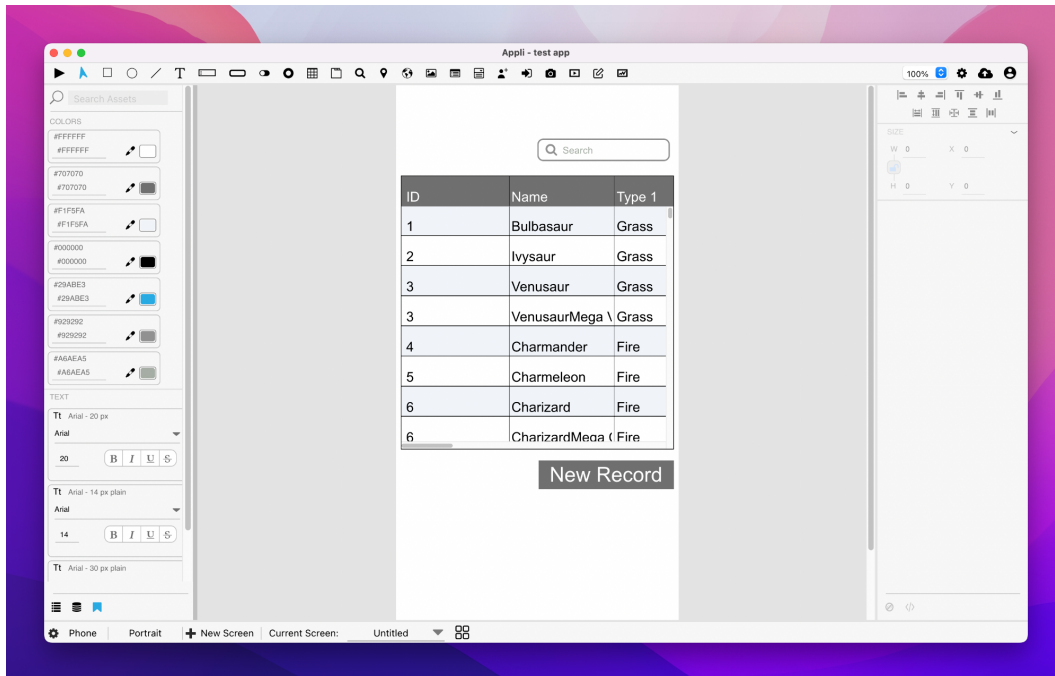
The data manager is used to add and configure data tables for your application.



Data manager

Tables can be local, cloud, or hybrid. The developer can decide which keys are exposed to the application and many other features. Dive deeper by reading the [data management](#) documentation.

Asset Manager

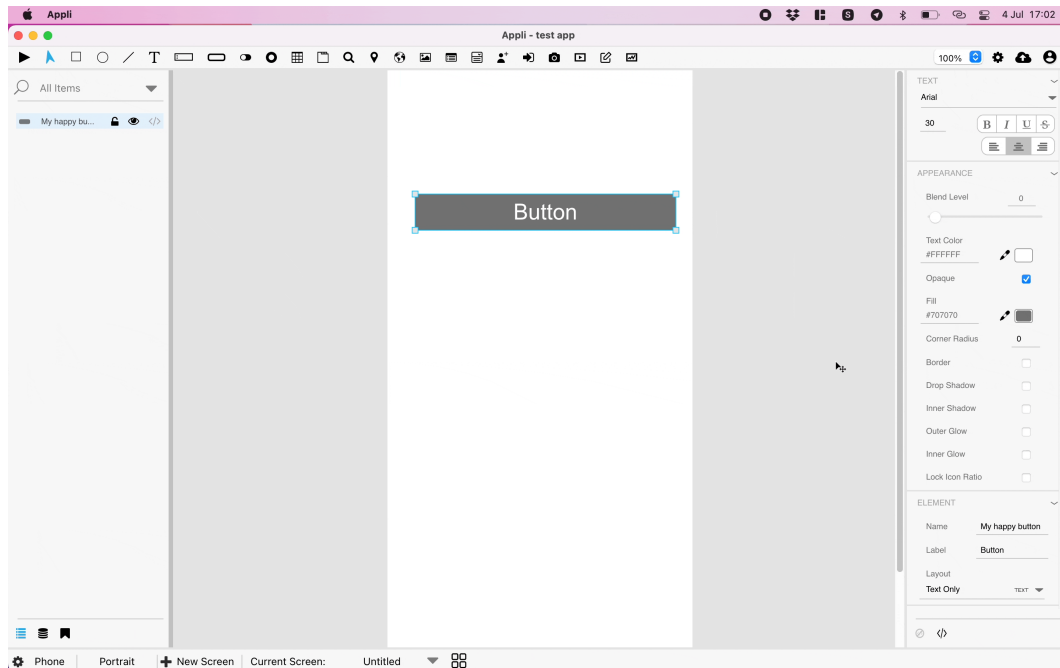


Asset manager

Use the asset manager to define a theme for your application. You can select default colors and text styles that will apply to all elements in the app.

PROPERTY INSPECTOR

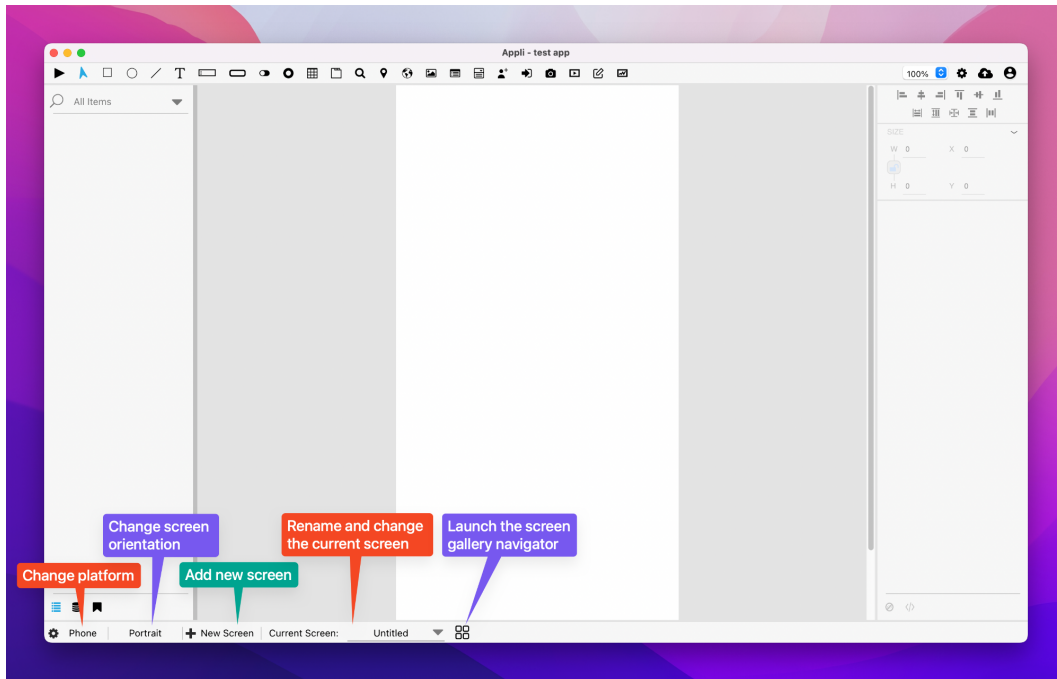
On the right side of the IDE, you'll find the property inspector. Elements are configurable via properties. You can change various characteristics and features by changing values in the property inspector. It is through the property inspector that the developer can change the behavior of an element, something that you can learn more by [diving deeper into the property inspector](#).



Changing properties on a button

FOOTER

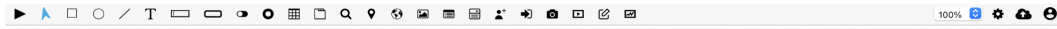
The footer is at the bottom of the interface. In that region are all the features related to managing the various screens and platforms in your project.



Footer features at a glance

Read the [screen management documentation](#) to learn how to create and manage multiple screens.

TOOLS PALETTE



Tools palette

The tools palette can be understood as serving three different broad roles:

- At the right side, we have project management tools.
- Most of the buttons in center of the palette are *elements* used to construct the application.
- The first button in the palette is a play tool used toggle between interacting or designing the app.

PROJECT MANAGEMENT



Project Management

These buttons deal with workflows related to your project and your Appli account.

The *gear button* opens the application-wide settings. Use it to change app name, icon, and some other general settings. Read the [application settings](#) documentation for more details.

The *cloud button* saves your application to the cloud.

The last *user button* is actually a dropdown navigation menu with multiple features. It can be used to *sign out* of Appli; go back to the [project selection screen](#); and view the most recent *patch notes*.

TESTING THE APP

The first button on the toolbar, the one that looks like a *play button* places Appli IDE in *interaction mode*. It disables both the left and right panes and lets the developer interact with the application as if it was running inside the player. To switch back to *design mode*, click it again.

ELEMENTS

Most of the rest of the palette is dedicated to hosting the *elements* used for app designing.



Elements

To draw an element in the playground, first select it on the tools palette and then draw it.

The *pointer tool*, which is the button next to the play button, can be used to select and alter elements already on the playground.

Check out the [elements gallery](#) to learn more about each element.

KEYBOARD SHORTCUTS

Appli has extensive keyboard shortcuts support. Learning them can speed up your development process.

GENERAL SHORTCUTS

Key Combination	Function
V	hide/show thumbnail view.
Shift-O	Alternate orientations.
o	hide/show off screen elements.
p	toggle Pointer/Player mode.
Shift-P	Engages pointer mode when you had previously selected an element.

COPY/PASTE/CUT OPERATIONS

Key Combination Operation

CMD-C	Copy.
CMD-V	Paste.
CMD-X	Cut.

SELECTION MANIPULATION

Key Combination Function

CMD-A	Select all.
Shift-CMD-A	Deselects all.

UNDO/REDO MANAGEMENT

Key Combination Function

CMD-Z	Undo.
Shift-CMD-Z	Redo.

LEFT PANE

Switching between the various tabs on the left pane can be done with simple keyboard shortcuts.

Key Combination	Function
CMD-D	show data manager.
Shift-CMD-Y	show asset manager.
CMD-Y	show project browser.

GUIDELINES

Creating a good design might require one to use multiple combinations of snapping, guidelines, and measuring. These shortcuts will help you handle guideline management with ease.

Key Combination	Function
Tab	Show distance to screen edge.
Space	Do not use guidelines and snapping.
G	Toggle Fixed Guidelines.
~	Prevents elements from snapping to the normal guidelines.

ELEMENTS SELECTION

Learning the element keyboard shortcuts will speed up your development workflow. Quickly change what element type is selected without the need to move the mouse to the toolbar.

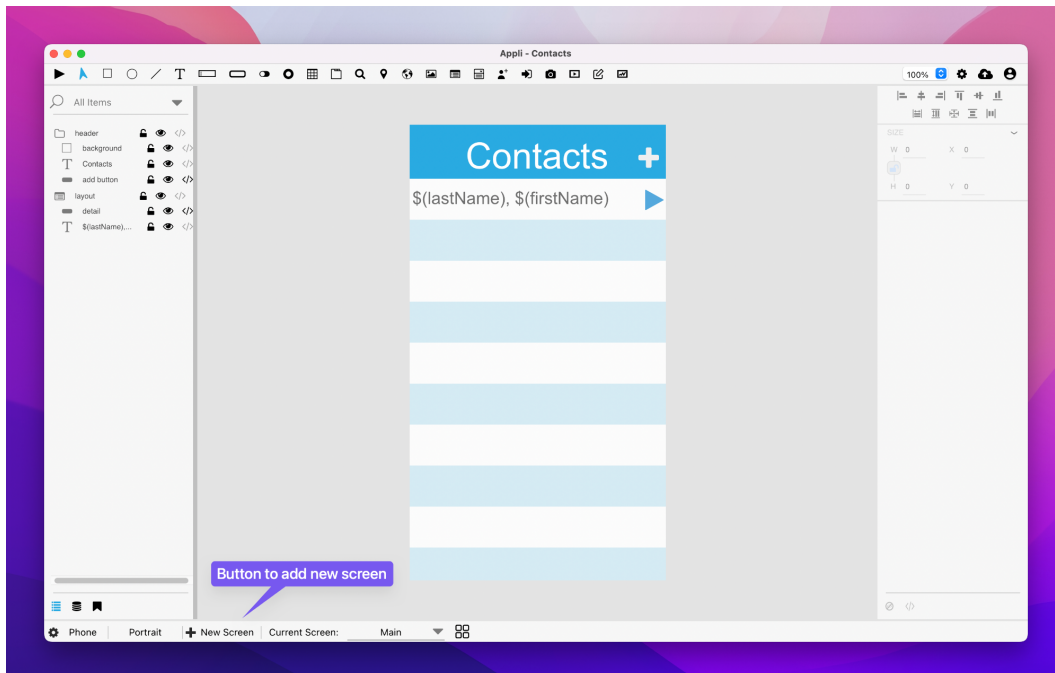
Key Combination	Element
a	create account.
b	button.
c	camera.
d	dropdown menu.
e	ellipse.
f	label/text field.
h	header.
i	image.
k	log in.
l	line.
m	map.
r	rectangle.
s	search.
t	input Field.
u	tab menu.
x	switch button.
Shift-B	browser.
Shift-F	form.
Shift-H	footer.
Shift-L	layout.
Shift-M	media.
Shift-R	radio button.
Shift-T	table.

SCREEN MANAGEMENT

A project can contain multiple apps, one per platform. Each app can contain multiple screens in different orientations. That is a mouthful to say that an Appli project is very flexible. You can build apps for various platforms — desktop, tablets, and smartphones — in a single project, and each app will have their own screens.

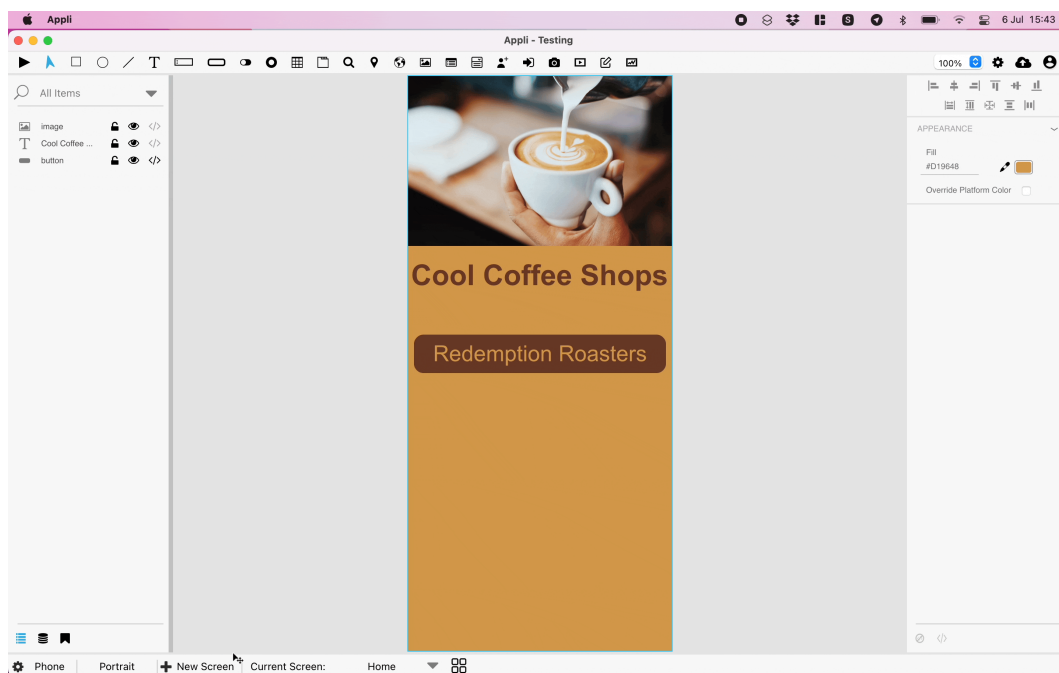
CREATING A NEW SCREEN

When you first create a project and an app, you select an initial platform and orientation for the app. The playground screen will open with that platform and screen selected. As your project grows, you'll may need to add more screens. The controls to do that are at the *footer*.



Button to add a new screen

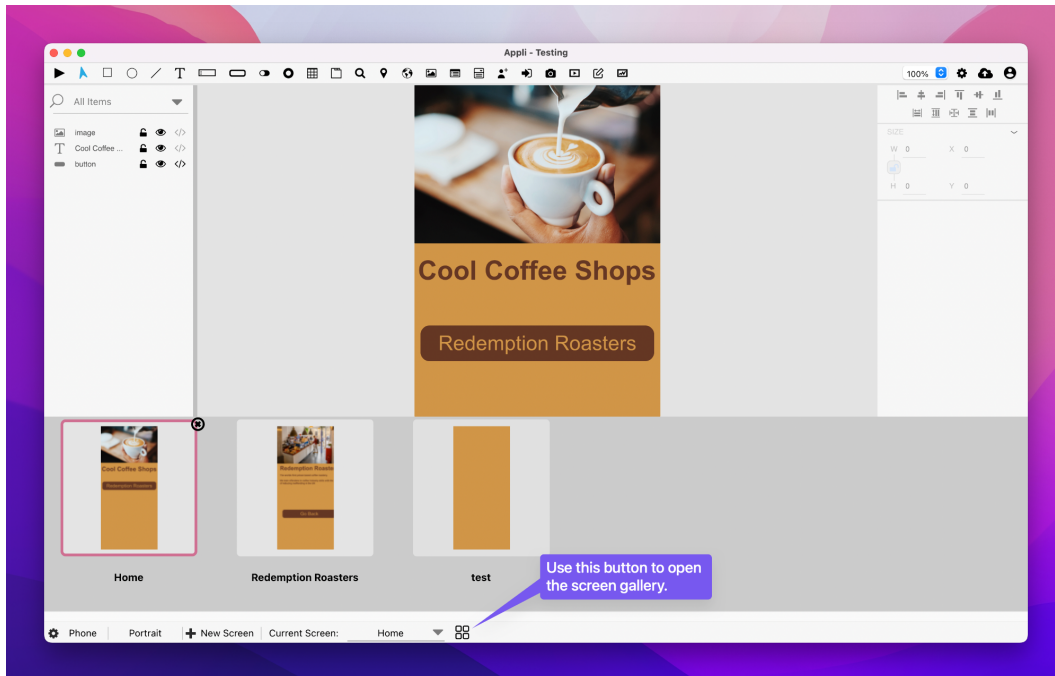
After clicking that button, Appli will ask for the name for this new screen. The playground will automatically switch to the newly created screen as shown in the *current screen display in the footer*. That display is a pop-up menu that allows the developer to switch between the screens in the current platform.



Adding a new screen to a project

SWITCHING BETWEEN SCREENS

Besides the pop-up menu shown above, there is another control that opens a gallery of screens thumbnails that makes it easier to find the screen you want.



Screen gallery

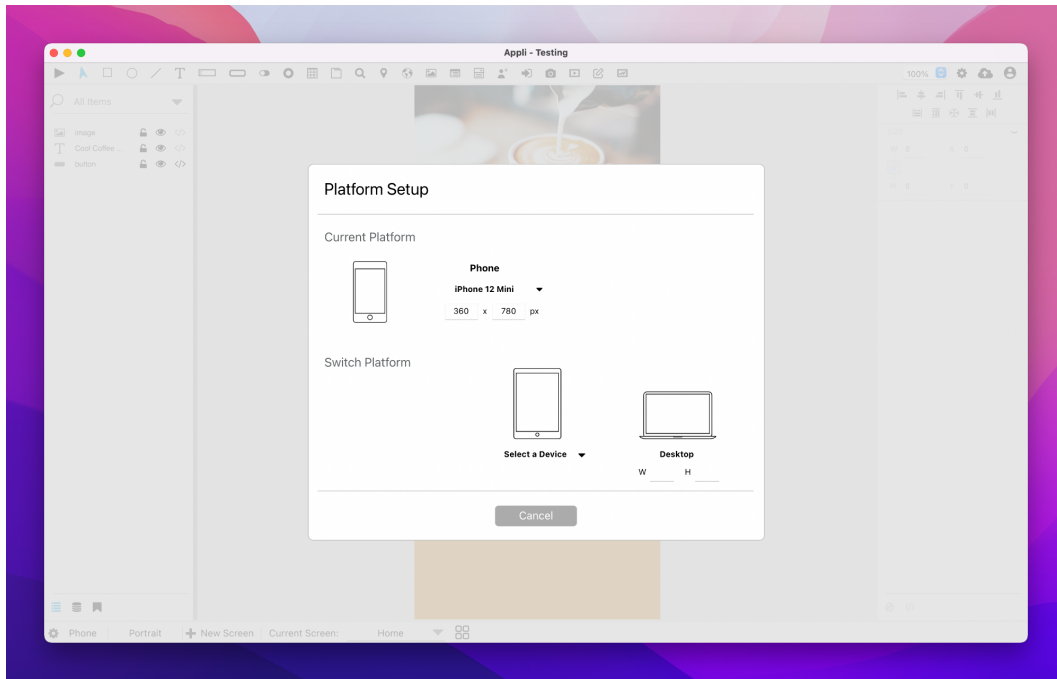
Important: It is in this gallery view that you can delete a screen. Use the little *crossed circle button* in the corner of the current screen thumbnail to delete it.

CHANGING ORIENTATION

Each screen can have independent versions for each orientation. You can switch between them by using the orientation button on the footer. Both orientations are versions of the same screen and the player will select the correct when it comes time to run the app later.

PLATFORM MANAGEMENT

The first button in the *footer* is used for changing the app platform. Clicking it will display the information about the current platform, and also allow the developer to switch to a new platform.



Screen gallery

FINAL REMARKS & WHAT TO READ NEXT

As we have shown, all the controls to manage both screens and platforms are at the *footer*. By now, you should be able to:

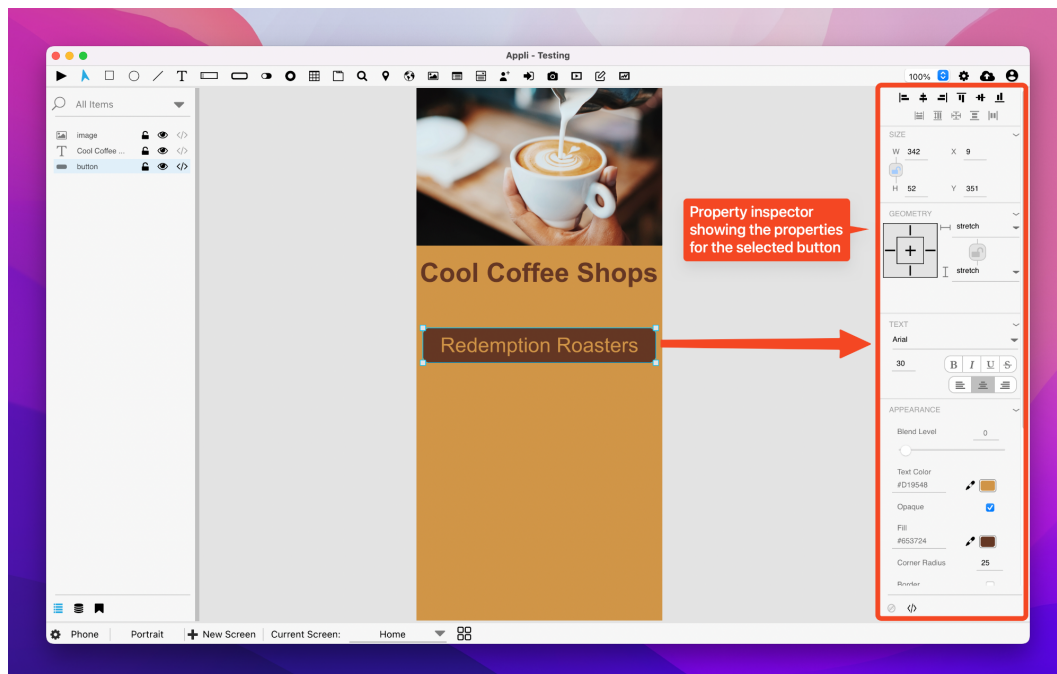
- Create new screens.
- Create versions of a screen for different orientations
- Switch to a new platform.
- Switch between screens.

Even though you can create as many screens as you want, you still need to learn about responsive design to make sure that your screens can adapt to different resolutions. A screen that was designed for an iPhone 13 Pro Max might not be suitable for a smaller Android device. Using responsive design, you can configure elements to adapt to different resolutions and make sure your app always looks perfect.

PROPERTY INSPECTOR

In Appli, one builds their app by creating and customizing elements on a screen. Each element has their own characteristics and behaviors. We refer to these characteristics as *properties*. To change them, the developer uses the *property inspector* in the right-side of the screen.

To view the properties of an element, select it using the *pointer tool*. Each type of element has properties that are suitable for them. For example, a *map element* will have a property to specify geolocation markers. The other elements have no need for such property. You won't see it on them. Some properties such as labels and colors are quite common and you'll find them over and over as you work in your app. All this will become second nature as the developer gets familiar with Appli.



Property Inspector

The interface of the property inspector is divided into sections that always appear in the same order regardless of the selected element's type.

ALIGNMENT TOOLS

Placed at the very top of the property inspector are the alignment tools.

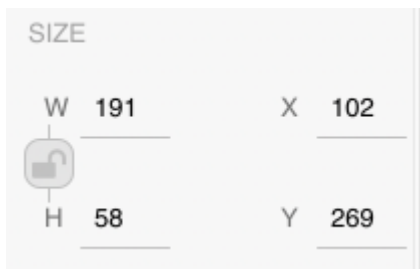


Alignment tools

These tools help the developer align their elements on the screen. Use them when you want to align an element based on the screen borders or other elements.

SIZE PROPERTIES

These are quick ways to change the position and dimensions of an element.



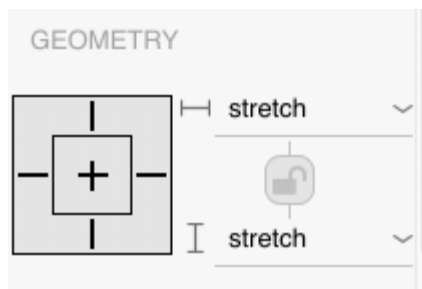
Size properties

There is a handy padlock that enables you to lock the dimensions of an element so that you don't resize them by accident.

GEOMETRY PROPERTIES

There are many variations of screen size and resolution for each platform when we factor in the thousands of devices that exist out in the world. A good example is smartphones. They come in various sizes and aspect ratios, even though most of them still qualify for a *portrait smartphone* category.

The geometry properties let the developer configure how an element should behave when the screen size differs from the one being used to design the application. Elements can grow, shrink, etc. To learn more about how to craft resolution independent screens, head on to the [responsive design](#) documentation.



Geometry properties

TEXT AND APPEARANCE

These represent all the properties that govern the appearance of an element. Colors, graphic effects, shadows, labels, all are a part of this text and appearances section. Be aware that these can vary depending on the kind of element selected. For example, elements without text will not contain a text section.

TEXT

Arial

30

B I U S

APPEARANCE

blendLevel 0

foregroundColor #FFFFFF

showBackground ☒

backgroundColor #707070

roundRadius 0

showBorder ☐

dropShadow ☐

Text and appearance properties

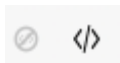
ELEMENT SPECIFIC

The properties that are specific for the type of element selected are displayed in a *element specific* section. To learn more about the properties of each kind of element, check out the [element gallery](#).

ELEMENT	
name	button
label	Button
layout	
Text Only	TEXT ▼

Element specific properties

NO-CODE AND LOW-CODE



no-code and low-code

These are small buttons at the end of the panel. Some properties can't fit well in a sidebar like this, so you can access them using these buttons.

Some elements are very complex, such as the *camera element*, to access things such as the camera output, use the first button, which is the *no-code* button.

The *low-code* button is used to configure flows. These are the behaviors attached to an element. You can learn more about them by reading the documentation on [flows and behaviors](#).

RESPONSIVE DESIGN

Computing devices come in various form-factors — from desktop computers to small smartphones — being to adapt and serve your customer is a crucial feature for any application.

No one wants to use an application that was designed for a smartphone and presents itself with the same smartphone-focused user interface on the desktop or tablet. The reverse is also true, an application designed for a desktop is usually a poor citizen on a small smartphone.

Responsive design is the technique of creating designs that adapt to run well in device.

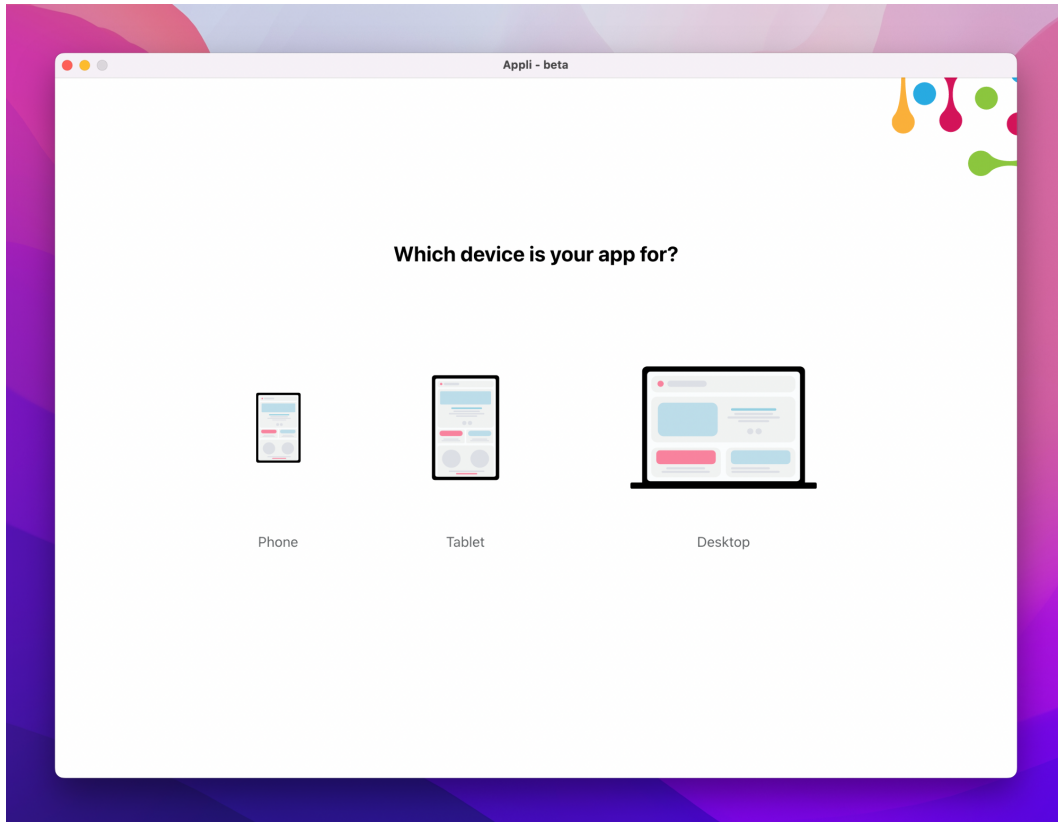
To achieve that, Appli uses three broad platform categories: desktops, tablets, and mobiles. Your design for each of those categories is independent from the others.

Even with per-platform designs, one might worry that many devices in a single category are still too different from one another. While that is definitely true, the properties in the *geometry section* of an element allow the developer to set how the element adapts to different screen sizes. Mobile phones might come in different resolutions and proportions, but they're all just phones: one screen that you interact using touches. Desktops and laptops have different screen sizes, and you mostly interact with them using keyboard and mouse. Tablets sit in between them both and deserve a special design that leverages the best of both worlds.

In this chapter, we're going to learn how to create independent designs for each platform and how to make effective usage of the geometry-related properties to make sure our application shines in all devices.

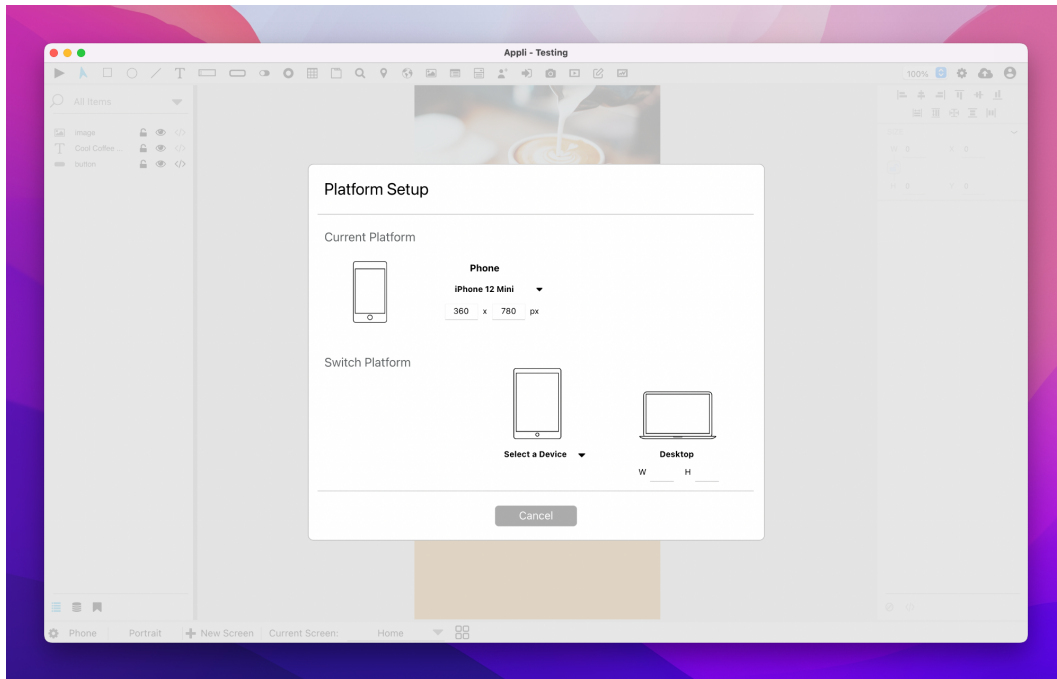
CREATING DESIGNS FOR EACH PLATFORM

When you are creating a new application, you need to choose the device category you want to work with. Do that by clicking on the desired device on the platform selection screen_:



Platform selection screen showing mobile, tablet, and desktop device categories.

Once you complete your application for that category, you can add additional applications with different platforms to your project. If you need to change the currently selected platform, use the button in the footer open the *Platform Setup* dialog.

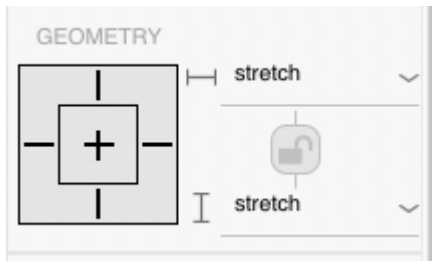


Platform setup dialog.

Be aware that tablets and mobile applications can adapt to orientation changes. Use the orientation toggle at the footer next to the platform indicator to switch between *portrait* and *landscape*. Each orientation design is independent from the other.

UNDERSTANDING THE GEOMETRY PROPERTIES

Besides creating new designs for each combination of platform and orientation, one should use the properties inside the *geometry section* to configure how an element should behave when the screen does not match the resolution that was used for the original design of the application. A typical case is when you design a mobile app for an iPhone and open it on a Samsung Galaxy Note. They are both phones, but they have different proportions and screen resolutions.



Geometry section as seen in the property inspector.

There are two settings: `responsive-x` and `responsive-y`. They are used to configure how the element behaves when there are changes to the width and the height of the screen.

The possible values for `responsive-x` are:

- left: move element to maintain its distance to the left side of the screen.
- right: move element to maintain its distance to the right side of the screen.
- left and right: same as having both left and right set.
- stretch: grow or shrink the element as needed.
- center: don't resize the element but center it considering the changes to the screen width.

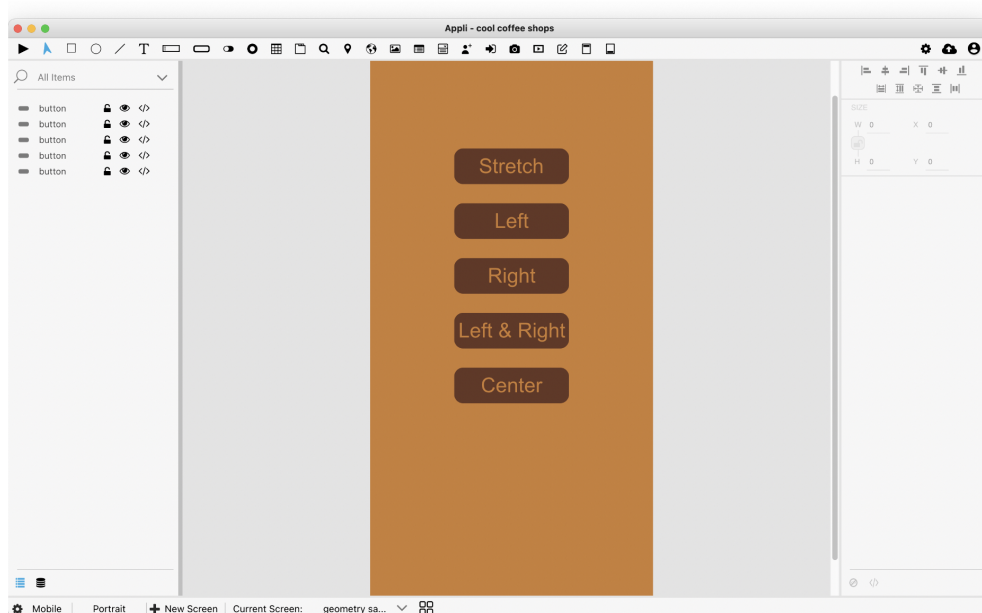
The values for `responsive-y` are:

- top: move element to maintain its original distance from the top of the screen.
- bottom: move element and maintain its original distance from the bottom of the screen.
- top and bottom: the same as setting both values above.
- stretch: grow or shrink the element as needed.
- center: don't resize the element but center it vertically considering the changes to the screen height.

The values can be changed using the dropdown menus on the section. The visual guide next to them show the current selection and can also be used to change the value for those properties, just click on the corresponding line.

Stretch is the default value used for all elements. It means that the object will grow or shrink to cope with the changes in width and height while preserving its aspect ratio.

To make it easier to visualise how these values change how an element behaves, consider the screen below. It was designed for the iPhone XR inside Appli.



A screen designed for the iPhone XR.

Notice how all button have the same size. The label of each button represents the value of the `responsive-x` property. Once I open it in Appli player using a different phone, in this case an iPhone 12 Mini, look how each button changes according to their `responsive-x` property.



13:44 🌙



Stretch

Left

Right

Left & Right

Center



That screen as seen in an iPhone 12 Mini.

As you can see, the default behaviour which is using `stretch` is the best option. With Appli, you get responsive design by default and you only need to change something when you have niche needs.

Oh, and in case you're wondering how their vertical position worked out of the box, that is because they're all have `responsive-y` set to `stretch`.

TIPS AND TRICKS

- When designing a user interface consider what kind of input method your user will be using. A finger is less precise than a mouse, so larger buttons make for easier targets in mobile platforms while smaller buttons allow you to make the best use of screen real estate on a desktop platform.
- When in doubt, set it to `stretch`.
- All tablet users have access can use touch input. Some of them might have keyboards or even a mouse connected to their tablet. Make sure your tablet design can work with just touch, but make sure that users leveraging keyboards and mouse will benefit from them.

LOW-CODE & ACTION SCRIPTS

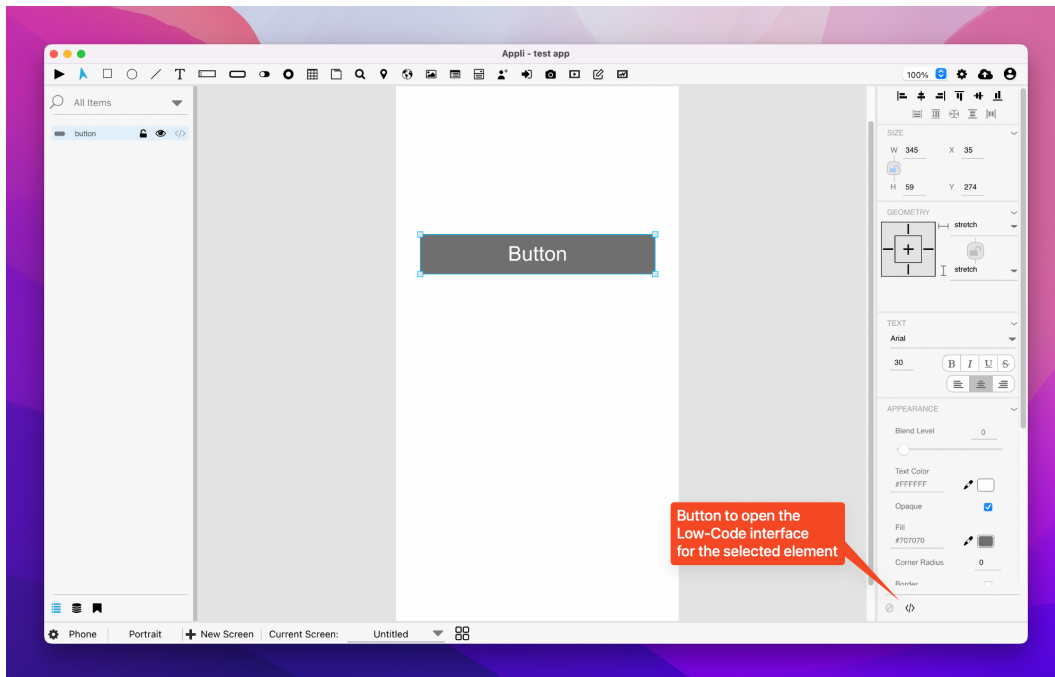
So far you have learned how to design your application by configuring platforms, creating screens, and laying out and customizing elements. That can get you very far, but to fully leverage the power of Appli, you need to understand how low-code works.

Most development workflows involves the developer typing our commands to craft their source-code in a way that is understood by that programming language. It is an error-prone process in which a typo can render a whole project impossible to compile, or worse it might render the final product buggy in a way that the consequences only surfaces later in its lifetime. The process of learning a new programming language requires a lot of effort and focus, it is not that different than learning any new human language. It is a rewarding process, but given how it works, it has a huge gap between first being exposed to the programming language and actually being an effective developer using that language.

Appli makes this gap smaller by being a next generation development environment. It doesn't force the developer to write their own source-code. Appli features an interactive development paradigm using a graphical interface; in essence Appli writes the source-code for you. This means that an Appli developer doesn't need to memorize all commands using an exotic language before they can be effective. Our interactive low-code system helps you learn how to be a developer, and enables you to craft complex programs in a fraction of the time that it would take to do the same task in a previous generation programming language.

THE LOW-CODE INTERFACE

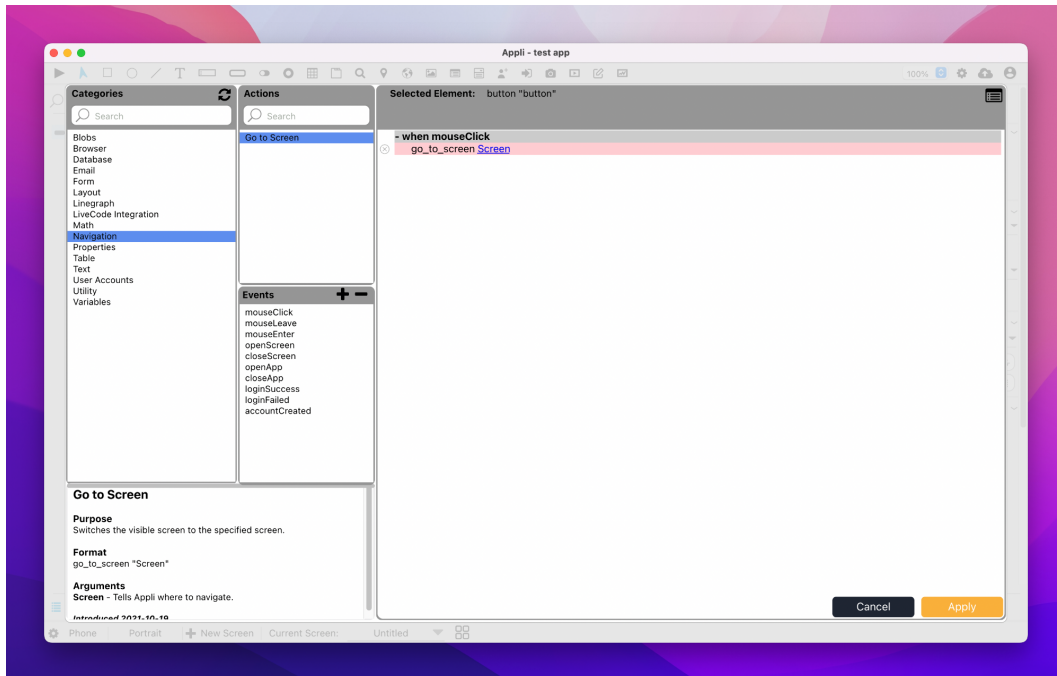
Low-Code is the method used to add behaviours to an element. The property inspector allows one to configure an element. At the bottom of the property inspector, you'll see the Low-Code button which opens the Low-Code interface for the selected element.



Button to open the Low-Code interface

Let's quickly define what we mean by *behaviour*. We're using it as a loose term to mean whatever happens when the user interacts with the element.

The Low-Code interface is divided into distinct sections that are easy to identify: categories, actions, events, action script, and help.



The Low-Code Interface

By selecting and configuring *actions* and *events*, you create *action scripts* which dictate what happens when a user interacts with the element.

Events are the trigger for *actions*. When an *event* occurs, it activates the *action* you selected. The most common event is *mouseClick* which is when the user clicks on an element using the mouse. If you don't select an *event* from the list, Appli will default to asking if you want to use *mouseClick*.

Actions are grouped using *categories*. Select a *category* to view all actions available in that group. Select an *event* and click on an *action* to add it to the *action script* on the right side of the interface. The selected *action* might contain some underlined text. That means that part of the *action* is an *argument*. To complete the action, you should click that *argument* and configure it.

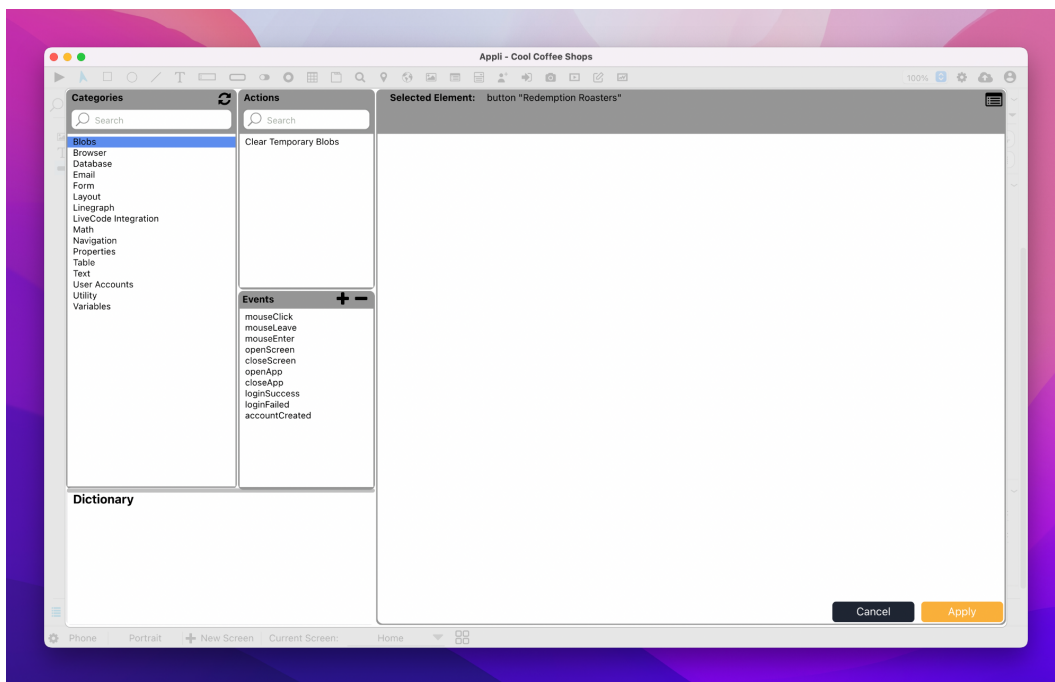
While the action is not properly configured, the *action script* will display with a red tint in the background. Once you set all the

arguments, that tint will change to green. This is a quick way to detect mistakes on your script.

On the bottom-left you'll see the help section for the selected *action* explaining all about it and what arguments it uses. Checking that section is a wonderful way to learn all the possible actions you can use.

EXAMPLE: NAVIGATING BETWEEN SCREENS WHEN A BUTTON IS CLICKED.

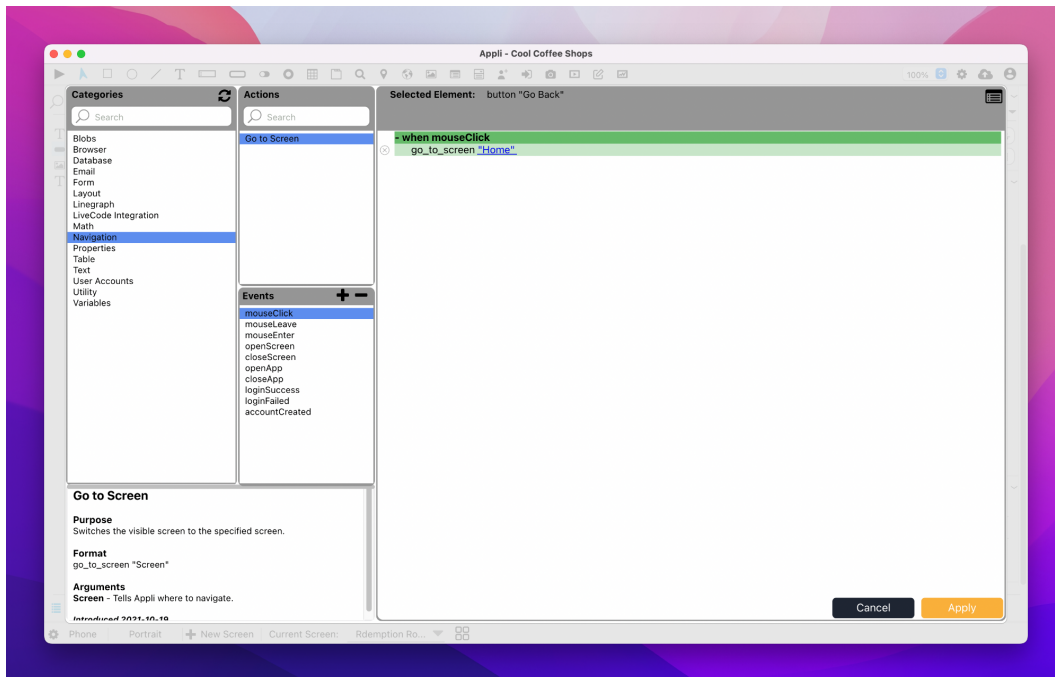
When you open the Low-Code interface for a button , you'll see an empty *action script*.



Low-Code interface showing empty action script.

Our objective is to create an *action script* that causes triggers the navigation to another screen when the button is clicked. To do that

we can go to the *Navigation* category and select the *Go to Screen* action.



Low-Code interface after selecting the Go to Screen action.

Appli wrote the *action script* for you:

```
when mouseClicked  
  
    go_to_screen Screen
```

The last thing you need to do is click on the *Screen* argument to select the destination screen for the *Go to Screen* action. Notice that before selecting the value for the *Screen* argument, the whole script background was tinted red. That is how Appli flags that there are mandatory steps needed before that script is ready. After selecting a screen, the background turns to green.

The *action script* starts with `mouseClick`, that tells you that this behaviour will happen when the user clicks the element, more precisely when they release the mouse button or lift their fingers from the touchpad or screen after clicking. That script can be understood as *go to this specific screen when the user clicks the*

element, that's not very different from the actual text of the script. Appli *action scripts* are easy to read and understand.

Action scripts are not restricted to a single action per script, you can select multiple actions to create complex behaviours. They'll happen in the order displayed on the *action script*. If you got something wrong, you can simply click the small x button next to the *action* in the script to remove it.

NEXT STEPS

Creating *action scripts* will become easier after you've read the next chapter which is a tutorial where we create an application from scratch. You'll create multiple *action scripts* for the various elements and that workflow will become a second nature to you in no time.

DATA MANAGEMENT

Appli offers robust data management features thanks to the many years of experience that the team spent developing LiveCloud. Appli itself is backed by the LiveCloud database. When you save your application to the Cloud, what happens behind the scene is that your app is stored in LiveCloud where it can be retrieved by the Appli Player.

The same powerful database system that powers Appli is available to you as a developer. It is easy to use and matches Appli development workflow in a way that other database systems simply can't.

LOCAL, CLOUD, AND HYBRID

There are three ways of storing data. Your choice is dictated by your applications requirements, but we believe that Hybrid storage is an ideal solution for most cases.

Storage type	Description
Local	The data is saved only to the user's device. Any data sharing or replication needs to be coded by the developer.
Cloud	The data is saved and retrieved from the Appli Cloud.
Hybrid	The data is cached locally and synchronized with the Appli Cloud.

Hybrid works best for *Table Elements* because:

- It is fast because most of the data manipulation is happening locally before being synchronized over the cloud.
- Works offline with eventual synchronization once the device is back online

- The robust safe online storage makes it a lot easier to develop networked applications.

Both *Form Elements* and *Layout Elements* only offer choices to store data locally or on the Cloud. Hybrid storage doesn't make sense for those elements.

TABLES

Tables are what represents a collection of data in Appli. You can think of them in the same mental model you use for spreadsheets or a stack of forms. A collection of structured data that you can manipulate and query to find the information you need.

Your app can have as many tables as you need, and you can choose to store them locally, on the cloud, or use hybrid storage as described above.

Each table has a set of keys. Much like a form would have fields. A good example is a contacts table, it could look like this:

Table Setup

LocalHybridCloud

Table Name

contacts

Keys +

id

first name

last name

email

☒ Show All

☒ Show in table

☒ Show in table

☒ Show in table

☒ Show in table

—

—

—

—

Selected Record IDs Variable

▼

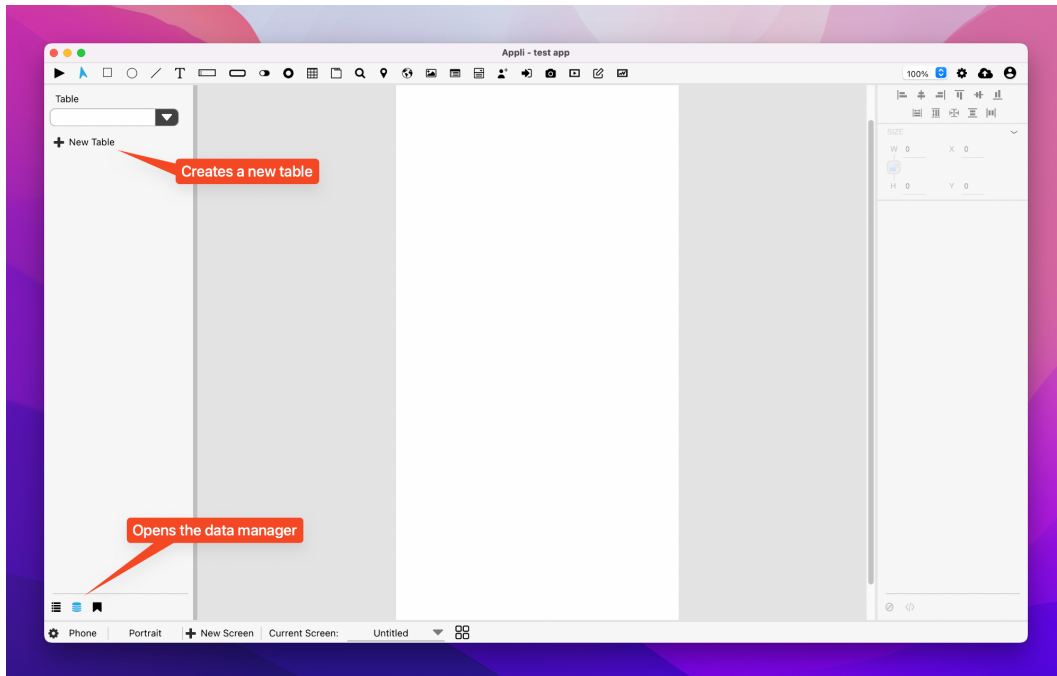
Not Now

Create Table

Sample contacts table

CREATING A NEW TABLE FROM SCRATCH

The *Data Manager* button is a series of stacked discs on the bottom of the left-sidebar. Clicking it opens the *Data Manager*. Create a new table by clicking the *New Table* button.



Opening the Data Manager and creating a new able

Once you have the *Table Setup* dialog open. You can fill in your table name, chose where it is going to be stored, and add keys.

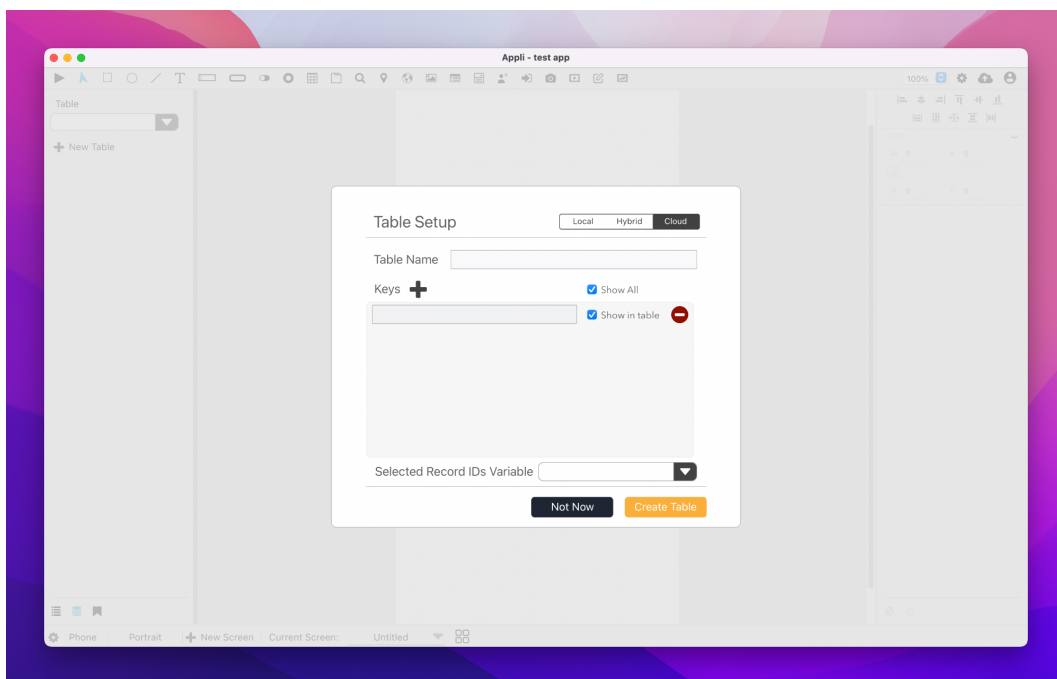


Table Setup

USING CSV FILES TO BOOTSTRAP A DATABASE

A good way to bootstrap a table from existing data is to export that data using the CSV format and dragging and dropping it into Appli.

Appli will process the CSV file and open the *Table Setup* dialog pre-filled with the data from that file.

BINDING DATA TO ELEMENTS

Once you have a table, you can use no-code to bind table data to elements. An obvious element to use is the table element. Using no-code you can select which table to display and which columns to use.

DISPLAYING RECORDS USING THE LAYOUT ELEMENT

A *Table* can be connected to a *Layout Element* Using *no-code*. Once connected, the elements placed inside the layout can be bound to data from the table.

The layout can be configured in a way that it is tied to a specific record in the table or display multiple records as a list. Use the *multiple rows* property to change between these two modes. Once that property is set to the desired value, the interface for *no-code* setup will change to reflect that option.

Data Context

layout local cloud

Connected Table: People ▼

Sort By: lastName ▼

Asc ▼ text ▼

Elements:

text $\$(lastName)$, $\$(firstName)$ ☐ Link to Key ☒ Use Templated Text

$\$(lastName)$, $\$(firstName)$ Edit

Done

Example of Layout no-code table setup with multiple rows.

Elements inside the layout can be linked direct to keys from the table or use *template text* to interpolate data from the table into a formatted text.

For more information, check the Data Management tutorial for a hands-on guide on building database-aware apps.

MANIPULATING DATA

Low-code has a category just for database manipulation. Everything is fully documented with dictionary entries for each action.

USING FORMS TO EDIT AND CREATE RECORDS

Similarly to *Layout Elements*, *Form Elements* can contain other elements inside it. The elements inside a form share a database context, they can be connected to a table via *no-code* configuration and can be further linked to a *Record ID* set using a combination of *low-code* actions:

- `set_variable_from_context`: to store the record id into a variable.
- `set_the_property_from_variable`: to set the `dataRecordID` property of the form to the value contained in the variable set with `set_variable_from_context`.

Elements inside the form will have access to the data from that record. Any change to them using bound fields or *low-code* can be saved back to the same record.

A form that is connected to a table but doesn't have a `dataRecordID` set can be used to create new records in the connected table via:

- `submit_form_to_db` which picks data from the form and save to the database.

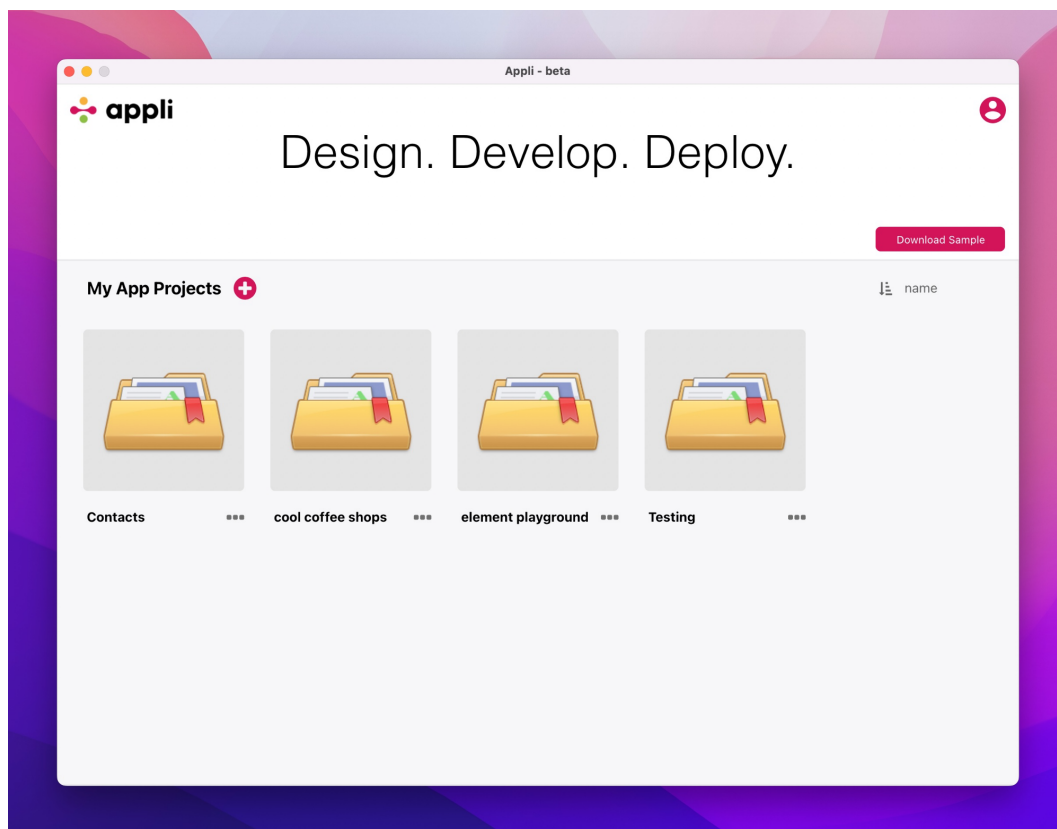
For more information, check the Data Management tutorial for a hands-on guide on building database-aware apps.

TUTORIAL: COOL COFFEE SHOPS

In this tutorial we're going to create a simple application to highlight my favorite coffee shops in London. You are going to go from zero to have a running app in your mobile device in less than one hour. So, fasten your seat belt because working with Appli is fast.

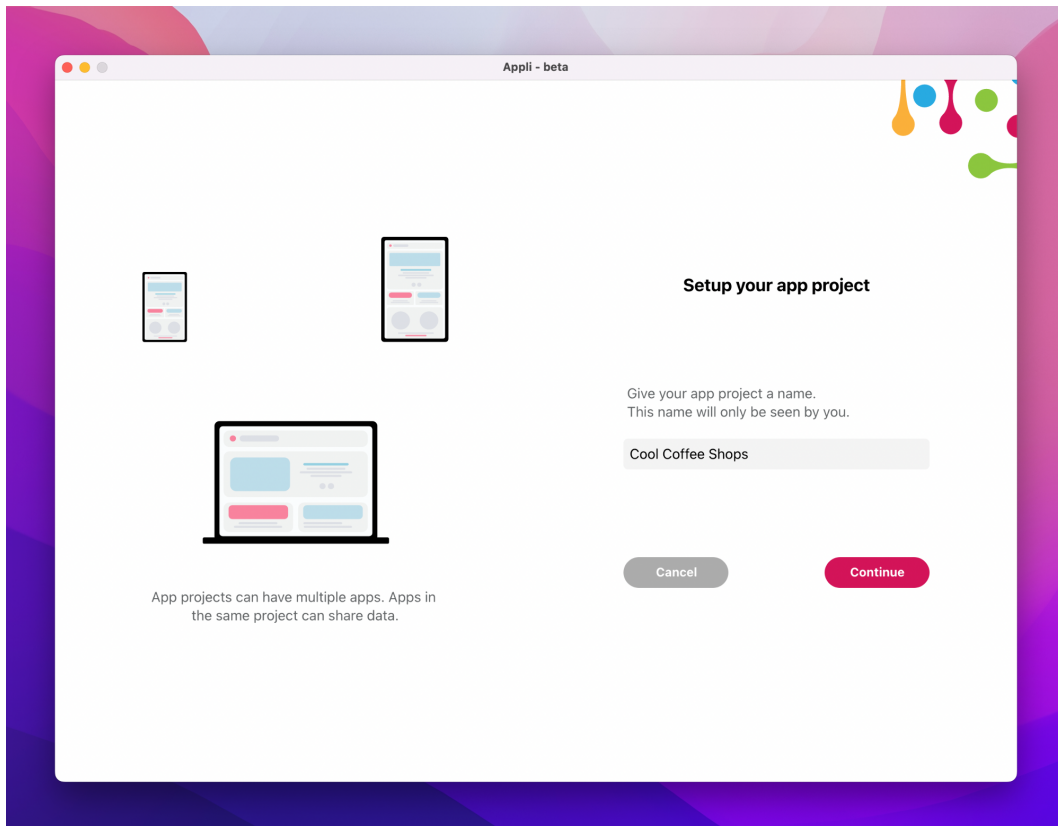
CREATING THE PROJECT

After logging in to our Appli account, we're presented with the project selection screen:



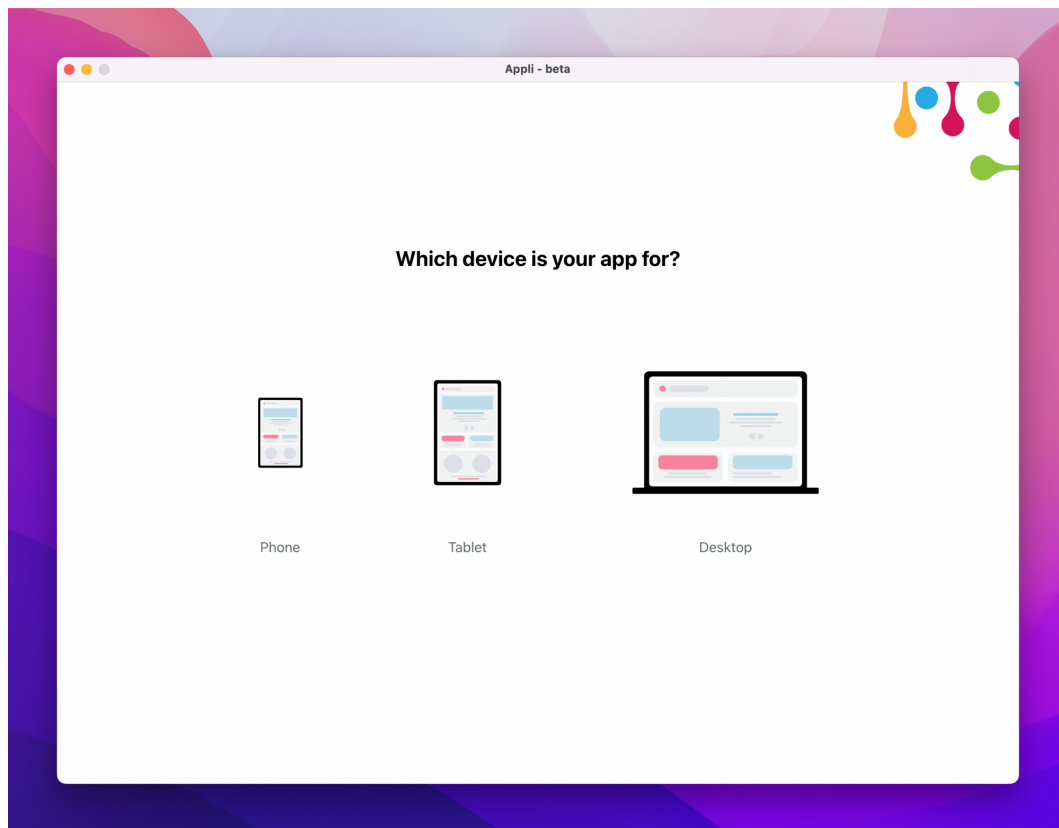
Project selection screen

Click on the *plus* button to create a new project named: Cool Coffee Shops.



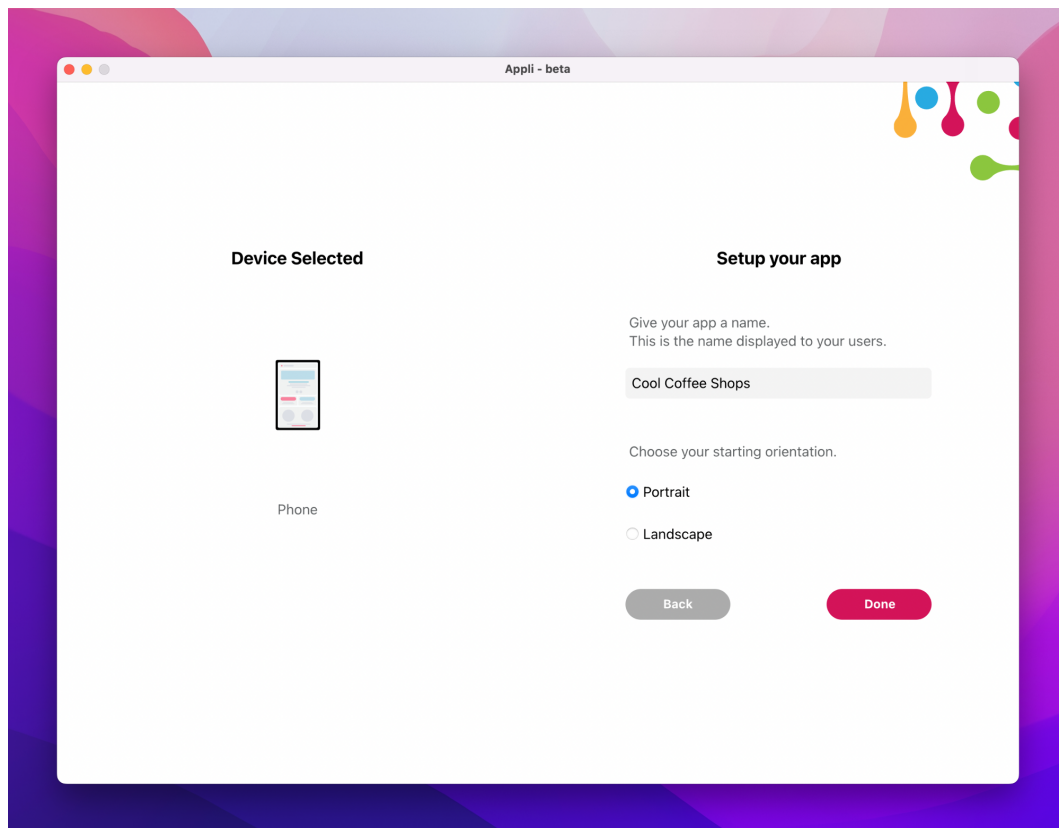
Project creation

Let's create a smartphone app first. Click on the phone button to select that platform.



Platform selection

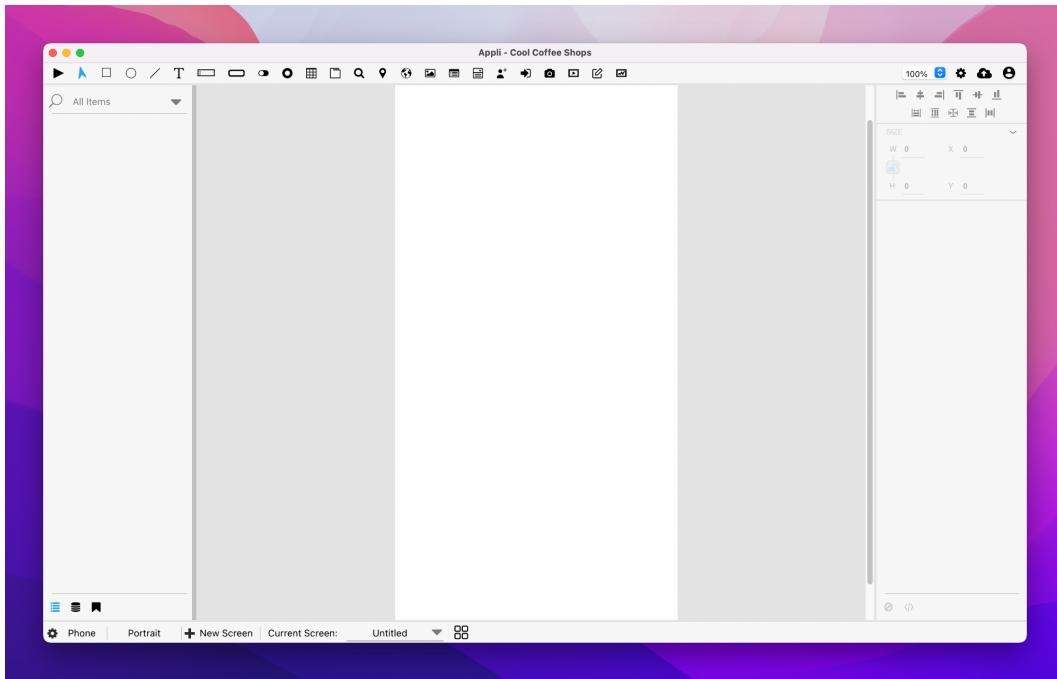
And then fill in the name for the app and select the *portrait orientation*.



Application settings

DESIGNING THE INTERFACE

After clicking save, your new project is automatically saved to the Appli cloud and opened in the playground screen:



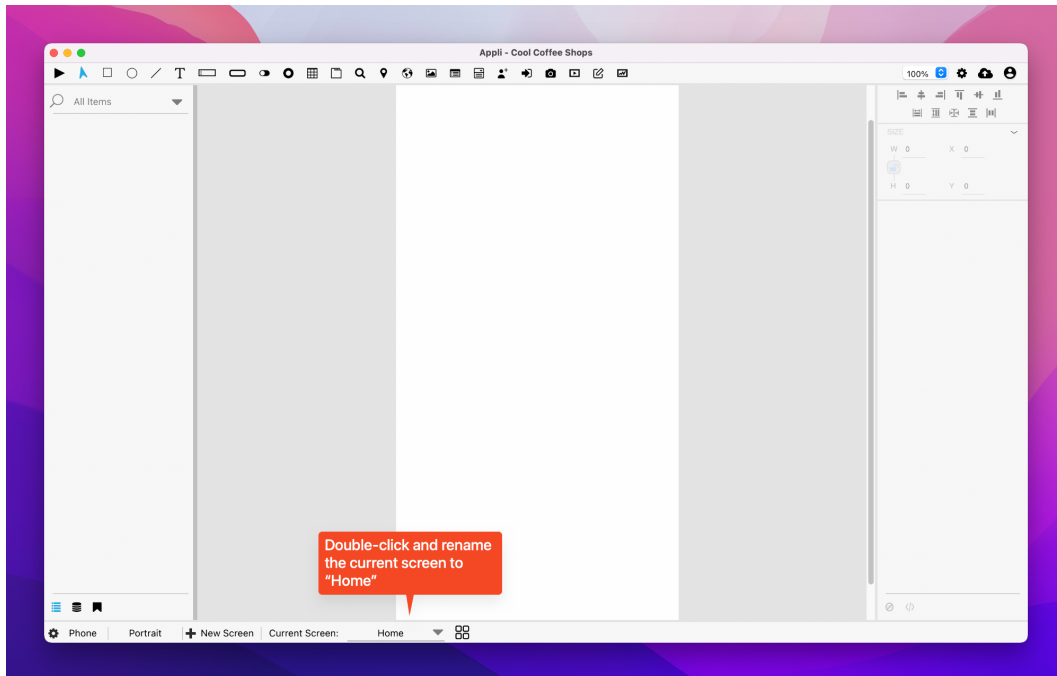
Empty playground

It may look a bit empty right now, but it is actually full of possibilities.

Creating the home screen

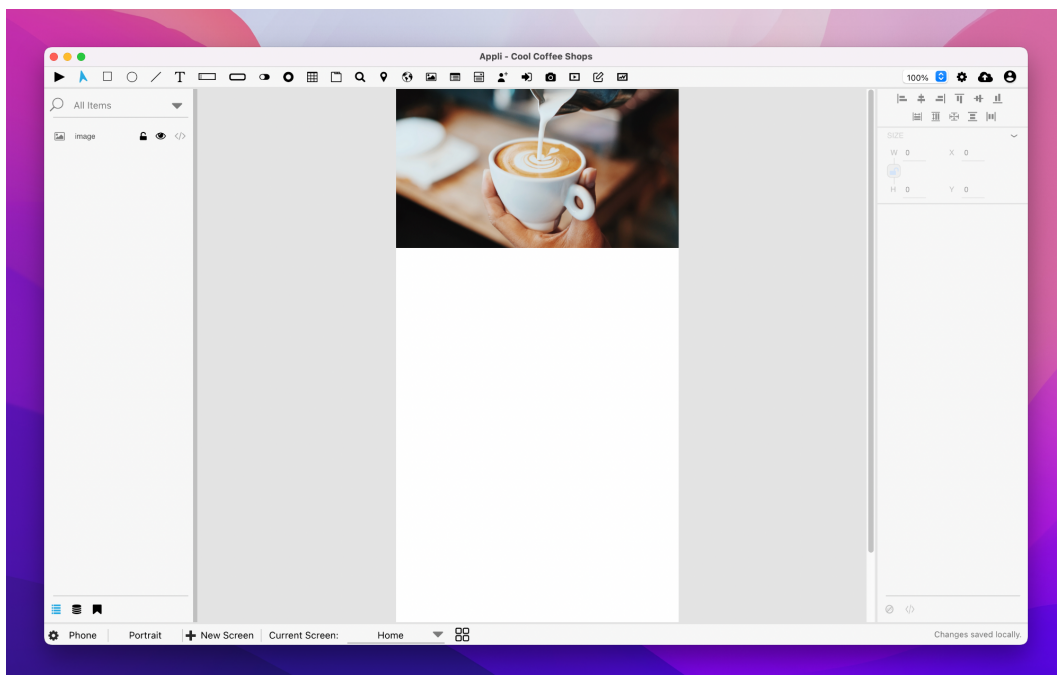
Let's begin by renaming our current screen using the controls in the footer.

Double-click the *"Untitled"* name in the footer to make it editable.
Type in *Home*:



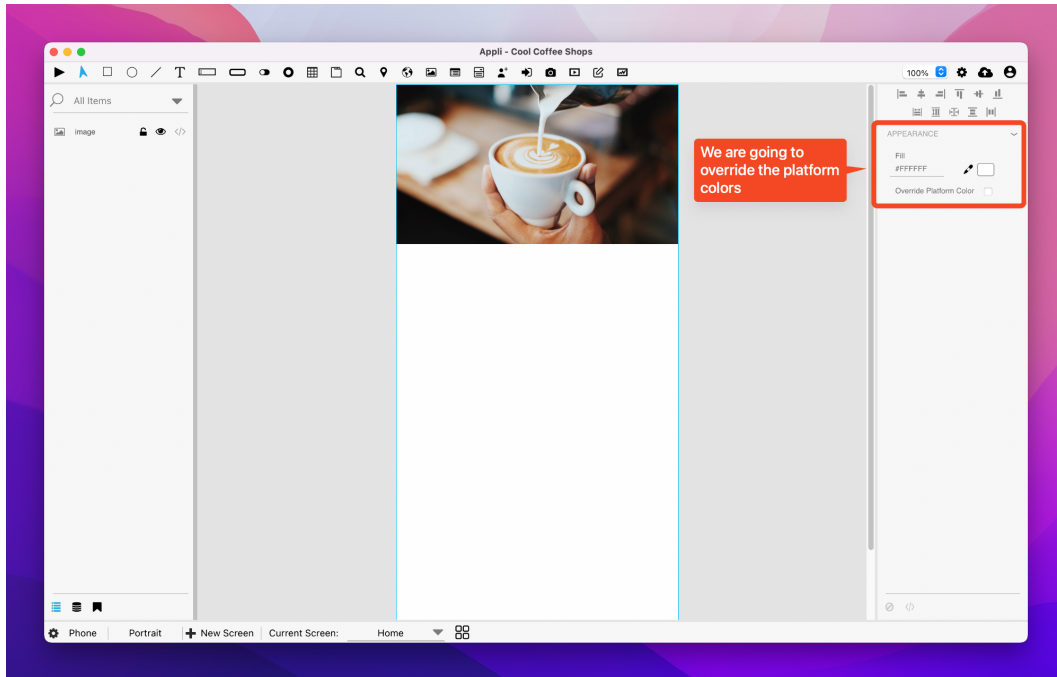
Screen rename: Home

Let's add an image to our home screen to make it more fun. You can download this [photo](#) to your computer and then just drag and drop it into Appli. Resize the image in Appli using the corners to make it fit onto the top of our app:



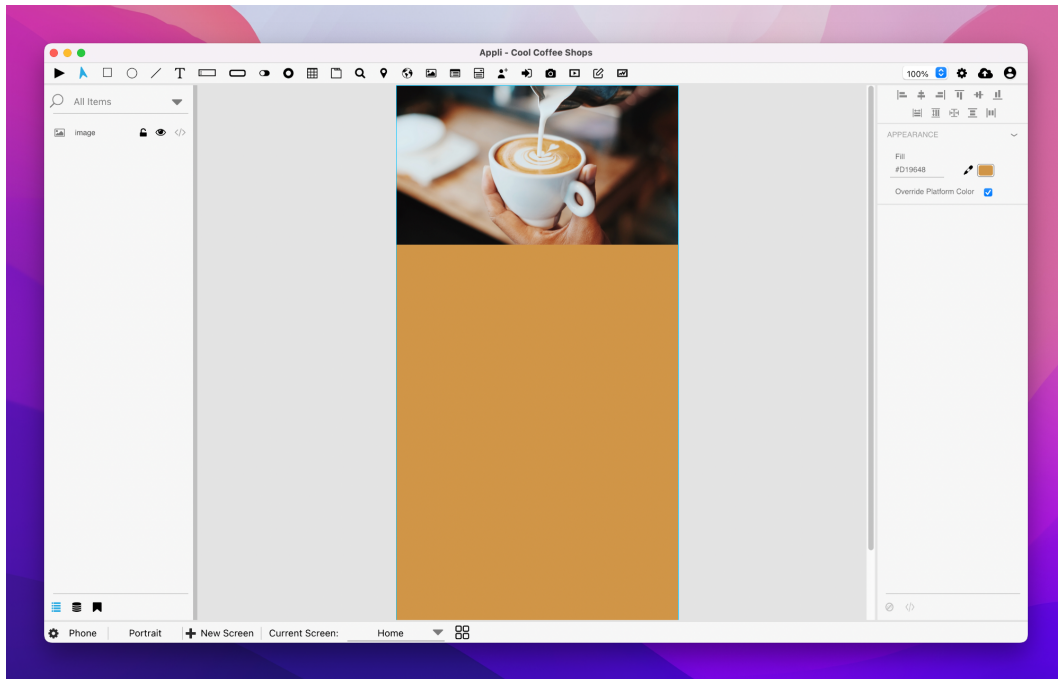
Header photo

Next, let's change the background color of the app to be more like coffee. Click anywhere on the white background of the playground, and let's change the `platformBackgroundColor` property:



Change background color

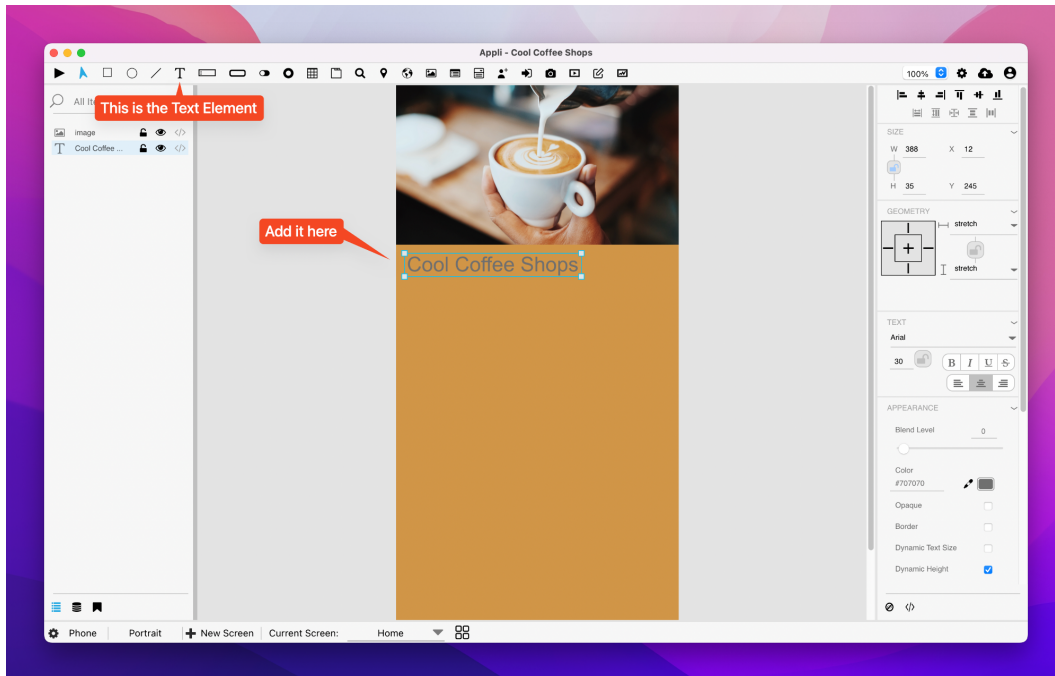
Using the color picker tool (the little eye dropper next to the *hexcoded color*) pick a nice shade of brown for the background.



With new background color

Next, we're adding a little header text to ensure everyone understands what our demo app is all about.

Add a *text element* by selecting it from the tools palette, and drawing it just below our header photo. When you add a text element to a screen, you have the opportunity to change its text. Change it to *Cool Coffee Shops*.

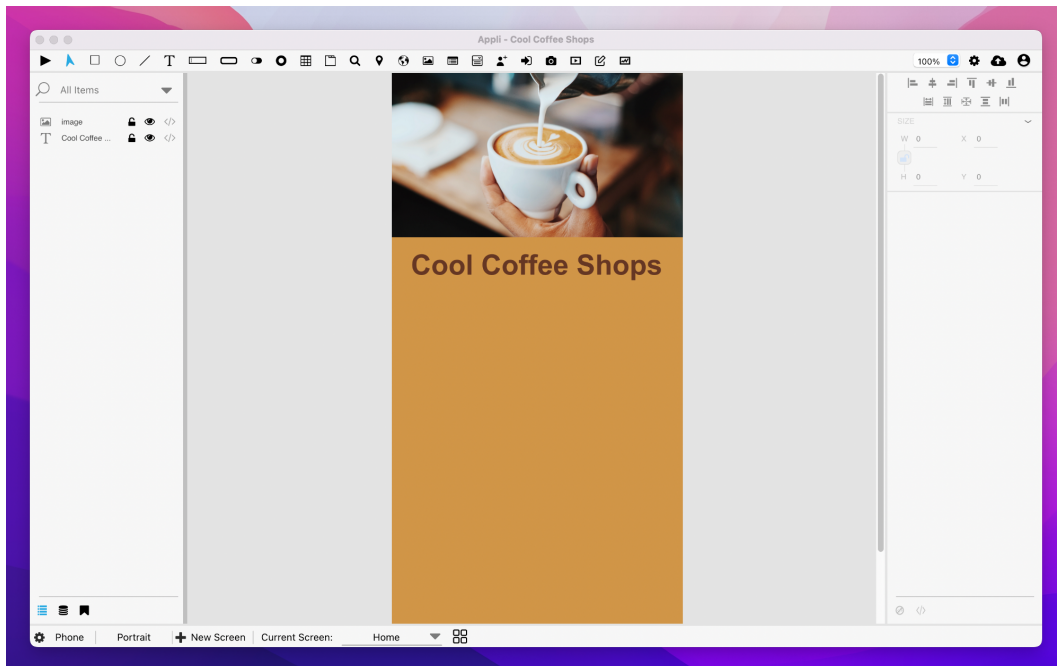


Change text color

We can dress that text element up using the property inspector. In the *Text & Appearance* sections, change the font to bold, the font size to 40, and center the text.

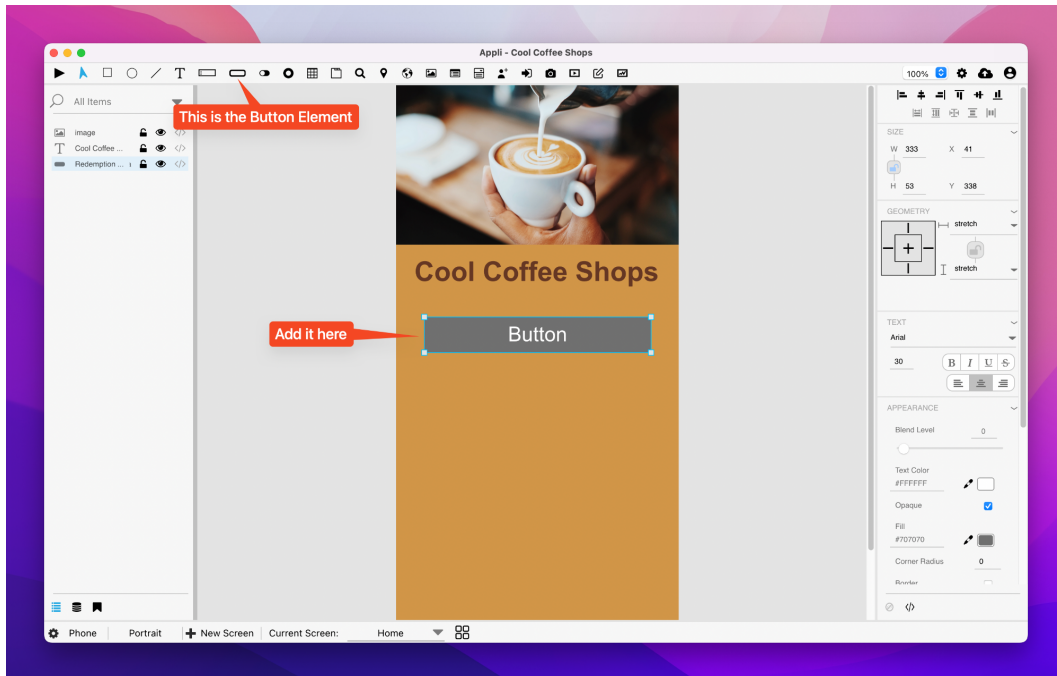
Be aware that to change the text to bold, you need to double-click the text element to make it editable and select the text you want to change.

Scrolling down on the property inspector, you'll find the *textColor* property. It also has an eye dropper tool next to it. Use it to pick some darker tone in the photo.



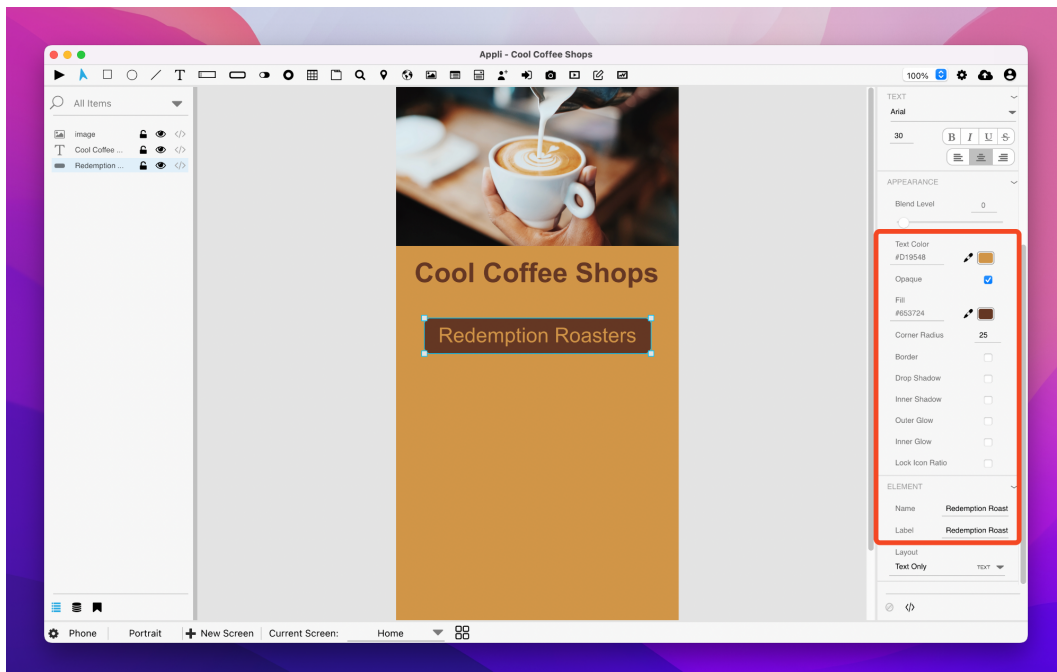
With new text color

Add a button to the screen. That button is what the user will use to navigate to a specific coffee shop. We'll make the first one, then duplicate it to make the second one. You can name that button whatever you want, but I named it *Redemption Roasters* because that is the name of the coffee shop it will go to.



Change button style

Much like our header text, the default appearance of the button doesn't match our application design. Let's change it using the property inspector on the right-side of the screen.



With new button style

The properties we changed:

- `Corner Radius: 25.` (be aware that)
- `Fill Color: #653724` (we actually used the eye dropper tool to pick the text header color).
- `Text Color: #D19548` (Used the eye dropper to pick the screen background color).

For the purposes of this demo application, we're going to list just two shops. Change the label of the button to *Redemption Roasters*.

Now, we're ready to create a our first *specific shop* screen.

Redemption Roasters Screen

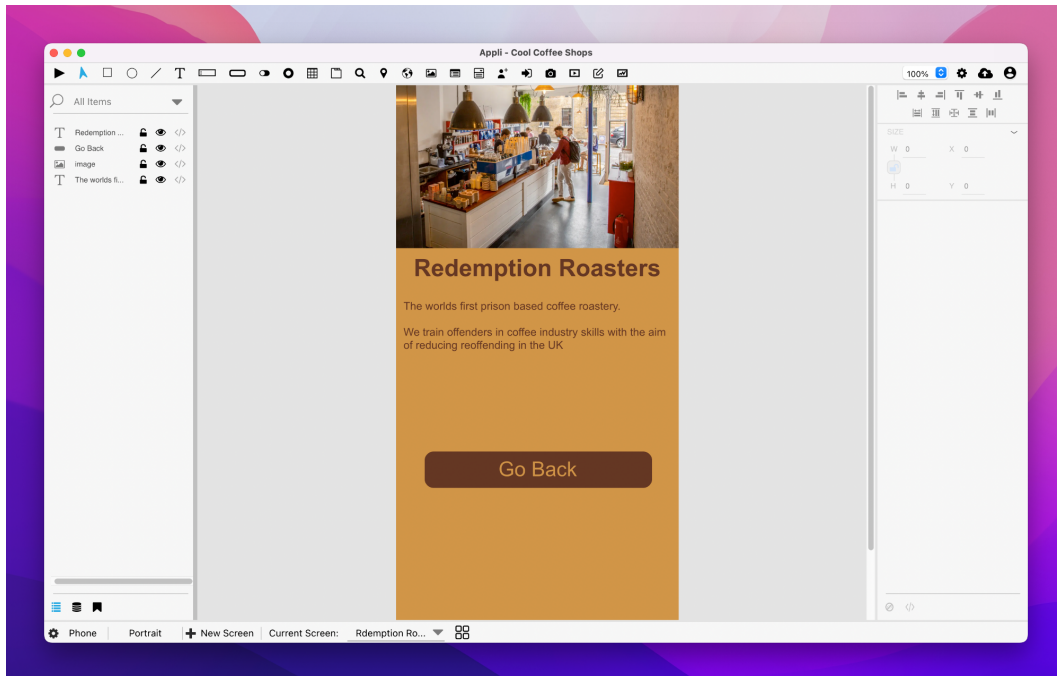
Using the controls at the footer, create a new screen named *Redemption Roasters*. You'll notice that the `backgroundColor` already matches the previous screen, that is because they're set per platform.

Our new screen will be quite simple. It will contain:

- A photo from the shop.
- Two text elements, one for the header, another for a description.
- A button to go back to the Home screen.

Quick tip: you can copy and paste elements between screens. I assembled the screen in the screenshot below by copying the elements from the Home screen, pasting them on the Redemption Roasters screen, and altering their text.

Make your version of the screen look like this:

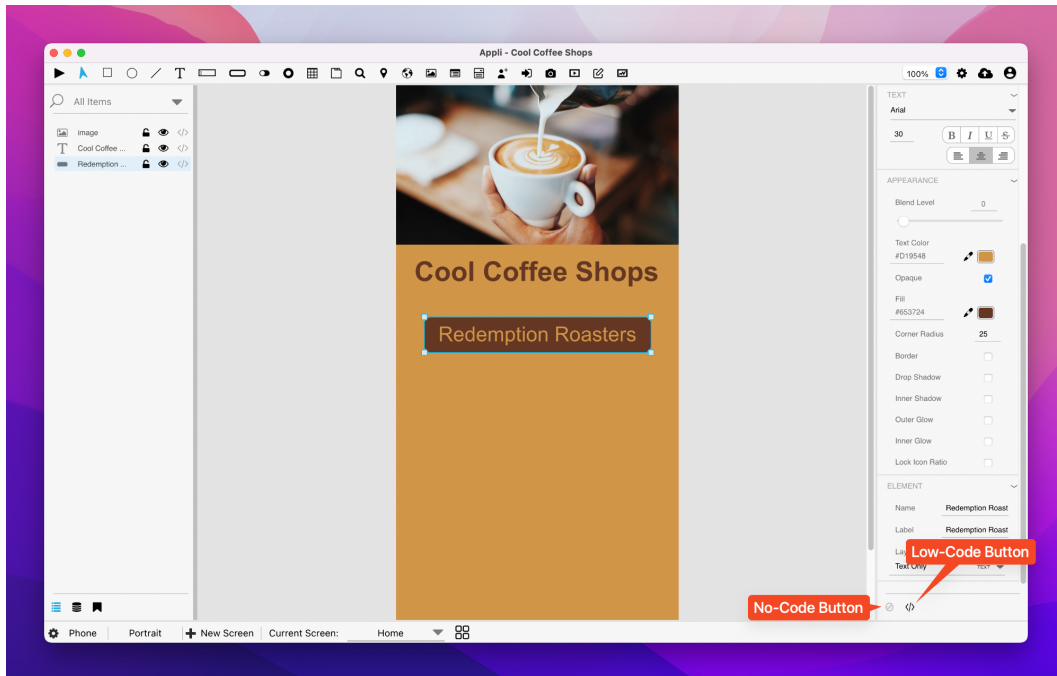


Redemption Roasters page

With those two screens in place, we're ready to add actions to those buttons and hook things up.

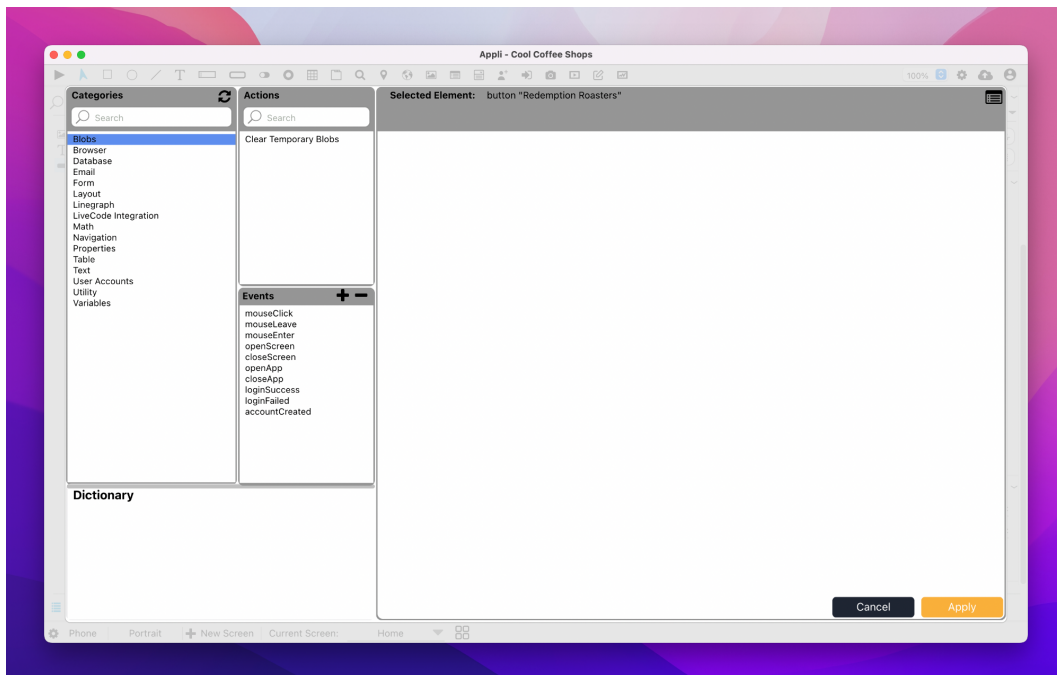
Adding an action to a button

Go back to the *Home* screen. Select the *Redemption Roasters* button. Look towards the property inspector. You'll see the *no-code button* and the *Low-Code button*. You can use them to set actions for the selected element or alter complex behaviours that are beyond what you can set using properties.



No-code and Low-code buttons

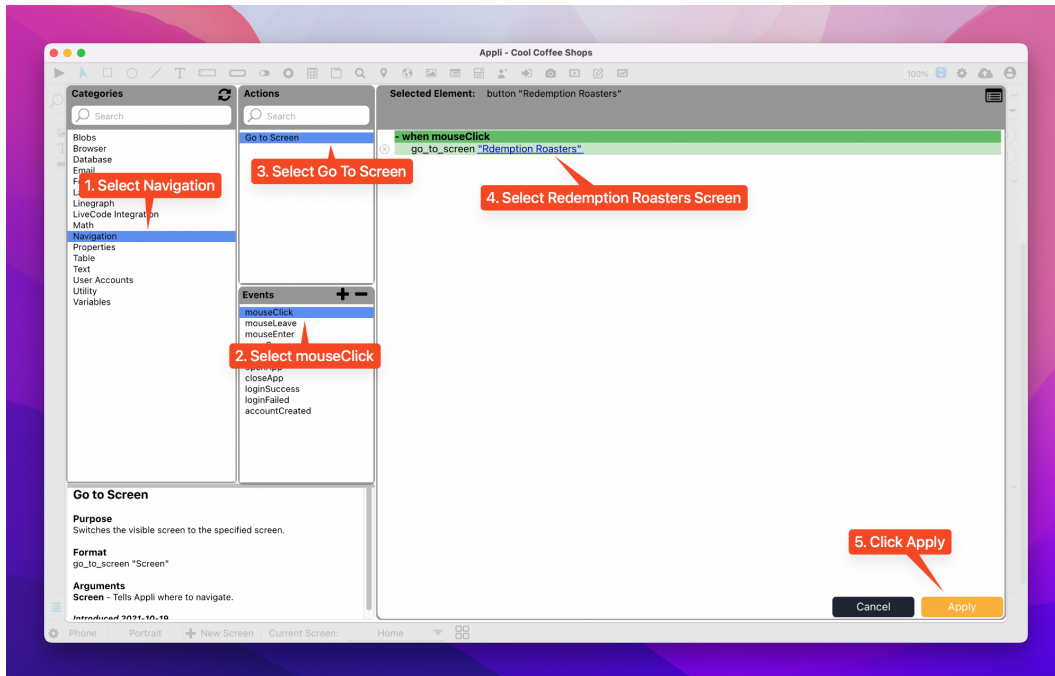
Buttons don't have *no-code* features, so that button is disabled. Clicking the *Low-Code button* will open the *Low-Code Action Script Editor*:



Low-Code interface

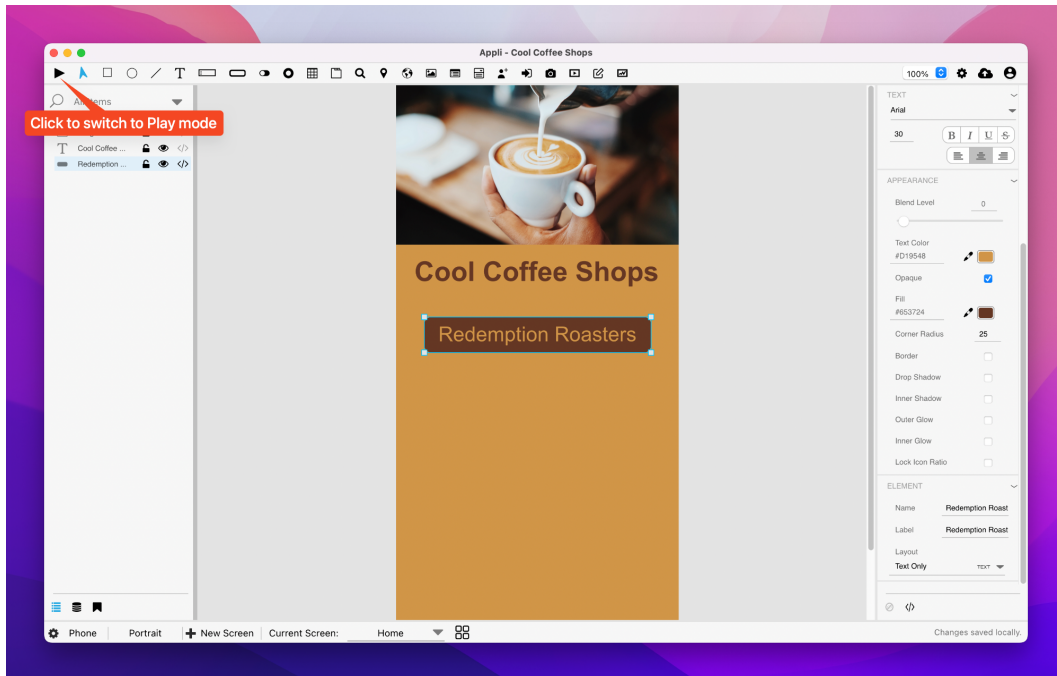
This editor is what we use to add actions to elements, you can learn more by reading the [action editor documentation](#).

The action we want to use is *Go To Screen* under the *Navigation* category. Once you select it, the *action script* will be displayed in the pane on the right. Click the blue *Screen* link to select which screen you want the button to go to when clicked.



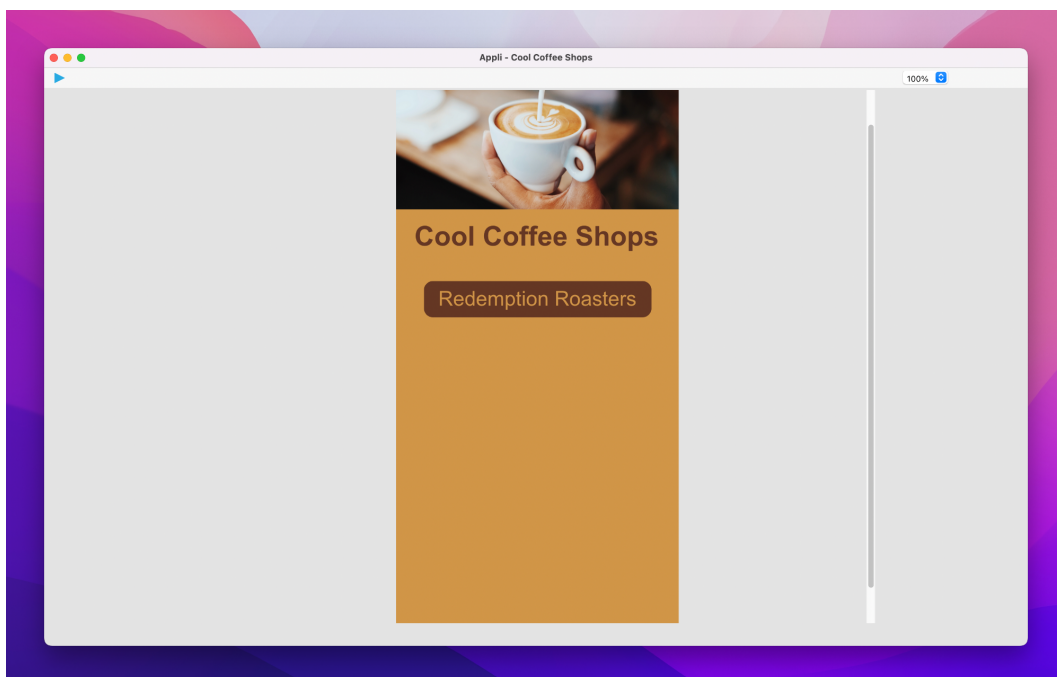
Low-Code script for the button

Done! You can click the *Play button* at the top-right of the Tools palette to switch to play mode. This mode allows you to test out your application.



Play mode switch

Try clicking the *Redemption Roasters* button, it should navigate to the *Redemption Roasters* page.



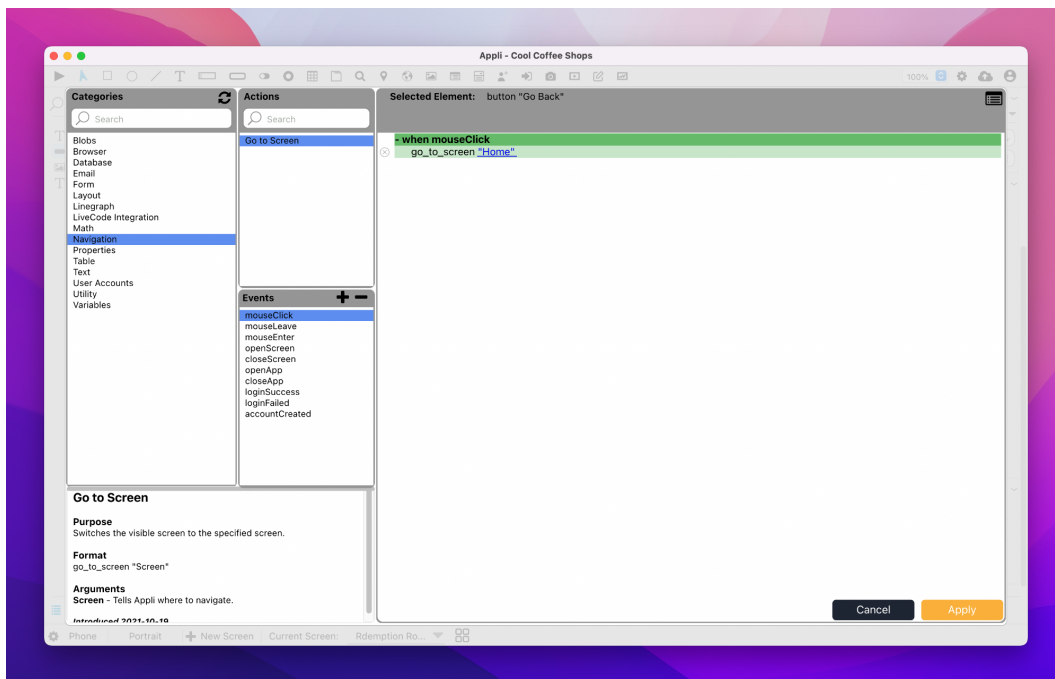
Interacting with the app in play mode

Once you're done with your testing, you can switch back to design mode by selecting the *Pointer tool* at the top-left side of the *Tools palette*.

Going back to the Home screen

Go to the *Redemption Roasters* screen. We need to wire the *Go Back* button so that it has an action to navigate back to the *Home* screen.

It is the exact same process as before. Select the element, click the *Low-Code button*, add a *Go To Screen* action and select the *Home* screen.



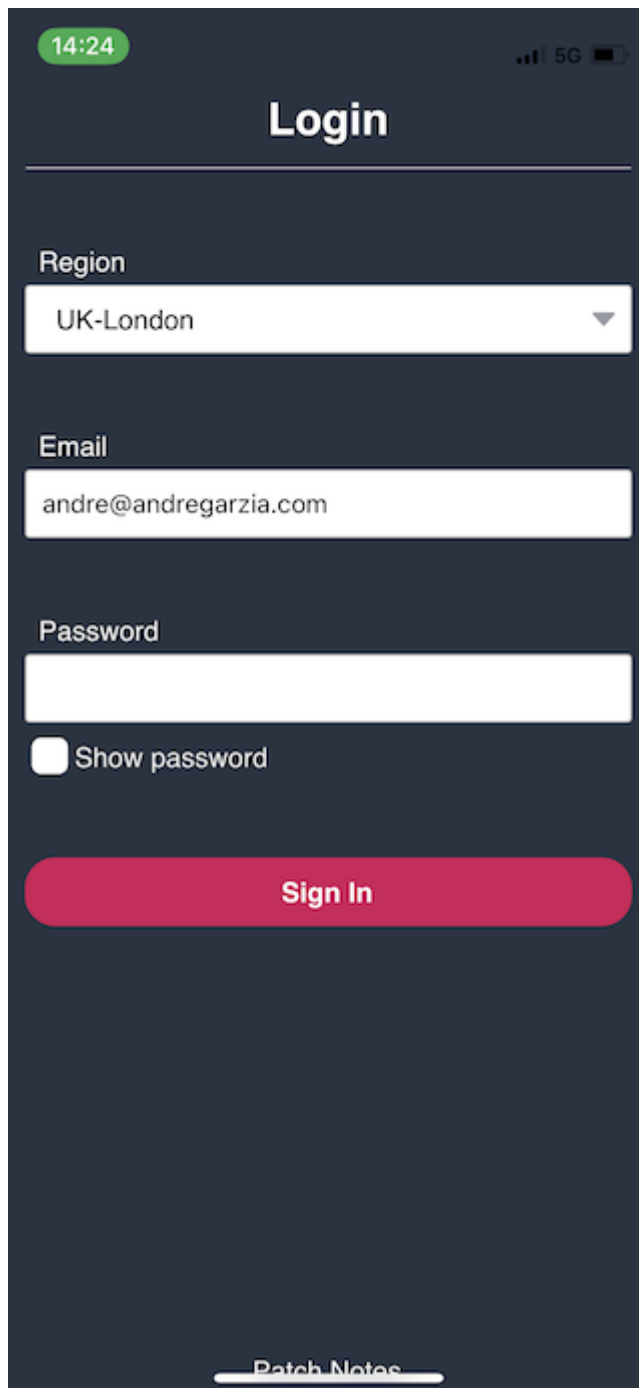
Low-Code for Go Back button

We're done with our little app! Read on to learn how to try it out in your mobile device.

RUNNING ON MOBILE

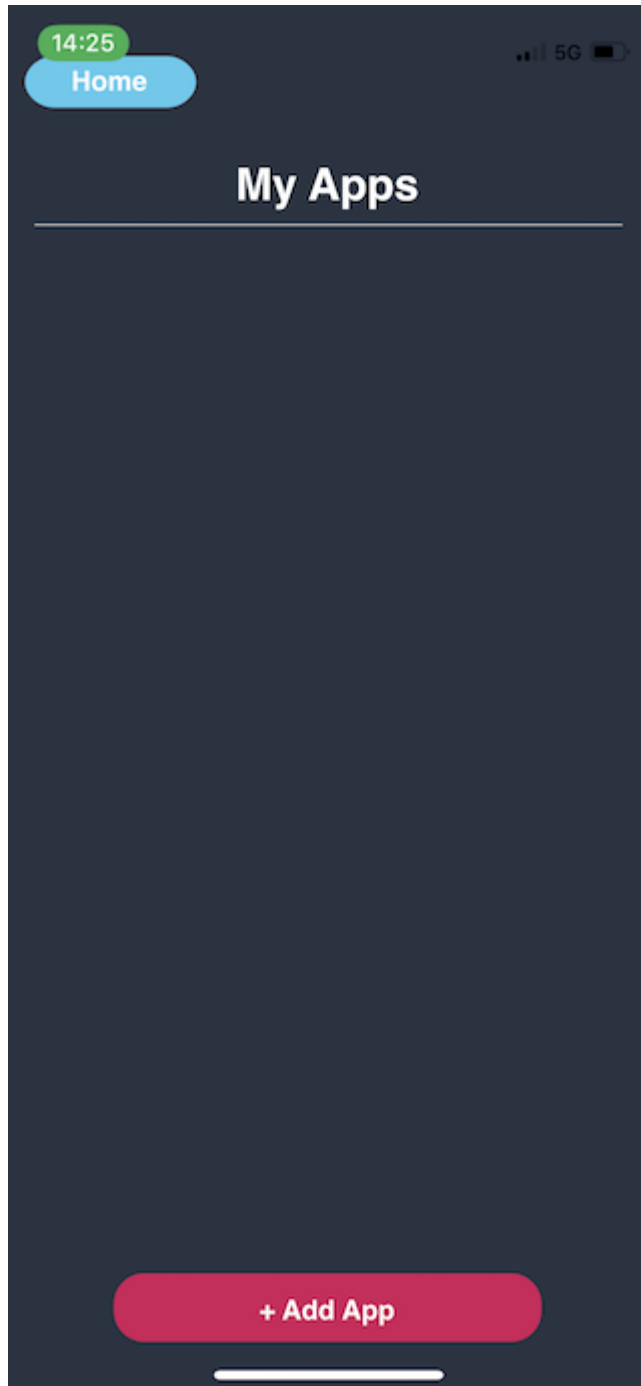
You should have the *Appli Player* installed on your mobile device for this section of the tutorial to work.

Launch the Appli Player app, and enter your login details. They are the same as the ones you used for your Appli IDE login.

A screenshot of the Appli Player mobile app's login screen. The screen has a dark blue background. At the top, the status bar shows the time 14:24, 5G signal, and battery level. The title 'Login' is centered at the top in white. Below it is a horizontal line. The form consists of three white input fields: 'Region' with a dropdown arrow showing 'UK-London', 'Email' with the text 'andre@andregarzia.com', and 'Password' which is empty. Below the password field is a checkbox labeled 'Show password'. A large red button with the text 'Sign In' is centered below the form. At the bottom, there is a link labeled 'Patch Notes'.

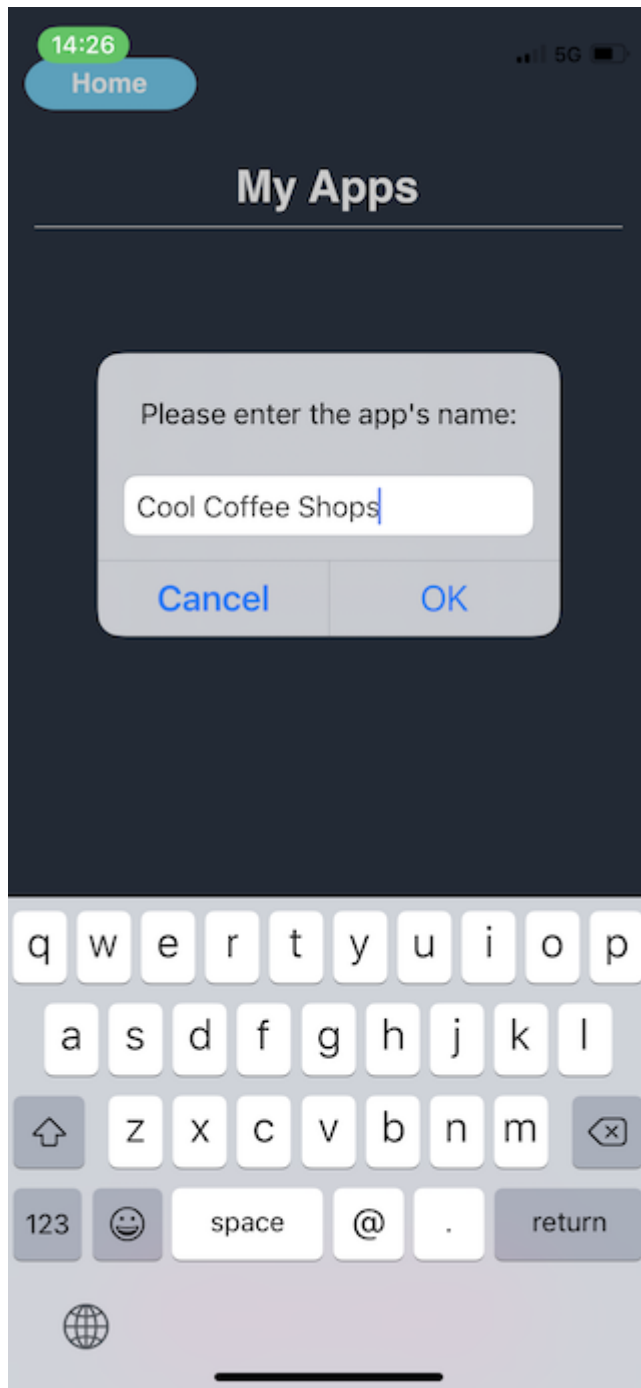
Player login

After logging in, you'll see the *Player Home Screen* that lists the apps *you have added to it*. It doesn't list your apps automatically. You need to click the *+ Add App* button at the bottom of the screen...



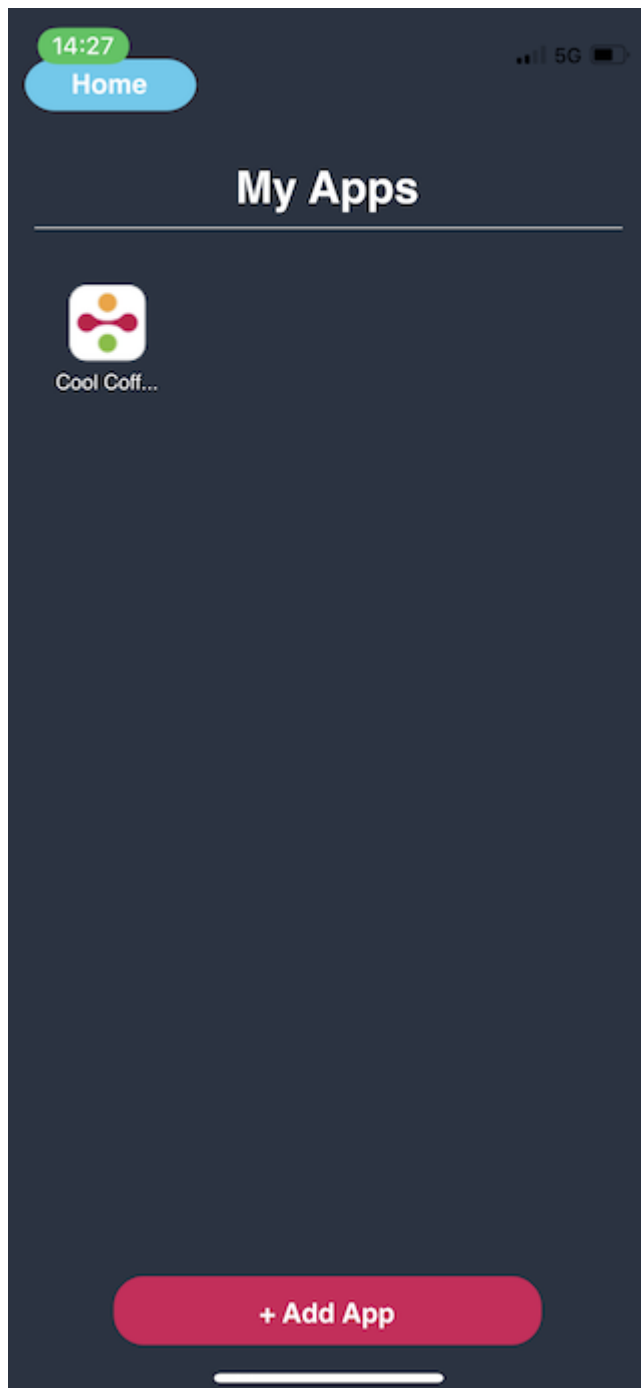
Adding an app

... and type in the name of the app you want to add.



Adding an app

The app will be appear listed on the Home screen.



Adding an app

Clicking on it will launch your app. You can interact with it. There is a floating draggable round button that you can use to refresh the app in case you made changes since launching it, and other controls to sign out and go back home.



Adding an app

FINAL WORDS AND WHERE TO GO
NEXT

You might have noticed that I mentioned that we would add two coffee shops to this app. That second coffee shop page is your homework. Add in your favourite coffee shop, or any other kind of shop, to the app and test it on the Appli player.

ELEMENTS

- Browser
- Button
- Camera
- Create Account
- Dropdown
- Field
- Form
- Graphic
- Image
- Layout
- LineGraph
- Login
- Map
- Media
- Radio Group
- Search Field
- Switch
- Tab Menu
- Table
- Text

ELEMENT: BROWSER

This element is a WebView that allows the developer to display content from a Web page inside their application.

The browser URL can be set using the property inspector or using the low-code action *Browser → Set Browser URL*.

PROPERTIES

Size Section

This section is used to configure the browser size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.

Property	Description
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Element Section

This section is contains properties that are specific to elements of type browser.

Property	Description
name	The name of the element. Displayed in the project browser.
url	A URL. Also known as a Web Address, it points to a network accessible resource.

ELEMENT: BUTTON

This element is a clickable button whose behavior can be configured using low-code ActionScript.

Buttons are among the most useful and common elements in an app.

EVENTS

Buttons have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the button.

PROPERTIES

Size Section

This section is used to configure the button size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the button.

Property	Description
textSize	The text size used in the element's text.
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.
textAlignment	The text alignment used for the element's text. It can be aligned to the left, right, or center.

Appearance Section

This section is used to configure how the button looks.

Property	Description
blendLevel	How transparent the element is.
foregroundColor	The color used in text.

Property	Description
showBackground	If the element background should be opaque or transparent.
backgroundColor	The color used for the element's background.
roundRadius	How round the corners of the element are.
showBorder	Turns the visibility of the element's borders on or off.
dropShadow	The drop shadow for the element.
innerShadow	The inner shadow for the element.
outerGlow	The outer glow for the element.
innerGlow	The inner glow for the element.
lockIconRatio	Locks the placement of the icon in relationship with the dimensions of the element.
lineSize	The size of a line in pixels.
borderColor	The color used for the border of the element.

Element Section

This section contains properties that are specific to elements of type button.

Property	Description
name	The name of the element. Displayed in the project browser.
label	The text label for the element.
layout	Element-specific property that configures how it should be displayed.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: CAMERA

This element displays the content of a camera. The camera can be a built-in camera or some attached webcam. It can also be used to scan barcodes and QR codes.

EXTRA CONFIGURATION

Using the no-code properties, the developer can configure which element receives the output from the camera element.

PROPERTIES

Size Section

This section is used to configure the camera size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
----------	-------------

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the camera.

Property	Description
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.

Appearance Section

This section is used to configure how the camera looks.

Property	Description
blendLevel	How transparent the element is.

Element Section

This section contains properties that are specific to elements of type camera.

Property	Description
----------	-------------

Property	Description
name	The name of the element. Displayed in the project browser.
device	Selects which camera device should be used for the camera element.
barcodeScanner	Toggles if the camera should behave like a barcode scanner or not.
flashMode	Configures the behavior of the flash for the camera element.
focusMode	Configures the focus for the camera element.

ELEMENT: CREATE ACCOUNT

This element is a form that creates accounts. It has all the necessary fields and buttons for account creation already in place.

The fields are: first name, last name, email, password, and a password confirmation dialog. If your needs are different, you can roll your own form very easily.

EVENTS

Create account forms have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks the *Create Account* button.

PROPERTIES

Size Section

This section is used to configure the create account size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the create account.

Property	Description
textSize	The text size used in the element's text.

Appearance Section

This section is used to configure how the create account looks.

Property	Description
blendLevel	How transparent the element is.
gap	The distance between various parts of an element. For example, in a field form it is the distance between fields.
fieldHeight	How tall the field is in pixels.
textColor	The color of the text in the element.
placeholderColor	The color used for placeholder text.
fieldColor	The color used for the background of a field.

Property	Description
showBorder	Turns the visibility of the element's borders on or off.
lineSize	The size of a line in pixels.
lineColor	The color of the line.
buttonHeight	How tall is the button.
buttonColor	The color for a button.
buttonLabelColor	The color used for the text label in a button.

Element Section

This section is contains properties that are specific to elements of type create account.

Property	Description
name	The name of the element. Displayed in the project browser.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: DROPDOWN

This element holds a menu. Clicking the element causes a scrolling list of options to open. Selecting an item will close back the dropdown menu.

EVENTS

Dropdowns have a *mouseUp* event that can be configured using low-code. This event is triggered when the user selects one of the available options.

PROPERTIES

Size Section

This section is used to configure the dropdown size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the dropdown.

Property	Description
textSize	The text size used in the element's text.
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.
textAlignment	The text alignment used for the element's text. It can be aligned to the left, right, or center.

Appearance Section

This section is used to configure how the dropdown looks.

Property	Description
blendLevel	How transparent the element is.
textColor	The color of the text in the element.
backgroundColor	The color used for the element's background.
labelColor	The color used for the text in the label of the element.
arrowColor	The color used for the arrow inside the element.

Property	Description
showBorder	Turns the visibility of the element's borders on or off.
lineSize	The size of a line in pixels.
lineColor	The color of the line.

Element Section

This section is contains properties that are specific to elements of type dropdown.

Property	Description
name	The name of the element. Displayed in the project browser.
options	A list of options used by the element.
optionsToDisplay	How many options should be displayed by the element. If there are more options available than the value of this property, the element will display a scrollbar.
label	The text label for the element.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: FIELD

This element allows the user to enter or edit text.

EVENTS

Fields have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the field.

PROPERTIES

Size Section

This section is used to configure the field size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.

Property	Description
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the field.

Property	Description
textSize	The text size used in the element's text.
textAlignment	The text alignment used for the element's text. It can be aligned to the left, right, or center.

Appearance Section

This section is used to configure how the field looks.

Property	Description
blendLevel	How transparent the element is.
hintTextColor	The color used by the hint text.
textColor	The color of the text in the element.
backgroundColor	The color used for the element's background.
showBorder	Turns the visibility of the element's borders on or off.
lineSize	The size of a line in pixels.
lineColor	The color of the line.

Element Section

This section contains properties that are specific to elements of type field.

Property	Description
name	The name of the element. Displayed in the project browser.
hintText	The hint text.
useNative	Replaces the element with the native version of that element for the running platform. Used by text fields.
multiLine	Configures the text field to accept multiline text instead of single line input.
passwordField	Marks the field as a password entry. This masks the input so that it is not readable.
keyboardType	Configure what kind of keyboard is to be used when entering data in the element.
autoCorrectionType	Configures if the element should use auto-correction features.
autoCapitalizationType	Configures if the element should use auto-capitalization of its data.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: GRAPHIC

This element represents a graphic on the screen. Appli has distinct controls on the toolbar to make the most common graphics: rectangles, ellipses, and lines.

EVENTS

Graphics have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the graphic.

PROPERTIES

Size Section

This section is used to configure the graphic size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Appearance Section

This section is used to configure how the graphic looks.

Property	Description
blendLevel	How transparent the element is.
opaque	If the element is opaque or if it's background is transparent.
foregroundColor	The color used in text.
showBackground	If the element background should be opaque or transparent.
backgroundColor	The color used for the element's background.
roundRadius	How round the corners of the element are.
showBorder	Turns the visibility of the element's borders on or off.
dropShadow	The drop shadow for the element.
innerShadow	The inner shadow for the element.
outerGlow	The outer glow for the element.
innerGlow	The inner glow for the element.
fillGradient	The gradient color used to fill the element.

Element Section

This section is contains properties that are specific to elements of type graphic.

Property	Description
name	The name of the element. Displayed in the project browser.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: IMAGE

This element contains an image.

Some properties are only available if you apply a mask to the image.

EVENTS

Images have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the image.

PROPERTIES

Size Section

This section is used to configure the image size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Appearance Section

This section is used to configure how the image looks.

Property	Description
blendLevel	How transparent the element is.
blur	How much blur is applied to the element content.
useMask	If a mask should be applied to the element.
shapeStyle	The shape to be used for masking the element.
lineSize	The size of a line in pixels.
lineColor	The color of the line.
preserveAspectRatio	If the element should preserve the aspect ratio of it's data.

Element Section

This section contains properties that are specific to elements of type image.

Property	Description
name	The name of the element. Displayed in the project browser.
imageFile	The file containing the image to be used in the element.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: LAYOUT

This element collects other elements as a group.

NO CODE

There are extra options available for Layout elements that can be configured using *no code*.

One can use records on a database to configure the elements inside a Layout.

ACTIONS

These are the *low code* actions available for Layout elements.

Refresh the layout

Used to refresh the records of a Layout element.

Example: after changing the records in a database, you can use this action to update the elements inside the Layout to reflect the current database state.

Set variable from Context

Each element inside a Layout has a *Context* (i.e. a reference to the *RecordID*). Using this action, one can update a variable based on a value from that context.

EVENTS

Layouts have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the layout.

PROPERTIES

Size Section

This section is used to configure the layout size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Appearance Section

This section is used to configure how the layout looks.

Property	Description
showBorder	Turns the visibility of the element's borders on or off.
borderColor	The color used for the border of the element.
lineSize	The size of a line in pixels.
opaque	If the element is opaque or if it's background is transparent.
backgroundColor	The color used for the element's background.
showScrollbar	If the scrollbar should be visible or not.

Element Section

This section contains properties that are specific to elements of type layout.

Property	Description
name	The name of the element. Displayed in the project browser.
multipleRows	If the element should contain multiple rows.
bottomMargin	The value in pixels of the bottom margin.

ELEMENT: LOGIN

This element represents a login form. It contains the most common fields that are needed to perform a login operation: email and password.

EVENTS

Login elements have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the Login button.

PROPERTIES

Size Section

This section is used to configure the login size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Appearance Section

This section is used to configure how the login looks.

Property	Description
blendLevel	How transparent the element is.
fieldHeight	How tall the field is in pixels.
textColor	The color of the text in the element.
placeholderColor	The color used for placeholder text.
fieldColor	The color used for the background of a field.
showBorder	Turns the visibility of the element's borders on or off.
lineSize	The size of a line in pixels.
borderColor	The color used for the border of the element.
buttonHeight	How tall is the button.
buttonColor	The color for a button.
buttonLabelColor	The color used for the text label in a button.

Element Section

This section contains properties that are specific to elements of type login.

Property	Description
-----------------	--------------------

Property	Description
name	The name of the element. Displayed in the project browser.

ELEMENT: MAP

This element contains a map with markers.

EVENTS

Maps have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the map.

PROPERTIES

Size Section

This section is used to configure the map size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.

Property	Description
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Element Section

This section is contains properties that are specific to elements of type map.

Property	Description
name	The name of the element. Displayed in the project browser.
Markers	A collection of geolocation markers to be added to a Map element.
APIKey	A Google Maps API Key.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: RADIO GROUP

This element is a group of radio buttons. The user can select only one radio button in a group.

EVENTS

Radio groups have a *mouseUp* event that can be configured using low-code. This event is triggered when the user selects one of the options.

PROPERTIES

Size Section

This section is used to configure the radio group size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
----------	-------------

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the radio group.

Property	Description
textSize	The text size used in the element's text.
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.

Appearance Section

This section is used to configure how the radio group looks.

Property	Description
textColor	The color of the text in the element.
blendLevel	How transparent the element is.
hiliteColor	The color of the hilited state of the element.

Element Section

This section contains properties that are specific to elements of type radio group.

Property	Description
name	The name of the element. Displayed in the project browser.
useIcons	If the element should use icons.
options	A list of options used by the element.
label	The text label for the element.
showLabel	Configures if the label is visible.
orientation	The element's orientation.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: SEARCH FIELD

This element represents a search box.

The element has two labels, one is used for the focused state and the other for the unfocused state.

EVENTS

Search elements have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the magnifying glass button inside the search element.

PROPERTIES

Size Section

This section is used to configure the search field size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the search field.

Property	Description
textSize	The text size used in the element's text.
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.
textAlignment	The text alignment used for the element's text. It can be aligned to the left, right, or center.

Appearance Section

This section is used to configure how the search field looks.

Property	Description
blendLevel	How transparent the element is.
textColor	The color of the text in the element.
iconColor	The color of the icon.
backgroundColor	The color used for the element's background.
showBorder	Turns the visibility of the element's borders on or off.

Property	Description
lineSize	The size of a line in pixels.
lineColor	The color of the line.
showLabel1	For elements with more than one label, this toggles the visibility of the first label.
label1Color	For elements with more than one label, this specifies the color for the first label.
showLabel2	For elements with more than one label, this toggles the visibility of the second label.
label2Color	For elements with more than one label, this specifies the color for the second label.

Element Section

This section is contains properties that are specific to elements of type search field.

Property	Description
name	The name of the element. Displayed in the project browser.
label1	For elements with more than one label, this specifies the content for the first label.
label2	For elements with more than one label, this specifies the content for the second label.
iconPlacement	How the icon should be placed inside the element.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: SWITCH

This element is a switch that can be toggled on or off.

EVENTS

Switches have a *mouseUp* event that can be configured using low-code. This event is triggered when the user toggles the switch.

Appli treats the on and off state of a switch a true or false value.

PROPERTIES

Size Section

This section is used to configure the switch size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.

Property	Description
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the switch.

Property	Description
textSize	The text size used in the element's text.
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.

Appearance Section

This section is used to configure how the switch looks.

Property	Description
blendLevel	How transparent the element is.
textColor	The color of the text in the element.
hiliteColor	The color of the hilited state of the element.

Element Section

This section contains properties that are specific to elements of type switch.

Property	Description
name	The name of the element. Displayed in the project browser.
trueLabel	The text for the true state of the element.
falseLabel	The text for the false state of the element.
value	If the element should show a value.
showLabel	Configures if the label is visible.
lowCode	ActionScript that controls the behavior of the element.

ELEMENT: TAB MENU

This element is a tab menu. It can contain multiple tabs each with their own collection of elements.

Each tab contains their own elements. To edit a specific tab, change the `selectedTab` property using the property inspector, and then add elements to it by dragging and dropping them on top of the tab.

To change a tab name, select the tab using the `selectedTab` property and change the `tabName` property. That property always display the name of the selected tab.

EVENTS

Tab menus have a *mouseUp* event that can be configured using low-code. This event is triggered when the user changes the selected tab.

PROPERTIES

Size Section

This section is used to configure the tab menu size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.

Property	Description
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the tab menu.

Property	Description
textSize	The text size used in the element's text.
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.
textAlignment	The text alignment used for the element's text. It can be aligned to the left, right, or center.

Appearance Section

This section is used to configure how the tab menu looks.

Property	Description
headerTextColor	The color used by the header text.
backgroundColor	The color used for the element's background.
headerColor	The color used by the header background.
blendLevel	How transparent the element is.
activeTabColor	The color used to hilite which tab is selected.

Element Section

This section is contains properties that are specific to elements of type tab menu.

Property	Description
name	The name of the element. Displayed in the project browser.
numberOfTabs	How many tabs the tab menu element contains.
selectedTab	Which of tabs is the active tab.
tabName	The name of the selected tab.

ELEMENT: TABLE

This element contains a table of records. Tables can be used to display and edit records. The `canEditData` property configures if the table is read-only or not.

TABLE SETUP

Use *no-code* to configure the table. Tables can be local, cloud, of hybrid. Using the *no-code* select which table and keys should be displayed on the table.

ACTIONS

These are the *low code* actions available for table elements.

Refresh the Table

Used to refresh the records of a table element.

Example: after changing the records in a database, you can use this action to update the elements inside the table to reflect the current database state.

Count Displayed Records

Counts the amount of records in the table and place that value in a variable or another element.

EVENTS

Tab menus have a *mouseUp* event that can be configured using low-code. This event is triggered when the user changes the selected tab.

PROPERTIES

Size Section

This section is used to configure the table size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the table.

Property	Description
textSize	The text size used in the element's text.
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.
textAlignment	The text alignment used for the element's text. It can be aligned to the left, right, or center.

Appearance Section

This section is used to configure how the table looks.

Property	Description
blendLevel	How transparent the element is.
headerColor	The color used by the header background.
headerTextColor	The color used by the header text.
headerVerticalAlign	The vertical alignment of the header.
rowColor1	For elements with alternating row colors, this is the color for the odd rows.
rowColor2	For elements with alternating row colors, this is the color for the even rows.
rowHeight	The height for the row.
rowVerticalAlign	The vertical alignment for the row.
bodyTextColor	The color used by the body text.
columnAlign	The alignment for the column.
showBorder	Turns the visibility of the element's borders on or off.

Property	Description
lineColor	The color of the line.
hiliteColor	The color of the hilited state of the element.
useRowColorForEditCell	If the element should use the row color for the edit cell background.
editCellTextColor	The color used by the text inside an edit cell.
editCellHiliteColor	The hilite color used by edit cells.

Element Section

This section is contains properties that are specific to elements of type table.

Property	Description
name	The name of the element. Displayed in the project browser.
canEditData	If the data shown in the element is editable.
adjustColumnsOnPlayer	If the columns should be adjusted depending on the player size.

ELEMENT: TEXT

This element contains non-editable text.

TEXT SETUP

The contents of a text element can be set from a database record. Use the *no-code* interface to select the table and recordID used for the text element.

ACTIONS

These are the *low code* actions available for text elements.

Format The Text

This action can be used to format the text. The current available format is *USD Currency*. The formatted text can be placed in a variable or inside an element.

EVENTS

Text elements have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the element.

PROPERTIES

Size Section

This section is used to configure the text size and position.

Property	Description
-----------------	--------------------

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Text Section

The properties in this section are used together with those in the appearance section to configure how the element is displayed on the screen. This section deals with the properties related to text used by the text.

Property	Description
textSize	The text size used in the element's text.

Property	Description
textStyle	The style used in the element's text. It can be bold, italic, underlined, and strikethrough.
textAlignment	The text alignment used for the element's text. It can be aligned to the left, right, or center.

Appearance Section

This section is used to configure how the text looks.

Property	Description
lockText	Prevents the text from the element from changing.
dynamicTextSize	Changes the size of the fontface used for the text depending on the dimensions of the element.
dynamicHeight	Change the height of the element based on the text content.
dynamicWidth	Change the width of the element based on the text content.
dontWrap	Prevents the text content from wrapping.
truncateWithEllipsis	Truncates the text and places an ellipsis character at the end.
showScrollbar	If the scrollbar should be visible or not.
verticalAlign	If the element should be aligned vertically.
leftMargin	The value in pixels for the left margin.
rightMargin	The value in pixels for the right margin.
fixedLineHeight	If the element should use a fixed line height.
textHeight	The line height in pixels.
blendLevel	How transparent the element is.
textColor	The color of the text in the element.
opaque	If the element is opaque or if it's background is transparent.

Element Section

This section is contains properties that are specific to elements of type text.

Property	Description
name	The name of the element. Displayed in the project browser.

ELEMENT: MEDIA

This element displays content from a media file.

EVENTS

Media elements have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the element.

PROPERTIES

Size Section

This section is used to configure the media size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.

Property	Description
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Appearance Section

This section is used to configure how the media looks.

Property	Description
showBorder	Turns the visibility of the element's borders on or off.
mirrored	If the content of the element should be mirrored (flipped horizontally).

Element Section

This section contains properties that are specific to elements of type media.

Property	Description
name	The name of the element. Displayed in the project browser.
mediaFile	The file used by the media element.
autoPlay	If the video should play automatically.
looping	If the video should restart automatically.
showController	If the video element should show user-accessible controls.

ELEMENT: FORM

This element collects other elements as part of a form.

NO CODE

Forms can be connected to Tables using *no-code*. This way they can be used to create or edit records.

LOW CODE

There is a property that can be set using *low-code* called `dataRecordID`. Use the `set_property_from_variable` action to set this property and connect a form to a specific record in a table.

ACTIONS

These are the *low code* actions available for Form elements.

Refresh the form

If the form is tied to a record in a table, this can be used to refresh the elements inside the form and update them to the new state of the record.

Submit Form To Database

Saves the data from the form to the associated table.

EVENTS

Forms have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the element.

PROPERTIES

Size Section

This section is used to configure the form size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Appearance Section

This section is used to configure how the form looks.

Property	Description
showBorder	Turns the visibility of the element's borders on or off.
borderColor	The color used for the border of the element.
lineSize	The size of a line in pixels.
opaque	If the element is opaque or if it's background is transparent.
backgroundColor	The color used for the element's background.

Element Section

This section contains properties that are specific to elements of type form.

Property	Description
name	The name of the element. Displayed in the project browser.

ELEMENT: LINEGRAPH

This element represents a line graph on the screen.

EVENTS

Graphics have a *mouseUp* event that can be configured using low-code. This event is triggered when the user clicks or touches the graphic.

PROPERTIES

Size Section

This section is used to configure the lineGraph size and position.

Property	Description
top	The value in pixels representing how far the element is from the top of the window.
left	The value in pixels representing how distant element is from the left side of the window.
width	The value in pixels representing the distance between the left side of the element and its right side.
height	The value in pixels representing the distance between the bottom side of the element and its top side.

Geometry Section

Use the properties in this section to configure the responsive design behaviors for the element.

Property	Description
lockAspectRatio	Locks the aspect ratio of the element.

Property	Description
responsive-x	How the element resizes responding to screen changes in the X axis.
responsive-y	How the element resizes responding to screen changes in the Y axis.
allowInIOSNotch	Allows the element to be drawn outside the safe area on iOS. Will draw underneath the notch in iOS.

Element Section

This section contains properties that are specific to elements of type lineGraph.

Property	Description
name	The name of the element. Displayed in the project browser.
showLine	Toggles the display of the line in a graph.
showXLines	Toggles the display of the vertical background lines in a graph.
showYLines	Toggles the display of the horizontal background lines in a graph.
opaque	If the element is opaque or if its background is transparent.
markerColor	Sets the color for markers in a graph.
markerShape	Configures which shape to use for markers in a graph.
axisData	The data to be used by the axis in a graph.
markerData	The data used to plot markers on a graph.
lowCode	ActionScript that controls the behavior of the element.

IMAGE CREDITS

-

Header Photo credits: Fahmi Fakhruudin.

- Redemption Roasters photo from their website.