

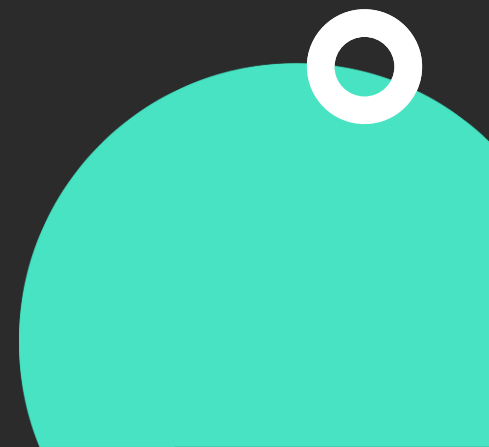


Diving into Autofill Framework in Android with Jetpack Compose to Improve User Experience



Gibson Ruitiari

[in](#) [in/GibsonRuitiari](#)



Overview

- What the **Autofill Framework** entails
- The Benefits of Using Autofill framework
- Components of the Autofill framework
- How to Integrate Autofill framework with your jetpack compose applications?
- How to test the Autofill framework integration?



The Autofill Framework

- Provides functionality for automatically filling out forms such as login, with user-related data
- The framework is available for android 8.0 and above
- The framework has two components: autofill service and autofill client



Benefits of Using Autofill in Your Jetpack Compose Apps

- Filling out forms is time consuming and error-prone
 - Autofill saves your users time from re-typing their information
 - Minimizing the need to type information minimizes errors in form of typos



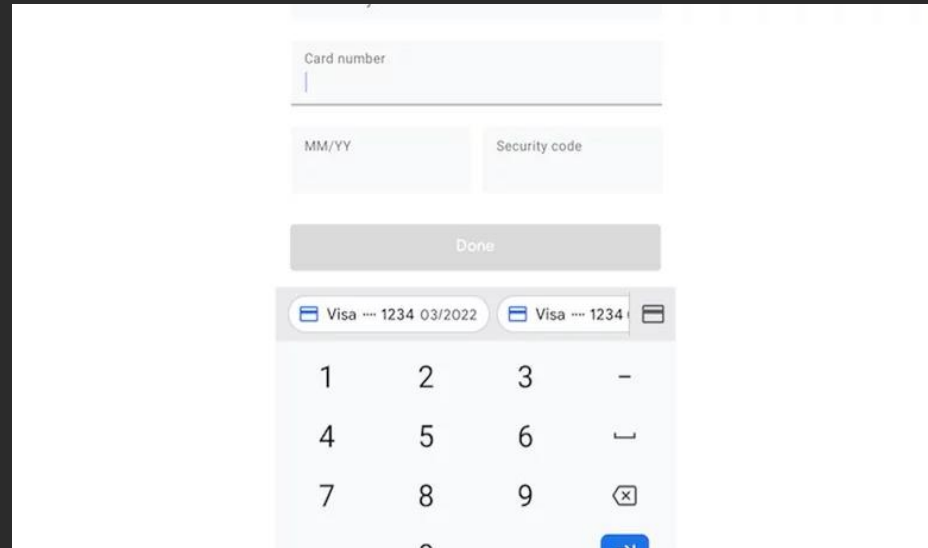
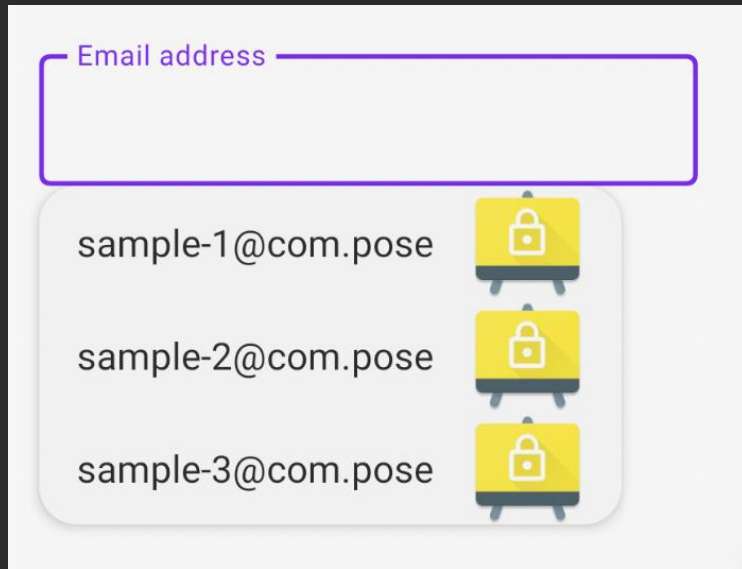
The Autofill Framework: Autofill service

- The **Autofill service** is a service that saves, stores and provides user-related data to multiple apps, and the data is used to fill out forms in those apps.
 - By default Android phones come with a pre-installed google autofill service hence in most cases you don't need to create an autofill service and you can rely on the default one



The Autofill Framework: Autofill client

- The **Autofill client** refers to application that has forms/edit texts that require to be filled out with user-related data such as password, username and email



Integrating Autofill framework with your jetpack compose applications

- Before the autofill service can provide suggestions to the client, the client needs to do several things:
 - The client needs to provide the autofill node
 - The client also has to specify the autofill hints
 - Then the client has to add the autofill node to the autofill tree
 - Lastly, the client needs to use the autofill node instance to request and cancel autofill action for the autofill node



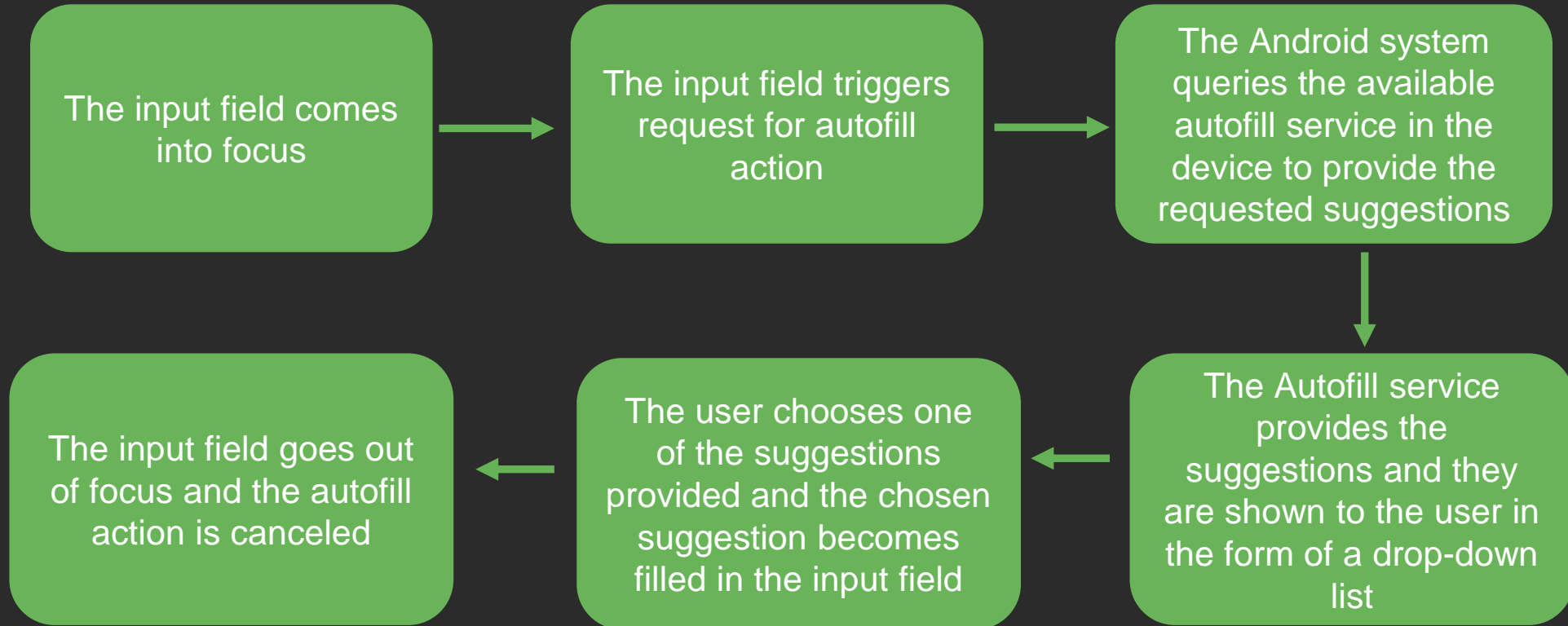
```
val email = remember{mutableStateOf("")}  
AutofillNode(autofillTypes = listOf(AutofillType.EmailAddress),  
onFill = { /* update the email textfield */ }) {autofillNode->  
    TextField(modifier=Modifier.onGloballyPosition{  
        autofillNode.boundingBox = it.boundsInWindow()  
    }, onValueChanged = {email.value=it},  
    onValue = email.value)  
}
```

An example of autofill node

- The **autofill types** are the types of data required by the autofill node, the **onFill** callback is the callback that will be called when the user selects on one of the suggestions and the **boundingBox** simply specifies the position of the suggestions drop down list

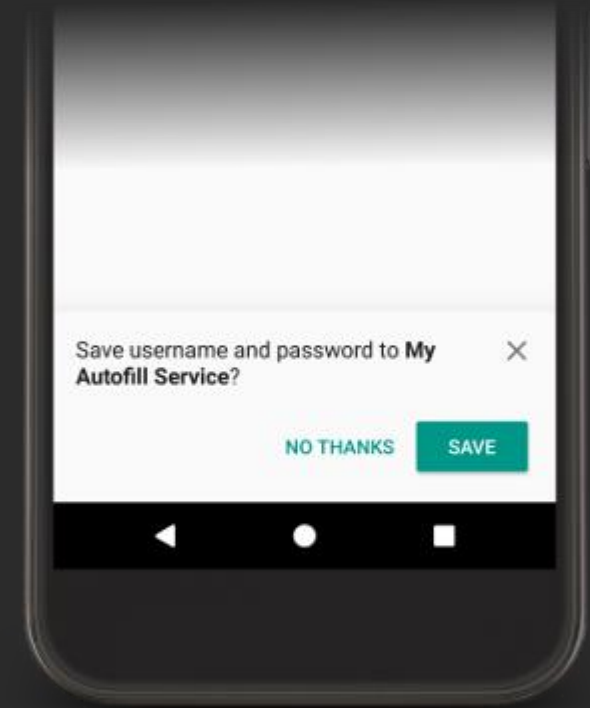


Autofill Workflow



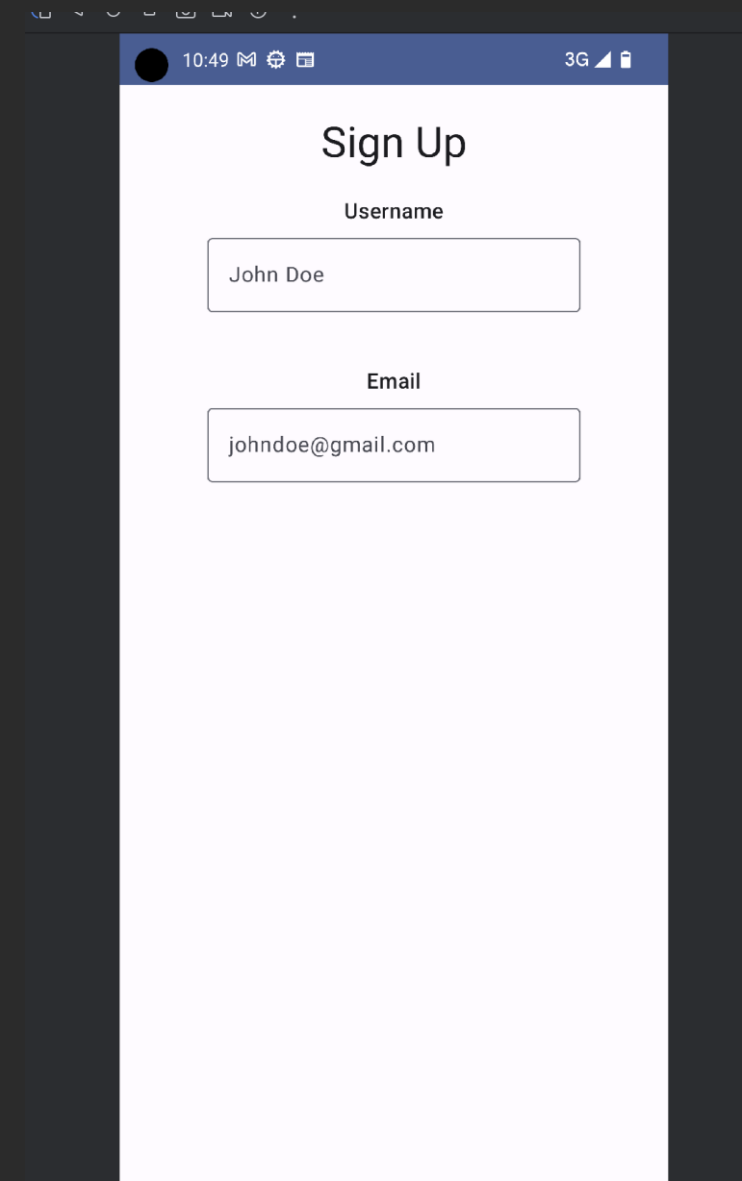
Autofill framework

- It is important to note several things when working with autofill
 - The device must have an **existing autofill** service and the service **must be enabled** on the user's device
 - The existing autofill service **may or may not have the data requested by autofill client** and this is because the user may have opted to not allow the autofill service to save the data required.



Sample App

- Sample app Overview
 - It contains a form that ought to be filled by the user
 - The **goal** is to intergrate the form's textfields with autofill framework so that the user does not have to fill each and every detail manually



Project Repository

- Explore the Starter project
 - Clone the project repository
 - Github repository link present in **study materials** of this course
 - Import the starter-project
 - Project Walk-through and explanation



Code Challenge

- In addition to the username and the email TextFields, it would be nice to have a phone number TextField.
- The challenge is to **add a phone number TextField** and integrate the TextField with autofill framework



Code Challenge Solution

- . Get the code challenge solution in the Study Materials



Summary

- Learnt what the autofill framework entails and its components
- Learnt the benefits of using autofill framework in your applications
- Learnt how to integrate autofill framework with your TextFields
- Learnt how to test the autofill framework and verify its working





Thank You!



Gibson Ruitiari

[in](#) [in/GibsonRuitiari](#)

