

Android

Increase productivity with **Gemini** in Android Studio



Adarsh Fernando

Senior Product Manager
Android Studio



@adarshfernando



Chris Sinco

UX Design Lead
Android Studio



@csinco

Table of contents

- 01 Why we're investing in Gemini in Android Studio

- 02 How to get started with Gemini and code context

- 03 Adding new functionality with Gemini in Android Studio

- 04 Getting changes ready for production

- 05 What's coming next

Why we're investing in AI



More time to focus on better experiences for users

Leveraging AI to take care of time-consuming tasks gives you and your team time back to make more amazing experiences for Android.



Higher quality code for a healthier ecosystem

The more we train Gemini to imbue best practices and help you address issues, the healthier the Android ecosystem becomes. This translates to higher user satisfaction.



Making it easier to harness the latest technology

Whether you're ramping up with Compose or trying to make your app more adaptable to larger screen sizes, Gemini can accelerate your rate of adoption with these new technologies..

Our guiding principles



Transparency and Control

Be transparent in what data we use and how we use it. Make sure the users have full control over the data they want to share.



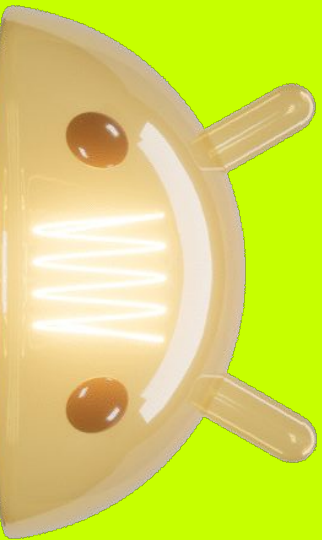
Accelerate productivity

Provide assistive tools that meet developers where they are, to accelerate their current workflow. These can be a number of smaller improvements, or large ones.



Maximize creativity

Provide experiences that allow developers to explore new ideas, experiment, iterate, and make developing with Gemini in Android Studio rewarding, productive, and fun.



Android

Getting set up with Gemini

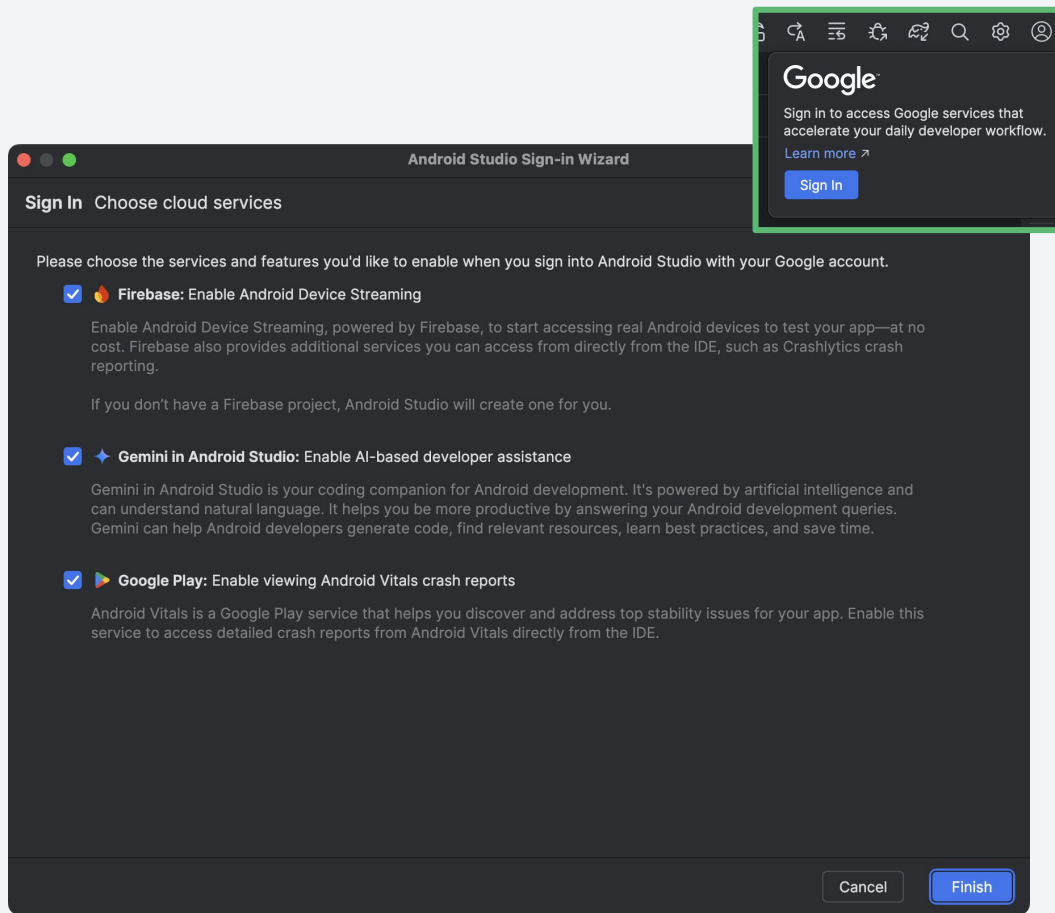
in Android Studio

Onboarding

Getting started is as easy as 1, 2, 3...

We've integrated Gemini into the canonical sign in flow

- Onboard to Gemini along with other useful developer services at no cost

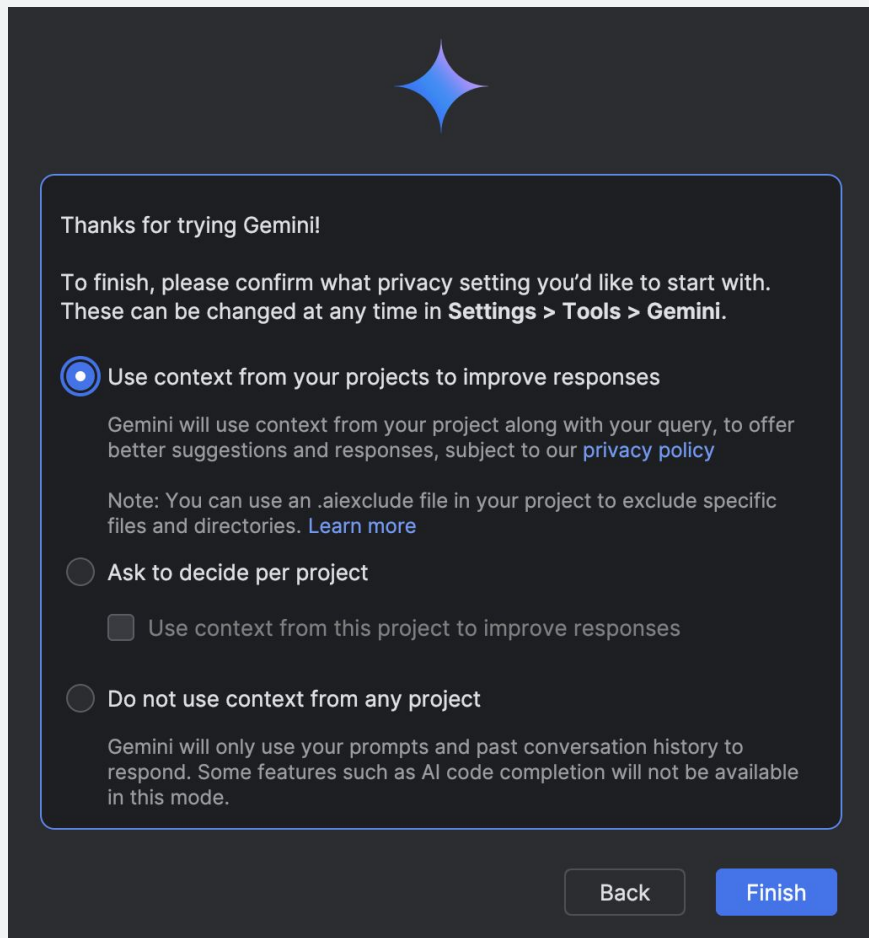


You are always in control

Share only the code context you want

As part of onboarding, Android Studio gives you control over whether to share code context with Gemini.

- Sharing code context is completely optional
- However, sharing code context improves the accuracy of the results and enables additional functionality
- You can choose to share context for all projects, or be asked to share context for each individual project



```
// .aiexclude uses same syntax/grammar as .gitignore
// exclude files or whole modules
my/sensitive/dir/

// exclude certain file extensions
*.keystore

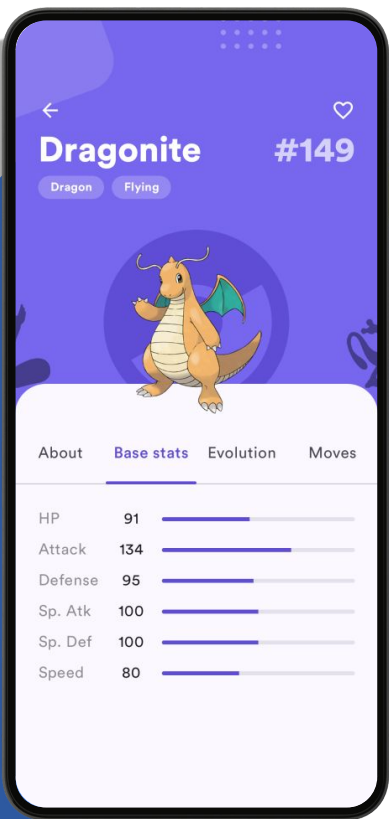
// exclude the entire project
*
```




Android

Let's get coding

with Gemini in Android Studio

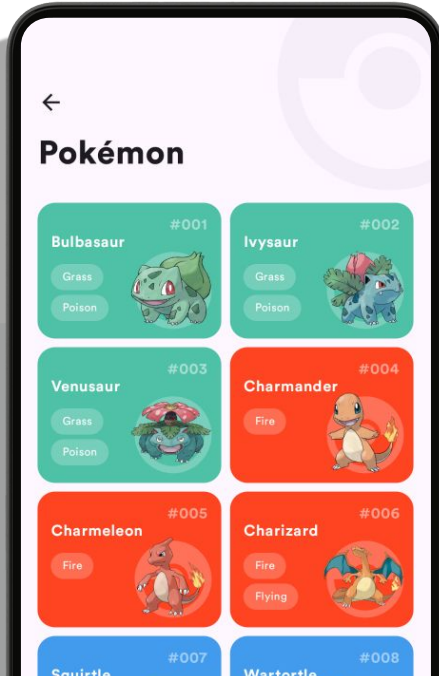


Compose Pokédexer

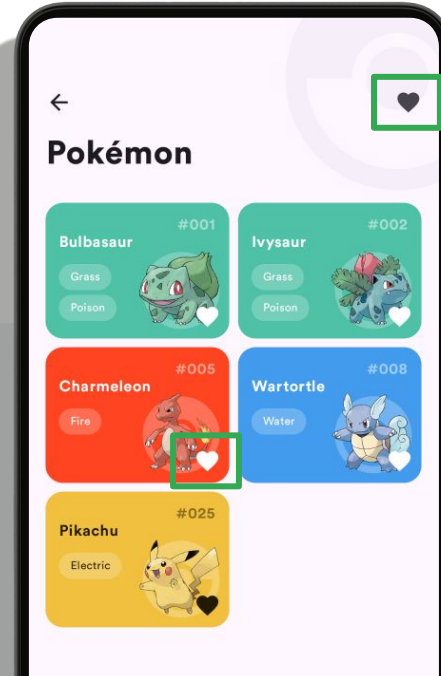
Pokedex sample app written in Compose,
powered by PokeAPI

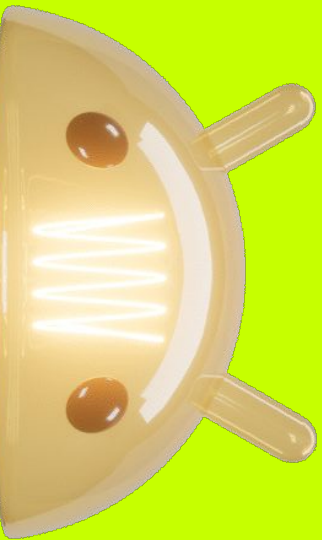
Task: Filtering favorites

Before



After





Android

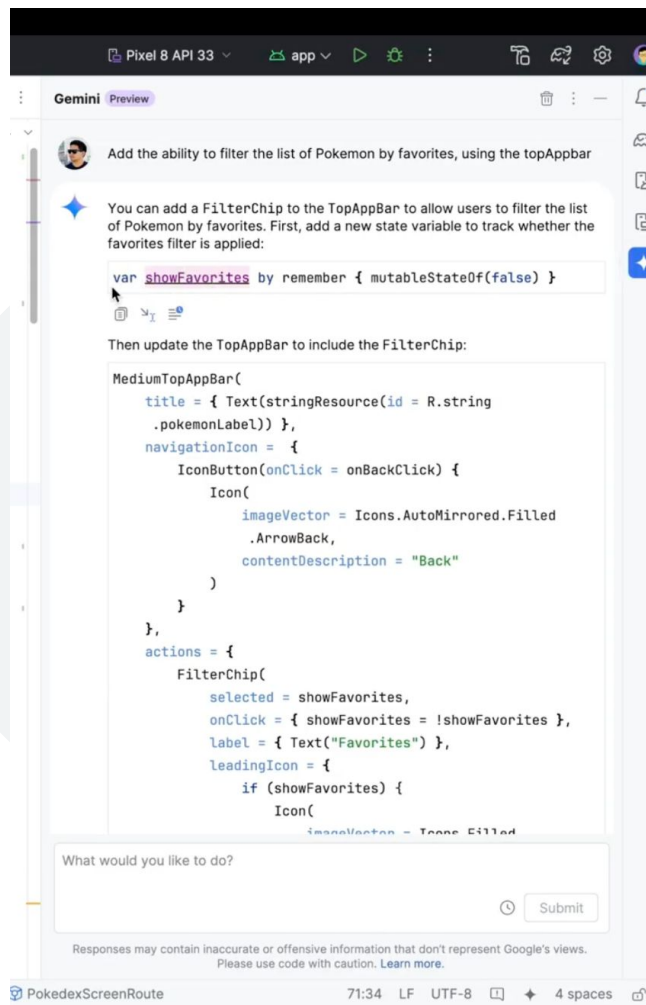
A dialog with Gemini

The Gemini tool window

Getting started by conversation

The Gemini tool window provides an opportunity to explore new problems or ideas in a conversational way. It's a great way to gain clarity on how to approach a code change before you get coding.

- Persistent context history
- Conversational interaction encourages iteration
- Quick actions to export suggestions to your code
- Even more powerful with code context enabled :)



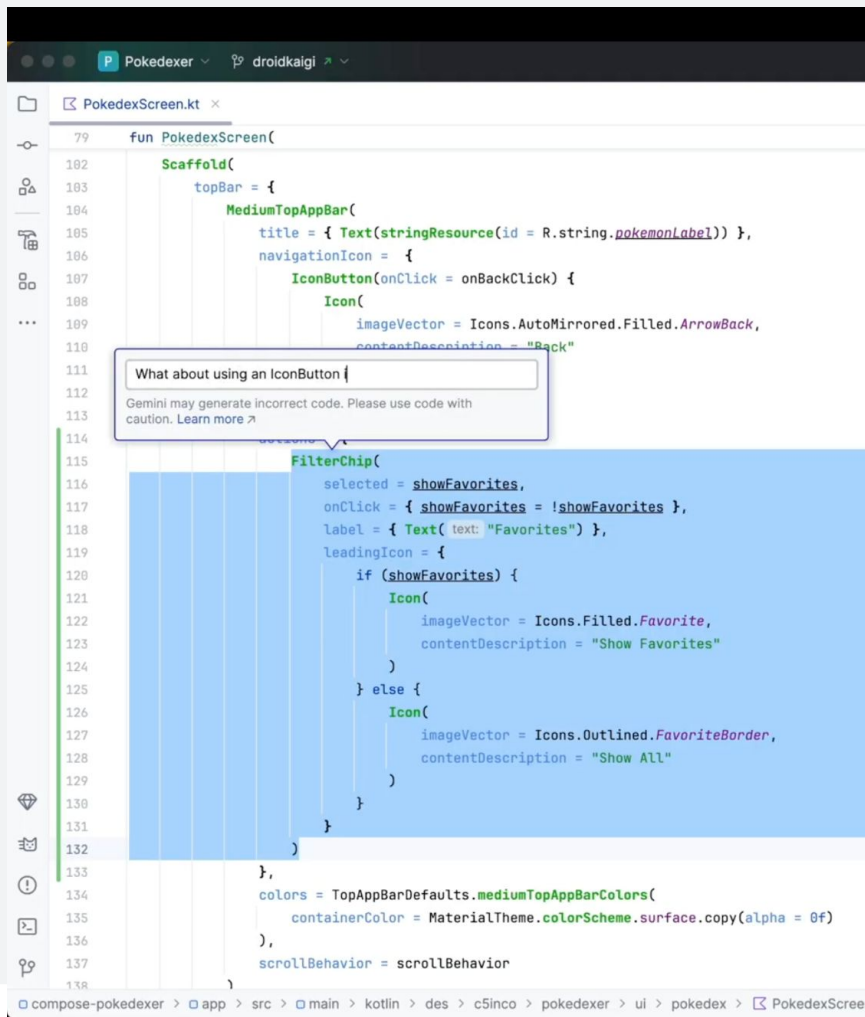
Available today

Custom Code Transformation

Prompt Gemini for immediate suggested code changes

If you have a good idea for how you want to modify your code, you can prompt Gemini directly from the code editor.

- Use the context menu: Gemini > Transform selected code
- Or keyboard shortcut:
 - Command + \
 - Control + \
- Gemini responds in the form of a code diff, where you can choose only the changes you want to accept.



Available today

Code Completion

Code context to anticipate your next lines of code

By enabling code context, Gemini can analyze the whole file context to suggest your next lines of code.

- Understands where your cursor is to determine what you're likely to type
- Utilizes whole file context to suggest code that is relevant and works with existing code
- Just hit Tab to accept

```
@Composable
private fun TypeFilterDropdownMenu(
    modifier: Modifier = Modifier,
    expanded: Boolean = false,
    onDismissRequest: () -> Unit,
    onTypeSelected: (Type) -> Unit,
) {
    DropdownMenu(
        modifier = modifier,
        expanded = expanded,
        onDismissRequest = onDismissRequest
    ) {
        |Type.values().forEach { type ->
            DropdownMenuItem(
                text = { Text(text = type.toString()) },
                onClick = {
                    onTypeSelected(type)
                }
            )
        }
    }
}

@Composable
private fun PokemonList(
    modifier: Modifier = Modifier,
    gridState: LazyGridState,
    isLoading: Boolean = false,
```

> pokdexer > ui > pokdex > PokedexScreen.kt > TypeFilterDropdownMenu

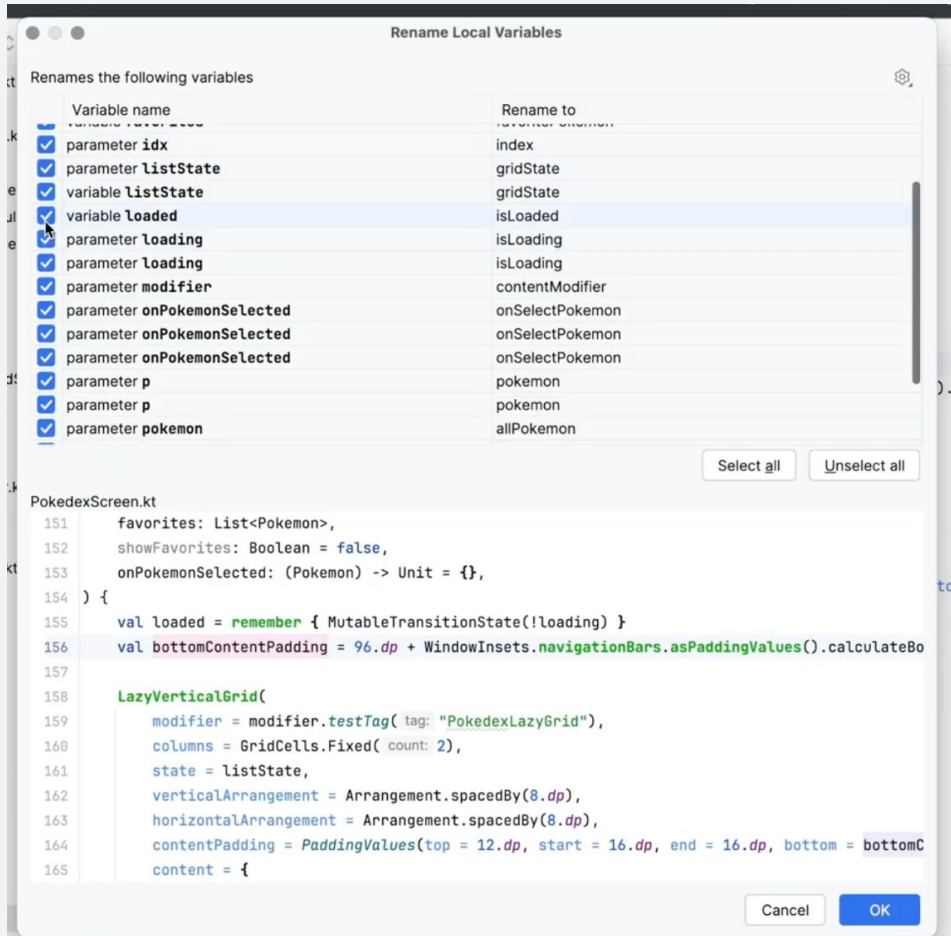
Available today

Rethink variables

Improve readability with this new Refactor operation

Refactor all variables in a file to make them more intuitive and descriptive

- Gemini analyzes whole file context
- Analyses how each variable is used to suggest more intuitive and descriptive names





Android

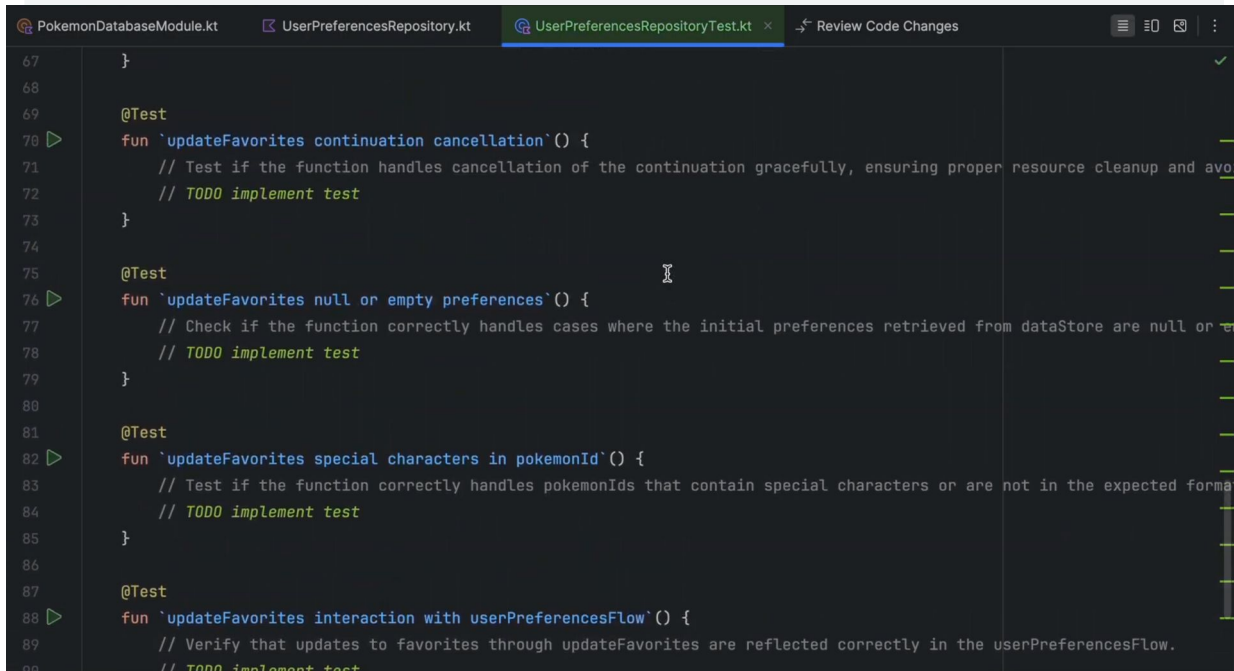
Getting ready for production

Available today

Generate unit test scenarios

Gemini can assist you in generating scenarios for your Unit tests. It analyzes each method to determine common and edge case scenarios for you to test

- Gemini > Unit test scenarios
- Scaffolds each unit test with descriptive name and comments for what each method should test
- You implement the test code
- We're working on how leverage Gemini to implement test code in a reliable way

A screenshot of an IDE window showing a Kotlin unit test file named 'UserPreferencesRepositoryTest.kt'. The file is part of a project with other files like 'PokemonDatabaseModule.kt' and 'UserPreferencesRepository.kt'. The code contains four test methods, each starting with '@Test' and a 'fun' declaration. The first method is 'updateFavorites continuation cancellation', the second is 'updateFavorites null or empty preferences', the third is 'updateFavorites special characters in pokemonId', and the fourth is 'updateFavorites interaction with userPreferencesFlow'. Each method has a comment indicating it's a test and a 'TODO implement test' placeholder. The IDE interface includes a tab bar at the top, a code editor with line numbers on the left, and a right sidebar with icons for navigation and search.

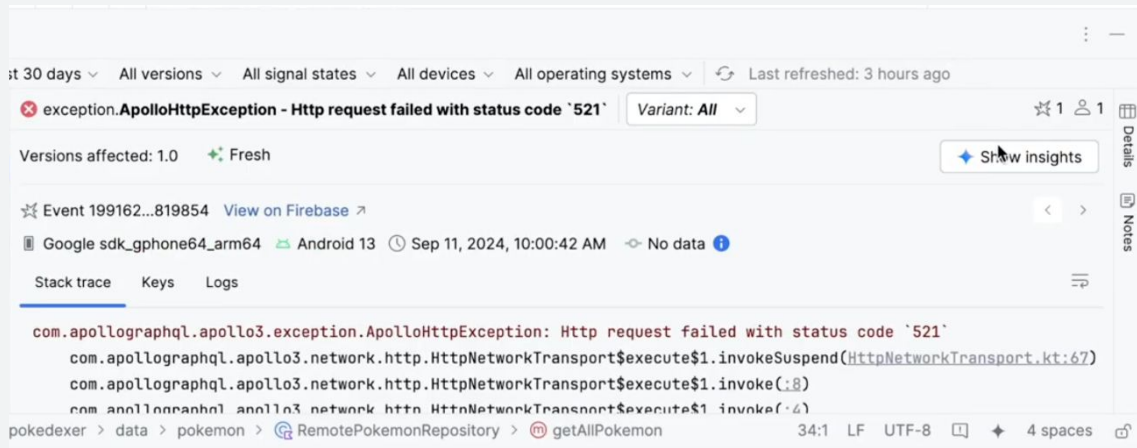
```
67 }  
68  
69 @Test  
70 fun `updateFavorites continuation cancellation`() {  
71     // Test if the function handles cancellation of the continuation gracefully, ensuring proper resource cleanup and ava  
72     // TODO implement test  
73 }  
74  
75 @Test  
76 fun `updateFavorites null or empty preferences`() {  
77     // Check if the function correctly handles cases where the initial preferences retrieved from dataStore are null or  
78     // TODO implement test  
79 }  
80  
81 @Test  
82 fun `updateFavorites special characters in pokemonId`() {  
83     // Test if the function correctly handles pokemonIds that contain special characters or are not in the expected forma  
84     // TODO implement test  
85 }  
86  
87 @Test  
88 fun `updateFavorites interaction with userPreferencesFlow`() {  
89     // Verify that updates to favorites through updateFavorites are reflected correctly in the userPreferencesFlow.  
90     // TODO implement test
```

Available today

Crashlytics and Vitals in Android Studio

You can now use Gemini in the existing App Quality Insights Window.

- Generate insights for each issue report to aid investigation in the root cause and explore suggested fixes
- Generating insights with code context is coming soon.
- Works for both Crashlytics and Android Vitals reports



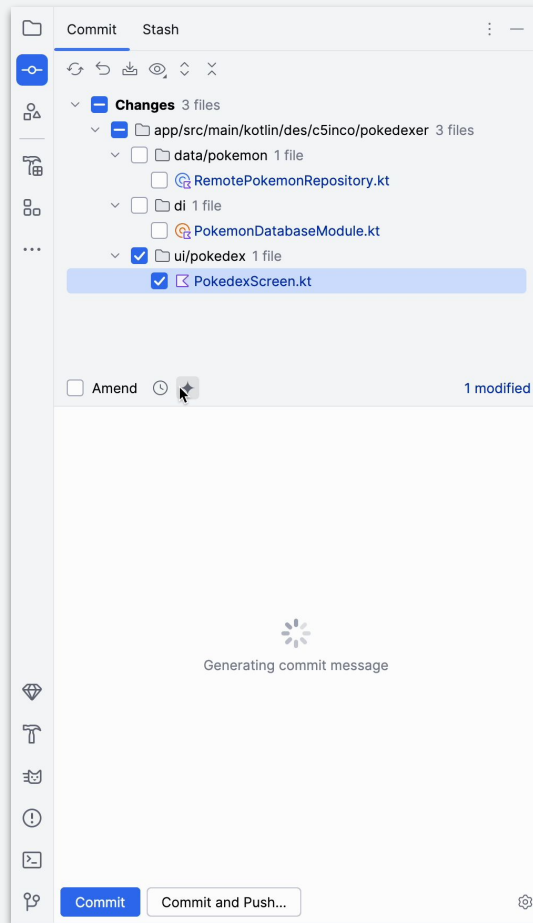
Available today

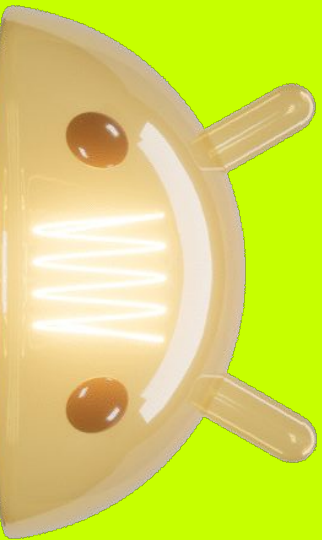
Commit message

Making it easier to maintain a healthy git history

Being descriptive in your commit messages not only helps you keep track of your work, but also help your teammates. But capturing all the details of your changes in a readable way is time consuming.

- Quick action in the Commit window leverages Gemini to analyze code changes
- Gemini responds with descriptive text to capture the nature if your changes, all in just one click.





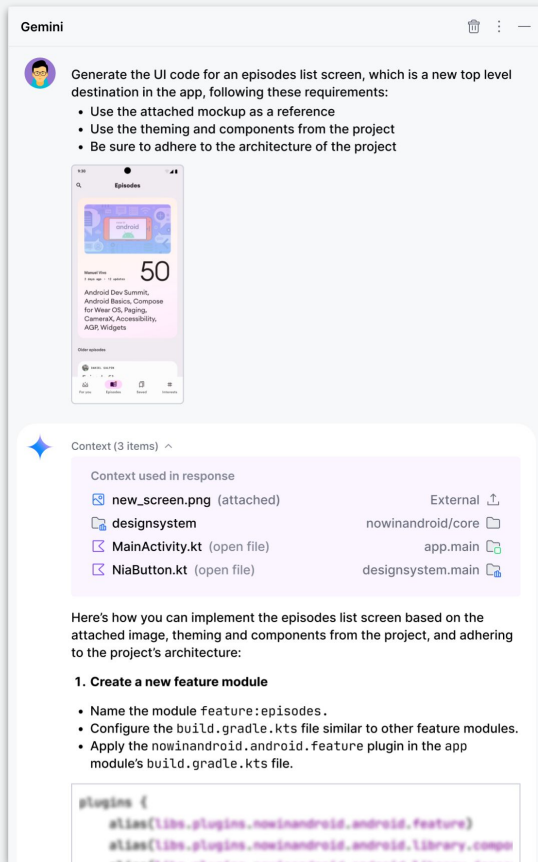
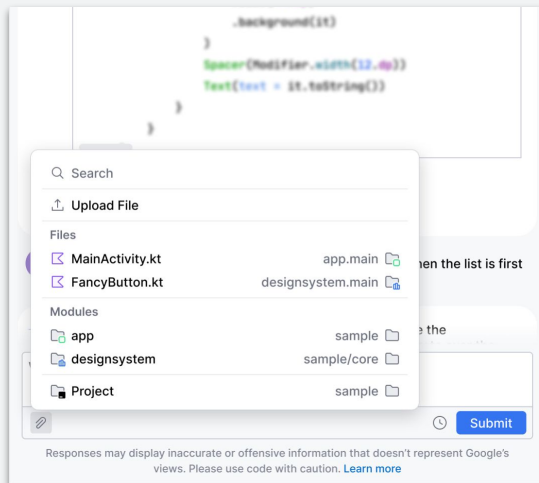
Android

**And more to
come...**

Attaching more context

Richer inputs, richer outputs

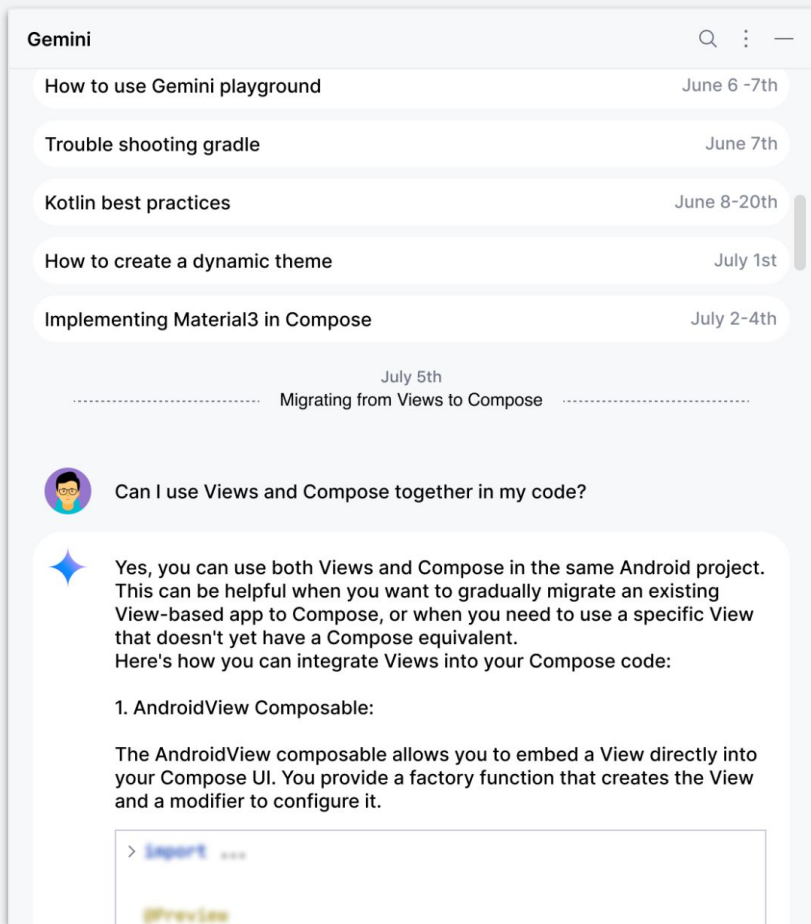
- We're working on the ability to have more control and flexibility with the context you provide Gemini
- See what files are being shared with Gemini, with the option to add more code files, or entire modules.
- Support for non-text inputs, such as screenshots.
- This taps into Gemini's multi-modal support for features, such as generating code from images



Multiple chat sessions

Separate and organize multiple conversations with Gemini

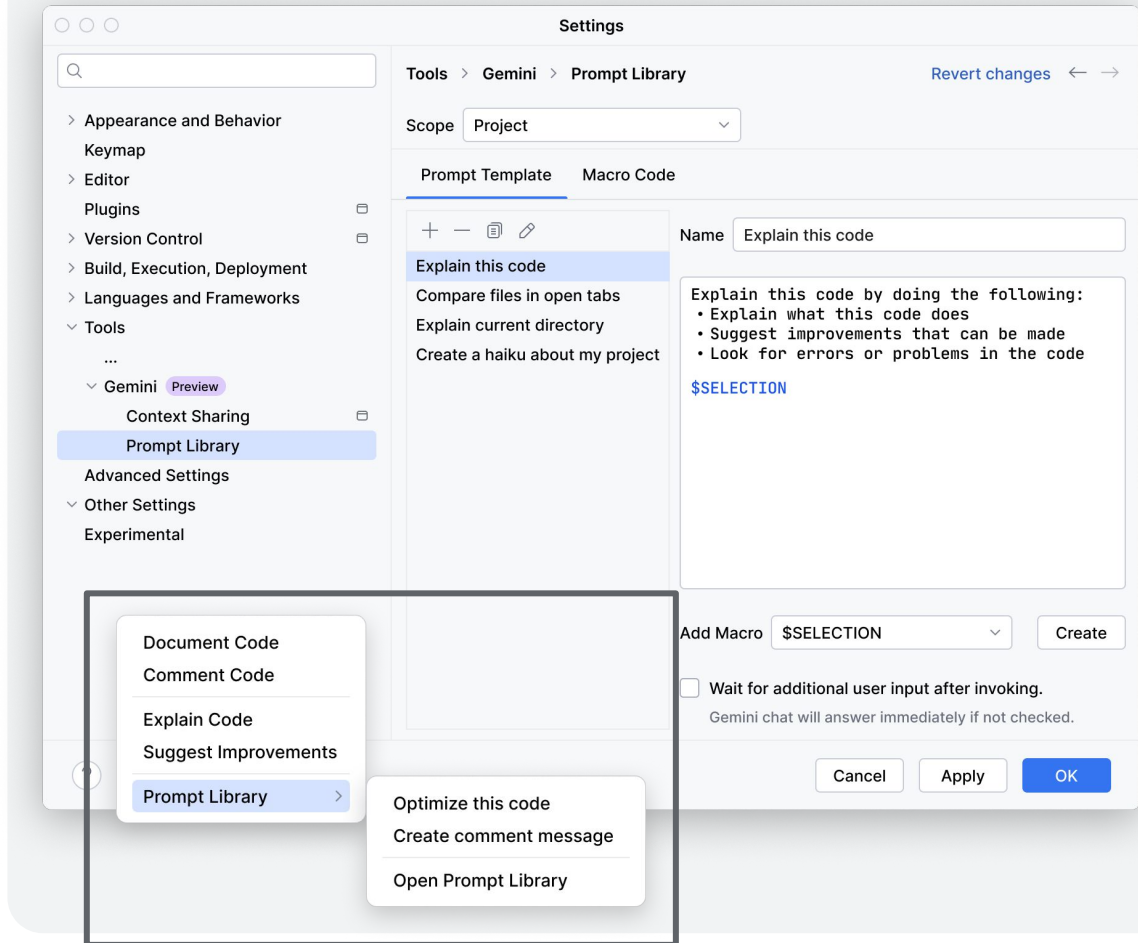
- Gemini in Android Studio is meant to be conversational and assistive in helping you explore ideas.
- You're likely to have more than one problem or idea you're exploring at any given time
- With multiple chat sessions, you can engage with Gemini across multiple conversations
- Each conversation has separate context and history



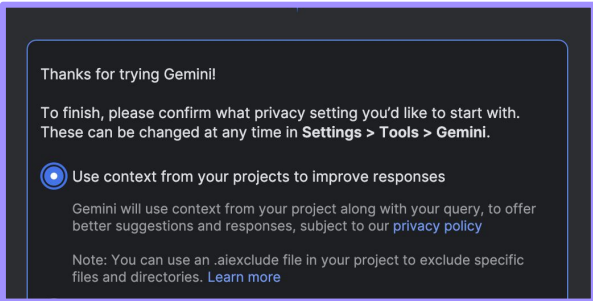
Prompt library

Ability to add and customize prompts

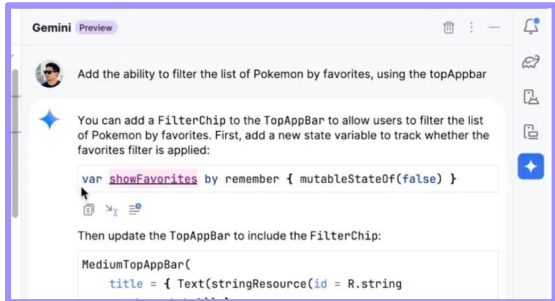
- Getting great responses often relies on having a well-tuned prompt
- As you explore using Gemini, we expect users will have certain prompts for adding new code in a certain style, or asking questions with specific context frequently.
- We're exploring including a prompt library that allows you to define prompts that pre-authored with specific requirements
- You can easily invoke those prompts with some current context



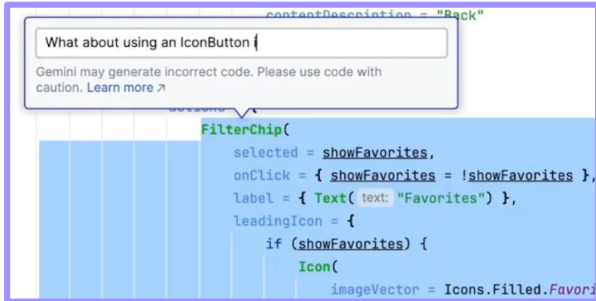
What we've covered today



Transparency and user-control



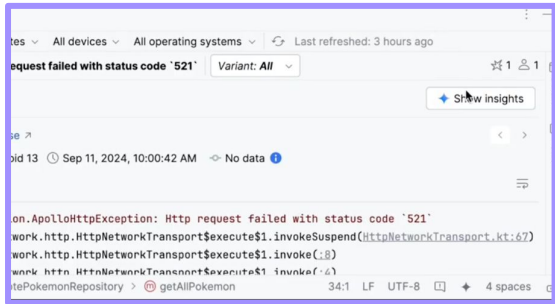
Have a conversation with Gemini



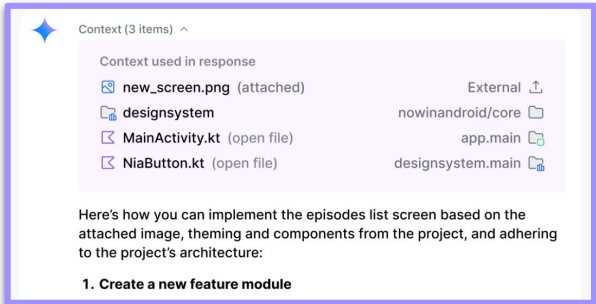
Prompt Gemini directly from the editor



Code completion that anticipates your needs



Deep insights into crash reports



More to come...

Thank you!



Adarsh Fernando

Senior Product Manager
Android Studio



@adarshfernando



Chris Sinco

UX Design Lead
Android Studio



@csinco