

M.ADI 성과보고 영상 자료

이재학 (AVNT설계1팀)

Exported on 10/14/2024

Table of Contents

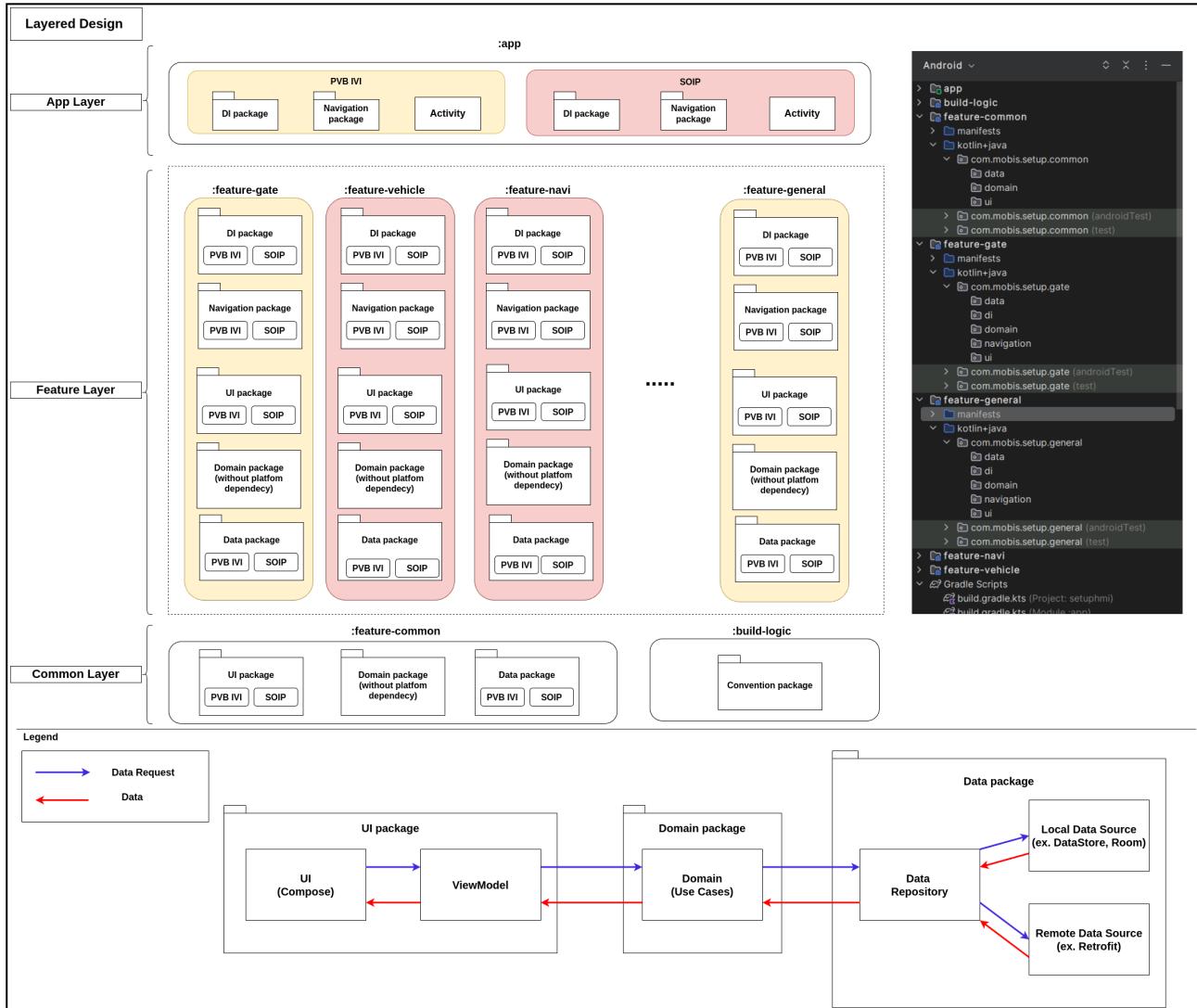
1	1. 표준화	3
1.1	1.1 AppTemplate.....	3
1.1.1	1.1.1 AppTemplate Architecture Design	4
1.1.2	1.1.2 AppTemplate Code.....	7
1.2	1.2 Project Wizard.....	9
2	2. 공용화	16
2.1	2.1 App 플러그인.....	16
2.1.1	2.1.1 Language Translator	16
2.1.2	2.1.2 Publish.....	18
2.1.3	2.1.3 Ktlint (Kotlin Lint).....	19
2.1.4	2.1.4 Build Extension	22
2.2	2.2 App 라이브러리	24
2.2.1	2.2.1 GUI Component	24
2.2.2	2.2.2 App Framework	30
2.3	2.3 M.ADI Maven Repository.....	32
3	3. 자동화	34
3.1	3.1 CI/CD	34
3.2	3.2 AI Code Reviewer	36

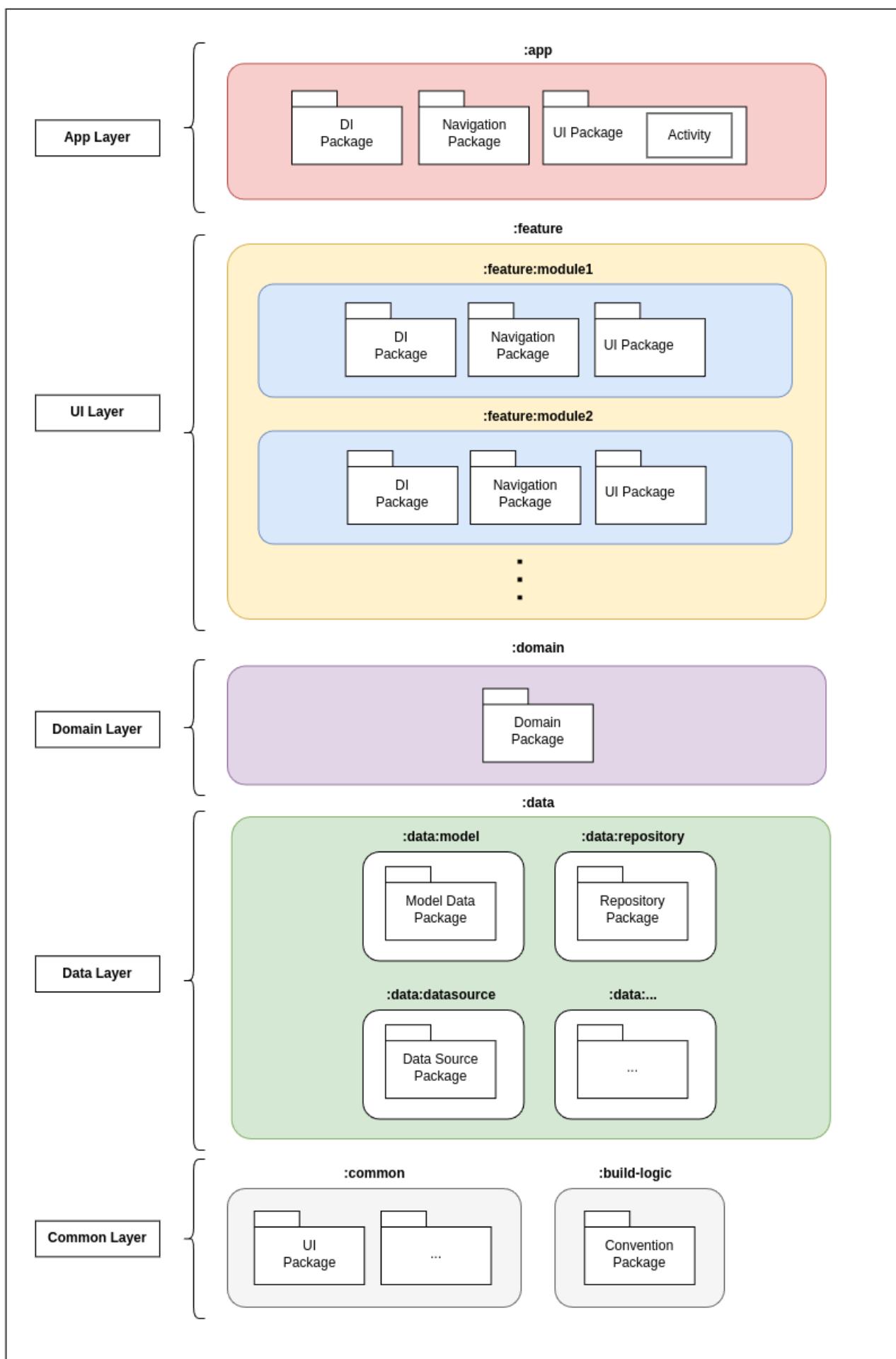
1 1. 표준화

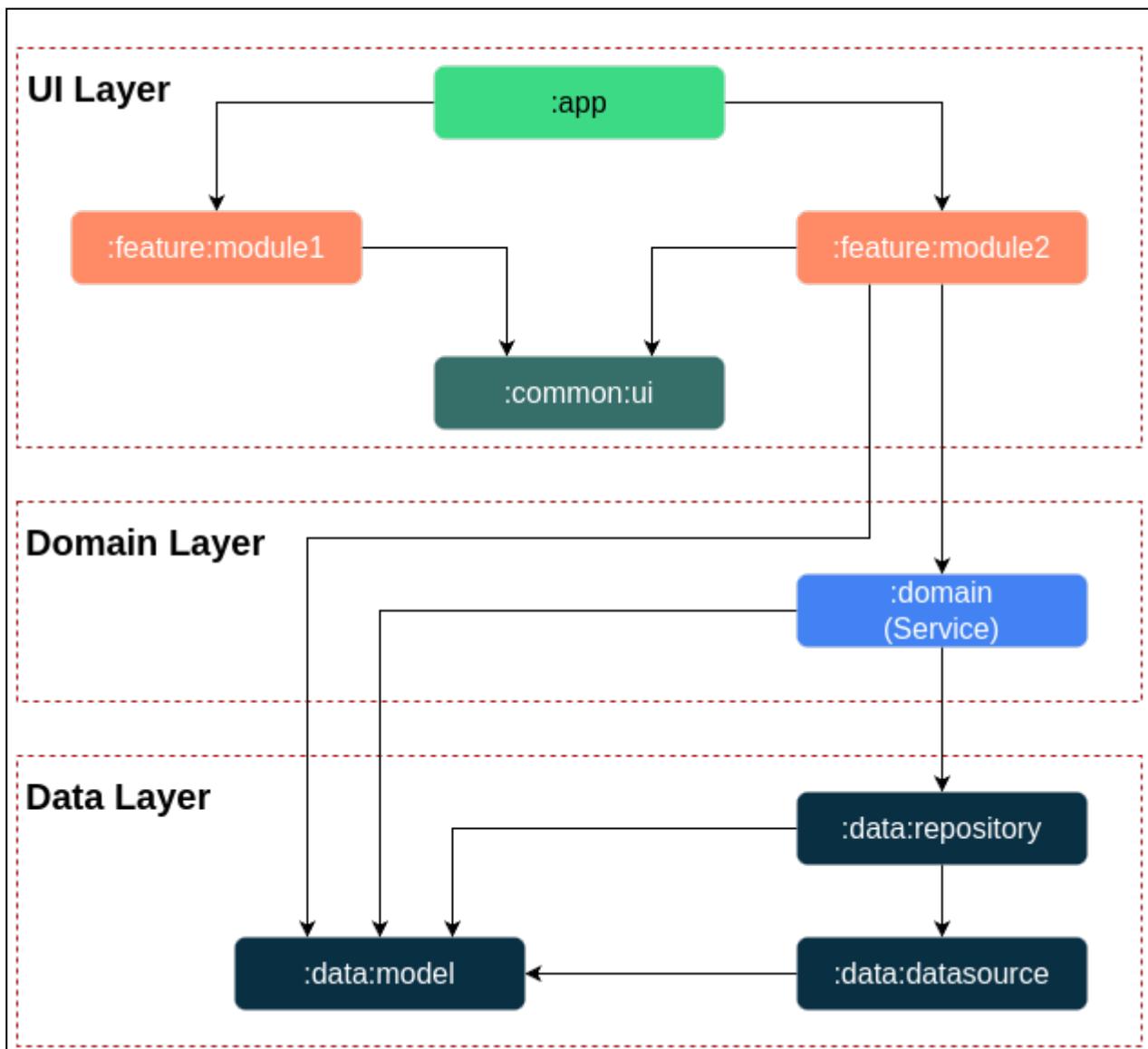
1.1 1.1 AppTemplate

- AppTemplate이란?
 - Application의 표준화된 Architecture를 실행 가능한 Code Level로 제공함으로써 Application의 Code Baseline을 제공함으로써 표준화 목표를 달성
- GAT(Generic AppTemplate), MAT(Media AppTemplate), SAT(Setting AppTemplate)의 3가지 Type의 AppTemplate을 제공
 - GAT는 일반적인 Application을 위한 Template 제공
 - MAT는 Media 특화 Application을 위한 Template 제공
 - SAT는 설정 특화 Application을 위한 Template 제공

1.1.1 AppTemplate Architecture Design







1.1.2 1.1.2 AppTemplate Code

1. GAT

Source	Description	Size	Last Modified
.editorconfig	ENH: Apply ktlint, and raise AGP version	852 B	28 Mar 2024
.gitignore	ENH: Apply the Kotlin 2.0	116 B	27 May 2024
build.gradle.kts	ENH: Apply the BuildExt plugin 1.0.0	498 B	01 Jun 2024
commit_template.txt	ENH: Add commit_template.txt	620 B	08 Dec 2023
gradle.properties	ENH: Configure product flavor, update strings.xml and create Android.bp	1.92 KB	11 Jul 2024
gradlew	ENH: Add basic module structure	5.63 KB	15 Nov 2023
gradlew.bat	ENH: Add basic module structure	2.70 KB	15 Nov 2023
README.md	DOC: Remove a comment about gat branch	199 B	01 Mar 2024
settings.gradle.kts	ENH: Use credentials to access M.ADI Maven repository in GAT	1.34 KB	25 Jul 2024

```

1 package adi.template.generic.app
2
3 > import ...
4
5 @AndroidEntryPoint
6 class MainActivity : ComponentActivity() {
7     override fun onCreate(savedInstanceState: Bundle?) {
8         super.onCreate(savedInstanceState)
9         setContent {
10             Theme(dynamicColor = false) {
11                 Surface(
12                     modifier = Modifier.fillMaxSize(),
13                     color = MaterialTheme.colorScheme.background,
14                 ) {
15                     App()
16                 }
17             }
18         }
19     }
20 }
21
22
23
24
25
26
27
28
29
30

```

2. MAT

The screenshot shows the MAT interface with the repository 'M.ADI / adi.template.media' selected. The left sidebar lists the project structure under 'Source'. The main area displays a table of files with their descriptions, sizes, and last modified dates.

File	Description	Size	Last Modified
.editorconfig	ENH: Apply official ktlint plugin	852 B	27 Feb 2024
.gitignore	ENH: Apply the Kotlin 2.0	116 B	27 May 2024
build.gradle.kts	ENH: Apply the BuildExt plugin 1.0.0	498 B	01 Jun 2024
commit_template.txt	DOC: Initial Commit	620 B	10 Jan 2024
gradle.properties	ENH: Configure product flavor, update strings.xml and create Android.bp	1.92 KB	11 Jul 2024
gradlew	ENH: Initial code	5.63 KB	10 Jan 2024
gradlew.bat	ENH: Initial code	2.70 KB	10 Jan 2024
README.md	DOC: Update README.md	2.18 KB	12 Apr 2024
settings.gradle.kts	ENH: Use credentials to access M.ADI Maven repository in MAT	1.43 KB	25 Jul 2024

The screenshot shows the Android Studio IDE with the project 'adi.template.media' open. The left sidebar shows the project structure. The right pane displays the code for 'MediaListScreen.kt'.

```

package adi.template.media.feature.medialist

import ...
import ...

@Composable
fun MediaListScreen(viewModel: MediaListViewModel = hiltViewModel()) {
    val mediaItems by viewModel.mediaItems.collectAsStateWithLifecycle()
    val currentItem by viewModel.currentItem.collectAsStateWithLifecycle()
    val uiState = viewModel.mediaListUiState.collectAsStateWithLifecycle()

    Box(
        modifier =
            Modifier
                .fillMaxSize()
                .background(MaterialTheme.colorScheme.background),
    ) {
        when (uiState.value) {
            is MediaListUiState.Ready ->
                Column(
                    modifier =
                        Modifier
                            .padding(start = 30.dp, end = 30.dp),
                ) {
                    BackButton(onClick = {
                        viewModel.displayPrevious()
                    })
                    LazyColumn { this: LazyListScope }
                        items(mediaItems) { this: LazyItemScope | item ->
...

```

3. SAT

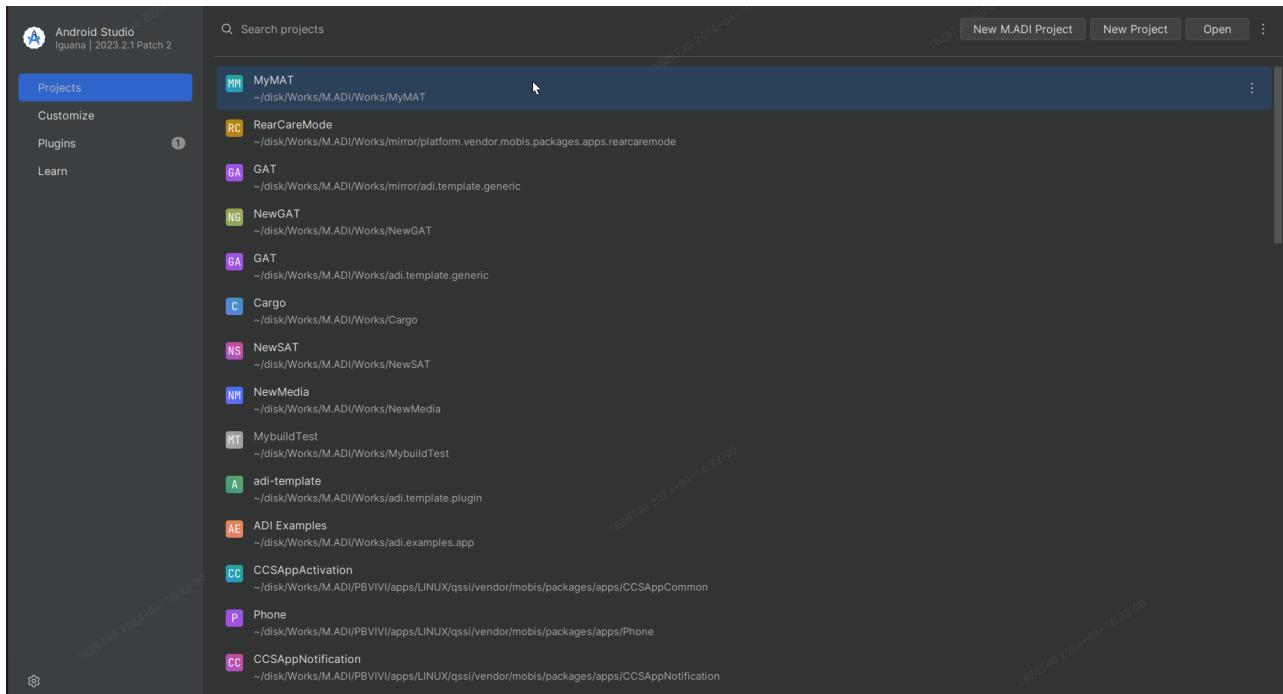
The screenshot shows the Android Studio interface with the following details:

- Top Bar:** Shows the project name "M.ADI / adi.template.settings" and the branch "master".
- Source Control:** Displays a list of files in the "adi.template.settings" repository, including .editorconfig, .gitignore, build.gradle.kts, commit_template.txt, gradle.properties, gradlew, gradlew.bat, README.md, and settings.gradle.kts. Each file has a brief description and its last modified date.
- Code Editor:** The main editor window displays the code for "SettingsPageProviderRepository.kt". The code is annotated with Korean comments explaining the logic of initializing page providers and handling root pages.
- Project Structure:** On the left, the "Android" view shows the project's module structure under "core". The "framework" module contains "common" and "compose" sub-modules. The "common" module includes classes like EntryMacro, EntrySearchData, EntrySliceData, EntryStatusData, PageModel, SettingsEntry, SettingsEntryBuilder, SettingsEntryGroup, SettingsEntryGroupBuilder, SettingsEntryRepository, SpaEnvironment, and SpaLogger.

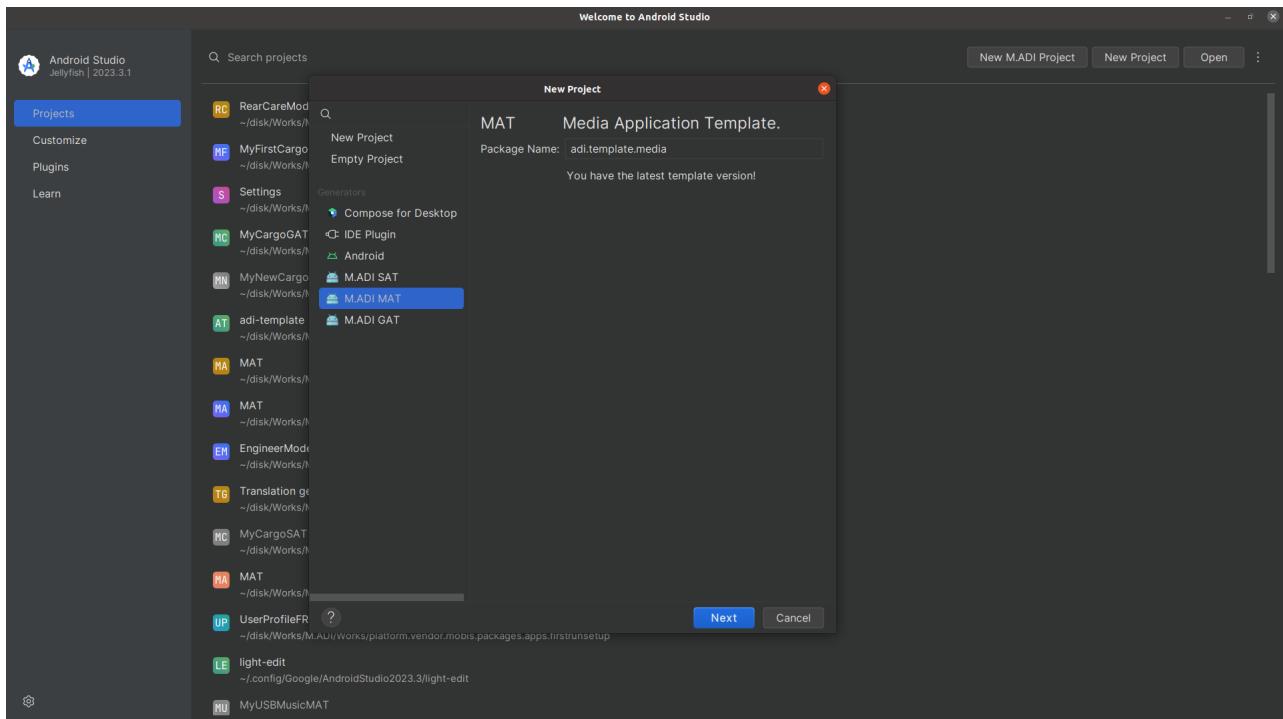
1.2 1.2 Project Wizard

- Android Studio IDE에서 몇번의 Click을 통해서 자동으로 실행 가능한 AppTemplate Code를 생성

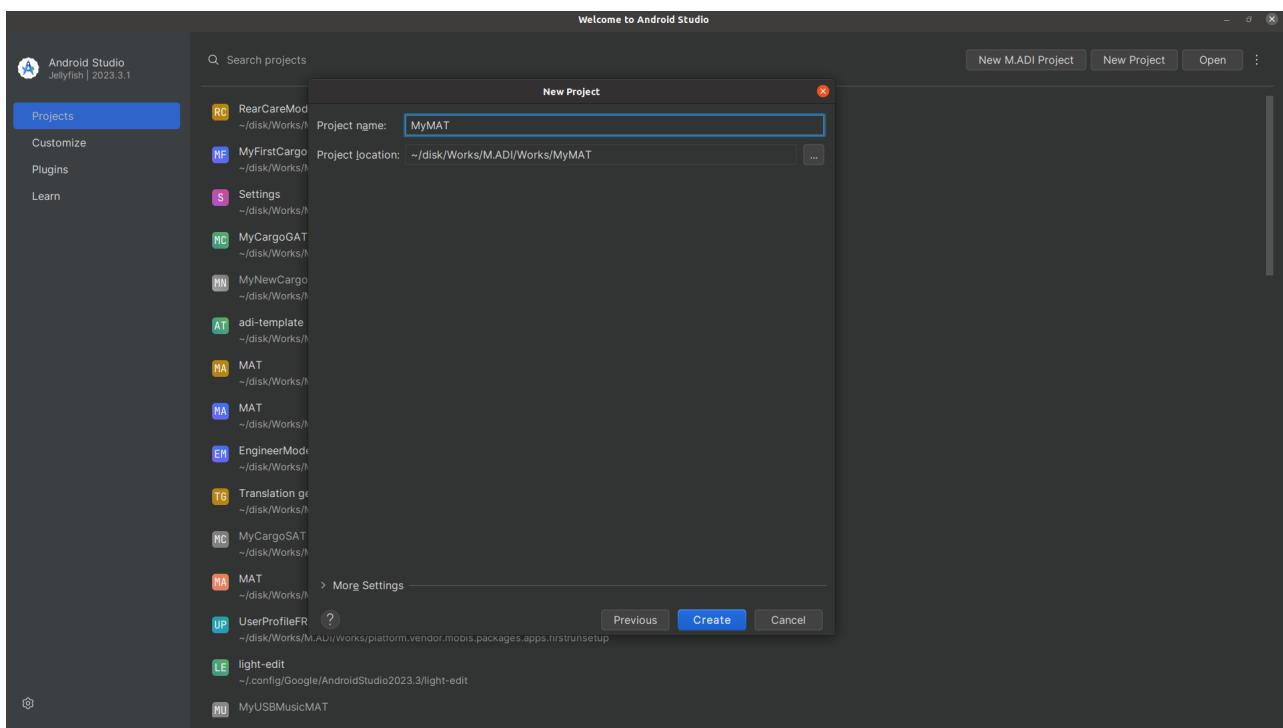
1. Android Studio IDE에서 화면 우측 상단의 "New M.ADI Project Button" 클릭



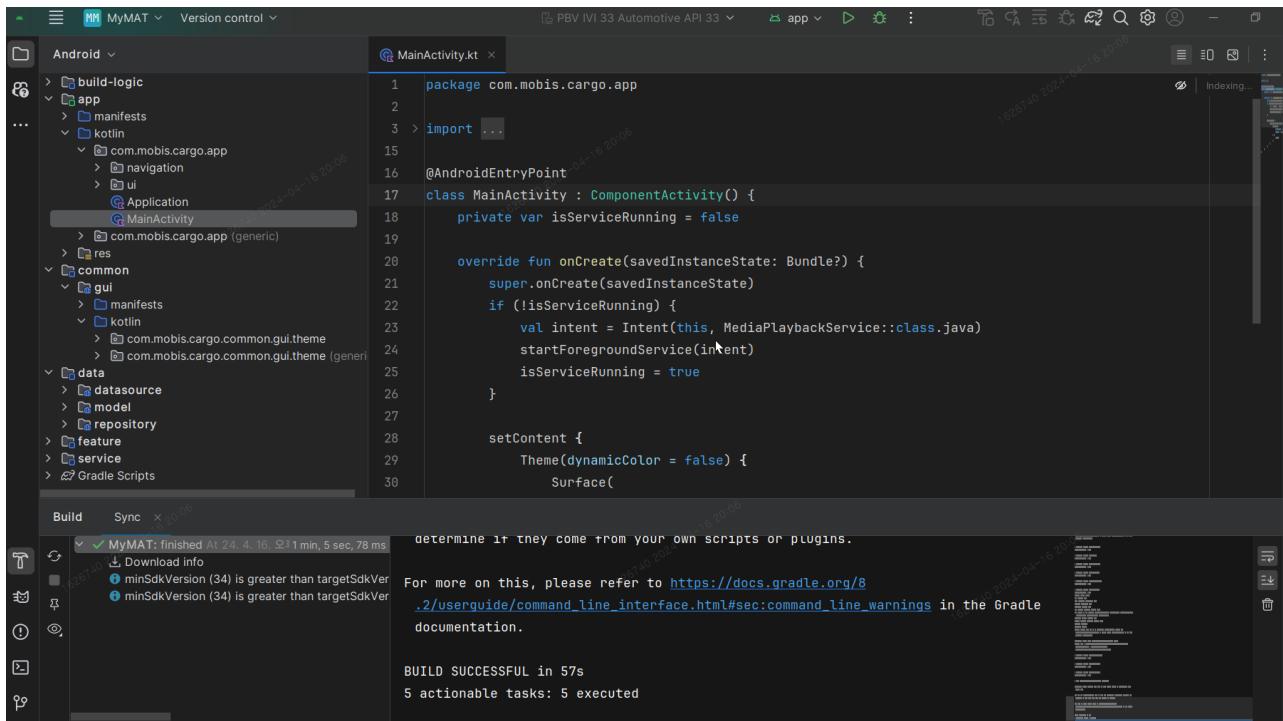
2. M.ADI GAT(Generic AppTemplate), MAT(Media AppTemplate), SAT(Setting AppTemplate) 중에서 원하는 AppTemplate을 선택 및 **개발하고자 하는 Application의 Package Name**을 입력



3. **개발하고자 하는 Application의 Project Name**을 설정 이후 Create 버튼 클릭

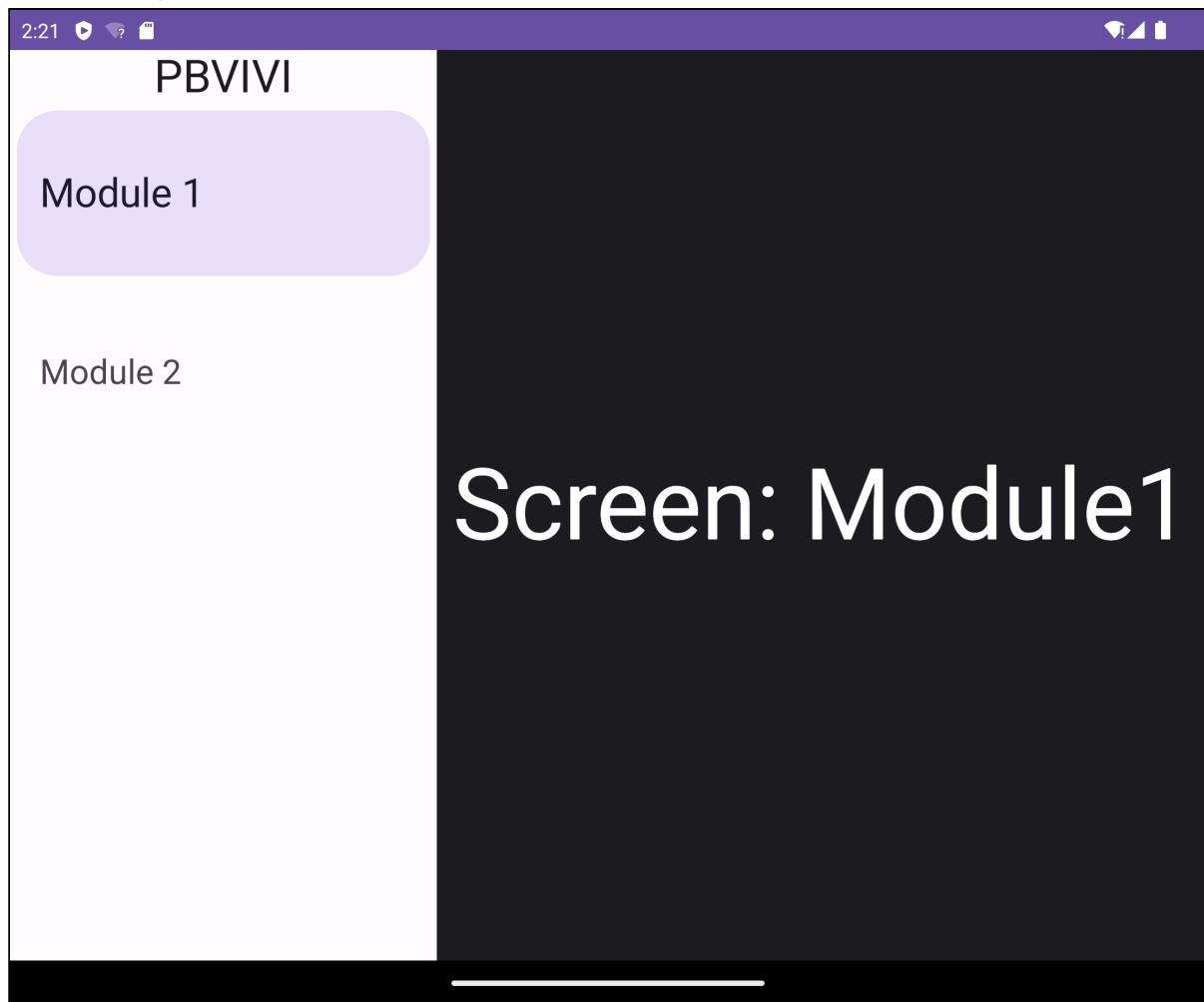


4. 개발자가 입력한 Project Name 및 Package Name으로 Application의 Base Line Code들에 대해서 자동 생성

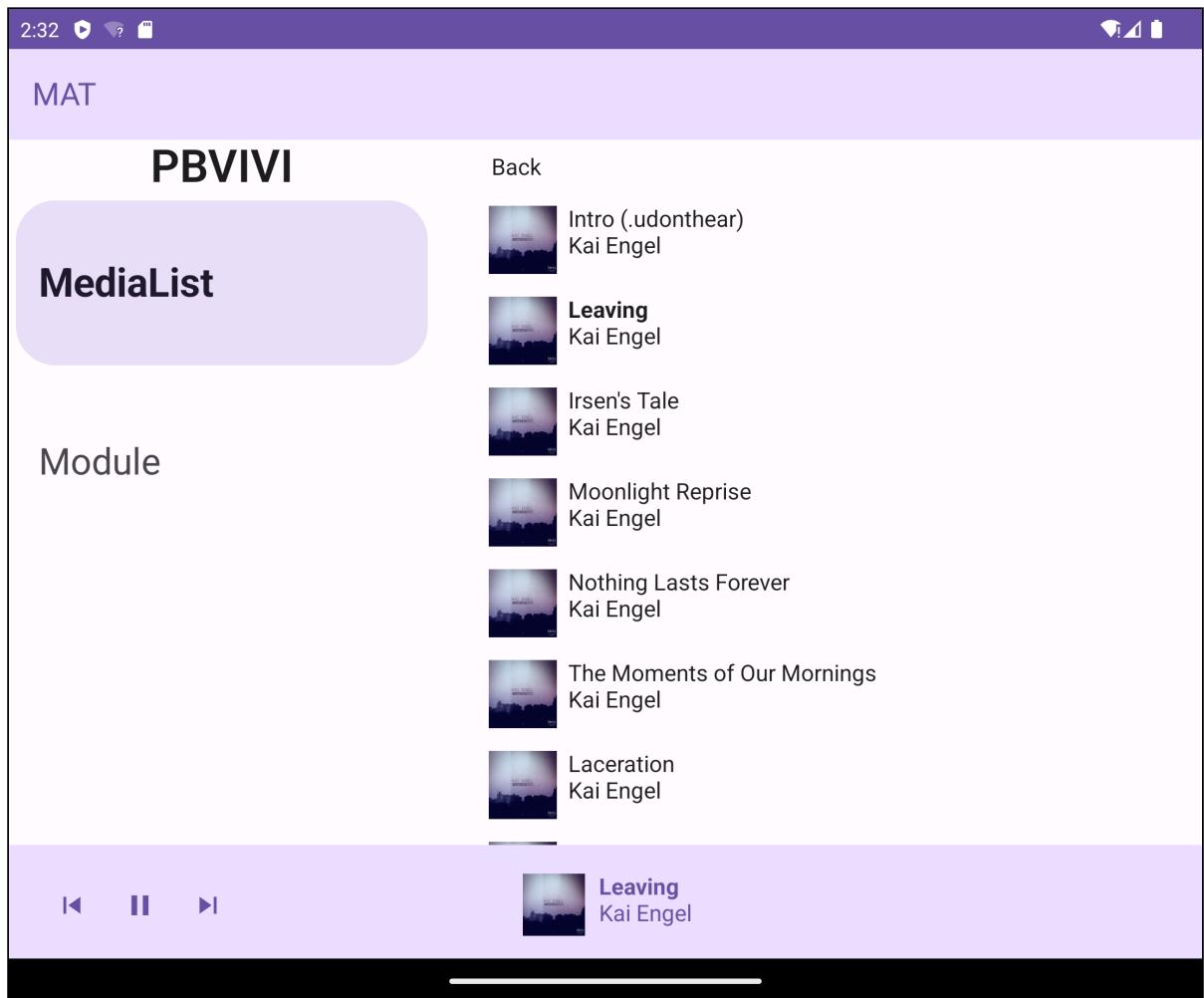


5. Application의 Base Code를 실행 하면 하기와 같이 기본 기능이 갖춰진 Application이 실행, 이 이후부터 각 Application의 특화 기능을 구현하여 제품화

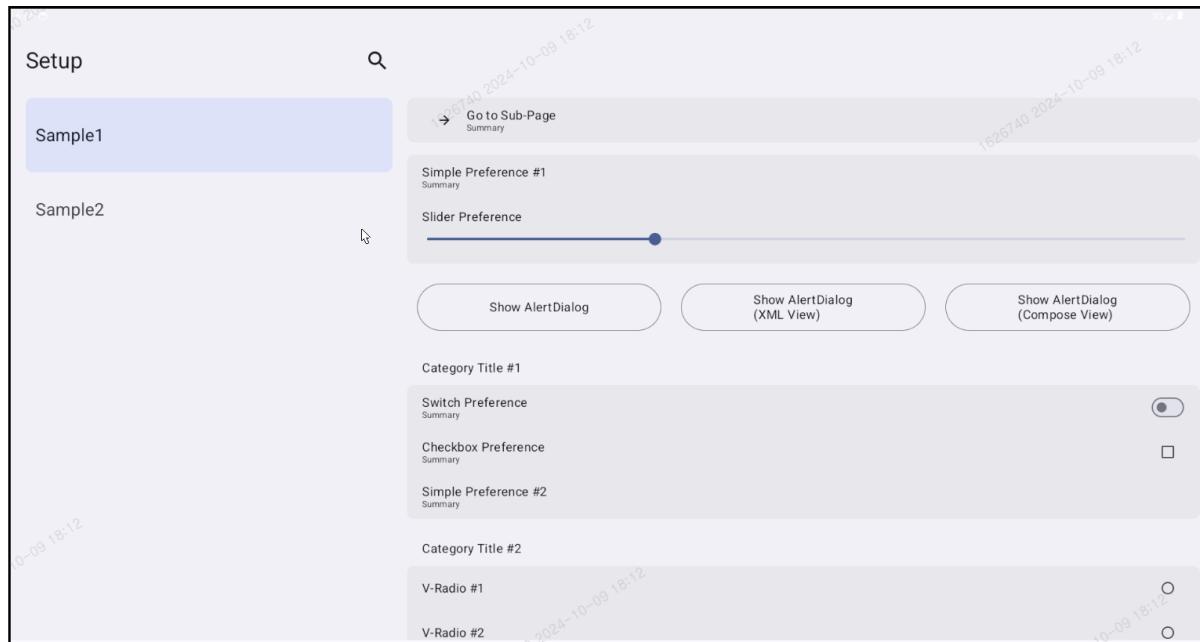
- GAT 실행 결과

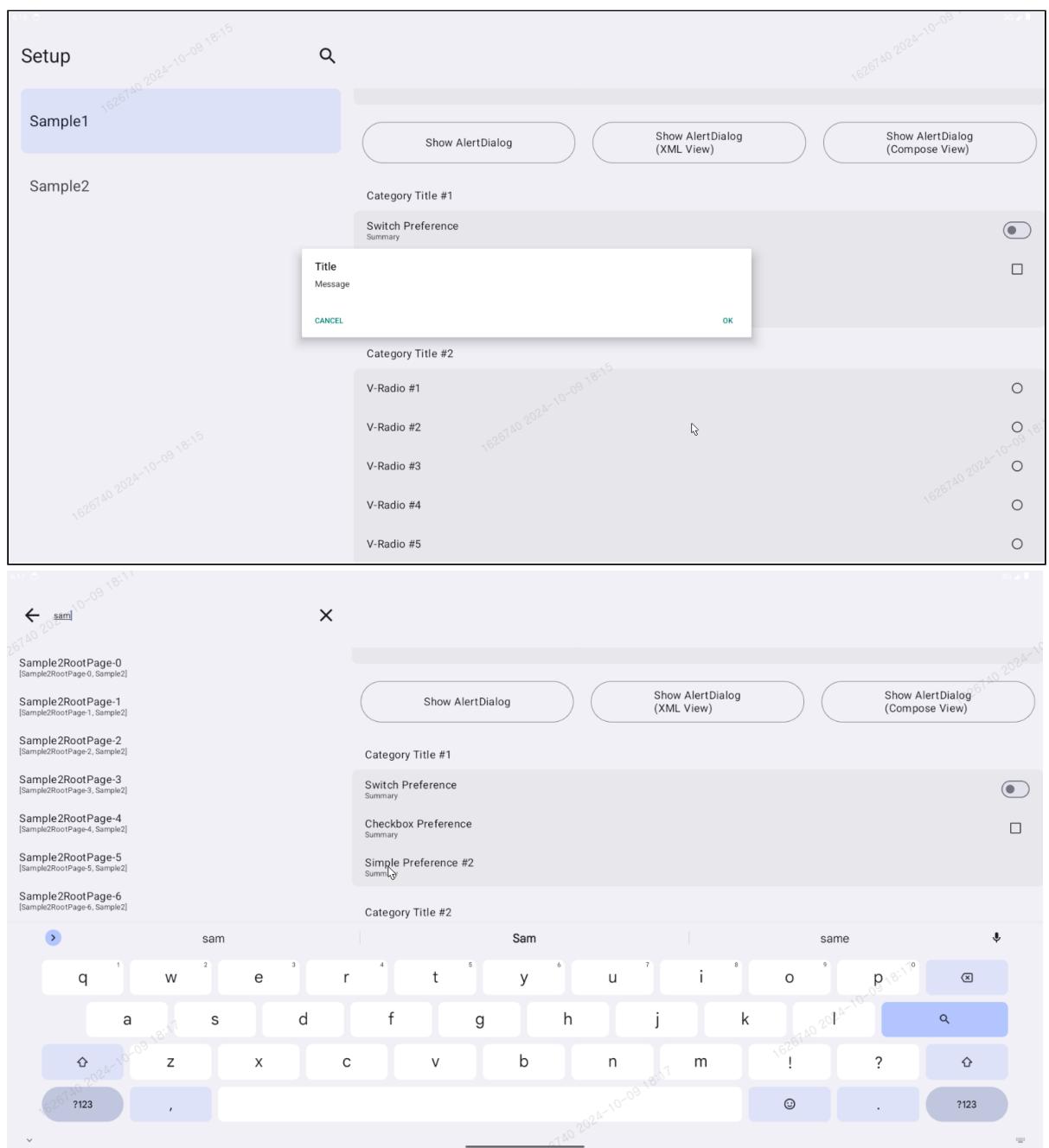


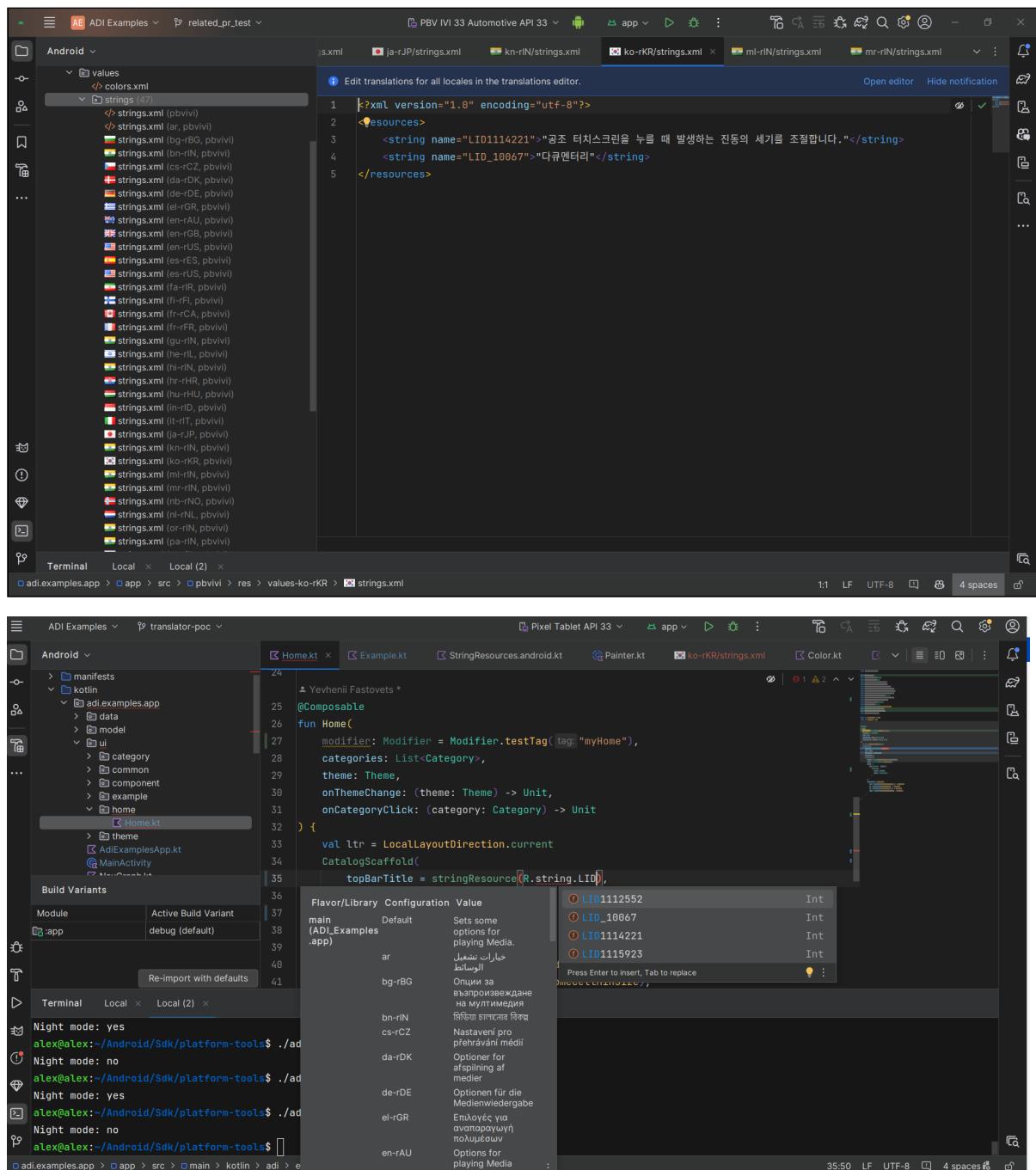
- MAT 실행 결과



▪ SAT 실행 결과







2.1.2 Publish

- 멀티 플랫폼 자동 configuration을 위한 App 플러그인
- PBVIVI 제품을 위한 Product Flavor 제공

```

CSTD API 34
app
Tasks.kt
if (addFlavors) {
    if (!android.flavorDimensionList.contains(FLAVOR_DIMENSION))
        android.flavorDimensionList.add(FLAVOR_DIMENSION)
}

android.productFlavors { this: NamedDomainObjectContainer<LibraryProductFlavor> -
    ALL_FLAVORS.forEach { flavor ->
        if (findByName(flavor) == null) {
            create(flavor) { it: LibraryProductFlavor -
                dimension = FLAVOR_DIMENSION
                minSdk = ALL_FLAVORS_SDK_VERSION
            }
        }
    }

    getByName(flavor).isDefault = (flavor == DEFAULT_FLAVOR_NAME)
}

if (!skipPublishing) {
    android.publishing { this: LibraryPublishing -
        multipleVariants { this: MultipleVariants -
            includeBuildTypeValues(RELEASE_VARIANT)
            includeFlavorDimensionAndValues(
                FLAVOR_DIMENSION,
                *PUBLISH_FLAVORS.toTypedArray()
            )
            withSourcesJar()
        }
    }
}

```

2. Connect 제품을 위한 Product Flavor 제공

```

CSTD API 34
app
Tasks.kt
if (addFlavors) {
    if (!android.flavorDimensionList.contains(FLAVOR_DIMENSION))
        android.flavorDimensionList.add(FLAVOR_DIMENSION)
}

android.productFlavors { this: NamedDomainObjectContainer<LibraryProductFlavor> -
    ALL_FLAVORS.forEach { flavor ->
        if (findByName(flavor) == null) {
            create(flavor) { it: LibraryProductFlavor -
                dimension = FLAVOR_DIMENSION
                minSdk = ALL_FLAVORS_SDK_VERSION
            }
        }
    }

    getByName(flavor).isDefault = (flavor == DEFAULT_FLAVOR_NAME)
}

if (!skipPublishing) {
    android.publishing { this: LibraryPublishing -
        multipleVariants { this: MultipleVariants -
            includeBuildTypeValues(RELEASE_VARIANT)
            includeFlavorDimensionAndValues(
                FLAVOR_DIMENSION,
                *PUBLISH_FLAVORS.toTypedArray()
            )
            withSourcesJar()
        }
    }
}

```

2.1.3 Ktlint (Kotlin Lint)

- Kotlin으로 작성된 code의 Coding Convention에 대한 Auto Code Formatting App 플러그인

❤️ Klint ❌ Disallowed

```
// A single annotation (without parameters) is allowed on same line as annotated code
@FunctionalInterface class FooBar {
    @JvmField var foo: String

    @Test fun bar() {}
}

// A class or function parameter may have a single annotation with parameter(s) on the same line
class Foo(
    @Path("fooId") val fooId: String,
    @NotNull("bar") bar: String,
)

// Multiple annotations (without parameters) are allowed on the same line
@Foo @Bar
class FooBar {
    @Foo @Bar
    var foo: String

    @Foo @Bar
    fun bar() {}
}

// An array of annotations (without parameters) is allowed on same line as annotated code
@[Foo Bar] class FooBar2 {
    @[Foo Bar] var foo: String

    @[Foo Bar] fun bar() {}
}
```

Rule-id: annotation (standard rule set)

❤️ Klint ❌ Disallowed

```
const val FOO_1 = "foo1"

class FooBar {
    val foo2 = "foo2"
    val foo3 = "foo3"

    fun bar1() {
        val foo4 = "foo4"
        val foo5 = "foo5"
    }

    fun bar2() = "bar"

    val foo6 = "foo3"
    val foo7 = "foo4"

    enum class Foo
}
```

Rule id: blank-line-before-declaration (standard rule set)

❤️ Klint ❌ Disallowed

```
/*
 * This comment is formatted well.
 */
```

Rule id: block-comment-initial-star-alignment (standard rule set)

Klint Disallowed

```
/*
 * This comment is formatted well.
 */
```

Rule id: `block-comment-initial-star-alignment` (standard rule set)

Klint Disallowed

```
fun foo(value: int) {
    if (value > 0) {
        doSomething()
    } else if (value < 0) {
        doSomethingElse()
    } else {
        doSomethingElse2()
    }
}
```

Rule id: `if-else-bracing` (standard rule set)

2.1.4 2.1.4 Build Extension

- Build 오류 검출 항목을 확장하기 위한 App 플러그인

1. Multi Module의 Android SDK Version의존 관계 자동 검출

```
31 issues were found when checking AAR metadata:
1. Dependency 'androidx.compose.material:material-ripple-android:1.6.0' requires libraries and applications that depend on it to compile against version 34 or later of the Android APIs.

:app is currently compiled against android-33.

Recommended action: Update this project to use a newer compileSdk of at least 34, for example 34.

Note that updating a library or application's compileSdk (which allows newer APIs to be used) can be done separately from updating targetSdk (which opts the app in to new runtime behavior) and minSdk (which determines which devices the app can be installed on).

2. Dependency 'androidx.compose.material:material-icons-core-android:1.6.0' requires libraries and applications that depend on it to compile against version 34 or later of the Android APIs.

:app is currently compiled against android-33.

Recommended action: Update this project to use a newer compileSdk of at least 34, for example 34.

Note that updating a library or application's compileSdk (which allows newer APIs to be used) can be done separately from updating targetSdk (which opts the app in to new runtime behavior) and minSdk (which determines which devices the app can be installed on).
```

```

3. Dependency 'androidx.compose.material:material-android:1.6.0' requires libraries and applications that
depend on it to compile against version 34 or later of the
Android APIs.

:app is currently compiled against android-33.

Recommended action: Update this project to use a newer compileSdk
of at least 34, for example 34.

Note that updating a library or application's compileSdk (which
allows newer APIs to be used) can be done separately from updating
targetSdk (which opts the app in to new runtime behavior) and
minSdk (which determines which devices the app can be installed
on).

4. Dependency 'androidx.compose.animation:animation-core-android:1.6.0' requires libraries and applications that
depend on it to compile against version 34 or later of the
Android APIs.

:app is currently compiled against android-33.

Recommended action: Update this project to use a newer compileSdk
of at least 34, for example 34.

Note that updating a library or application's compileSdk (which
allows newer APIs to be used) can be done separately from updating
targetSdk (which opts the app in to new runtime behavior) and
minSdk (which determines which devices the app can be installed
on).

5. Dependency 'androidx.compose.animation:animation-android:1.6.0' requires libraries and applications that
depend on it to compile against version 34 or later of the
Android APIs.

```

2. Android Lint 이슈에 대해서 Build 시점에 검출

The screenshot shows the 'Lint Report: 102 warnings' dialog in Android Studio. The report was performed on Thu Oct 03 09:54:43 KST 2024 by AGP (8.3.2). The overview section displays the following counts:

- Lint:** 1 warning (ObsoleteLintCustomCheck: Obsolete custom lint check)
- Correctness:** 9 warnings (AndroidGradlePluginVersion: Obsolete Android Gradle Plugin Version, GradleDependency: Obsolete Gradle Dependency)
- Performance:** 1 warning (ObsoleteSdkInt: Obsolete SDK_INT Version Check), 7 unused resources (UnusedResources: Unused resources)
- Included Additional Checks:** 68 additional checks
- Disabled Checks:** 41 disabled checks

A 'DISMISS' button is located at the bottom right of the dialog.

Lint Report: 102 warnings

Obsolete custom lint check

[/ / / / / gradle/caches/transforms-30776c0233c48126016e712e625741a/transformed/ui-release/ars/lint.jar](#): Library lint checks reference invalid APIs: these checks will be skipped!

Lint found an issue registry (`androidx.compose.ui.lint.UiIssueRegistry`) which contains some references to invalid APIs: `com.android.tools.lint.detector.api.UastLintListK$boolean isIncorrectImplicitReturningLambda(org.jetbrains.uast.UElement)` (`Referenced from androidx/compose/ui/lint/ReturnFromAwaitPointerEventScopeDetector.class`)

Therefore, this lint check library is not included in analysis. This affects the following lint checks:

- UnnecessaryClassifier
- ModifierForExtensionFunction
- ModifierFactoryReturnType
- ModifierFactoryUnreferencedReceiver
- ModifierNodeInspectableProperties
- ModifierParameter
- MultipleWaitPointerEventScopes
- ReturnFromAwaitPointerEventScope
- SuspiciousCompositionLocalModifierRead
- SuspiciousModifierThen

To use this lint check, upgrade to a more recent version of the library.

Lint can be extended with "custom checks": additional checks implemented by developers and libraries to for example enforce specific API usages required by a library or a company coding style guideline.

The Lint APIs are not yet stable, so these checks may either cause a performance degradation, or stop working, or provide wrong results.

This warning flags custom lint checks that are found to be using obsolete APIs and will need to be updated to run in the current lint environment.

It may also flag issues found to be using a newer version of the API, meaning that you need to use a newer version of lint (or Android Studio or Gradle plugin etc) to work with these checks.

To suppress this error, use the issue id "ObsoleteCustomCheck" as explained in the [Suppressing Warnings and Errors](#) section.

[ObsoleteCustomCheck](#) [Warning](#) [Priority 10/10](#)

[DISMISS](#)

Lint Report: 102 warnings

Obsolete Gradle Dependency

[/ / / gradlelibs/versions.toml](#): 2: A newer version of com.android.tools:desugar_jdk_libs than 2.0.3 is available: 2.1.2

```
1 [version]
2 androidxDesugarJdkLibs = "2.0.3"
3 androidGradlePlugin = "8.3.2"
4 androidMaterial = "1.10.0"
5 androidxActivity = "1.7.2"
```

[/ / / gradlelibs/versions.toml](#): 2: A newer version of com.android.tools:desugar_jdk_libs than 2.0.3 is available: 2.1.2

```
1 [version]
2 androidxDesugarJdkLibs = "2.0.3"
3 androidGradlePlugin = "8.3.2"
4 androidMaterial = "1.10.0"
5 androidxActivity = "1.7.2"
```

[/ / / gradlelibs/versions.toml](#): 2: A newer version of com.android.tools:desugar_jdk_libs than 2.0.3 is available: 2.1.2

```
1 [version]
2 androidxDesugarJdkLibs = "2.0.3"
3 androidGradlePlugin = "8.3.2"
4 androidMaterial = "1.10.0"
5 androidxActivity = "1.7.2"
```

[/ / / gradlelibs/versions.toml](#): 4: A newer version of com.google.android.material:material than 1.10.0 is available: 1.12.0

```
1 [version]
2 androidxDesugarJdkLibs = "2.0.3"
3 androidGradlePlugin = "8.3.2"
4 androidMaterial = "1.10.0"
5 androidxActivity = "1.7.2"
6 androidxAppcompat = "1.3.1"
7 androidxComposeBom = "2024.05.00"
```

[/ / / gradlelibs/versions.toml](#): 4: A newer version of com.google.android.material:material than 1.10.0 is available: 1.12.0

```
1 [version]
2 androidxDesugarJdkLibs = "2.0.3"
3 androidGradlePlugin = "8.3.2"
4 androidMaterial = "1.10.0"
5 androidxActivity = "1.7.2"
6 androidxAppcompat = "1.3.1"
7 androidxComposeBom = "2024.05.00"
```

+ 79 MORE OCCURRENCES...

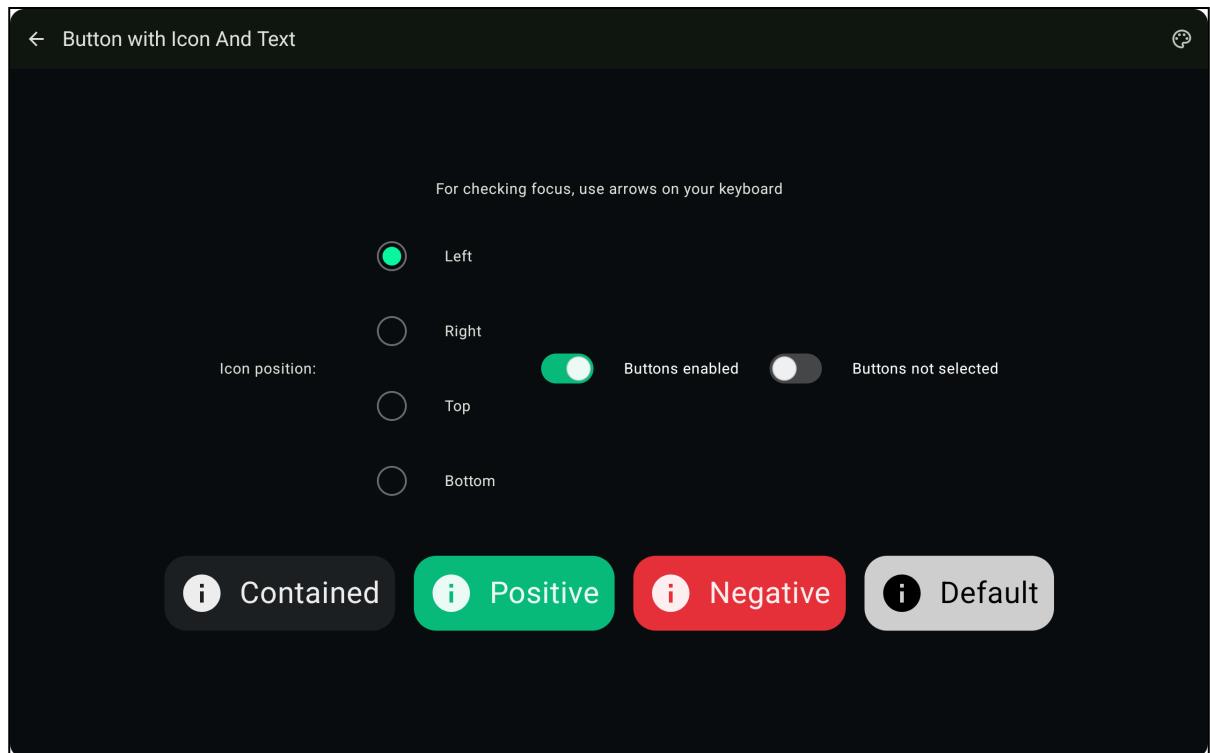
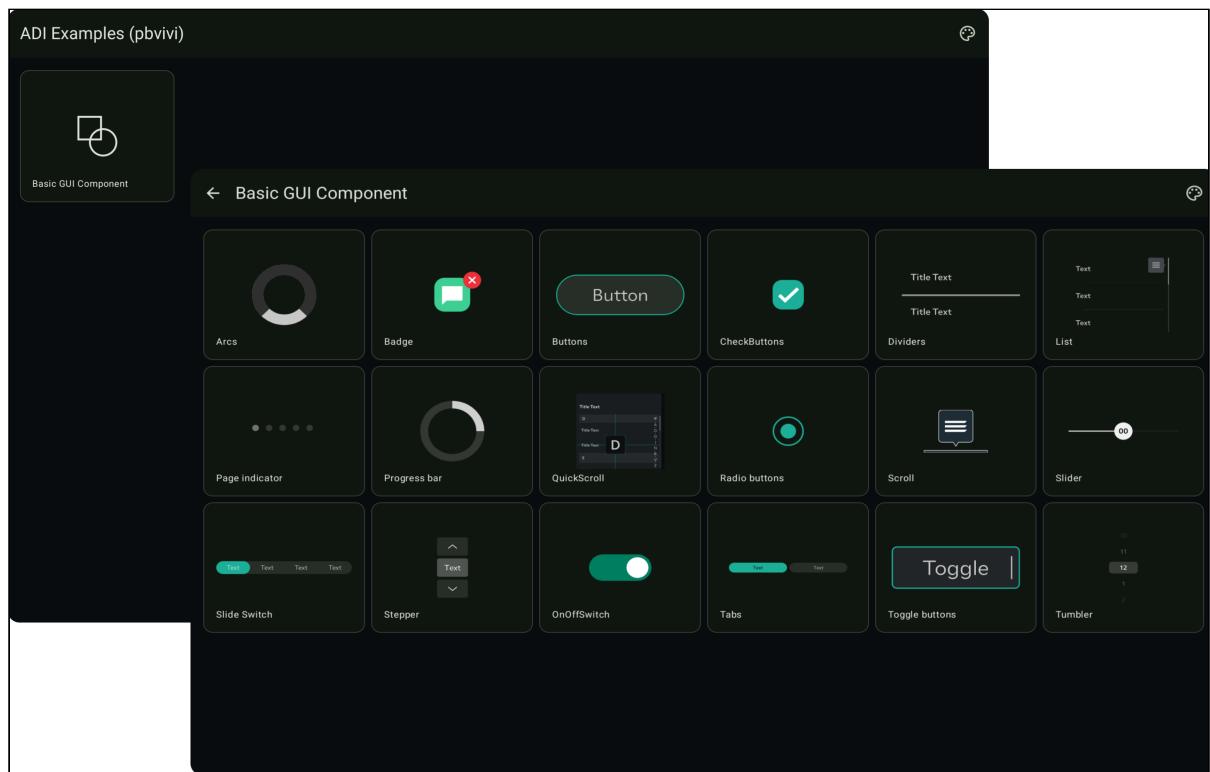
[GradleDependency](#) [Correctness](#) [Warning](#) [Priority 4/10](#)

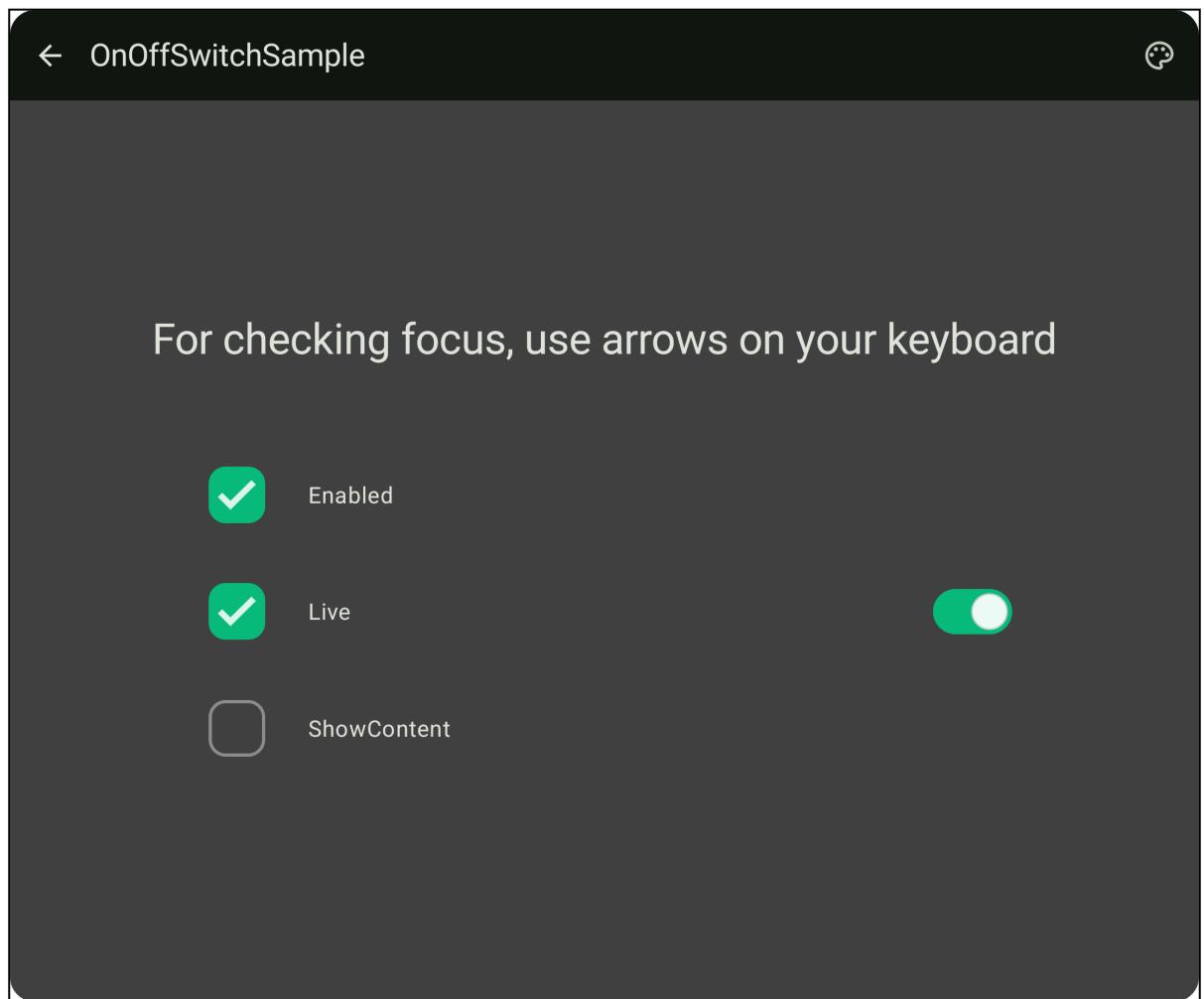
[EXPLAIN](#) [DISMISS](#)

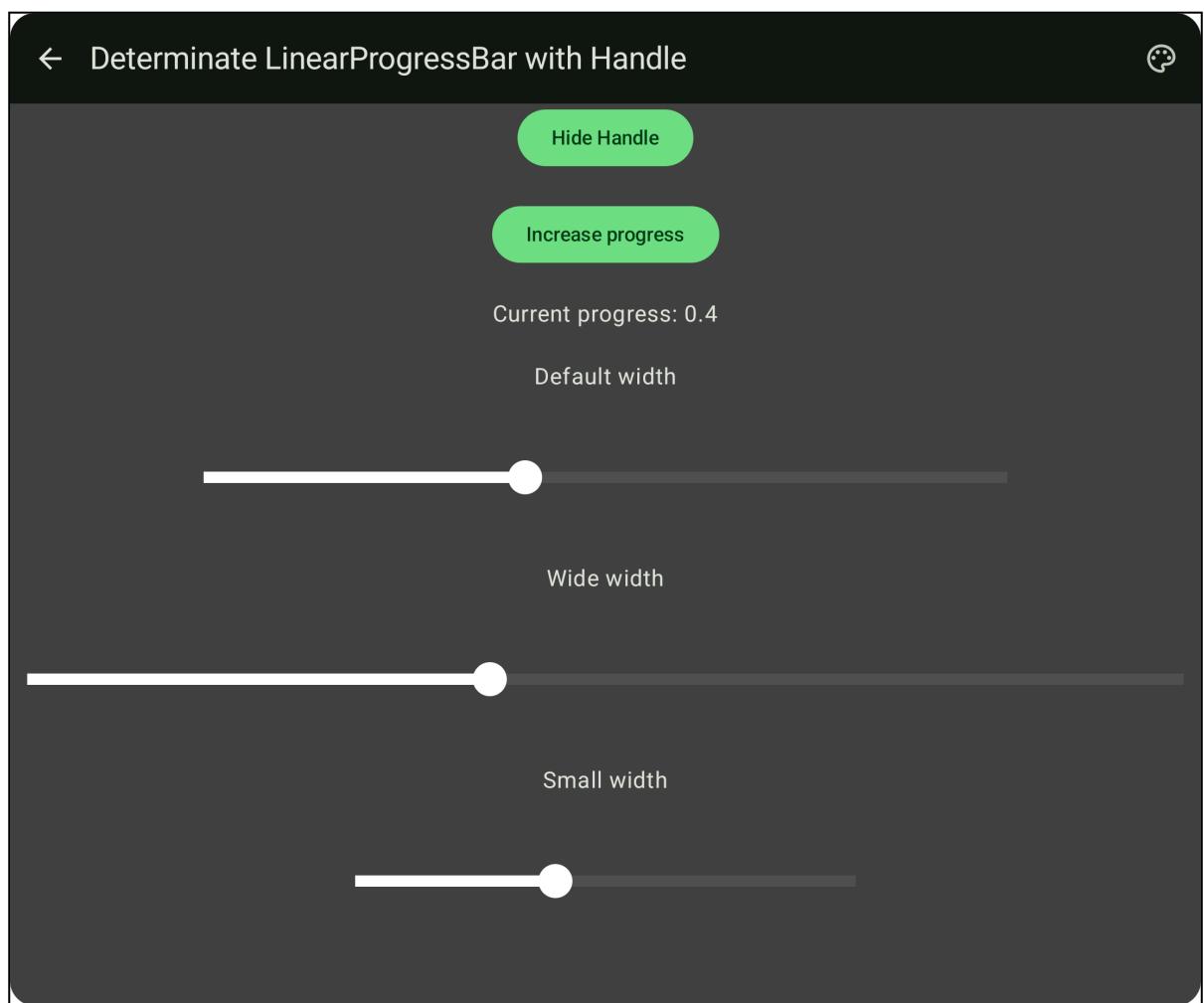
2.2 2.2 App 라이브러리

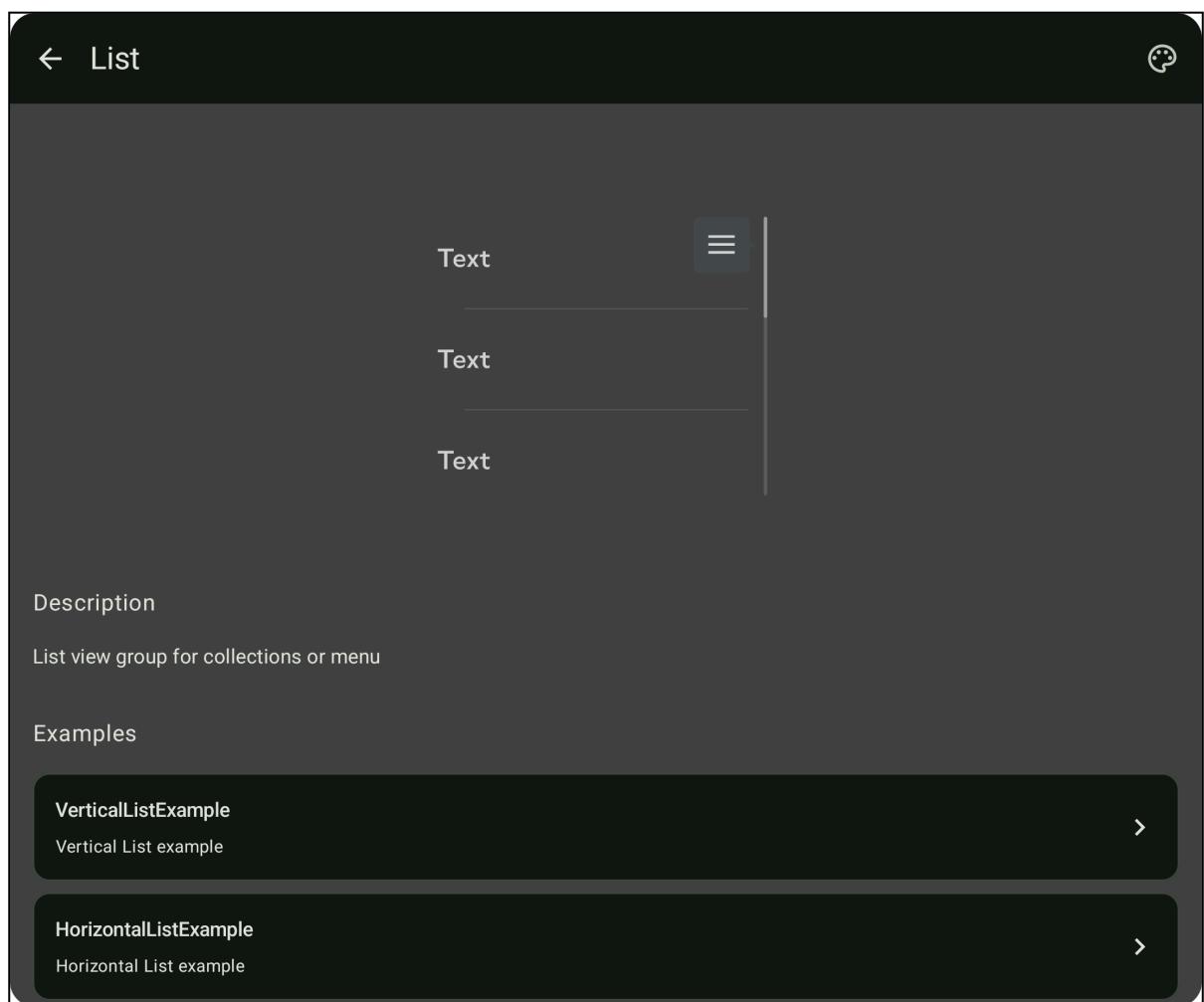
2.2.1 2.2.1 GUI Component

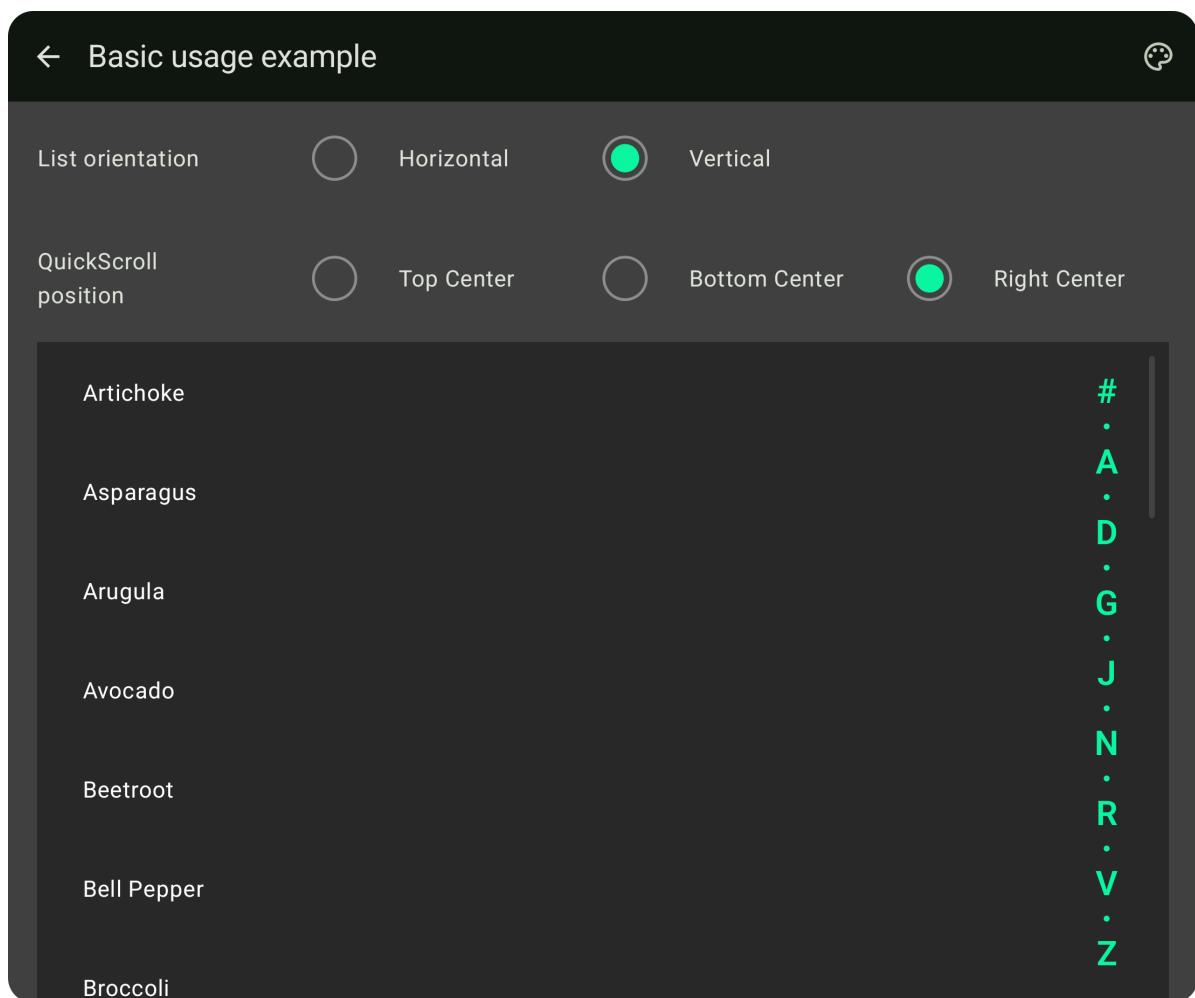
- 모든 HMI App.들의 GUI를 위한 Basic 한 GUI Component들을 제공

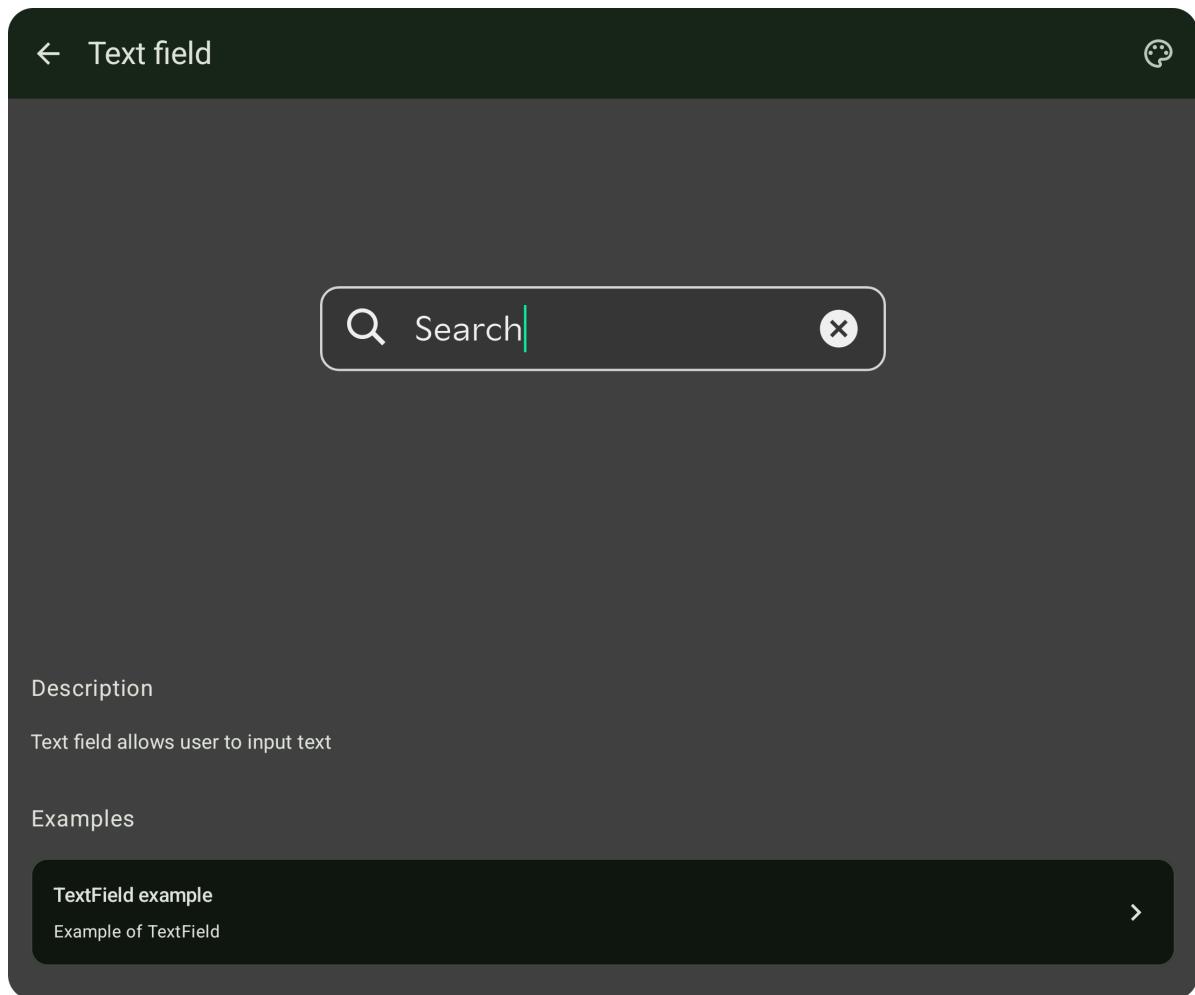












2.2.2 App Framework

- 모든 HMI App.들이 공용으로 사용할 수 있는 screen, navigation, search 및 Assembled GUI Component들과 Theme 등등의 기능들 제공

The build-logic module for Settings Application Framework			
Source	Description	Size	Last Modified
adi/master ... adi.framework.app.settings.build-logic / convention / src / main / java /		Browse Filter	
..			
adi/framework/app/settings/buildlogic			
ApplicationComposeConventionPlugin.kt	ENH: Initial Commit	582 B	15 Aug 2024
ApplicationConventionPlugin.kt	ENH: Initial Commit	835 B	15 Aug 2024
FeatureConventionPlugin.kt	ENH: Initial Commit	1.16 KB	15 Aug 2024
HiltConventionPlugin.kt	ENH: Initial Commit	931 B	15 Aug 2024
LibraryComposeConventionPlugin.kt	ENH: Initial Commit	570 B	15 Aug 2024
LibraryConventionPlugin.kt	ENH: Initial Commit	818 B	15 Aug 2024

Source

master ... adi.framework.app.settings.pbvivi / src / main / kotlin / com / android / settingslib / spa / framework / compose /

Browse Filter

Source	Description	Size	Last Modified
AnimatedNavGraphBuilder.kt	ENH: Apply page transition animation correctly	2.95 KB	12 Jun 2024
Debounce.kt	ENH: Apply debouncing logic to Reset Button in General module	1.26 KB	08 Aug 2024
DrawablePainter.kt	ENH: Apply ktlint plugin (v0.0.1) to all modules	6.63 KB	29 Feb 2024
Keyboards.kt	ENH: Enhance the logic for keyboard driving restriction	8.35 KB	5 days ago
LifecycleEffect.kt	ENH: Implementation of keyboard and search specifications	1.8 KB	18 Jul 2024
LogCompositions.kt	ENH: Change the top-module name from common to core	1.36 KB	14 Feb 2024
NavControllerWrapper.kt	ENH: Add navigate source type and highlightNavController	4.63 KB	5 days ago
OverrideableFlow.kt	ENH: Change the top-module name from common to core	1.08 KB	14 Feb 2024
PaddingValuesExt.kt	ENH: Refine submodule-template with samples and use GUI only in framework	1.87 KB	14 Feb 2024
RuntimeUtils.kt	ENH: Change the top-module name from common to core	1.3 KB	14 Feb 2024
TimeMeasurer.kt	ENH: Change the top-module name from common to core	1.68 KB	14 Feb 2024

MADI_maven

adt
 > cldc
 > example
 > examples
framework
 app
 connect
 settings
 framework-pbvivi
 1.0.0
 1.0.1
 1.0.1-alpha
 1.0.2
 1.0.3
 1.0.4
 1.0.4-alpha01
 1.0.6
 1.0.7
 maven-metadata.xml
 maven-metadata.xml.sha512

maven-metadata.xml

General XML View Properties

XML Content

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <metadata>
3    <groupId>adi.framework.app.settings</groupId>
4    <artifactId>framework-pbvivi</artifactId>
5    <versioning>
6      <latest>1.0.7</latest>
7      <release>1.0.7</release>
8      <versions>
9        <version>1.0.0</version>
10       <version>1.0.1-alpha</version>
11       <version>1.0.1</version>
12       <version>1.0.2</version>
13       <version>1.0.3</version>
14       <version>1.0.4-alpha01</version>
15       <version>1.0.4</version>
16       <version>1.0.6</version>
17       <version>1.0.7</version>
18     </version>
19     <lastUpdated>20240911051735</lastUpdated>
20   </versioning>
21 </metadata>

```

M.AF for Connect

EXAMPLE #1 Go to Sub-Page EXAMPLE #1

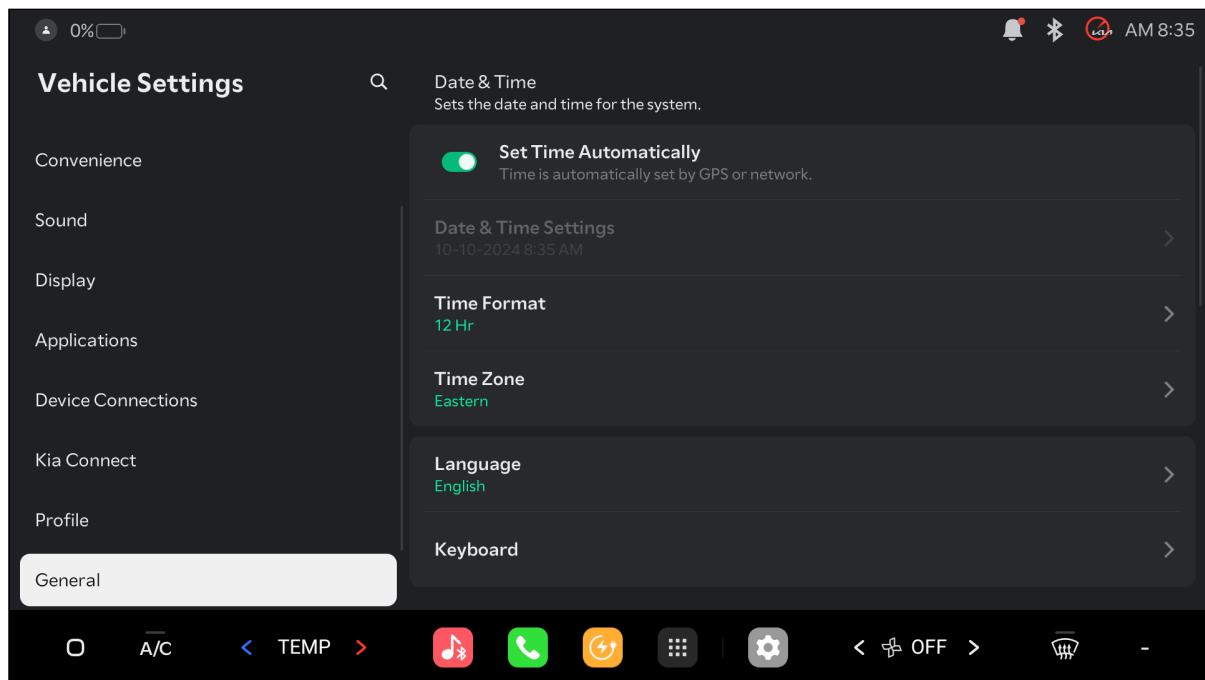
EXAMPLE #2 TabTop #1 TabTop #2

EXAMPLE #3 Switch Preference Summary

EXAMPLE #4 Checkbox Preference Summary

EXAMPLE #5 SliderSwitch Preference Summary

Item #1 Item #2 Item #3



2.3 2.3 M.ADI Maven Repository

- 공용화를 위한 App 플러그인들 및 라이브러리들의 배포 및 각 App에서 download 해서 사용하기 위한 저장소 구축

The screenshot shows the JFrog Platform interface with the following details:

- Left Sidebar:** Artifactory, Packages, Builds, Artifacts, Distribution, Pipelines, Security & Compliance.
- Top Bar:** Artifacts, Search Artifacts, Set Me Up, Deploy, My Favorites, Tree View.
- Middle Area:**
 - Repositories:** Happily serving 932,892 artifacts. A tree view shows 'MADI_maven' expanded, listing sub-folders like 'adt', 'ai', 'android', etc.
 - Selected Repository:** MADI_maven (Maven) with URL https://artifacts.mobis.com:443/artifactory/MADI_maven/
 - Properties:**
 - General:** Name: MADI_maven, Package Type: Maven, Repository Path: https://artifacts.mobis.com:443/artifactory/MADI_maven/, Repository Layout: maven-2-default.
 - Info:** Description: Private maven repository for MADI, Artifact Count / Size: Show, Created: 23-10-11 09:51:44 +09:00.
 - Package Information:** Dependency Declaration.
 - Virtual Repository Associations:**
 - Included Repositories:**
- Bottom Left:** Commercial License 7.11.3 rev 10000000000000000000000000000000 by Hyundai Motor © Copyright 2024 JFrog Ltd.

The screenshot shows the JFrog Platform interface, specifically the Artifactory section. On the left, there's a sidebar with navigation links: Artifactory, Packages, Builds, Artifacts (which is selected), Distribution, Pipelines, and Security & Compliance. Below the sidebar, there's a license notice: "Commercial license 7.11.5 rev 7115500 Licensed to Hyundai Mobis © copyright 2021 JFrog Ltd".
The main area displays a tree view of artifacts under the "MADI_maven" repository. The tree structure includes: adi, examples, framework, gpt, plugin, and several sub-folders like basic, basic-cstd, basic-generic, basic-pbvl, basic-soip, foundation-pbvl, gradle, buildext, ktlint, publish, language, and translator.
On the right, a detailed view of the artifact "1.0.6" is shown. The "General" tab is selected, displaying the following information:

- Name: 1.0.6
- Repository Path: MADI_maven/adi/plugin/language/translator/1.0.6
- URL to file: https://artifacts.mobis.com:443/artifactory/MADI_maven/adi/plugin/language/translator/1.0.6
- Deployed By: 1626740
- Artifact Count / Size: Show
- Created: 24-10-03 15:49:57 +09:00 (0d 23h 50m 42s ago)

Below the General tab, there are sections for Dependency Declaration, Virtual Repository Associations, Included Repositories, and Checksums.

3 3. 자동화

3.1 3.1 CI/CD

- 자동 Build 검증 수행 및 KtLint 및 Android Lint 검증 결과에 대한 Repotring을 통한 App. 품질 검토 기능 강화
1) 개발자가 PR 생성



김인규(KIM IN KYU) 인포홈/설정APP팀 OPENED the pull request 20 Sep 2024

- CI 검증 시작



인포홈/설정APP팀_공용01 인포홈/설정APP팀 20 Sep 2024

[CI] Build Started: https://ivici.mobis.com/job/M_HMI/job/repos/job/platform.vendor.mobis.packages.apps.settings/job/PR-525/3/

[Reply](#) · [Resolve](#) ☺ ...

- CI 검증 결과 리포팅



인포홈/설정APP팀_공용01 인포홈/설정APP팀 20 Sep 2024

[CI] Build finished.

Build Status: — https://ivici.mobis.com/job/M_HMI/job/repos/job/platform.vendor.mobis.packages.apps.settings/job/PR-525/3/

Downstream jobs:

⚠️ M_HMI (hmi_apps) | [Log](#) | [KTLINT](#) | [ANDROID_LINT](#)

[Reply](#) · [Resolve](#) ☺ ...

- 상기 KTLINT 링크를 통해서, 하기와 같이 KtLint 이슈에 대해서 file 단위로 문제점에 대해서 확인

Dashboard > M_ADI > Downstreams > linux.app.downstream > #1464 > KtLint Warnings

KtLint Warnings

Status: Success

Changes: 0

Console Output: View

View as plain text: View

View Build Information: View

Parameters: View

Timings: View

Git Build Data: View

KtLint Warnings (Selected)

Android Lint Warnings: View

See Fingerprints: View

Pipeline Overview: View

Pipeline Console: View

Pipeline Steps: View

Previous Build: View

Next Build: View

Overview

History

Details

Issues

Show 10 entries

Search:

Details	File	Package	Category	Type	Severity	Age
PageProvider.kt:27	adi.framework.app.connect.framework.common	Standard:no-unused-imports	standard:no-unused-imports	Error	1	

Dashboard > M_ADI > Downstreams > linux.app.downstream > #1464 > KtLint Warnings > PageProvider.kt

Content of file PageProvider.kt

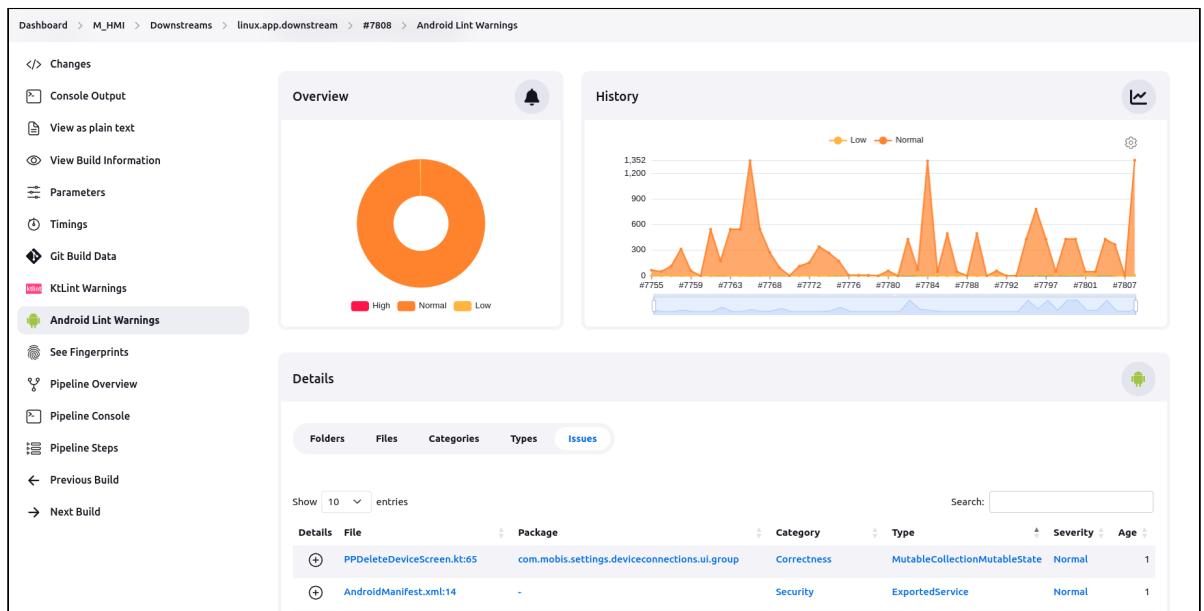
```

1  /*
2   * Copyright (C) 2022 The Android Open Source Project
3   *
4   * Licensed under the Apache License, Version 2.0 (the "License");
5   * you may not use this file except in compliance with the License.
6   * You may obtain a copy of the License at
7   *
8   *      http://www.apache.org/licenses/LICENSE-2.0
9   *
10  * Unless required by applicable law or agreed to in writing, software
11  * distributed under the License is distributed on an "AS IS" BASIS,
12  * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13  * See the License for the specific language governing permissions and
14  * limitations under the License.
15  */
16
17 package adi.framework.app.connect.framework.common
18
19 import adi.framework.app.connect.framework.compose.LocalNavController
20 import adi.framework.app.connect.framework.compose.NavControllerWrapperImpl
21 import adi.framework.app.connect.framework.theme.Dimension
22 import adi.framework.app.connect.framework.theme.Shape
23 import adi.framework.app.connect.framework.util.genPageId
24 import adi.framework.app.connect.framework.util.normalizeArglist
25 import adi.framework.app.connect.widget.scaffold.BaseScaffold
26 import adi.framework.app.connect.widget.scaffold.NavigateBack
27 import adi.framework.app.connect.widget.scaffold.OptionMenuAction
28 import adi.framework.app.connect.widget.scaffold.Topbar.MenuBar
29 import adi.framework.app.connect.widget.scaffold_topbar.TopBar

```

Unused import

5) 상기 ANDROID_LINT 링크를 통해서, 하기와 같이 Android Lint 이후에 대해서 file 단위로 문제점에 대해서 확인



```

50     ] {
51         val logTag = "PBV_PP_SETTINGS:DEL"
52         val context = LocalContext.current
53         val listMaxIndex = phoneProfileDataSource.ListInfo.LIST_MAX_SIZE
54         val data = when (ppType) {
55             ANDROID_AUTO -> [
56                 phoneProfileUseCase.getAndroidAutoData()
57             ]
58             CAR_PLAY -> [
59                 phoneProfileUseCase.getCarPlayData()
60             ]
61             else -> return
62         }
63
64         val viewModelData by data.collectAsState(initial = emptyList())
65         var deviceList by remember { mutableStateOf(phoneList.toMutableList()) }

Creating a MutableState object with a mutable collection type
Creating a MutableState object with a mutable collection type
Writes to mutable collections inside a MutableState will not cause a recomposition - only writes to the MutableState itself will. In most cases you should either use a read-only collection (such as List or Map) and assign a new instance to the MutableState when your data changes, or you can use an snapshot-backed collection such as SnapshotStateList or SnapshotStateMap which will correctly cause a recomposition when their contents are modified.

66     val selectedDeviceIds = remember { mutableStateListOf<String>() }
67     val markAllButtonEnabled = remember { mutableStateOf(true) }
68     val deleteButtonEnabled = remember { mutableStateOf(true) }
69
70     // 내비게이션 컨트롤러
71     val navController = rememberNavController()
72     val controller = LocalNavController.current as NavControllerWrapperImpl
73
74     // 삭제 버튼 클릭 이벤트
75     fun handleDeleteButtonClick() {
76         // 선택된 장치 ID 목록을 업데이트
77         selectedDeviceIds.clear()
78         deviceList.forEach { device ->
79             if (device.isChecked == true) {
80                 selectedDeviceIds.add(device.deviceID)
81             }
82         }
83     }

```

6) 최종적으로 CI 검증 결과 따라서 PR 승인



3.2 3.2 AI Code Reviewer

- AI를 통한 자동화된 Code Review를 통한 App. 품질 검토 기능 강화

1) 개발자가 PR¹ 생성



2) AI Code Reviewer가 PR에 대한 검토를 시작



3) AI Code Reviewer가 검토를 완료하면, 해당 결과를 comment로 하기와 같이 추가



¹ https://bitbucket.mobis.co.kr/projects/M_HMI/repos/platform.vendor.mobis.packages.apps.extcamera/pull-requests/878/overview

Defects, Runtime Errors, and Security Vulnerabilities(결함, 런타임 오류, 보안 취약점):

Defect(결함): Coroutine Scope Misuse

- File: AudioRepositoryImpl.kt
- Rationale: Creating a new CoroutineScope with Job(). For each function call can lead to resource leaks and improper job management.
- 이유: 각 함수 호출마다 Job()으로 새로운 CoroutineScope를 생성하면 리소스 누수 및 부적절한 작업 관리가 발생할 수 있습니다.
- Code Snippet:

```
CoroutineScope(Job()).launch {
    // ...
}
```
- Suggestion: Use a single CoroutineScope with a proper lifecycle management.
- 제안: 적절한 생명주기 관리를 통해 단일 CoroutineScope를 사용하십시오.

Runtime Error(런타임 오류): Potential NullPointerException

- File: DisplaySettingsRepositoryImpl.kt
- Rationale: Accessing properties without null checks can lead to NullPointerException.
- 이유: null 검사를 하지 않고 속성에 접근하면 NullPointerException이 발생할 수 있습니다.
- Code Snippet:

```
val brightness = if (nightMode.value) nightBrightness.value else dayBrightness.value
```
- Suggestion: Add null checks or use safe calls.
- 제안: null 검사를 추가하거나 안전 호출을 사용하십시오.

Security Vulnerability(보안 취약점): Hardcoded Sensitive Data

- File: InfoBigDataRepositoryImpl.kt
- Rationale: Hardcoding sensitive data like user IDs can lead to security vulnerabilities.
- 이유: 사용자 ID와 같은 민감한 데이터를 하드코딩하면 보안 취약점이 발생할 수 있습니다.
- Code Snippet:

```
val guestUserId = settingsDataSource.getSettingsData(String::class, SettingsTable.GLOBAL, "guest_user_id", 0)
```
- Suggestion: Use secure storage mechanisms for sensitive data.
- 제안: 민감한 데이터에 대해 보안 저장 메커니즘을 사용하십시오.

Structural Improvements, Optimization, and Best Practices(구조적 개선, 최적화 및 모범 사례):

Structural Improvements(구조적 개선): Use Dependency Injection

- File: AudioRepositoryImpl.kt
- Rationale: Use dependency injection (DI) for CoroutineScope to improve testability and maintainability.
- 이유: CoroutineScope에 대해 의존성 주입(DI)을 사용하여 테스트 가능성과 유지 관리성을 향상시키십시오.
- Code Snippet:

```
class AudioRepositoryImpl @Inject constructor(
    private val audioManagerDataSource: AudioManagerDataSource,
    private val openAudioManagerDataSource: OpenAudioManagerDataSource,
    private val coroutineScope: CoroutineScope
) : AudioRepository {
    // ...
}
```

Optimization(최적화): Avoid Redundant StateFlow Conversions

- File: RvnCameraSettingsRepositoryImpl.kt
- Rationale: Directly use StateFlow without redundant conversions to improve performance.
- 이유: 성능을 향상시키기 위해 불필요한 변환 없이 StateFlow를 직접 사용하십시오.
- Code Snippet:

```
override val extendedRearViewMonitor: StateFlow<Boolean> = _extendedRearViewMonitor.map { it == 1 }.stateIn(repositoryScope, Eagerly, false)
```

Best Practices(모범 사례): Use Constants for Keys

- File: InfoBigDataRepositoryImpl.kt
- Rationale: Use constants for keys to avoid hardcoding and improve maintainability.
- 이유: 하드코딩을 피하고 유지 관리성을 향상시키기 위해 키에 대해 상수를 사용하십시오.
- Code Snippet:

```
companion object {
    const val QUEST_USER_ID_KEY = "guest_user_id"
    const val DRIVER1_USER_ID_KEY = "driver1_user_id"
    const val DRIVER2_USER_ID_KEY = "driver2_user_id"
}
```

Performance and Memory Management(성능 및 메모리 관리):

Performance(성능): Use Dispatchers.IO for I/O Operations

- File: InfoBigDataRepositoryImpl.kt
- Rationale: Use Dispatchers.IO for I/O operations to avoid blocking the main thread.
- 이유: 메인 스레드를 차단하지 않기 위해 I/O 작업에 Dispatchers.IO를 사용하십시오.
- Code Snippet:

```
private val repositoryScope = CoroutineScope(Dispatchers.IO + supervisorJob)
```

Memory Management(메모리 관리): Proper Job Management

- File: AudioRepositoryImpl.kt
- Rationale: Ensure proper job management to avoid memory leaks.
- 이유: 메모리 누수를 방지하기 위해 적절한 작업 관리를 보장하십시오.
- Code Snippet:

```
private val job = Job()
private val scope = CoroutineScope(Dispatchers.IO + job)

fun clear() {
    job.cancel()
}
```

By addressing these issues, the code will be more robust, maintainable, and performant.

Reply ⚡ ...

▪ Code Review 결과에 대해 Reviewer가 추가적인 질문시 #AI와 함께 질문을 하면, 이에 대한 응답

1) 개발자가 AI Code Review 결과에 대해서 좀 더 상세히 알고자 하는 경우, 하기와 같이 [추가적으로 질문²](#)을 수행

이재학(LEE JAE HAK) 인포ழ/설정APP팀 A moment ago ⚡

#AI

🔒 Security Vulnerability(보안 취약점): Hardcoded Sensitive Data

- File: InfoBldDataRepositoryImpl.kt
- Rationale: Hardcoding sensitive data like user IDs can lead to security vulnerabilities.
- 이유: 사용자 ID와 같은 민감한 데이터를 하드코딩하면 보안 취약점이 발생할 수 있습니다.
- Code Snippet:

```
val guestUserId = settingsDataSource.getSettingData(String::class, SettingsTable.GLOBAL, "guest_user_id", 0)
Suggestion: Use secure storage mechanisms for sensitive data.
```
- 제안: 민감한 데이터에 대해 보안 저장 메커니즘을 사용하십시오.

상기 민감한 데이터에 대해 보안 저장 메커니즘에 대해서 좀 더 자세히 설명해줘

Reply · Edit · Resolve ⚡ ...

2) AI Code Reviewer가 개발자의 질문에 대해서 분석하기 위해 준비 (개발자의 질문에 연관된 code를 내부적으로 search 하는 단계)

Access Token User - M.HMI A moment ago ⚡

@이재학(LEE JAE HAK) 인포ழ/설정APP팀

The preprocessing is run for 1st M.GPT starting. So, please wait...

Reply ⚡ ...

3) AI Code Reviewer가 개발자의 질문에 대해서 분석 준비 완료 이후, 질문에 대한 분석 시작

Access Token User - M.HMI Yesterday ⚡

@이재학(LEE JAE HAK) 인포ழ/설정APP팀

Welcome to 1st M.GPT Comment Chat 🎉

To address the security vulnerability of hardcoding sensitive data, such as user IDs, you should use secure storage mechanisms. Here are some recommendations:

보안 취약점인 사용자 ID와 같은 민감한 데이터를 하드코딩하는 문제를 해결하기 위해 보안 저장 메커니즘을 사용해야 합니다. 다음은 몇 가지 권장 사항입니다:

Reply ⚡ ...

4) AI Code Reviewer가 개발자의 질문에 대한 분석 완료 이후 답변을 comment로 추가

Access Token User - M.HMI 1 week ago ⚡

@이재학(LEE JAE HAK) 인포ழ/설정APP팀

Welcome to 1st M.GPT Comment Chat 🎉

To address the security vulnerability of hardcoding sensitive data, such as user IDs, you should use secure storage mechanisms. Here are some recommendations:

보안 취약점인 사용자 ID와 같은 민감한 데이터를 하드코딩하는 문제를 해결하기 위해 보안 저장 메커니즘을 사용해야 합니다. 다음은 몇 가지 권장 사항입니다:

1. Encrypted Shared Preferences:

- Use the `EncryptedSharedPreferences` class to store sensitive data securely.
- EncryptedSharedPreferences 클래스를 사용하여 민감한 데이터를 안전하게 저장하십시오.

```
// Kotlin
import androidx.security.crypto.EncryptedSharedPreferences
import androidx.security.crypto.MasterKeys

val masterKeyAlias = MasterKeys.getOrCreate(MasterKeys.AES256_GCM_SPEC)
val sharedPreferences = EncryptedSharedPreferences.create(
    "secure_prefs",
    masterKeyAlias,
    context,
    EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256_SIV,
    EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256_GCM
)

val editor = sharedPreferences.edit()
editor.putString("guest_user_id", "your_guest_user_id")
editor.apply()
```

2 https://bitbucket.mobis.co.kr/projects/M_HMI/repos/platform.vendor.mobis.packages.apps.cargo/pull-requests/159/overview?commentId=775552

2. **Android Keystore System:**

- o Use the Android Keystore system to generate and store cryptographic keys.
- o Android Keystore 시스템을 사용하여 암호화 키를 생성하고 저장하십시오.

```
// Kotlin
import android.security.keystore.KeyGenParameterSpec
import android.security.keystore.KeyProperties
import java.security.KeyStore
import javax.crypto.Cipher
import javax.crypto.KeyGenerator
import javax.crypto.SecretKey
import javax.crypto.spec.GCMParameterSpec

val keyGenerator = KeyGenerator.getInstance(KeyProperties.KEY_ALGORITHM_AES, "AndroidKeyStore")
keyGenerator.init(
    KeyGenParameterSpec.Builder("keyAlias",
        KeyProperties.PURPOSE_ENCRYPT or KeyProperties.PURPOSE_DECRYPT)
        .setBlockModes(KeyProperties.BLOCK_MODE_GCM)
        .setEncryptionPadding(KeyProperties.ENCRYPTION_PADDING_NONE)
        .build()
)
val secretKey = keyGenerator.generateKey()

val cipher = Cipher.getInstance("AES/GCM/NoPadding")
cipher.init(Cipher.ENCRYPT_MODE, secretKey)
val encryptionIV = cipher.iv
val encryptedData = cipher.doFinal("your_guest_user_id".toByteArray())

By using these secure storage mechanisms, you can protect sensitive data from being exposed.
이러한 보안 저장 메커니즘을 사용하여 민감한 데이터가 노출되지 않도록 보호할 수 있습니다.
```

Reply ⚡ ...