CUSTOM **LINT RULES** FOR A CUSTOM **WORLD**

ONTRUCK

Chema
Android Engineer
@jmperezra

Who am I?

Chema
Android Engineer
@jmperezra

Agenda

1. Ontruck
2. Lint
3. Our use case

onTRUCK

Trucking is BROKEN

30%

kilometres are driven empty

Trucking is BROKEN

For every **1%** we reduce in empty km, we save…

1.2Bn €	100M	240M
Transport costs	Liters of fuel	Km being driven

Ontruck - Mobile Guild

Álvar Montes

Antonio Calvo

David Vallejo

Ontruck - Mobile Guild

Chema Pérez

Iván Martínez

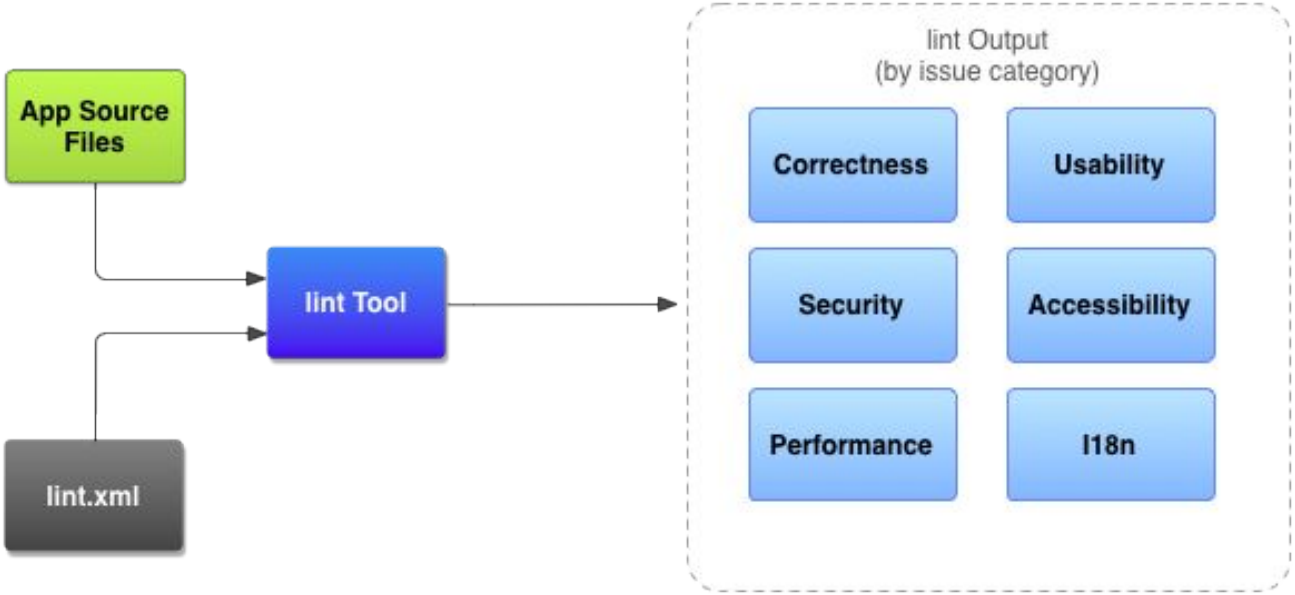
Ana Llera

What is **Lint?**

What is Lint?

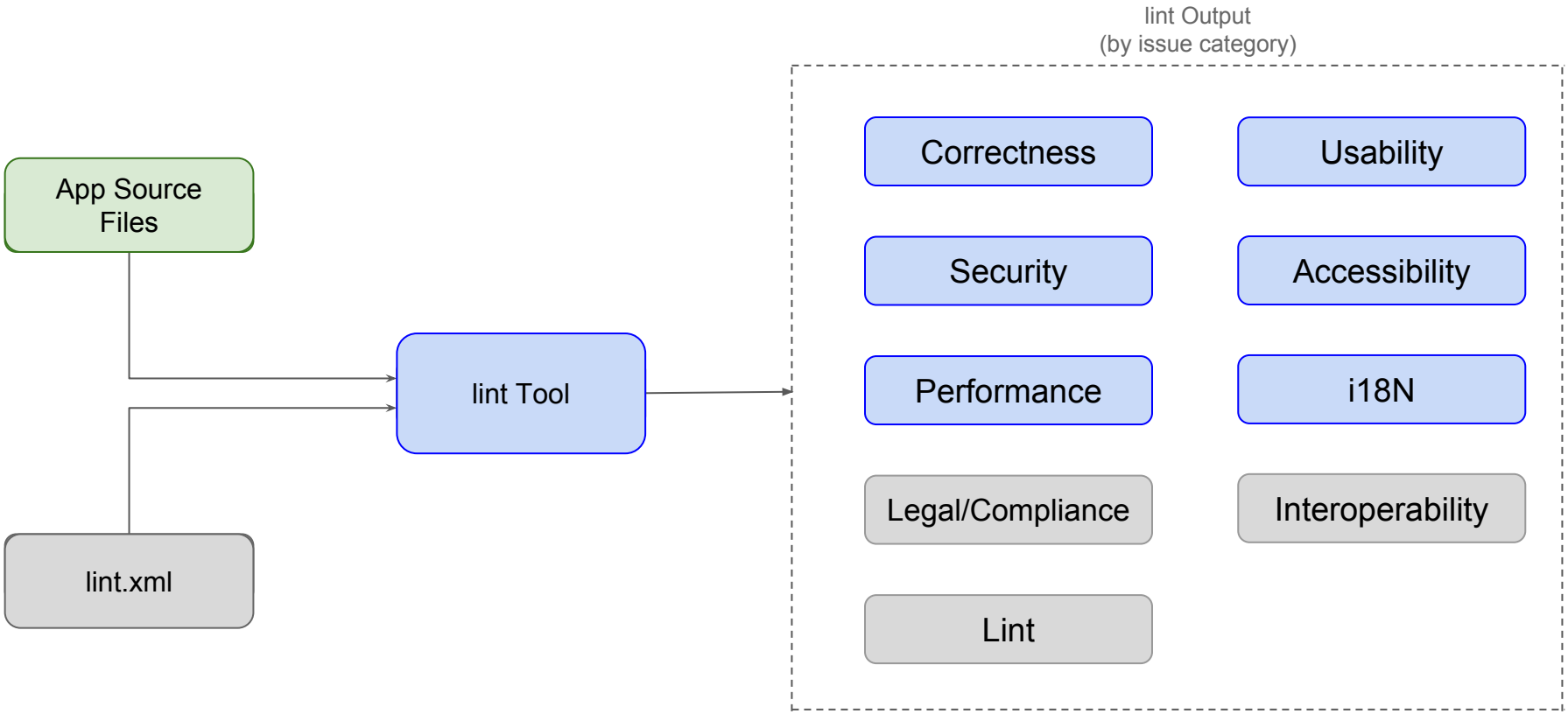
- A code scanning tool that helps prevent and fix structural code problems
- No need to run the app or write tests
- Each detected problem is communicated with a descriptive message and a security level

What is Lint?

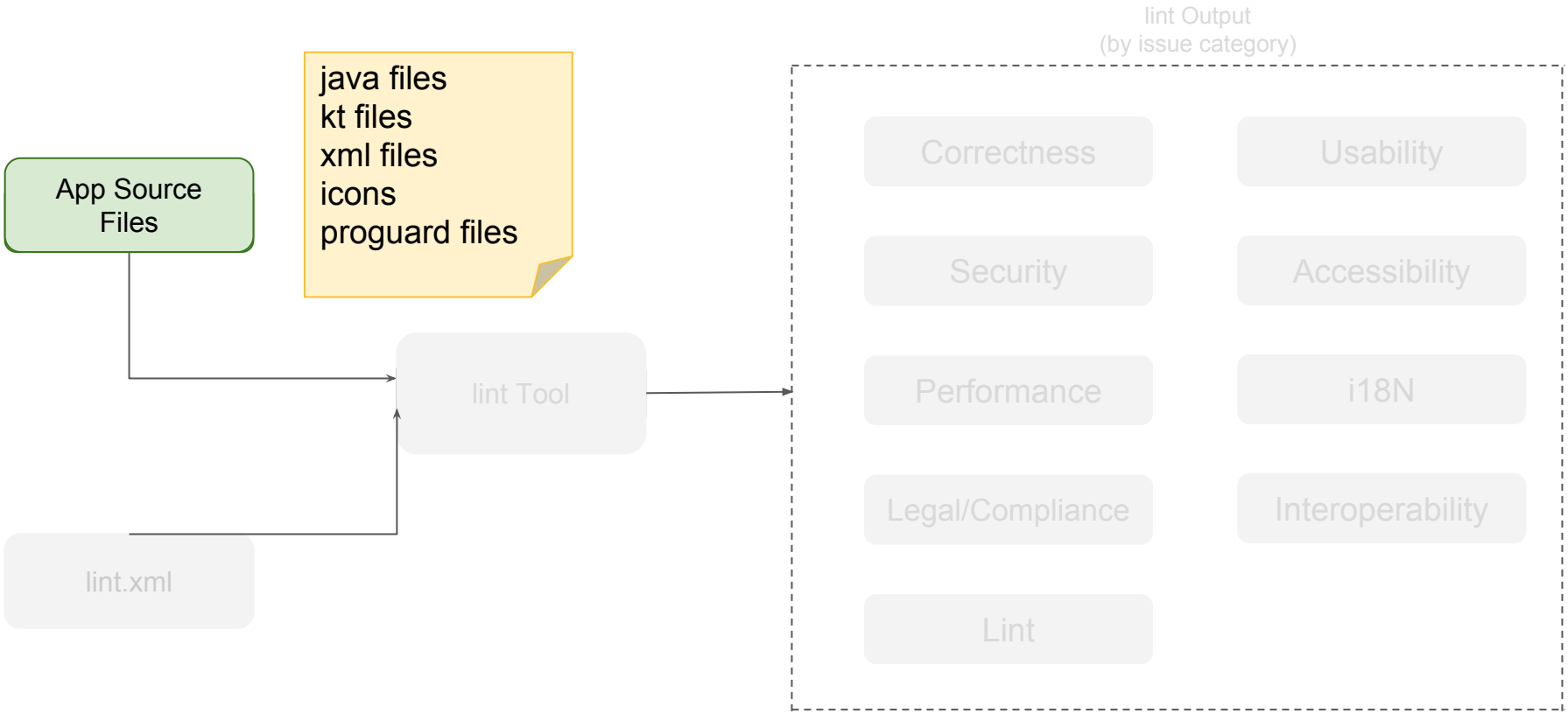


Source: https://developer.android.com/studio/images/write/lint.png

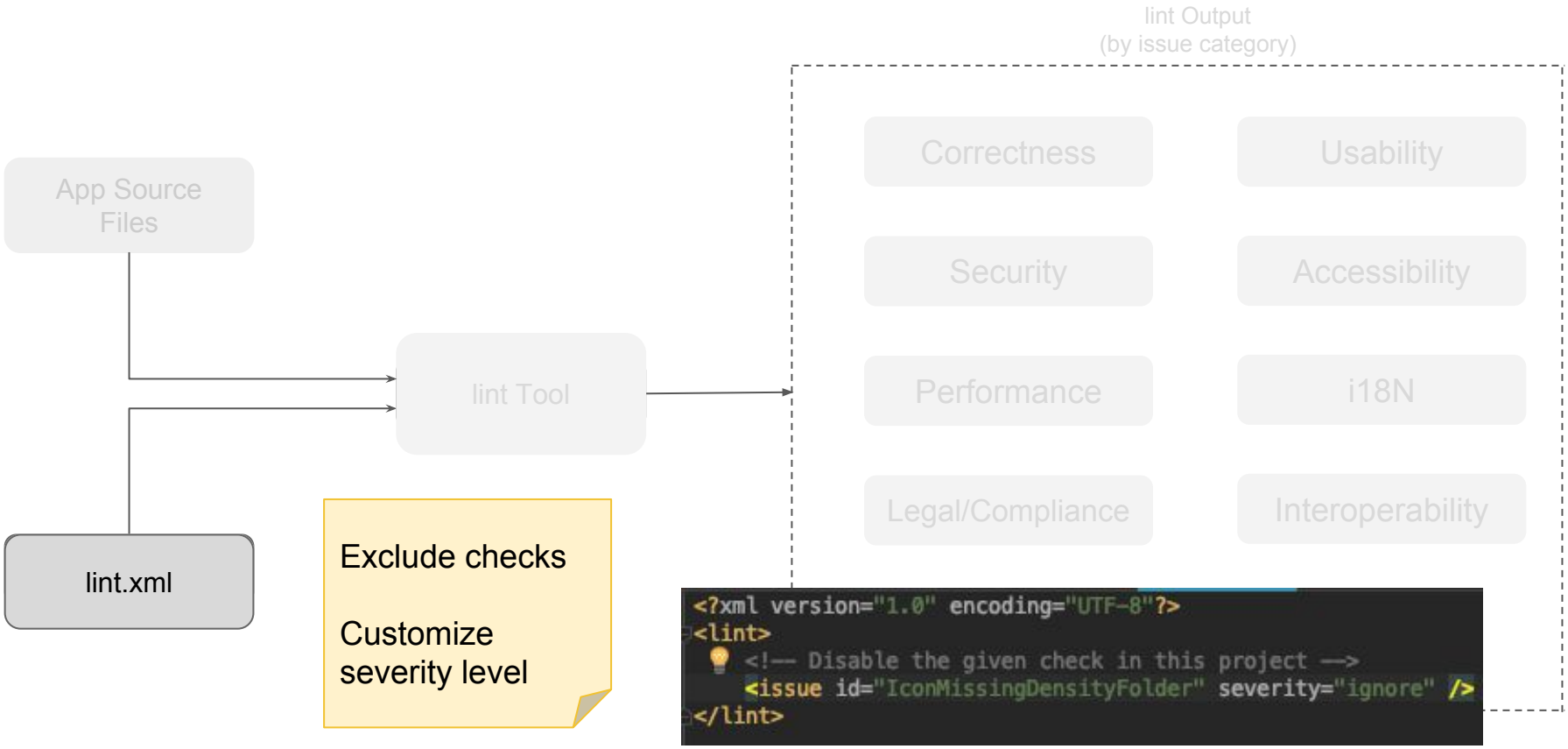
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

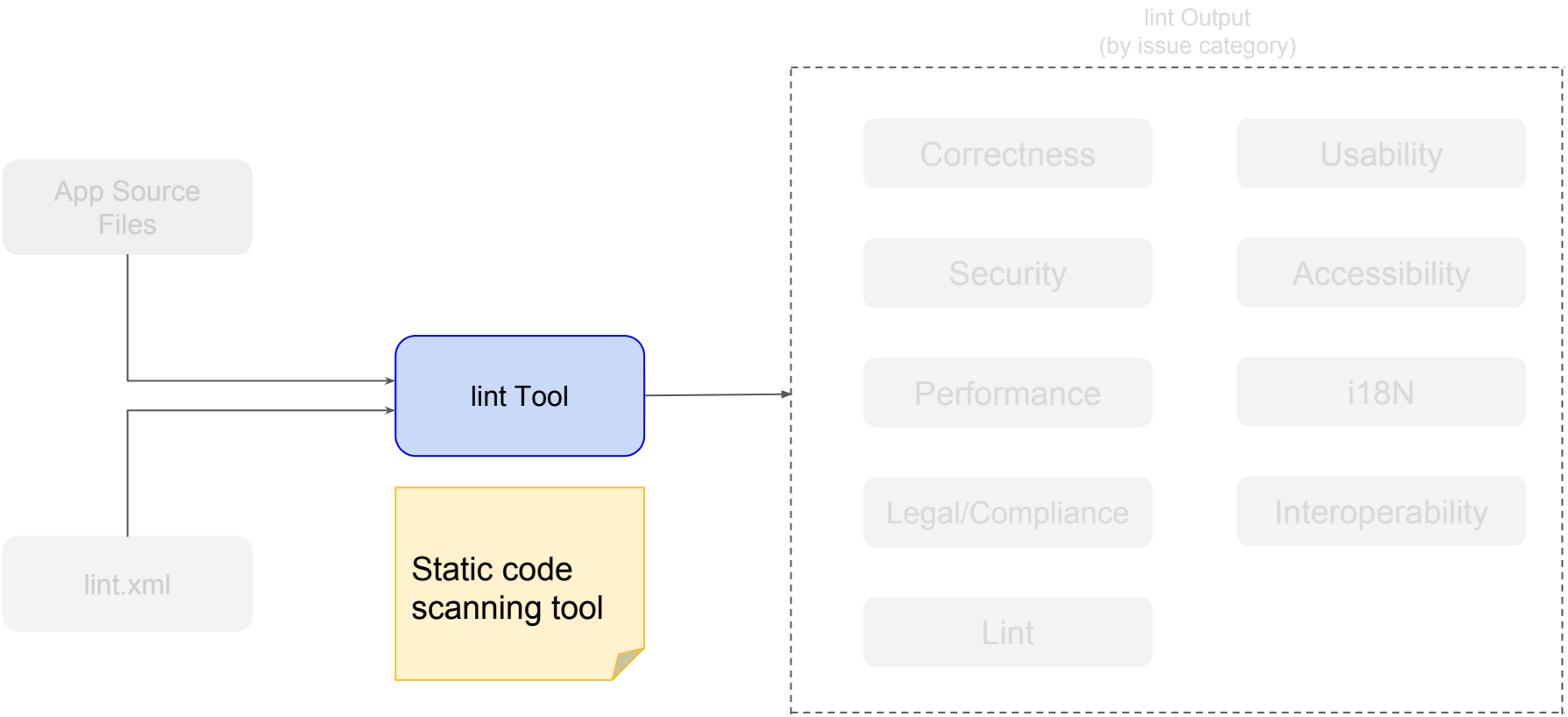
What is Lint?


App Source Files
java files
kt files
xml files
icons
proguard files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

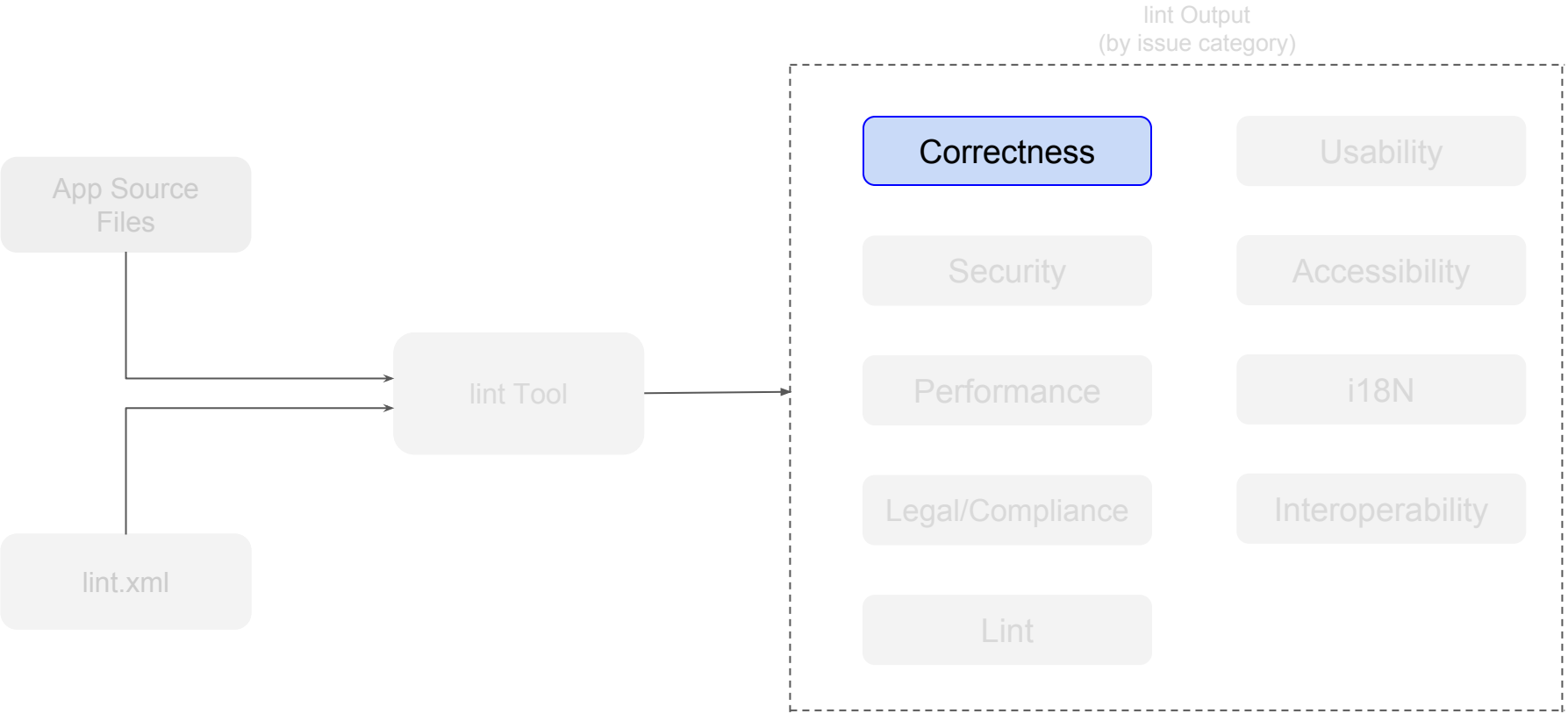
What is Lint?


lint Output
(by issue category)
App Source Files
lint Tool
lint.xml
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Exclude checks
Customize severity level
<?xml version="1.0" encoding="UTF-8"?>
<lint>
<!-- Disable the given check in this project -->
<issue id="IconMissingDensityFolder" severity="ignore" />
</lint>

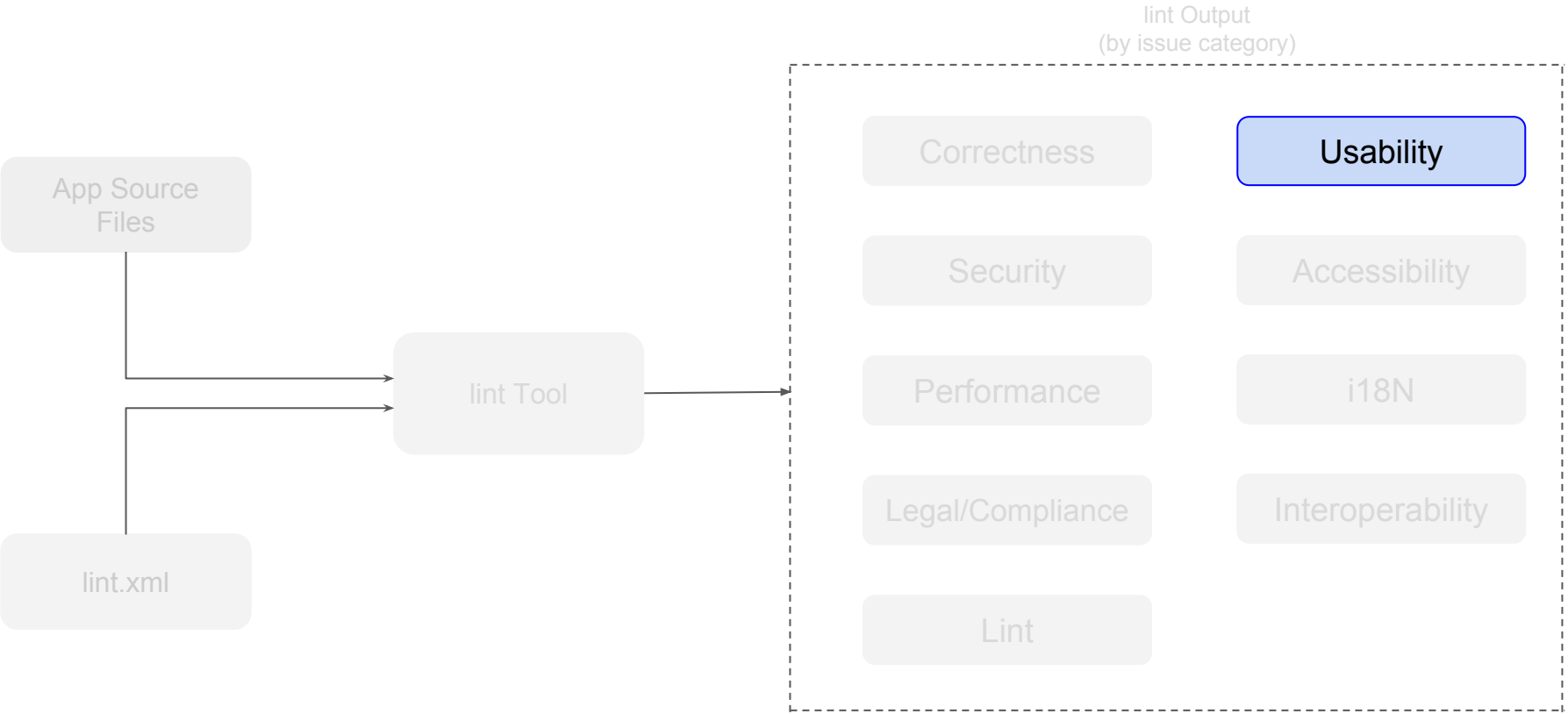
What is Lint?


App Source Files
lint.xml
lint Tool
Static code scanning tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

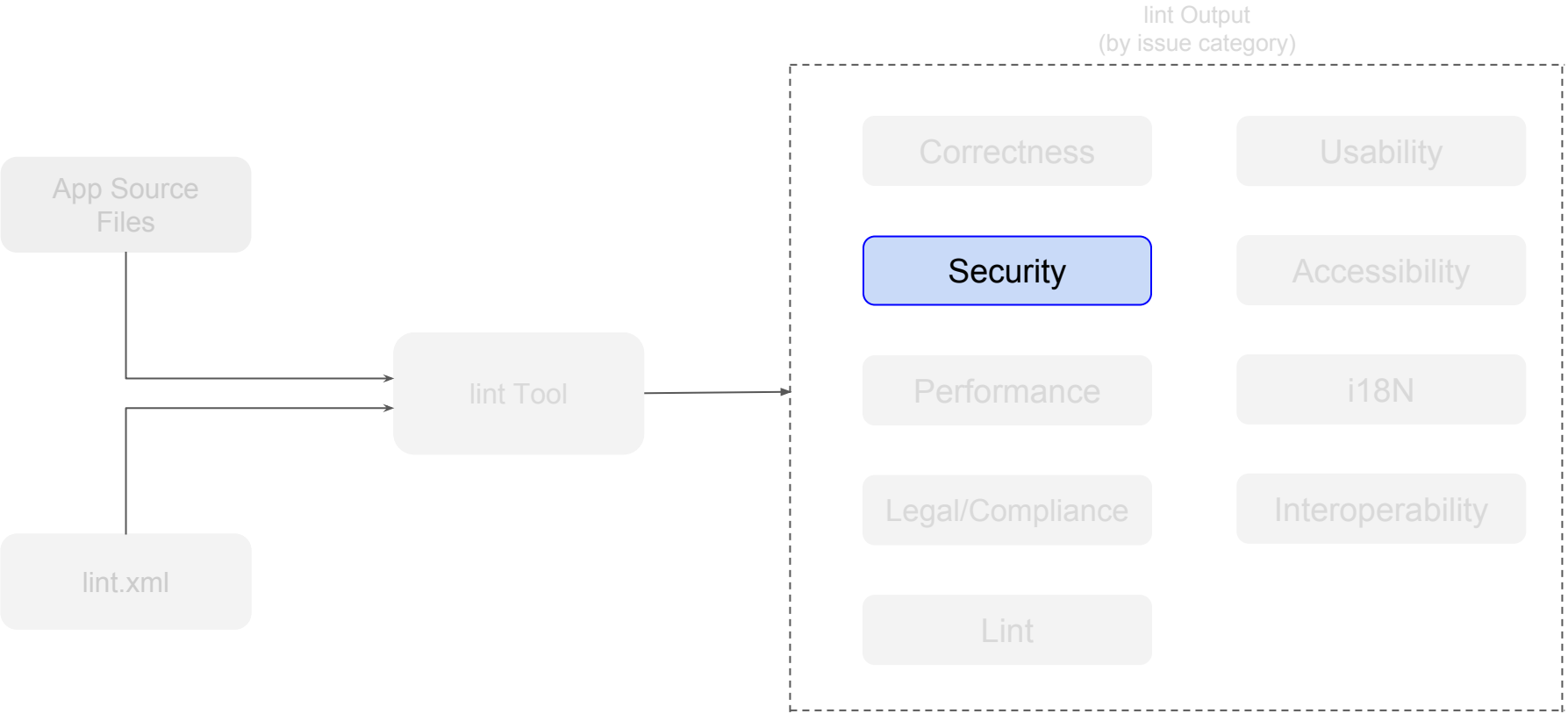
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

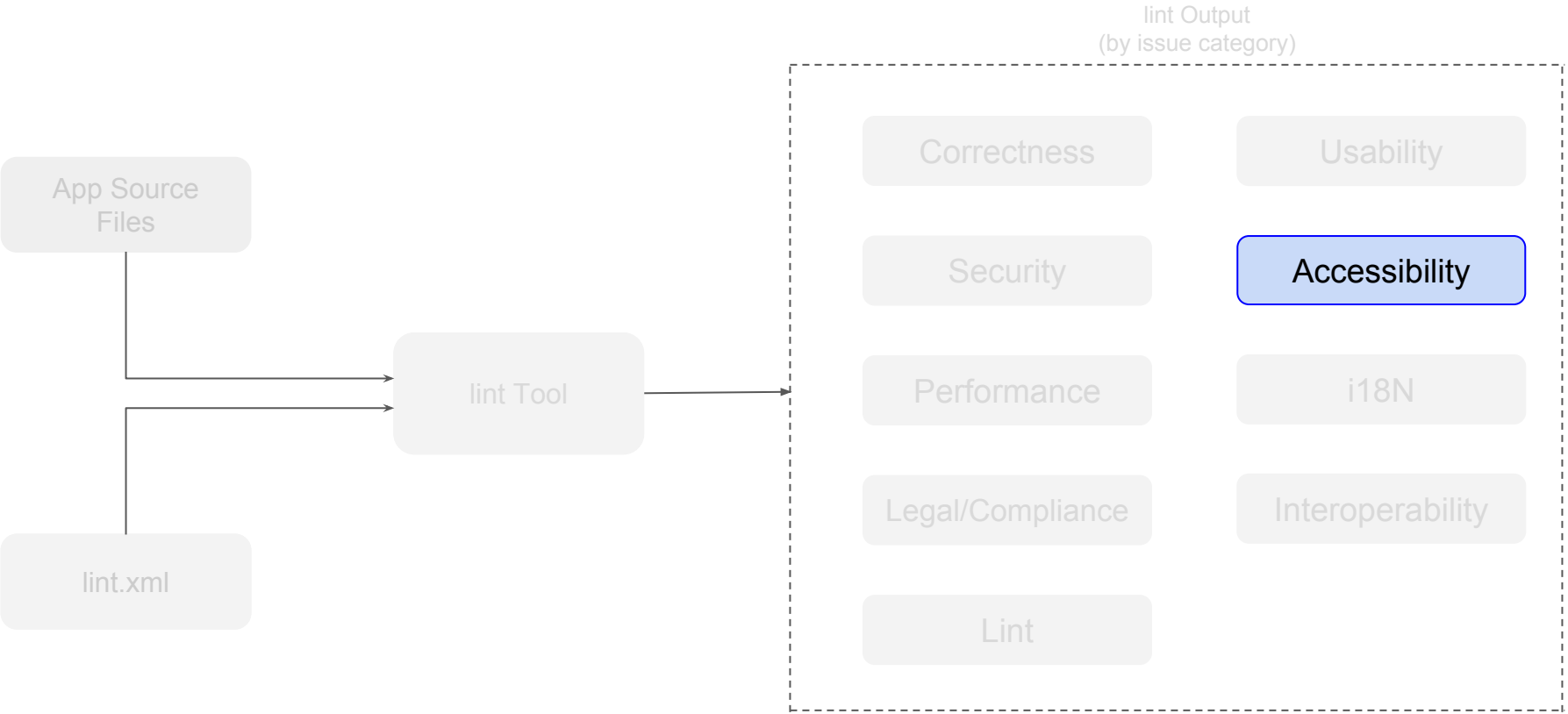
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

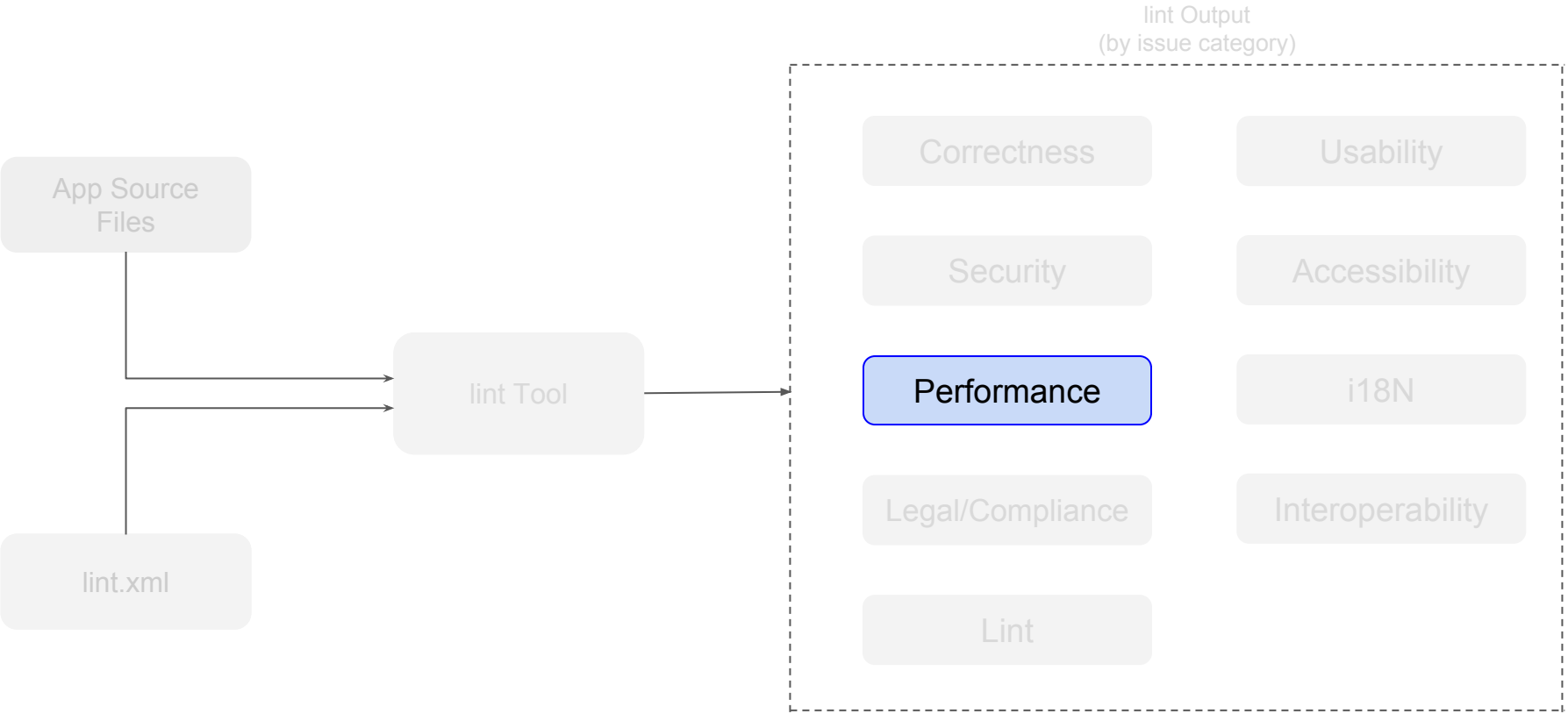
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

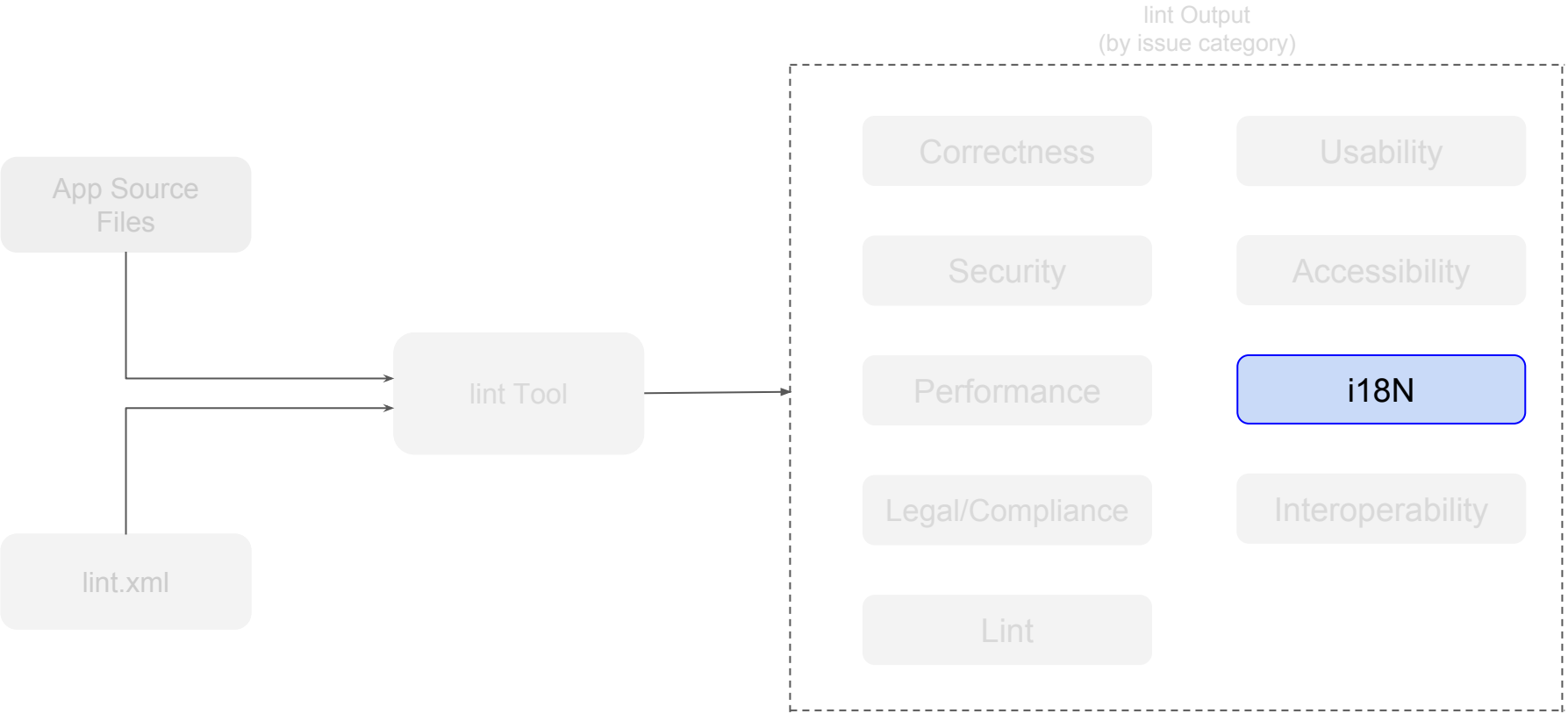
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

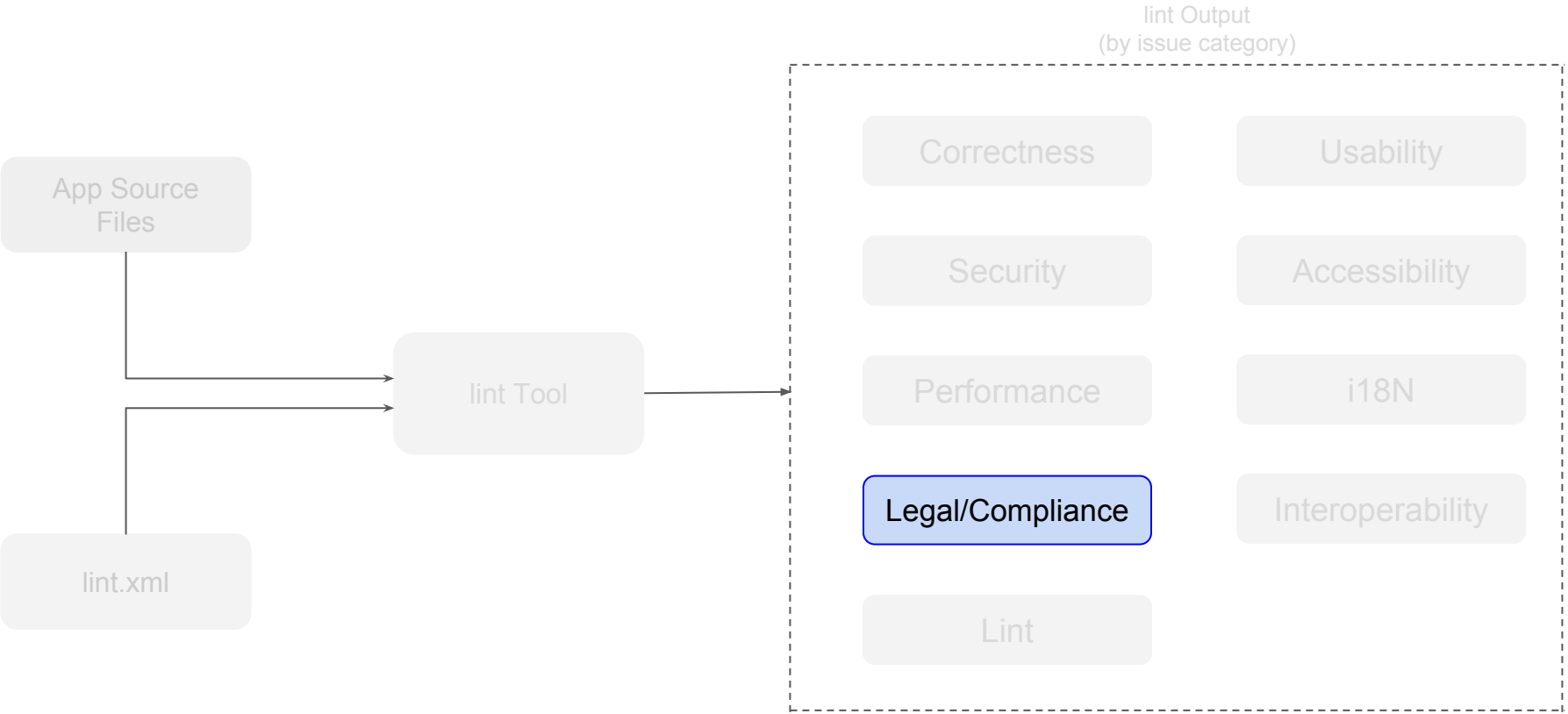
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

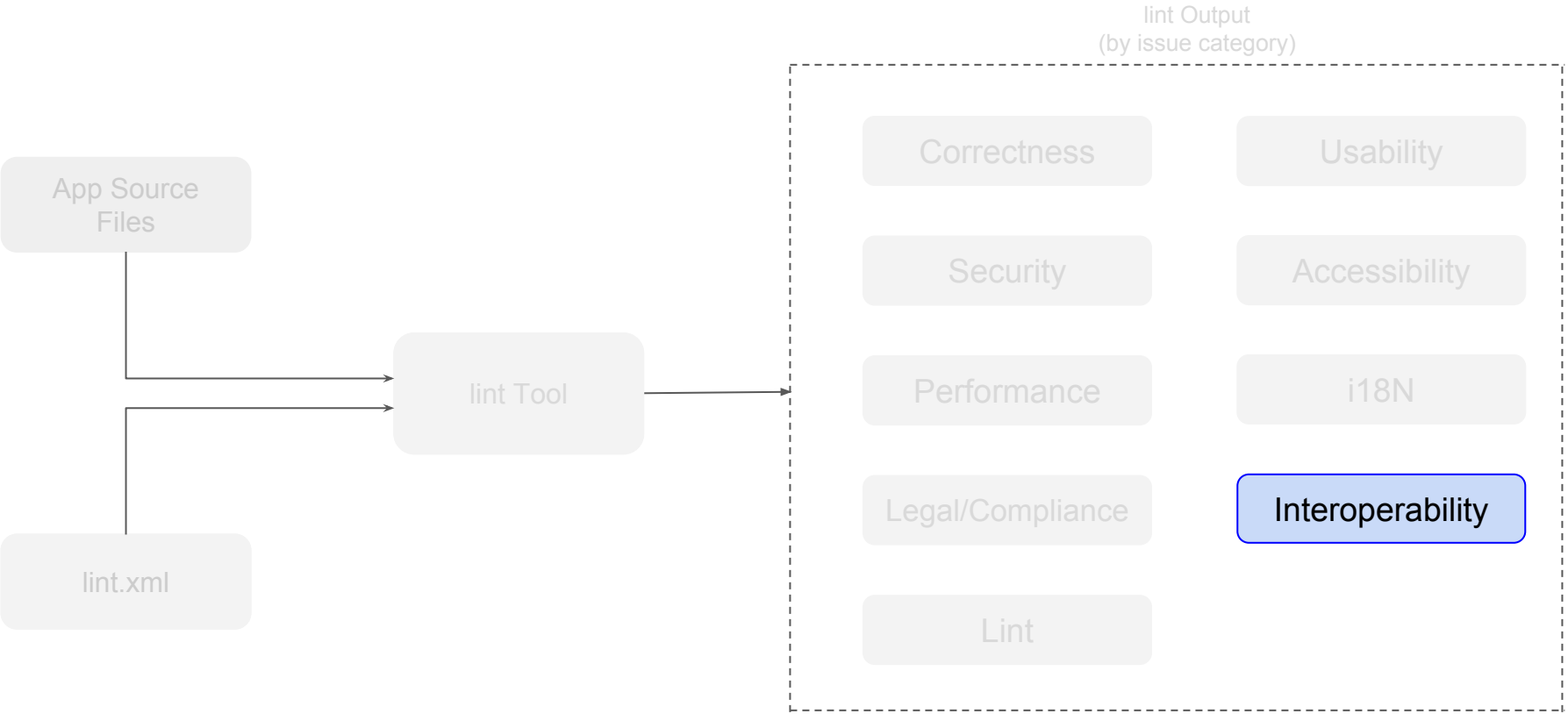
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

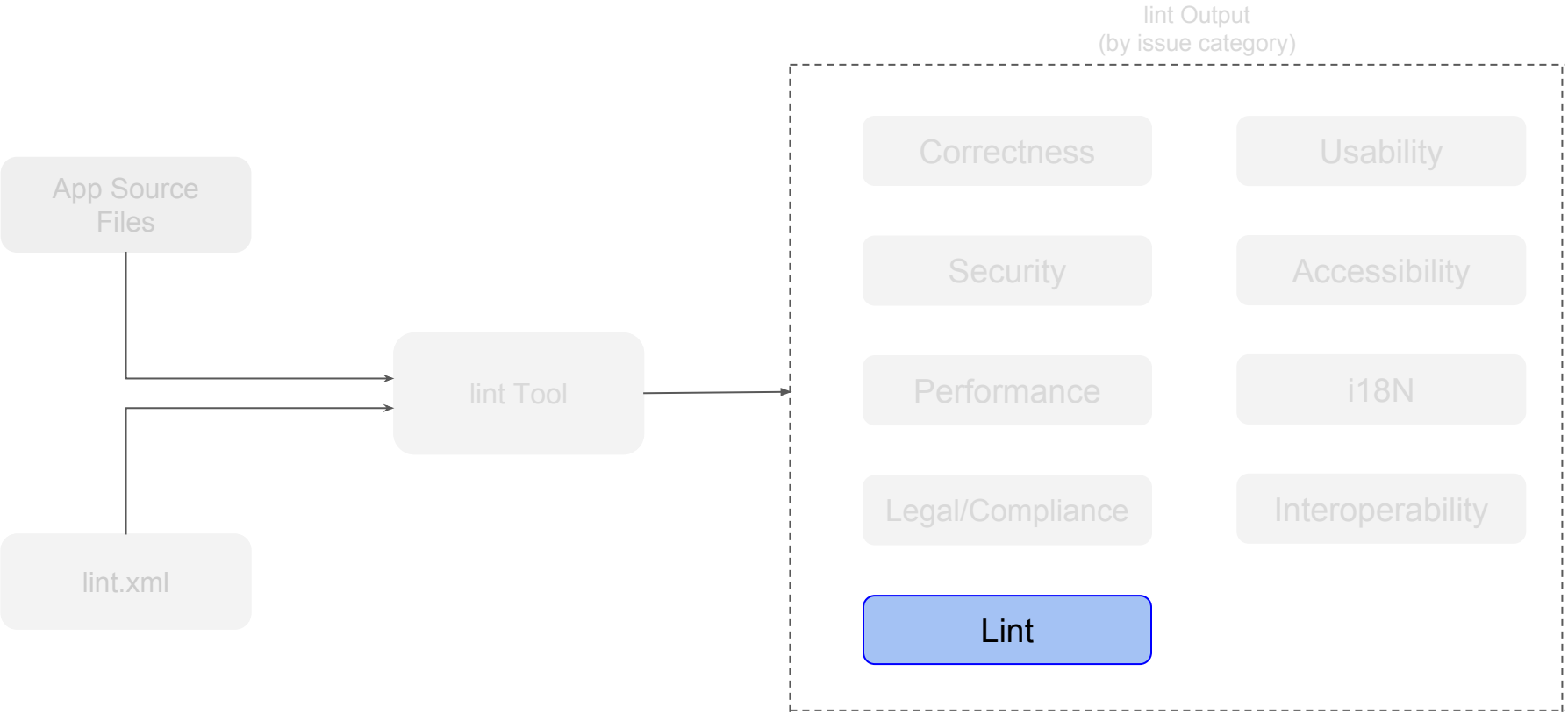
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

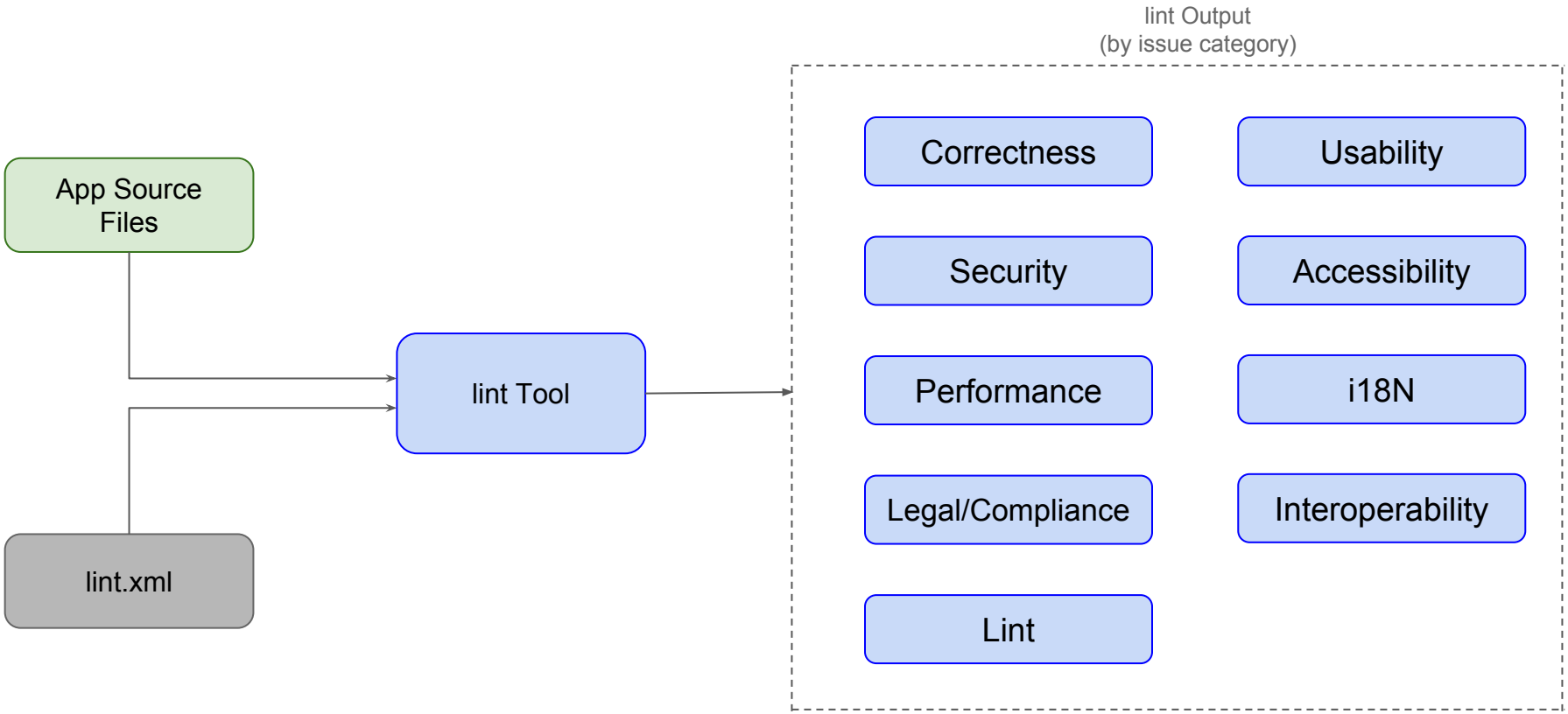
What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

What is Lint?


App Source Files
lint.xml
lint Tool
lint Output
(by issue category)
Correctness
Usability
Security
Accessibility
Performance
i18N
Legal/Compliance
Interoperability
Lint

What is Lint?

Lint & Gradle

```
android {
  ...
  lintOptions {
    // Turns off checks for the issue IDs you specify.
    disable 'TypographyFractions','TypographyQuotes'
    // Turns on checks for the issue IDs you specify. These checks are in
    // addition to the default lint checks.
    enable 'RtlHardcoded','RtlCompat', 'RtlEnabled'
    // To enable checks for only a subset of issue IDs and ignore all others,
    // list the issue IDs with the 'check' property instead. This property overrides
    // any issue IDs you enable or disable using the properties above.
    check 'NewApi', 'InlinedApi'
    // If set to true, turns off analysis progress reporting by lint.
    quiet true
    // if set to true (default), stops the build if errors are found.
    abortOnError false
    // if true, only report errors.
    ignoreWarnings true
  }
}
```

Source: https://developer.android.com/studio/write/lint

What is Lint?

Execution

- Automatic execution while coding
- Manual execution from Android Studio
- Manual execution through the console

What is Lint?

- Automatic execution while coding

```
private fun render() {
    localizationManager.getString( key: "action_accept_")
}
```

What is Lint?

- Automatic execution while coding

```
private fun render() {
    localizationManager.getString( key: "action_accept_")
}
```

```
62          localizationManager.getString( key: "action_accept_")
The key "action_accept_" is not present in local JSON translation file. more... (⌘F1)
65      }
```

What is Lint?

- Automatic execution while coding

```
private fun render() {
    localizationManager.getString( key: "action_accept_")
}
```

```
62        localizationManager.getString( key: "action_accept_")
The key "action_accept_" is not present in local JSON translation file. more... (⌘F1)
65    }
```

```
@SuppressLint( ...value: "LocalizedString")
private fun render() {
    localizationManager.getString( key: "action_accept_")
}
```

What is Lint?

- Automatic execution while coding

```
<drivers.ontruck.ontruck.uicomponents.LocalizedTextView
    android:id="@+id/visibilityTitleTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:textAppearance="@style/TextAppearance.Header"
    app:localizedText="driver_detail_choose_dispatcher_" />
```

What is Lint?

- Automatic execution while coding

```
<drivers.ontruck.ontruck.uicomponents.LocalizedTextView
    android:id="@+id/visibilityTitleTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:textAppearance="@style/TextAppearance.Header"
    app:localizedText="driver_detail_choose_dispatcher_" />
```

```
<drivers.ontruck.ontruck.uicomponents.LocalizedTextView
    android:id="@+id/visibilityTitleTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:textAppearance="@style/TextAppearance.Header"
    app:localizedText="driver_detail_choose_dispatcher_" />

<Ra  The key "driver_detail_choose_dispatcher_" is not present in local JSON translation file. n
```

What is Lint?

- Automatic execution while coding

```
<drivers.ontruck.ontruck.uicomponents.LocalizedTextView
    android:id="@+id/visibilityTitleTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:textAppearance="@style/TextAppearance.Header"
    app:localizedText="driver_detail_choose_dispatcher_"
    tools:ignore="LocalizedString" />
```

```
<drivers.ontruck.ontruck.uicomponents.LocalizedTextView
    android:id="@+id/visibilityTitleTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:textAppearance="@style/TextAppearance.Header"
    app:localizedText="driver_detail_choose_dispatcher_" />

The key "driver_detail_choose_dispatcher_" is not present in local JSON translation file.
```

What is Lint?

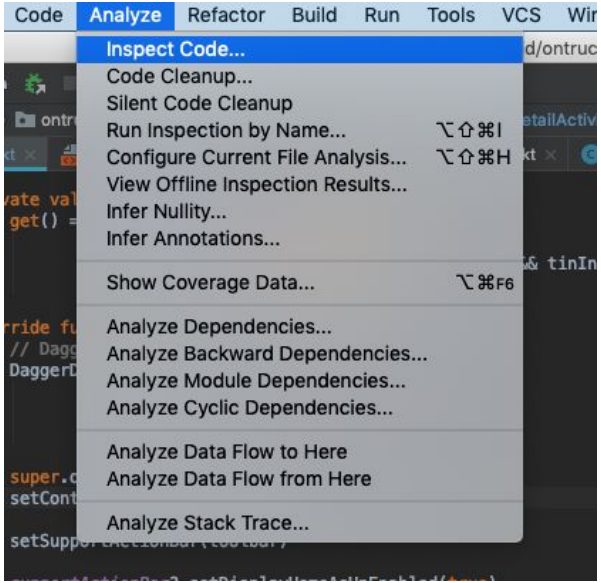
- Automatic execution while coding

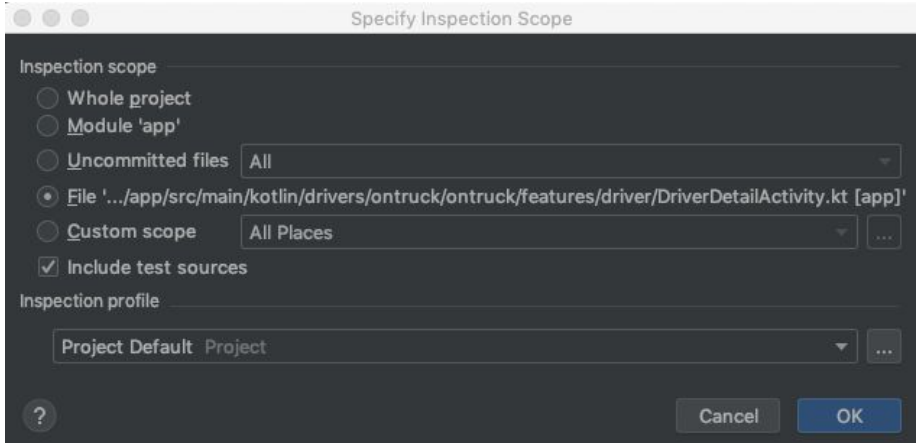
```
<drivers.ontruck.ontruck.uicomponents.LocalizedTextView
    android:id="@+id/visibilityTitleTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:textAppearance="@style/TextAppearance.Header"
    app:localizedText="driver_detail_choose_dispatcher_"
    tools:ignore="LocalizedString" />
```

```
<drivers.ontruck.ontruck.uicomponents.LocalizedTextView
    android:id="@+id/visibilityTitleTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:textAppearance="@style/TextAppearance.Header"
    app:localizedText="driver_detail_choose_dispatcher_" />

The key "driver_detail_choose_dispatcher_" is not present in local JSON translation file.
```

What is Lint?

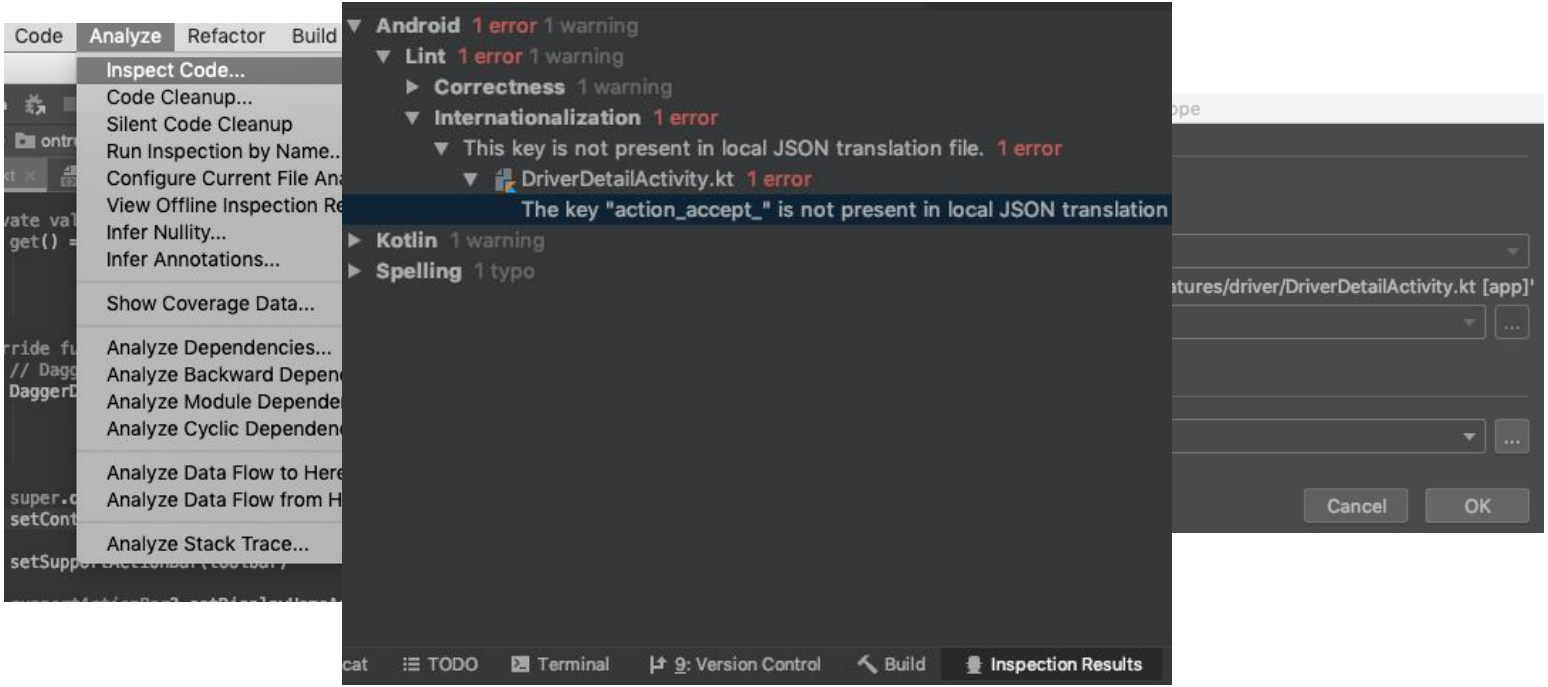
- Manual execution from Android Studio





What is Lint?

- Manual execution from Android Studio



What is Lint?

- Manual execution through the console

```
MacBook-Pro-de-Jose-Maria-Perez:ontruck josemariaperez$ ./gradlew lint

> Configure project :app
```

What is Lint?

- Manual execution through the console

```
MacBook-Pro-de-Jose-Maria-Perez:ontruck josemariaperez$ ./gradlew lint

> Configure project :app
```

```
Wrote HTML report to file:///Users/josemariaperez/Proyectos/Android/ontruck/app/build/reports/lint-results-developDebug.html
Wrote XML report to file:///Users/josemariaperez/Proyectos/Android/ontruck/app/build/reports/lint-results-developDebug.xml
```

What is Lint?

- Manual execution through the console

```
MacBook-Pro-de-Jose-Maria-Perez:ontruck josemariaperez$ ./gradlew lint
> Configure project :app
```

Lint Report: 30 errors and 294 warnings

```
Wrote HTML report to file:///Users/ ports/lint-results-developDebug.html
Wrote XML report to file:///Users/ orts/lint-results-developDebug.xml
```

Overview

Lint	
1	ObsoleteLintCustomCheck: Obsolete custom lint check
Correctness	
1	ScrollViewSize: ScrollView size validation
2	DefaultLocale: Implied default locale in case conversion
6	MissingConstraints: Missing Constraints in ConstraintLayout
1	NewApi: Calling new methods on older versions
1	OldTargetApi: Target SDK attribute is not targeting latest version

Our use case at **Ontruck**

Lint at Ontruck

Context

```
private void foo(){
    getString(R.string.app_name);
}
```

```
<androidx.appcompat.widget.AppCompatTextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/app_name"/>
```

Lint at Ontruck

Context

Lint at Ontruck

Context

```
<drivers.ontruck.ontruck.uicomponents.LocalizedTextView
    android:id="@+id/capacityLabel"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textAppearance="@style/TextAppearance.Header"
    app:localizedText="vehicle_detail_capacity_section" />
```

```
@Override
public void showLogoutProgress() {
    logoutProgressDialog = ProgressDialog.show( context: this,
            localizationManager.getString( key: "logout_loader_title"),
            localizationManager.getString( key: "logout_loader_message"),
            indeterminate: true,
            cancelable: false);
}
```

Problems

- Localization file sync is not automatic
- Sometimes there are keys missing from one language file - pending to translate
- Sometimes keys are removed from Lokalise but are still used in the apps

Lint at Ontruck

Solution, create a Lint rule that warns us when:

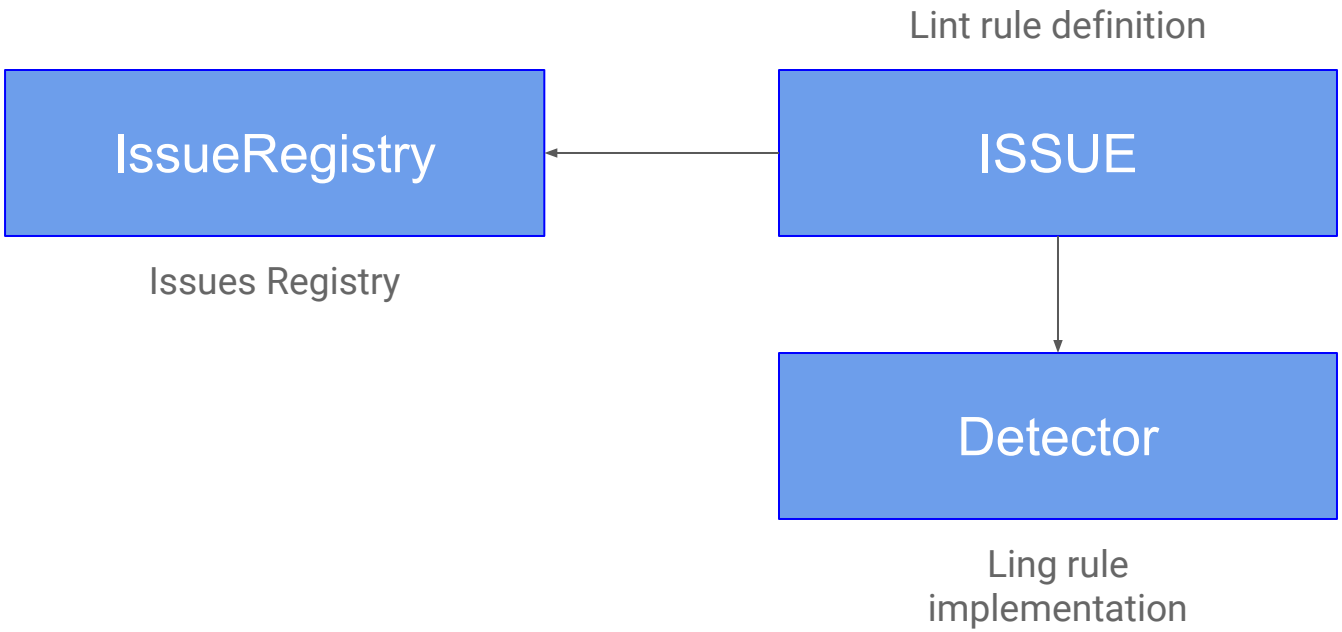
- We are using, from **code**, a key that is not present in the localization files (json)
- We are using, in a **xml file**, a key that is not present in the localization files (json)

Show me the Code

Custom Lint Rule

Custom Lint Rule: Localization

Lint: Scheme Class



Custom Lint Rule: Localization

IssueRegistry

```
class LintIssuesRegistry : IssueRegistry() {

    override val issues: List<Issue>
        get() = object : ArrayList<Issue>() {
            init {
                add(LocalizedStringDetector.ISSUE)
                add(LocalizedLayoutDetector.ISSUE)
            }
        }
```

Issue registry

Custom Lint Rule: Localization

IssueRegistry

```
class LintIssuesRegistry : IssueRegistry() {

    override val issues: List<Issue>
        get() = object : ArrayList<Issue>() {
            init {
                add(LocalizedStringDetector.ISSUE)
                add(LocalizedLayoutDetector.ISSUE)
            }
        }
```

Issue registry

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key is not present in local JSON translation file.
        LINT_EXPLANATION, // = The key you are setting is not present in local....
        I18N, // Category
        LINT_PRIORITY, //Priority: Current value is 10.
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class.java,
                Scope.JAVA_FILE_SCOPE)) // Implementation: Logic and Scope
```

Issue file Java and Kt

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key is not present in local JSON translation file.
        LINT_EXPLANATION, // = The key you are setting is not present in local....
        I18N, // Category
        LINT_PRIORITY, //Priority: Current value is 10.
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class.java,
                Scope.JAVA_FILE_SCOPE)) // Implementation: Logic and Scope
```

Issue file Java and Kt

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key is not present in local JSON translation file.
        LINT_EXPLANATION, // = The key you are setting is not present in local....
        I18N, // Category
        LINT_PRIORITY, //Priority: Current value is 10.
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class.java,
                Scope.JAVA_FILE_SCOPE)) // Implementation: Logic and Scope
```

Issue file Java and Kt

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key is not present in local JSON translation file.
        LINT_EXPLANATION, // = The key you are setting is not present in local....
        I18N, // Category
        LINT_PRIORITY, //Priority: Current value is 10.
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class.java,
                Scope.JAVA_FILE_SCOPE)) // Implementation: Logic and Scope
```

Issue file Java and Kt

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key is not present in local JSON translation file.
        LINT_EXPLANATION, // = The key you are setting is not present in local....
        I18N, // Category
        LINT_PRIORITY, //Priority: Current value is 10.
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class.java,
                Scope.JAVA_FILE_SCOPE)) // Implementation: Logic and Scope
```

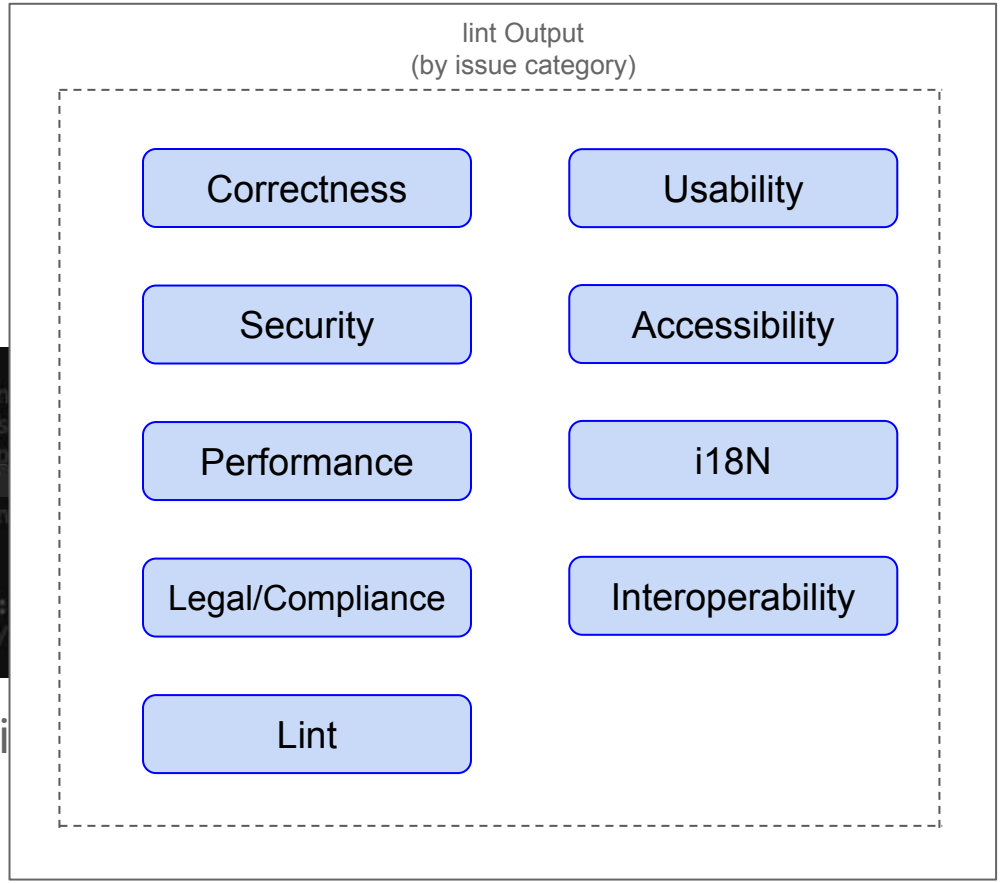
Issue file Java and Kt

Custom Lint Rule: Localization

Issue

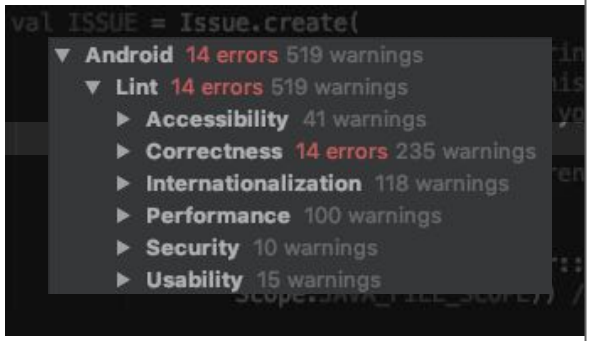
```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedStrin
        LINT_BRIEF_DESCRIPTION, // = This
        LINT_EXPLANATION, // = The key yo
        I18N, // Category
        LINT_PRIORITY, //Priority: Curren
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::
                Scope.JAVA_FILE_SCOPE)) /
```

Issue fi

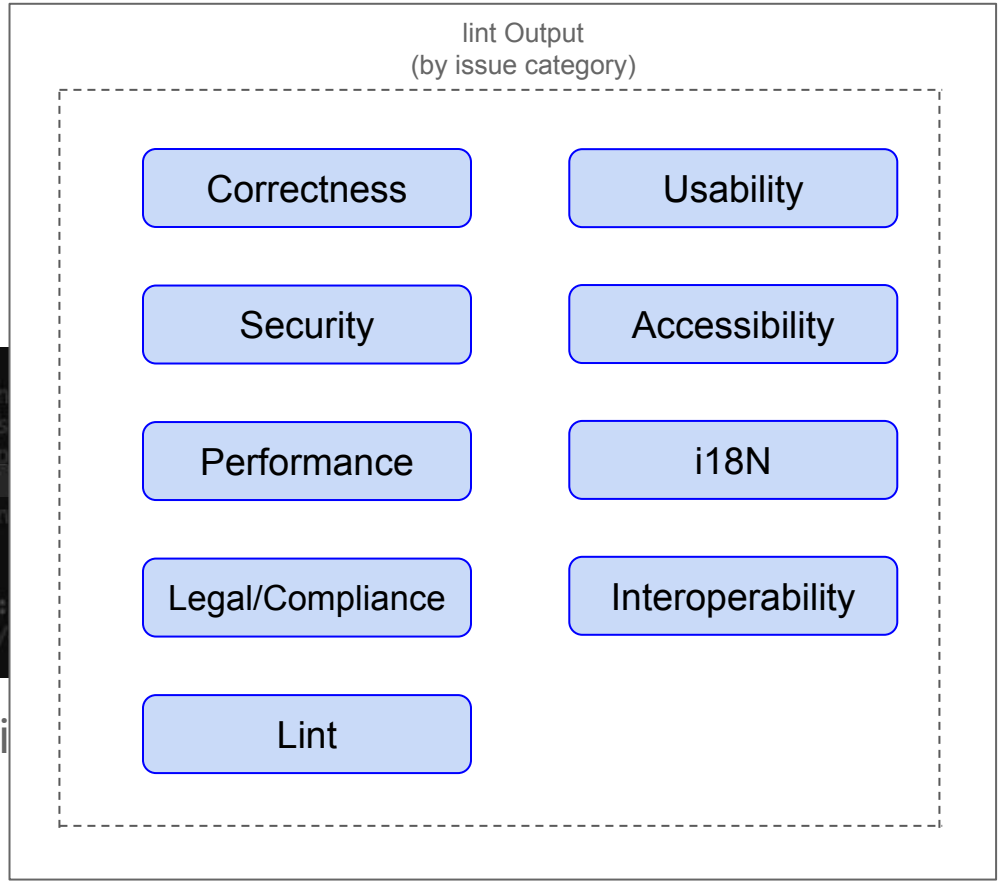


Custom Lint Rule: Localization

Issue



Issue fi



Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key is not present in local JSON translation file.
        LINT_EXPLANATION, // = The key you are setting is not present in local....
        I18N, // Category
        LINT_PRIORITY, //Priority: Current value is 10.
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class.java,
                Scope.JAVA_FILE_SCOPE)) // Implementation: Logic and Scope
```

Issue file Java and Kt

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key is not present in local JSON translation file.
        LINT_EXPLANATION, // = The key you are setting is not present in local....
        I18N, // Category
        LINT_PRIORITY, //Priority: Current value is 10.
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class.java,
                Scope.JAVA_FILE_SCOPE)) // Implementation: Logic and Scope
```

Issue file Java and Kt

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key
        LINT_EXPLANATION, // = The key you are
        I18N, // Category
        LINT_PRIORITY, //Priority: Current val
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class
                Scope.JAVA_FILE_SCOPE)) // Imp
```

Issue file J

```
/**
 * Fatal: Use sparingly because a warning marked as fatal will be
 * considered critical and will abort Export APK etc in ADT
 */
FATAL( description: "Fatal"),

/**
 * Errors: The issue is known to be a real error that must be addressed.
 */
ERROR( description: "Error"),

/**
 * Warning: Probably a problem.
 */
WARNING( description: "Warning"),

/**
 * Information only: Might not be a problem, but the check has found
 * something interesting to say about the code.
 */
INFORMATIONAL( description: "Information"),

/**
 * Ignore: The user doesn't want to see this issue
 */
IGNORE( description: "Ignore");
```

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE, // = LocalizedString
        LINT_BRIEF_DESCRIPTION, // = This key is not present in local JSON translation file.
        LINT_EXPLANATION, // = The key you are setting is not present in local....
        I18N, // Category
        LINT_PRIORITY, //Priority: Current value is 10.
        Severity.FATAL, //Severity
        Implementation(
                LocalizedStringDetector::class.java,
                Scope.JAVA_FILE_SCOPE)) // Implementation: Logic and Scope
```

Issue file Java and Kt

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE,
        LINT_BRIEF_DESCRIPTION,
        LINT_EXPLANATION,
        I18N,
        LINT_PRIORITY,
        Severity.FATAL,
        Implementation(
                LocalizedLayoutDetector::class.java,
                RESOURCE_FILE_SCOPE)
)
```

Issue file .xml

Custom Lint Rule: Localization

Issue

```
val ISSUE = Issue.create(
        ID_LINT_RULE,
        LINT_BRIEF_DESCRIPTION,
        LINT_EXPLANATION,
        I18N,
        LINT_PRIORITY,
        Severity.FATAL,
        Implementation(
                LocalizedLayoutDetector::class.java,
                RESOURCE_FILE_SCOPE)
)
```

Issue file .xml

Custom Lint Rule: Localization

Detector

LocalizedLayoutDetector

Files .xml

LocalizedStringDetector

Files java/kt

Custom Lint Rule: Localization

Detector

```
val ATTR_LOCALIZED = listOf("localizedText", "localizedTitle", "localizedHint", "localizedMessage")

class LocalizedLayoutDetector : LayoutDetector() {

    private val assetHandler = AssetHandler()

    override fun getApplicableAttributes() = ATTR_LOCALIZED

    override fun visitAttribute(context: XmlContext, attribute: Attr) {
        if (!assetHandler.existKeyInJson(context, attribute.value)) {
            context.report(
                    ISSUE,
                    attribute,
                    context.getValueLocation(attribute),
                    message: "The key \"${attribute.value}\" is not present in local JSON translation file.")
        }
    }

    companion object {...}
}
```

Files .xml

Custom Lint Rule: Localization

Detector

```
val ATTR_LOCALIZED = listOf("localizedText", "localizedTitle", "localizedHint", "localizedMessage")

class LocalizedLayoutDetector : LayoutDetector() {

    private val assetHandler = AssetHandler()

    override fun getApplicableAttributes() = ATTR_LOCALIZED

    override fun visitAttribute(context: XmlContext, attribute: Attr) {
        if (!assetHandler.existKeyInJson(context, attribute.value)) {
            context.report(
                    ISSUE,
                    attribute,
                    context.getValueLocation(attribute),
                    message: "The key \"${attribute.value}\" is not present in local JSON translation file.")
        }
    }

    companion object {...}
}
```

Files .xml

Custom Lint Rule: Localization

Detector

```
val ATTR_LOCALIZED = listOf("localizedText", "localizedTitle", "localizedHint", "localizedMessage")

class LocalizedLayoutDetector : LayoutDetector() {

    private val assetHandler = AssetHandler()

    override fun getApplicableAttributes() = ATTR_LOCALIZED

    override fun visitAttribute(context: XmlContext, attribute: Attr) {
        if (!assetHandler.existKeyInJson(context, attribute.value)) {
            context.report(
                    ISSUE,
                    attribute,
                    context.getValueLocation(attribute),
                    message: "The key \"${attribute.value}\" is not present in local JSON translation file.")
        }
    }

    companion object {...}
}
```

Files .xml

Custom Lint Rule: Localization

Detector

```kotlin
class LocalizedStringDetector : Detector(), Detector.UastScanner {

    private val assetHandler = AssetHandler()

    override fun getApplicableMethodNames(): List<String>? {
        return listOf("getString")
    }

    override fun visitMethodCall(context: JavaContext, node: UCallExpression, method: PsiMethod) {
        if (context.evaluator.isMemberInClass(method, NAME_CLASS_LOCALIZATION)) {
            val key = node.valueArguments[0].asSourceString().replace( oldValue: "\"",  newValue: "")
            if (!assetHandler.existKeyInJson(context, key)) {
                context.report(
                        ISSUE,
                        node,
                        context.getLocation(node),
                         message: "The key \"$key\" is not present in local JSON translation file.")
            }
            return
        }
    }

    companion object {...}
}
```

Files java/kt

Custom Lint Rule: Localization

Detector

```kotlin
class LocalizedStringDetector : Detector(), Detector.UastScanner {

    private val assetHandler = AssetHandler()

    override fun getApplicableMethodNames(): List<String>? {
        return listOf("getString")
    }

    override fun visitMethodCall(context: JavaContext, node: UCallExpression, method: PsiMethod) {
        if (context.evaluator.isMemberInClass(method, NAME_CLASS_LOCALIZATION)) {
            val key = node.valueArguments[0].asSourceString().replace( oldValue: "\"",  newValue: "")
            if (!assetHandler.existKeyInJson(context, key)) {
                context.report(
                        ISSUE,
                        node,
                        context.getLocation(node),
                         message: "The key \"$key\" is not present in local JSON translation file.")
            }
            return
        }
    }

    companion object {...}
}
```

Files java/kt

Custom Lint Rule: Localization

Detector

```kotlin
class LocalizedStringDetector : Detector(), Detector.UastScanner {

    private val assetHandler = AssetHandler()

    override fun getApplicableMethodNames(): List<String>? {
        return listOf("getString")
    }

    override fun visitMethodCall(context: JavaContext, node: UCallExpression, method: PsiMethod) {
        if (context.evaluator.isMemberInClass(method, NAME_CLASS_LOCALIZATION)) {
            val key = node.valueArguments[0].asSourceString().replace( oldValue: "\"",  newValue: "")
            if (!assetHandler.existKeyInJson(context, key)) {
                context.report(
                        ISSUE,
                        node,
                        context.getLocation(node),
                         message: "The key \"$key\" is not present in local JSON translation file.")
            }
            return
        }
    }

    companion object {...}
}
```

Files java/kt

Custom Lint Rule: Localization

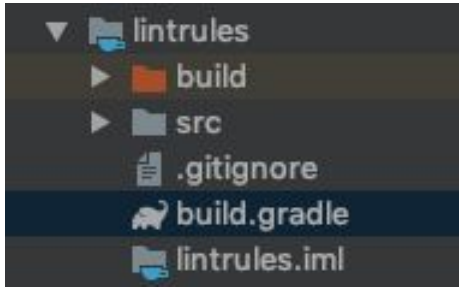
Detector

```kotlin
class LocalizedStringDetector : Detector(), Detector.UastScanner {

    private val assetHandler = AssetHandler()

    const val NAME_CLASS_LOCALIZATION = "drivers.ontruck.ontruck.localization.LocalizationManager"
    }

    override fun visitMethodCall(context: JavaContext, node: UCallExpression, method: PsiMethod) {
        if (context.evaluator.isMemberInClass(method, NAME_CLASS_LOCALIZATION)) {
            val key = node.valueArguments[0].asSourceString().replace( oldValue: "\"", newValue: "")
            if (!assetHandler.existKeyInJson(context, key)) {
                context.report(
                        ISSUE,
                        node,
                        context.getLocation(node),
                        message: "The key \"$key\" is not present in local JSON translation file.")
            }
            return
        }
    }

    companion object {...}
}
```

Files java/kt

Custom Lint Rule: Localization

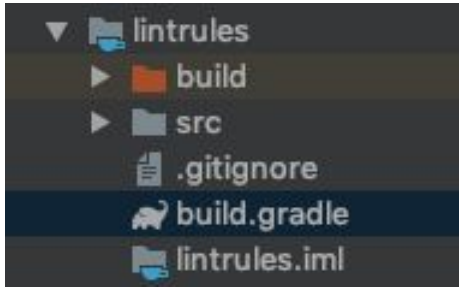
Module LintRules




```
apply plugin: 'java-library'
apply plugin: 'kotlin'

targetCompatibility = '1.8'
sourceCompatibility = '1.8'

repositories {
    jcenter()
}

dependencies {
    compileOnly 'com.android.tools.lint:lint-api:26.3.2'
    compileOnly 'com.android.tools.lint:lint-checks:26.3.2'
    testImplementation 'com.android.tools.lint:lint:26.3.2'
    testImplementation 'com.android.tools.lint:lint-tests:26.3.2'
}

jar {
    manifest {
        attributes 'Lint-Registry-v2': 'drivers.ontruck.ontruck.lintrules.LintIssuesRegistry'
    }
}
```

Custom Lint Rule: Idiomas

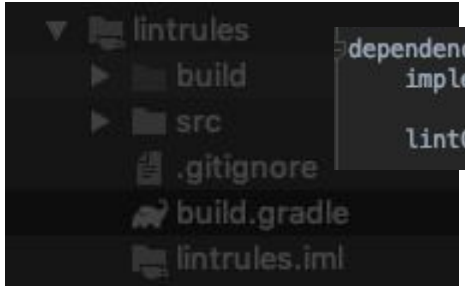
Module LintRules




```
apply plugin: 'java-library'
apply plugin: 'kotlin'

targetCompatibility = '1.8'
sourceCompatibility = '1.8'

repositories {
    jcenter()
}

dependencies {
    compileOnly 'com.android.tools.lint:lint-api:26.3.2'
    compileOnly 'com.android.tools.lint:lint-checks:26.3.2'
    testImplementation 'com.android.tools.lint:lint:26.3.2'
    testImplementation 'com.android.tools.lint:lint-tests:26.3.2'
}

jar {
    manifest {
        attributes 'Lint-Registry-v2': 'drivers.ontruck.ontruck.lintrules.LintIssuesRegistry'
    }
}
```

Custom Lint Rule: Idiomas

Module LintRules




```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])

    lintChecks project(':lintrules')
```

```
apply plugin: 'java-library'
apply plugin: 'kotlin'

targetCompatibility = '1.8'
sourceCompatibility = '1.8'

repositories {

    testImplementation 'com.android.tools.lint:lint:26.3.2'
    testImplementation 'com.android.tools.lint:lint-tests:26.3.2'
}

jar {
    manifest {
        attributes 'Lint-Registry-v2': 'drivers.ontruck.ontruck.lintrules.LintIssuesRegistry'
    }
}
```

Testing
Custom Lint Rule

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="keyNotFound" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expect("""
              |res/layout/layouts.xml:6: Error: The key "keyNotFound" is not present in local
              |     app:localizedText="keyNotFound" />
              |                        ~~~~~~~~~~~~
              |1 errors, 0 warnings""".trimMargin())
}
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    lint()
```

```
@Test
fun shouldShowErrorWhenKeyNotFound() {

    lint()
            .files(xml( to: "res/layout/layouts.xml", getFile( filePath: "layout_key_not_found.xml")).indented(),
                    jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expect(getFile( filePath: "assert_result_layout_key_not_found.txt").trimMargin())
}
```

```
|res/layout/layouts.xml:6: Error: The key "keyNotFound" is not present in local
|    app:localizedText="keyNotFound" />
|                       ~~~~~~~~~~~~
|1 errors, 0 warnings""".trimMargin())
}
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="keyNotFound" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expect("""
              |res/layout/layouts.xml:6: Error: The key "keyNotFound" is not present in local
              |     app:localizedText="keyNotFound" />
              |                        ~~~~~~~~~~~~
              |1 errors, 0 warnings""".trimMargin())
}
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="keyNotFound" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expect("""
              |res/layout/layouts.xml:6: Error: The key "keyNotFound" is not present in local
              |     app:localizedText="keyNotFound" />
              |                        ~~~~~~~~~~~
              |1 errors, 0 warnings""".trimMargin())
}
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="keyNotFound" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expect("""
              |res/layout/layouts.xml
              |     app:localizedText=
              |
              |1 errors, 0 warnings""".trimMargin())
}
```

```
private fun jsonTestFile(): TestFile {
    return source(
            DEFAULT_LOCALIZATION_JSON_RELATIVE_PATH,
            source: "{ \"key1\": \"value1\", \"key2\": \"value2\" }")
}
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="keyNotFound" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expect("""
              |res/layout/layouts.xml:6: Error: The key "keyNotFound" is not present in local
              |     app:localizedText="keyNotFound" />
              |                        ~~~~~~~~~~~
              |1 errors, 0 warnings""".trimMargin())
}
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="keyNotFound" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expect("""
              |res/layout/layouts.xml:6: Error: The key "keyNotFound" is not present in local
              |     app:localizedText="keyNotFound" />
              |                        ~~~~~~~~~~~~
              |1 errors, 0 warnings""".trimMargin())
}
```

FAIL

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="keyNotFound" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expect("""
                |res/layout/layouts.xml:6: Error: The key "keyNotFound" is not present in local
```

```
res/layout/layouts.xml:5: Error: The key "keyNotFound" is not present in local JSON translation file.
res/layout/layouts.xml:6: Error: The key "keyNotFound" is not present in local JSON translation file.
    app:localizedText="keyNotFound" />
                       ~~~~~~~~~~~
1 errors, 0 warnings
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowNotWarningsWhenKeyIsFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="key1" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expectClean()
}
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowNotWarningsWhenKeyIsFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="key1" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expectClean()
}
```

```
private fun jsonTestFile(): TestFile {
    return source(
            DEFAULT_LOCALIZATION_JSON_RELATIVE_PATH,
            source: "{ \"key1\": \"value1\", \"key2\": \"value2\" }")
}
```

XML: Testing Custom Lint Rule

```
@Test
fun shouldShowNotWarningsWhenKeyIsFound() {
    lint()
            .files(xml( to: "res/layout/layouts.xml",  source: """
        <drivers.ontruck.ontruck.uicomponents.LocalizedTextView
            xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:localizedText="key1" />""").indented(), jsonTestFile())
            .issues(LocalizedLayoutDetector.ISSUE)
            .run()
            .expectClean()
}
```

PASS

Actual

```
1    No warnings.
```

Java&Kt: Testing Custom Lint Rule

```
private val LOCALIZATION_MANAGER_STUB = java("""
  |package drivers.ontruck.ontruck.localization;
  |public class LocalizationManager {
  |   public String getString(@LocalizedString String key) {
  |       return localizationJsonObject.get(key).getAsString();
  |   }
  |}""".trimMargin())
```

Java&Kt: Testing Custom Lint Rule

```
private val LOCALIZATION_MANAGER_STUB = java("""
  |package drivers.ontruck.ontruck.localization;
  |public class LocalizationManager {
  |   public String getString(@LocalizedString String key) {
  |       return localizationJsonObject.get(key).getAsString();
  |   }
  |}""".trimMargin())
```

```
private val LOCALIZATION_MANAGER_STUB =
        java(getFile( filePath: "LocalizationManager.java").trimMargin())
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                        |package foo;
                        |
                        |import drivers.ontruck.ontruck.localization.LocalizationManager;
                        |
                        |public class TestClass {
                        |
                        |  public void foo(LocalizationManager localization) {
                        |    localization.getString("keyNotFound");
                        |  }
                        |
                        |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect("""
                    |src/foo/TestClass.java:8: Error: The key "keyNotFound" is not present in
                    |    localization.getString("keyNotFound");
                    |    ~~~
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
```

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java(getFile( filePath: "TestClassKeyNotFound.java")
                    .trimMargin()).indented(),
                    jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect(getFile( filePath: "assert_result_class_key_not_found.txt").trimMargin())
}
```

```
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect("""
                    |src/foo/TestClass.java:8: Error: The key "keyNotFound" is not present in
                    |    localization.getString("keyNotFound");
                    |    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                    |1 errors, 0 warnings""".trimMargin())
}
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                        |package foo;
                        |
                        |import drivers.ontruck.ontruck.localization.LocalizationManager;
                        |
                        |public class TestClass {
                        |
                        |  public void foo(LocalizationManager localization) {
                        |    localization.getString("keyNotFound");
                        |  }
                        |
                        |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect("""
                    |src/foo/TestClass.java:8: Error: The key "keyNotFound" is not present in
                    |    localization.getString("keyNotFound");
                    |    ~~~
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                        |package foo;
                        |
                        |import drivers.ontruck.ontruck.localization.LocalizationManager;
                        |
                        |public class TestClass {
                        |
                        |  public void foo(LocalizationManager localization) {
                        |    localization.getString("keyNotFound");
                        |  }
                        |
                        |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect("""
                    |src/foo/TestClass.java:8: Error: The key "keyNotFound" is not present in
                    |    localization.getString("keyNotFound");
                    |    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                    |1 errors, 0 warnings""".trimMargin())
}
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                    |package foo;
                    |
                    |import drivers.ontruck.ontruck.localization.LocalizationManager;
                    |
                    |public class TestClass {
                    |
                    |  public void foo(LocalizationManager localization) {
                    |    localization.getString("keyNotFound");
                    |  }
                    |
                    |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect("""
                    |src/foo/TestClass.java:8: Error: The key "keyNotFound" is not present in
                    |    localization.getString("keyNotFound");
                    |    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                    |1 errors, 0 warnings""".trimMargin())
}
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                        |package foo;
                        |
                        |import drivers.ontruck.ontruck.localization.LocalizationManager;
                        |
                        |public class TestClass {
                        |
                        |  public void foo(LocalizationManager localization) {
                        |    localization.getString("keyNotFound");
                        |  }
                        |
                        |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect("""
                    |sr
                    |
                    |
                    |1 errors, 0 warnings""".trimMargin())
}
```

```
private fun jsonTestFile(): TestFile {
    return source(
            DEFAULT_LOCALIZATION_JSON_RELATIVE_PATH,
            source: "{ \"key1\": \"value1\", \"key2\": \"value2\" }")
}
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                        |package foo;
                        |
                        |import drivers.ontruck.ontruck.localization.LocalizationManager;
                        |
                        |public class TestClass {
                        |
                        |  public void foo(LocalizationManager localization) {
                        |    localization.getString("keyNotFound");
                        |  }
                        |
                        |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect("""
                    |src/foo/TestClass.java:8: Error: The key "keyNotFound" is not present in
                    |    localization.getString("keyNotFound");
                    |    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                    |1 errors, 0 warnings""".trimMargin())
}
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowErrorWhenKeyNotFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                        |package foo;
                        |
                        |import drivers.ontruck.ontruck.localization.LocalizationManager;
                        |
                        |public class TestClass {
                        |
                        |  public void foo(LocalizationManager localization) {
                        |    localization.getString("keyNotFound");
                        |  }
                        |
                        |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expect("""
                    |src/foo/TestClass.java:8: Error: The key "keyNotFound" is not present in
                    |    localization.getString("keyNotFound");
                    |    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                    |1 errors, 0 warnings""".trimMargin())
}
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowNotWarningsWhenKeyIsFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                        |package foo;
                        |
                        |import drivers.ontruck.ontruck.localization.LocalizationManager;
                        |
                        |public class TestClass {
                        |
                        |  public void foo(LocalizationManager localization) {
                        |    localization.getString("key1");
                        |  }
                        |
                        |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expectClean()
}
```

Java&Kt: Testing Custom Lint Rule

```
@Test
fun shouldShowNotWarningsWhenKeyIsFound() {
    TestLintTask
            .lint()
            .allowCompilationErrors()
            .files(LOCALIZATION_MANAGER_STUB, java("""
                        |package foo;
                        |
                        |import drivers.ontruck.ontruck.localization.LocalizationManager;
                        |
                        |public class TestClass {
                        |
                        |  public void foo(LocalizationManager localization) {
                        |    localization.getString("key1");
                        |  }
                        |
                        |}""".trimMargin()).indented(), jsonTestFile())
            .issues(LocalizedStringDetector.ISSUE)
            .run()
            .expectClean()
}
```

```
Actual
1    No warnings.
```

PASS

References

- Android Doc:
  https://developer.android.com/studio/write/lint
- Github Sophie Dankel:
  https://github.com/sophiedankel/Android-lint
- Github Jake Warthon:
  https://github.com/JakeWharton/timber/tree/master/timber-lint
- Android Tools Doc:
  http://tools.android.com/tips/lint-checks

Thanks!
Questions?