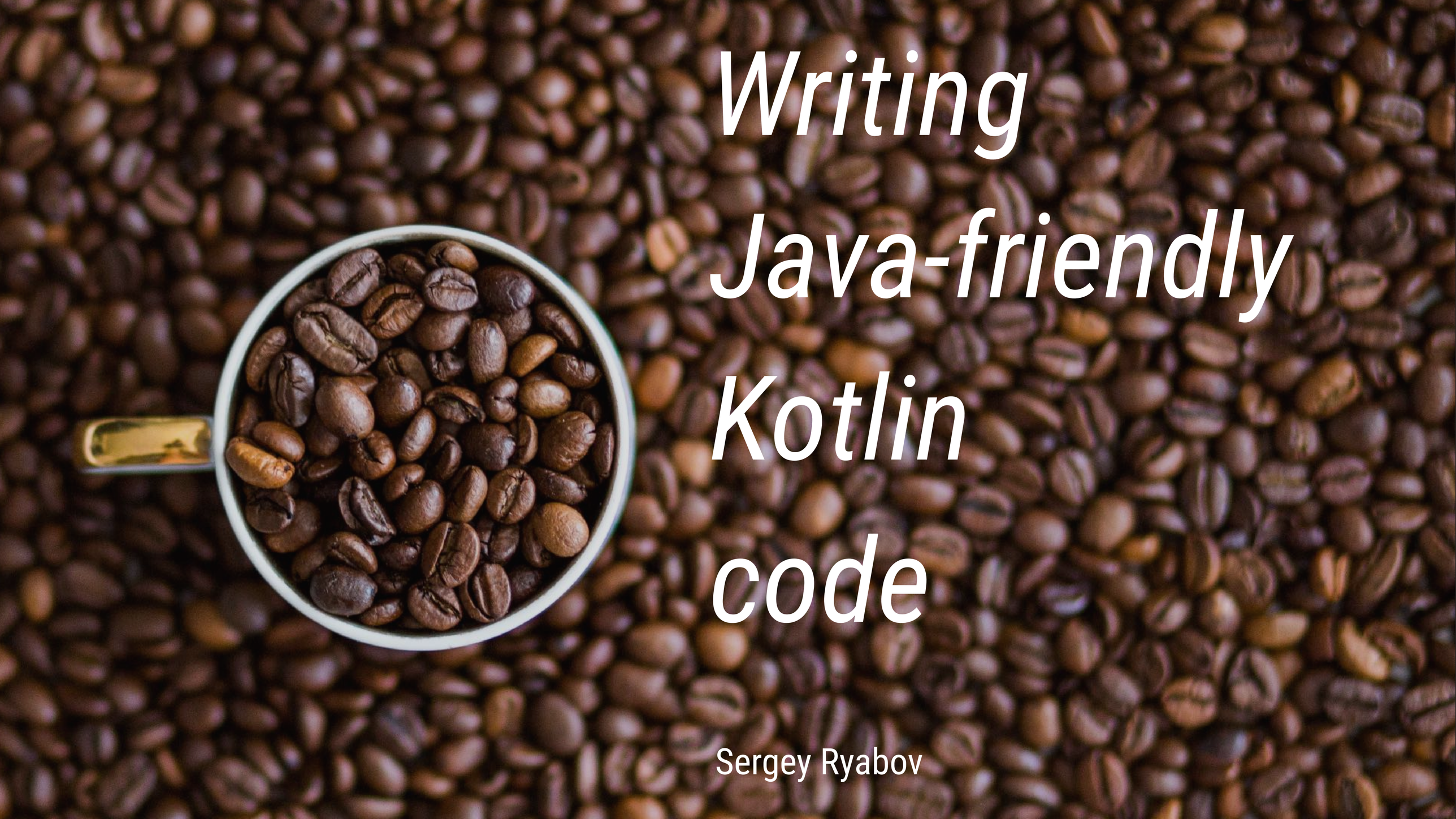


# Writing Java-friendly Kotlin code

Sergey Ryabov

```
compile 'rxbinding:x.y.x'
compile 'rxbinding-appcompat-v7:x.y.x'
compile 'rxbinding-design:x.y.x'

compile 'autodispose:x.y.z'
compile 'autodispose-android:x.y.z'
compile 'autodispose-android-archcomponents:x.y.z'
```

```
compile 'rxbinding:x.y.x'
compile 'rxbinding-kotlin:x.y.x'
compile 'rxbinding-appcompat-v7:x.y.x'
compile 'rxbinding-appcompat-v7-kotlin:x.y.x'
compile 'rxbinding-design:x.y.x'
compile 'rxbinding-design-kotlin:x.y.x'

compile 'autodispose:x.y.z'
compile 'autodispose-kotlin:x.y.z'
compile 'autodispose-android:x.y.z'
compile 'autodispose-android-kotlin:x.y.z'
compile 'autodispose-android-archcomponents:x.y.z'
compile 'autodispose-android-archcomponents-kotlin:x.y.z'
```

# JAVA

# JAVA

- **volatile**
- **synchronized**
- **strictfp**
- **transient**

# KOTLIN

- @Volatile
- @Synchronized
- @Strictfp
- @Transient

## KOTLIN

- @Volatile
- @Synchronized
- @Strictfp
- @Transient

```
package kotlin.jvm
```

# ANALYTICS ABSTRACTION LIBRARY

# ANALYTICS ABSTRACTION LIBRARY

```
object Analytics {
  fun send(event: Event) {}
  fun addPlugins(plugs: List<Plugin>) {}
  fun getPlugins(): List<Plugin> {}
}
```

# ANALYTICS ABSTRACTION LIBRARY

```
object Analytics {
  fun send(event: Event) {}
  fun addPlugins(plugs: List<Plugin>) {}
  fun getPlugins(): List<Plugin> {}
}

interface Plugin {
  fun init()
  fun send(event: Event)
  fun close()
}
```

# ANALYTICS ABSTRACTION LIBRARY

```
object Analytics {
  fun send(event: Event) {}
  fun addPlugins(plugs: List<Plugin>) {}
  fun getPlugins(): List<Plugin> {}
}

interface Plugin {
  fun init()
  fun send(event: Event)
  fun close()
}

data class Event(
    val name: String,
    val context: Map<String, Any> = emptyMap()
)
```


TIME FOR
SOME CODE!

# Generics

# VARIANCE

# VARIANCE

# VARIANCE

# VARIANCE

`ArrayList<Animal>`

`ArrayList<Dog>`

# VARIANCE

`ArrayList<Animal>`

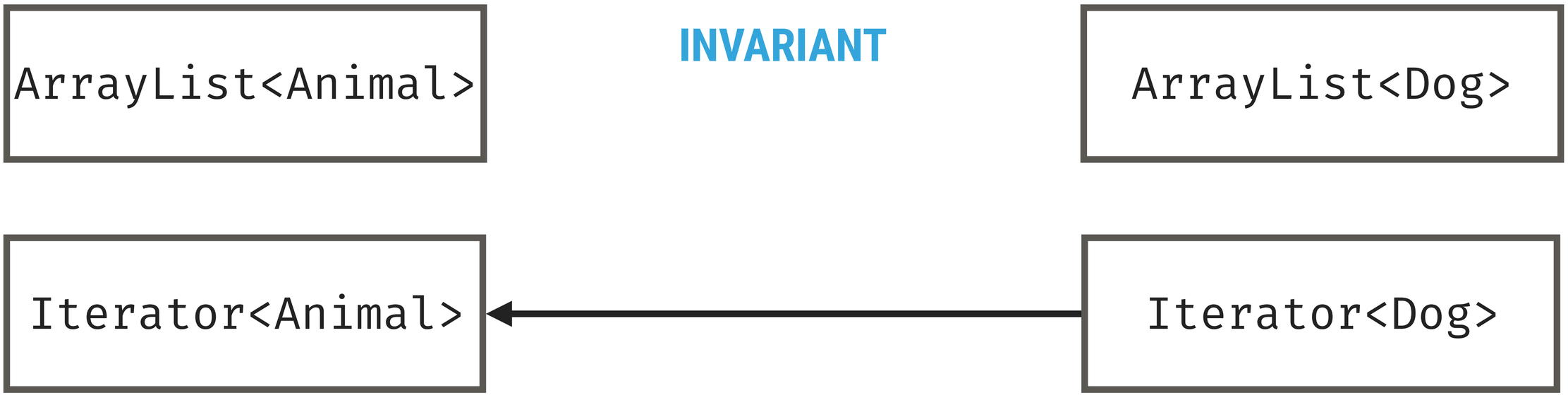
**INVARIANT**

`ArrayList<Dog>`

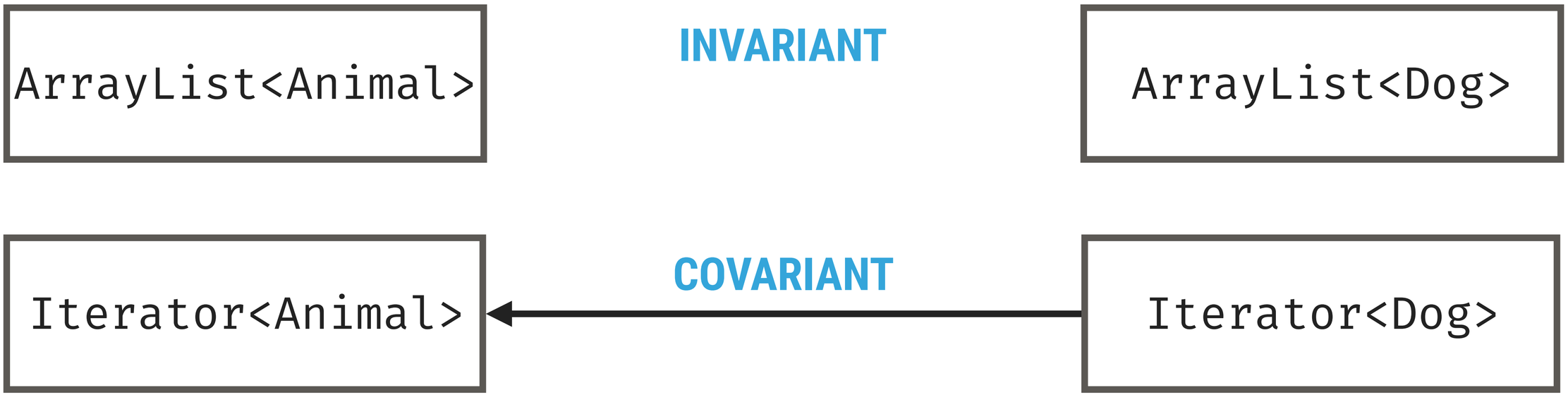
# VARIANCE


ArrayList<Animal>
INVARIANT
ArrayList<Dog>
Iterator<Animal>
Iterator<Dog>

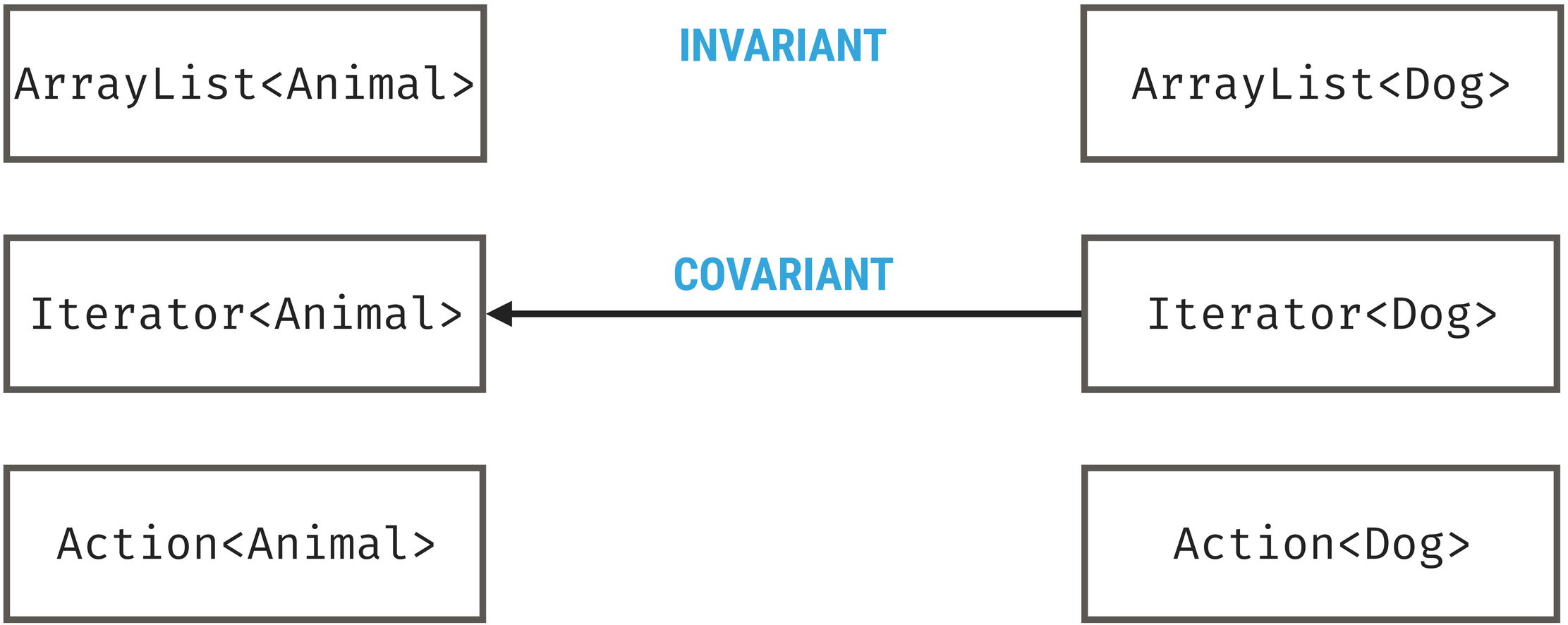
# VARIANCE


ArrayList<Animal>
INVARIANT
ArrayList<Dog>
Iterator<Animal>
Iterator<Dog>

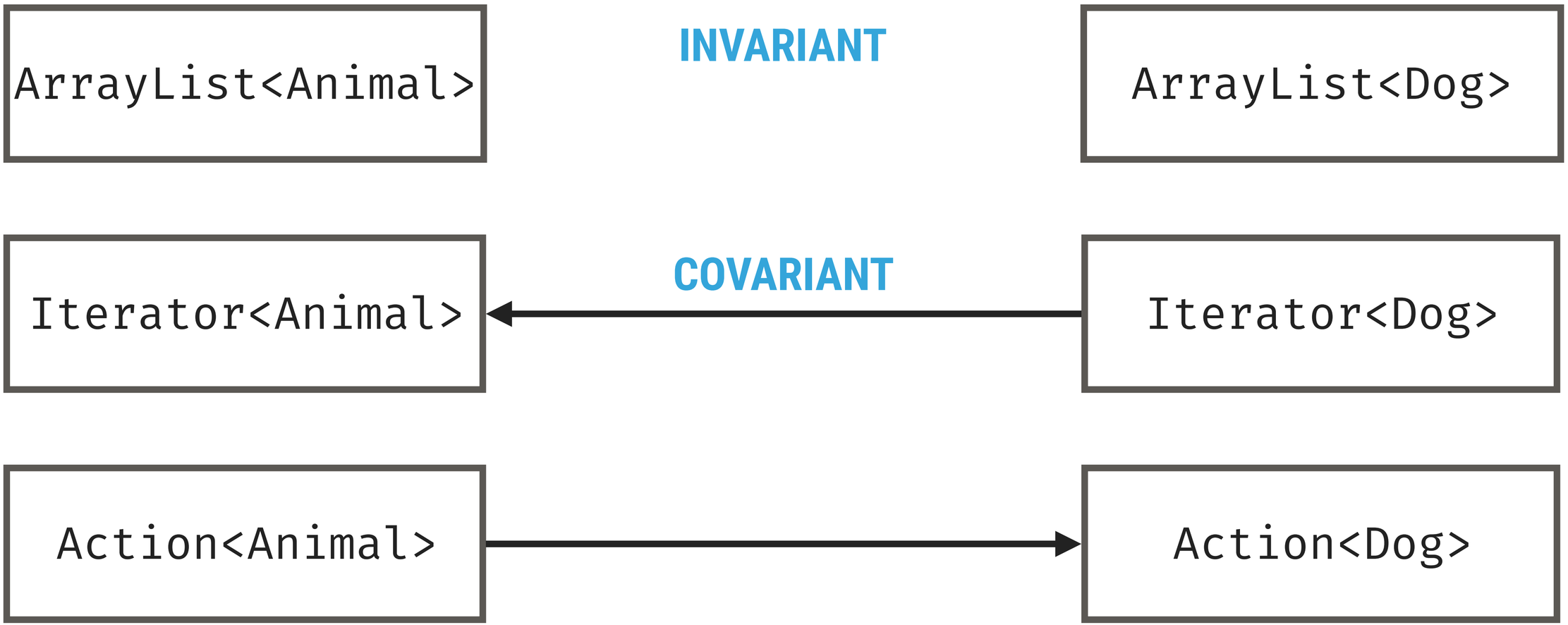
# VARIANCE


ArrayList<Animal>
INVARIANT
ArrayList<Dog>
Iterator<Animal>
COVARIANT
Iterator<Dog>

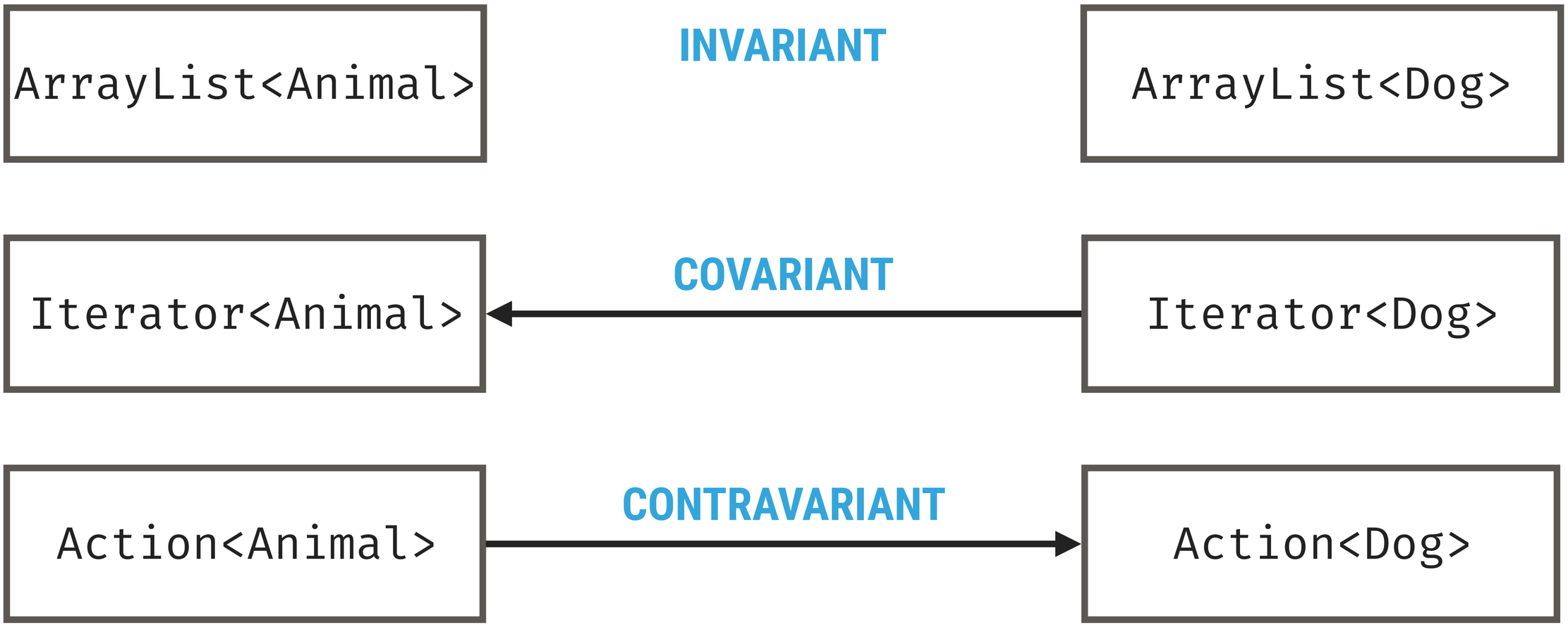
# VARIANCE


ArrayList<Animal>
INVARIANT
ArrayList<Dog>
Iterator<Animal>
COVARIANT
Iterator<Dog>
Action<Animal>
Action<Dog>

# VARIANCE


ArrayList<Animal>
INVARIANT
ArrayList<Dog>
Iterator<Animal>
COVARIANT
Iterator<Dog>
Action<Animal>
Action<Dog>

# VARIANCE


ArrayList<Animal>
INVARIANT
ArrayList<Dog>
Iterator<Animal>
COVARIANT
Iterator<Dog>
Action<Animal>
CONTRAVARIANT
Action<Dog>

# VARIANCE

```
// Java
List<Dog> dogs = new ArrayList<>();
List<Animal> animals = dogs;
```

# VARIANCE

```
// Java
List<Dog> dogs = new ArrayList<>();
List<Animal> animals = dogs;
```

# USE-SITE VS DECLARATION-SITE VARIANCE

```
// Java
List<Dog> dogs = new ArrayList<>();
List<? extends Animal> animals = dogs;
```

# USE-SITE VS DECLARATION-SITE VARIANCE

```java
// Java
List<Dog> dogs = new ArrayList<>();
List<? extends Animal> animals = dogs;
```

```kotlin
// Kotlin
val dogs: List<Dog> = ArrayList()
val animals: List<Animal> = dogs
```

# USE-SITE VS DECLARATION-SITE VARIANCE

```java
// Java
List<Dog> dogs = new ArrayList<>();
List<? extends Animal> animals = dogs;
```

```kotlin
// Kotlin
val dogs: List<Dog> = ArrayList()
val animals: List<Animal> = dogs

interface List<out E> : Collection<E>
```

# USE-SITE VS DECLARATION-SITE VARIANCE

```java
// Java
List<Dog> dogs = new ArrayList<>();
List<? extends Animal> animals = dogs;
```

```kotlin
// Kotlin
val dogs: List<Dog> = ArrayList()
val animals: List<Animal> = dogs

interface List<out E> : Collection<E>
```

# NOW BACK TO THE CODE

# MAY THE ANNOTATIONS BE WITH YOU

- `@JvmOverloads`
- `@JvmStatic`
- `@JvmField`
- `@JvmWildcard @JvmSuppressWildcards`
- `@Throws`
- `@JvmName`
- `@JvmMultifileClass`

# WORTH NOTING

# WORTH NOTING

- Inline functions

# WORTH NOTING

- Inline functions
- Reified type params

# WORTH NOTING

- Inline functions
- Reified type params
- `java.lang.Class`

# WORTH NOTING

- Inline functions
- Reified type params
- `java.lang.Class`
- `Unit`

# WORTH NOTING

- Inline functions
- Reified type params
- `java.lang.Class`
- `Unit`
- Typealiases

# WORTH NOTING

- Inline functions
- Reified type params
- `java.lang.Class`
- `Unit`
- Typealiases
- Visibility

# WORTH NOTING

- Inline functions
- Reified type params
- `java.lang.Class`
- `Unit`
- Typealiases
- `internal` visibility (!)

# Writing Java-friendly Kotlin code

@colriot