Android CI At Scale - How Square Makes It Work

Paul Hundal, Inez Korczyński
DPE Summit 2023

Square

Our Team

Paul Hundal

Senior Software Engineer

Inez Korczyński

Senior Software Engineer

Agenda

1 General Stats

2 Build Pipeline Composition

3 Git S3 strategy

4 UI Test Avoidance

5 Shard Avoidance

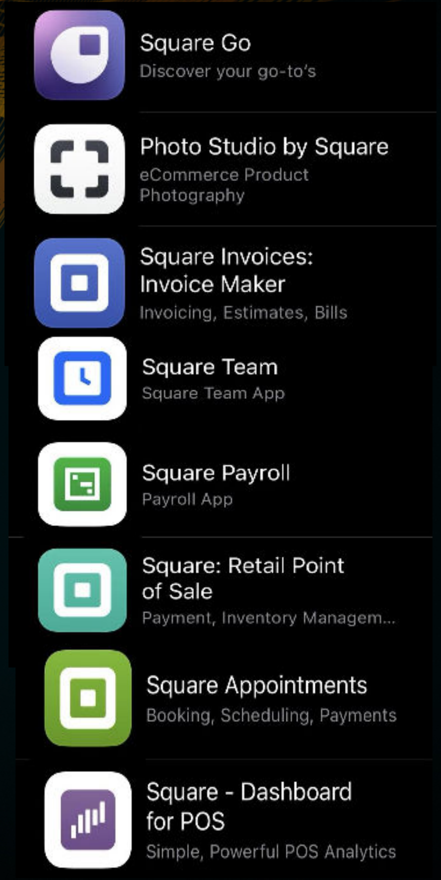
6 Results

7 Next Steps

1

General Stats

Overview & numbers



Square Go
Discover your go-to's

Photo Studio by Square
eCommerce Product Photography

Square Invoices: Invoice Maker
Invoicing, Estimates, Bills

Square Team
Square Team App

Square Payroll
Payroll App

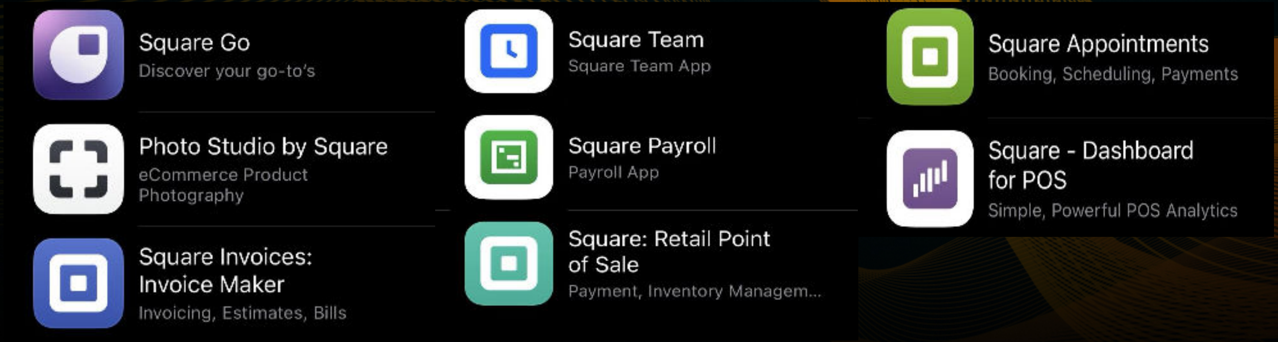
Square: Retail Point of Sale
Payment, Inventory Managem...

Square Appointments
Booking, Scheduling, Payments

Square - Dashboard for POS
Simple, Powerful POS Analytics

Some Stats

- 200 Android developers
- 11 apps in Google Play Store
- 300 demo/development apps
- 5,400 Android modules



Square Go
Discover your go-to's

Photo Studio by Square
eCommerce Product Photography

Square Invoices: Invoice Maker
Invoicing, Estimates, Bills

Square Team
Square Team App

Square Payroll
Payroll App

Square: Retail Point of Sale
Payment, Inventory Managem...

Square Appointments
Booking, Scheduling, Payments

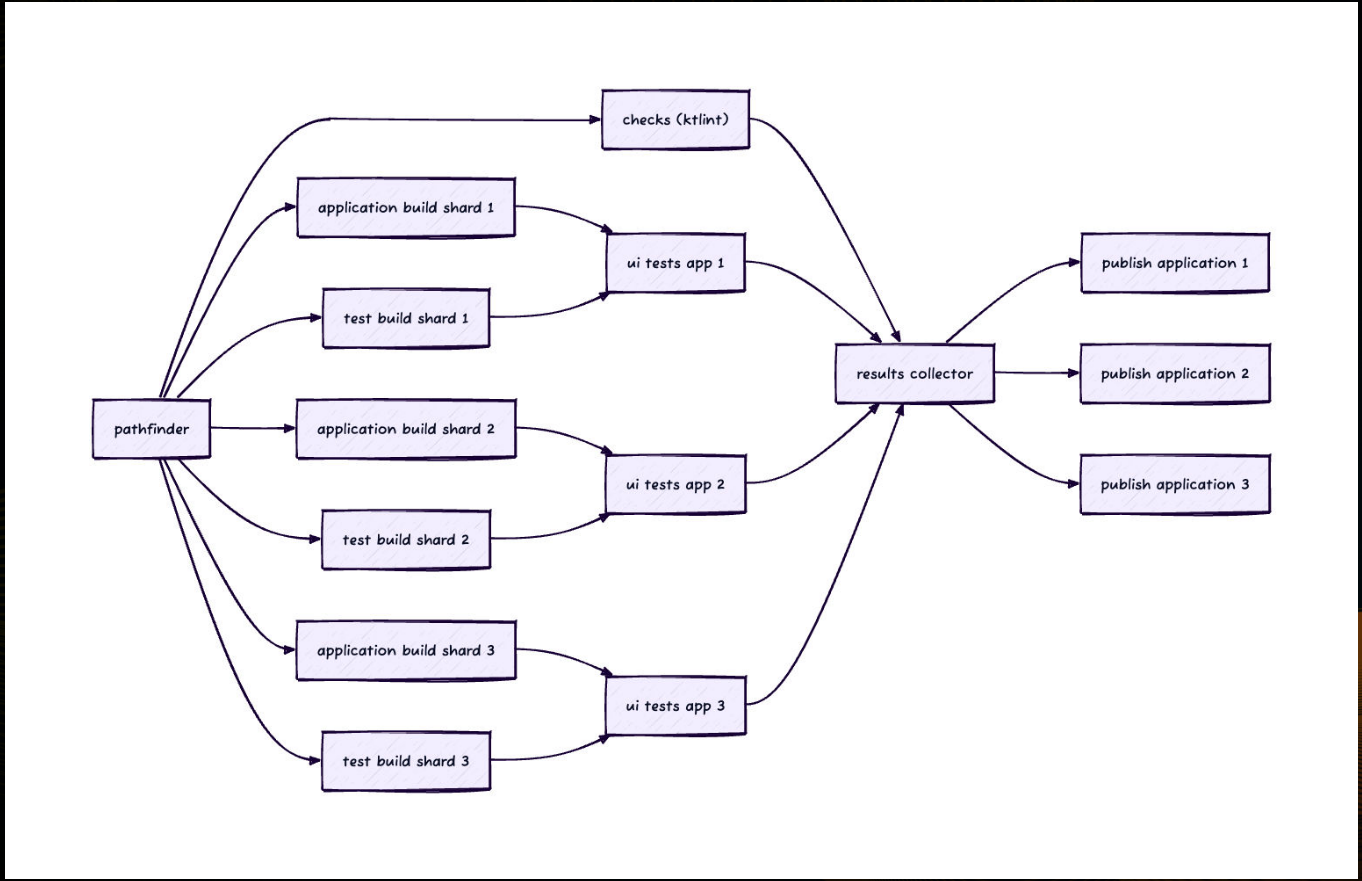
Square - Dashboard for POS
Simple, Powerful POS Analytics

2

Composition of our Build Pipeline

Types Of Jobs

- checks
  - e.g. check-ktlint, check-unused-dependencies
- builds
  - e.g. pos-assemble-release, pos-assemble-android-test, login-screen-assemble-debug,
- unit tests
  - e.g. pos-unit-tests
- ui-tests
  - e.g. pos-ui-mobile, point-of-sale-ui-tablet
- publish
  - pos-sign-and-upload


checks (ktlint)
application build shard 1
ui tests app 1
test build shard 1
results collector
publish application 1
publish application 2
publish application 3
pathfinder
application build shard 2
ui tests app 2
test build shard 2
application build shard 3
ui tests app 3
test build shard 3

Problems?

3

Git / S3 Strategy

- Git snapshot
  - Created daily
  - Shallow clone (depth=50)
- Git bundle
  - Created for each SHA
  - Differential (snapshot => SHA)

4

UI Test Avoidance

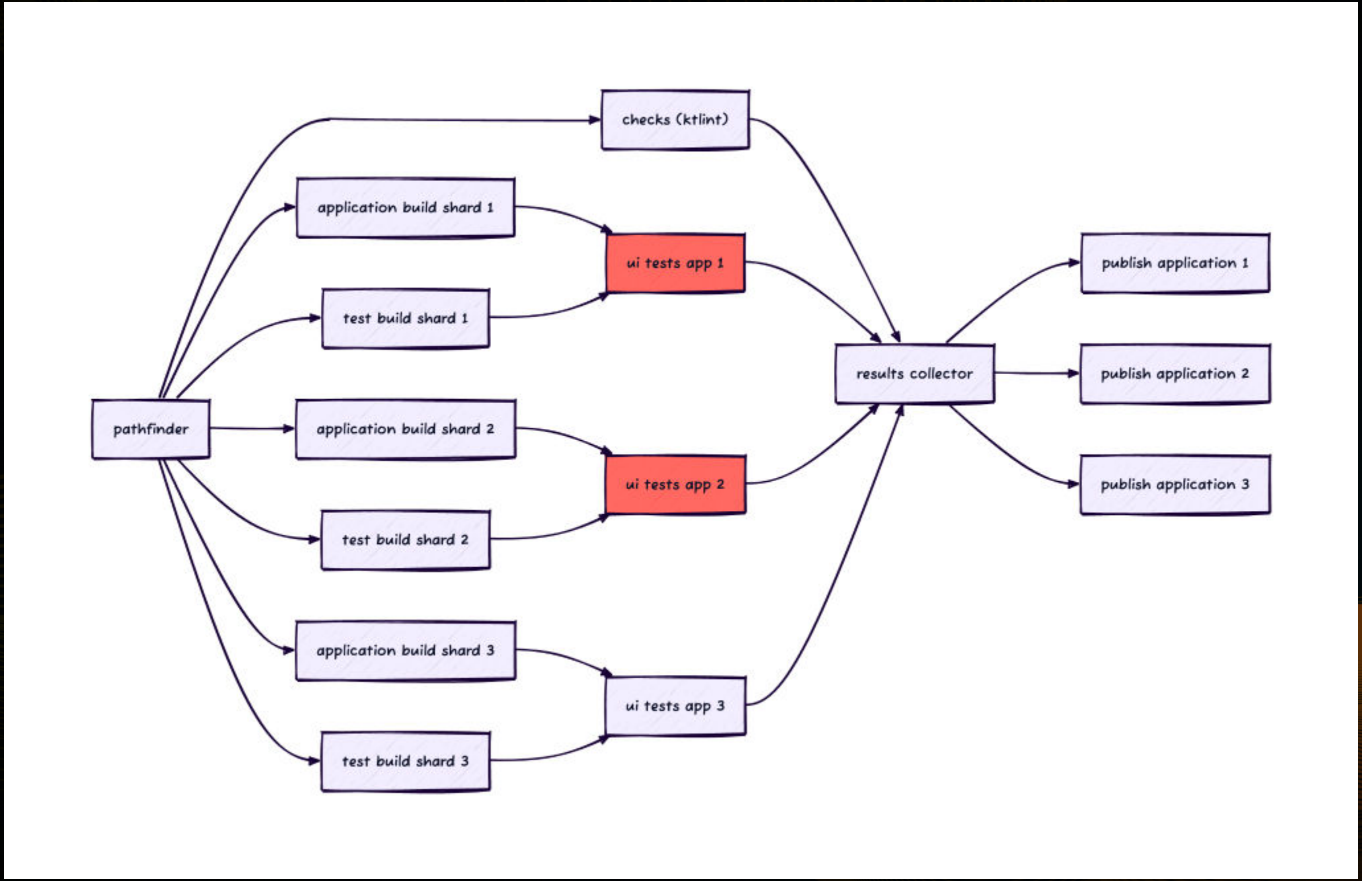
- 14,000 UI tests
- 500 CI UI tests jobs

pos-mobile-ui

3,100 tests & 120 emulators

login-screen-demo-mobile-ui

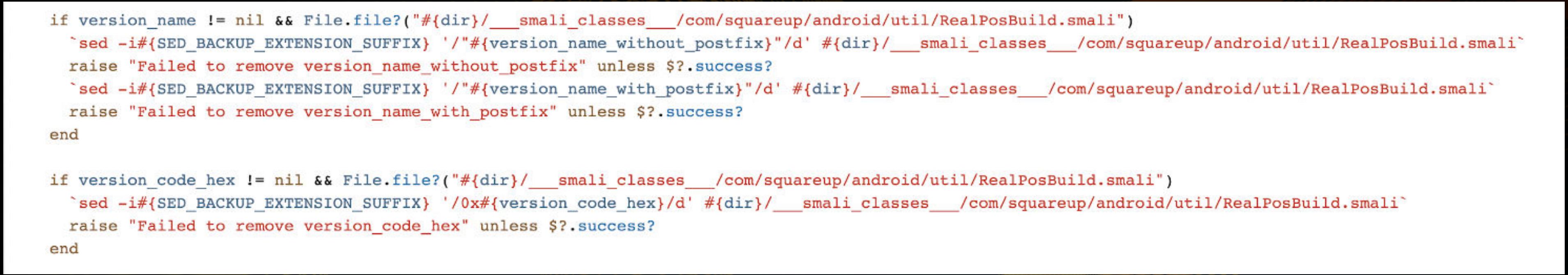
5 tests & 1 emulator


checks (ktlint)
application build shard 1
ui tests app 1
test build shard 1
pathfinder
application build shard 2
ui tests app 2
test build shard 2
application build shard 3
ui tests app 3
test build shard 3
results collector
publish application 1
publish application 2
publish application 3

Smali classes/files ending up in different directories

```ruby
# As it turns out in Android compilation with multidexing sometimes some classes will end up in
# for instance smali_classes2 and other time in smali_classes5 (it is not deterministic),
# however it does not have impact on how the application work. To work-around that we are moving
# all files/directories into common (`___smali_classes___`) directory first.
# Note: `smali_classesX` start from 2
FileUtils.mkdir("#{dir}/___smali_classes___/")
i = 2
while true
  if File.directory?("#{dir}/smali_classes#{i}")
    `cp -r #{dir}/smali_classes#{i}/. #{dir}/___smali_classes___/`
    raise "Copying smali_classes* to ___smali_classes___ failed" unless $?.success?
    FileUtils.rm_rf("#{dir}/smali_classes#{i}")
    i = i + 1
  else
    break
  end
end
```

Timestamp, versions, shas



```
if version_name != nil && File.file?("#{dir}/___smali_classes___/com/squareup/android/util/RealPosBuild.smali")
  `sed -i#{SED_BACKUP_EXTENSION_SUFFIX} '/"#{version_name_without_postfix}"/d' #{dir}/___smali_classes___/com/squareup/android/util/RealPosBuild.smali`
  raise "Failed to remove version_name_without_postfix" unless $?.success?
  `sed -i#{SED_BACKUP_EXTENSION_SUFFIX} '/"#{version_name_with_postfix}"/d' #{dir}/___smali_classes___/com/squareup/android/util/RealPosBuild.smali`
  raise "Failed to remove version_name_with_postfix" unless $?.success?
end

if version_code_hex != nil && File.file?("#{dir}/___smali_classes___/com/squareup/android/util/RealPosBuild.smali")
  `sed -i#{SED_BACKUP_EXTENSION_SUFFIX} '/0x#{version_code_hex}/d' #{dir}/___smali_classes___/com/squareup/android/util/RealPosBuild.smali`
  raise "Failed to remove version_code_hex" unless $?.success?
end
```

Results - hit ratios

98%

Demo/Development applications

25%

Large applications

5

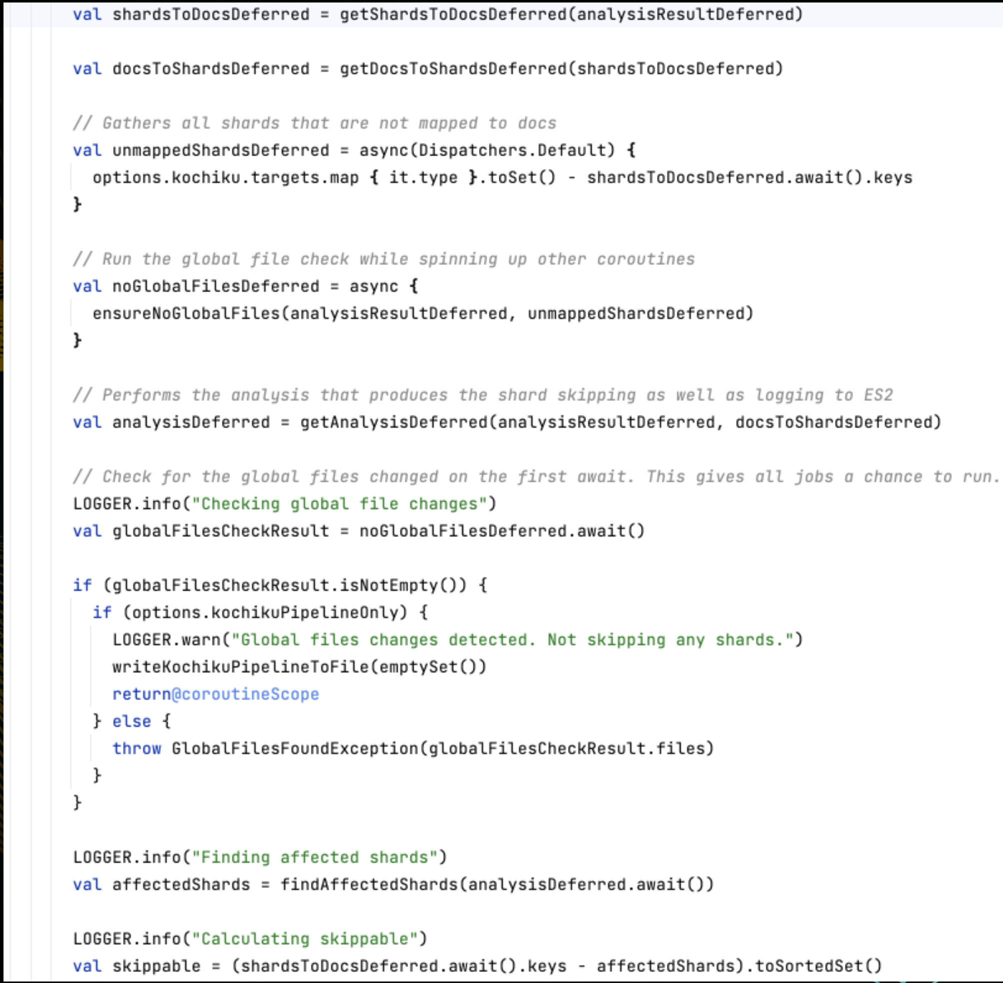
Shard Avoidance

Shard Avoidance Benefits

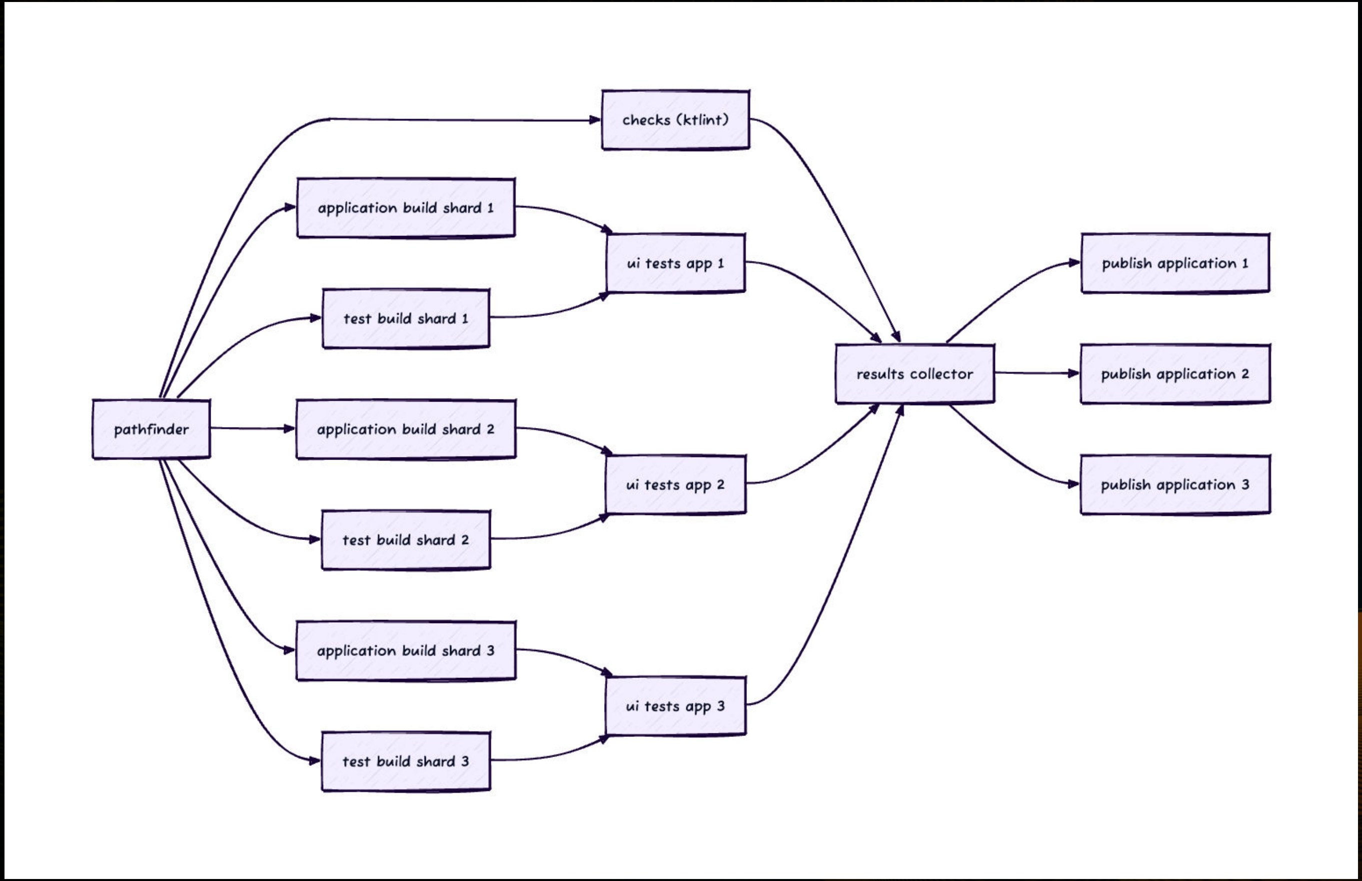
- Faster Builds
- Less potentially flakey shards to run
- Reduced worker queue
- Faster developer iterations

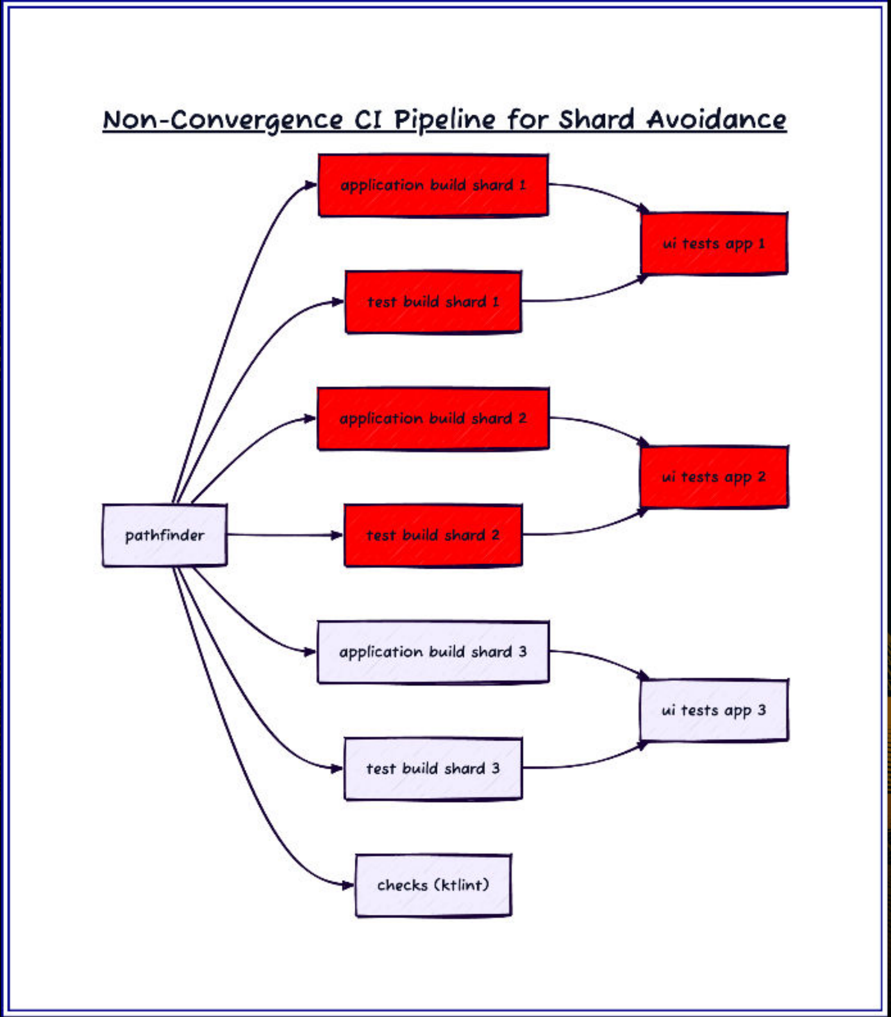
Shard Avoidance In Practice

- Compare Git SHA's
- Analyze modified files
- Map to Gradle Modules
- Find minimum set of CI shards to run



```kotlin
val shardsToDocsDeferred = getShardsToDocsDeferred(analysisResultDeferred)

val docsToShardsDeferred = getDocsToShardsDeferred(shardsToDocsDeferred)

// Gathers all shards that are not mapped to docs
val unmappedShardsDeferred = async(Dispatchers.Default) {
  options.kochiku.targets.map { it.type }.toSet() - shardsToDocsDeferred.await().keys
}

// Run the global file check while spinning up other coroutines
val noGlobalFilesDeferred = async {
  ensureNoGlobalFiles(analysisResultDeferred, unmappedShardsDeferred)
}

// Performs the analysis that produces the shard skipping as well as logging to ES2
val analysisDeferred = getAnalysisDeferred(analysisResultDeferred, docsToShardsDeferred)

// Check for the global files changed on the first await. This gives all jobs a chance to run.
LOGGER.info("Checking global file changes")
val globalFilesCheckResult = noGlobalFilesDeferred.await()

if (globalFilesCheckResult.isNotEmpty()) {
  if (options.kochikuPipelineOnly) {
    LOGGER.warn("Global files changes detected. Not skipping any shards.")
    writeKochikuPipelineToFile(emptySet())
    return@coroutineScope
  } else {
    throw GlobalFilesFoundException(globalFilesCheckResult.files)
  }
}

LOGGER.info("Finding affected shards")
val affectedShards = findAffectedShards(analysisDeferred.await())

LOGGER.info("Calculating skippable")
val skippable = (shardsToDocsDeferred.await().keys - affectedShards).toSortedSet()
```

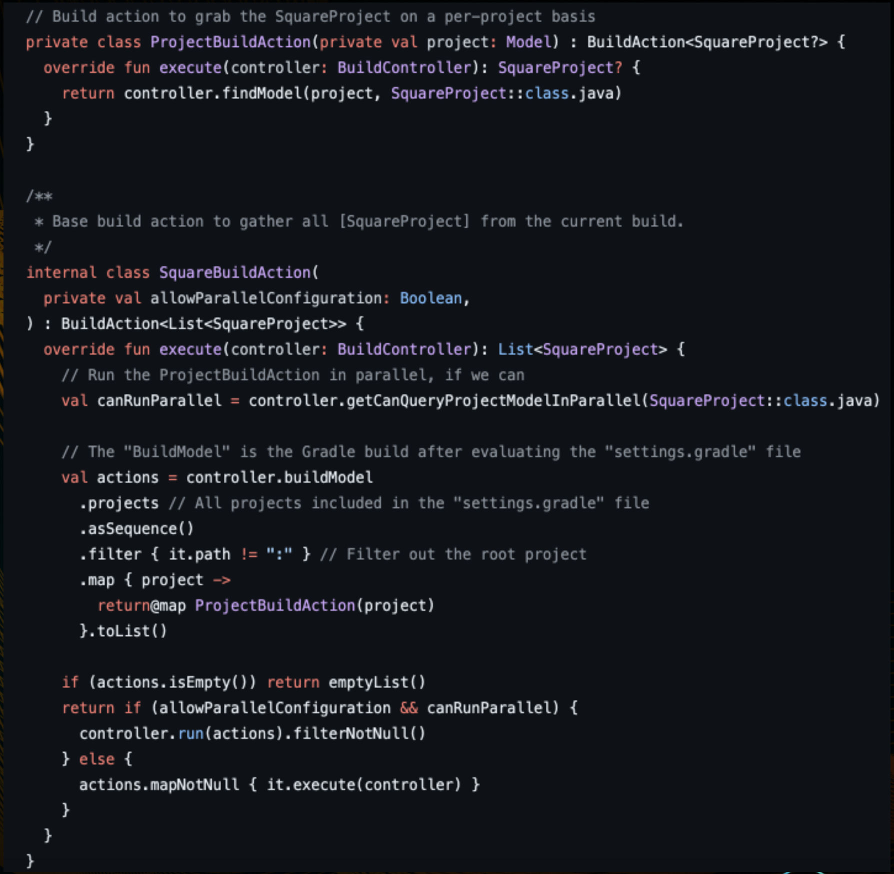

checks (ktlint)
application build shard 1
ui tests app 1
test build shard 1
pathfinder
application build shard 2
ui tests app 2
test build shard 2
application build shard 3
ui tests app 3
test build shard 3
results collector
publish application 1
publish application 2
publish application 3


Non-Convergence CI Pipeline for Shard Avoidance
pathfinder
application build shard 1
test build shard 1
ui tests app 1
application build shard 2
test build shard 2
ui tests app 2
application build shard 3
test build shard 3
ui tests app 3
checks (ktlint)

Gradle Tooling API

We analyzed how Gradle interacts with IntelliJ using the Tooling API to extract a custom model representation of our dependency graph.

- Extract the build graph
- Create model representation for analysis
- Inject models into our analyzer
- Determine which shards to skip



```kotlin
// Build action to grab the SquareProject on a per-project basis
private class ProjectBuildAction(private val project: Model) : BuildAction<SquareProject?> {
  override fun execute(controller: BuildController): SquareProject? {
    return controller.findModel(project, SquareProject::class.java)
  }
}

/**
 * Base build action to gather all [SquareProject] from the current build.
 */
internal class SquareBuildAction(
  private val allowParallelConfiguration: Boolean,
) : BuildAction<List<SquareProject>> {
  override fun execute(controller: BuildController): List<SquareProject> {
    // Run the ProjectBuildAction in parallel, if we can
    val canRunParallel = controller.getCanQueryProjectModelInParallel(SquareProject::class.java)

    // The "BuildModel" is the Gradle build after evaluating the "settings.gradle" file
    val actions = controller.buildModel
      .projects // All projects included in the "settings.gradle" file
      .asSequence()
      .filter { it.path != ":" } // Filter out the root project
      .map { project ->
        return@map ProjectBuildAction(project)
      }.toList()

    if (actions.isEmpty()) return emptyList()
    return if (allowParallelConfiguration && canRunParallel) {
      controller.run(actions).filterNotNull()
    } else {
      actions.mapNotNull { it.execute(controller) }
    }
  }
}
```

Shard Avoidance Results

50%

Shards Skipped

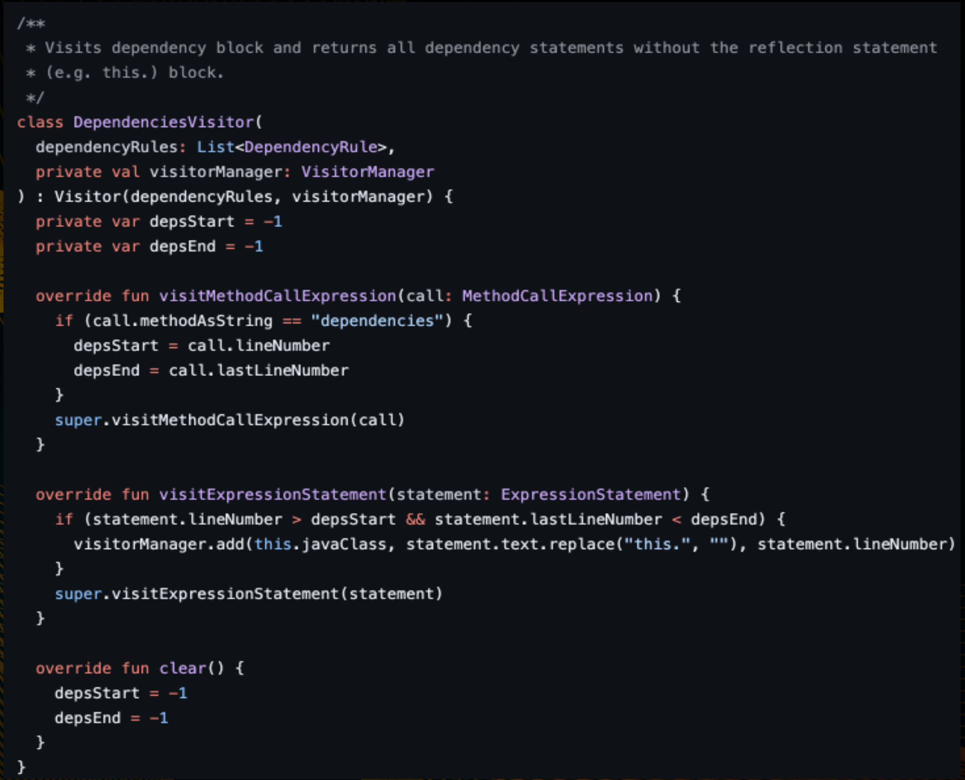
360

Time Saved (hrs)

Static Build Analyzer

We tried to bypass Gradle using Groovy's AST Parsing

- Mimic the model representation of Gradle
- Bypass the Tooling API completely
- Reduce the 3-5 minute runtime of the configuration phase to mere seconds!
- Lower level of correctness

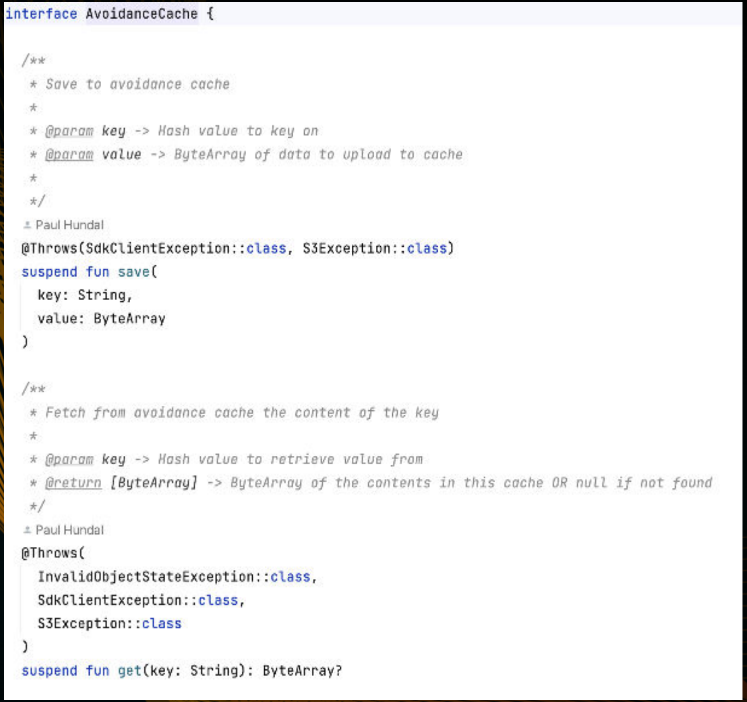


```
/**
 * Visits dependency block and returns all dependency statements without the reflection statement
 * (e.g. this.) block.
 */
class DependenciesVisitor(
  dependencyRules: List<DependencyRule>,
  private val visitorManager: VisitorManager
) : Visitor(dependencyRules, visitorManager) {
  private var depsStart = -1
  private var depsEnd = -1

  override fun visitMethodCallExpression(call: MethodCallExpression) {
    if (call.methodAsString == "dependencies") {
      depsStart = call.lineNumber
      depsEnd = call.lastLineNumber
    }
    super.visitMethodCallExpression(call)
  }

  override fun visitExpressionStatement(statement: ExpressionStatement) {
    if (statement.lineNumber > depsStart && statement.lastLineNumber < depsEnd) {
      visitorManager.add(this.javaClass, statement.text.replace("this.", ""), statement.lineNumber)
    }
    super.visitExpressionStatement(statement)
  }

  override fun clear() {
    depsStart = -1
    depsEnd = -1
  }
}
```

Shard Avoidance Cache

We realized that though expensive, Gradle produce an accurate representation of our build graph. But does it need to be invoked every time?

- Global files modified only 8% of the time
- Reusable build graph
- S3 Key/Value Store



```
interface AvoidanceCache {

  /**
   * Save to avoidance cache
   *
   * @param key -> Hash value to key on
   * @param value -> ByteArray of data to upload to cache
   *
   */
  ≗ Paul Hundal
  @Throws(SdkClientException::class, S3Exception::class)
  suspend fun save(
    key: String,
    value: ByteArray
  )

  /**
   * Fetch from avoidance cache the content of the key
   *
   * @param key -> Hash value to retrieve value from
   * @return [ByteArray] -> ByteArray of the contents in this cache OR null if not found
   */
  ≗ Paul Hundal
  @Throws(
    InvalidObjectStateException::class,
    SdkClientException::class,
    S3Exception::class
  )
  suspend fun get(key: String): ByteArray?
```

Results

63%

Avoidance Analysis Time Savings

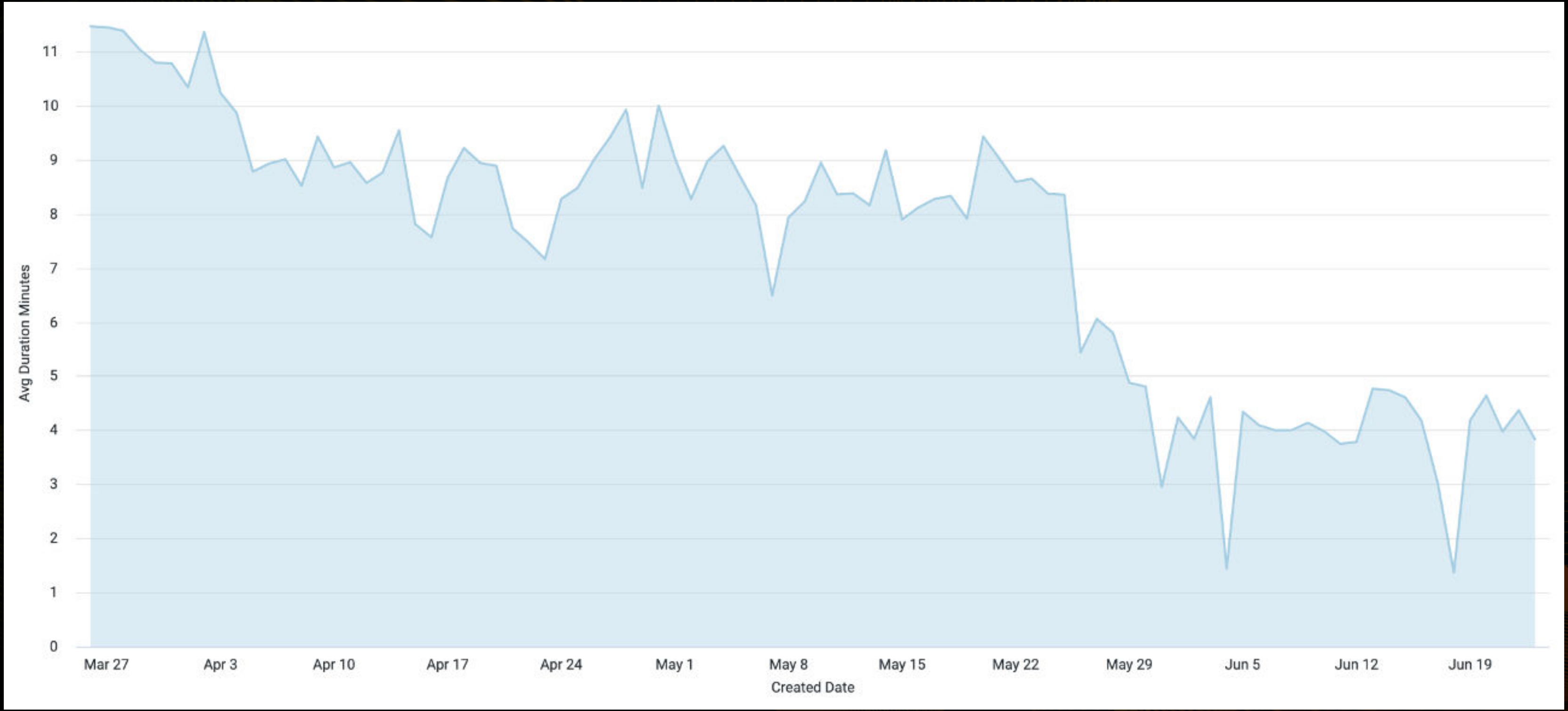
12%

Overall Build Time Reduction

90%

Avoidance Cache Hit Rate

Results



Avg Duration Minutes
0
1
2
3
4
5
6
7
8
9
10
11
Mar 27
Apr 3
Apr 10
Apr 17
Apr 24
May 1
May 8
May 15
May 22
May 29
Jun 5
Jun 12
Jun 19
Created Date

Recap

- Problem Space
    - 11 Apps
    - 200 Developers
    - 1,200 Shards
    - 14,000 UI Tests
    - 5,400 Android Modules
- Solutions
    - UI Test Avoidance
    - Shard Avoidance
    - Shard Caching

THANKS

paul.hundal@block.xyz
inez@block.xyz