Gradlin'

Pluggin' it in for Build Success

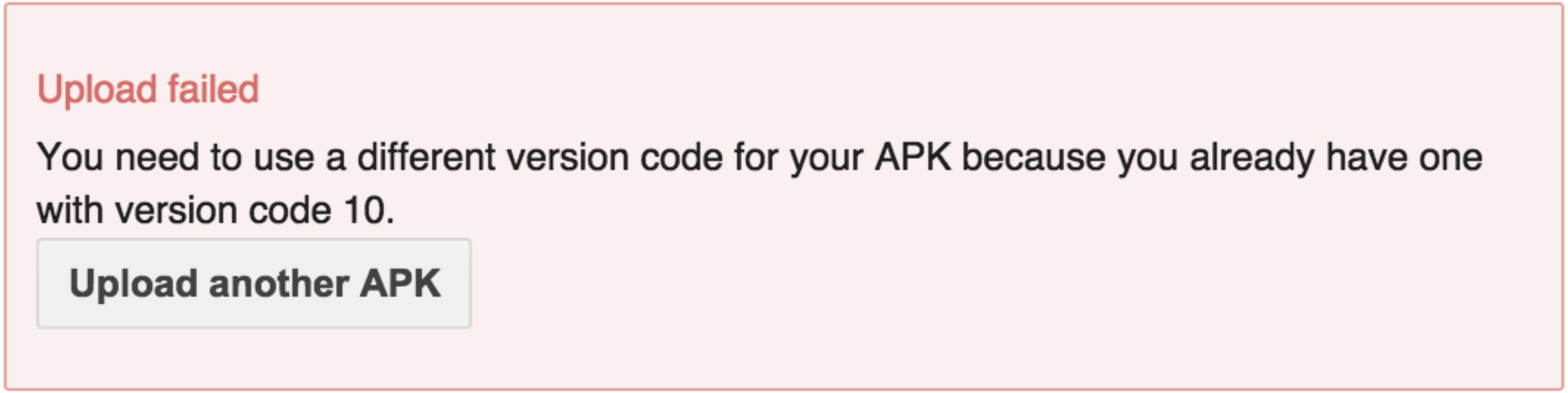
Versioning

From Project to Plugin

build.gradle

```
android {
  compileSdkVersion 23
  buildToolsVersion "23.0000000"

  defaultConfig {
    applicationId "neigut.lisa.gradlepractice"
    minSdkVersion 16
    targetSdkVersion 21
    versionCode 200
    versionName "200.0.1"
  }
}
```


Upload failed

You need to use a different version code for your APK because you already have one with version code 10.

```
* d2b1b18 (origin/master, origin/HEAD) bump version
...
* f0f0771 bump app version number
```

build.gradle

```
android {
  compileSdkVersion 23
  buildToolsVersion "23.0000000"

  defaultConfig {
    applicationId "neigut.lisa.gradlepractice"
    minSdkVersion 16
    targetSdkVersion 21
    versionCode lookupVersionCode()
    versionName lookupVersionName()
  }
}
```

build.gradle

```
def lookupVersionCode() {
  return 1
}

def lookupVersionName() {
  return "1.0"
}
```

build.gradle

```
project.ext.set("versionCode", 1);
project.ext.set("versionName", "1.0");
```

build.gradle

```
def lookupVersionCode() {
  return 1
}

def lookupVersionName() {
  return "1.0"
}
```

build.gradle

```
def lookupVersionCode() {
  return project.versionCode
}

def lookupVersionName() {
  return project.versionName
}
```

```
./gradlew :project:assemble -PversionCode=10 -PversionName="10"
```

Writing A Task To Bump Versions

build.java

```
project.tasks.create("bumpVersion") {
  ...
}
```

build.java

```
project.tasks.create("bumpVersion") {
  doLast {
    ...
  }
}
```

build.java

```
project.tasks.create("bumpVersion") {
  doLast {
    project.versionCode += 1;
    project.versionName = String.valueOf(project.versionCode + ".0")
    ...
  }
}
```

build.java

```
project.tasks.create("bumpVersion") {
  doLast {
    project.versionCode += 1;
    project.versionName = String.valueOf(project.versionCode + ".0")
    project.android.applicationVariants.all { variant ->
      variant.mergedFlavor.versionCode project.versionCode
      variant.mergedFlavor.versionName project.versionName
    }
  }
}
```

```
android {
  compileSdkVersion 23
  buildToolsVersion "23.0000000"

  defaultConfig {
    applicationId "neigut.lisa.gradlepractice"
    minSdkVersion 16
    targetSdkVersion 21
    versionCode project.versionCode
    versionName project.versionName
  }
}
```

Caveats:

- `applicationVariants` **is only available for** `com.android.application` **projects**
- **All product flavors will have the same** `versionCode` **and** `versionName`

$ ./gradlew tasks

```
Other tasks
-----------
bumpVersion
```

$ ./gradlew bumpVersion assembleDebug

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="neigut.lisa.gradlepractice"
    android:versionCode="2"
    android:versionName="2.0">
```

build.gradle

```
project.ext.set("versionCode", 1);
project.ext.set("versionName", "1.0");

android {
  compileSdkVersion 23
  buildToolsVersion "23.0000000"

  defaultConfig {
    applicationId "neigut.lisa.gradlepractice"
    minSdkVersion 16
    targetSdkVersion 21
    versionCode project.versionCode
    versionName project.versionName
  }
}

project.tasks.create("bumpVersion") {
  doLast {
    project.versionCode += 1;
    project.versionName = String.valueOf(project.versionCode + ".0")
    project.android.applicationVariants.all { variant ->
      variant.mergedFlavor.versionCode project.versionCode
      variant.mergedFlavor.versionName project.versionName
    }
  }
}
```

build.gradle old

```
defaultConfig {
  versionCode lookupVersionCode()
  versionName lookupVersionName()
  ...
}
```

build.gradle new

```
defaultConfig {
  versionCode project.versionCode
  versionName project.versionName
  ...
}
```

```
$ ./gradlew bumpVersion assembleDebug
```

$ ./gradlew bumpVersion assembleDebug

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="neigut.lisa.gradlepractice"
    android:versionCode="2"
    android:versionName="2.0">
```

Saving the State

```
AndroidApp
- build.gradle
- app/
-- src/
-- build.gradle
```

```
AndroidApp
- build.gradle
- app/
-- src/
-- build.gradle
-- versions.gradle
```

versions.gradle

```
majorVersion=2
minorVersion=0
bugFixVersion=0
```

versions.gradle

```
majorVersion=2
minorVersion=1
bugFixVersion=0
```

just released version: 2.0.0

version currently in testing: 2.1.0

Load the State

build.gradle

```
def String VERSION_FILE_NAME = "versions.gradle"
```

build.gradle

```
def String VERSION_FILE_NAME = "versions.gradle"

project.ext.set("majorVersion", 0);
project.ext.set("minorVersion", 0);
project.ext.set("bugFixVersion", 0);
```

build.gradle

```
loadVersion() {
  def versionFile = new File(project.projectDir, VERSION_FILE_NAME)
  versionFile.eachLine() { line ->
    def (key, value) = line.split("=").collect { it.trim() }
    if ("majorVersion".equals(key)) {
      project.majorVersion = Integer.parseInt(value)
    }
    ...
  }
}
```

Increment and Save To Disk

//TODO:
- Create 3 tasks (one for each version 'type')
- Each task increments the appropriate project property
- Write out the new values to `verisons.gradle`

build.gradle

```
def VERSIONS = ["majorVersion", "minorVersion", "bugFixVersion"]
...
   VERSIONS.each { versionType ->
      projects.tasks.create(name: "bumpVersionType") {
        doLast {
          project.ext[versionType] += 1
          ...
        }
      }
```

build.gradle

```
def VERSIONS = ["majorVersion", "minorVersion", "bugFixVersion"]
...
   VERSIONS.each { versionType ->
      projects.tasks.create(name: "bump$versionType") {
        doLast {
          project.ext[versionType] += 1
          // write to versions.gradle file
          // update the applicationVariants values
        }
      }
```

build.gradle

```
def VERSIONS = ["majorVersion", "minorVersion", "bugFixVersion"]
...
   VERSIONS.each { versionType ->
      projects.tasks.create(name: "bump$versionType") {
        doLast {
          project.ext[versionType] += 1
          new File(project.projectDir, VERSIONS_FILE_NAME).withWriter { out ->
            out.write {
               """majorVersion=${project.majorVersion}
               minorVersion=${project.minorVersion}
               bugFixVersion=${project.bugFixVersion}
               """
            }
          }
          // update the applicationVariants values
        }
      }
```

$ ./gradlew tasks

```
Other tasks
-----------
bumpmajorVersion
bumpminorVersion
bumpbugFixVersion
```

$ ./gradlew bumpmajorVersion

versions.gradle

```
majorVersion=3
minorVersion=0
bugFixVersion=0
```

$ ./gradlew bump<u>bugFix</u>Version

versions.gradle

```
majorVersion=3
minorVersion=0
bugFixVersion=1
```

AndroidManifest.xml

```
android:versionName="3.0.1"
```

build.gradle

```
def VERSIONS = ["majorVersion", "minorVersion", "bugFixVersion"]
...
   VERSIONS.each { versionType ->
      projects.tasks.create(name: "bump$versionType") {
        doLast {
          project.ext[versionType] += 1
          new File(project.projectDir, VERSIONS_FILE_NAME).withWriter { out ->
            out.write {
               """majorVersion=${project.majorVersion}
               minorVersion=${project.minorVersion}
               bugFixVersion=${project.bugFixVersion}
               """
            }
          }
          // update the applicationVariants values
        }
      }
```

build.gradle

```
def VERSIONS = ["majorVersion", "minorVersion", "bugFixVersion"]
afterEvaluate {
   VERSIONS.each { versionType ->
      projects.tasks.create(name: "bump$versionType") {
        doLast {
          project.ext[versionType] += 1
          new File(project.projectDir, VERSIONS_FILE_NAME).withWriter { out ->
            out.write {
               """majorVersion=${project.majorVersion}
               minorVersion=${project.minorVersion}
               bugFixVersion=${project.bugFixVersion}
               """
            }
          }
          // update the applicationVariants values
        }
      }
```

build.gradle

```
afterEvaluate {
  ...
}
```

```
beforeEvaluate { project -> ... }

afterEvaluate { project -> ... }
```

```
beforeEvaluate { project ->
  // set up project properties
  // load the versions from disk
}

afterEvaluate { project ->
  // create tasks to bump versions
}
```

Versioning

1. Stateful versions
2. Gradle tasks to change the version number
3. Build server (Jenkins) can easily manage version numbers
4. Can be checked into source control (Git)

Let's Make A New App

Options

— Ctrl-C, Ctrl-V

Options

- Ctrl-C, Ctrl-V
- Share logic via the root project

Options

- Ctrl-C, Ctrl-V
- Share logic via the root project
- Use a Gradle Plugin

Options

- Ctrl-C, Ctrl-V
- Share logic via the root project
- Use a Gradle Plugin

Multi-Project Builds

```
AndroidApp
- build.gradle
- app/
-- src/
-- build.gradle
```

```
AndroidApp
- build.gradle
- app/
-- src/
-- build.gradle
- app2/
-- src/
-- build.gradle
```

```
AndroidApp
- build.gradle  (root)
- app/
-- src/
-- build.gradle (app1)
- app2/
-- src/
-- build.gradle (app2)
```

Versioning Steps

- **Add** `bumpVersion` **tasks**
- **Set up project properties**
- **Load the versions file**
- **Set the variant version**

Root Project Hooks

- allprojects {}
- subprojects {}
- project(':app') {}

app/build.gradle

```
afterEvaluate { ...  }
```

app/build.gradle

```
afterEvaluate { ... }
```

build.gradle (root)

```
subprojects {
  project.afterEvaluate { ... }
}
```

Versioning Steps

- **Add** `bumpVersion` **tasks**
- **Set up project properties**
- **Load the versions file**
- **Set the variant version**

build.gradle (root)

```
def VERSIONS = ['majorVersion', 'minorVersion', 'bugFixVersion']
subprojects {
  project.afterEvaluate {
    // create bump version tasks here, dynamically
  }
}
```

Versioning Steps

- **Add** `bumpVersion` **tasks**
- **<u>Set up project properties</u>**
- **Load the versions file**
- **Set the variant version**

build.gradle (root)

```
def VERSIONS = ['majorVersion', 'minorVersion', 'bugFixVersion']
subprojects {
  project.afterEvaluate {
    // create bump version tasks here, dynamically
  }
  project.beforeEvaluate {
    VERSIONS.each { version ->
      project.ext.set(version, 0)
    }
  }
}
```

Versioning Steps

- **Add** `bumpVersion` **tasks**
- **Set up project properties**
- **<u>Load the versions file</u>**
- **Set the variant version**

build.gradle (root)

```
def VERSIONS = ['majorVersion', 'minorVersion', 'bugFixVersion']
subprojects {
  project.afterEvaluate {
    // create bump version tasks here, dynamically
  }
  project.beforeEvaluate {
    VERSIONS.each { version ->
      project.ext.set(version, 0)
    }
    loadVersions(project)
  }
}
```

Versioning Steps

- **Add** `bumpVersion` **tasks**
- **Set up project properties**
- **Load the versions file**
- **Set the variant version**

app/build.gradle (app1/app2)

```
defaultConfig {
  versionCode project.majorVersion * 10 + project.minorVersion // etc
  versionName project.majorVersion + " " + project.minorVersion // etc
  // ...
}
```

```
$ ./gradlew :app:bumpminorVersion :app2:bumpmajorVersion
$ ./gradlew :app:bumpminorVersion :app2:bumpmajorVersion

$ ./gradlew bumpminorVersion
```

Options

- Ctrl-C, Ctrl-V
- Share logic via the root project
- Use a Gradle Plugin

Gradle Plugins

Where can Plugin code live?

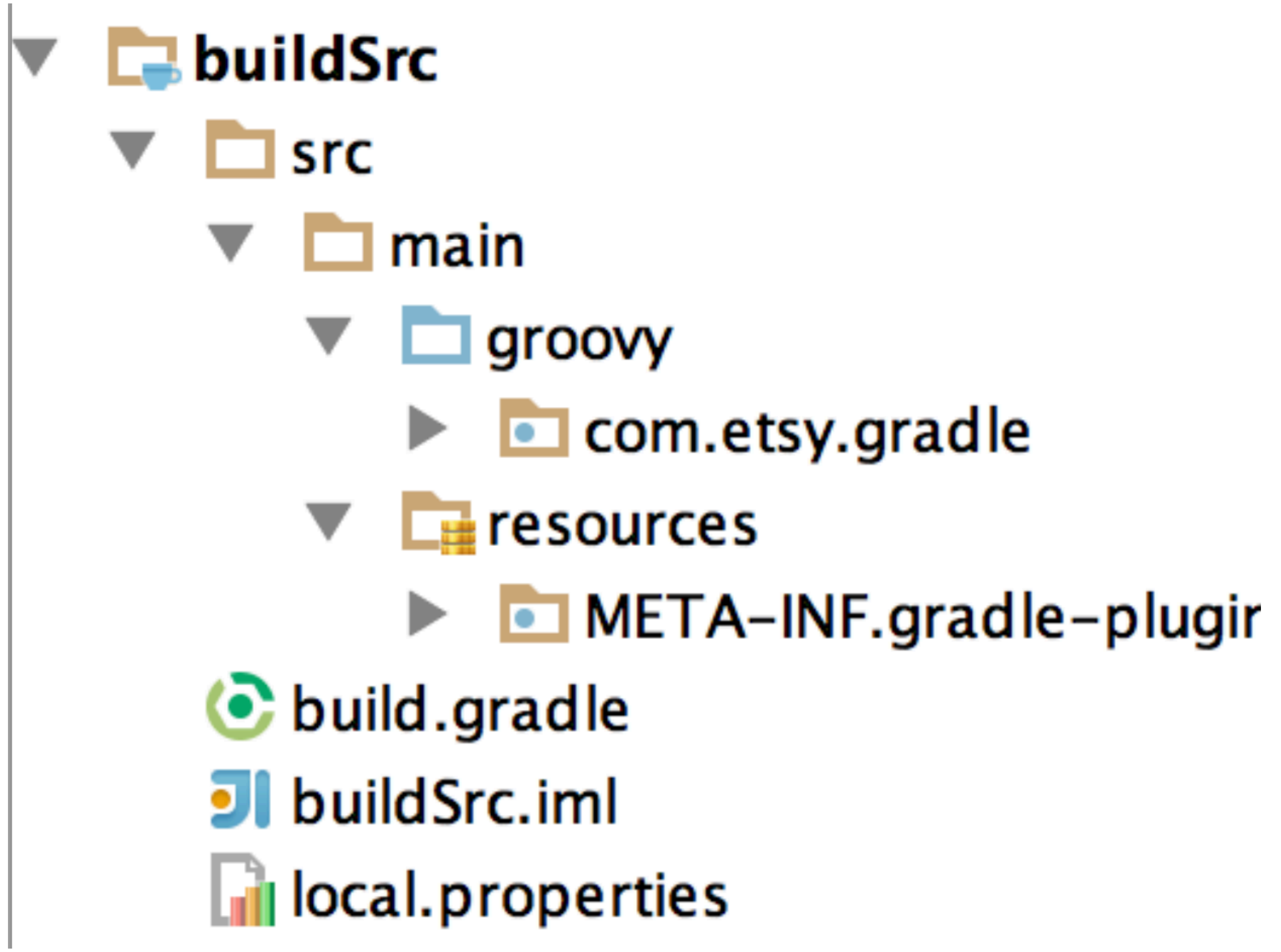
Where can Plugin code live?

- in the project class itself
- in `buildSrc`
- as a separate jar

```
AndroidApp
- build.gradle   (root)
- app/
- app2/
```

```
AndroidApp
- buildSrc/
- build.gradle   (root)
- app/
- app2/
```

- groovy package
- resources directory with a META-INF folder
- build.gradle file



```
▼ buildSrc
  ▼ src
    ▼ main
      ▼ groovy
        ▶ com.etsy.gradle
      ▼ resources
        ▶ META-INF.gradle-plugir
  build.gradle
  buildSrc.iml
  local.properties
```

- **an extension class**
- **the plugin class**

```
// Project Property
project.ext.set("majorVersion", 0)

// Project Extension
project.extensions.create("appVersion", VersionExtension)
project.appVersion.majorVersion = 0
```

VersionExtension.groovy

```
class VersionExtension {
  def int majorVersion
  def int minorVersion
  def int bugFixVersion
  // ...
}
```

VersionExtension.groovy

```
class VersionExtension {
  // ...
  def releaseString() { majorVersion + DOT + minorVersion + DOT + bugFixVersion }
  def code() { majorVersion * 10**6 + minorVersion * 10**4 + bugFixVersion }

  // 1.4.1
  // 1040001
}
```

VersionsPlugin.groovy

```groovy
class VersionPlugin implements Plugin<Project> {
```

VersionsPlugin.groovy

```
class VersionPlugin implements Plugin<Project> {
  void apply(Project project) {
    \\ plugin set up logic
  }
}
```

build.gradle (root)

```
subprojects {
  project.afterEvaluate { ... }
}
```

build.gradle (root)

```
subprojects {
  project.afterEvaluate { ... }
}
```

VersionsPlugin.groovy

```
void apply(Project project) {
  project.afterEvaluate { ... }
}
```

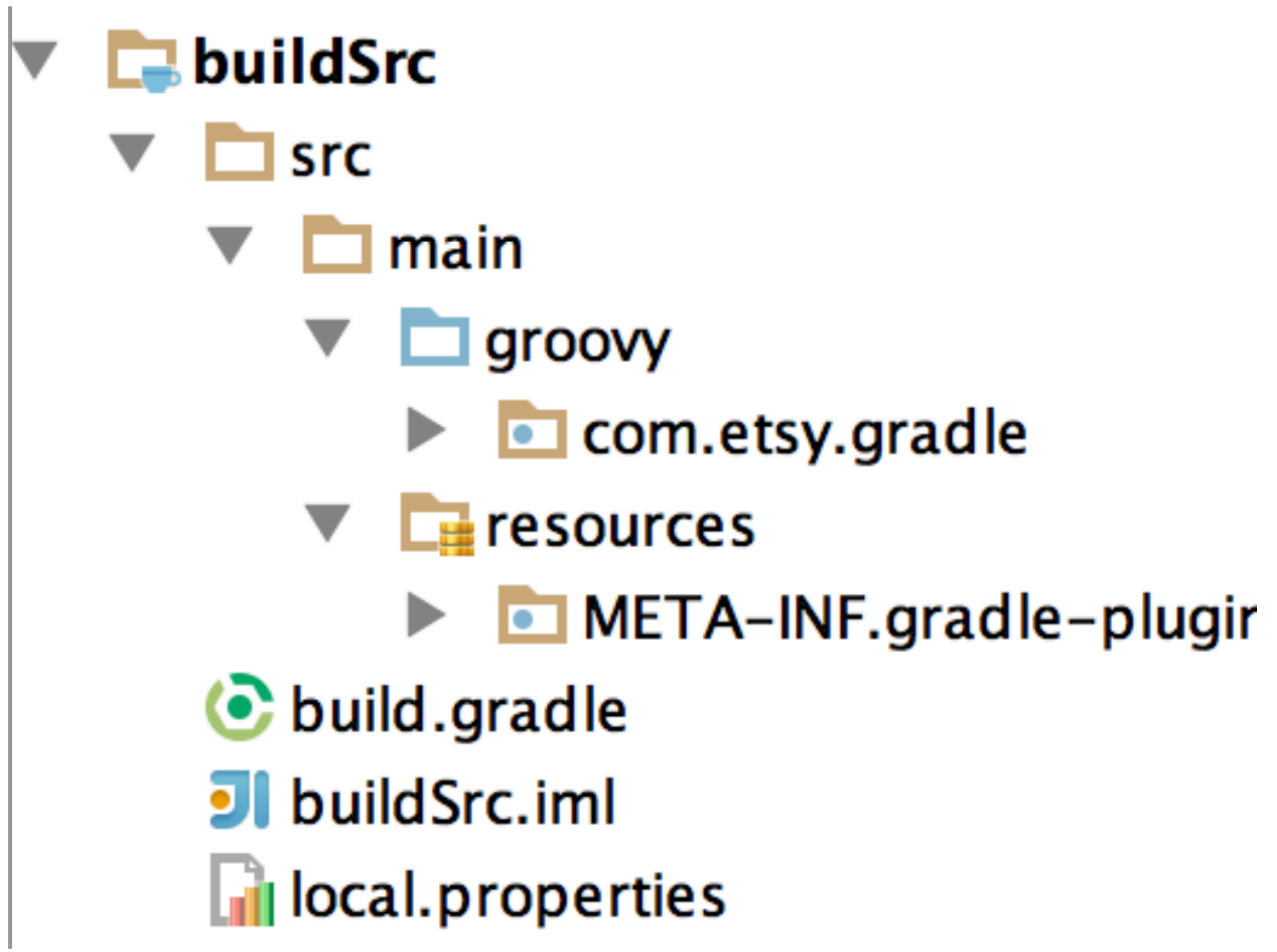
VersionsPlugin.groovy

```
class VersionPlugin implements Plugin<Project> {
  void apply(Project project) {
    project.extensions.create("appVersion", VersionExtension)
    project.appVersion.loadVersions(project)
    project.afterEvaluate {
      VERSIONS.each { version ->
        project.tasks.create(name: "bump$version") {
          doLast {
            project.appVersion.bump(version)
            // Write out to file
            // Update `android` plugin values
          }
        }
      }
    }
  }
}
```

Use your Plugin

— **Expose**
— **Apply**
— **Use**

Expose



```
buildSrc
  src
    main
      groovy
        com.etsy.gradle
      resources
        META-INF.gradle-plugi
  build.gradle
  buildSrc.iml
  local.properties
```

Expose

buildSrc/resources/META-INF.gradle-plugins/appVersion.properties

```
implementation-class=neigut.lisa.gradle.VersionsPlugin
```

Apply

app/build.gradle

```
apply plugin: 'com.android.application'
apply plugin: 'appVersion'
```

Use

app/build.gradle

```
android {
  defaultConfig {
    versionCode appVersion.code()
    versionName appVersion.releaseString()
  }
}
```

Kevin Grant's Sample Project

https://github.com/kevinthecity/GradlePluginExample

Lisa Neigut

work @electricobjects

me on the internet, @niftynei

~thank you~