



SNS COLLEGE OF TECHNOLOGY, COIMBATORE –35
(An Autonomous Institution)
19CSB303 and Composing Mobile Apps
UNIT 5



Versioning

Versioning is the creation and management of multiple releases of a product, all of which have the same general function but are improved, upgraded or customised.

To define the version information for your Android app, set values for the version settings in the Gradle build files. These values are then merged into your app's manifest file during the build process.

Two settings are available, and you should always define values for both of them:

Version Code: An integer used as an internal version number. This number is used only to determine whether one version is more recent than another, with higher numbers indicating more recent versions.

Version Name: A string used as the version number shown to users. This setting can be specified as a raw string or as a reference to a string resource.

Semantic Versioning:

Semantic Versioning (referred to, for short, as SemVer), is a versioning system that has been on the rise over the last few years. In this scheme, version numbers and the way they change convey meaning about the underlying code and what has been modified from one version to the next.

SemVer is a 3-component system in the format of x.y.z where:

- x stands for a major version
- y stands for a minor version
- z stands for a patch

So you have Major.Minor.Patch.

How to apply SemVer in Android ?

Based on the following changes we apply this versioning scheme

Major Version: API Changes, Adding new feature, Redesign the App, Increment the MAJOR after the MINOR version achieved the number 9

Minor Version: While adding a functionality, minor changes like color change or icon changes

Patch: Bug fixing

Precedence refers to how versions are compared to each other when ordered. Precedence MUST be calculated by separating the version into major, minor, patch and pre-release identifiers in that order

Pre-release:

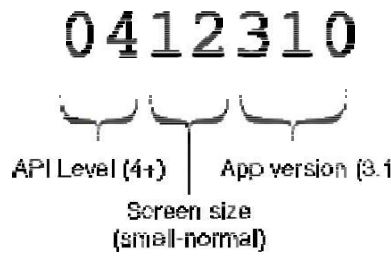
A pre-release version may be denoted by appending a version classifier that starts with a “-”. For example “1.1.0-alpha”, “1.2.1-beta”, “4.1.3-preview”, “1.1.8-demo”

When major, minor, and patch are equal, a pre-release version has lower precedence than a normal version. Example: 1.0.0-alpha < 1.0.0.

Using a version code scheme:

In order to allow different APKs to update their version codes independent of others. As per developer guidelines a version code with at least 7 digits.

0412310 For example, when the application version name is 3.1.0, version codes for an API level 4 would be something like 0412310. The first two digits are reserved for the API Level (4), the middle two digits are for either screen sizes or GL texture formats (12), and the last three digits are for the application’s version name (3.1.0).



Automate the versioning scheme with Gradle:



You can automate the versioning scheme using the following snippet on your build.gradle.

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

```
ext.majorVersion = 1
```

```
ext.minorVersion = 2
```

```
ext.patchVersion = 3
```

```
ext.preRelease= "DEMO"
```

```
ext.minSdkVersion = 14
```

```
android {
```

```
    compileSdkVersion 23
```

```
    buildToolsVersion "23.0.2"
```

```
    defaultConfig {
```

```
        applicationId "droidmentor.sample"
```

```
        targetSdkVersion 23
```

```
        minSdkVersion project.ext.minSdkVersion versionCode
```

```
        generateVersionCode() // 140010203 versionName
```

```
        generateVersionName() // 1.2.3-DEMO
```

```
    }
```

```
}
```

```
private Integer generateVersionCode() {
```

```
    return ext.minSdkVersion * 10000000 + ext.majorVersion * 10000 + ext.minorVersion *  
100 + ext.patchVersion
```

```
}
```

```
private String generateVersionName() {
```

```
    String versionName = "${ext.majorVersion}.${ext.minorVersion}.${ext.patchVersion}"
```

```
    if (ext.preRelease != null && !ext.preRelease.isEmpty()) {
```

```
        versionName = versionName + "-" + ext.preRelease
```

```
    }
```

```
    return versionName;
```

```
}
```