**Chapter 3 – Description of Tools**

**3.1 Android Studio**

Android Studio is the official[integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) for [Google](https://en.wikipedia.org/wiki/Google)'s [Android](https://en.wikipedia.org/wiki/Android_(operating_system)) [operating system](https://en.wikipedia.org/wiki/Operating_system), built on [JetBrains](https://en.wikipedia.org/wiki/JetBrains)' [IntelliJ IDEA](https://en.wikipedia.org/wiki/IntelliJ_IDEA) software and designed specifically for [Android development](https://en.wikipedia.org/wiki/Android_software_development). It is available for download on [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS) and [Linux](https://en.wikipedia.org/wiki/Linux) based operating systems. It is a replacement for the [Eclipse Android Development Tools](https://en.wikipedia.org/wiki/Eclipse_(software)#Android_Development_Tools) (ADT) as primary IDE for native Android application development.

Android Studio was announced on May 16, 2013 at the [Google I/O](https://en.wikipedia.org/wiki/Google_I/O) conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. The current stable version is 3.1 released in March 2018

The following features are provided in the current stable version:

* [Gradle](https://en.wikipedia.org/wiki/Gradle)-based build support
* Android-specific [refactoring](https://en.wikipedia.org/wiki/Code_refactoring) and quick fixes
* [Lint](https://en.wikipedia.org/wiki/Lint_(software)) tools to catch performance, usability, version compatibility and other problems
* [ProGuard](https://en.wikipedia.org/wiki/ProGuard_(software)) integration and app-signing capabilities
* Template-based wizards to create common Android designs and components
* A rich [layout editor](https://en.wikipedia.org/wiki/Graphical_user_interface_builder) that allows users to drag-and-drop UI components, option to [preview layouts](https://en.wikipedia.org/wiki/WYSIWYG) on multiple screen configurations
* Support for building [Android Wear](https://en.wikipedia.org/wiki/Android_Wear) apps
* Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine.Android Virtual Device (Emulator) to run and debug apps in the Android studio.

Android Studio supports all the same programming languages of [IntelliJ](https://en.wikipedia.org/wiki/IntelliJ), and [PyCharm](https://en.wikipedia.org/wiki/PyCharm) e.g. [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), and [Kotlin](https://en.wikipedia.org/wiki/Kotlin_(programming_language))  and Android Studio 3.0 supports "Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects [backport](https://en.wikipedia.org/wiki/Backporting) some Java 9 features.

## **3.1.1Advantages of Android studio:-**

## **Faster Deployment of Fresh Builds**

Bringing incremental changes to an existing app code or resource is now easier and faster. Thanks to Instant Run. Code changes can be witnessed in the emulator or physical device on real-time without restarting the app or building a new APK (Android Application Package file) every time.

* **More Accurate Programming**

Featuring an intelligent Code Editor equipped with  IntelliJ IDEA interface, ***Android Studio*** makes code writing and analysis faster, easier and more accurate. The most challenging area has become a cakewalk now.

## **Faster Programming and Testing**

The newly introduced emulator is 3x faster in CPU, RAM, & I/O in comparison to its predecessor. The virtual testing environment is faster than a real device and has a user-friendly UI. Sensor controls are effective to read every move of the developers. Developers can drag and drop APKs for quick installation, resize and rescale the window, use multi-touch actions (pinch & zoom, pan, rotate, tilt) and much more.

## **Inclusive App Development**

Making multiple builds is a past now. Build for one and test on multiple devices  using Cloud Test Lab Integration. Developers can check the compatibility and performance of an app on a wide range of physical Android devices from within Android Studio.

## **Better App Indexing**

Promoting is an important component of the app marketing, and **Android Studio 2.0**takes it to a new high. The App Indexing feature available in the IDE helps in creating and adding indexable URL links to the app.

Hope you found the information useful. Stay tuned with Root Info Solutions to receive more updates. Connect with us on [Facebook](https://www.facebook.com/RootInfoSolutions/), [Twitter](https://twitter.com/rootinfosol)and [LinkedIn](https://www.linkedin.com/company/contractor-for-php-drupal-mysql) to catch the latest trends, frameworks and practices linked with [Android apps development](https://www.rootinfosol.com/android-app-development).



**3.2 Web Application**

In computing, a **web application** or **web app** is a [client–server](https://en.wikipedia.org/wiki/Client%E2%80%93server_model) [computer program](https://en.wikipedia.org/wiki/Computer_program) which the client (including the user interface and client-side logic) runs in a [web browser](https://en.wikipedia.org/wiki/Web_browser).[]](https://en.wikipedia.org/wiki/Web_application#cite_note-1) Common web applications include [webmail](https://en.wikipedia.org/wiki/Webmail), [online retail sales](https://en.wikipedia.org/wiki/Online_shopping), [online auctions](https://en.wikipedia.org/wiki/Online_auction), [wikis](https://en.wikipedia.org/wiki/Wiki), [instant messaging services](https://en.wikipedia.org/wiki/Instant_messaging) and many other functions.

The general distinction between a [dynamic web page](https://en.wikipedia.org/wiki/Dynamic_web_page) of any kind and a "web application" is unclear. Web sites most likely to be referred to as "web applications" are those which have similar functionality to a desktop software application, or to a [mobile app](https://en.wikipedia.org/wiki/Mobile_App_Server). [HTML5](https://en.wikipedia.org/wiki/HTML5) introduced explicit language support for making applications that are loaded as web pages, but can store data locally and continue to function while offline.

[Single-page applications](https://en.wikipedia.org/wiki/Single-page_application) are more application-like because they reject the more typical web paradigm of moving between distinct pages with different URLs. Single-page frameworks like [Sencha Touch](https://en.wikipedia.org/wiki/Sencha_Touch) and [AngularJS](https://en.wikipedia.org/wiki/AngularJS) might be used to speed development of such a web app for a mobile platform.

### 3.2.1 Mobile web applications

Further information: [Multiple phone web-based application framework](https://en.wikipedia.org/wiki/Multiple_phone_web-based_application_framework)

There are several ways of targeting mobile devices when making a web application:

* [Responsive web design](https://en.wikipedia.org/wiki/Responsive_web_design) can be used to make a web application - whether a conventional web site or a single-page application viewable on small screens and work well with touchscreens.
* [Progressive Web Apps](https://en.wikipedia.org/wiki/Progressive_Web_Apps) are a hybrid of regular web pages (or websites) and a mobile application.
* [Native apps](https://en.wikipedia.org/wiki/Native_app) or "mobile apps" run directly on a mobile device, just as a conventional software application runs directly on a desktop computer, without a web browser (and potentially without the need for Internet connectivity); these are typically written in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) (for [Android](https://en.wikipedia.org/wiki/Android_(operating_system)) devices) or [Objective-C](https://en.wikipedia.org/wiki/Objective-C) or [Swift](https://en.wikipedia.org/wiki/Swift_(programming_language)) (for [iOS](https://en.wikipedia.org/wiki/IOS) devices). Recently, frameworks like [React Native](https://en.wikipedia.org/wiki/React_Native), [Flutter](https://en.wikipedia.org/wiki/Flutter_(software)) and [Xamarin](https://en.wikipedia.org/wiki/Xamarin) allow the development of native apps for all platforms using languages other than each standard native language.
* Hybrid apps embed a mobile web site inside a native app, possibly using a hybrid framework like [Apache Cordova](https://en.wikipedia.org/wiki/Apache_Cordova) and [Ionic](https://en.wikipedia.org/wiki/Ionic_(mobile_app_framework)) or [Appcelerator Titanium](https://en.wikipedia.org/wiki/Appcelerator_Titanium). This allows development using web technologies (and possibly directly copying code from an existing mobile web site) while also retaining certain advantages of native apps (e.g. direct access to device hardware, offline operation, [app store](https://en.wikipedia.org/wiki/App_store) visibility).

**3.2.2 Java Programming Language**

**Java** is a general-purpose [computer-programming language](https://en.wikipedia.org/wiki/Programming_language) that is [concurrent](https://en.wikipedia.org/wiki/Concurrent_computing), [class-based](https://en.wikipedia.org/wiki/Class-based_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "[write once, run anywhere](https://en.wikipedia.org/wiki/Write_once,_run_anywhere)" (WORA), meaning that [compiled](https://en.wikipedia.org/wiki/Compiler) Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to [bytecode](https://en.wikipedia.org/wiki/Java_bytecode) that can run on any [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine) (JVM) regardless of [computer architecture](https://en.wikipedia.org/wiki/Computer_architecture). As of 2016, Java is one of the most [popular programming languages in use](https://en.wikipedia.org/wiki/Measuring_programming_language_popularity)  particularly for client-server web applications, with a reported 9 million developers Java was originally developed by [James Gosling](https://en.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems)(which has since been [acquired by Oracle Corporation](https://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle)) and released in 1995 as a core component of Sun Microsystems' [Java platform](https://en.wikipedia.org/wiki/Java_(software_platform)). The language derives much of its [syntax](https://en.wikipedia.org/wiki/Syntax_(programming_languages)) from [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B), but it has fewer [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language) facilities than either of them.

The original and [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) Java [compilers](https://en.wikipedia.org/wiki/Compiler), virtual machines, and [class libraries](https://en.wikipedia.org/wiki/Library_(computing)) were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the [Java Community Process](https://en.wikipedia.org/wiki/Java_Community_Process), Sun [relicensed](https://en.wikipedia.org/wiki/Software_relicensing) most of its Java technologies under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License). Others have also developed alternative implementations of these Sun technologies, such as the [GNU Compiler for Java](https://en.wikipedia.org/wiki/GNU_Compiler_for_Java) (bytecode compiler), [GNU Classpath](https://en.wikipedia.org/wiki/GNU_Classpath) (standard libraries), and [IcedTea](https://en.wikipedia.org/wiki/IcedTea)-Web (browser plugin for applets).

The latest version is [Java 10](https://en.wikipedia.org/wiki/Java_version_history), released on March 20, 2018 which follows [Java 9](https://en.wikipedia.org/wiki/Java_version_history) after only six months in line with the new release schedule. Java 8 is still supported but there will be no more security updates for Java 9 Versions earlier than Java 8 are supported by companies on a commercial basis; e.g. by Oracle back to Java 6 as of October 2017 (while they still "highly recommend that you uninstall” pre-Java 8 from at least Windows computers).

### Versions

Main article: [Java version history](https://en.wikipedia.org/wiki/Java_version_history)

As of 20 March 2018, both Java 8 and 10 are officially supported. Major release versions of Java, along with their release dates:

* JDK 1.0 (January 23, 1996)
* JDK 1.1 (February 19, 1997)
* J2SE 1.2 (December 8, 1998)
* J2SE 1.3 (May 8, 2000)
* J2SE 1.4 (February 6, 2002)
* J2SE 5.0 (September 30, 2004)
* Java SE 6 (December 11, 2006)
* Java SE 7 (July 28, 2011)
* Java SE 8 (March 18, 2014)
* Java SE 9 (September 21, 2017)
* Java SE 10 (March 20, 2018)

#### Performance

Main article: [Java performance](https://en.wikipedia.org/wiki/Java_performance)

Programs written in Java have a reputation for being slower and requiring more memory than those written in C++. However, Java programs' execution speed improved significantly with the introduction of [just-in-time compilation](https://en.wikipedia.org/wiki/Just-in-time_compilation) in 1997/1998 for [Java 1.1](https://en.wikipedia.org/wiki/Java_version_history) the addition of language features supporting better code analysis (such as inner classes, the StringBuilder class, optional assertions, etc.), and optimizations in the Java virtual machine, such as [HotSpot](https://en.wikipedia.org/wiki/HotSpot) becoming the default for Sun's JVM in 2000. With Java 1.5, the performance was improved with the addition of the java.util.concurrent package, including [lock free](https://en.wikipedia.org/wiki/Lock_free) implementations of the [Concurrent Maps](https://en.wikipedia.org/wiki/Java_ConcurrentMap) and other multi-core collections, and it was improved further with Java 1.6.

Some platforms offer direct hardware support for Java; there are microcontrollers that can run Java in hardware instead of a software Java virtual machine, and some [ARM](https://en.wikipedia.org/wiki/ARM_architecture) based processors could have hardware support for executing Java bytecode through their [Jazelle](https://en.wikipedia.org/wiki/Jazelle) option, though support has mostly been dropped in current implementations of ARM.

**Use outside the Java Platform**

The Java programming language requires the presence of a software platform in order for compiled programs to be executed. Oracle supplies the [Java platform](https://en.wikipedia.org/wiki/Java_platform) for use with Java. The [Android SDK](https://en.wikipedia.org/wiki/Android_SDK) is an alternative software platform, used primarily for developing [Android applications](https://en.wikipedia.org/wiki/Android_application).

### Android:

[](https://en.wikipedia.org/wiki/File:Android_6.0-en.png)

The [Android](https://en.wikipedia.org/wiki/Android_(operating_system)) operating system makes extensive use of Java-related technology.

The Java language is a key pillar in [Android](https://en.wikipedia.org/wiki/Android_(operating_system)), an [open source](https://en.wikipedia.org/wiki/Open_source) [mobile operating system](https://en.wikipedia.org/wiki/Mobile_operating_system). Although Android, built on the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel), is written largely in C, the [Android SDK](https://en.wikipedia.org/wiki/Android_software_development#SDK) uses the Java language as the basis for Android applications. The bytecode language supported by the Android SDK is incompatible with Java bytecode and runs on its own virtual machine, optimized for low-memory devices such as [smartphones](https://en.wikipedia.org/wiki/Smartphone) and [tablet computers](https://en.wikipedia.org/wiki/Tablet_computer). Depending on the Android version, the bytecode is either interpreted by the [Dalvik virtual machine](https://en.wikipedia.org/wiki/Dalvik_virtual_machine) or compiled into native code by the [Android Runtime](https://en.wikipedia.org/wiki/Android_Runtime).

**3.3 Database**

SQLite is a [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) contained in a [C](https://en.wikipedia.org/wiki/C_(programming_language)) programming [library](https://en.wikipedia.org/wiki/Library_(computer_science)). In contrast to many other database management systems, SQLite is not a [client–server](https://en.wikipedia.org/wiki/Client%E2%80%93server) database engine. Rather, it is embedded into the end program.

SQLite is [ACID](https://en.wikipedia.org/wiki/ACID)-compliant and implements most of the [SQL](https://en.wikipedia.org/wiki/SQL) standard, using a dynamically and weakly [typed](https://en.wikipedia.org/wiki/Data_type) SQL [syntax](https://en.wikipedia.org/wiki/Syntax) that does not guarantee the [domain integrity](https://en.wikipedia.org/wiki/Domain_integrity).

SQLite is a popular choice as [embedded database](https://en.wikipedia.org/wiki/Embedded_database) software for local/client storage in [application software](https://en.wikipedia.org/wiki/Application_software) such as [web browsers](https://en.wikipedia.org/wiki/Web_browser). It is arguably the most widely deployed [database engine](https://en.wikipedia.org/wiki/Database_engine), as it is used today by several widespread browsers, [operating systems](https://en.wikipedia.org/wiki/Operating_system), and [embedded systems](https://en.wikipedia.org/wiki/Embedded_system) (such as mobile phones), among others. SQLite has [bindings](https://en.wikipedia.org/wiki/Language_binding) to many programming languages.

Unlike client–server database management systems, the SQLite engine has no standalone [processes](https://en.wikipedia.org/wiki/Process_(computing)) with which the application program communicates. Instead, the SQLite [library](https://en.wikipedia.org/wiki/Library_(computing))is [linked](https://en.wikipedia.org/wiki/Linker_(computing)) in and thus becomes an integral part of the application program. The library can also be called dynamically. The application program uses SQLite's functionality through simple [function calls](https://en.wikipedia.org/wiki/Subroutine), which reduce [latency](https://en.wikipedia.org/wiki/Latency_(engineering)) in database access: function calls within a single process are more efficient than [inter-process communication](https://en.wikipedia.org/wiki/Inter-process_communication). SQLite stores the entire database (definitions, tables, indices, and the data itself) as a single cross-platform [file](https://en.wikipedia.org/wiki/Computer_file) on a host machine. It implements this simple design by [locking](https://en.wikipedia.org/wiki/Lock_(computer_science)) the entire database file during writing. SQLite read operations can be multitasked, though writes can only be performed sequentially.

Due to the server-less design, SQLite applications require less configuration than client-server databases. SQLite is called zero-conf because it does not require service management (such as startup scripts) or access control based on [GRANT](https://en.wikipedia.org/wiki/Data_control_language) and passwords. Access control is handled by means of [file system permissions](https://en.wikipedia.org/wiki/File_system_permissions)

given to the database file itself. Databases in client-server systems use file system permissions which give access to the database files only to the daemon process.

Another implication of the serverless design is that several processes may not be able to write to the database file. In server-based databases, several writers will all connect to the same daemon, which is able to handle its locks internally. SQLite on the other hand has to rely on file-system locks. It has less knowledge of the other processes that are accessing the database at the same time. Therefore, SQLite is not the preferred choice for write-intensive deployments.

 However, for simple queries with little concurrency, SQLite performance profits from avoiding the overhead of passing its data to another process.

SQLite uses [PostgreSQL](https://en.wikipedia.org/wiki/PostgreSQL) as a reference platform. “What would PostgreSQL do” is used to make sense of the SQL standard. One major deviation is that, with the exception of primary keys, SQLite does not enforce type checking; the type of a value is dynamic and not strictly constrained by the schema (although the schema will trigger a conversion when storing, if such a conversion is potentially reversible). SQLite strives to follow [Postel's Rule](https://en.wikipedia.org/wiki/Robustness_principle).

**Features:**

SQLite implements most of the [SQL-92](https://en.wikipedia.org/wiki/SQL-92) standard for [SQL](https://en.wikipedia.org/wiki/SQL) but it lacks some features. For example, it partially provides [triggers](https://en.wikipedia.org/wiki/Database_trigger), and it cannot write to [views](https://en.wikipedia.org/wiki/View_(database)) (however it provides INSTEAD OF triggers that provide this functionality). While it provides complex queries, it still has limited [ALTER TABLE](https://en.wikipedia.org/wiki/Data_Definition_Language#ALTER_statements) function, as it cannot modify or delete columns.

SQLite uses an unusual [type system](https://en.wikipedia.org/wiki/Type_system) for an SQL-compatible DBMS; instead of assigning a [type](https://en.wikipedia.org/wiki/SQL_data_types) to a column as in most SQL database systems, types are assigned to individual values; in language terms it is dynamically typed. Moreover, it is weakly typed in some of the same ways that [Perl](https://en.wikipedia.org/wiki/Perl) is: one can insert a [string](https://en.wikipedia.org/wiki/String_(computer_science)) into an [integer](https://en.wikipedia.org/wiki/Integer) column (although SQLite will try to convert the string to an integer first, if the column's preferred type is integer). This adds flexibility to columns, especially when bound to a dynamically typed scripting language. However, the technique is not portable to other SQL products. A common criticism is that SQLite's type system lacks the data integrity mechanism provided by statically typed columns in other products. The SQLite web site describes a "strict affinity" mode, but this feature has not yet been added. However, it can be implemented with constraints like CHECK(typeof(x)='integer'). SQLite with full [Unicode](https://en.wikipedia.org/wiki/Unicode) function is optional.

Several [computer processes](https://en.wikipedia.org/wiki/Computer_process) or [threads](https://en.wikipedia.org/wiki/Thread_(computer_science)) may access the same database concurrently. Several read accesses can be satisfied in parallel. A write access can only be satisfied if no other accesses are currently being serviced. Otherwise, the write access fails with an [error code](https://en.wikipedia.org/wiki/Error_code) (or can automatically be retried until a configurable timeout expires). This concurrent access situation would change when dealing with temporary tables. This restriction is relaxed in version 3.7 when [write-ahead logging](https://en.wikipedia.org/wiki/Write-ahead_logging) (WAL) is turned on enabling concurrent reads and writes.

SQLite version 3.7.4 first saw the addition of the FTS4 (full text search) module, which features enhancements over the older FTS3 module. FTS4 allows users to perform full text searches on documents similar to how search engines search webpages. Version 3.8.2 added support for creating tables without [rowid](https://en.wikipedia.org/wiki/Pseudocolumn), which may provide space and performance improvements [Common table expressions](https://en.wikipedia.org/wiki/Common_table_expressions) support was added to SQLite in version 3.8.3. In 2015, with the json1 extension and new subtype interfaces, SQLite version 3.9 introduced [JSON](https://en.wikipedia.org/wiki/JSON) content managing.

**Development and Distribution**

SQLite's code is hosted with [Fossil](https://en.wikipedia.org/wiki/Fossil_(software)), a [distributed version control system](https://en.wikipedia.org/wiki/Distributed_version_control_system) that is itself built upon an SQLite database.

A standalone [command-line](https://en.wikipedia.org/wiki/Console_application) program is provided in SQLite's distribution. It can be used to create a database, define tables, insert and change rows, run queries and manage an SQLite database file. It also serves as an example for writing applications that use the SQLite library.

SQLite uses automated [regression testing](https://en.wikipedia.org/wiki/Regression_testing) prior to each release. Over 2 million tests are run as part of a release's verification. Starting with the August 10, 2009 release of SQLite 3.6.17, SQLite releases have 100% branch test coverage, one of the components of [code coverage](https://en.wikipedia.org/wiki/Code_coverage). The tests and [test harnesses](https://en.wikipedia.org/wiki/Test_harness) are partially public domain and partially [proprietary](https://en.wikipedia.org/wiki/Proprietary_software).

