

Software & SDLC

Jieung Kim

Aug. 2022

Software &
software engineer

Software concepts

- Computer
 - An electronic device that can perform various operations of computation
 - It is driven by many entities
 - **Hardware:** the physical and tangible components that can touch or feel
 - **Software:** Programs and commands that drive the hardware

Software vs. program

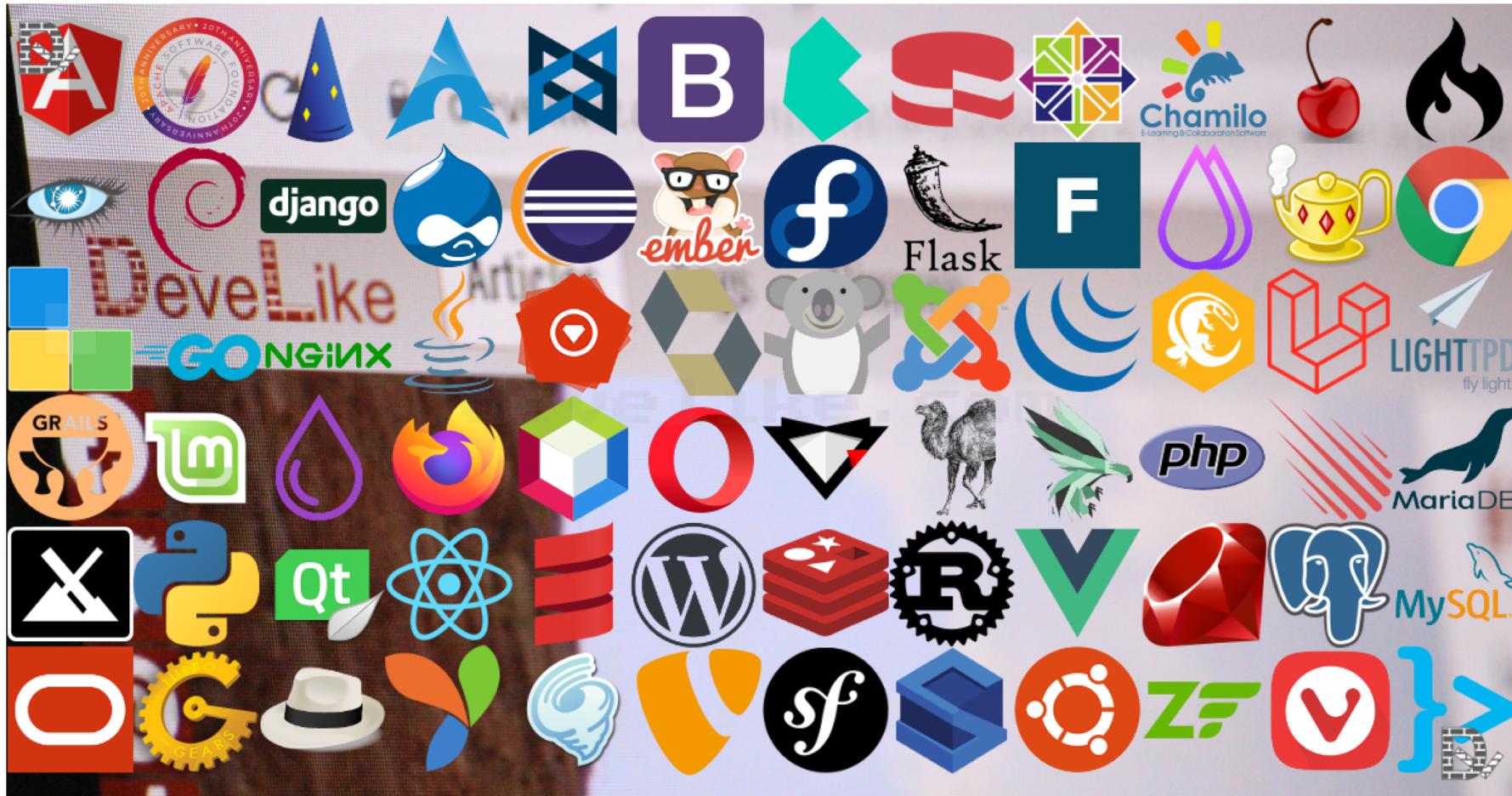
- Software
 - A collection or set of programs procedures, data or instructions
 - They instruct computer about what to do
 - They are designed to perform well defined functions
- Program
 - A collection of instructions or ordered operations for computer
 - They perform specific function or perform a particular task
 - They achieve a specific result



Software vs. program

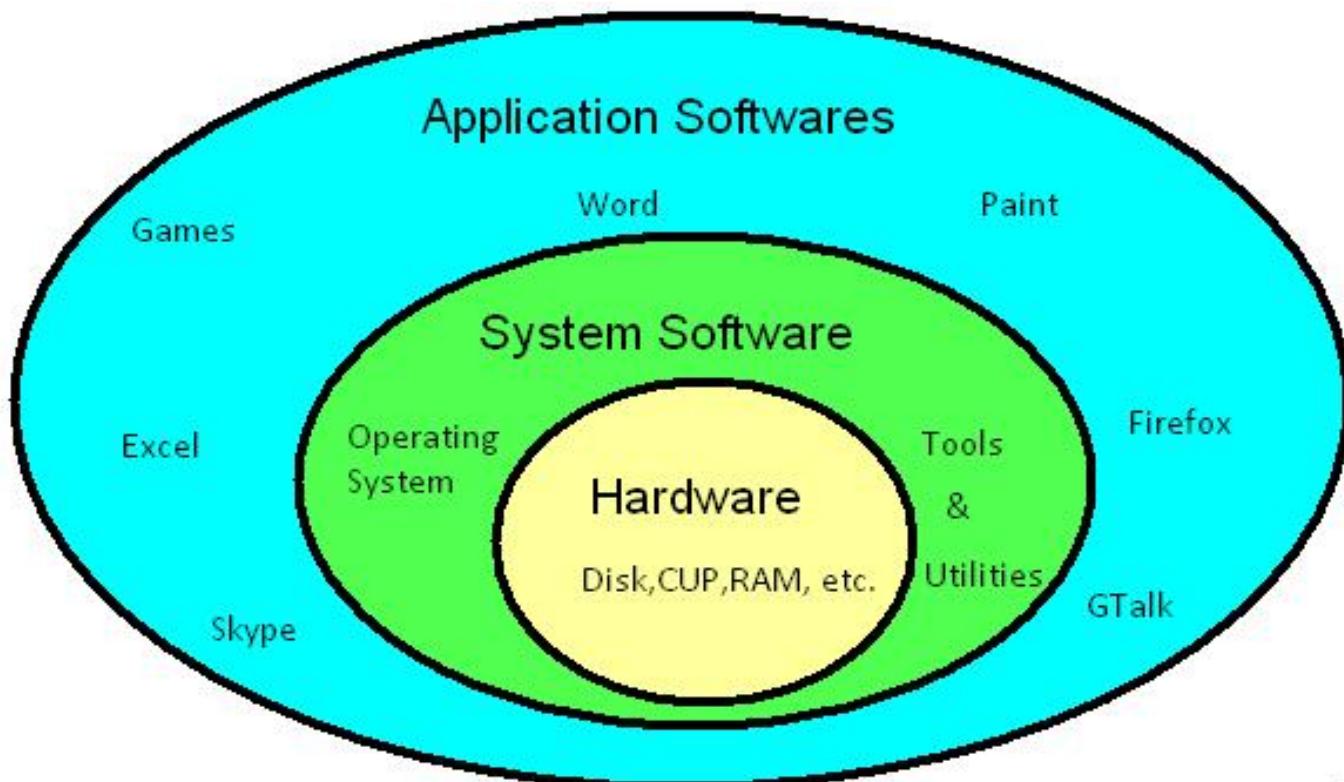
Software	Program
Various categories of software includes application software, system software, computer programming tools, etc.	There are no such categories of program.
Size of software generally ranges from megabytes (Mb) to gigabytes (Gb).	Size of program generally ranges from kilobytes (Kb) to megabytes (Mb).
Software's are usually developed by people having expert knowledge and experience as well as are trained in developing software and are also referred to as software developers.	Programs are usually developed by person who is beginner and have no prior experience.
Software's can be a program that generally runs on computer.	Programs cannot be a software.
It requires more time to create software than program.	It requires less time to create program than software.
...	...

Software



Software

- Depending on its use and area of implementation, Software can be divided into two major types
 - System Software
 - Application Software



System software

- It allows the user to **interact with the hardware components** of a computer system
- The **interface is provided** by the software
 - Interface: allows the users to interact with the core system
- The system software can be called **the main or the alpha software** of a computer system
 - It handles the major portion of running hardware
- It can be further divided into several major types

System software

- **The Operating System**
 - **Governs and maintains the inter-cooperation of the components** of a computer system
- **The Language Processor**
 - **Provides the methods for human-machine interactions with multiple languages** (i.e., machine-level, assembly-level, and high-level languages)
 - Can be divided into three types
 - **Assembler:** Convert the assembly language into machine level language
 - **Compiler:** Convert High-Level Language into machine level language in one go.
 - Execution time is fast, but error detection is difficult in a compiler
 - C, C++ and Scala use compilers
 - **Interpreter:** Convert High-Level Language into machine level language line-by-line
 - Execution time is slow, but error-detection is easier in an interpreter as it reports as soon as a bug is caught and restarts the process. This consumes unnecessary memory
 - Python, Ruby and Java use an interpreter

System software

- **Utility software**
 - The most basic type of software
 - Helping the management of the system
 - Examples
 - Antivirus Software
 - provide protection to the computer system from unwanted malware and viruses
 - V3, QuickHeal, McAfee etc
 - Disk Defragmenter Tools
 - Help the users to analyze the bad sectors of the disk and rearrange the files in a proper order
 - Text-editors
 - Help the users to take regular notes and create basic text files
 - Notepad, Gedit, Emacs, Vim, etc

System software

- **Device Drivers**
 - **Acts as an interface** between the various Input-Output **devices** and the **users** or the **operating system**
 - For example, printers, and web cameras come with a driver disk that is needed to be installed into the system to make the device run in the system
- **BIOS**
 - **Basic Input Output System**
 - It is a small firmware that **controls the peripheral or the input-output devices** attached to the system
 - Also **responsible for starting the OS** or **initiating the booting process**

Application software

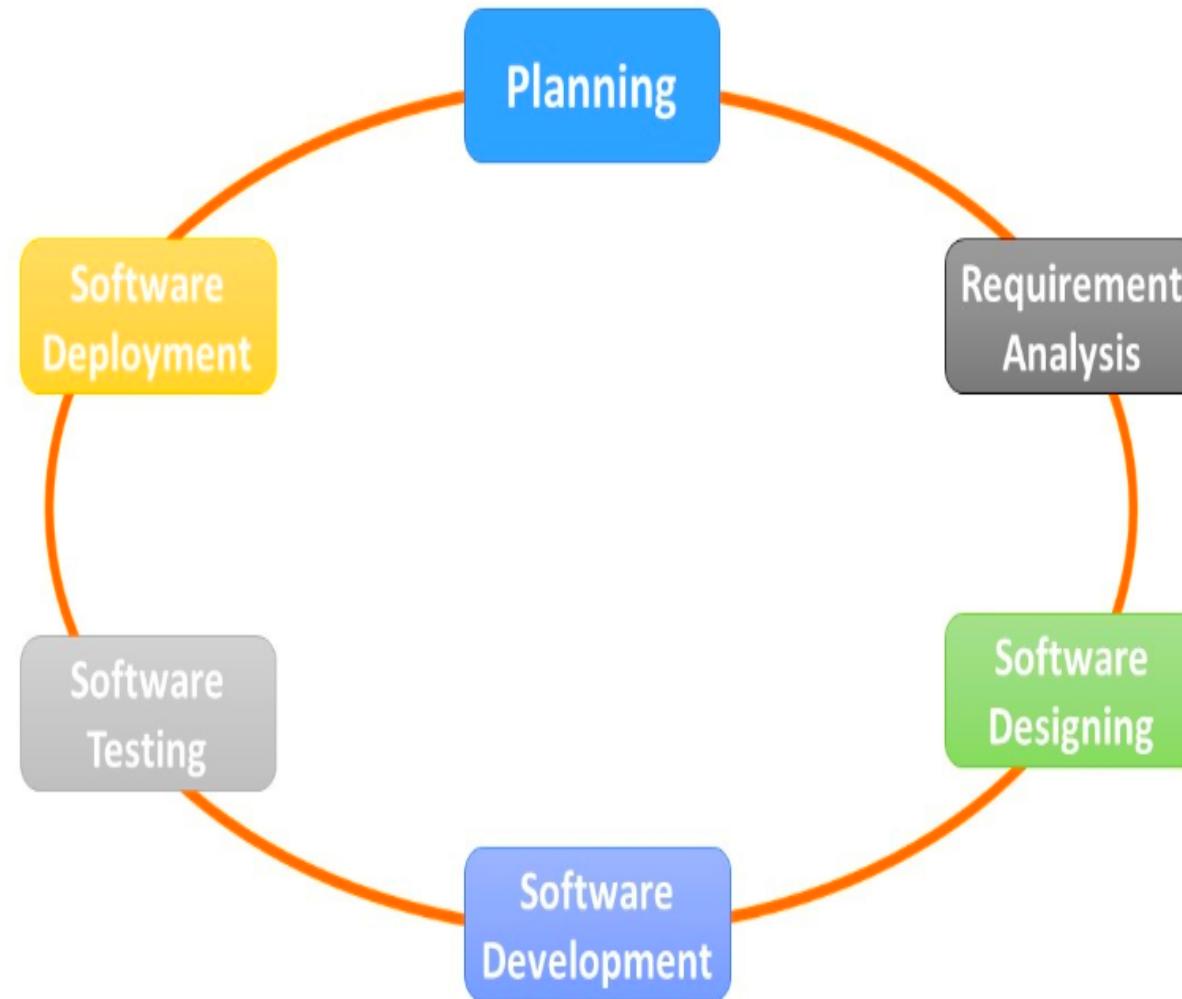
- Used to run to accomplish a particular action and task
- Dedicated to performing simple and single tasks
 - E.g., a single software cannot serve both the reservation system and banking system
- Divided into two types:
 - **General Purpose Application Software:** comes in-built and ready to use, manufactured by some company or someone
 - Microsoft Excel – Used to prepare excel sheets
 - VLC Media Player – Used to play audio/video files
 - Adobe Photoshop – Used for designing and animation and many more
 - **Specific Purpose Application Software:** Is customizable and mostly used in real-time or business environments
 - Ticket Reservation System
 - Healthcare Management System
 - Hotel Management System
 - Payroll Management System

Software development life cycle (SDLC)

Software development life cycle (SDLC)

- SDLC is a **Common term of an efficient approach** for the development
 - A process used in the software industry to produce software with the highest quality and lowest cost in the shortest time
- The entire software development process includes **6 stages**
- It **Specifies the task(s)** to be performed at various stages by a software engineer/developer
- It's **vital for a software developer to have prior knowledge** of this software development process
- Google also educates the similar thing with this during the orientation

Software development life cycle (SDLC)



1. Planning phase

- **Input:** Client proposal or requirements
- **Requirement analysis** is the most important and fundamental phase in SDLC
- Senior members (with other team members) perform the following things to define the **various technical approaches** to implement the project successfully with minimum risks
 - The basic project approach
 - The product feasibility study in the economical, operational and technical areas
 - Quality assurance requirements
 - Risk identification associated with the project
- **Output:** Technical Feasibility Study/Project initiation

2. Analysis phase

- **Input:** Technical feasibility study and funding
- Clearly **define and document the product requirements** and get them approved from the customer or the market analysts
- **Output**
 - Software Requirement Specification (SRS)
 - Consists of all the product requirements to be designed and developed during the project life cycle

3. Design phase

- **Input:** SRS
- Write a **Design Document Specification (DDS)**
 - Contains more than one design approach for the product architecture
 - Defines all the architectural modules of the product along with communications and data flows of sub modules and third parties
 - Be reviewed by all the important stakeholders and based on various parameter (e.g., risk assessment, design modularity, etc)
 - The internal design of all the modules should be clearly defined
- Write **high level design (HLD) / detail design (DDD)** based on **DDS**
- **Output:** DDS, HLD, and/or DDD

4. Build phase (implementation)

- **Input:** DDD (Detail Design Document)
- The **actual development starts** and the product is built based on DDD
 - *If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle*
- Developers must follow the coding guidelines (e.g., [Google Python style guide](#))
- **Output:** Program/software

5. Test phase

- **Input:** Un-tested program/software, Test plan document
- Perform tests based on SRS
 - Testing only stage where product defects are reported, tracked, fixed and retested
 - Continuing it until the result reaches the quality standards defined in the SRS
- **Output:** tested program/software

6. Deployment phase

- **Input:** Tested program/software, migration plan
- Released formally in the appropriate market as per the business strategy of that organization
- The product may first be released in a limited segment and tested in the real business environment (**UAT- User acceptance testing**)
- After the product is released in the market, its maintenance is done for the existing customer base
- **Output:** Tested program/software migrated into production and celebrate

SDLC models

- Waterfall model
- Agile model
- Iterative model
- V-shape model
- Big Bang model
- Spiral model
- ...

