Framework VS Library

Both the framework vs library is precoded support programs to develop complex software applications. However, libraries target a specific functionality, while a framework tries to provide everything required to develop a complete application. So, when you develop a software application, you will need many libraries, but often one or two frameworks. Popular examples of frameworks are Ext JS, Angular, Django, Spring, and Rails, which offer a comprehensive set of tools and components for application development. On the other hand, popular examples of libraries are React and jQuery, which focus on specific tasks or functionalities and can be used in conjunction with frameworks. The choice between a framework vs library depends on the specific requirements and scope of the project at hand. You decide to keep your code DRY (don’t repeat yourself) and write some reusable function.

System vs Application

system software is a designed platform to run the hardware resources, and application software is designed to fulfill the end-user's requirements. System software provides a platform for other software to run, while application software performs specific tasks for the user, so both are necessary in for your computer to work properly.

Task vs Thread

Asynchronous Programming has become a popular trend in modern software development. Two commonly used techniques for Asynchronous Programming in C# are Tasks and Threads.

Task is a higher-level abstraction for running code asynchronously. A Task denotes a unit of work that needs to be executed asynchronously, and it may or may not return a value. Tasks are more lightweight than Threads.

Thread is a lower-level abstraction for running code asynchronously. A Thread represents an operating system-level construct that is used to execute code asynchronously. A Thread may or may not return a value, and it is usually created with the help of the Thread class.