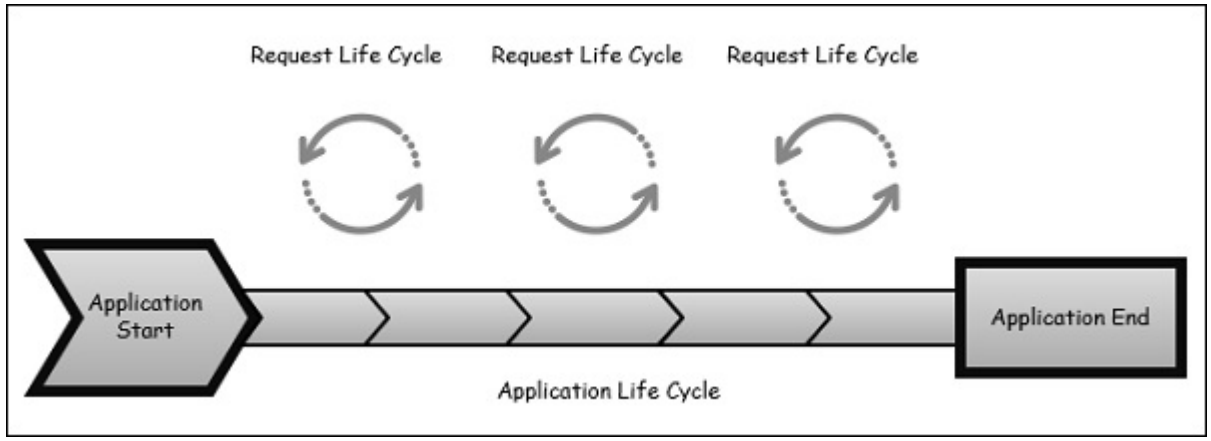
ASP.NET MVC - Life Cycle

In this chapter, we will discuss the overall MVC pipeline and the life of an HTTP request as it travels through the MVC framework in ASP.NET. At a high level, a life cycle is simply a series of steps or events used to handle some type of request or to change an application state. You may already be familiar with various framework life cycles, the concept is not unique to MVC.

For example, the ASP.NET webforms platform features a complex page life cycle. Other .NET platforms, like Windows phone apps, have their own application life cycles. One thing that is true for all these platforms regardless of the technology is that understanding the processing pipeline can help you better leverage the features available and MVC is no different.

MVC has two life cycles −

- The application life cycle
- The request life cycle



The Application Life Cycle

The application life cycle refers to the time at which the application process actually begins running IIS until the time it stops. This is marked by the application start and end events in the startup file of your application.

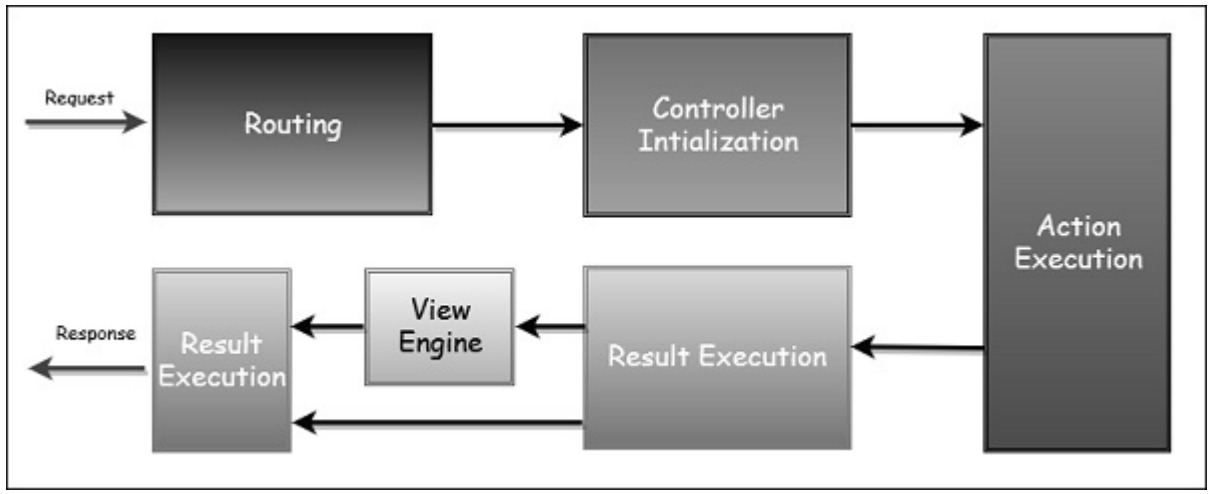
The Request Life Cycle

It is the sequence of events that happen every time an HTTP request is handled by our application.

The entry point for every MVC application begins with routing. After the ASP.NET platform has received a request, it figures out how it should be handled through the URL Routing Module.

Modules are .NET components that can hook into the application life cycle and add functionality. The routing module is responsible for matching the incoming URL to routes that we define in our application.

All routes have an associated route handler with them and this is the entry point to the MVC framework.



The MVC framework handles converting the route data into a concrete controller that can handle requests. After the controller has been created, the next major step is **Action Execution**. A component called the **action invoker** finds and selects an appropriate Action method to invoke the controller.

After our action result has been prepared, the next stage triggers, which is **Result Execution**. MVC separates declaring the result from executing the result. If the result is a view type, the View Engine will be called and it's responsible for finding and rending our view.

If the result is not a view, the action result will execute on its own. This Result Execution is what generates an actual response to the original HTTP request.