**Why should you care about how long it takes your page to load?**

**Well, studies have shown that most people won’t wait around longer than 5 seconds for your page to load and some will wait even less. If your application is slow to load then your customers will be less likely to stay on your site which translates into fewer conversions.**

So now I’ve got you convinced. You need lazy loading,Eager loading or preloading in your application. Only one problem, you still don’t know what are types of loading.

So let’s fix that:

**1. What is Eager Loading?**

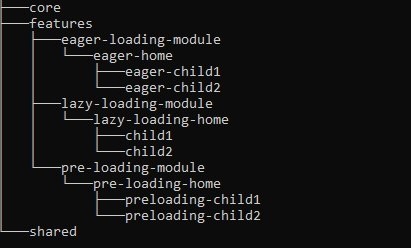
Feature modules under Eager Loading would be loaded before the application starts. This is the default module-loading strategy.

**2. What is Lazy Loading?**

Feature modules under Lazy Loading would be loaded on demand after the application starts. It helps to start application faster.

**3. What is Pre-Loading?**

Feature Modules under Pre-Loading would be loaded automatically after the application starts.



When you compile an Angular application, the Angular compiler creates a javascript bundle which is used to run your application on the client. These bundles include all of the transpiled javascript, the HTML, and CSS that you have written.

As the amount of features in your application grows, the amount of code also grows. This results in a large bundle size that will end up being served to your end-user. Most ordinary users will have no idea what a bundle of code is much less care about how big it is, however they will care about how long it takes for your application to load. The size of the javascript bundle has a direct correlation to the time that it takes for your page to load. Smaller bundle, quicker page load and a bigger bundle results in a longer page load.

**When to use *Eager Loading, Lazy Loading, Pre-Loading* in Angular?**

**When to use Eager Loading?**

Case 1: Small size applications. In this case, it’s not expensive to load all modules before the application starts, and the application will be faster and more responsive to process requests.

Case 2: Core modules and feature modules that are required to start the application. These modules could contain components of the initial page, interceptors (for authentication, authorization, and error handling, etc.), error response components, top-level routing, and localization, etc.

We just have to eagerly load these modules to make the application function properly despite the application size.

**When to use Lazy Loading?**

The scenario of applying Lazy Loading is relatively simple and straightforward. In a big-size web application, we can lazily load all other modules that are not required when the application starts.

**When to use Pre-Loading?**

Compared with Eager Loading and Lazy Loading, Pre-Loading is not so much frequently used in web application development. Based on my understanding of this loading strategy, Pre-Loading would be favorable for two cases though.

Case 1: Medium size application. In this scenario, we can make the application start faster since it will load all other modules later that are not required to run the application. And the application would be more responsive to process users’ requests than applying Lazy Loading strategy since the application will load all these modules after the application started.

Case 2: Some specific modules that users are very likely to use after the application started. In this scenario, we can pre-load these feature modules and still lazy load other modules.

**Summary**

* Eager Loading: used to load core modules and feature modules that are required to start the application.
* Pre-Loading: used to load specific feature modules that are very likely to be used soon after the application started.
* Lazy Loading: all other modules could be lazily loaded on demand after the application started.

**Conclusion**

**By now you should have some understanding how to implement Eager loding,lazy loading and Pre-loading angular (-- how to add it to your application--).**

**So,It is an essential part of any Angular application as it allows for a better user experience. I also find that it allows for better code organization which enhances the developer experience and optimization for his application.It really is a win-win.**

**threfore you can find more details on their angular.io as well**