Asynchronous providers

When the application start has to be delayed until some **asynchronous tasks** will be finished, for example, until the connection with the database will be established, you should consider using asynchronous providers. In order to create an `async` provider, we use the `useFactory`. The factory has to return a `Promise` (thus `async` functions fit as well).

```
{
  provide: 'ASYNC_CONNECTION',
  useFactory: async () => {
    const connection = await createConnection(options);
    return connection;
  },
}
```

> **HINT**
> Learn more about the custom providers syntax here.

Injection

The asynchronous providers can be simply injected to other components by their tokens (in the above case, by the `ASYNC_CONNECTION` token). Each class that depends on the asynchronous provider will be instantiated once the async provider is **already resolved**.

The above example is for demonstration purposes. If you're looking for more detailed one, see here.

Support us

Nest is an MIT-licensed open source project. It can grow thanks to the support by these awesome people. If you'd like to join them, please read more here.

Principal Sponsor

Sponsors / Partners

Become a sponsor