Controller

# Questions

1. How nestjs know to which controller to give requests?
2. What is routing table and how it works in nestjs
3. How Nest takes controller method and assign it specific path (how pipes, guards … are applied)?
4. What are controller http methods like @Post @Get?
5. How nest js assigns metadata to controller methods?
6. What is Injectable scope (Scope.REQUEST)
7. Difference between injectable and providers in Module file?
8. What are enhancers?
9. What is DependenciesScanner in NestJs and how it is working?
10. What is InternalCoreModule and why do I need it?
11. What is forwardRef in context of module
12. How moduleRef is working, how knows which provider to extract
13. How ApplicationConfig it is used?
14. What is InstanceLoader and how it is used?
15. What is serialize-graph and how it is used?
16. What is use of InternalCoreModuleFactory?
17. How controllers methods access external providers?
18. How nest pipes validate dto?
19. How controller method params are send in correct order to method?
20. How routing table is created
21. How incoming request is mapped and how it is matched to route from routing tree
22. What is routing mechanism?
23. What is routing map?
24. What metadata @Controller associate to decorated class
25. What metadata is associated with controller decorator?
26. How nest knows to which method to receive concrete request?
27. How nest injects @Req in controllers methods?
28. What algorithms nest uses to optimize the routing table
29. What is instance wrapper
30. What is Injectable scope (Scope.REQUEST)
31. How method param is destructed?
32. How request mapping is happening?

# How it works

Controllers are responsible for handling incoming request and returning some response to client

There is some thing named **routing mechanism** that receive request to specific controller

To create controller we use decorator for some class, @Controller decorator associate to our class specific metadata that will help nest to work with our controller class

Always use classes for DTO because once compiled classes are part of Javascript and they are preserved as real entities

# How could use

Controller is used to create endpoints through which we will receive request work with it and return some response

# Explain as simple as possible

Controller is handler of event (request)

# Responses for questions

1. *How nestjs know to which controller to give requests?*

Under the hood nestjs transform each controller and it’s methods in tree-structure named **routing tree**   
Once request is handled **nestjs search for appropriate route in routing tree** that to which it could receive that request, in moment that route is mached nest takes it’s method and receive to it incoming request

1. *What is routing table and how it works in nestjs?*

Routing table is data **structure that maps incoming http request to appropriate handler method** in controller. Routing table checks which method from controller should be invoked from controller by matching route path and http method. Route table is created in time of bootstrapping of application.

1. How Nest takes controller method and assign it specific path (how pipes, guards … are applied)?

Word file with explanations:

‘nest\_from\_path\_till\_controller\_method.docx’

Github:

<https://github.com/dimapascal/learn-nestjs/tree/explore_routing_table>

Folder with fotos:

‘nest\_from\_path\_till\_controller\_method’

1. What are controller http methods like @Post @Get?

@Get and @Post are decorators that assign to controller method metadata using Reflect which is required for nest to make route-map

1. How nest js assigns metadata to controller methods?

Nestjs assigns metadata to controllers methods through decorators like @Post @Get

Under the hood nestJs uses Reflect (<https://www.npmjs.com/package/reflect-metadata>) to assign some metadata to methods params and so on

Request methods file: packages/common/decorators/http/request-mapping.decorator.ts

In case of methods metadata nest using Reflect assigns to our controller method it’s path (‘/home’) and method (‘post’, ‘get’, ….)

Reflect.defineMetadata(PATH\_METADATA, path, descriptor.value);

Reflect.defineMetadata(METHOD\_METADATA, requestMethod, descriptor.value);

1. What is scope in nest (Scope.REQUEST)

Scope describes life-time of instance (dependency)

It means how much dependency should leave

Scope.DEFAULT or Scope.SINGLETON: Creates one instance across the application

Scope.REQUEST: Create istance per request and destroy when request is received.

Scope.TRANSIENT: Create new instance every time that dependency is requested

1. Difference between injectable and providers in Module file?

Providers: classes, are static scope (Scope.DEFAULT)

Injectable: pipes guards interceptors and filters, are generated per request or by requirement (Scope.TRANSIENT, Scope.REQUEST)

1. What are enhancers?

Enhancers is common name for guards pipes interceptors and middlewares

1. What is DependenciesScanner in NestJs and how it is working ?

Word file with explanations:

‘dependencies\_scanner\_what\_and\_how.docx’

Github:

<https://github.com/dimapascal/learn-nestjs/tree/dependencies_scanner>

1. What is InternalCoreModule and why do I need it?

Word file with explanations:

‘internal\_core\_module.docx’

Github:

<https://github.com/dimapascal/learn-nestjs/tree/internal_core_module>