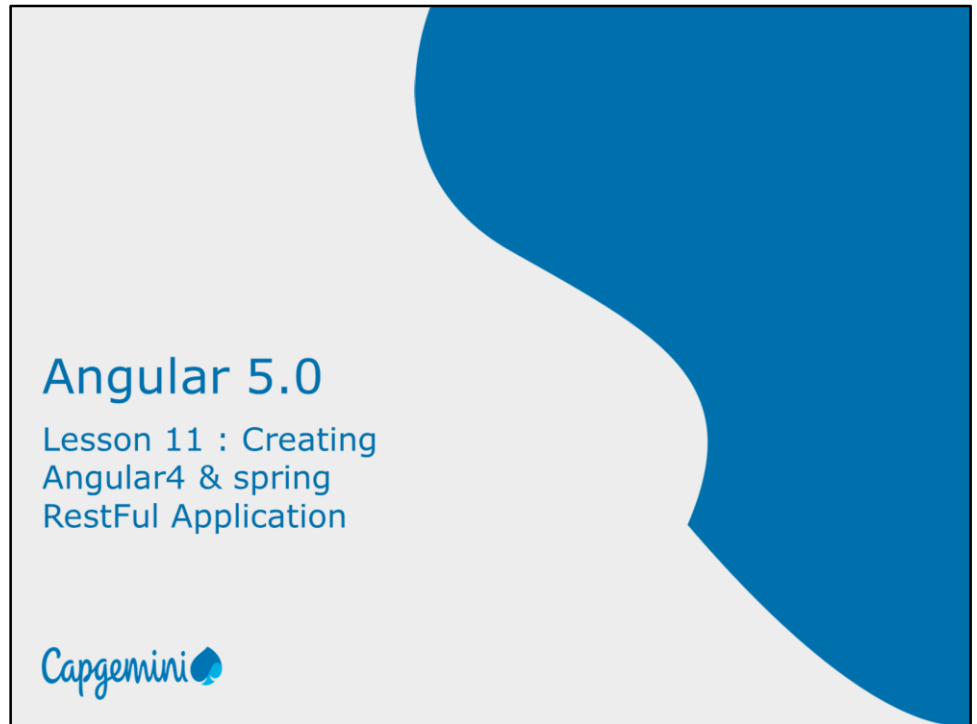


Instructor Notes:

Add instructor notes here.



Instructor Notes:

Add instructor notes here.

Lesson Objectives

- Angular5 & spring restful Application



Instructor Notes:

Angular With Java Base Middleware (Spring)



- Angular http is an angular module for interacting with REST based services.
- We are going to learn with a small project with Spring with RESTFUL
- Observable is a RxJS API. Observable is a representation of any set of values over any amount of time.
- All angular Http methods return instance of Observable. Find some of its operators.
- map: It applies a function to each value emitted by source Observable and returns finally an instance of Observable.
- catch: It is called when an error is occurred. catch also returns Observable.

We will perform create operation using Angular Http.post() method. It hits the request URL using HTTP POST method. Http.post() method syntax is as follows.

post(url: string, body: any, options?: RequestOptionsArgs) :
Observable<Response>

The description of parameters is given as below.

url: This is the REST web service URL to create article.

body: This is of any type object that will be passed to REST web service server. In our example we will create an Angular class as Article and pass its instance to body parameter.

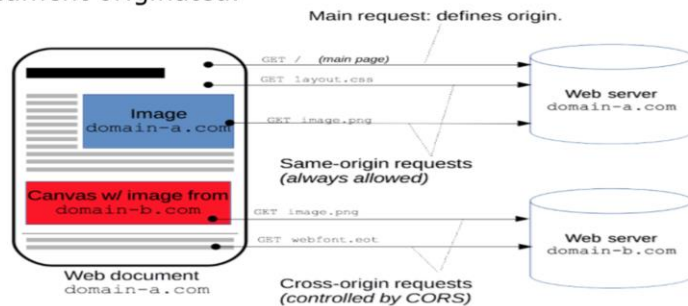
options: This is optional. This accepts the instance of Angular RequestOptions that is instantiated using Angular RequestOptionsArgs. Using RequestOptions we pass request parameter, request headers etc.

Http.post() returns instance of Observable. Observable is a representation of any set of values over any amount of time.

Instructor Notes:

Cross-Origin Resource Sharing (CORS)

- Cross-Origin Resource Sharing (CORS) is a mechanism that uses additional HTTP headers to let a user agent gain permission to access selected resources from a server on a different origin (domain) than the site currently in use.
- **HTTP request** when it requests a resource from a different domain, protocol, or port than the one from which the current document originated.



The only allowed methods are:

[GET](#)
[HEAD](#)
[POST](#)

Apart from the headers set automatically by the user agent (for example, [Connection](#), [User-Agent](#), or [any of the other headers with names defined in the Fetch spec as a "forbidden header name"](#)), the only headers which are allowed to be manually set are [those which the Fetch spec defines as being a "CORS-safelisted request-header"](#), which are:

[Accept](#)
[Accept-Language](#)
[Content-Language](#)
[Content-Type](#) (but note the additional requirements below)
[Last-Event-ID](#)
[DPR](#)
[Save-Data](#)
[Viewport-Width](#)
[Width](#)

The only allowed values for the [Content-Type](#) header are:

`application/x-www-form-urlencoded`
`multipart/form-data`
`text/plain`

No event listeners are registered on any [XMLHttpRequestUpload](#) object used in the request; these are accessed using the [XMLHttpRequest.upload](#) property.

No [ReadableStream](#) object is used in the request.

Instructor Notes:

Cross-Origin Resource Sharing (CORS)

- The CORS mechanism supports secure cross-domain requests and data transfers between browsers and web servers.
- Modern browsers use CORS in an API container such as XMLHttpRequest or Fetch to help mitigate the risks of cross-origin HTTP requests.
- This cross-origin sharing standard is used to enable cross-site HTTP requests for:
 - Invocations of the XMLHttpRequest or Fetch APIs in a cross-site manner, as discussed above.
 - Web Fonts (for cross-domain font usage in @font-face within CSS), so that servers can deploy TrueType fonts that can only be cross-site loaded and used by web sites that are permitted to do so.
 - WebGL textures.
 - Images/video frames drawn to a canvas using drawImage.
 - Stylesheets (for CSSOM access).
 - Scripts (for unmuted exceptions).

In our code created filter &
`response.addHeader("Access-Control-Allow-Origin", "*");`
`response.addHeader("Access-Control-Allow-Methods", "POST, GET, OPTIONS, PUT, DELETE, HEAD");`
`response.addHeader("Access-Control-Allow-Headers", "X-PINGOTHER, Origin, X-Requested-With, Content-Type, Accept");`
`response.addHeader("Access-Control-Max-Age", "1728000");`

Instructor Notes:

Angular With Java Base Middleware (Spring)



- Angular2 Method We are Using for Calling Spring Rest service
- `http.get` Shortcut method to perform GET request.
- `http.delete` shortcut method to perform DELETE request
- `http.post` shortcut method to perform post request
- `http.put` shortcut method to perform put request

We will perform read operation using Angular `Http.get()` method. It hits the URL using HTTP GET method. Find its syntax.

`get(url: string, options?: RequestOptionsArgs) : Observable<Response>`

Find the description of the parameters.

url: Web service URL to read article.

options: This is optional. It is used to pass request parameter, headers etc.

`Http.get()` returns the instance of `Observable`.

`Http.put`

We will perform update operation using Angular 2 `Http.put()` method. It hits the URL using HTTP PUT method. Find its syntax.

`put(url: string, body: any, options?: RequestOptionsArgs) :`

`Observable<Response>`

Find the description of parameters.

url: This is the REST web service URL to update article.

body: This is of any type object that will be passed to REST web service server. In our example we will create an Angular class as `Article` and pass its instance to `body` parameter.

options: This is optional. This is used to pass request parameter, request headers etc.

`Http.put()` returns the instance of `Observable`.

Instructor Notes:

Add instructor notes here.

Demo

- Crud Demo angular With spring
- Spring Static
- Spring with Jpa



Instructor Notes:

Lab

- Lab 05



Instructor Notes:

Add instructor notes here.

Summary

- Angular http is an angular module for interacting with REST based services. We are going to learn with a small project with Spring with RESTFUL
- CORS features



Add the notes here.