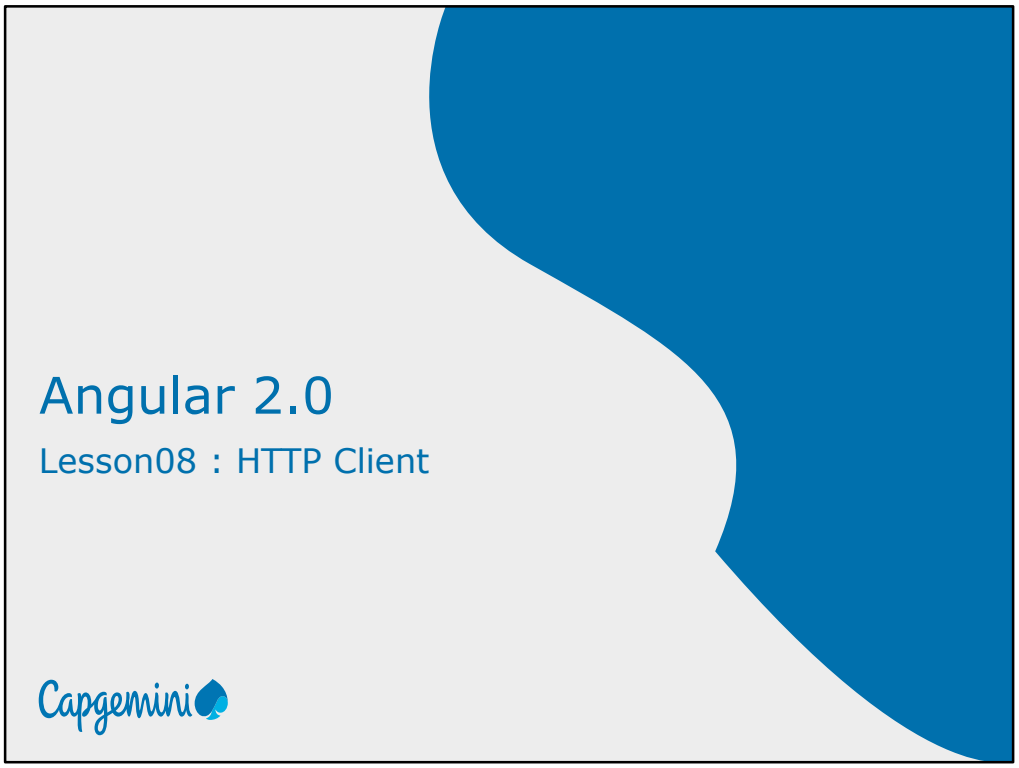
Instructor Notes:

Add instructor notes here.



Instructor Notes:

Add instructor notes here.

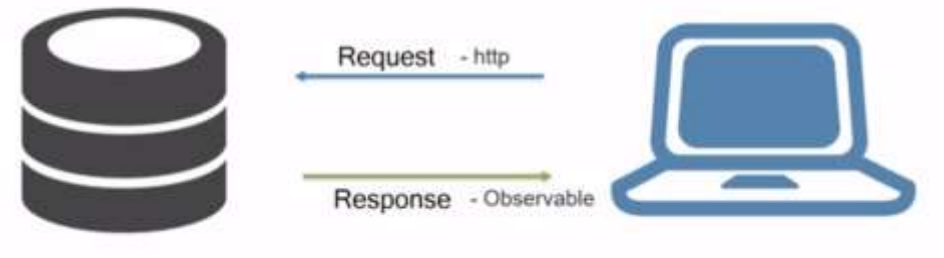
Lesson Objectives

- The HTTP providers
- Injecting the providers
- GET call
- Handling error
- About Observables
- POST request
- Working with headers

Instructor Notes:

HTTP

- Angular applications often obtain data using http
- Application issues http get requests to a web server which returns http response-Observable to the application.
- Application then processes that data



Instructor Notes:

Add instructor notes here.

Observables

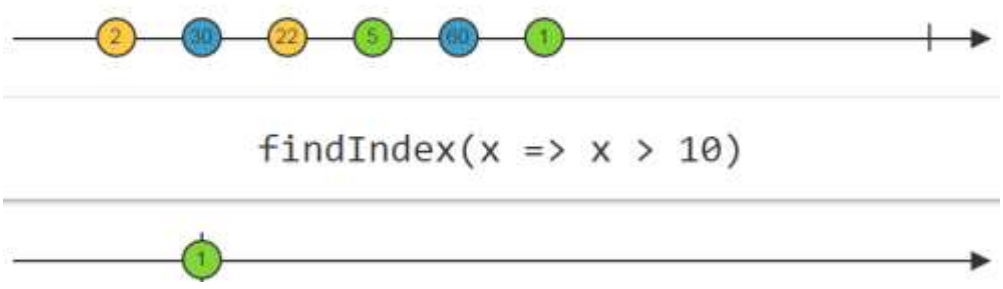
- Observables is like an array whose items arrived asynchronously.
- Observable help to manage asynchronous data, such as data coming from a backend service.
- Observables are proposed feature for ES 2016 the next version of JavaScript. To use observables now angular uses a third party library called reactive extensions.
- Observables are used with in angular itself including angular's event system and its http client service
- A method can be subscribed to an observable to receive asynchronous notifications as new data arrives.

Data sequences can take many forms such as a stream of data from a backend web service or a set of system notifications or a series of events such as user input.

Reactive extensions represent a data sequence as an observable sequence commonly just called an observable.

A method can be subscribed to an observable to receive asynchronous notifications as new data arrives. The method can then react with the arrived data. The method is notified when there is no more data or one an error occurs. Since an observable works like an array we can use the map operator.

We can visualizing observable sequences with interactive diagrams from http://rxmarbles.com/



Marble diagram

Instructor Notes:

Add instructor notes here.

Observables

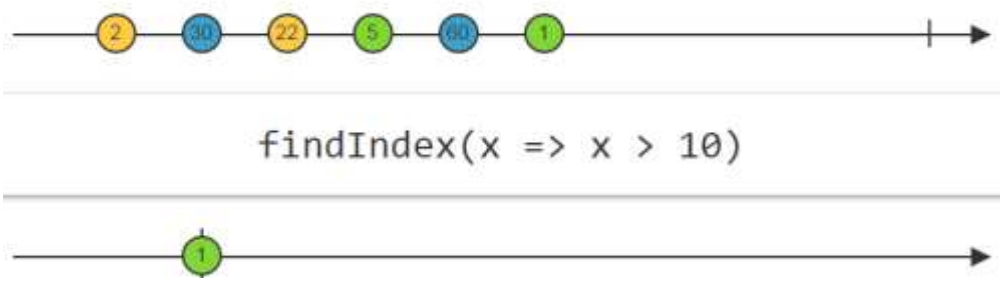
- Observables is a part of ReactiveX library also known as rxjs
  - import { Observable } from 'rxjs/Observable';
- Observables is like an array whose items arrived asynchronously. The role of ReactiveX to provide asynchronously programming
- Observable help to manage asynchronous data, such as data coming from a backend service. That data we are going to subscribe
- Observable work with multiple value
- Observable are cancellable
- Observable use javaScript function such as map filter & reduce

Data sequences can take many forms such as a stream of data from a backend web service or a set of system notifications or a series of events such as user input.

Reactive extensions represent a data sequence as an observable sequence commonly just called an observable.

A method can be subscribed to an observable to receive asynchronous notifications as new data arrives. The method can then react with the arrived data. The method is notified when there is no more data or one an error occurs. Since an observable works like an array we can use the map operator.

We can visualizing observable sequences with interactive diagrams from http://rxmarbles.com/



Marble diagram

Instructor Notes:

Add instructor notes here.

Introducing RxJs

- RxJs stands for Reactive Extensions for JavaScript, and its an implementation of Observables for JavaScript.
- It is a ReactiveX library for JavaScript.
- It provides an API for asynchronous programming with observable streams.
- ReactiveX is a combination of the best ideas from the Observer pattern, the Iterator pattern, and functional programming.
- 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.

The RxJS library is quite large.

It's up to us to add the operators we need

// Add map operator
https://cdnjs.cloudflare.com/ajax/libs/rxjs/4.1.0/rx.map

// Add all operators to Observable
https://cdnjs.cloudflare.com/ajax/libs/rxjs/4.1.0/rx.all.js

// Add map operator
import 'rxjs/add/operator/map'*;*

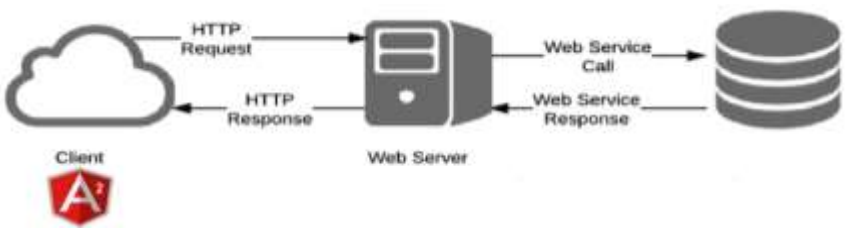
// Add all operators to Observable
import 'rxjs/Rx';

Instructor Notes:

Add instructor notes here.

Angular2 HTTP

- Angular applications often obtain data using http
- Application issues http get requests to a web server which returns http response to the application.
- Application then processes that data



Instructor Notes:

Add instructor notes here.

Http Class

- Performs http requests using `XMLHttpRequest` as the default backend.
- Http is available as an injectable class.
- Calling request returns an Observable which will emit a single Response when a response is received.
- To work with Http Class
  - Include Angular 2 Http script (http.dev.js) in index.html
  - Include script tag for the reactive extensions (Rx.js) in index.html
  - Register HTTP_PROVIDERS
  - Import RxJS

Http is not a part of angular2/core and it is not included in the main angular script file. It is an optional service available in its own library. It needs to be added in the index file.

Angular http library has several services that assist with using http. To use http from any of the components we can register this angular service with root component.

There are several services involved the angular http client library provides a single constant that defines the set of service providers from the angular http library called **HTTP_Providers**

Similarly there are several features for reactive extensions such as the map operator. To load them all in root component using import statement we need to use **import 'rxjs/Rx'**;. It tell to the module loader to load this library but imports nothing. When the library is loaded its javascript is executed and for this particular library executing the JavaScript loads the full set of observable operators that will need throughout application

Http methods
request(url, options)
get (url, options)
delete (url, options)
head (url, options)
post (url,body,options)
put (url,body,options)
patch(url,body,options)

Instructor Notes:

Add instructor notes here.

Catch Operator

- Reacts to the error case of an Observable.
- Need to return a new Observable to continue with

```
export class UserProxy{
    constructor(private http :Http){}
    load(){
        return this.http
            .get('http://api.randomuser.me/10')
            .map(res =>res.json())
            .catch(this.logAndPassOn);
}
private logAndPassOn (error: Error) {
    console.error(error);
    return Observable.throw(error);
}
```

Instructor Notes:

Add instructor notes here.

Communication with JSONP

- Angular provides us with a JSONP services which has the same API surface as the Http.
- Only difference that it restricts us to use GET requests only.
- JSONP service requires the JSONP_PROVIDERS.

Instructor Notes:

Add instructor notes here.

Server Simulation

- To enable our server simulation, we replace the XHRBackend service with the in-memory web api backend.
- The in-memory api must to implements ConnectionBackend

```
bootstrap(App,[ HTTP_PROVIDERS,
  // in-memory web api providers
  provide( XHRBackend, { useClass: InMemoryBackend
} )
]);
```

Instructor Notes:

Add instructor notes here.

Demo

- HttpDemo

Add the notes here.

Instructor Notes:

Add instructor notes here.

Lab

- Lab 4.1

Add the notes here.

Instructor Notes:

Add instructor notes here.

Summary

- RxJS is a third party library, endorsed by Angular, that implements the asynchronous observable pattern.
- Observables are used with in angular itself including angular's event system and its http client service.
- The Angular HTTP library simplifies application programming with the XHR and JSONP APIs.
- The Angular Http client communicates with the server using a familiar HTTP request/response protocol.