Injeçāo de dependencia/ Seçāo 5 aula 38

Injeçāo de dependência é um padrão de peojeto onde a aplicação deix de distanciar seus objetos manualmente e passa a depender do framework para obter os objetos que ela quer usar, o framework que gerencia a instanciação dos objetos bem como suas dependências e disponibiliza isso para os componentes da aplicação.

O que é um serviço ? / aula 39

Serviços são classes comuns em angular que você pode usar para injetar em componente e em outros serviços. Serviços são geralmente usados numa aplicação para encapsular o acesso a API do back-end. Serviços podem sem singleton, isso que dizer que sāo ótimos candidatos para guardar dados compartilhados para toda aplicação , se você quiser também serviços podem gravar dados apenas para uma parte de aplicação. Por exemplo quando declarado no mȯdulo central (app.module.ts) dessa forma instanciando a classe apenas uma vez, vc consegue manipulala em toda a aplicação.

Reactive Programming e HTTP/ 41.

A idéia de programação reativa é simples: um evento acontece e os que estão interessados são notificados e reagem a ele É baseado em um padrão bem comum da web, o observer, nesse padrão você vai ter o objeto que e o foco do interesse (pattern) e você vai ter outros objetos interessados em alguma mudança desse primero objeto que sāo os listeners

Reactive programming é a junção de dois padrões: iterator + observer. Iterartor pq vai item a item na stream mais observer pq notifica os listeners interessados.

A biblioteca utilizada pelo angular para reactive programming é a RXJS.

O observable possui uma serie de funções ou operadores para inscrição e manipulação de eventos, mas existe uma similaridade com funções de outro tipo bem conhecido: O array.

Comparaçāo:

([1,2,3,4]). // STREAM

.map( x => x + 3

.filter( x => x % 2 == 0 )

.forEach( x => console.log(x) )//4, 6, 8.

Observable.from([1,2,3,4]). // STREAM

.map( x => x + 3 ). // TRANSFORMAÇĀO

.filter( x => x % 2 == 0 )

.subscribe( x => console.log(x) )//4, 6, 8. –LISTENER – BASTANTE PARECIDO COM O FOREACH.

Por que nāo promise’s ?

Uma das grandes diferenças entre observable e promises, é que observables continuam disparando eventos até que sejam explicitamente fechados, já promises sāo consideradas resolvidas depois do primeiro evento. Isso da um pouco mais de flexibilidade com o observable como por exemplo a capacidade de usar web sockets.

Refazer chamadas HTTP com observable: retry(2);