**Activated route**

**ActivatedRoute** interface provides access to information about a route associated with a component that is loaded in an outlet. Use to traverse the RouterState tree and extract information from nodes

**Observables vs promise** – Both are used for asynchronous task only difference is that when a when a function returns promise it is either resolved or rejected only once, but when a function returns observable it gets subscribed and subscription is always called whenever there is change in data

<https://jasonwatmore.com/post/2018/11/22/angular-7-role-based-authorization-tutorial-with-example>

<https://stackblitz.com/edit/angular-7-role-based-authorization-example>

**Angular cli** – It is command line interface provided by angular, it has inbuild features for building project, test cases, run projects, run e2e testing etc

**Incremental DOM** - Angular Ivy is a new Angular renderer which is radically different from anything we have seen in mainstream frameworks, because it uses incremental DOM.

Incremental DOM is different from virtual DOM

Lets see how they update DOM when there change

In case **Incremental DOM** has one virtual DOM and walks along the tree to find changes then mutates the virtual DOM and then apply those changes to the actual DOM - (reduced memory size and garbage collection).

**RXJS operator usage**

Different types of rxjs operators

RxJS offers a number of functions that can be used to create new observables. These functions can simplify the process of creating observables from things such as events, timers, promises, and so on.

import { from } from 'rxjs'; // Create an Observable out of a promise

const data = from(fetch('/api/endpoint'));

import { interval } from 'rxjs'; // Create an Observable that will publish a value on an interval const secondsCounter = interval(1000);

## Operators

Operators are functions that build on the observables foundation to enable sophisticated manipulation of collections. For example, RxJS defines operators such as map(), filter(), concat(), and flatMap().

Normally with array we use js functions like map(), filter(), concat() etc.

Similarly if we get data as observables we use rxjs operators.

**Authorization service** – We use Authorization services if we want to add authorizations on page route for example we have web page were two type of users admin and normal user in case of normal users we want to restrict some of the routes we can do that using Authorization service

**Incremental VS virtual DOM**

Before virtual DOM to if we used to make any change in web page in order to update we can iterate complete page and find out

AOT and Ivy

Ivy is the code name for Angular's [next-generation compilation and rendering pipeline](https://blog.angular.io/a-plan-for-version-8-0-and-ivy-b3318dfc19f7). With the version 9 release of Angular, the new compiler and runtime instructions are used by default instead of the older compiler and runtime, known as View Engine.

Service workers: To cache backend response

AOT – Ahead of time compilation

#### **Question: Could you explain services in Angular?**

**Answer:** Singleton objects in Angular that get instantiated only once during the lifetime of an application are called services. An Angular service contains methods that maintain the data throughout the life of an application.

**Angular change detection**

<https://blog.angular-university.io/how-does-angular-2-change-detection-really-work/>

Angular uses Zones internally to trigger change detection,  based on that change it will rerender the page

many browser APIs are patched by Zone.js to transparently trigger Angular change detection,

One limitation of this mechanism is that if by some reason an asynchronous browser API is not supported by Zone.js, then change detection will not be triggered. This is, for example, the case of IndexedDB callbacks.

Each Angular component has an associated change detector, which is created at application startup time.

#### Q. What is Karma? What is the use in Angular?

Karma is a tool for executing source code against test code inside a browser environment. It supports the running of tests in each browser it’s configured for. Results are displayed on both the command line and on the browser for the developer to inspect which tests have passed or failed. Karma also watches the files and can trigger a test rerun whenever a file changes. At the root of the Angular project, we have the file karma.conf that’s used to configure Karma.

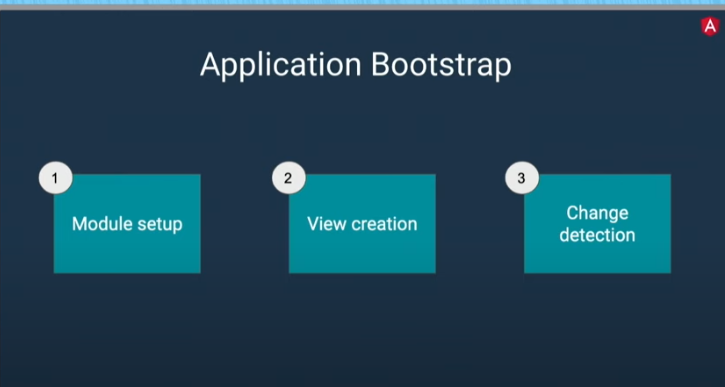
#### Q. What is Jasmine? What is the Use in Angular?

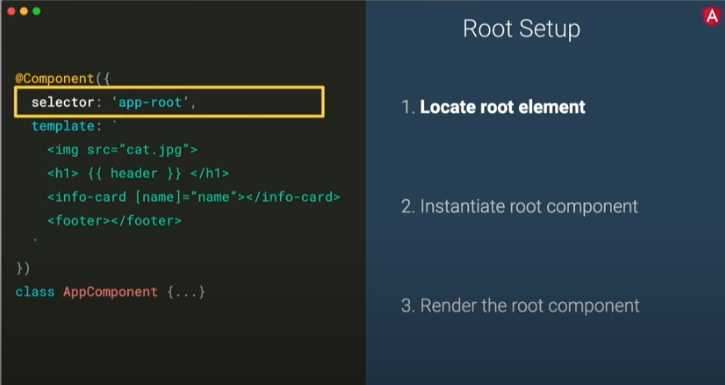
Jasmine is a javascript testing framework that supports a software development practice called Behaviour Driven Development, or BDD for short. It’s a specific flavour of Test Driven Development (TDD).

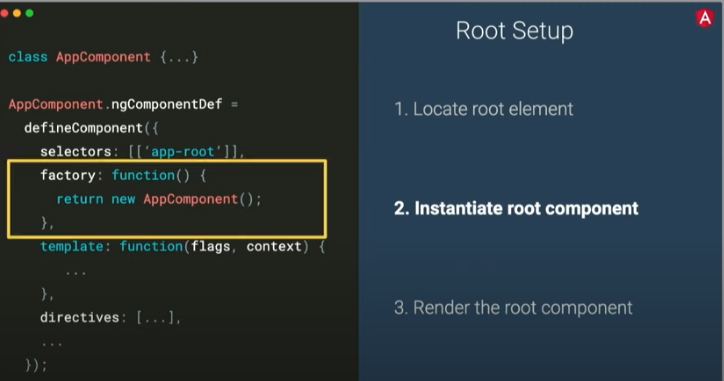
#### Q. What is Protractor?

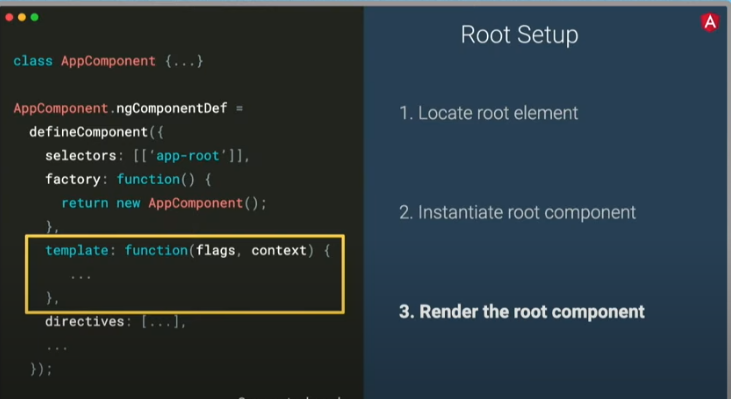
Protractor is an end-to-end test framework for Angular. It runs your tests inside a real browser, interacting with it as real person would. Unlike unit tests, where we test individual functions, here we test the entire logic. Protractor is able to fill in forms, click buttons and confirm that the expected data and styling is displayed in the HTML document. Just like Karma, Protractor has its own configuration file at the root of your Angular project, protractor.conf:

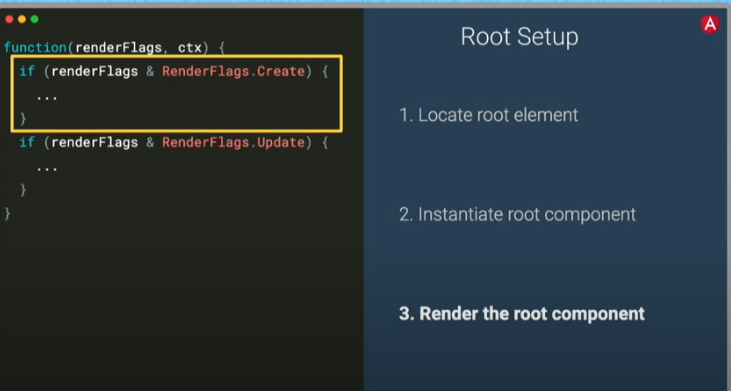
**How angular works**

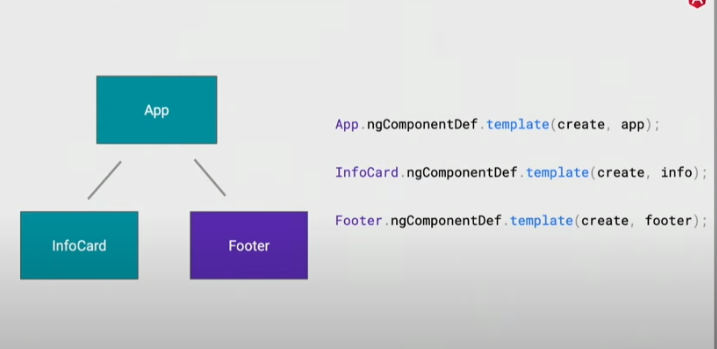




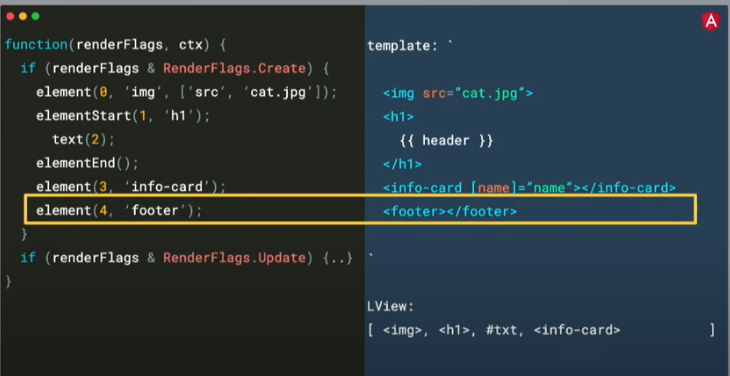




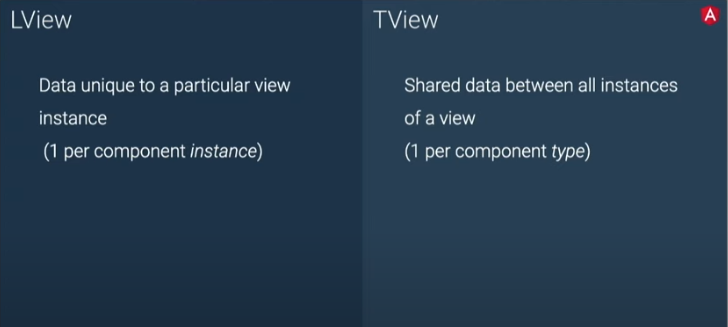


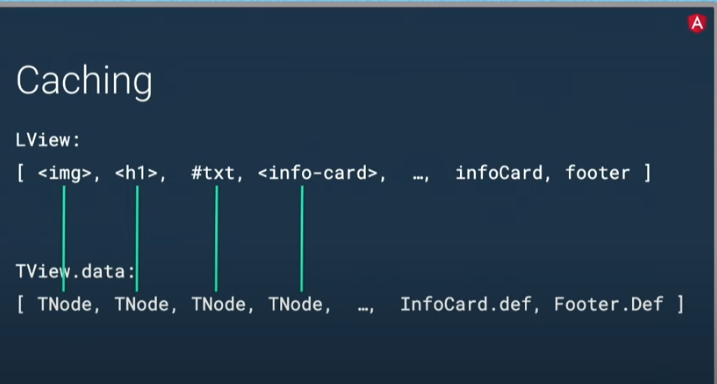


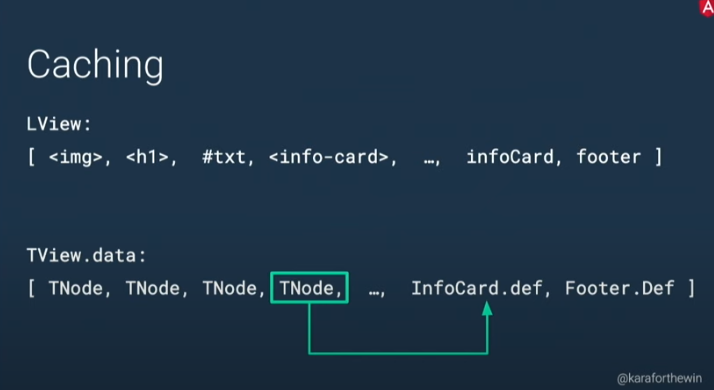
Every component they push into Lview(logical view) o that during change detection we don’t have to use dom apis

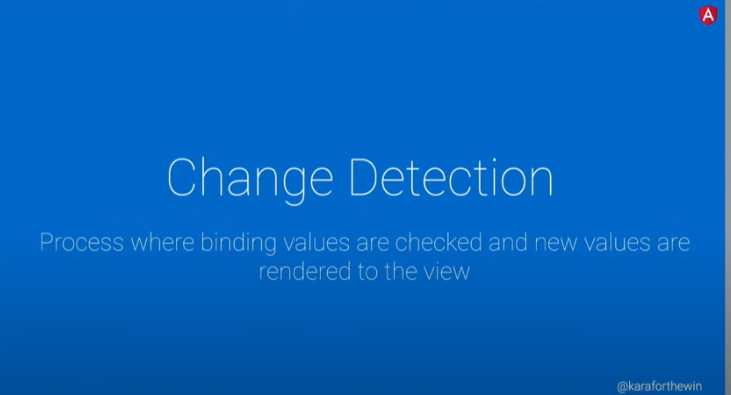


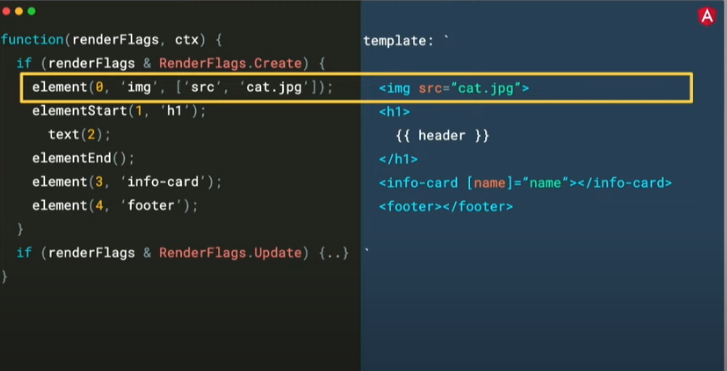
During compile time they maintain an array for what all directive being used and than instantiate thore directive for respective elements



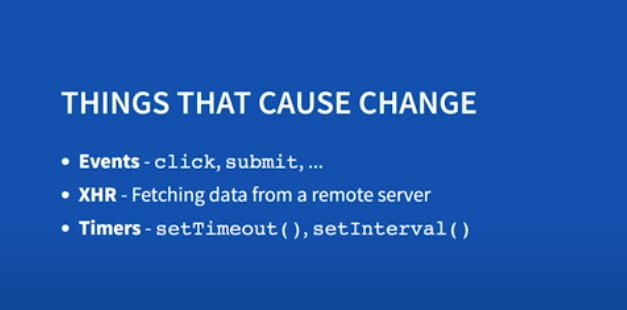








ChangeDetectionStragtegy: onpush - in this case angular is not going to run change detection when there is change in any of the property on input object it will trigger only when input value itself changes.



Who notifies angular about change has happened : zones

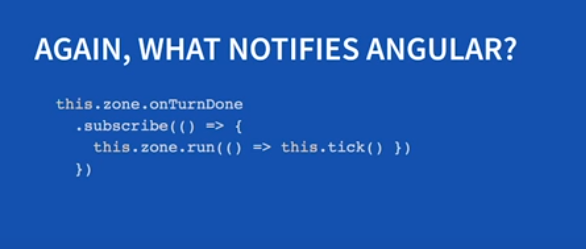
Zone.js is a library which angular uses for change detection

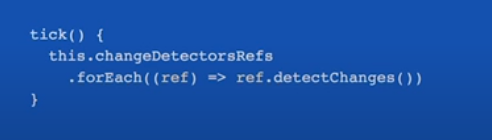
Zones can perform an operation each time code enter or exists task

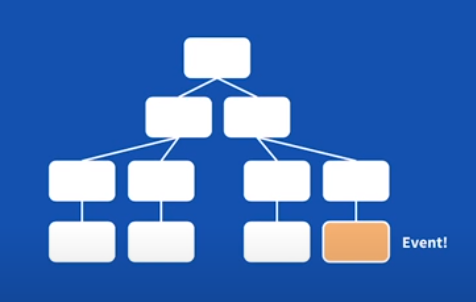
Again what notifies angular – angular internally uses **this.zone.onTurnDone,** angular subscribes it which is triggered everytime a task is completed

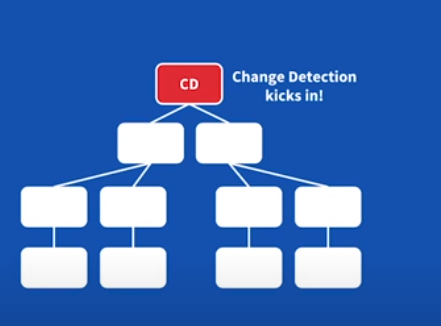
Once onTurnDone fires angular triggers this.tick and tick function will run changedetectors for all component

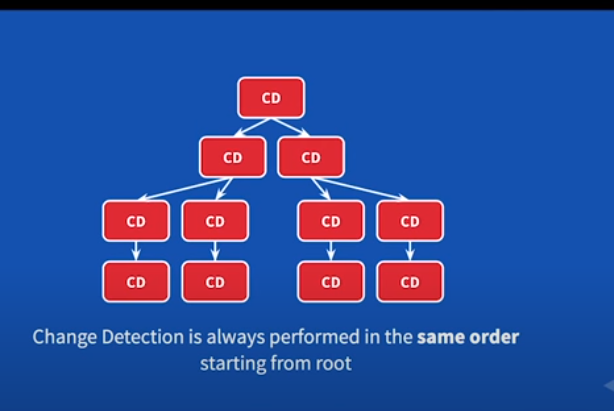
Note : There is one mechanism that triggers change detection but every component has its own change Detecter











How fast is change detection in angular – Its very fast as angular maintains monomorphic code.

Angular creates change detector classes at runtime for each component, which are monomorphic, because they know exactly what the shape of the component’s model is. VMs can perfectly optimize this code, which makes it very fast to execute.

Monomorphism - I**n short: functions that are invoked with the same types over and over again are optimized in the JIT compiler, hence faster.**

**Smarter change detection**

By default change detection is run for each and every class, but we can skip that by using immutable objects,

Mutable – That can be changed

Immutable which cannot be changed.

We can tell angular that for particular components perform change detection only if input to this component chages.

But we need to make sure that for Immutable object we always assign new reference if we want to change any of its property

**changeDetection: changeDetectionStrategyOnPush**

**There points to keep in mind for angular performance**

1. Use lazy loading and feature modules for
2. Unsubscribe observables when ever we subscribe
3. If possible to skip skip change detection using changeDetectionStrategyOnPush, do it always for presentation components, keep presentational component always clean of unnecessary things
4. Don’t keep reference of dom element this will lead to memory as we have if we are keeping it don’t forget to remove reference on destroy

**Angular vs react**

**Angular is ts based react is still on js**

Angular is framework angular can be used independently for development of complete application react is library it requires other libraries it not complete package in itself.

Learning curve if you are building complete react learning curve is very less it is just javascript and what ever feature you need use it.

But angular is complete framework in itself we need to understand it completely to work

Community support if you will production bug on react it is less compared to angular

Performance wise: React go much popularity as angularjs was poor performance and react introduced virtual DOM, with latest angular when angular ivy has come and incremental dom it has similar performance

**Virtual dom –** react doesnot operate on real dom to traverse despite full in-memory representation of the DOM it to traverse so that it can quickly identify which part of DOM to be updated and update it quickly

**Incremental DOM –** Does not keep in memory representation o fcomplete DOM so saves lot of memory.

Battle between incremental dom and virtual dom is about memory and speed .

Angular doesnot loose battle of speed but wins battle of memory for sure .

Which is very important for mobile devices and these days PWA is getting popularity over app. It is big win

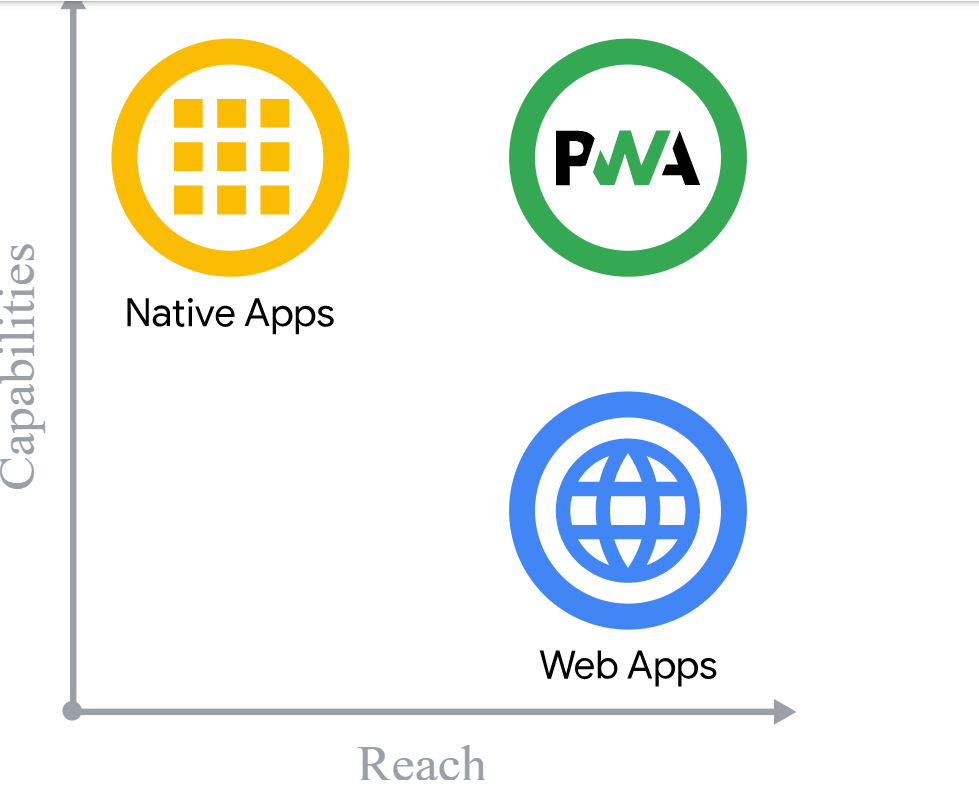
**PWA(progressive web app)**

We have web app and and native they have their own benefits

Web app has more capabilities and native app is more reachable PWA gets benefits of both

PWA are webs apps enchanced to get feature like native app

Progressive Web Apps are just web applications. Using progressive enhancement, new capabilities are enabled in modern browsers. Using service workers and a web app manifest, your web application becomes reliable and installable



**Service worker and web worker**

Ek to javascript engine hai aur ek browser, which also has capabilities to run code in background. Both run independent of javascript thread.

A service worker is a script that your browser runs in the background, separate from a web page, opening the door to features that don't need a web page or user interaction.

Today, they already include features like [push notifications](https://developers.google.com/web/updates/2015/03/push-notifications-on-the-open-web) and [background sync](https://developers.google.com/web/updates/2015/12/background-sync).

[ServiceWorkers](https://developer.mozilla.org/en-US/docs/Web/API/ServiceWorker_API) essentially act as proxy servers that sit between web applications, the browser, and the network (when available).

**Web Workers** makes it possible to run a script operation in a background thread separate from the main execution thread of a web application. The advantage of this is that laborious processing can be performed in a separate thread, allowing the main (usually the UI) thread to run without being blocked/slowed down.

|  |
| --- |
|  |
|  | Import { DynamicComponent } from ‘’ |
|  | @Component({ |
|  | selector: 'app-child', |
|  | template: ' <h4> I am child </h4> ', |
|  | styleUrls: ['./child.component.css'] |
|  | }) |
|  | export class ChildComponent implements OnInit { |
|  |  |
|  | constructor(private vf:ViewContainerRef,private componentFactoryResolver:ComponentFactoryResolver) { } |
|  |  |
|  | ngOnInit() { |
|  | //This pieces of code adds dynamic component ( Just trust me for now ) |
|  | let resolver = this.componentFactoryResolver.resolveComponentFactory(DynamicComponent); |
|  | let componentFactory = this.vf.createComponent(resolver); |
|  |  |
|  | } |
|  |  |
|  | } |