global function in Ionic 2 / Angular 2

Asked 2 years, 3 months ago Active 6 months ago Viewed 10k times

9

4

How can I setup a global function that can be accessed throughout all views?

In app.component.ts I added a simple method

```
openShare() {console.log("button clicked")}
```

and then in my nav I have

```
<button ion-button right (click)="openShare()">
<ion-icon name="md-share"></ion-icon>
</button>
```

When I try to access this from any page I get the following.

```
self.context.openShare is not a function
```

It does however execute fine if I put it directly in the constructor (e.g this.openShare();), but when calling from any page using a (click) function it just doesn't work.

Is app.component.ts not global? I thought this is where I would place it, but maybe I am missing something.

Basically its a function for a simple button on the nav, I need it to be global though since its used on every page.

Any help would be appreciated, still figuring Ionic 2 out.

angular typescript ionic2 ionic3

edited Jul 1 '18 at 8:39
sebaferreras
35.6k 8 87 106

asked Apr 2 '17 at 3:19
limit
367 5 19

Try a custom directive. – Harish Apr 2 '17 at 10:00

Any example code how I would do this? – limit Apr 2 '17 at 10:26

angular.io/docs/ts/latest/guide/attribute-directives.html – Harish Apr 2 '17 at 10:29

@HarishKommuri - This might also work. If anyone has a simple example would be appreciated. I am more visual, not asking to write the code just a simple example on a click method. – limit Apr 2 '17 at 10:36

3 Answers

11

You can use `Directive` .

Try as follows.

Put the following code in separate directive file. (social-sharing.directive.ts);

```
import { Directive, HostListener } from '@angular/core';

@Directive({
  selector: '[socialSharing]'
})
export class SocialSharing {

  constructor() { }

  @HostListener('click') onClick() {
    // Your click functionality
  }
}
```

Import it into `app.module.ts` and than add to `declarations` .

In HTML, just add `attribute` to any element which is the `selector` in your directive file.

```
<button ion-button right socialSharing>
 <ion-icon name="md-share"></ion-icon>
</button>
```

Additional Info:

Passing values from component to directive.

home.html

```
<button ion-button right [socialSharing]="property">
  <ion-icon name="md-share"></ion-icon>
</button>
```

home.component.ts

```
export class HomePage {

  property: string = 'some url';
   constructor() {}
}
```

social-sharing.directive.ts

Import `Input` along with others from `@angular/core` .

```
import { Directive, Input, HostListener } from '@angular/core';

@Directive({
   selector: '[socialSharing]'
})

export class SocialSharing {
  @Input('socialSharing') str: string;

  constructor() {}

  @HostListener('click') onClick() {
     console.log(this.str);
  }
};

  ngAfterInit() {
     console.log(this.str);
  };
}
```

Using Element in Directive:

social-sharing.directive.ts

Import `ElementRef` along with others from `@angular/core`

```
import { Directive, ElementRef, HostListener } from '@angular/core';

@Directive({
  selector: '[socialSharing]'
})

export class SocialSharing {

  // Add ElementRef to constructor

  constructor(el: ElementRef) {};

  ngOnInit() {
     let elRef = this.el.nativeElement;
     console.log(elRef.innerHTML);
  };
}
```

edited Jan 10 at 9:11

answered Apr 2 '17 at 10:39

Harish

2,558 1 15 21

- Hey thank you. This actually worked perfect. A lot of posts for people wanting to know how to do this. Hopefully they find your answer here. Thank you, much appreciated. – limit Apr 2 '17 at 19:16

8

You can easily do that using `Providers` .When you need to use that functionality (or provider), you just need to `inject` it into your related component.That is it.

Method 1:

You can create a Provider using CLI.

```
> ionic g provider YourProvider
```

your-provider.ts

```
import { Injectable } from '@angular/core';

   @Injectable()
   export class YourProvider {

     constructor() {
     }

      openShare() {console.log("button clicked")}
   }
```

app.module.ts

```
import { YourProvider } from "../pages/path";

   @NgModule({
     declarations: [
       MyApp,
     ],
     imports: [
       IonicModule.forRoot(MyApp),
      ],
     bootstrap: [IonicApp],
     entryComponents: [
       MyApp,
       ],
     providers: [{ provide: ErrorHandler, useClass: IonicErrorHandler },YourProvider  ]
   })

   export class AppModule { }
```

your-view.ts

```
    import { YourProvider } from '../../providers/your-provider';

     export class YourViewPage{

       constructor(public yourProvider : YourProvider) {

       }

     openShare(){
        this.yourProvider.openShare();
     }
```

Method 2: Creat an `abstract` base class.

my-base-class.ts

```
export abstract class MyBaseClass {

  constructor() {
  }

  protected openShare():void {
    console.log("button clicked");
  }

}
```

my-view.ts

```
export class MyViewPage extends MyBaseClass {
  constructor()
  {
     super();
  }

       openShare(){
              super.openShare();
            }
  }
```

edited Apr 2 '17 at 4:28

answered Apr 2 '17 at 3:27

Sampath
34.6k 21 156 259

hmm.. yeah this is kind of how I set things up now, but for a simple function seems like too many imports over 20 pages. Is this the only way? This is for a button in the nav header so basically called on every page. – limit Apr 2 '17 at 4:16

please see the update on my post. – Sampath Apr 2 '17 at 4:28

Thanks for the help. Interesting with the abstract class, will give this a try. – limit Apr 2 '17 at 10:32

sure.hope you'll share the result with us :) – Sampath Apr 2 '17 at 10:38

3

Just like @Sampath mentioned, one way would be to use a custom provider. But since you method is just a simple one, I think you can use Events instead. It'd be like this:

In your `app.component.ts` file, subscribe to the new event:

```
import { Events } from 'ionic-angular';

constructor(public events: Events, ...) {

  // ...

  events.subscribe('social:share', () => {

    // your code...
    console.log("button clicked");

  });

}
```

And then in any other page, just publish the event to execute that logic:

```
function anotherPageMethod() {
  this.events.publish('social:share');
}
```

answered Apr 2 '17 at 4:28

sebaferreras
35.6k 8 87 106

2 Thanks, this might work. Still though - I can't believe something so global as a nav bar, there is no simple way to declare common methods. So much for keeping it DRY. Was simple in Ionic 1. Oh well - Really appreciate the help. Thanks again. – limit Apr 2 '17 at 10:31