

emonney / ngx-toasta

Join GitHub today

Dismiss

GitHub is home to over 28 million developers working together to host and review code, manage projects, and build software together.

Sign up

Toast notification for AngularX (Angular2 and beyond) <https://emonney.github.io/ngx-toasta>

#toast #notifications #alerts #growl #angularx #angular2

8 commits

1 branch

0 releases

1 contributor

MIT

Branch: master

New pull request

Find file

Clone or download



emonney Updated readme with ghpages link

Latest commit f4fc077 on 28 Jun 2018

docs	github pages publish	7 months ago
e2e	initial commit	7 months ago
projects/ngx-toasta	Updated readme and some aot fine tuning	7 months ago
src	Updated readme and some aot fine tuning	7 months ago
.editorconfig	initial commit	7 months ago
.gitignore	Updated .gitignore	7 months ago
LICENSE	Create LICENSE	7 months ago
README.md	Updated readme with ghpages link	7 months ago
angular.json	Finished porting/convertng to angular6 & angular6 library	7 months ago
package.json	Finished porting/convertng to angular6 & angular6 library	7 months ago
tsconfig.json	Scaffolded angular library "ngx-toasta"	7 months ago
tslint.json	initial commit	7 months ago

README.md

ngx-toasta

npm package

0.1.0

downloads

6k/m

An angularX toast component that shows growl-style alerts and messages for your application. This is a port of [ng2-toasty](#) in order to get it to work with the latest versions of [angular](#) and [rxjs](#) with some enhancements.

Follow me [Follow](#) 394 to be notified about new releases.

Installation

```
npm install ngx-toasta
```

Demo

Online demo available [here](#)

Usage

If you use SystemJS to load your files, you might have to update your config:

```
System.config({
  map: {
    'ngx-toasta': 'node_modules/ngx-toasta/bundles/index.umd.js'
  }
});
```

1. Update the markup

- Import style into your web page. Choose one of the following files;
 - style-default.css - Contains DEFAULT theme
 - style-bootstrap.css - Contains Bootstrap 3 theme
 - style-material.css - Contains Material Design theme
- Assign the selected theme name [default , bootstrap , material] to the theme property of the instance of ToastaConfig.
- Add <ngx-toasta></ngx-toasta> tag in template of your application component.

2. Import the ToastaModule

Import ToastaModule.forRoot() in the NgModule of your application. The forRoot method is a convention for modules that provide a singleton service.

```
import {BrowserModule} from '@angular/platform-browser';
import {NgModule} from '@angular/core';
import {ToastaModule} from 'ngx-toasta';

@NgModule({
  imports: [
    BrowserModule,
    ToastaModule.forRoot()
  ],
  bootstrap: [AppComponent]
})
export class AppModule {
}
```

If you have multiple NgModules and you use one as a shared NgModule (that you import in all of your other NgModules), don't forget that you can use it to export the ToastaModule that you imported in order to avoid having to import it multiple times.

```
@NgModule({
  imports: [
    BrowserModule,
    ToastaModule.forRoot()
  ],
  exports: [BrowserModule, ToastaModule],
})
export class SharedModule {
}
```

3. Use the ToastaService for your application

- Import ToastaService from ngx-toasta in your application code:

```
import {Component} from '@angular/core';
import {ToastaService, ToastaConfig, ToastOptions, ToastData} from 'ngx-toasta';
```

```

@Component({
  selector: 'app',
  template: `
    <div>Hello world</div>
    <button (click)="addToast()">Add Toast</button>
    <ngx-toasta></ngx-toasta>
  `
})
export class AppComponent {

  constructor(private toastService: ToastService, private toastConfig: ToastConfig) {
    // Assign the selected theme name to the `theme` property of the instance of ToastConfig.
    // Possible values: default, bootstrap, material
    this.toastConfig.theme = 'material';
  }

  addToast() {
    // Just add default Toast with title only
    this.toastService.default('Hi there');
    // Or create the instance of ToastOptions
    var toastOptions: ToastOptions = {
      title: 'My title',
      msg: 'The message',
      showClose: true,
      timeout: 5000,
      theme: 'default',
      onAdd: (toast: ToastData) => {
        console.log('Toast ' + toast.id + ' has been added!');
      },
      onRemove: function(toast: ToastData) {
        console.log('Toast ' + toast.id + ' has been removed!');
      }
    };
    // Add see all possible types in one shot
    this.toastService.info(toastOptions);
    this.toastService.success(toastOptions);
    this.toastService.wait(toastOptions);
    this.toastService.error(toastOptions);
    this.toastService.warning(toastOptions);
  }
}

```

4. How to dynamically update title and message of a toast

Here is an example of how to dynamically update message and title of individual toast:

```

import {Component} from '@angular/core';
import {ToastService, ToastConfig, ToastComponent, ToastOptions, ToastData} from 'ngx-toasta';
import {Subject, Observable, Subscription} from 'rxjs/Rx';

@Component({
  selector: 'app',
  template: `
    <div>Hello world</div>
    <button (click)="addToast()">Add Toast</button>
    <ngx-toasta></ngx-toasta>
  `
})
export class AppComponent {

  getTitle(num: number): string {
    return 'Countdown: ' + num;
  }

  getMessage(num: number): string {

```

```

        return 'Seconds left: ' + num;
    }

    constructor(private toastaService:ToastaService) { }

    addToast() {
        let interval = 1000;
        let timeout = 5000;
        let seconds = timeout / 1000;
        let subscription: Subscription;

        let toastOptions: ToastOptions = {
            title: this.getTitle(seconds),
            msg: this.getMessage(seconds),
            showClose: true,
            timeout: timeout,
            onAdd: (toast: ToastData) => {
                console.log('Toast ' + toast.id + ' has been added!');
                // Run the timer with 1 second interval
                let observable = Observable.interval(interval).take(seconds);
                // Start listen seconds beat
                subscription = observable.subscribe((count: number) => {
                    // Update title of toast
                    toast.title = this.getTitle(seconds - count - 1);
                    // Update message of toast
                    toast.msg = this.getMessage(seconds - count - 1);
                });
            },
            onRemove: function(toast: ToastData) {
                console.log('Toast ' + toast.id + ' has been removed!');
                // Stop listenning
                subscription.unsubscribe();
            }
        };

        switch (this.options.type) {
            case 'default': this.toastaService.default(toastOptions); break;
            case 'info': this.toastaService.info(toastOptions); break;
            case 'success': this.toastaService.success(toastOptions); break;
            case 'wait': this.toastaService.wait(toastOptions); break;
            case 'error': this.toastaService.error(toastOptions); break;
            case 'warning': this.toastaService.warning(toastOptions); break;
        }
    }
}

```

5. How to close specific toast

Here is an example of how to close an individual toast:

```

import {Component} from '@angular/core';
import {ToastaService, ToastaConfig, ToastaComponent, ToastOptions, ToastData} from 'ngx-toasta';
import {Subject, Observable, Subscription} from 'rxjs/Rx';

@Component({
    selector: 'app',
    template: `
        <div>Hello world</div>
        <button (click)="addToast()">Add Toast</button>
        <ngx-toasta></ngx-toasta>
    `
})
export class AppComponent {

    getTitle(num: number): string {

```

```

        return 'Countdown: ' + num;
    }

    getMessage(num: number): string {
        return 'Seconds left: ' + num;
    }

    constructor(private toastService: ToastService) { }

    addToast() {
        let interval = 1000;
        let subscription: Subscription;

        let toastOptions: ToastOptions = {
            title: this.getTitle(0),
            msg: this.getMessage(0),
            showClose: true,
            onAdd: (toast: ToastData) => {
                console.log('Toast ' + toast.id + ' has been added!');
                // Run the timer with 1 second interval
                let observable = Observable.interval(interval);
                // Start listen seconds beat
                subscription = observable.subscribe((count: number) => {
                    // Update title of toast
                    toast.title = this.getTitle(count);
                    // Update message of toast
                    toast.msg = this.getMessage(count);
                    // Extra condition to hide Toast after 10 sec
                    if (count > 10) {
                        // We use toast id to identify and hide it
                        this.toastService.clear(toast.id);
                    }
                });
            },
            onRemove: function(toast: ToastData) {
                console.log('Toast ' + toast.id + ' has been removed!');
                // Stop listening
                subscription.unsubscribe();
            }
        };

        switch (this.options.type) {
            case 'default': this.toastService.default(toastOptions); break;
            case 'info': this.toastService.info(toastOptions); break;
            case 'success': this.toastService.success(toastOptions); break;
            case 'wait': this.toastService.wait(toastOptions); break;
            case 'error': this.toastService.error(toastOptions); break;
            case 'warning': this.toastService.warning(toastOptions); break;
        }
    }
}

```

6. Customize the ngx-toasta for your application in template

You can use the following properties to customize the ngx-toasta component in your template:

- **position** - The window position where the toast pops up. Default value is `bottom-right`. Possible values: `bottom-right`, `bottom-left`, `bottom-fullwidth`, `top-right`, `top-left`, `top-fullwidth`, `top-center`, `bottom-center`, `center-center`
Example:

```
<ngx-toasta [position]="top-center"></ngx-toasta>
```

7. Options

Use these options to configure individual or global toasts

Options specific to an individual toast:

```
ToastOptions
{
  "title": string,      //A string or html for the title
  "msg": string,        //A string or html for the message
  "showClose": true,    //Whether to show a close button
  "theme": "default",   //The theme to apply to this toast
  "timeout": 5000,      //Time to live until toast is removed. 0 is unlimited
  "onAdd": Function,    //Function that gets called after this toast is added
  "onRemove": Function  //Function that gets called after this toast is removed
}
```

Configurations that affects all toasts:

```
ToastaConfig
{
  "limit": 5,           //Maximum toasts that can be shown at once. Older toasts will be removed. 0 is unlimit
  "showClose": true,    //Whether to show the 'x' icon to close the toast
  "position": "bottom-right", //The window position where the toast pops up
  "timeout": 5000,      //Time to live in milliseconds. 0 is unlimited
  "theme": "default"    //What theme to use
}
```

Credits

Original work by [ng2-toasty](#)

License

[MIT](#)