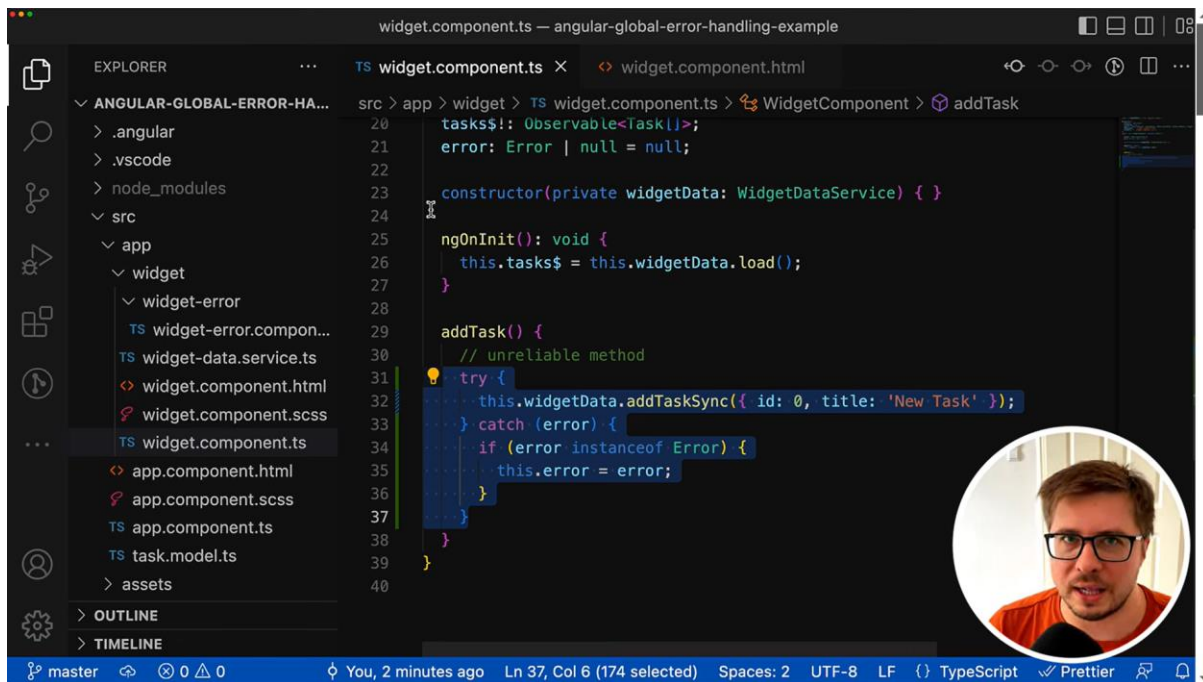
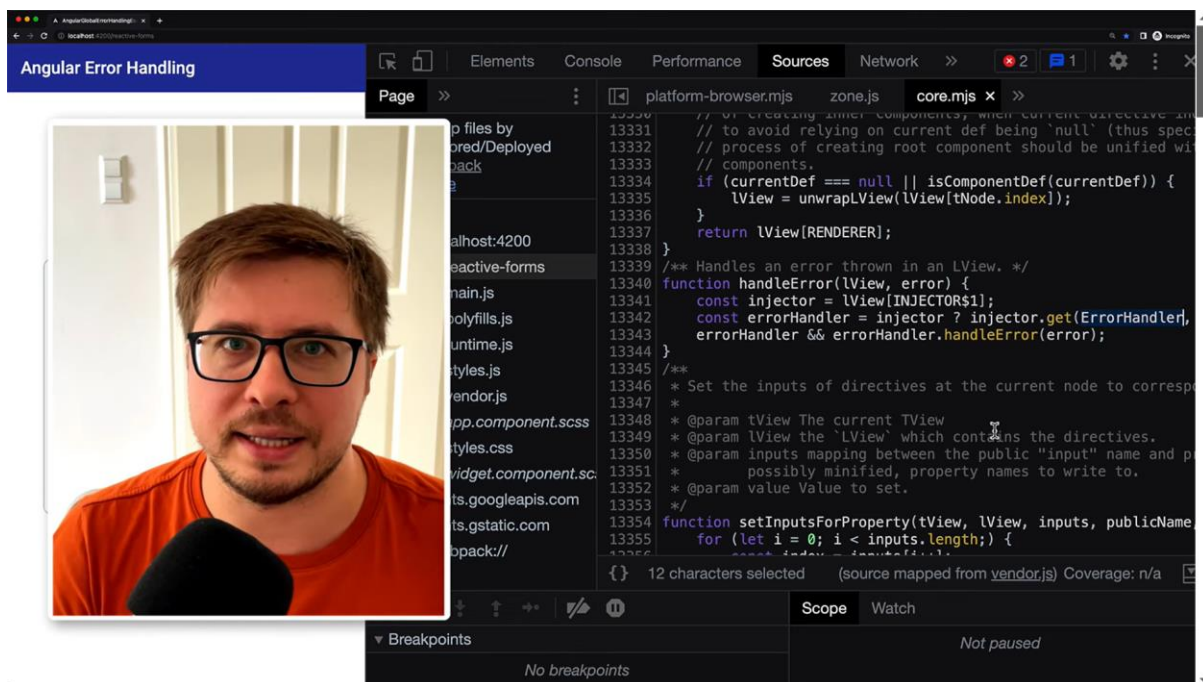


1. try-catch blocks



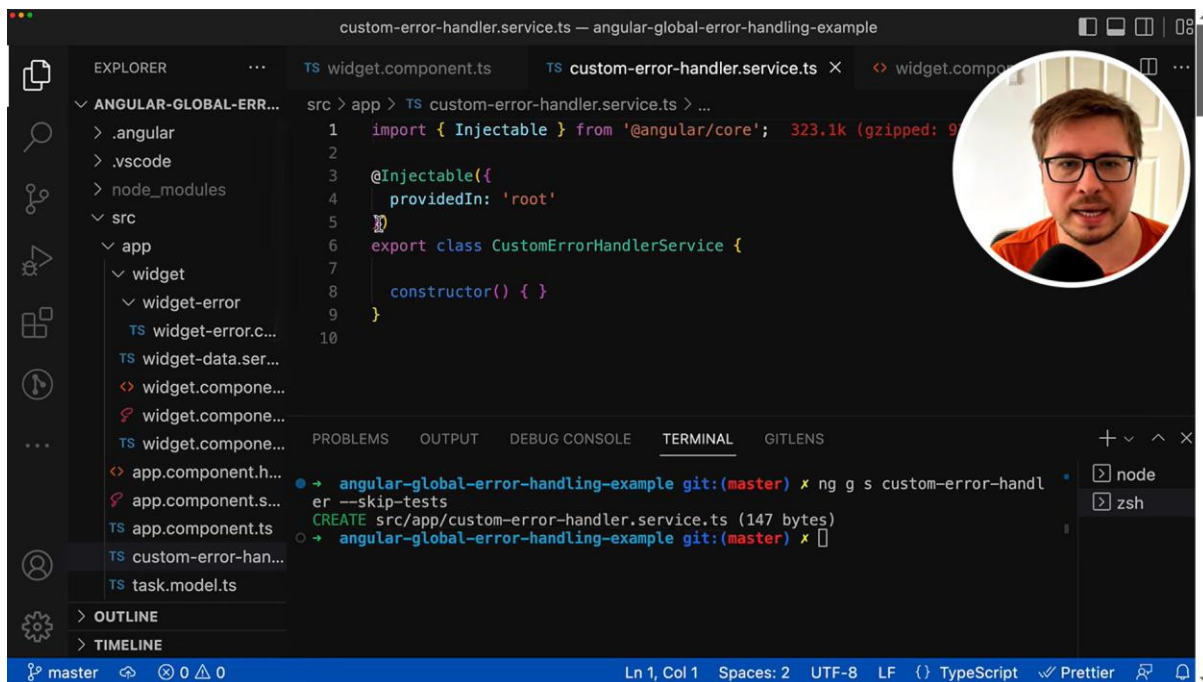
```
widget.component.ts — angular-global-error-handling-example
EXPLORER
  ANGULAR-GLOBAL-ERROR-HA...
    .angular
    .vscode
    node_modules
    src
      app
        widget
          widget-error
            widget-error.compon...
            widget-data.service.ts
            widget.component.html
            widget.component.scss
            widget.component.ts
          app.component.html
          app.component.scss
          app.component.ts
          task.model.ts
          assets
      OUTLINE
      TIMELINE
  master
  You, 2 minutes ago
  Ln 37, Col 6 (174 selected)
  Spaces: 2
  UTF-8
  LF
  TypeScript
  Prettier

src > app > widget > TS widget.component.ts > WidgetComponent > addTask
20 tasks$: Observable<Task[]>;
21 error: Error | null = null;
22
23
24 constructor(private widgetData: WidgetDataService) { }
25
26 ngOnInit(): void {
27   this.tasks$ = this.widgetData.load();
28 }
29
30 addTask() {
31   // unreliable method
32   try {
33     this.widgetData.addTaskSync({ id: 0, title: 'New Task' });
34   } catch (error) {
35     if (error instanceof Error) {
36       this.error = error;
37     }
38   }
39 }
40
```



```
Angular Error Handling
Page
  platform-browser.mjs
  zone.js
  core.mjs x
  13331 // or creating inner components, which current directive in
  13332 // to avoid relying on current def being 'null' (thus spec
  13333 // process of creating root component should be unified wi
  13334 // components.
  13335 if (currentDef === null || isComponentDef(currentDef)) {
  13336   lView = unwrapLView(lView[tNode.index]);
  13337 }
  13338 return lView[RENDERER];
  13339 }
  13340 /** Handles an error thrown in an LView. */
  13341 function handleError(lView, error) {
  13342   const injector = lView[INJECTOR$1];
  13343   const errorHandler = injector ? injector.get(ErrorHandler,
  13344     errorHandler && errorHandler.handleError(error);
  13345 }
  13346 /**
  13347 * Set the inputs of directives at the current node to corres
  13348 * @param tView The current TView
  13349 * @param lView the 'LView' which contains the directives.
  13350 * @param inputs mapping between the public "input" name and p
  13351 * possibly minified, property names to write to.
  13352 * @param value Value to set.
  13353 */
  13354 function setInputsForProperty(tView, lView, inputs, publicName
  13355   for (let i = 0; i < inputs.length;) {
  13356     // ...
  13357   }
  13358 }
  12 characters selected (source mapped from vendor.js) Coverage: n/a
Scope Watch
Breakpoints
  No breakpoints
  Not paused
```

Created custom-error-handler.service

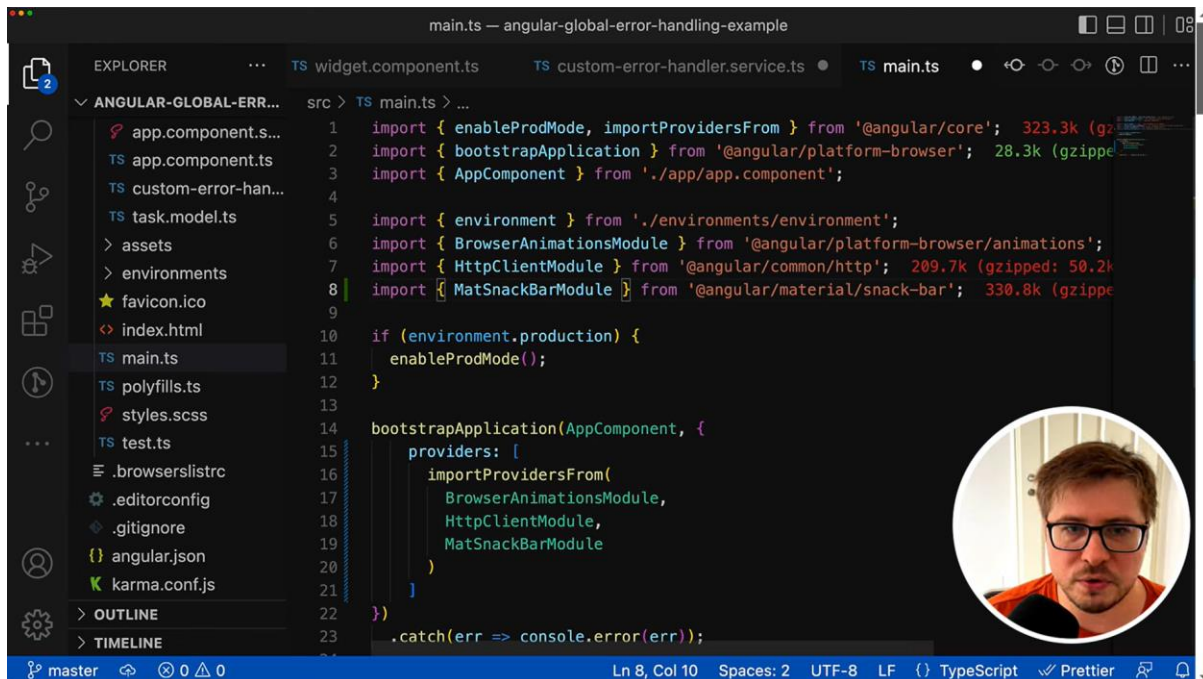


The image shows a VS Code editor window with the file explorer on the left and the editor on the right. The file explorer shows the project structure with a new file 'custom-error-handler.service.ts' created in the 'src/app' directory. The editor shows the code for 'custom-error-handler.service.ts' which is an injectable service.

```
1 import { Injectable } from '@angular/core';
2
3 @Injectable({
4   providedIn: 'root'
5 })
6 export class CustomErrorHandlerService {
7
8   constructor() {}
9
10 }
```

The terminal at the bottom shows the command to create the file: `ng g s custom-error-handl` and the output: `CREATE src/app/custom-error-handler.service.ts (147 bytes)`. A circular inset image of a man with glasses is visible in the top right corner.

Import MatSnackBar

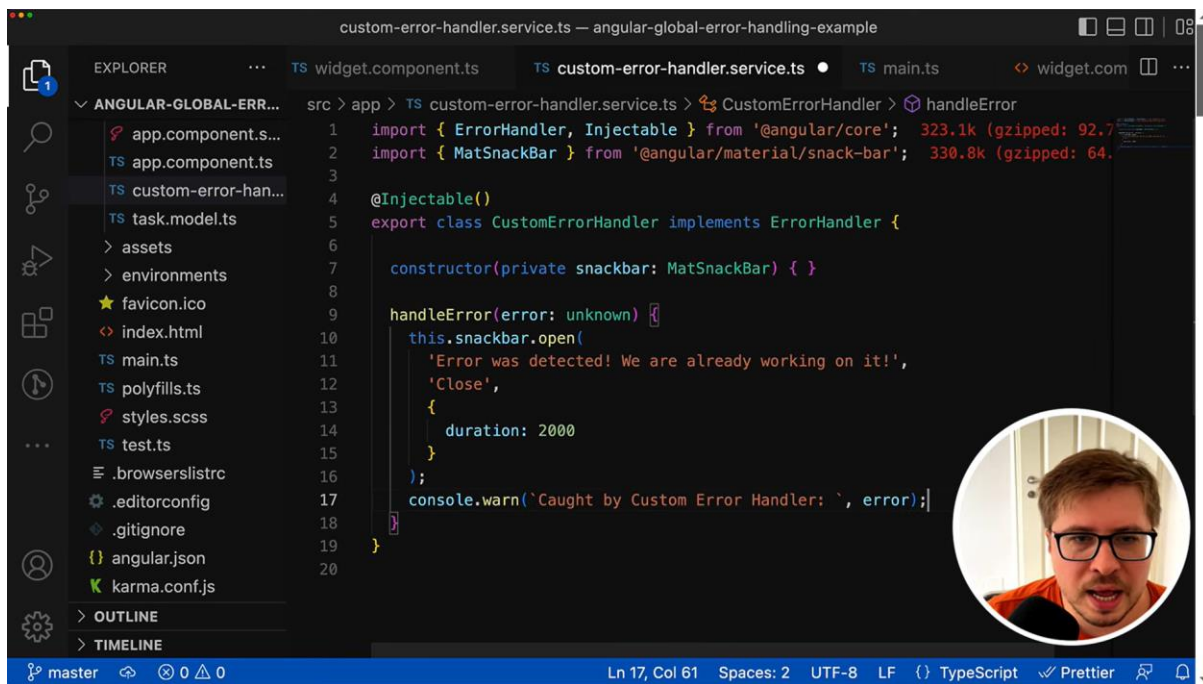


The image shows a VS Code editor window with the file explorer on the left and the editor on the right. The file explorer shows the project structure with a new file 'main.ts' created in the 'src' directory. The editor shows the code for 'main.ts' which imports 'MatSnackBarModule' from '@angular/material/snack-bar'.

```
1 import { enableProdMode, importProvidersFrom } from '@angular/core';
2 import { bootstrapApplication } from '@angular/platform-browser';
3 import { AppComponent } from './app/app.component';
4
5 import { environment } from './environments/environment';
6 import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
7 import { HttpClientModule } from '@angular/common/http';
8 import { MatSnackBarModule } from '@angular/material/snack-bar';
9
10 if (environment.production) {
11   enableProdMode();
12 }
13
14 bootstrapApplication(AppComponent, {
15   providers: [
16     importProvidersFrom(
17       BrowserAnimationsModule,
18       HttpClientModule,
19       MatSnackBarModule
20     )
21   ]
22 })
23 .catch(err => console.error(err));
```

The status bar at the bottom shows the file is at line 8, column 10. A circular inset image of a man with glasses is visible in the bottom right corner.

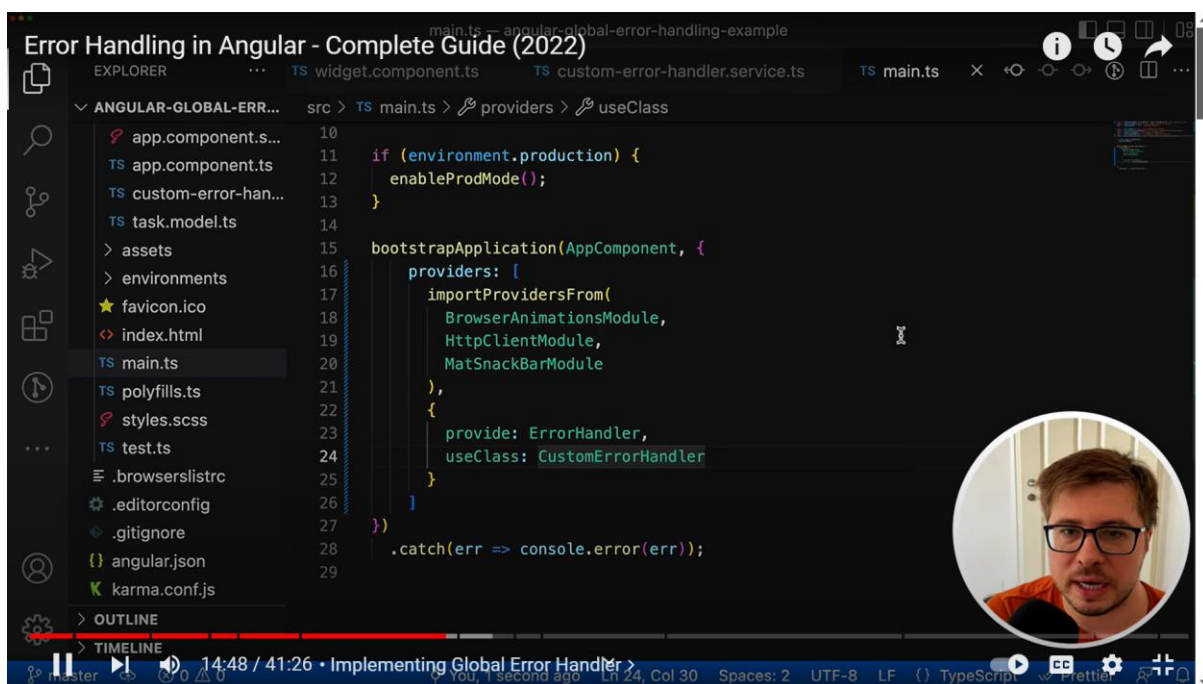
Error handler method



```
custom-error-handler.service.ts — angular-global-error-handling-example
EXPLORER
  ANGULAR-GLOBAL-ERR...
    app.component.s...
    app.component.ts
    custom-error-han...
    task.model.ts
    assets
    environments
    favicon.ico
    index.html
    main.ts
    polyfills.ts
    styles.scss
    test.ts
    .browserslistrc
    .editorconfig
    .gitignore
    angular.json
    karma.conf.js
    OUTLINE
    TIMELINE
  master
  0 0 0
  Ln 17, Col 61 Spaces: 2 UTF-8 LF TypeScript Prettier

src > app > TS custom-error-handler.service.ts > CustomErrorHandler > handleError
1 import { ErrorHandler, Injectable } from '@angular/core';
2 import { MatSnackBar } from '@angular/material/snack-bar';
3
4 @Injectable()
5 export class CustomErrorHandler implements ErrorHandler {
6
7   constructor(private snackbar: MatSnackBar) { }
8
9   handleError(error: unknown) {
10     this.snackbar.open(
11       'Error was detected! We are already working on it!',
12       'Close',
13       {
14         duration: 2000
15       }
16     );
17     console.warn('Caught by Custom Error Handler: ', error);
18   }
19 }
20
```

Inject this on main.ts or model for global level use



```
Error Handling in Angular - Complete Guide (2022)
main.ts — angular-global-error-handling-example
EXPLORER
  ANGULAR-GLOBAL-ERR...
    app.component.s...
    app.component.ts
    custom-error-han...
    task.model.ts
    assets
    environments
    favicon.ico
    index.html
    main.ts
    polyfills.ts
    styles.scss
    test.ts
    .browserslistrc
    .editorconfig
    .gitignore
    angular.json
    karma.conf.js
    OUTLINE
    TIMELINE
  master
  0 0 0
  Ln 24, Col 30 Spaces: 2 UTF-8 LF TypeScript Prettier

src > TS main.ts > providers > useClass
10
11 if (environment.production) {
12   enableProdMode();
13 }
14
15 bootstrapApplication(AppComponent, {
16   providers: [
17     importProvidersFrom(
18       BrowserModule,
19       HttpClientModule,
20       MatSnackBarModule
21     ),
22     {
23       provide: ErrorHandler,
24       useClass: CustomErrorHandler
25     }
26   ]
27 })
28 .catch(err => console.error(err));
29
```

Throw error for capturing global based

Angular Error Handling

Backlog Widget

Delectus Aut Autem

Quis Ut Nam Facilis Et Officia Qui

Fugiat Veniam Minus

Add new Task

Widget Error:
Value zero (0) is not allowed as a task id

[webpack-dev-server] Server started: Hot Module Replacement disabled, Live Reloading enabled, Progress disabled, Overlay enabled.

Angular is running in development mode. Call enableProdMode() to enable production mode.

Caught by Custom Error Handler: Error: Value zero (0) is not allowed as a task id

custom-error-handler.service.ts:17

WidgetDataService.addTaskSync (widget-data.service.ts:18:13)

WidgetComponent.addTask (widget.component.ts:32:23)

WidgetComponent.template_button_click_9_listener (widget.component.html:10:20)

executeListenerWithErrorHandling (core.mjs:15694:16)

wrapListenerIn_markDirtyAndPreventDefault (core.mjs:15729:22)


HTMLButtonElement.<anonymous> (platform-browser.mjs:459:38)

_ZoneDelegate.invokeTask (zone.js:406:31)

Object.onInvokeTask (core.mjs:26278:33)

_ZoneDelegate.invokeTask (zone.js:405:60)

Zone.runTask (zone.js:178:47)



Try-catch does not handle this error but by global

Error Handling in Angular - Complete Guide (2022)

EXPLORER


src > app > widget > TS widget.component.ts

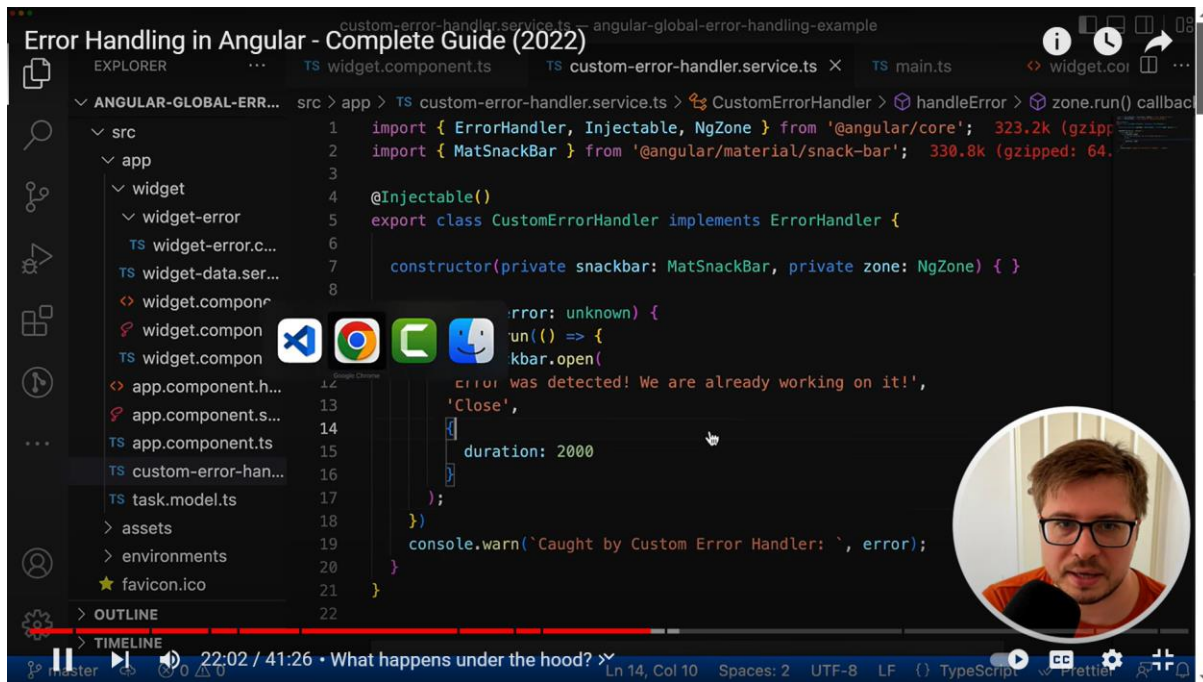
WidgetComponent

addTask

setTimeout() callback

```
22
23
24
25 constructor(private widgetData: WidgetDataService) { }
26
27 ngOnInit(): void {
28   this.tasks$ = this.widgetData.load();
29 }
30
31 addTask() {
32   // unreliable method
33   try {
34     setTimeout(() => {
35       this.widgetData.addTaskSync({ id: 0, title: 'New Task' });
36     });
37   } catch (error) {
38     if (error instanceof Error) {
39       this.error = error;
40       throw error;
41     }
42   }
43 }
```





`throw error` // used for send global based error

Try-catch will not work with Async call

```

try {
  // this is an example
  setTimeout(() => {
    this.widgetData.addTaskSync({ id: 0, title: 'New Task' });
  });
}
catch (error) {
  if (error instanceof Error) {
    this.error = error
    throw error // for global error
  }
}

```

Error Handling in Observables

2. error handling middleware
3. global error handling,
4. error handling within components.