**Angular video applicaton**

Developed by Josip Kenjerić

# 1. Getting started

Technologies used for this solutions are Angular JS 2, TypeScript, Node JS and Express JS. Angular 2 in combination with TypeScript is used to develop the application itself and Node with Express to develop API that will serve the application needs. Application was developed in Visual Studio Code.

# 2. Solution

## 2.1. Classes

Classes are written in TypeScript. There are two classes used for this solution: video class and VideoDetailed class. This approach is used because it is easier to have all videos saved at one JSON (JSON in this example) and get additional information for selected video trough new JSON. Like this it is possible to distinguish information that is needed for different views (although in this example both information will be used in presented views). Video class is going to present basic information about the videos such as id of the video, description and name of the video and thumbnail or image that is going to be set next to the information (Figure 1).

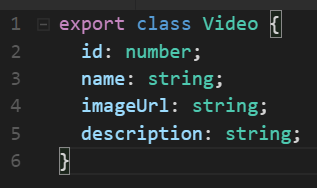


Figure 1 - Video class properties

VideoDetailed class on the other hand saves different information about the video such as size, author, date and link to video (Figure 2). In this example for each entry in videos JSON, application needs JSON that shares the name of the id property of video class to be able to fetch additional information about the video. If the file does not exist server will return 404.

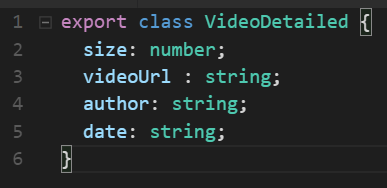


Figure 2 - VideoDetailed class properties

## 2.2. Angular Components

As mentioned earlier Angular 2 is used to develop this project. There are two main components apart from app.component (Figure 3). Components are great way to make code reusable.

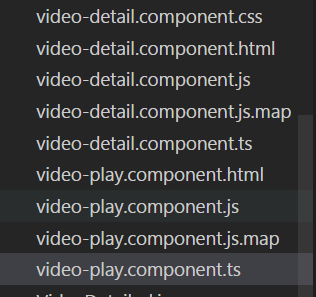


Figure 3 - List of projects components

First component is video-detail component. Video-detail component is used to show all the videos saved in JSON containing basic video information. This component upon initialisation fetches all the videos and places them on the site. When site is being initialised *ngOnInit()* method is called what triggers *getVideos()* method to be called and load information from JSON to scope. Videos are fetched trough service that will be explained later. There are three more methods used: *onSelect(video)* which highlights selected video, *goToDetail()* that will navigate to other view where video can be played and *generatePopUp(video)* which will fetch additional information from the selected video and present it in a pop up box above the additional information tab this method also grabs information trough service. \*ngFor directive is used to fill list with videos and \*ngIf directive is used to check if the object is ready. Rest of HTML and CSS for this example is basic and it will not be explained.

Second component is video-play. This component is called when user navigates from videos to desired video (*goToDetail()* navigates trough selected id). Here upon initialisation *ngOnInit()* method is called which grabs information about the video and once again this information is grabbed trough service. In HTML additional information about the video is presented. Video is played trough HTML5 video tag. Framework video.js is added to project and it is used for a player but somehow it is not giving desired result. As well three buttons are added to help navigate between videos: one button that returns user to videos page, one button for next and one for previous video. \*ngIf directive is used once again to check if object is ready.

Service is used to easily manage data calls - like this we don't have to rewrite the code over and over. Video service is used to get and parse JSON from our server. There are two main methods *getVideos()*  that fetch JSON with all videos and *getVideo(id)* that fetch JSON with additional information about the video. Third method *getVideoOriginal(id)* should return basic information about the video in play video view.

In app.component selector router-outlet is used. Trough this selector in app.module and ngModule we can assign our selectors that we have created in our components (my-video-detail and play-video) to different paths. Basically path will define component or selector that will be served.

## 2.3. Backend

To setup backend we have to install Node.js first on our machine. Framework express.js for node is used. Our sever is simple (Figure 4). Media folder have all JSON and static files images and videos.

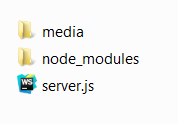


Figure 4 - Backend folder structure

Our server file is setup as follows: express is included, file reading is included, cross origin is enabled and media is set as static folder. To be able to read JSON files with data for our application we need read their content to be able to serve them (JSON files are not served as static). To be able to get resources from different domains Cross-Origin HTTP request has to be enabled. Since backend works on port 8081 and application on port 3000. All videos and images are served statically.

To get our JSON two different gets are in server file. When user sends request to "*get\_videos*" JSON with all videos will be in response. But when user sends request to "*video/:uid*" depending on id that was sent in :uid JSON with additional details about video will be in response.

# 3. Application

## 3.1. Navigating and views