Video Player Documentation

Files: script2.js

(Note: the line numbers may not be exactly accurate. When searching for a section of code it's probably best to Ctrl+f).

This file included all of the front-end functionality for the player. This includes the video player and controls, all even listeners for the creation of labels and annotations.

Video player functionality: Lines 141 – 310

Create New Annotation: Lines 311 – 321 & 449 – 498

Within lines 311 - 321 exist the event listener for the CREATE NEW LABEL BUTTON once it is pressed theses lines hear it and call the function **createNewLabel()** which is located on 449.



Once CREATE NEW LABEL pressed code below executes

```
newLabel = document.getElementById('create-new');

if (busy1 == false)

busy1 = true;

createNewLabel(); //Creates box to select color are select follows:

frequency for the select follows:

newLabel = document.getElementById('create-new');

//Create New');

function() {

if (busy1 == false)

function() {

if (busy1 == false)

//Creates box to select color are select follows:

//Creates box to select color are select follows:

//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box to select follows:
//Creates box follows:
/
```

Submit Button Pressed: Lines 324 – 340 & 502 - 561

Once label name and color selected press SUBMIT to execute creation of the new label. The event listener on 329 will get label name and color, them pass them to the function **submitLabel()** on line 502.



```
newBox = document.getElementById('new-new'); //container holding all new labels

//The next few event listeners use "event delegation" since the items created are done so dynamically within the browser. if(e.target && e.target.id == "<id of element clicked>" signifies event delegation);

//Once name is entered, color is picked and Submit is pressed

newBox.addEventListener('click', function(e){

if(e.target && e.target.id == "submit") //Event delagation

if(e.target && e.target.id == "submit") //Event delagation

color = document.getElementById("colorPick").value;

label = document.getElementById("namebox").value;

loadlock = false;

submitLabel(color, label); //New label is made using color and label name

auguments.getElementById("submit is pressed newById("namebox").value;

// New label is made using color and label name

// New label is made using color and label name

// New label is made using color and label name
```

Create new annotation: Lines 345 – 370 & 564 – 676

This event listener waits for the CREATE NEW ANNOTATION button to be pressed, which exists on the newly created label. Once the button is pressed the code below is executed. The event listener will call the **createNewAnnotation(color, label, kids, Grandparent Element)** on line 564, which will create the box which allows you to create annotation data.



```
labelContainer.addEventListener('click', function(e) {
345 ▼
                 var num = 0;
                 if(e.target && e.target.id == "addNew") //Event delegation
348 ▼
                      color = e.target.parentElement.style.backgroundColor;
                      label = e.target.parentElement.id;
                      var boxInd;
                      if(busy2 == false) {
353 ▼
                          busy2 = true;
                          var numibox = document.getElementsByTagName('tot');
358 ▼
                          for(var i = 0; i < numibox.length; i++) {</pre>
359 ▼
                              if(e.target.parentElement == numibox[i]) {
                          var kids = e.target.parentElement.parentElement.children;
                          kids = kids.length;
                          createNewAnnotation(color, label, kids,
                          e.target.parentElement.parentElement);
```

The if statement on 353 exists in this and other functions so that the user must create one annotation before they move onto the next

```
353 ▼ if(busy2 == false) {
354 busy2 = true;
```

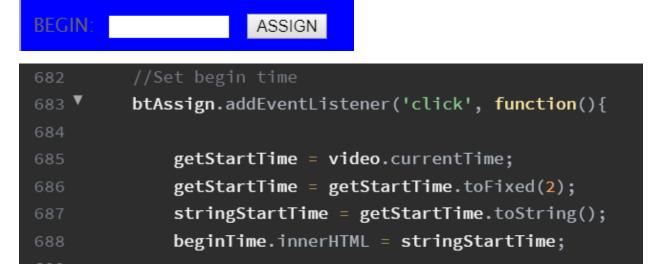
The code from 355 – 365 exists to determine the indices of the annotation that is created from the push of the CREATE NEW ANNOTATION button.

Most of the **createNewAnnotation()** function sets up the new annotation box, it also creates the box under the correct label. It does so using the GRAND PARENT ELEMENT passed to the function (the 4th parameter). On line 673 the function **drawEditAnnotation(color, label)** is called. This is the function that allows the user to create the annotation rectangle as well as set begin and end times.

The **drawEditAnnotation()** function is very important as this it contains the main functionality of the application.

drawEditAnnotation(): Lines 679 – 717

Lines 682 – 688: Event listener which allows user to set BEGIN time



Lines 693 – 701 Do the same thing but for the end time.

```
endAssign.addEventListener('click', function(){

694

695

696

getEndTime = video.currentTime;

getEndTime = getEndTime.toFixed(2);

stringEndTime = getEndTime.toString();

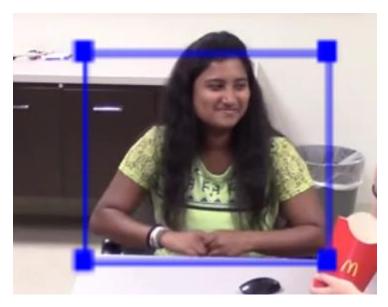
endTime.innerHTML = stringEndTime;

700

701

});
```

Before the user can finalize the annotation they must draw the annotation box on the canvas. Look at the example below.



Once the BEGIN and END times are set and the user has drawn the annotation box, they can then finalize the annotation.

Lines 703 - 715

These lines provide event listeners that actually draw and resize the annotation box.

Finalize Annotation: Lines 373 – 387

Once the user has set the begin and end times and drawn the annotation box they will press the *FINALIZE* button which will store the data by executing the code below. On line 381 the function **finalizeNewAnno(color)** is called.

The **finalizeNewAnno()** function updates the database with the annotation data the user just created.

```
//Finalize annotation by storing data in database
function finalizeNewAnno(color) {
    var x, y, w, h;
    x = currentAnnotation.x;
    y = currentAnnotation.y;
    w = currentAnnotation.w;
    h = currentAnnotation.h;

    var newAnnotationKey = database.ref().child('annotations').push().key;
    console.log(newAnnotationKey);
    var newAnnotationStatus = firebaseUpdate(newAnnotationKey, newAnnotationBox.id, getStartTime, getEndTime, x, y, w, h, videoName, color);
}
```

Delete Annotation: Lines 389 - 407

This event listener deletes an annotation. It does so when the user clicks the "X" on the annotation.



```
390 ▼
              labelContainer.addEventListener('click', function(e) {
391 ▼
                  if(e.target && e.target.id == "delete") {
                      var thisid = e.target.parentElement.id;
                      var kids = e.target.parentElement.parentElement.children;
395 ▼
                      for(var i = 0; i < kids.length; i++) {</pre>
396 ▼
                          if(kids[i].id == thisid) {
                              kids[i].parentElement.removeChild(kids[i]);
                          else if(kids[i].id == "nAB") {
399 ▼
                              kids[i].parentElement.removeChild(kids[i]);
                          busy2 = false;
                      removeAnnotation(thisid);
              });
```

Display finalized annotations:

Finalized annotations will minimize to reduce clutter. Once they have been minimized if the user wishes to see the annotations created they can click on the annotation (anywhere but the X on the example below) and the annotation box will display on the canvas where it was drawn. Also, the video current time will jump to the BEGIN time that the user set for the annotation.



```
labelContainer.addEventListener('click', function(e) {
                 if(e.target && e.target.name == "nabox")
                     if(busy1 == false && busy2 == false)
                         var myid = e.target.id;
                         var annotationPromise = database.ref().child(videoName + '/annotations
                         + myid).once("value")
                         .then(function(snapshot) {
                              return snapshot.val();
423 ▼
                         annotationPromise.then(function(key) {
                              var annotationDetails = database.ref().child('/annotations/' +
                              key).once("value")
                              .then(function(snapshot) {
                                  return snapshot.val();
428 ▼
                              .then(function(details) {
                                  showAnno(details.color, details.x, details.y, details.w,
                                 details.h, ctx);
                                 video.currentTime = details.start;
             });
```