ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΦΥΣΙΚΗΣ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΔΙΑΤΜΗΜΑΤΙΚΟ ΜΕΤΑΠΤΥΧΙΑΚΟ ΔΙΠΛΩΜΑ ΕΙΔΙΚΕΥΣΗΣ

ΣΤΟΝ ΗΛΕΚΤΡΟΝΙΚΟ ΑΥΤΟΜΑΤΙΣΜΟ

Ανάπτυξη λογισμικού για εμπλουτισμένη εμφάνιση περιεχομένου βίντεο σε HTML5

Δημήτριος Σ. Κατσίκης

Επιβλέπων: Γεώργιος Στασινόπουλος, Καθηγητής

ΑΘΗΝΑ

ΜΑΙΟΣ 2012

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη λογισμικού για εμπλουτισμένη εμφάνιση περιεχομένου βίντεο σε HTML5

Δημήτριος Σ. Κατσίκης
Α.Μ.: 2010506

ΕΠΙΒΛΕΠΩΝ: Γεώργιος Στασινόπουλος, Καθηγητής

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ: Διονύσιος Ρέϊσης, Επίκουρος Καθηγητής
Δημήτριος Φραντζεσκάκης, Αναπληρωτής Καθηγητής

Μάιος 2012

ΠΕΡΙΛΗΨΗ

Στόχος της παρούσας διπλωματικής διατριβής είναι η ανάπτυξη λογισμικού για την εμπλουτισμένη εμφάνιση περιεχομένου βίντεο. Αυτό το περιεχόμενο θα παρέχει πληροφορίες για τις διαφορετικές σκηνές του βίντεο. Ποιο συγκεκριμένα, θα εμφανίζονται εικόνες οι οποίες θα αντιπροσωπεύουν τις σκηνές του βίντεο. Το βίντεο και οι εμφανιζόμενες εικόνες θα είναι συγχρονισμένα και συντονισμένα. Για τον σκοπό αυτό θα εργαστούμε με Javascript και HTML5. Θα χρησιμοποιήσουμε κατά κύριο λόγο τα νέα element του HTML5 <video> και <canvas>.

Η παραπάνω λειτουργικότητα θα υλοποιηθεί ως ένα custom HTML tag με όνομα 'scenes'. Η υλοποίηση του scenes θα γίνει στην γλώσσα Javascript. Χρησιμοποιούμε μία τεχνική για την δημιουργία custom HTML tag από μία κλάση Javascript. Το αποτέλεσμα είναι το HTML tag <scenes>. Η συμπεριφορά του είναι όμοια με αυτήν των κανονικών HTML tag. Τα attribute που διαθέτει είναι: *videoID*, *number*, *frames*, *unit*, *fps*. Το **scenes**, επιπλέον, μπορεί να χρησιμοποιηθεί και ως αντικείμενο Javascript. Η μέθοδος δημιουργίας του (constructor method) είναι scenes(*tag*, *video*, *frames*, *number*).

Το tag <scenes> είναι μια συνεχώς μεταβαλλόμενη μπάρα από εικόνες τοποθετημένες η μία δίπλα στην άλλη. Οι εμφανιζόμενες εικόνες είναι επιλεγμένα frames από το βίντεο. Η κεντρική εικόνα είναι το πρώτο frame της τρέχουσας σκηνής του βίντεο. Οι εικόνες στα δεξιά είναι τα πρώτα frame των σκηνών που ακολουθούν και οι εικόνες στα αριστερά είναι τα πρώτα frame των σκηνών που προηγούνται. Ο αριθμός των εμφανιζόμενων σκηνών προσδιορίζεται από το attribute *number* και τα frame τα οποία θα αποσπαστούν από το βίντεο για να αντιπροσωπεύσουν την αρχή των σκηνών προσδιορίζονται από το attribute *frames*. Το μήκος του <scenes> είναι πάντα ίσο με το μήκος του βίντεο ανεξάρτητα από των αριθμό των εμφανιζόμενων εικόνων. Αν πατήσουμε με το ποντίκι πάνω σε μια εμφανιζόμενη εικόνα το βίντεο θα πάει στο αντίστοιχο frame. Περισσότερες λεπτομέρειες για την χρήση και την λειτουργικότητα του **scenes** στο documentation(τεκμηρίωση) στο παράρτημα 'Τεκμηρίωση'.

Στα επόμενα κεφάλαια εξηγούμε μέσω απλών παραδειγμάτων πως μπορούμε να χειριστούμε εφαρμογές που περιέχουν element <scenes> και πως μπορούμε να χρησιμοποιήσουμε το tag <scenes> για να αναπτύξουμε τις δικές μας εφαρμογές. Για να δοκιμάσουμε αυτές τις εφαρμογές χρησιμοποιούμε τον περιηγητή (browser) Firefox. Τέλος, στο πιο σημαντικό τμήμα της παρούσας εργασίας, παραθέτουμε ολόκληρο τον κώδικα της υλοποίησης μας και παρέχουμε μία λεπτομερή επεξήγηση αυτού του κώδικα γραμμή προς γραμμή.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Πολυμέσα
ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: HTML5, Javascript, video, σκηνές, frame

ABSTRACT

The aim of this master thesis is developing software for enhanced video content presentation. This content provides information about the different scenes of the video. Specifically, images represents video scenes will be displaying. The video and the displaying images will be synchronized and coordinated. For this purpose we worked with Javascript and HTML5. We will mainly use the new HTML5 elements <video> and <canvas>.

The functionality above implemented as a custom HTML tag named 'scenes'. The implementation of **scenes** was done in Javascript language. We used a technique to create a custom HTML tag from a Javascript class. The result is the HTML tag <scenes>. It's behavior is similar to normal HTML tags. Its attributes are: *videoID*, *number*, *frames*, *unit*, *fps*. **Scenes**, also, can used as a Javascript object. Its constructor method is scenes(*tag*, *video*, *frames*, *number*).

The tag <scenes> is an ever-changing bar of images placed next to each other. The displaying images are selected frames from the video. The central image is the first frame of the current scene of the video. The images in the right are the first frames of the following scenes and the images in the left are the first frames from the previews scenes. The number of displaying images is specified from attribute *number* and the frames will be captured from the video to represent the start of scenes are specified from attribute *frames*. The length of <scenes> is always equal to video length, regardless of the number of displaying images. If you click on an displaying image the video goes to the corresponding frame. More details about the use and functionality of **scenes** in the documentation in the annex 'Documentation'.

In the following chapters we explain via simple examples how to use applications which contains **scenes** elements and how to use **scenes** tag to develop custom applications. To test these applications we use the browser Firefox. Finally, in the most significant part of this thesis, we quote the whole code of our implementation and we provide a detailed illustration of this code line by line.

SUBJECT AREA: Multimedia
KEYWORDS: HTML5, Javascript, video, scenes, frame

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

1. ΠΡΟΛΟΓΟΣ

Η παρούσα διπλωματική διατριβή εκπονήθηκε κατά το χρονικό διάστημα από τον Σεπτέμβριο 2011 έως τον Απρίλιο του 2012 στα πλαίσια του Διατμηματικού Μεταπτυχιακού Διπλώματος Ειδίκευσης στον Ηλεκτρονικό Αυτοματισμό του Τμήματος Φυσικής και του Τμήματος Πληροφορικής και Τηλεπικοινωνιών του Εθνικού και Καποδιστριακού Πανεπιστημίου Αθηνών.

Θα ήθελα να εκφράσω τις θερμές μου ευχαριστίες στον Καθηγητή Γεώργιο Στασινόπουλο για την επίβλεψη και καθοδήγησή του κατά την διάρκεια υλοποίησης της παρούσας διατριβής.

2. ΕΙΣΑΓΩΓΗ

Η HTML5 παρέχει νέες δυνατότητες και ευκολίες στην ανάπτυξη διαδικτυακών εφαρμογών. Ιδιαίτερα σημαντικές είναι οι νέες δυνατότητες σε εφαρμογές που περιέχουν πολυμέσα και προκύπτουν από την εισαγωγή νέων HTML element.

Αντικείμενο της παρούσας διπλωματικής διατριβής είναι η ανάπτυξη λογισμικού για την εμπλουτισμένη εμφάνιση περιεχομένου βίντεο. Αυτό το περιεχόμενο θα παρέχει πληροφορίες για τις διαφορετικές σκηνές του βίντεο. Ποιο συγκεκριμένα, θα εμφανίζονται εικόνες οι οποίες θα αντιπροσωπεύουν τις σκηνές του βίντεο. Το βίντεο και οι εμφανιζόμενες εικόνες θα είναι συγχρονισμένα και συντονισμένα. Για τον σκοπό αυτό θα εργαστούμε με Javascript και HTML5. Θα χρησιμοποιήσουμε κατά κύριο λόγο τα νέα element του HTML5 <video> και <canvas>.

Πιο συγκεκριμένα η παραπάνω λειτουργικότητα θα ενταχθεί σε ένα custom HTML tag με το όνομα 'scenes'. Το scenes θα υλοποιηθεί σε γλώσσα Javascript ενώ θα χρησιμοποιηθεί σε εφαρμογές σε γλώσσα HTML5. Οι εφαρμογές αυτές θα δοκιμαστούν στον περιηγητή Firefox.

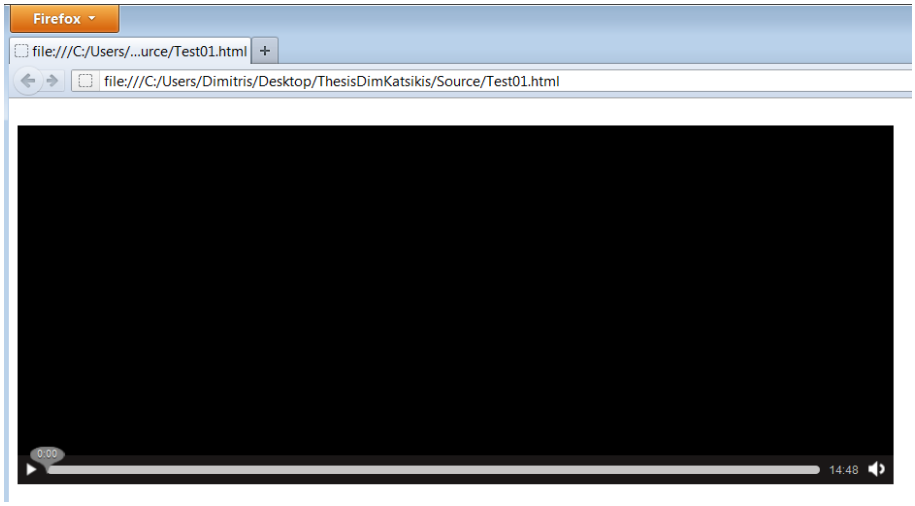
Αρχικά θα περιγράψουμε την λειτουργία του <scenes> στο κεφάλαιο 'ΠΕΡΙΓΡΑΦΗ ΛΕΙΤΟΥΡΓΙΑΣ'. Στην υποενότητα 'Χρήση εφαρμογών που περιέχουν το element <scenes>' θα δούμε πως μπορεί ο τελικός χρήστης να χειριστεί απλές εφαρμογές οι οποίες εμπεριέχουν το συγκεκριμένο element, μέσω του περιηγητή. Στην συνέχεια στην υποενότητα 'Τρόπος χρήσης του element <scenes> για την ανάπτυξη διαδικτυακών εφαρμογών' θα δούμε πως μπορούμε να χρησιμοποιήσουμε το tag <scenes> για την ανάπτυξη των δικών μας διαδικτυακών εφαρμογών μέσα από απλά παραδείγματα. Κατόπιν θα προχωρήσουμε στην αναλυτική επεξήγηση του τρόπου υλοποίησης του <scenes> στο κεφάλαιο 'ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ ΥΛΟΠΟΙΣΗΣΗΣ' όπου θα δώσουμε ιδιαίτερη έμφαση στην περιγραφή του κώδικα που χρησιμοποιήθηκε για την επίτευξη της προκύπτουσας λειτουργικότητας. Ο κώδικας θα εξεταστεί γραμμή προς γραμμή. Τέλος υπάρχει το κεφάλαιο 'ΠΡΟΤΑΣΕΙΣ ΓΙΑ ΜΕΛΛΟΝΤΙΚΗ ΕΠΕΚΤΑΣΗ' όπου παρατίθενται κάποιες σκέψεις για την περαιτέρω βελτίωση του **scenes**, προτάσεις για πιθανή χρησιμοποίηση του και ιδέες για ανάπτυξη λογισμικού που επεκτείνει την λειτουργικότητα που παρουσιάζεται στα πλαίσια της παρούσας εργασίας. Επίσης στο παρόν έγγραφο παρατίθενται και τρία παραρτήματα. Στο 'ΠΑΡΑΡΤΗΜΑ Ι - Documentation' και στο 'ΠΑΡΑΡΤΗΜΑ ΙΙ - ΤΕΚΜΗΡΙΩΣΗ' περιέχεται το documentation του αντικειμένου **scenes** στα αγγλικά και στα ελληνικά αντίστοιχα. Ενώ στο 'ΠΑΡΑΡΤΗΜΑ ΙΙΙ – ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ – ΧΡΗΣΙΜΕΣ ΔΙΑΣΥΝΔΕΣΕΙΣ' παρέχονται οι κύριες πηγές γνώσης οι οποίες χρησιμοποιήθηκαν ως θεωρητικό υπόβαθρο για την ανάπτυξη του λογισμικού που παρουσιάζουμε στο πλαίσιο της παρούσας διατριβής. Οι πηγές αυτές είναι κυρίως Tutorials, Specifications, βιβλία και σχετικές εφαρμογές που είναι δημοσιοποιημένες στο διαδίκτυο. Για κάθε πηγή δίνεται η περιγραφή της και το σχετικό link.

3. ΠΕΡΙΓΡΑΦΗ ΛΕΙΤΟΥΡΓΙΑΣ

1. Χρήση εφαρμογών που περιέχουν το element <scenes>

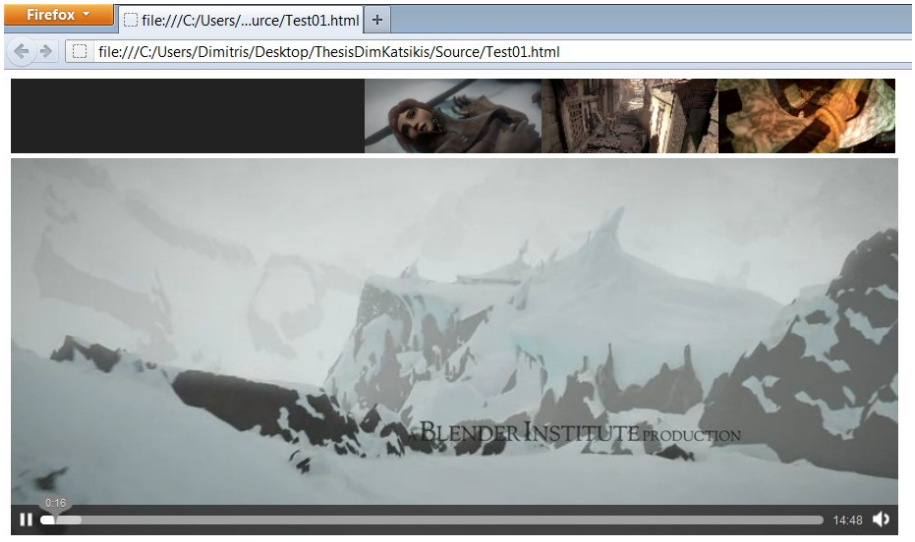
Στην παρούσα ενότητα θα περιγράψουμε τα βασικά σημεία της λειτουργίας του αντικειμένου **scenes** μέσα από μία απλή εφαρμογή. Η εφαρμογή μας δεν είναι τίποτα άλλο παρά μία ιστοσελίδα σε HTML5 η οποία περιέχει ένα HTML5 <video> element και το custom HTML5 element <scenes>. Το προβαλλόμενο βίντεο είναι τύπου webm και η δοκιμή θα γίνει στο περιηγητή Firefox.

Κατά την φόρτωση της ιστοσελίδας παρατηρούμε μια μαύρη εικόνα η οποία αντιπροσωπεύει το βίντεο το οποίο δεν έχει ξεκινήσει να παίζει.



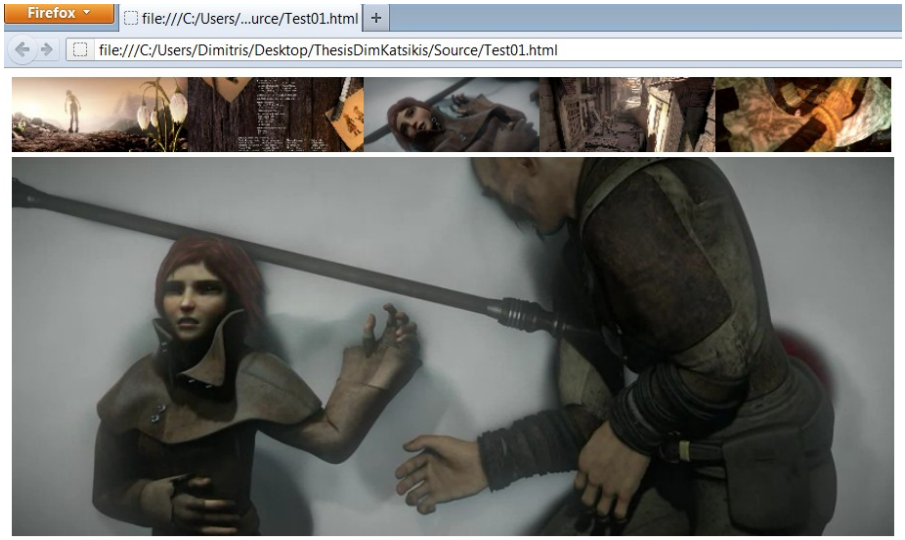
Εικόνα 1: Πριν πατήσουμε το play

Μόλις πατήσουμε το play θα εμφανισθεί η μία μπάρα η οποία περιέχει εικόνες οι οποίες αντιπροσωπεύουν τις σκηνές του βίντεο. Η μπάρα αυτή είναι το το περιεχόμενο του <scenes>. Κάθε εικόνα είναι το πρώτο frame κάθε σκηνής του βίντεο. Μέχρι να φορτωθούν όλα τα frames εμφανίζεται ένα σκούρο γκρι φόντο. Στην παρακάτω εικόνα εμφανίζονται με γκρι φόντο οι δύο αριστερότερες εικόνες καθώς τα αντίστοιχα frames δεν έχουν φορτωθεί ακόμα.



Εικόνα 2: Μόλις πατήσουμε το play

Στην παρακάτω εικόνα έχουν φορτωθεί όλα τα frames τα οποία αντιπροσωπεύουν τις διαφορετικές σκηνές του βίντεο. Η ταχύτητα με την οποία φορτώνονται τα frames εξαρτάται κυρίως από την ταχύτητα της σύνδεσης στο διαδίκτυο.

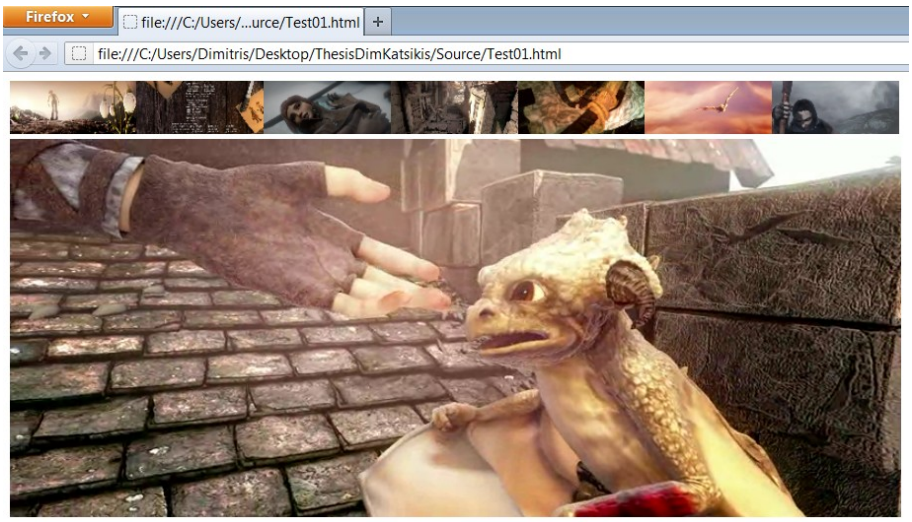


Εικόνα 3: Έχουν φορτωθεί όλα τα frames

Η εικόνα που εμφανίζεται στο κέντρο της μπάρας είναι το επιλεγμένο frame που μόλις παίχτηκε και αντιπροσωπεύει την τρέχουσα σκηνή. Συγκεκριμένα είναι το πρώτο frame της τρέχουσας σκηνής.

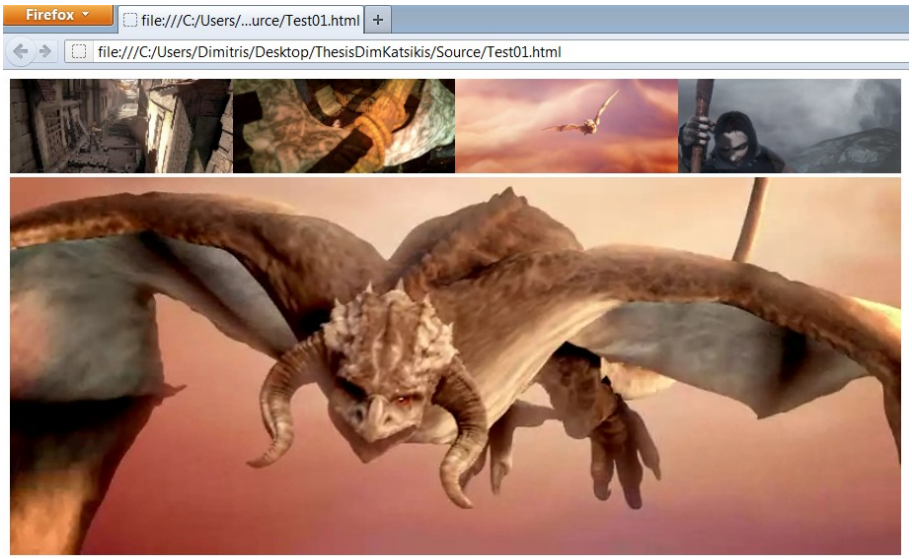
Οι εικόνες που εμφανίζονται στα δεξιά είναι τα αρχικά frames των σκηνών που ακολουθούν ενώ οι εικόνες που βρίσκονται στα δεξιά είναι τα αρχικά frames των σκηνών που έχουν προηγηθεί. Στην περίπτωση που δεν υπάρχουν προηγούμενες σκηνές, όταν για παράδειγμα η τρέχουσα σκηνή είναι η αρχική τότε στα αριστερά θα εμφανιστούν τα frames των τελευταίων σκηνών του βίντεο. Κάτι τέτοιο συμβαίνει και στην παραπάνω εικόνα που παρατηρούμε αριστερά του frame που μόλις παίχτηκε το frame που απεικονίζει τους τίτλους τέλους.

Ο αριθμός των εικόνων που θα απεικονίζονται δεν είναι συγκεκριμένος. Μπορούμε να έχουμε οποιονδήποτε αριθμό από frames για παράδειγμα στην παρακάτω εικόνα βλέπουμε 7.



Εικόνα 4: Εμφανίζονται 7 εικόνες

Συνίσταται να έχουμε περιττό αριθμό εικόνων έτσι ώστε η τρέχουσα σκηνή να αντιπροσωπεύεται από την κεντρική εικόνα. Ωστόσο, καθώς δεν υπάρχει περιορισμός σε αυτό, είναι δυνατόν να έχουμε και άρτιο αριθμό εικόνων για παράδειγμα 4. Σε αυτή την περίπτωση η εικόνα που αντιπροσωπεύει την τρέχουσα σκηνή είναι αυτή που βρίσκεται αμέσως δεξιότερα από την μέση της μπάρας.

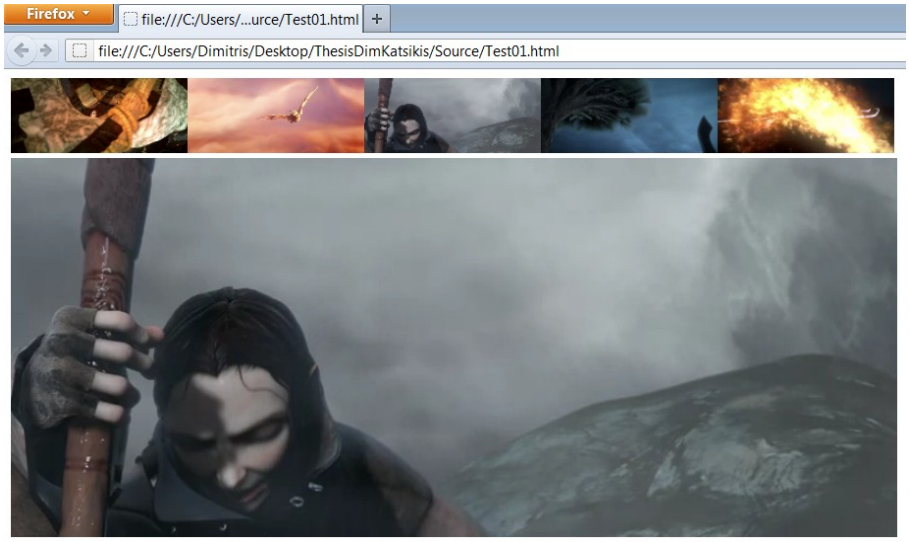


Εικόνα 5: Εμφανίζονται 4 εικόνες

Εδώ είναι σημαντικό να παρατηρήσουμε πως ανεξάρτητα από τον αριθμό των απεικονιζόμενων frames το μήκος της μπάρας είναι σταθερό και ισούται με το μήκος του βίντεο.

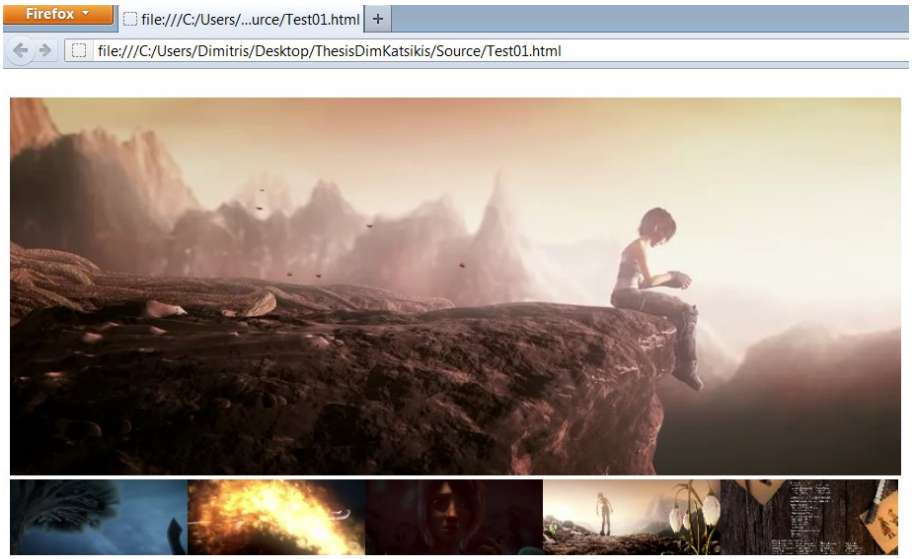
Αν μετακινηθούμε με χειροκίνητο τρόπο σε κάποιο σημείο του βίντεο η μπάρα προσαρμόζεται αυτόματα έτσι ώστε κάθε στιγμή η κεντρική εικόνα να αντιπροσωπεύει την τρέχουσα σκηνή, οι δεξιότερες εικόνες τις σκηνές που ακολουθούν και οι αριστερότερες τις σκηνές που προηγούνται. Με αυτόν τον τρόπο <scenes> και <video> είναι πάντα συγχρονισμένα.

Ακόμη ένα χαρακτηριστικό της εφαρμογής είναι πως αν πατήσουμε με το ποντίκι πάνω σε οποιαδήποτε εικόνα τότε το βίντεο μεταφέρεται αυτόματα στο σημείο που εμφανίζεται το αντίστοιχο frame ή αλλιώς στην αρχή της σκηνής που θα επιλέξουμε.



Εικόνα 6: Μεταφορά στην επιλεγμένη σκηνή

Το σημείο στο οποίο εμφανίζεται η μπάρα δεν είναι συγκεκριμένο. Το <scenes> μπορούμε να το τοποθετήσουμε πάνω από το προβαλλόμενο βίντεο, κάτω, δίπλα ή και οπουδήποτε αλλού μέσα στην ιστοσελίδα μας. Όπως, για παράδειγμα, στην παρακάτω εικόνα όπου η μπάρα εμφανίζεται κάτω από το προβαλλόμενο βίντεο.



Εικόνα 7: Η μπάρα κάτω από το βίντεο

2. Τρόπος χρήσης του element <scenes> για την ανάπτυξη διαδικτυακών εφαρμογών.

Σε αυτήν την ενότητα θα εξετάσουμε τους τρόπους χρήσης του custom multimedia HTML element <scenes>. Θα ξεκινήσουμε με μία απλή εφαρμογή. Πρόκειται για μια ιστοσελίδα HTML στην οποία έχουμε τοποθετήσει ένα <video> element και ένα <scenes> element. Η ιστοσελίδα αυτή είναι παρόμοια με την ιστοσελίδα που παρουσιάσαμε σε προηγούμενη ενότητα όταν περιγράφαμε την λειτουργία του HTML element. Παραθέτουμε τον HTML κώδικα που περιέχει η ιστοσελίδα:

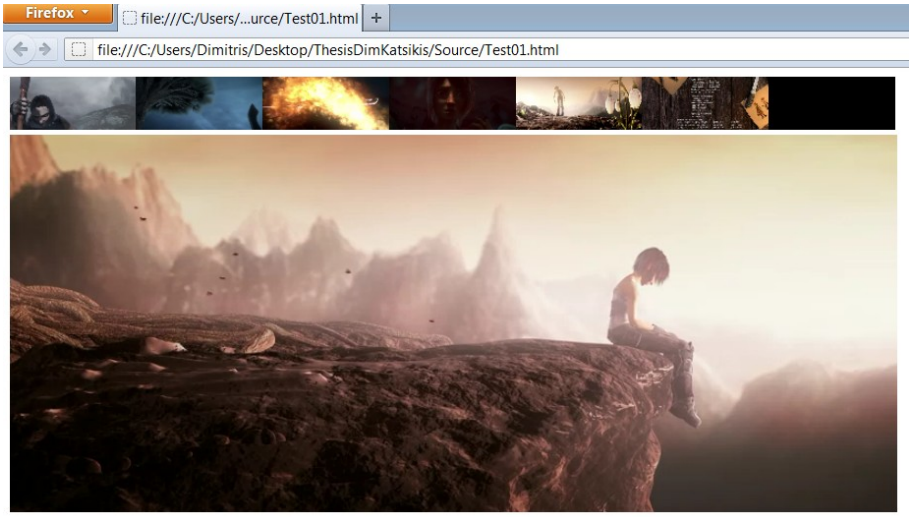
```
1. <!DOCTYPE HTML>
2. <html>
3. <head>
4. <script type="text/javascript" src="MyLibrary.js"></script>
5. </head>
6. <body onload = "initializeUI()">
7. <scenes number = "7" frames = "0 80 160 240 320 400 480 560 640 720 800"
   videoID = "vid1"></scenes>
8. </br>
9. <video
   src=http://cdnbakmi.kaltura.com/p/243342/sp/24334200/serveFlavor/flavorId/
   0_1c89hhxb/name/0_1c89hhxb.webm> id = "vid1" controls="controls">
10. </video>
11. </br>
```

12. `</body>`
13. `</html>`

Πρόκειται για ένα τυπικό HTML έγγραφο. Τα σημεία του κώδικα στα οποία θα εστιάσουμε περισσότερο είναι σε έντονη γραφή.

Στην σειρά 4 φορτώνεται η βιβλιοθήκη MyLibrary η οποία δεν είναι τίποτα άλλο παρά ένα Javascript αρχείο το οποίο περιέχει την υλοποίηση του element <scenes>. Απαραίτητη για την λειτουργία της εφαρμογής μας είναι και η σειρά 6 στην οποία γίνεται η αρχικοποίηση του element. Πιο συγκεκριμένα κατά την φόρτωση της σελίδας γίνεται κλήση της συνάρτησης **initializeUI**, η οποία βρίσκεται εντός της MyLibrary, και αναλαμβάνει την αρχικοποίηση του <scenes>. Περισσότερες λεπτομέρειες σχετικά με το περιεχόμενο της MyLibrary και της λειτουργίας της **initializeUI** θα δοθούν στην ενότητα 'ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ ΥΛΟΠΟΙΗΣΗΣ'. Αυτό που μας ενδιαφέρει για την ώρα είναι πως μετά την προσθήκη αυτών των δύο γραμμών κώδικα μπορούμε να χρησιμοποιούμε το tag <scenes> σαν να ήταν ένα οποιοδήποτε κοινό HTML tag.

Κατόπιν προχωράμε στην σειρά 7 όπου γίνεται η τοποθέτηση και η αρχικοποίηση των attributes του στοιχείου <scenes>. Συγκεκριμένα θέτουμε *number* = '7' έτσι ώστε στην μπάρα να εμφανίζονται 7 εικόνες. Στην συνέχεια θέτουμε *frames* = "0 80 160 240 320 400 480 560 640 720 800". Το string αυτό περιέχει αριθμούς που αντιστοιχούν στις χρονικές στιγμές, σε δευτερόλεπτα, στις οποίες εμφανίζονται επιλεγμένα frames από το βίντεο τα οποία αντιπροσωπεύουν τις διαφορετικές σκηνές. Οι αριθμοί αυτοί θα μπορούσαν να είναι και δεκαδικοί δηλαδή θα μπορούσαμε να είχαμε *frames* = "0 80.5 160.9 240.8 320 400 480.4 560 640 720 800.02". Οι αριθμοί αυτοί θα πρέπει να διαχωρίζονται μεταξύ τους με ένα κενό. Τέλος το attribute *videoID* παίρνει ως τιμή το id του <video> με το οποίο συσχετίζεται. Το αποτέλεσμα του παρακάτω κώδικα φαίνεται στην παρακάτω εικόνα.



Εικόνα 8: <scenes> με number = 7

Το attribute *frames* έχει την δυνατότητα να παίρνει αντί για τα δευτερόλεπτα τους αριθμούς που αντιστοιχούν στην σειρά εμφάνισης των επιλεγμένων frames στο βίντεο. Σε αυτήν την περίπτωση οι αριθμοί που θα περιέχονται στο string θα πρέπει απαραιτήτως να είναι ακέραιοι. Το αν θα παίρνει δευτερόλεπτα ή αριθμούς σειράς εμφάνισης των frames εξαρτάται από την τιμή του attribute *unit*. Το attribute *unit* μπορεί να πάρει δύο δυνατές τιμές 'second' ή 'number'. Παίρνει τιμή 'second' όταν θέλουμε να περάσουμε δευτερόλεπτα και 'number' όταν θέλουμε να περάσουμε αριθμούς σειράς. Στην περίπτωση όμως που μόλις εξετάσαμε δεν εμφανίζεται πουθενά το attribute *unit*. Τι συμβαίνει σε αυτή την περίπτωση; Αυτό που έχει συμβεί είναι πως η μεταβλητή *unit* παίρνει από προεπιλογή (by default) την τιμή 'second' οπότε οι παρακάτω δύο γραμμές κώδικα

είναι ισοδύναμες.

```
<scenes number = "7" frames = "0 80 160 240 320 400 480 560 640 720 800" videoID
= "vid1"></scenes>
```

```
<scenes number = "7" frames = "0 80 160 240 320 400 480 560 640 720 800" videoID
= "vid1" unit = "second"></scenes>
```

Στην περίπτωση τώρα που έχουμε θέσει *unit* = 'number' πρέπει να καθορίσουμε και το fps του βίντεο καθώς εσωτερικά πρέπει να γίνει η μετατροπή σε δευτερόλεπτα. Αυτό συμβαίνει γιατί το element <video> δεν έχει τρόπο για να διαχειρίζεται απευθείας αριθμούς frames αλλά έχει πρόσβαση στα frames μόνο μέσα από την χρονική στιγμή στην οποία εμφανίζονται. Για αυτόν τον λόγο υπάρχει το attribute *fps* το οποίο έχει νόημα μόνο εφόσον έχουμε θέσει *unit* = 'number.' Εφόσον γνωρίζουμε πως το παραπάνω βίντεο έχει fps = 29,97 αν αντικαταστήσουμε την γραμμή 7 του κώδικα του παραδείγματός μας με την παρακάτω γραμμή το αποτέλεσμα θα είναι ισοδύναμο.

```
<scenes number = "7" frames = "2400 4800 7200 9600 12000 14400 16800 19200 21600
24000" videoID = "vid1" unit = "number" fps = 29,97></scenes>
```

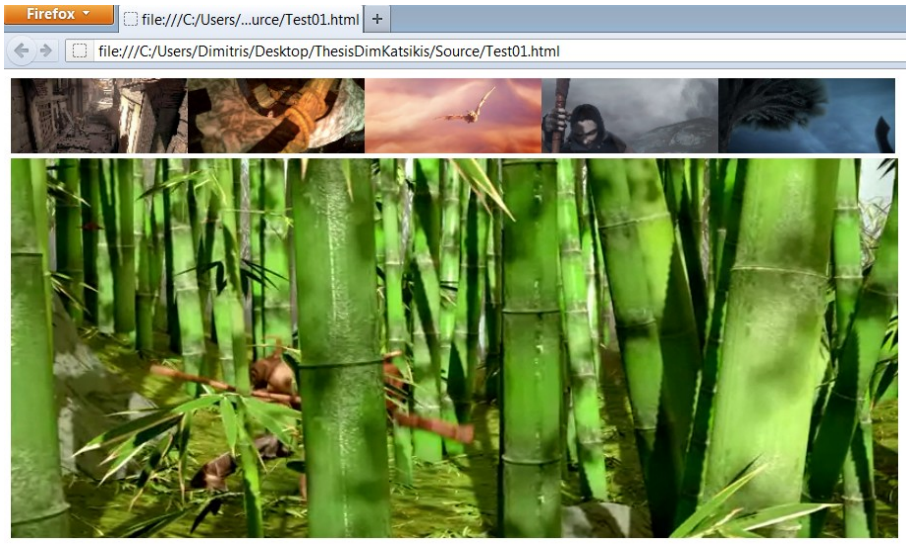
Το *fps* παίρνει από προεπιλογή την τιμή 29.97 οπότε η παρακάτω γραμμή κώδικα, στην οποία δεν υπάρχει το attribute *fps*, δίνει ισοδύναμο αποτέλεσμα με την προηγηθείσα γραμμή κώδικα.

```
<scenes number = "7" frames = "2400 4800 7200 9600 12000 14400 16800 19200 21600
24000" videoID = "vid1" unit = "number" ></scenes>
```

Ένα από τα κυριότερα χαρακτηριστικά του element <scenes> είναι πως έχει default τιμές για όλα του τα attributes. Έχουμε ήδη εξετάσει τι συμβαίνει στην περίπτωση που δεν δώσουμε τιμή στα attributes *unit* και *fps*. Τώρα θα δούμε τι θα συμβεί αν δεν δώσουμε τιμή στο attribute *number*. Αφαιρούμε, λοιπόν από την γραμμή 7 του παραδείγματός μας το attribute *number* και έχουμε:

```
<scenes frames = "0 80 160 240 320 400 480 560 640 720 800" videoID =
"vid1"></scenes>
```

Το αποτέλεσμα αυτής της αλλαγής φαίνεται στην επόμενη εικόνα, από την οποία γίνεται φανερό πως το attribute *number* παίρνει από προεπιλογή την τιμή 5.



Εικόνα 9: Attribute number από προεπιλογή

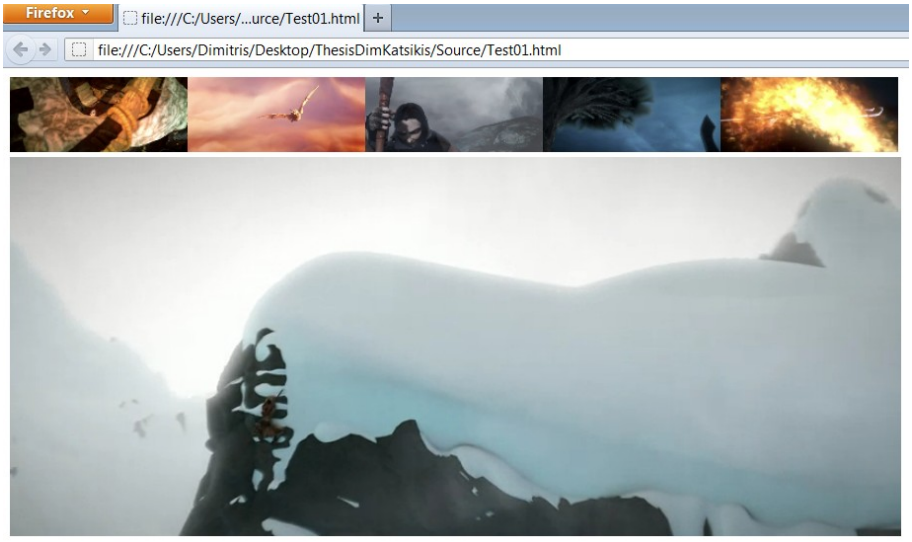
Στην συνέχεια θα εξετάσουμε τι θα συμβεί αν δεν ορίσουμε τιμή για το attribute *videoID*. Σε αυτή την περίπτωση αυτή θα έχουμε:

```
<scenes frames = "0 80 160 240 320 400 480 560 640 720 800" ></scenes>
```

Το αποτέλεσμα είναι πως το scenes εξακολουθεί να είναι συσχετισμένο με το ίδιο <video> χωρίς να προσδιορίσουμε το id του βίντεο. Αυτό που συμβαίνει είναι πως στην περίπτωση που δεν προσδιορίσουμε την τιμή του attribute *videoID* τότε θα επιλεχθεί ως συσχετιζόμενο βίντεο το πρώτο HTML <video> element που ακολουθεί το element <scenes> στον κώδικα.

Στην περίπτωσή μας το <video> element που ακολουθεί είναι:

```
<video
src=http://cdnbakmi.kaltura.com/p/243342/sp/24334200/serveFlavor/flavorId/0_1c89
hhxb/name/0_1c89hhxb.webm> id = "vid1" controls="controls">
```

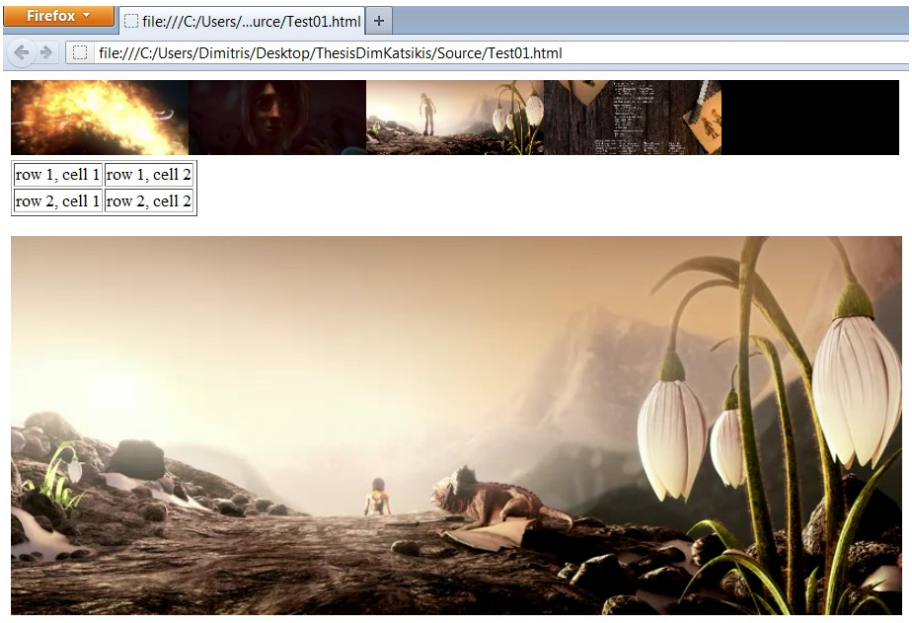


Εικόνα 10: Attribute videoID από προεπιλογή

Δεν έχει σημασία αν παρεμβάλλεται κάποιο άλλου είδους tag μεταξύ του <scenes> και του <video>. Έστω για παράδειγμα πως έχουμε τον παρακάτω κώδικα:

```
<scenes frames = "0 80 160 240 320 400 480 560 640 720 800"></scenes>
</br>
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
</br>
<video src=TestVideo.webm id = "vid1" controls="controls">
</video>
```

Σε αυτήν την περίπτωση μεταξύ του <scenes> και του <video> παρεμβάλλεται ένα element <table> (πίνακας), ενώ δεν προσδιορίζουμε με ποιο <video> συσχετίζεται το element <scenes> καθώς δεν προσδιορίζουμε το attribute *videoID*. Το αποτέλεσμα του παραπάνω κώδικα φαίνεται στην παρακάτω εικόνα, όπου γίνεται φανερό πως το element <table> δεν επηρεάζει καθόλου την εφαρμογή μας.

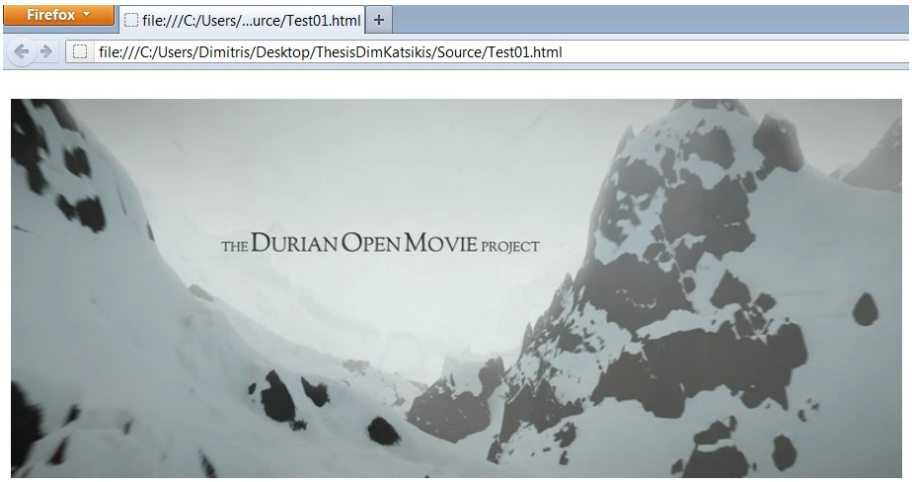


Εικόνα 11: Μεταξύ <scenes> και <video> παρεμβάλεται ένα <table>

Το element <scenes> μπορεί να τοποθετηθεί σε οποιοδήποτε σημείο της σελίδας. Τώρα θα δούμε τι θα συμβεί αν το τοποθετήσουμε στο κάτω μέρος του <video> χωρίς όμως να προσδιορίσουμε το attribute *videoID*. Ο αντίστοιχος κώδικας θα είναι:

```
<video src=TestVideo.webm id = "vid1" controls="controls">
</video>
</br>
<scenes frames = "0 80 160 240 320 400 480 560 640 720 800"></scenes>
```

Το αποτέλεσμα είναι να μην εμφανίζεται καθόλου η μπάρα με τις σκηνές. Αυτό συμβαίνει επειδή εφόσον δεν προσδιορίζουμε ρητά το id του συσχετιζόμενου <video>, το <scenes> θα επιχειρήσει να συσχετιστεί με το <video> που ακολουθεί, με αποτέλεσμα να αποτύχει καθώς δεν υπάρχει κάποιο <video> element μετά το <scenes> στον παραπάνω κώδικα.



Εικόνα 12: Τοποθετούμε το <scenes> κάτω από το <video> χωρίς να έχουμε προσδιορίσει το video ID

Προηγουμένως είπαμε πως το element <scenes> έχει default τιμές για όλα του τα attributes. Η μόνη περίπτωση που έχουμε αφήσει είναι να μην προσδιορίσουμε ρητά την τιμή του attribute *frames*. Μια τέτοια περίπτωση έχουμε στον παρακάτω κώδικα:

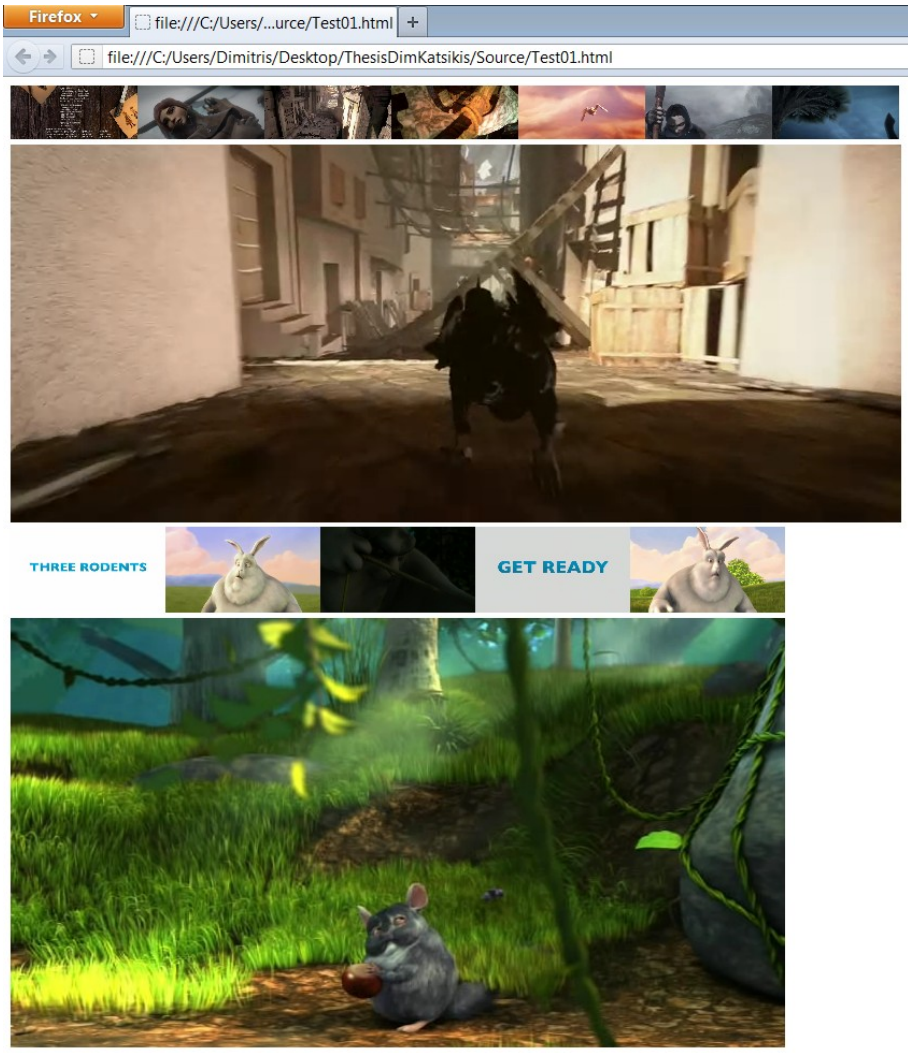
```
<scenes></scenes>
</br>
<video src=TestVideo.webm id = "vid1" controls="controls">
Your browser does not support the video tag.
</video>
```

Σε αυτήν την περίπτωση παρατηρούμε πως δεν έχουμε προσδιορίσει κανένα attribute του <scenes>, κάτι που σημαίνει πως όλα τα attributes θα πάρουν τις αντίστοιχες προεπιλεγμένες τιμές. Δεδομένου ότι έχουμε εξετάσει όλες τις άλλες περιπτώσεις εκτός από την περίπτωση που δεν προσδιορίζουμε το attribute *frames* θα εστιάσουμε εκεί.

Το attribute *frames* καθορίζει τις χρονικές στιγμές που εμφανίζονται στο βίντεο τα επιλεγμένα frames. Άρα, λοιπόν, μιλάμε για κάποιες χρονικές στιγμές που θα ορίζονται εσωτερικά σε περίπτωση που δεν προσδιορίζονται με ρητό τρόπο μέσα από το attribute *frames*. Το πρώτο που μας απασχολεί είναι πόσες είναι αυτές οι χρονικές στιγμές που επιλέγονται και το δεύτερο ποιες ακριβώς. Σχετικά με το πρώτο ο αριθμός των χρονικών στιγμών που θα επιλεγούν είναι ο διπλάσιος του αριθμού των εικόνων που εμφανίζονται στο <scenes>, δηλαδή είναι 2**number*, οπότε στην περίπτωση μας που το *number* παίρνει από προεπιλογή την τιμή 5 θα είναι 10. Τώρα για να επιλεχθούν ποιες ακριβώς θα είναι αυτές οι δέκα χρονικές στιγμές χωρίζουμε το video σε δέκα ίσα χρονικά διαστήματα και ξεκινώντας από την χρονική στιγμή 0 προχωράμε κάθε φορά το ένα

δέκατο της διάρκειας του βίντεο και επιλέγουμε την αντίστοιχη χρονική στιγμή μέχρι να επιλέξουμε τις δέκα πρώτες. Με αυτόν τον τρόπο γίνεται η by default επιλογή των frames που αντιπροσωπεύουν τις διαφορετικές σκηνές του βίντεο.

Σε μία ιστοσελίδα, προφανώς μπορούμε να έχουμε πολλά elements <scenes> θα πρέπει ωστόσο να καθορίζουμε το id του συσχετιζόμενου <video> αλλιώς, όπως εξετάσαμε προηγουμένως, το <scenes> θα συσχετιστεί με το ακριβώς επόμενο <video> element στον HTML κώδικα. Στην παρακάτω περίπτωση έχω μια ιστοσελίδα με δύο <video> elements πάνω από τα οποία βρίσκονται αντίστοιχα δύο <scenes> elements. Το κάθε <scenes> προσδιορίζει μέσω του *videoID* με ποιο video συσχετίζεται:



Εικόνα 13: Δύο <video> με αντίστοιχα δύο <scenes> στην ίδια ιστοσελίδα

Η παραπάνω εικόνα προκύπτει από τον HTML κώδικα που που παραθέτουμε ακολούθως.

```
<scenes videoID = "vid1" number = "7" frames = "80.5 160.9 240 320.25 400 480
560 640.05 720 800"></scenes>
</br>
<video src=TestVideo.webm id = "vid1" controls="controls">
</video>
</br>
```

```
<scenes videoID = "vid2" number = "5"></scenes>
</br>
<video src=TestVideo2.ogv id = "vid2" controls="controls">
</video>
```

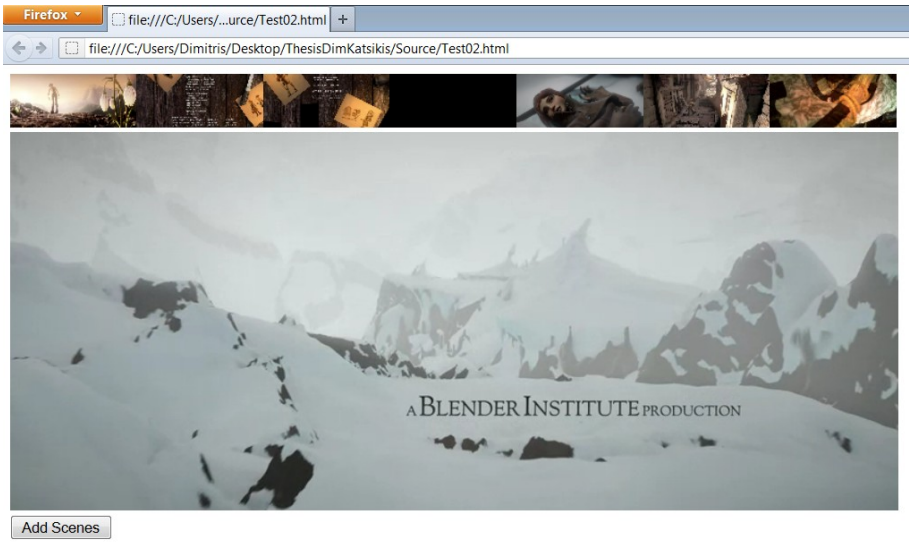
Στις εφαρμογές που εξετάσαμε ως τώρα το element <scenes> είναι τοποθετημένο στατικά στην ιστοσελίδα. Με την εφαρμογή που ακολουθεί θα δούμε πως μπορούμε να διαχειριστούμε το element αυτό με ποιο δυναμικό τρόπο. Παρακάτω παραθέτω τον κώδικα της ιστοσελίδας.

```
1. <!DOCTYPE HTML>
2. <html>
3. <head>
4. <script type="text/javascript" src="MyLibrary.js"></script>
5. <script type="text/javascript" src="Test02.js"></script>
6. </head>
7. <body onload = "initializeUI()">
8. <div id = "div1"></div>
9. <video src =
   http://cdnbakmi.kaltura.com/p/243342/sp/24334200/serveFlavor/flavorId/0_1c
   89hhxb/name/0_1c89hhxb.webm id = "vid1" controls="controls">
10. </video>
11. </br>
12. <input type="button" onclick="addScenes()" value="Add Scenes" />
13. </body>
14. </html>
```

Και σε αυτήν την περίπτωση τα σημεία στα οποία θα εστιάσουμε περισσότερο σημειώνονται με έντονη γραφή. Στην γραμμή 4 εισάγουμε ένα εξωτερικό Javascript αρχείο το οποίο περιέχει μία συνάρτηση η οποία θα καλείται από την εφαρμογή μας. Στην συνέχεια στην σειρά 8 έχουμε ένα element το οποίο λειτουργεί ως διαχωριστικό μέσα στο HTML έγγραφο και προσδιορίζει τον χώρο στον οποίο θα τοποθετηθεί το element <scenes> που θα δημιουργηθεί με δυναμικό τρόπο μέσα στην εφαρμογή. Και τέλος στην γραμμή 11 έχουμε τοποθετήσει, ακριβώς κάτω από το προβαλλόμενο βίντεο, ένα κουμπί το οποίο μόλις πατηθεί θα καλέσει την Javascript function η οποία περιέχεται στο προαναφερθέν εξωτερικό αρχείο. Στο κουμπί θα αναγράφεται η ένδειξη 'Add Scenes'. Μόλις φορτωθεί η ιστοσελίδα έχει την μορφή που φαίνεται στην παρακάτω εικόνα.

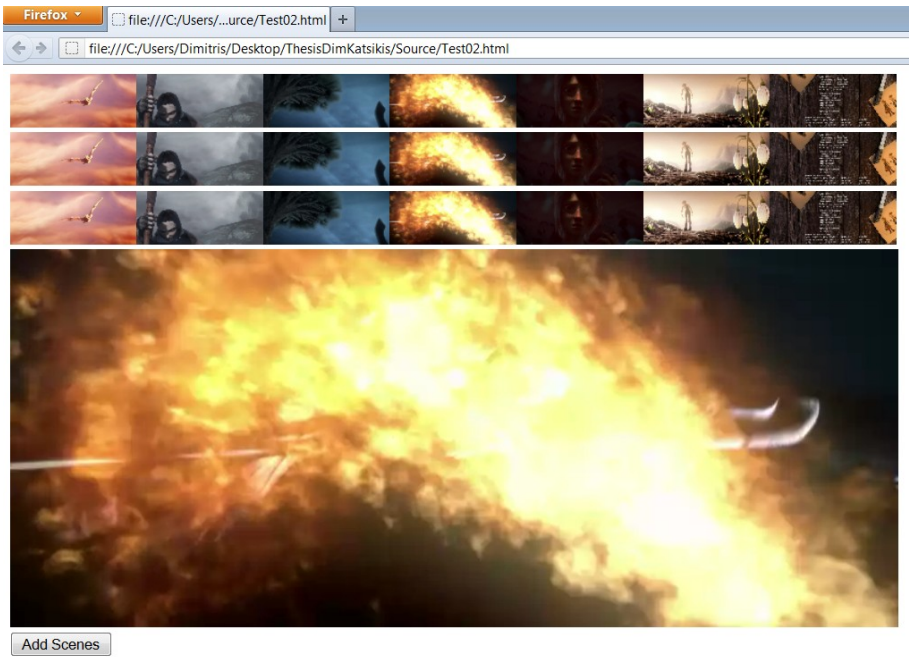
Εικόνα 14: Πριν το πάτημα του κουμπιού

Μόλις πατήσουμε το κουμπί θα προστεθεί ένα element <scenes> πάνω από το προβαλλόμενο βίντεο. Ποια θα είναι η μορφή της ιστοσελίδας αφού πατήσουμε το κουμπί περιγράφεται από την παρακάτω εικόνα.



Εικόνα 15: Μετά το πάτημα του κουμπιού

Ενώ αν ξαναπατήσουμε το κουμπί θα εμφανισθεί και δεύτερη μπάρα ακριβώς από κάτω από την πρώτη κοκ. Οι μπάρες είναι όλες συσχετισμένες με το προβαλλόμενο βίντεο και είναι πλήρως συγχρονισμένες με αυτό κάθε στιγμή.



Εικόνα 16: Μετά από τρία πατήματα του κουμπιού

Τώρα θα εξετάσουμε με αναλυτικό τρόπο τι έγινε μόλις πατήσαμε πάνω στο κουμπί. Μόλις λοιπόν πατήσαμε το κουμπί κλήθηκε η συνάρτηση **addScenes** η οποία βρίσκεται στο εξωτερικό Javascript αρχείο test02.js. Το περιεχόμενο του αρχείου αυτού παρουσιάζεται παρακάτω.

```
1. function addScenes(){
2. var fr = [0, 80, 160, 240, 320, 400, 480, 560, 640, 720, 800, 880];
3. vid = document.getElementById("vid1");
4. sc = new scenes(null, vid, fr, 7);
5. region = document.getElementById("div1");
6. region.appendChild(sc.self);
7. }
```

Η συνάρτηση **addScenes** δημιουργεί ένα <scenes> και το τοποθετεί στην περιοχή της ιστοσελίδας που έχουμε ορίσει στο HTML αρχείο που παρουσιάσαμε παραπάνω. Την λειτουργικότητα του **scenes** μπορούμε να την διαχειριστούμε είτε ως ένα HTML tag είτε ως μία Javascript κλάση. Στην γραμμή 2 αρχικοποιούμε την μεταβλητή fr στην οποία θέτουμε ένα πίνακα ο οποίος περιέχει τις χρονικές στιγμές των επιλεγμένων frames. Στην συνέχεια στην γραμμή 3 τοποθετούμε το <video> element με το προβαλλόμενο βίντεο στην μεταβλητή vid. Κατόπιν στην γραμμή 4 δημιουργούμε ένα νέο αντικείμενο **scenes** και το τοποθετούμε στην μεταβλητή sc. Αυτό το αντικείμενο περιέχει όλη την λειτουργικότητα ενός element <scenes>.

Παρατηρούμε πως ο constructor του **scenes** δέχεται τέσσερα ορίσματα. Στο πρώτο όρισμα δίνουμε την τιμή null. Για την ώρα αρκεί να κρατήσουμε πως όταν διαχειριζόμαστε το **scenes** με δυναμικό τρόπο με Javascript θα θέτουμε το πρώτο όρισμα constructor της **scenes** null. Το δεύτερο όρισμα δέχεται το <video> element με το οποίο θα συσχετιστεί οπότε θέτουμε το vid. Το τρίτο όρισμα

δέχεται έναν πίνακα που περιέχει τις χρονικές στιγμές που εμφανίζονται τα επιλεγμένα frames οπότε βάζουμε το fr. Εδώ πρέπει να προσέξουμε το γεγονός ότι κατά την δημιουργία ενός αντικειμένου scenes με αυτόν το τρόπο δεν μπορούμε να περάσουμε τους αριθμούς της σειράς εμφάνισης των επιλεγμένων frames παρά μόνο τα δευτερόλεπτα στα οποία εμφανίζονται στο βίντεο. Έτσι δεν υπάρχουν ορίσματα που να αντιστοιχούν στα attributes *unit* και *fps* του element <scenes>. Τέλος το τέταρτο όρισμα δέχεται των αριθμό των εικόνων που θα εμφανίζονται.

Για τις default τιμές των επιλεγμένων frames και του αριθμού των εμφανιζόμενων frames ισχύουν τα ίδια με την περίπτωση της πρώτης εφαρμογής. Το συσχετιζόμενο <video> element σε αυτήν την περίπτωση δεν παίρνει τιμή από προεπιλογή αλλά πρέπει, οπωσδήποτε, να ορίσουμε ρητά την τιμή του.

Στην γραμμή 6 τοποθετούμε στο με id = 'div1', δηλαδή στην περιοχή της ιστοσελίδας που έχουμε καθορίσει στον HTML κώδικα το *sc.self* που δεν είναι τίποτα άλλο παρά η γνωστή μπάρα στην οποία εμφανίζονται οι εικόνες που αντιπροσωπεύουν τις διαφορετικές σκηνές του βίντεο. Το *self* στην Javascript ονομάζεται property του αντικειμένου **scenes**. Το *self* είναι προσβάσιμο εξωτερικά δηλαδή έχει συμπεριφορά public πεδίου και στην περίπτωσή μας είναι ένας χώρος στον οποίο είναι τοποθετημένες οι εμφανιζόμενες εικόνες δηλαδή η εμφανιζόμενη μπάρα.

4. ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ ΥΛΟΠΟΙΗΣΗΣ

Στην ενότητα αυτή θα περιγράψουμε με αναλυτικό τρόπο τον κώδικα ο οποίος υλοποιεί την λειτουργικότητα του custom html element <scenes>. Ο κώδικας αυτός περιέχεται στο αρχείο MyLibrary.js και περιλαμβάνει την Javascript κλάση **scenes** και την συνάρτηση **initializeUI**. Παρακάτω παρατίθεται το περιεχόμενο του αρχείου MyLibrary.js. Οι γραμμές του κώδικα είναι αριθμημένες. Ανάμεσα στις γραμμές κώδικα θα γίνεται ο αναλυτικός σχολιασμός και περιγραφή της υλοποίησης ο οποίος θα ξεχωρίζει από τον κώδικα καθώς θα είναι τυπωμένος σε διαφορετική γραμματοσειρά και οι γραμμές του δεν θα είναι αριθμημένες.

```
1.    /*
2.     * This library contains the custom HTML5 tag <scenes>. This tag displays video frames
3.     * which represent the beginning of video scenes.
4.     *
5.     * Developer: Dimitris Katsikis
6.     */
7.
```

Στην γραμμή 11 δημιουργούμε ένα element με το όνομα scenes. Αυτό που επιτυγχάνουμε με αυτή την γραμμή είναι να πληροφορήσουμε το περιηγητή ότι μπορούμε να δημιουργούμε custom tag τύπου <scenes>. Αν παραλείψουμε αυτήν την γραμμή ο περιηγητής δεν μπορεί να αναγνωρίσει το tag <scenes>.

```
8.    /* Here browser be informed that we will be creating a custom tag of type <scenes>.
9.     * Without this statement the browser does not understand the tag <scenes>.
10.    */
11.   document.createElement("scenes");
12.
13.
```

Στα σχόλια που παρατίθενται από την γραμμή 14 έως την γραμμή 29 περιγράφονται τα ορίσματα του constructor scenes(*tag*, *video*, *frames*, *number*) καθώς και τα attributes του custom element <scenes>. Με εξαίρεση το όρισμα tag τα υπόλοιπα τα έχουμε ήδη αναλύσει σε προηγούμενες ενότητες. Στο όρισμα tag θα δέχεται το tag <scenes> το οποίο θα βρίσκεται στο HTML έγγραφο που θα εισάγει το παρόν αρχείο ή αλλιώς που θα χρησιμοποιεί την παρούσα βιβλιοθήκη.

```
14.   /*
15.    * Function (class) scenes provides the implementation of <scenes> functionality.
16.    * Class Parameters:
17.    *          tag - the HTML tag <scenes> from the HTML file
18.    *          video - the relative HTML video
```

```
19.     *          frames - array with seconds represents the beginning of every
    scene
20.     *          number - the number of scenes will be displaying
21.     * Attributes of <scenes>:
22.     *          videoID - the id of relative video element
23.     *          number - the number of scenes will be displaying
24.     *          fps - frames per second
25.     *          unit - unit of frames ("second" or "number")
26.     *          frames - array represents the beginning of every scene in
    seconds if unit is
27.     *                "second" or in integers if unit is "number"
28.     *
29.     */
30.    function scenes(tag, video, frames, number){
```

Στην γραμμή 31 δηλώνουμε την μεταβλητή that και θέτουμε σε αυτήν την this. Αυτό είναι ένα προγραμματιστικό κόλπο που κάνουμε έτσι ώστε να μπορούμε να χρησιμοποιήσουμε την this εντός των εσωτερικών συναρτήσεων της scenes. Αυτό συμβαίνει επειδή στην Javascript το this αναφέρεται στο αντικείμενο που δημιουργείται από την εσωτερική συνάρτηση στην οποία χρησιμοποιείται.

```
31.         var that = this;// a trick to use 'this' inside internal functions
32.
```

Στην γραμμή 34 γίνεται έλεγχος αν έχουμε δώσει όρισμα στο tag. Στην πράξη το tag θα ισούται με null στην περίπτωση δημιουργίας της λειτουργικότητας του <scenes> στο παρασκήνιο με δυναμικό τρόπο μέσα από κώδικα Javascript χρησιμοποιώντας ρητά τον constructor. Ενώ αν δεν είναι ούτε null ούτε undefined αυτό σημαίνει πως στο έγγραφό μας υπάρχει κάποιο element <scenes> το οποίο θα δημιουργηθεί.

```
33.         // if there is a <scenes> tag in the HTML file.
34.         if ((typeof tag != 'undefined') && tag !== null){
```

Ακολουθεί η δήλωση των μεταβλητών που θα δεχθούν τις τιμές των attributes του <scenes>. Εδώ πρέπει να πούμε πως οι μεταβλητές που δηλώνονται με αυτόν τον τρόπο δηλαδή με το προσδιοριστικό var είναι τοπικές μεταβλητές της κλάσης και έχουν την συμπεριφορά private μεταβλητών. Οι μεταβλητές που θα έχουν συμπεριφορά public μεταβλητών θα είναι αυτές που θα έχουν την μορφή that.xxx και θα μπορούμε να έχουμε πρόσβαση σε αυτές εξωτερικά. Υπενθυμίζουμε πως ότι συμβαίνει εντός του παρόντος if αφορά την περίπτωση δημιουργίας του αντικειμένου scenes κατά την φόρτωση μιας ιστοσελίδας που περιέχει κάποιο element <scenes>.

```
35.             // these variables and the variable 'number' represent the
    scenes' attributes
36.             var videoID, fps, unit, frameNumbers, frameSeconds;
37.
```

Στην συνέχεια ορίζουμε την μεταβλητή self η οποία έχει public συμπεριφορά και σε αυτήν θα θέσουμε την μεταβλητή tag η οποία περιέχει το <scenes> το οποίο βρίσκεται στην ιστοσελίδα. Στην παρούσα υλοποίηση η self είναι η μόνη μεταβλητή που έχει public συμπεριφορά.

```
38.             that.self = tag; // scenes.self will be the displaying tag
```

Στις επόμενες γραμμές γίνεται ο έλεγχος του attribute *videoID*. Αν είναι ορισμένο τότε η τιμή του τοποθετείται στο *videoID* στην γραμμή 40 και στην 41 λαμβάνεται το συσχετιζόμενο video element με το id = *videoID* και τοποθετείται στην μεταβλητή video. Στην περίπτωση που δεν ορίζεται το attribute *videoID* τότε θα επιλεγεί το συσχετιζόμενο <video element> με προκαθορισμένο τρόπο. Ο κώδικας που υλοποιεί την συγκεκριμένη συμπεριφορά βρίσκεται στις γραμμές 43 έως 51. Αναλυτικά, στην γραμμή 43 λαμβάνονται όλα τα elements που περιέχονται στην ιστοσελίδα (ή στο τμήμα της ιστοσελίδας που έχουμε επιλέξει). Κατόπιν μέσα στο for εξετάζονται ένα ένα τα elements αυτά. Όταν φτάσουμε στο παρόν element scenes, κάτι που ελέγχεται από την συνθήκη στην γραμμή 46, τότε αρχίζουμε να ψάχνουμε τα element που ακολουθούν. Το πρώτο <video> element που θα συναντήσουμε είναι αυτό με το οποίο θα συσχετιστεί το παρόν scenes. Ο έλεγχος αυτός γίνεται στην γραμμή 49 και η τοποθέτηση του προκαθορισμένου συσχετιζόμενου video γίνεται στην γραμμή 50.

```
39.             if ((typeof tag.attributes.videoID != 'undefined') &&
tag.attributes.videoID !== null){
40.                     videoID = tag.attributes.videoID.value;
41.                     video = document.getElementById(videoID);
42.             }else{
43.                     var tags = document.getElementsByTagName("*");
44.                     outer:
45.                     for(var  i=0; i<tags.length;  i++){
46.                             if(tag === tags.item(i)){
47.                                     for(i; i<tags.length;  i++){
48.                                             var tagName = tags.item(i).nodeName;
49.                                             if(tagName.toLowerCase()=='video'){
50.                                                     video = tags.item(i); // default
value
51.                                                     break outer;
52.                                             }
53.                                     }
54.                             }
55.                     }
56.             }
57.
```

Από την γραμμή 58 έως την γραμμή 62 ελέγχεται το attribute *number* και αν δεν είναι ορισμένο παίρνει την προκαθορισμένη τιμή. Κάτι παρόμοιο συμβαίνει και στις γραμμές 64 έως 68 για το attribute *fps*. Παρόμοια για το *unit* στις γραμμές 70 έως 74.

```
58.             if ((typeof tag.attributes.number != 'undefined') &&
tag.attributes.number !== null){
```

```
59.                     number = parseInt(tag.attributes.number.value,10);
60.               }else{
61.                     number = 5; // default value
62.               }
63.
64.               if    ((typeof    tag.attributes.fps    !=    'undefined')    &&
   tag.attributes.fps !== null){
65.                     fps = parseFloat(tag.attributes.fps.value);
66.               }else{
67.                     fps = 29.97; // default value
68.               }
69.
70.               if    ((typeof    tag.attributes.unit    !=    'undefined')    &&
   tag.attributes.unit !== null){
71.                     unit = tag.attributes.unit.value;
72.               }else{
73.                     unit = "second"; // default value
74.               }
75.
```

Στις επόμενες γραμμές αναγνωρίζεται αν τα επιλεγμένα frames προσδιορίζονται από τον αριθμό της σειράς εμφάνισής τους στο βίντεο ή από την χρονική στιγμή στην οποία εμφανίζονται στο βίντεο σε δευτερόλεπτα. Αν η τιμή του *unit* είναι 'number' τότε τοποθετούνται στον πίνακα frameNumbers αλλιώς τοποθετούνται στο πίνακα frameSeconds. Για την μετατροπή του string που περιέχει το attribute frames σε ξεχωριστούς αριθμούς που θα τοποθετηθούν στον αντίστοιχο πίνακα χρησιμοποιείται η split η οποία παίρνει ως όρισμα τον τύπο του διαχωριστικού που στην περίπτωσή μας είναι το space .

```
76.               if    ((typeof    tag.attributes.frames    !=    'undefined')    &&
   tag.attributes.frames !== null){
77.                     if (unit == "number"){
78.                           frameNumbers  =  tag.attributes.frames.value.split("
   "); // frames in integers
79.                     }else{
80.                           frameSeconds  =  tag.attributes.frames.value.split("
   "); // frames in seconds
81.
         }
```

Στην περίπτωση που δεν ορίζεται το attribute *frames* θα πρέπει τα επιλεγμένα frames να ορισθούν με προκαθορισμένο τρόπο. Αυτός ο προκαθορισμένος τρόπος περιγράφεται στις γραμμές 82 μέχρι 95. Το σκεπτικό είναι τα επιλεγμένα frames να είναι διπλάσια σε αριθμό από τις εικόνες που θα εμφανίζονται στην μπάρα και να είναι ισοκατανεμημένα μέσα στο βίντεο. Στην γραμμή 82 καθορίζουμε το χρονικό διάστημα που θα απέχει το ένα frame από το άλλο. Στις επόμενες δύο γραμμές ορίζουμε δύο πίνακες έναν για κάθε τιμή του *unit*. Στην συνέχεια δίνουμε τιμές σε κάθε πίνακα που ξεκινούν από το μηδέν και αυξάνονται κατά *step* μέχρι να δημιουργηθεί ένας πίνακας με 2**number* στοιχεία.

```
82.                 }else{
83.                       var step = video.duration/(2*number);
84.                       frameNumbers = new Array();
85.                       frameSeconds = new Array();
86.                       if (unit == "number"){
87.                             for (var i=0 ; i<2*number ; i++){
88.                                   frameNumbers[i]  =  i*step*fps;  //  default
   values
89.                             }
90.                       }else{
91.                             for (var i=0 ; i<2*number ; i++){
92.                                   frameSeconds[i] = i*step; // default values
93.                             }
94.                       }
95.                 }
```

Στις γραμμές από 96 έως 106 θα γίνει η εσωτερική μετατροπή του πίνακα που καθορίζει τα frames που θα επιλεγούν έτσι ώστε από εδώ και στο εξής τα επιλεγμένα frames να καθορίζονται μόνον από τη χρονική στιγμή στην οποία εμφανίζονται στο βίντεο. Αυτό συμβαίνει γιατί οι μέθοδοι που διαθέτει το <video> element υποστηρίζουν μόνο αυτόν τον τρόπο για να διαχειριστούμε τα frames.

```
96.
97.                 frames = new Array(); // frames in seconds
98.                 if(unit == "number"){
99.                       for (var i=0 ; i<frameNumbers.length ; i++){
100.                            frames[i] = frameNumbers[i]/fps;
101.                      }
102.                }else{
103.                      for (var i=0 ; i<frameSeconds.length ; i++){
104.                            frames[i] = frameSeconds[i];
105.                      }
106.                }
107.
```

Οι γραμμές από 108 έως 123 θα τρέξουν μόνο στην περίπτωση που το αντικείμενο **scenes** δημιουργηθεί με δυναμικό τρόπο αφού φορτωθεί η ιστοσελίδα με ρητή επίκληση του constructor. Εκεί που πρέπει να σταθούμε είναι η γραμμή 109 στην οποία στην public μεταβλητή self τοποθετούμε ένα element το οποίο παριστάνει τον χώρο στον οποίο θα τοποθετηθούν τα επιλεγμένα frames. Στο αυτό θα ενσωματωθεί όλη η λειτουργικότητα του **scenes**. Στην ουσία το *self* σε κάθε περίπτωση περιέχει όλη την λειτουργικότητα του **scenes**, είναι το ίδιο το <scenes>.

```
108.          }else{ // case of creation scenes tag in the background
109.                that.self = document.createElement("div");
```

```
110.
111.              if (number == null){
112.                    number = 5; // default value
113.              }
114.
115.              if (frames == null){
116.                    frames = new Array();
117.                    var step = video.duration/(2*number);
118.                    for (var i=0 ; i<2*number ; i++){
119.                          frames[i] = i*step*fps; // default values
120.                    }
121.              }
122.
123.        }
124.
125.
```

Η μεταβλητή screens είναι ο πίνακας στον οποίο θα τοποθετηθούν τα επιλεγμένα frames. Στην μεταβλητή *hiddenVideo* τοποθετείται ένα <video> element το οποίο αναφέρεται στην ίδια πηγή με το προβαλλόμενο βίντεο και θα χρησιμοποιηθεί στο παρασκήνιο. Την ώρα που θα προβάλλεται το video από το *hiddenvideo* θα τραβάμε τα επιλεγμένα frames για να τα τοποθετήσουμε στο screens.

```
126.        var screens = new Array(frames.length); // stores frames which
    represents scenes
127.        var hiddenVideo = createVideo(video.src); // video in the background
128.
```

Εδώ προσθέτουμε ένα listener στο *hiddenVideo*. Κατά την φόρτωση του βίντεο θα τρέξει μία φορά η συνάρτηση **makeScenes**. Με την **makeScenes** θα ασχοληθούμε αναλυτικότερα παρακάτω. Για την ώρα αρκεί να πούμε πως η **makeScenes** θα αναλάβει να γεμίσει τον πίνακα screens με τα κατάλληλα frames.

```
129.        // runs one time
130.        hiddenVideo.addEventListener("loadeddata", function(evt){
131.              makeScenes(hiddenVideo, video, frames, number, screens);
132.              }, false);
133.
```

Εδώ προσθέτουμε ένα listener στο video. Η συνάρτηση **displayScenes** θα τρέχει κάθε φορά που θα συμβαίνει ένα συμβάν *timeupdate*. Γενικά έχουμε ένα *timeupdate* συμβάν σε ένα βίντεο όταν έχουμε μια οποιοδήποτε αλλαγή σε αυτό το **video** για παράδειγμα καθώς αλλάζουν τα προβαλλόμενα frames από το **video** έχουμε *timeupdate* συμβάντα ή όταν πατάμε το play ή το pause. Για την **displayScenes** θα αναφερθούμε αναλυτικότερα παρακάτω. Για την ώρα αρκεί να πούμε πως η **displayingScenes** είναι υπεύθυνη για τον τρόπο με τον οποίο θα εμφανίζονται κάθε φορά στην μπάρα τα κατάλληλα frames με την κατάλληλη σειρά.

```
134.        // runs many times
135.        video.addEventListener("timeupdate", function(evt){
136.              displayScenes(video, frames, number, screens);
137.              }, false);
138.
```

Η συνάρτηση **createVideo** χρησιμοποιείται για την δημιουργία **video** element. Δέχεται ως όρισμα την URL του βίντεο και επιστρέφει το **video** element. Χρησιμοποιήθηκε παραπάνω στην γραμμή 127 κατά την δημιουργία του *hiddenVideo*.

```
139.        function createVideo(videoURL){
140.            var video = document.createElement("video");
141.            video.setAttribute("src", videoURL);
142.            //video.setAttribute("preload", "metadata");
143.            return video;
144.        }
145.
```

Η συνάρτηση **capture** αποσπά το τρέχον frame από ένα βίντεο και το επιστρέφει ως **canvas** element. Δέχεται ως ορίσματα το **video** element από το οποίο θα αποσπάσει το frame και έναν συντελεστή *scaleFactor* ο οποίος προσδιορίζει ποιο θα είναι το μέγεθος του επιστρεφόμενου **canvas** σε σχέση με το αρχικό frame. Όταν ο συντελεστής αυτός ισούται με την μονάδα τότε το **canvas** που προκύπτει έχει μέγεθος ίσο με το αρχικό frame, δηλαδή ίσο με τις διαστάσεις του **video**. Αν για παράδειγμα ο συντελεστής *scaleFactor* έχει τιμή 0,5 αυτό σημαίνει πως η κάθε διάσταση του επιστρεφόμενου **canvas** θα είναι η μισή σε σχέση με τις αντίστοιχες διαστάσεις του **video**.

```
146.        /*
147.         * Function 'capture' captures the current frame from a video and
    returns it as a canvas
148.         * element.
149.         * Arguments:
150.         *          video - the relative video
151.         *          scalefactor - specify the size of the captured frame in
    relation with its
152.         *original size
153.         */
154.        function capture(video, scaleFactor) {
```

Στην γραμμές 155 έως 157 παρατηρούμε πως σε περίπτωση που το *scaleFactor* δεν οριστεί τότε παίρνει προκαθορισμένη τιμή 1. Στις γραμμές 158 έως 162 καθορίζονται οι διαστάσεις που θα έχει το επιστρεφόμενο canvas και θα είναι οι διαστάσεις του video πολλαπλασιασμένες με τον συντελεστή *scaleFactor*. Στην γραμμή 160 δημιουργούμε ένα element **canvas**. Στην γραμμή 163 καθορίζεται πως το **canvas** θα είναι δύο διαστάσεων. Ενώ στην 164 είναι η στιγμή στην οποία αποσπάμε το τρέχον frame του **video** και το τοποθετούμε ως περιεχόμενο στο **canvas**. Και τέλος στην γραμμή 165 επιστρέφουμε το **canvas** που δημιουργήθηκε παραπάνω.

```
155.              if(scaleFactor == null){
156.                    scaleFactor = 1; // default value
157.              }
158.              var w = video.videoWidth * scaleFactor;
159.              var h = video.videoHeight * scaleFactor;
160.              var canvas = document.createElement('canvas');
161.                    canvas.width  = w;
162.                  canvas.height = h;
163.              var ctx = canvas.getContext('2d');
164.                    ctx.drawImage(video, 0, 0, w, h);
165.            return canvas;
166.        }
167.
```

Ακολουθεί η υλοποίηση της συνάρτησης **makeScenes**. Η συνάρτηση **makeScenes** μαζί με την **displayScenes** που θα δούμε παρακάτω αποτελούν την καρδιά της κλάσης **scenes**. Αυτό γιατί υλοποιούν το μεγαλύτερο μέρος της λειτουργικότητας της **scenes**, την δημιουργία των frames και την παρουσίασή τους. Η **makeScenes** παίρνει πέντε ορίσματα. Το *hiddenVideo* είναι το **video** το οποίο δημιουργήθηκε παραπάνω και μένει στο παρασκήνιο. Το *video* είναι το προβαλλόμενο **video**. Το *frames* είναι ο πίνακας που περιέχει τις χρονικές στιγμές των frames που θα επιλεγούν. Το *number* ο αριθμός των εικόνων που θα εμφανίζονται στην μπάρα. Και τέλος το *screens* είναι ο πίνακας στον οποίο θα τοποθετηθούν τα frames τα οποία θα εμφανίζονται ως εικόνες στην μπάρα.

```
168.        /*
169.         * Function 'makeScenes' fills an array with the selected frames and
    return it.
170.         * Arguments:
171.         *          hiddenVideo - the background video
172.         *          video - the relative video
173.         *          frames - the array with seconds represents the beginning
    of every scene
174.         *          number - the number of scenes will be displaying
175.         *          screens - the array will store frames represent scenes
176.         */
```

Στην γραμμή 178 δίνουμε την κατάλληλη τιμή στον συντελεστή *scaleFactor* που θα χρησιμοποιηθεί κατά την κλήση της συνάρτησης **capture** παρακάτω. Η τιμή αυτού του συντελεστή ορίζεται ως 1/*number* έτσι ώστε όλες οι εικόνες που θα εμφανίζονται μαζί στην μπάρα να έχουν μήκος ίδιο με το μήκος του video για λόγους αισθητικής. Κατόπιν στην γραμμή 179 τοποθετούμε στην μεταβλητή *duration* την χρονική διάρκεια του *hiddenVideo* καθώς με αυτό κυρίως θα δουλέψουμε παρακάτω σε αυτήν την συνάρτηση. Οι γραμμές 180 και 181 τοποθετούν το *hiddenVideo* στην αρχή του. Στην 182 τοποθετούμε στο *hiddenVideo* ένα listener το οποίο κάθε φορά που έχουμε ένα συμβάν *seeked* στο *hiddenVideo* καλεί την **seekedCallBack** που ορίζεται ακριβώς από κάτω. Ένα συμβάν *seeked* λαμβάνει χώρα κάθε φορά που το video πηγαίνει σε μια συγκεκριμένη χρονική στιγμή που ψάχνουμε. Συνήθως συμβαίνει μετά από

αλλαγή στο *currentTime* του video που ορίζει την τρέχουσα χρονική στιγμή στη οποία βρίσκεται το βίντεο. Για παράδειγμα όταν θέσω *video.currentTime* = 50 το βίντεο θα πάει στην στιγμή 50 sec. Μέχρι να πάει στην στιγμή 50 είναι σε κατάσταση *seeking* δηλαδή ψάχνει, μόλις φτάσει στην στιγμή 50 λαμβάνει χώρα το συμβάν *seeked* που σημαίνει πως η χρονική στιγμή βρέθηκε.

```
177.        function makeScenes(hiddenVideo, video, frames, number, screens){
178.              var scaleFactor = 1/number;
179.              var duration = hiddenVideo.duration;
180.              var i = 0;
181.              hiddenVideo.currentTime = frames[i];
182.              hiddenVideo.addEventListener("seeked", seekedCallBack, false);
```

Ακολουθεί η **seekedCallback** η οποία, όπως εξηγήσαμε παραπάνω, εκτελείται κάθε φορά που έχουμε ένα συμβάν *seeked*. Η υλοποίηση της **seekedCallBack** γίνεται στις γραμμές 183 έως 203. Στην γραμμή 184 συλλαμβάνεται το πρώτο επιλεγμένο frame (αντικείμενο τύπου canvas), που αντιστοιχεί στην αρχή της πρώτης σκηνής, και τοποθετείται στην θέση i του πίνακα *screens*. Στην συνέχεια στην γραμμή 185 προστίθεται ένα νέο πεδίο στο αντικείμενο το οποίο περιέχεται στην θέση i του πίνακα *screens*. Η Javascript δίνει την δυνατότητα να προσθέτουμε δυναμικά νέα πεδία κατευθείαν σε ένα αντικείμενο μετά την δημιουργία του. Το πεδίο αυτό είναι το *time* και ο ρόλος του είναι να προσδιορίζει την χρονική στιγμή στη οποία εφμανίζεται το συγκεκριμένο frame. Στην συνέχεια, στην γραμμή 188, προσθέτουμε με παρόμοιο τρόπο στο ίδιο αντικείμενο **canvas** έναν listener. Σύμφωνα με αυτόν το listener κάθε φορά που θα έχουμε ένα συμβάν *click* στο αντικείμενο αυτό τότε θα καλείται η συνάρτηση **sceneCallBack**, η οποία υλοποιείται ακριβώς παρακάτω. Η λειτουργιά αυτής της συνάρτησης είναι να πηγαίνει το συσχετιζόμενο video στην χρονική στιγμή που περιέχεται στην μεταβλητή time του αντικειμένου στο οποίο είχαμε το συμβάν click. Με πιο απλά λόγια με αυτές τις γραμμές κώδικα πετυχαίνουμε κάθε φορά που κάνουμε κλικ με το ποντίκι πάνω σε μια εικόνα που εμφανίζεται στην μπάρα το βίντεο να μεταφέρεται στην στιγμή που το αντίστοιχο frame προβάλλεται.

```
183.              function seekedCallBack(evt){ //capture a frame and search for
    the next
184.                    screens[i] = capture(hiddenVideo, scaleFactor);
185.                    screens[i].time    =    hiddenVideo.currentTime;    //add
    attribute 'time' to every frame
186.
187.                    //add an eventListener to every frame (canvas element)
    represents a scene
188.
       screens[i].addEventListener("click",sceneCallBack,false);
189.                    function sceneCallBack(){ // the video goes to the frame
    time
190.                          video.currentTime = this.time;
191.                    }
```

Εδώ ελέγχεται αν ο πίνακας πίνακας *screens* έχει γεμίσει οπότε το *hiddenVideo* σταματά. Και κατόπιν στην γραμμή 196 απομακρύνεται ο listener ο οποίος καλεί την **seekedCallBack** καθώς έχει ολοκληρώσει το έργο του. Στην περίπτωση που ο πίνακας *screens* δεν έχει γεμίσει, δηλαδή

όταν δεν έχει ολοκληρωθεί η συλλογή των επιλεγμένων frames, ψάχνουμε το επόμενο frame. Μόλις το frame αυτό βρεθεί θα πυροδοτήσει ένα συμβάν *seeked*, η seekedCallBack εκτελείται και αυτό γίνεται συνεχώς μέχρι να γεμίσει ο πίνακας *screens* και επομένως η συνθήκη if στην 192 να ικανοποιηθεί.

```
192.                    if(i >= frames.length-1){
193.                          //when array 'screens' be filled the video pauses and the listener removed
194.                          hiddenVideo.pause();
195.                          hiddenVideo.currentTime = 0;
196.                    hiddenVideo.removeEventListener("seeked", seekedCallBack, false);
197.                    //alert("completed");
198.              }
199.                    else{
200.                    hiddenVideo.currentTime = frames[i+1]; // search for the next frame
201.              i++;
202.                    }
203.              }
204.        }
```

Παρακάτω ακολουθεί η υλοποίηση της συνάρτησης **displayScenes** η οποία όπως είδαμε παραπάνω εκτελείται κάθε φορά που συμβαίνει ένα συμβάν *timeupdate* στο συσχετιζόμενο **video**. Η συνάρτηση αυτή καθορίζει ποιο από τα frames που περιέχει ο πίνακας *screens* που γεμίσαμε στην συνάρτηση **makeScenes** θα εμφανίζεται στην μπάρα και σε ποια θέση θα εμφανίζεται. Στην κεντρική θέση θα εμφανίζεται το επιλεγμένο frame που μόλις προβλήθηκε, δηλαδή το frame που αντιπροσωπεύει την αρχή της τρέχουσας σκηνής. Στα δεξιά προβάλλονται τα επιλεγμένα frames που προβάλλονται μετά το frame που εμφανίζεται στο κέντρο και στα αριστερά τα επιλεγμένα frame που προβάλλονται πριν από το frame που εμφανίζεται στο κέντρο. Η **displayScenes** δέχεται 4 ορίσματα, το *video* παίρνει το συσχετιζόμενο **video**, το frames παίρνει τον πίνακα με τις χρονικές στιγμές σε δευτερόλεπτα που εμφανίζονται τα frames που έχουν επιλεγεί να αντιπροσωπεύουν την αρχή διαφορετικών σκηνών. Το *number* το οποίο παίρνει τον αριθμό των frame που θα εμφανίζονται στην μπάρα. Και τέλος το *screens* το οποίο δέχεται τον ίδιο πίνακα με αυτόν που γεμίζεται στην **makeScenes**. Εδώ πρέπει να σημειώσουμε πως όταν καλείται η **displayScenes** ο πίνακας *screens* μπορεί να μην έχει συμπληρωθεί, σε αυτήν την περίπτωση η **displayScenes** εκτελείται κανονικά και για τα στοιχεία του *screens* που δεν έχουν ακόμη συμπληρωθεί εμφανίζει προκαθορισμένη συμπεριφορά.

```
205.
206.        /*
207.         * Function 'displayScenes' display the appropriate frames of the video. The central
208.         * frame represents the current scene. The left frames represent the previous scenes
209.         * and the right frames represent the next scenes.
210.         * Arguments:
211.         *          video - the relative video
```

```
212.         *          frames - the array with seconds represents the beginning
   of every scene
213.         *          number - the number of scenes will be displaying
214.         *          screens - the array stores frames represent scenes
215.         */
```

Στην γραμμή 217 δηλώνεται η μεταβλητή d. Το d είναι ο αριθμός των frames που θα εμφανίζονται δεξιά ή αριστερά του κεντρικού frame. Για παράδειγμα αν ο συνολικός αριθμός των frames που θα εμφανίζονται είναι 5 (*number* = 5) τότε ένα frame θα είναι στο κέντρο, δύο στα αριστερά και δύο στα δεξιά δηλαδή d = 2. Ο αριθμός αυτός προκύπτει αν διαιρέσουμε το *number* δια του δύο και κρατήσουμε το ακέραιο μέρος. Στην σειρά 218 δηλώνεται η μεταβλητή k, όπου k θα είναι η θέση του πίνακα *screens* στην οποία θα βρίσκεται το frame που θα τοποθετηθεί στην αρχή της μπάρας. Στην συνέχεια από την γραμμή 220 έως την γραμμή 234 γίνεται ο υπολογισμός του k. Ο υπολογισμός του k είναι ένα από τα σημαντικότερα σημεία της υλοποίησης καθώς αυτό είναι που καθορίζει την σειρά με την οποία θα προβάλλονται τα frames στην μπάρα. Εδώ μπορούμε να αναφέρουμε πως έχει εισαχθεί επιπλέον πολυπλοκότητα από το γεγονός πως ο αριθμός των προβαλλόμενων στην μπάρα εικόνων είναι παραμετροποιήσιμος. Πρώτα θα εξετάσουμε τις γραμμές 227 έως 234 και στην συνέχεια τις γραμμές 220 έως 226. Η συνθήκη στην 228 ικανοποιείται όταν το video βρίσκεται σε χρονική στιγμή που βρίσκεται ανάμεσα στην στιγμή που προβάλλεται το frame i και το frame i + 1. Στην γραμμή 230 περιγράφεται η περίπτωση που το βίντεο βρίσκεται σε μία χρονική περίοδο μετά την προβολή του τελευταίου επιλεγμένου frame. Σε αυτήν την περίπτωση το τελευταίο επιλεγμένο frame μένει στο κέντρο μέχρι το τέλος του βίντεο. Τώρα πάμε στον κώδικα στις γραμμές από 220 έως 226. Όταν βρισκόμαστε στην αρχή του βίντεο και το πρώτο επιλεγμένο frame βρίσκεται στο κέντρο της μπάρας στα αριστερά του επειδή δεν υπάρχουν προηγούμενα frames εμφανίζονται τα τελευταία επιλεγμένα frames. Αυτή η περίπτωση περιγράφεται στην γραμμή 220. Μερικές φορές η αρχή της πρώτης σκηνής μπορεί να μην συμπίπτει με την αρχή της ταινίας. Δηλαδή μπορεί να υπάρχει χρόνος μεταξύ της έναρξης του βίντεο και του πρώτου επιλεγμένου frame. Σε αυτή την περίπτωση έχουμε επιλέξει να εμφανίζεται στο κέντρο της μπάρας το πρώτο επιλεγμένο frame και ας μην έχει προβληθεί ακόμα. Για να το πετύχουμε αυτό πρέπει να κάνουμε μία διόρθωση η οποία περιγράφεται στις γραμμές 223 έως 225.

```
216.        function displayScenes(video, frames, number, screens){
217.              var d = Math.floor(number/2); // the number of left (or right)
   displayed frames
218.              var k; // screens[k] == position[0]
219.              for (var i = 0 ; i < screens.length ; i++){ // k calculation
220.                    if (i < d){ // case of displaying last frames at the
   beginning
221.                          if(video.currentTime >= frames[i] &&
   video.currentTime < frames[i+1]){
222.                                k = screens.length - d + i;
223.                          }else if (video.currentTime >= 0 &&
   video.currentTime < frames[0]){
224.                                k = screens.length - d;
225.                          }
226.                    }
227.                    else{ // case of other frames
228.                          if(video.currentTime >= frames[i] &&
```

```
   video.currentTime < frames[i+1]){
229.                                k = i - d;
230.                            }else        if        (video.currentTime        >=
   frames[frames.length - 1] && video.currentTime <= video.duration){
231.                                k = frames.length - 1 - d;
232.                            }
233.                    }
234.                }
```

Στην γραμμή 235 καθαρίζει η μπάρα. Κάθε φορά που καλείται η **displyScenes** το εσωτερικό της του <scenes> καθαρίζει από τις προηγούμενες εικόνες και ξαναγεμίζει με τις κατάλληλες εικόνες έτσι ώστε να το **scenes** να είναι πάντα συγχρονισμένο με το συσχετιζόμενο βίντεο. Αυτό όμως συμβαίνει στιγμιαία και δεν γίνεται αντιληπτό με το μάτι. Δηλαδή με το μάτι δεν προλαβαίνουμε να δούμε την μπάρα άδεια, αλλά την βλέπουμε να αλλάζει στιγμιαία έτσι ώστε να ακολουθεί το προβαλλόμενο βίντεο. Στην 236 ορίζεται ο πίνακας *position* ο οποίος θα περιέχει το frame που εμφανίζονται στην μπάρα. Κάθε θέση αυτού του πίνακα είναι και θέση στην εμφανιζόμενη μπάρα. Η μεταβλητή m στην σειρά 238 περιέχει την θέση του frame στον πίνακα *screens* που θα τοποθετηθεί στον πίνακα *position* στην θέση i. Στις γραμμές 240 έως 250 περιγράφεται τι θα συμβεί στην περίπτωση που το *screens*[m] είναι κενό, δηλαδή δεν έχει ακόμη συλληφθεί το συγκεκριμένο frame από το **video**. Τότε δημιουργείται ένα γκρίζο frame με τις διαστάσεις που θα πρέπει να είχε το frame που λείπει και τοποθετείται στην θέση του. Στην περίπτωση που το frame υπάρχει κανονικά τοποθετείται αυτό στον πίνακα *position*. Τέλος στην 254 το frame τοποθετείται στην μπάρα.

```
235.                that.self.innerHTML=""; // clear tag before update
236.              var position = new Array(number);
237.                for (var i = 0 ; i < number ; i++ ){ // placement of frames
238.                      var m; // will be position[i] = screens[m]
239.                      m = (i+k)%screens.length;
240.                      if(screens[m] == null){ // until proper frame be loaded
   display a default frame
241.                            var scaleFactor = 1/number;
242.                            var c = document.createElement('canvas');
243.                            c.width = scaleFactor * video.videoWidth;
244.                            c.height = scaleFactor * video.videoHeight;
245.                            var ctx=c.getContext("2d");
246.                            ctx.fillStyle="#222222";
247.                            ctx.fillRect(0,0,c.width,c.height);
248.                            ctx.fillStyle="#CCCCCC";
249.                            position[i] = c;
250.                      }
251.                      else{
252.                            position[i] = screens[m];
253.                      }
254.                      that.self.appendChild(position[i]); // add frames inside
   <scenes>
```

```
255.                }
256.          }
257.    }
258.
```

Η **intializeUI** σαρώνει το HTML έγγραφο και ψάχνει για custom tags, στην περίπτωσή μας ψάχνει για <scenes>. Η κλήση της συνίσταται να γίνεται κατά την φόρτωση του <body> της ιστοσελίδας. Στην γραμμή 265 όλα τα element του HTML εγγράφου αποθηκεύονται στον πίνακα *tags*. Στην συνέχεια εξετάζονται ένα ένα τα element που περιέχει ο πίνακας *tags* και αν κάποιο έχει όνομα scenes, δημιουργείται ένα object **scenes** το οποίο παίρνει ως όρισμα το *tag* με το όνομα scenes. Με αυτόν τον τρόπο το παρόν έγγραφο λειτουργεί ως βιβλιοθήκη custom tag. Η **initializeUI** κατάλληλα τροποποιημένη μπορεί να χρησιμοποιηθεί και για αλλά custom tags.

```
259.
260.    /*
261.      * Scan the HTML for custom tags and create an instance of the
    corresponding object.
262.      * Function 'initializeUI' can be invoked on body load event.
263.      */
264.    function initializeUI(){
265.          var tags = document.getElementsByTagName("*");
266.          /*
267.            * Iterate through the array and check for your custom tag. If
    found, instantiate the
268.           * corresponding class.
269.           */
270.          for(var  i=0; i<tags.length;  i++){
271.
272.                var tagName = tags.item(i).nodeName;
273.                var obj = null;
274.                if(tagName.toLowerCase()=='scenes'){
275.                      obj = new scenes(tags.item(i));
276.                }
277.          }
278.    }
```

- Την συνάρτηση **capture** την αποσπάσαμε από τον κώδικα που βρίσκεται δημοσιοποιημένος, υπό τον τίτλο 'Using HTML5 Canvas to capture frames from a video',στην ηλεκτρονική διεύθυνση:

http://appcropolis.com/blog/using-html5-canvas-to-capture-frames-from-a-video/ [11]

- Την συνάρτηση **initializeUI** την αποσπάσαμε από τον κώδικά που βρίσκεται δημοσιοποιημένος στην ηλεκτρονική διεύθυνση:

http://cherianajay.hubpages.com/hub/extented-HTML-tags [14]

5. ΠΡΟΤΑΣΕΙΣ ΓΙΑ ΜΕΛΛΟΝΤΙΚΗ ΕΠΕΚΤΑΣΗ

Στην ενότητα αυτή θα παρουσιαστούν προτάσεις για βελτίωση και επέκταση της λειτουργικότητας που παρέχεται μέσω του αντικειμένου **scenes** και θα προταθούν περιοχές στις οποίες θα μπορούσε αυτή η λειτουργικότητα να έχει πρακτική εφαρμογή.

- Κατά την υλοποίηση του κώδικα, λάβαμε υπόψη μας την περίπτωση του progressive download video, δηλαδή τον τρόπο διανομής video που χρησιμοποιείται από το YouTube. Αυτός ο τρόπος διανομής video είναι ο πιο συνηθισμένος. Κύριο χαρακτηριστικό του progressive download video είναι πως μπορείς να ψάχνεις σε οποιοδήποτε σημείο του video. Ωστόσο υπάρχει και η περίπτωση του streaming video η οποία έχει μεγάλο ενδιαφέρον και πολλές χρήσεις. Με αυτήν την περίπτωση δεν ασχοληθήκαμε ιδιαίτερα στα πλαίσια της παρούσας εργασίας. Θα είχε μεγάλο ενδιαφέρον η τροποποίηση του κώδικα υλοποίησης του **scenes** έτσι ώστε να προκύψει ένα αντικείμενο με παρόμοια λειτουργικότητα που να μπορεί να ανταποκριθεί στις απαιτήσεις του streaming video.
- Δύο είναι οι βασικές λειτουργίες του αντικειμένου **scenes**. Η μία είναι η δημιουργία των εικόνων που αντιπροσωπεύουν τις διαφορετικές σκηνές η οποία υλοποιείται μέσω της μεθόδου **makeScenes** και η άλλη η εμφάνιση των κατάλληλων εικόνων με την κατάλληλη σειρά σε συγχρονισμό με το συσχετιζόμενο βίντεο η οποία υλοποιείται μέσω της μεθόδου **displayScenes**. Η **makeScenes** είναι αργή σε σχέση με την **displayScenes** και η ταχύτητά της εξαρτάται από τον ταχύτητα με την οποία κατεβαίνει το βίντεο και εκτελείται μία φορά σε αντίθεση με την **displayScenes** που εκτελείται πολλές φορές. Συγκεκριμένα η **makeScenes** αναφέρεται σε ένα αντίγραφο **video** που βρίσκεται στο παρασκήνιο το οποίο το χρησιμοποιεί για να ψάξει το κατάλληλο frame που αντιπροσωπεύει μια σκηνή, να το αποσπάσει από το βίντεο και από εκεί να προκύψει ο πίνακας *screens* με τις εικόνες που αντιπροσωπεύουν τις διαφορετικές σκηνές του βίντεο. Το **scenes** θα ήταν πολύ πιο γρήγορο αν αντί να παίρνει ως είσοδο το δευτερόλεπτα των frames έπαιρνε τα ίδια τα frames. Αυτό είναι δυνατόν να συμβεί αν στον server έχουμε ήδη δημιουργήσει μέσω μιας μεθόδου με παρόμοια λειτουργικότητα με την **makeScenes** το σύνολο των εικόνων που αντιπροσωπεύουν τις σκηνές και το έχουμε αποθηκεύσει σε μια βάση δεδομένων.
- Στην παρούσα εργασία εντάξαμε όλη την λειτουργικότητα του **scenes** σε ένα HTML tag έτσι ώστε να μπορεί να χρησιμοποιηθεί ανεξάρτητα στην ανάπτυξη διαδικτυακών εφαρμογών. Η λειτουργικότητα αυτή θα μπορούσε να ενταχθεί και σε έναν media player με βάση το HTML5 και με αυτόν τον τρόπο να αυξήσει τις δυνατότητές του ως προς την παρουσίαση περιεχομένου βίντεο.
- Το αντικείμενο **scenes** μπορεί να πραγματοποιήσει αυτό που υπόσχεται το όνομά του δηλαδή την παρουσίαση των διαφορετικών σκηνών ενός βίντεο μόνον αν δεχθεί την κατάλληλη είσοδο. Η είσοδος αυτή είναι η πληροφορία για το που βρίσκονται στο βίντεο τα frames τα οποία αντιπροσωπεύουν την αρχή μιας νέας σκηνής. Αυτή η είσοδος στην παρούσα υλοποίηση είναι το όρισμα frames δηλαδή ο πίνακας που περιέχει τις χρονικές στιγμές στις οποίες ξεκινάει μια καινούρια σκηνή. Ωστόσο δεν ασχοληθήκαμε καθόλου με τον τρόπο με τον οποίο θα βρεθούν οι χρονικές στιγμές στις οποίες ξεκινάει μία καινούρια σκηνή. Η υλοποίησης μιας συνάρτησης η οποία θα έκανε αυτήν την δουλειά θα ολοκλήρωνε την εφαρμογή. Αυτή η συνάρτηση θα τροφοδοτούσε την **makeScenes** με την πληροφορία για το που βρίσκονται τα frames που πρέπει να αποσπάσει από το βίντεο ώστε οι εικόνες που θα προκύψουν να μην είναι τυχαίες αλλά να έχουν κάποια λογική σχετική με το περιεχόμενο του βίντεο. Η συνάρτηση αυτή θα ξέρει πότε περνάμε από την μία σκηνή στην άλλη.

6. ΟΡΟΛΟΓΙΑ

Ξενόγλωσσος Όρος	**Ελληνικός Όρος**
Attribute	Ιδιοχαρακτηριστικό
Browser	Περιηγητής
By default	Από προεπιλογή
Constructor method	Μέθοδος υλοποίησης
Documentation	Τεκμηρίωση
Element	Στοιχείο
Frame	Πλαίσιο
Function	Συνάρτηση
Object	Αντικείμενο
On line	Σε απευθείας σύνδεση
Progressive download video	Βίντεο προοδευτικής λήψεως
Property	Ιδιότητα
Server	Εξυπηρετητής
Specification	Προδιαγραφή
Streaming video	Βίντεο συνεχούς ροής
Tutorial	Οδηγός εκμάθησης

7. ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΑ

API	Application Programming Interface
DOM	Document Object Model
fps	frame per second
HTML	HyperText Markup Language
MDN	Mozila Developer Network
W3C	World Wide Web Consortium
XML	Extensible Markup Language

8. ΠΑΡΑΡΤΗΜΑ Ι – DOCUMENTATION

<scenes> is a custom HTML tag and its behavior is similar to normal HTML tags. **Scenes**, also, can used as a Javascript object. **Scenes** functionality is to display an ever-changing bar of images placed next to each other. The displaying images are selected frames from the video. The central image is the first frame of the current scene of the video. The images in the right are the first frames of the following scenes and the images in the left are the first frames from the previews scenes. The number of displaying images is specified from attribute *number* and the frames will be captured from the video to represent the start of scenes are specified from attribute *frames*. The length of <scenes> is always equal to video length, regardless of the number of displaying images. If you click on an displaying image the video goes to the corresponding frame.

Πίνακας 1: Attributes of custom multimedia HTML tag <scenes>

Attribute	**Value**	**Description**
videoID	[id]	The id of relative HTML <video> element. By default the relative HTML video element is the immediately following video element after the <scenes> element .
number	[number of displaying frames]	The integer that specify the number of displaying images in the bar. By default *number* = '5'.
fps	[frames per second]	The double that specify the frame per second of relative video. It is used only with *unit* = 'number'. By default *fps* = '29,97'.
unit	'second' or 'number'	It specifies the unit of attribute *frames*. If *unit* = 'second' attribute *frames* contains the seconds of selected frames. If *unit* = 'number' attribute *frames* contains the numbers of selected frames. By default *unit* = 'second'. It is recommended to assign *unit* = 'second' or not to assign.
frames	[Array of frame numbers] or [Array of frame seconds]	If *unit* = 'number', frames is a string that contains the integers, separated by a space, that specifies the number of selected frames. Else frames is a string of doubles, separated by space, that specifies the seconds in the video when selected frames occurs. By default the number of selected frames is twice the *number* attribute and *frames* contains the seconds of selected frames that are equidistant from each other.

Example:
```
<scenes number = "7" videoID = "vid1" frames = "80.5 160.9 240 320.25 400 480
560 640.05 720 800" ></scenes>
```

Example:
```
<scenes videoID = "vid1" frames = "2400 4800 7200 9600 12000 14400 16800 19200
21600 24000" unit = "number"></scenes>
```

Πίνακας 2: constructor method of Javascript class ***scenes***

Constructor method
scenes(tag, video, frames, number)

Πίνακας 3: Parameters of scenes' Javascript constructor

Parameter	**Type**	**Description**
tag	HTML element	The custom multimedia HTML element <scenes>
video	HTML5 video element	The relative **video** element.
frames	Array of doubles	It is an array of doubles that specifies the seconds in the video when selected frames occurs.
number	integer	The integer that specify the number of displaying images in the bar.

Example:
```
var fr = [0, 80, 160, 240, 320, 400, 480, 560, 640, 720, 800, 880];
vid = document.getElementById("vid1");
sc = new scenes(null, vid, fr, 7);
```

Πίνακας 4: Properties of Javascript object scenes

Property	**Type**	**Description**
self	HTML element	The custom tag. The bar with displaying images.

Example:
```
region = document.getElementById("div1");
region.appendChild(sc.self);
```

To use the the custom HTML tag <scenes> first you must import the file *MyLybrary.js* and you must execute the **initializeUI** function on loading of HTML <body>.

Example:
```
<html>
<head>
<script type="text/javascript" src="MyLibrary.js"></script>
</head>
<body onload = "initializeUI()">
...
</body>
```

9. ΠΑΡΑΡΤΗΜΑ ΙΙ – ΤΕΚΜΗΡΙΩΣΗ

Το **<scenes>** είναι ένα custom HTML tag και η συμπεριφορά του είναι παρόμοια με τα κανονικά HTML tag. Η λειτουργικότητά του **scenes** είναι να εμφανίζει μια συνεχώς μεταβαλλόμενη μπάρα από εικόνες τοποθετημένες η μία δίπλα στην άλλη. Οι εμφανιζόμενες εικόνες είναι επιλεγμένα frames από το βίντεο. Η κεντρική εικόνα είναι το πρώτο frame της τρέχουσας σκηνής του βίντεο. Οι εικόνες στα δεξιά είναι τα πρώτα frame των σκηνών που ακολουθούν και οι εικόνες στα δεξιά είναι τα πρώτα frame των σκηνών που προηγούνται. Ο αριθμός των εμφανιζόμενων σκηνών προσδιορίζεται από το attribute *number* και τα frame τα οποία θα αποσπαστούν από το βίντεο για να αντιπροσωπεύσουν την αρχή των σκηνών προσδιορίζονται από το attribute *frames*. Το μήκος του <scenes> είναι πάντα ίσο με το μήκος του βίντεο ανεξάρτητα από των αριθμό των εμφανιζόμενων εικόνων. Αν πατήσουμε με το ποντίκι πάνω σε μια εμφανιζόμενη εικόνα το βίντεο θα πάει στο αντίστοιχο frame.

Πίνακας 5: Attributes του custom multimedia HTML tag <scenes>

Attribute	**Value**	**Description**
videoID	[id]	Το id του συσχετιζόμενου HTML <video> element. Η προεπιλογή για το HTML <video> element είναι το ακριβώς επόμενο <video> element μετά το <scenes> element.
number	[number of displaying frames]	Ο ακέραιος που προσδιορίζει τον αριθμό των εμφανιζόμενων εικόνων στην μπάρα. Από προεπιλογή *number* = '5'.
fps	[frames per second]	Ο αριθμός (double) που προσδιορίζει το frame per second του συσχετιζόμενου βίντεο. Χρησιμοποιείται μόνο όταν έχουμε θέσει *unit* = 'number'. Από προεπιλογή *fps* = '29,97'.
unit	'second' or 'number'	Προσδιορίζει την μονάδα μέτρησης του attribute *frames*. Αν *unit* = 'second' το attribute *frames περιέχει* τα δευτερόλεπτα των επιλεγμένων frames. If *unit* = 'number' το attribute *frames* περιέχει τους αριθμούς των επιλεγμένων frames. Από προεπιλογή unit = 'second'. Συνίσταται να τίθεται *unit* = 'second' ή να μην τίθεται καμία τιμή.
frames	[frame numbers] or [frame seconds]	Αν *unit* = 'number' το *frames* είναι ένα string που περιέχει τους ακέραιους, διαχωρισμένων με space, οι οποίοι προσδιορίζουν τον αριθμό των επιλεγμένων frames. Διαφορετικά το *frames* είναι ένα string που περιέχει τους αριθμούς (doubles), διαχωρισμένων με space, οι οποίοι προσδιορίζουν τα δευτερόλεπτα στα οποία εμφανίζονται τα επιλεγμένα frames. Από προεπιλογή ο αριθμός των επιλεγμένων frames είναι διπλός του attribute *number* και το *frames* περιέχει τα δευτερόλεπτα των επιλεγμένων frames τα οποία είναι ισοκατανεμημένα.

Παράδειγμα:
```
<scenes number = "7" videoID = "vid1" frames = "80.5 160.9 240 320.25 400 480
560 640.05 720 800" ></scenes>
```

Πάραδειγμα:
```
<scenes videoID = "vid1" frames = "2400 4800 7200 9600 12000 14400 16800 19200
21600 24000" unit = "number"></scenes>
```

*Πίνακας 6: μέθοδος υλοποίησης Javascript κλάσης **scenes***

Constructor method
scenes(tag, video, frames, number)

*Πίνακας 7: Ορίσματα μεθόδου υλοποίησης Javascript κλάσης **scenes***

Parameter	**Type**	**Description**
tag	HTML element	Το custom multimedia HTML element <scenes>
video	HTML5 video element	Το συσχετιζόμενο **video** element.
frames	Array of doubles	Πίνακας από αριθμούς (doubles) που προσδιορίζουν τα τα δευτερόλεπτα του βίντεο στα οποία εμφανίζονται τα επιλεγμένα frames.
number	integer	Ο ακέραιος που προσδιορίζει των αριθμό των εμφανιζόμενων εικόνων στην μπάρα.

Παράδειγμα:
```
var fr = [0, 80, 160, 240, 320, 400, 480, 560, 640, 720, 800, 880];
vid = document.getElementById("vid1");
sc = new scenes(null, vid, fr, 7);
```

*Πίνακας 8: Properties του Javascript αντικειμένου **scenes***

Property	**Type**	**Description**
self	HTML element	Το custom tag. Η μπάρα με τα εμφανιζόμενα frames.

Παράδειγμα:
```
region = document.getElementById("div1");
region.appendChild(sc.self);
```

Για να χρησιμοποιήσουμε το custom HTML tag <scenes> first πρώτα πρέπει να εισάγουμε το αρχείο *MyLybrary.js* και στην συνέχεια πρέπει να εκτελέσουμε την συνάρτηση **initializeUI** κατά την φόρτωση του HTML <body>.

Παράδειγμα:

```
<html>
<head>
<script type="text/javascript" src="MyLibrary.js"></script>
</head>
<body onload = "initializeUI()">
...
</body>
```

10. ΠΑΡΑΡΤΗΜΑ III – ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ, ΧΡΗΣΙΜΕΣ ΔΙΑΣΥΝΔΕΣΕΙΣ

1. Θεωρία

Βασικές Γνώσεις

- HTML Tutorial [1]: http://www.w3schools.com/html/default.asp
- HTML5 Tutorial [2]: http://www.w3schools.com/html5/default.asp
- Javascript Tutorial [3]: http://www.w3schools.com/js/default.asp
- XML Tutorial [4]: http://www.w3schools.com/xml/default.asp
- HTML DOM Tutorial [5]: http://www.w3schools.com/htmldom/default.asp

Εξειδικευμένες γνώσεις

- HTML video element Specification [6]: http://dev.w3.org/html5/spec/Overview.html#the-video-element
- HTML canvas element Specification [7]: http://dev.w3.org/html5/spec/Overview.html#the-canvas-element
- HTML media elements Specification [8]: http://dev.w3.org/html5/spec/Overview.html#media-elements

Βιβλία

- "The Definitive Guide to HTML5 Video – Everything you need to know about the new HTML5 video element", Silvia Pfeiffer, Apreess, 2010 [9]
- "Object Oriented Javascript – Create scalable, reuasable high-quality JavaScript applications, and libraries", Stoyan Stefanov, Packt Publishing, 2008 [10]

2. Εφαρμογές

- **Using HTML5 Canvas to capture frames from a video**

Η εφαρμογή αφορά την λήψη του τρέχοντος frame ενός βίντεο με το πάτημα ενός κουμπιού και την παρουσίαση των των τελευταίων τεσσάρων frames με την σειρά που αποσπάστηκαν σε μια περιοχή κάτω από το video. [11]

http://appcropolis.com/blog/using-html5-canvas-to-capture-frames-from-a-video/

- **Video Timeline HTML5 and JavaScript**

Στην εφαρμογή αυτή λαμβάνονται περιοδικά frames από κάποιο video και τοποθετούνται σε μία περιοχή δίπλα στο προβαλλόμενο βίντεο. Μόλις η περιοχή αυτή γεμίσει με κάποιο συγκεκριμένο αριθμό από frames τότε το κάθε καινούριο frame που προσλαμβάνεται τοποθετείται στην θέση του παλιότερου από τα εμφανιζόμενα. Όταν πατήσουμε με το ποντίκι πάνω σε κάποιο frame τότε το βίντεο πηγαίνει πίσω στην αντίστοιχη χρονική στιγμή που προβάλλεται το συγκεκριμένο frame. Το αρχείο που δίνεται περιέχει μη ενεργά link για τα video οπότε πρέπει να τα αντικαταστήσουμε. Δοκιμάστηκε με Firefox, και Chrome και λειτουργεί σωστά μόνο στο Chrome. [12]

http://www.codeproject.com/Tips/226990/Video-Timeline-using-HTML5-and-javascript?display=Print

- **HTML5 Video Events and API**

Στην σελίδα αυτή εξετάζονται τα events και τα properties του **video**. Μέσω αυτής της εφαρμογής μπορούμε να ελέγχουμε κάθε στιγμή την τιμή των properties και να βλέπουμε πότε συμβαίνει κάποιο event καθώς το βίντεο παίζει, σταματά, ψάχνει κτλ. Χρησιμεύει πάρα πολύ στην καλύτερη κατανόηση της εσωτερικής λειτουργίας του **video** element. [13]

http://www.w3.org/2010/05/video/mediaevents.html

- **Creating custom HTML tags, DOM style**

Παρουσιάζεται μία τεχνική για την δημιουργία custom HTML tags σύμφωνα με την τεχνοτροπία DOM. Η λειτουργικότητα του custom tag υλοποιείται από μία κλάση Javascript. Το tag φορτώνεται από μία custom library. Η διαδικασία περιγράφεται αναλυτικά μέσω παραδείγματος. [14]

http://cherianajay.hubpages.com/hub/extented-HTML-tags

3. Χρήσιμα Links

- http://html5video.org/ - community site για HTML5 video
- http://html5doctor.com/ - site για HTML5
- https://developer.mozilla.org/en/HTML/HTML5 - docs για HTML5 από το MDN (Mozilla Developer Network)
- http://jsfiddle.net/ - on line εργαλείο ανάπτυξης web εφαρμογών

11. ΑΝΑΦΟΡΕΣ

[1] W3Schools, HTML Tutorial, http://www.w3schools.com/html/default.asp
[Προσπελάστηκε 9/4/12]
[2] W3Schools, HTML5 Tutorial, http://www.w3schools.com/html5/default.asp
[Προσπελάστηκε 9/4/12]
[3] W3Schools, Javascript Tutorial, http://www.w3schools.com/js/default.asp
[Προσπελάστηκε 9/4/12]
[4] W3Schools, XML Tutorial, http://www.w3schools.com/xml/default.asp
[Προσπελάστηκε 9/4/12]
[5] W3Schools, HTML DOM Tutorial, http://www.w3schools.com/htmldom/default.asp
[Προσπελάστηκε 9/4/12]
[6] Ian Hickson, World Wide Web Consortium (W3C) specification, A vocabulary and associated APIs for HTML and XHTML, HTML video element, March 2012 http://dev.w3.org/html5/spec/Overview.html#the-video-element [Προσπελάστηκε 9/4/12]
[7] Ian Hickson, World Wide Web Consortium (W3C) specification, A vocabulary and associated APIs for HTML and XHTML, HTML canvas element, March 2012 http://dev.w3.org/html5/spec/Overview.html#the-canvas-element [Προσπελάστηκε 9/4/12]
[8] Ian Hickson, World Wide Web Consortium (W3C) specification, A vocabulary and associated APIs for HTML and XHTML, HTML media elements, March 2012 http://dev.w3.org/html5/spec/Overview.html#media-elements [Προσπελάστηκε 9/4/12]
[9] "The Definitive Guide to HTML5 Video – Everything you need to know about the new HTML5 video element", Silvia Pfeiffer, Apreess, 2010
[10] "Object Oriented Javascript – Create scalable, reuasable high-quality JavaScript applications, and libraries", Stoyan Stefanov, Packt Publishing, 2008
[11] Raul Sanchez, 'Using HTML5 Canvas to capture frames from a video', September 2011 http://appcropolis.com/blog/using-html5-canvas-to-capture-frames-from-a-video/
[Προσπελάστηκε 9/4/12]
[12] 'Video Timeline HTML5 and JavaScript', http://www.codeproject.com/Tips/226990/Video-Timeline-using-HTML5-and-javascript?display=Print [Προσπελάστηκε 9/4/12]
[13] World Wide Web Consortium (W3C), 'HTML5 Video Events and API', http://www.w3.org/2010/05/video/mediaevents.html [Προσπελάστηκε 9/4/12]
[14] 'Creating custom HTML tags, DOM style', http://cherianajay.hubpages.com/hub/extented-HTML-tags [Προσπελάστηκε 9/4/12]