Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Εφαρμοσμενων Μαθηματικων
και Φυσικων Επιστημων

Τομεας Μαθηματικων

Διαδραστικότητα αρχείων με server

Πτυχιακή Εργασία

της

Αικατερίνης Μπαούση

Αθήνα, Ιούλιος 2016

Εθνικο Μετοβιο Πολυτεχνειο
Σχολη Εφαρμοσμενων Μαθηματικων και Φυσικων Επιστημων
Τομεας Μαθηματικων

Διαδραστικότητα αρχείων με server

Πτυχιακή Εργασία

της

Αικατερίνης Μπαούση

Επιβλέπων: Αντώνιος Συμβώνης
Καθηγητής ΕΜΠ

Τριμελής Επιτροπή

.....	.....	.....
Α. Συμβώνης	Ι. Κολέτσος	Π. Στεφανέας
Καθηγητής	Επικ. Καθηγητής	Επικ. Καθηγητής
ΕΜΠ	ΕΜΠ	ΕΜΠ
(επιβλέπων)		

Αθήνα, Ιούλιος 2016

.....

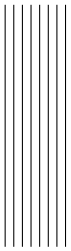
ΑΙΚΑΤΕΡΙΝΗ ΜΠΑΟΥΣΗ
Σχολή Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω ιδιαίτερα τον επιβλέποντα μου καθηγητή κ. Αντώνιο Συμβώνη, για τη συνεχή καθοδήγηση του στην εκπόνηση της παρούσας διπλωματικής, αλλά και για την ευκαιρία που μου έδωσε με την ανάθεσή της. Επιπρόσθετα, θα ήθελα να ευχαριστήσω τα μέλη της επιτροπής για τη σημαντική συμβολή τους στην περαίωση της διπλωματικής εργασίας. Ξεχωριστά θα ήθελα να ευχαριστήσω την διδάκτορα Χρυσάνθη Ραφτοπούλου για πολύτιμη βοήθεια και αμέριστη συμπαράσταση που μου προσέφερε καθόλη την διάρκεια της εκπόνησης της πτυχιακής μου εργασίας.Τέλος, ευχαριστώ θερμά τους δικούς μου ανθρώπους για την στήριξη και την υπομονή τους όλα αυτά τα χρόνια.

Περίληψη

Στόχος της παρούσας πτυχιακής εργασίας είναι η εξερεύνηση των δυνατοτήτων των αρχείων epub3 και των αρχείων PDF για το θέμα της απομακρυσμένης ανταλλαγής δεδομένων και της δυναμικής παραγωγής περιεχομένου. Εξετάζονται οι τεχνολογίες που αφορούν την αποστολή και λήψη πληροφοριών στο διαδίκτυο.Επιπρόσθετα ελέγχεται και το θέμα της αποθήκευσης δεδομένων τοπικά με στόχο την αμεταβλητότητά τους και την ανεξαρτησία τους από την διαδικτυακή εφαρμογή που τα παρουσιάζει.Η σκοπιά με την οποία εξετάζονται τα παραπάνω ζήτηματα είναι για την χρήση των αρχείων epub3 και PDF στην αντιμετώπιση μαθησιακών δυσκολιών ως παιχνίδιθεραπεία από τους παιδαγωγούς-λογοθεραπευτές.



Abstract

The main goal of this diploma thesis is to explore the possibilities of epub3 and pdf files concerning remote data transactions and dynamic content production. The technologies used for receiving and sending data to the Internet are examined closely. Furthermore, the local data storage issue is being inspected in order for the data to become immutable and independent from the web application that presents them. The spectrum under which the aforementioned issues are studied, is to apply epub3 and PDF files in the form of game/treatment that special therapists will use for helping people facing learning difficulties.

Περιεχόμενα

Περιεχόμενα **13**

1 Εισαγωγή **15**
1.1 Γενική Αναδρομή . . . 15
1.2 Σκοπιά Πτυχιακής Εργασίας . . . 16
1.3 Οργάνωση της Πτυχιακής Εργασίας . . . 16

2 Τεχνολογίες Διάδρασης **19**
2.1 Επικοινωνία Με Server . . . 19
2.1.1 Get . . . 20
2.1.2 Post . . . 24
2.2 Αποθήκευση σε Τοπικά Αρχεία . . . 26
2.2.1 Cookies . . . 26
2.2.2 Web Storage & Local Storage . . . 29
2.2.3 Automation & ActiveXObject . . . 31

3 Αρχεία epub3 **33**
3.1 Γενικά στοιχεία αρχείων Epub . . . 33
3.1.1 Το ζήτημα της διαδραστικότητας . . . 35
3.1.2 Editors & Viewers . . . 37
3.2 Η γλώσσα Javascript και η βιβλιοθήκη της jQuery . . . 38
3.3 Διαδραστικότητα αρχείων epub3 . . . 41
3.4 Εφαρμογή σε αρχείο epub3 . . . 43

4 Αρχεία PDF **51**
4.1 Εισαγωγή στην Acrobat Javascript . . . 51
4.2 Δυνατότητες δημιουργίας τοπικών αρχείων . . . 55
4.2.1 Δημιουργία σχολίων μέσα στο αρχείο . . . 55
4.2.2 Αποθήκευση αρχείου σε άλλο format . . . 56
4.3 Επικοινωνία PDF αρχείων με βάσεις δεδομένων και server . . . 58
4.3.1 Το αντικείμενο ADBC . . . 58

4.3.2 Χρήση των forms για επικοινωνιία με τον server 59

5 Συμπεράσματα και μελλοντικές επεκτάσεις **63**

Βιβλιογραφία **65**

1 Εισαγωγή

1.1 Γενική Αναδρομή

Το διαδίκτυο έχει γίνει πλέον αναπόσπαστο κομματι της ζωής μας. Όλοι είμαστε εξοικειωμένοι με την χρήση του χωρίς όμως να γνωρίζουμε σε βάθος την δομή και τον τρόπο λειτουργίας του.Πρακτικά το διαδίκτυο αποτελεί ένα παγκόσμιο σύστημα διασυνδεδεμένων δικτύων υπολογιστών, οι οποίοι χρησιμοποιούν καθιερωμένη ομάδα πρωτοκόλλων, η οποία συχνά αποκαλείται "TCP/IP" για να εξυπηρετεί εκατομμύρια χρηστών καθημερινά σε ολόκληρο τον κόσμο. Οι διασυνδεδεμένοι ηλεκτρονικοί υπολογιστές ανά τον κόσμο, οι οποίοι βρίσκονται σε ένα κοινό δίκτυο επικοινωνίας, ανταλλάσσουν μηνύματα (πακέτα) με τη χρήση διαφόρων πρωτοκόλλων (τυποποιημένοι κανόνες επικοινωνίας), τα οποία υλοποιούνται σε επίπεδο υλικού και λογισμικού.

Αναλύοντας σε βάθος τους όρους που αναφέρονται παραπάνω αξίζει να σημειωθεί ότι το TCP/ IP είναι το πρωτόκολλο που ελέγχει το ρυθμό / ροή της επικοινωνίας. Οι κύριοι στόχοι του πρωτοκόλλου TCP είναι να επιβεβαιώνεται η αξιόπιστη αποστολή και λήψη δεδομένων, επίσης να μεταφέρονται τα δεδομένα χωρίς λάθη μεταξύ του στρώματος δικτύου (network layer) και του στρώματος εφαρμογής (application layer) και, φτάνοντας στο πρόγραμμα του στρώματος εφαρμογής, να έχουν σωστή σειρά. Οι περισσότερες σύγχρονες υπηρεσίες στο Διαδίκτυο βασίζονται στο TCP.Κάθε συγκεκριμένος τύπος επικοινωνίας που πραγματοποιείται, χρειάζεται επιπλέον κανόνες που καθορίζουν πώς θα πραγματοποιηθεί. Για παράδειγμα το SMTP (port 25), το παλαιότερο (και μη-ασφαλές) Telnet (port 23), το FTP και πιο σημαντικό το HTTP (port 80) το οποίο αναλύεται στο δεύτερο κεφάλαιο εκτενέστερα.

Οι εφαρμογές λογισμικού που επιτρέπουν την οπτικοποίηση των αποτελεσμάτων που λαμβάνονται μέσω της επικοινωνίας των πρωτοκόλλων με το στρώμα του δικτύου είναι οι web browsers (φυλλομετρητές).Ένας web browser επιτρέπει στον χρήστη του να προβάλλει, και να αλληλεπιδρά με, κείμενα, εικόνες, βίντεο, μουσική, παιχνίδια και άλλες πληροφορίες συνήθως αναρτημένες σε μια ιστοσελίδα ενός ιστότοπου στον Παγκόσμιο Ιστό ή σε ένα τοπικό δίκτυο.Χρησιμοποιεί τη γλώσσα μορφοποίησης HTML για την προβολή των ιστοσελίδων, για αυτό η εμφάνιση μιας ιστοσελίδας μπορεί να διαφέρει ανάλογα με τον browser.Οι πιο διαδεδομένοι web browsers είναι ο

Google Chrome, Mozilla Firefox, Safari και Opera.

Το διαδίκτυο μπορεί να επιφέρει σημαντική βελτίωση σε ποικίλους τομείς της ζωής μας και στην αντιμετώπιση διαφόρων δυσκολιών που την πλήττουν. Συγκεκριμένα θα μπορούσε να ενισχύσει το έργο των παιδαγωγών και των λογοθεραπευτών για την παροχή εξατομικευμένης και άμμεσης θεραπείας σε μαθησιακές δυσκολίες όπως δυσλεξία, δυσορθογραφία που αντιμετωπίζουν πολλοί μαθητές.Παρακάτω θα δούμε πώς αυτό μπορεί να γίνει δυνατό.

1.2 Σκοπιά Πτυχιακής Εργασίας

Σκοπός αυτής της εργασίας είναι η εξερεύνεση της ύπαρξης και εφαρμογής της απαραίτητης τεχνολογίας για την δημιουργία διαδραστικών αρχείων.Τα αρχεία αυτά θα χρησιμοποιηθούν για την αντιμετώπιση μαθησιακών δυσκολιών ως παιχνίδι-θεραπεία από τους παιδαγωγούς-λογοθεραπευτές που αναφέρθηκαν παραπάνω.

Τα αρχεία θα πρέπει να είναι ικάνα να λαμβάνουν δεδομένα από συγκεκρίμενη πηγή του δικτύου αλλά και να αποστέλουν πληροφόριες σχετικές με την πορεία του χρήστη στο παιχνίδι-θεραπεία πίσω στην ίδια πηγή. Παράλληλα τα αρχεία αυτά πρεπεί να έχουν ελκυστικό περιεχόμενο καθώς η πλειοψηφία των χρηστών θα είναι άτομα μικρής ηλικίας.

Η μορφή των αρχείων που εξετάστηκε αρχικά είναι αυτή των epub3 αρχείων τα οποία όπως θα δούμε στην συνέχεια αποδείχθηκαν ικανά να αποδώσουν το περιεχόμενο και να καλύψουν όλες τις ανάγκες της εφαρμογής που παρατέθηκε παραπάνω. Συγκεκριμένα αναπτύχθηκε ένα παράδειγμα αρχείου epub3 που θα αναλυθεί στα επόμενα κεφάλαια. Τέλος έγινε προσπάθεια επέκτασης και σε pdf αρχεία τα οποία μπορούν να καλύψουν αρκετές από τις προαναφερθείσες ανάγκες.

1.3 Οργάνωση της Πτυχιακής Εργασίας

Ξεκινώντας στο δεύτερο κεφάλαιο αναφέρονται οι βασικές τεχνολογίες που χρησιμοποιήθηκαν για την δημιουργία διαδραστικών αρχείων. Αναλυτικότερα εξετάζεται η επικοινωνία με τον server και μελετούνται οι μέθοδοι του HTTP πρωτοκόλλου GET και POST.Στη συνέχεια εξετάζονται τα τοπικά αρχεία που παρέχουν την δυνατότητα αποθήκευσης πληροφοριών σχετικές με τις ενέργειες του χρήστη.

Στο τρίτο κεφάλαιο αναλύονται τα αρχεία epub3. Παρατίθενται γενικά χαρακτηριστικά που αφορούν την μορφή τους και διέπουν την λειτουργία τους. Για την κωδικοποίηση των αρχείων αυτών χρησιμοποιήθηκαν η γλώσσα Javascript και η βιβλιοθήκη της jQuery όπου βλέπουμε γενικά τους χαρακτηριστικά σε αυτό το σημείο. Εν συνεχεία εξετάζεται και παρουσιάζεται η διαδραστικότητα μέσω μιας εφαρμογής κώδικα

Javascript σε αρχείο epub3.

Ολοκληρώνοντας στο τέταρτο κεφάλαιο επεκτείνεται η μελέτη για διαδραστικότητα σε αρχεία μορφής pdf. Ειδικότερα γίνεται μια εισαγωγή στην γλώσσα Acrobat Javascript. Επιπλέον αναφέρεται η δυνατότητα δημιουργίας τοπικών αρχείων αλλά και η επικοινωνία αυτών με βάσεις δεδομένων και server.

2 Τεχνολογίες Διάδρασης

2.1 Επικοινωνία Με Server

Το πρωτόκολλο HTTP (Hyper Text Transfer Protocol) είναι ένα πρωτόκολλο επιπέδου εφαρμογής.Είναι υπεύθυνο για την ορθή μετάδοση των ζητούμενων πληροφοριών-δεδομένων(HTML αρχεία, εικόνες, αποτελέσματα αναζήτησης κ.α) ανάμεσα σε μια κατανεμημένη εφαρμογή και έναν χρήστη. Ορίζει τον τρόπο με τον οποίο οι χρήστες μπορούν να στείλουν ένα αίτημα για δεδομένα από το server και πώς θα απαντήσει σε αυτό το αίτημα στην συνέχεια ο server. Τέλος, αποτελεί το θεμελιώδες σύνολο κανόνων στην μετάδοση δεδομένων στο World Wide Web από το 1990 μέχρι σήμερα.

Βασικά χαρακτηριστικά του πρωτοκόλλου HTTP είναι ότι είναι ανεξάρτητο της σύνδεσης, του τύπου των δεδομένων που μεταφέρονται και της ύπαρξης του server. Η ανεξαρτησία του πρωτοκόλλου όσον αφοράν την σύνδεση έγκειται στο γεγονός ότι από την στιγμή που γίνεται το HTTP αίτημα από τον browser η εφαρμογή παύει να είναι συνδεδεμένη με τον server και η σύνδεση αποκαθίσταται μόνο για την στιγμή που λαμβάνεται η απάντηση στο αίτημα. Επιπλέον οποιοσδήποτε τύπος δεδομένων μπορεί να σταλεί απο την εφαρμογή στον server και αντίστροφα αρκεί να υπάρχει το κατάλληλο λογισμικό για την διαχείριση τους. Συμπερασματικά το HTTP πρωτόκολλο είναι ανεξάρτητο και του τύπου των δεδομένων. Λόγω της μη αναγκαιότητας ύπαρξης σύνδεσης το HTTP μπορεί να χαρακτηριστεί και ”stateless” διότι η εφαρμογή ”γνωρίζει” την ύπαρξη του server μόνο κατά την διάρκεια της μεταξύ τους αλληλεπίδρασης, εκ’ τότε ο ένας αγνοεί την ύπαρξη του άλλου.

Αναφέρονται βασικές μέθοδοι που χρησιμοποιούν HTTP πρωτόκολλο για να ζητήσουν/στείλουν δεδομένα:

GET

Η μέθοδος GET ζητά μια αναπαράσταση συγκεκριμένης πηγής δεδομένων. Τα αιτήματα GET πρέπει μόνο να ζητούν την ανάκτηση δεδομένων από το server.

HEAD

Η μέθοδος HEAD στέλνει ένα ίδιου τύπου αίτημα με την μέθοδο GET με την βασική διαφορά ότι αναφέρεται όχι στην πλήρη πληροφορία που παρέχουν τα δεδομένα αλλά στην φύση των ίδιων των δεδομένων (πχ στην κωδικοποίησή τους).

POST

Η μέθοδος POST ζητά από τον server να δεχθεί τα δεδομένα που περικλύονται στο αίτημα ως μια νέα προσθήκη στην πηγή που προσδιορίζεται από το URI (string που αναφέρεται στην διεύθυνση της πηγής). Τα δεδομένα που στέλνονται μέσω της μεθόδου POST μπορεί να είναι για παράδειγμα ένας σχολιασμός για τους υπάρχουσες πηγές, ένα μπλοκ δεδομένων που είναι το αποτέλεσμα της υποβολής μιας φόρμας web σε μια διαδικασία επεξεργασίας δεδομένων ή ένα στοιχείο που πρέπει να προστεθεί σε μια βάση δεδομένων.

DELETE

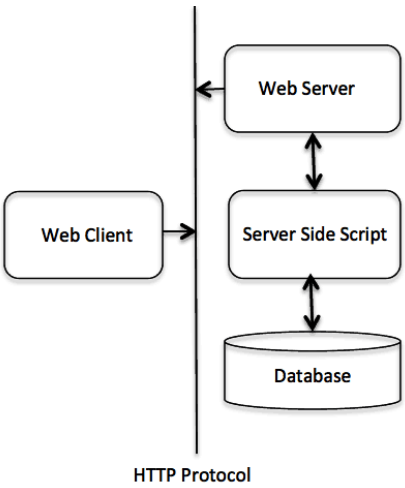
Η μέθοδος DELETE διαγράφει μια συγκεκριμένη πηγή.

TRACE

Η μέθοδος TRACE επιτρέπει στον χρήστη να δει αν έχουν συμβεί μεταβολές στις πηγές του server λόγω της επέμβαση ενδιάμεσων server σε αυτά.

OPTIONS

Η μέθοδος OPTIONS επιστρέφει τις μεθόδους HTTP που υποστηρίζει ο server για ένα συγκεκριμένο URL.



2.1.1 Get

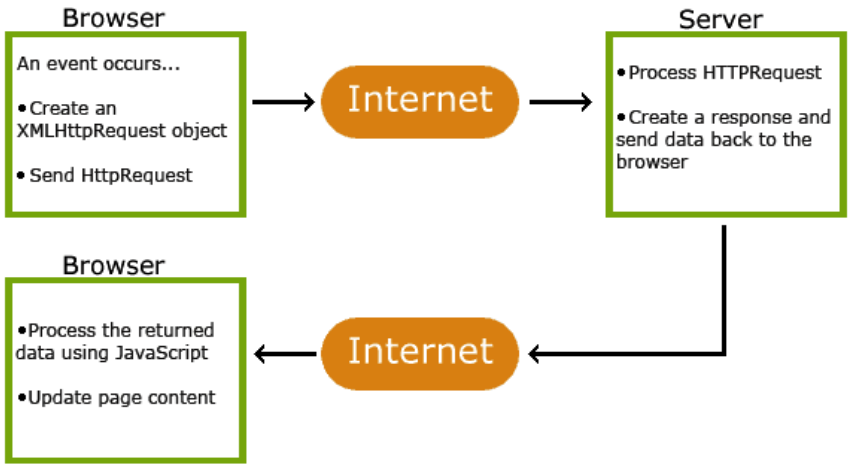
Η μέθοδος GET χρησιμοποιείται για την ανάκτηση πληροφοριών από κάποιο συγκεκριμένο URI που θεωρείται ασφαλης πηγή. Η μέθοδος αυτή είναι ασφαλής και μπορεί να επαναλαμβάνεται από τον browser χωρίς την άμμεση έγκρισή της αποστολής του αιτήματος για GET από τον χρήστη.

Για παράδειγμα η χρήση της μεθόδου GET για την εμφάνιση του υπολοίπου ενός τραπεζικού λογαριασμού ενός χρήστη σε μια ιστοσελίδα κάποιας τράπεζας δεν έχει καμία επίδραση στον ίδιο τον λογαριασμό και σαν διαδικασιά μπορεί να επαναληφθεί με ασφάλεια. Ο web browser που στέλνει το αίτημα θα πρέπει να επιτρέπει σε έναν

χρήστη να ανανεώνει τις ιστοσελίδες που έχουν προκύψει από μια μέθοδο GET χωρίς την εμφάνιση προειδοποιητικών μηνυμάτων.

Ένα μειονέκτημα της χρήσης της μεθόδου GET είναι ότι τα δεδομένα που αυτή μας επιστρέφει είναι εμφανή στο URL της σελίδας που έχει προκύψει ως αποτέλεσμα ενός αιτήματος με αυτή τη μέθοδο. Αυτό κάνει την χρήση αυτής της μεθόδου απαγορευτική όταν πρόκειται για αιτήματα που σχετίζονται με λογαριασμούς χρηστών γιατί αν χρησιμοποιούνταν η μέθοδος GET ο κωδικός πρόσβασης και το όνομα χρήστη θα ήταν εμφανή στο URL κάθε σελίδας μετά την εξακρίβωση των στοιχείων.

Τα αιτήματα μέσω της μεθόδου GET στην Javascript μπορούν να υλοποιηθούν μέσω της τεχνολογίας Asynchronous Javascript XML (AJAX)[1]. Χάρη σε αυτή την τεχνολογία έχουμε την δυνατότητα να διαμορφώσουμε ένα αίτημα HTTP με χρήση του αντικειμένου XMLHttRequest. Αυτό το αντικείμενο προσφέρει έναν πολύ εύκολο τρόπο σύνταξης και κατασκευής ενός αιτήματος προς τον server και στην συνέχεια επικοινωνεί με τον κώδικα που βρίσκενται εντός του server για να φέρει την ζητούμενη απάντηση. Επίσης η τεχνολογία αυτή ονομάστηκε ασύγχρονη διότι η ιστοσελίδα δεν παραμένει αδρανής μέχρι να πάρει την απάντηση από τον server αλλά ο κώδικας Javascript που δεν σχετίζεται με το αίτημα μπορεί να υλοποιηθεί άσχετα με το αν έχουμε πάρει την απάντηση από τον server ή όχι. Όταν λάβουμε την απάντηση του server τότε το κομμάτι του κώδικα που είναι άμμεσα συνδεδεμένο με αυτή θα εκτελεστεί κανονικά. Ένας τρόπος για να καταλάβουμε πώς λειτουργεί η τεχνολογία AJAX είναι αν σκεφτούμε τι συμβαίνει όταν επιλέγουμε την αναπαραγωγή ενός βίντεο στην ιστοσελίδα www.youtube.com .



Παρατίθεται παράδειγμα κώδικα Javascript με χρήση AJAX που συντάσει ένα αίτημα XMLHttpRequest το αποστέλει στον server και εμφανίζει την απάντηση:

```
1 function loadDoc() {
2   var xhttp = new XMLHttpRequest();
```

```
3   xhttp.onreadystatechange = function show() {
4     if (xhttp.readyState == 4 && xhttp.status == 200) {
5       document.getElementById("demo").innerHTML = xhttp.responseText;
6     }
7   };
8   xhttp.open("GET", "http://localhost/text.txt", true);
9   xhttp.send();
10 }
```

Η διαδικασία loadDoc( ) δημιουργεί αρχικά ένα αντικείμενο XMLHttpRequest και το αποθηκεύει στη μεταβλητή xhttp. Μέσω της μεθόδου open(method,url,async) δηλώνονται η μέθοδος HTTP που αντιστοιχεί στο αίτημα, το URL στο οποίο απευθύνεται το αίτημα και εάν το αίτημα γίνεται ασύγχρονα ή όχι κατά την σειρά που εμφανίζονται και παραπάνω.Με την μέθοδο send( ) το αίτημα αποστέλεται.

Το αντικείμενο xhttp φέρει τις ιδιότητες readyState και status προκειμένου να μπορεί να προσφέρει πληροφορίες σχετικά με την εξέλιξη του αιτήματος στον server. Η ιδιότητα readyState λαμβάνει τιμές από 0 έως 4. Για την τιμή 0 συμπαιρένουμε ότι το αίτημα έχει δημιουργηθεί, για την τιμή 1 η σύνδεση του browser με τον server έχει εγκατασταθεί, για την τιμή 2 το αίτημα έχει παραδοθεί στον server, για την τιμή 3 το αίτημα αναλύεται και τέλος για την τιμή 4 το αίτημα έχει ολοκληρωθεί και η απάντηση σε αυτό είναι έτοιμη. Τέλος η ιδιότητα status προσδιορίζει αν η πληροφορία που ζητάται υπάρχει στο server ή όχι λαμβάνοντας τις τιμές 200 ή 404 αντίστοιχα.

Η ιδιότητα onreadystatechange καλεί την διαδικασία show κάθε φορά που η ιδιότητα readyState λαμβάνει διαφορετική τιμή. Όταν τελικά ληφθεί η απάντηση από τον server το μήνυμα κειμένου που έχει σταλεί εμφανίζεται στη θέση του στοιχείου 'demo' της HTML χάρη στην ιδιότητα responseText.

Η διαχείριση των αιτημάτων προς τον server μπορεί να απλοποιηθεί ακόμα περισσότερο μέσω της μεθόδου $.ajax(settings) που μας παρέχεται μέσω της βιβλιοθήκης της Javascript, jQuery. Μπορούμε να προσδιορίσουμε πλήρως το αίτημα μας δίνοντας στο αντικείμενο που δέχεται σαν όρισμα η μέθοδος $.ajax(settings) τις κατάλληλες τιμές.

Αναφέρουμε μερικά από τα ζεύγη τιμών-μεταβλητών που μπορεί να περιέχει το αντικείμενο settings της μεθόδου ajax[2]:

async

Λαμβάνει λογικές τιμές (true ή false) ανάλογα με το εάν το αίτημα πρέπει να χειριστεί ασύγχρονα ή όχι. Σε περίπτωση παράλειψης τίθεται true.

beforeSend

Διαδικασία που θα εκτελεστεί πριν την αποστολή του αιτήματος. *cache*

Λαμβάνει λογικές τιμές (true ή false) και υποδεικνύει στον browser αν θα αποθηκεύσει τις σελίδες που ζητήθηκαν στην προσωρινή μνήμη. Σε περίπτωση παράλειψης

τίθεται true.

complete

Διαδικασία η οποία υλοποιείται όταν το αίτημα ολοκληρωθεί, δηλαδή όταν ο browser λάβει την απάντηση του server.

dataFilter

Διαδικασία που υλοποιείται προκειμένου να εξυγιάνει την απάντηση του server. Για παράδειγμα μπορεί να χρειαστεί να δεδομένα της απάντησης να μετατραπούν σε άλλο τύπο προκειμένουν να ακολουθήσει η επεξεργασία του (πχ να χρησιμοποιηθεί η μέθοδος parseJSON(data)).

dataType

Λαμβάνει τιμές τύπου string που υποδηλώνουν τον τύπο των δεδομένων που αναμένονται ως απάντηση από τον server. Ενδεικτικές τιμές που μπορεί να πάρει είναι ”script”,”json”, ”text”.

error

Διαδικασία που υλοποιείται αν απορριφθεί το αίτημα από τον server.

method

Λαμβάνει τιμές τύπου sting για να υποδηλώσει την μέθοδο HTTP που υλοποιείται για το αίτημα. Σε περίπτωση παράλειψης τίθεται ”GET”.

password

Προσδιορίζει τον κωδικό πρόσβασης που μπορεί να ζητηθεί για αιτήματα που απαιτούν εξακρίβωση στοιχείων.

scriptCharset

Προσδιορίζει την κωδικοποίηση που θα έχει το αίτημα (πχ ”UTF-8”)

statusCode

Αντικείμενο που λαμβάνει ως ονόματα των πεδίων του τα πιθανά status του αιτήματος (200 για επιτυχία ή 404 για αποτυχία) και εκτελεί κάποια διαδικασία στην αντίστοιχη περίπτωση.

success

Διαδικασία που υλοποιείται σε περίπτωση επιτυχίας του αιτήματος.

type

Έχει ακριβώς την ίδια λειτουργία με το method που ορίστηκε παραπάνω και προτιμάται στις προγενέστερες εκδόσεις της jQuery 1.9.0.

url

Λαμβάνει τιμές τύπου string και υποδηλώνει την ακριβή τοποθεσία του αρχείου, στο οποίο απευθύνεται το αίτημα, στον server.

username

Λαμβάνει τιμές τύπου string και προσδιορίζει τον κωδικό πρόσβασης που μπορεί να ζητηθεί για αιτήματα που απαιτούν εξακρίβωση στοιχείων.

Παρατίθεται παράδειγμα κώδικα που επιστρέφει ακριβώς το ίδιο αποτέλεσμα με το προηγούμενο παράδειγμα υλοποιημένο με την μέθοδο $.ajax( ).

```
function loadDoc() {
  \$.ajax({
     url: "http://localhost/text.txt",
     type: 'GET',
     async: true,
     success:function (data)
     {
        document.getElementById("demo").innerHTML =data;
     }

  })
}
```

2.1.2 Post

Η μέθοδος POST είναι και αυτή μέθοδος που υποστηρίζεται από το πρωτόκολλο HTTP και χρησιμοποιείται πολύ συχνά στο Διαδίκτυο. Το αίτημα που στέλνεται μέσω αυτής της μεθόδου ζητά από τον server να δεχτεί και να αποθηκεύσει τα δεδομένα που περικλείονται εντός του αιτήματος. Τα δεδομένα μπορεί να είναι από ελάχιστες πληροφορίες που βρίσκονται μέσα σε πεδία κειμένου μιας ιστοσελίδας μέχρι και ολόκληρα αρχεία. Η ποσότητα των δεδομένων που μπορούν να αποσταλούν μέσω αυτής της μεθόδου είναι απεριόριστη σε αντίθεση με την ποσότητα των δεδομένων που μπορούν να αποσταλούν μέσω ενός αιτήματος GET που μπορούν να είναι το πολύ μέχρι 2048 χαρακτήρες.

Η μέθοδος χρησιμοποιείται για διαδικασίες που δεν μπορούν να επαναληφθουν με την ανανέωση της ιστοσελίδας χωρίς την έγκριση του χρήστη διαφορετικά θα υπήρχαν προβλήματα διαρροής προσωπικών πληροφοριών. Για παράδειγμα η μεταφορά χρηματικού ποσού από έναν τραπεζικό λογαριασμό σε έναν άλλο δεν θα πρέπει να επαναλαμβάνεται με την ανανέωση της σελίδας χωρίς την ύπαρξη προειδοποιητικού μηνύματος.Επιπλεόν σε αντιδιαστολή με τα αιτήματα που υλοποιούνται με την μέθοδο GET, στην μέθοδο POST δεν είναι εμφανή τα δεδομένα του αιτήματος στο URL.

Το πού αποθηκεύονται τα δεδομένα που αποστέλονται μέσω ενός αιτήματος με την μέθοδο POST δεν είναι διακριτό στο URL που απευθύνεται το αίτημα. Στις περισσότερες περιπτώσεις το τελευταίο τμήμα του URL του αιτήματος θα περιγράφει την διεύθυνση μιας εφαρμογής του server που είναι υπεύθυνη για την επεξεργασία και την αποθήκευση των δεδομένων που παρέχονται μέσω του POST.

Τα αιτήματα POST μπορούν να ακολουθήσουν και αυτά την ίδια σύνταξη με τα αιτήματα GET. Μέσω της Javascript μπορούμε να στείλουμε αιτήματα POST χάρη στην τεχνολογία Asynchronous Javascript XML (AJAX) [1] και στο αντικείμενο XMLHTtpRequest.

```
function submit() {
  var data="Hello World!"
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function show() {
    if (xhttp.readyState == 4 && xhttp.status == 200) {
      alert(xhttp.responseText);
    }
  };
  xhttp.open("POST", "http://localhost/server.php", true);
  xhttp.send(data);
}
```

Λόγω της μεγάλης χρησιμότητας της αποστολής αιτημάτων POST στον server η διαδικασία αυτή τυποποιήθηκε και πλέον υπάρχουν στοιχεία της γλώσσα HTML που την αναλαμβάνουν εξ'ολοκλήρου. Αυτά είναι τα στοιχεία form που σε συνδιασμό με τα στοιχεία input συγκεντρώνουν την πληροφορία που εισάγει ο χρήστης και την στέλνουν στον server. Η σύνταξη αυτών των στοιχείων σε HTML είναι η ακόλουθη:

```
<form action="http://localhost/profile/server.php" method="POST">
    <input name="field1" type="text" />
    <input name="field2" type="text" />
    <input type="submit" name="submit" value="Save Data">
</form>
```

Το form προδιορίζει τις πληροφορίες γύρω από την αποστολή του αιτήματος με την μέθοδο POST. Η παράμετρος action προσδιορίζει την τοποθεσία που πρέπει να σταλούν οι πληροφορίες στον server. Όπως αναφέρθηκε και παραπάνω τα δεδομένα δεν αποστέλονται άμεσα σε ένα αρχείο του server αλλά σε ένα server-side scipt που είναι υπεύθυνο για την ανάγνωση και αποθήκευσή τους. Η παράμετρος method προσδιορίζει την μέθοδο του HTTP πρωτοκόλλου που θα χρησιμοποιηθεί.Μερικές ακόμα παράμετροι που μπορεί να δεχτεί το πεδίο form είναι:

accept-charset

Προσδιορίζει την κωδικοποίηση του αιτήματος προς αποστολή και σε περίπτωση που παραληφθεί παίρνει την ίδια τιμή με τα υπόλοιπα στοιχεία της ιστοσελίδας.

autocomplete

Προσδιορίζει αν ο browser πρέπει να επέμβει για την ολοκλήρωση του αιτήματος.

name

Προσδιορίζει το όνομα του form. Αυτή η παράμετρος χρησιμοποιείται κυρίως για την διευκόλυνση της επεξεργασίας των στοιχείων της HTML σελίδας από την Javascript

target

Προσδιορίζει την τοποθεσία στη οποία πρέπει να σταλεί η απάντηση του server και σε περίπτωση παράλειψης λαμβάνει τιμή "_self"

Το input στοιχειοθετεί την είσοδο του χρήστη και διακρίνεται σε τρείς τύπου: στον

τύπο text που όπως στο παράδειγμα χρησιμοποιείται για να δημιουργήσει πλέσια κειμένου, στον τύπο radio που δημιουργεί μια λίστα πολλαπλής επιλογής και στον τύπο submit που στέλνει ολόκληρο το form στην διεύθυνση που δηλώνεται από την παράμετρο action στον server.

2.2 Αποθήκευση σε Τοπικά Αρχεία

2.2.1 Cookies

Τα cookies είναι μικρά αρχεία δεδομένων κειμένου που αποθηκεύονται προσωρινά στον browser του χρήστη καθώς αυτός πλοηγείται σε ιστοσελίδες. Η επικοινωνία αυτών των δύο συστημάτων διακόπτεται μόλις ο server στείλει την ιστοσελίδα που ζητήθηκε από τον browser και έτσι ο πρώτος δεν μπορεί να ξέρει την δραστηριότητα του δεύτερου. Χάρη στα cookies η ιστοσελίδα καθοδηγεί τον browser να αποθηκεύσει τις πληροφορίες του cookie και στην συνέχεια να το στείλει πίσω στον server σε κάθε μεταγενέστερη αίτηση υπό κάποιο καθορισμένο πρωτόκολλο. Για παράδειγμα οι περισσότερες ιστοσελίδες που απαιτούν είσοδο θα κατασκευάσουν ένα cookie με τα διαπιστευτήρια του χρήστη μόλις αυτά έχουν ελεγχθεί και τότε ο χρήστης θα είναι ελεύθερος να περιηγηθεί σε όλα τα μέρη του ιστότοπου εφ' όσον το cookie παραμένει αποθηκευμένο στον browser. Επίσης τα cookies μπορούν να αποθηκεύουν μέχρι και τις ενέργειες του ίδιου του χρήστη (πχ ποιά κουμπιά πάτησε, ποιές ιστοσελίδες επισκέπτηκε) καθώς και πληροφορίες γενικότερου χαρακτήρα όπως στοιχεία που συμπλήρωσε ο χρήστης σε πεδία της ιστοσελίδας.

Τα cookies χωρίζονται σε κατηγορίες ανάλογα με σκοπιμότητα που αυτά εξυπηρετούν.Οι πιο γνωστές κατηγορίες είναι οι παρακάτω:

Authentication cookies

Η χρήση αυτών των cookies είναι η πιο διαδεδομένη διότι μέσω αυτών οι web servers μπορούν να πληροφορηθούν σχετικά με το εάν ένας χρήστης έχει συνδεθεί στην ιστοσελίδα ή όχι και με ποιόν λογαριασμό έχει συνδεθεί. Χωρίς αυτό τον μηχανισμό θα ήταν πιθανή η διαρροή σελιδών με προσωπικά στοιχεία ή θα ήταν αναγκαία η απόδειξη της ταυτότητας του χρήστη ζητώντας του να συνδεθεί κάθε φορά εκ νέου στο σύστημα. Το εάν η ύπαρξη αυτού του τύπου cookies θέτει σε κίνδυνο την ασφάλεια του χρήστη ή όχι εξαρτάται άμμεσα από την ασφάλεια της ιστοσελίδας που τα φέρει και του browser που τα περιέχει καθώς και από την κρυπτογράφηση των ίδιων των δεδομένων του cookie.

Session cookie

Τα session cookies έχουν χρόνο ζωής όσο ο χρήστη πλοηγείται στην ιστοσελίδα που τα περιέχει. Οι web browsers διαγράφουν τα session cookies όταν ο χρήστης κλείσει την εφαρμογή ή την καρτέλα που τα περιέχει. Τα cookies αυτά δεν έχουν ημερομη-

νία λήξης και γι'αυτό ο web browser μπορεί να τα αναγνωρίσει και να τα διαγράψει.

Persistent cookies

Σε αντιδιαστολή με τα session cookies τα persistent cookies έχουν ημερομηνία λήξης η οποία ορίζεται ανάλογα με τον σκοπό που πρέπει να εξυπηρετήσουν. Μέχρι να λήξει το cookie οι πληροφορίες που φέρει μεταδίδονται στον web server κάθε φορά που ο χρήστης μεταβαίνει στην ιστοσελίδα που τα φέρει ή κάθε φορά που ο χρήστης βρίσκεται σε σελίδα που συνδέεται με την ιστοσελίδα αυτή.Γι'αυτό το λόγο τα persistent cookies αναφέρονται συχνά ως tracking cookies (cookies παρακολούθησης) γιατί χρησιμοποιούνται κυρίως από διαφημιστές προκειμένου να παρακολουθούν τις προτιμήσεις των καταναλωτών σε μια μεγάλη περιόδο χρόνου.

Secure cookies

Ένα secure cookie μπορεί να μεταφέρει την πληροφορία του στο server μόνο σε κρυπτογραφημένες συνδέσεις με ασφαλές πρωτόκολλο (HTTPS). Ένα cookie μπορεί να μετατραπεί σε secure cookie εάν προσθέσουμε την μεταβλητή secure κατά τον ορισμό του.

HTTPOnly cookies

Τα HTTPOnly cookies δεν μπορούν να γίνουν προσβάσιμα από την Javascript μέσω του document.cookie. Αυτά τα cookies έχουν σχεδιαστεί ως μέτρο ασφάλειας για να βοηθήσουν στην πρόληψη των επιθέσεων μέσω cross-site scripting που διαπράττονται από την κλοπή cookies μέσω Javascript.

Μπορούμε να διαχειριστούμε τα cookies μέσω της Javascript με την παρακάτω σύνταξη:

```
1  //Create the cookie
2  document.cookie = "cookieName=cookieNameValue; expires=Fri, 08 Jul 2016
       16:00:00 UTC"
3
4  //Read a cookie
5  var x = document.cookie;
6
7  //Change the cookie
8  var newValue="cookieNameValue2"
9  document.cookie = "cookieName="+newValue+"; expires=Fri, 08 Jul 2016
       16:00:00 UTC"
10
11 //Delete cookie
12 document.cookie = "cookieName=; expires=Thu, 01 Jan 1970 00:00:00 UTC";
```

Η Javascript μπορεί να δημιουργήσει cookies μέσω της document.cookie ιδιότητας. Δίνουμε στο cookie την τιμή που περιγράφεται από το παραπάνω string η οποία δηλώνει το όνομα με το οποίο μπορούμε να αναφερόμαστε στο cookie αυτό και την τιμή που αυτό φέρει. Επιπλέον κατά την δημιουργία του cookie δηλώνουμε και την ημερο-

μηνία λήξης του. Τέλος κατά τον ορισμό του cookie μπορούν να χρησιμοποιηθούν και τα χαρακτηριστικά domain που αναφέρεται στο πλήθος των ιστοσελίδων που φέρουν αυτό το cookie (πχ yahoo.com), path που αναφέρεται στην συγκεκριμένη σελίδα του domain για την οποία έχει δημιουργηθεί το cookie(πχ /blog) και secure που όπως αναφέρθηκε παραπάνω μεταφέρει πληροφορία μόνο για HTTPS πρωτόκολα ανταλλαγής πληροφοριών.

Η ανάγνωση και η αλλαγή της τιμής του cookie μπορούν να πραγματοποιηθούν εύκολα και άμμεσα. Στην πρώτη περίπτωση απλά αναθέτουμε την τιμή του cookie σε μια μεταβλητή και στην δεύτερη περίπτωση επεμβαίνουμε άμμεσα στην τιμή του cookie που έχουμε ορίσει και επιβάλλουμε την επιθυμητή αλλαγή. Για την διαγραφή του cookie με όνομα cookieName απλά αρκεί να θέσουμε την ημερομηνία λήξης του σε μια περασμένη ημερομηνία δηλαδή να αναγκάσουμε το cookie μας να λήξει και στην συνέχεια ο browser το διαγράφει από μόνος του.

Παρατίθεται παράδειγμα κώδικα που δημιουργεί cookies με τιμές που εισάγει ο χρήστης σε πεδίο κειμένου και τα οποία στην συνέχεια μπορεί να διαγράψει.

```
1  var count=0;
2
3  function setCookie() {
4      count++
5      var d = new Date();
6      var cvalue=\$("input").val()
7       var exdays=30
8       var cname="test"+count
9      d.setTime(d.getTime() + (exdays*24*60*60*1000));
10      var expires = "expires=" + d.toGMTString();
11     document.cookie = cname+"="+cvalue+"; "+expires;
12
13 }
14
15 function alertCookie() {
16   alert(document.cookie);
17 }
18
19 function deleteAllCookies() {
20     var cookies = document.cookie.split(";");
21
22     for (var i = 0; i < cookies.length; i++) {
23       var cookie = cookies[i];
24       var eqPos = cookie.indexOf("=");
25       var name = eqPos > -1 ? cookie.substr(0, eqPos) : cookie;
26       document.cookie = name + "=;expires=Thu, 01 Jan 1970 00:00:00 GMT";
27     }
28
```

```
29      count=0;
30  }
```

Η διαδικασία setCookies( ) δημιουργεί κάθε φορά που την καλεί ο χρήστης ένα καινούργιο cookie.Την πρώτη φορά που την καλεί ο χρήστης δημιουργεί ένα cookie που λήγει σε 30 ημέρες από την τρέχουσα ημέρα και ώρα , με όνομα test1 και τιμή την τιμή που έχει εισάγει ο χρήστης στο πεδίο input. Η διαδικασία επαναλαμβάνετε αυτούσια κάθε φορά και τα cookies που δημιουργούνται είναι persistent cookies όπως αυτά περιγράφηκαν παραπάνω.

Αν ο χρήστης καλέσει την διαδικασία deleteAllCookies( ) τα cookies της σελίδας μετατρέπονται σε ένα διάνυσμα χάρη στη μέθοδο split(;), η οποία αντικαθιστά τα ';' που χρησιμοποιοούνται στο εσωτερικό των cookies με ',' και στην συνέχεια αποθηκεύονται στην μεταβλητή cookie. Για κάθε στοιχείο του διανύσματος cookie αποκόπτουμε το κομμάτι που φέρει την πληροφοριά για το όνομα και την τιμή του cookie και επεμβαίνουμε στο τμήμα που ορίζει την ημερομηνία λήξης του cookie και το αναγκάζουμε να λήξει βάζοντας μια παλαιότερη ημερομηνία.Τέλος η διαδικασία alertCookie( ) ανοίγει ένα προειδοποιητικό παράθυρο στον browser και εμφανίζει τις τιμές που έχουν λάβει τα cookies που έχουν δημιουργηθεί μέχρι τώρα.

Το θέμα τις χρήσης των cookies για εφαρμογές Javascript οι οποίες υλοποιούνται τοπικά (χωρίς έκθεση στο δίκτυο και κατ'επέκταση χωρίς επικοινωνιά με server) είναι ιδιαίτερα αμφιλεγόμενο. Ο web browser Google Chrome δεν υποστηρίζει αυτή την δυνατότητα διότι μπορεί να προκαλέσει θέματα ασφάλειας και να επιρρεάσει αρνητικά την σωστή λειτουργία της εφαρμογής.Παρόλα αυτά, οι browsers Internet Explorer και Mozilla Firefox υποστηρίζουν τα cookies από τοπικές εφαρμογές.

2.2.2 Web Storage & Local Storage

Ως Web Storage χαρακτηρίζονται οι μέθοδοι του λογισμικού των web applications που είναι υπεύθυνες για την αποθήκευση δεδομένων στον browser. Όπως και στην περίπτωση των cookies τα δεδομένα δεν αποθηκεύονται κάθε φορά εκ νέου αλλά ανανεώνονται από την προηγούμενη τους κατάσταση. Βασική διαφορά των cookies και του web storage είναι ότι το δεύτερο έχει περισσότερες δυνατότητες όσον αφοράν τον όγκο των αποθηκευμένων δεδομένων και τον χρόνο που αυτά μπορούν να παραμείνουν στην εφαρμογή. Τέλος ενώ τα δεδομένα των cookies μπορούν να διαβαστούν άμεσα από τον server , η επικοινωνία του Web Storage και του server δεν είναι άμεση και μπορεί να επιτευχθεί με την εφαρμογή client-side scripts.

Το Web Storage παρέχει δύο διαφορετικά αντικείμενα που υλοποιούν διαφορετικούς μηχανισμούς αποθήκευσης δεδομένων: το sessionStorage και το localStorage. Ο πρώτος διατηρεί τα δεδομένα μέχρις ότου να τερματιστεί η λειτουργία της εφαρμογής ενώ ο δεύτερος δεν επηρεάζεται από τον τερματισμό της εφαρμογής.

Τα αντικείμενα αυτά μπορούμε να τα διαχειριστούμε με Javascript μέσω της παρακάτω σύνταξης:

```
//Store
localStorage.variableName= variableValue
//Retrieve
localStorage.getItem('variableName' )
//Delete
localStorage.removeItem('variableName')
```

Ομοίως και για το αντικείμενο sessionStorage.

Παρατίθεται ένα παράδειγμα κώδικα που κάνει χρήση του localStorage.

```
function reset()
{
    localStorage.removeItem("clickcount");
}
function clickCounter() {
    if(typeof(Storage) !== "undefined") {
        if (localStorage.clickcount) {
            localStorage.clickcount = Number(localStorage.clickcount)+1;
        } else {
            localStorage.clickcount = 1;
        }
        document.getElementById("result").innerHTML = "You have clicked
            the button " + localStorage.clickcount + " time(s).";
    } else {
        document.getElementById("result").innerHTML = "Sorry, your
            browser does not support web storage...";
    }
}
```

Στο παραπάνω παράδειγμα δημιουργείται η μεταβλητή clickCount του αντικειμένου localStorage. Η μεταβλητή αυτή αυξάνεται κατά 1 κάθε φορά που ο χρήστης κάνει click στο κουμπί "'Click Me!" και αποθηκεύεται στον browser. Όσες φορές και αν πατήσει ο χρήστης το κουμπί η μεταβλητή θα συνεχίσει την καταμέτρηση από την τελευταία αποθηκευμένη τιμή. Επιπλέον αν ο χρήστης τερματίσει την εφαρμογή του browser και στην συνέχεια την ανοίξει και πατήσει εκ νέου το κουμπί "Click Me!" η καταμέτρηση θα συνεχίσει κανονικά από την τελευταία καταγεγραμμένη τιμή. Τέλος αν το παραπάνω παράδειγμα υλοποιούνταν με χρήση του αντικειμένου sessionStorage κάθε φορά που ο χρήστης θα τερμάτιζε την εφαρμογή και επέστρεφε αργότερα σε αυτή η καταμέτρηση θα ξεκινούσε πάλι από το 1.

Τέλος αναφέρεται πως το Web Storage υποστηρίζεται από τις εξής εκδόσεις Web Browser και τις μεταγενέστερες τους: Google Chrome 4, Mozilla Firefox 3.5 , Internet Explorer 8.0, Safari 4.0 και Opera 11.5.

2.2.3 Automation & ActiveXObject

Με τον όρο Automation αναφερόμαστε στην τεχνολογία που έχει αναπτυχθεί από την εταιρεία Microsoft με στόχο την αυτόματη σύνδεση και ενσωμάτωση αντικειμένων σε έγγραφα και σε άλλα αντικείμενα.Αυτός ο μηχανισμός δίνει τη δυνατότητα σε πακέτα λογισμικού των Windows να εκθέτουν τα αντικείμενά τους προκειμένου αυτά να μπορούν να τροποποιηθούν από άλλες εφαρμογές.

Τα αντικείμενα που εφαρμόζουν αυτή την τεχνολογία είναι τα ActiveX τα οποία δημιουργήθηκαν από την Microsoft το 1996. Ενώ ο αρχικός στόχος της δημιουργία αυτών των αντικειμένων ήταν να λειτουργούν ανεξαρτήτως λογισμικού συστήματος στην πράξη μπορούν να αλληλεπιδράσουν αποτελσματικά μόνο για το λογισμικό των Windows.Τέλος, τα αντικείμενα ActiveX θέτουν συνήθως θέματα ασφάλειας λόγω της άμεσης επαφής τους με τοπικά αρχεία του υπολογιστή και για αυτό το λόγο οι σχετικές με την ασφάλεια ρυθμίσεις θα πρέπει να τροποποιηθούν, διαφορετικά το αντικείμενο θα παραμείνει ανενεργό.

Τα αντικείμενα ActiveX μπορούν να ορισθούν σε Javascript με την παρακάτω σύνταξη[3]:

```
1 var newObj= new ActiveXObject(servername.typename,location)
```

servername

Υποχρεωτικό πεδίο τύπου string που δηλώνει την εφαρμογή που παρέχει το αντικείμενο.(πχ. Excel, Word, Scripting)

typename

Υποχρεωτικό πεδίο τύπου string που δηλώνει τον τύπο ή την κλάση του αντικειμένου που θα δημιουργηθεί.(πχ. Application, Chart, FileSystemObject)

location

Προαιρετικό πεδίο τύπου string που δηλώνει την τοποθεσία στην οποία το αρχείο θα δημιουργηθεί.

Παρατίθεται παράδειγμα κώδικα σε Javascript για τον Internet Explorer:

```
1  function writeToFile(d1, d2){
2      var fso = new ActiveXObject("Scripting.FileSystemObject");
3      var fh = fso.OpenTextFile("logfile.txt", 8, false, 0);
4      fh.WriteLine(d1 + ',' + d2);
5      fh.Close();
6      }
7
8     var submit = document.getElementById("submit");
9
10     submit.onclick = function getTheValues () {
11     var id      = document.getElementById("id").value;
12     var content = document.getElementById("content").value;
13     writeToFile(id, content);
14 }
```

Για την κατανόηση του παραδείγματος είναι απαραίτητη η ανάλυση της σύνταξης της μεθόδου 'OpenTextFile'.

```
1 object.OpenTextFile(filename,iomode,create,format)
```

filename

Υποχρεωτικό πεδίο τύπου string που δηλώνει το όνομα του αρχείου στο οποίο επεμβαίνει η μέθοδος

iomode

Προαιρετικό πεδίο που δηλώνει αν η μέθοδος θα εισάγει στοιχεία στο αρχείο ή θα εξάγει από αυτό και λαμβάνει τιμές 8 ή 1 αντίστοιχα.

create

Προαιρετικό πεδίο τύπου boolean που δηλώνει αν το αρχείο υπάρχει ή αν θα πρέπει να δημιουργηθεί (false ή true αντίστοιχα).

format

Προαιρετικό πεδίο που δηλώνει την μορφοποίηση που πρέπει να εφαρμοστεί στο αρχείο που επεξεργαζόμαστε και λαμβάνει τιμές 2, 1 ή 0 για την προκαθορισμένη μορφοποίηση του συστήματος, για μορφοποίηση Unicode και για μορφοποίηση ASCII αντίστοιχα.

Στο παραπάνω παράδειγμα με το πάτημα του κουμπιού 'Submit' ενεργοποιείται η διαδικασία 'getTheValues' που αποθηκεύει τις τιμές των πεδίων 'id' και 'content' της HTML. Στην συνέχεια καλεί την διαδικασία 'writeToFile' με ορίσματα τις τιμές αυτές. Η 'writeToFile' ενεργοποιεί το ActiveXObject της εφαρμογής 'Scripting' το οποίο θα είναι τύπου 'FileSystemObject'. Το αντικείμενο αυτό καλεί την μέθοδο 'OpenTextFile' η οποία ανοίγει το υπάρχον αρχείο 'logfile.txt' για να εισάγει σε αυτό κείμενο υπό την μορφοποίηση ASCII, δημιουργεί δηλαδή έναν δίαυλο επικοινωνίας με το αρχείο 'logfile.txt'. Έπειτα μέσω της μεθόδου 'writeLine' γράφει σε αυτό τις τιμές που είχε λάβει ως ορίσματα και τέλος με την μέθοδο 'close' κλείνει το αρχείο.

Οι εκδόσεις του Internet Explorer που επιτρέπουν τις εφαρμογές Javascript με χρήση αντικειμένων ActiveX ξεκινούν από τον IE 6 και μετά.

3 Αρχεία epub3

3.1 Γενικά στοιχεία αρχείων Epub

Συχνά συγχεόμενος με τον όρο ebook , ο όρος epub χρησιμοποιείται για να περιγράψει το format (διαμόρφωση) σύμφωνα με το οποίο παρουσιάζονται έγγραφα σε ηλεκτρονική μορφή. Όπως το ebook, το όνομα epub προέρχεται από τη σύντμηση των όρων Elecrtonic PUBlication. Η επιλογή του πιο γενικού όρου "publication" (δημοσίευση) ήταν σκόπιμη και επιλέχτηκε προκειμένου τα αρχεία που φέρουν το format epub να είναι ευρέος τύπου από περιοδικά, εφημερίδες, ημερολόγια μέχρι έγγραφα γραφείου, συμβόλαια και άλλα. Σχεδόν οποιοσδήποτε τύπος εγγράφου που διανέμεται ηλεκτρονικά μπορεί να παρουσιαστεί ως EPUB.

Πιο πρακτικά, το epub καθορίζει τόσο το format του αρχείου όσο και τον τρόπο που τα συστήματα ανάγνωσης το ανακαλύπτουν και το αποδίδουν στους αναγνώστες.

Προκειμένου να γίνει αντιληπτή η δομή ενός αρχείου epub, παρατίθεται μια σύντομη περιγραφή των δομικών του στοιχείων του [4] :

- **Φάκελος OEBPS(Open Publication Structure)**

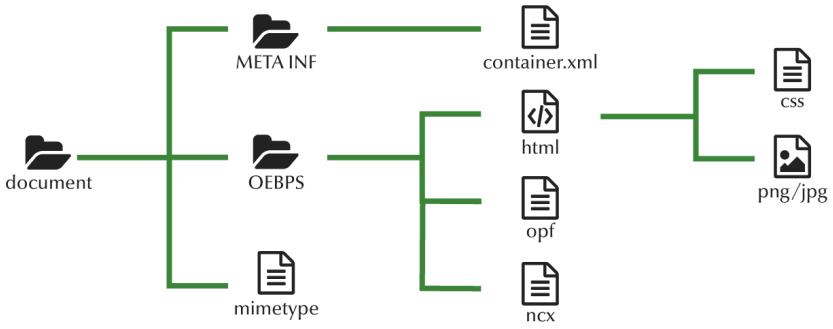
  Περιέχει τα αρχεία:

  1. *HTML5, CSS, Javascript, img, SVG* που συνθέτουν το περιεχόμενο του βιβλίου.
  2. *content.opf* (open packaging format) που περιέχει πληροφορρίες για τον συγγραφεά, τον τίτλο και την σειρά εμφάνισης των αρχείων του βιβλίου καθώς και σε ποια γλώσσα είναι γραμμένο.
  3. *toc.ncx*(navigation control XML) που είναι υπεύθυνο για την κατασκευή των περιεχομένων του βιβλίου με βάση την σειρά εμφάνισης των αρχείων όπως αναφέρεται στο προηγουμένως.

- **Φάκελος META-INF**

  Εδώ εμπεριέχεται το αρχείο container.xml το οποίο ¨κατευθύνει¨ το σύστημα ανάγνωσης στο φάκελο OEBPS και πιο συγκεκριμένα στο αρχείο content.opf. Τέλος για την αναγνώριση του βιβλίου από το λειτουργικό σύστημα ηλεκτρονικής

ανάγνωσης ως αρχεία σε format .epub προστίθεται ένα αρχείο κειμένου ASCII με όνομα mimetype.



Σύμφωνα με την παραπάνω λίστα μπορεί να γίνει αντιληπτό πώς λειτουργούν τα συστήματα ανάγνωσης (Ebook Readers).Ακολουθούν την αντίθετη διαδρομή δηλαδή εξετάζουν τον περιέκτη συμπίεσης (zip container), προσδιορίζουν ότι είναι ένα EPUB (mimetype), βρίσκουν το αρχείο toc.ncx και ανακαλύπτουν πώς να προσφέρουν τις πηγές στους αναγνώστες μέσω του αρχείου content.opf.

Η πλειοψηφία των τεχνολογιών που χρησιμοποιούνται για το EPUB έχουν αναπτυχθεί από το International Digital Publishing Forum (IDPF), ο οργανισμός που υποστηρίζει τις προδιαγραφές του EPUB και μέλη του οποίου είναι εκδότες, διανομείς και προγραμματιστές αναγνωστικών συστημάτων. Οι περισσότερες προδιαγραφές είναι διεθνώς αναγνωρισμένες και θα πρέπει να είναι γνωστές στους περισσότερους αναγνώστες.

Μερικά από τα προβλήματα που αντιμετώπιζαν τα προηγούμενα format την έλλειψη διαδραστικού περιεχομένου ,την αδυναμία καλύτερης υποστήριξης στην παγκόσμια γλώσσα και την ποιότητα των χαρακτηριστικών προσβασιμότητας. Επομένως, έπρεπε να γίνει μια αναθεώρηση στα πλαίσια των απαιτήσεων του IDPF.

Η τελευταία τεχνολογία στον τομέα των δημοσιεύσεων είναι τα αρχεία epub3 τα οποία έχουν επεκτείνει σημαντικά την λίστα με τις διαθέσιμες τεχνολογίες που χρησιμοποιούσαν οι προγενέστερες μορφές epub αρχείων. Μερικές από τις τεχνολογίες που χρησιμοποιούνται από αυτά τα αρχεία είναι η γλώσσα προγραμματισμού HTML5 επιτρέποντας την ύπαρξη βίντεο και ήχου στη δημοσίευση,JavaScript επιτρέποντας την διαδραστικότητα χρήστη-ebook και την αυτοματοποίηση πολλών διαδικασιών ,CSS3 και SMIL 3 διευκολύνοντας τον συγχρονισμό ήχου εικόνας και την δημιουργία ¨ομιλούντων¨ βιβλίων.

Οι ίδιες ανοιχτές προδιαγραφές που ισχύουν για τον Διεθνή Παγκόσμιο Ιστό είναι κατάλληλες για ψηφιακή ανάγνωση και αναπόσπαστο μέρος για τη μακροπρόθεσμη επιβίωση του συγκεκριμένου format[5]. Είναι κατάλληλες, γιατί η γλώσσα HTML έχει κυριαρχήσει στο χώρο λόγο της δυνατότητας της να παράγει περιεχόμενο που προ-

σαρμόζεται εύκολα στο οπτικό πεδίο διαθέσιμο σε κάθε συσκευή, σε αντίθεση με τις σταθερές διαμορφώσεις όπως το PDF που στηρίζεται στην ακριβή τοποθέτηση.

Το epub είναι συνδεμένο με τη δομή του συστήματος πλοήγησης (browser stack) έχει τεχνικά πλεονεκτήματα που το βοηθούν να ευδοκιμήσει, ενώ άλλες διαμορφώσεις έχουν έλθει και παρέλθει. Επιπλέον, η CSS του δίνει τη δυνατότητα να απευθύνεται σε ένα διεθνές κοινό και να παρέχει μια πλούσια εμπειρία θέασης.

Αν και η JavaScript ήταν πάντα μέρος της δομής του συστήματος πλοήγησης, το epub3 τελικά επανέφερε αυτό το επίμαχο χαρακτηριστικό στο χώρο των ebook για να διευκολύνει τη διαδραστικότητα στα έγγραφα .

3.1.1 Το ζήτημα της διαδραστικότητας

Ο ενσωματωμένος πηγαίος κώδικας μιας γλώσσας προγραμματισμού, είναι ένα προαπαιτούμενο για τις περισσότερες μορφές ενός διαδραστικού ebook. Ενώ τα epub3 αρχεία επιτρέπουν στους συντάκτες να ενσωματώσουν ένα επιπρόσθετο εκτελέσιμο αρχείο (Object), στην πλειονότητα των περιπτώσεων, τα διαδραστικά αρχεία θα δημιουργηθούν με τη χρήση του πηγαίου κώδικα . Επειδή η JavaScript είναι η κατ' εξοχήν γλώσσα προγραμματισμού scripting για SVG και HTML5, τα epub3 αρχεία μπορούν να χαρακτηριστούν διαθέσιμα προς scripting, μόνο εάν περιέχουν κώδικα JavaScript. Τέλος δεν υπάρχει κανόνας που να καθορίζει ποιες εκδόσεις της JavaScript είναι απαραίτητο να χρησιμοποιηθούν για αυτό το σκοπό.

Το θέμα της διαδραστικότητας των ηλεκτρονικών εκδόσεων είναι ιδιαίτερα αμφιλεγόμενο. Για πολλούς αναγνώστες, η αμεταβλητότητα ενός βιβλίου είναι ένα θετικό χαρακτηριστικό γνώρισμα παρά ένα μειονέκτημα. Ένα τυπωμένο βιβλίο δεν απαιτεί τίποτα παραπάνω από το χρήστη, παρά την πλήρη προσοχή του. Δεν προτρέπει τον αναγνώστη να κάνει κλικ ή να σχολιάσει. Υπόσχεται απόλυτη συγκέντρωση στο κείμενο, ως ένας άμεσος αγωγός που οδηγεί στις σκέψεις του συγγραφέα.

Ωστόσο, υπάρχουν πολλές απόψεις που χαρακτηρίζουν τη στατική φύση του παραδοσιακού βιβλίου, ένα αποτέλεσμα της τεχνολογίας γύρω από την εκτύπωση και τίποτα παραπάνω. Βιβλία που στοχεύουν να διδάξουν πολύπλοκα θέματα, θα είχαν την δυνατότητα να γίνουν πιο κατανοητά και ευχάριστα στην ανάγνωση, από την ευκαιρία που θα παρείχαν στους αναγνώστες να αλληλοεπιδράσουν με το κείμενο. Νέοι τρόποι αφήγησης θα μπορούσαν να ενθαρρύνουν τον αναγνώστη να ανακαλύψει νέα μονοπάτια, ή να του επιτρέψει να "σκάψει" πιο βαθιά στο κείμενο, αποκαλύπτοντας κρυφά κίνητρα του συγγραφέα.

Ένα αρχείο epub3 παρέχει τη δυνατότητα στο χρήστη να κάνει όλα τα παραπάνω. Ωστόσο, υπάρχουν κάποια σημεία που χρήζουν την προσοχή κάθε συγγραφέα. Η διαδραστικότητα ενός ηλεκτρονικού βιβλίου παραμένει ένα πρωτοποριακό χαρακτηριστικό. Όσο περισσότερο αποκλίνει από ένα παραδοσιακό βιβλίο, τόσο λιγότερο πι-

θανό είναι να είναι πλήρως συμβατό με τα υπάρχοντα συστήματα ανάγνωσης. Πολλά συστήματα ανάγνωσης αρχείων epub3 δεν θα υποστηρίξουν ποτέ την διαδραστικότητα στα αρχεία τους και δεν υπάρχει κάποιος κανόνας που να υποχρεώνει σε μερική ή υπό όρους υποστήριξη αυτών.

Οι προδιαγραφές ενός epub3 αρχείου ορίζουν δύο μοντέλα που επηρεάζουν το πεδίο εφαρμογής του εγγράφου, όπου το κείμενο μπορεί να τροποποιείτε δυνητικά από τον χρήστη. Όσο μεγαλύτερο το πεδίο της αλληλεπίδρασης, τόσο λιγότερο πιθανό το epub3 αρχείο να είναι λειτουργικό.

Τα ebooks, σε αυτό το σημείο, παρουσιάζουν μια ιδιαίτερη πρόκληση για δύο λόγους[6]:

- Τα συστήματα ανάγνωσης epub3 αρχείων, χρησιμοποιούν διαφορετικά μοντέλα προβολής του περιεχομένου. Ενώ συνήθως αυτό είναι μια επιλογή μεταξύ προβολής του κειμένου σε αριθμημένες σελίδες, όπως σε ένα βιβλίο ή με δυνατότητα κύλισης του κειμένου, όπως στο διαδίκτυο, μπορούν επίσης να περιλαμβάνουν δυνατότητα ανάγνωσης ηχητικά ή σε κώδικα Braille, εξειδικευμένα μεγέθη παραθύρων ανάγνωσης (σειριακή παρουσίαση αποσπασμάτων ή μονές γραμμές) ή πιο ασυνήθιστων διαμορφώσεων όπως προβολή δύο σελίδων ή ακόμα και τριών ή τεσσάρων. Οποιαδήποτε από αυτές τις διατάξεις ενδέχεται να παρουσιάσει απροσδόκητη συμπεριφορά όταν το DOM (Document Object Model) του αρχείου τροποποιηθεί.
- Σε αντίθεση με τα προγράμματα περιήγησης, τα συστήματα ανάγνωσης ebook έχουν μια σειρά από μοναδικές λειτουργίες προκειμένου να ανατρέξουν σε στοιχεία ελέγχου πλοήγησης, ρυθμίσεις ή metadata. Ένα συντάκτης δεν μπορεί να ξέρει ακριβώς ποιες ενέργειες του χρήστη μπορεί να παραληφθούν , από το ίδιο το σύστημα, λόγο των παραπάνω λειτουργιών ούτε την ακριβή χρονική σειρά στην οποία συνέβησαν αυτές οι παραλήψεις.

Η Προοδευτική ενίσχυση αποτελεί μία αρχή σχεδιασμού στην οποία ο στόχος είναι να αναπτυχθεί η πιο προσβάσιμη έκδοση του περιεχομένου αρχικά, ως την κύρια έκδοση, και σταδιακά να προστίθενται ενισχυτικά επίπεδα που θα βασίζονται στις δυνατότητες του συστήματος ανάγνωσης, της συσκευής και του τελικού χρήστη. Ενώ το αποτέλεσμα μιας σταδιακά βελτιωμένης εργασίας μπορεί να διακριθεί από το εάν ολοκληρώθηκε επιτυχώς μέσω μιας αρμονικά εκτελεσμένης προσέγγισης, η πρόθεση πίσω από την προοδευτική ενίσχυση είναι να προβληθεί η προσβάσιμη έκδοση του περιεχομένου ως η κανονική, και όχι να προβάλλετε ο σχεδιασμός μιας οπτικής έκδοσης, με βαρύ περιεχόμενο, υψηλής ευκρίνειας ως το «πραγματικό» περιεχόμενο με αποτέλεσμα την υποβάθμιση της παρουσίαση σε μια προσβάσιμη έκδοση εκ των υστέρων.

Η χρήση των scripts σε αρχεία προηγούμενης γενιάς ήταν αποθαρρυντική, αν και δεν ήταν ρητά απαγορευμένη. Παρόλα αυτά, δεν υπάρχουν πολλές δυνατότητες όσον

αφορά το scripting σε συστήματα ανάγνωσης μόνο αρχείων με format epub2 και παλαιότερα. Η αναθεώρηση με το epub3 παρουσίασε μια ευκαιρία να εξεταστούν εκ νέου θέματα που προέκυψαν ή που είχαν μείνει ανοικτά από την προηγούμενες αναθεωρήσεις και να βρεθούν ,αν είναι δυνατόν, νέες λύσεις. Εν ολίγοις, η εξέλιξη του epub δεν σταματά με την πρόσφατη αναθεώρηση και, ήδη, έχουν ξεκινήσει βελτιώσεις.

3.1.2 Editors & Viewers

Η κατασκευή epub3 αρχείων μπορεί να γίνει είτε χειροκίνητα με την κατασκευή όλων των φακέλων που αναφέρθηκαν παραπάνω είτε με την χρήση κατάλληλων εφαρμογών σύνταξης (epub3 Editors). Αυτά τα εργαλεία εμφανίζουν μεγάλες διαφορές ως προς τις προαπαιτούμενες γνώσεις που πρέπει να έχει ο χειριστής τους και ως προς την συνδρομή που πρέπει να καταβάλει κάποιος για να τα χρησιμοποιήσει.

Οι πιο διαδεδομένοι epub3 Editors είναι οι ακόλουθοι:

Sigil

Το εργαλείο Sigil είναι ένα έργο ανοιχτού κώδικα για κατασκευή ηλεκτρονικών βιβλίων σε format epub. Δημιουργήθηκε το 2009 από τον Strahinja Markovic. Είναι μια εφαρμογή συμβατή με όλα τα λειτουργικά συστήματα. Από την έκδοση 0.9.3 και μετά επιτρέπει την δημιουργία epub3 αρχείων που μπορούν να κάνουν XMLHttpRequest και κατ' επέκταση να επικοινωνήσουν με τον server.

Η χρήση αυτού του εργαλείου προαπαιτεί την γνώση HTML, CSS και Javascript καθώς ο χρήστης θα δημιουργήσει εξ' ολοκλήρου το epub3 αρχείο.

Calibre

Το εργαλείο Calibre είναι και αυτό ένα εργαλείο ανοιχτού κώδικα που χρησιμοποιείται κυρίως για την διαχείριση ηλεκτρονικών βιβλιοθηκών. Μπορεί και να χρησιμοποιηθεί σαν σύστημα ηλεκτρονικής ανάγνωσης και οι τελευταίες εκδόσεις περιέχουν και ενσωματωμένο editor που χρησιμεύει κυρίως στην επισκόπηση του κώδικα του βιβλίου.

Η χρήση αυτού του εργαλείου με στόχο την δημιουργία αρχείων epub3 προαπαιτεί την γνώση HTML, CSS και Javascript.

oXygen

Το εργαλείο oXygen είναι ένας XML editor που υποστηρίζει την επεξεργασία αρχείων epub3. Το oXygen χρησιμοποιείται κυρίως για την επεξεργασία αρχείων epub3 και όχι για την εξ' ολοκλήρου δημιουργία τους. Έχει την δυνατότητα να παράξει ένα κενό αρχείο epub3 και στην συνέχεια ο χρήστης θα πρέπει να συμπεριλάβει τα αρχεία HTML με το βασικό περιεχόμενο μέσα στο κενό αρχείο.

Και αυτό το εργαλείο προαπαιτεί τις γνώσεις HTML, CSS και Javascript και σε αντίθεση με τα προαναφερθέντα εργαλεία δεν είναι δωρεάν.

Τα συστήματα ανάγνωσης (Reading Systems) που έχουν δημιουργηθεί για αναπαραγωγή των αρχείων epub είναι αναρίθμητα. Οι νέες τεχνολογίες που εισήχθησαν στον κόσμο των epub μετά την δημιουργία των epub3 αρχείων οδήγησαν στην αναγκαιότητα του προσδιορισμού των δυντατότήτων των συστημάτων ανάγνωσης εκ νέου.

Ο οργανισμός IDPF σχεδίασε ένα σύστημα αξιολόγησης των συστημάτων ανάγνωσης (epub3 Test Suite) προκειμένου αυτά να κατηγοριοποιηθούν με βάση τα χαρακτηριστικά των αρχείων epub3 που υποστηρίζουν.

Παρατίθεται πίνακας των δέκα καλύτερων συστημάτων ανάγνωσης με γνώμονα τις δυνατότητες τους για αλληλεπίδραση με κώδικα[7]:

NAME	SCRIPT	XHTM	SVG	FONTS	MEDIA	GLS	FIXED LAYOUTS
Bureau van Dijk Reader	93.9%	60.5%	100%	100%	89.5%	97.6%	100%
Readium	89.8%	37.2%	100%	100%	84.2%	100%	85.7%
OverDrive Media Console	8.2%	14%	73.7%	0%	0%	78.6%	7.1%
Gitden Reader	77.6%	37.2%	100%	53.8%	76.3%	97.6%	35.7%
Adobe Digital Editions	77.6%	76.7%	100%	100%	79%	92.9%	92.9%
VitalSource Bookshelf	75.5%	39.5%	94.7%	76.9%	18.4%	23.8%	85.7%
iBooks	61.2%	34.9%	94.7%	100%	21%	90.5%	42.9%
Tolino	59.2%	44.2%	100%	23.1%	23.7%	85.7%	50%
Azardi	55.1%	16.3%	79%	53.8%	0%	11.9%	50%
Kobo	55.1%	58.1%	100%	100%	47.4%	100%	57.1%

3.2 Η γλώσσα Javascript και η βιβλιοθήκη της jQuery

Η JavaScript είναι η γλώσσα προγραμματισμού του διαδικτύου. Η συντριπτική πλειοψηφία των σύγχρονων ιστοσελίδων χρησιμοποιούν JavaScript, και όλα τα σύγχρονα προγράμματα περιήγησης — σε επιτραπέζιους υπολογιστές, κονσόλες παιχνιδιών, tablets, και smartphones — περιλαμβάνουν διερμηνείς της JavaScript, καθιστώντας την, την πλέον διαδεδομένη γλώσσα προγραμματισμού στην ιστορία. Η JavaScript είναι μέρος της Τριάδας των Τεχνολογιών που όλοι οι Web developers πρέπει να μάθουν:η HTML καθορίζει το περιεχόμενο των ιστοσελίδων, η CSS καθορίζει την παρουσίαση των ιστοσελίδων και η JavaScript καθορίζει τη συμπεριφορά των ιστοσελίδων.

Η JavaScript είναι μια υψηλού επιπέδου, δυναμική γλώσσα προγραμματισμού που είναι κατάλληλη για αντικειμενοστραφή (object-oriented) και λειτουργικό (functional) προγραμματισμό[1]. Είναι ελαφριά και πιο συχνά χρησιμοποιείται ως μέρος των ιστο-

σελίδων, των οποίων οι εφαρμογές επιτρέπουν το client-side script να αλληλεπιδρά με το χρήστη κάνοντας τις σελίδες δυναμικές. Είναι μία διερμηνευόμενη γλώσσα προγραμματισμού με αντικειμενοστραφής δυνατότητες.

Η JavaScript δημιουργήθηκε το 1995 από τον Brendan Eich, έναν μηχανικό της Netscape και εκδόθηκε με τον Netscape 2 στις αρχές του 1996. Το όνομα "JavaScript" είναι στην πραγματικότητα κάπως παραπλανητικό. Εκτός από μια επιφανειακή συντακτική ομοιότητα, η JavaScript είναι εντελώς διαφορετική από την Java. Το αρχικό της όνομα ήταν LiveScript αλλά πιθανώς για λόγους μάρκετινγκ, καθώς η εξάπλωση της Java ήταν μεγάλη, μετονομάστηκε σε JavaScript. Είναι μία γλώσσα προγραμματισμού επηρεασμένη από αντικειμενοστραφείς γλώσσες προγραμματισμού όπως η C++ και Java, αλλά η ίδια δεν είναι αντικειμενοστραφής γλώσσα προγραμματισμού.

Ο κώδικας της JavaScript γράφεται σε καθαρό κείμενο (ASCII μορφή) και ενσωματώνεται μέσα στον κώδικα της HTML, μπορεί δε να εκτελεστεί αμέσως ή όταν λαμβάνει χώρα ένα συμβάν (event). Δεν γίνεται μεταγλώττιση (compilation) του κώδικα της JavaScript, αρκεί μόνο ο φυλλομετρητής (browser) να υποστηρίζει την JavaScript.

Η Netscape υπέβαλλε τη γλώσσα για τυποποίηση στον ECMA (Σύλλογο Ευρωπαϊκών κατασκευαστών υπολογιστών) και αυτός όρισε την βασική έκδοση της JavaScript ως μια ελαφριά γλώσσα προγραμματισμού, σχεδιασμένη για τη δημιουργία δικτυακών εφαρμογών, συμπληρώνει και ενσωματώνεται με την Java και την HTML και είναι ανοικτή και συμβατή με όλα τα προγράμματα περιήγησης.

Η client-side χρήση της JavaScript είναι η πιο κοινή χρήση της γλώσσας. Το script θα πρέπει να αναφέρεται ή να περιλαμβάνεται σε ένα έγγραφο HTML προκειμένου ο κώδικας να ερμηνευτεί από το πρόγραμμα περιήγησης.

Αυτό σημαίνει ότι μια ιστοσελίδα δεν χρειάζεται να είναι μια στατική HTML, αλλά μπορεί να περιλαμβάνει προγράμματα που αλληλεπιδρούν με το χρήστη, ελέγχουν το πρόγραμμα περιήγησης, και δημιουργούν ένα δυναμικό περιεχόμενο HTML.

Τα client-side scripts παρέχουν πολλά πλεονεκτήματα σε σύγκριση με τα παραδοσιακά server-side scripts. Για παράδειγμα, ο προγραμματιστής μπορεί να χρησιμοποιήσει JavaScript για να ελέγξει εάν ο χρήστης έχει εισάγει μια έγκυρη διεύθυνση ηλεκτρονικού ταχυδρομείου σε ένα πεδίο φόρμας. Ο κώδικας εκτελείτε όταν ο χρήστης υποβάλλει τη φόρμα, και μόνο εάν όλες οι εγγραφές είναι έγκυρες, τα καταθέτει στο Web server.

Η JavaScript μπορεί να χρησιμοποιηθεί για να εντοπιστούν κάποια συμβάντα που ενεργοποιούνται από το χρήστη, όπως το να κάνει κλικ σε ενα κουμπί, σε κάποιο σύνδεσμο και άλλες ενέργειες που ο χρήστης ενεργοποιεί ρητά ή σιωπηρά.

Μερικά από τα βασικά πλεονεκτήματα που παρουσιάζει η Javascript είναι:

- **Λιγότερη αλληλεπίδραση με τον server**

Ο προγραμματιστής μπορεί να επικυρώσει μια εισαγωγή από τον χρήστη πριν η σελίδα σταλεί στον server. Αυτό εξοικονομεί κυκλοφορία, πράγμα που σημαίνει λιγότερο φορτίο στον server.

- **Άμεση απάντηση στους επισκέπτες**

  Οι επισκέπτες μιας ιστοσελίδας δεν χρειάζεται να περιμένουν μια σελίδα να επαναφορτώσει προκυμμένου να δουν εάν έχουν ξεχάσει να εισάγουν κάτι.

- **Αυξημένη διαδραστικότητα**

  Ο προγραμματιστής μπορεί να δημιουργήσει στοιχεία που αντιδρούν όταν ο χρήστης περνάει από πάνω τους με το ποντίκι ή της ενεργοποιεί μέσω του πληκτρολογίου.

- **Καλύτερες διασυνδέσεις**

  Ο προγραμματιστής μπορεί να χρησιμοποιήσει JavaScript για να περιλάβει στοιχεία όπως drag and drop και sliders για να δημιουργήσει μια πλούσια αλληλεπίδραση των επισκεπτών με τη σελίδα.

Παρά τα σημαντικά της πλεονεκτήματα η Javascript παρουσιάζει και μειονεκτήματα που προκύπτουν από το γεγονός ότι δεν μπορεί να μεταχειριστεί ως μια πλήρως αναπτυγμένη γλώσσα προγραμματισμού. Για παράδειγμα η Javascript JavaScript δεν επιτρέπει την ανάγνωση ή το γράψιμο αρχείων. Αυτό έχει διατηρηθεί για λόγους ασφαλείας.

Η σημαντικότερη βιβλιοθήκη της Javascript είναι η jQuery. Η jQuery είναι μια γρήγορη και περιεκτική βιβλιοθήκη της JavaScript που δημιουργήθηκε από τον John Resig το 2006 με το σύνθημα – «γράψε λιγότερα, κάνε περισσότερα»[2].

Η jQuery απλοποιεί κάποιες λειτουργίες του εγγράφου HTML όπως την διέλευση, τον χειρισμό συμβάντων, το animation και τις αλληλεπιδράσεις Ajax για γρήγορο web development[8].

Η jQuery είναι μια εργαλειοθήκη της JavaScript που σχεδιάστηκε για να απλοποιεί διάφορες εργασίες γράφοντας λιγότερο κώδικα. Παρουσιάζεται ο κατάλογος των πιο σημαντικών διευκολύνσεων που προσφέρονται από τη jQuery:

- **DOM manipulation** -Η jQuery κατέστησε εύκολη την επιλογή στοιχείων DOM, την παρακολούθηση τους και τροποποίηση του περιεχομένου τους, χρησιμοποιώντας μια μηχανή επιλογής ανοικτού κώδικα που ονομάζεται Sizzle.
- **Event handling**-Η jQuery προσφέρει έναν εύκολο τρόπο σύλληψης ενός μεγάλου αριθμού από events, όπως για παράδειγμα όταν ένας χρήστης κάνει κλικ σε μια σύνδεση, χωρίς την ανάγκη να υπερφωτωσει τον HTML κώδικα με event handlers.

- **AJAX Support**– Η jQuery βοηθά πολύ στο να αναπτυχθεί μια πλούσια ιστοσελίδα χρησιμοποιώντας την τεχνολογία AJAX.
- **Animations**– Η jQuery έρχεται με πολλά ενσωματωμένα εφέ κίνησης που μπορεί ο χρήστης να χρησιμοποιήσει στην ιστοσελίδα του.
- **Cross Browser Support**- Η jQuery έχει πολλαπλή υποστήριξη από τους browsers, και λειτουργεί καλά σε IE 6.0+, 2. 0 + FF, Safari 3.0 +, Chrome και Opera 9.0 +.

Η βιβλιοθήκη jQuery μπορεί να συμπεριληφθεί στον HTML κώδικα απευθείας από το CDN (Content Delivery Network). Η Google και η Microsoft παρέχουν αυτή την δυνατότητα στις πιο πρόσφατες εκδόσεις τους.

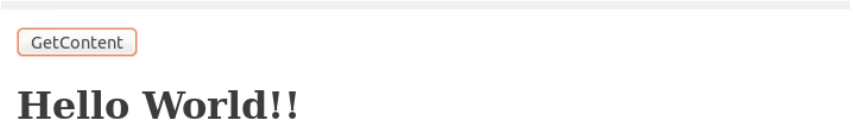
3.3 Διαδραστικότητα αρχείων epub3

Με την χρήση της γλώσσας Javascript και της βιβλιοθήκης της jQuery που αναφέρθηκαν παραπάνω τα αρχεία epub3 εγκαταλείπουν την στατική τους φύση και γίνονται διαδραστικά. Έχουν την δυνατότητα να παράξουν δυναμικά το περιεχόμενο τους και να εκμεταλευτούν τις μεθόδους GET και POST του πρωτοκόλλου HTTP. Παρακάτω αναλύονται διεξοδικά οι δυνατότητες που αναφέρθηκαν και παρατίθενται τμήματα κώδικα ως παραδείγματα.

Δυναμική Παραγωγή Περιεχομένου

Με την χρήση των event της Javascript μπορούμε με το πάτημα ενός κουμπιού να εισάγουμε περιεχόμενο σε ένα κενό αρχείο epub3. Στο παρακάτω παράδειγμα κώδικα που έχουμε εισάγει σε αρχείο epub3 βλέπουμε πως με το πάτημα του κουμπιού Get Content μπορούμε να εμφανίσουμε δυναμικά το μήνυμα ”Hello World!”.

```
1  <html>
2  <head>
3    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery
         .min.js"></script>
4
5
6  </head>
7  <body>
8  <button id="getContent">GetContent</button>
9  <h1></h1>
10 <script type="text/javascript">
11   \$("#getContent").click(function()
12       {
13         \$("h1").text("Hello World!!")
14         })
15      </script>
16 </body>
17 </html>
```

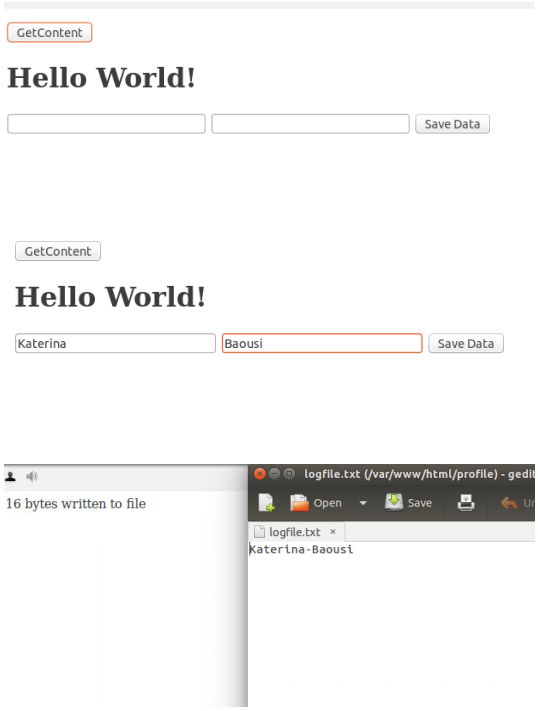


Χρήση μεθόδων GET & POST

Η μέθοδος του της jQuery $.ajax( ) μπορεί να εφαρμοστεί στα αρχεία epub3 και να βοηθήσει στην απόκτηση πληροφοριών από server και την απόκτηση πληροφοριών από αυτούς. Στο επόμενο παράδειγμα κώδικα που έχουμε εισάγει σε αρχείο epub3 βλέπουμε πως μπορούμε να φορτώσουμε το περιεχόμενο του αρχείου και να το εμφανίσουμε μέσα στο epub3 . Επίσης έχουμε την δυνατότητα να συμπληρώσουμε τα πεδία κειμένου των submit forms και να τα στείλουμε πίσω στο server. Στις εικόνες βλέπουμε ότι λαμβάνουμε το απαντητικό μήνυμα που επιβεβαιώνει την σωστή εγγραφή των δεδομένων στο αρχείο logfile.txt και βλέπουμε και το εσωτερικό του αρχείου.

```
1  <html>
2  <head>
3    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery
       .min.js"></script>
4
5
6  </head>
7  <body>
8  <button id="getContent">GetContent</button>
9  <h1></h1>
10
11 <form action="http://localhost/profile/server.php" method="POST">
12     <input name="field1" type="text" />
13     <input name="field2" type="text" />
14     <input type="submit" name="submit" value="Save Data">
15 </form>
16
17 <script type="text/javascript">
18   \$("#getContent").click(function()
19       {   \$.ajax
20       ({
21
22             url: 'http://localhost/getTest.json',
23           type: 'GET',
24           // callback: '?',
25           datatype: 'application/json',
```

```
26              success:function (data){
27
28
29                          answers=data.category[0];
30                      \$("h1").text(answers)
31
32              }
33
34        })
35
36      })
37       </script>
38  </body>
39  </html>
```



3.4 Εφαρμογή σε αρχείο epub3

Στα πλαίσια της εξερεύνησης των δυνατοτήτων των epub3 αρχείων αναπτύχθηκε μια εφαρμογή με στόχο την αξιοποίηση των παναφερθέντων τεχνολογιών. Το αρχείο epub3 που προκύπτει από αυτή την εφαρμογή μπορεί να δημιουργήσει δυναμικά το περιεχόμενο του,να αντλήσει πληροφορίες από ένα server, να αλληλεπιδράσει με τον χρήστη μέσω κουμπιών και πεδίων κειμένου και να στείλει δεδομένα σχετικά με την κατάστασή του πίσω στον server.

Η εφαρμογή αυτή είναι το κλασικό παιχνίδι σταυρόλεξου που δημιουργήθηκε με ειδικές λέξεις για άτομα με μαθησιακές δυσκολίες.Είναι μια εφαρμογή που κάνει χρήση των γλωσσών HTML, Javascript και της βιβλιοθήκης της jQuery και η μορφοποίηση των στοιχείων της έγινε με CSS.Αναπτύχθηκε στον epub3 editor Sigil και οπτικοποιήθηκε στον epub3 viewer Azardi. Αποτελείται από οχτώ αρχεία javascript , ένα αρχείο CSS και ένα αρχείο HTML το οποίο είναι υπεύθυνο για την κλήση και την οπτικοποίηση των προαναφερθέντων αρχείων. Τα αρχεία Javascript καθώς και οι λειτουργίες τους αναφέρονται επιγραμματικά παρακάτω.

initialize.js

Το αρχείο αυτό είναι υπεύθυνο για την δημιουργία της αρχικής εικόνας της εφαρμογής. Δημιουργεί το κουμπί "New" που με το πάτημα του αρχικοποιείται η οθόνη με την κλήση της διαδικασίας clear All( ) που βρίσκεται στο αρχείο clearContent.js.Στην συνέχεια καλείται η διαδικασία get( ) που βρίσκεται στο αρχείο ajaxCalls.js και που είναι υπεύθυνη για την αποστολή του αιτήματος GET στον server και την λήψη των λέξεων του σταυρολέξου ως απάντηση στο αίτημα αυτό.

ajaxCalls.js

Το αρχείο αυτό είναι υπεύθυνο για την επικοινωνία της εφαρμογής με τον server. Εδώ υλοποιείται το GET με την μέθοδο της jQuery $.ajax( ) η οποία λαμβάνει από την αντίστοιχη διεύθυνση τις λέξεις που είναι υπεύθυνες για την δημιουργία του σταυρολέξου.Μόλις ληφθούν και αποθηκευτούν οι λέξεις καλείται την διαδικασία play( ) που λαμβάνει τις λέξεις ως παράμετρο και προχωρά στην κατασκευή του σταυρολέξου. Επίσης σε αυτό το αρχείο βρίσκεται και η διαδικασία που είναι υπεύθυνη για την υλοποίηση της POST μεθόδου με την χρήση των submit form. Τέλος, πρέπει να σημειωθεί ότι το περιεχόμενο των submit form δημιουργείται εδώ εξ'ολοκλήρου με την επέμβαση της Javascript και όχι μέσω του κώδικα της HTML όπως στο παράδειγμα του κεφαλαίου δύο και γι'αυτό το λόγο το περιεχόμενό τους αποστέλεται και πάλι με την μέθοδο $.ajax( ).

crossword.js

Η διαδικασία play( ) που αναφέρεται παραπάνω υλοποιείται εδώ. Η παράμετρος answers που έχει προέλθει από το αίτημα GET χρησιμοποιείται στην δημιουργία του αντικειμένου cw που κατασκευάζεται από την κλάση Crossword του αρχείου crosswordClass.js. Εν συνεχεία καλούνται οι διαδικασίες visualization( ) του αρχείου visualization.js και addFunctionality() του αρχείου buttonCreation.js, που δημιουργούν το οπτικό πλέγμα του σταυρολέξου και δημιουργούν τα διαδραστικά κουμπιά αντίστοιχα.

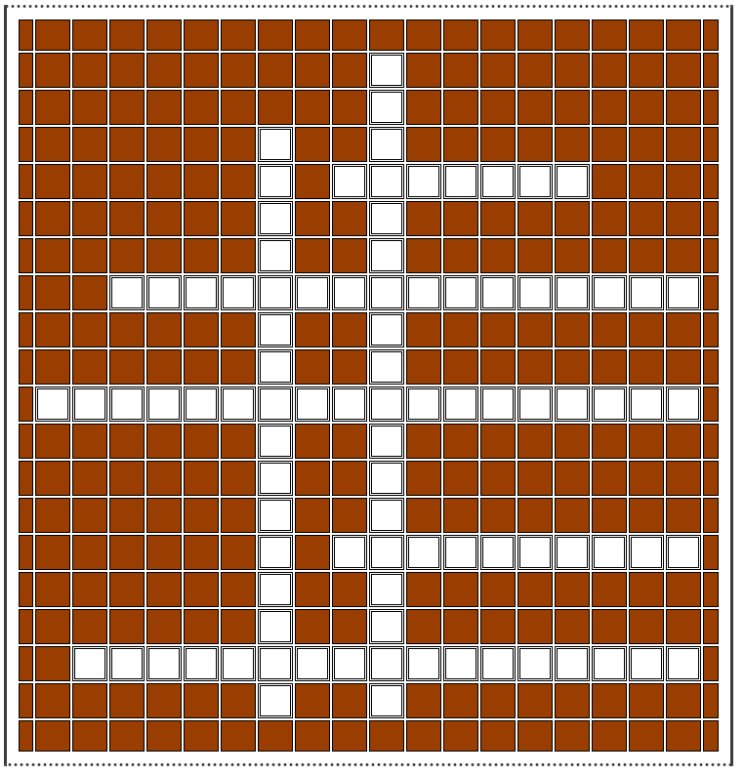
crosswordClass.js

Εδώ δημιουργείται το αντικείμενο Crossword που φέρει τα βασικά πεδία του σταυρολέξου. Οι λέξεις που ήρθαν από το αίτημα στον server συνθέτουν τον πίνακα answerArray ο οποίος αποτελεί τον οδηγό της δημιουργίας του σταυρολέξου. Ο πίνακας αυτός ταξι-

νομείται σύμφωνα με το μήκος των λέξεων με φθίνουσα σειρά και έπειτα εισάγεται η πρώτη και μεγαλύτερη λέξη εντός του σταυρολέξου. Για την τοποθέτηση κάθε επόμενης λέξης ελέγχεται εάν αυτή έχει κοινά γράμματα με το σύνολο των ήδη εισαγμένων λέξεων και στην συνέχεια βρίσκεται η πιθανή κατεύθυνση που μπορεί να έχει μέσα στο πλέγμα (οριζόντια ή κάθετα).Για να χαρακτηριστεί η υποψήφια λέξη για εισαγωγή ως νόμιμη ελέγχεται επιπλέον αν χωράει εντός του οριοθετημένου πλέγματός του σταυρολέξου και αν παραβαίνει τους κανόνες αυτου παιχνιδιού ως προς τον σχεδιασμό του. Τελειώνοντας εδώ περιέχονται όλες οι βασικές μεταβλητές που καλούνται στον υπόλοιπο κώδικα και έχουν ζωτική σημασία στην ορθή λειτουργία του σταυρολέξου.

visualization.js

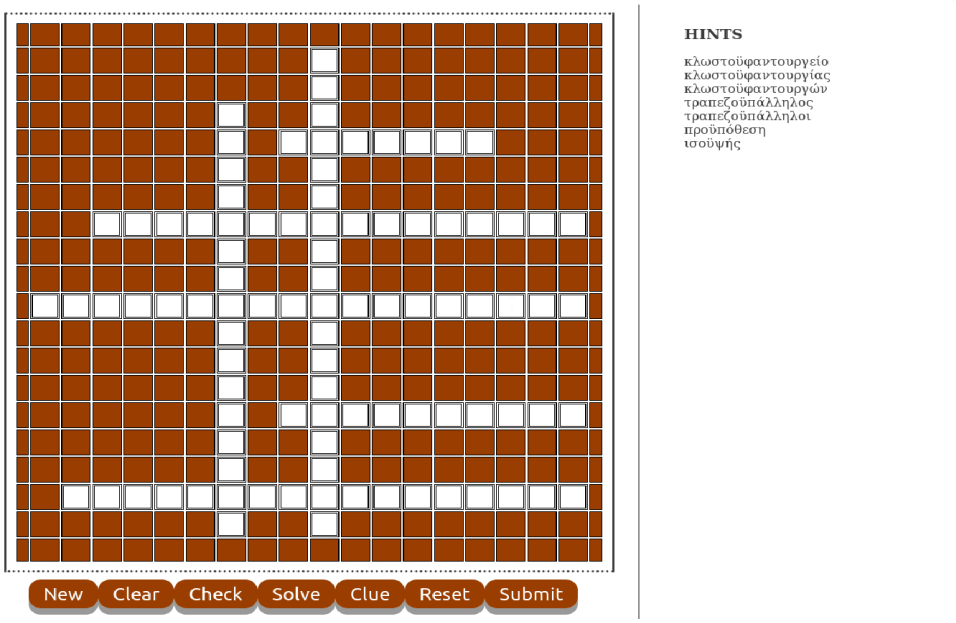
Το αρχείο αυτό είναι υπεύθυνο για την οπτικοποίηση του σταυρόλεξου που έχει δημιουργηθεί στο προηγούμενο βήμα.Δημιουργούνται δυναμικά όλα τα στοιχεία του HTML κώδικα που θα συνθέσουν το σταυρόλεξο που θα γίνει εμφανές στην οθόνη του χρήστη. Στα στοιχεία που δημιουργούνται προστίθεται και οι ειδικά διαμορφωμένες κλάσεις CSS προκειμένου να έχει το σταυρόλεξο τη κατάλληλη εμφάνιση. Αφού τερματίσει την λειτουργία της η διαδικασία visualization( ) που βρίσκεται εντός του συγκεκριμένου αρχείου είναι υπεύθυνη για το ακόλουθο αποτέλεσμα στην οθόνη:



buttonCreation.js

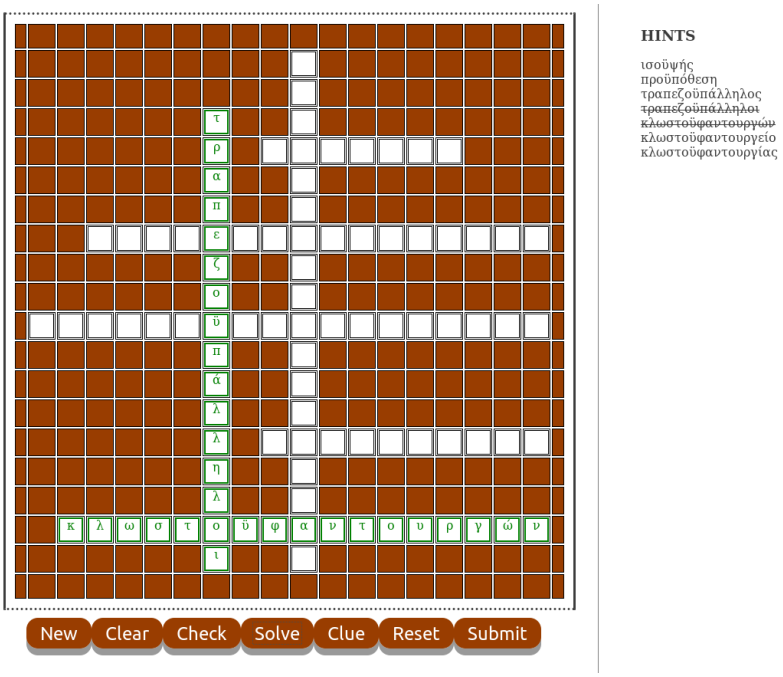
Το αρχείο αυτό περιέχει την διαδικασία addFunctionality( ) που αναφέρθηκε πα-

ραπάνω. Αυτή είναι υπεύθυνη για την δημιουργία των κουμπιών κάτω από το σταυρόλεξο και για την διαχείριση των event του χρήστη. Από το πάτημα κάποιου κουμπιού μέχρι την πληκτρολόγηση λέξεων εντός του πλέγματος του σταυρολέξεων από τον χρήστη , η συνάρτηση αυτή ελέγχει ότι συμβάνει μέσα στην HTML σελίδα από τον χρήστη. Εδώ καλούνται και οι διαδικασίες που θα αναφερθούν παρακάτω. Λόγω της δημιουργίας κουμπιών και της προσθήκης του τμήματος της σελίδας HINTS η εικόνα του σταυρολέξου αλλάζει και γίνεται:

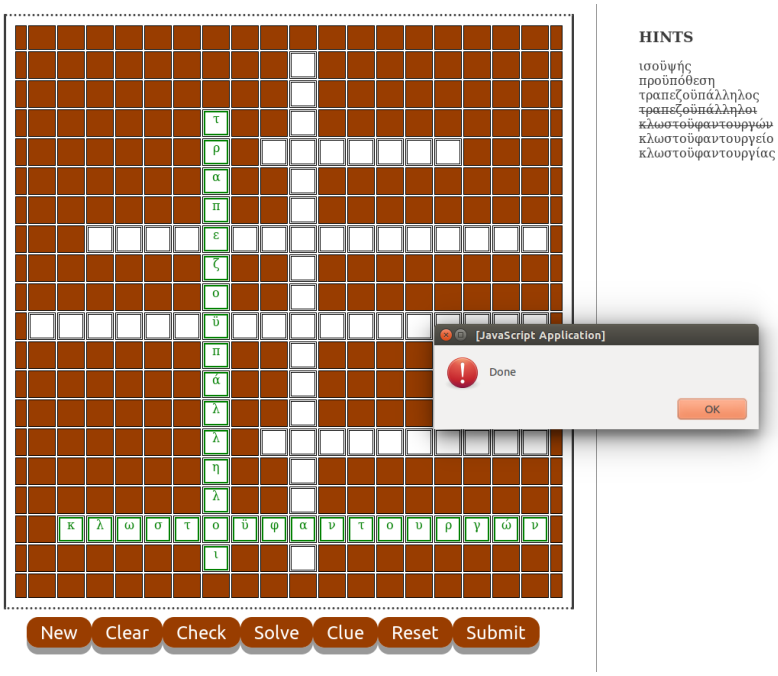


buttonFunctions.js

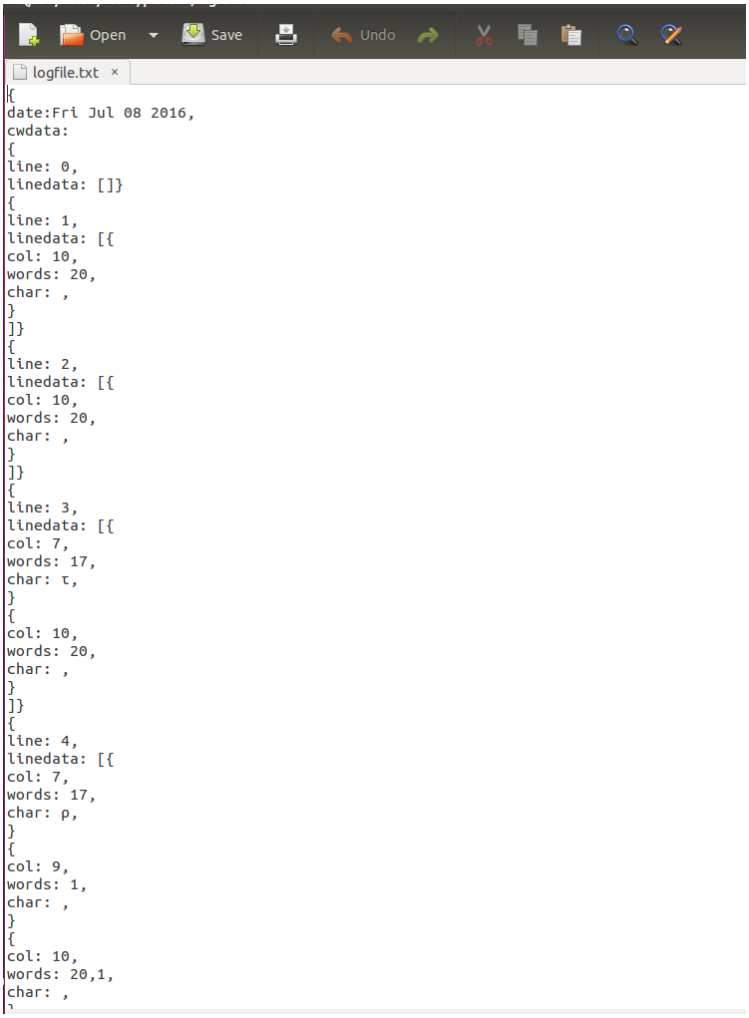
Εδώ είναι οι διαδικασίες που εκτελούνται με το πάτημα των κουμπιών που αναφέρθηκαν παραπάνω. Οι δυνατότητες που παρέχουν αυτά τα κουμπιά στον χρήστη είναι η εκ νέου δημιουργία του σταυρολέξου (”New”), η διαγραφή της λέξης κάθετα ή οριζόντια που έχει εισάγει ανάλογα με το κελί στο οποίο βρίσκεται (”Clear”), ο έλεγχος όλων των στοιχείων που έχει εισάγει ο χρήστης και ο χρωματισμός του ανάλογα με το εάν η εισαγωγή είναι σωστή ή λάθος (”Check”), η επίλυση της λέξης του τρέχοντος κελιού (”Solve”), η εμφάνιση του σωστού γράμματος στο τρέχον κελί για να βοηθηθεί ο χρήστης (”Clue”), η διαγραφή όλων των εισαγμένων πληροφοριών στο πλέγμα (”Reset”) και τέλος η υποβολή των καταγεγραμμένων λέξεων πίσω στον server(”Submit”). Για παράδειγμα αν ο χρήστης πατήσει το κουμπί ”Solve” για την θέση τομής των λέξεων ”κλωστοϋφαντουργών” και ”τραπεζοϋπάλληλοι” το αποτέλεσμα που θα φανερωθεί στην οθόνη είναι το ακόλουθο.



Αν ο χρήστης στην συνέχεια ο χρήστης πατήσει το κουμπί ”Submit” λαμβάνει ενημερωτικό μήνυμα όπως φαίνεται παρακάτω



και τα δεδομένα εισάγονται με μορφοποίηση τύπου JSON αρχείων στο αρχείο logfile.txt του server σύμφωνα με την επόμενη εικόνα.



clearContent.js

Το αρχείο αυτό περιέχει την διαδικασία clearAll( ) η οποία αδείαζει την HTML σελίδα από το περιεχόμενο της.

Σύμφωνα με τα παραπάνω με το πάτημα του κουμπιού ”Submit” όλες οι εισαγωγές του χρήστη αποστέλονται στον server. Όταν οι εισαγμένες πληροφορίες σταλούν στον server επεξεργάζονται από τον παρακάτω κώδικα PHP ο οποίος αναλαμβάνει την αποκωδικοποίηση τους και την εγγραφή τους στο αρχείο logfile.txt που βρίσκεται στην ίδια τοποθεσία. Όπως είναι εμφανές από τον κώδικα η PHP στέλνει ειδοποιητικό μήνυμα σχετικά με την εξέλιξη της εγγραφής των πληροφοριών στο αρχείο.

```
1 <?php
2 if(isset(\$_POST['data'])) {
3     \$data = \$_POST['data'] ;
4     \$ret = file_put_contents('logfile.txt', \$data, FILE_APPEND |
          LOCK_EX);
5     if(\$ret === false) {
```

```
6           die('There was an error while writing this file');
7       }
8       else {
9           echo "\$ret bytes written to file";
10      }
11  }
12  else {
13     die('no post data to process');
14  }
15  ?>
```

4 Αρχεία PDF

4.1 Εισαγωγή στην Acrobat Javascript

Η Acrobat Javascript είναι μια γλώσσα που αναπτύχθηκε βασιζόμενη στον πηγαίο κώδικα της έκδοσης Javascript 1.5 (ISO-16262) γνωστή ως ECMAScript, μια αντικειμενοστρεφής γλώσσα προγραμματισμού αναπτυγμένη από της εταιρεία Netscape. Η δημιουργία της Javascript είχε ως σκοπό την αποσυμφόρηση των διαδικτυακών εφαρμογών που βασίζονταν σε server-side scripts. Η Acrobat Javascript επεκτείνει την χρήση της Javascript δημιουργώντας καινούργια αντικείμενα και μεθόδους και ιδιότητες που τα χαρακτηρίζουν. Τα αντικείμενα που βασίζονται στην δομή του αρχείου Acrobat δίνουν την δυνατότητα στον χρήστη να διαχειριστεί την ασφάλεια του αρχείου, να επικοινωνεί με βάσεις δεδομένων, να χειρίζεται τα επισυνημμένα αρχεία, να κάνει ένα αρχείο PDF διαδραστικό. Δεδομένου ότι τα αντικείμενα της Acrobat Javascript έχουν δημιουργηθεί με βάση την Javascript, είναι δυνατή η χρήση των βασικών κλάσεων της γλώσσας αυτής δηλαδή των κλάσεων Math, String, Date, Array και RegExp.

Τα PDF αρχεία είναι ιδιαίτερα ευέλικτα αφού το περιεχόμενο τους μπορεί να γίνει αντιληπτό τόσο από το λογισμικό της Acrobat όσο και από τους σύγχρονους web browsers. Για αυτό το λόγο είναι πολύ σημαντικό να γίνουν αντιληπτές οι διαφορές μεταξύ της Javascript που χρησιμοποιείται για τους web browsers και για της Acrobat Javascript[9]. Ενδεικτικά αναφέρεται ότι:

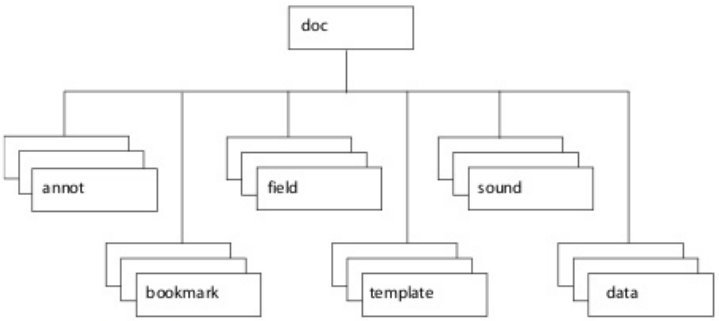
- Η Acrobat Javascript δεν μπορεί να επηρρεάσει τα αντικείμενα μιας ιστοσελίδας HTML και αντίστοιχα η Javascript δεν μπορεί να διαμορφώσει το περιεχόμενου ενός PDF αρχείου.
- Η Javascript διαθέτει το αντικείμενο window που αναφέρεται άμεσα στην εφαρμογή του web browser ενώ η Acrobat Javascript δεν μπορεί. Μπορεί παρόλα αυτά να επηρεάσει αντίστοιχα αντικείμενα εντός της εφαρμογής της Adobe.

Τα αντικείμενα της Acrobat Javascript δίνουν την δυνατότητα στον κώδικα να αλληλεπιδρά με την ίδια την εφαρμογή του Acrobat, το αρχείο PDF ή με τα form fields (πεδία φόρμας στα οποία ο χρήστης μπορεί να εισάγει πληροφορία). Τα πιο σημαντικά αντικείμενα της Acrobat Javscript είναι[10]:

app

Το αντικείμενο app είναι ένα στατικό αντικείμενο που αντιπροσωπεύει την ίδια την εφαρμογή του Acrobat. Προσφέρει έναν αριθμό από διαδικασίες που σχετίζονται με την εφαρμογή του Acrobat. Μέσω της αλληλεπίδρασης με αυτό το αντικείμενο ένας χρήστης μπορεί να ανοίξει ή να δημιουργήσει ένα PDF αρχείο , να μορφοποιήσει την διαπροσωπία της εφαρμογής του Acrobat (πχ να δημιουργήσει παράθυρα διαλόγου, προειδοποιητικά μηνύματα κλπ).

doc Το αντικείμενο doc μπορεί να χρησιμοποιηθεί για την πρόσβαση και την διαχείριση του περιεχομένου του αρχείου PDF. Μέσω του αντικειμένου αυτού είναι εφικτή η πρόσβαση σε γενικές πληροφορίες που αφορούν το αρχείο, η πλοήγηση στο εσωτερικό του, η διαχείριση της δομής του, του περιεχομένου του, η δημιουργία νέου περιεχομένου και η πρόσβαση σε αντικείμενα που ανήκουν στο αρχείο όπως οι σελιδοδείκτες του, τα πεδία φόρμας του, τα σχόλιά του.Τέλος, η πρόσβαση στο αντικείμενο doc μπορεί να γίνει με διάφορους τρόπους μέσω της Acrobat Javascript, ο πιο συνηθισμένος είναι μέσω του αντικειμένου this το οποίο κατά γενικό κανόνα αναφέρεται στον τρέχον αρχείο.



global

Το αντικείμενο global χρησιμοποιείται προκειμένου να αποθηκεύσει δεδομένα που χρήστης επιθυμεί να μπορούν να χρησιμοποιηθούν σε πολλές κλήσεις διαδικασιών Acrobat Javascript ή να μπορούν να χρησιμοποιηθούν από πολλά αρχεία. Η διαμοίραση global δεδομένων μπορεί να επιτευχθεί μέσω ενός μηχανισμού που παρακολουθεί της αλλαγές στα δεδομένα και ενημερώνει τα αρχεία που συνδέονται με αυτά για τυχόν αλλαγές. Επιπλεόν το αντικείμενο global μπορεί να χρησιμποποιηθεί για την αποθήκευσει πληροφοριών που αφορούν μια συγκεκριμένη ομάδα αρχείων. Για παράδειγμα μια ιδιότητα του global αντικειμένου πληροφορεί για το πλήθος των αρχείων που ανήκουν σε μια ομάδα αρχείων.

security

Το αντικείμενο security είναι ένα στατικό αντικείμενο Javascript, διαθέσιμο σε όλες τις εφαρμογές της Acrobat, συμπεριλαμβανομένου και του Acrobat Reader, που υλο-

ποιεί ένα σύστημα ασφάλειας για να διευκολύνει την διαδικασία της δημιουργίας και διαχείρισης μιας ψηφιακής υπογραφής (όνομα χρήστη και κωδικός πρόσβασης). Το αντικείμενο αυτό μπορεί να χρησιμοποιηθεί προκειμένου να προστεθεί ένας νέος κωδικός χρήστη στο αρχείο, να προστεθούν δικαιώματα χρήσης στο αρχείο, να δημιουρηθουν προσαρμοσμένες πολιτικές ασφάλειας, να δημιουργηθεί μια λίστα με τις ταυτότητες των χρηστών και άλλα.

SOAP

Το αντικείμενο SOAP μπορεί να χρησιμοποιηθεί προκειμένου να κάνει κλήσεις σε απομακρυσμένους server και να επικαλεστεί μεθόδους που υποστηρίζονται από το πρωτόκολλο SOAP 1.1 και 1.2. Οι μέθοδοι αυτοί είναι διαθέσιμες για χρήση στις εκδόσεις Acrobat Professional, Acrobat Standard και τους μεταγενέστερους reader του Acrobat Reader 6.0 για αρχεία που έχουν δικαιώματα αποστολής πληροφοριών μέσω πεδίων φόρμας (form fields). Το αντικείμενο SOAP δίνει την δυνατότητα απομακρυσμένης ανταλλαγής σχολίων και την κλήση μεθόδων web service μέσω της πυροδότησης event σε πεδία φόρμας (form field events).

ADBC

Το αντικείμενο ADBC χρησιμοποιείται σε συνεργασία με τα αντικείμενα connection και statement προκειμένου να αλληλεπιδράσει με μια βάση δεδομένων. Αυτά τα αντικείμενα της Acrobat Javascript συνθέτουν τον μηχανισμό Acrobat Database Conectivity (ADBC). Τα SQL queries δηλώνονται μέσα στην μέθοδο execute() του αντικειμένου statement προκειμένου να υπάρχει η δυνατότητα εισαγωγής, ανανέωσης, πρόσβασης και διαγραφής δεδομένων.

event

Όλες οι διαδικασίες της Javascript υλοποιούνται όταν συμβεί ένα συγκεκριμένο γεγονός (event). Για κάθε γεγονός η Acorbat Javascript δημιουργεί ένα αντικείμενο event. Με την πυροδότηση αυτού του γεγονότος το αντικείμενο μπορεί να χρησιμοποιηθεί προκείμενου να εξάγει και να διαχειριστεί οποιαδήποτε πληροφορία σχετίζεται με αυτό το γεγονός. Ένα αντικείμενο event μπορεί να δημιουργηθεί όταν εκκινήσει η εφαρμογή του Acrobat Viewer, ο χρήστης κάνει κλίκ σε κάποιο σελιδοδείκτη ή πεδίο φόρμας, το αρχείο εκτυπώνεται ή αποθηκεύεται και πολλά άλλα.

Η Acrobat Javascript μπορεί να εφαρμοστεί σε επίπεδο φακέλου αρχείων (folder level), σε επίπεδο μεμονωμένου αρχείου (document level), σε επίπεδο μεμονωμένου πεδίου εντός αρχείου (field level) και σε επίπεδο ομάδας αρχείων (batch level). Αυτά τα επίπεδα αναπαριστούν το περιεχόμενο του αρχείου και κατ'επέκταση τον τρόπο με τον οποίο ο κώδικας φορτώνεται στο αρχείο και την προσβασιμότητα του κώδικα εντός και εκτός του αρχείου. Για παράδειγμα η εφαρμογή κώδικα σε επίπεδο φακέλου αρχείων καθιστά τον κώδικα διαθέσιμο σε όλα τα αρχεία, η εφαρμογή κώδικα σε επίπεδο αρχείου καθιστά τον κώδικα διαθέσιμο σε όλα τα πεδία φόρμας εντός του

αρχείου και η εφαρμογή κώδικα σε επίπεδο πεδιού εντός του αρχείου καθιστά τον κώδικα διαθέσιμο για όλα συνδεόμενα με αυτό πεδία.

Folder Level Javascripts

Ο κώδικας Javascript σε επίπεδο Folder Level περιέχει μεταβλητές και διαδικασίες χρήσιμες στην εφαρμογή Acrobat και που είναι ορατά από όλα τα αρχεία. Υπάρχουν δύο ειδών Folder Lever Javascripts αυτά που σχετίζονται με την ίδια την εφαρμογή του Acrobat (App) και αυτά που σχετίζονται με τον χρήστη (User). Για παράδειγμα η δημιουργία εξειδικευμένων μενού που θα είναι προσβάσιμα από όλα τα αρχεία που ανοίγονται μέσω Acrobat επιτυγχάνεται με την δημιουργία Folder Level Javascripts.

Γενικά τα Folder Level Javascripts αποθηκεύονται σε ξεχωριστά αρχεία με κατάληξη ".js" . Τα Folder Level Javascripts τύπου App αποθηκεύονται εντός του φακέλου της εφαρμογής με όνομα "Javascripts" ενώ Folder Level Javascripts τύπου User αποθηκεύονται εντός του φακέλου του χρήστη με όνομα "Javascripts". Όλα αυτά τα αρχεία κώδικα φορτώνονται όταν η εφαρμογή του Acrobat εκκινεί και σχετίζονται με το γεγονός αντικειμένου event με όνομα App/Init.

Μερικά από τα Folder Level Javascripts που παρέχονται απευθείας με την εγκατάσταση της εφαρμογής Acrobat είναι τα Aform.js, Annot.js και ADBC.js(App folder level Javascripts). Επιπλέον ο φάκελος χρήστη περιέχει τα glob.js που σχετίζεται με την χρήση του αντικειμένου global και Config.js που σχετίζεται με την μορφοποίηση της εμφάνισης της εφαρμογής (προσθήκη μενού, διαχείρηση κουμπιών) του Acrobat. Για παράδειγμα η κλήση στην μέθοδο app.addMenuItem( ) βρίσκεται εντός του Config.js.

Document Level Javascripts

Ο κώδικας Javascript σε επίπεδο Document Level περιέχει μεταβλητές και διαδικασίες χρήσιμες σε ένα δεδομένο αρχείο και δεν μπορούν να εφαρμοστούν έξω από αυτό. Ο κώδικας επιπεδου Document Level αποθηκεύεται εντός του αρχείου PDF στο οποίο αναφέρεται και υλοποιείται με το άνοιγμα αυτού του αρχείου.

Field Level Javascripts

Η Acrobat Javascript μπορεί να σχετιστεί με κάθε στοιχείο του αρχείου. Στην περίπτωση δυναμικών πεδίων που κάνουν χρήση XML για αποστολή αιτημάτων στον server, ο κώδικας που σχετίζεται με αυτά τα πεδία είναι εμφανής μόνο στο εσωτερικό του εκάστοτε πεδίου και εκτελείται μόλις συμβεί κάποιο event που σχετίζεται με το συγκεκριμένο πεδίο. Για παράδειγμα είναι δυνατή η εκτέλεση του κώδικα που θα μεταβάλει το περιεχόμενο του πεδίου μόλις ο χρήστης κάνει κλικ στο συγκεκριμένο πεδίο (MouseUp event). Όπως και τα Document Level Javascripts τα Field Level Javascripts αποθηκεύονται εντός του αρχείου PDF στο οποίο ανταφέρονται.

Batch Level Javascripts

Ο κώδικας Javascript σε επίπεδο Batch Level μπορεί να εφαρμοστεί σε μια συλλογή αρχείων και επιδρά στο επίπεδο της εφαρμογής. Για παράδειγμα ο χρήστης μπορεί

να δημιουργήσει ένα αρχείο κώδικα που κατά την εκτέλεση του θα εκτυπώνει την συγκεκριμένη συλλογή αρχείων ή θα εφαρμόζει κάποιους κανόνες ασφάλειας σε αυτά.

4.2 Δυνατότητες δημιουργίας τοπικών αρχείων

4.2.1 Δημιουργία σχολίων μέσα στο αρχείο

Η Acrobat Javascript δίνει την δυνατότητα εισαγωγής και διαχείρισης σχολίων μέσα στο αρχείο με δυναμικό τρόπο. Η δυνατότητα της εισαγωγής σχολίων παρέχεται μέσω του αντικειμένου annot και των τύπων του Text και FreeText. Μπορούμε να εισάγουμε σχόλια σε ένα Text Box με χρήση της Acrobat Javascript μέσω της παρακάτω σύνταξης:

```
this.addAnnot({
   page:0,
   type:"Square",
   rect:[0,0,100,100],
   name:"This is a Test.",
   author:"Katerina",
   content:"Hello World!"
})
```

Στο παραπάνω παράδειγμα η μέθοδος addAnnot( ) λαμβάνει σαν παράμετρο ένα αντικείμενο που χαρακτηρίζει πλήρως το πού θα τοποθετηθεί το σχόλιο, τον τύπο του, το περιεχόμενό του και το όνομά του. Δημιουργεί ένα τετράγωνο πλαίσιο σχολίου στην πρώτη σελίδα του αρχείου. (το αντικείμενο this αναφέρεται στο αντικείμενο doc όπως έχει αναφερθεί νωρίτερα). Το όνομα του αντικειμένου-σχολίου που δημιουργήθηκε δηλώνεται μέσω της παραμέτρου name, ο δημιουργός του μέσω της author και το περιεχόμενό του μεσω της content.

Η Acrobat Javascript δίνει και την δυνατότητα επεξεργασίας των σχολίων που εμπεριέχονται σε ένα αρχείο. Η ταξινόμηση της λίστας των σχολίων επιτυγχάνεται με την κλήση της μεθόδου getAnnots στο αντικείμενο doc. Ένα παράδειγμα της μεθόδου παρατίθεται παρακάτω:

```
var sortedAnnot=this.getAnnots({
                              nPage:2,
                              nSortBy:ANSB_Author,
                              bReverse:true
                                   })
```

Στο παραπάνω παράδειγμα η μέθοδος getAnnots( ) λαμβάνει σαν παράμετρο ένα αντικείμενο που χαρακτηρίζει σε ποιά σελίδα αυτά ανήκουν τα σχόλια, ως προς ποιά μεταβλητή ταξινομούνται αυτά και αν τα σχόλια θέλουμε να ταξινομηθούν σε αύξουσα ή φθίνουσα σειρά. Η παράμετρος nSortBy μπορεί να λάβει τιμές ANSB_None για να μην γίνει ταξινόμηση, ANSB_Page για να ταξινομηθούν τα σχόλια ανά αριθμό σελίδας,

ANSB_Author για να ταξινομηθούν ανά όνομα συγγραφέα και ANSB_ModDate για να ταξινομηθούν ανά ημέρα τροποποίησης.

Τα σχόλια μπορούν να είναι είτε εμφανή μέσα στο αρχείο είτε κρυμμένα αρκεί να δώσουμε την αντίστοιχη τιμή στην ιδιότητα του αντικειμένου annot hidden.

```
1 myAnnot.hidden=true
```

Η πρόσβαση στις πληροφορίες των σχολίων μπορεί να γίνει μέσω της μεθόδου getAnnot( ) αν είναι γνωστό το όνομα του σχολίου ή με την μέθοδο getAnnots( ) που επιστρέφει όλα τα σχόλια του αρχείου και αναζητώντας μέσα στην λίστα των σχολίων να βρούμε το ζητούμενο.

Εάν αναζητούμε για παράδειγμα στην πρώτη σελίδα το σχόλιο με όνομα "This is a Test." αντίστοιχος κώδικας θα είναι ο παρακάτω:

```
1 var content=this.getAnnot(0,"This is a Test.")
```

Η εισαγωγή σχολίων και η επισύναψη αρχείων σε ένα αρχείο PDF είναι δύο διαδικασίες που συμπληρώνουν η μια την άλλη. Τα επισυννημένα αρχεία συνήθως συνοδεύονται από κάποιο σχόλιο που προσδιορίζει το περιεχόμενό τους.

Η επισύναψη αρχείων στην Acrobat Javascript υλοποιείται μέσω της μεθόδου importDataObject( ) του αντικειμένου doc. Η μέθοδος αυτή εισάγει στο αρχείο τα δεδομένα και στην συνέχεια η μέθοδος getDataObject( ) τα επεξεργάζεται δεδομένα του επισυνημμένου αρχείου.

Ο κώδικας που υλοποιεί την διαδικασία που περιγράφηκε παραπάνω και προσθέτει συνοδευτικό σχόλιο στο επισυνημμένο αρχείο είναι ο ακόλουθος:

```
1  //E           XML
2  this.importDataObject(myData,"/folder/myFile.xml")
3  //Δ
4  this.addAnnot({
5                  page:0,
6                 name:"myFileAnnot",
7                 attachIcon:"myData"
8                 });
9  //E
10 var content=this.getDataObject("myData");
11 //
```

4.2.2 Αποθήκευση αρχείου σε άλλο format

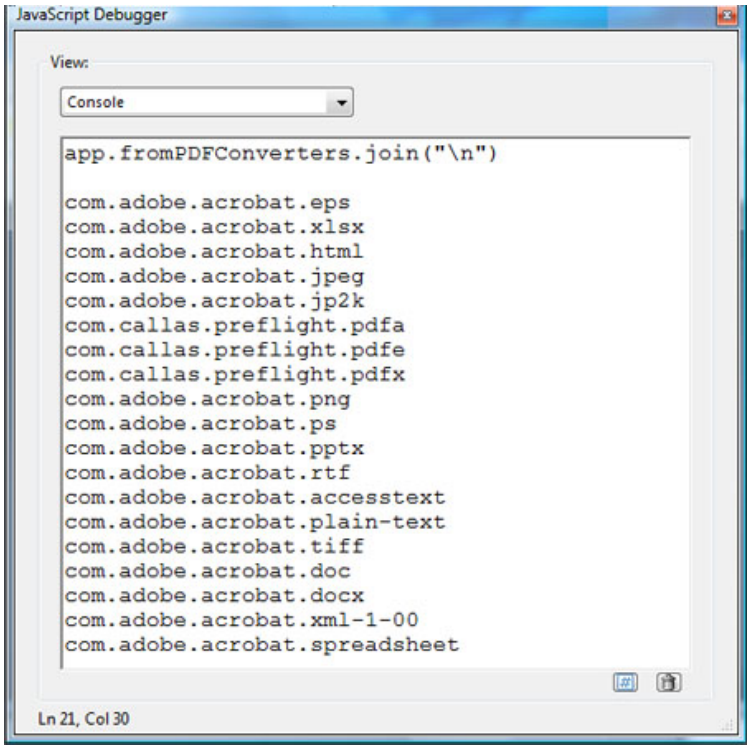
Η αυτοματοποίηση της διαδικασίας αποθήκευσης αρχείων PDF μέσω της Acrobat Javascript είναι ένα πολύ σημαντικό εργαλείο γιατί μπορεί να διευκολύνει την επεξεργασία των αρχείων με βάση τις ανάγκες του κάθε χρήστη. Η διαδικασία doc.saveAs() επιτρέπει την αποθήκευση του αρχείου σε τοποθεσία και με όνομα που υποδεικνύει ο χρήστης μέσω του παρακάτω κώδικα:

```
this.saveAs("/folderName/fileName.pdf");
```

Η παραπάνω διαδικασία μπορεί να δεχθεί και ως παράμετρο ένα αντικείμενο που θα της δίνει την δυνατότητα να ανανεώσει το αρχείο αν αυτό βρίσκεται εκεί που δηλώνει η παράμετρος cPath.

```
this.saveAs({cPath:"/folderName/fileName.pdf", bPromptToOverwrite:true});
```

Η πιο σημαντική δυνατότητα της διαδικασίας doc.saveAs( ) είναι ότι μπορεί να μετατρέψει το αρχείο στο format που επιθυμεί ο χρήστης και στην συνέχεια να το αποθηκεύσει στην τοποθεσία που επιθυμεί. Η μετατροπή επιτυγχάνεται χάρη στην ύπαρξη αντικειμένων app.formPDFConverters που είναι αποθηκευμένα μέσα στην εφαρμογή. Παρατίθεται η λίστα των παραμέτρων που εισάγονται στην διαδικασία doc.saveAs( ) για την μετατροπή των αρχείων στο αντίστοιχων format όπως αυτή εμφανίζεται στην κονσόλα της εφαρμογής Adobe Acrobat Pro:



Μπορεί λοιπόν να γίνει έυκολα αντιληπτό ότι αν θέλουμε να μετατρέψουμε και να σώσουμε το τρέχον αρχείο σε format txt η εντολή του κώδικα Acrobat Javascript είναι:

```
this.saveAs("/folderName/fileName.txt","com.adobe.acrobat.plain-text");
```

4.3 Επικοινωνία PDF αρχείων με βάσεις δεδομένων και server

4.3.1 Το αντικείμενο ADBC

Η Acrobat Javascript προσφέρει ένα αντικείμενο με όνομα ADBC(Acrobat DataBase Conectivity) το οποίο επιτρέπει στον κώδικα επιπέδου document level να συνδεθεί με μια βάση δεδομένων προκειμένου να εισάγει νέα πληροφορία στο αρχείο, να ανανεώσει υπάρχουσες πληροφορίες και να διαγράψει πεδία από την βάση δεδομένων. Απαραίτητη για την λειτουργία του αντικειμένου ADBC είναι η ύπαρξη του λειτουργικού συστήματος Microsoft Windows και λογισμικού που επιτρέπει την αλληλεπίδραση του χρήστη με την βάση από μερίας του χρήστη.

Για να επιτευχθεί μια σύνδεση σε μια βάση δεδομένων μέσω του αντικείμενου ADBC της Acrobat Javascript πρέπει να υλοποιηθούν δύο βασικά βήματα. Αρχικά η κλήση της μεθόδου getDataSourceList που επιστρέφει ένα διάνυσμα αντικειμένων με πληροφορίες για την βάση και στην συνέχεια την κλήση της μεθόδου newConnection που επιτρέπει την σύνδεση στην βάση δεδομένων.

Η πρώτη μέθοδος επιστρέφει ένα διάνυσμα αντικειμένων που έχει τις ιδιότητες name και description. Η ιδιότητα name είναι αυτή που χρειάζεται σαν παράμετρο η μέθοδος connection για να ανοίξει την δίοδο επικοινωνίας του αρχείου με την βάση δεδομένων.

Παρατίθεται σχετικό παράδειγμα κώδικα:

```
var dataBaseList=ADBC.getDataSourceList();

for (var i=0;i<dataBaseList.length;i++)
{
  if(dataBaseList[i].name="Q32016Data")
  {
    myDB=dataBaseList[i].name
    break;
  }
}

if(myDB!="")
{
  var connection=ADBC.newConnection(myDB.name);
}
```

Η μέθοδος newConnection επιστρέφει ένα νέο αντικείμενο που παρέχει τις μεθόδους close( ) για τερρματισμό της σύνδεσης με την βάση, newStatement( ) για την αποστολή SQL queries στην βάση, getTableList( ) για την ανάκτηση δεδομένων από τους πίνακες της βάσης και getColumnList( ) για την ανάκτηση δεδομένων από τις στήλες των πινάκων.

4.3.2 Χρήση των forms για επικοινωνιία με τον server

Η βάση για τα περισσότερα τρέχοντα μοντέλα των διαδικτυακών υπηρεσιών είναι η Extensible Markup Language(XML), που διευκολύνει τη διαλειτουργικότητα, επιτρέποντας την ανταλλαγή πληροφοριών μεταξύ των εφαρμογών και πλατφορμών κατά τρόπο που βασίζεται σε κείμενο.Το πρωτόκολλο SOAP (Simple Object Access Protocol) είναι ένα πρωτόκολλο ανταλλαγής μηνυμάτων που βασίζεται στην XML. Πληροφορίες που αποστέλλονται με το πρότυπο SOAP τοποθετείται σε ένα αρχείο που ονομάζεται ”φάκελος” SOAP (SOAP envelope).

Η Acrobat Javascript παρέχει το αντικείμενο SOAP που υποστηρίζει την επικοινωνία με server. Οι πιο βασικές μέθοδοι του αντικειμένου που υλοποιούν αυτή την επικοινωνιά είναι η μέθοδος request( ) που είναι η βασική μέθοδος για την επίκληση κάποιου web service και η μέθοδος response( ) που χρησιμοποιείται κυριώς σε περιπτώσεις ασύγχρονων κλήσεων στον server.

Η μέθοδος request( ) λαμβάνει ως παράμετρο ένα αντικείμενο προκειμένου να προσδιορίσει πλήρως το αίτημα που θα αποστείλει στον server. Οι μεταβλητές του αντικειμένου περιγράφουν το πού θα αποσταλεί το αίτημα για πληροφορία , τον τύπο της πληροφορίας που ζητάμε, το αν η κλήση είναι ασύχρονη ή όχι, το εάν είναι απαραίτητη η κατάθεση διαπιστευτηρίων για την έναρξη της επικοινωνίας με τον server και άλλα.

Ένα παράδειγμα ασύγχρονης κλήσης στον server είναι το παρακάτω:

```
1
2  // Create the aSync parameter:
3  var mySync = {
4  isDone: false,
5  val: null,
6
7  // Generates the result of the web method:
8  result: function(cMethod)
9  {
10 this.isDone = false;
11 var name = "http://mydomain/methods/:" + cMethod + "Response";
12 if (typeof this.val[name] == "undefined")
13 return null;
14 else
15 return this.val[name]["return"];
16 },
17 // The method called by the web service after completion:
18 response: function(oResult, cURI)
19 {
20 this.val = oResult;
21 this.isDone = true;
22 },
```

```
// While the web service is not done, do something else:
wait: function()
{
while (!this.isDone) doSomethingElse();
}
};

// Invoke the web service:
SOAP.request({
cURL: myURL,
oRequest: echoIntegerRequest,
oAsync: mySync,
cAction: mySOAPAction
});
// The response callback function could contain the following code:
// Handle the asynchronous response:
var result = mySync.result("echoInteger");
// Display the response in the console:
console.println("Result is " + result);
```

Για web services που απαιτούν ταυτοποίηση είναι αναγκαία η χρήση της παραμέτρου oAuthentication της μεθόδου request προκειμένου να χειριστεί το ζήτημα της εξακρίβωσης στοιχείων του HTTP πρωτοκόλλου. Η παράμετρος αυτή έχει τις ιδιότητες Username που είναι μια μεταβλητή τύπου string που καθορίζει το απαιτούμενο όνομα χρήστη και Password που είναι και αυτή μια μεταβλητή τύπου string που περιέχει τον κωδικό πρόσβασης. Ένα παράδειγμα χρήσης της μεθόδου request( ) για την περίπτωση που είναι απαραίτητη η ταυτοποίηση είναι το ακόλουθο[9]:

```
// Create the oAuthenticate object:
var myAuthentication = {
Username: "myUsername",
Password: "myPassword"
};
// Invoke the web service using the username and password:
var response = SOAP.request ({
cURL: myURL,
oRequest: echoStringRequest,
cAction: mySOAPAction
oAuthenticate: myAuthentication
});
```

Τα αντικείμενα της Acrobat Javascript που χρησιμοποιούνται κατά κύριο λόγο για αλληλεπίδραση με τον server είναι τα αντικείμενα form fields.Τα αντικείμενα form fields της Acrobat Javascript μπορούν να παρουσιάσουν τις πληροφορίες στον χρήστη ή να ζητήσουν πληροφορίες από αυτόν χρησιμοποιώντας πεδία φόρμας. Παρέχουν έναν δομημένο τρόπο παρουσίασης των δεδομένων και επιτρέπουν στον χρήστη να

συμπληρώσει προσωπικές πληροφορίες, να επιλέξει από μια λίστα και να υπογράψει ηλεκτρονικά το αρχείο. Μόλις συμπληρωθούν τα πεδία, η πληροφορία που εμπεριέχουν μπορεί να δεχθεί παραπάνω επεξεργασία πριν την τελική της υποβολή ή μπορεί να σταλεί άμεσα σε μια βάση δεδομένων αν έχουμε να κάνουμε με ένα διαδικτυακά συνδεδεμένο πεδίο φόρμας.

Τα πεδία φόρμας που υποστηρίζονται από την Acrobat είναι επτά τύπων. Αυτοί είναι τα κουμπιά, τα τετραγωνίδια ελέγχου, τα στοιχεία ελέγχου σύνθετου πλαισίου, οι λίστες επιλογής, τα κουμπιά πολλαπλής επιλογής, τα πεδία κειμένου και πεδία για τις ηλεκτρονικές υπογραφές. Η δημιουργία τους βασίζεται στην μέθοδο addField( ) του αντικειμένου doc η οποία παίρνει ως παράμετρους το όνομα του πεδίου, τον έναν από τους προαναφερθέντες επτά τύπους που το χαρακτηρίζουν, τον αριθμό της σελίδας που το πεδίο πρέπει να τοποθετηθεί και την ακριβή τοποθεσία στην σελίδα που πρέπει να τοποθετηθεί. Παραδείγματος χάρη για την δημιουργία ενός κουμπιού στην πρώτη σελίδα του αρχείου με τις συντεταγμένες της πάνω αριστερά γωνίας και της κάτω δεξιά γωνίας του να είναι αντίστοιχα (100,472) και (172,400) , ο κώδικας της Acrobat Javascript θα ήταν:

```
1 var name = "myButton";
2 var type = "button";
3 var page = 0;
4 var location = [100, 472, 172, 400];
5 var myField = this.addField(name, type, page, location);
```

Μέσω της Acrobat Javascript μπορούμε να στείλουμε και μαζικά πληροφορίες που υπάρχουν εντός των form fields ενός αρχείου PDF. Αυτό θα μπορούσε να συμβεί με την κατασκευή ενός κουμπιού το οποίο θα επιτρέπει την αποστολή αρχείων στον server όταν γινόταν κλικ σε αυτό. Η κλήση της μεθόδου openDataObject ( ) είναι απαραίτητη για να ανοιχτούν τα πεδία του αρχείου και στη συνέχεια η αποστολή των δεδομένων υλοποιείται με την κλήση της μεθόδου submitForm( ). Παρατίθεται παράδειγμα που υλοποιεί την προναφερθείσα διαδικασία[9]:

```
1  var oParent = event.target;
2
3  // Get the list of attachments:
4
5  var oDataObjects = oParent.dataObjects;
6
7  if (oDataObjects == null)
8  app.alert("This form has no attachments!");
9
10 else {
11
12 // Create the root node for the global submit:
13
```

```
14 var oSubmitData = oParent.xfa.dataSets.createNode(
15 "dataGroup",
16 "globalSubmitRootNode"
17 );
18
19 // Iterate through all the attachments:
20
21 var nChildren = oDataObjects.length;
22 for (var iChild = 0; iChild < nChildren; i++) {
23 // Open the next attachment:
24 var oNextChild = oParent.openDataObject(
25 oDataObjects[iChild].name
26 );
27
28 // Transfer its data to the XML collection:
29
30 oSubmitData.nodes.append(
31 oNextChild.xfa.data.nodes.item(0)
32 );
33 close the attachment//
34 oNextChild.closeDoc();
35 }
36
37 // Submit the XML data collection to the server
38
39 oParent.submitForm({
40 cURL: "http://theserver.com/cgi-bin/thescript.cgi",
41 cSubmitAs: "XML",
42 oXML: oSubmitData
43 });
44 }
```

5 Συμπεράσματα και μελλοντικές επεκτάσεις

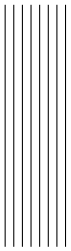
Η συμβολή του Διαδικτύου,και των πρωτοκόλλων που το διέπουν, στην επικοινωνία και στον τρόπο μετάδοσης της πληροφορίας είναι αδιαμφισβήτητα πολύ σημαντική.Οι πληροφορίες μπορούν να μεταδίδονται με ασφάλεια από και προς τον χρήστη και το περιεχόμενο τους να παραμένει έγκυρο στην πάροδο του χρόνου.Επιπλέον, οι εφαρμογές του Διαδικτύου έχουν βελτιώσει την ποιότητα της ζωής μας σε πολύ μεγάλο βαθμό και μπορούν πλέον να βοηθήσουν και στην αντιμετώπιση προβλημάτων όπως αυτό των μαθησιακών δυσκολιών.

Στα πλαίσια αυτής της πτυχιακής εργασίας εξερευνήθηκε η δυνατότητα της χρήσης των τεχνολογιών των ηλεκτρονικών αρχείων σε format epub3 και PDF για την δημιουργία διαδραστικής εφαρμογής. Η εφαρμογή που κατασκευάστηκε δημιουργεί αρχεία που δέχονται πληροφορίες από έναν απομακρυσμένο server, επικοινωνεί διαδραστικά με τον χρήστη, δέχεται επεξεργασία από αυτόν και τέλος στέλνει πληροφορίες πίσω στον server σχετικές με την πρόοδο του.

Τα αρχεία epub3 αποδείχθηκαν κατάλληλα για την ανάπτυξη της εφαρμογής αυτής. Η δυνατότητα που παρέχουν για απόστολή και λήψη μηνυμάτων από και προς τον server καθώς και για δημιουργία δυναμικού περιεχομένου αποτελεί σημαντικό πλεονέκτημα τόσο για τον μαθητή-χρήστη που αντιμετωπίζει μαθησιακές δυσκολίες όσο και για τον παιδαγωγό - λογοθεραπευτή που καλείται να του παρέχει θεραπεία. Ο πρώτος μπορεί έτσι να διασκεδάσει λύνοντας την εξατομικευμένη άσκηση που του έχει ανατεθεί ενώ ο δεύτερος μπορεί να αξιολογήσει βάσει των αποτελεσμάτων την εξέλιξη του χρήστη.Αξίζει να σημειωθεί ότι τα αρχεία PDF αναδείχθηκαν ικανά να υποστηρίξουν την ίδια εφαρμογή αφού παρέχουν και αυτά παρόμοιες δυνατότητες με τα αρχεία epub3. Τέλος διαπιστώθηκε πως η χρήση των τεχνολογιών που σχετίζονται με την αποθήκευση σε τοπικά αρχεία (cookies, localStorage, activeXobject) μπορεί να βελτιώσει την απόδοση της εφαρμογής γιατί με αυτό τον τρόπο η πρόοδος του χρήστη θα είναι καταγεγραμμένη και άμεσα προσβάσιμη.

Η εφαρμογή αυτή μπορεί να αποτελέσει την βάση για περεταίρω εξέλιξη. Αρχικά η δημιουργία μια βάσης που θα περιέχει παιχνίδια διαφόρων ειδών (κρεμάλα, κρυπτόλεξο, σταυρόλεξο κ.α) θα μπορούσε να δώσει στον χρήστη την δυνατότητα επιπλέον επιλογών που αφορούν τις προτιμήσεις αλλά και τις ανάγκες του. Η επέκταση του

συνόλου των λέξεων και σε άλλες γλώσσες θα μπορούσε να αποδώσει παγκόσμιο χαρακτήρα στην εφαρμογή.Εν κατακλείδι για την βελτίωση της απόδοσης της εφαρμογής είναι αναγκάια η διαμόρφωση ειδικού viewer ηλεκτρονικών αρχείων ο οποίος θα υποστηρίζει την εγγραφή σχετικών με την πρόοδο του χρήστη δεδομένων σε τοπικά αρχεία.



Βιβλιογραφία

[1] David Flanagan. *Javascript: The Definitive Guide,Sixth Edition*. O'Reilly, 2011.

[2] The jQuery Foundation. jquery api. https://api.jquery.com.

[3] Microsoft Developer Network. Activexobject. https://msdn.microsoft.com.

[4] Matt Garrish. *What is EPUB3?* O'Reilly, 2011.

[5] Matt Garrish. *Accessible EPUB3*. O'Reilly, 2012.

[6] Matt Garrish & Markus Gulling. *EPUB 3 Best Practices*. O'Reilly, 2013.

[7] IDPF. Epub3 support grid. http://www.epubtest.org.

[8] jQuery Community Experts. *jQuery Cookbook*. O'Reilly, 2010.

[9] Adobe Solutions Network. *Acrobat Javascript Scripting Guide*. Adobe, 2005.

[10] Adobe Systems Incorporated. *Javascript for Acrobat API Reference*. Adobe, 2007.