ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΕΞΑΣΚΗΣΗ ΜΝΗΜΗΣ ΓΙΑ ΤΗ ΝΟΣΟ ΑΛΤΣΧΑΙΜΕΡ ΜΕ ΕΦΑΡΜΟΓΗ ANDROID

ΠΑΠΑΔΟΠΟΥΛΟΥ ΠΑΝΑΓΙΩΤΑ ΑΜ: 15242

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΓΕΩΡΓΙΟΣ ΑΣΗΜΑΚΟΠΟΥΛΟΣ

ΠΑΤΡΑ

ΟΚΤΩΒΡΙΟΣ 2020

ΠΡΟΛΟΓΟΣ

Η παρούσα πτυχιακή εργασία εκπονήθηκε στο πλαίσιο της ολοκλήρωσης των σπουδών στο Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών, πρώην Τεχνολογικό Εκπαιδευτικό Ίδρυμα Δυτικής Ελλάδας Μηχανικών Πληροφορικής. Πρόκειται για μια εργασία με ανάπτυξη εφαρμογής πάνω στο λειτουργικό σύστημα Android. Η υλοποίηση της εφαρμογής συμπληρώνεται με ένα βασικό θεωρητικό μέρος γύρω από το συγκεκριμένο λειτουργικό σύστημα και γενικότερα τις εφαρμογές του καθώς επίσης και μια παρουσίαση της νόσου Αλτσχάιμερ.

ΠΕΡΙΛΗΨΗ

Στην παρούσα πτυχιακή εργασία αναλύονται και περιγράφονται τεχνικές ανάπτυξης εφαρμογών για έξυπνες κινητές συσκευές και πραγματοποιείται ανάπτυξη μιας εφαρμογής σε λειτουργικό Android.

Στην εισαγωγή γίνεται αναφορά στην συνεχή αύξηση του πλήθους των νέων εφαρμογών που υλοποιούνται και αφορούν τις έξυπνες κινητές συσκευές πάνω σε διάφορα λειτουργικά συστήματα καθώς επίσης και τον λόγο που επιλέχθηκε σαν θέμα της εργασίας η ανάπτυξης μιας εφαρμογής σε λειτουργικό Android.

Στο δεύτερο κεφάλαιο γίνεται μια περιγραφή των κινητών εφαρμογών (mobile applications) και αμέσως μετά γίνεται μια παρουσίαση του λειτουργικού συστήματος Android στην οποία περιέχονται πληροφορίες σχετικά με τη φύση του και την εξέλιξη του μέσα στον χρόνο από τότε που έκανε τα πρώτα του βήματα στο τεχνολογικό χώρο. Επιπροσθέτως στο ίδιο κεφάλαιο πραγματοποιείται μια σύγκριση του Android με άλλα λειτουργικά συστήματα της αγοράς.

Στο επόμενο κεφάλαιο περιλαμβάνεται μια περιγραφή για το πώς αναπτύσσεται μια εφαρμογή σε Android , ενώ στη συνέχεια και συγκεκριμένα στο κεφάλαιο πέντε παρουσιάζονται οι πιο σημαντικές πληροφορίες για την νόσο Αλτσχάιμερ, Οι πληροφορίες αυτές σχετίζονται με τα στάδια της νόσου καθώς επίσης και με τους τρόπου αντιμετώπισής της.

Στην συνέχεια, στο κεφάλαιο έξι παρουσιάζονται τα βασικά εργαλεία ανάπτυξης που χρειάζεται κάποιος για να δημιουργήσει μια εφαρμογή Android, ενώ αμέσως μετά τα βασικά δομικά μέρη μιας τέτοιας εφαρμογής.

Στο κεφάλαιο οκτώ περιγράφεται η υλοποιήσιμη εφαρμογή της εργασίας μαζί με τις βασικές τεχνικές που αποτέλεσαν τη βάση της ανάπτυξης της. Η κεντρική ιδέα της εφαρμογής είναι ένα διαδραστικό παιχνίδι με το οποίο ο χρήστης έχει την ευκαιρία να εξασκεί τη μνήμη του, κάτι που αποτελεί καταλυτικό παράγοντα στην αντιμετώπιση της νόσου Αλτσχάιμερ.

ABSTRACT

This diploma thesis analyzes and describes application development techniques for smart mobile devices and as well as an application on Android operating system.

The introduction refer to the continuous increase of the number of new applications that are implemented for smart mobile devices on various operating systems as well as the reason that the development of an Android application was chosen as subject for the current diploma thesis.

The second chapter describes mobile applications and after that there is a presentation of the Android operating system that includes information about its nature and its evolution from the moment it appeared in the field of technology. In addition, in the same chapter there is a comparison between Android and other operating systems.

Next chapter includes a description of how an Android application can be developed and then in chapter five includes the most important information about Alzheimer's disease. This information is related to the stages of the disease and its treatment.

Chapter six presents the basic development tool that someone needs to develop an Android application and after that there are the basic structural parts of such an application.

In the last chapter there is a description for the application of diploma thesis along with the basic techniques that the application based on. The main idea of the application is an interactive game with which the user has the opportunity to practice his/her memory, which is catalyst in the treatment of Alzheimer's disease.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον κύριο Γεώργιο Ασημακόπουλο που μου εμπιστεύτηκε το θέμα της πτυχιακής μου εργασίας δίνοντας μου την ευκαιρία να καταπιαστώ με ένα τόσο ενδιαφέρον και σύγχρονο θέμα, η τριβή με το οποίο μου παρείχε ένα ακόμη εφόδιο καθώς η ολοκλήρωση της φοιτητικής μου σταδιοδρομίας φθάνει στο τέλος της.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

1. ΕΙΣΑΓΩΓΗ

Στην εποχή που διανύουμε οι κινητές προσωπικές συσκευές αποτελούν αναπόσπαστο κομμάτι της καθημερινότητας μας και γίνονται όλο ένα εξυπνότερες. Όσο τα χαρακτηρίστηκα τους εμπλουτίζονται και η δυναμική τους αυξάνεται το ίδιο συμβαίνει και με τις παροχές που μας δίνουν. Αυτές οι παροχές αντικατοπτρίζονται μέσω των εκατομμύρια εφαρμογών που αναπτύσσονται για αυτές τις συσκευές. Δεν είναι λίγες οι εφαρμογές αυτές οι οποίες συσχετίζονται με το πεδίο της υγείας παρέχοντας άμεση ή έμμεση βοήθεια στο καταναλωτικό κοινό. Μια από τις πιο γνωστές νόσους των ημερών μας είναι το Αλτσχάιμερ. Θέλοντας να εκμεταλλευτούμε την ικανότητα της πληροφορικής και της ανάπτυξης εφαρμογών με σκοπό την αντιμετώπιση μιας τόσο διαδεδομένης νόσου σκεφτήκαμε να δημιουργήσουμε ένα εργαλείο που δίνει την δυνατότητα στους ασθενείς της νόσου αυτής να εξασκήσουν την μνήμη τους. Ουσιαστικά πρόκειται για μια έμμεση αντιμετώπιση της νόσου αφού η εξάσκηση της μνήμης αποτελεί την πιο απλή και προπάντων την πιο βασική μη φαρμακευτική μέθοδο αντιμετώπισης της νόσου. Επιλέγοντας το λογισμικό του Android , όντας το πιο διαδεδομένο μεταξύ αυτών που υπάρχουν, αναπτύχθηκε μια εφαρμογή παιχνιδιού καρτών με σκοπό την όξυνση και την βελτίωση της μνήμης του χρήστη μέσω ενός διαδραστικού και ταυτόχρονα διασκεδαστικού τρόπου.

2. MOBILE APPLICATIONS

Όταν μιλάμε για μια κινητή εφαρμογή απευθυνόμαστε σε εφαρμογή λογισμικού που αναπτύχθηκε ειδικά για χρήση σε μικρές ασύρματες υπολογιστικές συσκευές όπως έξυπνα τηλέφωνα (smartphones) και ταμπλέτες (tablets) αντί για επιτραπέζιους ή φορητούς υπολογιστές. Οι εφαρμογές αυτές σχεδιάζονται παίρνοντας ως παράγοντα τις απαιτήσεις καθώς και τους περιορισμούς των εκάστοτε συσκευών. Πέραν αυτού οι εφαρμογές επιδιώκουν να επωφελούνται από τις δυνατότητες που παρέχουν οι κινητές συσκευές. Για παράδειγμα ένα παιχνίδι που περιλαμβάνει χάρτες ή τοποθεσίες θα μπορούσε να χρησιμοποιήσει την τεχνολογία GPS που περιλαμβάνεται στις συσκευές.

Οι προγραμματιστές κινητών εφαρμογών επιδιώκουν να προσεγγίζουν την ευρύτερη ποικιλία συσκευών που απαιτείτε να υποστηρίζουν τις εφαρμογές τους. Παρακάτω ακολουθούν μερικές από τις πιο κοινές κινητές πλατφόρμες:

Έξυπνα τηλέφωνα (Smartphones)

Συσκευές κινητού διαδικτύου(Mobile Internet Devices - MID)

Φορητά μέσα αναπαραγωγής πολυμέσων(Portable media players - PMPs)

Φορητοί Υπολογιστές (Laptops)

Netbooks

Υπερκινητοί προσωπικοί υπολογιστές (Ultra mobile personal computers – UMPC)

Υπολογιστές Tablet

Από άποψη λειτουργικότητας, η ανάπτυξη κινητών εφαρμογών για κινητές πλατφόρμες συνηθίζεται να είναι αρκετά δύσκολη λόγω της ποικιλίας αυτών των συσκευών. Μια εφαρμογή θα πρέπει να εμφανίζεται σωστά σε συσκευές που κυμαίνονται σε διαφορετικό εύρος μεγέθους οθόνες (από έξυπνα τηλέφωνα με οθόνη τριών ιντσών έως φορητούς υπολογιστές με οθόνες 17

ιντσών). Το ίδιο συμβαίνει και για ότι αφορά τις απαιτήσεις ασφαλείας για αυτές τις συσκευές και τις πλατφόρμες τους, οι οποίες θα διαφέρουν ανάλογα με τις προδιαγραφές για κάθε υποστηριζόμενη κινητή συσκευή.

Οι εφαρμογές πολύ συχνά κατηγοριοποιούνται σύμφωνα με το αν είναι βασισμένες στο διαδίκτυο (web-based apps) ή είναι εγγενείς εφαρμογές (native apps) που σημαίνει ότι έχουν δημιουργηθεί ειδικά για την συγκεκριμένη πλατφόρμα. Εκτός αυτόν το δύο κατηγοριών συναντάμε και τις εφαρμογές που συνδυάζουν στοιχεία και των δύο αυτών δημιουργώντας μια νέα η οποία τις χαρακτηρίζει ως υβριδικές εφαρμογές (hybrid apps). Γενικότερα επικρατεί η άποψη ότι με την πάροδο του χρόνου και όσο οι τεχνολογίες γύρω από αυτές αναπτύσσονται θα επικεντρώνονται στη δημιουργία εφαρμογών που στοχεύουν σε συγκεκριμένα προγράμματα περιήγησης (browser-based apps) και εφαρμογές οι οποίες δεν θα χρειάζεται να γνωρίζουν λεπτομέρειες της συσκευής στην οποία εκτελούνται (device-agnostic apps).

- Native applications

Πρόκειται για τις εφαρμογές οι οποίες έχουν υλοποιηθεί προκειμένου να χρησιμοποιηθούν σε μια συγκεκριμένη πλατφόρμα ή λειτουργικό σύστημα. Συνήθως γράφονται σε γλώσσες που δέχεται η πλατφόρμα καθώς επίσης και στο αντίστοιχο περιβάλλον ανάπτυξης που υποστηρίζει αυτή. Για παράδειγμα στα windows οι εφαρμογές γράφονται σε γλώσσα C# ή Visual Basic .Net, στο Android σε Java ή Kotlin και στο iOS σε Swift ή Objective C.

Πλεονεκτήματα:

Είναι πολύ γρήγορες και ανταποκρίνονται αρτιότερα

Έχουν την καλύτερη απόδοση

Είναι πιο διαδραστικά και τρέχουν πιο ομαλά σε ότι σχετίζεται με την διαχείριση από τον χρήστη

Μειονεκτήματα:

Είναι πιο ακριβές στο να αναπτυχθούν σε σχέση με τις εφαρμογές πολλαπλών πλατφορμών (cross platform applications) και των web εφαρμογών

Χρειάζονται περισσότερο χρόνο για να υλοποιηθούν μιας και για να τρέξουν σε άλλη πλατφόρμα πρέπει να γραφτούν εκ νέου σε άλλη γλώσσα

Έχουν υψηλότερο κόστος συντήρησης και ενημέρωσης

- Hybrid Applications

Οι εφαρμογές οι οποίες αναπτύσσονται με σκοπό να εκτελεστούν σε παραπάνω από μια πλατφόρμα ή λογισμικό. Κατασκευάζονται με τρόπο παρόμοιο με εκείνο των ιστότοπων (websites). Χρησιμοποιούν ένα συνδυασμό τεχνολογιών όπως HTML, CSS και JavaScript. Ουσιαστικά στοχεύουν σε ένα WebView που φιλοξενείται μέσα σε ένα εγγενές κοντέϊνερ αντί να στοχεύουν σε ένα πρόγραμμα περιήγησης (mobile browser). Με αυτόν τον τρόπο τους επιτρέπεται να κάνουν πράγματα όπως η δυνατότητα πρόσβασης στο hardware της συσκευής.

Πλεονεκτήματα:

Ο ίδιος κώδικας μπορεί να χρησιμοποιηθεί ξανά για διάφορες πλατφόρμες και λογισμικά δίνοντας τους το χαρακτηριστικό της προσαρμοστικότητας

Ενιαία και λιγότερο δαπανηρή ανάπτυξη

Γρηγορότερος χρόνος ανάπτυξης σε σχέση με τις εγγενείς εφαρμογές

Μειονεκτήματα:

Λιγότερο αποδοτικές σε σχέση με τις εγγενείς εφαρμογές, επειδή πρέπει να υπάρξει μια γέφυρα επικοινωνίας μεταξύ του κώδικα του υβριδικού πλαισίου με τα εγγενή χαρακτηριστικά του τηλεφώνου.

Οι εφαρμογές με βαριά κινούμενα σχέδια και ηχητικά εφέ δεν είναι τόσο απρόσκοπτα όσο στις εγγενείς εκδόσεις τους.

- Progressive Web Applications

Είναι οι εφαρμογές ιστού που χρησιμοποιούν τις σύγχρονες δυνατότητες του ιστού προκειμένου να παρέχουν στον χρήστη μια εμπειρία απλής εφαρμογής

(που δεν μοιάζει δηλαδή με εφαρμογή ιστού) χωρίς να απαιτείται από αυτόν να εγκαταστήσει μια εφαρμογή από το AppStore / PlayStore. Είναι συνήθως προσβάσιμες από μια διεύθυνση URL(Uniform Resource Locator) που μπορεί να αποθηκεύεται στην αρχική οθόνη του τηλεφώνου του. Οι εφαρμογές αυτές κατασκευάζονται τις περισσότερες φορές χρησιμοποιώντας HTML, CSS και JavaScript.

Πλεονεκτήματα:

Εύκολη κατασκευή

Εύκολο στη συντήρηση

Λειτουργεί εκτός σύνδεσης και λειτουργεί καλά σε δίκτυα χαμηλής ποιότητας

Δημιουργείτε μια εφαρμογή για όλες τις πλατφόρμες (αρκεί να μπορεί να εκτελέσει ένα πρόγραμμα περιήγησης)

Μειονεκτήματα:

Απαιτεί ένα πρόγραμμα περιήγησης για εκτελεστεί (οι χρήστες πρέπει να κάνουν ένα ακόμα βήμα για να πληκτρολογήσουν τη διεύθυνση URL της εφαρμογής, κάτι που δυσχεραίνει την εμπειρία του χρήστη)

Λιγότερο διαδραστικές και διαισθητικές από τις εγγενείς εφαρμογές.

3. ΤΟ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ ANDROID

3.1 Τι είναι το Android

Πρόκειται για ένα λειτουργικό σύστημα που απευθύνεται σε κινητές συσκευές και το οποίο εκτελεί τον πυρήνα του λειτουργικού Linux. Η πρωταρχική ανάπτυξη έγινε από την Google και στην συνέχεια την ανέλαβε η Open Handset Alliance. Μέσω των βιβλιοθηκών που παρέχει η Google , οι προγραμματιστές χρησιμοποιώντας την γλώσσα προγραμματισμού Java έχουν την δυνατότητα να αναπτύσσουν τα δικά τους προγράμματα και εφαρμογές καθώς και επίσης να ελέγχου την συσκευή. Τα τελευταία χρόνια το λογισμικό έχει χρησιμοποιηθεί σε διάφορα περιβάλλοντα όπως τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear), κάνοντας ευρύτερο το αρχικό πλάνο το οποίο αφορούσε κατά βάση σε συσκευές με οθόνη αφής όπως τα έξυπνα τηλέφωνα (smart phones) και τα τάμπλετ. Εκτός από τα παραπάνω , μέχρι σήμερα έχει χρησιμοποιηθεί ακόμη σε κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, συνηθισμένους Η/Υ (π.χ. το HP Slate 21) και σε άλλες ηλεκτρονικές συσκευές. Στις μέρες μας αποτελεί το πιο γνωστό και διαδεδομένο λογισμικό στον κόσμο. Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007 μαζί με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια Ιρίνα Μπλόκ. Η τελευταία έκδοση καλείται Android 10 Queen Cake η οποία κυκλοφόρησε επίσημα στις 3 Σεπτέμβρη 2019 (ήδη είχαν κυκλοφορήσει δοκιμαστικές εκδόσεις αυτής από τις 13 Μαρτίου του ίδιου έτους) και φέρνει σημαντικές αλλαγές. Στις 19 Φεβρουαρίου ανακοινώθηκε η νέα έκδοση όπου η πρώτη δοκιμαστική έκδοση θα κυκλοφορήσει στις 3 Ιουνίου του 2020.

3.2 Εξέλιξη του Android

Το λειτουργικό σύστημα Android ιδρύθηκε στο Palo Alto της Καλιφόρνια τον Οκτώβριο του 2003 από τους Andy Rubin, Rich Miner, Nick Sears και Chris White. Οι πρώτες προθέσεις της εταιρείας ήταν να αναπτύξει ένα προηγμένο λειτουργικό σύστημα για ψηφιακές φωτογραφικές μηχανές και αυτή ήταν η βάση του πεδίου για τους επενδυτές τον Απρίλιο του 2004. Η εταιρεία αποφάσισε τότε ότι η αγορά των φωτογραφικών μηχανών δεν ήταν αρκετά μεγάλη για τους στόχους της, πέντε μήνες αργότερα είχε εκτρέψει τις προσπάθειές της και ανέβαζε το Android ως λειτουργικό σύστημα χειρός που θα ανταγωνιζόταν τα Symbian και τα Microsoft Windows Mobile. Ο Ρούμπιν είχε δυσκολία να προσελκύσει επενδυτές από νωρίς και το Android αντιμετώπιζε έξωση από το χώρο γραφείων του. Τον Ιούλιο του 2005, η Google απέκτησε το Android Inc. για τουλάχιστον 50 εκατομμύρια δολάρια. Οι βασικοί υπάλληλοί της, όπως οι Rubin, Miner και White, εντάχθηκαν στο Google ως μέρος της εξαγοράς. Δεν γνώριζαν πολλά για το μυστικό Android την εποχή εκείνη, καθώς η εταιρεία παρέσχε λίγες λεπτομέρειες εκτός από το ότι έκανε λογισμικό για κινητά τηλέφωνα. Στη Google, η ομάδα με επικεφαλής τον Rubin ανέπτυξε μια πλατφόρμα κινητών συσκευών που τροφοδοτείται από τον πυρήνα του Linux. Η Google προώθησε την πλατφόρμα στους κατασκευαστές και τους μεταφορείς με την υπόσχεση παροχής ενός ευέλικτου, αναβαθμίσιμου συστήματος. Μέχρι το 2008, τόσο η Nokia όσο και το BlackBerry ανακοίνωσαν έξυπνα τηλέφωνα με βάση το touch για να ανταγωνιστεί το iPhone 3G και η εστίαση του Android τελικά μετατράπηκε σε απλές οθόνες αφής. Το πρώτο εμπορικά διαθέσιμο smartphone που τρέχει το Android ήταν το HTC Dream, επίσης γνωστό ως T-Mobile G1, που ανακοινώθηκε στις 23 Σεπτεμβρίου 2008.

Στις 5 Νοεμβρίου 2007, η Open Handset Alliance, μια κοινοπραξία εταιρειών τεχνολογίας όπως η Google, κατασκευαστές συσκευών όπως HTC, Motorola και Samsung, ασύρματες εταιρείες όπως η Sprint και η T-Mobile και κατασκευαστές chipset όπως η Qualcomm και η Texas Instruments με στόχο την ανάπτυξη της πρώτης πραγματικά ανοικτής και ολοκληρωμένης

πλατφόρμας για κινητές συσκευές. Μέσα σε ένα χρόνο η Open Handset Alliance αντιμετώπισε δύο άλλους ανταγωνιστές ανοιχτής πηγής, το Symbian Foundation και το LiMo Foundation, το τελευταίο αναπτύσσοντας επίσης ένα κινητό λειτουργικό σύστημα όπως το Google. Τον Σεπτέμβριο του 2007, το InformationWeek κάλυψε μια μελέτη Evalueserve που ανέφερε ότι η Google είχε καταθέσει αρκετές αιτήσεις ευρεσιτεχνίας στον τομέα της κινητής τηλεφωνίας.

Εικόνα 3.1: Το σήμα κατατεθέν του Android

Από το 2008, το Android έχει δει πολλές αναβαθμίσεις, οι οποίες έχουν βελτιώσει σταδιακά το λειτουργικό σύστημα, προσθέτοντας νέα χαρακτηριστικά και διορθώνοντας σφάλματα σε προηγούμενες κυκλοφορίες. Κάθε μεγάλη κυκλοφορία ονομάζεται με αλφαβητική σειρά μετά από επιδόρπιο ή ζαχαρούχο σκεύασμα, με τις πρώτες εκδόσεις του Android να ονομάζονται "Cupcake", "Donut", "Eclair" και "Froyo", με αυτή τη σειρά. Το 2010, η Google παρουσίασε τη σειρά συσκευών Nexus, μια σειρά στα οποία η Google συνεργάστηκε με διαφορετικούς κατασκευαστές συσκευών για την παραγωγή νέων συσκευών και την εισαγωγή νέων εκδόσεων Android. Η σειρά περιγράφηκε ότι έχει διαδραματίσει βασικό ρόλο στο ιστορικό του Android εισάγοντας νέες επαναλήψεις λογισμικού και πρότυπα υλικού σε όλο το πλάτος και έγινε γνωστός για το λογισμικό "free-bloat" με "έγκαιρες ενημερώσεις". Στη διάσκεψη προγραμματιστών τον Μάιο του 2013, η Google ανακοίνωσε μια ειδική έκδοση του Samsung Galaxy S4, όπου, αντί να χρησιμοποιήσει την προσαρμογή του Android, το τηλέφωνο έτρεξε "stock Android" και υποσχέθηκε να λάβει γρήγορα νέες ενημερώσεις συστήματος. Η συσκευή θα γίνει η αρχή του προγράμματος Google Play και θα ακολουθήσει άλλες συσκευές, όπως η έκδοση HTC One Google Play και η έκδοση MotoG

Google Play. Από το 2008 έως το 2013, ο Hugo Barra υπηρέτησε ως εκπρόσωπος των προϊόντων, εκπροσωπώντας το Android σε συνεντεύξεις Τύπου και το Google I / O, το ετήσιο συνέδριο της Google για τους προγραμματιστές. Έφυγε από την Google τον Αύγουστο του 2013 για να ενώσει την κινεζική εταιρεία κατασκευής τηλεφώνων Xiaomi. Λιγότερο από έξι μήνες νωρίτερα, ο τότε διευθύνων σύμβουλος της Google, Larry Page ανακοίνωσε σε blog, ότι ο Andy Rubin είχε μετακομίσει από το τμήμα Android για να αναλάβει νέα έργα στο Google και ότι η Sundar Pichai θα γίνει το νέο Android. Τον Ιούνιο του 2014, η Google ανακοίνωσε το Android One, ένα σύνολο μοντέλων αναφοράς υλικού που θα επιτρέψουν στους κατασκευαστές συσκευών να δημιουργήσουν εύκολα τηλέφωνα υψηλής ποιότητας με χαμηλό κόστος, σχεδιασμένα για καταναλωτές στις αναπτυσσόμενες χώρες. Τον Σεπτέμβριο, η Google ανακοίνωσε το πρώτο σετ Android One phones για κυκλοφορία στην Ινδία. Μια αναφορά από τις πληροφορίες τον Ιανουάριο του 2017 δήλωσε ότι η Google επεκτείνει το πρόγραμμα Android One με χαμηλό κόστος στις Ηνωμένες Πολιτείες, αν και η The Verge σημειώνει ότι η εταιρεία πιθανότατα δεν θα παράγει τις ίδιες τις ίδιες τις συσκευές.

Εικόνα 3.2: Το λογότυπο του Android

Η Google εισήγαγε τα smartphone Pixel και Pixel XL τον Οκτώβριο του 2016, τα οποία διατέθηκαν στο εμπόριο ως τα πρώτα τηλέφωνα που έκανε η Google και παρουσίαζαν αποκλειστικά ορισμένες λειτουργίες λογισμικού, όπως ο Βοηθός Google (Google Assistant), πριν από την ευρύτερη ανάπτυξη. Τα τηλέφωνα Pixel αντικατέστησαν τη σειρά Nexus, με μια νέα γενιά τηλεφώνων Pixel που ξεκίνησε τον Οκτώβριο του 2017.

Πίνακας 3.1: Οι εκδόσεις του Android

Εκδοση	Κωδική Ονομασία	Ημερομηνία
11.0	R	19-Φεβ-20
10.0	Q	3-Σεπ-19
9.0	Pie	6-Αυγ-18
8.1	Oreo	5-Δεκ-17
8.0		21-Αυγ-17
7.1	*Nougat*	4-Οκτ-16
7.0		22-Αυγ-16
6.0	*Marshmallow*	5-Οκτ-15
5.1	*Lollipop*	9-Μαρ-15
5.0		3-Νοε-14
4.4	*KitKat*	31-Οκτ-13
4.3	*Jelly Bean*	24-Ιουλ-13
4.2		13-Νοε-12
4.1		9-Ιουλ-12
4.0	*Ice Cream Sandwich*	16-Δεκ-11
3.2	*Honeycomb*	15-Ιουλ-11
2.3	*Gingerbread*	9-Φεβ-11
2.2	*Froyo*	20-Μαϊ-10
2.0	*Eclair*	26-Οκτ-09
1.6	*Donut*	15-Σεπ-09

3.3 Σύγκριση με άλλα λογισμικά

Η ανάπτυξη της τεχνολογίας έχει οδηγήσει στην αύξηση του ανταγωνισμού μεταξύ των εταιριών που κατασκευάζουν τις έξυπνες συσκευές και ιδιαίτερα των έξυπνων κινητών (smart phones) ο οποίος μεγαλώνει συνεχώς. Τα πεδία ανταγωνισμού ποικίλουν και είναι συνήθως σε συγκεκριμένα χαρακτηριστικά των συσκευών όπως η ποιότητα της κάμερας, ο σχεδιασμός της συσκευής (design), η δυναμική του επεξεργαστή , η χωρητικότητα αποθήκευσης. Όλα αυτά τα χαρακτηριστικά αποτελούν τη βάση της σύγκρισης μεταξύ των συσκευών αλλά χωρίς το κατάλληλο λειτουργικό σύστημα δεν θα είναι αρκετά ώστε κάποιος να κάνει την καλύτερη επιλογή συσκευής. Όταν γίνεται αναφορά στο λειτουργικό σύστημα δεν περιλαμβάνεται μόνο το interface αλλά όλο το σύνολο των χαρακτηριστικών που παρέχει. Για αυτό το λόγο παρακάτω παρατίθεται μια σύντομη σύγκριση του Android με ορισμένα άλλα λειτουργικά συστήματα. Παρότι το μερίδιο της αγοράς μοιράζεται μεταξύ αρκετών λειτουργικών συστημάτων η σύγκριση θα πραγματοποιηθεί σε σχέση με τα λειτουργικά iOS της Apple και των Windows αφού είναι τα πιο διαδεδομένα και αυτά που συγκεντρώνουν την προσοχή του καταναλωτικού κοινού. Η σύγκριση στηρίζεται σε πέντε συγκεκριμένες κατηγορίες χαρακτηριστικών οι οποίες είναι η προσιτότητα, οι εφαρμογές, η ενημέρωση, η προσαρμοστικότητα(Customizability) και το περιβάλλον (interface).

- Προσιτότητα

Το πρώτο χαρακτηριστικό που θα συγκρίνουμε είναι ίσως και το πιο βασικό είναι η τιμή. Είναι γεγονός ότι η τιμή αφορά ολόκληρη τη συσκευή και όχι μεμονωμένα το λειτουργικό αλλά επειδή αυτά τα δύο είναι άρρηκτα συνδεδεμένα , η τιμή της συσκευής αντικατοπτρίζει σε μεγάλο βαθμό και εκείνη του λειτουργικού. Η Apple όπως ξέρουμε διαθέτει smart phones στην αγορά τα οποία έχουν αρκετά τσιμπημένη τιμή, που δεν την αντέχουν πολλοί. Το Android απ' την άλλη κατέχει μεγαλύτερη γκάμα κινητών με ποίκιλες τιμές. Παρομοίως και τα Windows έχουν αρκετά προσιτές-χαμηλές τιμές. Σε αυτό που υπερτερεί το Android είναι η τεράστια ποικιλία συσκευών , μιας και αρκετοί κατασκευαστές το έχουν επιλέξει σαν λειτουργικό των συσκευών τους.

- Εφαρμογές

Εδώ υπάρχει μια μεγάλη διαμάχη όχι τόσο σχετικά με τον αριθμό αλλά με την ποιότητα των εφαρμογών. Παραθέτοντας ενδεικτικά κάποια νούμερα το Android κατέχει περίπου 3 εκατομμύρια εφαρμογές , το iOS 2.2 εκατομμύρια και τα Windows περίπου 500.000. Παρ όλα αυτά παραδοσιακά το iOS υπήρξε μια πιο κερδοφόρα πλατφόρμα για προγραμματιστές οπότε υπήρξε μια τάση για νέες εφαρμογές να εμφανίζονται εκεί πρώτα, κάτι που αρχίζει να αλλάζει καθώς το μερίδιο αγοράς του Android συνεχίζει να αυξάνεται. Επίσης υπάρχει η αντίληψη ότι η Apple έχει περισσότερος «ποιοτικές» εφαρμογές σε σχέση με το Android. Αυτό αλλάζει αλλά εν μέρει ισχύει καθώς εφαρμογές για μουσική, φωτογραφίες, βίντεο και άλλα υπάρχουν σε μεγάλη γκάμα και καλύτερης ποιότητας στο Apple Store από ότι στο Google Play. Δεν μπορούμε να πούμε το ίδιο για τα Windows καθώς βρίσκονται αρκετά πίσω όσο αναφορά και αριθμό και ποιότητα, συνεπώς είναι εκτός ανταγωνισμού.

- Ενημέρωση Λογισμικού

Όσο αφορά την ενημέρωση λογισμικού να αναφέρουμε ότι και οι 3 ενημερώνονται τακτικά. Οι εταιρίες φροντίζουν απ' την μία να βγάζουν μεγάλες αναβαθμίσεις με επανασχεδιασμό και νέες ενέργειες και απ' την άλλη μικρότερες με διορθώσεις σφαλμάτων. Ίσως η Apple να έχει ένα μικρό προβάδισμα καθώς κάθε χρόνο αφήνει πίσω της μια συσκευή όσο αφορά τις αναβαθμίσεις, αλλά τα καταφέρνει καλύτερα απ' το Android που είναι στο έλεος των εκάστοτε εταιριών, και έχει τους πελάτες να περιμένουν για τις αναβαθμίσεις.

- Προσαρμοστικότητα (Customization)

Σε αυτή την κατηγορία το Android φαίνεται να υπερτερεί σημαντικά. Διαθέτει ποίκιλες εφαρμογές για να αλλάξεις το καθετί. Μπορείς να εγκαταστήσεις διαφορετικούς launcher που θα αλλάξουν το στυλ του UI. Επίσης μπορείς να τοποθετήσεις lock screen, background, widgets, ενώ στο συγκεκριμένο κομμάτι τα άλλα 2 λογισμικά έχουν περιορισμένες δυνατότητες. Το iOS από τη μεριά του παρέχει κάποιες αλλαγές αλλά περιορισμένες. Πλέον

υποστηρίζει ξεχωριστά πληκτρολόγια αλλά δεν φτάνει το Android. Σε μια παρόμοια κατάσταση βρίσκονται και τα Windows.

Υπάρχουν πολλές κατηγορίες για να συγκρίνουμε τα λογισμικά όπως, συνδεσιμότητα, ασφάλεια, χάρτες, φωνητικοί βοηθοί κτλ αλλά πιστεύω περιπλέκουν την κατάσταση και την διχόνοια. Ίσως βέβαια να τα αναφέρω σε ένα επόμενο άρθρο ακόμα πιο αναλυτικά.

Καταλαβαίνουμε βέβαια ότι η επιλογή κινητού είναι μία καθαρά προσωπική άποψη, ανάλογα με τις ανάγκες του καθενός, και δεν υπάρχει υποβαθμισμένο κινητό, αναφερόμενος βέβαια στις 2 μεγάλες εταιρίες καθώς το WP έχει ακόμα πολλά να δώσει. Κάθε λογισμικό έχει τα υπέρ και τα κατά που μας οδηγούν στην απόκτηση του, και αν πρόκειται να συγκρίνουμε 2 διαφορετικά software αυτό δεν είναι τα τεχνικά χαρακτηριστικά.

Εσείς είστε ευχαριστημένοι από τις αναβαθμίσεις τις εταιρίας σας; Θα αλλάζατε στην επόμενη αγορά κινητού για να πάτε σε άλλο λογισμικό;

- Περιβάλλον και Σχεδιασμός

Σε ότι αφορά το εσωτερικό του software και συγκεκριμένα στο UI (User Interface) καταλαβαίνουμε καλύτερα την μεγάλη διαφορά μεταξύ των λογισμικών. Το Android απ' μεριά του διαφέρει έστω και λίγο απ' κινητό σε κινητό καθώς οι εταιρίες αφήνουν το δικό τους στίγμα στο UI. Η βασική αρχή όμως παραμένει η ίδια, στην αρχική οθόνη υπάρχουν συντομεύσεις εφαρμογών, με την μπάρα ειδοποιήσεων στο πάνω μέρος, και το συρτάρι εφαρμογών όπως ονομάζεται στην συνέχεια. Το iOS έκανε μια μεγάλη αλλαγή στις τελευταίες ενημερώσεις. Οι εφαρμογές βρίσκονται όλες στην αρχική, πλέον διαθέτει και μία μπάρα ειδοποιήσεων απο την οποία είναι δυνατή η πρόσβαση σε ένα control center για βασικές ρυθμίσεις και τις ειδοποιήσεις. Τα Windows μοιάζουν αρκετά με εκείνα του υπολογιστή, καθώς αποτελούνται απ' εικονίδια τα οποία αλλάζουν μέγεθος και θέση για να ταιριάζουν στον χρήστη. Επίσης μπορείτε να τα διαμορφώσετε όπως εσείς προτιμάτε.

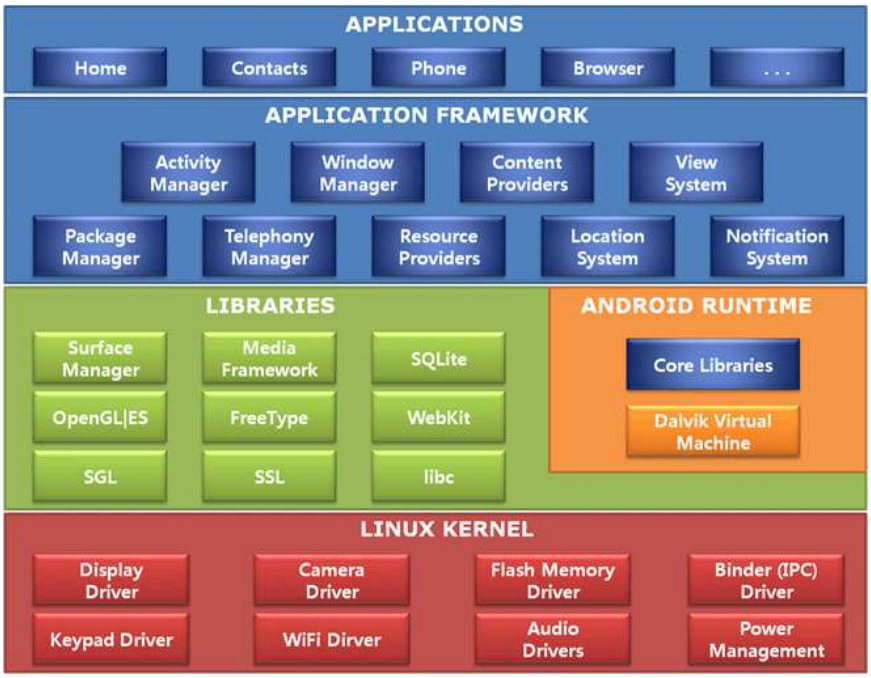
3.4 Επίπεδα Αρχιτεκτονικής

Η αρχιτεκτονική του android στηρίζεται πάνω σε μια στοίβα επίπεδων. Όπως βλέπουμε στην εικόνα 2.3 τα επίπεδα αυτά είναι τέσσερα (4) και παρουσιάζονται στην συνέχεια.

- Linux Kernel

Στο χαμηλότερο επίπεδο δηλαδή στη βάση της αρχιτεκτονικής βρίσκεται ο πυρήνας των linux. Πρόκειται για ένα πυρήνα λειτουργικού συστήματος που χρησιμοποιείτε από την οικογένεια των λειτουργικών συστημάτων Linux , το οποίο βασίζεται στο Unix(δεν χρησιμοποιεί κώδικα από αυτό αλλά θεωρείτε παρεμφερές σύστημα – ο ακριβής όρος είναι Unix –οειδές). Ο ρόλος του είναι να παρέχει τα απαιτούμενα προγράμματα οδήγησης (drivers). Πιο συγκεκριμένα , κάποια από τα βασικά στοιχεία που παρέχει είναι:

- υποστήριξη πολυεπεξεργαστικών συστημάτων (SMP)
- πραγματική πολυδιεργασία
- εικονική μνήμη
- διαμοιραζόμενες βιβλιοθήκες
- σωστή διαχείριση μνήμης
- δικτύωση μέσω TCP/IP



Εικόνα 3.3: Το διάγραμμα της αρχιτεκτονικής του Android

- Libraries

Πάνω από τον πυρήνα του Linux υπάρχουν οι φυσικές βιβλιοθήκες που περιέχουν των κώδικα του λειτουργικού και βοηθάνε ώστε να παρέχονται τα απαιτούμενα χαρακτηριστικά στον χρήστη. Για παράδειγμα η βιβλιοθήκη WebKit είναι υπεύθυνη για την υποστήριξη του προγράμματος περιήγησης (web browser), η SQLite παρέχει υποστήριξη αποθήκευσης στην βάση δεδομένων , ενώ η FreeType για την υποστήριξη της γραμματοσειράς.

- Android Runtime

Στο ίδιο επίπεδο συναντάμε το Android Runtime, το οποίο περιέχει τις βιβλιοθήκες πυρήνα (core libraries) και την εικονική μηχανή Dalvik (Dalvik virtual machine). Οι βιβλιοθήκες πυρήνα είναι ένα σύνολο βασικών βιβλιοθηκών που επιτρέπουν στους προγραμματιστές να γράψουν εφαρμογές χρησιμοποιώντας JAVA. Η εικονική μηχανή Dalvik επιτρέπει κάθε εφαρμογή να τρέξει την δικιά της εργασία. Πρόκειται για μια εξειδικευμένη εικονική μηχανή, ειδικά διαμορφωμένη για κινητές συσκευές που έχουν περιορισμένη μνήμη και ισχύ με στόχο την γρηγορότερη απόδοση.

- Application Framework

Στο επόμενο επίπεδο βρίσκεται το Android framework το οποίο περιλαμβάνει τα API(Application Programming Interface) του Android, όπως UI (User Interface), τηλεφωνία, πόρους, τοποθεσίες, παροχείς περιεχομένου (δεδομένα) και διαχειριστές πακέτων. Πέραν αυτών παρέχει πολλές κατηγορίες και διεπαφές για ανάπτυξη εφαρμογών Android από τους προγραμματιστές.

- Applications

Στην κορυφή της αρχιτεκτονικής βρίσκεται το επίπεδο εφαρμογών. Σε αυτό περιλαμβάνονται τόσο οι εφαρμογές που είναι προ-εγκατεστημένες και αφορούν τη λειτουργία της συσκευή του χρήστη (τηλέφωνο, επαφές, μουσική κ.α.), όσο και εκείνες που εγκαθιστά από μόνος του ο χρήστης σύμφωνα με τις ανάγκες του. Όλες αυτές χρησιμοποιούν το Android framework , το οποίο με τη σειρά του χρησιμοποιεί το android runtime και τις βιβλιοθήκες του Android και αυτά με τη σειρά τους χρησιμοποιούν το Linux Kernel.

4. ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΗΣ ΣΕ ANDROID

Προκειμένου ένα προγραμματιστής να αναπτύξει μια εφαρμογή για κάποια κινητή συσκευή θα πρέπει να λάβει υπόψη αρκετές παραμέτρους. Αυτές οι παράμετροι σχετίζονται με τα χαρακτηριστικά και τις δυνατότητες που έχουν οι συσκευές. Τέτοια χαρακτηριστικά είναι η ανίχνευση της θέσης της συσκευής και η κάμερα. Επίσης μπορεί να είναι η κατανάλωση της ενέργειας της μπαταρίας ή η μικρότερη ισχύς των επεξεργαστών (σε σχέση με συσκευές που διακρίνονται από ποιοτικότερα χαρακτηριστικά όπως οι σταθεροί υπολογιστές). Σημαντική παράμετρος στην ανάπτυξη της εφαρμογής μιας τέτοιας συσκευής αποτελεί επίσης και το ευρύ φάσμα στα μεγέθη των οθονών, των προδιαγραφών υλικού, στο customization που διαφοροποιείτε αναλόγως του λογισμικού των συσκευών καθώς επίσης και των αλλαγών σε κάθε πλατφόρμα.

Το βασικό εργαλείο που απαιτείτε για να δημιουργηθεί μια εφαρμογή είναι ένα ολοκληρωμένο αναπτυξιακό περιβάλλον (integrated development environment - IDE). Μέσα σε αυτό γίνονται οι απαραίτητες δοκιμές με τη χρήση εξομοιωτών που περιλαμβάνονται σε αυτά τα περιβάλλοντα και στη συνέχεια υποβάλλονται σε δοκιμές πεδίου. Η ύπαρξη αυτών των εξομοιωτών είναι πολύ σημαντική αφού παρέχουν την δυνατότητα στους προγραμματιστές να κάνουν τις δοκιμές που χρειάζονται χωρίς δαπάνη και χωρίς να απαιτείτε φυσική πρόσβαση σε κάποια κινητή συσκευή.

Ο σχεδιασμός διεπαφής χρήστη για κινητά (User Interface - UI) είναι επίσης απαραίτητος. Πρόκειται για το οπτικό μέρος της εφαρμογής με το οποίο αλληλεπιδρά ο χρήστης. Το UI εξετάζει τους περιορισμούς και τα περιβάλλοντα, την οθόνη, την είσοδο και την κινητικότητα ως περιγράμματα για το σχεδιασμό. Ο χρήστης είναι συχνά το επίκεντρο αλληλεπίδρασης με τη συσκευή του και η διασύνδεση περιλαμβάνει στοιχεία υλικού και λογισμικού. Η είσοδος χρήστη επιτρέπει στους χρήστες να χειρίζονται ένα σύστημα και η έξοδος της συσκευής επιτρέπει στο σύστημα να υποδείξει τα αποτελέσματα του χειρισμού των χρηστών. Οι περιορισμοί σχεδιασμού του UI περιλαμβάνουν περιορισμένους παράγοντες προσοχής και μορφής, όπως το μέγεθος της οθόνης μιας κινητής συσκευής για το χέρι του χρήστη. Το γενικό

πλαίσιο του UI υποδεικνύεται από τη δραστηριότητα των χρηστών, όπως η τοποθεσία και ο προγραμματισμός που μπορούν να εμφανιστούν από τις αλληλεπιδράσεις των χρηστών μέσα σε μια εφαρμογή. Γενικά, ο στόχος του σχεδιασμού του UI είναι η δημιουργία ενός περιβάλλοντος που είναι κύριο λόγο κατανοητό και προπάντων εύκολο στη χρήση.

Το UI (πολλές φορές ονομάζεται και front-end) βασίζεται στην υποστήριξη που παρέχει το επίπεδο πρόσβασης δεδομένων (data access layer ή πιο γνωστό ως back-end) προκειμένου να αποκτήσει πρόσβασης στο σύστημα. Το back-end διευκολύνει τη δρομολόγηση δεδομένων, την ασφάλεια, τον έλεγχο ταυτότητας, την εξουσιοδότηση, την εκτός σύνδεσης λειτουργία (off-line) και την ενορχήστρωση των υπηρεσιών. Αυτή η λειτουργικότητα υποστηρίζεται από ένα μείγμα μέσων του λογισμικού , συμπεριλαμβανομένων των διακομιστών κινητής εφαρμογής (mobile app servers), του MBaaS (Mobile Backend as a service) και της υποδομής SOA(service-oriented architecture).

5. Η ΝΟΣΟΣ ΑΛΤΣΧΑΙΜΕΡ

5.1 Γενική παρουσίαση

Πρόκειται για μια νόσος η οποία αναπτύσσεται προοδευτικά με κύριο χαρακτηριστικό της την καταστροφή της μνήμης καθώς επίσης και άλλων νοητικών λειτουργιών του εγκεφάλου. Αποτελεί τον πρωτεύονται παράγοντα της άνοιας η οποία περιλαμβάνει μια ποικιλία διαταραχών το εγκεφάλου που οδηγούν σε εξάλειψη της πνευματικής λειτουργίας και των κοινωνικών δεξιοτήτων. Στη νόσο Αλτσχάιμερ, τα κύτταρα του εγκεφάλου εκφυλίζονται και πεθαίνουν με αποτέλεσμα σταδιακά την απώλεια της μνήμης και την εμφάνιση διανοητικής δυσλειτουργίας. Οι φαρμακευτικές αγωγές που χορηγούνται στους ασθενείς μαζί με τις στρατηγικές διαχείρισης (παρέχονται από τους θεραπευτές) επιβραδύνουν την εξέλιξη της νόσου παρέχοντας μια προσωρινή βελτίωση των συμπτωμάτων. Στόχος της θεραπείας είναι η συντήρηση της ανεξαρτησίας των ασθενών μέσω της μεγιστοποίησης της λειτουργίας.

Συμπτώματα

Τα πιο βασικά συμπτώματα της νόσου είναι η διαταραχή της μνήμης καθώς και των νοητικών λειτουργιών που παρουσιάζονται αργότερα όπως οι διαταραχές προσοχής και της συγκέντρωσης, του προσανατολισμού όσο αναφορά τον χρόνο και τον τόπο, η εξασθένιση των λειτουργικών ικανοτήτων του ασθενούς, τα ψυχολογικά συμπτώματα , απώλεια γλωσσικών ικανοτήτων (αφασία).

Διαταραχή της μνήμης

Αποτελεί το πρωτεύον γνώρισμα της νόσου. Αρχικά η σημασιολογική μνήμη (το είδος της μνήμης που περιλαμβάνονται οι γενικές αρχές και γνώσεις μας για τον κόσμο που πηγάζουν μέσω των εμπειριών, χωρίς όμως της παρουσίας των αναπαραστάσεων αυτών των εμπειριών δηλαδή το πότε και το που λάβαμε αυτές τις γνώσεις) δεν επηρεάζεται τόσο. Αντιθέτως, πάσχουν η λειτουργική και η επεισοδιακή μνήμη. Αξίζει να σημειωθεί ότι σύμφωνα με τον νόμο του Ribot η παλαιά μνήμη διατηρείται καλύτερα από την πρόσφατη.

Κατά τον εκφυλισμός της νόσου επιφέρεται και διαταραχή της σημασιολογικής μνήμης.

Διαταραχή γλωσσικών ικανοτήτων

Το επόμενο σημαντικό πράγμα που επηρεάζεται από τη νόσο είναι ο προφορικός και γραπτός λόγος. Ο ασθενής ενώ αρχικά κατανοεί τις λέξεις, μετέπειτα δεν κατανοεί σύνθετες εντολές και παρουσιάζει δυσχέρεια στην ομιλία καθώς και στη γραφή. Το λεξιλόγιο του γίνεται όλο και πιο φτωχό και συχνά αναζητεί την κατάλληλη λέξη και αφήνει ατελείς τις προτάσεις. Στα τελευταία στάδια της νόσου παρουσιάζει το φαινόμενο της αυτούσιας επανάληψης ενός λεκτικού μηνύματος το οποίο μόλις έχει ακούσει (ηχολαλία). Πρόβλημα παρουσιάζει και ο γραπτός λόγος. Ένα φαινόμενου αυτού είναι η γραφή με κεφαλαία γράμματα , ο λάθος τονισμός ή σύνθεσης των γραμμάτων καθώς επίσης και κακή ευθυγράμμιση του κειμένου.

Ψυχικές διαταραχές

Οι ασθενείς της νόσου μπορεί να εμφανίσουν μια ποικιλία από ψυχικές διαταραχές. Τέτοιες μπορεί να είναι η εμφάνιση ανησυχίας, η αδράνεια και η παθητικότητα. Παρουσιάζουν άγχος και φοβίες ακόμη και ανασφάλεια. Συχνά αδιαφορούν για την εμφάνιση τους και την προσωπική τους υγιεινή. Πέραν αυτών , παρατηρείται διαταραχή του ύπνου όπως αϋπνίες ή αναστροφής του ωραρίου του ύπνου δηλαδή να κοιμούνται την ημέρα με αποτέλεσμα να μην μπορούν να κοιμηθούν τη νύχτα. Με την πάροδο της νόσου οι ασθενείς είναι πιθανό να παρουσιάσουν ψευδαισθήσεις και παραληρηματικά επεισόδια.

5.2 Στάδια της Νόσου

Ένα χαρακτηριστικό της νόσου είναι ότι δεν παρουσιάζει μια σταθερή ομοιογένεια στο εξελικτικό της κομμάτι κάτι που την κάνει απρόβλεπτη στην πορεία της εξέλιξης της από τους γιατρούς. Άλλοι ασθενής βιώνουν μια ταχεία κλιμάκωση της με αποτέλεσμα να φεύγουν από τη ζωή σε διάστημα λίγων μηνών , ενώ σε άλλους εξελίσσεται με αργούς ρυθμούς παρατείνοντας τη διάρκεια ζωής τους αρκετά χρόνια. Προκειμένου να γίνεται διάκριση της σοβαρότητας κάθε ασθενούς έχουν χρησιμοποιηθεί διάφορες φόρμουλες όπως το Mini Mental State Examination (αναλόγως αν το αποτέλεσμα που

προκύπτει είναι μεγαλύτερο του 20 , μεταξύ 10 και 19 ή μικρότερο του 10 η νόσος χαρακτηρίζεται ως ήπια , μέτρια ή σοβαρή αντίστοιχα). Η πιο διαδεδομένη κλίμακα είναι εκείνη του Reisberg η οποία διαιρεί τη γνωστική εξασθένιση σε επτά στάδια άνοιας:

Στάδιο 1ο : Καμία νοητική εξασθένιση: Η κλινική εικόνα του ασθενή δεν παρουσιάζει καμία νοητική δυσχέρεια.

Στάδιο 2ο : Μέτρια νοητική εξασθένιση: Ο ασθενής αναφέρει προβλήματα μνήμης που αφορούν είτε τοποθέτηση οικείων αντικειμένων είτε οικείων ονομάτων. Κλινική εικόνα επίσης φυσιολογική.

Στάδιο 3ο : Μέτρια νοητική εξασθένιση: το πρώτο ανησυχητικό στάδιο. Σε αυτό ο ασθενής εκδηλώνει ανεπάρκεια μνήμης σε περισσότερους από ένα τομείς όπως α) έντονος αποπροσανατολισμός σε ταξίδια άγνωστων περιοχών, β) σχετικά χαμηλή εργασιακή απόδοση, γ) παρατήρηση των οικείων δυσχέρειας του ασθενούς σε ονόματα και λεξιλόγιο, δ) χαμηλή απομνημόνευση περιεχομένου μετά από ανάγνωση ενός κειμένου, ε) λανθασμένη τοποθέτηση αντικειμένων αξίας, στ) Κλινική εικόνα που παρουσιάζει δυσκολία συγκέντρωσης.

Στάδιο 4ο : Μέτρια νοητική εξασθένιση: Πιο ξεκάθαρα συμπτώματα εξασθένισης στην κλινική εικόνα. Οι τομείς της ζωής του ασθενούς που επηρεάζονται είναι α) μειωμένη γνώση πρόσφατων συμβάντων, β) πρώιμο στάδιο απώλειας μνήμης για την προσωπικά του βιώματα, γ) αντιμετώπιση δυσκολίας σε επίλυση μαθηματικών πράξεων, δ) δυσκολία διαχείρισης οικονομικών συναλλαγών καθώς και μετακινήσεων.

Στάδιο 5ο : Σχετικά σοβαρή νοητική εξασθένιση: ο ασθενής πλέον χρειάζεται βοήθεια προκειμένου να ζήσει στην καθημερινότητα του. Δυσκολεύεται να φέρει στη μνήμη του διευθύνσεις που γνωρίζει, τηλεφωνικούς αριθμούς που χρησιμοποιούσε με άνεση μέχρι τότε, ονόματα άμεσων συγγενών του (π.χ. εγγόνια), το όνομα του σχολείου ή του Πανεπιστημιακού ιδρύματος που μπορεί να φοίτησε και γενικότερα σημαντικές στιγμές που έζησε πρόσφατα. Συχνά μπορεί να χάνει την αίσθηση της χρονικής περιόδου που ζει (ημερομηνία , εποχή κλπ) ή της τοποθεσίας που διαμένει. Οι ασθενείς της

συγκεκριμένης κατηγορίας μπορούν να θυμούνται μόνο βασικών γεγονότων που αφορούν εκείνους ή τους οικείους τους. Συγκρατούν ακόμα την ταυτότητα τους (το όνομα τους) και των οικογενειακών μελών τους (σύζυγο , τέκνων). Επίσης μπορούν να ανταπεξέλθουν χωρίς βοήθεια σε οικιακές εργασίες, όπως την ετοιμασία του φαγητού , αλλά μπορεί να δυσκολεύονται να επιλέξουν την σωστή ένδυση τους.

Στάδιο 6ο : Σοβαρή νοητική εξασθένιση: Ανά διαστήματα μπορεί να μην θυμούνται το όνομα της/του γυναίκας/άντρα τους στους οποίους στηρίζονται προκειμένου να ανταπεξέλθουν καθημερινά. Είναι συχνό το φαινόμενο να ξεχνούν πράγματα τα οποία έζησαν τελευταία , καθώς επίσης και παλαιότερες ιστορίες που τους έχουν βιώσει. Στην μνήμη τους παραμένουν κάποιες γνώσεις από την πρότερη ζωή τους αλλά χωρίς λεπτομέρειες και συχνά μπερδεμένες. Υπάρχει απουσία πληροφορίας σχετικά με το πότε και που ζουν και διαμένουν. Συχνά παρουσιάζουν πρόβλημα στην απλή αρίθμηση προς το 10 και πάνω. Η βοήθεια σχετικά με τις απλές τους ανάγκες γίνεται ακόμα πιο απαραίτητη (πχ στο να πηγαίνουν στην τουαλέτα μιας και παρουσιάζουν ακράτεια). Βοήθεια επίσης προβλέπεται και στις μετακινήσεις τους (σποραδικά φαίνεται να ανταπεξέρχονται στην μετακίνηση τους προς γνωστές τους τοποθεσίες). Παρουσιάζουν επίσης δυσκολίες στο πρόγραμμα του ύπνου τους. Διατηρούν την γνώση του ονόματος τους και αρκετές φορές μπορούν να ξεχωρίσουν αν όντως γνωρίζουν ή όχι πρόσωπα. Σε αυτό το στάδιο παρουσιάζονται για πρώτη φορά αλλοιώσεις στην ψυχοσύνθεση και τον χαρακτήρα του ασθενή. Σε αυτές τις αλλοιώσεις κυριαρχεί μια ποικιλομορφία όπως α) παραισθησιακή συμπεριφορά (π.χ. να φαντάζονται παρουσίες στο χώρο ή γεγονότα που δεν έχουν συμβεί) β) ιδεοψυχαναγκαστικά συμπτώματα (π.χ. ο ασθενής μπορεί να επαναλαμβάνει συνεχώς μια ενέργεια της καθημερινότητας του) , γ) νοητική αβουλία (π.χ. η βραχύβια συντήρησης μια σκέψης στο νου του οδηγεί στην έλλειψη ικανότητας να θέλει να πράξει το οτιδήποτε που έχει σαν αποτέλεσμα την ανικανότητα να ορίσει τον λόγο των πράξεων του).

Στάδιο 7ο : Πολύ Σοβαρή νοητική εξασθένιση: Απώλεια των λεκτικών ικανοτήτων και έλλειψη έναρθρου λόγου. Δυσχέρεια στις ψυχοκινητικές

ικανότητες όπως το βάδισμα και απαραίτητη η βοήθεια σε καθημερινές βασικές λειτουργίες όπως σίτιση και προσωπική υγιεινή. Ο εγκέφαλος χωλαίνει στον χειρισμό του σώματος.

5.3 Αντιμετώπιση της Νόσου

Προκειμένου να μιλήσουμε για αντιμετώπιση της άνοιας γίνεται αντιληπτό ότι πρέπει αρχικά να κατηγοριοποιήσουμε κάποιες μορφές της με γνώμονα το βαθμό θεραπείας της. Με αυτή την παράμετρο η άνοιες διακρίνονται αναστρέψιμες (δηλαδή υπάρχει αιτιολογική θεραπεία) και εκείνες που είναι μη αναστρέψιμες. Οι αναστρέψιμες μορφές άνοιας αποτελούν ένα ποσοστό της τάξεως του 10% όλων των νέων περιπτώσεων. Οι μη αναστρέψιμες περιπτώσεις κατέχουν το μεγαλύτερο μέρων των περιπτώσεων. Μέχρι στιγμής στις μη αναστρέψιμες μορφές δεν έχει βρεθεί κάποια θεραπεία που να αντιμετωπίσει πλήρως τη νόσο αλλά υπάρχουν τακτικές προσέγγισης οι οποίες βελτιώνουν τα συμπτώματα. Αυτή η θεραπεία είναι γνωστή ως συμπτωματική και είναι υπεύθυνη για τη συντήρηση και την επιμήκυνση της καλύτερης λειτουργικότητας του ασθενούς και κατ' επέκταση της βελτίωσης της ποιότητας ζωή του ασθενούς. Το βέλτιστο αποτέλεσμα επιτυγχάνεται με το συνδυασμό της φαρμακευτικής αγωγής με τακτικές μη φαρμακευτικής προσέγγισης καθώς επίσης και ενός υποστηρικτικού οικογενειακού περιβάλλοντος.

Η Νόσος πέραν του ασθενούς απασχολεί και τον στενό κοινωνικό του κύκλο. Η μια προσέγγιση της θεραπείας της σχετίζεται με τη λήψη φαρμάκων. Η σφαιρική προσέγγιση της συνήθως χρειάζεται την από κοινού συνεννόηση μεταξύ του νευρολόγου και των φροντιστών έτσι ώστε οι πρακτικές να επιφέρουν το βέλτιστο αποτέλεσμα για τον ασθενή. Μέχρι τώρα δεν έχει ανακαλυφθεί κάποια θεραπεία με την οποία επιβραδύνεται η εξέλιξη του εκφυλισμού των καταστάσεων που σχετίζονται με το νευρολογικό σύστημα και των μη-αναστρέψιμων μορφών άνοιας. Επομένως όσες τακτικές αντιμετώπισης εφαρμόζονται με σκοπό καλυτερεύσουν τα συμπτώματα της νόσου.

Φαρμακευτική

Μέχρι πρότινος δεν έχει ανακαλυφθεί κάποια θεραπεία η οποία επιβραδύνει την κλιμάκωση των παθολογικών καταστάσεων που πρόκειται να επέλθουν οι οποίες σχετίζονται με τις νευροκεκφυλιστικές μορφές άνοιας. Οι θεραπείες που υφίστανται μέχρι τώρα καταφέρνουν μόνο να καλυτερεύουν τα συμπτώματα τις νόσου. Αυτό γίνεται πραγματικότητα με την προσέγγιση των υγειών νευρικών κυκλωμάτων, αλλάζοντας τη λειτουργία τους έτσι ώστε αδράξουν τους μηχανισμούς εφεδρείας και βιολογικής προσαρμογής. Όμως, όσο οι αρνητικές μεταβολές στον εγκέφαλο γίνονται όλο και μεγαλύτερες όσο τα χρόνια περνούν , οι μηχανισμοί που ενεργοποιήθηκαν για να αντιστρέψουν την κατάσταση γίνονται περισσότερο αναποτελεσματικοί και η κλινική εικόνα του ασθενούς χειροτερεύει. Τα φάρμακα τα οποία έχουν εγκριθεί και παρέχονται για τη θεραπεία του Alzheimer είναι:

Α) Αναστολείς ακετυλοχολινεστεράσης

Η λειτουργία αυτών των φαρμάκων είναι να αυξάνουν τη δραστηριότητα της ακετυλοχολίνης , που είναι ένας νευροδιαδιβαστής. Το χολινεργικό σύστημα είναι κατά μεγάλο μέρος υπεύθυνο για τη λειτουργία της μνήμης καθώς και της συγκέντρωση. Πιο συγκεκριμένα:

- Δρουν κατά το ήπιο και μέτριας βαρύτητας στάδιο της νόσου
- Τα αποτελέσματα που θα έχει στους ασθενείς δεν μπορούν να υπολογιστούν μιας και σε κάποιους μπορεί να δράσουν σε μεγαλύτερο βαθμό με αποτέλεσμα την καλυτέρευση των νοητικών τους λειτουργιών ενώ σε άλλους η βελτίωση να είναι πολύ μικρή.
- Κατά τη λήψη αυτών παρουσιάζονται τις περισσότερες φορές παρενέργειες όπως ναυτία, έμετοι και διάρροιες που διαρκούν μία με δύο εβδομάδες
- Σε κάποιες περιπτώσεις παρουσιάζεται και βελτίωση ορισμένων συμπεριφορικών συμπτωμάτων όπως η απάθεια και οι οπτικές ψευδαισθήσεις.

Β) Μεμαντίνη

Έχει αντίκτυπο στο γλουταμινεργικό σύστημα του εγκεφάλου το οποίο σχετίζεται με ένα μεγάλο αριθμό λειτουργιών με τις πιο σημαντικές να είναι η μνήμη και η μάθηση.

- Και σε αυτή τη κατηγορία φαρμάκων η δράση τους δεν μπορεί να υπολογιστεί με ακρίβεια μιας και διαφέρει από ασθενή σε ασθενή
- Σε μέτρια έως σοβαρά στάδια της νόσου παρουσιάζει την μεγαλύτερη αποτελεσματικότητα.

Σε ότι αφορά τις παρενέργειες η ζάλη είναι αυτή που εμφανίζεται πιο συχνά.

Μη φαρμακευτική

Έχει παρατηρηθεί ότι οι ασθενείς της νόσου μπορούν να ευεργετηθούν και από μη φαρμακευτικές τακτικές. Σε πολλές περιπτώσεις η εξέλιξη της νόσου μπορεί να εμφανίζει θετικότερο πρόσωπο εάν οι ασθενείς καταπιάνονται με δραστηριότητες στην καθημερινότητας τους. Αρχικά αν συντηρούνται σε επίπεδο σωματικής άσκησης (όπως περπάτημα ή γυμναστική) καθώς επίσης και σε πνευματικό επίπεδο μέσω απλών δραστηριοτήτων όπως ζωγραφική, πάζλ, εκμάθηση μουσικού οργάνου κλπ. Σημαντικό κομμάτι στη μη φαρμακευτική αντιμετώπιση καταλαμβάνει και η καλή διατροφή που μπορεί να ακολουθεί ο ασθενής.

Ασκήσεις για τη μνήμη

Όπως προαναφέρθηκε υπάρχουν τρόποι με τους οποίους μπορεί ένας ασθενής να καταπολεμήσει την αδυναμία της μνήμης μέσω μη φαρμακευτικών τακτικών. Σύμφωνα με τους επιστήμονες οι ασθενείς μπορούν να προβούν σε διάφορες ασκήσεις προκειμένου να ενισχύσουν τη λειτουργία του εγκεφάλου. Αυτές οι ασκήσεις θα πρέπει αρχικά να είναι διασκεδαστικές και ταυτόχρονα να παρέχουν πρόκληση και καινοτομία. Σύμφωνα με τον νευροεπιστήμονα David Eagleman προτείνονται στους ασθενείς να ξεκινούν με απλές καθημερινές αλλαγές όπως να βουρτσίζουν τα δόντια τους με το αντίθετο χέρι από ότι συνήθως ή να αλλάζουν διαδρομές από και προς το σπίτι σε σχέση με τα καθιερωμένα τους δρομολόγια. Η ιδέα πίσω από αυτό έχει να κάνει με το ότι ο εγκέφαλος εργάζεται μέσω συσχετισμών και γενικότερα όσο περισσότερες αισθήσεις χρησιμοποιούνται στην εκτέλεση των δραστηριοτήτων τόσο καλύτερα. Άλλες συνηθισμένες προτάσεις είναι η επίλυση παιχνιδιών Sudoku ή παιχνιδιών στα οποία αναζητούνται οι διαφορές μεταξύ εικόνων. Παρακάτω ακολουθεί μια λίστα με μια ποικιλία ασκήσεων:

- Ο ασθενής μπορεί να φτιάξει μια λίστα στην οποία να περιλαμβάνει διάφορα προϊόντα που θέλει να ψωνίσει από το super market. Στην συνέχεια προσπαθεί να την απομνημονεύσει. Μετά από μία ώρα δοκιμάζει να δει πόσα από αυτά μπορεί να θυμηθεί. Αυτή η άσκηση σχετίζεται με την ανάκληση της μνήμης.
- Αφού επισκεφθεί κάποιο νέο μέρος το οποίο δεν συνηθίζει να βρίσκεται, μπορεί να φτιάχνει νοερώς ένα χάρτη. Αυτή την άσκηση μπορεί να την πραγματοποιεί κάθε φορά που πηγαίνει σε ένα καινούριο μέρος.
- Προσπάθεια επίλυσης μαθηματικών πράξεων επίσης νοερώς χωρίς τη βοήθεια γραφικής ύλης ή υπολογιστικής μηχανής. Αν αυτό γίνεται και εν κινήσει θα κάνει πιο δύσκολη την άσκηση και πιο αποδοτική.
- Εκτέλεση συνταγών μαγειρικής (αν ο ασθενής συνηθίζει να μαγειρεύει μπορεί να δοκιμάσει κάποια τελείως νέα συνταγή ή ακόμη και να κάνει μαθήματα). Αυτό αποτελεί μια πολύ καλή άσκηση αφού εμπλέκονται πολλές αισθήσεις(γεύση , όραση, ακοή, αφή , όσφρηση).
- Εκμάθηση μιας ξένης γλώσσας. Ο εγκέφαλος θα ενεργοποιηθεί μέσω της ακοής.
- Εκμάθηση ενός μουσικού οργάνου.
- Αναπτύξτε μια χειρονακτική δραστηριότητα όπως ζωγραφική , πλέξιμο ή πάζλ. Αυτό θα εξασκήσει τις ικανότητες χεριού – ματιού.
- Ενασχόληση με ένα άθλημα. Μια πολύ καλή άσκηση αφού συνδυάζει τόσο σωματική όσο και πνευματική εξάσκηση.

6 ΒΑΣΙΚΑ ΕΡΓΑΛΕΙΑ ΑΝΑΠΤΥΞΗΣ

Προκειμένου να υλοποιηθεί η εφαρμογή της παρούσας εργασίας ήταν απαραίτητη η χρήση ενός περιβάλλοντος ανάπτυξης κώδικα. Πιο συγκεκριμένα χρειάστηκε ένα προγραμματιστικό περιβάλλον το οποίο θα έδινε την δυνατότητα ανάπτυξης μιας εφαρμογής για το λογισμικό του Android. Τέτοια περιβάλλοντα είναι το Eclipse και το Android Studio. Και τα δυο είναι ιδιαίτερα διαδεδομένα στους κύκλους των προγραμματιστών εφαρμογών Android. Το Eclipse για πολλά χρόνια ήταν αυτό που επιλεγόταν περισσότερο , αλλά τα τελευταία χρόνια το Android Studio είναι αυτό που έχει κερδίσει τη μερίδα του λέοντος μιας και έχει εκτοξευτεί. Το Eclipse παρείχε την δυνατότητα ανάπτυξης κώδικα σε πολλές γλώσσες προγραμματισμού όπως C, C ++, C #, COBOL, Fortran , JavaScript, Perl, PHP, Prolog, Python, R, Ruby. Το Android Studio αντιθέτως απευθύνεται αποκλειστικά στην ανάπτυξη εφαρμογών Android λογισμικού παρέχοντας ένα πλήρες περιβάλλον ανάπτυξης το οποίο περιλαμβάνει όλα τα απαραίτητα στοιχεία που χρειάζονται.

6.1 Android Studio

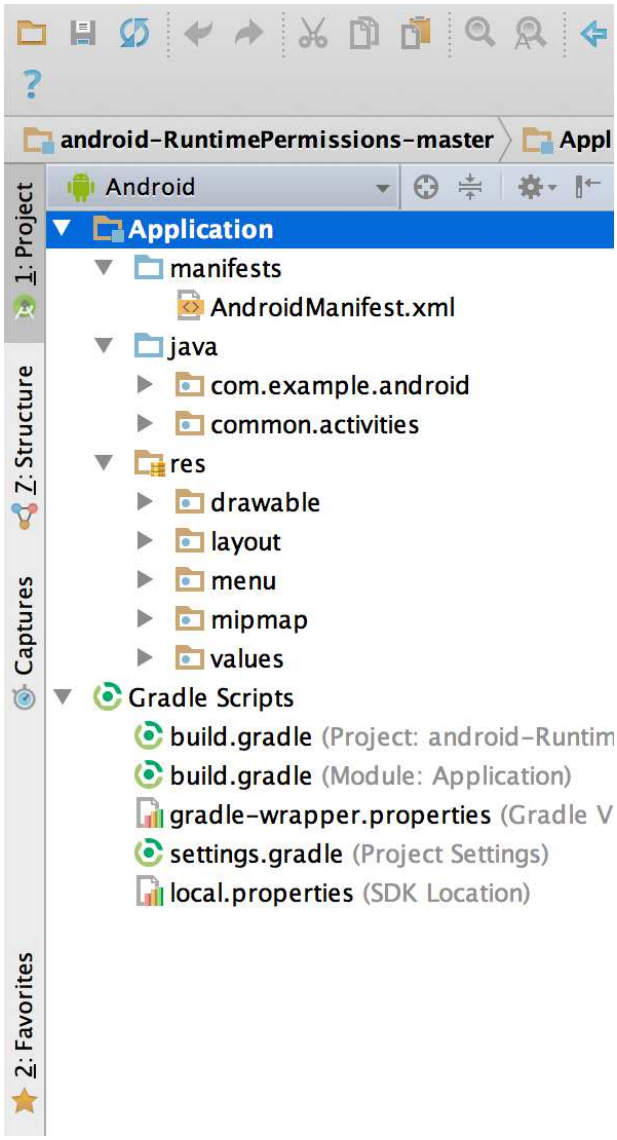
Το Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment - IDE) για το λειτουργικό σύστημα Android, το οποίο βασίζεται στο λογισμικό IntelliJ IDEA της JetBrains και έχει σχεδιαστεί ειδικά για την ανάπτυξη Android εφαρμογών.

Βασικά χαρακτηριστικά

- Ένα ευέλικτο σύστημα κατασκευής με βάση το Gradle
- Ένας γρήγορος και πλούσιος σε χαρακτηριστικά εξομοιωτής
- Ένα ενιαίο περιβάλλον όπου ο προγραμματιστής μπορεί να αναπτύξει μια εφαρμογή για όλες τις συσκευές Android
- Πρότυπα κώδικα για να σας βοηθήσουν να δημιουργήσετε κοινές λειτουργίες εφαρμογής
- Ενσωματωμένη υποστήριξη για την πλατφόρμα Google Cloud, διευκολύνοντας την ενσωμάτωση του Google Cloud Messaging και του App Engine

Δομή ενός project

Κάθε καινούριο project που ξεκινάει στο περιβάλλον του Android Studio περιλαμβάνει μία ή περισσότερες ενότητες (module) με αρχεία πηγαίου κώδικα και αρχεία πόρων. Οι τύποι ενοτήτων περιλαμβάνουν την ενότητα android εφαρμογής, την ενότητα της βιβλιοθήκης και την ενότητα της εφαρμογής της μηχανής Google. Από προεπιλογή , το android studio εμφανίζει τα αρχεία του project όπως φαίνεται στη εικόνα 6.1.1



Εικόνα 6.1.1 Τα αρχεία ενός project

Αυτή η προβολή οργανώνεται κατά ενότητα έτσι ώστε να παρέχεται στο χρήστη γρηγορότερη πρόσβαση στα βασικά αρχεία του project που αναπτύσσει. Όλα τα αρχεία που κτίζονται είναι ορατά στο υψηλότερο επίπεδο

κάτω από την ενότητα Gradle Scripts και κάθε ενότητα της εφαρμογής περιέχει τους εξής φακέλους:

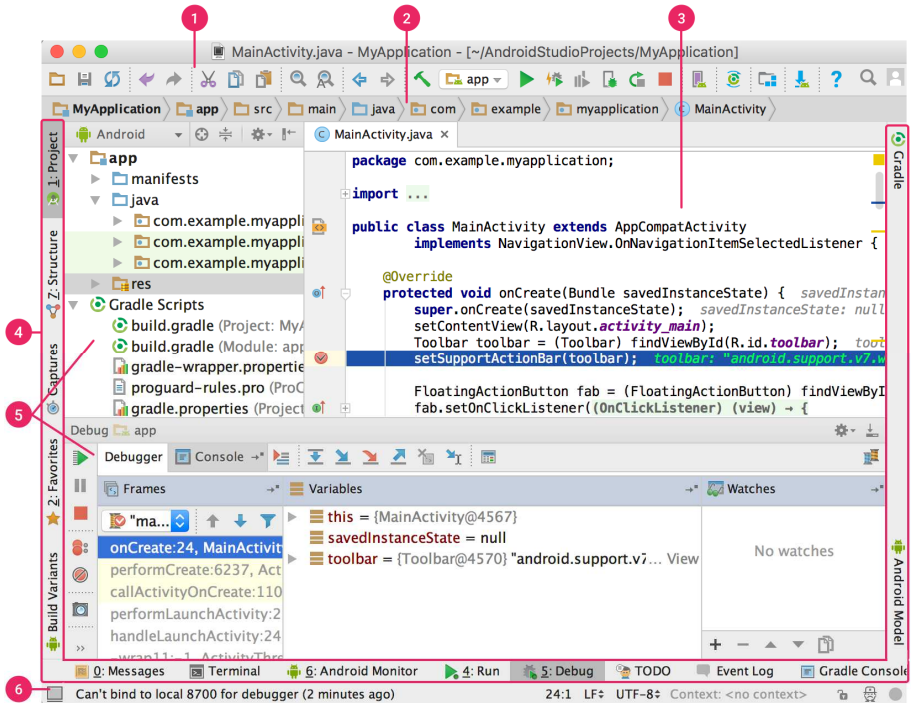
Φάκελος Manifest: περιέχει το αρχείο AndroidManifest.xml το οποίο περιγράφει βασικές πληροφορίες σχετικά με την εφαρμογή στο Android build tool (είναι το στοιχείο του Android SDK το οποίο κτίζει την εφαρμογή) και στο λειτουργικό του android στο οποίο θα τρέξει η εφαρμογή.

Φάκελος java: περιέχει τα αρχεία του πηγαίου κώδικα της Java

Φάκελος res: περιέχει όλα του πόρους που δεν έχουν πηγαίο κώδικα , όπως τις XML διατάξεις(layouts), τις συμβολοσειρές της διεπαφής του χρήστη (UI strings) και τις bitmap εικόνες.

Η διεπαφή του χρήστη(User Interface)

Το βασικό παράθυρο του Android Studio αποτελείται από διάφορες περιοχές όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 6.1.2 Τα αρχεία ενός project

- Η γραμμή εργαλείων (toolbar) επιτρέπει στον χρήστη να εκτελεί ένα εύρος ενεργειών όπως για παράδειγμα το να εκτελεί την εφαρμογή του καθώς επίσης και την χρήση των android εργαλείων.

- Η γραμμή πλοήγησης (navigation bar) βοηθάει στην περιήγηση εντός του project και στο άνοιγμα των αρχείων του. Παρέχει μια πιο συμπαγή παρουσίαση της δομής του project.
- Το παράθυρο σύνταξης (editor window) είναι η περιοχή όπου δημιουργείτε και τροποποιείτε ο κώδικας του χρήστη.
- Το παράθυρο γραμμής εργαλείων (tool window bar) βρίσκεται γύρω από το παράθυρο του IDE (Integrated Development Environment) και περιέχει κουμπιά τα οποία δίνουν τη δυνατότητα στο χρήστη να επεκτείνει ή να συμπτύξει της ενότητες και τους φακέλους του παραθύρου εργαλείων.
- Το παράθυρο εργαλείων (tool windows) δίνει πρόσβαση σε συγκεκριμένες ενέργειες όπως την διαχείριση του project , την αναζήτηση, τον έλεγχο εκδόσεων (version control) και άλλα.
- Η γραμμή κατάστασης (status bar) εμφανίζει την κατάσταση του project καθώς και του IDE , καθώς και διάφορα μηνύματα και προειδοποιήσεις.

Ο χρήστης μπορεί να οργανώσει το βασικό παράθυρο με τέτοιο τρόπο ώστε να δώσει περισσότερο χώρο στην οθόνη του , κρύβοντας ή μετακινώντας γραμμές εργαλείων και παράθυρα εργαλείων. Επίσης μπορεί να χρησιμοποιεί συντόμευση πληκτρολογίου προκειμένου να έχει πρόσβαση στις δυνατότητες που του παρέχει το περιβάλλον.

6.2 Η γλώσσα προγραμματισμού Java

Η Java είναι μια γλώσσα προγραμματισμού γενικής χρήσης που είναι ταυτόχρονα βασισμένη σε αντικείμενα αλλά και σχεδιασμένη ώστε να έχει όσο το δυνατόν λιγότερες εξαρτήσεις εφαρμογής.

Σκοπός είναι να επιτρέψει στους προγραμματιστές εφαρμογών να «γράψουν μία φορά και να τρέξουν οπουδήποτε» (Write Once, Run Anywhere - WORA), δηλαδή ο μεταγλωττισμένος κώδικας Java να μπορεί να τρέξει σε όλες τις πλατφόρμες που υποστηρίζουν την Java χωρίς την ανάγκη αντιγραφής . Οι εφαρμογές Java συνήθως μεταγλωττίζονται σε bytecode που μπορεί να τρέξει σε οποιαδήποτε εικονική μηχανή Java (Java Virtual Machine - JVM) ανεξάρτητα από την αρχιτεκτονική του υπολογιστή.

Από το 2016, η Java είναι μια από τις πιο δημοφιλείς γλώσσες προγραμματισμού που χρησιμοποιούνται , ιδιαίτερα για web εφαρμογές πελάτη-διακομιστή (client – server web applications). Η Java αναπτύχθηκε

αρχικά από τον James Gosling στην Sun Microsystems (η οποία από τότε έχει αποκτηθεί από την Oracle Corporation) και κυκλοφόρησε το 1995 ως βασικό στοιχείο της πλατφόρμας Java της Sun Microsystems. Η γλώσσα αποδίδει μεγάλο μέρος της σύνταξης της στη C και τη C ++ , αλλά έχει λιγότερες εγκαταστάσεις χαμηλού επιπέδου από ότι αυτές οι δύο. Η τελευταία έκδοση είναι η Java 10 , η οποία κυκλοφόρησε στις 20 Μαρτίου 2018, η οποία ακολούθησε την Java 9 μετά από μόλις έξι μήνες. Η Java 8 εξακολουθεί να υποστηρίζεται, αλλά δεν θα υπάρχουν περισσότερες ενημερώσεις ασφαλείας. Οι εκδόσεις που προηγούνται της Java 8 υποστηρίζονται από εταιρείες σε εμπορική βάση. π.χ. από την Oracle.

- Αρχές

Υπήρχαν πέντε πρωταρχικοί στόχοι στη δημιουργία της γλώσσας , οι οποίοι δήλωναν ότι πρέπει:

- να είναι απλή, αντικειμενοστρεφής και οικεία
- να είναι ανθεκτική και ασφαλής
- να είναι ουδέτερη στην αρχιτεκτονική και φορητή.
- να εκτελείται με υψηλή απόδοση
- να διερμηνεύεται, να νηματοποιείται και να είναι δυναμική

- Σύστημα Εκτέλεσης

Java JVM και Bytecode

Ένας σχεδιαστικός στόχος της Java είναι η φορητότητα, πράγμα που σημαίνει ότι τα προγράμματα που έχουν γραφτεί για την πλατφόρμα αυτή πρέπει να λειτουργούν ομοίως σε οποιοδήποτε συνδυασμό υλικού και λειτουργικού συστήματος με επαρκή υποστήριξη χρόνου εκτέλεσης. Αυτό επιτυγχάνεται με την κατάρτιση του κώδικα γλώσσας Java σε μια ενδιάμεση αναπαράσταση που ονομάζεται Java bytecode , αντί για τον κώδικα μηχανής που αφορά συγκεκριμένα την αρχιτεκτονική. Οι οδηγίες Java bytecode είναι ανάλογες με τον κώδικα του μηχανήματος, αλλά προορίζονται να εκτελούνται από μια εικονική μηχανή (Virtual Machine - VM) γραμμένη ειδικά για το υλικό

του κεντρικού υπολογιστή. Οι τελικοί χρήστες χρησιμοποιούν συνήθως ένα περιβάλλον εκτέλεσης Java (Java Runtime Environment - JRE) εγκατεστημένο στο δικό τους μηχάνημα για αυτόνομες εφαρμογές Java ή σε ένα πρόγραμμα περιήγησης ιστού για εφαρμογές Java. Οι τυπικές βιβλιοθήκες παρέχουν έναν γενικό τρόπο για την πρόσβαση σε λειτουργίες συγκεκριμένου κεντρικού υπολογιστή, όπως γραφικά, νήματα και δικτύωση .Η χρήση του καθολικού bytecode καθιστά εύκολη τη μεταφορά. Ωστόσο, η επιβάρυνση της διερμηνείας του bytecode σε οδηγίες μηχανής έκαναν τα διερμηνευμένα προγράμματα σχεδόν πάντα να τρέχουν πιο αργά από τα εγγενή εκτελέσιμα . Οι μεταγλωττιστές Just-in-Time (JIT) που μεταγλωτίζουν τα bytecodes σε κώδικα μηχανής κατά τη διάρκεια του χρόνου εκτέλεσης εισήχθησαν σε μεταγενέστερο στάδιο. Η ίδια η Java είναι ανεξάρτητη από πλατφόρμα και είναι προσαρμοσμένη στη συγκεκριμένη πλατφόρμα που πρόκειται να εκτελέσει μια εικονική μηχανή Java γι 'αυτήν, η οποία μεταφράζει τον Java bytecode στη γλώσσα της μηχανής της πλατφόρμας.

Απόδοση

Τα προγράμματα γραμμένα σε Java έχουν τη φήμη ότι είναι πιο αργά και απαιτούν περισσότερη μνήμη από αυτά που γράφονται στην C ++. Ωστόσο, η ταχύτητα εκτέλεσης των προγραμμάτων Java βελτιώθηκε σημαντικά με την εισαγωγή των Just-in-Time το 1997/1998 για την Java 1.1. , την προσθήκη χαρακτηριστικών γλώσσας που υποστηρίζουν καλύτερη ανάλυση κώδικα (όπως την κλάση StringBuilder, τις εσωτερικές κλάσεις, τις προαιρετικούς ισχυρισμούς κ.λπ.) καθώς και με την βελτιστοποίηση στην εικονική μηχανή Java. Με την Java 1.5, η απόδοση βελτιώθηκε με την προσθήκη του πακέτου java.util.concurrent συμπεριλαμβανομένης των υλοποιήσιμων χωρίς κλειδαριές(lock free) των ConcurrentMaps και άλλων συλλογών πολλαπλών πυρήνων, και βελτιώθηκε περαιτέρω με την Java 1.6.

Αυτόματη διαχείριση μνήμης

Η Java χρησιμοποιεί έναν αυτόματο συλλέκτη σκουπιδιών(garbage collector) για τη διαχείριση μνήμης στον κύκλο ζωής του προγράμματος. Ο προγραμματιστής καθορίζει πότε δημιουργούνται τα αντικείμενα και ο χρόνος

εκτέλεσης Java είναι υπεύθυνος για την ανάκτηση της μνήμης μόλις τα αντικείμενα δεν χρησιμοποιούνται πλέον. Μόλις δεν παραμένουν αναφορές σε ένα αντικείμενο, η μνήμη που κατανέμεται δυναμικά (unreachable memory) καθίσταται επιλέξιμη για να ελευθερωθεί αυτόματα από τον συλλέκτη απορριμμάτων. Καταλαβαίνουμε την σημαντικότητα του συλλέκτη σκουπιδιών αν αναλογιστούμε ότι κάτι παρόμοιο με διαρροή μνήμης μπορεί να συμβεί αν ο κώδικας ενός προγραμματιστή κρατά μια αναφορά σε ένα αντικείμενο που δεν χρειάζεται πλέον, συνήθως όταν αντικείμενα που δεν χρειάζονται πλέον αποθηκεύονται σε θέσεις που είναι ακόμα σε χρήση. Εάν κληθούν μέθοδοι για ανύπαρκτο αντικείμενο, προκύπτει μια "εξαίρεση μηδενικού δείκτη" («null pointer exception»).

Μια από τις ιδέες πίσω από το μοντέλο αυτόματης διαχείρισης μνήμης της Java είναι ότι οι προγραμματιστές μπορούν να απαλλαγούν από την επιβάρυνση που απαιτείται για τη χειρωνακτική διαχείριση της μνήμης. Σε ορισμένες γλώσσες, η μνήμη για τη δημιουργία αντικειμένων κατανέμεται στη στοίβα ή εκχωρείται και καταργείτε από τον σωρό . Στην τελευταία περίπτωση, η ευθύνη της διαχείρισης της μνήμης βρίσκεται στον προγραμματιστή. Εάν το πρόγραμμα δεν απομακρύνει ένα αντικείμενο, εμφανίζεται διαρροή μνήμης . Εάν το πρόγραμμα προσπαθεί να αποκτήσει πρόσβαση ή να απομακρύνει μνήμη που έχει ήδη αφαιρεθεί, το αποτέλεσμα είναι απροσδιόριστο και δύσκολο να προβλεφθεί και το πρόγραμμα είναι πιθανό να γίνει ασταθές ή να καταρρεύσει. Αυτό μπορεί να αντιμετωπιστεί εν μέρει με τη χρήση έξυπνων δεικτών , οι οποίοι όμως προσθέτουν επιβάρυνση και πολυπλοκότητα. Σημειώστε ότι η συλλογή απορριμμάτων δεν εμποδίζει τη διαρροή "λογικής" μνήμης, δηλαδή εκείνες όπου η μνήμη εξακολουθεί να αναφέρεται, αλλά δεν χρησιμοποιείται ποτέ.

Η συλλογή σκουπιδιών μπορεί να συμβεί ανά πάσα στιγμή. Στην ιδανική περίπτωση, θα συμβεί όταν ένα πρόγραμμα είναι αδρανές. Είναι εγγυημένη η ενεργοποίησή του εάν δεν υπάρχει αρκετή ελεύθερη μνήμη στον σωρό για να φιλοξενήσει ένα νέο αντικείμενο. Αυτό μπορεί να προκαλέσει προσωρινή διακοπή ενός προγράμματος. Δεν είναι δυνατή η διαχείριση ρητής μνήμης στην Java.

Η Java δεν υποστηρίζει την αριθμητική δείκτη όπως συμβαίνει στη C / C ++, όπου οι διευθύνσεις αντικειμένων και οι μη υπογεγραμμένοι ακέραιοι (συνήθως μεγάλοι ακέραιοι) μπορούν να χρησιμοποιηθούν εναλλακτικά. Αυτό επιτρέπει στον συλλέκτη απορριμμάτων να μεταφέρει τα αναφερόμενα αντικείμενα και εξασφαλίζει ασφάλεια και ασφάλεια τύπου(type safety).

Η Java περιέχει πολλούς τύπους συλλεκτών σκουπιδιών. Από προεπιλογή, χρησιμοποιείτε ο παράλληλος συλλέκτης απορριμμάτων. Ωστόσο, υπάρχουν επίσης αρκετοί άλλοι συλλέκτες απορριμμάτων που μπορούν να χρησιμοποιηθούν για τη διαχείριση του σωρού.

7. ΒΑΣΙΚΑ ΔΟΜΙΚΑ ΜΕΡΗ ΜΙΑΣ ANDROID ΕΦΑΡΜΟΓΗΣ

Τα στοιχεία της εφαρμογής (application components) είναι τα βασικά δομικά στοιχεία μιας εφαρμογής Android. Κάθε στοιχείο είναι ένα σημείο εισόδου μέσω του οποίου το σύστημα ή ένας χρήστης μπορεί να εισάγει την εφαρμογή σας. Υπάρχουν τέσσερις διαφορετικοί τύποι στοιχείων της εφαρμογής:

- Δραστηριότητες (Activities)
- Υπηρεσίες (Services)
- Δέκτες εκπομπής (Broadcast Receivers)
- Παροχείς περιεχομένου (Content Providers)

Κάθε τύπος εξυπηρετεί έναν ξεχωριστό σκοπό και έχει έναν ξεχωριστό κύκλο ζωής ο οποίος καθορίζει τον τρόπο με τον οποίο το στοιχείο δημιουργείται και καταστρέφεται. Στη συνέχεια ακολουθεί συνοπτική περιγραφή για κάθε έναν τύπο στοιχείου από τους παραπάνω.

7.1 Δραστηριότητες

Μια δραστηριότητα είναι το σημείο εισόδου για την αλληλεπίδραση με τον χρήστη. Αντιπροσωπεύει μια ενιαία οθόνη με μια διεπαφή χρήστη (User Interface - UI). Για παράδειγμα, μια εφαρμογή ηλεκτρονικού ταχυδρομείου μπορεί να έχει μια δραστηριότητα που εμφανίζει μια λίστα με νέα μηνύματα ηλεκτρονικού ταχυδρομείου, μια άλλη δραστηριότητα για τη σύνταξη ενός μηνύματος ηλεκτρονικού ταχυδρομείου και μια άλλη δραστηριότητα για την ανάγνωση μηνυμάτων ηλεκτρονικού ταχυδρομείου. Αν και οι δραστηριότητες συνεργάζονται για να σχηματίσουν μια συνολική εμπειρία χρήστη στην εφαρμογή email, κάθε μία είναι ανεξάρτητη από τις άλλες. Ως εκ τούτου, μια διαφορετική εφαρμογή μπορεί να ξεκινήσει οποιαδήποτε από αυτές τις δραστηριότητες αν το επιτρέπει η εφαρμογή ηλεκτρονικού ταχυδρομείου. Για παράδειγμα, μια εφαρμογή κάμερας μπορεί να ξεκινήσει τη δραστηριότητα στην εφαρμογή ηλεκτρονικού ταχυδρομείου που συνθέτει νέα μηνύματα ηλεκτρονικού ταχυδρομείου για να επιτρέψει στο χρήστη να μοιραστεί μια φωτογραφία.

7.2 Υπηρεσίες

Μια υπηρεσία είναι ένα σημείο εισόδου γενικού σκοπού για τη διατήρηση μιας εφαρμογής που εκτελείται στο παρασκήνιο για κάθε είδους λόγο. Πρόκειται για ένα στοιχείο που εκτελείται στο παρασκήνιο για εκτέλεση εκτεταμένων εργασιών ή για εκτέλεση εργασίας για απομακρυσμένες διεργασίες. Μια υπηρεσία δεν παρέχει διεπαφή χρήστη. Για παράδειγμα, μια υπηρεσία μπορεί να αναπαράγει μουσική στο παρασκήνιο ενώ ο χρήστης βρίσκεται σε διαφορετική εφαρμογή ή μπορεί να ανακτήσει δεδομένα μέσω του δικτύου χωρίς να εμποδίζει την αλληλεπίδραση των χρηστών με μια δραστηριότητα. Ένα άλλο στοιχείο, όπως μια δραστηριότητα, μπορεί να ξεκινήσει την υπηρεσία και να την αφήσει να τρέξει ή να συνδεθεί σε αυτήν για να αλληλεπιδράσει με αυτήν. Υπάρχουν δύο πολύ ξεχωριστές υπηρεσίες σημασιολογίας που λένε στο σύστημα πώς να διαχειριστεί μια εφαρμογή: Οι υπηρεσίες που ξεκίνησαν λένε στο σύστημα να τις κρατήσει σε λειτουργία μέχρι να ολοκληρωθεί η εργασία τους. Αυτό θα μπορούσε να είναι ο συγχρονισμός ορισμένων δεδομένων στο παρασκήνιο ή η αναπαραγωγή μουσικής ακόμα και μετά την έξοδο από την εφαρμογή από τον χρήστη. Ο συγχρονισμός δεδομένων στο παρασκήνιο ή η αναπαραγωγή μουσικής αντιπροσωπεύουν επίσης δύο διαφορετικούς τύπους υπηρεσιών που ξεκινούν και οι οποίες τροποποιούν τον τρόπο χειρισμού του συστήματος.

7.3 Δέκτες εκπομπής

Ένας δέκτης εκπομπής είναι ένα στοιχείο το οποίο επιτρέπει στο σύστημα να εκπέμπει συμβάντα στην εφαρμογή εκτός μιας κανονικής ροής του χρήστη, επιτρέποντας στην εφαρμογή να ανταποκρίνεται σε ανακοινώσεις εκπομπής σε όλο το σύστημα. Επειδή οι δέκτες εκπομπής είναι μια άλλη καλά καθορισμένη είσοδος στην εφαρμογή, το σύστημα μπορεί να παραδώσει εκπομπές ακόμα και σε εφαρμογές που δεν εκτελούνται αυτήν τη στιγμή. Έτσι, για παράδειγμα, μια εφαρμογή μπορεί να προγραμματίσει ένα συναγερμό για να δημοσιεύσει μια ειδοποίηση για να πει στο χρήστη για ένα επερχόμενο συμβάν και με την παράδοση αυτού του συναγερμού σε έναν Broadcast Receiver της εφαρμογής, δεν υπάρχει ανάγκη η εφαρμογή να παραμείνει σε λειτουργία μέχρι ο συναγερμός να σβήσει. Πολλές εκπομπές

προέρχονται από το σύστημα - για παράδειγμα, μια εκπομπή που αναγγέλλει ότι η οθόνη έχει απενεργοποιηθεί, η μπαταρία είναι χαμηλή ή έχει τραβηχτεί μια φωτογραφία. Οι εφαρμογές μπορούν επίσης να εκκινήσουν εκπομπές - για παράδειγμα, για να επιτρέψουν σε άλλες εφαρμογές να γνωρίζουν ότι ορισμένα δεδομένα έχουν ληφθεί στη συσκευή και είναι διαθέσιμα για χρήση από αυτούς. Αν και οι δέκτες εκπομπής δεν εμφανίζουν διεπαφή χρήστη, ενδέχεται να δημιουργήσουν μια ειδοποίηση γραμμής κατάστασης για την ειδοποίηση του χρήστη όταν συμβεί κάποιο γεγονός εκπομπής. Συνηθέστερα, όμως, ένας δέκτης εκπομπής είναι απλώς μια πύλη προς άλλα εξαρτήματα και προορίζεται να κάνει πολύ ελάχιστη ποσότητα εργασίας.

7.4 Παροχείς περιεχομένου

Ένας πάροχος περιεχομένου διαχειρίζεται ένα κοινό σύνολο δεδομένων της εφαρμογής που μπορείτε να αποθηκεύσετε στο σύστημα αρχείων, σε μια βάση δεδομένων SQLite, στον ιστό ή σε οποιαδήποτε άλλη μόνιμη τοποθεσία αποθήκευσης στην οποία μπορεί να έχει πρόσβαση η εφαρμογή σας. Μέσω του παρόχου περιεχομένου, άλλες εφαρμογές μπορούν να ζητήσουν ή να τροποποιήσουν τα δεδομένα, αν το επιτρέπει ο πάροχος περιεχομένου. Για παράδειγμα, το σύστημα Android παρέχει έναν παροχέα περιεχομένου που διαχειρίζεται τα στοιχεία επικοινωνίας του χρήστη. Ως εκ τούτου, οποιαδήποτε εφαρμογή με τα κατάλληλα δικαιώματα μπορεί να ζητήσει από τον πάροχο περιεχομένου, όπως `ContactsContract.Data` , να διαβάσει και να γράψει πληροφορίες σχετικά με ένα συγκεκριμένο άτομο. Είναι δελεαστικό να σκεφτόμαστε έναν πάροχο περιεχομένου ως μια αφαίρεση σε μια βάση δεδομένων, επειδή υπάρχουν πολλά API και υποστήριξη που ενσωματώνονται σε αυτά για αυτή την κοινή υπόθεση. Ωστόσο, έχουν διαφορετικό βασικό σκοπό από την άποψη του σχεδιασμού του συστήματος. Στο σύστημα, ένας πάροχος περιεχομένου είναι ένα σημείο εισόδου σε μια εφαρμογή για τη δημοσίευση ονομάτων δεδομένων που προσδιορίζονται από ένα σχήμα URI. Έτσι, μια εφαρμογή μπορεί να αποφασίσει πώς θέλει να αντιστοιχίσει τα δεδομένα που περιέχει σε ένα χώρο ονομάτων URI, παραδίδοντας τα URI σε άλλες οντότητες, οι οποίες με τη

σειρά τους μπορούν να τις χρησιμοποιήσουν για να έχουν πρόσβαση στα δεδομένα.

Οι παροχείς περιεχομένου είναι επίσης χρήσιμοι για την ανάγνωση και τη σύνταξη δεδομένων που είναι ιδιωτικά στην εφαρμογή σας και δεν μοιράζονται. Για παράδειγμα, η εφαρμογή δειγμάτων Note Pad χρησιμοποιεί έναν παροχέα περιεχομένου για την αποθήκευση σημειώσεων. Ένας πάροχος περιεχομένου υλοποιείται ως υποκατηγορία του ContentProvider και πρέπει να εφαρμόσει ένα πρότυπο σύνολο API που επιτρέπουν σε άλλες εφαρμογές να εκτελούν συναλλαγές. Για περισσότερες πληροφορίες, ανατρέξτε στον οδηγό προγραμματιστών Providers περιεχομένου.

Μια μοναδική πτυχή του σχεδιασμού συστήματος Android είναι ότι κάθε εφαρμογή μπορεί να ξεκινήσει μια συνιστώσα μιας άλλης εφαρμογής. Για παράδειγμα, εάν θέλετε ο χρήστης να τραβήξει μια φωτογραφία με τη φωτογραφική μηχανή της συσκευής, ίσως υπάρχει κάποια άλλη εφαρμογή που το κάνει και η εφαρμογή σας μπορεί να το χρησιμοποιήσει αντί να αναπτύξει μια δραστηριότητα για να τραβήξετε μόνοι σας μια φωτογραφία. Δεν χρειάζεται να ενσωματώσετε ή ακόμα και να συνδέσετε τον κώδικα από την εφαρμογή φωτογραφικής μηχανής. Αντί αυτού, μπορείτε απλά να ξεκινήσετε τη δραστηριότητα στην εφαρμογή φωτογραφικής μηχανής που συλλαμβάνει μια φωτογραφία. Όταν ολοκληρωθεί, η φωτογραφία επιστρέφεται στην εφαρμογή σας, ώστε να μπορείτε να την χρησιμοποιήσετε. Για τον χρήστη, φαίνεται ότι η κάμερα είναι στην πραγματικότητα μέρος της εφαρμογής σας.

Όταν το σύστημα ξεκινά ένα στοιχείο, ξεκινά τη διαδικασία για αυτήν την εφαρμογή αν δεν εκτελείται ήδη και παράγει τις κλάσεις που απαιτούνται για το στοιχείο. Για παράδειγμα, αν η εφαρμογή σας ξεκινήσει τη δραστηριότητα στην εφαρμογή φωτογραφικής μηχανής που συλλαμβάνει μια φωτογραφία, αυτή η δραστηριότητα εκτελείται στη διαδικασία που ανήκει στην εφαρμογή φωτογραφικής μηχανής και όχι στη διαδικασία της εφαρμογής σας. Επομένως, σε αντίθεση με τις εφαρμογές στα περισσότερα άλλα συστήματα, οι εφαρμογές Android δεν έχουν ένα μόνο σημείο εισόδου (δεν υπάρχει λειτουργία `main()`).

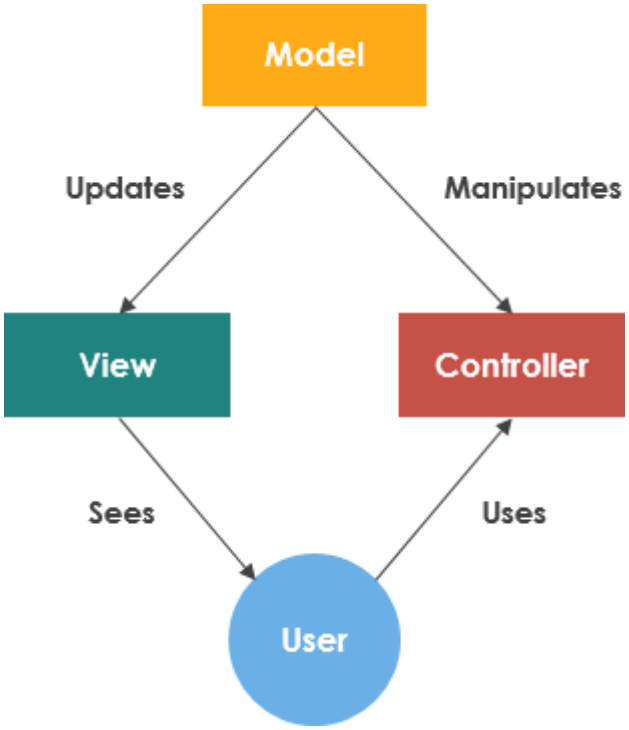
Επειδή το σύστημα εκτελεί κάθε εφαρμογή σε ξεχωριστή διαδικασία με δικαιώματα αρχείων που περιορίζουν την πρόσβαση σε άλλες εφαρμογές, η εφαρμογή σας δεν μπορεί να ενεργοποιήσει άμεσα ένα στοιχείο από άλλη εφαρμογή. Ωστόσο, το σύστημα Android μπορεί. Για να ενεργοποιήσετε ένα στοιχείο σε άλλη εφαρμογή, παραδώστε ένα μήνυμα στο σύστημα που καθορίζει την πρόθεσή σας να ξεκινήσετε ένα συγκεκριμένο στοιχείο. Το σύστημα ενεργοποιεί τότε το στοιχείο για εσάς.

8. ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Στο κεφάλαιο αυτό παρουσιάζεται λεπτομερώς η εφαρμογή που αναπτύχθηκε. Πριν γίνει αυτό γίνεται επεξήγηση ορισμένων σημαντικών στοιχείων στα οποία βασίστηκε η υλοποίηση.

8.1 Αρχές και τεχνικές αντικειμενοστραφή προγραμματισμού

Η ανάπτυξη της εφαρμογής της εργασίας έγινε με τη χρήση της γλώσσας προγραμματισμού Java. Είναι μια αντικειμενοστραφής γλώσσα που είναι ευρέως διαδεδομένη και χρησιμοποιείτε από πολλούς προγραμματιστές τόσο για την ανάπτυξη εφαρμογών στο λογισμικό του Android όσο και για άλλες εφαρμογές. Πρόκειται για μια γλώσσα που υπακούει στις αρχές και τις ιδέες του αντικειμενοστραφή προγραμματισμού (Object Oriented Programming - OOP). Στην παρούσα εφαρμογή χρησιμοποιούνται κάποιες από αυτές τις αρχές. Αρχικά χρησιμοποιείτε η αφαίρεση των δεδομένων (data abstraction). Είναι η ιδιότητα που έχουν οι κλάσεις να αναπαριστούν αφαιρετικά σύνθετες οντότητες σε ένα προγραμματιστικό περιβάλλον. Ουσιαστικά μια κλάση αποτελεί ένα αφαιρετικό μοντέλο κάποιας κατηγορίας αντικειμένων.



Εικόνα 8.1.1 Το μοντέλο MVC

Μια ακόμη ιδέα του αντικειμενοστραφή προγραμματισμού που χρησιμοποιείτε στην εφαρμογή είναι η ενθυλάκωση δεδομένων (data encapsulation). Αυτή είναι η ιδιότητα που προσφέρουν οι κλάσεις να «κρύβουν» τα ιδιωτικά τους δεδομένα από τα υπόλοιπα αντικείμενα του κώδικα. Η προσπέλαση αυτών των δεδομένων πραγματοποιείτε μόνο από τις δημόσιες μεθόδους τους. Επίσης χρησιμοποιείτε και η αρχή της κληρονομικότητας. Πρόκειται για ένα μηχανισμό στον οποίο μια κλάση κληρονομεί, δηλαδή αποκτά τις ιδιότητες μίας άλλης κλάσης ή αντικειμένου. Η κλάση που αντλεί χαρακτηριστικά ονομάζεται κλάση παιδί (child class) ή παράγωγη κλάση (derived class) ενώ εκείνη που παρέχει τις ιδιότητες της ονομάζεται βασική κλάση(base class) ή κλάση γονέας (parent class). Με την κληρονομικότητα παρέχεται η δυνατότητα επαναχρησιμοποίησης των πεδίων και των μεθόδων μιας ήδη υπάρχουσας κλάσης. Πέραν των παραπάνω αρχών και ιδεών η υλοποίηση της εφαρμογής βασίζεται πάνω σε ένα μοντέλο αρχιτεκτονικής λογισμικού που ονομάζεται Model – View – Controller (MVC). Στο μοντέλο αυτό (βλ. εικόνα 8.1.1) η εφαρμογή διαιρείται σε τρία διασυνδεδεμένα μέρη ώστε να διαχωριστεί η παρουσίαση της πληροφορίας (View) στον χρήστη(User) όπως έχει αποθηκευτεί στο σύστημα. Το κύριο μέρος της αρχιτεκτονικής αυτής είναι το μοντέλο (Model) το οποίο διαχειρίζεται την ανάκτηση/αποθήκευση των δεδομένων του συστήματος. Το αντικείμενο View χρησιμοποιείται μόνο για την οπτικοποίηση της πληροφορίας προς τον χρήστη (στην εφαρμογή της εργασίας είναι ένα User Interface). Το τρίτο μέρος , ο Controller , δέχεται την είσοδο και αλληλεπιδρά με το αντικείμενο Model και το View.

8.2 Το μοντέλο της εφαρμογής

Έχοντας δει μια σύντομη παρουσίαση της αρχιτεκτονικής του MVC στην προηγούμενη υποενότητα σε αυτή θα γίνει επεξήγηση του μοντέλου της εφαρμογής που υλοποιήθηκε. Αποτελείτε από τις κλάσεις Card, DataProvider, ItemsAdapter. Ουσιαστικά πρόκειται για τον κορμό της εφαρμογής γύρω από τον οποίο αλληλεπιδρούν τα αρχεία κώδικα. Αρχικά υπάρχει η κλάση Card η οποία αντιπροσωπεύει ένα αντικείμενο κάρτας για το παιχνίδι. Προκειμένου να προσομοιωθεί απαιτείται να υπάρχουν μεταβλητές που θα διατηρούν πληροφορίες σχετικά με την θέση των καρτών στην οθόνη (οι κάρτες

συγκεκριμένα περιέχονται σε ένα πλέγμα) καθώς και τις εικόνες που θα φέρουν επάνω τους. Στην συνέχεια υπάρχει η κλάση DataProvider η οποία περιέχει τις πληροφορίες και τις διαδικασίες που χρειάζονται στην εφαρμογή. Πιο συγκεκριμένα είναι υπεύθυνη ώστε να διατηρεί ακέραιες τιμές των εικόνων που θα πάρουν οι κάρτες καθώς επίσης και διάφορες μεθόδους οι οποίες εκτελούν βασικές διαδικασίες του παιχνιδιού όπως η αρχικοποίηση των καρτών(δηλαδή η ανάθεση συγκεκριμένης κατηγορίας στις κάρτες καθώς και ο ορισμός του πλήθους τους). Τέλος, το μοντέλο ολοκληρώνεται με την κλάση ItemsAdapter που είναι υπεύθυνη για την διάταξη των καρτών στην οθόνη (και συγκεκριμένα στο πλέγμα). Στο κεφάλαιο 8.4 υπάρχει και ανάλυση του κώδικα του μοντέλου.

8.3 Παρουσίαση εφαρμογής «MemorizeIt»

Η εφαρμογή που υλοποιήθηκε σχετίζεται με την αντιμετώπιση της νόσου Αλτσχάιμερ. Πιο συγκεκριμένα πρόκειται για ένα διαδραστικό παιχνίδι το οποίο έχει στόχο να εξασκήσει την μνήμη του χρήστη, κάτι το οποίο αποτελεί ένα βασικό κομμάτι ενάντιας στη νόσο.

Κατά την εκκίνηση της εφαρμογής εμφανίζεται η αρχική οθόνη η οποία περιλαμβάνει το λογότυπο της εφαρμογής και μετά το πέρας τριών δευτερόλεπτων εμφανίζεται η επόμενη οθόνη.

Εικόνα 8.3.1: Το λογότυπο της εφαρμογής κατά το άνοιγμα της

Στην δεύτερη οθόνη εμφανίζεται το μενού του παιχνιδιού στο οποίο μπορεί να κάνει ο χρήστης τις επιλογές του ώστε να διαμορφώσει τις ρυθμίσεις του παιχνιδιού. Πιο συγκεκριμένα, ο χρήστης έχει τη δυνατότητα να επιλέξει το είδος των καρτών που θα παίξει, τα οποία είναι τρία. Αυτά είναι εικόνες ζώων, εικόνες σημαιών χωρών και εικόνες φαγητών. Επιπλέον ο χρήστης επιλέγει το πλήθος των καρτών που θα παίξει.

Εικόνα 8.3.2: Το μενού του παιχνιδιού

Μπορεί να επιλέξει μεταξύ δέκα(10), δώδεκα(12) και δεκαέξι(16) κάρτες. Ο χρήστης έχει την δυνατότητα να αλλάξει οποιαδήποτε στιγμή τις επιλογές του όσο βρίσκεται σε αυτή την οθόνη.

Εικόνα 8.3.3: Το μενού του παιχνιδιού μετά από επιλογές

Όταν συμβεί αυτό ταυτόχρονα γίνεται και επαναφορά των χρωμάτων των προηγούμενων επιλογών(δηλαδή από-επιλέγονται) που είχε κάνει ο χρήστης ή που ήταν ήδη προεπιλεγμένες.

Εικόνα 8.3.4: Το μενού του παιχνιδιού με άλλες επιλογές

Όταν πλέον ο χρήστης καταλήξει στις επιλογές του, πατάει το κουμπί «ΠΑΙΞΕ» προκειμένου να ξεκινήσει το παιχνίδι. Όταν συμβεί αυτό εμφανίζεται η βασική οθόνη του παιχνιδιού.

Εικόνα 8.3.3: Αρχική κατάσταση παιχνιδιού

Σε αυτήν εμφανίζονται όλες οι κάρτες «αναποδογυρισμένες» έτσι ώστε να μην ξέρει ο χρήστης τι έχει η κάθε μια. Ο χρήστης παίζει το παιχνίδι γυρνώντας τις κάρτες προσπαθώντας να τις ταιριάξει χρησιμοποιώντας την μνήμη του. Πατώντας πάνω σε μία από αυτές εκείνη περιστρέφεται και εμφανίζεται η εικόνα που έχει από κάτω της.

Εικόνα 8.3.6: Κατάσταση παιχνιδιού μετά από μια αναποδογυρισμένη κάρτα

Στη συνέχεια ο χρήστης είτε μπορεί να ξανά επιλέξει την ίδια κάρτα σε περίπτωση που θέλει να αναιρέσει την επιλογή που έκανα είτε προχωράει στην επόμενη επιλογή του. Όταν επιλέξει την δεύτερη κάρτα εκείνη αναποδογυρίζεται και εμφανίζει την εικόνα της. Αν οι δυο κάρτες είναι ίδιες τότε οι εικόνες παραμένουν ανοιχτές

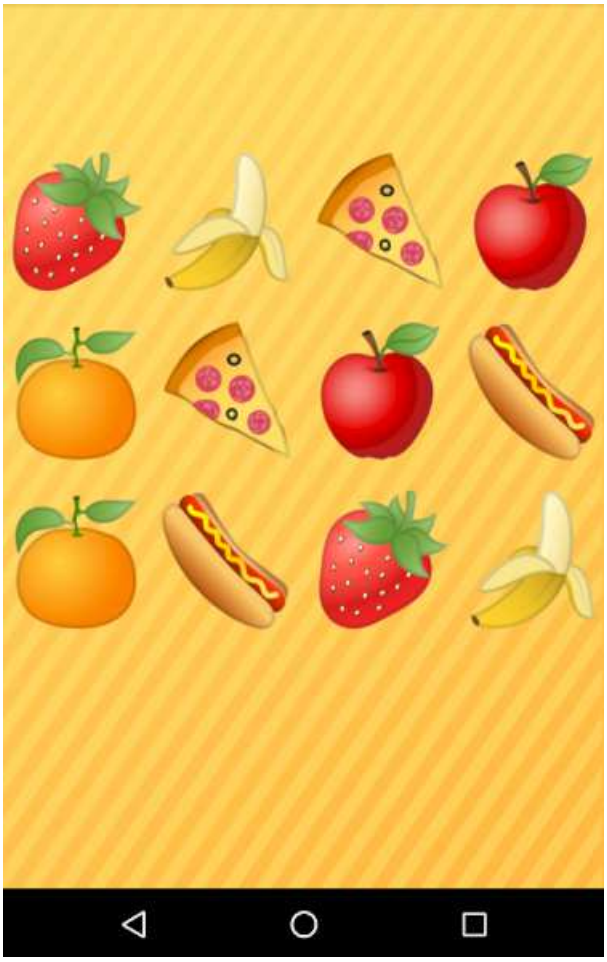
Εικόνα 8.3.7: Κατάσταση παιχνιδιού με δυο διαφορετικές κάρτες

Εάν οι κάρτες δεν είναι ίδιες τότε ξανά επιστρέφουν στην αρχική τους – κρυφή – κατάσταση.

Εικόνα 8.3.8: Κατάσταση παιχνιδιού με δυο όμοιες κάρτες

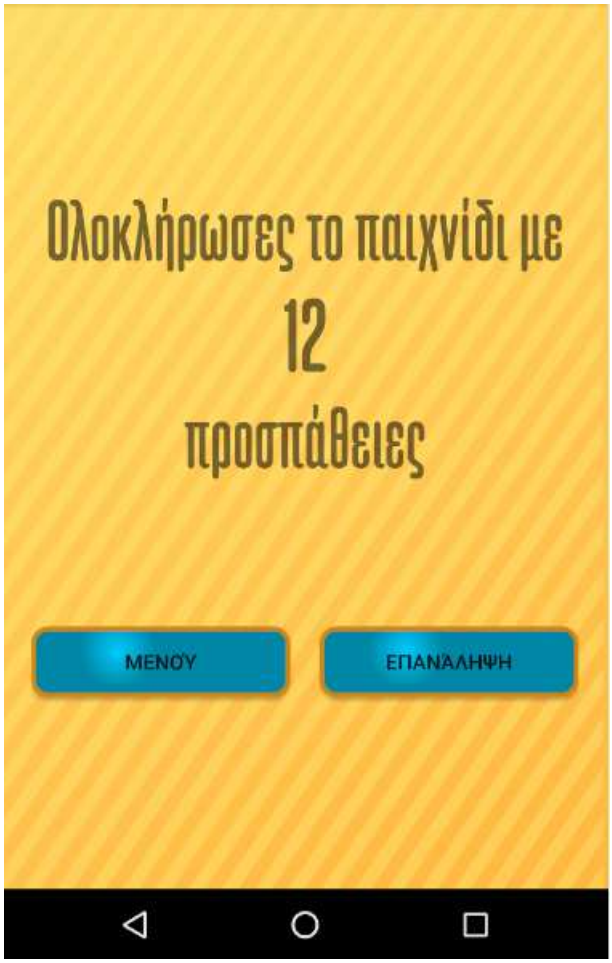
Το παιχνίδι τελειώνει όταν ο χρήστης καταφέρει να ταιριάξει όλες τις κάρτες δηλαδή όταν εμφανίσει όλα τα ζευγάρια.

Εικόνα 8.3.9: Κατάσταση παιχνιδιού με δυο ζευγάρια όμοιων καρτών και μια επιλογή



Εικόνα 8.3.10: Κατάσταση παιχνιδιού όταν έχουν ολοκληρωθεί όλα τα ταιριάσματα

Το παιχνίδι ολοκληρώνεται με την τελευταία οθόνη όπου εμφανίζεται στο χρήστη ο αριθμός των προσπαθειών που χρειάστηκε να κάνει προκειμένου να ταιριάξει όλες τις κάρτες.



Εικόνα 8.3.11: Ενημέρωση του χρήστη για τον αριθμό των προσπαθειών του

Στην ίδια οθόνη υπάρχουν επίσης δύο κουμπιά. Με το ένα (Μενού) ο χρήστης πλοηγείτε στην οθόνη όπου κάνει τις επιλογές του παιχνιδιού που θα παίξει (βλ. εικόνα 8.3.2). Με το άλλο (Επανάληψη) ξανά παίζει το παιχνίδι με τις ίδιες ακριβώς επιλογές που είχε κάνει στο προηγούμενο που μόλις τελείωσε.

8.4 Ανάλυση τμημάτων πηγαίου κώδικα

Στην προηγούμενη υποενότητα έγινε η παρουσίαση της εφαρμογής όπως αυτή εκτελείτε σε μια συσκευή και πώς ο χρήστης αλληλεπιδρά μαζί της. Σε αυτή την ενότητα θα γίνει παρουσίαση και επεξήγηση ορισμένων τμημάτων κώδικα από αυτόν που γράφτηκε κατά την υλοποίηση της εφαρμογής. Αρχικά θα γίνει ανάλυση του κώδικα των κλάσεων του μοντέλου που περιγράφτηκε στην ενότητα 8.2 και στη συνέχεια ο κώδικας ορισμένων activities τα οποία αντιστοιχούν στις οθόνες της εφαρμογής που περιγράφηκαν στην ακριβώς προηγούμενη ενότητα. Μαζί με αυτά τα activities θα συνοδεύονται και τα

αντίστοιχα layouts αυτών τα οποία είναι υπεύθυνα για την χωρική ταξινόμιση των επιμέρους στοιχείων(ImageView, TextView, Button κλπ) στην οθόνη.

Κλάσεις μοντέλου

Στην ενότητα 8.2 έγινε αναφορά στις κλάσεις που αποτελούν το μοντέλο της εφαρμογής. Αυτές είναι οι Card, DataProvider, ItemsAdapter. Ακολουθεί ο κώδικας τους:

Πίνακας 8.4.1 Κώδικας Μοντέλου Μέρος Α (κλάσης Card)

```
public class Card
{
    private int[] imagesArray;
    private Integer[] positions;

    public Card()
    {

    }

    public void setImagesArray(int[] array)
    {
        imagesArray = array;
    }

    public int[] getImagesArray()
    {
        return imagesArray;
    }

    public void setPositions(Integer[] pos)
    {
        positions = pos;
    }

    public Integer[] getPositions()
    {
        return positions;
    }
}
```

Η κλάση Card περιέχει δύο βασικές παραμέτρους, τον πίνακα ακεραίων imagesArray και τον πίνακα ακεραίων positions. Αυτές οι μεταβλητές είναι ιδιωτικές (private). Για αυτό το λόγο απαιτείτε η χρήση μεθόδων με τις οποίες θα μπορούν να διαπεραστούν οι τιμές τους. Οι μέθοδοι αυτές είναι οι setImagesArray, getImagesArray,setPositions και getPositions. Οι

setImagesArray και setPositions παίρνουν σαν όρισμα τους ένα πίνακα ακεραίων και αναθέτουν τις τιμές του στους πίνακες imagesArray και positions αντίστοιχα. Οι getImagesArray, getPositions επιστρέφουν τις τιμές των πινάκων imagesArray και positions αντίστοιχα. Δηλώνοντας αυτούς τους πίνακες ως ιδιωτικούς και διαχειρίζοντας τα περιεχόμενα τους μόνο μέσω αυτών των μεθόδων υλοποιείται η τεχνική της ενθυλάκωσης (encapsulation - βλ. ενότητα 8.1)

Πίνακας 8.4.2 Κώδικας μοντέλου Μέρος Β (κλάση DataProvider)

```
public class DataProvider
{
    final int[] drawableArray = new int[]{R.drawable.bee,
R.drawable.cow,R.drawable.crab, R.drawable.elephant,
R.drawable.lion,R.drawable.owl,R.drawable.sheep,R.drawable.snake};

    final int[] drawableArrayFlags = new
int[]{R.drawable.france,R.drawable.germany,R.drawable.greece,R.drawab
le.netherland,R.drawable.portugal,
R.drawable.spain,R.drawable.sweden, R.drawable.united_kindom};

    final int[] drawableArrayFoods = new
int[]{R.drawable.strawberry,R.drawable.orange,R.drawable.apple,R.draw
able.banana,R.drawable.pizza, R.drawable.hotdog, R.drawable.burger,
R.drawable.fries};

    private Card current_cards;

    public Card InitializeCards(String kind, int amount)
    {
        current_cards = new Card();

        current_cards.setImagesArray(PickCardsCategory(kind));
        current_cards.setPositions(PickAmountOfCards(amount));

        return current_cards;
    }

    private int[] PickCardsCategory(String kind_of_cards)
    {
        if(kind_of_cards.equals("animals"))
            return drawableArray;
        if(kind_of_cards.equals("flags"))
            return drawableArrayFlags;
        if(kind_of_cards.equals("foods"))
            return drawableArrayFoods;

        return drawableArray;
    }

    private Integer[] PickAmountOfCards(int cards_amount)
    {
```

```
        Integer[] gridPositions = null;

        if(cards_amount == 10)
            gridPositions = new Integer[]{0,1,2,3,4,0,1,2,3,4};
        if(cards_amount == 12)
            gridPositions = new Integer[]{0,1,2,3,4,5,0,1,2,3,4,5};
        if(cards_amount == 16)
            gridPositions = new
Integer[]{0,1,2,3,4,5,6,7,0,1,2,3,4,5,6,7};

        return gridPositions;
    }
}
```

Η κλάση DataProvider περιέχει τρείς πίνακες ακέραιων, οι οποίοι είναι οι drawableArray, drawableArrayFlags και drawableArrayFoods. Κάθε ένας από αυτούς περιέχει τις ακέραιες τιμές που αντιστοιχούν σε αρχεία εικόνων(.png extension). Αυτές οι εικόνες βρίσκονται στη διαδρομή app\src\main\res\drawable του project. Αυτές είναι οι εικόνες οι οποίες θα εμφανίζονται στον χρήστη κατά την εκτέλεση του παιχνιδιού και με τις οποίες έρχεται σε αλληλεπίδραση. Κάθε ένας πίνακας περιέχει διαφορετικής κατηγορίας εικόνες. Ο drawableArray περιέχει ζώα, ο drawableArrayFlags σημαίες χωρών και ο drawableArrayFoods φαγητά. Εκτός αυτών περιέχει το αντικείμενο current_cards που είναι τύπου Card. Επίσης περιέχει και κάποιες μεθόδους. Την InitializeCards η οποία παίρνει σαν παράμετρο ένα αλφαριθμητικό (string) που ορίζει το είδος των καρτών του παιχνιδιού(μεταβλητή kind) και έναν ακέραιο που ορίζει το πλήθος των καρτών (παράμετρος amount). Ο ρόλος της μεθόδου αυτής είναι να αρχικοποιήσει και να επιστρέψει το αντικείμενο current_cards. Για να το κάνει αυτό χρησιμοποιεί άλλες δύο μεθόδους. Η πρώτη είναι η PickCardsCategory η οποία παίρνει σαν παράμετρο ένα αλφαριθμητικό που ορίζει την κατηγορία των καρτών(συγκεκριμένα των εικόνων που θα έχουν οι κάρτες) και επιστρέφει έναν απο τους τρείς πίνακες της. Η δεύτερη είναι PickAmountOfCards η οποία παίρνει σαν παράμετρο έναν ακέραιο και επιστρέφει έναν πίνακα ακεραίων.

Πίνακας 8.4.3 Κώδικας Μοντέλου Μέρος Γ(κλάση ItemsAdapter)

```
public class ItemsAdapter extends BaseAdapter
{
```

```
    private final Context mContext;
    private Card obj_cards;

    public ItemsAdapter(Context context, Card oCards)
    {
        this.mContext = context;
        this.obj_cards = oCards;
    }

    public int getCount()
    {
        return obj_cards.getPositions().length;
    }

    public long getItemId(int position)
    {
        return 0;
    }

    public Object getItem(int position)
    {
        return null;
    }

    public View getView(int position, View convertView, ViewGroup
parent)
    {
        ImageView anImageView;

        if(convertView == null)
        {
            anImageView = new ImageView(this.mContext);
            anImageView.setLayoutParams(new
GridView.LayoutParams(350,350));
            anImageView.setScaleType(ImageView.ScaleType.FIT_XY);
        }
        else
        {
            anImageView = (ImageView)convertView;
        }
        anImageView.setImageResource(R.drawable.hidden);

        return anImageView;
    }
}
```

Η κλάση ItemsAdapter κάνει χρήση της αρχής της κληρονομικότητας (βλ. ενότητα 8.1).Συγκεκριμένα κληρονομεί τις ιδιότητες της αφηρημένης κλάσης BaseAdapter. Αρχικά η ItemsAdapter έχει δυο ιδιωτικές μεταβλητές, τη mContext που είναι τύπου Context και έχει το περιεχόμενο της κλάσης από την οποία καλείται και την obj_cards που είναι τύπου Card. Προχωρώντας στις μεθόδους της αρχικά υπάρχει μια μέθοδος δημιουργός οι οποία αρχικοποιεί τις δυο μεταβλητές. Στην συνέχεια τέσσερεις μέθοδοι οι οποίες

προέρχονται από την κλάση γονέα και οι οποίες υποσκελίζονται(overriding).Η getItemId και η getItem απλώς υλοποιούνται επειδή είναι υποχρεωτικό και δεν αλλάζει κάτι σε αυτές. Η getCount επιστρέφει το πλήθος των καρτών που έχει το obj_cards (μέσω της μεθόδου getPositions) ενώ η getView διαμορφώνει την εικόνα που θα έχει αρχικά κάθε κάρτα(είναι μια εικόνα με ένα ερωτηματικό ώστε να φαίνεται σαν κρυφή).

Αρχείο Activity και Layout της βασικής οθόνης

Η ιδέα γύρω από τον προγραμματισμό της εφαρμογής βασίζεται πάνω στα activities(βλ. κεφ. 7.1). Ουσιαστικά κάθε διαφορετική οθόνη που βλέπει ο χρήστης αποτελεί και ένα ξεχωριστό activity. Όταν μιλάμε για ένα activity αναφερόμαστε σε ένα αρχείο κώδικα java (.java extension) και συγκεκριμένα μια κλάσης με το όνομα του activity η οποία κληρονομεί απο το interface AppCompatActivity. Σε κάθε τέτοιο αρχείο περιέχεται η λογική των στοιχείων που περιέχονται στην οθόνη που αντιπροσωπεύουν. Πιο συγκεκριμένα στον κώδικα των κλάσεων αυτόν καθορίζεται το πως λειτουργούν και το πως συμπεριφέρονται τα αντικείμενα που βλέπει και αλληλεπιδρά ο χρήστης. Για παράδειγμα τι θα συμβεί όταν ο χρήστης πατήσει ένα κουμπί της επιλογής του που βλέπει στην οθόνη ή πως θα γίνει η μετάβαση από μια τρέχουσα οθόνη σε μια άλλη. Κάθε τέτοιο activity συνδέεται άμεσα με ένα άλλο αρχείο που περιέχει κώδικα xml (.xml extension). Σε αυτά τα αρχείο καθορίζετε η δομή και η εμφάνιση των στοιχείων μιας οθόνης. Πιο συγκεκριμένα καθορίζεται η διάταξη τους καθώς και η οπτική τους απεικόνιση ως προς το σχήμα, το χρώμα και γενικότερα την εμφάνιση. Επομένως όπως γίνεται κατανοητό για να αναπαρασταθεί πλήρως μία οθόνη(ως προς την εμφάνιση και τη λειτουργικότητα της) απαιτούνται δύο αρχεία κώδικα τα οποία αλληλεπιδρούν μεταξύ τους. Παρακάτω ακολουθεί ο κώδικας του MainActivity και μια σύντομη εξήγηση του

Πίνακας 8.4.4 Κώδικας της Main Activity

```
public class MainActivity extends AppCompatActivity
{
    Card cards_object;
    boolean locker = false;
    int attempts_counter = 0;
```

```
    String kind_of_cards;
    int amount_of_cards;

    ImageView current_view = null;
    ImageView second_current_view = null;
    private int countPair=0;

    private  int current_position = -1;
    List<Integer> myPositionsList;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Intent intent = getIntent();
        String recived_json = intent.getExtras().getString("json");

        try
        {
            JSONObject parentObject = new JSONObject(recived_json);
            JSONArray parentArray =
parentObject.getJSONArray("settings");

            JSONObject finalObject = parentArray.getJSONObject(0);
            kind_of_cards = finalObject.getString("kind");
            amount_of_cards = finalObject.getInt("amount");
        }
        catch(JSONException e)
        {
            System.out.println("error message: "+e.getMessage());
        }

        DataProvider data = new DataProvider();
        cards_object = data.InitializeCards(kind_of_cards,
amount_of_cards);

        myPositionsList = new
ArrayList<Integer>(Arrays.asList(cards_object.getPositions()));
        Collections.shuffle(myPositionsList);

        GridView gridView = (GridView)findViewById(R.id.gridview);
        ItemsAdapter cardsAdapter = new ItemsAdapter(this,
cards_object );
        gridView.setAdapter(cardsAdapter);

        int[] current_images = cards_object.getImagesArray();

        gridView.setOnItemClickListener(new
AdapterView.OnItemClickListener()
        {
            @Override
            public void onItemClick(AdapterView<?> parent, View view,
final int position, long id)
            {
                if(locker) return;

                if(current_position<0)
```

```
                {
                    current_position = position;
                    current_view = (ImageView)view;


((ImageView)view).setImageResource(cards_object.getImagesArray()[myPo
sitionsList.get(position)]);

                }
                else
                {
                    if(locker == false)
                    {
                        second_current_view = (ImageView)view;
                        locker = true;

                        if(current_position == position)
                        {

second_current_view.setImageResource(R.drawable.hidden);

                            current_position = -1;
                            locker = false;
                        }
                        else
if(myPositionsList.get(current_position)!=
myPositionsList.get(position))
                        {

second_current_view.setImageResource(cards_object.getImagesArray()[my
PositionsList.get(position)]);

                            new Handler().postDelayed(new Runnable()
                            {
                                @Override
                                public void run() {


current_view.setImageResource(R.drawable.hidden);

second_current_view.setImageResource(R.drawable.hidden);

                                    current_position = -1;
                                    locker = false;

                                    attempts_counter++;
                                }
                            }, 1000);
                        }
                        else
                        {

second_current_view.setImageResource(cards_object.getImagesArray()[my
PositionsList.get(position)]);
                            countPair++;

                            current_position = -1;
                            locker = false;
```

```
                                attempts_counter++;

                                if(countPair == amount_of_cards/2)
                                {

                                    JSONObject mainObject = new
JSONObject();
                                    JSONArray settings = new JSONArray();
                                    JSONObject finalObject = new
JSONObject();

                                    try
                                    {

mainObject.put("kind",kind_of_cards);

mainObject.put("amount",amount_of_cards);

mainObject.put("attempts",attempts_counter);
                                    settings.put(mainObject);

finalObject.put("settings",settings);

                                    String json =
finalObject.toString();

                                    Intent i = new
Intent(MainActivity.this, FinalActivity.class);
                                    i.putExtra("json", json);
                                    startActivity(i);
                                }
                                catch(JSONException e)
                                {
                                    System.out.println("error
message: "+e.getMessage());
                                }
                            }
                        }
                    }

                }
            }
        });
    }
}
```

Είναι το βασικό activity του παιχνιδιού μιας και περιέχει την βασική του λειτουργία και ουσιαστικά αποτελεί την βασική οθόνη. Σε αυτήν δηλαδή ο χρήστης παίζει το παιχνίδι γυρνώντας τις κάρτες προσπαθώντας να τις ταιριάξει χρησιμοποιώντας την μνήμη του. Περιλαμβάνει σαν μεταβλητές την cards_object η οποία είναι τύπου Card, την locker η οποία είναι τύπου

boolean , την attempts_counter που είναι ακέραια, το αλφαριθμητικό kind_of_cards, τον ακέραιο amount_of_cards, τις current_view και second_current_view που είναι τύπου ImageView,τον ακέραιο countPair , τον ακέραιο current_position, μια λίστα myPositionsList που περιέχει ακεραίους. Πέραν αυτών περιέχει μια βασική μέθοδο. Αυτή ειναι η onCreate , η οποία βρίσκεται σε κάθε activity και ο κώδικας της εκτελείτε κατά την δημιουργία του activity που την περιέχει. Μέσα σε αυτή υπάρχει όλος ο κώδικας ο οποίος καθορίζει την λειτουργία του παιχνιδιού. Το πρώτο πράγμα που συμβαίνει είναι να λαμβάνετε από το προηγούμενο activity (συγκεκριμένα το MenuActivity) οι επιλογές που έχει κάνει ο χρήστης ως προς την κατηγορία των καρτών που θέλει να παίξει (ζώα, σημαίες, φαγητά) καθώς επίσης και των συνολικό αριθμό των καρτών που θα έχει το παιχνίδι. Αυτές οι δύο πληροφορίες έρχονται σε ένα αλφαριθμητικό το οποίο αποθηκεύεται στην τοπική μεταβλητή recived_json. Πραγματοποιείτε επεξεργασία σε αυτήν έτσι ώστε να διαχωριστούν αυτές οι πληροφορίες και στη συνέχεια να αποθηκευτούν στις μεταβλητές kind_of_cards και amount_of_cards αντίστοιχα. Συνεχίζοντας, γίνεται η αρχικοποίηση του cards_object (τύπου Card) με την χρήση της κλάσης DataProvider, καθώς και της λίστας myPositionsList χρησιμοποιώντας την μέθοδο getPositions του αντικειμένου cards_object. Προχωρώντας στο κώδικα δημιουργείτε μια τυχαία αναδιοργάνωση των περιεχομένων της λίστας myPositionsList ώστε να υπάρχει τυχαιότητα στην πορεία του προγράμματος ως προς την τοποθέτηση των εικόνων που θα έχουν οι κάρτες στο παιχνίδι. Επίσης χρησιμοποιείτε η μεταβλητή gridView η οποία είναι ένα GridView(μια δομή πλέγματος) στην οποία θα τοποθετηθούν οι κάρτες του παιχνιδιού. Στο τέλος της κλάσης υπάρχει η μέθοδος ακροατής(listener) αυτού του πλέγματος. Μέσα σε αυτή την μέθοδο φιλοξενείτε ο κώδικας ο οποίος εκτελείτε όταν ο χρήστης πατήσει κάποια από τις κάρτες που υπάρχουν διατεταγμένες στο πλέγμα. Αυτός ο κώδικας είναι υπεύθυνος για το πως θα εμφανίζονται οι κάρτες (αν θα παραμένουν ανοιχτές σε περίπτωση ταιριάσματος ή αν θα γυρνάνε στην αρχική τους κατάσταση). Πιο συγκεκριμένα ελέγχεται αρχικά ελέγχεται αν η μεταβλητή locker είναι αληθής. Αν συμβαίνει αυτό τότε δεν θα

πραγματοποιηθεί κάποια ενέργεια. Συγκεκριμένα η μεταβλητή αυτή χρησιμοποιείτε προκειμένου να διασφαλίσει ότι μετά την δεύτερη επιλογή που θα κάνει δεν θα του επιτραπεί άλλη επιλογή μέχρι να καθοριστεί αν οι κάρτες που επέλεξε ταιριάζουν ή όχι. Εφόσον δεν ισχύει αυτό ελέγχεται η μεταβλητή current_position αν είναι μικρότερη του μηδενός. Εάν αυτό συμβαίνει τότε σημαίνει ότι δεν έχει αρχικοποιηθεί ποτέ (δηλαδή ο χρήστης δεν έχει πατήσει ποτέ σε κάποια κάρτα). Επομένως ο κώδικας συνεχίζει και η μεταβλητή παίρνει την τιμή της από την μεταβλητή position η οποία διατηρεί τη θέση της κάρτας που επέλεξε ο χρήστης πάνω στο πλέγμα των εικόνων (grid view). Επίσης αρχικοποιείται η current_view (η μεταβλητή η οποία κρατάει την πρώτη επιλογή του χρήστη) με την τιμή της view δηλαδή της εικόνας που επέλεξε ο χρήστης. Εάν η current_position είναι ίση ή μεγαλύτερη από μηδέν, ελέγχεται η μεταβλητή locker για το αν είναι false. Εάν αυτό συμβαίνει τότε αρχικοποιείτε η second_current_view η οποία κρατάει την δεύτερη επιλογή του χρήστη και επίσης η μεταβλητή locker γίνεται αληθής. Στο ίδιο μπλοκ κώδικα ελέγχεται αν η current_position είναι ίδια με την position. Αυτό σημαίνει ότι ο χρήστης σαν δεύτερη επιλογή επέλεξε πάλι τη ίδια κάρτα επομένως η κάρτα αναποδογυρίζεται πάλι (δηλαδή κρύβετε). Εάν αυτό δεν συμβαίνει σημαίνει ότι η δεύτερη επιλογή του χρήστη είναι διαφορετική από την πρώτη επομένως θα πρέπει να γίνει έλεγχος για το αν οι δυο κάρτες είναι ίδιες. Αν δεν είναι ίδιες τότε αναποδογυρίζονται (δηλαδή οι θέσεις των δυο καρτών παίρνουν την κρυφή εικόνα), η current_position γίνεται -1,η locker γίνεται ψευδής και η attempt_counter (μετράει τις προσπάθειες του χρήστη) αυξάνετε κατά ένα. Αυτές οι ενέργειες εκτελούνται με καθυστέρηση ενός δευτερολέπτου προκειμένου η επαναφορά των καρτών στην αρχική κρυφή τους κατάσταση να γίνει ορατή στο μάτι του χρήστη. Σε περίπτωση που οι κάρτες είναι ίδιες αυξάνεται ο μετρητής countPair (μετράει τον αριθμό των ζευγαριών που έχουν βρεθεί) το current_position γίνεται -1 και η locker ψευδής. Επίσης κάθε φορά που δυο κάρτες ταυτίζονται γίνεται έλεγχος για το αν έχουν βρεθεί όλες. Αν αυτό συμβεί καλείτε η FinalActivity η οποία είναι η οθόνη που θα παρουσιάσει τα αποτελέσματα των προσπαθειών του χρήστη. Προκειμένου να γίνει αυτό στο activity αυτό περνιούνται σαν παράμετροι οι

μεταβλητές kind_of_cards, amount_of_cards, attempts_counter όπου έχουν τον τύπο των καρτών, το πλήθος των καρτών καθώς και το σύνολο των προσπαθειών που χρειάστηκε ο χρήστης προκειμένου να ταιριάξει όλες τις κάρτες. Έχοντας δει τον κώδικα της Main Activity θα γίνει και μια παρουσιάσει του layout (πως τοποθετούνται τα components του activity στην οθόνη)

Πίνακας 8.4.5 Κώδικας του layout της Main Activity

```
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/menu_backround_1"
    tools:context=".MainActivity">

    <GridView
        android:id="@+id/gridview"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:columnWidth="70dp"
        android:numColumns="4"
        android:verticalSpacing="15dp"
        android:horizontalSpacing="5dp"
        android:stretchMode="columnWidth"
        android:layout_marginTop="100dp"
        />
</FrameLayout>
```

Όπως φαίνεται στον πίνακα 8.4.5 το activity αυτό ουσιαστικά περιλαμβάνει ένα Grid View το οποίο φιλοξενεί τις εικόνες. Σε αυτόν τον κώδικα ορίζεται αρχικά ένα αντιπροσωπευτικό όνομα (id). Στην συνέχεια η τοποθεσία του πλέγματος εντός της οθόνης. Συγκεκριμένα το μήκος και το πλάτος του παίρνει τις διαστάσεις της οθόνης (match_parent). Ορίζεται ο αριθμός των στηλών που θα έχει (numColumns) καθώς επίσης και οι αποστάσεις που θα έχει η κάθε θέση του πλέγματος στον κάθετο και οριζόντιο άξονα (verticalSpacing και horizontalSpacing). Επίσης ορίζεται η απόσταση από οτιδήποτε υπάρχει πάνω από το πλέγμα (layout_marginTop). Εκτός από τα δεδομένα σχετικά με το grid view στον παραπάνω κώδικα περιλαμβάνονται και διάφορες άλλες πληροφορίες σχετικά με το activity. Ορίζεται και εδώ οι

διαστάσεις που θα πάρει (layout_width και layout_height) καθώς επίσης και η εικόνα που θα έχει η οθόνη στο background.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] https://www.nstr.gr (Προσπελάστηκε 11/2019)

[2] https://neyrologos.gr/alzheimer-symptomata-stadia-aitia-therapeia/ (Προσπελάστηκε 11/2019)

[3] https://neyrologos.gr/anoia-antimetopisi-frontida (Προσπελάστηκε 11/2019)

[4] https://virus.com.gr/askisi-quot-kleidi-quot-gia-ti-veltiosi-ton-asthenon-me-altschaimer/ (Προσπελάστηκε 11/2019)

[5] https://www.in.gr/2009/07/06/health/health-news/askiseis-noitikis-endynamwsis/ (Προσπελάστηκε 11/2019)

[6] https://whatis.techtarget.com/definition/mobile-app (Προσπελάστηκε 11/2019)

[7] https://www.sciencedirect.com/topics/computer-science/mobile-application (Προσπελάστηκε 11/2019)

[8] https://codeburst.io/introduction-to-mobile-applications-ddf9732b690a (Προσπελάστηκε 12/2019)

[9] https://developer.android.com/studio/intro (Προσπελάστηκε 12/2019)

[10] https://en.wikipedia.org/wiki/Android_(operating_system) (Προσπελάστηκε 12/2019)

[11] https://www.lifewire.com/what-is-a-mobile-application-2373354(Προσπελάστηκε 2/2020)

[12] https://howtodoinjava.com/android/android-tutorial-android-project-structure-files-and-resources/(Προσπελάστηκε 2/2020)

[13] https://techblog.gr/software/android-vs-apple-vs-windows-phone/(Προσπελάστηκε 2/2020)

[14] http://android4beginners.com/2013/06/lesson-0-8-additional-information-about-project-structure-in-android-studio-java-and-resources-folders/(Προσπελάστηκε 2/2020)

[15] https://stackoverflow.com(Προσπελάστηκε 4/2020)