

**ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ FLEX – BISON**

ΑΚΑΔΗΜΑΪΚΟ ΕΤΟΣ 2022-2023

ΣΥΝΤΑΚΤΕΣ :

|  |  |  |  |
| --- | --- | --- | --- |
| Ονοματεπώνυμο | ΑΜ | ΕΤΟΣ | EMAIL |
| ΜΙΡΑ ΙΣΛΑΜΑΪ | 1070736 | 5ο | miraslamaj@gmail.com |
| ΔΗΜΗΤΡΑ ΜΑΥΡΙΔΟΥ | 1070770 | 6ο | up1070770@upnet.gr |
| ΧΡΥΣΟΥΛΑ ΚΑΤΣΑΝΤΑ | 1067503 | 5ο | up1067503@upnet.gr |

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ

Αρχές Γλωσσών Προγραμματισμού & Μεταφραστών

ΔΙΔΑΣΚΟΝΤΕΣ: Ιωάννης Γαροφαλάκης, Σπυρίδων Σιούτας, Παναγιώτης Χατζηδούκας

# BNF ΓΡΑΜΜΑΤΙΚΗ

ΕΔΩ ΘΑ ΜΠΕΙ Η ΤΕΛΙΚΉ ΜΟΡΦΉ ΤΗΣ ΓΡΑΜΜΑΤΙΚΗΣ ΚΑΙ ΜΕ ΤΙΣ ΔΙΑΦΟΡΕΣ ΠΑΡΑΔΟΧΕΣ ΠΟΥ ΜΠΟΡΕΙ ΝΑ ΠΡΟΣΤΕΘΟΥΝ.

### ΠΑΡΑΔΟΧΕΣ

* Για δική μας διευκόλυνση, θεωρήσαμε πως τα χαρακτηριστικά των στοιχείων θα μπαίνουν μόνο με μια συγκεκριμένη σειρά. Συγκεκριμένα , θα πρέπει να γράφονται πάντα τα υποχρεωτικά χαρακτηριστικά πρώτα και μετά τα προαιρετικά του κάθε στοιχείου. Τα χαρακτηριστικά “android:layout\_width” και “android:layout\_height” (που είναι και καθολικά υποχρεωτικά χαρακτηριστικά) θα πρέπει να γραφτούν πρώτα για να θεωρηθεί σωστή η σύνταξη του xml αρχείου. Στην συνέχεια, θα πρέπει να γραφτεί το ειδικό υποχρεωτικό χαρακτηριστικό του στοιχείου , εάν έχει , για παραδειγμα στο TextView μετα τα χαρακτηριστικα “layout\_width” και “layout\_height” θα πρέπει να εμφανίζεται το “android:text” και τέλος μπαίνουν τα προαιρετικά χαρακτηριστικά. Γενικότερα , ακολουθήθηκε η σειρά με την οποία τα χαρακτηριστικά αυτα εμφανίζονται στην εκφώνηση της άσκησης .
* Χρησιμοποιήθηκαν τα διπλά εισαγωγικά **“ ”**, με την λογική που παρατίθεται στο παρακάτω λινκ:

<https://eng.libretexts.org/Bookshelves/Computer_Science/Programming_and_Computation_Fundamentals/Foundations_of_Computation_(Critchlow_and_Eck)/04%3A_Grammars/4.02%3A_Application_-_BNF>

Ειδικότερα, έγινε η χρήση τους μόνο για τα “<” και “>” , διότι αυτά τα δύο σύμβολα έχουν συγκεκριμένη σημασία στην BNF γραμματική και θα θέλαμε να αποφύγουμε την σύγχυση με την ύπαρξή τους στην γλώσσα XML. Το ίδιο έγινε και με τα σύμβολα **“** και **|**, που χρησιμοποιούνται και στην xml και στην bnf.

# FLEX ΚΩΔΙΚΑΣ

### DECLARATIONS

Αρχικά ορίζονται οι βιβλιοθήκες που θα χρησιμοποιήσουμε , όπως και οι συναρτήσεις και μεταβλητές. Συγκεκριμένα, η συνάρτηση yyerror() ,ορίζεται στον τμήμα των auxiliary functions, δηλώνεται για την διαχείριση των ορθογραφικών σφαλμάτων. Η μεταβλητή errors αρχικοποιείτε και χρησιμοποιείται ως μετρητής σφαλμάτων.

Δηλώνεται και μία μεταβλητή τύπου int ενσωματωμένη στο Flex (%option yylineno), όπου κάθε φορά που θα συναντά το Flex τον χαρακτήρα νέας γραμμής ('\n'), η μεταβλητή θα αυξάνεται κατά 1 αυτόματα.

Δηλώθηκαν επίσης δύο exclusive start condition, για την περιγραφή των τιμών μέσα σε διπλά ειαγωγικά (%x VALUE) και για την περιγραφή των σχολίων (%x COMMENTS) , όπως αναφέρεται στο Flex manual από την σελ. 21.

Τέλος, δίνονται κάποιες ονομασίες για κάποιες κανονικές εκφράσεις που θα χρειαστούν, με τις ποιο σημαντικές τα {ΙΝΤ} και {STRING}. Το {INT}, περιγράφει όλους τους ακέραιους αριθμούς μεγαλύτερο ή ίσο του μηδενός. Το {STRING} περιγράφει αλφαριθμητικά που ξεκινούν από τουλάχιστον ένα γράμμα και μπορούν να έχουν ανάμεσα ή στο τέλος κάτω παύλα “\_” , παύλα “-”κενό “ ” και αριθμούς. Κάποια παραδείγματα : “match\_parent”, “TV2”, “button-12 red” ενώ το “12something” θεωρείται λάθος.

%%

### TRANSLATION RULES

Στο τμήμα των translation rules ορίστηκαν αρχικά όλες οι λέξεις-κλειδιά , καθώς ο flex λεκτικός αναλυτής αναζητεί από πάνω προς τα κάτω εάν ένας κανόνας υφίσταται στο αρχείο που εξετάζουμε. Συγκεκριμένα οι λέξεις-κλειδιά που ορίστηκαν είναι όλα τα ονόματα των στοιχείων (LinearLayout, RelativeLayout κλπ.) και τα χαρακτηριστικά των στοιχείων (layout\_width, layout\_height, textColor κλπ.) ξεχωριστά από την λέξη-κλειδί “android:” που επίσης ορίζουμε.

Μετά τις λέξεις-κλειδιά , περιγράφονται οι τιμές που παίρνουν τα χαρακτηριστικά με τον ύπο-αναλυτή <VALUE>. Μέσα σε αυτόν τον αναλυτή ,που εφόσον είναι exclusive δεν επηρεάζεται από τους κανόνες “έξω” από αυτόν, μόλις βρεθεί ένας ακέραιος σύμφωνα με τον ορισμό του {INT} μέσα σε εισαγωγικά επιστρέφεται ένα συγκεκριμένο token για τους αριθμούς, μόλις βρεθεί ένα αλφαριθμητικό που συμπίπτει με τον ορισμό του {STRING} επιστρέφετε το αντίστοιχο token , και σε οποιαδήποτε άλλη περίπτωση έχουμε να κάνουμε με σφάλματα που τα διαχειρίζουμε μέσω της yyerror , εμφανίζοντας ένα μήνυμα λάθους.

Με έναν ίδιο τρόπο ορίζονται και τα σχόλια με τον υπο-αναλυτή <COMMENTS>. Σαν κανόνες σε αυτόν τον υπο-αναλυτή έχουμε πως αγνοούμε ότι υπάρχει ενδιάμεσα στα “<!--” και “-->” , με μόνη εξαίρεση το “--” που μόλις διαβαστεί από το flex εμφανίζεται μήνυμα ορθογραφικού λάθους.

Και στους δύο υπο-αναλυτές άμα τελειώσει το αρχείο χωρίς να τους κλείσουμε, τα <comment> με “-->” και τα <VALUE> με διπλό εισαγωγικό **“** , εμφανίζεται σφάλμα.

Πολύ σημαντικό ήταν να ορίσουμε κανόνες για ετικέτες αρχής , ετικέτες τέλους και για το “=”, όπου αυτό βλέπουμε στην συνέχεια.

Με κάθε κανόνα , επιστρέφεται ένα συγκεκριμένο token (λεκτική μονάδα) εκτός από τα κενά και τα σχόλια.

%%

### AUXILIARY FUNCTIONS

Εδώ ορίσαμε την συνάρτηση yyerror() , όπου αναφέραμε και παραπάνω, εκτυπώνει το σφάλμα που προέκυψε και σε ποια γραμμή. Άμα εμφανιστούν 10 σφάλματα συνολικά τερματίζεται η εκτέλεση του προγράμματος.

# BISON ΚΩΔΙΚΑΣ

Στο πρώτο κομμάτι του bison, ορίσαμε τις απαραίτητες αρχικοποιήσεις των βιβλιοθηκών της c, καθώς και κάποιες βοηθητικές μεταβλητές για την εκτέλεση της συντακτικής ανάλυσης. Σε αυτό το κομμάτι καλέσαμε επίσης τις συναρτήσεις ,που ορίζονται στο flex, με το keyword extern. Με αυτό τον τρόπο καταφέρνουμε να μεγαλώσουμε την εμβέλεια ορατότητας της c με σκοπό να έχουμε πρόσβαση στα εξής του flex :

**yyin**, **yylineno**(που έχει περιγραφεί παραπάνω), **yyparse**() για να την επικυροποίηση των δεδομένων εισόδου, **yylineno , yyval** (τιμές της εισόδου) **, yylex.**

### DEFINITIONS

Μετά τις δηλώσεις σε c προχωρήσαμε στην δήλωση των tokens. Σε αυτό το σημείο δηλώνονται όλα τα tokens που έχουν δηλωθεί ότι επιστρέφονται από το flex και μέσω του bison δημιουργείται το αρχείο bisonFile.tab.h που περιλαμβάνει κάποιες μοναδικές τιμες (κωδικούς) για όλα τα tokens. Το σημείο αυτό είναι πολύ σημαντικό καθώς εδώ γίνεται η σύνδεση των εισόδων που αναγνωρίζει το flex με τους κανόνες που υπακούν στην γλώσσα που ορίζουμε.

### RULES

Το κομμάτι των κανόνων στο bison είναι και αυτό που ορίζει το συντακτικό που πρέπει να ακολουθεί οποιοδήποτε αρχείο εισόδου. Ανάλογα με την αλληλουχία χαρακτήρων που διαβάζει το flex, επιστρέφει τα αντίστοιχα tokens στο bison, όπου και μέσω των εκφράσεων ορίζονται οι πιθανοί αποδεκτοί συνδιασμοί που αναγνωρίζει ο compiler που σχεδιάσαμε.

Για την σύνθεση των διαφορετικών κανόνων πρώτα σχεδιάσαμε τους κανόνες με το BNF και μετά ανάλογα με την σύνταξη του bison κάναμε τις απαραίτητες αλλαγές. Στις περιπτώσεις που θέλαμε να έχουμε το κενό αλφαριθμητικό ως είσοδο, χρησιμοποιήσαμε το %empty.

Μετά την εμφάνιση των τελευταίων %% συμβόλων προχωρήσαμε στην δήλωση των συναρτήσεων.

### SUBROUTINES

Σε αυτή την περιοχή του bison μεταφέραμε την συνάρτηση yyerror που στην αρχή είχαμε ορίσει στο flex. Η συνάρτηση σε αυτή την περίπτωση εμφανίζει κατάλληλα μυνήματα για να ενημερώσει τον χρήστη σχετικά με τα λάθη που αναγνώρισε το bison σε συνδιασμό με το flex.

Επίσης σε αυτό το σημείο έχουμε όσες συναρτήσεις προέκυψαν από τα ερωτήματα 2 και 3.

Πιο συγκεκριμένα, η συνάρτηση idUnique χρησιμοποιείται για την αποθήκευση των id των στοιχείων σε array και τον έλεγχο μοναδικότητάς τους. Αρχικά, υπάρχει ένας βρόχος που ελέγχει τις ήδη υπάρχουσες τιμές id και αν κάποια από αυτές είναι ίδια με την τρέχουσα τιμή, εμφανίζει ανάλογο μήνυμα λάθους. Στη συνέχεια, υπάρχει μία if, η οποία χρησιμεύει για την αύξηση του μεγέθους του array, κάθε φορά που δεν υπάρχει χώρος για την αποθήκευση κάποιας τιμής. Τέλος, το μοναδικό id προστείθεται στο array.

Τέλος, εδώ ορίζουμε την main συνάρτηση όπου ανοίγουμε το αρχείο εισόδου που δίνει ο χρήστης και το διατρέχουμε με το flex. Μέσω της yyparse κάνουμε διάβασμα των αντίστοιχων tokens και αντιστοιχία στους κανόνες. Αν δεν υπάρχει κάποιο error από την εκτέλεση τότε εμφανίζουμε το κατάλληλο μήνυμα επιτυχίας.

# ΚΑΤΑ ΤΗΝ ΕΚΤΕΛΕΣΗ

### MAKE FILE

Η εκτέλεση του flex και bison έγινε σε ένα περιβάλλον linux με την προεγκατάσταση των flex, bison μέσω terminal. Για την εκτέλεση του κάθε αντίστοιχου αρχείου πραγματοποιούσαμε επαναλαμβανόμενες κινήσεις, οπότε μέσω του αρχείου Makefile καταγράψαμε τις γραμμές που επανεκτελούσαμε στο terminal και με την εντολή make στο σωστό directory εκτελούσαμε ταυτόχρονα bison,flex και gcc compiler. Επομένως, καταλήγαμε με το αρχείο **run ,**που είναι εκτελέσιμο, μέσω του οποίου μπορούμε να ελέγξουμε το αρχείοο εισόδου που ειπθυμούμε.

### TESTING ΚΑΙ ΣΤΙΓΜΙΟΤΥΠΑ ΟΘΟΝΗΣ ΑΠΟ ΤΗΝ ΕΚΤΕΛΕΣΗ

Παρακάτω, παραθέτουμε κάποια στιγμιότυπα οθόνης, όπου δοκιμάζουμε τον αναλυτή μας με διαφορετικά test files.

Περίπτωση: Ίδιο χαρακτηριστικό android:id

Αποτέλεσμα: Μήνυμα σφάλματος

A screen shot of a computer program

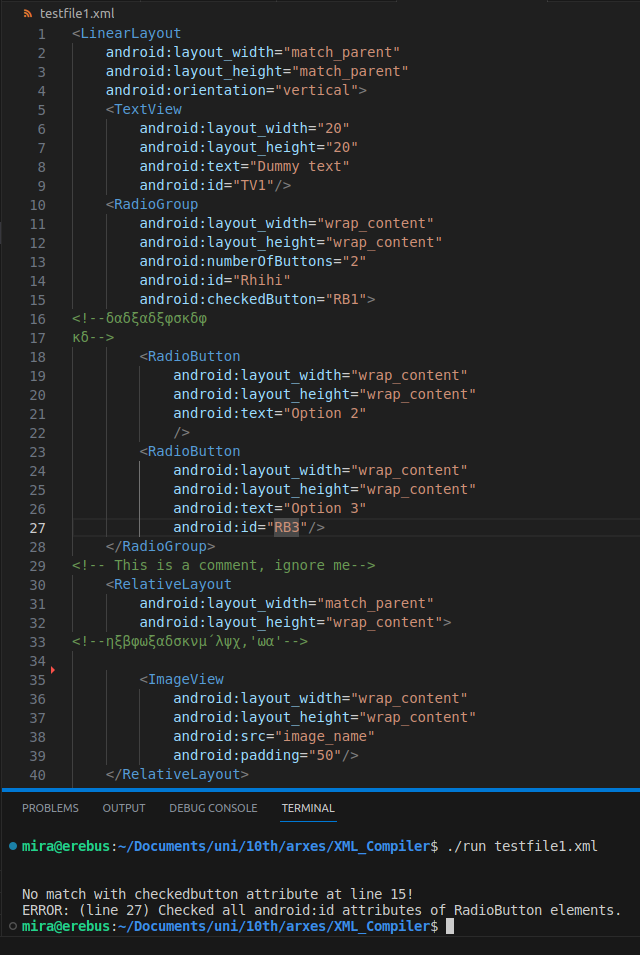
Description automatically generated

A screen shot of a computer

Description automatically generated

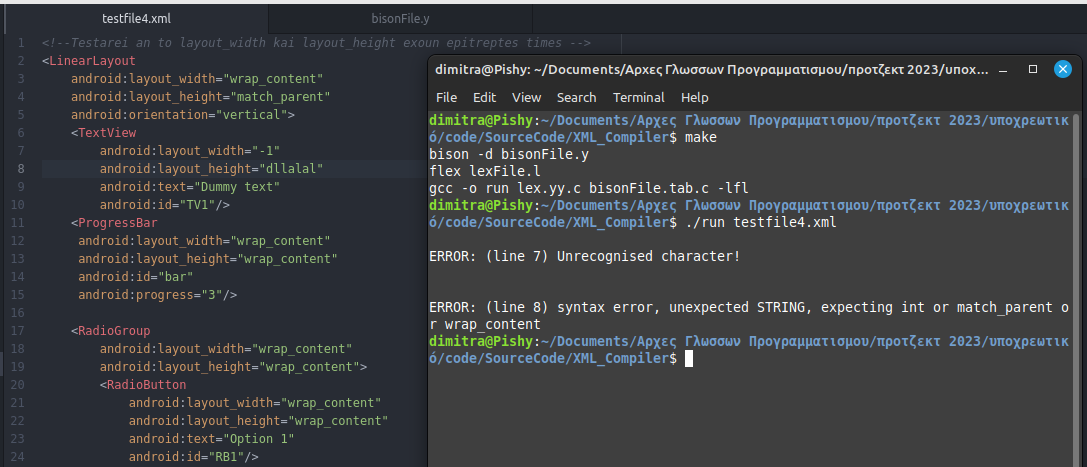
Περίπτωση: Δεν έχει δηλωθεί κανένα χαρακτηριστικό id και επίσης τα radio buttons δεν έχουν id

Αποτέλεσμα: Το checked button μπορεί να πάρει οποιαδήποτε τιμή



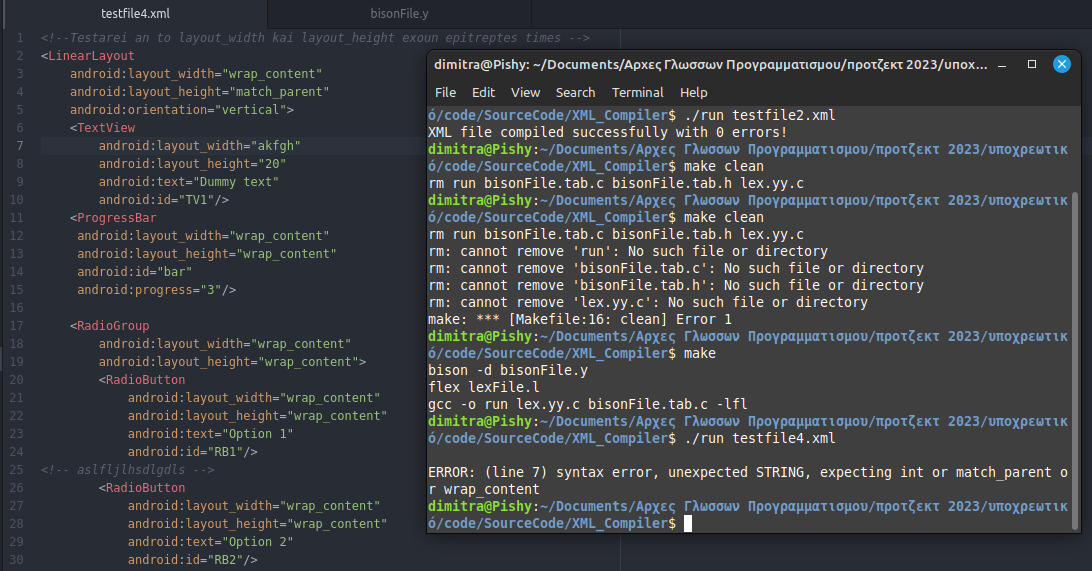
Περίπτωση: Τα χαρακτηριστικά android:layout\_width και android:layout\_height έχουν μη επιτρεπτές τιμές

Αποτέλεσμα: Εμφάνιση σφάλματος



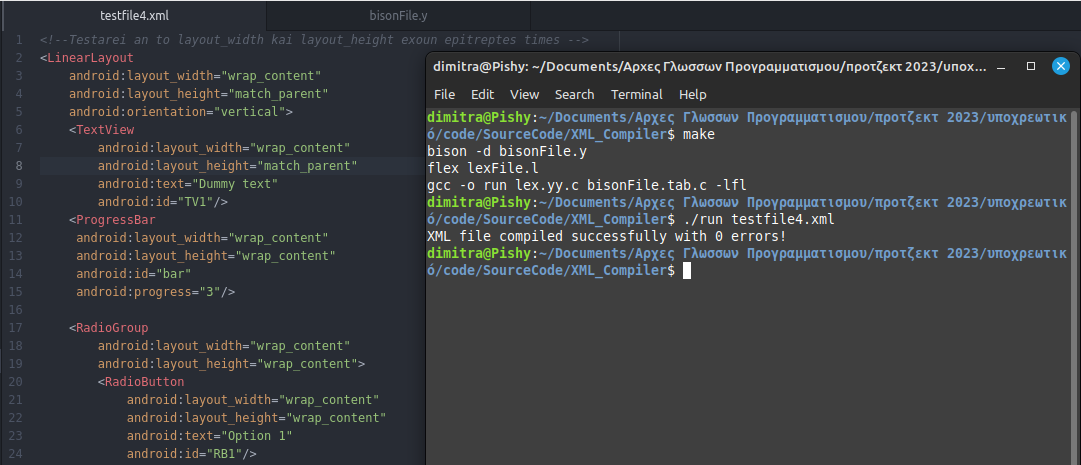
Περίπτωση: Το χαρακτηριστικό android:layout\_width έχει μη επιτρεπτή τιμή

Αποτέλεσμα: Εμφάνιση σφάλματος



Περίπτωση: Τα χαρακτηριστικά android:layout\_width και android:layout\_height έχουν επιτρεπτές τιμές.

Αποτέλεσμα: Δεν υπάρχει κανένα σφάλμα.



Περίπτωση:

Αποτέλεσμα: