ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΑΡΟΥΣΙΑΣΗ / ΕΞΕΤΑΣΗ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ντούλας Μάριος
Μεταπτυχιακός Φοιτητής

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης
Επόπτης Μεταπτυχιακής Εργασίας: Καθηγητής, Α. Σαββίδης

Μ. Τρίτη, 27 Απριλίου 2021 , ώρα 15:00 μ.μ.

Join Zoom Meeting

https://zoom.us/j/97710869575

" Βελτιωμένο IntelliSense με Μεταφορείς Τύπων"

Περίληψη

Το IntelliSense έχει σημαντικό αντίκτυπο στη διαδικασία ανάπτυξης λογισμικού. Οι αυτοματισμοί κατά την επεξεργασία του πηγαίου κώδικα βοηθούν τους προγραμματιστές στην πλοήγηση, την κατανόηση, την αποφυγή σφαλμάτων και την επιτάχυνση της συνολικής διαδικασίας επεξεργασίας. Στις μέρες μας, η χρήση γλωσσών προγραμματισμού χωρίς τύπους και η ποσότητα του κώδικα χωρίς τύπους σε συστήματα λογισμικού τείνει να κλιμακώνεται, όπως και η χρήση βιβλιοθηκών κώδικα τρίτων. Ωστόσο, πολλά σφάλματα εξακολουθούν να επισκιάζονται από τη δυναμική φύση του κώδικα χωρίς τύπους που καθιστά τη σημασιολογική ανάλυση μια δύσκολη και μερικές φορές μη υπολογίσιμη εργασία. Ενώ υπάρχουν επεκτάσεις για τέτοιες γλώσσες και νεότερες εκδόσεις που εισάγουν κατασκευές βασισμένες στις κλάσεις, υπάρχει ακόμα πολύς κώδικας χωρίς τύπους που χρησιμοποιείται αλλά δεν υποστηρίζεται, αλλά και πολλοί προγραμματιστές προτιμούν την ευελιξία και την εκφραστική οικονομία του σύμπαντος χωρίς τύπους, αποδεχόμενοι την έλλειψη της ασφάλειας που θα τους προσέφεραν αυτοί. Σε αυτό το πλαίσιο πιστεύουμε ότι υπάρχει ανάγκη για βελτιωμένα εργαλεία επεξεργασίας, ικανά να αναλύουν και να αποτιμούν σταδιακά τα κομμάτια

πηγαίου κώδικα των γλωσσών χωρίς τύπους κατά την επεξεργασία τους, τα οποία τελικά παρέχουν βελτιωμένη ανατροφοδότηση τύπων κατά απαίτηση στους προγραμματιστές.

Παρουσιάζουμε τις τεχνικές για τη σημασιολογική ανάλυση του πηγαίου κώδικα χωρίς τύπους κατά την επεξεργασία εστιάζοντας στη γλώσσα JavaScript. Το σύστημά μας υλοποιείται πάνω από το Visual Studio Code IDE και εκμεταλλεύεται τα άγκιστρα επέκτασης προγράμματος που προσφέρει το Language Server Protocol για αυξητική επεξεργασία και αυτοματισμούς. Η προσέγγισή μας βασίζεται στην έννοια των μεταφορέων τύπων οι οποίοι, σχετίζονται με τις εντολές που αλλάζουν την τιμή ή τον τύπο μιας μεταβλητής και την σύνδεσή αυτών κατά την επεξεργασία με τρόπο που επιτρέπει να παρακολουθείται με ακρίβεια από οποιαδήποτε τοποθεσία προέλευσης στον κώδικα η στοίβα των ενεργών μεταφορέων τύπων ανά σύμβολο, ώστε να μπορεί κατά απαίτηση να αποφασιστεί ο εξαρτώμενος από τα συμφραζόμενα εύλογος τύπος.

University of Crete

Computer Science Department

M.Sc. Thesis presentation / examination

Ntoulas Marios

Master's Thesis Supervisor: Professor, A. Savidis

Tuesday, 27 April 2021, 15:00 p.m.

Join Zoom Meeting

https://zoom.us/j/97710869575

“More Informative Untyped IntelliSense with Type Carriers”

Abstract

IntelliSense has a major impact in the development process. The automations during source code editing assist developers in navigating, understanding, avoiding errors, and speeding-up the overall editing process. Currently, the use of untyped languages and the quantity of untyped code in software systems tends to escalate, including the deployment of third-party untyped code libraries. However, many errors are still shadowed by the dynamic nature of untyped code making semantic analysis a difficult and sometimes undecidable job. While there are typed language extensions, and sometimes newer versions introducing class-based constructs, not only there is still a lot of untyped legacy

code, but many programmers prefer the abstraction flexibility and expressive economy of the untyped universe, although acknowledging this is traded for lack of type safety. In this context we believe there is a need for improved editing tools capable to analyze and evaluate incrementally the source code fragments of untyped languages while being edited, which eventually deliver more informative on-demand type feedback to programmers.

We present the techniques for the semantic analysis of untyped source code during editing focusing on the JavaScript language. Our system is implemented on top of the Visual Studio Code IDE and exploits the editor extension hooks offered by the Language Server Protocol for incremental parsing and editing automations. Our approach is based on the notion of type carriers which are associated to instructions that change the value or the type of a variable, and their chaining during editing in a way enabling to precisely track from any given source location the stack of active type carriers per symbol, thus being able to on-demand tell its plausible context-dependent type.