Angaben zur Person

Ort und Art der Durchführung: Vor Ort, Windows 10, Remetschwil
Datum der Durchführung: 27.Mai.2020
Test wird durchgeführt von: Marcel Hess

Vorinterview

Dient zur Einschätzung der Fähigkeiten des Probanden und klärt ihn über die Ziele und Absichten des Tests auf.

Vorabinformationen:
Dem Probanden darüber informieren, dass dieser Test nicht ihn sondern unser Plugin testet. Er darf jederzeit unbeschwert allfällige Probleme äussern. Er kann in dem Sinne nichts falsch machen. Den Probanden grob mit unserer Arbeit vertraut machen und was das Ziel unseres Plugins ist. Grobinformation über Dafny selbst, und dass für den Test keine Dafny-Kenntnisse benötigt werden, da Codebeispiele zur Syntax bereitstehen. Für den Probanden werden grundsätzliche Programmierkenntnisse vorausgesetzt. In dieses Dokument hat nur der Testdurchführer Einsicht.

Wie heisst du, wie alt bist du, programmierst du beruflich oder in deiner Freizeit?
Remo Herzog, 25 Jahre, primär in der Freizeit.

Welche Programmier-/Skriptsprachen verwendest du am meisten?
Ich verwende primär Python und selten JavaScript.

Weisst du was eine IDE ist? Kannst du mir grob dein Verständnis einer guten IDE schildern?
Zum Beispiel VSCode. Eine Umgebung, in welcher man programmieren kann, und gleich alle nötigen Tools integriert sind.

Welche IDE verwendest du am häufigsten/liebsten?
Ich arbeite fast ausschliesslich mit VSCodium auf Linux.
Von der Bedienung ist es sozusagen dasselbe wie VSCode.

Welche Features einer IDE schätzt du besonders?
Für mich ist das automatische Markieren von Fehlern eine essenzielle Funktion. Auch sehr wichtig für mich in Verbindung mit Python ist das direkte Ausführen eines Codeteils zum Prüfen, obs funktioniert wie gewollt. Dass die Entwicklungsumgebund ein modernes Aussehen hat ist auch kein unwesentlicher Bestandteil. Inklusive Darkmode versteht sich. Ein altmodischer Auftritt hat meiner Meinung nach beispielsweise Notepad++.

Hast du bisher schon mal etwas sehr störendes an einer IDE erlebt? Wenn ja, was?
Bei Python gibt es eine Umgebung, mit welcher man Pakete installiert. Codium erkennt das leider nicht richtig. So kann ich als Entwickler nicht einfach «run» klicken und die Pakete werden installiert, sondern ich muss Manuel in der Konsole die Pakete installieren. Das empfinde ich als unnötigen Zusatzaufwand.

Hast du VSCode schon einmal verwendet? Für was/welche Sprachen? Positives Erlebnis?
Ja - sozusagen. Primär für Pyhton.

Kennst du Dafny und hast allenfalls schon mal mit Dafny etwas programmiert?
Ich hab schonmal davon gehört, es aber selbst nie verwendet.

Szenarien

Vorbedingung: *VSCode und der Dafny Language Server sind gestartet und einsatzbereit. Die benötigte Dateien sind bereits im Workspace geöffnet. Andere Plugins von Dritten für VSCode sind deaktiviert.*

Aufgabe 1 – Zahl hochzählen
Du findest eine Datei «Aufgabe1_Verwenden» und eine Date «Aufgabe1_NichtVerwenden».
Bitte öffne die Datei «Aufgabe1_Verwenden».
Darin findest du bereits ein kleines Dafny-Programm inklusive Einstiegspunkt (Main).
Im *Include* auf der ersten Zeile wird die Datei «Aufgabe1_NichtVerwenden» eingebunden.
Bitte schau dir diese Datei NICHT an.

Finde heraus, was für ein Typ die Variable «number» hat und wo sie deklariert wurde.
Ziel: Wird Hover Information genutzt?

> *«In Python hätte ich jetzt beispielsweise mit «type» gearbeitet und eine entsprechende Ausgabe getätigt.» Der Proband sucht nach Debugging. Ist etwas verwirrt. Kleine Hilfestellung, dass man nicht zu weit suchen muss.*
>
> *Nun fährt er mit der Maus über die Klasse. Dies liefert zwar den Hover, aber nicht die gewünschten Informationen. Er erwartet hier eine Art CodeDoc der Klasse. Anschliessend fährt er über die Variable und er erhält die gesuchten Informationen.*
>
> *Die vielen Informationen auf viel Raum verwirren etwas. Weniger Text wäre wünschenswert, eine bessere Gruppierung. Das Feld «Kind» verwirrt etwas. Ein Hinweis «was genau was macht bzw bedeutet» wäre hilfreich. Das liegt aber allenfalls auch an der fehlenden Vertrautheit zu Dafny. Etwas in Richtung einer Entwicklungsdokumentation wäre wünschenswert.*
>
> *Ausserdem haben «Symbol» und «Declaration» zu viel Text. Weniger Text, höherer Informationsgehalt. Beispielsweise die Position in Klammern als reine Zahl*

Verwende die dir nicht weiter bekannte Klasse «Counter» des inkludierten Files um die Variable «Number» um eins hochzuzählen.
Wird die Autocompletion verwendet? Ausreichende Hilfe? Bsp: c.increase(c.number);

> *Als erstes fährt der Proband wieder mit der Maus über «Counter» und «Number». Er erwartet eine Art Code Dokumentation, findet aber keine. Er fängt an zu tippen «c.». Nun ist er verwundert, dass keine Autocompletion auftaucht. Nach einem Punkt erwartet er automatisch eine Autocompletion. Er erhält den Hinweis, dass man aktuell Steuerung + Leerschlag drücken muss.*

Bei den vorgeschlagenen Methoden fehlen die Parameter. Das verwirrt etwas. Bei der Fehlermeldung (rot unterstrichen) würde die Information enthalten sein, dies fällt ihn aber nicht auf. Bessere Benutzerführung mit dem Augenfokus bei Fehlermeldungen wären allenfalls hilfreich.
Auch für die Behebung des fehlenden Semikolons wird ein Moment gebraucht. Er findet dies allerdings von alleine heraus und nicht mit Hilfe des Fehlertexts.

Bitte Kompiliere dein Programm, führe es aus und überprüfe die Ausgabe.
Wird «compile + run» entdeckt und verwendet?

Oben rechts wird ein grüner «Play Button» gesucht. Er ist verwirrt, dass dieser nicht dort ist. (Gewohnheit von Python). Solch ein Button wäre hilfreich.

Als nächstes wird ein Rechtsklick versucht. Dort findet der Proband sofort die «Compile and Run» Option.

Die vielen Meldungen verwirren ihn etwas. «Was ist jetzt passiert.»
Bei der Ausgabe wünscht er sich weniger Text. Der Pfad zum aktuellen Prozess sowie die Ausgabe der ausgeführten Executable sind zum Grossteil identisch. Das könnte man vereinfachen, dass nicht so viel Text ausgegeben wird. Sprich dass vom aktuellen Pfad ausgegangen wird. Das wäre ihm sympathischer.

Szenario 2 – Codebereinigung
Öffne die Datei «Aufgabe2» und scroll in der Datei NICHT herunter. Du möchtest herausfinden, welche Klasse im oberen Teil der Datei wie oft verwendet wurde. Gibt es dead-code? Wo wurde die Klasse ClassA überall verwendet?
Wird CodeLens verwendet? Wird die nicht verwendete Methode in ClassC entdeckt?

CodeLens wird direkt verwendet, sofort gefunden. Der tote Code wird auch sofort erkannt.

Zusatzfragen im Gespräch:
Bei TypeScript werden die Referenzen aufgeteilt nach Scope. Dann gibt es beispielsweise die Auszeichnung «1 Reference| 1 Reference» wobei der erste Link die internen Referenzen (this) der Klasse und die zweiten Referenzen die Verwendungen von aussen (über Instanzvariablen) auszeichnen. Fändest du diese Aufteilung angenehmer?
Nein. Ich will ja mit einem Klick gleich alle Referenzen sehen; egal über welchen Weg darauf zugegriffen wird. Im Popup sehe ich ja dann, was von wo kommt.

Wenn eine Klasse oder Methode keine Referenz hat, wird diese mit dem Vermerk «wird noch nicht verwendet – kann dieser Code gelöscht werden?» ausgezeichnet. Bei Typescript wird beispielsweise einfach «0 Referenzen» ausgewiesen. Welche Variante bevorzugst du und warum?
Ich finde das mit dem Text eigentlich besser. Es ist «mehr Text» und «anders» wodurch es sofort auffällt. Dead Code will man ja grundsätzlich nicht haben. Also darf das ruhig etwas ins Auge stechen.

Szenario 3 – Fehlerbehebung
Öffne die Datei «Aufgabe3». Jemand hat hier nicht sauber gearbeitet. Findest du den Fehler? Kannst du dir Beispiele anzeigen lassen, für welche konkreten Fälle die Funktion sich falsch verhalten wird?
Wird CounterExample verwendet?

> Der Proband erkennt durch Überlegen sofort, dass «less» grösser sein könnte als «more». Dann verwendet er den Rechtsklick und geht Punkt für Punkt durch. «CounterExample» kennt er zwar nicht, hört sich aber passend an. Das gewünschte Ergebnis wurde erzielt.
>
> Zusatzfrage im Gespräch:
> Du versuchst jetzt grundsätzlich eher den Rechtsklick. Bei VSCode gäbe es ja noch die integrierte Eingabeaufforderung für Befehle. Dort hätten wir die Befehle auch hinterlegt. Verwendest du diese grundsätzlich weniger?
> *Diese verwende ich eigentlich nie.*

Du möchtest nun die Variable «more» zu «bigger» umbenennen. Wie gehst du vor? Kannst du das automatisiert so umsetzen bitte?
Wird Rename verwendet?

> Der Proband verwendet «Control + H» und ersetzt die Begriffe mit «Suchen und ersetzen». Zusätzlich aktiviert er die Option, nur ganze Wörter und nicht Wortbestandteile zu ersetzen. Nach dem Hinweis, dass dies in gewissen Fällen auch ungewollte Ersetzungen zur Folge haben kann, verwendet er erfolgreich den Rechtsklick und «Rename».
> *Dannach verwendet er rechtsklick.*

Nachinterview

Sind Probleme während der Tests aufgetreten?

Der Play Button oben rechts habe ich etwas vermisst. Bei der Completion erwarte ich Vorschläge sobald ich einen Punkt eingetippt habe.

Hast du etwas vermisst? Fehlte etwas bei den bisherigen Feature? Ein ganz anderes Feature?

Grundsätzlich die oberen beiden Funktionen. Ansonsten eigentlich nicht.

Hat dir etwas besonders gut gefallen?

Das Counter Example war sehr cool. Das ist sicher sehr praktisch.
Das Underlining ist sicher auch nützlich wenn was falsch ist. Falls die Meldungen kürzer uns aussagekräftiger wären, würde ich es aber noch besser finden.

Empfandst du die Features als hilfreich? Würdest du sie verwenden? (Mehrwert)
Welches der Features findest du am nützlichsten? Warum?

Das Umbenennen ist noch überraschend praktisch (statt Suchen & Ersetzen).

Allgemeine Anmerkungen, Inputs was man sich noch wünschen würde:

Das Wichtigste fänd ich das Autocompletion. Und falls Dokumentationsstrings für Klassen und Methoden hinterlegt werden könnten, fände ich das sehr nutzbringend.