

Explore It!



Elisabeth Hendrickson (@testobsessed) arbeitet als Testerin, Entwicklerin und »Agile-Enabler«. Im Jahr 1980 schrieb sie ihre erste Codezeile und fand sofort ihre ersten Fehler. 2010 gewann sie den renommierten Gordon Pask Award von der Agile Alliance. Sie ist bekannt für ihren Google Tech Talk über Agile Testing sowie ihre beliebten Testheuristiken-Spickzettel. Sie teilt ihre Zeit auf zwischen lehren, vortragen, schreiben, programmieren und der Arbeit in agilen Teams, die ihr Engagement beim Testen sehr schätzen.



Übersetzerin: Meike Mertsch arbeitet als begeisterte Testerin für Magine AB in Stockholm, Schweden. Sie hat einen Hintergrund als agiler Entwickler und Coach und ein Faible für leichtgewichtige Methoden wie Kanban und Personal Kanban sowie agiles Entwickeln und Testen. In ihrer Freizeit läuft und skatet sie durch Stockholm oder macht die Kletterhallen und Felsen in der Nähe unsicher.

Elisabeth Hendrickson

Explore It!

**Wie Softwareentwickler und Tester
mit explorativem Testen Risiken reduzieren
und Fehler aufdecken**

Aus dem Amerikanischen übersetzt von Meike Mertsch



dpunkt.verlag

Elisabeth Hendrickson
elisabeth@testobsessed.com

Übersetzung: Meike Mertsch, Stockholm
Lektorat: Christa Preisendanz
Copy-Editing: Ursula Zimpfer, Herrenberg
Satz: Petra Strauch, just in print, Bonn
Herstellung: Birgit Bäuerlein
Umschlaggestaltung: Helmut Kraus, www.exclam.de
Druck und Bindung: M.P. Media-Print Informationstechnologie GmbH, 33100 Paderborn

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

ISBN:
Buch 978-3-86490-093-8

1. Auflage, Translation Copyright für die deutschsprachige Ausgabe © 2014 dpunkt.verlag GmbH
Wiebinger Weg 17
69123 Heidelberg

Copyright der amerikanischen Originalausgabe © 2013 The Pragmatic Programmers, LLC.
Title of American original: Explore It! Reduce Risk and Increase Confidence with Exploratory Testing
Pragmatic Bookshelf, The Pragmatic Programmers, LLC, Dallas, Texas, Raleigh, North Carolina
<http://pragprog.com>
ISBN-13: 978-1-937785-02-4

Die vorliegende Publikation ist urheberrechtlich geschützt. Alle Rechte vorbehalten. Die Verwendung der Texte und Abbildungen, auch auszugsweise, ist ohne die schriftliche Zustimmung des Verlags urheberrechtswidrig und daher strafbar. Dies gilt insbesondere für die Vervielfältigung, Übersetzung oder die Verwendung in elektronischen Systemen.

Es wird darauf hingewiesen, dass die im Buch verwendeten Soft- und Hardware-Bezeichnungen sowie Markennamen und Produktbezeichnungen der jeweiligen Firmen im Allgemeinen warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.

Alle Angaben und Programme in diesem Buch wurden mit größter Sorgfalt kontrolliert. Weder Autor noch Verlag können jedoch für Schäden haftbar gemacht werden, die in Zusammenhang mit der Verwendung dieses Buches stehen.

5 4 3 2 1 0

Inhaltsverzeichnis

Teil 1

Grundlagen schaffen	1
1 Testen und Erforschen	3
1.1 Zwei Seiten des Testens	4
1.1.1 Checking (dt.: Prüfen)	5
1.1.2 Erforschen	5
1.1.3 Getestet = geprüft und erforscht.	6
1.2 Hauptbestandteile von explorativem Testen.	6
1.2.1 Tests erstellen.	6
1.2.2 Durchführen	7
1.2.3 Lernen	7
1.2.4 Steuern.	7
1.3 In zeitbeschränkten Sessions arbeiten	8
1.4 Für die Praxis	8
2 Ihre Forschungen chartern	11
2.1 Die Expedition chartern	12
2.2 Eine einfache Chartervorlage	13
2.3 Gute Charter	15
2.4 Charter erstellen.	17
2.4.1 Anforderungen.	17
2.4.2 Implizite Erwartungen	19
2.4.3 Charter koordinieren Ziele.	19
2.4.4 Die Fragen der Stakeholder	20
2.4.5 Bestehende Artefakte	21
2.4.6 Neue Erkenntnisse und Entdeckungen	22

2.5	Das Spiel mit Horrorschlagzeilen.	22
2.5.1	Schritt 1: Voraussetzungen schaffen	22
2.5.2	Schritt 2: Sammeln Sie Schlagzeilen.	23
2.5.3	Schritt 3: Suchen Sie sich ein großes Risiko zum Arbeiten aus	23
2.5.4	Schritt 4: Beteiligte Ursachen finden	24
2.5.5	Schritt 5: Ursachen zu Charter weiterentwickeln.	24
2.6	Charter planen	25
2.7	Für die Praxis	25
3	Details beobachten	27
3.1	Aber haben Sie auch den tanzenden Bären gesehen?	27
3.2	Tiefer graben.	29
3.2.1	Die tiefer gehende Frage stellen	30
3.2.2	Rechnen Sie mit unterschwelligem Hinweisen	31
3.2.3	Ein unerwartetes Geräusch	31
3.3	Testbarkeit und Unsichtbares sichtbar machen	32
3.4	Konsolen und Logs	33
3.5	Für die Praxis	35
4	Interessante Variationen finden	37
4.1	Variablen sind Dinge, die sich verändern.	38
4.1.1	Offensichtliche Variablen	38
4.1.2	Weniger offensichtliche Variablen.	39
4.1.3	Indirekt zugängliche Variablen	40
4.2	Weniger offensichtliche Variablen und große Katastrophen	40
4.2.1	Der Therac-25-Fall	40
4.2.2	Die Ariane 5	41
4.2.3	Die Marssonde	41
4.3	Variablen erkennen	42
4.3.1	Dinge, die Sie zählen können.	42
4.3.2	Relative Position	43
4.3.3	Dateien und Speicher.	44
4.3.4	Geografische Lage.	44
4.3.5	Formate	45
4.3.6	Größe	46

4.3.7	Tiefe	47
4.3.8	Timing, Frequenzen, Dauer	47
4.3.9	Eingaben und Navigation	48
4.4	Variablen! Sie sind überall!	48
4.5	Für die Praxis	49
5	Testergebnisse beurteilen	51
5.1	Nie und immer	52
5.1.1	Kernfähigkeiten	53
5.1.2	Qualitätsfaktoren	53
5.1.3	Risiken	54
5.2	Weitere Quellen	55
5.2.1	Interne Konsistenz	55
5.2.2	Standards	55
5.2.3	Vergleichsprodukte	56
5.3	Näherungen	57
5.3.1	Mit einem Wertebereich bewerten	58
5.3.2	Eigenschaften untersuchen	58
5.3.3	Das Ergebnis umkehren	59
5.3.4	Voraussetzungen festlegen	59
5.4	Für die Praxis	60
Teil 2		
Weitere Möglichkeiten ins Spiel bringen		
<hr/>		
6	Handlungsabläufe und Interaktionen abwandeln	63
6.1	Substantive und Verben	64
6.2	Zufällige Navigation	66
6.3	Personas	67
6.4	Für die Praxis	69
7	Entitäten und ihre Beziehungen erforschen	71
7.1	Entitäten, Merkmale und Abhängigkeiten erkennen	71
7.1.1	Nicht offensichtliche Entitäten finden	72
7.1.2	Beziehungen ausarbeiten	73

7.2	CRUD: Create, Read, Update, Delete	74
7.2.1	CRUD mit Datenvariationen.	75
7.2.2	Alle Wege zu CRUD	76
7.2.3	Abhängigkeiten zwischen CRUD und »Null, eins, viele«.	76
7.3	Den Daten folgen	77
7.4	Für die Praxis	78
8	Zustände und Übergänge entdecken	79
8.1	Zustände und Ereignisse erkennen.	80
8.1.1	Zustände sind Verhaltensformen.	80
8.1.2	Ereignisse lösen Zustandsübergänge aus	81
8.2	Ein Zustandsmodell erstellen.	82
8.2.1	Beschränken Sie Ihren Fokus.	83
8.2.2	Erkennen Sie verschiedene Perspektiven	83
8.2.3	Ändern Sie die Abstraktionsebene.	84
8.3	Mit Zustandsmodellen forschen	85
8.3.1	Alle Wege	85
8.3.2	Unterbrechen.	86
8.3.3	Zurück zu den Variablen.	87
8.4	Die Darstellung ändern: Zustandstabellen.	88
8.5	Für die Praxis	90
9	Das Ökosystem untersuchen	93
9.1	Das Ökosystem grafisch darstellen	94
9.1.1	Schritt 1: Die Schnittstellen bezeichnen	94
9.1.2	Schritt 2: Externe Abhängigkeiten aufzeichnen	95
9.1.3	Schritt 3: Die Interna ausfüllen	96
9.1.4	Vermeiden Sie die Analyse-Paralyse.	97
9.2	Vertrauensgrenzen.	98
9.3	Was wenn ...?	99
9.3.1	Netzwerkverbindungen	100
9.3.2	Dateien	100
9.4	Den Daten folgen	101
9.5	Für die Praxis	102

Teil 3**Zusammenhänge herstellen 103****10 Ohne Benutzeroberfläche forschen 105**

- 10.1 Eine API untersuchen 106
- 10.2 Eine Programmiersprache erforschen 108
- 10.3 Einen Webservice untersuchen 110
- 10.4 Fehler beschreiben 112
- 10.5 Für die Praxis 113

11 Ein existierendes System erforschen 115

- 11.1 Mit einer Aufklärungssession beginnen 115
- 11.2 Beobachtungen gemeinsam machen 118
- 11.3 Stakeholder befragen, um Problemstellungen zu sammeln 120
- 11.4 Techniken auswählen 122
- 11.5 Ihre Ergebnisse festhalten 123
- 11.6 Was ist mit den gefürchteten nicht reproduzierbaren Fehlern? .. 124
 - 11.6.1 Sammeln Sie Hinweise 125
 - 11.6.2 Listen Sie Variablen auf, die dazu beitragen können . . . 125
 - 11.6.3 Nutzen Sie Zustandsdiagramme, um Zeitzusammenhänge zu verstehen 126
 - 11.6.4 Arbeiten Sie zusammen 126
- 11.7 Für die Praxis 126

12 Anforderungen untersuchen 129

- 12.1 In das Anforderungsmeeting hineinkommen. 130
 - 12.1.1 Wenn man außen vor gelassen wird 130
 - 12.1.2 Missverständnisse aus Einzeldiskussionen 131
 - 12.1.3 Eine Testbesprechung wird zum Anforderungsmeeting . 132
- 12.2 Im Anforderungsmeeting 133
 - 12.2.1 Zentrale Werte bestimmen 133
 - 12.2.2 Fragen Sie »Was wenn ...?« 134
 - 12.2.3 Erwartungen schärfen 135
- 12.3 Charter in Anforderungsdiskussionen. 136

12.4	Aktives Lesen	138
12.4.1	Das Papier befragen	138
12.4.2	In kleinere Stücke aufteilen	139
12.4.3	Ein Modell aufstellen	139
12.5	Für die Praxis	140
13	Forschungen in den gesamten Entwicklungsprozess einbinden	141
13.1	Forschungen als Teil Ihrer Teststrategie.	141
13.1.1	Beispiele aus dem wahren Leben	141
13.1.2	Prüfen und Forschen in die Entwicklung einbeziehen	143
13.1.3	Früh und oft forschen	144
13.1.4	Aufwände für Forschungen einplanen	145
13.2	Forschen in Paaren	145
13.3	Systemische Problemquellen finden	147
13.4	Forschungen abschätzen	148
13.5	Wann haben Sie genug geforscht?	149
13.6	Nachbesprechungen mit Stakeholdern.	150
13.6.1	Im Daily Standup berichten.	150
13.6.2	Nachbesprechung mit Entscheidern.	151
13.7	Nützliche Lebensweisheiten erfassen	153
13.8	Für die Praxis	153
Anhang		155
A.1	Bewerbungsgespräche zu Fähigkeiten in explorativem Testen	157
A.1.1	Exploratives Testen im Paar als Teil eines Bewerbungsgesprächs	158
	Vorbereitung	158
	Start	158
	Den Kandidaten beobachten	159
	Nachbereitung.	159
	Steuern	160
A.1.2	Kandidaten einschätzen.	160

A.2 Testheuristiken-Spickzettel	161
A.2.1 Generelle Heuristiken	161
Alles dezentralisieren	161
Alles zentralisieren	161
Anfang, Mitte, Ende	162
CRUD	162
Das Modell ändern	163
Datenformatregeln missachten	163
Den Daten folgen	163
Einige, keine, alle	164
Goldlückchen	164
Nie und immer	164
Null	164
Null, eins, viele	165
Nützliche Näherungen	165
Umkehren	165
Unterbrechung	165
Verhungern	166
Zoom	166
Zu viele	166
Zu wenige	166
Zusammenfassen	167
A.2.2 Webheuristiken	167
Lesezeichen setzen	167
Zurück, vorwärts, Verlauf	167
 Literatur	 169
 Index	 171