RICK KUIPERS
Developer bij Enrise
enrise.com / @rskuipers

TEST-DRIVEN DEVELOPMENT IN DE PRAKTIJK

Wil je complexe logica testen, dan is TDD misschien iets voor je

WD: Sinds wanneer werkt Enrise met test-driven development?
Begin 2010, tijdens de ontwikkeling van de nieuwe website van AutoTrack, zijn wij begonnen met het toepassen van test-driven development (TDD). AutoTrack heeft de kwaliteit van auto-informatie hoog in het vaandel staan. Voor ons reden genoeg om de zoekmachine- en businesslogica voor de weergave van deze informatie met TDD te ontwikkelen.

WD: Wat is test-driven development precies?
Test-driven development is een werkwijze waarbij er eerst op basis van criteria een geautomatiseerde test wordt geschreven. Deze test voer je vervolgens uit en zal in eerste instantie falen. Het idee is om op basis van die test je code te gaan schrijven. Zodra je denkt de functionaliteit te hebben geïmplementeerd, draai je de test nogmaals. Op deze manier creëer je een hele korte feedbackloop. Dankzij de geautomatiseerde tests worden continu alle criteria getest, terwijl jij je kunt focussen op het schrijven van de daadwerkelijke code.

Het grote verschil tussen eerst je test schrijven en je test pas schrijven nadat de functionaliteit is gebouwd, is dat het een architecturele opzet forceert. Met de vooraf geschreven test bepaal je hoe je met de code wilt gaan communiceren (een contract). Het gevolg hiervan is vaak nettere en duidelijkere code die is bedacht vanuit het oogpunt van de gebruiker. TDD komt dus zowel ten goede van de klant als van de code.

Naast dat TDD interessant is bij het ontwikkelen van nieuwe features of bugfixes, is het ook een handige aanpak bij refactoring. Refactoring is waar je veranderingen in de code aanbrengt, zonder per se de functionaliteit aan te passen. TDD is perfect om tijdens het refactoren in de gaten te houden of je applicatie nog naar behoren functioneert.

WD: Waarom is hiervoor gekozen en hoe is het ingepast in de werkwijze?
Businesslogica is het belangrijkste onderdeel van een website, denk bijvoorbeeld aan het beheren van bedrijfsprocessen of iets als een webshop. Dat er tests voor moeten worden geschreven is een feit, maar met TDD forceer je deze mindset en zal een test nooit achterwege worden gelaten. Hiervoor is TDD een belangrijk onderdeel van onze werkwijze geworden. Om TDD goed te ondersteunen, beginnen de projecten bij Enrise altijd met het opzetten van de testtools en de testomgeving. De tests draaien automatisch zodra een ontwikkelaar nieuwe code wil introduceren. Bij een falende test wordt de code geweigerd en moet de ontwikkelaar de code updaten om deze wel te laten voldoen aan de testcriteria. De codebase zal dus altijd bestaan uit alleen maar succesvolle tests.

Het is ook een goede manier om bugs aan te pakken. Je schrijft eerst een test die de bug reproduceert, vervolgens ga je de code tweaken totdat deze test en alle andere tests succesvol zijn. Dit zorgt er ook meteen voor dat je een geautomatiseerde test hebt voor die bug en die kan in die vorm dus ook niet meer terugkomen.

WD: Waarom is het zo belangrijk om te testen tijdens het developmentproces?
Bij een traditioneel developmentproces gebeurt het nog weleens dat bij het ontwikkelen van een feature of bugfix een andere feature omvalt. Dit kan komen doordat de ontwikkelaar een stuk code ergens heeft moeten aanpassen om de nieuwe feature te kunnen bouwen. Echter gebeurt het vaak dat dit pas erg laat, soms zelfs pas bij oplevering, ontdekt wordt. Om dit zoveel mogelijk te voorkomen, testen we tijdens het ontwikkelen. Dankzij de korte feedbackloop komt dit probleem snel ter sprake en kan de ontwikkelaar hier ook direct een oplossing voor vinden.

WD: In welke situaties is het handig om een dergelijk testproces te hebben? En wanneer niet?
TDD is niet voor elk project de juiste keuze. Bij Enrise hebben we veel te maken met complexe website-applicaties, echter hebben veel ontwikkelaars ook te maken met simpelere contentsites waar de businesslogica zich beperkt tot het kunnen publiceren van content. In dit geval wordt TDD vaak ervaren als overhead. De tijd die je erin stopt uit zich in dit geval vaak niet tot toegevoegde waarde. Daarnaast zijn er al veel goed geteste contentmanagementsystemen beschikbaar waarop je kunt vertrouwen. TDD vereist ook een bepaalde investering van de klant. Dat het waarde toevoegt is duidelijk, maar dit betekent niet altijd dat de klant bereid is daarvoor te betalen.

WD: Wat hebben jullie geleerd tijdens het implementeren/uitvoeren van dergelijke tests?
Het schrijven van tests is stap één, maar de uitdaging ligt vooral in het schrijven van goede tests. Tests helpen je bij het vinden van problemen, maar tests kunnen ook je productiviteit in de weg gaan zitten als je voor elke regel code die je aanpast één of meerdere regels testcode moet aanpassen. *Sensible testing* is een term die vaak gebruikt wordt in zulke situaties. Er moet een goede balans zijn die je ontwikkelproces ten goede komt en niet in de weg zit. Het vinden van deze balans is een kwestie van vallen en opstaan.

Zoek naar deze balans en voorkom de drang naar dingen als 100% code coverage, het is niet erg dat niet elke regel code getest is. Houd de focus op het toevoegen van waarde en voorkom dat je verzeild raakt in het idee dat TDD de enige methode om te werken is. Als je begint met TDD in een project, zorg ervoor dat je dit met het hele team omarmt. Doe je dit niet, dan resulteert dit al snel in achterstallige tests en dat heeft een averechts effect.