

- Learn and understand the concepts of Behavior-Driven Development (BDD)
- Build the right thing through collaboration and shared understanding
- Increase innovation, lower stress, lower risk, and lower bug count
- Explore the relationships between BDD and its surrounding ideal environment – DevOps and agile
- Solidify your understanding with hands-on exercises

If agile is correctly implemented, adding Behavior-Driven Development (BDD) to the development process will not be difficult. It is an enhancement that can be used alongside other development strategies (e.g. TDD) to greatly increase the quality of the final product. With small incremental changes, BDD can give your team that fresh start and a new way of thinking that will take your programming to the next level.

Learn the key concepts of BDD, including business and team effects, participant roles, difficulties and traps, and more. Leave being able to practice and promote BDD at your company with primary and supporting technologies, transitions to using BDD, and more.

## Who Should Attend

This course is appropriate for those who lead teams in the definition, development, and quality assurance of software related products. It is recommended that you are familiar with concepts of the agile methodology, as taught in our [Fundamentals of Agile class](#) [1], before taking this course.

## Hands-on Exercises

In this workshop you will learn about testing application behavior through hands-on activities, exercises, discussions, and demos.

- Demonstrations and exercises using BDD
- Simulation of sprint planning incorporating BDD
- Presentations, discussions, and debate of BDD culture and mindset
- Discussion of transitioning into using BDD
- Convincing your team and management of its value

## Course Outline

---

### Introduction to BDD

General introduction to BDD concept  
Show demo of running in Jenkins pipeline  
Terminology  
Building the right thing

### Benefits of usage

BDD Demo  
3 amigos meeting  
Terminology  
Focusing on domain

### Practice

Discovering user stories  
Difference between feature and user story  
Expounding on user stories  
A great user story!  
Well-written user story  
Poor user stories  
Clarity  
Adjustments to the schedule  
How leaderships can help  
Gherkin syntax

Becoming a more valuable tester  
Becoming a more valuable developer  
Awesome business results  
Business effects  
Live documentation  
Higher productivity  
Higher morale

### **Origins**

Kent Beck  
Dan North

### **Roles**

Business stakeholder  
Developer  
Tester  
Special advisers  
End users

### **Mindset and Culture**

Communication isn't easy  
Collaboration  
Sustainability  
Maintainability and testability  
Colocation versus remote  
Trust and empowerment  
Zen Mindset  
Psychological safety  
Theory versus practice

### **Difficulties and Traps**

It is not about the tools  
Reduce redundancy  
Minimum viable product  
General organizing principles  
Common antipatterns  
Avoid fragile scenarios  
Company culture preventing collaboration and trust  
Changing existing precedent and culture is hard  
Transitioning to a new technique will cause productivity drop  
Poorly written scenarios can increase maintenance and derail progress  
Business should be available to team  
Not following Agile tenets

### **Transitioning to BDD**

Small incremental changes  
Team and management buy-in  
Fresh start - leaving behind the baggage  
Simple and small

### **Case Studies**

**Price:** \$1495