ProTech protechtraining.com *"Charting the Course ... ... to Your Success!"*

Introduction to Gradle

Course Summary

Description

Within Agile methods such as SCRUM and Kanban automated build management is essential to enable continuous testing, integration and deployment. Without a fully autonomous 'build->test->release' cycle to act as its heartbeat an Agile team has no effective means of detecting errors or measuring progress. Additionally if the build system is not self-contained and isolated from external dependencies it is difficult for developers to transfer their work between machines and setup new ones.

Traditionally build management in Java has been based around Maven. Whilst Maven enables rapid progress at the beginning of a project its inflexibility quickly leads to issues as the project grows. In particular the XML based syntax and complex underlying architecture means that customizing the framework for bespoke project needs is an arduous process.

Gradle is an alternative to Maven which has quickly overtaken it as the default build tool in JEE. Unlike Maven a Gradle build script is a program written in the Groovy JVM scripting language, making it trivial to inject arbitrary functionality into the build process. This DSL (Domain Specific Language) based approach is the same as that used to great effect within the 'Ruby on Rails' web framework.

Gradle embeds micro frameworks (aka Plugins) which support all the standard tasks required of a built tool. It is fully interoperable with Maven and can (for example) download libraries from existing Maven repositories. In addition it is designed to enable the modern style of 'polyglot programming' – where multiple JVM languages are used within the same project.

This course will enable developers to manage their builds with Gradle, extend the tool to shorten and simplify their deployment process and use Gradles power features to explore new forms of software development.

Topics

- Overview of Build Automation
- Introduction to Groovy
- Introducing Gradle
- Gradle Builds in Depth
- Advanced Uses of Gradle

Prerequisites

Students must be proficient with a JVM language (Java, Groovy, Scala or Clojure) with at least one year's experience developing applications in industry.

Duration

Two days

Introduction to Gradle

Course Outline

I. Overview of Build Automation
- A. The early days of Make and Ant
- B. How Rake pioneered DSL's in 'RoR'
- C. Why Maven became dominant in JEE
- D. Issues with the Maven approach
- E. The emergence of Gradle

II. Introduction to Groovy
- A. Dynamic typing within the JVM
- B. Basic syntax and concepts
- C. Closures and functional programming
- D. Techniques for building DSL's

III. Introducing Gradle
- A. Creating a build script for a console application
- B. Making the script self-contained via the Wrapper
- C. Configuring and connecting to repositories
- D. Managing dependencies and creating discrete scopes
- E. Exclusions and controlling transitive dependencies

IV. Gradle Builds in Depth
- A. The lifecycle of a build within Gradle
- B. Creating a graph of inter-dependent tasks
- C. Managing the file system and source code control (VCS)
- D. Configuring source sets using the Java plugin
- E. Installing and managing additional plugins
- F. Enabling TDD, BDD and CI using Gradle plugins
- G. How the Gradle Daemon reduces startup times
- H. Documenting your script for external use

V. Advanced Uses of Gradle
- A. Building multi-module JEE Web Applications
- B. Enabling use of Java, Scala and Clojure within the same project
- C. Running code quality tools (Checkstyle, Jacoco, PMD etc…)
- D. Using Gradle to build Android and iOS mobile applications
- E. Supporting alternative Java Web Architectures (e.g. Play)
- F. Advantages of using Gradle to build C and C++ projects