Mathematical Problems of Computer Science 50, 107--110, 2018.

Research and Comparative Analysis of Build Technology Programming in Java

Aren K. Mayilyan and Levon M. Hovsepyan

National Polytechnic University of Armenia
e-mail: mayilyan96@mail.ru, lehovsepyan@gmail.com

Abstract

Build tools have been studied in Java environment, and a comparable analysis has been done according to the following parameters: initial learning curve, the speed of different builds, complexity, plugins, documentation and community, developer tools integrity.

Keywords: Compilation, Project, Tool, Template, Script, Configuration, Logic, Concept, Binary.

1. Introduction

Many programmers start a project with selecting the language and framework for programming, then the most important problem is to choose the build and deploy tool, for what you need to know which parameters to compare and determine which is the most appropriate for the project to be performed. Developers spend a substantial amount of time doing mundane tasks like build and deployment that include:

- ➢ Compiling the code
- ➢ Packaging the binaries
- ➢ Deploying the binaries to the test server
- ➢ Testing the changes
- ➢ Copying the code from one location to another

Build tools are programs that automate the creation of executable applications from source code. Building incorporates compiling, linking and packaging the code into a usable or executable form. In small projects, developers will often manually invoke the build process. This is not practical for larger projects, where it is very hard to keep track of what needs to be built, in what sequence, and what dependencies exist in the building process. Using an automation tool allows the build process to be more consistent [1].

2. Definition and Comparative Analysis of Build Tools

Apache Ant (Another Neat Tool) is a Java-based build tool from Apache Software Foundation. Apache Ant's build files are written in XML, and they take advantage of being open standard, portable and easy to understand [2].

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information. Using maven we can build and manage any Java-based project [3].

Gradle is an open source, advanced general purpose build management system. It is built on ANT, Maven, and lvy repositories. It supports Groovy-based Domain Specific Language (DSL) over XML. [4]:

So, let’s make a comparative analysis of the build tools described above and summarize in Table 1.

Table 1: Comparative analysis of build tools.

Comparison parameters	Build tools		
	Apache Ant	Apache Maven	Gradle
Initial learning curve	Learning is not hard, it takes a long time for the practical works done for build tools, dependency and XML. There are a lot of templates in internet.	Learning Maven is not a hard thing to do and you do not need to be familiar with Maven to be able to build and package the artifact or run unit tests on some existing project [5].	The learning curve for Gradle is partially affected by another JVM language–Groovy. With just a little understanding of Groovy, it is really easy to get it working and understand it thoroughly.
Different builds speed	Clean function without tests lasts 7.378sec, with tests - 13.496sec, build function without tests lasts 4.829sec, with tests - 10.705sec.	Clean function without tests lasts 6.279sec, with tests - 13.451sec, build function without tests lasts 5.552sec, with tests - 13.357sec.	Clean function without tests lasts 3.342sec, with tests - 13.081sec, build function without tests lasts 3.638sec, with tests - 10.030sec.

Create and maintain the build script	The complexity of Ant build scripts varies a lot, depending on what you're trying to accomplish. Especially when dealing with many targets, the dependency graph between them can easily be unclear, but still enjoy the benefit of the dependencies being explicitly stated in the script!	When reading Maven build scripts, there are some conventions that you should be aware of this is sometimes known as "The Maven Way". Mostly, this refers to the super POM and contains some relatively non-obvious (i.e., invisible) inheritance and aggregation rules, giving Maven an implicit order of lifecycles.	The readability of Gradle build scripts is quite simple. It depends who wrote the script and how, but in general, while taking a look at these scripts, it can roughly see what is the script doing, even without much Groovy experience.
Count of plugins and how simple it is to customize your own plugins	Ant has as many plugins as Maven. Creating your own plugin is also as simple as Maven, but more specifications in the script itself are needed to make it work. Can't write code directly in the build script.	Sometimes Maven is called "plugin execution framework", because if you want to do something that Maven currently can't do, you'll need to find a plugin or write one yourself. There is simply no other alternative.	Excellent flexibility, but the library of existing plugins is meager– perhaps since it's easy to write your own plugins, users might do that rather than creating and hosting plugins in the community [6].
Community and documentation	Many experts who have used it for many years, but the community activity is pretty much dead. Documentation is outdated looking but sufficient [7].	Documentation is relatively good, but the community, forum and activity is past its prime.	A lot of active users, wikis and excellent documentation–all backed up by Gradleware, an actual company that supports the tool's ecosystem. You can even email someone who will write back to you. It has a free book.
Integration with developer tools	It integrates in more environment than the gradle, but less than maven.	Full support for each tool and every category. There is a reason that the majority of developers use Maven, after all.	Doesn't integrate in very old developer tools, because is a new build tool.

3. Conclusion

For initial training, maven and gradle build tools are preferred. If required to build simple and shorter scripts, then the gradle build tool is preferred. For the number of plugins, the preference is given to maven, but its own plugin is easier to write with gradle. All build tools can be selected from the viewpoint of study and acquaintance, however, gradle has more recent and plain documents, free book, and maven is also rich in documents, and most of them are outdated in the case of ANT. Maven integrates in all environments and is ideal for large projects. Gradle is more suitable for medium sized projects.

Referenses

[1] [Online]. Available: https://www.techopedia.com/definition/16359/build-tool
[2] [Online]. Available: https://www.tutorialspoint.com/ant/
[3] [Online]. Available: https://www.tutorialspoint.com/maven/
[4] [Online]. Available: https://www.tutorialspoint.com/gradle/
[5] [Online]. Available: https://www.baeldung.com/ant-maven-gradle
[6] [Online]. Available: https://technologyconversations.com/2014/06/18/build-tools/
[7] [Online]. Available: https://devopscube.com/list-of-popular-open-source-java-build-tools/

Submitted 08.06.2018, accepted 02.11.2018.

JAVA միջավայրում build ծրագրավորման տեխնոլոգիաների հետազոտումը և համեմատական վերլուծությունը

Ա. Մայիլյան և Լ. Հովսեփյան

Ամփոփում

Հետազոտվել են Java միջավայրում build գործիքները և կատարվել է վերջիններիս համեմատական վերլուծությունն ըստ հետևյալ պարամետրերի․ սկզբնական ուսուցում, տարբեր կառուցումների արագություն, կառուցման սկրիպտների կառուցումն ու պահպանումը, plagin-ների քանակը և սեփական plugin-ների կոնֆիգուրացումը նախագծում, փաստաթղթեր և հետադարձ կապ, ինտեգրումը ծրագրավորման միջավայրերի հետ:

Исследование и сравнительный анализ технологии build программирования в среде JAVA

А. Маилян и Л. Овсепян

Аннотация

Исследованы технологии build программирования в среде Java. Проведен сравнительный анализ по следуюшим параметрам: начальное знакомство, скорость построения, сложность, плагины, документация и обратная связь, интегрирование в программные среды.