Dokumentationstools

Tool/Kriterium	HTML-Kentnisse?	Programmiersprachen	Diagramme	Betriebssystem	Eingabeformate	Ausgabeformate
Javadoc	Ja	Java	Ja	windows, OS X, Linux, BSD, Unix	Text, nicht Binär	HTML, CHM, RTF, PDF, LaTeX, PostScript, man pages, DocBook, XML
Doxygen	Nein	Java, C, C++, Objective-C, Python, Fortran, IDL, PHP, C#, D, VHDL, Tcl, Object Pascal	ja	windows, OS X, Linux, BSD, Unix	Text, nicht Binär	HTML, CHM, RTF, PDF, LaTeX, PostScript, man pages, DocBook, XML

Entscheidung: Wir haben uns als Dokumentationstool für Javadoc entschieden, da es Industriestandard zur Dokumentation von Java-Klassen und in JDK enthalten ist.

Javadoc kann außerdem auch eine Vielzahl von Diagrammen erstellen und bietet eine strukturierte Ansicht des Quellcodes.

Versionierung

Tool/Kriterium	Versionskon-trollsystem	Binärdateien	Branching	Sicherheit	BS
hg	mehrere dezentrale Repositorys aber ein Zentrales	Zum Bearbeiten wird lokale Kopie des Projekt-Repository erstellt	Branches können nicht entfernt werden; Risiko, Code an falschen Branch zu übergeben	Standardmäßig nicht möglich, Historie zu verändern	Verwendung leichter auf Windows
git	mehrere dezentrale Repositorys aber ein Zentrales	Speichern problematisch -> wachsen bei jeder hochgeladenen Änderung um die gesamte Speichergröße	leicht, Branch ist nur Referenz auf Commit in lokalem Repository	Historien von lokalen Repository-Kopien durch jeden änderbar, Schäden am Projekt möglich	Auf Linux fokussiert; komplizierter auf Windows
svn	zentrales Respository aber dezentrale Arbeit an Dateizweigen möglich	nur zu bearbeitender Zweig wird heruntergeladen und neu gespeichert	Verzeichnisse innerhalb Repositorys	Historie-Änderung nur mit Zugang zu zentralem Server möglich	Keine großen Unterschiede

Entscheidung: Unser Versionierungstool soll für dieses Projekt Git sein. Der Hauptgrund hierbei besteht darin, dass Git ein sehr bekanntes und weit verbreitetes Versionierungswerkzeug ist.

Es erwies sich in vergangenen Projekten, dass es sich durch das einfache „mergen“ von „Branches“ hervorragend zum Zusammenarbeiten und Strukturieren unsere Projekte eignet.

IDE

Tool/Kriterium	Programmiersprache	Programmart	Reaktionszeit	Erweiterungen	kosten
Eclipse	Java, über Plug-ins auch andere möglich	IDE	mit zunehmenden Prjekten langsamer und reagiert träger	viele Plug-ins erhältlich	kostenlos
IntelliJ IDEA	Java, Kotlin, Groovy, Scala	IDE	schnell und komfortabel	geringere Anzahl als bei Eclipse	kostenlos
NetBeans	Java, auch C & C++ möglich	IDE	langsamer als Eclipse	Plug-ins für weitere Sprachen und	kostenlos

Entscheidung: Als IDE werden wir Eclipse benutzen, da wir diese Entwicklungsumgebung im Rahmen vorheriger Projekte ins Herz geschlossen haben. Es ist kostenlos und eignet sich hervorragend zur Entwicklung in der Programmiersprache JAVA.

Codeconventions

Tool/Kriterium	freier Platz	Linienlänge	Ausnahmen	Klammern für Klarheit im Code
Google	2 Leerzeichen	100	keine leeren catch-Klammern	wird Empfohlen
Mozilla	4 Leerzeichen	angepasst	keine Angabe	wie im Java-Standard
Oracle	4 Leerzeichen	80	keine Angabe	wie im Java-Standard

Entscheidung: Wir möchten für unseren Quellcode die Code-Conventions von Google benutzen. Es ist ein großer anerkannter Standard für die Gestaltung von Code und erfüllt seine Rolle in unserem Projekt hinsichtlich der einheitlichen Strukturierung des Quellcodes.

Build-Tool

Tool/Kriterium	Programmiersprache	Build-Sprache	Build-Prozess	Benutzerfreundlichkeit
ant	Java	XML	Befehle müssen aufgelistet und beschrieben werden	niedrig
maven	Java	XML	Projekte werden in kleine Teile zerlegt->POM	hoch
gradle	Java, Groovy	DSL	Azyklisch gerichteter Graph bestimmt Reihenfolge der Tasks	hoch

Entscheidung: Wir werden Maven benutzen, da es XML als Build-Sprache benutzt und gut strukturiert ist. Das Programm ist unserer Meinung außerdem recht benutzerfreundlich.

Kolaborationstools

Tools/Kriterium	Effektivität	Preis	Aufwand	Vorerfahrung	Sichtkontakt	Struktur
Mailinglist	gering	kostenlos	wenig	ja	nein	wenig
Treffen im Praktikum	hoch	kostenlos	hoch	ja	ja	wenig
Issue Tracker	mittel	kostenlos	mittel	nein	nein	hoch

Entscheidung: Wir benutzen "Treffen im Praktikum" um miteinander zu kolaborieren. Es bietet sich für uns als IT-Dienstleister am besten an, da wir im Rahmen unserer vorherigen Projekte gute Erfahrung damit gemacht haben. Mit der Namensgebung machen wir eine Anspielung auf die Zeit als wir noch studierten, da unsere Projekte in der Studienzeit damit immer gut funktioniert haben. Es ist außerdem hoch effizient.

Obfuscator

Obfuscator:	Betriebssystem	Programmiersprache
ProGuard	plattformunabhängig	Java
JavaGuard	plattformunabhängig	Java

Kategorie	Lizenz
Obfuscator	GPL2+ mit optionalen Ergänzungen
Build Tools, Code Generators, Compilers	GNU Library or Lesser General Public License version 2
TestNG	plattformübergreifend	Java

Beschreibung
freie Software, die kompilierte Java-Dateien komprimiert & optimiert. Erkennt und entfernt nicht genutzte Kategorien, Felder und Attribute.
Javaguard ist ein allgemeiner Byte-code Obfuscator

Wahl: Proguard: Aufgrund der Tatsache, dass sich Javaguard aktuell noch in der Beta-Version befindet, ist damit zu rechnen, dass viele Erneuerungen auf das Tool drauf zu kommen. Insofern ist ProGuard die weit aus bessere Wahl.

Test-Automatisierung

Test Automatisierung:	Betriebssystem	Programmiersprache
Junit	plattformübergreifend	Java
TestNG	plattformübergreifend	Java

Kategorie	Lizenz
Test-Framework	Eclipse Public License
Test-Framework	Apache-Lizenz 2.0

Beschreibung
Unterscheidung zwischen gelungenen Test (grün) und misslungenen Test (rot). Bei einem misslungenen Test wird eine entsprechende Exception ausgegeben. Es wird in Failure(erwarteter Fehler) und Error(unerwarteteer Fehler) unterschieden

Wahl: Junit: Aufgrund guter Arbeit des gesamten Teams mit der Entwicklungsumgebung Eclipse, wird als Test-Automatisierungstool Junit gewählt um somit Kosten beim Lizenzkauf zu sparen.

Prototyping der Benutzerschnittstelle

Tool\Kriterien	Effektivität	Preis	Aktualität	Vorkenntnisse
gomockingbird	Einfaches UI, kein Download nötig	Kostenlos / aber Anmeldung nötig	keine weiteren Updates seit 2017	keine
popapp	Anmeldung nötig, Bilder laden möglich (Skizzen), schwieriges Interface, kein Download, auch als App	Kostenlos / aber Anmeldung nötig	wird weiterentwickelt	keine
hotgloo	ähnlich wie mockingbird, einfach, mobil verwendbar	7 Tage kostenlos / 12€ pro Monat	aktuell	keine
ninjamock	Gute Zusammenarbeit durch Kommentare, einfach gehalte	Kostenlos / aber Anmeldung nötig	bekommt Updates	keine
moqups	Original Grafiken, Konzept Diagramme möglich	Kostenlos / bezahlen für weitere Features	aktuell	keine
pencil	Download, Diagramme möglich, Export in z.B. Bild möglich templates werden zur Verfügung gestellt	Kostenlos	keine neue Version seit 2017	keine

Entscheidung : Unsere Entscheidung fällt nicht auf Grund von Vorerfahrungen, da wir in diesem Bereich noch keine haben.

Wir entscheiden uns für NinjaMock, weil es kostenlos und einfach zu benutzen ist.

Uml-Tools

Tool/Kriterium	Typ	Plattform	Preis
TopCased	nutzt Infrastruktur von Eclipse	plattformunabhängig	kostenlos
StarUML	selbstständiges Softwareprodukt	plattformunabhängig	69 Euro
Papyrus	nutzt Infrastruktur von Eclipse	plattformunabhängig	kostenlos

Programmiersprachen	UML-Version	aktuell weiterentwickelt	Vorkenntnisse
C/C++, Java	UML 2.1.	nein	ja
C/C++, Java, C#, Python	UML	ja	nein
C/C++, Java	UML 2.5.	ja	nein

Entscheidung: Wir möchten TopCased als UML Tool verwenden, da wir bereits im Praktikum Erfahrungen damit sammeln konnten. Es ist in unseren Büros bereits installiert und steht uns kostenlos zur Verfügung.