**Software Engineering IV**

| **Thema** | **Auswahl** | **Seite** |
| --- | --- | --- |
| Versionierung | Git | 2 |
| UML-Tool mit Quellcodegenerierung | Papyrus | 3 |
| Build-Tool | Gradle | 4 |
| Tool für Prototyping der Benutzerschnittstelle | Figma | 5 |
| IDE/Editor | Intellij Community | 6 |
| Test-Automatisierung | Junit | 7 |
| Dokumentationstool | Doxygen | 8 |
| Obfuscator | ProGuard | 9 |
| Codeconventions | Google | 10 |
| Kollaborationstool | Discord | 11 |

**Versionierung**

|  | Mercurial  (hg) | **git** | Subversions  (svn) |
| --- | --- | --- | --- |
| Revisionshistorie | Unveränderlich (kein Rewrite) | Änderbar (z. B. Rebase) | Linear, zentral verwaltet |
| Branching-Modell | Einfach, isolierte Branches | Sehr flexibel, aber komplex | Umständlich (Kopien von Verzeichnissen) |
| Installation & Setup | Einfach, wenige Abhängigkeiten | Komplexer (Hooks, Configs) | Einfach für zentrale Teams |
| Geschwindigkeit | Schnell, aber etwas langsamer als Git | Sehr schnell, optimiert für große Repos | Langsamer, speziell bei großen Projekten |
| Verbreitung | Gering (hauptsächlich ältere Projekte) | Sehr weit verbreitet | Noch genutzt in Unternehmen |
| Ideal für | Einsteigerfreundliche verteilte Versionierung | Open-Source- und große Softwareprojekte | Unternehmen mit strikten Workflows |

***Fazit***: *Das von uns bevorzugtes Versionierungstool ist* ***Git****, da dieses sehr weit verbreitet ist und jeder aus unserer Gruppe damit Erfahrung gemacht hat.*

**UML-Tool**

|  | **Papyrus** | StarUML | VS |
| --- | --- | --- | --- |
| UML-Unterstützung | Vollständig (alle UML-Diagramme) | Vollständig (fokus auf UML 2.x) | Eingeschränkt (Klassendiagramm) |
| Geeignet für | Modellgetriebene Entwicklung (MDE) | Mittelgroße Softwareprojekte | .NET-basierte Software |
| Zielgruppe | Forschung, komplexe Projekte | Softwareentwickler, Architekten | .NET-Entwickler |
| Installation & Setup | Komplex (benötigt Eclipse) | Einfach (Standalone-Anwendung) | Einfach (integriert in VS) |
| Dokumentation & Community | Groß, aber komplex | Gute Dokumentation | Hervorragend für .NET |
| Anpassbarkeit | Sehr hoch (Eclipse-Plug-ins, MDA) | Mittel (Erweiterungen, einfache Konfiguration) | Gering (stark an .NET gebunden) |

***Fazit****: Das von uns bevorzugte UML-Tool ist* ***Papyrus****, da dieses perfekt für modellgetriebene Entwicklungen geeignet ist und vollständige UML-Unterstützung besitzt. Zwar müssen wir hierfür unseren Code in Eclipse integrieren, da Intellij Papyrus nicht unterstützt, allerdings sind die Alternativen nicht besser.*

**Build-Tool**

|  | Ant + Ivy | Maven | **Gradle** |
| --- | --- | --- | --- |
| Alter | 24 Jahre | 20 Jahre | 16 Jahre |
| Flexibilität | Hoch, keine spezifische Projekt Struktur vorgegeben | Niedrig, Konvention bevorzugt | Mitte, Konvention und Flexibilität |
| Syntax | XML, verboss, alle commands müssen selbst geschrieben werden | XML, verboss | DSL, ausdrucksstark, besser lesbar |
| Support | kleine Community | größte Community | Seit 2013 von Google für Android genutzt, wachsende Community, reguläre Updates |
| IDE Unterstützung | von 8 IDEs unterstützt  (https://ant.apache.org/manual/ide.html) | aufgrund längerer Existenz für die meisten Nutzer besser | Kotlin basierte DSL mit besser IDE Erfahrung, von vielen IDEs schon unterstützt |
| Abhängigkeitsmanagement | über Apache Ivy | rudimentäre Abhängigkeits Auswahl | Problemlösung mit transitiven Abhängigkeiten besser als Maven, anpassbare Auswahl- und Substitutionsregeln, Zusammenbau mehrere Quellprojekte |
| inkrementelle Builds | nein | nicht wirklich unterstützt | ja, bis zu 100 mal schneller als Maven |

***Fazit****:* ***Gradle*** *ist das hier bevorzugte Tool, es besitzt nicht nur eine hohe Kompatibilität mit Intellij, sondern auch reguläre Updates, sowie eine hohe Geschwindigkeit, ist generell gut zu verstehen und angenehm zu benutzen.*

**Prototyping Tool**

|  | **Figma** | Miro | PenPot |
| --- | --- | --- | --- |
| Plattform | Browser-basiert (Windows, Mac, Linux) | Browser-basiert (Windows, Mac, Linux) | Browser-basiert & selbst hostbar |
| Zielgruppe | UI/UX-Designer, Web- & App-Entwicklung | Teams für Brainstorming, Wireframes, Mindmaps | Open-Source- & Datenschutz-orientierte Designer |
| Kollaboration | Echtzeit-Kollabora-  tion, Kommentare | Echtzeit-  Whiteboard, viele Integrationen | Echtzeit-Kollabo-  ration |
| Benutzer-  freundlichkeit | Sehr intuitiv, moderne UI | Sehr einfach, aber weniger für UI-Design optimiert | Etwas komplexer, da neue Plattform |
| Offline-Nutzung | Desktop-App verfügbar | Nein | Desktop-App verfügbar |
| Zusätzliches |  |  | Open-Source |

***Fazit****:* ***Figma*** *überzeugt durch seine intuitive Benutzeroberfläche, leistungsstarken Prototyping-Funktionen und einfache Echtzeit-Kollaboration. Zudem ist es plattformunabhängig und die kostenlose Version bietet ausreichend Funktionen für viele Anwendungsfälle.*

**IDE/Editor**

|  | Eclipse | **Intellij Community** | Visual Studio Code |
| --- | --- | --- | --- |
| Komplexität | Hoch – viele Funktionen, steile Lernkurve | recht komplexer Aufbau | Gering – leichtgewichtige IDE |
| Geschwindigkeit | Mittel – kann bei großen Projekten langsam sein | Schnell – optimierte Performance | Sehr schnell – minimaler Ressourcenverbrauch |
| Zusätzliche Features | -Papyrus würde funktionieren | -viele Shortcuts für repetitive Codeabschnitte(zB. Variablenausgaben) | -Plugins sehr einfach zu integrieren |
| Versionierungs-  integration | Sehr gute Git-Unterstützung | Nahtlose Git-Integration | Sehr gute Git-Unterstützung |

***Fazit****:* ***Intellij*** *ist das modernste der 3 Umgebungen und bietet eine Menge Vorteile, trotz der hohen Komplexität besitzt Intellij eine Menge nützlicher Funktionen, sowie eine schnelle Geschwindigkeit bei der Ausführung.*

**Test-Automatisierung**

|  | **Junit** | Test-NG | Cucumber |
| --- | --- | --- | --- |
| Komplexität | recht simpel | extrem simpel | normal |
| Anwendungsmöglichkeiten | viele verschiedene Testarten | nur Standart Test Fälle | scenario cases für tests, keine “Standart” Tests |
| Java-Kompatibilität | Java 8 und höher | Java 11 oder höher | Nur Java 8 bis jetzt |

***Fazit****:* ***Junit*** *bietet ein breites Spektrum an Möglichkeiten, sowie eine recht simple Bedienungswiese und hat eine breitflächige Kompatibilität mit Java.*

**Dokumentationstool**

|  | Evernote | Javadoc | **Doxygen** |
| --- | --- | --- | --- |
| Kosten | 10$ pro Monat/bzw sehr eingeschränkte gratis Version | Kostenlos | Kostenlos |
| Komplexität | sehr simpel und einfach zu verstehen | macht der PC für dich, sehr simpel | macht der Pc für dich, sehr simpel |
| Aufbau | klar Strukturiert mit Pages geordnet nach Baumstruktur | direkt im Code an bestimmten Abschnitten, teilweise schwieriger Gesamtprojekt zu verstehen | Außerhalb des Codes, mit Beschreibungen für Klassen, pages usw. |

***Fazit****: Evernote ist zu teuer und Javadoc ist zwar gut, allerdings ist* ***Doxygen*** *für eine richtige Dokumentation besser geeignet, da alles klar strukturiert außerhalb des Codes dargestellt wird.*

**Obfuscator**

|  | **ProGuard** | JavaGuard | yGuard |
| --- | --- | --- | --- |
| Offenheit, Preis | Open-Source, gratis | Open-Source, gratis | Open-Source, gratis |
| kompatibel mit welchen Build-Tools | kann als Task in Gradle ausgeführt werden, Maven Plugin, beliebt auch für Android Apps |  | Maven plugin, Gradle |
| Vergleich 3 | Java Version>=7 | JDK>=1.2 | JDK>=1.7.x, Ant>=1.5 |
| Updates | aktuell, Update im März 2025 | zuletzt 2016 | zuletzt in August 2024 |

***Fazit****:* ***ProGuard*** *ist am besten, da es mit Gradle kompatibel ist und aktiv geupdated wird.*

**Code Conventions**

|  | Oracle | **Google** | Mozilla |
| --- | --- | --- | --- |
| Relevanz | Offizielle Java Code Conventions | große Firma mit viel Erfahrung | nutzen den Oracle Java Code Style |
| Revision | letzte Revision von 1999 | letztes Update 2018, behandeln auch neuere Features wie lambdas & Streams |  |
| Einrückung | Tabs oder Spaces, Tabs exact 8 Spaces lang (nicht 4) | 2 Spaces |  |
| Zeilenlänge | 100 | 80 |  |
| Anmerkungen |  | bei einigen Themen etwas zu nachsichtig |  |

***Fazit****: Die* ***Google*** *Code Conventions sind moderner und Standards anderer Programmiersprachen ähnlich, deswegen werden wir diese verwenden.*

**Kollaborationstool**

|  | Microsoft Teams | **Discord** | NextCloud |
| --- | --- | --- | --- |
| Komplexität | simpel | sehr simpel | medium |
| Funktionen | Anrufe,Chats, Verknüpfung mit OneDrive | vielfältige Servererstellungsmöglichkeiten, Chats, Anrufe | von Hochschule gegebener Account, Dateien, Emails, Kontakte, Kalender |
| Einschränkungen | Microsoft Account benötigt | Account benötigt, max. 10 MB Dateigröße | schwer, einen Einstieg zu finden |

***Fazit****: Microsoft Teams ist zwar gut, hat jedoch den Nachteil, dass es nicht ganz so einfach zu benutzen ist, wie* ***Discord****, welches vielfältige Kommunikationsmöglichkeiten hat. NextCloud hat den großen Vorteil, dass es viele breit gefächerte Funktionen besitzt, jedoch ist Discord einfacher zu benutzen.*