7. APRIL 2022

TABELLARISCHER VERGLEICH VON TOOLS ZUR ERSTELLUNG DER ENTWICKLUNGSUMGEBUNG

Gauger Games Studios GmbH
Vogelsang 33d
01067 Dresden
Tel.-Nr.: 0351-32277691
E-Mail: info@gaugergames.de

Projektleiterin: Felizitas Ebermann

Gauger Games Studios

Einleitung

In dem folgenden Dokument wird evaluiert welche Software zur Bearbeitung des Projektes „Brick-Breaker“ genutzt wird.
Dabei wird zur Bewertung ein System, ähnlich den Schulnoten in Deutschland benutzt, bei dem Zahlen von 1 bis 6 verwendet werden, wobei eine 1 sehr gut, also dem angestrebten Ziel entspricht und eine 6 sehr schlecht wäre.

Kollaborationstool

Kollaborationstool		Discord		Mailinglist		Treffen im Praktikum		Google Docs		WhatsApp	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Lizenzgebühr/ Aufwand	20	1	0,2	1	0,2	3	0,6	1	0,2	1	0,2
Verfügbare Tools	35	2	0,7	5	1,75	3	1,05	3	1,05	4	1,4
Erfahrung	30	1	0,3	4	1,2	2	0,6	2	0,6	3	0,9
Aktualität	15	1	0,15	5	0,75	2	0,3	2	0,3	3	0,45
Gesamt		1,35		3,9		2,55		2,15		2,95	

Für die Kollaborationstools fiel die Entscheidung aufgrund vorheriger gemeinsamer Projekte sehr leicht. Allgemein wurden nur kostenfreie Möglichkeiten zur Auswahl gewählt, da es keine kostenpflichtigen Programme gibt, die sich besonders von kostenfreien hervorheben. Trotzdem stellt das Treffen im Praktikum eine Besonderheit dar, da hiermit ein erhöhter Aufwand verbunden ist. Weiterhin wurden die angebotenen Features verglichen, welche den wichtigsten Teil der Wichtung ausmachen, wobei besonders Discord als Programm mit vielen Möglichkeiten heraussticht, da hier neben Video- und Sprachanrufen mit möglicher Bildschirmübertragung in guter Qualität, auch eine Chatfunktion mit möglichem Dateiaustausch angeboten wird. Besonders hilfreich ist die simple und schnelle Möglichkeit des Austausches mit anderen Projektteams. Eine Mailinglist hingegen bietet nur die Möglichkeit des schriftlichen Austauschs bzw. der Übertragung von Dateien und erschien hierbei als wenig sinnvoll. Das Treffen im Praktikum wird vor allem durch die Möglichkeit der direkten Nachfrage bei Hr. Prof. Dr. Müller attraktiv. Google Docs bietet die Möglichkeit einer schriftlichen Zusammenarbeit an ein und demselben Dokument, weswegen auch hierauf nicht verzichtet wurde. WhatsApp bietet lediglich eine Video- und Sprachanruffunktion sowie einen Chat mit der Möglichkeit des Dateiaustausches. Schon allein da dies bereits durch Discord geboten wird, haben wir uns dagegen entschieden WhatsApp zu nutzen. Erfahrung haben wir in allen Punkten, wobei Teamwork noch nicht so oft über WhatsApp oder Mailinglisten praktiziert haben und uns dies eher fremd ist. Zudem sind Mailinglisten eher eine altmodische Methode um sich untereinander abzustimmen. Und bieten kaum Spielraum für schnelles Abstimmen innerhalb des Teams. Damit bleibt Discord der Spitzenreiter der verglichenen Programme/Möglichkeiten wobei auch auf die zwei besonderen Möglichkeiten bei einem Treffen im Praktikum oder der Zusammenarbeit in Google Docs nicht verzichtet werden konnte und wir uns hiermit für mehrere der möglichen Kollaborationstools entschieden haben.

Obfuscator

Obfuscator		ProGuard		Y-Guard		DashO	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Lizenzgebühr	15	1	0,15	1	0,15	5	0,75
Benutzerfreundlichkeit	20	1	0,2	1	0,2	1	0,2
Erfahrung	25	6	1,5	6	1,5	6	1,5
Support	5	4	0,2	1	0,05	2	0,1
Dokumentation	10	1	0,1	2	0,2	3	0,75
Kompatibilität	25	2	0,5	1	0,25	3	0,75
Gesamt		2,65		2,35		4,05	

Da zuvor noch nie mit Obfuscatorn gearbeitet wurde, gestaltete sich hier die Auswahl durch die mangelnde Erfahrung nicht einfach. Nach erfolgreichem Einarbeiten in die Tools wurden ProGuard, Y-Guard und DashO als relevante Software ausgewählt, da diese die aktuell weit verbreiteten sind und beispielsweise JavaGuard veraltet ist. Als offensichtliche Kriterien wurden zunächst die ggf. anfallenden Kosten, die Nutzbarkeit und die Erfahrung eingeführt. Hierbei hob sich DashO im ersten Punkt negativ von den anderen beiden hervor, da es nicht kostenfrei ist und die Lizenzgebühr nur auf Nachfrage mitgeteilt wird. Allerdings werden verschiedene „Pakete“ angeboten. Die fehlende Erfahrung bezieht sich wie o.g. auf alle der 3 Auswahlmöglichkeiten und auch die Nutzbarkeit ist bei allen 3 Möglichkeiten sehr ausgeprägt und eindeutig und jedes der Tools kann intuitiv genutzt werden. Bzgl. des Supportes wird bei ProGuard von verzögerten Antworten berichtet, während DashO und Y-Guard für eine gute und schnelle Hilfeleistung bekannt sind. Bei Y-Guard sticht zudem der gute Customer-Service heraus. In den letzten beiden Punkten, der Dokumentation und der Kompatibilität mit Java-Versionen, waren für DashO keine Informationen zu finden, da dies vermutlich Paket-abhängig ist, weswegen wir hier einen Mittelwert gewählt haben. In ProGuard wird eine ausführliche englische Dokumentation angelegt und in Y-Guard gibt es immerhin ein Changelog. Außerdem sind sowohl ProGuard als auch Y-Guard mit allen bis dato veröffentlichten Java-Versionen (bis Java 17) kompatibel und somit ideal geeignet. Letztendlich fiel die Wahl auf Y-Guard, da hier die Vorteile gegenüber den anderen beiden Konkurrenten überwogen und es zudem mit dem gewählten Buildtool Ant kompatibel ist und es sich daher anbietet sich hierfür zu entscheiden.

Dokumentationstool

Dokumentationstool		Javadoc		Doxygen	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**
Lizenzgebühr	15	1	0,15	1	0,15
Unterstützte Systeme	5	1	0,05	1	0,05
Abhängigkeit	10	2	0,2	3	0,3
Benutzerfreundlichkeit	25	3	0,75	4	1,0
Support	5	1	0,05	4	0,2
Komfortabilität	20	2	0,4	3	0,6
Erfahrung	15	4	0,6	6	0,9
Gesamt		2,2		3,2	

Besonders wichtig waren uns bei dem Dokumentationstool die Benutzerfreundlichkeit und generelle Komfortabilität. Bei Javadoc hat uns das schnellere und einfache Setup, sowie die allgemeine Leistung mehr überzeugt. Bei Doxygen ist die Installation schon komplizierter und bei großen Projekten kann das Programm langsamer werden. Das war ein großer Minuspunkt, da wir nicht wollen, dass unser Arbeitsprozess von so etwas beeinflusst wird. Ansonsten fanden wir gut, dass man in der IDE einen direkten Zugriff auf Javadoc hat, was den Arbeitsprozess beschleunigen sollte. Des weiteren ist es speziell für Java designed und da wir mit Java arbeiten ist das auch ein positiver Aspekt. Dem entgegen steht das eigenständige Doxygen, bei welchem man aber erst das Programm wechseln müsste um die Dokumentation zu machen. Da beide Programme OpenSource sind gab es bei der Lizenzgebühr keine Unterschiede für uns. Bei den unterstützten Systemen gab es ebenfalls keine Unterschiede, da beide Windows, Linux und MacOS unterstützen. Als Dokumentationstool haben wir uns in Anbetracht der vorliegenden Informationen für Javadoc entschieden. Es hat insgesamt die höchste Wertung erreicht, aber auch in vielen Kategorien mehr als Doxygen.

Testtool

Testtool		QF-Test		TestComplete		TestNG	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Lizenzgebühren	40	6	2,4	6	2,4	1	0,4
Einarbeitung	30	3	0,9	4	1,2	2	0,6
Kompatibilität mit anderen genutzten Programmen	30	2	0,6	2	0,6	2	0,6
Gesamt		3,9		4,2		1,6	

Zwar lassen sich alle dieser Tools als Framework oder Plug-In in Eclipse einbinden, aber im Gegensatz zu den beiden anderen Tools ist TestNG kostenlos. WF-Test kostet mindestens 1.045€, TestComplete sogar mindestens 2,905€. Schlussendlich haben wir uns für TestNG Und da wir von diesen Tools bis jetzt nur mit TestNG gearbeitet haben, wurde entschieden, dieses auch weiterhin zu verwenden.

UML-Tool

UML-Tool		Papyrus		StarUML		Visio	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Lizenzgebühr	15	1	0,15	3	0,45	5	0,75
Unterstützte Systeme	10	1	0,1	4	0,4	4	0,4
Abhängigkeiten	10	2	0,2	1	0,1	5	0,5
Support	5	1	0,05	2	0,1	3	0,15
Komfortabilität	25	2	0,5	2	0,5	3	0,75
Kollaborative Funktionalität	15	6	0,9	3	0,45	1	0,15
Erfahrung	20	4	0,8	6	1,2	6	1,2
Gesamt		2,7		3,2		3,9	

Bei der Auswahl vom UML-Tool fiel zuerst stark auf, wie unterschiedlich viel die Tools kosten. Da der Preis bei Visio als einmalige Zahlung sehr hoch ist wurde es schlechter gewertet. Dagegen ist eine StarUML Lizenz viel günstiger, man kann es auch kostenlos unbegrenzt testen, und Papyrus ist sogar komplett kostenlos. Bei den unterstützten Systemen, unterstützt nur Papyrus alle gängigen. In der Kategorie Support hat man gesehen, dass zwar grundsätzlicher Support bei allen gegeben ist, aber auffällig war, dass Papyrus einen wöchentlichen Update Zyklus hat, was natürlich hoffnungsvoll auf die Behebung möglicher Bugs in der Anwendung blicken lässt. Für die Nutzung der Anwendung braucht man bei Papyrus Eclipse, Visio braucht MS Office und StarUML benötigt nichts weiter. Da die Anschaffung von Eclipse um einiges einfacher und günstiger ist, wurde Papyrus besser bewertet als Visio. Ebenso haben wir beschlossen Eclipse zu nutzen, wodurch das auch kein Problem mehr wäre. Was die Komfortabilität angeht hat jedes Programm gewisse Vorteile. Was bei Visio jedoch aufgefallen ist, ist dass es zwar eine modern gestaltete Oberfläche hat, aber die Inhalte sind veraltet, weshalb es weniger Punkte als der Rest erreicht hat. Der letzte Punkt, die kollaborative Funktionalität, ist in Visio gegeben und für StarUML ist es möglich mit einer Extension. Für Papyrus haben wir leider keine Extension gefunden, aber da wir in diesem Programm schon ein wenig Arbeitserfahrung haben, im Gegensatz zu den anderen, und es uns auch in den anderen Aspekten überzeugt hat, haben wir uns schlussendlich trotzdem für Papyrus entscheiden.

Versionierung

Versionierung		hg		git		svn	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Lizenzgebühr	20	2	0,4	2	0,4	2	0,4
Erfahrung	35	5	1,75	3	1,05	6	2,1
Benutzerfreundlichkeit	35	3	1,05	3	1,05	5	1,75
Support	10	3	0,3	1	0,1	3	0,3
Gesamt		3,5		2,6		4,55	

Da alle drei Softwares frei zugänglich sind wurden ihnen bei der Lizenzgebühr die höchste Punktzahl gegeben.

Die höchste Punktzahl bei der Arbeitserfahrung hat git bekommen, da wir schon mit dieser Software gearbeitet haben.

Hg und git haben 4 Punkte bei der Usability bekommen, da verteilte Versionskontrollsysteme handelt, die über eine Kommandozeile bedient werden kann, aber auch grafische Benutzeroberflächen zur Verfügung stehen. Alle drei Softwares sind übersichtlich und relativ schnell zu erlernen. Svn hat nur 2 Punkte bekommen, Da es sich um ein zentrale Versionsverwaltung handelt und man sich mit zentraler Authentifizierung und Hosting herumschlagen muss, was bei den anderen nicht der Fall ist.

Build-Tool

Build-Tool		ant		maven		gradle	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Lizenzgebühr	20	1	0,2	1	0,2	1	0,2
Erfahrung	35	3	1,05	6	2,1	6	2,1
Benutzerfreundlichkeit	35	2	0,7	3	1,05	3	1,05
Support	10	3	0,3	2	0,2	2	0,2
Gesamt		2,25		3,55		3,55	

Alle drei Softwares besitzen die Apache-Lizenz 2.0, das bedeutet, dass sie alle kostenlos sind und bekommen dementsprechend auch die volle Punktzahl.

Bei der Arbeitserfahrung hat ant vier Punkte bekommen, da wir schon vorher mit dieser Software gearbeitet haben. Maven und gradle haben null Punkte bekommen, da diese Software für uns unbekannt ist.

Bei der Usability

IDE/Editor

IDE		Eclipse		NetBeans		IntelliJ IDEA	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Erfahrung	50	1	0,5	3	1,5	2	1,0
Lizenzgebühren	20	1	0,2	1	0,2	6	1,2
Komfortabilität	10	2	0,2	2	0,2	1	0,1
Debuggen	20	2	0,4	2	0,4	2	0,4
Gesamt		1,3		2,3		2,7	

Bei der Entscheidung der zu Benutzenden IDE liegt das größte Augenmerk auf der bereits vorhandenen oder auch nicht vorhanden Arbeitserfahrung mit dieser, um einen flüssigen und Problemlosen Ablauf der Programmierung des Projektes zu gewährleisten.
Unsere Entwickler sind in Eclipse eingearbeitet und mit der Software vertraut, wohingegen mit IntelliJ IDEA nur ein Mitarbeiter Erfahrungen hat und mit NetBeans in unserem Teamüberhaupt keine Erfahrungen vorliegen.

Lizenzgebühren sind nach Möglichkeit gering zu halten, es liegen keine existierenden Lizenzen vor.
Bei Eclipse, sowie NetBeans handelt es sich um Kostenfreie Programme, wohingegen für IntelliJ IDEA eine Lizenz notwendig ist (489,00€/Jahr je Entwickler).

Komfortabilität**:** Um eine flüssige und angenehme Programmiererfahrung zu bieten, sollte der Code einfach lesbar (Farbcodiert) sein und eine schlaue Auto Vervollständigung besitzen.
Alle drei IDEs besitzen dieses Feature in einem zufriedenstellenden Ausmaß, allerdings hebt sich IntelliJ IDEA positiv ab.

In jedem Programm schleichen sich Fehler und Probleme ein, Ziel ist es diese schnellstmöglich und effizient zu beheben, dafür ist eine gute interne Fehlererkennung sowie Debugging Features von Vorteil.
Alle drei IDEs verfügen über eingebaute Debugger.

Codeconvention

Codeconvention		nach Oracle		nach Google		nach Mozilla	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Erfahrung	50	3	1,5	2	1,0	4	2,0
Support/Dokumentation	30	1	0,3	1	0,3	2	0,6
Aktualität	20	2	0,4	1	0,2	3	0,6
Gesamt		2,2		1,5		3,2	

Wie bei der Auswahl der DIE ist bei den Codeconvention die Arbeitserfahrung das wichtigste Kriterium.
Unsere Entwickler sind in dem Style von Google geübt, für die anderen wäre eine Einarbeitung von Nöten.
Um bei Unklarheiten möglichst schnell weiterzukommen und keine Zeit zu verschwenden ist eine gute Dokumentation des Coding Styles notwendig.
Alle drei Lösungen sind gut dokumentiert, allerdings findet man für Oracle und Google eher Lösungen für spezifische Fragen als für Mozilla.

Die Aktualität des Codingstyles hat Auswirkung auf die Zukunftsfähigkeit und Fortbestand des Programms.
Codeconvention nach Google ist die aktuellste der drei Möglichkeiten und unterstützt die neuesten Features von Java.

Tool für Prototyping der Benutzerschnittstelle

Tool für Prototyping der Benutzerschnittstelle		Figma		Moqups		NinjaMock		Pencil	
Kriterien	**Gewichtung in %**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**	**Bewertung**	**Total**
Lizenzgebühren	15	2	0,3	5	0,75	4	0,6	1	0,15
Erfahrung	20	4	0,8	6	1,2	6	1,2	6	1,2
Workflow	20	1	0,2	2	0,4	2	0,4	4	0,8
Unterstützte Systeme	10	1	0,1	2	0,2	1	0,1	1	0,1
Umfang der Elementbibliothek	15	1	0,15	2	0,3	2	0,3	3	0,45
Kollaborationsmöglichkeit	15	1	0,15	1	0,15	1	0,15	6	0,9
Updates	5	1	0,05	4	0,2	6	0,3	5	0,25
Gesamt		1,75		3,2		3,05		3,85	

Mit Prototyping der Benutzerschnittstelle haben wir bisher wenig Erfahrung. Mit Figma kamen wir durch andere Projekte bereits in Berührung und können daher ungefähr einschätzen wie es funktioniert. Alle der verglichenen Tools bieten zumindest eine kostenfreie Variante an, wobei die Kosten der vollwertigen Version bei Moqups mit 17€/Monat am höchsten liegen. Nur Pencil ist auch in der Vollversion komplett kostenfrei nutzbar. An sich sind alle Tools relativ schnell zu verstehen, wobei Pencil nicht ganz so übersichtlich ist. Dahingegen hebt sich Figma durch viele Shortcuts hervor. Alle der verglichenen Programme sind auf allen gängigen Betriebssystemen nutzbar wobei von Moqups nur eine Webversion verfügbar ist und kein installierbares Programm. Außerdem punktet Figma auch bei dem Umfang der Elementbibliothek und Features die es bietet, da großartige Designfunktionen verfügbar sind. Auch Moqups und NinjaMock bieten viele Funktionen wohingegen Pencil nur einige integrierte Zeichenfunktionen und Form-Sammlungen anbietet. Auch bei der Kollaborationsmöglichkeit sticht Pencil negativ heraus, da es im Gegensatz zu den anderen Programmen für Teamarbeit ungeeignet ist. Die anderen 3 bieten Echtzeit-Zusammenarbeit, Feedback- oder Präsentationsmöglichkeiten. Regelmäßige Updates finden eigentlich nur bei Figma statt, wobei bei Moqups im letzten Jahr immerhin 2 geboten wurden. Bei Pencil wurde sehr lang nichts mehr geupdatet und NinjaMock bietet gar keine Updates mehr an. In Anbetracht dieser Ergebnisse sind wir zu dem Entschluss gekommen Figma zu benutzen. Es hat die beste Wertung in unserer Gegenüberstellung erreicht und uns mit seinen Funktionen überzeugt.

QUELLEN

UML-Tool:
https://www.eclipse.org/papyrus/index.php#page-top
https://staruml.io/
https://www.microsoft.com/de-de/microsoft-365/visio/flowchart-software
https://www.ionos.de/digitalguide/websites/web-entwicklung/die-besten-uml-tools/
https://ijrest.net/vol-1,issue-1/pid-11201404.pdf

Dokumentationstool:
https://www.oracle.com/java/technologies/javase/javadoc-tool.html
https://www.doxygen.nl/index.html
https://www.oracle.com/java/technologies/javase/javadoc-faq.html
https://stackoverflow.com/questions/225447/doxygen-vs-javadoc
https://localcoder.org/doxygen-vs-javadoc

IDE:
https://de.education-wiki.com/9314585-eclipse-vs-intellij
https://www.makeuseof.com/best-java-ide-netbeans-eclipse-intellij-compared/#:~:text=NetBeans%20is%20the%20perfect%20IDE,a%20beginner%20and%20an%20enterprise.

Codeconvention:
https://www-archive.mozilla.org/hacking/mozilla-style-guide
https://codegym.cc/groups/posts/491-java-coding-conventions-which-ones-to-follow-and-why

Testtools:
https://de.wikipedia.org/wiki/QF-Test
https://www.qfs.de/produkt/preise.html
https://de.wikipedia.org/wiki/TestComplete
https://smartbear.com/product/testcomplete/pricing/
https://testng.org/doc/

Tool für Prototyping der Benutzerschnittstelle:
https://www.g2.com/compare/figma-vs-pencil
https://www.g2.com/compare/figma-vs-ninjamock
https://www.getapp.de/software/105959/ninjamock
https://ninjamock.com/
https://chrome.google.com/webstore/detail/ninjamock/hgiegfdifagakbonggdpbnkgallhjadj
https://www.getapp.de/compare/122951/122959/figma/vs/mogups
https://www.getapp.de/reviews/122959/mogups
https://mogups.com/

Versionierung:
https://www.ionos.de/digitalguide/websites/web-entwicklung/git-vs-svn-versionsverwaltung-im-vergleich/
https://entwickler.de/programmierung/git-oder-mercurial-was-sollte-man-nutzen/
https://www.g2.com/compare/git-vs-mercurial

Build-Tool:
https://www.baeldung.com/ant-maven-gradle
https://www.dev-tek.de/lexicon/entry/151-buildautomation-maven-gradle-ant/
https://medium.com/@257ramanrb/ant-vs-maven-vs-gradle-cd8ab4c2735f
https://www.jrebel.com/blog/java-build-tools-comparison

Obfuscator:
https://www.preemptive.com/products/dasho/
https://www.programmersought.net/en/article/324375704.html
https://stackoverflow.com/questions/150653/java-obfuscation-proguard-yguard-other
https://www.yworks.com/products/yguard