ENTWICKLUNG EINES WEB BASED TRAINIGS ZUM THEMA FOTOGRAFIE

A. Saskia Schreiber, B. Thomas Rehm

Technische Hochschule Mittelhessen
Standort Friedberg, Fachbereich IEM

C. Teresa Hoffmann

Justus-Liebig-Universität Gießen
Fachbereich Psychologie

1. EINLEITUNG

Im Rahmen der Veranstaltung *Lehren und Lernen mit neuen Medien* wurde von der Psychologiestudentin Teresa Hoffmann (Justus-Liebig-Universität Giessen, JLU) und den Medieninformatik-Masterstudenten Thomas Rehm und Saskia Schreiber (Technische Hochschule Mittelhessen, THM) ein Web Based Training (WBT) zum Thema Fotografie entwickelt.

Das primäre Ziel des Trainings sollte die Vermittlung grundlegender Kenntnisse über die Funktionalität einer Kamera sein. Darauf aufbauend sollte der Trainee Anregungen über die kreative Nutzung dieser Kenntnisse erhalten. Aus dieser Zielsetzung ergab sich folgende thematische Unterteilung:

- Kameratypen und ihre Unterschiede
- Technische Grundlagen Schulung
- Kreative Aspekte

2. PROJEKTORGANISATION

Neben der inhaltlichen und technischen Umsetzung des WBT stand die interdisziplinäre Zusammenarbeit zwischen Medieninformatik- und Psychologiestudierenden im Vordergrund. Um eine gute und möglichst effektive Realisierung des WBT zu gewährleisten, wurde von Anfang an des Projektes im verteilten Team gearbeitet. Die Arbeit wurde durch den Einsatz von Tools für die kollaborative Zusammenarbeit wesentlich unterstützt. Dazu sind die Google© Apps Documents, SpreadSheets & Drive sowie die kollaborative Code-Versionsverwaltung GitHub verwendet worden. Die Tools wurden nicht nur für die Zusammenarbeit, sondern auch für die inhaltliche und technische Bereitstellung des WBT genutzt (siehe Abschnitt Einleitung). Die Kommunikation erfolgte per Instant Messenger.

In den gemeinsamen Treffen der ersten Projektphase wurden die Projektdetails festgelegt, um einen gemeinsamen Leitplan für das Projekt zu haben. Dieser beinhaltete das inhaltliche sowie strukturelle Konzept, an dem sich das Team weitestgehend bis zur Fertigstellung orientierte.

In der zweiten Phase wurde parallel an unterschiedlichen Aufgaben gearbeitet. Texte und Grafiken wurden anhand des Leitplans erstellt, die technische und gestalterische Umsetzung des WBT wurde geplant und mit der Umsetzung in HTML, CSS und JS begonnen.

In der dritten und abschließenden Phase des Projekts wurden Inhalt und Technik zusammengeführt, Inhalte geprüft und erweitert, technische Feinheiten ergänzt und Fehler behoben.

3. ASPEKTE DER UMSETZUNG

Im Folgenden wird tiefer auf die psychologischen, inhaltlichen und technischen Aspekte eingegangen, die bei der Umsetzung des WBT von Bedeutung waren.

3.1. Psychologische & inhaltliche Aspekte

Bei der Konzeption des Trainings wurden verschiedene wissenschaftliche Erkenntnisse aus lern- und medienpsychologischer Forschung berücksichtigt.

Inhaltlich wurde in den Abschnitten *Kameratypen* und *Technische Grundlagen* zunächst deklaratives Grundlagenwissen vermittelt und in den anschließenden Tests abgefragt. Auf diesem Wissen baute die Lektion *Kreative Aspekte* auf. Damit Lernende mit unterschiedlichem Vorwissen von dem Training profitieren können, waren alle Hauptthemenbereiche von Anfang an zugänglich und nicht nacheinander freigeschaltet. Auf diese Weise werden (fortgeschrittene) Lernende nicht zu sehr in ihren Interessen eingeschränkt und explorativ es Lernen ermöglicht.

Die Tests wurden gezielt nicht nur nach Abschluss der Hauptkapitel platziert, sondern als obligatorische Zwischenprüfungen nach einzelnen Themenblöcken eingesetzt. Dadurch erhält der Lernende direkt ein Feedback zu seinem Wissensstand und kann gegebenenfalls Inhalte erneut lesen, wenn er Teile nicht verstanden hat.

Da jedoch das Ziel war, nicht nur deklaratives Wissen, sondern die richtige und kreative Bedienung einer Kamera zu vermitteln, wurde ein Kamerasimulator integriert. Der Kamerasimulator ermöglicht es zum Beispiel, änderungen an den Blendeneinstellungen live am Bild zu sehen. über einen Button konnte so jederzeit das Gelernte ausprobiert und dadurch in prozedurales Wissen transferiert werden.

Bei der Erstellung der Inhalte wurde auf die gute Verständlichkeit der Texte geachtet. Als Orientierung diente dabei das "Hamburger Verständlichkeitskonzept" von Langer, Schulz von Thun und Tausch (2006)[1], laut dem sich verständliche Texte durch vier Merkmale auszeichnen: Einfachheit, Gliederung und Ordnung, Kürze und Prägnanz und anregende Zusätze. Weiterhin wurde eine informelle Sprache und persönliche Anrede benutzt, was nach dem Personalisierungsprinzip der kognitiven Theorie multimedialen Lernens (Cognitive Theory of Multimedia Learning von Richard E. Mayer)[1] das Lernen erleichtert. Zur besseren Verständlichkeit wurden ebenfalls viele Bilder eingefügt, die die Inhalte verdeutlichen. Um dem negativen Effekt der geteilten Aufmerksamkeit (Split-Attention Effect)[1] entgegenzuwirken, wurden die Bilder immer direkt beim dazugehörigen Text platziert.

Die Inhalte des Trainings wurden nach aktuellen Erkenntnissen der Lerntheorie konzipiert, um sowohl Novizen als auch fortgeschrittenen Benutzern einen schnellen Lernerfolg zu ermöglichen. Ein Aspekt dieses Konzeptes war die Entscheidung, die Hauptthemen nicht nacheinander freizuschalten, sondern als getrennte Bereiche zu behandeln, deren Unterthemen aufeinander aufbauen. Auf diese Weise wird der (fortgeschrittene) Lernende nicht zu sehr in seinen Interessen eingeschränkt. Weiterhin wurde für die Texte auf informelle Sprache und persönliche Anrede geachtet, was nach der kognitiven Theorie das Lernen erleichtert.

Ein besonders wichtiges Feature ist der integrierte Kamerasimulator, der jederzeit über einen Button zu erreichen ist. Da es sich bei den technischen Themen teils um recht trockene Theorie handelt, muss es eine Möglichkeit geben, die gelernten Informationen in der Praxis zu erleben. Der Kamerasimulator ermöglicht es zum Beispiel, änderungen an den Blendeneinstellungen live am Bild zu sehen.

3.2. Technische Aspekte

Die technische Grundlage des WBT bildet das von Studenten der Technischen Hochschule Mittelhessen entwickelte WBT-Framework.

Um dieses Framework sinnvoll um eigene Komponenten erweitern zu können, wurde sich für eine Einbindung als Git-Submodul[2, S. 341 ff.] entschieden, sodass das Hauptrepository allein aus den neu erstellen Inhalten besteht. Das hat den Vorteil, dass Updates und Bugfixes am Framework ohne Konflikte in das Projekt integriert werden können. Auch mit den in diesem Kapitel erläuterten Erweiterungen wurde so verfahren, sofern es sich um Fremdcode handelt.

Um die Veröffentlichung des WBT so direkt und unkompliziert für die Entwickler zu machen, wurde sich für die Nutzung des frei verfügbaren GitHub Pages[1] Services entschieden, der sich optimal in den bestehenden Git-Workflow einfügt. GitHub hostet mit GitHub Pages kostenlos statische Webanwendungen, die Code-Projekte auf GitHub mit einer einfachen Projektpage versorgen. Pages integriert sich insofern nahtlos in den Workflow mit Git und GitHub da zur Veröffentlichung einer GitHub Pages Webseite lediglich der Branch "gh-pages" angelegt werden muss. GitHub crawlt jedes Repository und erstellt dann die Webseite aus dem gh-pages-Branch. Durch pushen in diesen Branch wird die neue Version im Hintergrund gebaut. GitHub Pages ist kompatibel mit Git Submodulen prüft beim Build-Prozess ob alles gelingt. Treten Fehler beim Build auf, wird der Entwickler benachrichtigt.

Da das Framework aus dem DOM[2] der HTML-Seite die eigentliche Webanwendung (Sections, Article, Navigation, Fragemodule etc.) beim Laden generiert, müssen alle Inhalte zum Zeitpunkt des Initialisierungsprozesses des Frameworks im DOM vorhanden sein. Ein solches HTML-Dokument kann schnell einige hundert Zeilen lang werden und ist damit sehr unangenehm für einen Redakteur mit Inhalten zu befüllen und zu pflegen.

Das Team entschloss sich die verwendenten Tools zu verwenden, um eine elegantere Lösung zu schaffen: Der sog. ContentLoader vereint mehrere Techniken, um die Arbeit an den Inhalten zu vereinfachen. Die Funktionsweise sieht folgenden Ablauf vor:

- Redakteur schreibt Inhalte und Struktur in einem veröffentlichten Google Spreadsheet[3]
- Das Spreadsheet sieht folgende Spalten vor: SectionHeadline, ArticleHeadline, ContentType, Content
- Sections und Articles können definiert werden
- Verlinkung von Bildern aus einem Google Drive Ordner erfolgt über Namensnennung des Bildes + Dateiendung
- Redakteur hat zusätzlich mehrere ContentTypes zur Verfügung: Listen (neutral, positiv, negativ), Subheadlines, Bilder (aus Drive-Ordner oder URL)

Der ContentLoader ist ein JavaScript Script[4], das per AJAX die Inhalte aus dem Spreadsheet im JSON Format[5] lädt und dann in den DOM einfügt. Das Script liest die Inhaltselemente linear ein und verarbeitet diese je nach ContentType.

[1] https://help.github.com/articles/what-are-github-pages/

[2] https://developer.mozilla.org/en-US/docs/Glossary/DOM

[3] https://docs.google.com/spreadsheets/d/1U7imA8NCahaqeIT3jlz-3J-mTuRZyrYs1rUlTZsnO_M/edit?usp=sharing

[4] https://github.com/thomasrehm/wbt-fotografie/blob/master/res/js/script.js#L40-L99

[5] https://spreadsheets.google.com/feeds/list/1U7imA8NCahaqeIT3jlz-3J-mTuRZyrYs1rUlTZsnO_M/od6/public/values?alt=json-in-script

Um den ContentLoader realisieren zu können wurde der Initialisierungsaufruf des Frameworks vom an den GlobalEventHandler `window.load`[6] gebunden. Dieses Event wird erst ausgelöst, wenn alle Elemente des DOM fertig aufgebaut sind. Bisher wurde das Framework mit dem Event `document.ready`[7] initialisiert, was vorausgesetzt hat, das die Inhalte komplett im HTML Dokument vorhanden sind.

Wie bereits in den inhaltlichen Aspekten erwähnt, war es bei der Wahl des Themas Fotografie sinnvoll, eine interaktive Möglichkeit zum Ausprobieren der gelernten Einstellungen zu schaffen. Dafür wurde die JavaScript-Applikation bethecamera[8] verwendet, ein interaktiver HTML5-Kamerasimulator, der Änderungen an Kameraeinstellungen in Echtzeit an verschiedenen Beispielbildern demonstriert.

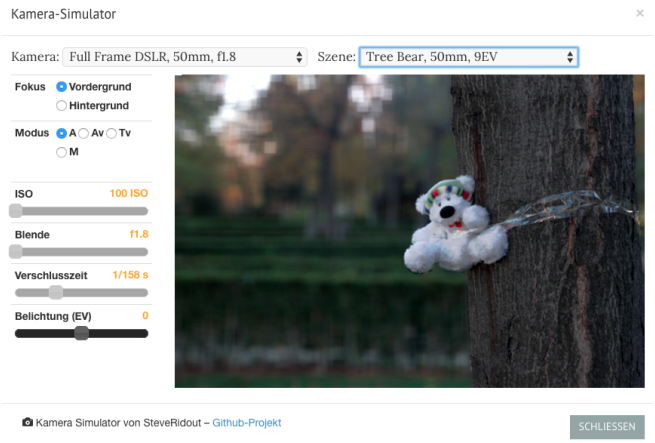

Fig. 1. Screenshot Kamerasimulator.

Der Simulator verwendet zum Rendern der Bilder das HTML5-Canvas-Element. Die verwendeten Bilder müssen vorher entsprechend aufbereitet werden: Das Bild sollte idealerweise als HDRI (High Dynamic Range Image) vorliegen, damit Änderungen der Helligkeit im Simulator die gewünschten Effekte zeigen können.

Zusätzlich muss der Vordergrund des Bildes ausgeschnitten und als PNG mit Alphakanal abgespeichert werden, damit Blureffekte der Blende sich auch wirklich nur auf den Fokuspunkt (Vorder- oder Hintergrund) auswirken.

Da die im Simulator mitgelieferten Bilder sehr gut funktionieren, wurde auf die Ergänzung weiterer Bilder verzichtet. Die aktuelle Umsetzung ermöglicht aber eine Weiterentwicklung in dieser Hinsicht.

Integriert wurde der Simulator so, dass er bei Bedarf per AJAX in ein Bootstrap-Modal[9] geladen wird. Das erhöht die initiale Ladezeit der Seite und ermöglicht außerdem, dass der Nutzer das aktuell aufgerufene Thema nicht verlassen muss. Gerade für das Ausprobieren zwischendurch ist dieses Kriterium essenziell.

Zuguterletzt wurde für eine optimale Präsentation des Themas ein eigenes Stylesheet angelegt, welches auf dem WBT-Stylesheet aufbaut und dieses teilweise überschreibt. Dadurch wird ein gewisser Wiedererkennungswert geschaffen, zumal das Standard-Theme von Bootstrap mittlerweile sehr häufig im Web zum Einsatz kommt.

[6] `https://developer.mozilla.org/en-US/docs/Web/API/GlobalEventHandlers/onload`

[7] `https://developer.mozilla.org/en-US/docs/Web/API/Document/readyState`

[8] `http://bethecamera.com/`

[9] `http://getbootstrap.com/javascript/#modals`

4. REFLEKTION

4.1. Schwierigkeiten & Herausforderungen

4.2. Aufgabenverteilung & Wissenstransfer

4.3. Lessons Learned

4.4. Kritische Beurteilung des Autorentools/WBT Frameworks

5. REFERENZEN

[1] Günter Rey, *E-Learning Theorien, Gestaltungsempfehlungen und Forschung*, Huber, Bern, 2009.

[2] Scott Chacon, *Pro Git*, Apress,Distributed to the book trade worldwide by Spring Science+Business Media, Berkeley, CA New York, NY, 2014.