|  |  |
| --- | --- |
| **VON** | Belegarbeit im Fach „Entwicklung von Multimediasystemen“ an der HTW Berlin |
|  |  |
| Programmierung einer gestengesteuerten Bildergalerie |  |
|  | *Eingereicht bei Sebastian Bauer und Sebastian Keppler* |
|  | *Vorgelegt von Delia Metzler und Christopher Heiden* |
|  | Berlin, 18. Juni 2016 |

**Inhaltsverzeichnis**

1. Einleitung **1**
2. Planung des Projektes **1**
3. Gestellte Anforderungen an des Projektes **1**
4. Aufbau des Projektes **1**

**4.1** WindowManager **1**

**4.2** Begruessung **1**

**4.2.1** Sound **1**

**4.2.2** Vorder- und Hintergrund **1**

**4.3** Hauptinterface **1**

**4.3.1** Optionsleiste **1**

**4.3.1.1** Vollbild**1**

**4.3.1.2** Hintergrund**1**

**4.3.1.3** BilderGroesse**1**

**4.3.1.3** Sprache ändern**1**

**4.3.1.4** Filtern**1**

**4.3.1.5** Hilfe**1**

**4.3.2** Infoleiste**1**

**4.3.3** Bilder**1**

**4.3.3.1** Suche**1**

**4.3.3.2** QStringZuQImage**1**

**4.3.3.3** Anzeigen**1**

**4.3.3.4** MyLabel**1**

**4.4** Diashow**1**

**4.4.1** Querformat**1**

**4.4.2** Hochformat**1**

**4.5** Datenbank **1**

**5.** Anforderungen **1**

**5.1** Gelöste Anforderungen 5

**5.2** Ungelöste Anforderungen 6

**6.** Tests 6

**7.** Vergleich mit Zielsetzung 6

**8.** Fazit 6

**9.** Selbstständigkeitserklärung **4**

**10.** Quellen **4**

I. Anhang **4**

Anhang: Bilder 6

Anhang: Anforderungen 6

Anhang: Quelltext 6

1. Einleitung

In einer Zeit, in der Virtual-Reality (Oculus Rift) real ist und Augmented-Reality (HoloLens) in den Startlöchern für die Programmierer steht, darf man sich nicht nur mit der Programmierung von Spielen oder anderer Arten von Software am PC auseinandersetzen. Der Trend entfernt sich immer weiter von den PCs und geht zu VR oder AR. Aus diesem Grund ist es wichtig sich Wissen über andere Fachgebiete der Informatik zu erschließen, wie zum Beispiel die Programmierung der Kinect von Microsoft, welche einen guten Einstieg in die Welt der Virtualität bietet.

Eine Möglichkeit sein Wissen zu erweitern und mal einen anderen Weg der Programmierung einzuschlagen, bietet der Kurs „Entwicklung von Multimediasysteme“. In diesem hatte man die Aufgabe mithilfe des Qt Frameworks eines vom Dozenten vorgeschlagenes Projekt zu realisieren oder ein eigenes Projekt zu bearbeiten. Dazu konnte man eine Gruppe aus maximal 3 Personen bilden, wobei sich der Umfang, der zu erbringen ist, pro Person erhöht. Für die Bearbeitung eines eigenständigen Projektes hatte man insgesamt 14 Wochen Zeit.

Für die Entwicklung eines Projektes bildete sich eine Gruppe aus zwei Personen. Diese setzt sich aus Delia Metzler (s0…) und Christopher Heiden (s0549453) zusammen. Wir entschieden uns für ein eigenes Projekt. Um Mal was Anderes zu programmieren, was man nicht nur mit der Maus steuern kann, sondern auch auf eine andere Art und Weise, entschied man sich die Kinect v2 in das Projekt zu integrieren.

Das Projekt, welches wir realisieren wollte, ist eine Gestengesteuerte Bildergalerie. Eine Bildergalerie ist viel mehr als nur das darstellen von Bilder, welche der Nutzer mit einer Kamera geschossen hat. Es ist eine Erinnerung an eine Zeit in der man glücklich und fröhlich war und man sich gerne zurückerinnert. Es ist Wunsch und Hoffnung zugleich, neue Eindrücke zu sammeln und zu erfahren. Der Titel des Projekts, welcher „VON“ lautet, bezieht sich auf dieses Gefühl, den ist dies ein Isländischer Begriff, der übersetzt „Hoffnung“ bedeutet.

1. Planung des Projektes

Um dieses Projekt zu realisieren galt es als erstes Ideen zu entwickeln, wie die Bildergalerie am Ende aussehen könnte und welche Anforderungen die Applikation am Ende haben soll. Dafür skizzierte man zu Beginn bei OneNote (siehe Anhang Bild 1), um sich designerisch an das Projekt anzutasten. Währenddessen entstanden bereits die ersten Anforderungen, wie zum Beispiel, dass man die Hintergrundfarbe ändern oder nach bestimmten Tags filtern kann. Nachdem man die ersten Anforderungen festhielt, wurde ein Papierprototyp entwickelt, welcher das Hauptinterface mit seinen Interaktionsmöglichkeiten und der Darstellung der Bilder, die von PC des Nutzers stammen (siehe Anhang: Bild 2-3). Durch die Prototypen hat man eine klarere Sicht auf das Projekt und den Anforderungen an das Projekt. Diese Art des Prototypens hat den Vorteil, dass man sich frühzeitig damit beschäftigt, welche Funktionalitäten mit in die Applikation kommen sollen und was passieren soll, wenn der Nutzer bestimmte Buttons bedient. Bei der Entwicklung dessen entstanden weitere Anforderungen, die mit in die Liste aufgenommen wurden.

1. Gestellte Anforderungen des Projektes

Nachdem die designspezifischen Eigenschaften beschlossen waren, war es an der Zeit, die Anforderungen zu spezifizieren. Dazu wurde eine Liste aufgestellt, die sämtliche Eigenschaften enthält (siehe Tab. 1).

|  |  |
| --- | --- |
| Beim Starten des Programms wird ein Willkommenston abgespielt | 1 |
| Beim Doppelklick auf Bild, startet ein Diashowmodus | 5 |
| Hintergrundgraphik beim ersten Fenster | 1 |
| Der Benutzer soll Bilder per Dateiauswahl ins Programm laden können | 2 |
| Es besteht die Möglichkeit die Hintergrundfarbe zu ändern | 1 |
| Der Benutzer kann die Anzahl der in einem Fenster dargestellten Bilder ändern. | 1 |
| Nach dem Beenden des Programms wird beim nächsten Neustart gefragt, ob der zuletzt geöffnete Ordner wieder geöffnet werden soll | 1 |
| Es gibt die Menüpunkte Hilfe und Optionen | 3 |
| Erstellen eines Desktop-Icons | 1 |
| Der Benutzer kann den Bildern Noten von 1 bis 5 geben | 2 |
| Im Programm ist eine Datenbank integriert, um IDs, Tags, Pfad und Bewertungen abzuspeichern | 4 |
| Der Benutzer kann die Bilder mit Tags versehen | 2 |
| Der Benutzer kann nach Tags und nach Noten filtern. | 2 |
| Die Bildinformationen werden beim Anklicken in der Informationsleiste angezeigt | 3 |
| Schnittstelle mit Kinect | 5 |
| Der Benutzer soll den Großteil des Programms mit Gesten steuern können. | 9 |
| Auf dem Rechner wird nach jpg\_Bildern gesucht | 2 |

Tab. 1 zeigt die Anforderungen, die zu Beginn des Projektes spezifiziert wurden.

Durch die offene Aufgabenstellung, die der Dozent im Kurs „Entwicklung von Multimediasystemen“ stellt, bietet der Kurs die Möglichkeit eines Einblickes in die Programmierung abseits des Computers.

Einer der wenigen Anforderungen, die der Dozent an alle Teams hatte, war das nutzen des Qt Frameworks. Wir beschlossen die Möglichkeit zu ergreifen und entschieden uns für die Integrierung der Kinect in das Projekt. Dafür nutzt man die Kinect v2, mit welcher wir die Gestensteuerung realisieren wollen.

Das ist einer der Hauptaspekte des Projektes, weshalb diese Anforderung am höchsten gewichtet ist. Diese 9 Punkte errechnet sich durch die Unterteilung der Aufgabe in Funktionen (siehe Tab. 2), die durch Gesten der Hand/Hände realisiert werden sollen.

|  |  |  |
| --- | --- | --- |
| Geste | Funktion | Punkte |
| Pushen des Bildes | Doppelklick auf das Bild | 1 |
| Hand nach links/rechts bewegen | Bild davor oder danach öffnen und anzeigen | 2 |
| Hand in Kreis drehen | Bild wird um 45° gedreht | 1 |
| Daumen und Zeigefinger bewegen sich voneinander oder nähern sich an | Vergrößerung bzw. Verkleinerung des Bildes | 2 |
| Bildbewertung/ Vollbildmodus beenden | Über Sterne wischen | 2 |
| Gestenmodus aktivieren | Aktivierung der Gesten | 1 |

Tab. 2 zeigt die Verteilung der Punkte zu den jeweiligen Gesten.

Die Anforderungen der Gruppe zum Projekt ist nur ein Teil der Aufgabenstellung, welche der Dozent an das Projekt hat. Die gesamte Liste der Anforderungen befindet sich im Anhang (Anhang: Anforderungen).

1. Aufbau des Projektes

Für die Ausarbeitung des Projektes wurde Visual Paradigm 12.2 verwendet, mit dessen Hilfe man ein UML Diagramm skizzierte (siehe Bild … ). Diese theoretische Struktur wurde versucht in dem Vorhaben zu implementieren.

Grob lässt sich sagen, dass das Projekt aus 5 Oberklassen zusammensetzt, dem WindowManger, der Begruessung, dem Hauptinterface, der Diashow und der Datenbank. Diese wiederum haben Unterklassen mit denen Sie wiederum verbunden sind, um damit beispielsweise die Bilder im Hauptinterface oder die Optionsleiste darzustellen.

In den nachfolgenden Unterpunkten wird der Aufbau des Projektes näher erläutert und dargestellt.

* 1. WindowManager

Die WindowManager-Klasse dient dazu zwischen den Einzelnen Layouts, die angezeigt werden sollen zu wechseln, wenn der Nutzer bestimmte Buttons oder andere Aktivitäten betätigt. Dafür wird das SIGNAL-SLOTS-Prinzip von Qt verwendet. Wenn einer der definierten Signale ausgesendet wird, wird dieses verwertet indem ein neues Layout gesetzt wird.   
Zudem wird in dieser Klasse das Icon, welches sich im Ressourcenordner befindet gesetzt, sowie der Name der Applikation.

* 1. Begruessung

Zu Beginn der Interaktion mit dem Nutzer wird diesem ein Begrüßungsinterface angezeigt, welches auf eine Aktion des Nutzers wartet in Form eines drücken eines der Buttons. Dem Nutzer wird beim ersten Öffnen des Programmes nur ein Button präsentiert. Erst beim zweiten Öffnen, wenn die Datenbank bereits Bilder enthält, des Programms, wird ihm ein zweiter Button zu Verfügung gestellt. (siehe Bild 2 und 3).

|  |  |
| --- | --- |
|  | Begruessung 2 Buttons |
| Bild 2 zeigt das Begruessunginterface, wenn sich in der Datenbank noch kein Bild enthalten ist. | Bild 3 zeigt das Begruessunginterface, wenn sich in der Datenbank bereits Bilder enthalten sind. |

* + 1. Sound

Bei jedem Ausführen des Programms wird ein Willkommenston abgespielt. Das Lied, welches abgespielt wird, befindet sich im Ressourcenordner. Da allerdings der Ressourcenordner nicht dazu geeignet ist Sounds zu beinhalten, wird ein neuer Ordner erzeugt, der das Lied beinhalten. Dieser Ordner wird aufgerufen um den Sound abzuspielen.

* + 1. Vorder- und Hintergrund

Der Hintergrund setzt sich aus einer QGraphicScene und einer QGraphicView zusammen, die 800 x 600 Pixel groß ist. Dieser wiederum beinhaltenen Linien unterschiedlicher Farben.

Auf dem Hintergrund befinden sich Bottons und das Label, welche in der Klasse Vordergrund definiert sind. Dort werden auch die Stille der Buttons und des Labels gesetzt, sowie deren Signale, die beim Klicken aktiviert werden. Zudem gibt es noch einen Button, der die Sprache ändert und einen Hilfebutton, die ebenfalls in der Vordergrundklasse definiert worden sind.

* 1. Hauptinterface

Nach der Begrüßung und das Betätigen eines der Buttons gelangt der Nutzer der Software zum Hauptinterface (siehe Bild 4). In diesem werden sämtliche vom Nutzer vorher bestimmten Bilder in klein dargestellt und kann Änderungen am Interface und der Galerie vornehmen, wie zum Beispiel die Farbe des Hintergrundes ändern oder nach bestimmten Bildern filtern.

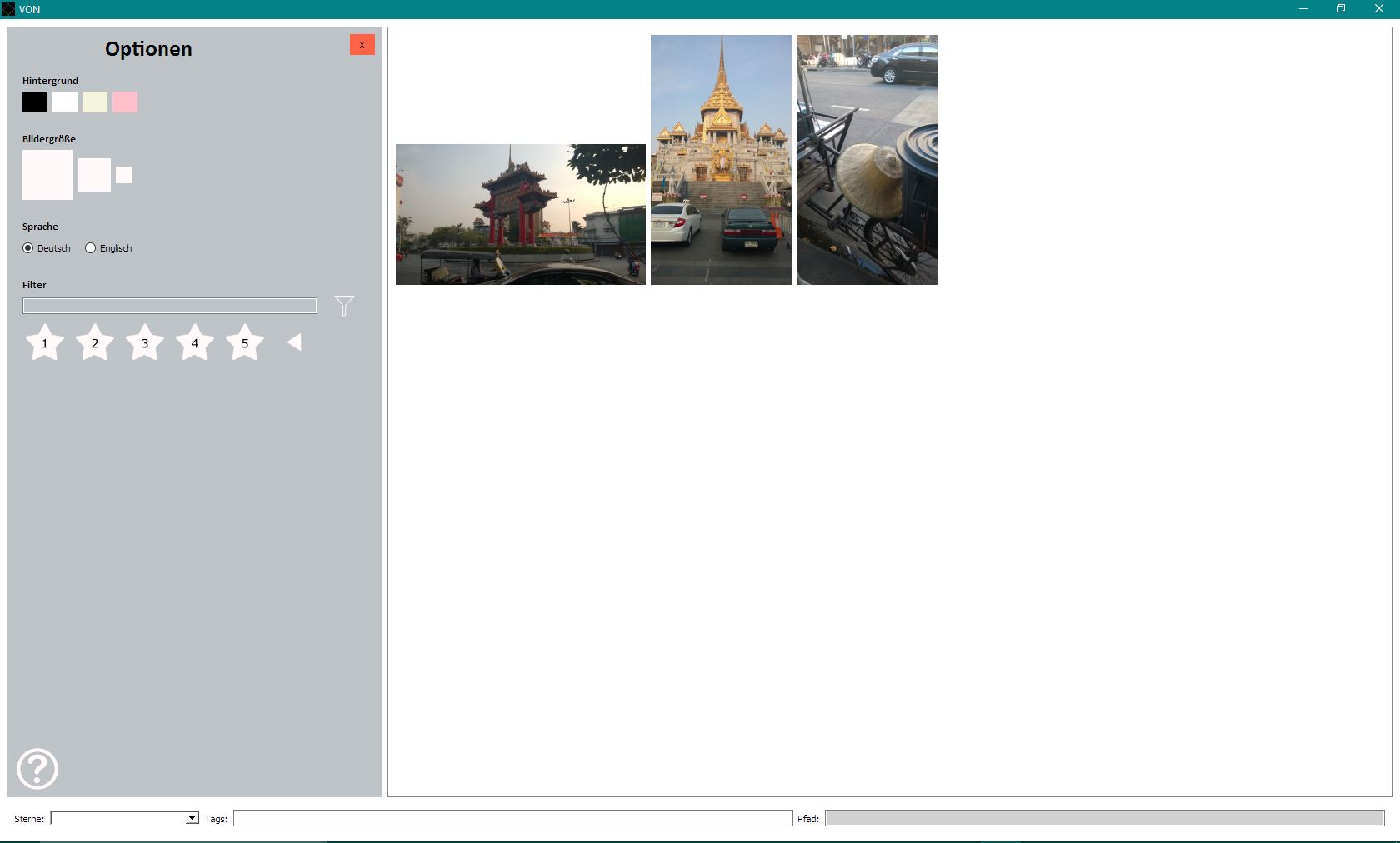


Bild 4 zeigt ein Hauptinterface, welches sich zusammensetzt aus einer Optionsleiste, einer Informationsliste und der Bildergalerien, die in dem Beispiel Beispielhaft Bilder zeigt.

Hauptaugenmerk des Interfaces ist die Bildergalerie und die Informationsleiste, sie sich im rechts neben der Optionsleiste bzw. darunter befindet. Das Interface wird im Gegensatz zum Begruessungsinterface diesmal maximal angezeigt. In den nachfolgenden Kapiteln werden die Options-, die Informationsleisten, die Bildergalerie, sowie der Diashow genauer erläutert.

* + 1. Optionsleiste

Die meisten Möglichkeiten, um mit der Software zu interagieren, ermöglicht die Optionsleiste (siehe Bild 5). Diese bietet dem Nutzer die Möglichkeit, diese zu entfernen, die Farbe des Hintergrundes und die Größe der Bilder, sowie die Sprache zu ändern. Um sich bei Fragen zu behelfen, bietet man einen Hilfebutton an.

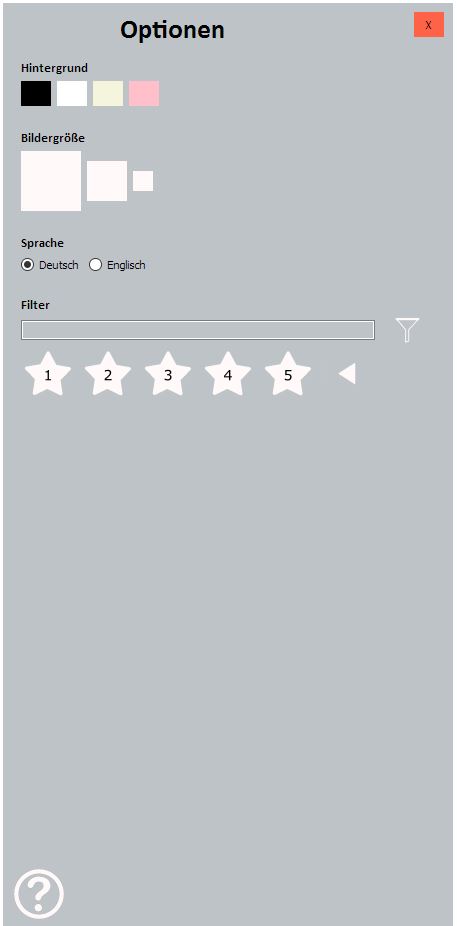


Bild 5 zeigt, Optionsleiste der Software mit der der User mit der Applikation interagieren kann.

* + - 1. Vollbild

Der Vollbildmodus wird aktiviert, wenn man auf den roten x-Button (siehe Bild 8) drückt, der sich neben dem Label „Optionen“ befindet. Dieser bewirkt, dass die Optionsleiste verschwindet, sodass man eine größere Fläche für die Bildergalerie hat und das die Bilder größer dargestellt werden. Um die Optionsleiste wieder sichtbar zu machen muss man den Button, der sich nun in der Informationsleiste befindet, drücken.

C:\Users\Christopher\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Vollbildmodus aktivieren.jpg

Bild 8 zeigt den Button zur Aktivierung des Vollbildmodus.

Die Informationsleiste existiert trotz dessen, sodass der Nutzer Bilder bewerten und Tags setzen kann. Zudem befindet sich ein nun ein Button in der Leiste (siehe Bild 9), die den Modus beendet und die Optionsleiste wieder erzeugen kann.

C:\Users\Christopher\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Informationsleiste + Vollbild deaktivieren.jpg

Bild 9 zeigt die Informationsleiste, wenn der Vollbildmodus aktiviert wurde

* + - 1. Hintergrund

Die Hintergrundfarbe des Hauptinterfaces kann geändert werden. Der Nutzer kann sich zwischen Schwarz, Weiß, Beige und Pink zu entscheiden (siehe Bild 10). Bei der Betätigung einer der Buttons wird die Hintergrundfarbe geändert. Zudem ändert sich auch die Farbe der Texte und der Labels, sodass man diese in jedem Modus lesen kann.

Hintergundfarben einstellen

Bild 10 zeigt die Farben, in der die Hintergrundfarbe der Software verändert werden kann.

* + - 1. BilderGroesse

Auch die Bildergröße kann der Nutzer einstellen. Dafür existieren drei Buttons (siehe Bild 11), die bei der Betätigung, die Bilder, die dargestellt werde, vergrößert bzw. verkleinert. Standard ist, dass 20 Bilder dargestellt werden, ohne das der Nutzer scrollen muss. Dies wurde so implementiert, dass die Bildergröße fest definiert ist, je nachdem, ob dieser 20, 40 oder 60 Bilder auf einmal sehen möchte.

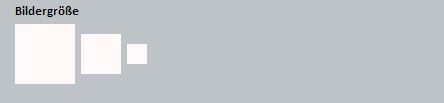


Bild 11 zeigt wie der Nutzer der Software die Größe der Bilder anhand eines Klickens auf eine der Buttons beeinflussen kann.

* + - 1. Sprache ändern

Auch die Sprache der Applikation kann geändert werden. Zur Auswahl stehen dem Nutzer zwei Sprachen zur Verfügung (Deutsch und Englisch). Da das Team davon ausgegangen ist, dass die Applikation meist nur für Personen, die als Muttersprache Deutsch gelernt haben, ist der Standard auf die deutsche Sprache gesetzt. Damit man die Sprache ändern kann, muss der Nutzer der Software auf den Radiobutton „Englisch“ (siehe Bild 12) aktiviert werden. Mit diesem Klick werden alle Labels und Text auf Englisch übersetzt.



Bild 12 zeigt die Möglichkeit die Sprach der Applikation zu ändern, indem man auf einend er beiden Radiobuttons drückt.

* + - 1. Filtern

Das Filtern ermöglicht dem Nutzer nach Tags oder nach Bewertungen, welche zuvor in der Informationsleiste für Bilder gesetzt worden sind zu suchen (siehe Kapitel 4.3.2 Informationsleiste), zu filtern (siehe Bild 13). Innerhalb des Textfeldes kann man die Tags angeben, nach denen man filtern möchte. Wenn es mehrere Tags sind muss man die Wörter mit einem Leerzeichen trennen. Wenn man den Tag gesetzt hat, kann man entweder die Entertaste drücken oder aber auf das „Filtern Symbol“ rechts neben dem Textfeld klicken. In der Datenbank wir dann nach Bildern gesucht, die diese Tags enthält (siehe Kapitel Datenbank4.5). Die gefundenen Bilder werden in einem Vector geschrieben, mit dessen Hilfe man die Bilder darstellen kann (siehe Kapitel4.3.3.2 QStringZuQImage und 5.3.3.3 Anzeigen).

Um hingegen nach Bewertungen zu filtern, muss man auf eine der Sterne drücken. Dann wird nach einer bestimmten Nummer, die man gedrückt hat, gefiltert. Dabei wird in der Datenbank (siehe Kapitel Datenbank 4.5) nach Bildern gesucht, die diese Bewertung besitzen. Diese werden in ein Vector geschrieben und dargestellt.



Bild 13 zeigt die Möglichkeit, die der Nutzer hat um nach Bildern zu filtern.

* + - 1. Hilfe

Der Hilfebutton dient zur Unterstützung bei unverständlichen Symbolen oder Labeln, die für die Interaktion der Software mit dem Nutzer diesen sollen. Zudem werden Funktionen dem Nutzer erklärt die beim ersten Nutzen nicht auffindbar sind (Hidden Affordance), wie das Einfach und Doppelklicken auf ein Bild (siehe Kapitel 4.3.3.4 MyLabel) oder dem Enterdrücken beim Filtern (siehe Kapitel 4.3.1.4 Filtern).

Beim Anklicken des Buttons (siehe Bild 14) öffnet sich ein weiteres Fenster welches das alle Erklärungen bietet.

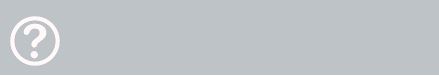


Bild 14 zeigt den Button der bei Fragen gedrückt werden soll.

* + 1. Informationsleiste

Unterhalb er Optionsleiste und der Bildergalerie befindet sich die Informationsleiste (siehe Bild 15), die Informationen über ausgewählte Bilder anzeigen soll. Zudem kann man dort die Bilder bewerten und Tags setzen, nach denen man in der Optionsleiste filtern kann (siehe Kapitel 4.3.1.2 Filter). Darüber hinaus wird der Pfad angezeigt, welchen man nicht verändern kann. Aus diesem Grund ist das Textfeld auch ausgegraut, um dies dem Nutzer anzudeuten. Die Funktionen stehen in enger Zusammenarbeit mit der Datenbank (siehe Kapitel …. ….).



Bild 15 zeigt die Informationsleiste, in der die Bewertung, die Tags und der Pfad des Bildes angezeigt werden, welches angeklickt wurde.

* + 1. Bilder

In diesem Kapitel wird darauf eingegangen, wie nach den Bildern gesucht, diese umgewandelt, anzeigt und wie man das drücken bzw. doppelklicken auf die Bilder realisiert hat, um die Informationen des jeweiligen Bildes auf der Informationsleiste anzugeben.

* + - 1. Suche

Die Suche beginnt mit dem Drücken einer des eine bzw. einer der Buttons aus dem Begruessungsinterface.

Wenn der Nutzer den Button mit der Aufschrift „Startverzeichnis“ drückt, öffnet sich die Ordnerstruktur des Nutzers (siehe Bild 16). Mit dessen Hilfe, kann sich der Nutzer für ein Startverzeichnis von dem aus das Programm nach Bildern suchen soll. Dabei ist zu bemerken, dass das nur nach Bildern gesucht werden kann, die vom Datentyp .jpg sind. Wenn sich der Nutzer für ein Verzeichnis entschieden hat, wird nach Bilder, die solch eine Endung haben iterativ gesucht. Bilder, die solch eine Endung besitzen werden als Strings (Pfad des Bilds) in einem Vector gespeichert mit dem im Anschluss weitergearbeitet wird.

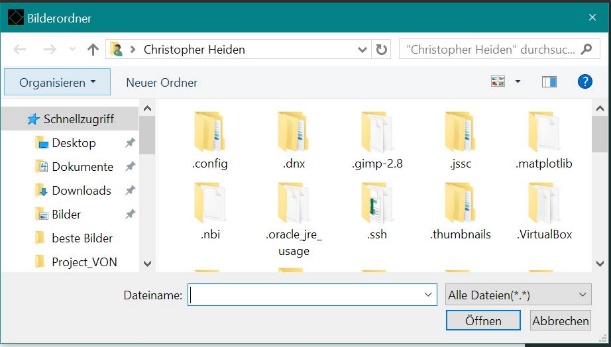


Bild 16 zeigt eine beispielhafte Ordnerstruktur.

Wenn der Nutzer den anderen Button mit der Aufschrift „Letzte Bilder anzeigen“ im Begruessungsinterface betätigt, wird nach Bildern (deren Pfade) in der Datenbank gesucht, die sich der Nutzer bei letzter Nutzung der Applikation angesehen hat. Diese werden ebenfalls in einem Vector gespeichert.

* + - 1. QStringZuQImage

Die Pfade der gefundenen Bilder, welche man in dem Vector gespeichert hat, werden nun zu QImages umgewandelt. Dieses ist sehr Zeitaufwändig, weshalb man hier auf Threading setzt, um die Bearbeitungszeit zu verringern. Bei der Umwandlung spielt auch die Anzahl der Bilder, die ohne scrollen angezeigt werden sollen eine Rolle. Umso mehr Bilder angezeigt werden sollen, umso kleiner werden die Bilder angezeigt. Die angaben, wie groß die Bilder angezeigt werden sollen ist hart kodiert. Die QImages werden wieder in einem Vector gespeichert.

* + - 1. Anzeigen

Die Bilder, die sich in einer Map befindet, werden durchgegangen durch eine Iterator. Dabei wird jedes QImages einem vom Autor selbst erweiterten Klasse „MyLabel“ zugewiesen und mit Signals versehen. Daraufhin wird jedem Bild ein QPixmap gesetzt.

Wie schon im Kapitel 4.3.3.2 QStringsZuQImages spielt auch hier die Bildergröße eine Rolle. Wenn man 20 Bilder ohne scrollen anzeigen möchte, werden 5 Bilder in einer Reihe angezeigt (siehe Bild 17). Bei 40 bzw. 60 Bilde, die auf einmal sichtbar sein sollen, werden 10 bzw. 15 Bilder nebeneinander angezeigt.

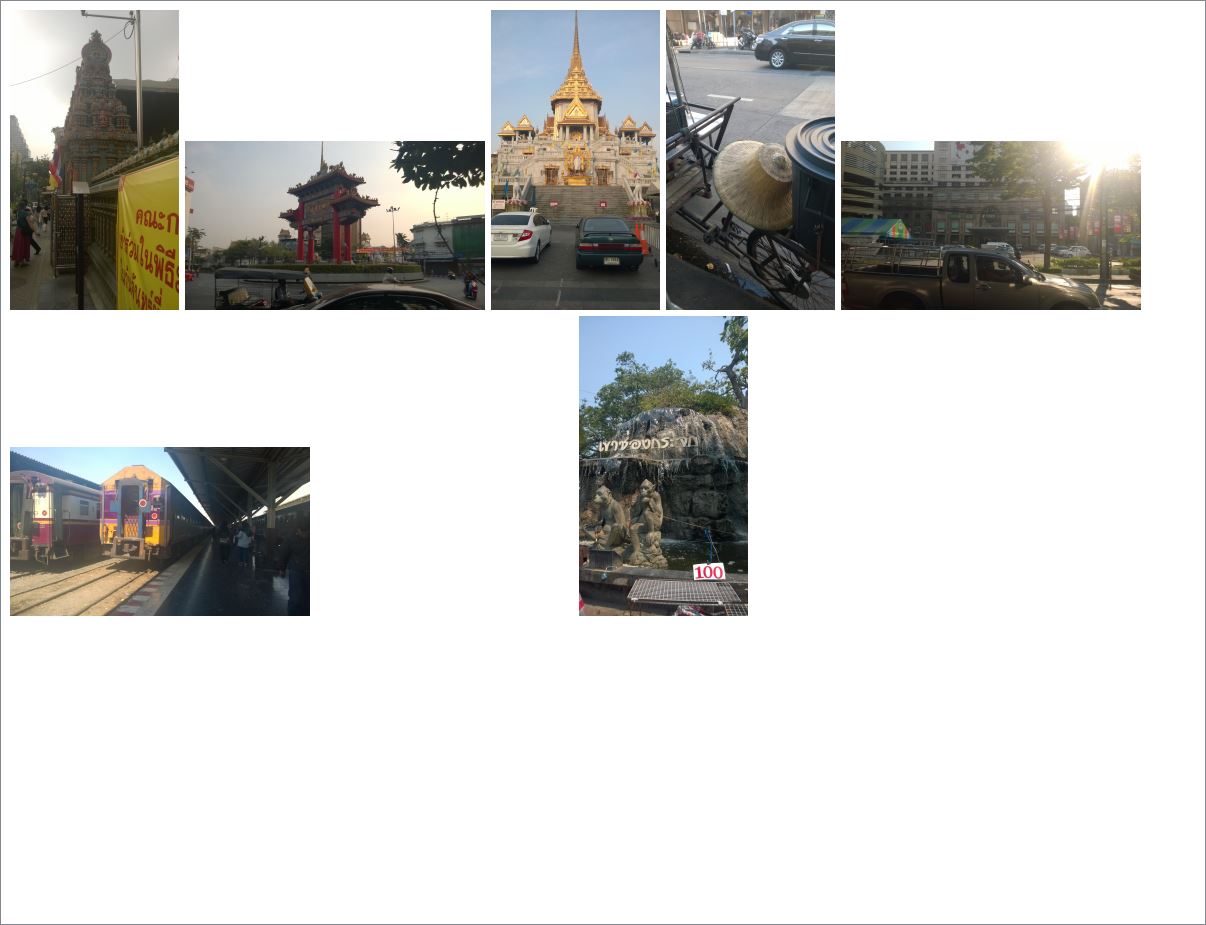


Bild 17 zeigt ein Beispiel für das Anzeigend er Bilder im Hauptinterface.

* + - 1. MyLabel

Die Klasse MyLabel erbt von der Klasse QLabel. Somit hat MyLabel alle Funktionen, die auch die Klasse Label besitzt und noch weitere, die man ergänzen kann. Funktionen, um die man die Klasse Label ergänzen wollte, war die Möglichkeit ein Bild anzuklicken. Mit dieser neuen Möglichkeit realisiert man die Status von der Informationsleiste, denn durch das anklicken eines Bildes, werden dessen Pfad, Bewertung und die Tags in der Informationsleiste angezeigt (siehe Kapitel 4.3.2 Informationsleiste).

Zudem konnte man dadurch das Doppelklicken auf ein Bild realisieren. Durch das doppelklicken auf ein Bild wird der Diashowmodus (siehe Kapitel 4.4 Diashowmodus) geöffnet in der das Bild in Großformat angezeigt wird.

* 1. Diashow

Der Diashowmodus ermöglich den Nutzer das Betrachten seiner Bilder in Groß. Dabei spielt es eine Rolle, ob die Bilder im Hoch- oder Querformat sind. Je nachdem ob die Höhe bzw. die breite größer sind, wird ein anderes Layout erzeugt (siehe Bild 18 und 19).

|  |  |
| --- | --- |
| C:\Users\Christopher\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Querformat.jpg | Unbenannt |
| Bild 18 zeigt das Layout, wenn das Bild im Querformat geschossen wurde. | Bild 19 zeigt, das Layout, wenn das Bild im Hochformat geschossen wurde. |

Durch die Datenbank, die eine Attribut hat welches speichert, ob ein Bilder in der Session angeschaut wird, kann man diese Funktionalität nutzen, um die vom Nutzer angezeigten Bilder in einem Vektor zu speichern. Beim bedienend einer der Pfleilbuttons, wir das nächste bzw. das vorherigen Bild dargestellt.

Eine andere Möglichkeit, die man durch die Verknüpfung der Datenbank erreicht, ist das auch hier das bewerten der Bilder möglich ist.

Man hat auch die Möglichkeit die Bilder zu drehen. Durch das drücken des Knopfes wird das jeweilige Bild, welches gerade angezeigt wird, nach rechts um 45 Grad gedreht. Die Änderungen werden in der Datenbank mit aufgenommen.

Wenn man aus diesem Modus wieder zu seiner Bildergalerie gehen möchte drückt man auf den Button mit dem Fluchtsymbol.

Die Funktionalitäten sind auch mit der Kinect nutzbar. Um dies den Nutzer kenntlich zu machen, wurde ein Symbol innerhalb der Optionsleiste implementiert , der, wenn man auf ihn drückt, man ein neues Fenster angezeigt bekommt, in dem der Nutzer auf die Möglichkeiten der Nutzer mit der Kinect hingewiesen wird.

* 1. Datenbank

1. Anforderungen

Für die Bearbeitung der Projekte innerhalb des Kurses „Entwicklung von Multimendiasystemen“ gab es vom Dozenten und vom Projektteam Anforderungen, die es zu erfüllen galt. Bei diesen stellte man bei einigen fest, dass manche nicht zu erfüllen sind. In den nachfolgenden Kapiteln geht es um die gelösten Anforderung und was es bei einigen zu lösen galt und die ungelösten Anforderungen mit einer Erklärung, weshalb man diese nicht wie gewünscht lösen konnte.

* 1. Gelöste Anforderungen

Einer der zu Beginn schwierigsten Probleme, die es zu lösen galt, war das wechseln auf ein neues Layout um beispielsweise vom Begruessungslayout zu dem Hauptinterface zu gelangen. Um dieses Problem zu lösen nutzt man nun eine Art Manager, der die Layouts bei bestimmten Signalen ändern. Dafür nutzt man die Funktion setCurrentWidget.

Ein anderes Problem entstand beim Abspielen des Sounds im Begruessungsinterface. Damit alles in einem Ordner gespeichert wird, entschied man sich den Sound ebenfalls als eine Ressource, wie auch schon das Icon, zu setzen. Bei Versuch den Sound abzuspielen, wurde festgestellt, dass der Ressourcenorder nicht geeignet dafür ist Sound abzuspielen. Deshalb entschied man sich einen versteckten Ordner zu erzeugen in dem sich der Sound befinden und abgespielte wird. Dies funktionierte, auch wenn es sehr lange gedauert hat bist ein Ton zu hören war. Aus diesem Grund entschied man sich mit Threads zu arbeiten in der Hoffnung, dass es schneller ginge. Der Versuch ging schief und es dauerte immer noch zu lange. Schlussendlich hat man sich dafür entschieden den Sound zu kürzen, so dass es anstelle von 3 Minuten, nur noch sieben Sekunden sind, die es zu laden gilt.

Eines der größten Probleme bei der Abarbeitung der Anforderungen entstand bei der Darstellung der Bilder. Das Problem was, dass zu Beginn anstelle von 98 Bilden nur an die 40 dargestellt wurden. Dies lag wahrscheinlich an der Größe der Bilder, die im MegaByte- Bereich liegen. Aus diesem Grund musste man die Bilder verkleinern. Um dies zu realisieren, durfte man nicht, wie man es zu Beginn getan hat, mit Buttons arbeiten, auf denen man die Bilder als Icons implementiert hat, sondern musste die Bilder umwandeln von QString zu QImages. Durch diese Umwandlung war man in der Lage, alle Bilder verkleinert darzustellen.

Doch durch die Lösung dieses Problems entstand schon ein Neues. Die Umwandlung von QString zu QImages dauerte zu lange. Deshalb entschied man sich für die Verwendung von Threads. Doch auch nachdem man eine Therad implementierte, dauert es immer noch zu lange große Mengen von Bildern in das Programm zu laden.Deshalb entschied man sich für die Implementierung von „BLOB“, um die Bilder als QImages in der Datenbank zu speichern und nicht als QStrings, welche man wieder umwandeln muss zu QImages.

Ein anderes Problem entstand als man die den Button implementiert hat, mit dessen Hilfe man vom Diashowmodus wider in das Hauptinterface zu gelangen versuchte. Jedes Mal, wenn man auf den Button gedrückt hat und man zurück zur Bildergalerie kehrte, entstanden innerhalb der Optionsleiste weitere Buttons, welche oberhalb der Leiste dargestellt wurden (siehe Bild 20). Das Problem war, dass neue Connections deklariert wurden. Um dies zu umgehen, wurden die Connections von Signals und Slots in innerhalb des Konstruktor geschrieben. Dasselbe Problem entstand im Diashowmodus, wenn man sich das nächste Bild ansehen wollte. Dort trat es allerdings so auf, dass man anstelle des nächsten Bilder, man das übernächste Bild dargestellt bekommen hat. Diese Art von Fehler potenzierte sich bei jedem Mal, wenn man das nächste oder das vorherige Bild sich ansehen wollte.

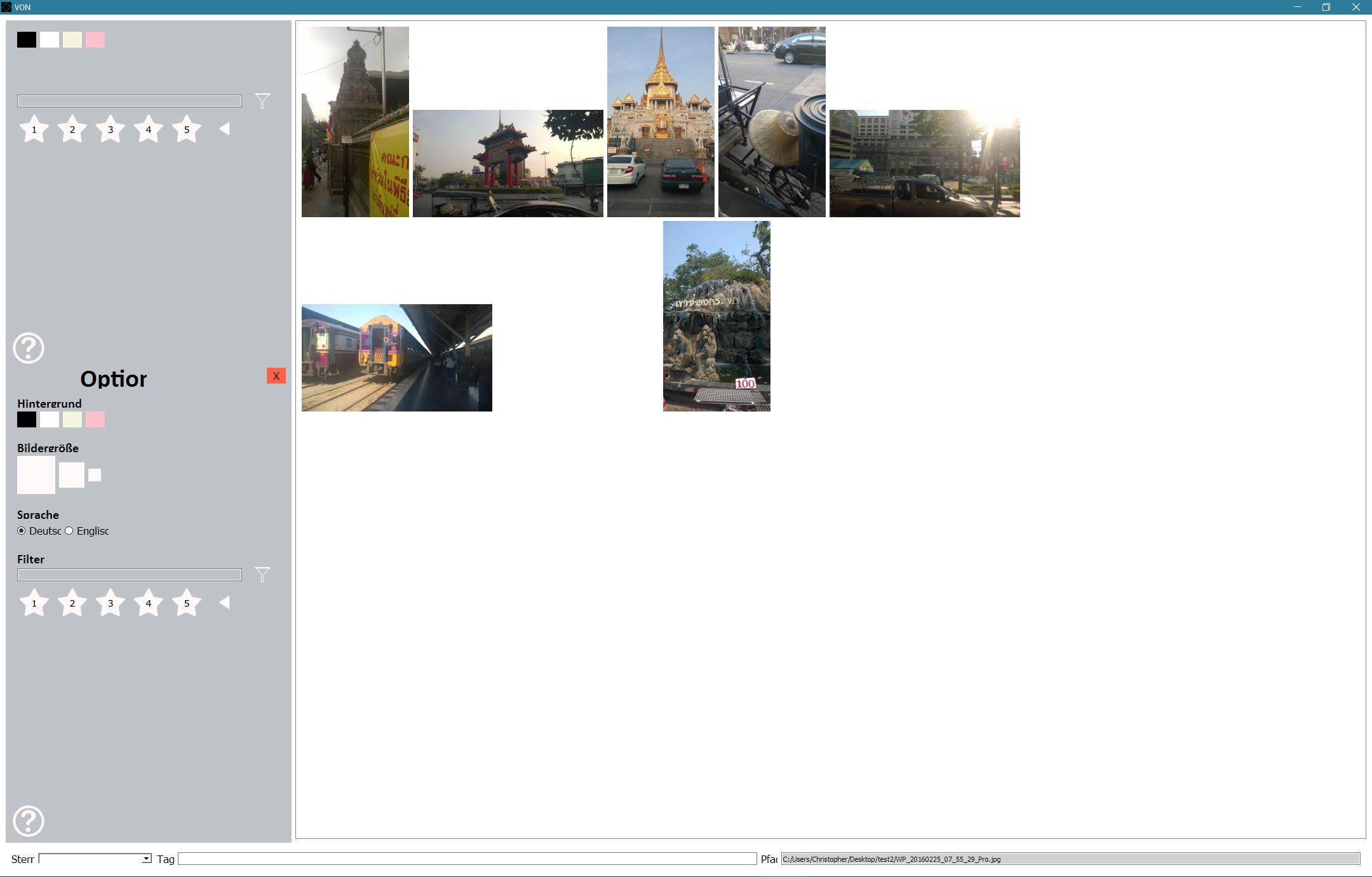


Bild 20 zeigt das Problem beim Zurückkehren vom Diashowmodus zur Bildergalerie.

* 1. Ungelöste Anforderungen

Eines der vom Dozenten spezifizierten Anforderungen an die Applikation, war es, dass man diese auch am Smartphone testet. Schließlich sagen die Entwickler von QT, dass man mit diesem Framework Gerätunabhängig programmieren kann und das entstandene Projekt leicht auch auf andere Geräte zu bringen ist, bzw. nur wenige Anpassungen benötigt. Eine Anforderung war es dies zu testen. Aufgrund der Problematik, das Smartphones einen anderen Aufbau und Struktur besitzen als die Betriebssysteme für PCs, ist ein Zugriff auf ein Bildverzeichnis anderes zu bewältigen als bei den klassischen Betriebssystemen. Somit ist das Laden von Bildern nicht möglich. Zudem sind auch das Speichern und der Zugriff auf diese anderes zu realisieren. Aus diesem Grund ist die Verwendung des Projektes auf dem Smartphone nicht realisierbar.

Mit den Dockern schafft man es Programme, die auf Windowsrechnern laufen, auch auf Linux Rechnern zu importieren. Auch dies sollte man als eine der Anforderungen umsetzten. Dieses ist in dem Fall nicht möglich, da man die Kinect v2 mit in das Projekt integrierte hat und man dieses mit dem Kinect SDK von Microsoft nutzt. Da die nötigen Treiber nicht installiert werden können auf Linux Rechnern, hat man davon abgesehen, dies zu realisieren.

1. Tests

Mithilfe von Tests soll überprüft und bewiesen werden, ob die Funktionen, welche man programmiert hat, funktionieren. Dies tat man für …. Was und wie man getestet hat, wird in den nachfolgenden Paragraphen beschrieben.

1. Fazit

Eine solch große Aufgabe wurde zum ersten Mal innerhalb des Studiums gestellt. Allerdings ist dies gut, da man dadurch erfasst, wie schnell ein Projekt wachsen kann und wie schwer es ist die letzten 20-30% der Anforderungen zu bearbeiten und wie falsch man mit der Zeit Einplanung sein kann. Auch wenn man nicht alles geschafft hat, was es sich an Anforderungen gestellt hat, hat es der Kurs geschafft, dass man sich nicht nur mit der Materie „Programmierung“ beschäftigt hat, sondern auch mit Softwareengineering, Kontrollstrukturen und Projektmanagement.

1. Selbstständigkeitserklärung

Hiermit erklären wir, dass wir die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst und keine anderen Hilfsmittel als angegeben verwendet haben. Insbesondere versichern wir, dass wir alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken als solche kenntlich gemacht haben.

Ort: Berlin



Datum: 18.07.20014 Unterschriften:

1. Anhang

8.1 Anhang: Bilder

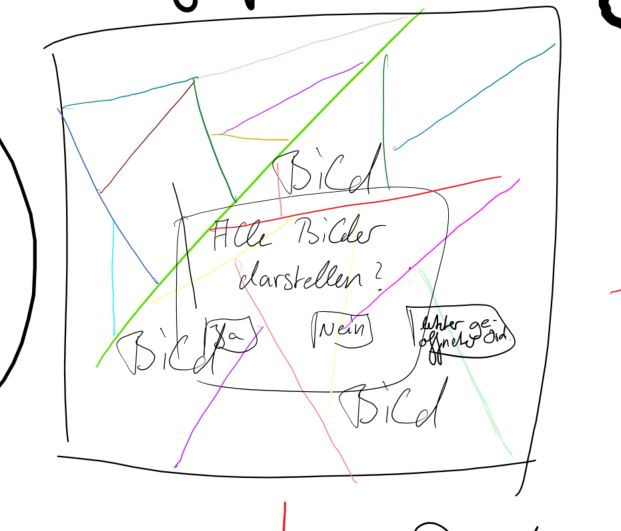


Bild 1 zeigt die Entstehung des Projektes. Für Skizzierung nutze man OneNote. Diese Skizze zeigt den ersten Startbildschirm, welcher beim Öffnen des Programms angezeigt werden soll.

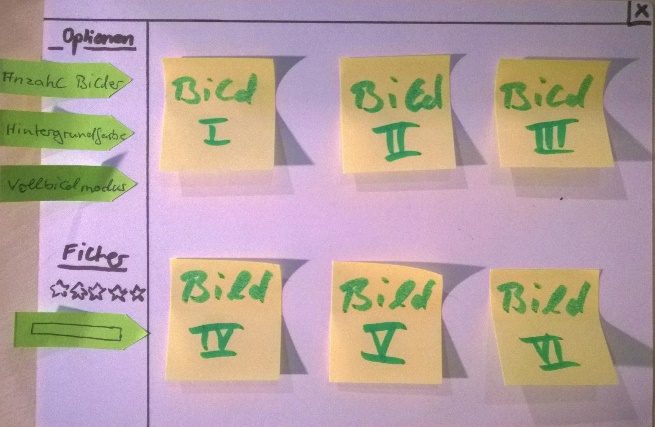


Bild 2 zeigt den Papierprototypen der Bildergalerie und bereits die meistern Funktionen die diese im späteren Verlauf des Projektes implementiert habe soll.

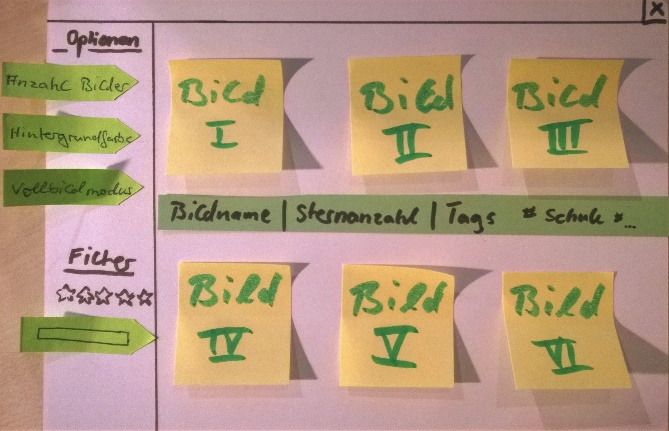


Bild 3 zeigt den Papierprototypen der Bildergalerie und bereits die meistern Funktionen die diese im späteren Verlauf des Projektes implementiert habe soll, sowie, was geschehen soll, wenn der Nutzer auf eines der Bilder geklickt hat.

5.2. Anhang: Anforderungen

|  |  |
| --- | --- |
| Funktion | Punkte |
| Programm soll eine QT-Anwendung mit einem Hauptfenster | 1 |
| Die Größe des Fensters kann mit den üblichen Methoden verändert werden, das GUI passt sich der Größenveränderung automatisch an. | 1 |
| Es gibt es einen Vollbild-Modus | 1 |
| Die Anwendung ist lokalisiert und das Benutzerinterface ist in mind. zwei Sprachen (z.B. englisch und deutsch) verfügbar. | 4 |
| Die Applikation läuft auf einem Desktop-Rechner, wird aber auch auf einer mobilen Plattform getestet und wenn nötig angepasst. Das Ergebnis ist in der Dokumentation festzuhalten | 4 |
| Die Anwendung reagiert weiterhin auf Benutzereingaben, wenn längere Berechnungen anstehen. Beispielsweise soll es möglich sein, eine länger dauernde Berechnung abzubrechen. | 4 |
| Es wird ein Versionsverwaltungssystem wie git oder svn benutzt. Die Commit-  Kommentare beschreiben genau, was gemacht wurde. Build-Artefakte beenden sich  nicht im Repository. | 4 |
| Klassen, Funktionen und Schnittstellen sind ausführlich im Quelltext im Doxygen oder qdoc-Format dokumentiert. Die Doxygen- bzw. qdoc-basierende Dokumentation lässt sich einfach mittels make generieren. | 4 |
| Verwendete Algorithmen und Datenstrukturen sind möglichst Qt-unabhängig zu entwickeln, um eine anderweitige Nutzung zu gewährleisten. | 4 |
| Fehlerfreies Funktionieren der implementierten Algorithmen und Datenstrukturen ist mit Unit-Tests zu belegen und deren Erfolg ist im Beleg zu dokumentieren. | 5 |
| Werkzeuge wie valgrind und AddressSanitizer werfen bei der Ausführung des Programms und der Unit-Tests keine Fehlermeldungen aus. Der Erfolg ist im Beleg zu dokumentieren. | 4 |
| Das Programm lässt sich einfach für eine Linux-Plattform übersetzen und die Tests laufen dabei automatisch ab. Build-Artefakte werden sowohl für Linux und für eine weitere Plattform wie Windows oder OS X erzeugt und optional für eine mobilen Plattform. Das Bauen der Artefakte wird über einer Containerlösung wie Docker realisiert. | 4 |
| Beim Starten des Programms wird ein Willkommenston abgespielt | 1 |
| Beim Doppelklick auf Bild, startet ein Diashowmodus | 5 |
| Hintergrundgraphik beim ersten Fenster | 1 |
| Der Benutzer soll Bilder per Dateiauswahl ins Programm laden können | 2 |
| Es besteht die Möglichkeit die Hintergrundfarbe zu ändern | 1 |
| Der Benutzer kann die Anzahl der in einem Fenster dargestellten Bilder ändern. | 1 |
| Nach dem Beenden des Programms wird beim nächsten Neustart gefragt, ob der zuletzt geöffnete Ordner wieder geöffnet werden soll | 1 |
| Es gibt die Menüpunkte Hilfe und Optionen | 3 |
| Erstellen eines Desktop-Icons | 1 |
| Der Benutzer kann den Bildern Noten von 1 bis 5 geben | 2 |
| Im Programm ist eine Datenbank integriert, um IDs, Tags, Pfad und Bewertungen abzuspeichern | 4 |
| Der Benutzer kann die Bilder mit Tags versehen | 2 |
| Der Benutzer kann nach Tags und nach Noten filtern. | 2 |
| Die Bildinformationen werden beim Anklicken in der Informationsleiste angezeigt | 3 |
| Schnittstelle mit Kinect | 5 |
| Der Benutzer soll den Großteil des Programms mit Gesten steuern können. | 9 |
| Auf dem Rechner wird nach jpg\_Bildern gesucht | 2 |

|  |  |  |
| --- | --- | --- |
| Geste | Funktion | Punkte |
| Pushen des Bildes | Doppelklick auf das Bild | 1 |
| Hand nach links/rechts bewegen | Bild davor oder danach öffnen und anzeigen | 2 |
| Hand in Kreis drehen | Bild wird um 45° gedreht | 1 |
| Daumen und Zeigefinger bewegen sich voneinander oder nähern sich an | Vergrößerung bzw. Verkleinerung des Bildes | 2 |
| Bildbewertung/ Vollbildmodus beenden | Über Sterne wischen | 2 |
| Gestenmodus aktivieren | Aktivierung der Gesten | 1 |