Globetrotter – Dokumentation

Gümüs Ali

Mayer Stephan

Moosbrugger Simon

Spiegel Philipp

Inhaltsverzeichnis

# 1. Thema

## 1.1. Neuausrichtung nach ÖkoSwim

### 1.1.1. Warum nicht mehr ÖkoSwim

- Daten von OÖ teilweise veraltet

- sporadisch erfasst

- nicht vollständig

- Datenformate nicht immer gut maschinenlesbar

### 1.1.2. Analyse anderer Open Data Datenquellen

|  |  |
| --- | --- |
| Land Vorarlberg | - geringe Auswahl an Datensätzen  - Datensätze enthalten nicht viel Informationen  - alle Formate fast nicht maschinenlesbar |
| Stadt Wien | + oft gut maschinenlesbar  + Daten werden aktualisiert  - viele Applikationen bereits umgesetzt |
| Schweiz | + viele Daten  + Aktualität OK  - Datenformat mehrmals nur schwer maschinenlesbar |
| Weltbank | + viele verschiedene Datensätze  + gute Aktualität  + REST Webservice zur Datenabfrage  - manchmal fehlen erfasste Länder oder Jahre |

## 1.2. Globetrotter

+ Daten der Weltbank

+ Welt auf dem Globus erkunden können

+ grundlegendes geographisches Wissen aufbauen

+ tiefere Details durch Datensätze der Weltbank

+ Vergleich von Ländern

+ Einsatz im E-Learning denkbar

# 2. Technologien

## 2.1. Applikation

Für die Applikation wird 3D benötigt. Folgende Technologien beziehungsweise Game Engines stehen unter anderem zur Auswahl:

* WebGL
* Blender
* Unity3D

### 2.1.1. HTML 5 mit WebGL

Die Recommendations von HTML 5 und WebGL stehen frei und plattformunabhängig zur Verfügung. Da sie aber noch relativ neu sind, können mehrere Schwierigkeiten auftreten. Nicht alle Webbrowser unterstützen HTML und WebGL im nötigen Ausmaß (Microsoft Internet Explorer kann kein WebGL; Apple Safari muss WebGL zuerst umständlich aktivieren). Zudem muss eine aktuellere Grafikkarte mit aktuellem Grafiktreiber am System installiert sein. Somit würden viele potentielle User die Applikation nicht nutzen können. Die Teammitglieder haben kaum Erfahrung mit diesen Technologien. Zudem scheint die Performance nach eigenen Tests noch nicht ganz ausgereift zu sein.

### 2.1.2. Blender

Blender steht für verschiedene Plattformen (Linux, Mac, Windows) frei zur Verfügung. Es enthält neben der 3D-Modellierung und –Animation die sogenannte Blender Game Engine. Für die Programmierung stellt die Blender Game Engine eine Python API zur Verfügung. Da wir im Team aber keine Erfahrungen mit der Blender Game Engine und Python haben, haben wir und gegen die Blender Game Engine entschieden.

### 2.1.3. Unity3D

Neben der kostenpflichtigen Version Unity3D Pro gibt es die kostenfreie Version Unity3D Free. Seit der Programmversion 4.x ist für Unity3D Free lediglich ein kostenfreier Account beim Hersteller nötig. Mit Mac und Windows werden auch die zwei großen Plattformen unterstützt. Bei der Programmierung in Unity3D kann zwischen Boo, C# und JavaScript gewählt werden. Da Unity3D eine gute Verbreitung in der einschlägigen Branche aber auch darüber hinaus hat bietet es sich an. Außerdem ist allen Teammitglieder die Programmiersprache C# gut bekannt. Der Umfang der Dokumentation und eine aktive Community sorgen für viele Hilfestellungen. Leider fehlen die wichtigen Teile der .NET API zur Erzeugung und Bearbeitung der Grafik. Deshalb müssen die Diagramme per Komposition und Manipulation von 3D-Objekten erstellt werden. Dennoch habe wir und entschlossen, dass Unity3D für die Applikation zum Einsatz kommen soll.

## 2.2. Steuerung

Die Steuerung der Applikation soll nicht direkt erfolgen. Stattdessen ist geplant, dass der User die Applikation über ein eigenes Gerät steuern kann. Neben Smartphones und Tablets könnten auch Bewegungs- und Tiefensensoren sowie Kameras zum Einsatz kommen. Dabei soll auf der Steuerungsseite kein Art der Visualisierung geben.

### 2.2.1. Android Smartphone

Das mobile Betriebssystem Android läuft auf einer immensen Anzahl an Smartphones weltweit. Die nötigen Entwicklertools für Android stellt Google für die verschiedensten Desktopplattformen kostenlos zur Verfügung. Da auch eine gute Kenntnis über die Androidprogrammierung im Team ist, ist Android die Plattform unserer Wahl.

### 2.2.2. iOS Smartphone

Smartphones mit dem iOS Betriebssystem sind ebenfalls weit verbreitet. In der Programmierung auf dieser Plattform haben wir aber keine Erfahrungen. Außerdem müssten uns Mac-Computer zur Verfügung stehen, da Apple die nötigen Entwicklertools nur für sein Betriebssystem bereitstellt.

### 2.2.3. Einsatz von Bewegungssensoren und Kameras

Im Bereich der Sensoren ist die Hersteller Primesense nennenswert. Dieser stellt zu den eigenen Sensoren Treiber und Entwicklertools kostenfrei zur Verfügung. Der Preis für einen Sensor befindet sich aber im gehobenen Preissegment. Außerdem kann der Sensor nicht im lokalen Handel gekauft werden. Der Name der Primesense ist auch nicht vielen Personen ein Begriff, obwohl Primesense die Sensoren für Microsoft Kinect entwickelt. Angesichts dieser Nachteile wollen wir vorerst von diesen Sensoren Abstand halten, da wir denken, dass die Applikation auf sonst möglicherweise kaum Verbreitung finden kann. Über eine abstrahierte Schnittstelle soll eine spätere Erweiterung mit Sensoren aber möglich sein.

# 3. Paperprototypes und Interaktionen

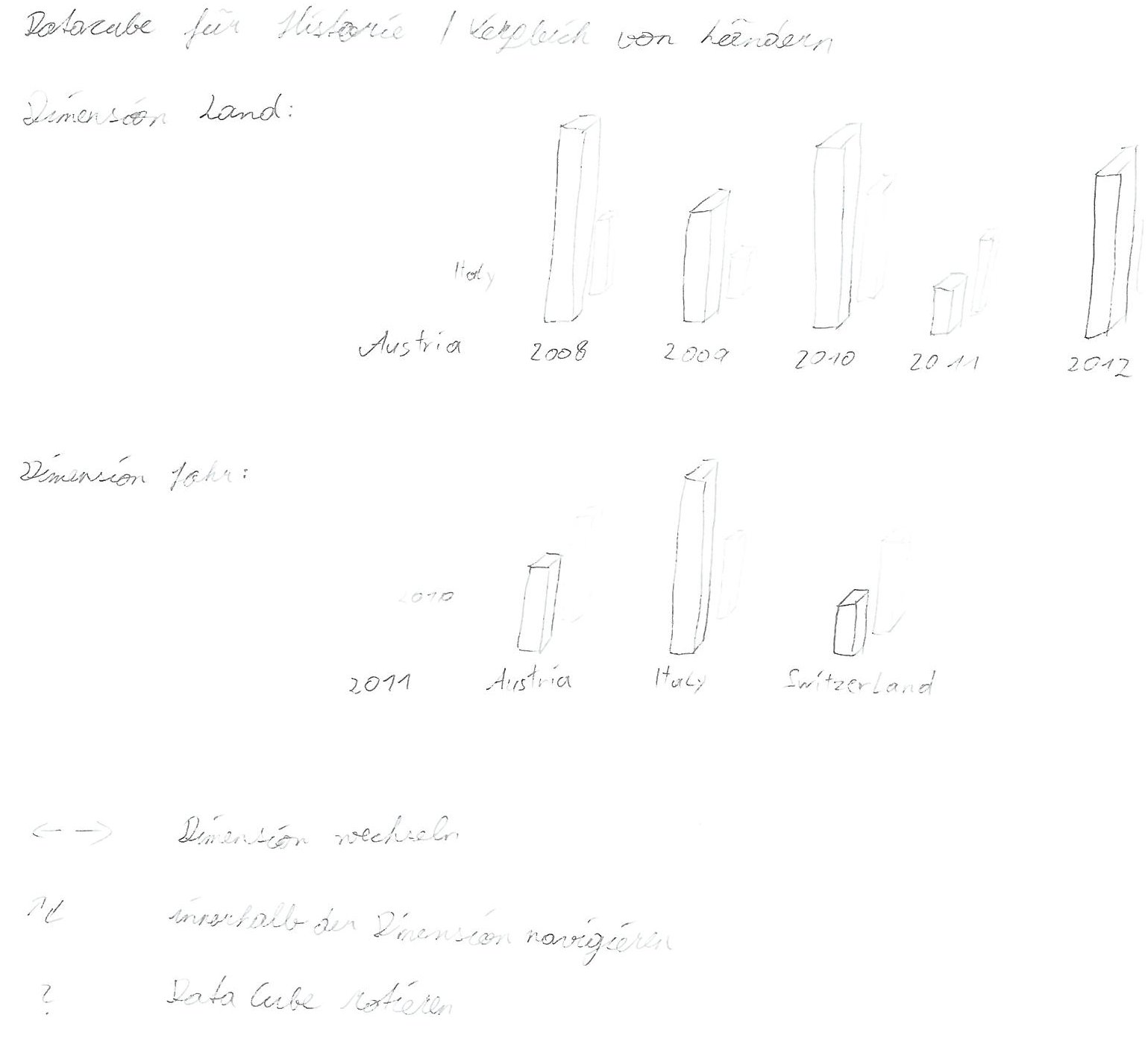
## 3.1. Erste Skizzen

### 3.1.1. Auswahl von Kontinenten, Ländern und Indikatoren



Die Auswahl von Ländern über die Kontinente und dann die Indikatoren der Weltbank enthält zu viele Ebenen. Das Auswahlverfahren basiert auf Listen. Dadurch muss der User möglicherweise sehr lange Listen durchforsten, bis er oder sie die gewünschten Einträge erreicht. Zusätzlich kommen hier keinerlei 3D-Fähigkeiten zum Einsatz.

### 3.1.2. Vergleich von Ländern

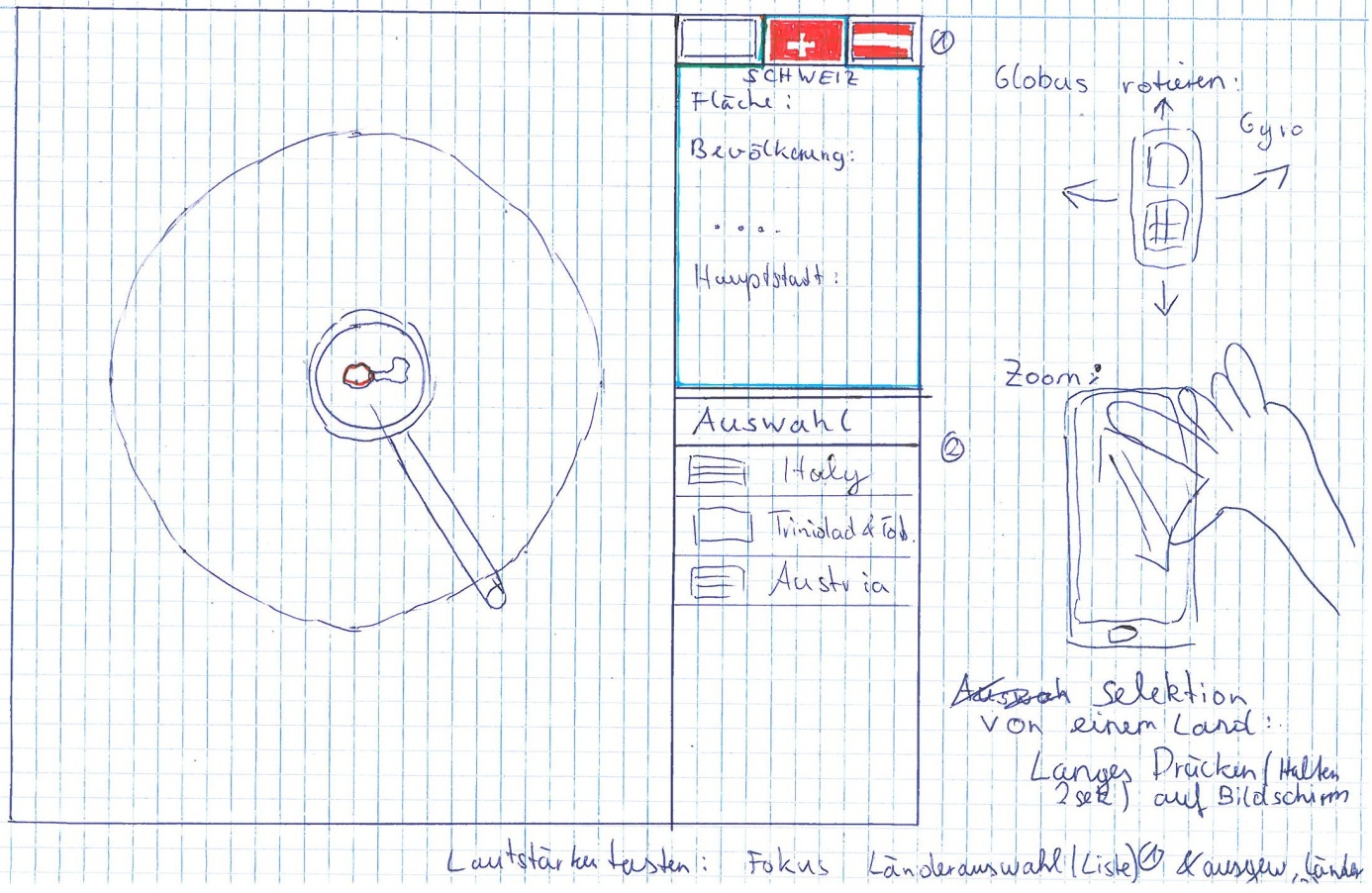


- viele Daten können irritieren

- je nach gewünschter Vergleichsart unpraktisch

## 3.2. Skizzen zum User Interface

## 3.2.1. Länderauswahl

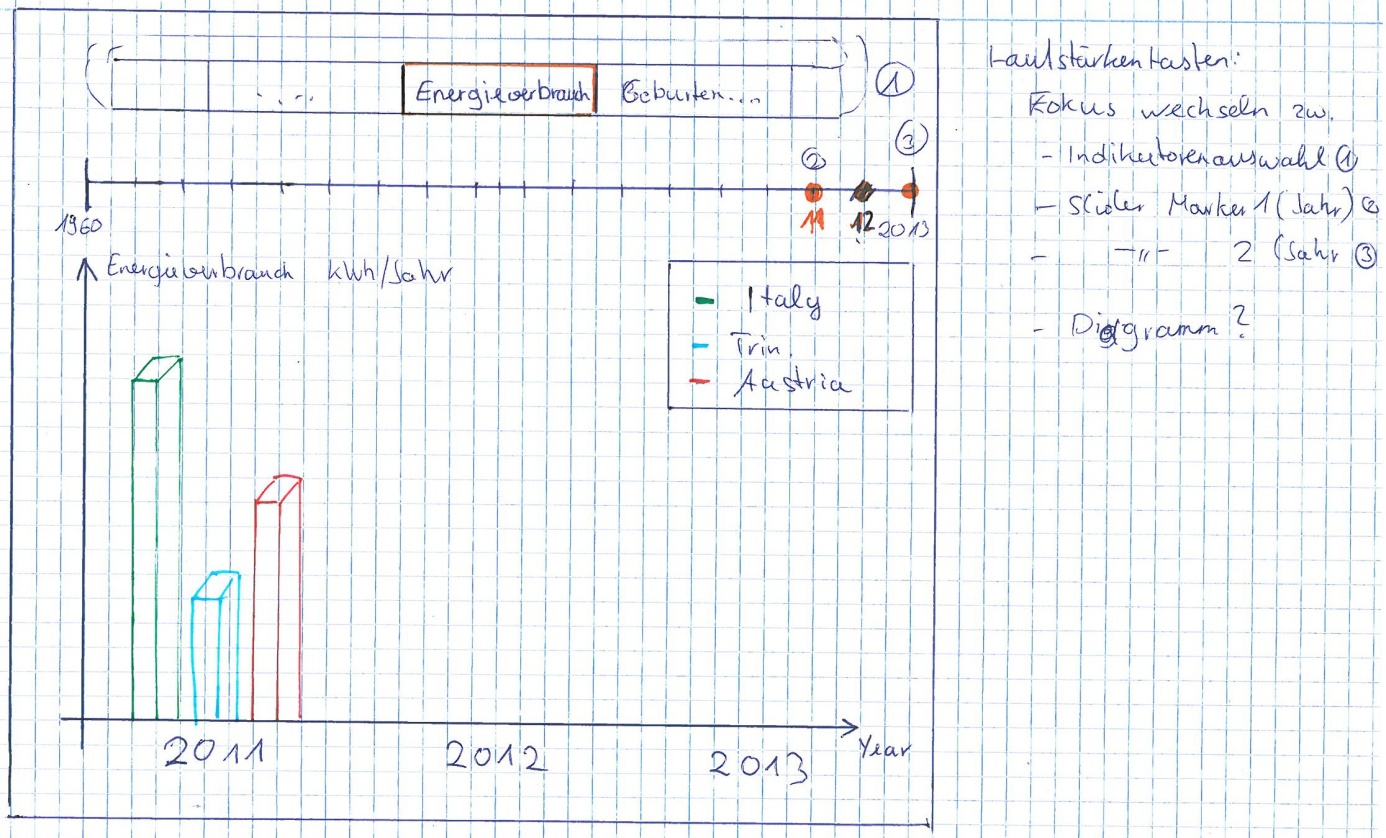


In der zweiten Version der Länderauswahl steht der Globus im Mittelpunkt. Ein Land soll mittels Rotation des Globus und Zooming gefunden und ausgewählt werden. Der Listenansatz steht alternativ für dem User unbekannte Länder bereit. Dieser bietet sich aber auch für zur Auswahl von kleinen Ländern an, die nicht gut über die Rotation zentriert werden können.

Ein Infofeld rechts oben soll grundlegende Informationen zum aktuell fokussierten Land anzeigen. Darunter gehören zum Beispiel die Hauptstadt und die Bevölkerungsanzahl.

Unten auf dem Bildschirm wird zusätzlich eine Liste der ausgewählten Länder für den Datenvergleich verwaltet.

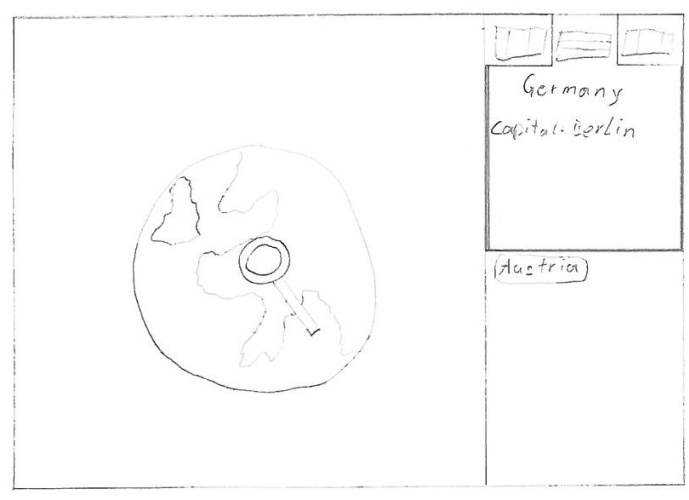
### 3.2.2. Vergleich von Ländern



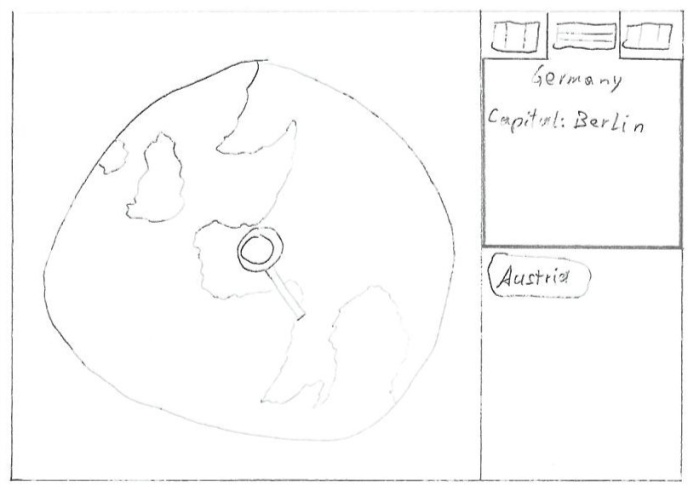
## 3.3. Interaktionen

### 3.3.1. Länderauswahl

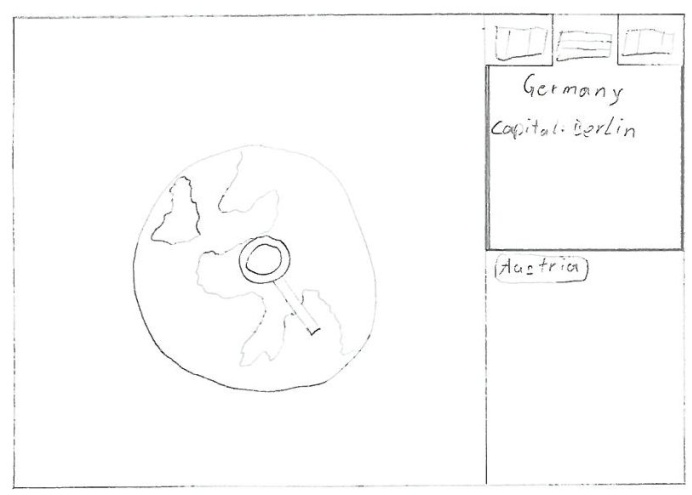
#### 3.3.1.1. Zoom



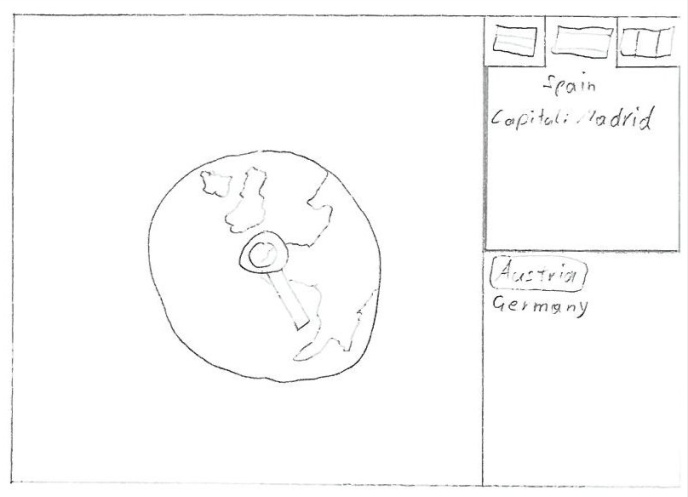
Bekannte Zoomgeste von Smartphones (zwei Finger spreizen oder zusammenziehen)



#### 3.3.1.2. Land auswählen



Smartphone rotieren um Globus zu drehen; Fokus oben gesetzt und horizontal wischen



Weitere gezeichnete „Screenshots“ für verschiedene Aktionen in Scan/pp005.jpg

# 4. Probleme während der Entwicklung und deren Lösung

## 4.1. Anfrage an den Webservice der Weltbank

- Weltbank scheinbar über Weihnachten umgestellt

- Countries OK; Indikatoren nicht mehr? Ohne \r (nur \nHost:) -> \r\n für jede Zeile nötig

- HTTP/1.1 Implementierung blockt weil Connection wohl doch offen bleibt

- HTTP/1.0 funktioniert nun weil Connection nicht offen gehalten wird -> kein Blocken mehr beim read

## 4.2. Textencoding in Unity

- bei XML

- ohne BOM (byte order mark) am Anfang sonst merkwürdige Exception

- System.Xml.XmlException: Text node cannot appear in this state. Line 1, position 1.

- mit MemoryStream read und danach write gelöst

- Fehler in C# Doku (<http://msdn.microsoft.com/en-us/library/system.io.bufferedstream%28v=vs.110%29.aspx>): falsche Offsets

# 5. Systemanforderungen

|  |  |
| --- | --- |
| **Globetrotter** | |
| Betriebssystem | Windows oder Mac |
| Bildschirmauflösung | Empfohlen: 1280 x 720  16:9 Seitenverhältnis für korrekte Darstellung erforderlich |
| **Globetrotter-Steuerungsapp** |  |
| Betriebssystem | Android Version ? |
| Speicher |  |
| Sensoren | - Gyrosensor |