Überlegungen zum Beleg "Schnitzeljagd" in Technik mobiler Systeme

Stand: 22.05.2015

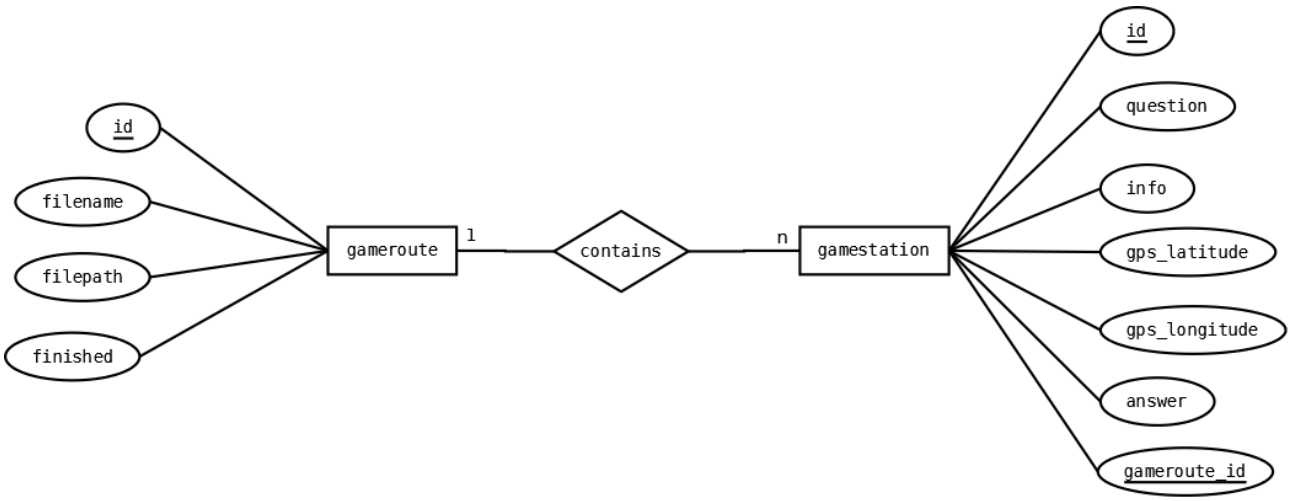
Anforderungen

- Spielroute erstellen, spielen, importieren, exportieren, löschen
- GPS-Position einer Spielstation muss vom Benutzer (bzw. Mobiltelefon) vor Ort erfasst werden, kann nicht manuell eingegeben werden
- 4.3 (Jelly Bean) als Minimalversion
- SQLite als App-DB verwenden
- Wenn GPS ausgestellt: Sperre + Notification "Bitte GPS wieder anschalten" anzeigen
- Zu erstellendes Spiel speichern: erst beim Klick auf "Route fertig", keine Zwischenstände speichern
- Spiel absolviert: Klick auf "Antworten" der letzten Stationsfrage + Antwort ist richtig, keine Zwischenstände speichern
- Information, welche Spiele absolviert wurden, in SQLite-DB der App abspeichern
- Für die Spieldateien (JSON-Dateiein die ein Spiel darstellen) soll die Dateiendung ".schnitzel" verwendet werden

Datenhaltung

Datenmodell

Alle Daten der App werden in einer SQLite-Datenbank gespeichert. Die App bezieht demzufolge alle Informationen aus dieser. Folgend ist das ERM zu sehen, was die DB abbilden soll:



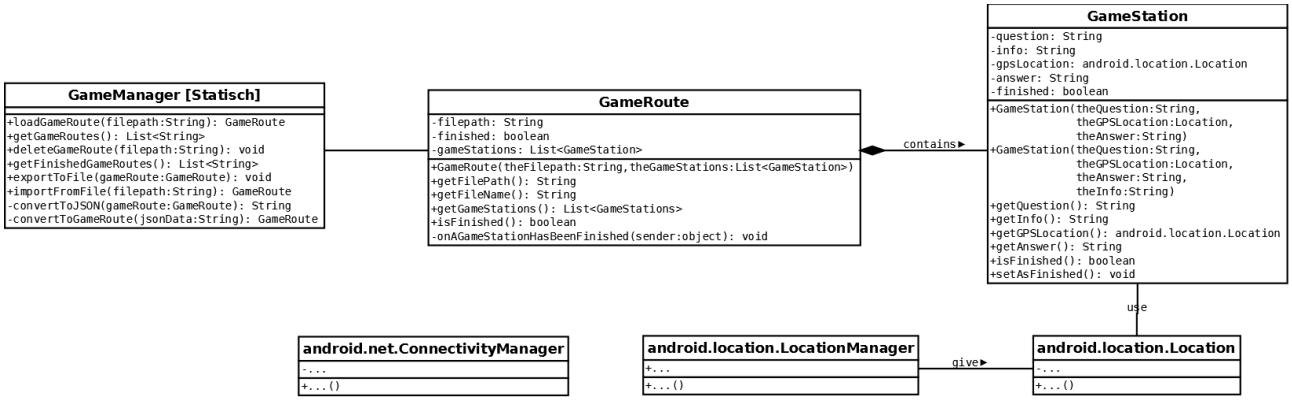
Beispieldatensatz Gameroute:

id	filename	filepath	finished
1	spiel123.schnitzel	Data/app/	0

Beispieldatensatz Gamestation:

id	question	info	gps_latitude	gps_l*ongitude*	answer	*gameroute_id*
1	"Was siehst du dort?"	"Es ist blau"	52.4933819	13.52949	"Briefkasten"	1

Klassendiagramm / Ablauf



Im Arbeitsspeicher soll es immer nur ein Spielobjekt geben, egal ob es gerade erstellt oder gespielt wird. Es soll also vermieden werden, alle verfügbaren Spielrouten bei Start der App als Objekte zu laden (Speicherplatz). Ist ein Spiel absolviert, werden dessen Ressourcen im RAM danach wieder freigegeben. Nachfolgend ist ein erster Entwurf der App-spezifischen Klassen zu sehen:

Der GameManager ist eine statische Klasse, also eine Sammlung von Hilfsfunktionen für die App. Diese soll nur Funktionen beinhalten, welche App-spezifisch aber nicht System-spezifisch sind. Das bedeutet, dass z.B. die vom Gerät bereitgestellte GPS- oder Internet-Funktionalität hier nicht behandelt werden soll.

Die GameRoute-Klasse repräsentiert eine Spielroute bzw. einen Spieldurchlauf, d.h. das Objekt hält alle nötigen Informationen für diese. Dieses Objekt unterstützt die zwei Zustände "absolviert" und "nicht absolviert". Nach Erstellung des Objektes befindet sich dieses im Zustand "nicht absolviert". Damit das GameRoute-Objekt weiß, wann eine Spielstation erfolgreich absolviert wurde, sendet die Spielstation ein Event an das GameRoute-Objekt. Das GameRoute-Objekt prüft dann, ob alle Spielstationen und somit das Spiel absolviert wurde oder nicht. Wurden alle Spielstationen absolviert ändert sich der Zustand zu "absolviert". Nachdem eine Spielroute bzw. Spiel absolviert wurde, wird diese Information in der Spiel-DB abgespeichert und das Spiel-Objekt mit seinen Spielstation-Objekten im RAM freigegeben.

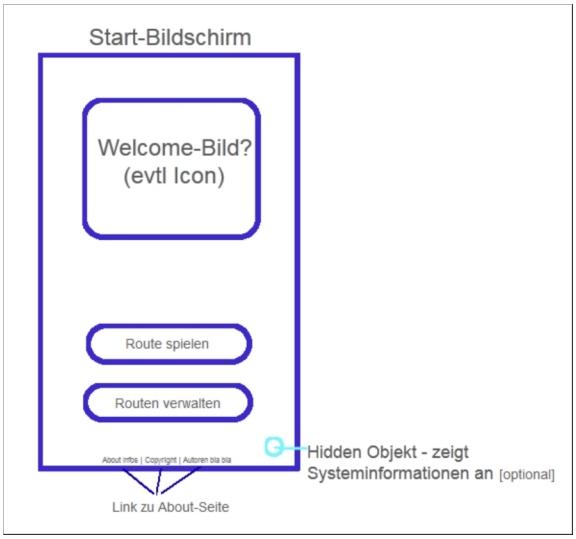
Die GameStation-Klasse repräsentiert eine Spielstation bzw. Level, d.h. das Objekt hält alle nötigen Informationen für diese. Dieses Objekt unterstützt die zwei Zustände "absolviert" und "nicht absolviert". Nach Erstellung des Objektes befindet sich dieses im Zustand "nicht absolviert". Die Funktion setAsFinished() sorgt dafür, dass der Zustand des Objektes von "nicht absolviert" zu "absolviert" übergeht und ein entsprechendes Event an das Mutter-Objekt "GameRoute" gesendet wird. Da die "Info"-Eigenschaft der Klasse optional ist, gibt es einen Konstruktor ohne "Info"-Parameter. Die Klasse nutzt zudem als Speicherung der Standortinformationen die "Location"-Klasse aus dem android-Framework.

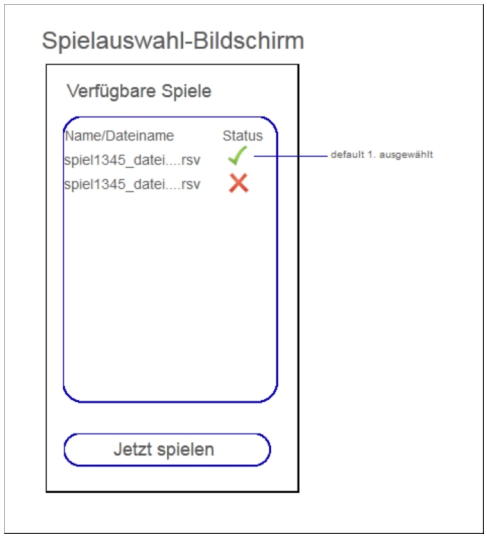
Der LocationManager aus dem android-Framework stellt alle nötigen GPS-Informationen zur Verfügung.

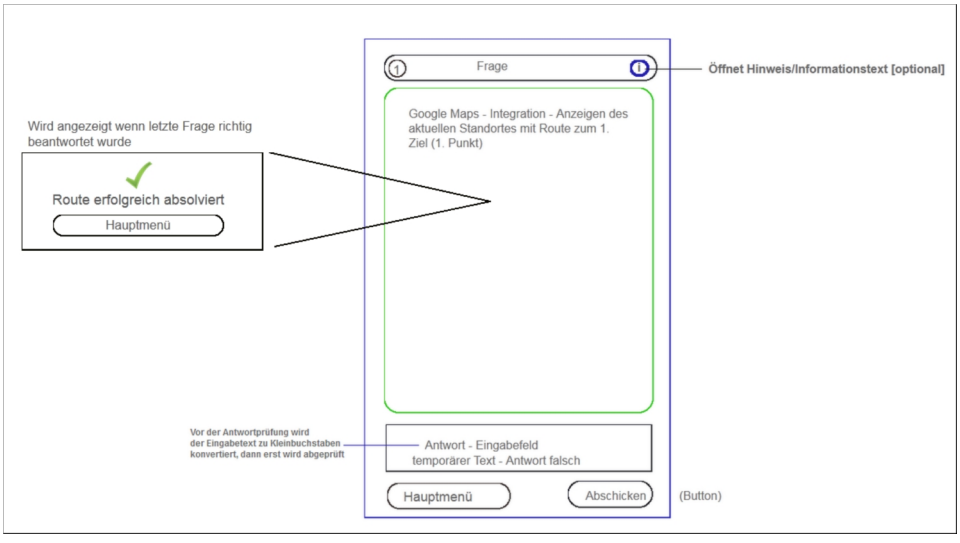
Der ConnectivityManager aus dem android-Framework stellt alle nötigen Verbindungsinformationen (Internet) zur Verfügung.

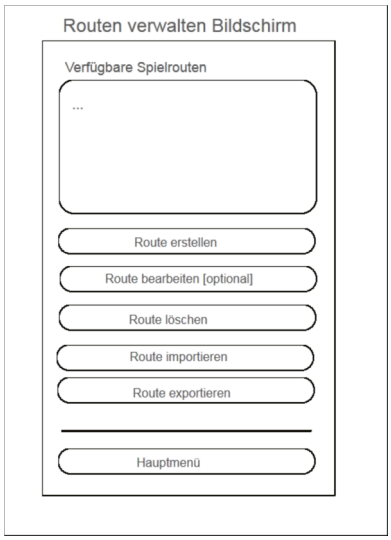
Benutzerinterface

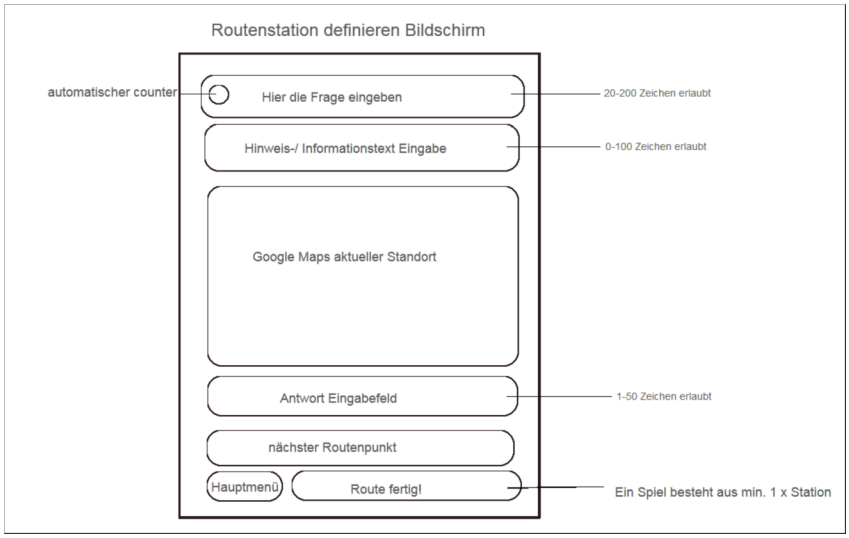
Nachfolgend sind alle View's der App aufgelistet:








Routen verwalten Bildschirm
Verfügbare Spielrouten
...
Route erstellen
Route bearbeiten [optional]
Route löschen
Route importieren
Route exportieren
Hauptmenü


Routenstation definieren Bildschirm
automatischer counter
Hier die Frage eingeben
20-200 Zeichen erlaubt
Hinweis-/ Informationstext Eingabe
0-100 Zeichen erlaubt
Google Maps aktueller Standort
Antwort Eingabefeld
1-50 Zeichen erlaubt
nächster Routenpunkt
Hauptmenü
Route fertig!
Ein Spiel besteht aus min. 1 x Station

Exportformat

Als Export-/Austausch- bzw. Dateiformat wurde JSON festgelegt. Eine Spielroute soll wie folgt in JSON abgespeichert sein:

spiel123.schnitzel

```
{
  "number of gameroutes": 2,
  "gameroutes": [
    {
      "number": 1,
      "question": "Was siehst du?",
      "answer": "Briefkasten",
      "info": "Es ist blau.",
      "longitude": 23.34123,
      "latitude": 45.23451
    },
    {
      "number": 2,
      "question": "Was ist dort?",
      "answer": "Weihnachtsmann",
      "info": "",
      "longitude": 14.34109,
      "latitude": 67.76451
    }
  ]
}
```