

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0) License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/; or, (b) send a letter to Creative Commons, 171 2nd Street, Suite 300, San Francisco, California, 94105, USA.”

Copyright (C) 2010 Fraunhofer Institute for Open Communication Systems (FOKUS)

Fraunhofer FOKUS

Kaiserin-Augusta-Allee 31

10589 Berlin

Tel: +49 30 3463-7000

[info@fokus.fraunhofer.de](mailto:info@fokus.fraunhofer.de)

# OpenRide: Update-Benachrichtigungen und –Tracking

## Mobiler Client

### Allgemeines Verhalten

* Pollt alle 15 Sekunden den Webservice “**/configuration/updates**” und erhält als Antwort die Anzahl der für den eingeloggten Benutzer neu vorliegenden Status-Updates  
  (als Update einer Fahrt gelten **neu hinzugekommene Matches** sowie solche, bei denen sich der **Status der Gegenseite seit dem letzten Abruf verändert** hat)
* Anzahl der Updates wird für Angebote und Gesuche (Fahrer-/Mitfahrermodus) separat abgelegt als “**driverupdatecount**” und “**riderupdatecount**”
* Wenn für Angebote oder Gesuche eine **Anzahl größer als null** festgestellt wird, wird
  + **Auf dem** **Homescreen** die entsprechende Zahl angezeigt
  + Falls der jeweilige Modus gerade aktiv ist, auch **in der** **oberen Tableiste innerhalb des Fahrtentabs** die entsprechende Anzahl angezeigt

### Verhalten der Liste der aktiven Angebote / Gesuche (Fahrtentab)

* Einträge für Fahrten mit einem Status-Update werden durch roten Schriftzug “**Update!**” markiert
* Beim Aufklappen eines markierten Eintrags wird vom Client selbstständig der Wert der jeweiligen “**\*updatecount**”-Variaben heruntergezählt und dementsprechend die in der Tableiste und auf dem Homescreen angezeigte Zahl aktualisiert  
  (Es wird im Moment des Aufklappens also das Ergebnis des nächsten Webservice-Pollings vorweggenommen)

## Server

### Webservice

* “**/configuration/updates**” liefert eine “**UpdateResponse**” mit der Anzahl der Fahrten und Gesuche, zu denen für den angemeldeten Benutzer Updates vorliegen, die er noch nicht abgerufen hat (in zwei separaten Variablen)
* Zur Feststellung der zu liefernden Zahlen greifen sie auf die Methoden “**getActiveDrives**” und “**getActiveRideRequests**” des “**driver/riderUndertakesRideController**” zurück und konsultieren für jede aktive Fahrt anschließend die Controller-Methode “**isDriveUpdated**” bzw. “**isRideUpdated**”

### Datenbank

* In der Tabelle „**match**“ werden die Spalten „**driver\_change**“, „**rider\_change**“, „**driver\_access**“ und „**rider\_access**“ (jeweils vom Typ timestamp) zur Verfolgung der Änderungen und Zugriffe auf einzelne Matches genutzt
* Für einen **Fahrer** gilt ein Angebot als „updated“, wenn bei mindestens einem dazugehörigen Match der Zeitpunkt des **„driver\_access“ vor dem „rider\_change“** liegt ODER **„driver\_access“** **NULL** ist
* Für einen **Mitfahrer** gilt ein Gesuch als „updated“, wenn bei mindestens einem dazugehörigen Match der Zeitpunkt des **„rider\_access“ vor dem „driver\_change“** liegt ODER **„rider\_access“** **NULL** ist

### EJBs

* “driverUndertakesRideController” und “riderUndertakesRideController” besitzen die Methoden “**isDriveUpdated**” und “**isRideUpdated**”, über die sich anhand der “rideId” bzw. “riderrouteId” feststellen lässt, ob es zu der Fahrt für den angemeldeten Benutzer neue oder geänderte Matches gibt  
  (die MatchEntity besitzt dazu die beiden Named Queries „findChangesSinceAccessByDriverByRideId“ / „findChangesSinceAccessByRiderByRiderrouteId“)
* Das Attribut „**driver\_change**“ eines Matches wird auf die aktuelle Uhrzeit gesetzt, wenn eine **Statusänderung durch den Fahrer** ausgelöst wird:
  + Status eines gebuchten Matches ändert sich in „ACCEPTED“, nicht zum Zuge kommende Matches in „NO\_MORE\_AVAILABLE“  
    (RiderUntertakesRiderController.addRiderToRide, DriverUntertakesRideController.getMatches)
  + Status eines Matches ändert sich in „COUNTERMANDED“  
    (DriverUntertakesRideController.removeRide)
  + Status eines Matches ändert sich in „ACCEPTED“  
    (DriverUntertakesRideController.acceptRider)
  + Status eines Matches ändert sich in „REJECTED“  
    (DriverUntertakesRideController.rejectRider)
* Das Attribut „**rider\_change**“ eines Matches wird auf die aktuelle Uhrzeit gesetzt, wenn eine **Statusänderung durch den Mitfahrer** ausgelöst wird:
  + Status eines gebuchten Matches ändert sich in „ACCEPTED“, nicht zum Zuge kommende Matches in „NO\_MORE\_AVAILABLE“  
    (RiderUntertakesRiderController.addRiderToRide, DriverUntertakesRideController.getMatches)
  + Status eines Matches ändert sich in „COUNTERMANDED“  
    (RiderUntertakesRideController.removeRide, RiderUntertakesRideController.removeRiderFromRide, RiderUntertakesRideController.setMatchCountermand)
  + Status eines Matches ändert sich in „ACCEPTED“  
    (DriverUntertakesRideController.acceptRider)
  + Status eines Matches ändert sich in „REJECTED“  
    (DriverUntertakesRideController.rejectRider)
* Das Attribut „**driver\_access**“ eines Matches wird auf die aktuelle Uhrzeit gesetzt, wenn ein   
  **Zugriff des Fahrers** stattfindet, z.B. über den mobilen Client / Webservice  
  (DriverUntertakesRideController.getMatches)
* Das Attribut „**rider\_access**“ eines Matches wird auf die aktuelle Uhrzeit gesetzt, wenn ein   
  **Zugriff des Mitfahrers** stattfindet, z.B. über den mobilen Client / Webservice  
  (RiderUntertakesRideController.getMatches)

## UML-Diagramm

See documentation

