CarDuinoDroid

Entwicklerdokumentation Carduinodroid

released by
Alexander Rose, Christoph Braun, Jenja Dietrich,
Marc Brunner, Michael Röding, Sven-Leonhard Weiler, Vincenz Vogel

Inhaltsverzeichnis

Kapitel 1

Einleitung

Unsere 3. Phase, die Implementierungsphase ist die wichtigste und gleichzeitig längste Phase in unserem Projekt gewesen. In dieser versuchten wir, aus unserem Anfangsprototyp einen voll funktionsfähigen Prototyp zu erschaffen, der es uns ermöglicht alle nötigen Funktionen auszuführen zu können. Dies versuchten wir durch Einteilung des Projektes in verschiedene Teilaufgaben.

Dies führte zur Einteilung in folgende Unterpunkte:

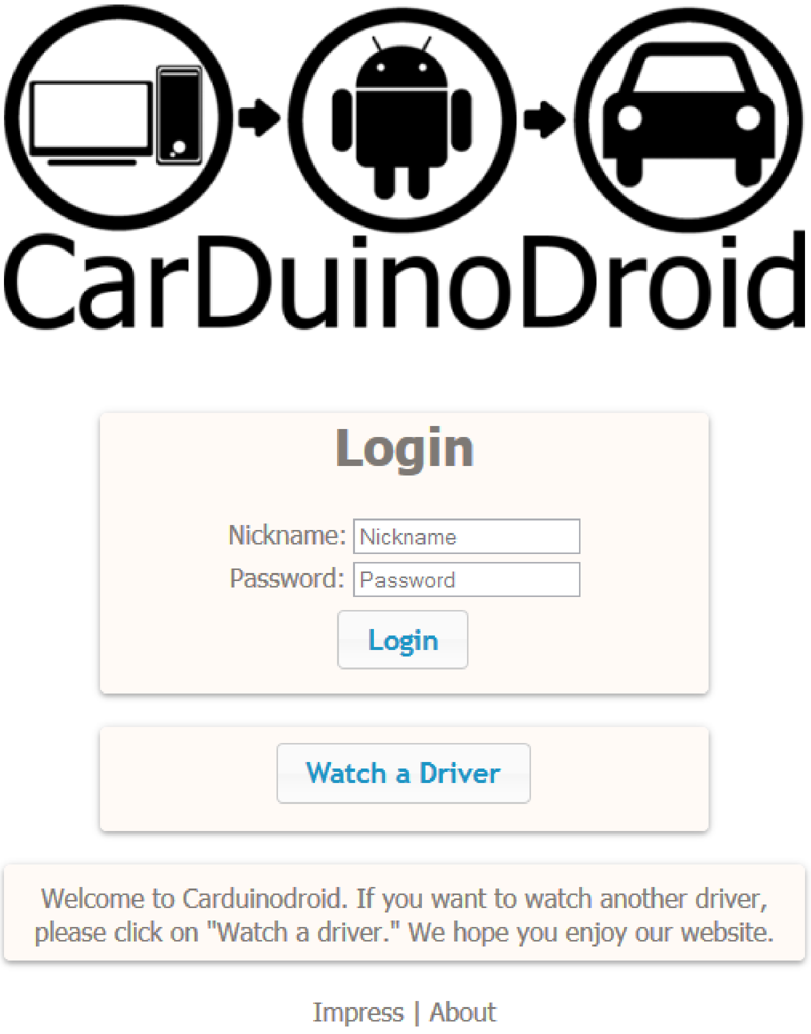
- GUI
- Datenbank
- Server
- WebSocket
- Chat
- Dokumentation

Diese Unterpunkte wurden alleine oder in Zweiergruppen aufgeteilt, wobei aber untereinander auch Aufgaben übernommen, Hilfestellungen gegeben und verschiedene Probleme gelöst wurden. Da in dieser Phase viel Quellcode entstand und wir diesen möglichst ordentlich und strukturiert abgeben wollten, entschieden wir uns für das Software-Dokumentationswerkzeug “Doxygen“. Mit Latex war es möglich daraus ein PDF zu erzeugen das unseren Quellcode gut strukturiert darstellte.

Kapitel 2

Implementierungsaufgaben

2.1 GUI



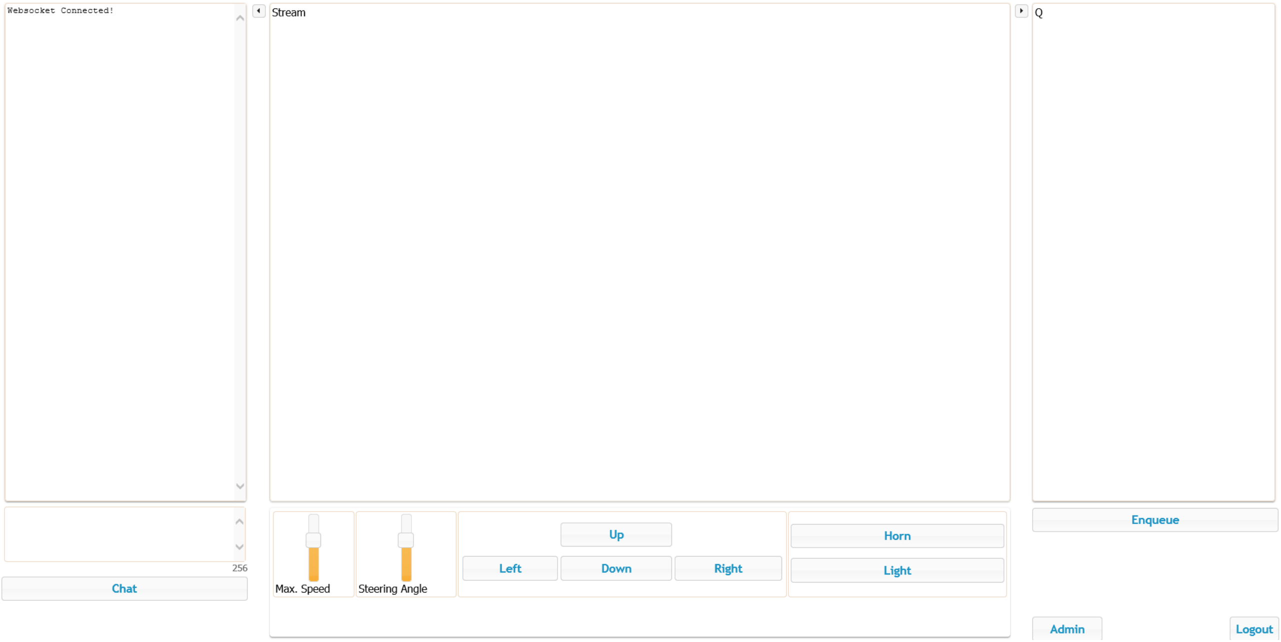
Nickname: Nickname

Password: Password

Welcome to Carduinodroid. If you want to watch another driver, please click on "Watch a driver." We hope you enjoy our website.

Startseite

Die Grafische Benutzeroberfläche von unserem Projekt ist sehr wichtig, da die Benutzer mit dieser Oberfläche arbeiten und sich zurechtfinden müssen. Wir haben sie deshalb schlicht und einfach gehalten. Die UserID sowie das Password erhält man vom Administrator um sich einzuloggen und das Carduinodroid zu steuern. Alternativ dazu ist es möglich ohne Benutzerdaten einem Fahrer zuzuschauen und im Chat zu kommunizieren.



Benutzerseite

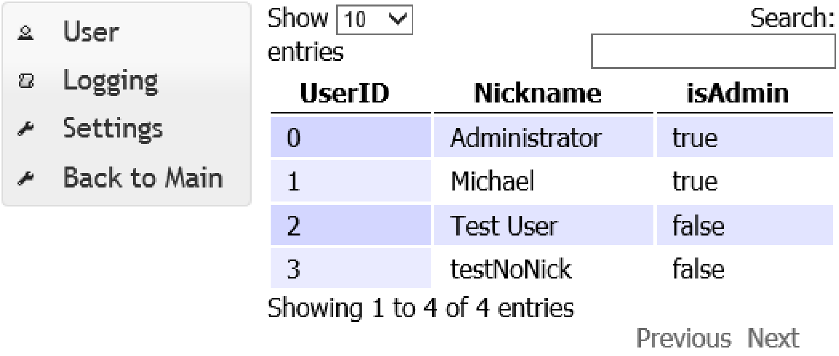
Hier ist die Hauptseite zu sehen, mit der es möglich sein soll, das Carduinodroid von überall aus der Welt zu steuern, sobald ein Computer mit Internetanschluss zur Verfügung steht.

Links ist der Chat zu sehen, Rechts die Warteschlange und in der Mitte soll das Bild der Handykamera erscheinen.

Es gibt verschiedene Einstellungen:

- Maximalgeschwindigkeit
- Lenkungswinkel
- Licht ein-/ausschalten
- Signalton erzeugen

Sowie Richtungstasten, mit denen man das Fahrzeug in die vorgesehene Richtung steuern kann. Weiterhin noch einen Logout-Button und einen Button für den Administratorbereich
(nur zu sehen, wenn die nötigen Recht vorhanden sind)



UserID	Nickname	isAdmin
0	Administrator	true
1	Michael	true
2	Test User	false
3	testNoNick	false

Showing 1 to 4 of 4 entries

Administratorseite

Wenn man sich auf der Startseite mit einem Administratornamen und dem dazugehörigem Passwort einloggt, gelangt man zur Hauptseite. Auf dieser befindet sich dann ein zusätzlicher Button, um in den Administratorbereich zu kommen. In diesem ist es möglich einen neuen Benutzer mit verschiedenen Rechten zu erstellen. Auch das Protokoll kann angezeigt werden und es gibt die Möglichkeit verschiedene Filter einzusetzen.

Was wurde gemacht:

- Front-End für alle gängigen Browser anpassen
  (getestet mit: Internet Explorer, Mozilla Firefox, Google Chrome)
- jQuery einarbeiten
- Front-End anpassen damit alle Interaktionen möglich sind
- Hilfe bei Front-End-Spezifischer Implementierung
- Custom Tags schreiben
- Dynamischen Content erzeugen

Probleme waren:

- Wie verschaffe ich mir Zugang zum Java-Teil
- Wie trenne ich Businesscode von Präsentationscode (Java vs JSP)

Lösung:

- JSTL (Java Standard Tag Library)
- Wiederverwendbar
- Debugbar
  (Scriplets erzeugen bei einer Exception nur eine weiße Seite)

2.2 Datenbank

Was wurde gemacht:

- ein Java-Datenbank-Interface zur Kommunikation mit einer MySQL Datenbank
- Datenbank nutzt MariaDB JDBC Treiber
- Die Datenbank-Klasse ist die einzige Verbindung zur Datenbank
  (Anderer Code kann nicht auf die Datenbank zugreifen)

Probleme waren:

- Es sollte nur eine Verbindung geben
- MySQL constraints funktionierten anfangs nicht
- Eine SQL-Injection soll verhindert werden

Lösung:

- das Verbindungsobjekt ist static - Singleton ähnliche Implementation
- MySQL constraints müssen einen eindeutigen Namen besitzen
- PreparedStatements nutzen um SQL-Injection zu verhindern

2.3 Server

Was wurde gemacht:

- Datenstrukturen für aktive Session (von Tomcat und von der Datenbank) und für die Warteschlange
- Timerhandling für die Warteschlange
  (bei leerer Warteschlange erfolgt eine Überprüfung jede Sekunde / wenn jemand fährt wird ihm nach der eingestellten Fahrzeit das Fahrrecht wieder entzogen)
- Filtern bzw. Reagieren auf ServletRequests
  (Login, Einreihen in die Warteschlange, Gast-Login, Logout)
- Logging von Benutzeraktionen
  (Login,Logout, Warteschlange, Fahrer, GPS (in einstellbaren Intervallen))
- Implementierung der Videoübertragung für Client und Server
  (variable Auflösungen sind möglich)

Probleme waren:

- TimerTasks lassen sich nicht von innen heraus neu aufrufen
- Tomcat-Session gibt manchmal IPv4 und manchmal IPv6 zurück
- Unterscheidung von Tomcat-Session und Datenbank Session
- Übertragung via Internet über Websockets
  (Übertragung via LAN stellte kein Problem dar)
- mangelhafte Recherchemöglichkeiten bezüglich Websockets für Tomcat führte zu zeitintensivere Arbeit, als ursprünglich angedacht

Lösung:

- DummyTask kann den TimerTask aufrufen
- Datenbank wurde angepasst um allgemein IP's als String aufzunehmen
- 2 "synchrone" ArrayLists mit identischen Indizes für zusammengehörige Elemente
  (Verbesserungsmöglichkeiten vorhanden, da unschöne Lösung)
- Dokumentation der einzelnen Anwendungen/Aufrufe von Websockets

2.4 WebSocket

Was wurde gemacht:

- MyWebsocketServlet für die Verarbeitung von serverseitigen Websocketverbindungen zur Steuerung und für den Chat
- Implementierung über Tomcats WebSocketServlet
  (funktioniert nur im Tomcat Servlet Container)
- Javascript/jQuery Code für die clientseitig Websocketverbindung zur Steuerung und für den Chat
- Es werden nur reine Websockets verwendet, deshalb ist eine aktuelle Browserversion notwendig

Probleme waren:

- Erster Ansatz mit einem Websocket Framework (Atmosphere) konnte auch nach langer Recherche nicht zum Laufen gebracht werden, dadurch war es nicht möglich eine Abstraktion der Websocket Verbindung herzustellen
- wenig brauchbare Informationen und Dokumentation zum Thema

Lösung:

- Nur Tomcat Websockets
- Plain Javascript Websockets, da diese keine fallback Alternative bieten

2.5 Chat

Was wurde gemacht:

- Broadcast-Funktion um Chat Nachrichten an alle verbundene Clients weiterzuleiten
- Verbindung von GUI und Websocket Code
- Logging der Chat Nachrichten in der Datenbank

Probleme waren:

- Einarbeitung in Websockets und Javascript/jQuery
- Fehler waren nicht einfach zu finden, somit viel *trial and error*

Lösung:

- Chrome Dev Tools
- Server Konsolenausgaben für jeden einzelnen Schritt

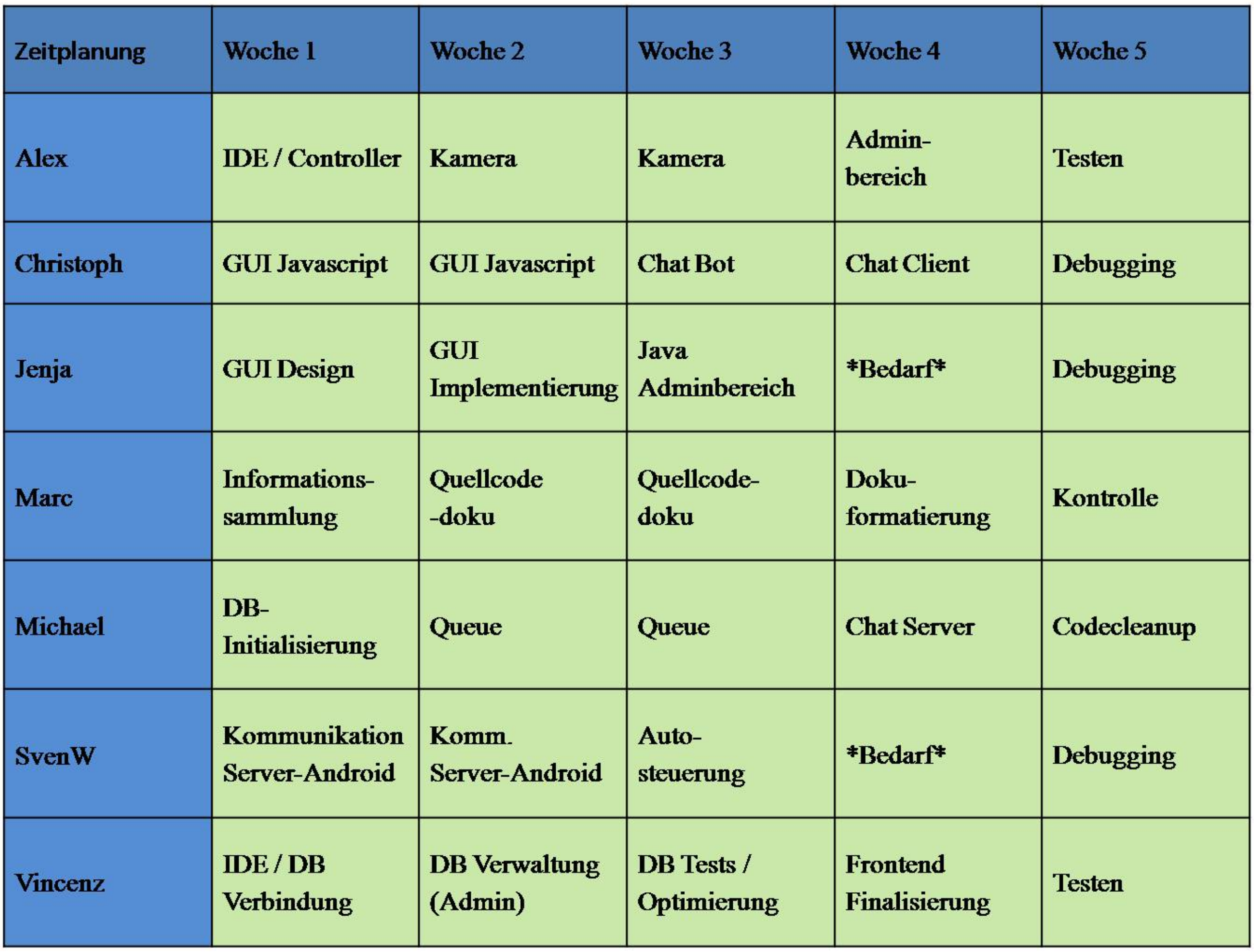
Kapitel 3

Zeit- und Arbeitseinteilung

3.1 Allgemein

Wie schon in der Einleitung zu lesen ist, haben wir unser Projekt in verschiedene Unterpunkte aufgeteilt. Dadurch war es uns möglich effizienter und unabhängiger zu arbeiten. Um dies zu realisieren und um überhaupt erst eine Übersicht über die einzelnen Projektarbeiten zu bekommen, stellten wir in der 2. Phase einen Zeitplan vor und versuchten darin möglichst realitätsnah die nötigen Aufgaben und Schritte im voraus zu planen. Da es nie so ist, wie man denkt, vergleichen wir nun den Erwartungswert mit dem tatsächlichen Wert. Um dies möglichst einfach zu gestalten, benutzen wir verschiedene Farben in der zweiten Tabelle.

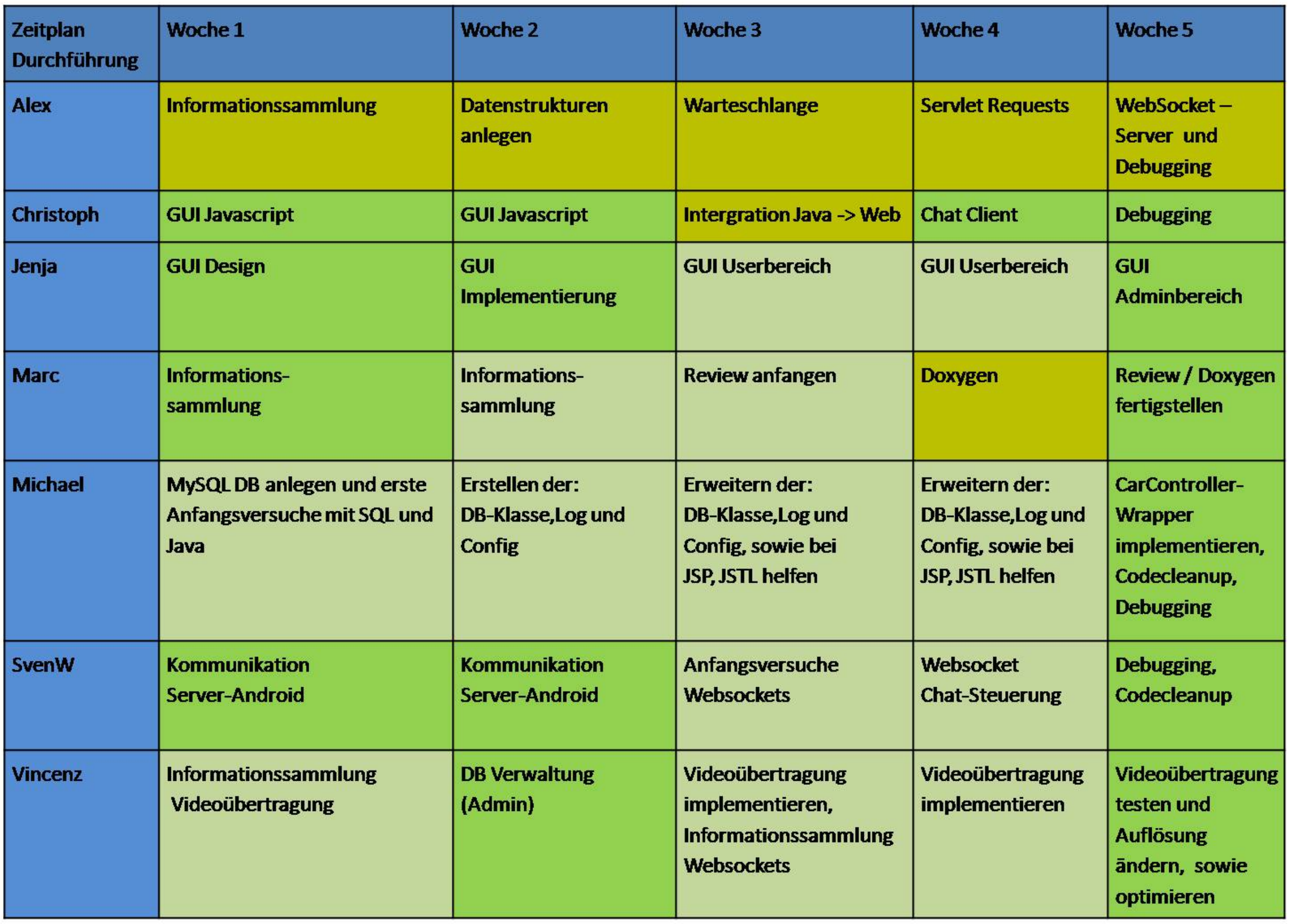
- Hellgrün zeigt uns an, dass wir mit unserer Planung sowohl zeitlich als auch arbeitstechnisch richtig lagen
- Ein blasses Grün zeigt uns an, dass wir zwar nicht planmäßig vorgegangen sind, aber die Richtung der Aufgaben stimmte und in dieses Ausmaß zu erwarten waren
- Gelb zeigt uns an, dass wir neue, nicht geplante Aufgaben bearbeiten mussten



Zeitplanung	Woche 1	Woche 2	Woche 3	Woche 4	Woche 5
Alex	IDE / Controller	Kamera	Kamera	Admin-bereich	Testen
Christoph	GUI Javascript	GUI Javascript	Chat Bot	Chat Client	Debugging
Jenja	GUI Design	GUI Implementierung	Java Adminbereich	*Bedarf*	Debugging
Marc	Informations-sammlung	Quellcode -doku	Quellcode-doku	Doku-formatierung	Kontrolle
Michael	DB-Initialisierung	Queue	Queue	Chat Server	Codecleanup
SvenW	Kommunikation Server-Android	Komm. Server-Android	Auto-steuerung	*Bedarf*	Debugging
Vincenz	IDE / DB Verbindung	DB Verwaltung (Admin)	DB Tests / Optimierung	Frontend Finalisierung	Testen

3.2 Planungstabelle

In der Planungstabelle versuchten wir, dass kommende Projekt möglichst realitätsnah abzubilden. Aus unserem Pflichtenheft wussten wir, was ungefähr auf uns zukommen wird und versuchten die Beschreibung allgemein zu halten.



Zeitplan Durchführung	Woche 1	Woche 2	Woche 3	Woche 4	Woche 5
Alex	Informationssammlung	Datenstrukturen anlegen	Warteschlange	Servlet Requests	WebSocket – Server und Debugging
Christoph	GUI Javascript	GUI Javascript	Intergration Java -> Web	Chat Client	Debugging
Jenja	GUI Design	GUI Implementierung	GUI Userbereich	GUI Userbereich	GUI Adminbereich
Marc	Informations-sammlung	Informations-sammlung	Review anfangen	Doxygen	Review / Doxygen fertigstellen
Michael	MySQL DB anlegen und erste Anfangsversuche mit SQL und Java	Erstellen der: DB-Klasse,Log und Config	Erweitern der: DB-Klasse,Log und Config, sowie bei JSP, JSTL helfen	Erweitern der: DB-Klasse,Log und Config, sowie bei JSP, JSTL helfen	CarController-Wrapper implementieren, Codecleanup, Debugging
SvenW	Kommunikation Server-Android	Kommunikation Server-Android	Anfangsversuche Websockets	Websocket Chat-Steuerung	Debugging, Codecleanup
Vincenz	Informationssammlung Videoübertragung	DB Verwaltung (Admin)	Videoübertragung implementieren, Informationssammlung Websockets	Videoübertragung implementieren	Videoübertragung testen und Auflösung ändern, sowie optimieren

3.3 Dokumentationstabelle

In unserer Dokumentationstabelle realisierten wir die Arbeit der letzten Wochen und verglichen diese dann, mit unserer erstellten Planungstabelle. Dabei fällt auf, dass es kaum größere, außerplanmäßige Arbeiten gab, aber auch das wir nur einen groben Rahmen zu Anfang festgesteckt hatten. Alex befasst sich mit allen anstehenden, wichtigen Aufgaben, sodass er auch nicht die geplanten Aufgaben erledigten konnte, da diese teilweise weggefallen oder übernommen worden sind. Dadurch entstand uns kein Nachteil, sondern wir konnten das Projekt dadurch flüssiger und schneller abarbeiten. Unser Dokumentationswerkzeug stand anfangs noch nicht fest, weshalb hier "Doxygen"gelb unterlegt auftaucht. Der *Chat Bot* viel aufgrund der Websockets weg, weshalb Christoph sich ausgiebig mit der Javaintegration in das Web beschäftigte. Zusammengefasst kann man sagen, dass wir in unserem Projekt in der dritten Phase eine klare Vorstellung hatte, was, wann zu erledigen ist und wir dadurch die dritte Phase gut meistern konnten.

Kapitel 4

Anhang Doxygen

CarDuinoDroid

Generated by Doxygen 1.8.3.1-20130402

released by
Alexander Rose, Christoph Braun, Jenja Dietrich,
Marc Brunner, Michael Röding, Sven-Leonhard Weiler, Vincenz Vogel

Mon Jun 17 2013 11:56:58

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 de.carduinodroid.shared.activeSession Class Reference

This Class is used to store all active sessions and provides access to them.

Static Public Member Functions

- static void init ()

  initializes the activeSession queue
- static void insertSession (String SessionID, String ipadress, String userid)

  inserts a session into the queue and also creates a session in the DB
- static String[] getAllSessions ()
- static void deleteSession (String SessionID)

  removes a session from the queue and closes the DB Session
- static void deleteAll ()

  removes all sessions from the queue and closes all DB sessions
- static int getSessionInt (String SessionID)

  returns the SessionID of the DB which belongs to the given Tomcat SessionID
- static String **getSession** (String SessionID)

3.1.1 Detailed Description

This Class is used to store all active sessions and provides access to them.

Author

Alexander ROse

3.1.2 Member Function Documentation

3.1.2.1 static void de.carduinodroid.shared.activeSession.deleteSession (String *SessionID*) [inline], [static]

removes a session from the queue and closes the DB Session

Parameters

SessionID	of the user

3.1.2.2 static String [] de.carduinodroid.shared.activeSession.getAllSessions ( ) [inline], [static]

Returns

Returns all SessionID's in the queue

3.1.2.3 static int de.carduinodroid.shared.activeSession.getSessionInt (String *SessionID*) [inline], [static]

returns the SessionID of the DB which belongs to the given Tomcat SessionID

Parameters

SessionID	from Tomcat

Returns

Returns SessionID from DB

3.1.2.4 static void de.carduinodroid.shared.activeSession.insertSession (String *SessionID*, String *ipadress*, String *userid*) [inline], [static]

inserts a session into the queue and also creates a session in the DB

Parameters

SessionID	Tomcat session of the user
ipadress	of the user (ipv4 and ipv6 are possible)
userid	of the user

The documentation for this class was generated from the following file:

- src/de/carduinodroid/shared/activeSession.java

3.2 de.carduinodroid.utilities.CarControllerWrapper Class Reference

Wrapper to combine the given source code in one class.

Public Member Functions

- void **confirmButtonUp** (boolean b)
- void **confirmButtonDown** (boolean b)
- void **confirmButtonRight** (boolean b)
- void **confirmButtonLeft** (boolean b)
- void setSpeed (int speed)

  Sets the current max speed of the car.
- void setAngle (int angle)

  Sets the current steering angle of the car.
- void setImg (BufferedImage img)

  Sets the current image.
- void setResolutions (String[] resolutions)

  Sets the possible resolutions.
- void setLatitude (String latitude)

Sets the current position (latitude) of the car.

- void setLongitude (String longitude)

  Sets the current position (longitude) of the car.

Static Public Member Functions

- static CarControllerWrapper getCarController (LogNG log)

  Use this constructor for first time instancing.
- static CarControllerWrapper getCarController () throws Exception

  Use this function to access the static CarControllerWrapper object.
- static void setDirection (boolean up, boolean down, boolean right, boolean left)

  Send control signal to the car.
- static void **driveForward** ()
- static void **driveBackward** ()
- static void **driveRight** ()
- static void **driveLeft** ()
- static int getSpeed ()

  Returns the current max speed of the car.
- static int getAngle ()

  Returns the current steering angle of the car.
- static BufferedImage getImg ()

  Returns the current image.
- static String[] getResolutions ()

  Returns all possible resolutions.
- static String getLatitude ()

  Returns the current latitude.
- static String getLongitude ()

  Returns the current longitude.

3.2.1 Detailed Description

Wrapper to combine the given source code in one class.

Since the code in de.carduinodroid.desktop was given to us we created this function to have one interface to interact with it.

This is implemented as a Singleton.

Author

Michael Röding

3.2.2 Member Function Documentation

3.2.2.1 static int de.carduinodroid.utilities.CarControllerWrapper.getAngle ( ) [inline], [static]

Returns the current steering angle of the car.

Returns

Returns the current steering angle of the car.

3.2.2.2 static CarControllerWrapper de.carduinodroid.utilities.CarControllerWrapper.getCarController (LogNG *log*) [inline], [static]

Use this constructor for first time instancing.

if necessary it creates the static CarControllerWrapper object.

If this function was called once use getCarController() instead.

Parameters

log	Log for logging.

Returns

Returns the static CarControllerWrapper object.

3.2.2.3 static CarControllerWrapper de.carduinodroid.utilities.CarControllerWrapper.getCarController ( ) throws Exception [inline], [static]

Use this function to access the static CarControllerWrapper object.

Returns

Returns the static CarControllerWrapper object.

Exceptions

Exception	in case the static CarControllerWrapper object is not present.

3.2.2.4 static BufferedImage de.carduinodroid.utilities.CarControllerWrapper.getImg ( ) [inline], [static]

Returns the current image.

Returns

Returns the current image.

3.2.2.5 static String de.carduinodroid.utilities.CarControllerWrapper.getLatitude ( ) [inline], [static]

Returns the current latitude.

Returns

Returns the current latitude or null if not available.

3.2.2.6 static String de.carduinodroid.utilities.CarControllerWrapper.getLongitude ( ) [inline], [static]

Returns the current longitude.

Returns

Returns the current longitude or null if not available.

3.2.2.7 static String [] de.carduinodroid.utilities.CarControllerWrapper.getResolutions ( ) [inline], [static]

Returns all possible resolutions.

Returns

Returns all possible resolutions as a String array.

3.2.2.8 static int de.carduinodroid.utilities.CarControllerWrapper.getSpeed ( ) [inline], [static]

Returns the current max speed of the car.

Returns

Returns the current max speed of the car.

3.2.2.9 void de.carduinodroid.utilities.CarControllerWrapper.setAngle (int *angle*) [inline]

Sets the current steering angle of the car.

Parameters

angle	The angle to set.

3.2.2.10 static void de.carduinodroid.utilities.CarControllerWrapper.setDirection (boolean *up,* boolean *down,* boolean *right,* boolean *left*) [inline], [static]

Send control signal to the car.

Parameters

up	
down	
right	
left	

3.2.2.11 void de.carduinodroid.utilities.CarControllerWrapper.setImg (BufferedImage *img*) [inline]

Sets the current image.

Parameters

img	The image to set.

3.2.2.12 void de.carduinodroid.utilities.CarControllerWrapper.setLatitude (String *latitude*) [inline]

Sets the current position (latitude) of the car.

Parameters

latitude	The latitude to set.

3.2.2.13 void de.carduinodroid.utilities.CarControllerWrapper.setLongitude (String *longitude*) [inline]

Sets the current position (longitude) of the car.

Parameters

longitude	The longitude to set.

3.2.2.14 void de.carduinodroid.utilities.CarControllerWrapper.setResolutions (String[] *resolutions*) [inline]

Sets the possible resolutions.

Parameters

resolutions	The resolutions to set.

3.2.2.15 void de.carduinodroid.utilities.CarControllerWrapper.setSpeed (int *speed*) [inline]

Sets the current max speed of the car.

Parameters

speed	The speed to set.

The documentation for this class was generated from the following file:

- src/de/carduinodroid/utilities/CarControllerWrapper.java

3.3 de.carduinodroid.Chat Class Reference

This Class is used to forward the Chat-messages to all users.

Static Public Member Functions

- static void BroadcastChat ()

3.3.1 Detailed Description

This Class is used to forward the Chat-messages to all users.

Author

Alexander Rose

3.3.2 Member Function Documentation

3.3.2.1 static void de.carduinodroid.Chat.BroadcastChat ( ) [inline], [static]

The documentation for this class was generated from the following file:

- src/de/carduinodroid/Chat.java

3.4 de.carduinodroid.utilities.Config Class Reference

Config class managing options.

Classes

- class Options

  Actual options.

Public Member Functions

- Config (LogNG logIN, String filePath_in)

  Sets log, file path and creates the options file if necessary.
- void readOptions ()

  Loads options from file.
- void saveOptions ()

  Saves options to file.
- void setDefault (boolean localhost)

  Creates default options.
- Options getOptions ()

  Returns the options.
- void setOptions (Options options)

  Set options.

3.4.1 Detailed Description

Config class managing options.

This class proved methods to load and save options. The options will be saved in config.properties and can be edited with a normal text editor.

To change the options call setOptions(Options options) followed by saveOptions()

Author

Michael Röding

3.4.2 Constructor & Destructor Documentation

3.4.2.1 de.carduinodroid.utilities.Config.Config (LogNG *logIN*, String *filePath_in*) [inline]

Sets log, file path and creates the options file if necessary.

Parameters

logIN	Log to use for logging
filePath_in	File path which should be used as root folder for everything else

3.4.3 Member Function Documentation

3.4.3.1 Options de.carduinodroid.utilities.Config.getOptions ( ) [inline]

Returns the options.

Returns

Returns the options.

3.4.3.2 void de.carduinodroid.utilities.Config.setDefault (boolean *localhost*) [inline]

Creates default options.

Parameters

3.4.3.3 void de.carduinodroid.utilities.Config.setOptions (Options *options*) [inline]

Set options.

Parameters

options	The options to set.

The documentation for this class was generated from the following file:

- src/de/carduinodroid/utilities/Config.java

3.5 de.carduinodroid.web.ConfigTag Class Reference

The documentation for this class was generated from the following file:

- src/de/carduinodroid/web/ConfigTag.java

3.6 de.carduinodroid.utilities.DBConnector Class Reference

This Class implements all functions which interact the database.

Public Member Functions

- DBConnector (LogNG logIN, Options opt)

  Initializes database connection and sets static variables.
- DBConnector () throws Exception
- void shutDown ()

  Commits all pending queries/updates and closes database connecting.
- int startDrive (String userID)

  Creates a new drive.
- boolean stopDrive (int driveID)

  Adds stop time timestamp.
- List< GPS > getGPSByDriveID (int driveID)

  Gets all GPS records for one drive.
- List< GPS > getGPSByTime (Timestamp begin, Timestamp end)

  Gets all GPS records between a given time frame.
- int createSession (String userID, String ip)

  Creates a new session.

- boolean closeSession (int sessionID)

  Adds logout time to a given session.
- User loginUser (String userID, String pw)

  Tries to login with a given userID and password.
- boolean changeNickname (String userID, String newNick)

  Changes the nickname of a user.
- boolean createUser (String userID, String nick, String pw, Right r)

  Creates a new user.
- List< User > getAllUser ()

  Gets all user from the database .
- int getAllUserCount ()

  Gets the number of all user in the database.
- String getUserIdBySession (int sessionID)

  Looks up a userID by a given sessionID.
- User getUserBySession (int sessionID)

  Gets a user object based on an given session ID.
- void dbTest ()

  Database self test.

Protected Member Functions

- boolean logChat (String userID, int sessionID, String text)

  Saves a chat text to database.
- boolean logGPS (int driveID, String latitude, String longitude)

  Logs GPS coordinates.
- int logQueue (String userID, int sessionID)

3.6.1 Detailed Description

This Class implements all functions which interact the database.

The constructor DBConnector(LogNG logIN, Options opt) is needed for first time initialization.

Author

Michael Röding

3.6.2 Constructor & Destructor Documentation

3.6.2.1 de.carduinodroid.utilities.DBConnector.DBConnector (LogNG *logIN*, Options *opt*) `[inline]`

Initializes database connection and sets static variables.

This constructor sets or updates the static log and options. It also calls connect() is necessary.

Parameters

logIN	Log which should be used for logging.
opt	Options which contains database connection parameters (server address, username and password).

3.6.2.2 de.carduinodroid.utilities.DBConnector.DBConnector ( ) throws Exception [inline]

This constructor checks if all necessary variables are set.

Exceptions

Exception	This constructor should never be used for first time instancing since log and options are not set

3.6.3 Member Function Documentation

3.6.3.1 boolean de.carduinodroid.utilities.DBConnector.changeNickname (String *userID*, String *newNick*) [inline]

Changes the nickname of a user.

Parameters

userID	UserID to change the nickname of.
newNick	new nickname.

Returns

Returns "true" if the SessionID was found or "false" if not.

3.6.3.2 boolean de.carduinodroid.utilities.DBConnector.closeSession (int *sessionID*) [inline]

Adds logout time to a given session.

Parameters

sessionID	Session to close.

Returns

Returns "true" if successful or "false" if an error occurs.

3.6.3.3 int de.carduinodroid.utilities.DBConnector.createSession (String *userID*, String *ip*) [inline]

Creates a new session.

Creates a database entry for the given userID and IP.

Parameters

userID	User who creates a new session.
ip	address IP of the user.

Returns

SessionID assigned by the database.

3.6.3.4 boolean de.carduinodroid.utilities.DBConnector.createUser (String *userID*, String *nick*, String *pw*, Right *r*) [inline]

Creates a new user.

Parameters

userID	UserID to login with.
nickname	Nickname to display. (can be null)
password	Password for the new user.
right	Right the new user has.

Returns

Returns "true" if the SessionID was found or "false" if not.

3.6.3.5 void de.carduinodroid.utilities.DBConnector.dbTest ( ) [inline]

Database self test.

Testing most function of this class. A test user is reuquired -> username: "test" password: "test456". This functions adds database entries and do not delete all of them! testet der Verbindung Erwartung: Verbindung ist ok

3.6.3.6 List<User> de.carduinodroid.utilities.DBConnector.getAllUser ( ) [inline]

Gets all user from the database .

Returns

Returns all users in the database in an ArrayList.

3.6.3.7 int de.carduinodroid.utilities.DBConnector.getAllUserCount ( ) [inline]

Gets the number of all user in the database.

Returns

Number of users.

3.6.3.8 List<GPS> de.carduinodroid.utilities.DBConnector.getGPSByDriveID (int *driveID*) [inline]

Gets all GPS records for one drive.

Parameters

driveID	DriveID to look up GPS records for.

Returns

ArrayList with GPS records.

3.6.3.9 List<GPS> de.carduinodroid.utilities.DBConnector.getGPSByTime (Timestamp *begin*, Timestamp *end*) [inline]

Gets all GPS records between a given time frame.

Parameters

begin	Time frame begin.
end	Time frame end.

Returns

ArrayList with GPS records.

3.6.3.10 User de.carduinodroid.utilities.DBConnector.getUserBySession (int *sessionID*) `[inline]`

Gets a user object based on an given session ID.

Similar to getUserIdBySession(int sessionID).

Parameters

sessionID	SessionID to search for.

Returns

Returns an User-object associated with the session or null if no session was found.

3.6.3.11 String de.carduinodroid.utilities.DBConnector.getUserIdBySession (int *sessionID*) `[inline]`

Looks up a userID by a given sessionID.

Similar to getUserBySession(int sessionID).

Parameters

sessionID	SessionID to search for.

Returns

UserID associated with the session or null if no session was found.

3.6.3.12 boolean de.carduinodroid.utilities.DBConnector.logChat (String *userID,* int *sessionID,* String *text*) `[inline]`, `[protected]`

Saves a chat text to database.

The current timestamp is added by this function.

Parameters

userID	The UserID of the user who sad that.
sessionID	The associated SessionID.
text	The actual chat text (max length is 256!).

Returns

Returns "true" if successful or "false" if an error occurs.

3.6.3.13 boolean de.carduinodroid.utilities.DBConnector.logGPS (int *driveID,* String *latitude,* String *longitude*) `[inline]`, `[protected]`

Logs GPS coordinates.

Save lat, long and the current timestamp in the database.

Parameters

driveID	DriveID to search GPS coordinates to.
longitude	Longitude
latitude	Latitude

Returns

Returns "true" if successful or "false" if an error occurs.

3.6.3.14 User de.carduinodroid.utilities.DBConnector.loginUser (String *userID*, String *pw*) `[inline]`

Tries to login with a given userID and password.

Parameters

userID	UserID to login with.
password	Password to login with.

Returns

Returns the user as a User object or null if the login was invalid.

3.6.3.15 int de.carduinodroid.utilities.DBConnector.logQueue (String *userID*, int *sessionID*) `[inline]`, `[protected]`

Logs a user who enqueued himself.

Parameters

userID	UserID from the user who is enqueued.
sessionID	SessionID from the current session.

Returns

QueueID assigned by the database.

3.6.3.16 int de.carduinodroid.utilities.DBConnector.startDrive (String *userID*) `[inline]`

Creates a new drive.

Create a database entry for a given userID and sets the current timestamp.

Parameters

userID	UserID of the driver.

Returns

DriverID assigned by the database or -1 if an error occurs.

3.6.3.17 boolean de.carduinodroid.utilities.DBConnector.stopDrive (int *driveID*) `[inline]`

Adds stop time timestamp.

End a drive by inserting the current timestamp in the database.

Parameters

driveID	DriverID of the current driver.

Returns

Returns "true" if successful or "false" if an error occurs.

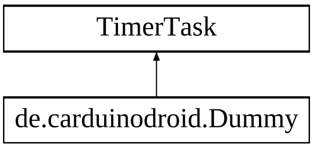
The documentation for this class was generated from the following file:

- src/de/carduinodroid/utilities/DBConnector.java

3.7 de.carduinodroid.Dummy Class Reference

This Class is used to start TimerTasks from the inside of the same TimerTask.

Inheritance diagram for de.carduinodroid.Dummy:



Public Member Functions

- **Dummy** (TimerTask tt)
- void **run** ()

3.7.1 Detailed Description

This Class is used to start TimerTasks from the inside of the same TimerTask.

Author

Alexander Rose

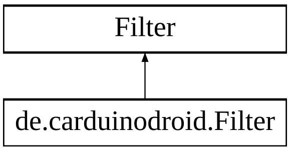
The documentation for this class was generated from the following file:

- src/de/carduinodroid/Dummy.java

3.8 de.carduinodroid.Filter Class Reference

This Class is used to catch ServletRequest e.g. Login,Logout or enqueue into the waiting queue.

Inheritance diagram for de.carduinodroid.Filter:



Public Member Functions

- void doFilter (ServletRequest request, ServletResponse res, FilterChain chain) throws IOException, Servlet-Exception
- void init (FilterConfig fConfig) throws ServletException
- void destroy ()

3.8.1 Detailed Description

This Class is used to catch ServletRequest e.g. Login,Logout or enqueue into the waiting queue.

Servlet Filter implementation class Filter

Author

Alexander Rose, wer noch?

3.8.2 Member Function Documentation

3.8.2.1 void de.carduinodroid.Filter.destroy ( ) [inline]

See Also

Filter::destroy()

3.8.2.2 void de.carduinodroid.Filter.doFilter (ServletRequest *request*, ServletResponse *res*, FilterChain *chain*) throws IOException, ServletException [inline]

See Also

Filter::doFilter(ServletRequest, ServletResponse, FilterChain)

3.8.2.3 void de.carduinodroid.Filter.init (FilterConfig *fConfig*) throws ServletException [inline]

See Also

Filter::init(FilterConfig)

The documentation for this class was generated from the following file:

- src/de/carduinodroid/Filter.java

3.9 de.carduinodroid.shared.GPS Class Reference

Public Member Functions

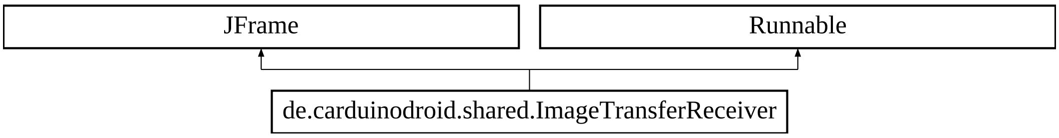
- **GPS** (int driveID, String longi, String lati, Timestamp time)
- String **getLongitude** ()
- String **getLatitude** ()
- int **getDriveID** ()
- Timestamp **getDateTime** ()
- String **getDateTimeFormated** ()

The documentation for this class was generated from the following file:

- src/de/carduinodroid/shared/GPS.java

3.10 de.carduinodroid.shared.ImageTransferReceiver Class Reference

Inheritance diagram for de.carduinodroid.shared.ImageTransferReceiver:



Public Member Functions

- void **readBytes** (final InputStream is, final byte[] buffer, int l) throws IOException
- void **run** ()

Static Public Member Functions

- static void **main** (final String[] args)

The documentation for this class was generated from the following file:

- src/de/carduinodroid/shared/ImageTransferReceiver.java

3.11 de.carduinodroid.shared.ImageTransferSender Class Reference

Static Public Member Functions

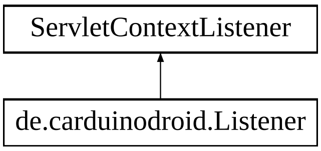
- static void **main** (String[] args) throws IOException, InterruptedException

The documentation for this class was generated from the following file:

- src/de/carduinodroid/shared/ImageTransferSender.java

3.12 de.carduinodroid.Listener Class Reference

Inheritance diagram for de.carduinodroid.Listener:



Public Member Functions

- void contextInitialized (ServletContextEvent arg0)
- void contextDestroyed (ServletContextEvent arg0)

3.12.1 Detailed Description

Application Lifecycle Listener implementation class Listener

3.12.2 Member Function Documentation

3.12.2.1 void de.carduinodroid.Listener.contextDestroyed (ServletContextEvent *arg0*) [inline]

See Also

ServletContextListener::contextDestroyed(ServletContextEvent)

3.12.2.2 void de.carduinodroid.Listener.contextInitialized (ServletContextEvent *arg0*) [inline]

See Also

ServletContextListener::contextInitialized(ServletContextEvent)

The documentation for this class was generated from the following file:

- src/de/carduinodroid/Listener.java

3.13 de.carduinodroid.utilities.LogNG Class Reference

Log class for logging everything.

Public Member Functions

- boolean logChat (String userID, int sessionID, String text)

  Logs chat to database or file depending on settings.
- boolean logGPS (int driveID, String longitude, String latitude)

  Logs GPS coordinates to database or file depending on settings.
- boolean logGPS (GPS gps)

  Logs GPS coordinates to database or file depending on settings.
- int logQueue (String userID, int sessionID)

  Logs enqueue event to database or file depending on settings.
- void writelogfile (String string)

  General purpose log function.
- void setOptions (Options options)

  Sets the options.
- void setDB (DBConnector database)

  Sets the database to use for logging.
- LinkedList< String > **getLog** ()

3.13.1 Detailed Description

Log class for logging everything.

This class logs everything to a file or to the database. The log is instanced before everything else. Therefore it has no options in the beginning and doesn't know where to save the log file. The solution to this is to save everything in a list and copy the list to the log file as soon as the options are set.

Author

Michael Röding

3.13.2 Member Function Documentation

3.13.2.1 boolean de.carduinodroid.utilities.LogNG.logChat (String *userID,* int *sessionID,* String *text*) [inline]

Logs chat to database or file depending on settings.

Parameters

userID	The UserID of the user who said that.
sessionID	The associated SessionID.
text	The actual chat text (max length is 256!).

Returns

Returns "true" if successful or "false" if an error occurs.

3.13.2.2 boolean de.carduinodroid.utilities.LogNG.logGPS (int *driveID,* String *longitude,* String *latitude*) [inline]

Logs GPS coordinates to database or file depending on settings.

Parameters

driveID	DriveID to search GPS coordinates to.
longitude	Longitude
latitude	Latitude

Returns

Returns "true" if successful or "false" if an error occurs.

3.13.2.3 boolean de.carduinodroid.utilities.LogNG.logGPS (GPS *gps*) [inline]

Logs GPS coordinates to database or file depending on settings.

Parameters

gps	GPS coordinates to log.

Returns

Returns "true" if successful or "false" if an error occurs.

3.13.2.4 int de.carduinodroid.utilities.LogNG.logQueue (String *userID,* int *sessionID*) [inline]

Logs enqueue event to database or file depending on settings.

Parameters

userID	UserID from the user who is enqueued.
sessionID	SessionID from the current session.

Returns

QueueID assigned by the database.

3.13.2.5 void de.carduinodroid.utilities.LogNG.setDB (**DBConnector** *database*) [inline]

Sets the database to use for logging.

Parameters

database	The database to set.

3.13.2.6 void de.carduinodroid.utilities.LogNG.setOptions (**Options** *options*) [inline]

Sets the options.

Parameters

options	The options to set.

3.13.2.7 void de.carduinodroid.utilities.LogNG.writelogfile (String *string*) [inline]

General purpose log function.

The current timestamp will be added last.

Parameters

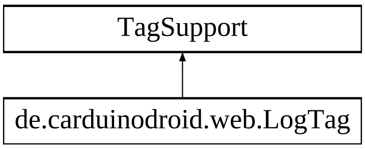
string	Text to save to the log.

The documentation for this class was generated from the following file:

- src/de/carduinodroid/utilities/LogNG.java

3.14 de.carduinodroid.web.LogTag Class Reference

Inheritance diagram for de.carduinodroid.web.LogTag:



Public Member Functions

- int **doStartTag** () throws JspException
- int **doEndTag** () throws JspException

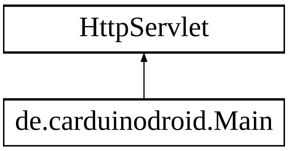
The documentation for this class was generated from the following file:

- src/de/carduinodroid/web/LogTag.java

3.15 de.carduinodroid.Main Class Reference

This Class is used to handle user-timeouts and to log GPS in a given interval and to handle the waiting queue.

Inheritance diagram for de.carduinodroid.Main:



Public Member Functions

- Main ()

Static Public Member Functions

- static void refresh (Options opt)
  refreshes the options needed in this class (Fahrzeit and GPS interval)
- static void shutDown ()
  deletes all Timers and TimerTasks and also deletes all active sessions
- static void main (Options opt, DBConnector db, LogNG logng)

Protected Member Functions

- void doGet (HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
- void doPost (HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException

3.15.1 Detailed Description

This Class is used to handle user-timeouts and to log GPS in a given interval and to handle the waiting queue.

Servlet implementation class Main

Author

Alexander Rose

3.15.2 Constructor & Destructor Documentation

3.15.2.1 de.carduinodroid.Main.Main ( ) [inline]

See Also

HttpServlet::HttpServlet()

3.15.3 Member Function Documentation

3.15.3.1 void de.carduinodroid.Main.doGet (HttpServletRequest *request,* HttpServletResponse *response*) throws ServletException, IOException [inline], [protected]

See Also

HttpServlet::doGet(HttpServletRequest request, HttpServletResponse response)

3.15.3.2 void de.carduinodroid.Main.doPost (HttpServletRequest *request,* HttpServletResponse *response*) throws ServletException, IOException [inline], [protected]

See Also

HttpServlet::doPost(HttpServletRequest request, HttpServletResponse response)

3.15.3.3 static void de.carduinodroid.Main.main (**Options** *opt,* **DBConnector** *db,* **LogNG** *logng*) [inline], [static]

3.15.3.4 static void de.carduinodroid.Main.refresh (**Options** *opt*) [inline], [static]

refreshes the options needed in this class (Fahrzeit and GPS interval)

Parameters

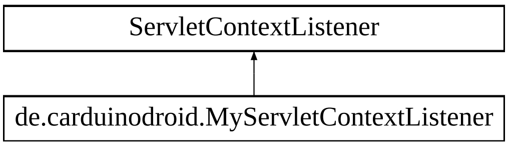
opt	Options object

The documentation for this class was generated from the following file:

- src/de/carduinodroid/Main.java

3.16 de.carduinodroid.MyServletContextListener Class Reference

Inheritance diagram for de.carduinodroid.MyServletContextListener:



Public Member Functions

- void **contextDestroyed** (ServletContextEvent event)
- void **contextInitialized** (ServletContextEvent event)

The documentation for this class was generated from the following file:

- src/de/carduinodroid/MyServletContextListener.java

3.17 de.carduinodroid.utilities.Config.Options Class Reference

Actual options.

Public Attributes

- int **fahrZeit**
- String dbAddress
- String dbUser
- String dbPW
- String filePath
- boolean logChat
- boolean logChatToFile
- boolean logGPS
- boolean logGPSToFile
- int logGPSInterval
- boolean logQueue
- boolean logQueueToFile

3.17.1 Detailed Description

Actual options.

This class contains all available options.

Author

Michael Röding

3.17.2 Member Data Documentation

3.17.2.1 String de.carduinodroid.utilities.Config.Options.dbAddress

in minutes

3.17.2.2 String de.carduinodroid.utilities.Config.Options.dbPW

database username

3.17.2.3 String de.carduinodroid.utilities.Config.Options.dbUser

Internet address of the database server

3.17.2.4 String de.carduinodroid.utilities.Config.Options.filePath

database password

3.17.2.5 boolean de.carduinodroid.utilities.Config.Options.logChat

file path which should be used as root folder for everything else (like log and options)

3.17.2.6 boolean de.carduinodroid.utilities.Config.Options.logChatToFile

should the chat be logged

3.17.2.7 boolean de.carduinodroid.utilities.Config.Options.logGPS

should the chat be logged to file

3.17.2.8 int de.carduinodroid.utilities.Config.Options.logGPSInterval

should gps be logged to file

3.17.2.9 boolean de.carduinodroid.utilities.Config.Options.logGPSToFile

should gps be logged

3.17.2.10 boolean de.carduinodroid.utilities.Config.Options.logQueue

in seconds

3.17.2.11 boolean de.carduinodroid.utilities.Config.Options.logQueueToFile

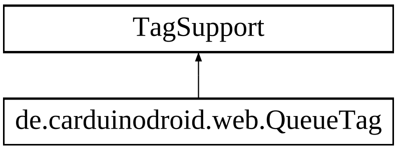
should queue events be logged

The documentation for this class was generated from the following file:

- src/de/carduinodroid/utilities/Config.java

3.18 de.carduinodroid.web.QueueTag Class Reference

Inheritance diagram for de.carduinodroid.web.QueueTag:



Public Member Functions

- int **doStartTag** () throws JspException
- int **doEndTag** () throws JspException

The documentation for this class was generated from the following file:

- src/de/carduinodroid/web/QueueTag.java

3.19 de.carduinodroid.shared.User.Right Enum Reference

Public Attributes

- **GUEST**
- **USER**

The documentation for this enum was generated from the following file:

- src/de/carduinodroid/shared/User.java

3.20 de.carduinodroid.shared.Session Class Reference

Public Member Functions

- **Session** (String id, Inet4Address ipadress, Timestamp Login)
- String **getSessionId** ()
- Inet4Address **getIP** ()
- Timestamp **getLoginTme** ()

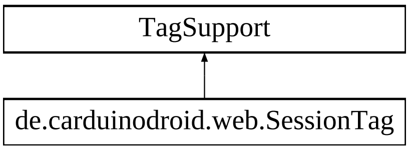
The documentation for this class was generated from the following file:

- src/de/carduinodroid/shared/Session.java

3.21 de.carduinodroid.web.SessionTag Class Reference

This class handles the Session related custom tags.

Inheritance diagram for de.carduinodroid.web.SessionTag:



Public Member Functions

- int doStartTag () throws JspException
  Starting tag handler.
- int **doEndTag** () throws JspException

3.21.1 Detailed Description

This class handles the Session related custom tags.

Currently there is only the isAdmin tag implemented but this can easily be expanded

Author

Christoph Braun

3.21.2 Member Function Documentation

3.21.2.1 int de.carduinodroid.web.SessionTag.doStartTag ( ) throws JspException [inline]

Starting tag handler.

Returns

EVAL_PAGE. Tells the Server to continue evaluating Tags.

Exceptions

Exception	Throws a default Jsp Exception with full stacktrace.

The documentation for this class was generated from the following file:

- src/de/carduinodroid/web/SessionTag.java

3.22 de.carduinodroid.shared.User Class Reference

Classes

- enum Right

Public Member Functions

- **User** (String id, Right r)
- **User** (String id, String nick, Right r)
- boolean **isAdmin** ()
- boolean **isUser** ()

- boolean **isGuest** ()
- String getUserID ()
- String getNickname ()
- void setNickname (String userNickname)

3.22.1 Member Function Documentation

3.22.1.1 String de.carduinodroid.shared.User.getNickname ( ) [inline]

Returns

the userNickname

3.22.1.2 String de.carduinodroid.shared.User.getUserID ( ) [inline]

Returns

the userID

3.22.1.3 void de.carduinodroid.shared.User.setNickname (String *userNickname*) [inline]

Parameters

userNickname	the userNickname to set

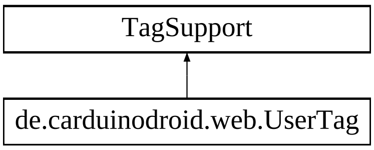
The documentation for this class was generated from the following file:

- src/de/carduinodroid/shared/User.java

3.23 de.carduinodroid.web.UserTag Class Reference

This class handles the User related custom tags.

Inheritance diagram for de.carduinodroid.web.UserTag:



Public Member Functions

- void setNum (int i)
  Setter methods.
- void **setPar** (int o)
- int doStartTag () throws JspException
  Starting tag handler.
- int **doEndTag** () throws JspException

3.23.1 Detailed Description

This class handles the User related custom tags.

The User tag returns User data (Nickname,Rights,UserCount).

Author

Christoph Braun

3.23.2 Member Function Documentation

3.23.2.1 int de.carduinodroid.web.UserTag.doStartTag ( ) throws JspException [inline]

Starting tag handler.

This tag has 3 Functions.

1. Get how many Users are registered.
2. Get the Nickname of a specific User.
3. Check if the User is an admin.

Returns

EVAL_PAGE. Tells the Server to continue evaluating Tags.

Exceptions

Exception	Throws a default Jsp Exception with full stacktrace.

3.23.2.2 void de.carduinodroid.web.UserTag.setNum (int *i*) [inline]

Setter methods.

Parameters

i	Number in UserList
o	Parameter on which function to call

The documentation for this class was generated from the following file:

- src/de/carduinodroid/web/UserTag.java

3.24 de.carduinodroid.shared.waitingqueue Class Reference

This Class is used to queue all users who want to get driver-rights.

Static Public Member Functions

- static void initqueue (DBConnector db2)
  creates the internal waitingqueue and receives Connection to DB
- static LinkedList< String > getqueue ()
  returns the waiting queue
- static String getNextUser ()
  returns the first User from the Queue and removes the User from the Queue
- static void insertUser (String SessionID)
  inserts a new User at the end of the waiting queue
- static int getLength ()
- static boolean isEmpty ()
- static void deleteTicket (String SessionID)
  removes a User from the waiting queue
- static void InsertFirst (String SessionID)
  inserts a User at the first position of the waiting queue
- static String[] getNickname ()
- static String[] getAllSessions ()

3.24.1 Detailed Description

This Class is used to queue all users who want to get driver-rights.

Author

Alexander Rose

3.24.2 Member Function Documentation

3.24.2.1 static void de.carduinodroid.shared.waitingqueue.deleteTicket (String *SessionID*) [inline], [static]

removes a User from the waiting queue

Parameters

SessionID	of the User

3.24.2.2 static String [] de.carduinodroid.shared.waitingqueue.getAllSessions ( ) [inline], [static]

Returns

Returns the UserID from every User in the waiting queue in the right order

3.24.2.3 static int de.carduinodroid.shared.waitingqueue.getLength ( ) [inline], [static]

Returns

returns the Length of the waiting queue

3.24.2.4 static String de.carduinodroid.shared.waitingqueue.getNextUser ( ) [inline], [static]

returns the first User from the Queue and removes the User from the Queue

Returns

returns the UserID from the first User as String

3.24.2.5 static String [] de.carduinodroid.shared.waitingqueue.getNickname ( ) [inline], [static]

Returns

Returns the nicknames of all users in the waiting queue in the right order

3.24.2.6 static LinkedList<String> de.carduinodroid.shared.waitingqueue.getqueue ( ) [inline], [static]

returns the waiting queue

Returns

returns a LinkedList of Strings

3.24.2.7 static void de.carduinodroid.shared.waitingqueue.initqueue (DBConnector *db2*) [inline], [static]

creates the internal waitingqueue and receives Connection to DB

Parameters

db2	DBConnector to our DB

3.24.2.8 static void de.carduinodroid.shared.waitingqueue.InsertFirst (String *SessionID*) [inline], [static]

inserts a User at the first position of the waiting queue

Parameters

SessionID	of the User

3.24.2.9 static void de.carduinodroid.shared.waitingqueue.insertUser (String *SessionID*) [inline], [static]

inserts a new User at the end of the waiting queue

Parameters

SessionID	from the User

3.24.2.10 static boolean de.carduinodroid.shared.waitingqueue.isEmpty ( ) [inline], [static]

Returns

Returns true if waiting queue is empty and false if its not empty

The documentation for this class was generated from the following file:

- src/de/carduinodroid/shared/waitingqueue.java

Index