Программа для ЭВМ
Румхаус

Исходный текст

Авторы:
Шевляков Александр Михайлович
Елизаров Михаил Львович

г. Москва, 2025

Оглавление

1. Серверная часть 3
1.1. Room.java 3
1.2. CallHandler.java 19
1.3. UserSession.java 31
1.4. UserRegistry.java 36
1.5. RoomManager.java 37
1.6. GroupCallApp.java 38
1.7. GeoLite.java 39

2. Клиентская часть 41
2.1. conferenceroom_144a.js 41
2.2. participant_139a.js 102
2.3. starter_a.js 133
2.4. finisher_a.js 155
2.5. i18n_120.js 157

1. Серверная часть

1.1. Room.java

```
package org.kurento.roomhouse;

import java.io.Closeable;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.List;
import java.util.concurrent.ConcurrentHashMap;
import java.util.concurrent.ConcurrentMap;

import javax.annotation.PreDestroy;
import javax.servlet.http.HttpServletRequest;

import org.kurento.client.Continuation;
import org.kurento.client.MediaPipeline;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.web.socket.WebSocketSession;

import com.google.gson.JsonArray;
import com.google.gson.JsonElement;
import com.google.gson.JsonObject;
import com.google.gson.JsonPrimitive;

import java.time.format.DateTimeFormatter;
import java.time.LocalDateTime;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;

public class Room implements Closeable {
  private final Logger log = LoggerFactory.getLogger(Room.class);

  private final ConcurrentMap<String, UserSession> participants = new
ConcurrentHashMap<>();
  private final ConcurrentMap<String, UserSession> viewers = new
ConcurrentHashMap<>();
  private final MediaPipeline pipeline;
  private final String name;

  public String getName() {
    return name;
  }

  public Room(String roomName, MediaPipeline pipeline) {
    final String pgurl = "jdbc:postgresql://localhost:5432/cp";
    final String pguser = "postgres";
    final String pgpass = "x";

    this.name = roomName;
    this.pipeline = pipeline;
    log.info("ROOM {} has been created", roomName);
    try (Connection con = DriverManager.getConnection(pgurl, pguser, pgpass);
    Statement st = con.createStatement();
```

```
    ResultSet rs = st.executeQuery("UPDATE rooms SET movie_name='',
movie_player='', dtm=current_timestamp WHERE name='" + roomName + "'")) {
    } catch (SQLException ex) {log.debug("PG update err for room {}",
roomName);}
  }

  @PreDestroy
  private void shutdown() {
    this.close();
  }

  public UserSession join(String userName, String userMode, String userCurip,
String userAccId, WebSocketSession session, String roleId, int num_guests, int
room_limit, int num_rooms, String currRoom, String Roomees) throws IOException {

    final String empty_anno = "";
    final UserSession participant = new UserSession(userName, userMode, "a",
userCurip, userAccId, roleId, empty_anno, currRoom, this.name, session,
this.pipeline);


    if (roleId.equals("1") || roleId.equals("2") || roleId.equals("3")) {
      log.info("ROOM {}: adding participant {}", this.name, userName);
      joinRoom(participant, num_guests, room_limit, currRoom);
      participants.put(participant.getName(), participant);
    } else {

      //limit num of viewers hard way

      if(viewers.size() < 100) {
            log.info("ROOM {}: adding viewer {}", this.name, userName);
            viewers.put(participant.getName(), participant);
            viewRoom(participant, num_guests, room_limit, currRoom);
      } else {
            log.info("ROOM {}: reached limit of viewers {}", this.name,
viewers.size());
      }
    }

    sendParticipantNamesModesCuripsAccIdsAnnos(participant, num_rooms, currRoom,
Roomees);
    sendViewerNamesCurips(participant, num_guests, room_limit, currRoom);

    return participant;
  }

  public void leave(UserSession user) throws IOException {

    String isViewer = "false";

    if (user != null) {
      log.debug("SOMEONE {}: Leaving room {}", user.getName(), this.name);

      for (final UserSession viewer : this.getViewers()) {
             if (viewer.equals(user)) {
            isViewer = "true";
              }
      }

      if (isViewer.equals("false")) {
            this.removeParticipant(user.getName());
      } else {
```

```
            viewers.remove(user.getName());
            final JsonObject viewerLeftJson = new JsonObject();
            viewerLeftJson.addProperty("id", "viewerLeft");
            viewerLeftJson.addProperty("name", user.getName());
            for (final UserSession participant : participants.values()) {
                  try {
                        participant.sendMessage(viewerLeftJson);
                  } catch (final IOException e) {
                        log.debug("ROOM {}: Participant {} could not be notified
on viewer {} leaving room", this.name, participant, user.getName());
                  }
            }
            for (final UserSession viewer : viewers.values()) {
                  try {
                        viewer.sendMessage(viewerLeftJson);
                  } catch (final IOException e) {
                        log.debug("ROOM {}: Viewer {} could not be notified on
viewer {} leaving room", this.name, viewer, user.getName());
                  }
            }
      }

      user.close();
    }
  }

  public void make_leave_room(UserSession user, String who) throws IOException {
    if (user != null) {
      final List<String> unnotifiedParticipants = new ArrayList<>();
      final JsonObject makeLeaveJson = new JsonObject();
      makeLeaveJson.addProperty("id", "clearCoo");

      for (final UserSession participant : participants.values()) {
        try {
            if (who.equals("all")) {
                  if (user.getName() != participant.getName())
participant.sendMessage(makeLeaveJson);
            } else {
                  if (who.equals(participant.getName()))
participant.sendMessage(makeLeaveJson);
            }
        } catch (final IOException e) {
            unnotifiedParticipants.add(participant.getName());
        }
      }

      for (final UserSession viewer : viewers.values()) {
        try {
            if (user.getName() != viewer.getName())
viewer.sendMessage(makeLeaveJson);
        } catch (final IOException e) {
            unnotifiedParticipants.add(viewer.getName());
        }
      }

      if (!unnotifiedParticipants.isEmpty()) {
            log.debug("ROOM {}: The users {} could not be made to leave room",
this.name, unnotifiedParticipants);
      }

    }
  }
```

```
  public void set_asking_mode(UserSession user) throws IOException {
    if (user != null) {
      final List<String> unnotifiedParticipants = new ArrayList<>();
      final JsonObject askGuruJson = new JsonObject();
      askGuruJson.addProperty("id", "askGuru");
      askGuruJson.addProperty("name", user.getName());

      for (final UserSession participant : participants.values()) {
        try {
            if (user.getName() != participant.getName())
participant.sendMessage(askGuruJson);
        } catch (final IOException e) {
            unnotifiedParticipants.add(participant.getName());
        }
      }

      for (final UserSession viewer : viewers.values()) {
        try {
                  viewer.sendMessage(askGuruJson);
        } catch (final IOException e) {
            unnotifiedParticipants.add(viewer.getName());
        }
      }

      if (!unnotifiedParticipants.isEmpty()) {
            log.debug("ROOM {}: The users {} could not be informed on asking
guru", this.name, unnotifiedParticipants);
      }

    }
  }
  public void reply_ping(UserSession user) throws IOException {

    if (user != null) {
      user.setAct("a");
    }
  }
  public void check_conn(UserSession user) throws IOException {

    if (user != null) {
      final String pgurl = "jdbc:postgresql://localhost:5432/cp";
      final String pguser = "postgres";
      final String pgpass = "x";

      // now check if room is closed

      String sta = "0";
      String roomees = "";

      try (Connection con = DriverManager.getConnection(pgurl, pguser, pgpass);
      Statement st = con.createStatement();
      ResultSet rs = st.executeQuery("select status, roomees from rooms where
name='" + this.name + "' and (current_timestamp < valid_till or valid_till is
null)")) {
      if (rs.next()) {if (rs.getString(1).equals("1")) {sta = "1";} else
{/*log.info("err1");*/} roomees = rs.getString(2);} else {log.info("err2");}
      } catch (SQLException ex) {log.debug("PG join err5 from {}: ",
user.getName());}

      String _role = user.getRole();
```

```
     if (sta.equals("1") && _role.equals("0")) {log.info("User {} locked off: ",
user.getName()); leave(user); return;} else {}

     user.setAct("a");
     final JsonObject checkConnJson = new JsonObject();
     checkConnJson.addProperty("id", "goodConnection");
     checkConnJson.addProperty("room", this.name);
     checkConnJson.addProperty("roomees", roomees);

     try {
          user.sendMessage(checkConnJson);
     } catch (final IOException e) {
          log.debug("ROOM {}: user {} could not be notified on good
connection", this.name, user.getName());
          // leave(user);
     }
     final JsonObject youLeft = new JsonObject();
     youLeft.addProperty("id", "youLeft");
     final JsonObject pingJson = new JsonObject();
     pingJson.addProperty("id", "pingConn");

     for (final UserSession participant : participants.values()) {
       try {
          if (user.getName() != participant.getName()) {
               String _rolee = participant.getRole();
               if (sta.equals("1") && _rolee.equals("0"))
{log.info("Participant {} locked off: ", participant.getName());
leave(participant); break;} else {}
               final String a = participant.getAct();
               if (a.equals("a")) {
                 participant.setAct("n");
                 participant.sendMessage(pingJson);
               } else {
                 log.info("ROOM {}: participant {} was ousted as inactive",
this.name, participant.getName());
                 leave(participant);
                 participant.sendMessage(youLeft);
               }
          }
       } catch (final IOException e) {
          //leave(participant);
          log.debug("ROOM {}: participant {} could not be pinged", this.name,
participant.getName());
       }
     }

     for (final UserSession viewer : viewers.values()) {
       try {
          if (user.getName() != viewer.getName()) {
               if (sta.equals("1")) {log.info("Viewer {} locked off: ",
viewer.getName()); leave(viewer); break;} else {}
               final String b = viewer.getAct();
               if (b.equals("a")) {
                 viewer.setAct("n");
                 viewer.sendMessage(pingJson);
               } else {
                 log.info("ROOM {}: viewer {} was ousted as inactive",
this.name, viewer.getName());
                 leave(viewer);
                 viewer.sendMessage(youLeft);
               }
          }
       } catch (final IOException e) {
```

```
            log.debug("ROOM {}: viewer {} could not be pinged", this.name,
viewer.getName());
        }
      }

    }
  }

  public void notify_tableau_change(UserSession user, String side, String num)
throws IOException {
    if (user != null) {
      final List<String> unnotifiedParticipants = new ArrayList<>();
      final JsonObject tableauChangeJson = new JsonObject();
      tableauChangeJson.addProperty("id", "changeTab");
      tableauChangeJson.addProperty("side", side);
      tableauChangeJson.addProperty("num", num);
      for (final UserSession participant : participants.values()) {
        try {
            if (user.getName() != participant.getName())
participant.sendMessage(tableauChangeJson);
        } catch (final IOException e) {
            unnotifiedParticipants.add(participant.getName());
        }
      }

      for (final UserSession viewer : viewers.values()) {
        try {
            if (user.getName() != viewer.getName())
viewer.sendMessage(tableauChangeJson);
        } catch (final IOException e) {
            unnotifiedParticipants.add(viewer.getName());
        }
      }

      if (!unnotifiedParticipants.isEmpty()) {
            log.debug("ROOM {}: The users {} could not be notified on tableau
change", this.name, unnotifiedParticipants);
      }

      String w = (side.equals("l")) ? "leftnum" : (side.equals("r")) ? "rightnum"
: "wrongnum";
      String p = (side.equals("l")) ? "MESSI" : (side.equals("r")) ? "RONALDO" :
"NONE";

      String[] cmdline1 = { "sh", "-c", "sed -E -i 's/
(id="+w+">)..*(<\\/span>)/\\1"+num+"\\2/g'
/home/nobody/vchat/target/classes/static/index.html"};

      Process pr1 = Runtime.getRuntime().exec(cmdline1);

      String[] cmdline2 = { "sh", "-c", "sed -E -i 's/
(id="+w+">)..*(<\\/span>)/\\1"+num+"\\2/g'
/home/nobody/vchat/src/main/resources/static/index.html"};

      Process pr2 = Runtime.getRuntime().exec(cmdline2);

      DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
      LocalDateTime now = LocalDateTime.now();

      String[] parts = user.getName().split("_");

      String[] copy = new String[parts.length - 1];
```

```
      for (int i = 0; i < parts.length-1; i++) {
           copy[i] = parts[i];
      }

      String str = String.join("_", copy);

      String loco = user.getCurip();

      String[] cmdline4 = { "sh", "-c", "sed -i '1s/^/"+str+" from "+loco+" voted
for "+p+" at "+dtf.format(now)+"<br><br>\\n/' /opt/nvme/ssd/nobody/log.html"};

      Process pr4 = Runtime.getRuntime().exec(cmdline4);

    }
  }

  public void notify_key_down(UserSession user, String num) throws IOException {
    if (user != null) {
      final List<String> unnotifiedParticipants = new ArrayList<>();
      final JsonObject keyDownJson = new JsonObject();
      keyDownJson.addProperty("id", "bongoKey");
      keyDownJson.addProperty("num", num);

      for (final UserSession participant : participants.values()) {
        try {
           if (user.getName() != participant.getName() && !
(user.getAccId().equals("") && participant.getMode().equals("c")))
participant.sendMessage(keyDownJson);
        } catch (final IOException e) {
           unnotifiedParticipants.add(participant.getName());
        }
      }

      for (final UserSession viewer : viewers.values()) {
        try {
           if (user.getName() != viewer.getName())
viewer.sendMessage(keyDownJson);
        } catch (final IOException e) {
           unnotifiedParticipants.add(viewer.getName());
        }
      }

      if (!unnotifiedParticipants.isEmpty()) {
           log.debug("ROOM {}: The users {} could not be notified on key down",
this.name, unnotifiedParticipants);
      }

    }
  }

  public void write_message_to_log(UserSession user, String mes) throws
IOException {
    if (user != null) {

      DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
      LocalDateTime now = LocalDateTime.now();

      String[] parts = user.getName().split("_");

      String[] copy = new String[parts.length - 1];
```

```
      for (int i = 0; i < parts.length-1; i++) {
           copy[i] = parts[i];
      }

      String str = String.join("_", copy);

      String loco = user.getCurip();

      if (mes.matches("^5[a-zA-Z0-9]{47}$")) {
        String[] cmdline3 = { "sh", "-c", "/home/nobody/check_eligible_drop.pl
"+user.getName()+" "+mes};

        Process pr3 = Runtime.getRuntime().exec(cmdline3);

        log.info("ROOM check_eligible_for_drop {}: running shell command {}",
name, cmdline3);

        final JsonObject newDrop = new JsonObject();
        newDrop.addProperty("id", "newDrop");
        newDrop.addProperty("user", user.getName());

        for (final UserSession participant : participants.values()) {
            try {
                        if (user.getName() != participant.getName())
participant.sendMessage(newDrop);
            } catch (final IOException e) {
                        log.debug("ROOM {}: participant {} could not be notified",
name, participant.getName());
            }
        }

      } else {

        //String[] cmdline4 = { "sh", "-c", "sed -i '1s/^/"+str+" from "+loco+"
wrote: <<<span class=mes>"+mes+"<\\/span>>> at "+dtf.format(now)
+"<br><br>\\n/' /var/www/html/cp/public_html/log.html"};
        String[] cmdline4 = { "sh", "-c", "sed -i '1s/^/"+str+" from "+loco+"
wrote: <<<span class=mes>"+mes+"<\\/span>>> at "+dtf.format(now)
+"<br><br>\\n/' /opt/nvme/ssd/nobody/log.html"};

        Process pr4 = Runtime.getRuntime().exec(cmdline4);

        log.info("ROOM write_message_to_log {}: running shell command {}", name,
cmdline4);

        final JsonObject newChatMessage = new JsonObject();
        newChatMessage.addProperty("id", "newChatMessage");

        for (final UserSession participant : participants.values()) {
            try {
                        if (user.getName() != participant.getName())
participant.sendMessage(newChatMessage);
            } catch (final IOException e) {
                        log.debug("ROOM {}: participant {} could not be notified",
name, participant.getName());
            }
        }

        for (final UserSession viewer : viewers.values()) {
            try {
```

```
                if (user.getName() != viewer.getName())
viewer.sendMessage(newChatMessage);
              } catch (final IOException e) {
                log.debug("ROOM {}: viewer {} could not be notified", name,
viewer.getName());
            }
          }
        }
      }
    }

  public void set_anno_to_text(UserSession user, String anno, String who_to)
throws IOException {

    if (user != null) {
      final JsonObject setAnno = new JsonObject();

      final String who_from = user.getName();

      String[] parts = user.getName().split("_");
      String[] copy = new String[parts.length - 1];

      for (int i = 0; i < parts.length-1; i++) {
           copy[i] = parts[i];
      }

      final String who_f = String.join("_", copy);

      String userName = who_to;
      String full_anno = anno;

      if (who_to.substring(0,Math.min(5, who_to.length())).equals("GURU:"))
{userName = who_from;}

      if (who_from.equals(who_to) || who_from.substring(0,Math.min(5,
who_to.length())).equals("GURU:") ) {} else {full_anno = who_f + ": " + anno;}

      setAnno.addProperty("id", "setAnno");
      setAnno.addProperty("participant", userName);
      setAnno.addProperty("anno", full_anno);

      for (final UserSession participant : participants.values()) {
           try {
                participant.sendMessage(setAnno);
                if (userName.equals(participant.getName()))
{participant.setAnno(anno);}
           } catch (final IOException e) {
                log.debug("ROOM {}: could not set anno for {}", name, userName);
           }
      }

      for (final UserSession viewer : viewers.values()) {
           try {
                viewer.sendMessage(setAnno);
           } catch (final IOException e) {
                log.debug("ROOM {}: could not push new anno of {} to {}", name,
userName, viewer.getName());
           }
      }
    }
  }
```

```
  public void set_movies_list_to_text(UserSession user, String ml, String
who_to, String cs) throws IOException {

    if (user != null) {
      final JsonObject setMoviesList = new JsonObject();

      String userName = who_to;

      setMoviesList.addProperty("id", "setMoviesList");
      setMoviesList.addProperty("participant", userName);
      setMoviesList.addProperty("ml", ml);
      setMoviesList.addProperty("cs", cs);

      for (final UserSession participant : participants.values()) {
            try {
                  participant.sendMessage(setMoviesList);
                  if (userName.equals(participant.getName()))
{participant.setAnno(ml+"@#%"+cs);}
            } catch (final IOException e) {
                  log.debug("ROOM {}: could not set anno for {}", name, userName);
            }
      }

      for (final UserSession viewer : viewers.values()) {
            try {
                  viewer.sendMessage(setMoviesList);
            } catch (final IOException e) {
                  log.debug("ROOM {}: could not push new anno of {} to {}", name,
userName, viewer.getName());
            }
      }
    }
  }

  public void set_guru_in_mode(UserSession user, String userName, String
userMode) throws IOException {
    if (user != null) {
      final JsonObject setGuru = new JsonObject();
      setGuru.addProperty("id", "setGuru");
      setGuru.addProperty("mode", userMode);

      if (userMode.equals("1")) {
            for (final UserSession viewer : viewers.values()) {
                  try {
                        if (userName.equals(viewer.getName()))
{viewer.sendMessage(setGuru); viewer.setRole("3");}
                  } catch (final IOException e) {
                        log.debug("ROOM {}: could not set as GURU viewer {}",
name, viewer.getName());
                  }
            }
      } else if (userMode.equals("2") || userMode.equals("3")) {
            for (final UserSession viewer : viewers.values()) {
                  try {
                        if (userName.equals(viewer.getName()))
{viewer.sendMessage(setGuru);}
                  } catch (final IOException e) {
                        log.debug("ROOM {}: could not set as GURU viewer {}",
name, viewer.getName());
                  }
            }
      } else if (userMode.equals("0")) {
            for (final UserSession participant : participants.values()) {
```

```
                try {
                     if (userName.equals(participant.getName()))
{participant.sendMessage(setGuru); participant.setRole("0");}
                } catch (final IOException e) {
                     log.debug("ROOM {}: could not unset as GURU viewer {}",
name, participant.getName());
                }
           }
      }
    }
  }

  public void request_movie(UserSession user, String name) throws IOException {
    if (user != null) {
      final String pgurl = "jdbc:postgresql://localhost:5432/cp";
      final String pguser = "postgres";
      final String pgpass = "x";

      String userName = "DUMMY";

      try (Connection con = DriverManager.getConnection(pgurl, pguser, pgpass);
      Statement st = con.createStatement();
      ResultSet rs = st.executeQuery("select movie_player from rooms where
name='" + this.name + "' and (current_timestamp < valid_till or valid_till is
null)")) {
      if (rs.next()) {userName = rs.getString(1);} else {log.info("PG select
movie_player err2");}
      } catch (SQLException ex) {log.debug("PG select movie_player err1 from {}:
", user.getName());}

      final JsonObject reqMovie = new JsonObject();

      reqMovie.addProperty("id", "requestFilm");
      reqMovie.addProperty("name", name);

      for (final UserSession participant : participants.values()) {
           try {
                if (userName.equals(participant.getName()) && !
participant.getMode().equals("c")) { participant.sendMessage(reqMovie); }
           } catch (final IOException e) {
                log.debug("ROOM {}: could not inform of Cinema {}", name,
participant.getName());
           }
      }


    }
  }

  public void deny_movie(UserSession user, int num) throws IOException {
    if (user != null) {
      final JsonObject denyMovie = new JsonObject();

      denyMovie.addProperty("id", "denyMovie");
      denyMovie.addProperty("n", num);
      user.sendMessage(denyMovie);
    }
  }

  public void set_cinema_in_mode(UserSession user, String userName, String
userMode) throws IOException {
    if (user != null) {
```

```
      final JsonObject setCinema = new JsonObject();

      setCinema.addProperty("id", "setCinema");
      setCinema.addProperty("name", userName);
      setCinema.addProperty("mode", userMode);

      for (final UserSession participant : participants.values()) {
            try {
                  if (userName.equals(participant.getName()))
{ participant.setMode(userMode);}
                   participant.sendMessage(setCinema);
            } catch (final IOException e) {
                  log.debug("ROOM {}: could not inform of Cinema {}", name,
participant.getName());
            }
      }

      for (final UserSession viewer : viewers.values()) {
            try {
                  viewer.sendMessage(setCinema);
            } catch (final IOException e) {
                  log.debug("ROOM {}: viewer {} could not be informed of Cinema",
name, viewer.getName());
            }
      }

    }
  }

  private Collection<String> joinRoom(UserSession newParticipant, int ng, int
rl, String currRoom) throws IOException {
    final JsonObject newParticipantMsg = new JsonObject();
    newParticipantMsg.addProperty("id", "newParticipantArrived");
    newParticipantMsg.addProperty("name", newParticipant.getName());
    newParticipantMsg.addProperty("mode", newParticipant.getMode());
    newParticipantMsg.addProperty("curip", newParticipant.getCurip());
    newParticipantMsg.addProperty("acc_id", newParticipant.getAccId());
    newParticipantMsg.addProperty("ng", ng);
    newParticipantMsg.addProperty("rl", rl);
    newParticipantMsg.addProperty("currRoom", currRoom);

    final List<String> participantsList = new
ArrayList<>(participants.values().size());
    log.debug("ROOM {}: notifying other participants of new participant {}",
name,
        newParticipant.getName());

    for (final UserSession participant : participants.values()) {
      try {
        participant.sendMessage(newParticipantMsg);
      } catch (final IOException e) {
        log.debug("ROOM {}: participant {} could not be notified", name,
participant.getName());
      }
      participantsList.add(participant.getName());
    }


    for (final UserSession viewer : viewers.values()) {
      try {
        viewer.sendMessage(newParticipantMsg);
      } catch (final IOException e) {
```

```
        log.debug("ROOM {}: viewer {} could not be notified", name,
viewer.getName());
      }
      participantsList.add(viewer.getName());
    }

    return participantsList;
  }

  private void viewRoom(UserSession newViewer, int ng, int rl, String currRoom)
throws IOException {
    final JsonObject newViewerMsg = new JsonObject();
    newViewerMsg.addProperty("id", "newViewerArrived");
    newViewerMsg.addProperty("name", newViewer.getName());
    newViewerMsg.addProperty("curip", newViewer.getCurip());
    newViewerMsg.addProperty("ng", ng);
    newViewerMsg.addProperty("rl", rl);
    newViewerMsg.addProperty("currRoom", currRoom);
    newViewerMsg.addProperty("accid", newViewer.getAccId());

    for (final UserSession participant : participants.values()) {
      try {
        participant.sendMessage(newViewerMsg);
      } catch (final IOException e) {
        log.debug("ROOM {}: participant {} could not be notified", name,
participant.getName());
      }
    }


    for (final UserSession viewer : viewers.values()) {
      try {
        viewer.sendMessage(newViewerMsg);
      } catch (final IOException e) {
        log.debug("ROOM {}: viewer {} could not be notified", name,
viewer.getName());
      }
    }

  }

  private void removeParticipant(String name) throws IOException {
    participants.remove(name);

    log.debug("ROOM {}: notifying all users that {} is leaving the room",
this.name, name);

    final List<String> unnotifiedParticipants = new ArrayList<>();
    final JsonObject participantLeftJson = new JsonObject();
    participantLeftJson.addProperty("id", "participantLeft");
    participantLeftJson.addProperty("name", name);

    for (final UserSession participant : participants.values()) {
      try {
        participant.cancelVideoFrom(name);
        participant.sendMessage(participantLeftJson);
      } catch (final IOException e) {
        unnotifiedParticipants.add(participant.getName());
      }
    }

    for (final UserSession viewer : viewers.values()) {
      try {
```

```
        viewer.cancelVideoFrom(name);
        viewer.sendMessage(participantLeftJson);
      } catch (final IOException e) {
        unnotifiedParticipants.add(viewer.getName());
      }
    }

    if (!unnotifiedParticipants.isEmpty()) {
      log.debug("ROOM {}: The users {} could not be notified that {} left the
room", this.name,
          unnotifiedParticipants, name);
    }

  }

  private void removeViewer(String name) throws IOException {

    viewers.remove(name);

    final JsonArray viewersArray = new JsonArray();
    for (final UserSession viewer : this.getViewers()) {

        final JsonElement viewerNameCurip = new JsonPrimitive(viewer.getName() +
"_|_" + viewer.getCurip());
        viewersArray.add(viewerNameCurip);

    }

    log.debug("ROOM {}: notifying all users that viewer {} is leaving the room",
this.name, name);

    final List<String> unnotifiedParticipants = new ArrayList<>();

    final JsonObject existingViewersMsg = new JsonObject();
    existingViewersMsg.addProperty("id", "existingViewers");
    existingViewersMsg.add("data", viewersArray);

    for (final UserSession participant : participants.values()) {
      try {
        participant.sendMessage(existingViewersMsg);
      } catch (final IOException e) {
        unnotifiedParticipants.add(participant.getName());
      }
    }

    for (final UserSession viewer : viewers.values()) {
      try {
        viewer.sendMessage(existingViewersMsg);
      } catch (final IOException e) {
        unnotifiedParticipants.add(viewer.getName());
      }
    }

    if (!unnotifiedParticipants.isEmpty()) {
      log.debug("ROOM {}: The users {} could not be notified that {} left the
room", this.name,
          unnotifiedParticipants, name);
    }

  }

  private static String getClientIp(HttpServletRequest request) {
```

```
        String remoteAddr = "";

        if (request != null) {
            remoteAddr = request.getHeader("X-FORWARDED-FOR");
            if (remoteAddr == null || "".equals(remoteAddr)) {
                remoteAddr = request.getRemoteAddr();
            }
        }

        return remoteAddr;
    }

    public void sendParticipantNamesModesCuripsAccIdsAnnos(UserSession user, int
num_rooms, String currRoom, String Roomees) throws IOException {

      final JsonArray participantsArray = new JsonArray();

      for (final UserSession participant : this.getParticipants()) {
        if (!participant.getCurrRoom().equals(currRoom)) continue;

          final JsonElement participantNameModeCuripAccIdAnno = new
JsonPrimitive(participant.getName() + "_|_" + participant.getMode() + "_|_" +
participant.getCurip() + "_|_" + participant.getAccId() + "_|_" +
participant.getAnno());

        String[] parts = participant.getName().split("_");

        String[] copy = new String[parts.length];

        for (int i = 1; i < parts.length; i++) {
             copy[i] = parts[i];
        }
        copy[0] = "DUMMY";

        String str = String.join("_", copy);

        final JsonElement participantNameModeCuripAccIdAnno_DUMMY = new
JsonPrimitive(str + "_|_" + participant.getMode() + "_|_" +
participant.getCurip() + "_|_" + participant.getAccId() + "_|_" +
participant.getAnno());
          if (user.getAccId().length() == 0 && participant.getMode().equals("c"))
{participantsArray.add(participantNameModeCuripAccIdAnno_DUMMY);} else
{participantsArray.add(participantNameModeCuripAccIdAnno);}

     }

      final JsonObject existingParticipantsMsg = new JsonObject();
      existingParticipantsMsg.addProperty("id", "existingParticipants");
      existingParticipantsMsg.add("data", participantsArray);
      existingParticipantsMsg.addProperty("num_rooms", num_rooms);
      existingParticipantsMsg.addProperty("roomees", Roomees);

      log.debug("SOMEONE {}: sending him a list of {} participants",
user.getName(),
          participantsArray.size());
      user.sendMessage(existingParticipantsMsg);
    }

    public void sendViewerNamesCurips(UserSession user, int ng, int rl, String
currRoom) throws IOException {
```

```
    final JsonArray viewersArray = new JsonArray();

    for (final UserSession viewer : this.getViewers()) {
      if (!viewer.getCurrRoom().equals(currRoom)) continue;

        final JsonElement viewerNameCurip = new JsonPrimitive(viewer.getName() +
"_|_" + viewer.getCurip());
        viewersArray.add(viewerNameCurip);

    }

    final JsonObject existingViewersMsg = new JsonObject();
    existingViewersMsg.addProperty("id", "existingViewers");
    existingViewersMsg.addProperty("ng", ng);
    existingViewersMsg.addProperty("rl", rl);
    existingViewersMsg.add("data", viewersArray);
    log.debug("SOMEONE {}: sending him a list of {} viewers", user.getName(),
        viewersArray.size());
    user.sendMessage(existingViewersMsg);
  }

  public Collection<UserSession> getParticipants() {
    return participants.values();
  }

  public Collection<UserSession> getViewers() {
    return viewers.values();
  }

  public UserSession getParticipant(String name) {
    return participants.get(name);
  }

  @Override
  public void close() {
    for (final UserSession user : participants.values()) {
      try {
        user.close();
      } catch (IOException e) {
        log.debug("ROOM {}: Could not invoke close on participant {}",
this.name, user.getName(),
            e);
      }
    }

    participants.clear();

    pipeline.release(new Continuation<Void>() {

      @Override
      public void onSuccess(Void result) throws Exception {
        log.trace("ROOM {}: Released Pipeline", Room.this.name);
      }

      @Override
      public void onError(Throwable cause) throws Exception {
        log.warn("PARTICIPANT {}: Could not release Pipeline", Room.this.name);
      }
    });

    log.debug("Room {} closed", this.name);
  }
```

```
}
```

1.2. CallHandler.java

```
package org.kurento.roomhouse;

import java.io.*;
import java.net.InetAddress;

import org.kurento.client.IceCandidate;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;
import org.springframework.web.socket.CloseStatus;
import org.springframework.web.socket.TextMessage;
import org.springframework.web.socket.WebSocketSession;
import org.springframework.web.socket.handler.TextWebSocketHandler;

import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.google.gson.JsonObject;

import com.maxmind.geoip2.DatabaseReader;
import com.maxmind.geoip2.exception.GeoIp2Exception;

import com.maxmind.geoip2.model.CityResponse;
import com.maxmind.geoip2.record.City;
import com.maxmind.geoip2.record.Country;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;


@Component
public class CallHandler extends TextWebSocketHandler {

  private static final Logger log = LoggerFactory.getLogger(CallHandler.class);

  private static final Gson gson = new GsonBuilder().create();

  @Autowired
  private RoomManager roomManager;

  @Autowired
  private UserRegistry registry;

  @Autowired
  private GeoLite geoLite;

  @Override
  public void handleTextMessage(WebSocketSession session, TextMessage message)
throws Exception {
    final JsonObject jsonMessage = gson.fromJson(message.getPayload(),
JsonObject.class);

    final UserSession user = registry.getBySession(session);
```

```
    final String pgurl = "jdbc:postgresql://localhost:5432/cp";
    final String pguser = "postgres";
    final String pgpass = "x";

    int room_limit = 0;
    int num_rooms = 0;
    String Roomees = "";

    if (user != null) {
      log.debug("Incoming message from user '{}': {}", user.getName(),
jsonMessage);
    } else {
    }

    switch (jsonMessage.get("id").getAsString()) {
      case "joinRoom":

                if (user == null) {log.info("Incoming message from new user:
{}", jsonMessage);}

           final String joinerName = jsonMessage.get("name").getAsString();
           final UserSession who = registry.getByName(joinerName);

           if (who != null) {
                if (user == null) {
                     leaveRoom(who);
                }
           }

           String joinerToken = jsonMessage.get("token").getAsString();
           String joinerRoom = jsonMessage.get("room").getAsString();
           String joinerHouse = jsonMessage.get("house").getAsString();
           String joinerRole = jsonMessage.get("role").getAsString();
           String joinerCurrRoom = jsonMessage.get("currRoom").getAsString();

           joinerRoom = joinerRoom.replaceAll("[;'\"]*", ""); //protect against
sql injection
           joinerRole = joinerRole.replaceAll("[;'\"]*", "");
           joinerCurrRoom = joinerCurrRoom.replaceAll("[;'\"]*", "");

           BufferedReader br = new BufferedReader(new
FileReader("/home/nobody/"+joinerRoom+"_room_limit"));
           try {
                StringBuilder sb = new StringBuilder();
                String line = br.readLine();

                while (line != null) {
                     sb.append(line);
                     sb.append(System.lineSeparator());
                     line = br.readLine();
                }

                room_limit = Integer.parseInt(sb.toString().replaceAll("\r",
"").replaceAll("\n", ""));
           } catch (FileNotFoundException fnfe) {
                fnfe.printStackTrace();
                } finally {
                br.close();
           }
```

```
          String role = "0"; String _role = "0"; String noSuchRoom = "0";
String temporary = "0";

// check if guru
          String cmd = "/home/nobody/a.sh";
          Runtime run = Runtime.getRuntime();
          Process pr = run.exec(cmd);
          pr.waitFor();
          BufferedReader buf = new BufferedReader(new
InputStreamReader(pr.getInputStream()));
          String line = "";
          while ((line=buf.readLine())!=null) {
               if (line.equals(joinerToken) && joinerToken.length() == 32)
{role = "1";}
          }
// then check if auth user

          if (role.equals("0")) {
               try (Connection con = DriverManager.getConnection(pgurl, pguser,
pgpass);
               Statement st = con.createStatement();
               ResultSet rs = st.executeQuery("select md5(concat(proj_code,
pass)) from members where proj_code != 'admin' and room = '" + joinerRoom +
"'")) {
                              while(rs.next()) {if
(rs.getString(1).equals(joinerToken)) {role = "2"; break;}}
               } catch (SQLException ex) {log.debug("PG join err1 from {}: ",
joinerName);}
          }
// now check if room is closed
          String sta = "0";
          try (Connection con = DriverManager.getConnection(pgurl, pguser,
pgpass);
               Statement st = con.createStatement();
          ResultSet rs = st.executeQuery("select status, num_rooms, roomees
from rooms where name='" + joinerRoom + "' and (current_timestamp < valid_till
or valid_till is null)")) {
                         if (rs.next()) {if (rs.getString(1).equals("1")) {sta =
"1";}; num_rooms = Integer.parseInt(rs.getString(2)); Roomees =
rs.getString(3);} else {noSuchRoom = "1";}
          } catch (SQLException ex) {log.debug("PG join err2 from {}: ",
joinerName);}


// now check if room is in the house

          try (Connection con = DriverManager.getConnection(pgurl, pguser,
pgpass);
               Statement st = con.createStatement();
          //ResultSet rs = st.executeQuery("select status from rooms where
name='" + joinerRoom + "' and (house='" + joinerHouse + "' or house is null)"))
{
          ResultSet rs = st.executeQuery("select status from rooms where
name='" + joinerRoom + "'")) {
                         if (rs.next()) {if (rs.getString(1).equals("1")) {sta =
"1";}} else {noSuchRoom = "1";}
          } catch (SQLException ex) {log.debug("PG join err3 from {}: ",
joinerName);}


// now make temp permissson if required

          if (user != null) {_role = user.getRole();}
```

```
           if (joinerRole.equals("3") && !role.equals("1") && _role.equals("3")
&& noSuchRoom.equals("0") ) { temporary = "1";}
// now check if we're to let join & count number of participants
                Room roo = roomManager.getRoom(joinerRoom);
                int cou = 0;
                for (final UserSession participant : roo.getParticipants()) {
                        cou++;
                }
// we allow room_limit joins in many-to-many
                if (role.equals("0") && cou < room_limit && sta.equals("0") &&
user != null ) { role = "1"; user.setRole("3"); }
           if (room_limit == 2 && cou >= room_limit) { noSuchRoom = "1"; }
           log.info("JOINER {}: SESSION {}, ROLE TOKEN {}, ROLE RECEIVED {},
ROOM STATUS {}, ROOM LIMIT {}", joinerName, session, role, joinerRole, sta,
room_limit);

           if ( (sta.equals("1") && role.equals("0") && temporary.equals("0") &&
_role.equals("0")) || noSuchRoom.equals("1") ) {
                log.info("ALARM1: joiner {} ", joinerName);
           } else {
                joinRoom(jsonMessage, session, room_limit, num_rooms, Roomees);
           }
         break;
       case "receiveVideoFrom":
         if (user != null) {
           final String senderName = jsonMessage.get("sender").getAsString();
           final UserSession sender = registry.getByName(senderName);
           final String sdpOffer = jsonMessage.get("sdpOffer").getAsString();
           log.info("SENDER {}: RECEIVED by USER {}", senderName,
user.getName());
           boolean cinema_ticket = true;
           if (sender != null && user.getAccId().length() == 0 &&
sender.getMode().equals("c")) cinema_ticket = false;
           if (sender != null && cinema_ticket) user.receiveVideoFrom(sender,
sdpOffer);
      }
         break;
       case "leaveRoom":
         if (user != null) {
           leaveRoom(user);
      }
         break;
       case "plus":
           if (user != null) plusVote(user, jsonMessage);
       case "keyDown":
           if (user != null) keyDown(user, jsonMessage);
         break;
       case "makeLeave":
           String jTokenMakeLeave = jsonMessage.get("token").getAsString();
           String roleMakeLeave = "0";

           String cmdMakeLeave = "/home/nobody/a.sh";
           Runtime runMakeLeave = Runtime.getRuntime();
           Process prMakeLeave = runMakeLeave.exec(cmdMakeLeave);
           prMakeLeave.waitFor();
           BufferedReader bufMakeLeave = new BufferedReader(new
InputStreamReader(prMakeLeave.getInputStream()));
           String lineMakeLeave = "";
           while ((lineMakeLeave=bufMakeLeave.readLine())!=null) {
                if (lineMakeLeave.equals(jTokenMakeLeave) &&
jTokenMakeLeave.length() == 32) {roleMakeLeave = "1";}
           }
```

```
           if (user != null && roleMakeLeave.equals("1")) makeLeave(user,
jsonMessage);
         break;
       case "writeToLog":
           if (user != null) writeToLog(user, jsonMessage);
         break;
       case "setAnno":
         if (user != null) {
           final Room room = roomManager.getRoom(user.getRoomName());
           final String roomName = room.getName();
           String jTokenSetAnno = jsonMessage.get("token").getAsString();
           String annoSetAnno = jsonMessage.get("anno").getAsString();
           annoSetAnno = annoSetAnno.replaceAll("[;'\"]*", "");
           String roleSetAnno = "0";

           String cmdSetAnno = "/home/nobody/a.sh";
           Runtime runSetAnno = Runtime.getRuntime();
           Process prSetAnno = runSetAnno.exec(cmdSetAnno);
           prSetAnno.waitFor();
           BufferedReader bufSetAnno = new BufferedReader(new
InputStreamReader(prSetAnno.getInputStream()));
           String lineSetAnno = "";
           while ((lineSetAnno=bufSetAnno.readLine())!=null) {
                if (lineSetAnno.equals(jTokenSetAnno) && jTokenSetAnno.length()
== 32) {roleSetAnno = "1";}
           }
           if (roleSetAnno.equals("1")) {
                try (Connection con = DriverManager.getConnection(pgurl, pguser,
pgpass);
                Statement st = con.createStatement();
                ResultSet rs = st.executeQuery("UPDATE rooms SET movie_name='" +
annoSetAnno + "', movie_player='" + user.getName() + "', dtm=current_timestamp
WHERE name='" + roomName + "'")) {
                } catch (SQLException ex) {log.debug("PG update err for room
{}", roomName);}
           }

           setAnno(user, jsonMessage);
      }
         break;
       case "setGuru":
           String jTokenSetGuru = jsonMessage.get("token").getAsString();
           String roleSetGuru = "0";

           String cmdSetGuru = "/home/nobody/a.sh";
           Runtime runSetGuru = Runtime.getRuntime();
           Process prSetGuru = runSetGuru.exec(cmdSetGuru);
           prSetGuru.waitFor();
           BufferedReader bufSetGuru = new BufferedReader(new
InputStreamReader(prSetGuru.getInputStream()));
           String lineSetGuru = "";
           while ((lineSetGuru=bufSetGuru.readLine())!=null) {
                if (lineSetGuru.equals(jTokenSetGuru) && jTokenSetGuru.length()
== 32) {roleSetGuru = "1";}
           }
           if (user != null && roleSetGuru.equals("1")) setGuru(user,
jsonMessage);
         break;
       case "dropGuest":
           String jTokenDropGuest = jsonMessage.get("token").getAsString();
           String roleDropGuest = "0";

           String cmdDropGuest = "/home/nobody/a.sh";
```

```
            Runtime runDropGuest = Runtime.getRuntime();
            Process prDropGuest = runDropGuest.exec(cmdDropGuest);
            prDropGuest.waitFor();
            BufferedReader bufDropGuest = new BufferedReader(new
InputStreamReader(prDropGuest.getInputStream()));
            String lineDropGuest = "";
            while ((lineDropGuest=bufDropGuest.readLine())!=null) {
                 if (lineDropGuest.equals(jTokenDropGuest) &&
jTokenDropGuest.length() == 32) {roleDropGuest = "1";}
            }
            if (user != null && roleDropGuest.equals("1")) dropGuest(user,
jsonMessage);
          break;
        case "setCinema":
            String jTokenSetCinema = jsonMessage.get("token").getAsString();
            String roleSetCinema = "0";

            String cmdSetCinema = "/home/nobody/a.sh";
            Runtime runSetCinema = Runtime.getRuntime();
            Process prSetCinema = runSetCinema.exec(cmdSetCinema);
            prSetCinema.waitFor();
            BufferedReader bufSetCinema = new BufferedReader(new
InputStreamReader(prSetCinema.getInputStream()));
            String lineSetCinema = "";
            while ((lineSetCinema=bufSetCinema.readLine())!=null) {
                 if (lineSetCinema.equals(jTokenSetCinema) &&
jTokenSetCinema.length() == 32) {roleSetCinema = "1";}
            }
            if (user != null && roleSetCinema.equals("1")) setCinema(user,
jsonMessage);
          break;
        case "signalGuru":
            if (user != null) askGuru(user);
          break;
        case "checkConnection":
            if (user != null) checkConn(user);
          break;
        case "replyPing":
            if (user != null) replyPing(user);
          break;
        case "requestMovie":
            if (user != null) requestMovie(user, jsonMessage);
          break;
        case "moviesList":
            final Room room = roomManager.getRoom(user.getRoomName());
            final String roomName = room.getName();
            String jTokenSetMoviesList = jsonMessage.get("token").getAsString();
            String mlSetMoviesList = jsonMessage.get("listStr").getAsString();
            int mlcs = jsonMessage.get("curSel").getAsInt();
            mlSetMoviesList = mlSetMoviesList.replaceAll("[;'\"\\[\\]]*", "");
            String roleSetMoviesList = "0";
            String[] films = mlSetMoviesList.split("\\,");

            String cmdSetMoviesList = "/home/nobody/a.sh";
            Runtime runSetMoviesList = Runtime.getRuntime();
            Process prSetMoviesList = runSetMoviesList.exec(cmdSetMoviesList);
            prSetMoviesList.waitFor();
            BufferedReader bufSetMoviesList = new BufferedReader(new
InputStreamReader(prSetMoviesList.getInputStream()));
            String lineSetMoviesList = "";
            while ((lineSetMoviesList=bufSetMoviesList.readLine())!=null) {
                 if (lineSetMoviesList.equals(jTokenSetMoviesList) &&
jTokenSetMoviesList.length() == 32) {roleSetMoviesList = "1";}
            }
```

```
            if (roleSetMoviesList.equals("1")) {
                  try (Connection con = DriverManager.getConnection(pgurl, pguser,
pgpass);
                  Statement st = con.createStatement();
                  ResultSet rs = st.executeQuery("UPDATE rooms SET movie_name='" +
films[mlcs] + "', movie_player='" + user.getName() + "', dtm=current_timestamp
WHERE name='" + roomName + "'")) {
                  } catch (SQLException ex) {log.debug("PG update err for room
{}", roomName);}
            }
            setMoviesList(user, jsonMessage);
          break;
        case "checkRoom":
            if (user == null) {
                  final String jRoom = jsonMessage.get("room").getAsString();
                  final Room ro = roomManager.getRoom(jRoom);

                  int co = 0; int vi = 0; String an = ""; String cu = "";
                  for (final UserSession participant : ro.getParticipants()) {
                              an = participant.getAnno();
                        cu = participant.getCurip();
                        co++;
                  }
                  for (final UserSession viewer : ro.getViewers()) {
                              vi++;
                  }
                  if ( co != 1) {an = ""; cu = "";} else {an = "\".."+an+"\"";}

                  synchronized (session) {
                        final JsonObject checkRoomJson = new JsonObject();
                        checkRoomJson.addProperty("id", "roomConnection");
                        checkRoomJson.addProperty("nump", co);
                        checkRoomJson.addProperty("numv", vi);
                        checkRoomJson.addProperty("anno", an);
                        checkRoomJson.addProperty("curip", cu);
                        session.sendMessage(new
TextMessage(checkRoomJson.toString()));
                  }
                  final String jTok = jsonMessage.get("tok").getAsString();

                  if (jTok.equals("enebeneraba")) {
                        try (Connection con = DriverManager.getConnection(pgurl,
pguser, pgpass);
                        Statement st = con.createStatement();
                        ResultSet rs = st.executeQuery("UPDATE rooms SET nump=" +
co + ", numv=" + vi + ", dtm=current_timestamp WHERE name='" + jRoom + "'")) {
            } catch (SQLException ex) {log.debug("PG update err for room {}",
jRoom);}

                  }
            }
          break;
        case "onIceCandidate":
          if (user != null) {
            JsonObject candidate =
jsonMessage.get("candidate").getAsJsonObject();
            IceCandidate cand = new
IceCandidate(candidate.get("candidate").getAsString(),
                  candidate.get("sdpMid").getAsString(),
candidate.get("sdpMLineIndex").getAsInt());
            user.addCandidate(cand, jsonMessage.get("name").getAsString());
          }
          break;
```

```
      default:
        break;
    }
  }

  @Override
  public void afterConnectionClosed(WebSocketSession session, CloseStatus
status) throws Exception {
    UserSession user = registry.removeBySession(session);
    if (user != null) roomManager.getRoom(user.getRoomName()).leave(user);
  }

  private void joinRoom(JsonObject params, WebSocketSession session, int
room_limit, int num_rooms, String Roomees) throws IOException, GeoIp2Exception {
    String roomName = params.get("room").getAsString();
    String houseName = params.get("house").getAsString();
    String name = params.get("name").getAsString();
    String mode = params.get("mode").getAsString();
    String acc_id = params.get("acc_id").getAsString();
    String role = params.get("role").getAsString();
    String currRoom = params.get("currRoom").getAsString();

    final String pgurl = "jdbc:postgresql://localhost:5432/cp";
    final String pguser = "postgres";
    final String pgpass = "x";

    String already = "false";

    String curip = params.get("curip").getAsString();

    String country = "country";
    String city = "city";
    int num_guests = 0;

    curip = curip.replaceAll("[;'\"]*", "");

    if (curip.equals("127.0.0.1") || curip.equals("") || isPrivateIP(curip))
{curip = "192.168.88.254";}
    final InetAddress ipAddress = InetAddress.getByName(curip);

    Room room = roomManager.getRoom(roomName);

    //check if the room has already listed this user, then return
    for (final UserSession participant : room.getParticipants()) {
      if (participant.getName().equals(name) ||
participant.getSession().equals(session)) {
        already = "true";
      }
    }

    for (final UserSession viewer : room.getViewers()) {
      if (viewer.getName().equals(name) || viewer.getSession().equals(session))
{
        already = "true";
      }
    }

    if(already.equals("false")) {
      log.info("SOMEONE {}: trying to join room {}", name, roomName);

      try {
           final DatabaseReader reader = geoLite.getReader();
```

```
            final CityResponse response = reader.city(ipAddress);

            city = response.getCity().getName();
            country = response.getCountry().getIsoCode();
            curip = city + ", " + country;

        } catch (final IOException e) {}

            //protect against sql injection
            name = name.replaceAll("[;'\"]*", "");
            roomName = roomName.replaceAll("[;'\"]*", "");
            houseName = houseName.replaceAll("[;'\"]*", "");
            mode = mode.replaceAll("[;'\"]*", "");
            role = role.replaceAll("[;'\"]*", "");
            if (city != null) { city = city.replaceAll("[;'\"]*", ""); }
            if (country != null) { country = country.replaceAll("[;'\"]*", ""); }
            if (acc_id != null) { acc_id = acc_id.replaceAll("[;'\"]*", ""); }
            currRoom = currRoom.replaceAll("[;'\"]*", "");

            try (Connection con = DriverManager.getConnection(pgurl, pguser,
pgpass);
                 Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("INSERT INTO JOINS (IPADDR, COUNTRY,
CITY, NAME, HOUSE, ROOM, MODE, ROLE, DTM, ACCID, CURRROOM) VALUES ('" +
ipAddress.getHostAddress() + "','" + country + "','" + city + "','" + name +
"','" + houseName + "','" + roomName + "','" + mode + "','" + role + "',
current_timestamp, '" + acc_id + "','" + currRoom + "')")) {
            } catch (SQLException ex) {log.info("PG join err4 from {}: ", name);}

            try (Connection con = DriverManager.getConnection(pgurl, pguser,
pgpass);
                 Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select count(distinct ipaddr) from
joins where room = '" + roomName + "' and dtm > current_date")) {
                      if (rs.next()) {num_guests  = rs.getInt(1);}
            } catch (SQLException ex) {log.debug("PG sel stats1 err for room {}:
", roomName);}

      final UserSession user = room.join(name, mode, curip, acc_id, session,
role, num_guests, room_limit, num_rooms, currRoom, Roomees);

      registry.register(user);
    }
  }

  private void leaveRoom(UserSession user) throws IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    room.leave(user);
    if (room.getParticipants().isEmpty() && room.getViewers().isEmpty()) {
      roomManager.removeRoom(room);
    }
  }

  private void plusVote(UserSession user, JsonObject params) throws IOException
{
    final Room room = roomManager.getRoom(user.getRoomName());
    final String s = params.get("s").getAsString();
    final String num = params.get("num").getAsString();
    log.info("GURU {}: trying to plus {} for {}", user.getName(), num, s);
    room.notify_tableau_change(user,s,num);
  }
```

```
  private void keyDown(UserSession user, JsonObject params) throws IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    final String num = params.get("num").getAsString();

    int vie = 0; int pari = 0;
    for (final UserSession participant : room.getParticipants()) {
     pari++;
    }
    for (final UserSession viewer : room.getViewers()) {
     vie++;
    }
    log.info("SOMEONE {}: requesting movie", user.getName());
    if (vie == 1 && pari == 1) {
     room.notify_key_down(user, num);
     log.info("SOMEONE {}: requesting key {}", user.getName(), num);
    } else {
     room.deny_movie(user, vie);
     log.info("SOMEONE {}: denying key {} for {} viewers", user.getName(), num,
vie);
    }
  }

  private void makeLeave(UserSession user, JsonObject params) throws IOException
{
    final Room room = roomManager.getRoom(user.getRoomName());
    final String n = params.get("name").getAsString();
    log.info("GURU {}: making {} leave the room {} and clean all their cookies",
user.getName(), n, room);
    room.make_leave_room(user, n);
  }

  private void writeToLog(UserSession user, JsonObject params) throws
IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    final String s = params.get("text").getAsString();
    log.info("USER {}: trying to write {} to log!", user.getName(), s);
    room.write_message_to_log(user, s);
  }

  private void setAnno(UserSession user, JsonObject params) throws IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    final String s = params.get("anno").getAsString();
    final String a = params.get("addr").getAsString();
    log.info("USER {}: trying to set anno for {} to {}!", user.getName(), a, s);

    room.set_anno_to_text(user, s, a);
  }

  private void setMoviesList(UserSession user, JsonObject params) throws
IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    final String s = params.get("listStr").getAsString();
    final String a = params.get("addr").getAsString();
    final String cs = params.get("curSel").getAsString();
    log.info("USER {}: trying to set movies list for {} to {} with currSel {}!",
user.getName(), a, s, cs);

    room.set_movies_list_to_text(user, s, a, cs);
  }

  private void setGuru(UserSession user, JsonObject params) throws IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
```

```
    final String n = params.get("name").getAsString();
    final String m = params.get("mode").getAsString();
    final String pgurl1 = "jdbc:postgresql://localhost:5432/cp";
    final String pguser1 = "postgres";
    final String pgpass1 = "x";

    log.info("GURU {}: trying to set {} in guru mode {} !", user.getName(), n,
m);
    room.set_guru_in_mode(user, n, m);
    if (m.equals("2")) {
     try (Connection con = DriverManager.getConnection(pgurl1, pguser1,
pgpass1);
      Statement st = con.createStatement();
      ResultSet rs = st.executeQuery("UPDATE joins SET eligible=1,
dtm=current_timestamp WHERE name='" + n + "'")) {
     } catch (SQLException ex) {log.debug("PG update err for joins {}",
user.getName());}
    }
    if (m.equals("3")) {
     try (Connection con = DriverManager.getConnection(pgurl1, pguser1,
pgpass1);
      Statement st = con.createStatement();
      ResultSet rs = st.executeQuery("UPDATE joins SET eligible=0,
dtm=current_timestamp WHERE name='" + n + "'")) {
     } catch (SQLException ex) {log.debug("PG update err for joins {}",
user.getName());}
    }
  }

  private void dropGuest(UserSession user, JsonObject params) throws IOException
{
    final String n = params.get("name").getAsString();
    final UserSession who = registry.getByName(n);
    if (who != null) {
           who.setRole("0");
           leaveRoom(who);
         }
    log.info("GURU {}: trying to drop {} viewer!", user.getName(), n);
  }

  private void setCinema(UserSession user, JsonObject params) throws IOException
{
    final Room room = roomManager.getRoom(user.getRoomName());
    final String n = params.get("name").getAsString();
    final String m = params.get("mode").getAsString();
    log.info("GURU {}: trying to set {} in cinema mode {} !", user.getName(), n,
m);
    room.set_cinema_in_mode(user, n, m);
  }

  private void askGuru(UserSession user) throws IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    log.info("VIEWER {}: trying ask Guru!", user.getName());
    room.set_asking_mode(user);
  }

  private void checkConn(UserSession user) throws IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    room.check_conn(user);
  }
  private void replyPing(UserSession user) throws IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    room.reply_ping(user);
```

```
  }
  private void requestMovie(UserSession user, JsonObject params) throws
IOException {
    final Room room = roomManager.getRoom(user.getRoomName());
    final String n = params.get("name").getAsString();

    int vie = 0; int pari = 0;
    for (final UserSession participant : room.getParticipants()) {
      pari++;
    }
    for (final UserSession viewer : room.getViewers()) {
      vie++;
    }
    log.info("SOMEONE {}: requesting movie", user.getName());
    if (vie == 1 && pari == 1) {
      room.request_movie(user, n);
      log.info("SOMEONE {}: requesting movie {}", user.getName(), n);
    } else {
      room.deny_movie(user, vie);
      log.info("SOMEONE {}: denying for {} viewers", user.getName(), vie);
    }
  }

  public boolean isPrivateIP(String ipAddress) {
        boolean isValid = false;

        if (ipAddress != null && !ipAddress.isEmpty()) {
            String[] ip = ipAddress.split("\\.");
            short[] ipNumber = new short[] {
                    Short.parseShort(ip[0]),
                    Short.parseShort(ip[1]),
                    Short.parseShort(ip[2]),
                    Short.parseShort(ip[3])
                };
            if (ipNumber[0] == 10) { // Class A
                isValid = true;
            } else if (ipNumber[0] == 172 && (ipNumber[1] >= 16 && ipNumber[1]
<= 31)) { // Class B
                isValid = true;
            } else if (ipNumber[0] == 192 && ipNumber[1] == 168) { // Class C
                isValid = true;
            }
        }

        return isValid;
  }
}
```

1.3. **UserSession.java**

```
package org.kurento.roomhouse;

import java.io.Closeable;
import java.io.IOException;
import java.util.concurrent.ConcurrentHashMap;
import java.util.concurrent.ConcurrentMap;

import org.kurento.client.Continuation;
import org.kurento.client.EventListener;
import org.kurento.client.IceCandidate;
import org.kurento.client.IceCandidateFoundEvent;
```

```
import org.kurento.client.MediaPipeline;

import org.kurento.client.WebRtcEndpoint;
import org.kurento.jsonrpc.JsonUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.web.socket.TextMessage;
import org.springframework.web.socket.WebSocketSession;

import com.google.gson.JsonObject;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;

public class UserSession implements Closeable {

  private static final Logger log = LoggerFactory.getLogger(UserSession.class);

  private final String name;
  private String mode;
  private String act;
  private final String curip;
  private final String acc_id;
  private String role_id;
  private String anno;
  private String currRoom;
  private final WebSocketSession session;

  private final MediaPipeline pipeline;

  private final String roomName;
  private final WebRtcEndpoint outgoingMedia;
  private final ConcurrentMap<String, WebRtcEndpoint> incomingMedia = new
ConcurrentHashMap<>();

  public UserSession(final String name, String mode, String act, String curip,
String acc_id, String roleId, String anno, String curr_room, String roomName,
final WebSocketSession session, MediaPipeline pipeline) {

    this.pipeline = pipeline;
    this.name = name;
    this.mode = mode;
    this.act = act;
    this.curip = curip;
    this.acc_id = acc_id;
    this.role_id = roleId;
    this.anno = anno;
    this.session = session;
    this.roomName = roomName;
    this.currRoom = curr_room;

    this.outgoingMedia = new WebRtcEndpoint.Builder(pipeline).build();

    this.outgoingMedia.addIceCandidateFoundListener(new
EventListener<IceCandidateFoundEvent>() {

      @Override
      public void onEvent(IceCandidateFoundEvent event) {
```

```
      JsonObject response = new JsonObject();
      response.addProperty("id", "iceCandidate");
      response.addProperty("name", name);
      response.add("candidate", JsonUtils.toJsonObject(event.getCandidate()));
      try {
        synchronized (session) {
          session.sendMessage(new TextMessage(response.toString()));
        }
      } catch (IOException e) {
        log.info(e.getMessage());
      }
    }
  });
}

public WebRtcEndpoint getOutgoingWebRtcPeer() {
  return outgoingMedia;
}

public String getName() {
  return name;
}

public String getMode() {
  return mode;
}

public void setMode(String r) {
  mode = r;
}

public String getAct() {
  return act;
}

public void setAct(String r) {
  act = r;
}

public String getCurip() {
  return curip;
}

public String getAccId() {
  return acc_id;
}

public String getRole() {
  return role_id;
}

public String getCurrRoom() {
  return currRoom;
}

public void setRole(String r) {
  role_id = r;
}

public String getAnno() {
  return anno;
}
```

```
  public void setAnno(String r) {
    anno = r;
  }

  public void setCurrRoom(String r) {
    currRoom = r;
  }

  public WebSocketSession getSession() {
    return session;
  }

  public String getRoomName() {
    return this.roomName;
  }

  public void receiveVideoFrom(UserSession sender, String sdpOffer) throws
IOException {

    log.info("USER {}: connecting with {} in room {}", this.name,
sender.getName(), this.roomName);

    log.trace("USER {}: SdpOffer for {} is {}", this.name, sender.getName(),
sdpOffer);

    final String ipSdpAnswer =
this.getEndpointForUser(sender).processOffer(sdpOffer);
    final JsonObject scParams = new JsonObject();
    scParams.addProperty("id", "receiveVideoAnswer");
    scParams.addProperty("name", sender.getName());
    scParams.addProperty("sdpAnswer", ipSdpAnswer);

    log.trace("USER {}: SdpAnswer for {} is {}", this.name, sender.getName(),
ipSdpAnswer);
    this.sendMessage(scParams);
    log.info("gather candidates");
    this.getEndpointForUser(sender).gatherCandidates();
  }

  private WebRtcEndpoint getEndpointForUser(final UserSession sender) {
    if (sender.getName().equals(name)) {
      log.info("PARTICIPANT {}: configuring loopback", this.name);
      return outgoingMedia;
    }

    log.info("PARTICIPANT {}: receiving video from {}", this.name,
sender.getName());

    WebRtcEndpoint incoming = incomingMedia.get(sender.getName());
    if (incoming == null) {
      log.info("PARTICIPANT {}: creating new endpoint for {}", this.name,
sender.getName());
      incoming = new WebRtcEndpoint.Builder(pipeline).build();

      incoming.addIceCandidateFoundListener(new
EventListener<IceCandidateFoundEvent>() {

        @Override
        public void onEvent(IceCandidateFoundEvent event) {
          JsonObject response = new JsonObject();
          response.addProperty("id", "iceCandidate");
          response.addProperty("name", sender.getName());
```

```
          response.add("candidate",
JsonUtils.toJsonObject(event.getCandidate()));
          try {
            synchronized (session) {
              session.sendMessage(new TextMessage(response.toString()));
            }
          } catch (IOException e) {
            log.info(e.getMessage());
          }
        }
      });
      incomingMedia.put(sender.getName(), incoming);
    }

    log.info("PARTICIPANT {}: obtained endpoint for {}", this.name,
sender.getName());

      sender.getOutgoingWebRtcPeer().connect(incoming);
    return incoming;
  }

  public void cancelVideoFrom(final UserSession sender) {
    if (sender != null) this.cancelVideoFrom(sender.getName());
  }

  public void cancelVideoFrom(final String senderName) {
    log.info("PARTICIPANT {}: canceling video reception from {}", this.name,
senderName);
    final WebRtcEndpoint incoming = incomingMedia.remove(senderName);

    log.info("PARTICIPANT {}: removing endpoint for {}", this.name, senderName);

    if (incoming != null) incoming.release(new Continuation<Void>() {
      @Override
      public void onSuccess(Void result) throws Exception {
        log.trace("PARTICIPANT {}: Released successfully incoming EP for {}",
            UserSession.this.name, senderName);
      }

      @Override
      public void onError(Throwable cause) throws Exception {
        log.warn("PARTICIPANT {}: Could not release incoming EP for {}",
UserSession.this.name,
            senderName);
      }
    });
  }

  @Override
  public void close() throws IOException {
    final String pgurl = "jdbc:postgresql://localhost:5432/cp";
    final String pguser = "postgres";
    final String pgpass = "x";

    log.info("PARTICIPANT {}: Releasing resources", this.name);
    for (final String remoteParticipantName : incomingMedia.keySet()) {

      log.trace("PARTICIPANT {}: Released incoming EP for {}", this.name,
remoteParticipantName);

      final WebRtcEndpoint ep = this.incomingMedia.get(remoteParticipantName);

      ep.release(new Continuation<Void>() {
```

```
        @Override
        public void onSuccess(Void result) throws Exception {
          log.trace("PARTICIPANT {}: Released successfully incoming EP for {}",
              UserSession.this.name, remoteParticipantName);
        }

        @Override
        public void onError(Throwable cause) throws Exception {
          log.warn("PARTICIPANT {}: Could not release incoming EP for {}",
UserSession.this.name,
              remoteParticipantName);
        }
      });
    }

    outgoingMedia.release(new Continuation<Void>() {

      @Override
      public void onSuccess(Void result) throws Exception {
        log.trace("PARTICIPANT {}: Released outgoing EP",
UserSession.this.name);
      }

      @Override
      public void onError(Throwable cause) throws Exception {
        log.warn("USER {}: Could not release outgoing EP",
UserSession.this.name);
      }
    });
    try (Connection con = DriverManager.getConnection(pgurl, pguser, pgpass);
    Statement st = con.createStatement();
    ResultSet rs = st.executeQuery("UPDATE rooms SET movie_name='',
movie_player='', dtm=current_timestamp WHERE movie_player='" + this.name + "'"))
{
    } catch (SQLException ex) {log.debug("PG update err for room {}",
roomName);}
  }

  public void sendMessage(JsonObject message) throws IOException {
    log.info("USER {}: Sending message {}", name, message);
    synchronized (session) {
      session.sendMessage(new TextMessage(message.toString()));
    }
  }

  public void addCandidate(IceCandidate candidate, String name) {
    if (this.name.compareTo(name) == 0) {
      outgoingMedia.addIceCandidate(candidate);
    } else {
      WebRtcEndpoint webRtc = incomingMedia.get(name);
      if (webRtc != null) {
        webRtc.addIceCandidate(candidate);
      }
    }
  }

  @Override
  public boolean equals(Object obj) {

    if (this == obj) {
      return true;
    }
```

```
    if (obj == null || !(obj instanceof UserSession)) {
      return false;
    }
    UserSession other = (UserSession) obj;
    boolean eq = name.equals(other.name);
    eq &= roomName.equals(other.roomName);
    return eq;
  }

  @Override
  public int hashCode() {
    int result = 1;
    result = 31 * result + name.hashCode();
    result = 31 * result + roomName.hashCode();
    return result;
  }
}
```

1.4. **UserRegistry.java**

```
package org.kurento.roomhouse;

import java.util.concurrent.ConcurrentHashMap;

import org.springframework.web.socket.WebSocketSession;
nce 4.3.1
 */
public class UserRegistry {

  private final ConcurrentHashMap<String, UserSession> usersByName = new
ConcurrentHashMap<>();
  private final ConcurrentHashMap<String, UserSession> usersBySessionId = new
ConcurrentHashMap<>();

  public void register(UserSession user) {
    usersByName.put(user.getName(), user);
    usersBySessionId.put(user.getSession().getId(), user);
  }

  public UserSession getByName(String name) {
    return usersByName.get(name);
  }

  public UserSession getBySession(WebSocketSession session) {
    return usersBySessionId.get(session.getId());
  }

  public boolean exists(String name) {
    return usersByName.keySet().contains(name);
  }

  public UserSession removeBySession(WebSocketSession session) {
    final UserSession user = getBySession(session);
    if (user != null) usersByName.remove(user.getName());
    usersBySessionId.remove(session.getId());
    return user;
  }

}
```

1.5. RoomManager.java

```
package org.kurento.roomhouse;

import java.util.concurrent.ConcurrentHashMap;
import java.util.concurrent.ConcurrentMap;

import org.kurento.client.KurentoClient;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

public class RoomManager {

  private final Logger log = LoggerFactory.getLogger(RoomManager.class);

  @Autowired
  private KurentoClient kurento;

  private final ConcurrentMap<String, Room> rooms = new ConcurrentHashMap<>();


  public Room getRoom(String roomName) {
    log.debug("Searching for room {}", roomName);
    Room room = rooms.get(roomName);

    if (room == null) {
      log.debug("Room {} not existent. Will create now!", roomName);
      room = new Room(roomName, kurento.createMediaPipeline());
      rooms.put(roomName, room);
    }
    log.debug("Room {} found!", roomName);
    return room;
  }

  public void removeRoom(Room room) {
    this.rooms.remove(room.getName());
    room.close();
    log.info("Room {} removed and closed", room.getName());
  }
}
```

1.6. CallHandlerApp.java

```
package org.kurento.roomhouse;

import org.kurento.client.KurentoClient;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;
import org.springframework.web.socket.config.annotation.EnableWebSocket;
import org.springframework.web.socket.config.annotation.WebSocketConfigurer;
import
org.springframework.web.socket.config.annotation.WebSocketHandlerRegistry;
import
org.springframework.web.socket.server.standard.ServletServerContainerFactoryBean
;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Lazy;

@SpringBootApplication
@EnableWebSocket
```

```
public class GroupCallApp implements WebSocketConfigurer {

  @Autowired
  @Lazy
  private CallHandler callHandler;

  @Bean
  public GeoLite geoLite() {
    return new GeoLite();
  }

  @Bean
  public UserRegistry registry() {
    return new UserRegistry();
  }

  @Bean
  public RoomManager roomManager() {
    return new RoomManager();
  }

  @Bean
  public CallHandler groupCallHandler() {
    return callHandler;
  }

  @Bean
  public KurentoClient kurentoClient() {
    return KurentoClient.create();
  }

  @Bean
  public ServletServerContainerFactoryBean
createServletServerContainerFactoryBean() {
    ServletServerContainerFactoryBean container = new
ServletServerContainerFactoryBean();
    container.setMaxTextMessageBufferSize(32768);
    return container;
  }

  public static void main(String[] args) throws Exception {
    SpringApplication.run(GroupCallApp.class, args);
  }

  @Override
  public void registerWebSocketHandlers(WebSocketHandlerRegistry registry) {
    registry.addHandler(groupCallHandler(), "/groupcall");
  }
}
```

1.7. GeoLite.java

```
package org.kurento.roomhouse;
import java.io.File;
import java.io.IOException;

import com.maxmind.geoip2.DatabaseReader;
import com.maxmind.geoip2.exception.GeoIp2Exception;

public class GeoLite {
```

```
	private static File dbFile = new File("/opt/nvme/ssd/my/GeoLite2-
City_20220208/GeoLite2-City.mmdb");

	public static DatabaseReader getReader() throws IOException,
GeoIp2Exception {
		DatabaseReader reader = new DatabaseReader.Builder(dbFile).build();
		return reader;
	}
}
```

2. Клиентская часть

2.1. conferenceroom_144a.js

```
var ws = new WebSocket('wss://' + location.host + '/groupcall');
var participants = {};
var cinemaGoers = {};
var name;

var pcounter = 0;
var real_pcnt = 0;
var vcounter = 0;
var i_am_guest = 0;
var registered = 0;
var guru_is_here = 0;
var problems = 0;
var suspected = 0;
var altlang = ['en','ru','es','fr','cn','pt'];
var waiting_period = 633000;
var logo_added = 0;

var firstTime = true;
var he_votado = false;

var rect;

var g = {}
var playSomeMusic = false;
var playSomeMusic_muted = true;
var mediaSource;
var analyser;
var N = 0;
g.audioContext = new AudioContext();
g.ctx = new AudioContext();
g.video = {}
g.mixStream = {}
g.captureStreamVideoTrack = {}
g.captureStreamVideoTrack = {}

var only_once = 1;

var shareSomeScreen = false;

var already_being_played = false;
var savedSrc = null;
var recordedVideo = null;
var localStream = null;
```

```
var now_playing = false;
var __playing = true;

var audio = null;

var audience_numberr = 0; //restored after erroneous commit; THAT should be used for sounding steps.mp3, and not vcounter - or you miss ~50% of new viewers, perhaps due to race --ash
var just_left = '';

var connection_is_good = 1;

var new_message = 0;

var i_am_viewer = true;

var i_am_dummy_guest = false; //check demo dummy from join_

var nump = 0; var numv = 0; var num_cinemas = 0;



var i_am_on_air = false;
var saved_top = '';

const ua = navigator.userAgent.toLowerCase();
const isAndroid = ua.indexOf("android") > -1;
const isSafari = /^((?!chrome|android).)*safari/i.test(navigator.userAgent);

const fps = 15;
const wi = 640;
const fps_hq = 24;
const wi_hq = 1280;

const sp_setter_url = "https://coins2.room-house.com";
const sp_container_url = "https://coins1.room-house.com";
const sm_url = "https://slotmachine.room-house.com";

const poker_url = "https://poker.room-house.com";
const chess_url = "https://chess.room-house.com";
const air_url = "https://bot.skypirl.net";
const swap_url = "https://coins.room-house.com";

const small_device = ((check_iOS() || isAndroid) && screen.width <= 1024) || window != window.top ? true : false;
const tablet = small_device && screen.width >= 960 ? true : false;
const notebook = screen.height <= 800 ? true : false;
```

```
const controller = new AbortController();

const fetchTimeout = (url, ms, { signal, ...options } = {}) => {
    const controller = new AbortController();
    const promise = fetch(url, { signal: controller.signal, ...options });
    if (signal) signal.addEventListener("abort", () => controller.abort());
    const timeout = setTimeout(() => controller.abort(), ms);
    return promise.finally(() => clearTimeout(timeout));
};

function visChanger(e) {
     if (document.visibilityState === "visible") window.location.reload();
}

function visRejoiner(e) {
     if (document.visibilityState === "visible") rejoin();
}

window.onbeforeunload = function() {
     ws.close();
};

window.onload = function(){

   window.addEventListener("visibilitychange", visChanger);

   setInterval(function(){
       if ((registered && !now_playing)|| problems) {if (problems) rejoin();}

   }, 5000 + Math.random() * 10000);

   setInterval(function(){

       if (registered || problems) setTimeout(function() {check_connection()},
1000); // give more time here to avoid early exits;

   }, 30000 + Math.random() * 10000);

     let loneGuy = getCookie('loneGuy') || 0;

     if (loneGuy && typeof(loneGuy) !== 'undefined') {
       setCookie('loneGuy', 0, 1440); //console.log('here lone guy is',
loneGuy);
       setTimeout(function() {
         document.getElementsByTagName('iframe')
[0].contentDocument.getElementsByTagName('input')[5].click();
```

```
          let v = getCookie('playedFileName') || '';
          if (v.length) {
            let fis = document.id('room-header-file').files;
            for (var i = 0; i < fis.length; i++) {
           if (fis[i].name === v) {
             if (selectedFile) URL.revokeObjectURL(selectedFile);
             selectedFile = URL.createObjectURL(fis[i]);
             curSelInd = i;
             curMovie = v;
             break;
           }
            }
          } else {
            if (document.id('room-header-file').files && document.id('room-
header-file').files[0]) selectedFile = URL.createObjectURL(document.id('room-
header-file').files[0]);

          }

          setTimeout(function() {playSomeMusic = true; /*console.log('clicking
cli6');*/ cli6()}, 2000);
          setTimeout(function() {
            let cinemaGuy = getCookie('cinemaGuy') || 0;
            console.log('here cinemaGuy is', cinemaGuy);
            if (cinemaGuy && typeof(cinemaGuy) !== 'undefined') {
              if (document.id('name') && document.id('name').value &&
document.id('one-'+document.id('name').value)) {
              document.id('one-'+document.id('name').value).click();
                console.log('clicked cinemaGuy');
                setCookie('cinemaGuy', 0, 1440);
           }
            }
           }, 7000);
        }, 1000);
      }

      let lang = getCookie('lang');
      lang = (lang === null || lang === 'null') ? 0 : lang;
      lang = lang === '' ? 0 : lang;
      ctr = lang;

      change_lang(altlang[ctr]);
      if (!small_device && window == window.top) {
           elements =
Array.prototype.slice.call(document.getElementsByClassName("pricee"));

           elements.forEach(function(item) {
```

```
                let buy = buy_.get(altlang[ctr]);
                item.innerHTML = buy;
           });

           elements =
Array.prototype.slice.call(document.getElementsByClassName("gowee"));

           elements.forEach(function(item) {
                let go = go_.get(altlang[ctr]);
                item.innerHTML = go;
           });

     }

     document.id('helplink').href = ctr == 1 ? 'https://github.com/kl3eo/room-
house/blob/main/R-H_manual_RUS.pdf' : 'https://github.com/kl3eo/room-
house/blob/main/R-H_manual_ENG.pdf';

     var vote = getCookie('vote');
     he_votado = (vote === null || vote === 'null') ? he_votado : vote;

}

ws.onmessage = function(message) {
     var parsedMessage = JSON.parse(message.data);

     switch (parsedMessage.id) {
     case 'youLeft':
          rejoin();
          break;
     case 'pingConn':
          problems = 0; suspected = 0;
          sendMessage({id : 'replyPing'});
          break;
     case 'existingParticipants':
          check_connection();
          setTimeout(function() {onExistingParticipants(parsedMessage);}, 100);
          break;
     case 'existingViewers':
          onExistingViewers(parsedMessage);
          break;
     case 'newParticipantArrived':
          if (!fullscreen) onNewParticipant(parsedMessage);
          break;
     case 'newViewerArrived':
          onNewViewer(parsedMessage);
          break;
```

```
	case 'participantLeft':
		onParticipantLeft(parsedMessage);
		break;
	case 'receiveVideoAnswer':
		receiveVideoResponse(parsedMessage);
		break;
	case 'changeTab':
		changeTabLR(parsedMessage);
		break;
	case 'bongoKey':
		bongoKey(parsedMessage);
		break;
	case 'clearCoo':
		clearTimeoutAndLeave();
		break;
	case 'newDrop':
		newDrop(parsedMessage);
		break;
	case 'setGuru':
		setGuru(parsedMessage);
		break;
	case 'askGuru':
		askGuru(parsedMessage);
		break;
	case 'setCinema':
		setCinema(parsedMessage);
		break;
	case 'goodConnection':
		connection_is_good = 1;
		roomees_str = parsedMessage.roomees;
		// console.log('goo1', roomees_str);
		break;
	case 'roomConnection':
		nump = parsedMessage.nump; numv = parsedMessage.numv;  (function(){if
(document.id('bstats')) document.id('bstats').fade(1);}).delay(1000);
		break;
	case 'newChatMessage':
		newChatMessage();
		break;
	case 'setAnno':
		setAnno(parsedMessage);
		break;
	case 'setMoviesList':
		setMoviesList(parsedMessage);
		break;
	case 'requestFilm':
		requestFilm(parsedMessage);
```

```
            break;
      case 'denyMovie':
            denyMovie(parsedMessage);
            break;
      case 'viewerLeft':
            just_left = parsedMessage.name;
            onViewerLeft(just_left);
            break;
      case 'iceCandidate':
            if (participants[parsedMessage.name])
participants[parsedMessage.name].rtcPeer.addIceCandidate(parsedMessage.candidate
, function (error) {
                  if (error) {
                   console.error("Error adding candidate: " + error);
                   return;
                  }
            })
            else
            if (cinemaGoers[parsedMessage.name])
cinemaGoers[parsedMessage.name].rtcPeer.addIceCandidate(parsedMessage.candidate,
function (error) {
                  if (error) {
                   console.error("Error adding candidate: " + error);
                   return;
                  }
            })
          break;
      default:
            console.error('Unrecognized message', parsedMessage);
      }
}

if (isAndroid) {
screen.orientation.onchange = function (){

    if (document.id('viewer_menu'))
document.id('viewer_menu').setStyles({'opacity': 0,'zIndex': 0});
    let vis = i_am_viewer && vcounter === 1 && cine ? small_device ? 1 : 1 : 0;
    let zi = i_am_viewer && vcounter === 1 && cine  ? 10111 : 0;
    if (document.id('viewer_menu')) setTimeout(function() {window.innerWidth <
window.innerHeight && document.id('viewer_menu').setStyles({'zIndex': zi}) &&
document.id('viewer_menu').fade(vis);}, 1000);
};
} else if (check_iOS()) {
    window.onorientationchange = function () {
      if (document.id('viewer_menu'))
document.id('viewer_menu').setStyles({'opacity': 0,'zIndex': 0});
```

```
      let vis = i_am_viewer && vcounter === 1 && cine ? small_device ? 1 : 1 :
0;
      let zi = i_am_viewer && vcounter === 1 && cine  ? 10111 : 0;
      if (Math.abs(window.orientation) === 90) {
        // Landscape
      } else {
     if (document.id('viewer_menu')) setTimeout(function()
{document.id('viewer_menu').setStyles({'zIndex':
zi});document.id('viewer_menu').fade(vis);}, 1000);
      }
    }
}

function rejoin(){
     already_being_played = false; // ?! to prevent local sound from played
video
     leaveRoom(); if (i_am_viewer) leaveRoom(); register();
}

function check_locked(par) { // let guru/user keep connected on server resets,
guest leave
     if ((vcounter == 0 || vcounter == '0') && (pcounter == 0 || pcounter ==
'0') && !cine) { console.log('locked!'); if (role == 0) (function() {if (par)
clearAllCookies(); location.reload();}).delay(100)}
}

const check_connection = () => {

     connection_is_good = 0;
     var message={id : 'checkConnection'};
     sendMessage(message);
     setTimeout(function() { if (!connection_is_good) { if (suspected) problems
= 1; if (!problems) suspected = 1; already_clicked = false; if (playSomeMusic)
{setCookie('fmode',22,14400); let myname = document.id('name').value; let
myvideo = 'video-' + myname; if (document.id(myvideo))
{setCookie('cT_'+curMoviesList[0], document.id(myvideo).currentTime, 14400);
/*console.log('saving cT as', document.id(myvideo).currentTime, 'for movie',
curMoviesList[0]);*/}} aonly = 1; cammode = 0; playSomeMusic = 0;
shareSomeScreen = 0; console.log('resetting connection');
document.id('phones').innerHTML = warning;
document.id('room_selector_box').style.zIndex = -10000; (function()
{ document.id('phones').fade(1)}).delay(1000);} else {if (problems) {} else {if
(pcounter == 0 && role != 0) {}}}}, 1200);
}

const check_fullscreen_strict = () => {
```

```
request('https://'+window.location.hostname+port+'/cgi/genc/get_acc_id.pl').then
(data => {
        if (fullscreen) {
           if ( !data.length || window.screenTop ||  window.screenY || (!
(window.innerWidth == screen.width && window.innerHeight == screen.height) &&
isAndroid) ) {
                 fullscreen = false; if (!data.length && cinemaEnabled) rejoin();
else fullscreen = true;
           }
        } else {
           if ((window.innerWidth == screen.width && window.innerHeight ==
screen.height) || (!window.screenTop && !window.screenY) || window.fullScreen )
                 if (!data.length) rejoin();
        }
     });

}

function check_fullscreen() {

request('https://'+window.location.hostname+port+'/cgi/genc/get_acc_id.pl').then
(data => {
        if (fullscreen) {
           if ( !data.length || window.screenTop ||  window.screenY || (!
(window.innerWidth == screen.width && window.innerHeight == screen.height) &&
isAndroid) ) {
                 fullscreen = false; if (!data.length) rejoin(); else {fullscreen
= true;}
           }
        }
     });
}

function signalGuru(e) {

     e.preventDefault();
     e.stopPropagation();

     var message={id : 'signalGuru'};
     sendMessage(message);

}

function eventHandler(n) {

           var message={id :'keyDown', num: n};
           sendMessage(message);
```

```
}

function leftHandler(e) {
     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {
          let role = respo || 0;
          e.preventDefault();
          e.stopPropagation();
          let n=parseInt(document.id('leftnum').innerHTML)+1;
          document.id('leftnum').innerHTML = n.toString();
          var message={id :'plus', s: 'l', num: n};
          sendMessage(message);

          if (role != 1) {
               setCookie('vote',1,144000);
               he_votado = true;
               //document.id('leftplus').removeEventListener('click',
leftHandler);
               //document.id('rightplus').removeEventListener('click',
rightHandler);
          }
     }).catch(err => console.log(err));
}

function noHandler(e) { }
function rightHandler(e) {
     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {
          let role = respo || 0;
          e.preventDefault();
          e.stopPropagation();
          let n=parseInt(document.id('rightnum').innerHTML)+1;
          document.id('rightnum').innerHTML = n.toString();
          var message={id :'plus', s: 'r', num: n};
          sendMessage(message);
          he_votado = true;

          if (role != 1) {
               setCookie('vote',1,144000);
               he_votado = true;
               //document.id('leftplus').removeEventListener('click',
leftHandler);
               //document.id('rightplus').removeEventListener('click',
rightHandler);
          }
     }).catch(err => console.log(err));
}
```

```
const register = () => {

     if (document.id('house')) document.id('house').style.visibility='hidden';
     if (document.id('room_name_id'))
document.id('room_name_id').style.display='none';
     if (document.id('preroom'))
document.id('preroom').style.visibility='hidden';
     var av = getCookie('av');
     if (av && guru_is_here) aonly = 0;

     let sem  = window.innerWidth > 1024 ? '7' : '';

     if (!aonly) { if (document.id('av_toggler'))
document.id('av_toggler').className = "bigO av_toggler_f" } else { if
(document.id('av_toggler')) document.id('av_toggler').className = "bigO
av_toggler"; document.id('bcam').className="bigO bcam";
document.id('fcam').className="bigO fcam"; }

     if (ws.readyState === WebSocket.OPEN) { problems = 0; suspected = 0; }

     if (ws.readyState === WebSocket.CLOSED) {

          ws = new WebSocket('wss://' + location.host + '/groupcall');
          ws.onmessage = function(message) {
               var parsedMessage = JSON.parse(message.data);
               switch (parsedMessage.id) {
                    case 'youLeft':
                         rejoin();
                         break;
                    case 'pingConn':
                         problems = 0; suspected = 0;
                         sendMessage({id : 'replyPing'});
                         break;
                    case 'existingParticipants':
                         check_connection();
                         setTimeout(function()
{onExistingParticipants(parsedMessage);}, 100);
                         break;
                    case 'existingViewers':
                         onExistingViewers(parsedMessage);
                         break;
                    case 'newParticipantArrived':
                         if (!fullscreen) onNewParticipant(parsedMessage);
                         break;
```

```
                        case 'newViewerArrived':
                             onNewViewer(parsedMessage);
                             break;
                        case 'participantLeft':
                             onParticipantLeft(parsedMessage);
                             break;
                        case 'receiveVideoAnswer':
                             receiveVideoResponse(parsedMessage);
                             break;
                        case 'changeTab':
                             changeTabLR(parsedMessage);
                             break;
                        case 'bongoKey':
                             bongoKey(parsedMessage);
                             break;
                        case 'clearCoo':
                             clearTimeoutAndLeave();
                             break;
                        case 'newDrop':
                             newDrop(parsedMessage);
                             break;
                        case 'setGuru':
                             setGuru(parsedMessage);
                             break;
                        case 'askGuru':
                             askGuru(parsedMessage);
                             break;
                        case 'setCinema':
                             setCinema(parsedMessage);
                             break;
                        case 'goodConnection':
                             connection_is_good = 1;
                             roomees_str = parsedMessage.roomees;
                             // console.log('goo2', roomees_str);
                             break;
                        case 'roomConnection':
                             nump = parsedMessage.nump; numv = parsedMessage.numv;
(function(){if (document.id('bstats'))
document.id('bstats').fade(1);}).delay(2000);
                             break;
                        case 'newChatMessage':
                             newChatMessage();
                             break;
                        case 'setAnno':
                             setAnno(parsedMessage);
                             break;
                        case 'setMoviesList':
```

```
                        setMoviesList(parsedMessage);
                        break;
                    case 'requestFilm':
                        requestFilm(parsedMessage);
                        break;
                    case 'denyMovie':
                        denyMovie(parsedMessage);
                        break;
                    case 'viewerLeft':
                        just_left = parsedMessage.name;
                        onViewerLeft(just_left);
                        break;
                    case 'iceCandidate':
                        if (participants[parsedMessage.name])
participants[parsedMessage.name].rtcPeer.addIceCandidate(parsedMessage.candidate
, function (error) {
                            if (error) {
                                    console.error("Error adding candidate: " +
error);
                                    return;
                            }
                        })
                        else
                        if (cinemaGoers[parsedMessage.name])
cinemaGoers[parsedMessage.name].rtcPeer.addIceCandidate(parsedMessage.candidate,
function (error) {
                            if (error) {
                               console.error("Error adding candidate: " +
error);
                               return;
                                }
                        })
                    break;
                    default: console.error('Unrecognized message',
parsedMessage);
                }
            }
        } else {}

        name = document.id('name').value; let cookie_name = loadData('name');
        if (!name.length) name = cookie_name;
        if (!name.length || name === 'null') {name=makeid(8); name = name+'_'+name;
saveData('name', name, 1440);}

        var gi = new RegExp('GURU:','g');
        if (name.match(gi) && cine) {name=makeid(8); name = 'GURU:'+name+'_'+name;
saveData('name', name, 1440);document.id('name').value=name}
```

```
     if (i_am_dummy_guest) { //it wouldn't help?!


     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {

                let role = respo || 0;

                register_body(role);
                i_am_dummy_guest = false; //only one time is enough?!

           }).catch(err => console.log(err));
     } else { register_body(role);}

}

const register_body = (ro) => {

           var plr = getCookie('player');
           //var mo = getCookie('cinemaMode');

           let w = window.location.hostname.split('.');
           let room = document.id('roomName').value == '' ? w[0] :
document.id('roomName').value;
           let house = w[0];

           let curip = document.id('curip').value;
           registered = 1;
           window.removeEventListener("visibilitychange", visChanger);
           if (small_device && window == window.top)
window.addEventListener("visibilitychange", visRejoiner);

           let sem  = window.innerWidth > 1024 ? '7' : '';
           if (temporary && ro == 0) role = 3;

           if (ro == -1) {soundEffect.src = "/sounds/lock.mp3";
setTimeout(function() {location.reload()}, 1200); console.log('Knock out2');
return false;}

           //define in case the onload retarded; doesn't help in iOS?
           let l = checkLang();
           change_lang(altlang[l]);

           document.id('room-header').innerText = 'ROOM ' + room;

           document.id('join').style.display = 'none';
```

```
            document.id('room').style.display = 'block';

            document.id('preroom').innerHTML=' ';
            document.id('postroom').style.display = 'none';

            (function() {
                  document.id('house').style.minHeight = '0px';
                  document.id('house').style.background = 'transparent';
                  document.id('house').style.display = 'block';
                  document.id('house').style.visibility = 'visible';

            }).delay(1000);

            document.id('phones').style.display = 'block';
            document.id('phones').style.visibility = 'visible';
            document.id('cr').style.display = 'none';
            document.id('av_selector').style.display = 'block';
            document.id('poll').style.display = 'block';
            if (voting_shown) {document.id('leftnum').style.display = 'block';
document.id('rightnum').style.display = 'block';}

            let curr_all_muted = getCookie('all_muted') || false;
            let my_mic_muted = loadData(document.id('name').value+'_muted');

            document.id('all_muter').className = check_iOS() && my_mic_muted !==
true && my_mic_muted !== 'true' ? "bigO my_mic_on_all_off" : (my_mic_muted ===
true || my_mic_muted === 'true') && !curr_all_muted && role ? "bigO
my_mic_off" : curr_all_muted ? "bigO allmuter_off" : "bigO allmuter_on";



            document.id('all_muter').title = document.id('all_muter').className
=== "bigO my_mic_on_all_off" ? 'Turn on sound' :
document.id('all_muter').className === "bigO allmuter_off" ? 'Turn on sound' :
document.id('all_muter').className === "bigO my_mic_off" ? 'Turn on
microphone' : 'Turn off sound'

            document.id('leftplus').style.display = 'block';
            document.id('rightplus').style.display = 'block';

            if (( (ro == 1) || !he_votado) && firstTime) {

                  //document.id('leftplus').addEventListener('click',
leftHandler);
                  //document.id('rightplus').addEventListener('click',
rightHandler);
                  addEventListener("keydown", (event) => {
                        //event.keyCode === 65 ? leftHandler(event) :
event.keyCode === 76 ? rightHandler(event) : noHandler(event);
```

```
                    let n = event.keyCode ? event.keyCode : 65;
                    if (n === 66) { if (document.id('controls').style.display
=== 'none' && !small_device) document.id('message_wrap').style.marginTop =
'48vh'; cli5(); }
                    eventHandler(n);
               });
               firstTime = false;
          }

          document.id('fmode_selector').style.display = 'block';
               document.id('newsub').style.display = 'block';



          if(document.id('rh_container'))
document.id('rh_container').style.display = 'none';

          // brute force
          all_muted = getCookie('all_muted');

          if (all_muted === true || all_muted === 'true' || my_mic_muted ===
true || my_mic_muted === 'true') i_am_muted = true;



          let mode = (i_am_muted === true || i_am_muted === 'true') ? 'm' :
aonly ? 'a' : 'v';
          mode = plr ? 'p' : mode;

          let tok = getCookie('authtoken') || '';

          if (ro == 0 && hack) role = 1;

          let formData = new FormData();
          formData.append('addr', curip);



     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{body: formData, method: 'post', credentials: 'include'}).then(respo =>
respo.text()).then((respo) => {
               let role = respo || 0;
          }).catch(err => console.log(err));

request('https://'+window.location.hostname+port+'/cgi/genc/get_acc_id.pl').then
(data=>{
```

```
            let message = {
              id : 'joinRoom',
              name : name,
              mode : mode,
              room : room,
              house : house,
              curip: curip,
              acc_id: data,
              token: tok,
              role: role,
              currRoom: currRoom
            }

            sendMessage(message);
          });

          if (problems) {
               document.id('phones').style.paddingTop = small_device ? '39vh' :
'45vh'; document.id('phones').style.lineHeight = '36px';
document.id('phones').innerHTML = warning;
               (function() { document.id('phones').fade(1)}).delay(500);
          }

          if(stats_shown) { (function()
{document.id('stats').style.display='block';
document.id('stats').fade(1);}).delay(1000);}

          if (!small_device && document.id('helpdoc')) (function() {let l =
checkLang();

          if (small_device) {document.id('helpdoc').style.paddingTop = "0.4vh";
document.id('helpdoc').style.paddingRight = "2vw";}

          if (!small_device && window != window.top)
{document.id('helpdoc').style.marginRight = "2vw";}
document.id('helpdoc').fade(1);}).delay(500);
          if (small_device && window != window.top)
{document.id('message_wrap').style.marginLeft = "-
10px";document.id('message_wrap').style.width =
"330px";document.id('antichatter').style.marginLeft = "290px";}

          document.body.style.overflowY = 'auto';

          (function() {dummies = true;}).delay(3000);

          document.body.scrollTop = document.documentElement.scrollTop = 0;
```

```
          // IMPORTANT: get SDP_ALREADY_NEGOTIATED for camera auto re-
activation, so leave it only for players?!

          if (role == 1 && !already_clicked && problems) {
               already_clicked = true; let ca = getCookie('fmode'); let av =
getCookie('av');
               (function() {
                    if (ca == 0 && av) {console.log('clicking 2!');cli2();}
                    if (ca == 1 && av) {console.log('clicking 3!');cli3();}
                    if (ca == 22) { playSomeMusic=true; getFile(); cli6();
                         if (document.id('asender').value.length) { //hack ash
: this value is set only manually?
                              let  a =
document.id('asender').value.replace('/g, ''); let tok =
getCookie('authtoken');let counter = 0;
                              const inter = setInterval(function() {
                                   counter++; sendMessage({id : 'setAnno',
anno:a, addr: document.id('name').value, token: tok});
                                   /*console.log('here token',
tok,'val',document.id('asender').value,'name',document.id('name').value,
'counter', counter);*/
                                   if (document.id('anno_' +
document.id('name').value) || counter > 5) {
                                        clearInterval(inter);
                                        let myvideo = 'video-' +
document.id('name').value;
                                        if (document.id(myvideo)) {

     document.id(myvideo).pause();__playing = false;
                                        }
                                   }
                              }, 3000);
                         }
                    }
                    console.log('auto re-connect, ca is', ca, 'av is', av);
               }).delay(200);
          }

          setTimeout(function() {check_locked(1)}, 5000); // give more time
here to avoid slow server push to exit
}

function checkLang() {
     let l = getCookie('lang');
     l = (l === null || l === 'null') ? w[0].match(new RegExp('rgsu','g')) ? 1 :
0 : l === '' ? 0 : l;
```

```
     return l;
}

const onNewViewer = (request) => {

     if (request.currRoom && currRoom != request.currRoom) return false; // do
not connect those coming to a room other than mine current

     let myname = document.id('name').value; let myvideo = 'video-' + myname;
     if ((now_playing && cinemaEnabled && playSomeMusic && request.accid.length)
|| (now_playing && !cinemaEnabled && playSomeMusic)) (function() {if
(document.id(myvideo)) document.id(myvideo).play();__playing =
true;}).delay(1000);

     if (request.ng) {if (document.id('num_guests'))
document.id('num_guests').innerHTML = request.ng;}
     room_limit = (typeof request.rl !== 'undefined') ? request.rl : room_limit;

          let na = request.name.split('_');
          let short_name = na[0];
          let suf = na[na.length-1];

          let f = request.name;
          let t = request.curip;
          let vi = 'video-' + f;

          if (document.id('_au_'+suf)) document.id('_au_'+suf).dispose();

          let medal = !temporary ? '<span id=_o_'+suf+'
style="display:none;cursor:pointer;font-size:36px;border-radius:20px;"
onclick="if (this.style.background === \'#9cf\') {set_guru(3,\'' + f +
'\');this.style.background=\'transparent\';this.onclick=function()
{set_guru(2,f);this.style.background=\'#9cf\';}} else {set_guru(2,\'' + f +
'\');this.style.background=\'#9cf\';this.onclick=function(){set_guru(3,\'' + f +
'\');this.style.background=\'transparent\';}}">O</span> ' : '';

          let audi = document.id('audience_boxx').innerHTML == 'Audience is
empty :(' ? '' : document.id('audience_boxx').innerHTML;
          audi = '<div id=_au_'+suf +'>' + medal + '<span id=au_' + suf + '
style="color:#9cf;cursor:pointer;" onclick="set_guru(1,\'' + f +
'\');">'+short_name + '</span>, ' + t + ' <span style="cursor:pointer;"
onclick="drop_guest(\'' + f + '\')" >X</span></div>' + audi;
          document.id('audience_boxx').innerHTML = audi;

          let theati = document.id('audience_boxx').innerHTML == 'Audience is
empty :(' ? '' : document.id('theater_rows_boxx').innerHTML;
```

```
            if (myname != request.name) {

               let left_random = 60 + Math.floor(Math.random() * 400);
               let top_random = 60 + (-1) * Math.floor(Math.random() *30);

               let x = document.id('video-' + f) && document.id('video-' +
f).muted == true ? 'X' : '';

               theati = theati + '<div id=_th_'+suf +'
style="position:absolute;left:'+left_random+'px;top:'+top_random+'px;width:60px;
height:60px;border-radius:30px;background:#012;padding-
top:45px;cursor:pointer;z-index:2000;" onclick="if (document.id(\'' + vi + '\'))
{ document.id(\'' + vi + '\').muted=(document.id(\'' + vi + '\') &&
document.id(\'' + vi + '\').muted == true) ? false : true; document.id(\'sndm_'
+ f + '\').innerText=(document.id(\'' + vi + '\') && document.id(\'' + vi +
'\').muted == true) ? \'X\' : \'\';}"><div id="sndm_'+f+'" class="snd_meters"
style="z-index:10112;font-size:16px;color:#fed;">' + x + '</div><span id=th_' +
suf + ' style="color:#9cf;cursor:pointer;">'+t+'</span></div>';
            }

            if (document.id('theater_rows_boxx') && cine && !normal_mode &&
i_am_viewer) (function() {document.id('theater_rows_boxx').innerHTML =
theati;}).delay(2000);

            let ar_split = audi.split('_au_');
            let cur = ar_split.length – 1;

            if (myname != request.name && i_am_viewer && cine)
receiveCGAudio(request.name, 'a');

            document.id('audience_numbers').innerHTML = cur;
            let col = cur > 0 ? '#369' : '#ccc';
            document.id('audience_numbers').setStyles({'color': col});

            vcounter = cur; if (document.id('vcounter'))
document.id('vcounter').innerHTML = vcounter;

            let vis = i_am_viewer && vcounter === 1 && cine ? small_device ? 1 :
1 : 0;
            let zi = i_am_viewer && vcounter === 1 && cine  ? 10111 : 0;
            if (document.id('viewer_menu') && !lori) setTimeout(function()
{document.id('viewer_menu').setStyles({'zIndex':
zi});document.id('viewer_menu').fade(vis);}, 1000);

            if (just_left != f && document.id('name').value != f &&
document.id('name').value != just_left) { if (cine) {soundEffect.src =
"/sounds/coin.mp3";} else {soundEffect.src = "/sounds/steps.mp3";}}
}
```

```
const onNewParticipant = (request) => {

  if (request.currRoom && currRoom != request.currRoom) return false; // do not
connect those coming to a room other than mine current

  if (request.ng) {if (document.id('num_guests'))
document.id('num_guests').innerHTML = request.ng;}

      let myrole = role;

      let myname = document.id('name').value;
      let myvideo = 'video-' + myname;
      if (now_playing) (function() {document.id(myvideo).play();__playing =
true;}).delay(3000);

      var theCookies = document.cookie.split(';');
      var really_new = 1;
      var video_hidden = 0;

      room_limit = (typeof request.rl !== 'undefined') ? request.rl : room_limit;

      for (var i = 1 ; i <= theCookies.length; i++) {

            if ( theCookies[i-1].match(request.name+'_hidden') )
            {

                        video_hidden = 1;
                        really_new = 0;
                        break;
            }

            if (!video_hidden) {
                        var ls_muted =
localStorage.getItem(request.name+'_muted');
                  if ( theCookies[i-1].match(request.name+'_muted') || (ls_muted !
== null && ls_muted !== 'null') )
                  {

                        really_new = 0;
                        break;
                  }
            }

      }

      if (really_new) soundEffect.src = "/sounds/coin.mp3";
```

```
      let my_mic_muted = loadData(document.id('name').value+'_muted');
      document.id('all_muter').className = check_iOS() && my_mic_muted !== true
&& my_mic_muted !== 'true' ? "bigO my_mic_on_all_off" :
document.id('all_muter').className;

      if (!video_hidden) {

            //preparing logics in advance
            let pctr = pcounter + 1;
            if (pctr > room_limit - 1 && i_am_viewer) {if (document.id('bell'))
document.id('bell').style.display = 'block'; if (document.id('av_toggler'))
document.id('av_toggler').style.display='none';}

            if (!small_device && window == window.top) resizer(pctr);

            if (small_device || tablet) {
             // new guy
             document.id('participants').style.marginTop = '-68vh';
             if (small_device && pcounter === 0)
document.id('participants').style.height = window == window.top ? '50vh' :
'63vh';
             if (small_device && pcounter === 1)
document.id('participants').style.height = window == window.top ? '110vh' :
tablet ? '140vh' : '128vh';
             if (small_device && pcounter === 2)
document.id('participants').style.height = window == window.top ? '160vh' :
'190vh';
             if (small_device && pcounter === 3)
document.id('participants').style.height = window == window.top ? '220vh' :
'250vh';
             if (small_device && pcounter > 3)
document.id('participants').style.height = window == window.top ? '275vh' :
'300vh';

            }

            receiveVideo(request.name, request.mode, myrole);

            (function() {if (document.id(request.name))
{document.id(request.name).style.display='block';
document.id(request.name).fade(1);}}).delay(500); //need this animation because
the new video appears under the row, so we hide it

            //set new sound according to all_muter
            let noSound = document.id('all_muter').className === "bigO
my_mic_on_all_off" || document.id('all_muter').className === "bigO allmuter_off"
            let vid = document.id('video-' + request.name);
```

```
            vid.muted = noSound;
            if (cine) vid.muted = true; // serpom po mandarinam

            // should check muted mic here, too
            if (request.mode === 'm') vid.muted = true;

            let spea = document.id('speaker-' + request.name);
            spea.removeChild(spea.childNodes[0]);
            request.mode !== 'm' && (noSound || cine) &&
spea.appendChild(document.createTextNode('\uD83D\uDD07'));//muted icon
            request.mode !== 'm' && !noSound && !cine &&
spea.appendChild(document.createTextNode('\uD83D\uDD0A'));//speaker icon
            request.mode === 'm' &&
spea.appendChild(document.createTextNode('X'));//x icon



            if (request.curip.length && document.id('loco_'+request.name) && !
ValidateIPaddress(request.curip)) {
                  document.id('loco_'+request.name).innerHTML = request.curip;
                  document.id('loco_'+request.name).style.display='block';
                  document.id('loco_'+request.name).fade(1);
            }

            if (cine && role != 1) {
                  (function() {document.id('loco_' +
request.name).fade(0);document.id('span_' +
request.name).fade(0);}).delay(2000);
            }

            if (document.id('acco_'+request.name) &&
ValidateAccountId(request.acc_id) ) {
                  document.id('acco_'+request.name).style.display='block';
                  let na = request.name.split('_');
                  let ac = request.acc_id;
                  document.id('acco_'+ request.name).fade(1);
                  document.id('acco_' + request.name).onclick = function(e) {
                        e.preventDefault(); e.stopPropagation(); copy(ac);
flashText('copied '+ na[0]);

request('https://'+window.location.hostname+port+'/cgi/genc/get_acc_id.pl').then
(data => {
                              setTimeout(function() {
                                    if (document.id('removerA'))
{document.id('removerA').innerHTML = 'Error: Service unavailable';
                                    (function()
{document.id('removerA').fade(0)}).delay(1000);}
                              }, 10000);
```

```
                              (function() {
                                   if (document.id('sp_balance')){

     document.id('sp_balance').style.display='block';

     document.id('sp_balance').src=sp_container_url+'/?acc='+data;
                                   }
                              }).delay(1000);
                              document.id('sp_container').style.display = 'block';
sp_shown = 1;
                              ch_int = setInterval(function() {
                                   if (document.id('sp_balance')) {

     document.id('sp_balance').style.display='block';
                                        document.id('sp_balance').src =
sp_container_url + '/?acc=' + data;
                                   }
                              }, 300000);
                         });
                }
                if (document.id('sp_container' && sp_shown) &&
document.id('sp_container').style.display != 'block')
document.id('acco_'+request.name).style.visibility='hidden';
           }

     }
           let vis = i_am_viewer && vcounter === 1 && cine && (request.mode ==
'p' || request.mode == 'c')? small_device ? 1 : 1 : 0;
           let zi = i_am_viewer && vcounter === 1 && cine  ? 10111 : 0;
           if (document.id('viewer_menu') && !lori) setTimeout(function()
{document.id('viewer_menu').setStyles({'zIndex':
zi});document.id('viewer_menu').fade(vis);}, 1000);
}

function receiveVideoResponse(result) {

     if (participants[result.name])
{participants[result.name].rtcPeer.processAnswer (result.sdpAnswer, function
(error) {
           if (error) return console.error (error);
     })

     }
     else {
      if (cinemaGoers[result.name])
{cinemaGoers[result.name].rtcPeer.processAnswer (result.sdpAnswer, function
(error) {
           if (error) return console.error (error);
```

```
       })
       }
      }
}

function callResponse(message) {
      if (message.response != 'accepted') {
            console.info('Call not accepted by peer. Closing call');
            stop();
      } else {
            webRtcPeer.processAnswer(message.sdpAnswer, function (error) {
                  if (error) return console.error (error);
            });
      }
}

async function startCapture(displayMediaOptions) {

      let captureStream = null;

      try {
            captureStream = await
navigator.mediaDevices.getDisplayMedia(displayMediaOptions);
      } catch(err) {
            console.error("Error: " + err);
      }

      return captureStream;
}

const onExistingViewers = (msg) => {

   let myname = document.id('name').value;

   let audience = '';
   let theater_rows = '';

   let play_sound = myname == just_left ? false : true;

   if (msg.ng) {if (document.id('num_guests'))
document.id('num_guests').innerHTML = msg.ng;}

   room_limit = (typeof msg.rl !== 'undefined') ? msg.rl : room_limit;

   if (msg.data) {
         var arr = msg.data.sort();
```

```
        for (var i = 0; i < arr.length; i++) {
          var chu = arr[i].split('_|_');

          let f = chu[0];
          let t = chu[1];

          if (f == just_left) {
               just_left = '';
               play_sound = false;
          }

          let na = f.split('_');
          let short_name = na[0];
          let suf = na[na.length-1];

          if (f == myname) {

               play_sound = false;
               var lang = getCookie('lang');
               if (t.length && (lang === null || lang === 'null')) {
                    let ar = /, AR$/.test(t);
                    let br = /, BR$/.test(t);
                    let ru = /, RU$/.test(t);

                    if (ar) {ctr = 2} else {if (br) {ctr = 5} else {if (ru)
{ctr = 1} else {}}}
                    change_lang(altlang[ctr]);
               }
          }

          let left_random = f != myname ? 60+ Math.floor(Math.random() * 400) :
10;
          let top_random = f != myname ? 60 + (-1) * Math.floor(Math.random() *
30) : 90;

          let rs = f != myname ? '60px': '60px';
          let br = f != myname ? '30px': '30px';
          let bg = f != myname ? '#012': '#012';

          if (i_am_viewer && myname != f && cine) {

               receiveCGAudio(f, 'a');

               let x = document.id('video-' + f) && document.id('video-' +
f).muted == true && myname != f ? 'X' : '';
               x = document.id('video-' + f) && myname === f ?
localStream.getAudioTracks()[0].enabled ? '' : 'X' : x;
```

```
                x = document.id('video-' + f) ? x : '';

                theater_rows = theater_rows + '<div id=_th_'+suf +'
style="position:absolute;left:'+left_random+'px;top:'+top_random+'px;width:'+rs+
';height:'+rs+';border-radius:'+br+';background:'+bg+';padding-
top:45px;cursor:pointer;z-index:2001;" onclick="if (document.id(\'video-' + f +
'\')) {document.id(\'video-' + f + '\').muted=document.id(\'video-' + f +
'\').muted == true ? false : true; document.id(\'sndm_' + f +
'\').innerText=document.id(\'video-' + f + '\').muted ==
true ? \'X\' : \'\';}"><div id="sndm_'+f+'" class="snd_meters" style="z-
index:10112;font-size:16px;color:#fed">' + x + '</div><span id=th_' + suf + '
style="color:#9cf;cursor:pointer;">' + t + '</span></div>';
           }

           if (i_am_viewer && myname === f && !i_am_muted && cine) {
                //console.log('Here we activate local camera!')
                var cinemaGoer = new CinemaGoer(f, document.id('name').value,
'a' );
                cinemaGoers[f] = cinemaGoer;let vi =

                g.video = cinemaGoer.getVideoElement();

                var constraints = {
                     audio : true,
                     video: false
                };

                var options = {
                     localVideo: g.video,
                     mediaConstraints: constraints,
                     onicecandidate: cinemaGoer.onIceCandidate.bind(cinemaGoer)
                }

                cinemaGoer.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(options, function(error) {
                        if(error) {
                     var ff = new RegExp('closed','ig');
                     if (error.toString().match(ff)) { } else {
                          cinemaGoer.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(options, function (error) {
                                        if(error) {
                                                return console.error(error);
                                        }

                                            startVideo(g.video);
                                            this.generateOffer
(cinemaGoer.offerToReceiveVideo.bind(cinemaGoer));
```

```
                                localStream =
cinemaGoer.rtcPeer.getLocalStream();
                                });
                        }

                        return false;
                           } else {
                              startVideo(g.video);
                              this.generateOffer
(cinemaGoer.offerToReceiveVideo.bind(cinemaGoer));
                        localStream = cinemaGoer.rtcPeer.getLocalStream();

                        x = document.id('video-' + f) && myname === f ?
localStream.getAudioTracks()[0].enabled ? '' : 'X' : x;

                        theater_rows = theater_rows + '<div id=_th_'+suf +'
style="position:absolute;left:'+left_random+'px;top:'+top_random+'px;width:'+rs+
';height:'+rs+';border-radius:'+br+';background:'+bg+';padding-
top:45px;cursor:pointer;z-index:2001;" onclick="if (document.id(\'video-' + f +
'\')) {localStream.getAudioTracks()[0].enabled = localStream.getAudioTracks()
[0].enabled === true ? false : true; document.id(\'sndm_' + f + '\').innerText =
localStream.getAudioTracks()[0].enabled === true ? \'\' : \'X\';}"><div
id="sndm_'+f+'" class="snd_meters" style="z-index:10112;font-
size:16px;color:#fed">' + x + '</div><span id=th_' + suf + '
style="color:#9cf;cursor:pointer;">' + t + '</span></div>';
                     }
                  });
            }

            let coo_volume = loadData(f+'_volume');
            if (document.id('slider_' + f)) document.id('slider_' + f).value =
coo_volume;
            if (document.id('video-' + f)) document.id('video-' + f).volume =
coo_volume;

            if (document.id('_au_'+suf)) document.id('_au_'+suf).dispose();
            if (document.id('_th_'+suf)) document.id('_th_'+suf).dispose();

            let medal = !temporary ? '<span id=_o_'+suf+'
style="display:none;cursor:pointer;font-size:36px;border-radius:20px;"
onclick="if (this.style.background === \'#9cf\') {set_guru(3,\'' + f +
'\');this.style.background=\'transparent\';this.onclick=function()
{set_guru(2,f);this.style.background=\'#9cf\';}} else {set_guru(2,\'' + f +
'\');this.style.background=\'#9cf\';this.onclick=function(){set_guru(3,\'' + f +
'\');this.style.background=\'transparent\';}}">O</span> ' : '';
            audience = audience + '<div id=_au_'+suf +'>' + medal + '<span
id=au_' + suf + ' style="color:#9cf;cursor:pointer;" onclick="set_guru(1,\'' + f
+ '\');">'+short_name + '</span>, ' + t + ' <span style="cursor:pointer;"
onclick="drop_guest(\'' + f + '\')" >X</span></div>';
```

```
        } //for

        if (arr.length == 0) {audience = 'Audience is empty :
(';document.id('audience_numbers').setStyles({'color':'#ccc'});document.id('audi
ence_numbers').innerHTML = '...'; audience_numberr = 0; vcounter = 0; if
(document.id('vcounter')) (function(){document.id('vcounter').innerHTML =
vcounter;}).delay(500);} else
{document.id('audience_numbers').setStyles({'color':'#369'});
document.id('audience_numbers').innerHTML = arr.length;

          if (arr.length > audience_numberr && arr.length > 0 && play_sound) {
               soundEffect.src = "/sounds/steps.mp3";
          }
          audience_numberr = arr.length; vcounter = arr.length; if
(document.id('vcounter')) document.id('vcounter').innerHTML = vcounter;
        }
        if (document.id('audience_boxx')) document.id('audience_boxx').innerHTML
= audience;
        if (document.id('theater_rows_boxx') && cine && !normal_mode &&
i_am_viewer) (function() {document.id('theater_rows_boxx').innerHTML =
theater_rows;}).delay(2000); // wait localstream

   }

   let vis = i_am_viewer && vcounter === 1 && cine ? small_device ? 1 : 1 : 0;
   let zi = i_am_viewer && vcounter === 1 && cine  ? 10111 : 0;
   if (document.id('viewer_menu') && !lori) setTimeout(function()
{document.id('viewer_menu').setStyles({'zIndex':
zi});document.id('viewer_menu').fade(vis);}, 1000);
}

const set_guru = (par, who) => {
     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {
          let role = respo || 0;

          if (role == 1) {
               let tok = getCookie('authtoken') || '';
               let message = {
                    id : 'setGuru',
                    name : who,
                    mode : par,
                    token: tok
               }
               sendMessage(message);
```

```
          } else {

               if (document.id('name').value == who ) {
                    chat_shown = 1; document.id('logger').click();
document.id('audience').click(); rejoin();
               }
          }
     }).catch(err => console.log(err));
}

function drop_guest(who) {

     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {
          let role = respo || 0;

          if (role == 1) {
               let tok = getCookie('authtoken') || '';
               let message = {
                    id : 'dropGuest',
                    name : who,
                    token: tok
               }
               sendMessage(message);
               vcounter--; if (document.id('vcounter'))
document.id('vcounter').innerText = vcounter;
               if (vcounter === 0)
{document.id('audience_numbers').setStyles({'color':'#ccc'});document.id('audien
ce_numbers').innerHTML = '...'; document.id('audience_boxx').innerText =
'Audience is empty :(';}
               if (vcounter) document.id('audience_numbers').innerText =
vcounter;

          } else {
               onViewerLeft(who);
          }

          let l = who.split('_');
          if (document.id('_au_'+l[1])) document.id('_au_'+l[1]).dispose();

     }).catch(err => console.log(err));
}

const onExistingParticipants = (msg) => {
//sets up every video in the room I just joined
```

```
  let myname = document.id('name').value;
  let cRoom= '';


   if (role == 0 && hack) role = 1;

   if (msg.num_rooms) num_rooms = msg.num_rooms;
   if (msg.roomees) roomees_str = msg.roomees;

   let room_sel = ''; var sym = 'A';
   roomees_str = roomees_str ? roomees_str :
'room1;room2;room3;room4;room5;room6';
   let roomees_arr = roomees_str ? roomees_str.split(';') : [];

   for (i = 1; i <= num_rooms; i++) {
     sym = i == 2 ? 'B' : i == 3 ? 'C' : i == 4 ? 'D' : i == 5 ? 'E' : i == 6 ?
'F' : sym;
     // let addon = sym == currRoom ? '  <span
style="color:#fed;border:0px solid #fed; padding:3px">ROOM' + i +
'</span>  ' : '  <span style="color:#9cf;cursor:pointer;
padding:3px" onclick="currRoom = \'' + sym +
'\';/*console.log(\'sym\',currRoom);*/(function()
{document.id(\'bg_switch\').click()}).delay(500);flashText_and_rejoin(\'SKIP TO
ROOM \' + currRoom);">ROOM' + i + '</span>  ';
     let addon = sym == currRoom ? '  <span
style="color:#fed;border:0px solid #fed; padding:3px">' + roomees_arr[i-1] +
'</span>  ' : '  <span style="color:#9cf;cursor:pointer;
padding:3px" onclick="currRoom = \'' + sym + '\'; cRoom = \'' + roomees_arr[i-1]
+ '\';/*console.log(\'sym\',currRoom);*/(function()
{document.id(\'bg_switch\').click()}).delay(500);flashText_and_rejoin(\'SKIP
TO \' + cRoom);">\"' + roomees_arr[i-1] + '\"</span>  ';

     room_sel = room_sel + addon;
   }
   if (num_rooms == 1) room_sel = '';
   if (num_rooms < 4) document.id('room_selector_box').style.left='-10px';
   if (num_rooms === 4) document.id('room_selector_box').style.left='-25px';
   if (num_rooms === 5) document.id('room_selector_box').style.left='-80px';
   if (num_rooms > 5) document.id('room_selector_box').style.left='-135px';

   document.id('room_selector').innerHTML = room_sel;

if (temporary && role == 0) role = 3;
```

```
   if (role == 1 || role == 2 || role == 3)  {

      let fmode = getCookie('fmode') ? 'environment' : 'user';

      i_am_viewer = false;

all_muted = getCookie('all_muted');
if (all_muted === true || all_muted === 'true') i_am_muted = true;

      let mode = (i_am_muted === true || i_am_muted === 'true') ? 'm' : aonly ?
'a' : 'v';
      var participant = new Participant(name, myname, mode, role);
      participants[name] = participant;

      g.video = recordedVideo ?  recordedVideo : participant.getVideoElement();

      var canvas = check_iOS() ? participant.getCanvasElement() : '';

      var constraints = {
                audio: true,
                video: {
                        maxWidth : wi_hq,
                        maxFrameRate : fps_hq,
                        minFrameRate : fps_hq,
                facingMode: fmode
                }
      };

      var constraints_vonly = {
                audio: false,
                video: {
                facingMode: fmode
                }
      };

      var constraints_aonly = {
                audio: true,
                video: false
      };

      var constraints_dumb = {
                audio: false,
                video: false
```

```
	};

	if (aonly) constraints = constraints_aonly;

	var constraints_alt = (i_am_muted === true || i_am_muted === 'true' || role
=== 3) ? constraints_vonly : constraints_aonly; // hack ash

	constraints = (i_am_muted === true || i_am_muted === 'true') && aonly ?
constraints_dumb : (i_am_muted === true || i_am_muted === 'true') ?
constraints_vonly : constraints;
	constraints_alt = (i_am_muted === true || i_am_muted === 'true') && aonly ?
constraints_dumb : (i_am_muted === true || i_am_muted === 'true') ?
constraints_vonly : constraints_alt;



	var options = {
		localVideo: g.video,
		mediaConstraints: constraints,
		onicecandidate: participant.onIceCandidate.bind(participant)
	}

	var options_alt = {
		localVideo: g.video,
		mediaConstraints: constraints_alt,
		onicecandidate: participant.onIceCandidate.bind(participant)
	}
	var options_aonly = {
		localVideo: g.video,
		mediaConstraints: constraints_aonly,
		onicecandidate: participant.onIceCandidate.bind(participant)
	}

	if(recordedVideo && savedSrc) {

		shareSomeStream = true;
		var cstrx = {
			audio: true,
			video:{
				maxWidth : wi,
				maxFrameRate : fps,
				minFrameRate : fps
			}
		};

		options = {
			videoStream: savedSrc,
			onicecandidate: participant.onIceCandidate.bind(participant),
```

```
                mediaConstraints : cstrx
            }
            option_alt = {
                videoStream: savedSrc,
                onicecandidate: participant.onIceCandidate.bind(participant),
                mediaConstraints :{audio: true, video: false}
            }

            participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(options, function(error) {
                     if(error) {
                var ff = new RegExp('closed','ig');
                if (error.toString().match(ff)) {
                } else {
                     participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(option_alt,
                           function (error) {
                                          if(error) {
                                               return console.error(error);
                                          }

                                          startVideo(g.video);
                                          this.generateOffer
(participant.offerToReceiveVideo.bind(participant));
                                localStream =
participant.rtcPeer.getLocalStream();

                                if (small_device)
document.id(myname).style.float = 'none';
                                document.id('room-header-
file').style.display='none';
                           });
                }
                return false;
                      } else {
                           startVideo(g.video);
                     this.generateOffer
(participant.offerToReceiveVideo.bind(participant));
                     localStream = participant.rtcPeer.getLocalStream();

                     if (small_device)  document.id(myname).style.float =
'none';
                     document.id('room-header-file').style.display='none';
                      }
                 (function(){document.id('phones').fade(0);}).delay(1000);
            }); //rtcPeer
```

```
     } else if (shareSomeScreen) {

          shareSomeScreen = true;
          document.id('room-header').style.color = oldColor;

          startCapture({video: true}).then(stream => {
/* mix microphone
*/
          navigator.mediaDevices.getUserMedia({audio: true})
          .then(function(mediaStream) {
          var audioTrack = mediaStream.getAudioTracks()[0] ?
mediaStream.getAudioTracks()[0] : null;
          if (audioTrack && i_am_muted !== true && i_am_muted !== 'true')
stream.addTrack(audioTrack);

          g.video.srcObject = stream;

          var options_sshare = {
               videoStream: stream,
               onicecandidate: participant.onIceCandidate.bind(participant),
               mediaConstraints : {
                    audio: true,
                    video:{
                              maxWidth : wi_hq,
                              maxFrameRate : fps_hq,
                              minFrameRate : fps_hq
                    }
               },
               sendSource : 'screen'
          }

          participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(options_sshare, function(error) {
                 if(error) {
               var ff = new RegExp('closed','ig');
               if (error.toString().match(ff)) {
               } else {
                         participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(options_alt,
                              function(error) {
                                       if(error) {
                                                return console.error(error);
                                       }

                                       startVideo(g.video);
                                       this.generateOffer
(participant.offerToReceiveVideo.bind(participant));
```

```
                            localStream = participant.rtcPeer.getLocalStream();

                            if (small_device)  document.id(myname).style.float =
'none';
                           });
                }

                return false;
                  } else {
                      startVideo(g.video);
                      this.generateOffer
(participant.offerToReceiveVideo.bind(participant));
                localStream = participant.rtcPeer.getLocalStream();

                if (small_device)  document.id(myname).style.float = 'none';
              }

              (function(){document.id('phones').fade(0);}).delay(1000);
            });
            }).catch(function(err){console.log(err.name + ": " + err.message);});
//mediaStream
            }).catch(e => console.log(e)); //startCapture

      } else if (playSomeMusic) {

            if (playSomeMusic_muted === false) {
                  if (!g.audioContext) g.audioContext = new AudioContext();

                  mediaSource = g.audioContext.createMediaElementSource(g.video);
                  analyser = g.audioContext.createAnalyser();
                  mediaSource.connect(analyser);
                  analyser.connect(g.audioContext.destination);
            }

            g.video.autoplay = true;
            g.video.playsInline = true;
            g.video.controls = true;
            g.video.crossOrigin = 'anonymous';
            g.video.volume = 1;
            g.video.loop = true;
            g.video.src = selectedFile ? selectedFile : null;
            g.video.muted = false;

            let cT = curMovie ? getCookie('cT_'+curMovie) : 0;
```

```
            g.video.currentTime = cT > 0 ? cT : 0;

            g.video.addEventListener('canplay', (event) => {

              now_playing = true;

              if (!already_being_played) {

console.log('running', N++);

                  already_being_played = true;
                  let captureStream = null;
/* mix microphone
*/
                  navigator.mediaDevices.getUserMedia({audio: true})
                  .then(function(mediaStream) {
                  var audioTrack = mediaStream.getAudioTracks()[0] ?
mediaStream.getAudioTracks()[0] : null;
                  if (canvas.captureStream) {
                        captureStream = canvas.captureStream(fps_hq);
                        if (audioTrack && i_am_muted !== true && i_am_muted !==
'true') captureStream.addTrack(audioTrack);
                  } else
                  if (g.video.captureStream) {
                        captureStream = g.video.captureStream(fps_hq);
                  } else if (g.video.mozCaptureStream) {
                        captureStream = g.video.mozCaptureStream(fps_hq);
                  } else {
                        console.error('Stream capture is not supported');
                  }

                  //NB: mix audio from mic to that of the video, except when in
cinema
                  let audioStreamTrack = null;
                  if (!cine) {
console.log('doing mic mix in normal mode');
                     if (!g.ctx) g.ctx = new AudioContext();
                     const dest = g.ctx.createMediaStreamDestination();
                     const audioIn_01 = audioTrack ?
g.ctx.createMediaStreamSource(mediaStream) : null;
                     const audioIn_02 = captureStream.getAudioTracks()[0] ?
g.ctx.createMediaStreamSource(captureStream) : null;

                     if (i_am_muted !== true && i_am_muted !== 'true' &&
audioIn_01) audioIn_01.connect(dest);
                     if (audioIn_02) audioIn_02.connect(dest);
```

```
                  audioStreamTrack = dest.stream.getAudioTracks()[0] ?
dest.stream.getAudioTracks()[0] : null;
                }// not cine

                g.captureStreamVideoTrack = captureStream.getVideoTracks()[0] ?
captureStream.getVideoTracks()[0] : null;
                g.captureStreamAudioTrack = captureStream.getAudioTracks()[0] ?
captureStream.getAudioTracks()[0] : null;

                g.mixStream = new MediaStream();
                if (g.captureStreamVideoTrack)
g.mixStream.addTrack(g.captureStreamVideoTrack);
                if (!cine && audioStreamTrack)
g.mixStream.addTrack(audioStreamTrack);
                if (cine && g.captureStreamAudioTrack)
g.mixStream.addTrack(g.captureStreamAudioTrack);

                var cstrx = {
                      audio: true,
                      video:{
                            maxWidth : wi_hq,
                            maxFrameRate : fps_hq,
                            minFrameRate : fps_hq
                      }
                };

                options = {
                      videoStream: g.mixStream,
                      onicecandidate:
participant.onIceCandidate.bind(participant),
                      mediaConstraints : cstrx
                }
                option_alt = {
                      videoStream: g.mixStream,
                      onicecandidate:
participant.onIceCandidate.bind(participant),
                      mediaConstraints :{audio: true, video: false}
                }

                participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(options, function(error) {
                       if(error) {
                      var ff = new RegExp('closed','ig');
                      if (error.toString().match(ff)) {
                      } else {
                                        participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(option_alt,
```

```
                                    function (error) {
                                            if(error) {
                                                  return console.error(error);
                                            }

                                            startVideo(g.video);
                                            this.generateOffer
(participant.offerToReceiveVideo.bind(participant));
                                    localStream =
participant.rtcPeer.getLocalStream();

                                    if (small_device)
document.id(myname).style.float = 'none';
                                    document.id('room-header-
file').style.display='none';
                                       }
                              );
                        }
                        return false;
                         } else {

                              g.video.play();

                        if (playSomeMusic_muted === false) {
                            function getSoundData() {
                               var sample = new
Float32Array(analyser.frequencyBinCount);
                               return analyser.getFloatFrequencyData(sample);
                            }
                        }

                        this.generateOffer
(participant.offerToReceiveVideo.bind(participant));
                        localStream = participant.rtcPeer.getLocalStream();

                        if (small_device)  document.id(myname).style.float =
'none';
                              document.id('room-header-file').style.display='none';
                         }
                  (function(){document.id('phones').fade(0);}).delay(1000);

                  if (!shareSomeStream) getList_names();

                 }); //rtcPeer
                }).catch(function(err){console.log(err.name + ": " +
err.message);}); //mediaStream
```

```
            } //already_being_played
           });//video.addEventListener

     } else {

           participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(options, function(error) {
                 if(error) {
                var ff = new RegExp('closed','ig');
                if (error.toString().match(ff)) {
                } else {
                  participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerSendonly(options,
                         function (error) {
                                      if(error) {
                                              return console.error(error);
                                      }

                                      startVideo(g.video);
                                      this.generateOffer
(participant.offerToReceiveVideo.bind(participant));
                          localStream = participant.rtcPeer.getLocalStream();

                          if (small_device)  document.id(myname).style.float =
'none';
                         });
                }
                return false;
                 } else {
                     startVideo(g.video);
                     this.generateOffer
(participant.offerToReceiveVideo.bind(participant));
                localStream = participant.rtcPeer.getLocalStream();

                i_am_on_air = true;

                if (small_device || tablet) {
                     // myself
                     document.id('participants').style.marginTop =
isIOSFirefox() ? '-46vh' : '-55vh';
```

```
                    if (pcounter === 1)
document.id('participants').style.height = window == window.top ? '50vh' :
'63vh';
                    if (pcounter === 2)
document.id('participants').style.height = window == window.top ? '110vh' :
tablet ? '140vh' : '128vh';
                    if (pcounter === 3)
document.id('participants').style.height = window == window.top ? '160vh' :
'190vh';
                    if (pcounter === 4)
document.id('participants').style.height = window == window.top ? '220vh' :
'250vh';
                    if (pcounter > 4) document.id('participants').style.height
= window == window.top ? '275vh' : '300vh';

                    document.id(myname).style.position = 'absolute';
                    document.id(myname).style.bottom = '0vh';
                    if (pcounter > 1) {
                         document.id('participants').style.position =
'absolute';
                         saved_top = document.id('participants').style.top;
                         document.id('participants').style.top = '81vh';
                    } else {
                         document.id('participants').style.position =
'relative';
                         document.id('participants').style.top =
saved_top.length ? saved_top : document.id('participants').style.top ;
                    }
               }

               if (!small_device && window == window.top) resizer(1);
            }

            (function(){document.id('phones').fade(0);}).delay(1000);
          });

     }

     (function() {document.id(myname).style.display='block';
document.id(myname).fade(1);}).delay(500);//need this animation because the new
video appears under the row, so we hide it
   }// if role

   if (msg.data) {

        cinemaEnabled = false; num_cinemas = 0;
        let top_buf = 0;
        for (var i = 0; i < msg.data.length; i++) {
```

```
            var chu = msg.data[i].split('_|_');

            let f = chu[0];
            let s = chu[1];
            let t = chu[2];
            let ac = chu[3];
            let a = chu[4];

            let na = chu[0].split('_');

            if (s === 'c') {cinemaEnabled = true; num_cinemas += 1;}
            if (f != myname) {
                  //prepare logics in advance
                  let pctr = pcounter +1;
                  if (pctr < room_limit && role == 0)
{document.id('bell').style.display = 'none';
document.id('av_toggler').style.display='block';}
                  if (pctr > room_limit - 1 && role == 0)
{document.id('bell').style.display = 'block';
document.id('av_toggler').style.display='none';}

                  if (!small_device) resizer(pctr);

                  receiveVideo(f, s, role);
                  let coo_volume = loadData(f+'_volume');

                  if (document.id('slider_' + f)) document.id('slider_' + f).value
= coo_volume;
                  if (document.id('video-' + f)) document.id('video-' + f).volume
= coo_volume;

                  (function() {if (document.id(f))
{document.id(f).style.display='block';
document.id(f).fade(1);}}).delay(500);//need this animation because the new
video appears under the row, so we hide it



            } else {
                  var lang = getCookie('lang');
                  if (t.length && (lang === null || lang === 'null')) {
                        let ar = /, AR$/.test(t);
                        let br = /, BR$/.test(t);
                        let ru = /, RU$/.test(t);

                        if (ar) {ctr = 2} else {if (br) {ctr = 5} else {if (ru)
{ctr = 1} else {}}}
                        change_lang(altlang[ctr]);
                  }
```

```
          }

          if (t.length && document.id('loco_' + f) && !ValidateIPaddress(t)) {
               document.id('loco_' + f).innerHTML = t;
               document.id('loco_' + f).style.display='block';
               document.id('loco_' + f).fade(1);
          }

          if (cine) {
               (function() {if (document.id('loco_' + f)) document.id('loco_' +
f).fade(0); if (document.id('span_' + f))document.id('span_' +
f).fade(0);}).delay(2000);
          }

          if (document.id('acco_' + f) && ValidateAccountId(ac)) {
               document.id('acco_' + f).style.display='block';

               document.id('acco_' + f).fade(1);
               document.id('acco_' + f).onclick = function() {
                    copy(ac); flashText('copied '+ na[0]);


request('https://'+window.location.hostname+port+'/cgi/genc/get_acc_id.pl').then
(data => {
                         setTimeout(function() {
                              if (document.id('removerA'))
{document.id('removerA').innerHTML = 'Error: Service unavailable';
                              (function()
{document.id('removerA').fade(0)}).delay(1000);}
                         }, 10000);
                         (function() {
                              if (document.id('sp_balance')){

     document.id('sp_balance').style.display='block';

     document.id('sp_balance').src=sp_container_url+'/?acc='+data;
                              }
                         }).delay(1000);
                         document.id('sp_container').style.display = 'block';
sp_shown = 1;
                         ch_int = setInterval(function() {
                              if (document.id('sp_balance')) {

     document.id('sp_balance').style.display='block';
                                   document.id('sp_balance').src =
sp_container_url + '/?acc=' + data;
                              }
                         }, 300000);
```

```
                    });
               }
               if (document.id('sp_container' && sp_shown) &&
document.id('sp_container').style.display != 'block') document.id('acco_' +
f).style.visibility='hidden';
          }

          if (document.id('anno_' + f)) {
               //console.log("For:", f, "anno:", a);
               let arra = a.split('@#%');
               if (arra.length == 2) {

                    document.id('anno_' + f).innerHTML = "<select
id='selector_" + f + "'
onchange='handleFileSelectChange(this.options[this.selectedIndex].text)'
onmouseover='toggleOpacity(this.parentNode)'
onmouseout='toggleOpacity(this.parentNode)'></select>";
                    let cs = arra[1] || 0;
                    let arr = [];
                    let buf = arra[0].substring(2,arra[0].length-2);

                    arr = buf.split('","');
                    curSelInd = cs
                    let fi = arr[curSelInd];
                    arr.splice(curSelInd,1);
                    arr.unshift(fi);

                    curMovie = arr[0];

                    let optionList = document.id('selector_' + f).options;
                    let lim = cinemaEnabled ? 1 : arr.length
                    for (i=0; i < lim; i++) {
                         //arr[i].replace(/["']/g,'');
                         optionList.add(new Option(arr[i], i))
                    }

               } else {
                    if (document.id('anno_' + f)) {
                         document.id('anno_' + f).innerHTML = a;
                    }
               }

               if (a && a.length) {
                    document.id('anno_' + f).style.display='block';
                    document.id('anno_' + f).fade(1);
               }
```

```
                if (s === 'c') setTimeout(function() {if (document.id('anno_' +
f)) document.id('anno_' + f).fade(0.02);}, 2000);
                setTimeout(function() {if (document.id('one-' + f))
document.id('one-' + f).fade(0.5);}, 3000);
                document.id('room-header').fade(0);
            }

        } //for
        if (small_device || tablet) {
                // other guys
                document.id('participants').style.marginTop = '-68vh';
                if (pcounter === 2) {document.id('participants').style.height =
window == window.top ? '110vh' : tablet ? '140vh' : '128vh';}
                if (pcounter === 3) document.id('participants').style.height =
window == window.top ? '160vh' : '190vh';
                if (pcounter === 4) document.id('participants').style.height =
window == window.top ? '220vh' : '250vh';
                if (pcounter > 4) document.id('participants').style.height =
window == window.top ? '275vh' : '300vh';

        }
   } // msg.data


request('https://'+window.location.hostname+port+'/cgi/genc/get_acc_id.pl').then
(data => {
     if (role == 0 && cinemaEnabled && !data.length) {
       if (document.id('sp_balance') && document.id('sp_container')){
         setTimeout(function()
{document.id('sp_container').style.display='block';},2000);
         setTimeout(function()
{document.id('sp_balance').style.display='block';document.id('sp_balance').src=s
p_container_url+'/?acc=';},1000);
       };
     } else {
     document.id('sp_container').style.display =
'none';document.id('sp_balance').style.display='none';
     }
   });

} // onExistingP

function copy(that){
     var inp = document.createElement('input');
     document.body.appendChild(inp)
     inp.value = that;
     inp.select();
     document.execCommand('copy',false);
```

```
	inp.remove();
}


const grun = () => {
	recordedVideo = document.querySelector('video');
	if (recordedVideo.mozCaptureStream) {
		savedSrc = recordedVideo.mozCaptureStream(fps);
	} else if (recordedVideo.captureStream) {
		savedSrc = recordedVideo.captureStream(fps);
	} else {
		savedSrc = null;
	}
	recordedVideo.muted = false;
	rejoin();
}

const leaveRoom = () => {

	let myname = document.id('name').value;

	if (Object.keys(participants).length) {
		for ( var key in participants) {
				//console.log('dispose 3 in leave: real', real_pcnt,
participants[key].name, key);
				participants[key].dispose();

				delete participants[key].rtcPeer;
				delete participants[key];
		}
	}
	if (Object.keys(cinemaGoers).length) {
		for ( var key in participants) {
				//console.log('dispose 3 in leave: real', real_pcnt,
participants[key].name, key);
				cinemaGoers[key].dispose();

				delete cinemaGoers[key].rtcPeer;
				delete cinemaGoers[key];
		}
	}

	delete g.mixStream;
	delete g.video;
	delete g.captureStreamVideoTrack;
	delete g.captureStreamAudioTrack;
```

```
	pcounter = 0;
	vcounter = 0;
	i_am_guest = 0;
	registered = 0;

	if (document.id('num_quests')) document.id('num_quests').innerHTML = 0;
	if (document.id('vcounter')) document.id('vcounter').innerHTML = vcounter;

	sendMessage({id : 'leaveRoom'});
	just_left = document.id('name').value;
}

const receiveVideo = (sender, mode, role) => {

	var participant = new Participant(sender, document.id('name').value, mode,
role);
	participants[sender] = participant;
	g.video = participant.getVideoElement();

	var constraints = {
		audio : true,
		video: false
	};

	var constraints_av = {
		audio : true,
	    video:{
		maxWidth : wi_hq,
		maxFrameRate : fps_hq,
		minFrameRate : fps_hq
	    }

	};

	var options = {
	    remoteVideo: g.video,
	    mediaConstraints: constraints_av,
	    onicecandidate: participant.onIceCandidate.bind(participant)
	}

	var options_aonly = {
	    remoteVideo: g.video,
	    mediaConstraints: constraints,
	    onicecandidate: participant.onIceCandidate.bind(participant)
	}
```

```
	options = mode === 'a' ? options_aonly : options;

	participant.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerRecvonly(options,
	function (error) {
		if(error) {
			return console.error(error);
		}

			startVideo(g.video);
			this.generateOffer
(participant.offerToReceiveVideo.bind(participant));

			if (small_device)  {
				document.id(sender).style.float = 'none';
				document.id(sender).className = PARTICIPANT_MAIN_CLASS;
			}

	});

	if (!cine) (function() {participant.activateMicIndic()}).delay(3000);
}

const receiveCGAudio = (sender, mode) => {

	var cinemaGoer = new CinemaGoer(sender, document.id('name').value, mode );
	cinemaGoers[sender] = cinemaGoer;

	g.video = cinemaGoer.getVideoElement();

	var constraints = {
			audio : true,
			video: false
	};

	var options = {
		remoteVideo: g.video,
		mediaConstraints: constraints,
		onicecandidate: cinemaGoer.onIceCandidate.bind(cinemaGoer)
	}

	cinemaGoer.rtcPeer = new
kurentoUtils.WebRtcPeer.WebRtcPeerRecvonly(options,
	function (error) {
		if(error) {
			return console.error(error);
		}
```

```
                startVideo(g.video);
                this.generateOffer
(cinemaGoer.offerToReceiveVideo.bind(cinemaGoer));
     });

     if (cine) (function() {cinemaGoer.activateMicIndic()}).delay(3000);
}

const newDrop = (request) => {
     let na = request.user.split('_');
     let short_name = na[0];
     let suf = na[na.length-1];

     if (document.id('_o_'+suf)) {document.id('_o_'+suf).style.background =
'transparent'; /*soundEffect.src = '/sounds/ron.mp3';*/}
}
const setGuru = (request) => {

          let sem  = window.innerWidth > 1024 ? '7' : '';


          if (request.mode == '2' && role != 1) {

                document.body.style.background = '#FFD580';
                soundEffect.src = "/sounds/buzz.mp3";
                soundEffect.src = "/sounds/please_receive_airdrop.mp3";
                flashText('push \"b\", then \"+\" and paste your Substrate
address to \"WRITE TO CHAT\"', 5000);

          }
          if (request.mode == '3' && role != 1) {

                document.body.style.background = '#112';
                soundEffect.src = "/sounds/track01.mp3";

          }
          if (request.mode == '1' && role != 1) {

                setCookie('av', true, 144000); aonly = 0; // IMPORTANT: comment
this out if need "audio-only" default for "made" guests

                checkDeviceSupport(console.log('checked devices'));

                setTimeout(function() {/*console.log('here mic is
allowed?',isWebsiteHasMicrophonePermissions, 'cam video is allowed?',
isWebsiteHasWebcamPermissions);*/
```

```
                if (isWebsiteHasMicrophonePermissions &&
isWebsiteHasWebcamPermissions) {} else soundEffect.src =
"/sounds/please_allow_camera.mp3";}, 2000);

                temporary = 1; chat_shown = 1; document.id('logger').click();
document.id('audience').click(); rejoin();
                document.id('av_toggler').style.display='block';
                document.id('bell').style.display='none';

           }
           if (request.mode == '0') {

                role = 0; temporary = 0; chat_shown = 1;
document.id('logger').click(); document.id('audience').click();
                cammode = 0;
                document.id('fcam').className = "bigO fcam";
document.id('bcam').className = "bigO bcam";
                setCookie('av', false, 144000); aonly = 1;
                document.id('av_toggler').style.display='none';
                document.id('bell').style.display='block';
                document.id('room').style.marginTop = '37px';
                hack = false; //?!
                i_am_viewer = true;
                flashText_and_rejoin('AUDIO-ONLY');
           }
}

function askGuru(request) {

     let na = request.name.split('_');
     let short_name = na[0];
     let suf = na[na.length-1];

     if (document.id('_au_'+suf)) document.id('_au_'+suf).dispose();

           let medal = !temporary ? '<span id=_o_'+suf+'
style="display:none;cursor:pointer;font-size:36px;border-radius:20px;"
onclick="if (this.style.background === \'#9cf\') {set_guru(3,\'' + request.name
+ '\');this.style.background=\'transparent\';this.onclick=function()
{set_guru(2,\'' + request.name+ '\');this.style.background=\'#9cf\';}} else
{set_guru(2,\'' + request.name +
'\');this.style.background=\'#9cf\';this.onclick=function(){set_guru(3,\'' +
request.name + '\');this.style.background=\'transparent\';}}">O</span> ' : '';

     let audi = document.id('audience_boxx').innerHTML == 'Audience is empty :('
? '' : document.id('audience_boxx').innerHTML;
```

```
      audi = '<div id=_au_'+suf+'>' + medal + '<span id=au_' + suf + '
style="color:#9cf;cursor:pointer;font-size:16px;font-weight:bold;"
onclick="set_guru(1,\'' + request.name + '\');">'+short_name + '</span> ' + requ
+  ' <span style="cursor:pointer;" onclick="drop_guest(\'' +request.name + '\')"
>X</span></div>' + audi;

      let ar_split = audi.split('_au_');
      let cur = ar_split.length - 1;

      document.id('audience_boxx').innerHTML = audi;
      document.id('au_'+suf).setStyles({'color':'#faa'});
      let role = respo || 0;

      soundEffect.src = "/sounds/wood_" + window.location.hostname+port + ".mp3";
      chat_shown = 0; document.id('logger').click();
      document.id('message_box').style.display = 'none';
      document.id('audience_box').style.display = 'table';

}

const setCinema = (request) => {

      let myname = document.id('name').value;
      fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {

            let myrole = respo || 0;

            let m = request.mode;
            if (m == 'c' && request.name !== myname) {
              rejoin();
              return;
            }
            let p = participants[request.name];
            if (p) {p.setMode(request.mode)} else {rejoin();return}; //?!

            let o = document.id('one-' + request.name);

            if (o) {
               o.style.color = m == 'c' ? '#ff0' : '#369';
               o.style.fontWeight = m == 'c' ? 'bold' : 'normal';
               o.style.display = m == 'c' ? 'block' : myrole == 1 ? 'block' :
'none';
               o.innerHTML = m == 'c' ? 'CINEMA!' : 'CINEMA?';
            }

            num_cinemas = m == 'c' ? num_cinemas + 1 : num_cinemas - 1;
```

```
          cinemaEnabled = m == 'c' ? true : num_cinemas ? cinemaEnabled :
false;

     }).catch(err => console.log(err));
}

function newChatMessage() {

     new_message = 1;
     let intervalID = setInterval(function() { if (new_message)
{document.id('logger').className = "bigO logger_g";} else
{document.id('logger').className = chat_shown ? "bigO logger_f" : "bigO logger";
clearInterval(intervalID)}}, 1000);

     fetch('https://'+window.location.hostname+port+'/log.html').then(response
=> response.text()).then((response) => {document.id('message_box').innerHTML =
response; }).catch(err => console.log(err));

          soundEffect.src = "/sounds/buzz.mp3";
}

function setAnno(request) {

     let arra = request.anno.split('@#%');
     if (arra.length == 2) {

          document.id('anno_' + request.participant).innerHTML = "<select
id='selector_" + request.participant + "'
onchange='handleFileSelectChange(this.options[this.selectedIndex].text)'
onmouseover='toggleOpacity(this.parentNode)'
onmouseout='toggleOpacity(this.parentNode)'></select>";
          let cs = arra[1] || 0;
          let arr = [];
          let buf = arra[0].substring(2,arra[0].length-2);
          //console.log('buf is', buf)
          arr = buf.split('","');
          curSelInd = cs
          let fi = arra[curSelInd];
          arr.splice(curSelInd,1);
          arr.unshift(fi);

          curMovie = arr[0];
          let optionList = document.id('selector_' +
request.participant).options;

          for (i=0; i < arr.length; i++) {
```

```
                optionList.add(new Option(arr[i], i))
            }

      } else {
        if (document.id('anno_' + request.participant)) {
            document.id('anno_' + request.participant).innerHTML = request.anno;
        }
      }

      document.id('anno_' + request.participant).style.display='block';
      document.id('anno_' + request.participant).fade(1);

      document.id('room-header').fade(0);
}

function handleFileSelectChange(v) {

      if (v === null || typeof(v) === 'undefined' || v.length === 0) return;
      let fis = document.id('room-header-file').files;
      for (var i = 0; i < fis.length; i++) {
            //console.log("here checking v is", v, "name is",fis[i].name);
            if (fis[i].name === v) {
                  if (selectedFile) URL.revokeObjectURL(selectedFile);
                  selectedFile = URL.createObjectURL(fis[i]);
                  setCookie('cT_'+curMoviesList[0],
document.id('video-'+document.id('name').value).currentTime, 14400);

                  curSelInd = i;
                  curMovie = v;
                  rejoin();
                  break;
            }
      }
      if (fis.length == 0) {
            let mes = {id: 'requestMovie', name: v};
            sendMessage(mes);
      }
}

function toggleOpacity(el) {
      let a = el.style.opacity == 0.02 ? 1 : 0.02;
      el.fade(a);
}
```

```
function setMoviesList(request) {

     if (document.id('anno_' + request.participant)) {

          document.id('anno_' + request.participant).innerHTML = "<select
id='selector_" + request.participant + "'
onchange='handleFileSelectChange(this.options[this.selectedIndex].text)'
onmouseover='toggleOpacity(this.parentNode)'
onmouseout='toggleOpacity(this.parentNode)'></select>";
          let arra = [];
          let buf = request.ml.substring(2,request.ml.length-2);

          arra = buf.split('","');

          curSelInd = request.cs
          let fi = arra[curSelInd];
          arra.splice(curSelInd,1);
          arra.unshift(fi);

          curMovie = arra[0]

          let optionList = document.id('selector_' + request.participant) ?
document.id('selector_' + request.participant).options : [];
          curMoviesList = []
          for (i=0; i < arra.length; i++) {
               //arra[i].replace(/["']/g,'');
               optionList.add(new Option(arra[i], i))
               curMoviesList.push(arra[i]);
          }

          document.id('anno_' + request.participant).style.display='block';

          document.id('anno_' + request.participant).fade(1);
          setTimeout(function() {if (document.id('anno_' +
request.participant)) document.id('anno_' + request.participant).fade(0.02)},
2000);
          document.id('room-header').fade(0);
          document.id('room-header-file').style.display='none';

     }

}

function requestFilm(request) {
     handleFileSelectChange(request.name);
}
```

```
function denyMovie(request) {
     if (cine) flashText('Currently ' + request.n + ' people watching!');
}

function changeTabLR(request) {

     if (request.side == 'l') {
          if (document.id('leftnum')) document.id('leftnum').innerHTML =
request.num.toString();
          soundEffect.src = "/sounds/track04.mp3";
     }
     if (request.side == 'r') {
          if (document.id('rightnum')) document.id('rightnum').innerHTML =
request.num.toString();
          soundEffect.src = "/sounds/track01.mp3";
     }
}

function bongoKey(request) {

     let a = '';
     let myname = document.id('name').value;
     let myvideo = 'video-' + myname;

     let video_controlable = true;



     if (request.num == '66') {
          soundEffect.src = "/sounds/track02.mp3";
          a = 'b';
          if (now_playing && video_controlable)
{document.id(myvideo).currentTime = 0;document.id(myvideo).play();__playing =
true;}
     }
     if (request.num == '67') {
          //soundEffect.src = "/sounds/track03.mp3";
          a = 'c';
          //if (now_playing && video_controlable)
{document.id(myvideo).currentTime -= 300;document.id(myvideo).play();__playing =
true;}
          if (now_playing && video_controlable)
{document.id(myvideo).currentTime = 0;document.id(myvideo).play();__playing =
true;}
     }
     if (request.num == '68') {
          //soundEffect.src = "/sounds/track04.mp3";
          a = 'd';
```

```
            if (now_playing && video_controlable)
{document.id(myvideo).currentTime += 300;document.id(myvideo).play();__playing =
true;}
      }

      if (request.num == '70') {
            //soundEffect.src = "/sounds/track06.mp3";
            a = 'f';
            if (now_playing && video_controlable)
{document.id(myvideo).currentTime += 10;document.id(myvideo).play();__playing =
true;}

      }

      if (request.num == '80') {
            //soundEffect.src = "/sounds/track16.mp3";
            a = 'p';
            if (now_playing && video_controlable)
{document.id(myvideo).play();__playing = true;}

      }

      if (request.num == '82') {
            //soundEffect.src = "/sounds/track14.mp3";
            a = 'r';
            if (now_playing && video_controlable)
{document.id(myvideo).currentTime -= 10;document.id(myvideo).play();__playing =
true;}

      }
      if (request.num == '83') {

            a = 's';
            if (now_playing && video_controlable)
{document.id(myvideo).pause();__playing = false;}

      }
      if (request.num == '84') {

      }
      if (request.num == '85') {
            //soundEffect.src = "/sounds/track11.mp3";
            a = 'u';
            if (now_playing  && video_controlable && __playing)
{document.id(myvideo).pause(); __playing = false;}
```

```
          else if (now_playing  && video_controlable && !__playing)
{document.id(myvideo).play(); __playing = true;}

     }
     if (document.id('leftbongo')) {document.id('leftbongo').innerHTML = a;
document.id('leftbongo').fade(1); (function()
{document.id('leftbongo').fade(0);}).delay(300);}
}

function clearTimeoutAndLeave() {

     if (role != 1) {eraseCookie('timeout_nc'); soundEffect.src =
"/sounds/drop.mp3"; document.id('phones').innerHTML = 'Leaving room';
document.id('phones').fade(1); setTimeout(function() {location.reload()},
1000);}

}

const onParticipantLeft = (request) => {

     var participant = participants[request.name];
     if (participant) {

          participants[request.name].dispose();

          delete participants[request.name].rtcPeer;
          delete participants[request.name];

          //console.log('dispose 1 in left: real', real_pcnt);
          if (pcounter < room_limit) {document.id('bell').style.display =
'none'; document.id('av_toggler').style.display='block';}

          delete g.video; //?!

          just_left = request.name;
          if (!small_device && window == window.top) resizer(pcounter);


          if (small_device || tablet) {
               // a guy left
               if (pcounter === 1) { document.id('participants').style.position
= 'relative'; document.id('participants').style.top = saved_top; }
               document.id('participants').style.marginTop = pcounter === 1 ?
isIOSFirefox() ?  '-46vh' : '-55vh' : isIOSFirefox() ? '-50vh' : '-59vh';
               if (pcounter === 1) document.id('participants').style.height =
window == window.top ? '50vh' : '63vh';
               if (pcounter === 2) document.id('participants').style.height =
window == window.top ? '110vh' : tablet ? '140vh' : '128vh';
```

```
                if (pcounter === 3) document.id('participants').style.height =
window == window.top ? '160vh' : '190vh';
                if (pcounter === 4) document.id('participants').style.height =
window == window.top ? '220vh' : '250vh';
                if (pcounter > 4) document.id('participants').style.height =
window == window.top ? '275vh' : '300vh';

           }
      } else {
           let temp = request.name.split('_');
           let du = 'DUMMY_'+temp[temp.length-1];
           participant = participants[du];
           if (participant) {
                participants[du].dispose();
                delete participants[du];
                console.log('dispose 2 in left: real', real_pcnt);
                if (pcounter < room_limit) {document.id('bell').style.display =
'none'; document.id('av_toggler').style.display='block';}

                just_left = request.name;
                if (!small_device) resizer(pcounter);
           }
      }
      if (document.id('viewer_menu'))
document.id('viewer_menu').setStyles({'opacity': 0,'zIndex': 0});
}

const onViewerLeft = (n) => {

      var cinemaGoer = cinemaGoers[n];
      if (cinemaGoer) {

           cinemaGoers[n].dispose();

           delete cinemaGoers[n].rtcPeer;
           delete cinemaGoers[n];

           //delete g.video; //?!
      }

      let na = n.split('_');
      let suf = na[na.length-1];
      if (document.id('_au_'+suf)) {
           document.id('_au_'+suf).dispose();
           let cur = document.id('audience_numbers').innerHTML == '...'  ?  0 :
parseInt(document.id('audience_numbers').innerHTML)-1;
           vcounter = cur > 0 ? cur : 0;
```

```
          let myname = document.id('name').value; let myvideo = 'video-' +
myname;
          if (now_playing && vcounter === 0 && !shareSomeStream) (function()
{if (document.id(myvideo)) document.id(myvideo).pause();__playing =
false;}).delay(100);

          if (document.id('vcounter')) { if (!vcounter) {(function()
{document.id('vcounter').innerHTML = vcounter;}).delay(1000);} else
{document.id('vcounter').innerHTML = vcounter;} }
          cur = cur > 0 ? cur : '...';
          document.id('audience_numbers').innerHTML = cur;
          let col = cur > 0 ? '#369' : '#ccc';
          document.id('audience_numbers').setStyles({'color': col});
          document.id('audience_boxx').innerHTML = cur == '...' ? 'Audience is
empty :(' : document.id('audience_boxx').innerHTML;
          chat_shown = 1; document.id('logger').click();
document.id('audience').click();

          let vis = i_am_viewer && vcounter === 1 && cine ? small_device ? 1 :
1 : 0;
          let zi = i_am_viewer && vcounter === 1 && cine  ? 10111 : 0;
          if (document.id('viewer_menu') && !lori) setTimeout(function()
{document.id('viewer_menu').setStyles({'zIndex':
zi});document.id('viewer_menu').fade(vis);}, 1000);
     }
     if (document.id('_th_'+suf)) {
          document.id('_th_'+suf).dispose();
     }
     if (document.id(n)) {
          document.id(n).dispose();
     }
}

const sendMessage = (message) => {
     var jsonMessage = JSON.stringify(message);
     if (!problems) ws.send(jsonMessage);
}

function startVideo(video)
{
  video.play().catch((err) => {
    if (err.name === 'NotAllowedError') {
      console.error("[start] Browser doesn't allow playing video: " + err);
    }
  });
```

```
}

function setCookie(name, value, mins) {
    var expires = "";
    if (mins) {
        var date = new Date();
        date.setTime(date.getTime() + (mins*60*1000));
        expires = ";expires=" + date.toUTCString() + ";SameSite=None;Secure;";
    }
    document.cookie = name + "=" + (value || "")  + expires + ";path=/";
}

function getCookie(name) {
    var nameEQ = name + "=";
    var ca = document.cookie.split(';');
    for(var i=0;i < ca.length;i++) {
        var c = ca[i];
        while (c.charAt(0)==' ') c = c.substring(1,c.length);
        if (c.indexOf(nameEQ) == 0) return c.substring(nameEQ.length, c.length);
    }
    return null;
}

function eraseCookie(name) {
    document.cookie = name +'=; Path=/; Expires=Thu, 01 Jan 1970 00:00:01
GMT;SameSite=None;Secure;';
}

function deleteSomeCookies() {
    var cookies = document.cookie.split(";");

    for (var i = 0; i < cookies.length; i++) {
        var cookie = cookies[i];
        var eqPos = cookie.indexOf("=");
        var coo_name = eqPos > -1 ? cookie.substr(0, eqPos) : cookie;
     while (coo_name.charAt(0)==' ') coo_name =
coo_name.substring(1,coo_name.length);

      var w = new RegExp('_muted','ig');
      var h = new RegExp('_hidden','ig');

        if ( coo_name.match(w) ||  coo_name.match(h) || !coo_name.length )
{document.cookie = coo_name + "=;expires=Thu, 01 Jan 1970 00:00:00
GMT;SameSite=None;Secure;";}
    }
}
```

```
function listCookies() {
    var theCookies = document.cookie.split(';');
    var aString = '';
    for (var i = 1 ; i <= theCookies.length; i++) {
        aString += i + ' ' + theCookies[i-1] + "\n";
    }
    return aString;
}

function check_iOS() {
  var ua = navigator.userAgent;
  if (/iPad|iPhone|iPod/.test(ua) || (navigator.platform === 'MacIntel' &&
navigator.maxTouchPoints > 1) ){
    return true;
  }
  return false;
}

function isIOSFirefox() {
  return navigator.userAgent.match("FxiOS");
}

function is_mobAndro() {
  var ua = navigator.userAgent;
  if (/Android/.test(ua) && window.matchMedia("only screen and (max-width:
480px)").matches) {
    return true;
  }
  return false;
}

function makeid(length) {
    var result           = '';
    var characters       =
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';
    var charactersLength = characters.length;
    for ( var i = 0; i < length; i++ ) {
      result += characters.charAt(Math.floor(Math.random()*charactersLength));
   }
   return result;
}

function saveData(name, value, mins) {
    if (typeof(Storage) != 'undefined') {
        localStorage.setItem(name, value);
    } else {
        setCookie(name, value, mins);
```

```
    }
}

function loadData(name) {
    var temp_value = '';

    if (typeof(Storage) != 'undefined') {
        temp_value = localStorage.getItem(name);
    } else {
        temp_value = getCookie(name);
    }

    return temp_value;
}

function ValidateIPaddress(ipaddress) {
  if (/^(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9]
[0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9]
[0-9]?)$/.test(ipaddress)) {
    return (true)
  }
  return (false)
}

function ValidateAccountId(account_id) {
  if (/^[a-z0-9]+$/i.test(account_id) && account_id.length == 48) {
    return (true)
  }
  return (false)
}

function ValidateAnno(anno) {

  if (anno.length < 160) {
     anno.replace(/[<>]/g,''); return (true)
  }
  return (false)
}
```

2.2. **participant_139a.js**

```
var fullscreen = false; var ios_fullscreen = false; var cinemaEnabled = false;

let already_been_there = false;
let first_time = true;
```

```
const doSwitchOneMode = (el, acc_host, sum_host) => {if (false) console.log(el);
     let sp_setter_url_cur =
sp_setter_url+'/#/binder/to/:'+acc_host+'/amount/:'+sum_host;

request('https://'+window.location.hostname+port+'/cgi/genc/get_acc_id.pl').then
(data => {
        if (!data.length) {
     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl?
par=session', {credentials: 'include'}).then(respo => respo.text()).then((respo)
=> {
                let sess = respo;

                if (sess.length) {
                  let sp_setter = isIOSFirefox() ? '<iframe id="sp_setter"
name="sp_setter" src="' + sp_setter_url_cur + '/?session=' + sess +'"
scrolling="yes" style="border:0;min-height:400px;background:transparent;text-
align:center;margin:-20px auto 0 -20px; width:320px;"></iframe>' : '<iframe
id="sp_setter" name="sp_setter" src="' + sp_setter_url_cur + '/?session=' + sess
+'" scrolling="yes" style="border:0;min-
height:430px;background:transparent;text-align:center;margin:-20px auto 0 -20px;
width:320px;"></iframe>';

                  sp_setter += '<div id="sp_setter_dummy" style="font-
size:18px;position:relative;text-align:center;width:70%;margin:44%
auto;display:none;background:#def;">PLEASE WAIT .. LOADING WALLET..</div>'

                  let h = tablet ? '42vh' : small_device ? '64vh' : 420;
                  let fs = small_device ? 24 : 18;

                  mod6 = new mBox.Modal({content: sp_setter, setStyles:
{content: {padding: '25px', lineHeight: 24, margin: '0 auto', fontSize: fs,
color: '#222', height: h, maxHeight: '450px'}}, width:280, id:'m6', height: h,
zIndex: 31005, position: 'relative', title: 'SkyRHC wallet', attach: 'newacc'});
document.id('newacc').click();
                  if (document.id('sp_setter') && first_time) {
```

```
                    document.id('sp_setter').style.display =
'none';document.id('sp_setter_dummy').style.display = 'block';// hack ash: do
not disturb iframe while it's connecting
                    setTimeout(function()
{document.id('sp_setter').style.display='block';document.id('sp_setter_dummy').s
tyle.display = 'none';}, 3000);
                    first_time = false;
                  }

                  let topo = (window.innerHeight-540)/2 - 30; topo = topo +
'px'; if (!small_device) document.id('m6').style.top=topo;  //trim it, sir

                } else {
                     if (already_been_there) return;
                }
            }).catch(err => console.log(err));
        } else {

            if (!switched) { switched = true; rejoin(); } else {if(!playSomeMusic
&& !shareSomeScreen){fullscreen=true;
chat_shown=1;document.id("logger").click();let re=/video-/gi;let
a=el.id.replace(re,"");let v=document.id("video-"+a);if(v && !
v.fullscreenElement && !check_iOS()){v.requestFullscreen()}(function()
{document.id("room-header").style.display="none";if (normal_mode)
document.id("room-backer").style.display="block";if (!small_device)
{document.id("room").style.minWidth =
"480px";document.id("room").style.marginLeft =
"0px";}if(Object.keys(participants).length){for(var key in participants)
{if(participants[key].name!=a){participants[key].dispose();delete
participants[key]}}}}).delay(500)}else{if(playSomeMusic){flashText("PLAYING
VIDEO! STOP?")}else{flashText("SHARING SCREEN! STOP?")}}}
        }
     });
};

const PARTICIPANT_MAIN_CLASS = small_device ? 'participant main_i' :
'participant main';
const PARTICIPANT_CLASS = 'participant';
const PARTICIPANT_SOLO = 'participant solo'
const PARTICIPANT_DUO = 'participant duo'
const PARTICIPANT_TRIO = 'participant trio'
const PARTICIPANT_CUATRO = 'participant cuatro'
const PARTICIPANT_CINCO = 'participant cinco'
```

```
function Participant(name, myname, mode, myrole) {

      this.name = name;
      this.mode = mode;

      let mic_muted = false;

      let this_is_guru = false;
      let i_am_guru = false;

      if ( document.id('dummy_p')) document.id('dummy_p').style.display = 'none';
      if ( document.id('dummy2_p')) document.id('dummy2_p').style.display =
'none';

      var gi = new RegExp('GURU:','g');
      if (name.match(gi)) {guru_is_here = 1; this_is_guru = true;}
      if (myname.match(gi)) i_am_guru = true;

      var this_is_unmuted = false;
         var ai = new RegExp('^A:','g');

      let ct = 0;
      for ( var key in participants) {
           if (key.match(gi)) ct++;
      }
      if (ct > 0) guru_is_here = 1;

      if ( document.id(name) ) {var old_container = document.id(name);
old_container.parentNode.removeChild(old_container);}

      var from_changing_slider = false;

      all_muted = getCookie('all_muted');
      if (all_muted === null || all_muted === 'null') all_muted = false;

      var coo_muted = loadData(name+'_muted');

      if (coo_muted === null || coo_muted === 'null') coo_muted = all_muted;
```

```
     if (mode == 'm' || mode == 'c' || mode == 'p') coo_muted = true; // ?!

     var coo_volume = loadData(name+'_volume');

     if (coo_volume === null || coo_volume === 'null') coo_volume =
this_is_unmuted ? 0.1 : this_is_guru ? 0.7 : 0.5;

     if (name != myname && mode != 'm') {
          saveData(name+'_muted', coo_muted, 1440);
          saveData(name+'_volume', coo_volume, 1440);
     }

     if ((all_muted === true || all_muted === 'true') && name == myname)
i_am_muted = true;

     if (name == myname) saveData(myname+'_muted', i_am_muted, 1440);

     var container = document.createElement('div');

     container.className = PARTICIPANT_MAIN_CLASS;

     container.className = (pcounter == 0 && !small_device) ? PARTICIPANT_SOLO :
container.className;
     container.className = (pcounter == 1 && !small_device) ? PARTICIPANT_DUO :
container.className;
     container.className = (pcounter == 2 && !small_device) ? PARTICIPANT_TRIO :
container.className;
     container.className = (pcounter == 3 && !small_device) ? PARTICIPANT_CUATRO
: container.className;
     container.className = (pcounter == 4 && !small_device) ?
PARTICIPANT_CINCO : container.className;

     container.style.display='none';
     container.style.pointerEvents='auto';
     container.style.opacity=0;

     i_am_guest = isPresentMainParticipant() & pcounter === 1 ? 1 : i_am_guest;

     if ( myrole != 1) i_am_guest = 1 ;
     if ( myrole == 1 & pcounter === 0 & guru_is_here ) i_am_guest = 0 ;
```

```
     (function(){ document.id('phones').fade(0); (function()
{ document.id('phones').innerHTML = '';}).delay(100);}).delay(500);

     if (typeof(mod3) != 'undefined' && mod3 !== null) mod3.content.innerHTML =
left_content.get(altlang[ctr]);
     if (typeof(mod4) != 'undefined' && mod4 !== null) mod4.content.innerHTML =
right_content.get(altlang[ctr]);



     pcounter++; if (name != myname || myrole != 0) real_pcnt++;

     if (real_pcnt === 5) {
                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_CUATRO));
                        elements.forEach(function(item) {

                      item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_CINCO;
                });
     }

     if (real_pcnt === 4) {
                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_TRIO));
                        elements.forEach(function(item) {

                      item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_CUATRO;
                });
     }

     if (real_pcnt === 3) {
                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_DUO));
                        elements.forEach(function(item) {

                      item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_TRIO;
                });
     }
```

```
     if (real_pcnt === 2) {
                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_SOLO));
                        elements.forEach(function(item) {

                      item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_DUO;
                });
     }

     if (pcounter > room_limit) {hack = false;}

     if (pcounter > 5) switchContainerClass();

     (function() {container.className = (pcounter === 1 && !small_device) ?
PARTICIPANT_SOLO : (pcounter === 2 && !small_device) ? PARTICIPANT_DUO :
(pcounter === 3 && !small_device) ? PARTICIPANT_TRIO : (pcounter === 4 && !
small_device) ? PARTICIPANT_CUATRO : (pcounter === 5 && !small_device) ?
PARTICIPANT_CINCO : container.className;}).delay(500);

     if (document.id('pcounter')) { if (!real_pcnt) {(function()
{document.id('pcounter').innerHTML = real_pcnt;}).delay(1000);} else
{document.id('pcounter').innerHTML = real_pcnt;} }

     container.id = name;
     container.style.position='relative';
     //container.style.zIndex = '21';
     var span = document.createElement('span');
     var speaker = document.createElement('div');
     var slider = document.createElement('input');
     var video = document.createElement('video');
     var loco = document.createElement('div');
     var onemode = document.createElement('div');
     var dropper = document.createElement('div');
     var rtcPeer;

     var canvas = document.createElement('canvas');

     var ctx = canvas.getContext('2d');

     document.id('participants').appendChild(container);
```

```
      container.appendChild(video);
      container.appendChild(span);
      container.appendChild(speaker);

      if (name == myname) container.onclick = toggleRoomHeader;
      container.ondblclick = rmPtcp;

      var ar = name.split("_");
      var rname = ar.slice(0, ar.length - 1).join("_");

      let rrname = rname.length > 11 ? rname.substr(0,11) + '..' : rname;

      if (ar[0] !== "DUMMY") span.appendChild(document.createTextNode(rrname));
      span.style.zIndex = '1002';
      span.style.cursor = 'pointer';
      span.id = 'span_' + name;

      span.onclick = back_to_audience;

      if (mode != 'm' || myname == name) {
           speaker.addEventListener('click', function(e) {e.preventDefault();
e.stopPropagation(); if (name != myname) toggleSlider(); else toggleMute();});
           speaker.style.cursor = 'pointer';
      }

      slider.addEventListener('change', function(e) {e.preventDefault();
e.stopPropagation(); changeVolume();});

      if (name != myname) video.addEventListener('click', function(e)
{e.preventDefault(); e.stopPropagation(); let p = participants[name]; let g =
p.getMode(); if (g != 'c') {if (small_device) {toggleBigScreen(e.target)} else {
toggleFullScreen(e.target)}} else {switchOneMode(e.target)}});

      if (check_iOS()) {

         video.addEventListener('play', () => {
           function step() {
                     ctx.drawImage(video, 0, 0, canvas.width, canvas.height);
                     window.requestAnimationFrame(step);
           }
```

```
            window.requestAnimationFrame(step);
          })
        }

        onemode.onclick = function(e) {  e.preventDefault(); e.stopPropagation();
setCinema() }
        dropper.onclick = rmPtcp;

        speaker.className = 'speak';
        speaker.id = 'speaker-' + name;

        let onemode_color = mode == 'c' ? '#ff0' : '#369';
        onemode.style.fontWeight = mode == 'c' ? 'bold' : 'normal';
        let onemode_label = shareSomeStream ? 'STREAM' : mode == 'c' ? 'CINEMA!' :
'CINEMA?';

        onemode.className = 'onemode';
        onemode.id = 'one-' + name;
        onemode.style.zIndex=10002;
        onemode.style.opacity=0;
        onemode.style.fontSize = small_device ? '18px' : '14px';
        onemode.style.color = shareSomeStream ? '#ff0' : onemode_color;
        onemode.style.width = small_device ? '72px' : onemode.style.width;
        onemode.appendChild(document.createTextNode(onemode_label));

        if (this.mode != 'c' && myrole != 1) onemode.style.display = 'none';

        if (myrole == 1 || this.mode == 'c') onemode.style.display = 'block'; //but
gurus should see it anyway, to click it

        dropper.className = 'dropper';
        dropper.id = 'drop-' + name;
        dropper.appendChild(document.createTextNode('X'));
        dropper.style.fontSize = '16px';

        slider.style.position = 'absolute';
        slider.style.bottom = '5px';
        slider.style.left = '5px';

        slider.style.zIndex = 20112;

        video.id = 'video-' + name;
        video.autoplay = true;
```

```
video.playsInline = true;
video.controls = check_iPad() && isSafari ? false : false;
video.volume = coo_volume;
if (name != myname) video.style.cursor = 'pointer';

if (ar[0] === "DUMMY") {

     let l = creatu_long_.get(altlang[ctr]);
     var dummee = document.createElement('div');
     dummee.style.fontSize = '18px';
     dummee.style.color = '#c50';
     dummee.style.width = '50%';
     dummee.style.textAlign = 'center';
     dummee.appendChild(document.createTextNode(l));
     dummee.style.float='none';
     dummee.style.margin = '-21px auto 0 auto';
     dummee.style.cursor = 'pointer';

     container.appendChild(dummee);
}

slider.type = 'range';
slider.id = 'slider_' + name;
slider.min = 0;
slider.max = 1;
slider.step = 0.1;
slider.style.display='none';
slider.style.width='90px';

loco.className = 'locos';
loco.id = 'loco_' + name;

loco.style.fontSize = '14px';
loco.style.color = '#9cf';

var acco = document.createElement('div');
acco.id = 'acco_' + name;
acco.className = 'accos';
container.appendChild(acco);

container.appendChild(slider);
container.appendChild(loco);
```

```
container.appendChild(onemode);
if (name != myname) {container.appendChild(dropper);}

var anno = document.createElement('div');
anno.className = 'annos';
anno.id = 'anno_' + name;
anno.style.fontSize = '16px';
anno.style.paddingLeft = '10px';
anno.style.paddingRight = '10px';
anno.style.opacity = 0;
anno.onclick = function(e) {toggleAnnoVisibility(e.target)};
container.appendChild(anno);

var lol = document.createElement('div');
lol.className = 'lols';
lol.id = 'lol_' + name;
lol.style.fontSize = '14px';
lol.onclick = toggleSignal;
lol.style.position = 'absolute';
lol.style.zIndex = '1001';
container.appendChild(lol);

var rew = document.createElement('div');
rew.className = 'lols';
rew.id = 'rew_' + name;
rew.style.right = '120px';
rew.style.fontSize = '14px';
rew.onclick = rewind;
rew.style.position = 'absolute';
rew.style.zIndex = '1001';
container.appendChild(rew);

var sndMeter = document.createElement('div');
sndMeter.className = 'snd_meters';
sndMeter.id = 'sndm_' + name;
container.appendChild(sndMeter);

var adder = document.createElement('div');
adder.className = 'adders';
adder.style.fontSize = '18px';
adder.style.cursor = 'pointer';
adder.style.background = name == myname ? '#369' : '#900';
adder.id = 'adder_' + name;
```

```
     adder.appendChild(document.createTextNode('A'));
     adder.onclick = set_Anno;

     adder.style.display = myrole == 0 && myname == name ? 'none': 'block';

     adder.style.display = (this_is_guru && name != myname) || ar[0] === "DUMMY"
? 'none' : adder.style.display;
     adder.style.right = name == myname ? '0px' : '24px';

     container.appendChild(adder);

     document.id(video.id).style.opacity = (i_am_muted === true || i_am_muted
=== 'true') && aonly && name == myname? 0 : 1;
     document.id(video.id).style.maxHeight = (i_am_muted === true || i_am_muted
=== 'true') && aonly && name == myname ? notebook ? '310px' : '365px':
document.id(video.id).style.maxHeight;

     document.id(video.id).style.maxHeight = notebook ? '310px' : '365px';

     if ((all_muted === true || all_muted === 'true') || (coo_muted === true ||
coo_muted === 'true') || name == myname) video.muted = true;

     if (video.muted !== true){
          speaker.appendChild(document.createTextNode('\uD83D\uDD0A'));
//speaker icon
     } else {
          if (name != myname && mode != 'm')
speaker.appendChild(document.createTextNode('\uD83D\uDD07')); //muted icon
          if (name != myname && mode == 'm')
speaker.appendChild(document.createTextNode('X'));
          if (name == myname && (i_am_muted === true || i_am_muted === 'true')
&& !playSomeMusic) speaker.appendChild(document.createTextNode('X'));
          if (name == myname && playSomeMusic_muted === true && playSomeMusic)
speaker.appendChild(document.createTextNode('\uD83D\uDD07')); //muted icon
          if (name == myname && playSomeMusic_muted === false && playSomeMusic)
speaker.appendChild(document.createTextNode('\uD83D\uDD0A')); //speaker icon
```

```
          if (name == myname && (i_am_muted === false || i_am_muted ===
'false') && !playSomeMusic)
speaker.appendChild(document.createTextNode('\uD83C\uDFA4')); //microphone icon
     }

     speaker.style.fontSize = "42px";
     speaker.style.right = small_device && window.top != window ? "10px" :
speaker.style.right;

     this.getVideoElement = function() {
          return video;
     }

     this.getCanvasElement = function() {
          return canvas;
     }

     this.activateMicIndic = function() {

       if (name != myname && !playSomeMusic) {
          let ac = new AudioContext();
          console.log('newed ac1..');
          let soundMeter = new SoundMeter(ac);

          let vidStream = this.rtcPeer.getRemoteStream(); let audio =
vidStream.getAudioTracks()[0];

          const audioContext = new AudioContext();
          const analyzer = audioContext.createAnalyser();
          analyzer.fftSize = 512;
          analyzer.smoothingTimeConstant = 0.1;
          const sourceNode = audioContext.createMediaStreamSource(vidStream);
          sourceNode.connect(analyzer);

          if (vidStream.getAudioTracks()[0])
soundMeter.connectToSource(vidStream, function(e) {if (e) {console.log('error
connectToSource',e); return;}
          let inte;
          if (document.id('sndm_'+name)) {inte = setInterval(() => {
```

```
          const fftBins = new Float32Array(analyzer.frequencyBinCount); //
Number of values manipulated for each sample
          analyzer.getFloatFrequencyData(fftBins);

          const audioPeakDB = Math.max(...fftBins);

          const frequencyRangeData = new
Uint8Array(analyzer.frequencyBinCount);
          analyzer.getByteFrequencyData(frequencyRangeData);
          const sum = frequencyRangeData.reduce((p, c) => p + c, 0);

          const audioMeter = Math.sqrt(sum / frequencyRangeData.length);
            if (document.id('sndm_'+name)) {/*console.log('..
received',soundMeter.instant.toFixed(3));*/document.id('sndm_'+name).style.backg
round = soundMeter.instant.toFixed(3) > 0.001 ? '#090' : 'transparent'; if
(audioMeter === 0 && audioPeakDB < -200 && !mic_muted) { mic_muted = true; let
speaker = document.id('speaker-'+name);
speaker.removeChild(speaker.childNodes[0]);speaker.appendChild(document.createTe
xtNode('X'));speaker.style.pointerEvents='none';} if (audioMeter > 0 &&
mic_muted) { mic_muted = false; let speaker = document.id('speaker-'+name);
speaker.removeChild(speaker.childNodes[0]);if (!video.muted)
speaker.appendChild(document.createTextNode('\uD83D\uDD0A'));if (video.muted)
speaker.appendChild(document.createTextNode('\uD83D\uDD07'));speaker.style.point
erEvents='auto';}} else {clearInterval(inte)}

          }, 200);}

          });
      }
    }


    function toggleRoomHeader() {

              if(!small_device) {document.id('room-header').style.display =
document.id('room-header').style.display === 'none' ? 'block' : 'none'; let a =
document.id('room-header').style.display === 'none' ? 0 : 1; document.id('room-
header').fade(a); document.id('room-header-file').style.display =
document.id('room-header').style.display === 'none' ? 'none' :
document.id('room-header-file').style.display;}
    }
```

```
     function set_Anno() {
                    who_to = name;
                    document.id('anno_adder').click();
                    //console.log('clicked set_anno!');
     }

     function setCinema() {

     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {
                let role = respo;
                if (role == 1) {
                     let p = participants[name]; let g = p.getMode();
                     let m = g == 'c' ? 'v'  : 'c'; //toggle
                     let mo =  g == 'c' ? false : true;
                     let tok = getCookie('authtoken') || '';

                     let message = {
                          id : 'setCinema',
                          name : name,
                          mode : m,
                          token: tok
                     }
                     sendMessage(message);
                }

          }).catch(err => console.log(err));
     }

     function back_to_audience() {

     fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {
                let role = respo;
                if (role == 1 && name !== myname) set_guru(0, name);

          }).catch(err => console.log(err));
     }

     function toggleSignal() {
```

```
            let message = {
                  id :'keyDown', num: 85, name: name //pause/play toggle
            }
            sendMessage(message);
      }

      function toggleAnnoVisibility(el) {
            el.style.opacity = el.style.opacity == 0.02 ? 1 : 0.02;
      }

      function rewind() {
            let message = {
                  id :'keyDown', num: 82, name: name //rewind 10sec
            }
            sendMessage(message);
      }

      const switchOneMode = (el) => {
            document.id('house').style.display = 'block';
document.id('house').style.visibility='visible';


request('https://'+window.location.hostname+port+'/cgi/genc/get_acc_id.pl').then
(data => {
                  if (data.length) {

doSwitchOneMode(el,'5ENzTTUL3zvnMP8usRo3ZcGmMhkaHsvFUP6PMedLV9EWtLFx',5);
//sorba
                  } else {
                        if (!afterBinding) {

                              setTimeout(function() {
                                    if (document.id('removerA'))
{document.id('removerA').innerHTML = 'Error: Service unavailable';
                                    (function()
{document.id('removerA').fade(0)}).delay(1000);}
                              }, 10000);
                              (function() {
                                    if (document.id('sp_balance')){

      document.id('sp_balance').style.display='block';

      document.id('sp_balance').src=sp_container_url+'/?acc='+data;
                                    }
```

```
                        }).delay(1000);
                        document.id('sp_container').style.display = 'block';
sp_shown = 1;
                        ch_int = setInterval(function() {
                              if (document.id('sp_balance')) {

     document.id('sp_balance').style.display='block';
                                    document.id('sp_balance').src =
sp_container_url + '/?acc=' + data;
                              }
                        }, 300000);
                        document.body.scrollTop =
document.documentElement.scrollTop = 0;

                  }
                  if (afterBinding) rejoin();

            }
      });

     }

     function switchContainerClass() {

           if (!small_device) {

           if (container.className === PARTICIPANT_CLASS) {
                var elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_MAIN_CLAS
S));
                elements.forEach(function(item) {

                  item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_CLASS;
                });

                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_SOLO));
                    elements.forEach(function(item) {

                  item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_CLASS;
                });
```

```
                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_DUO));
                        elements.forEach(function(item) {

                      item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_CLASS;
                });

                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_TRIO));
                        elements.forEach(function(item) {

                      item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_CLASS;
                });

                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_CUATRO));
                        elements.forEach(function(item) {

                      item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_CLASS;
                });

                elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_CINCO));
                        elements.forEach(function(item) {

                      item.className = small_device ? PARTICIPANT_MAIN_CLASS :
PARTICIPANT_CLASS;
                });

                      container.className = PARTICIPANT_MAIN_CLASS;
           } else {
                      container.className = PARTICIPANT_CLASS;
           }

            }
     }
```

```
	function toggleFullScreen(el) {

		if (!el.fullscreenElement) {

			el.requestFullscreen();
			document.id('room_selector').fade(0);


		} else {
			if (el.exitFullscreen) {
				//fullscreen = false; //uncomment for 'strict' chasing
				el.exitFullscreen();
				document.id('room_selector').fade(1);

			}
		}

	}

	function toggleBigScreen(el) {

			if(!isAndroid && !ios_fullscreen) {
				document.id('house').style.visibility='hidden';
				document.id('controls').style.display='none';
				container.style.position='fixed';
				container.style.top='0px';
				container.style.left='0px';
				container.style.width= window.innerHeight >
window.innerWidth ? '97vw' : '98vw';

flashText('swipe screen');
				ios_fullscreen = true;
				document.id('room_selector').fade(0);
			}
			else if(!isAndroid && ios_fullscreen) {
				container.style.position='relative';
				container.style.width='96%';
				document.id('house').style.visibility='visible';
				document.id('controls').style.display='block';

				ios_fullscreen = false;
```

```
                    document.id('room_selector').fade(1);
               }
               else if (isAndroid) {
                    if (container.className == PARTICIPANT_MAIN_CLASS)
toggleFullScreen(el);
                    if (container.className == PARTICIPANT_CLASS)
switchContainerClass();
               }
     }

     function toggleBigScreen_new(el) {

               if(!ios_fullscreen) {
                    document.id('house').style.visibility='hidden';
                    document.id('controls').style.display='none';
                    container.style.position='fixed';
                    container.style.top='0px';
                    container.style.left='0px';
                    container.style.width= window.innerHeight >
window.innerWidth ? '97vw' : '98vw';

flashText('swipe screen');
                    ios_fullscreen = true;
               }
               else if(ios_fullscreen) {
                    container.style.position='relative';
                    container.style.width='96%';
                    document.id('house').style.visibility='visible';
                    document.id('controls').style.display='block';
                    //document.id('second_logger').style.visibility='visible';
                    //screen.orientation.unlock();
                    ios_fullscreen = false;
               }
     }

     function toggleSlider() {
          if (check_iOS()) from_changing_slider = false;
          else slider.style.display = slider.style.display == 'block' ?
'none' : 'block';

          toggleMute();
     }
```

```
	function changeVolume() {

		coo_volume=slider.value;
		saveData(name+'_volume', coo_volume, 1440);
		coo_muted  = coo_volume > 0 ? false : true;
		saveData(name+'_muted', coo_muted, 1440);

		video.volume = coo_volume;
		from_changing_slider = true;

		if ( (video.volume === 0 && !video.muted) || (video.volume &&
video.muted) ) toggleMute();
	}

	function check_iPad() {
		var ua = navigator.userAgent;
		if (/iPad/.test(ua) ){
			return true;
		}
		return false;
	}

	function toggleMute() {
		if ( document.id('video-' + name) ) {
		   var video = document.id('video-' + name);
		   if ( !from_changing_slider || (from_changing_slider &&
((video.volume === 0 && !video.muted) || (video.volume > 0))) ) {

			if (name != myname)
			{
				video.muted = video.volume === 0 ? true : !video.muted;
				slider.value =  video.volume;

				from_changing_slider = false;

				saveData(name+'_muted', video.muted, 1440);
				saveData(name+'_volume', video.volume, 1440);
			}

			if (name == myname && playSomeMusic)
			{
```

```
                    playSomeMusic_muted = playSomeMusic_muted === true ? false
: true;
                }

                speaker.removeChild(speaker.childNodes[0]);

                if (video.muted || (name == myname && playSomeMusic &&
playSomeMusic_muted === true )){
                    if (name != myname || (name == myname && playSomeMusic))
{

     speaker.appendChild(document.createTextNode('\uD83D\uDD07'));//muted icon

                        if (name == myname && playSomeMusic)
analyser.disconnect();
                    }
                    if (name == myname && !playSomeMusic) {
                        if (i_am_muted === true || i_am_muted === 'true') {
                            saveData(myname+'_muted', false, 1440);
i_am_muted = false;
                            setCookie('all_muted', false, 1440);// have to
add this too

                            localStream.getAudioTracks()[0].enabled = true;

     speaker.appendChild(document.createTextNode('\uD83C\uDFA4'));
                        }
                        else {
                            saveData(myname+'_muted', true, 1440);
i_am_muted = true;

                            localStream.getAudioTracks()[0].enabled =
false;

     speaker.appendChild(document.createTextNode('X'));
                        }
                    }

                } else {
```

```
      speaker.appendChild(document.createTextNode('\uD83D\uDD0A'));//speaker icon

                        if (name == myname && playSomeMusic)
                        {
                              if (!g.audioContext) g.audioContext = new
AudioContext();
                              mediaSource =
g.audioContext.createMediaElementSource(video);
                              analyser = g.audioContext.createAnalyser();
                              mediaSource.connect(analyser);

                              analyser.connect(g.audioContext.destination);

                        }
                  }
               }
            }
      }

      function rmPtcp() {
        if (!normal_mode) {let foo = document.location.href.split('#'); if
(true_cine) {document.location.href = foo[0];} else
{document.location.reload();}} else {
          if (name != myname) {
           var yon = window.confirm('Drop '+rname+'?!');
           if (yon) {

      fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {
                        let role = respo;
                        if (role == 1) {

                              let tok = getCookie('authtoken') || '';

                              let message = {
                                    id : 'dropGuest',
                                    name : name,
                                    token: tok
                              }
                              sendMessage(message);
```

```
                    } else {

                          for ( var key in participants) {
                               if (participants[key].name === name) {
                                    participants[key].dispose();
                                    delete participants[key].rtcPeer;
                                    delete participants[key];
                               }
                          }
                    }

               }).catch(err => console.log(err));



          }
         }
       }
     }

     function isPresentMainParticipant() {
          return
( (document.getElementsByClassName(PARTICIPANT_MAIN_CLASS)).length != 0 ||
(document.getElementsByClassName(PARTICIPANT_SOLO)).length != 0 ||
(document.getElementsByClassName(PARTICIPANT_DUO)).length != 0 ||
(document.getElementsByClassName(PARTICIPANT_TRIO)).length != 0 ||
(document.getElementsByClassName(PARTICIPANT_CUATRO)).length != 0 ||
(document.getElementsByClassName(PARTICIPANT_CINCO)).length != 0);
     }



     this.offerToReceiveVideo = function(error, offerSdp, wp){
          if (error) return console.error ("sdp offer error")

          var msg =  { id : "receiveVideoFrom",
                       sender : name,
                       sdpOffer : offerSdp
                    };
          sendMessage(msg);
     }

     this.onIceCandidate = function (candidate, wp) {
```

```
            var message = {
              id: 'onIceCandidate',
              candidate: candidate,
              name: name
            };
            sendMessage(message);
      }

      Object.defineProperty(this, 'rtcPeer', { writable: true});

      this.dispose = function() {

            var gi = new RegExp('GURU:','g');
            let ct = 0;
            for ( var key in participants) {
                  if (key.match(gi)) ct++;
            }

            if (name.match(gi) && ct == 1) {
                  guru_is_here = 0;
                  setTimeout(function() {
                        if (!guru_is_here && false) {
                              flashText(waiter);
                        }
                  }, 2000);
            }
            if (this.rtcPeer && typeof(this.rtcPeer != 'undefined') )
this.rtcPeer.dispose();
            if (container && container.parentNode)
container.parentNode.removeChild(container);
            pcounter--;

            if (pcounter <= room_limit && !i_am_viewer) {hack = true;}

            if (pcounter < 6) {
                        var elements =
Array.prototype.slice.call(document.getElementsByClassName(PARTICIPANT_CLASS));
                        elements.forEach(function(item) {
```

```
                                item.className = small_device ?
PARTICIPANT_MAIN_CLASS : pcounter === 1 ? PARTICIPANT_SOLO : pcounter === 2 ?
PARTICIPANT_DUO : pcounter === 3 ? PARTICIPANT_TRIO : pcounter === 4 ?
PARTICIPANT_CUATRO : pcounter === 5 ? PARTICIPANT_CINCO : item.className;
                        });

          }

          if (name != myname || myrole != 0) real_pcnt--;
          if (document.id('pcounter')) { if (!real_pcnt) {(function()
{document.id('pcounter').innerHTML = real_pcnt;}).delay(1000);} else
{document.id('pcounter').innerHTML = real_pcnt;} }

          if ( guru_is_here == 0 & pcounter) i_am_guest = 1;
          if ( guru_is_here == 1 & pcounter === 1) i_am_guest = 0;
          if ( ct > 1) {i_am_guest = 0 ; guru_is_here = 1;}

     }

     this.setMode = function(r) {
          this.mode = r;
     }

     this.getMode = function() {
          return this.mode;
     }
}

const CINEMA_GOER_MAIN_CLASS = small_device ? 'cinema_goer main_i' :
'cinema_goer main';

function CinemaGoer(name, myname, mode) {

     this.name = name;
     this.mode = mode;

     if ( document.id(name) ) {var old_container = document.id(name);
old_container.parentNode.removeChild(old_container);}

     var from_changing_slider = false;

     all_muted = getCookie('all_muted');
     if (all_muted === null || all_muted === 'null') all_muted = false;
```

```
	var coo_muted = loadData(name+'_muted');

	if (coo_muted === null || coo_muted === 'null') coo_muted = all_muted;

	if (mode == 'm' || mode == 'c' || mode == 'p') coo_muted = true; // ?!

	var coo_volume = loadData(name+'_volume');

	if (coo_volume === null || coo_volume === 'null') coo_volume = 0.3; // set
default volume on cinema goers

	if (name != myname && mode != 'm') {
		saveData(name+'_muted', coo_muted, 1440);
		saveData(name+'_volume', coo_volume, 1440);
	}

	if ((all_muted === true || all_muted === 'true') && name == myname)
i_am_muted = true;

	if (name == myname) saveData(myname+'_muted', i_am_muted, 1440);

	var container = document.createElement('div');

	container.className = CINEMA_GOER_MAIN_CLASS;

	container.style.display='block';
	container.style.opacity=0;

	container.id = name;
	container.style.position='absolute';
	container.style.zIndex = 10112;

	container.style.bottom = '0px';

	var speaker = document.createElement('div');
	var slider = document.createElement('input');
	var video = document.createElement('video');

	var rtcPeer;
```

```
     document.id('participants').appendChild(container);

     container.appendChild(video);
     container.appendChild(speaker);

     if (mode != 'm' || myname == name) {
          speaker.addEventListener('click', function(e) {e.preventDefault();
e.stopPropagation(); if (name != myname) toggleSlider(); else toggleMute();});
          speaker.style.cursor = 'pointer';
     }

     slider.addEventListener('change', function(e) {e.preventDefault();
e.stopPropagation(); changeVolume();});

     speaker.className = 'speak';
     speaker.id = 'speaker-' + name;
     speaker.style.zIndex = 10112;

     slider.style.position = 'relative';
     slider.style.zIndex = 10112;

     video.id = 'video-' + name;
     video.autoplay = true;
     video.playsInline = true;
     video.controls = false;
     video.volume = coo_volume;
     if (name != myname) video.style.cursor = 'pointer';

     slider.type = 'range';
     slider.id = 'slider_' + name;
     slider.min = 0;
     slider.max = 1;
     slider.step = 0.1;
     slider.style.display='none';
     slider.style.width='90px';

     container.appendChild(slider);

     speaker.style.display = 'none';

     if ((all_muted === true || all_muted === 'true') || (coo_muted === true ||
coo_muted === 'true') || name == myname) video.muted = true;
```

```
     if (video.muted !== true){
          speaker.appendChild(document.createTextNode('\uD83D\uDD0A'));
//speaker icon

     } else {
          if (name != myname && mode != 'm')
speaker.appendChild(document.createTextNode('\uD83D\uDD07')); //muted icon
          if (name != myname && mode == 'm')
speaker.appendChild(document.createTextNode('X'));
          if (name == myname && (i_am_muted === true || i_am_muted === 'true'))
speaker.appendChild(document.createTextNode('X'));
          if (name == myname && (i_am_muted === false || i_am_muted ===
'false')) speaker.appendChild(document.createTextNode('\uD83C\uDFA4'));
//microphone icon
     }

     speaker.style.fontSize = "42px";
     speaker.style.right = small_device && window.top != window ? "10px" :
speaker.style.right;

     this.getVideoElement = function() {
          return video;
     }

     this.activateMicIndic = function() {

       if (name != myname) {
          let ac = new AudioContext();
          console.log('newed ac2..');
          let soundMeter = new SoundMeter(ac);

          let vidStream = this.rtcPeer.getRemoteStream(); let audio =
vidStream.getAudioTracks()[0];

          if (vidStream.getAudioTracks()[0])
soundMeter.connectToSource(vidStream, function(e) {if (e) {console.log('error
connectToSource',e); return;}
          let inte;
          if (document.id('sndm_'+name)) {inte = setInterval(() => {
```

```
            if (document.id('sndm_'+name)) {/*console.log('..
received',soundMeter.instant.toFixed(3));*/document.id('sndm_'+name).style.backg
round = soundMeter.instant.toFixed(3) > 0.001 ? '#090' : 'transparent'} else
{clearInterval(inte)}

          }, 200);}

          });
       }
     }

     function toggleSlider() {
          if (check_iOS()) from_changing_slider = false;
          else slider.style.display = slider.style.display == 'block' ?
'none' : 'block';

          toggleMute();
     }

     function changeVolume() {

          coo_volume=slider.value;
          saveData(name+'_volume', coo_volume, 1440);
          coo_muted  = coo_volume > 0 ? false : true;
          saveData(name+'_muted', coo_muted, 1440);

          video.volume = coo_volume;
          from_changing_slider = true;

          if ( (video.volume === 0 && !video.muted) || (video.volume &&
video.muted) ) toggleMute();
     }

     function toggleMute() {
          if ( document.id('video-' + name) ) {
             var video = document.id('video-' + name);
             if ( !from_changing_slider || (from_changing_slider &&
((video.volume === 0 && !video.muted) || (video.volume > 0))) ) {

               if (name != myname)
               {
                    video.muted = video.volume === 0 ? true : !video.muted;
                    slider.value =  video.volume;
```

```
                        from_changing_slider = false;

                        saveData(name+'_muted', video.muted, 1440);
                        saveData(name+'_volume', video.volume, 1440);
                  }

                  speaker.removeChild(speaker.childNodes[0]);

                  if (video.muted){
                        if (name != myname)  {

      speaker.appendChild(document.createTextNode('\uD83D\uDD07'));//muted icon

                        }
                        if (name == myname) {
                              if (i_am_muted === true || i_am_muted === 'true') {
                                    saveData(myname+'_muted', false, 1440);
i_am_muted = false;
                                    setCookie('all_muted', false, 1440);// have to
add this too
                                    flashText_and_rejoin('microphone ON!');
                              }
                              else {
                                    saveData(myname+'_muted', true, 1440);
i_am_muted = true;
                                    flashText_and_rejoin('microphone OFF!');
                              }
                        }

                  } else {

      speaker.appendChild(document.createTextNode('\uD83D\uDD0A'));//speaker icon
                  }
               }
            }
      }

      this.offerToReceiveVideo = function(error, offerSdp, wp){
            if (error) return console.error ("sdp offer error")
```

```
            var msg =  { id : "receiveVideoFrom",
                        sender : name,
                        sdpOffer : offerSdp
                     };
            sendMessage(msg);
      }


      this.onIceCandidate = function (candidate, wp) {

              var message = {
                id: 'onIceCandidate',
                candidate: candidate,
                name: name
              };
              sendMessage(message);
      }

      Object.defineProperty(this, 'rtcPeer', { writable: true});

      this.dispose = function() {}

      this.setMode = function(r) {
            this.mode = r;
      }

      this.getMode = function() {
            return this.mode;
      }
}
```

2.3. starter_a.js

```
var curIP = '';

var aonly = 1;

var mod1; var mod2; var mod5; var mod8;

var altlang = ['en','ru','es','fr','cn','pt'];

var cam1 = getCookie('d1');
var cam2 = getCookie('d2');
var scanned = getCookie('s1');
```

```
const soundEffect = new Audio();
soundEffect.autoplay = true;

var snd_clicked = false;
var dummies = false;

var oldColor = null;

var selectedFile = null;
var movieList_names = [];

var loado = ''; var warning = ''; var waiter = ''; var sorry = ''; var hola =
'''; var caller = ''; var requ = ''; var creatu = ''; var badger = ''; var
learner = ''; var morer = ''; var hea = ''; var now = ''; var today = ''; var
roo = ''; var buy = ''; var helpcapo = '';
var chat_shown = 1;

var temporary = 0;

var cammode = 0;

var stats_shown = 1;

var sp_shown = (window == window.top && (w[0].match(new RegExp('skypirl','g'))
|| w[0].match(new RegExp('club','g')) || w[0].match(new
RegExp('milan','g')) )) ? 0 : 0;

var voting_shown = w[0] === "club" && small_device && !sp_shown ? 0 : 0;

var sound_on_played = w[0] === "club" || w[0].match(new RegExp('rgsu','g')) ?
1 : 0;

var scrolled = false;

var heard_info = getCookie('heard_info') || false;

var who_to = '';

var already_clicked = false;

var afterBinding = false;

var ch_int; var chr_int;

var curSelInd = 0;

var curMoviesList = []
```

```
var curMovie = ''

var switched = false;

var lori = small_device && window.innerWidth > window.innerHeight ? true :
false;

function getIP(json) {
    curIP = json.ip;
     document.id('curip').value = curIP;
}

function ajax_chat() {
     fetch('https://'+window.location.hostname+port+'/log.html').then(response
=> response.text()).then((response) => {document.id('message_box').innerHTML =
response; }).catch(err => console.log(err));
}

function ajax_room(name) {
     let banner = name == 'REDHALL' ? 'red_hall_span' : name == 'BLUEHALL' ?
'blue_hall_span' : name == 'GREENHALL' ? 'green_hall_span' :''; if
(banner.length == 0) return;
     fetch('https://'+window.location.hostname+'/cgi/genc/get_room_data.pl?
room='+name).then(response => response.text()).then((response) => {if
(response.length) {document.id(banner).innerHTML = '"'+response+'"';} else
{document.id(banner).innerHTML = '-'} }).catch(err => console.log(err));
}

function change_lang(l) {

     if (document.id('head1')) document.id('head1').innerHTML=heads1.get(l);
     if (document.id('abouter')) document.id('abouter').innerHTML=about.get(l);
     if (document.id('helper')) document.id('helper').innerHTML=help.get(l);
     if (document.id('leftlab'))
document.id('leftlab').innerHTML=left_label.get(l);
     if (document.id('rightlab'))
document.id('rightlab').innerHTML=right_label.get(l);
     if (document.id('dummy_p')) document.id('dummy_p').innerHTML=du_.get(l);
     if (document.id('dummy2_p')) document.id('dummy2_p').innerHTML=du2_.get(l);

     warning = warner.get(l);
     waiter = waiter_.get(l);
     sorry = sorry_.get(l);
     loado = loado_.get(l);

     caller = caller_.get(l);
     requ = requ_.get(l);
```

```
      creatu = creatu_.get(l);
      badger = badger_.get(l);
      learner = learner_.get(l);
      morer = morer_.get(l);
      hea = hea_.get(l);
      now = now_.get(l);
      today = today_.get(l);
      roo = roo_.get(l); roo = roo.match(new RegExp('room-house','g')) ? '' :
roo;// hack --ash

      if (typeof(mod1) != 'undefined' && mod1 !== null) mod1.content.innerHTML =
about_content.get(l);
      if (typeof(mod2) != 'undefined' && mod2 !== null) mod2.content.innerHTML =
help_content.get(l);

      if (typeof(mod3) != 'undefined' && mod3 !== null) mod3.content.innerHTML =
left_content.get(l);
      if (typeof(mod4) != 'undefined' && mod4 !== null) mod4.content.innerHTML =
right_content.get(l);
}

function checkWebRTC() {

        var isWebRTCSupported = navigator.getUserMedia ||
        navigator.webkitGetUserMedia ||
        navigator.mozGetUserMedia ||
        navigator.msGetUserMedia ||
        window.RTCPeerConnection;

        if (window.navigator.userAgent.indexOf("Edge") > -1 || !
isWebRTCSupported) {
            alert('Please use a different browser with WebRTC support!');
          console.log('Bad WebRTC non-compatible bro');
          return false;

        }

      try {
           const pc = new RTCPeerConnection();
           const transceiver = pc.addTransceiver('audio');

      } catch (e) {
           console.log(e);
           alert('Please use a newer browser with full WebRTC support!');
      }
}
```

```
function toggleHeader(i) {

           let role = 1; //hack ash
           if (i == 1 && role == 1 && aonly)  {
                document.id('room-header').style.color = document.id('room-
header').style.color == 'rgb(153, 204, 255)' ? oldColor : '#9cf';
                document.id('room-header-file').style.display =
document.id('room-header-file').style.display == 'block' ? 'none' : 'block';
                playSomeMusic = playSomeMusic == true ? false : true;
                setCookie('player', playSomeMusic);
           }
           if (i == 2 && role == 1 && aonly)  {
                document.id('room-header-file').style.display='none';
                document.id('room-header').style.color = document.id('room-
header').style.color == 'rgb(153, 255, 204)' ? oldColor : '#9fc';
                shareSomeScreen = shareSomeScreen == true ? false : true;
                setCookie('player', '');
           }
}

function getFile() {

     let fi = document.id('room-header-file').files[curSelInd];
     if (selectedFile) URL.revokeObjectURL(selectedFile);
     selectedFile = URL.createObjectURL(fi);
}

function getList_names() {
     movieList_names = [];
     let fis = document.id('room-header-file').files;
     for (var i = 0; i < fis.length; i++) {
          movieList_names.push(fis[i].name);
     }

     let str = JSON.stringify(movieList_names);
     let tok = getCookie('authtoken') || '';
     let mes = {id : 'moviesList', listStr : str, token: tok, addr:
document.id('name').value, curSel: curSelInd};
     sendMessage(mes);
}

function clearAllCookies() {

    var cookies = document.cookie.split(";");

    for (var i = 0; i < cookies.length; i++) {
        var cookie = cookies[i];
```

```
        var eqPos = cookie.indexOf("=");
        var coo_name = eqPos > -1 ? cookie.substr(0, eqPos) : cookie;
      while (coo_name.charAt(0)==' ') coo_name =
coo_name.substring(1,coo_name.length);

        document.cookie = coo_name + "=;expires=Thu, 01 Jan 1970 00:00:00
GMT;SameSite=None;Secure;";
    }

    if (typeof(Storage) != 'undefined') localStorage.clear();
}

function flashText(t, tt) {
     let ti = tt ? tt : 1000;
     document.id('phones').innerHTML = t; document.id('phones').fade(1);
(function(){document.id('phones').fade(0);}).delay(ti);
}

function flashText_and_rejoin(t) {
     document.id('room-header').style.display = 'none';document.id('room-
header').fade(0); document.id('phones').innerHTML = t;
document.id('phones').fade(1); if (i_am_viewer || true) {leaveRoom();
register();} else {/*showMeAsParticipant();*/}
     (function(){document.id('phones').fade(0);}).delay(1000);
}

function resizer(pctr) {
}

function toggleAllMuted() {

     let curr_all_muted = getCookie('all_muted') || false;
     let my_mic_muted = loadData(document.id('name').value+'_muted');

     if (!i_am_muted && document.id('all_muter').className == "bigO
my_mic_on_all_off") {

          document.id('all_muter').className = "bigO allmuter_on";
          document.id('all_muter').title = 'Turn off sound';
          setCookie('all_muted', false, 1440);
          if (Object.keys(participants).length) {
               for (var key in participants) {
                 let video = document.id('video-' + participants[key].name);
                 let spea = document.id('speaker-' + participants[key].name);

                 if (participants[key].name != document.id('name').value) {
```

```
                    video.muted = false;
                    video.volume = 1;
                    spea.removeChild(spea.childNodes[0]);
                    participants[key].mode !== 'm' &&
spea.appendChild(document.createTextNode('\uD83D\uDD0A'));//speaker icon
                    participants[key].mode === 'm' &&
spea.appendChild(document.createTextNode('X'));//x icon
                  }
                }
          }
     } else if (!i_am_muted && document.id('all_muter').className == "bigO
allmuter_on") {

          document.id('all_muter').className = "bigO my_mic_on_all_off";
          document.id('all_muter').title = 'Turn on sound';

          if (Object.keys(participants).length) {
               for (var key in participants) {
                 let video = document.id('video-' + participants[key].name);
                 let spea = document.id('speaker-' + participants[key].name);

                 if (participants[key].name != document.id('name').value) {

                    video.muted = true;
                    video.volume = 0;
                    spea.removeChild(spea.childNodes[0]);
                    participants[key].mode !== 'm' &&
spea.appendChild(document.createTextNode('\uD83D\uDD07'));//muted icon
                    participants[key].mode === 'm' &&
spea.appendChild(document.createTextNode('X'));//x icon
                  }
                }
          }
     } else {

          let n = curr_all_muted ? false : true;
          let t = curr_all_muted ? 'UNMUTED' : 'MUTED ALL';
          if ((my_mic_muted === true || my_mic_muted === 'true') && !
curr_all_muted) {saveData(document.id('name').value+'_muted', false, 1440);
i_am_muted = false; t = 'MICROPHONE ON';} else {setCookie('all_muted', n,
1440);}

          if (n === false) {saveData(document.id('name').value+'_muted', false,
1440); i_am_muted = false;} //?! need this for brute force, unmute mic by
toggle_mute

          document.id('all_muter').title = curr_all_muted ? 'Turn on sound' :
'Turn off all sound';
```

```
          if (Object.keys(participants).length) {
               for (var key in participants) {
                if (participants[key].name != document.id('name').value) {
                     let n = participants[key].name + '_muted';

                     if (typeof(Storage) != 'undefined') {
                          localStorage.removeItem(n);

                     } else {
                          document.cookie = n + "=;expires=Thu, 01 Jan 1970
00:00:00 GMT;SameSite=None;Secure;";
                     }

                     var video = document.id('video-' +
participants[key].name);
                 }
               }
          }

          flashText_and_rejoin(t);
     }
}

window.addEventListener("message", function(event) {

let da = event.data; // console.log ('da0 is', da[0]);

if (typeof(da[0]) === 'undefined' || da[0] == "esms" || (event.origin !=
'https://'+window.location.hostname+':18443' && event.origin !=
'https://'+window.location.hostname+port && event.origin != sp_container_url &&
event.origin != sp_setter_url && event.origin != sm_url && event.origin !=
"https://room-house.com" && event.origin != chess_url && event.origin !=
poker_url && event.origin != air_url && event.origin != swap_url)) {
    return;
  }

  if ((event.origin == 'https://'+window.location.hostname+port) ||
(event.origin == 'https://'+window.location.hostname+':18443')) {

var obj = JSON.parse(da);

if (obj.replacer && obj.replacer.length) {
     var roo = window.location.hostname.split('.');
```

```
     var foo = document.location.href.split('#'); var zoo = foo[1] &&
foo[1].length ? foo[1] : roo[0];
     console.log('replaced', zoo, 'with', obj.replacer, 'in',
document.location.href);

     document.id('phones').style.display='none';
     document.id('container').style.display='none';
     //works different for tower name and apart name
     if (zoo == roo[0]) {

     document.location.href=document.location.href.replace(zoo,obj.replacer);
     } else {
          history.pushState({}, null,
document.location.href.replace(zoo,obj.replacer));
          window.location.reload();
     }
}
     soundEffect.src =
'data:audio/mpeg;base64,SUQzBAAAAAABEVRYWFgAAAAtAAADY29tbWVudABCaWdTb3VuZEJhbmsu
Y29tIC8gTGFTb25vdGhlcXVlLm9yZwBURU5DAAAAHQAAA1N3aXRjaCBQbHVzIMKpIE5DSCBTb2Z0d2Fy
ZQBUSVQyAAAABgAAAzIyMzUAVFNTRQAAAA8AAANMYXZmNTcuODMuMTAwAAAAAAAAAAAAAAD/80DEAAAA
A0gAAAAATEFNRTMuMTAwVVVVVVVVVVVVVUxBTUUzLjEwMFVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVV';

const sho = () => {

     (function() {if (document.id('red_banner')) {if
(document.id('red_banner').style.display == 'none')
{document.id('red_banner').style.display='block';
document.id('red_banner').fade(1);} else { document.id('red_banner').fade(0);
(function() { document.id('red_banner').style.display =
'none';}).delay(500);}}}).delay(100);
     (function() {if (document.id('green_banner')) {if
(document.id('green_banner').style.display == 'none')
{document.id('green_banner').style.display='block';
document.id('green_banner').fade(1);} else
{ document.id('green_banner').fade(0); (function()
{ document.id('green_banner').style.display =
'none';}).delay(500);}}}).delay(100);
     (function() {if (document.id('blue_banner')) {if
(document.id('blue_banner').style.display == 'none')
{document.id('blue_banner').style.display='block';
document.id('blue_banner').fade(1);} else { document.id('blue_banner').fade(0);
(function() { document.id('blue_banner').style.display =
'none';}).delay(500);}}}).delay(100);
```

```
     setTimeout(function()
{ajax_room('BLUEHALL');ajax_room('GREENHALL');ajax_room('REDHALL');}, 1000);
     chr_int = setInterval(function() { if (document.id('ph_b') &&
document.id('ph_b').style.visibility == 'visible' && homee == w[0])
{ajax_room('BLUEHALL');ajax_room('GREENHALL');ajax_room('REDHALL');}}, 30000);
}

const ed = () => { //code to run on receive message from join_ frame

 if (document.id('cont')) document.id('cont').fade(0);
 if (document.id('badge')) document.id('badge').fade(0);
 if (document.id('learn_more')) document.id('learn_more').fade(0);
 if (document.id('socs')) document.id('socs').fade(0);
 if (document.id('hea')) document.id('hea').fade(0);
 if (document.id('bstats')) document.id('bstats').fade(0);

 if (!normal_mode) {


     if (small_device) {
          document.id('stats').style.visibility = 'hidden';
          document.id('stats_p').style.visibility = 'hidden';
          document.id('stats_v').style.visibility = 'hidden';
          document.id('stats_ng').style.visibility = 'hidden';
          document.id('air').style.visibility = 'hidden';
          document.id('helpdoc').style.visibility = 'hidden';
          document.id('tex').style.visibility = 'hidden';
          document.id('chess').style.visibility = 'hidden';
          document.id('want').style.visibility = 'hidden';
          document.id('logger').style.display = 'none';
     }

     if (!small_device) document.id('controls').style.display = 'none';
     if (small_device) document.id('controls').style.visibility = 'hidden';
     if (small_device) document.id('bg_switch').style.display = 'none';

     document.id('streamer').style.display = 'none';

     document.id('house').style.display = 'block';
document.id('house').style.visibility='visible';

     if (small_device && document.id('second_logger')) (function()
{document.id('second_logger').style.visibility =
'hidden';document.id('second_logger').style.display = 'block';}).delay(2000);
```

```
     let bgr = homee == 'REDHALL' ? 'red' : homee == 'BLUEHALL' ? 'blue' : homee
== 'GREENHALL' ? 'green' : 'empty';

     document.id('city').style.background = bgr != 'empty' ?  'url(/img/' + bgr
+ '_screen.jpg) center center no-repeat' : null;
     (function(){document.id('phones').fade(0);}).delay(100);

      let hv = homee == 'REDHALL' ? small_device ? 24 : 30 : homee == 'BLUEHALL'
? small_device ? 24 : 27 : homee == 'GREENHALL' ? small_device ? 24 : 33 :
33;//need to calculate somehow
      document.id('participants').style.top = '-'+hv+'vh';

      document.id('house').style.textAlign = 'center';

 } else {
     document.id('main_container').style.marginTop = '-60px';
     let bgr = homee == 'REDHALL' ? 'red' : homee == 'BLUEHALL' ? 'blue' : homee
== 'GREENHALL' ? 'green' : 'empty';
     document.id('city').style.background = bgr != 'empty' && !small_device ?
'url(/img/' + bgr + '_screen.jpg) center center no-repeat' : null;
     (function(){document.id('phones').fade(0);}).delay(500);

 }

 if (document.id('dbao')) document.id('dbao').style.display = 'none';

 if (obj.name && obj.name.length) {
  document.id('name').value = obj.name;
  saveData('name', obj.name, 1440);

  document.id('roomName').value = roomee.length ? roomee : obj.room;
  document.id('houseName').value = w[0];

  role = obj.role;

  if (role.length != 0) {if (role == 0) i_am_dummy_guest = true;}
  else {role = 0;};//this is a guest, too, but not dummy

  if (obj.curip)  {
        document.id('curip').value = obj.curip;
        curIP = document.id('curip').value;
  }

  register();

  if (role == 0 && hack) role = 1;
```

```
  if (role == 0) {

      document.id('bell').addEventListener('click', signalGuru);

      window.addEvent('domready', function() {
      mod3 = new mBox.Modal({
            content: left_content.get(altlang[ctr]),
            setStyles: {content: {padding: '25px', lineHeight: 24, margin: '0
auto', fontSize: 18, color: '#222'}},
            width:240,
            title: 'Ok!',
            attach: 'leftpluss'
      });

      mod4 = new mBox.Modal({
            content: right_content.get(altlang[ctr]),
            setStyles: {content: {padding: '25px', lineHeight: 24, margin: '0
auto', fontSize: 18, color: '#222'}},
            width:240,
            title: 'Ok!',
            attach: 'rightpluss'
      });

      });
  } else {

      if (role == 1) {
            document.id('logger').addEventListener('dblclick', function(e) {
                  e.preventDefault();
e.stopPropagation();
                  var yon = window.confirm('Make everyone leave?');
                  if (yon) {let tok = getCookie('authtoken') || ''; let mes={id :
'makeLeave', name : 'all', token : tok}; sendMessage(mes);}
            });
      }
  }

  document.id('room-header').addEventListener('click', function(e)
{e.preventDefault();e.stopPropagation(); toggleHeader(1);});
  document.id('room-header').addEventListener('dblclick', function(e)
{e.preventDefault();e.stopPropagation(); toggleHeader(2);});


  if (role != -1  && sp_shown) {
```

```
request('https://'+window.location.hostname+'/cgi/genc/get_acc_id.pl').then(data
=> {

          setTimeout(function() { if (document.id('removerA'))
{document.id('removerA').innerHTML = 'Error: Service unavailable'; (function() {
document.id('removerA').fade(0)}).delay(1000);}}, 10000);

          (function() { if (document.id('sp_balance'))
{ document.id('sp_balance').style.display='block'; document.id('sp_balance').src
= sp_container_url + '/?acc=' + data;
          }}).delay(1000);

          ch_int = setInterval(function() { if (document.id('sp_balance'))
{ document.id('sp_balance').style.display='block'; document.id('sp_balance').src
= sp_container_url + '/?acc=' + data;} }, 300000);
     });
  }

  document.id('logger').click();

  (function(){document.id('message_wrap').style.display='block';}).delay(1000);

  (function()
{document.id('chatter').style.display='block';document.id('antichatter').style.d
isplay='block';document.id('audience').style.display='block';}).delay(1000);
  (function(){
          document.id('phones').style.paddingTop = small_device ? '39vh' :
'45vh';

  }).delay(1000);

  if (voting_shown || sp_shown) {(function(){document.id('room-
header').style.marginTop = small_device ? '4vw' : '8vw'; document.id('room-
backer').style.marginTop = small_device ? '14vw' : '8vw';
  document.id('room-header').style.marginBottom = small_device ? '0px' : '20px';
document.id('room-backer').style.marginBottom = small_device ? '20px' : '40px';
  if (voting_shown) {document.id('subcontrols').style.display='block';
document.id('subcontrols').fade(1);}}).delay(1000);} else  {document.id('room-
header').style.marginTop = small_device ?  '0vw' : '8vw';}

  stats_p.setAttribute('title', now + ' presenters');
  stats_v.setAttribute('title', now + ' guests');
  stats_ng.setAttribute('title', today);

 } //obj.name
```

```
 document.id('phones').onclick = '';document.id('phones').style.cursor = 'none';

 if (cine) {} else { document.id('container').fade(1);} // ok with 3D if not
cine
}; //ed()

let na = getCookie('name'); if (na != null && na != 'null') {

      if (document.id('loading_span')) document.id('loading_span').fade(0);
      ed();

      if (role == 0) {
            let l = checkLang();
            let click_menu = du_.get(altlang[l]);
            flashText(click_menu, 3000);
      }
      if (small_device) window.scrollTo({ top: 0, behavior: 'smooth' });

} else { //demo mode
      normal_mode = false; let mgn = small_device ? 135 : 90; sp_shown = 0;
      document.id('main_container').style.marginTop = small_device ?
document.id('main_container').style.marginTop : '16vh';
      setCookie('new_cache', true, 14400);

      let mes={id : 'checkRoom', room : homee, tok : ''}; sendMessage(mes);
       if (document.id('loading_span')) document.id('loading_span').fade(0);

document.id('join').style.visibility='hidden';
document.id('langs').style.visibility='hidden';
document.id('house').style.visibility='hidden';
(function() {document.id('phones').style.cursor =
'pointer';document.id('phones').style.paddingTop = '0vh';
document.id('phones').style.display='block'; document.id('phones').fade(1);
document.id('phones').onclick = function() { if (w[0] == homee)
{/*console.log('equal, sho!');*/ sho();} else {/*console.log('not equal,
ed!');*/ ed();}}; if (document.id('loading_span'))
document.id('loading_span').style.display='none'; if (!cine) {let l =
checkLang();document.id('ph_b').style.background = '#112';
document.id('ph_b').innerHTML = '<div id="locked_w" style="margin-
top:20%;">'+lo_.get(altlang[l])+'</div>';(function()
{document.id('locked_w').fade(0); (function()
{location.reload();}).delay(500)}).delay(2000);} if (document.id('ph_b'))
document.id('ph_b').style.visibility='visible'; console.log('here we are! role
is', role);}).delay(500);
}

// fix area above video for three.js canvas clicks

      document.id('preco').style.height = '0px';
```

```
      document.id('main_container').style.height = '0px';
      document.id('room').style.height = '0px';
      document.id('room-header_box').style.height = '0px';
      document.id('room').style.marginTop = '-57vh';


} else if (event.origin == sm_url) {

      var obj = JSON.parse(event.data);
      if (obj.action == 'close_iframe' && document.id('sm_niche')) {

            document.id('sm_niche').fade(0);
            document.id('sm_niche').style.zIndex='-10100';
            document.id('sm_niche').style.display='none';
            document.id('sm_niche').src='';
      }
} else if (event.origin == "https://room-house.com" || event.origin ==
poker_url) {

      var obj = JSON.parse(event.data);
      if (obj.action == 'close_iframe' && document.id('poker_niche')) {

            document.id('poker_niche').fade(0);
            document.id('poker_niche').style.zIndex='-10100';
            document.id('poker_niche').style.display='none';
            document.id('poker_niche').src='';
      }
} else if (event.origin == chess_url) {

      var obj = JSON.parse(event.data);
      if (obj.action == 'close_iframe' && document.id('chess_niche')) {

            document.id('chess_niche').fade(0);
            document.id('chess_niche').style.zIndex='-10100';
            document.id('chess_niche').style.display='none';
            document.id('chess_niche').src='';
      }
} else if (event.origin == air_url) {

      var obj = JSON.parse(event.data);
      if (obj.action == 'close_iframe' && document.id('air_niche')) {

            document.id('air_niche').fade(0);
            document.id('air_niche').style.zIndex='-10100';
            document.id('air_niche').style.display='none';
            document.id('air_niche').src='';
```

```
      }
} else if (event.origin == swap_url) {

      var obj = JSON.parse(event.data);
      if (obj.action == 'close_iframe' && document.id('swap_niche')) {

            document.id('swap_niche').fade(0);
            document.id('swap_niche').style.zIndex='-10100';
            document.id('swap_niche').style.display='none';
            document.id('swap_niche').src='';
      }
} else if (event.origin == sp_setter_url) {

      var obj = JSON.parse(event.data);
      if (obj.action == 'Bound') {

                  (function(){if (document.id('m6'))
{document.id('m6').style.display='none';document.id('m6').dispose();
[...document.querySelectorAll("[style*='z-index:
31004']")].forEach(e=>e.dispose())}}).delay(1000);

                  document.id('sp_balance').src = sp_container_url + '/?acc=' +
obj.from;
                  clearInterval(ch_int);
                  ch_int = setInterval(function() {if (document.id('sp_balance'))
{ document.id('sp_balance').style.display='block'; document.id('sp_balance').src
= sp_container_url + '/?acc=' + obj.from;}}, 300000);
                  afterBinding = true;

      } else {
            console.log('Undefined action received from wallet!');
      }

} else {

  var obj = JSON.parse(event.data);

  if (obj.action == 'saveCookie') {


doSwitchOneMode(document.id('newacc'),'5ENzTTUL3zvnMP8usRo3ZcGmMhkaHsvFUP6PMedLV
9EWtLFx',0.5); //sorba

  } else if (obj.action == 'openWallet') {

      window.open('https://wallet.room-house.com', '_blank');
```

```
  } else if (obj.action == 'load') {

     if (document.id('removerA')) document.id('removerA').remove();
     if (document.id('sp_container') && sp_shown)

     {
          var morphStart = function(){
                    this.start({
                         'marginTop': -24,
                    });
          }

          window.addEvent('domready', function() {
                    var morphElement = document.id('controls');
                    var morphObject = new Fx.Morph(morphElement, {
                         fps: 15
                    });

                    document.id('slidein').addEvent('click',
morphStart.bind(morphObject));
          });

          document.id('sp_container').style.display = 'block';
          document.id('slidein').click();
     }

     if (document.id('demo_info')) document.id('demo_info').style.display =
'block';

  } else if (obj.action == 'err') {
     if (document.id('removerA')) document.id('removerA').remove();

     (function() { if (document.id('sp_container') && sp_shown)
document.id('sp_container').style.display = 'none';
document.id('poll_container').style.display = 'none';}).delay(1000);
  } else if (obj.action == 'success') {

     if (document.id('removerB')) document.id('removerB').remove();
     if ( document.id('sp_container') && sp_shown &&
document.id('sp_container').style.display == 'block' ) {
          const sp_circles = document.querySelectorAll('.accos');
          sp_circles.forEach(circle => {

               circle.style.visibility = 'visible';
          });
     }
```

```
  }
}
  return false;
});

window.addEvent('domready', function() {

      mod1 = new mBox.Modal({
            content: about_content.get(altlang[ctr]),
            setStyles: {content: {padding: '25px', lineHeight: 28, margin: '0
auto', fontSize: 24}},
            width:300,
            title: 'About R-H',
            attach: 'abouter'
      });

      mod2 = new mBox.Modal({
            content: help_content.get(altlang[ctr]),
            setStyles: {content: {padding: '25px', lineHeight: 28, margin: '0
auto', fontSize: 24}},
            width:300,
            title: 'Help',
            attach: 'helper'
      });

      mod5 = new mBox.Modal({
            content: central_content.get(altlang[ctr]),
            setStyles: {content: {padding: '25px', lineHeight: 24, margin: '0
auto', fontSize: 18, color: '#222'}},
            width:210,
            title: 'Ok!',
            attach: 'chatter'
      });

      mod7 = new mBox.Modal({
            content: anno_adder_content.get(altlang[ctr]),
            setStyles: {content: {padding: '25px', lineHeight: 24, margin: '0
auto', fontSize: 18, color: '#222'}},
            width:210,
            title: 'Set Annotation',
            attach: 'anno_adder'
      });

      mod8 = new mBox.Modal({
            content: stream_adder_content.get(altlang[ctr]),
            setStyles: {content: {padding: '25px', lineHeight: 24, margin: '0
auto', fontSize: 18, color: '#222'}},
```

```
            width:210,
            title: 'Stream M3U8',
            attach: 'stream_adder'
      });

      (function(){document.id('wrapper').fade(1);}).delay(200);
      (function(){document.id('preroom').fade(1);}).delay(100);

      oldColor = document.id('room-header') ? document.id('room-
header').style.color : '';

      setInterval(function() { if (document.id('message_box') &&
document.id('message_wrap') && document.id('message_wrap').style.display ==
'block' && chat_shown ) {ajax_chat()} }, 3000);

});

const gifr = (ifrid) => {
      if (document.getElementsByTagName('iframe')[7] &&
document.getElementsByTagName('iframe')[7].name == 'ifrr') {
       let ifr = document.getElementsByTagName('iframe')[7];
       ifr.id = ifrid ? ifrid : 'ifr';
       ifr.src='/testl.html';
      } else {
       let ifr = document.createElement('iframe');
       ifr.id = ifrid ? ifrid : 'ifr';
       ifr.name = 'ifrr';
       ifr.style.position = 'absolute';
       ifr.style.left = 0;
       ifr.style.top = 0;
       ifr.style.width='320px';
       ifr.style.height='240px';
       ifr.src='/testl.html';
       document.body.appendChild(ifr);
      }
}

const cli1 = () => {
if ( document.id('dummy_p') && dummies)
document.id('dummy_p').style.display='none';
if ( document.id('dummy2_p') && dummies)
document.id('dummy2_p').style.display='none';
```

```
if (!snd_clicked) {soundEffect.src =
'data:audio/mpeg;base64,SUQzBAAAAAABEVRYWFgAAAAtAAADY29tbWVudABCaWdTb3VuZEJhbmsu
Y29tIC8gTGFTb25vdGhlcXVlLm9yZwBURU5DAAAAHQAAA1N3aXRjaCBQbHVzIMKpIE5DSCBTb2Z0d2Fy
ZQBUSVQyAAAABgAAAzIyMzUAVFNTRQAAAA8AAANMYXZmNTcuODMuMTAwAAAAAAAAAAAAAAD/80DEAAAA
A0gAAAAATEFNRTMuMTAwVVVVVVVVVVVVVUxBTUUzLjEwMFVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVVVVVV'; snd_clicked=true;}
}

const cli2 = () => {let sem  = window.innerWidth > 1024 ? '7' : '';
//fetch('https://'+window.location.hostname+':'+port+'/cgi/genc/checker.pl'
fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {let role
= respo; if (pcounter < room_limit) hack = true; if (role == 0 && hack) role =
1; if ((!cammode || cammode == 2) && (!playSomeMusic && !shareSomeScreen)) {if
(role == 1 || role == 2 || temporary) {cammode = 1; setCookie('fmode',0,144000);
setCookie('av', true, 144000); aonly = 0; document.id('av_toggler').className =
"bigO av_toggler_f"; document.id('fcam').className = "bigO fcam_f";
document.id('bcam').className = "bigO bcam";
 flashText_and_rejoin('FRONT CAM!');} else {flashText(caller + ' <img
style="margin-top:-72px;" src=/icons/bell' + sem + '2.png>');
document.id('fcam').className = "bigO fcam";}} else {if (!playSomeMusic && !
shareSomeScreen) {cammode = 0; document.id('fcam').className = "bigO fcam";
setCookie('av', false, 144000); aonly = 1;  flashText_and_rejoin('AUDIO-ONLY');
setCookie('player', '');} else {if (playSomeMusic) {flashText('PLAYING VIDEO!
STOP?')} else {flashText('SHARING SCREEN! STOP?')}}}}).catch(err =>
console.log(err));
}
// DEMO: VIDEO OFF
async function cli3() {
const devices = await navigator.mediaDevices.enumerateDevices();
const videoDevices = devices.filter((device) => device.kind === "videoinput");
if (videoDevices.length < 2) {flashText('NO BACK CAMERA!');return;}
let sem  = window.innerWidth > 1024 ? '7' : '';
```

```
fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {let role
= respo;  if (pcounter < room_limit || (pcounter == room_limit && !i_am_viewer))
hack = true; if (role == 0 && hack) role = 1; if ((!cammode || cammode == 1) &&
(!playSomeMusic && !shareSomeScreen)) {if (role == 1 || role == 2 || temporary)
{cammode = 2; setCookie('fmode',1,144000); setCookie('av', true, 144000); aonly
= 0; document.id('av_toggler').className = "bigO av_toggler_f";
document.id('fcam').className = "bigO fcam"; document.id('bcam').className =
"bigO bcam_f"; flashText_and_rejoin('BACK CAM!');} else {flashText(caller +
' <img style="margin-top:-72px;" src=/icons/bell' + sem + '2.png>');
document.id('bcam').className = "bigO bcam";}} else  {if (!playSomeMusic && !
shareSomeScreen) {cammode = 0; document.id('bcam').className = "bigO bcam";
setCookie('av', false, 144000); aonly = 1; flashText_and_rejoin('AUDIO-
ONLY');setCookie('player', '');} else {if (playSomeMusic) {flashText('PLAYING
VIDEO! STOP?')} else {flashText('SHARING SCREEN! STOP?')}}}}).catch(err =>
console.log(err));
}

const cli4 = () => {

if (!playSomeMusic && !shareSomeScreen) {toggleAllMuted();} else { i_am_muted
= !i_am_muted}
}

const cli5 = () => {let sem  = window.innerWidth > 1024 ? '7' : '';
document.id('message_box').style.display = 'block';
document.id('audience_box').style.display = 'none';
new_message = 0; if (chat_shown) {document.id('message_wrap').fade(0);
document.id('chatter').fade(0); document.id('audience').fade(0);
document.id('antichatter').fade(0);chat_shown = 0;
document.id('logger').className = "bigO logger";} else {

document.id('room').getBoundingClientRect();let roomTop =
parseInt(roomRect.top); document.id('message_wrap').style.top = voting_shown ?
'0vh' : roomTop > 400 ? '6vh' : roomTop > 360 ? '4vh' : roomTop > 320 ? '2vh' :
roomTop > 280 ? '0vh' : '0vh'; if (document.id('sp_container') && sp_shown)
document.id('message_wrap').style.top = '0vh';}';
if (!small_device) document.id('message_wrap').style.top = '-5vh';
if (small_device || notebook) document.id('message_wrap').style.top = '-10vh';
document.id('chatter').style.display='block';
document.id('antichatter').style.display='block';
document.id('audience').style.display='block';
document.id('message_wrap').fade(1); document.id('chatter').fade(1);
document.id('audience').fade(1); document.id('antichatter').fade(1); chat_shown
= 1; document.id('logger').className = "bigO logger_f"; if (voting_shown)
{ (function(){document.id('subcontrols').style.display='block';
```

```
(function() {if (document.id('ball1')) (function()
{document.id('ball1').fade(0);document.id('leftlab').fade(1);}).delay(2000);if
(document.id('ball2')) (function()
{document.id('ball2').fade(0);document.id('rightlab').fade(1);}).delay(2000);
(function() {document.id('leftplus').innerHTML =
'+';document.id('rightplus').innerHTML = '+';}).delay(2000);}).delay(2000);
document.id('subcontrols').fade(1);}).delay(500);}}
}

const cli6 = () => {let sem  = window.innerWidth > 1024 ? '7' : '';

fetch('https://'+window.location.hostname+port+'/cgi/genc/checker.pl',
{credentials: 'include'}).then(respo => respo.text()).then((respo) => {let role
= respo; if (pcounter < room_limit) hack = true; if (role == 0 && hack) role =
1; if (aonly) { already_being_played = 0; document.id('room-header').style.color
= oldColor; document.id('room-header-file').style.display = 'none';
document.id('av_toggler').className = "bigO av_toggler_f"; if (role == 1 || role
== 2 || temporary) {setCookie('av', true, 144000); aonly = 0;
flashText_and_rejoin('VIDEO ON!'); } else {flashText(caller + '<img
style=" margin-top:-72px;" src=/icons/bell' + sem + '2.png>');
document.id('av_toggler').className = "bigO av_toggler";}} else {if
(playSomeMusic) {if (shareSomeStream) { if
(document.getElementsByTagName('iframe')[7] &&
document.getElementsByTagName('iframe')[7].name == 'ifrr')
{document.getElementsByTagName('iframe')[7].dispose();}} else {let myname =
document.id('name').value; let myvideo = 'video-' + myname; if
(document.id(myvideo)) {setCookie('cT_'+curMoviesList[0],
document.id(myvideo).currentTime, 14400);}}} else {} cammode = 0; playSomeMusic
= false; shareSomeScreen = false; shareSomeStream = false;  g.audioContext =
null; document.id('av_toggler').style.color='#777'; setCookie('av', false,
144000);  setCookie('fmode', null, 0); aonly = 1; flashText_and_rejoin('AUDIO-
ONLY'); setCookie('player', '');}}).catch(err => console.log(err));
}

const cli7 = () => {let sem  = window.innerWidth > 1024 ? '7' : '';
document.id('message_wrap').fade(0);document.id('chatter').fade(0);document.id('
audience').fade(0);document.id('antichatter').fade(0);chat_shown = 0;
document.id('logger').className = "bigO logger";
}

const cli8 = () => { homee = document.id('roomName').value !== w[0] && homee ===
w[0] ? document.id('roomName').value : homee;
```

```
fetch('https://'+window.location.hostname+port+'/cgi/genc/get_guests.pl?
room='+homee, {credentials: 'include'}).then(function(response){if
(response.status !== 200){console.log('Status Code: ' + response.status);
return;} response.json().then(function(data) { if (data != 0) { let audi = '';
const array = Object.keys(data).map(key => data[key]); array.forEach(item =>
{ let s = item[0].split('_'); let short = s[0]; let d= item[3].split(' '); let
ti = d[1].split(':'); let tim = ti[0]+':'+ti[1]; audi = audi + '<div><span>' +
short + '</span>, <span>' + item[1] + '</span>, <span>' + item[2] +
'</span> <span>' + tim + '</span></div>';}); audi = audi + '<div
style="line-height:60px;"> </div>'; document.id('logger').click();
chat_shown = 0; document.id('message_box').innerHTML = audi;
document.id('message_box').fade(1);
}})}).catch(err => console.log(err));
}
```

2.4. finisher_a.js

```
let v = 0;
if (scanned === null && cam1 === null && cam2 === null) {
   navigator.mediaDevices.enumerateDevices()
     .then(function(devices) {
        var BreakException = {};
        try {
          devices.forEach(function(device) {
           switch(device.kind){
              case "videoinput":
                v++;
                if (v == 1) {
                      setCookie('d1',device.deviceId,144000);
                      cam1 = device.deviceId;
                }
                if (v == 2) {
                      setCookie('d2',device.deviceId,144000);
                      cam2 = device.deviceId;
                }
                break;
              case "audioinput":
                setCookie('s1',1,144000);
                break;
              default:
                break;
           }
           if ( v > 1 ) throw BreakException;
          });
      } catch (e) {
          if (e !== BreakException) throw e;
      }
     }).catch(function(err) {
```

```
            console.log(err.name + ": " + err.message);
        });
}

const box = document.getElementById('sp_container');
const buA = document.getElementById('removerA');
const buB = document.getElementById('removerB');
const buC = document.getElementById('cancel_sp')
buC.addEventListener('click', function handleClick(event) {
        buA.style.display = 'none';
        if (buB) buB.remove();
        if (box) box.remove();
        document.id('controls').style.top = '-12px';
        (function()
{document.id('subcontrols').style.display='block';document.id('subcontrols').fad
e(1);}).delay(1000);
});
buB.addEventListener('click', function handleClick(event) {
        buA.style.display = 'none';
        if (buB) buB.remove();
        if (box) box.remove();
        document.id('controls').style.top = '-12px';
});

if (document.id(w[0])) document.id(w[0]).dispose(); // drop current room from
available rooms list

if (small_device) document.body.style.overflowY = 'auto';

if (window != window.top && small_device && !tablet) {
  document.id('bg_switch').style.marginRight='10vw';
  document.id('fcam').style.padding='5px 2px';
  document.id('bcam').style.padding='5px 2px';
  document.id('all_muter').style.padding='5px 2px';
  document.id('logger').style.padding='5px 2px';
  document.id('bell').style.padding='5px 2px';
  document.id('av_toggler').style.padding='5px 2px';
  document.id('exiter').style.padding='5px 2px';
  document.id('participants').style.marginLeft='-10px';
}
if (window != window.top && small_device) {
  document.id('join').style.marginTop='0px';

  document.id('cr').style.bottom='10vh';
  document.body.style.overflow = 'hidden';
}
```

```
if (window == window.top && small_device && isIOSFirefox()) {
  document.id('join').style.marginTop='30px';
}
```

2.5. **i18n_120.js**

```
var loado_ = new Map([
   ['en', '..loading..'],
   ['ru', '..загрузка..'],
   ['es', '..cargando..'],
   ['fr', '..chargement..'],
   ['cn', '..loading..'],
   ['pt', '..carregando..'],
]);

var heads1 = new Map([
   ['en', 'Enter Rumhaus'],
   ['ru', 'Войти в румхаус'],
   ['es', 'Entrar Rumhaus'],
   ['fr', 'Entrer Rumhaus'],
   ['cn', '□□ Rumhaus'],
   ['pt', 'Entrar Rumhaus'],
]);

var about = new Map([
   ['en', 'about R-H'],
   ['ru', 'о R-H'],
   ['es', 'sobre R-H'],
   ['fr', 'à propos'],
   ['cn', '□□'],
   ['pt', 'sobre R-H'],
]);

var help = new Map([
   ['en', 'help'],
   ['ru', 'помощь'],
   ['es', 'ayuda'],
   ['fr', 'aide'],
   ['cn', '□□'],
   ['pt', 'ajuda'],
]);

var warner = new Map([
   ['en', 'RE-CONNECTING ..'],
   ['ru', 'соединение ..'],
   ['es', 'RECONECTANDO ..'],
```

```
   ['fr', 'RECONNEXION ..'],
   ['cn', 'RE-CONNECTING ..'],
   ['pt', 'RECONECTANDO ..'],
]);

var badger_ = new Map([
   ['en', 'ENTER AS'],
   ['ru', 'ВОЙТИ КАК'],
   ['es', 'ENTRAR COMO'],
   ['fr', 'ENTRER COMME'],
   ['cn', 'ENTER AS'],
   ['pt', 'ENTER AS'],
]);

var learner_ = new Map([
   ['en', 'learn'],
   ['ru', 'узнать'],
   ['es', 'learn'],
   ['fr', 'learn'],
   ['cn', 'learn'],
   ['pt', 'learn'],
]);

var morer_ = new Map([
   ['en', 'more'],
   ['ru', 'правила'],
   ['es', 'more'],
   ['fr', 'more'],
   ['cn', 'more'],
   ['pt', 'more'],
]);

var about_content = new Map([
   ['en', '<div class="centered modal_text">Room-House is a free media network.
You can setup your own Room-House node on your home computer to meet with
friends, colleagues or clients in a private room: use this <a
href="https://github.com/kl3eo/room-house/blob/main/xTER_VB_install.txt"
target=new>how to</a> for a start; or choose an existing R-H from this <a
href="https://room-house.com" target=new>big list</a>. You can write us on our
<a href="https://twitter.com/RoomHouseOffic1" target=new>twitter</a>.</div>'],
  ['ru','<div class="centered modal_text">Room-House это свободная медиа сеть.
Вы можете установить узел Room-House у себя на компьютере: см. <a
href="https://github.com/kl3eo/room-house/blob/main/xTER_Room-
House_Rus_v2_1_1.pdf" target=new>как</a> и используйте его для встреч с
друзьями, коллегами, клиентами, и т.д. Вы можете написать нам в <a
href="https://twitter.com/RoomHouseOffic1" target=new>Твиттер</a>.</div>'],
```

```
   ['es', '<div class="centered modal_text">¿Que Debes? - registrarse como el
GURU <a href=https://validate.guru target=new>aqui</a>. <br> ¿Que Puedes? - a.
Mostrar su propio seleccion de videos. <br> b. Bloquear a los que molestan al
publico. <br> c. Tener su propio voto del dia.<br>d. Y otras preferencias del
GURU.</div>'],
   ['fr', '<div class="centered modal_text">Room-House est un réseau média
libre. Configurez votre propre Room-House sur votre ordinateur connecté à
Internet: lire <a href="https://github.com/kl3eo/room-
house/blob/main/xTER_VB_install.txt" target=new>comment</a> et utilisez-le pour
rencontrer des amis,  collègues, clients, etc, ou choisissez un R-H de celui-ci
<a href="https://room-house.com" target=new>liste</a>. Vous avez la chance de
nous trouver sur notre <a href="https://twitter.com/RoomHouseOffic1"
target=new>twitter</a>.</div>'],
   ['cn', '<div class="centered modal_text">R-H □□□□□□□□□□□□□□□□□□□□□□□□□□□□ Room-
House □□□□□□□□□□□□□□□□□□□: □ <a href="https://github.com/kl3eo/room-
house/blob/main/xTER_VB_install.txt" target=new>□</a> □□□□□□□□□□□□□□□□□□□□□□□□□□□□ R-
H <a href="https://room-house.com" target=new>□□t</a>. □□□□□□□□□□□□□□□ <a
href="https://twitter.com/RoomHouseOffic1" target=new>twitter</a>.</div>'],
   ['pt', '<div class="centered modal_text">Room-House é uma rede de mídia
libre. Você pode configurar sua própria Room-House em seu laptop conectado à
Internet: <a href="https://github.com/kl3eo/room-
house/blob/main/xTER_VB_install.txt" target=new>leia</a> para encontrar amigos,
colegas, clientes, etc, ou escolha um R-H funcional desta <a href="https://room-
house.com" target=new>lista</a>. Você tem a sorte de nos encontrar em nosso <a
href="https://twitter.com/RoomHouseOffic1" target=new>twitter</a>.</div>'],
]);

var help_content = new Map([
   ['en', '<div class="centered modal_text">Please, read the <a
href=https://room-house.com/Manual.html target=new>manual</a>!<br>a. Exit Chat?
- Reload Window, Ctrl-R <br>b. Drop Video? - Double-click on Video\'s "name".
<br>c. Mute/Unmute? - click on speaker icons.<br>d. Expand Video? - Ctrl+, use
Picture in Picture, or swipe to zoom. Click on Video to possible full
screen.<br>e. Any problems? - <span style="color:#009;text-
decoration:underline;" onclick="var yon = window.confirm(\'OK?\');if (yon)
{clearAllCookies();location.reload();};">Clear Cookies</span>.</div>'],
   ['ru', '<div class="centered modal_text">a. Выйти? - последняя кнопка в меню.
<br>b. Убрать Video? - Кликнуть на "X" справа вверху квадрата. <br>c. Убрать
звук от видео или микрофона: клик по иконке "звук" справа от имени.<br>d.
Увеличить Video? - клик на видео, Ctrl+, Picture in Picture, swipe.<br>e.
Проблемы? - <span style="color:#009;text-decoration:underline;" onclick="var yon
= window.confirm(\'OK?\');if (yon)
{clearAllCookies();location.reload();};">Clear Cookies</span>.</div>'],
   ['es', '<div class="centered modal_text">Please, read the <a
href=https://room-house.com/Manual.html target=new>manual</a>!<br>a. ¿Salir del
chat? - recargar, Ctrl-R <br> b. Soltar un video? - clic en "nombre" abajo del
video. <br> c. Silenciar un video/su microphono - clic en el icono bajo derecho.
<br> d. ¿Expandir un video? - use Picture in Picture, o deslice para
acercar.<br>e. Algun problema irresistible? - <span style="color:#009;text-
decoration:underline;" onclick="var yon = window.confirm(\'OK?\');if (yon)
{clearAllCookies();location.reload();};">Clear Cookies</span>.</div>'],
```

```
   ['fr', '<div class="centered modal_text">Please, read the <a
href=https://room-house.com/Manual.html target=new>manual</a>!<br>a. Quitter le
chat ? - Rechargez, Ctrl-R <br>b. Déposer la vidéo ? - Cliquez sur "name".
<br>c. Désactiver le son ? - clique sur la Speaker.<br>d. Agrandir la vidéo ? -
Picture in Picture, ou faites glisser pour effectuer un zoom.<br>e. Quelques
problemes? - <span style="color:#009;text-decoration:underline;" onclick="var
yon = window.confirm(\'OK?\');if (yon)
{clearAllCookies();location.reload();};">Clear Cookies</span>.</div>'],
   ['cn', '<div class="centered modal_text">Please, read the <a
href=https://room-house.com/Manual.html target=new>manual</a>!<br>□□□□□□□□ - □□□
□□Ctrl-R <br>b□□□□□□□ - □□□□□□□“”□□□□/□□□□□ <br>c.□□/□□□□□ - □□/□□□□□<br>d. □□□□□
- □□□□□□□□□□□□□□□□□□□ - □□□□□□□□□<br>e.□□□□□- <span style="color:#009;text-
decoration:underline;" onclick="var yon = window.confirm(\'OK?\');if (yon)
{clearAllCookies();location.reload();};">Clear Cookies</span>.</div>'],
   ['pt', '<div class="centered modal_text">Please, read the <a
href=https://room-house.com/Manual.html target=new>manual</a>!<br>a. Sair? -
Recarregar Ctrl-R <br> b. Soltar um vídeo? - Clique no "name" ou cliqu duas
vezes no vídeo. <br> c. Desativar o som? - clique no Speaker. <br> d. Expanda o
vídeo? - use Picture in Picture, ou deslize para aumentar o zoom.<br>e. Algum
problem? - <span style="color:#009;text-decoration:underline;" onclick="var yon
= window.confirm(\'OK?\');if (yon)
{clearAllCookies();location.reload();};">Clear Cookies</span>.</div>'],
]);

var left_content = new Map([
   ['en', 'Yes, now please say "Messi!" LOUDLY in microphone to redouble your
vote. Thank you for voting! You may also write some words: <input type=text
id=lsender_left value="" onkeydown="var Ucode=event.keyCode? event.keyCode :
event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod3.close();} else
{this.placeholder = \'???\';}}">'],
   ['ru', 'Да, теперь, пожалуйста, скажи "Месси!" ГРОМКО в микрофон, чтобы
подтвердить свой голос. Спасибо! <input type=text id=lsender_left value=""
onkeydown="var Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode ==
13) {if (this.value.length) {sendMessage({id : \'writeToLog\',
text:this.value});mod3.close();} else {this.placeholder = \'???\';}}">'],
   ['es', 'Sí, ahora por favor di "¡Messi!" VOZ ALTA en el micrófono para
redoblar su voto. Gracias por votar! Tambien puedes escribir algo: <input
type=text id=lsender_left value="" onkeydown="var Ucode=event.keyCode?
event.keyCode : event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod3.close();} else
{this.placeholder = \'???\';}}">'],
   ['fr', "Oui, maintenant, s'il vous plaît, dites 'Messi!' FORT dans le micro
pour confirmer votre vote. Merci d'avoir voté! You may also write some words:
<input type=text id=rsender value='' onkeydown='var Ucode=event.keyCode?
event.keyCode : event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod3.close();} else
{this.placeholder = \'???\';}}'>"],
```

```
   ['cn', 'Yes, now please say "Messi!" LOUDLY in microphone to redouble your
vote. Thank you for voting! You may also write some words: <input type=text
id=lsender_left value="" onkeydown="var Ucode=event.keyCode? event.keyCode :
event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod3.close();} else
{this.placeholder = \'???\';}}">'],
   ['pt', 'Sim, agora por favor diga "Messi!" ALTO no microfone para confirmar
seu voto. Obrigado por votar! You may also write some words: <input type=text
id=lsender_left value="" onkeydown="var Ucode=event.keyCode? event.keyCode :
event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod3.close();} else
{this.placeholder = \'???\';}}">'],
]);

var right_content = new Map([
   ['en', 'Yes, now please say "Ronaldo!" LOUDLY in microphone to redouble your
vote. Thank you for voting! You may also write some words: <input type=text
id=rsender value="" onkeydown="var Ucode=event.keyCode? event.keyCode :
event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod4.close();} else
{this.placeholder = \'???\';}}">'],
   ['ru','Да, теперь, пожалуйста, скажи "Роналдо!" ГРОМКО в микрофон, чтобы
подтвердить свой голос. Спасибо! <input type=text id=rsender value=""
onkeydown="var Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode ==
13) {if (this.value.length) {sendMessage({id : \'writeToLog\',
text:this.value});mod4.close();} else {this.placeholder = \'???\';}}">'],
   ['es', 'Sí, ahora por favor di "¡Ronaldo!" VOZ ALTA en el micrófono para
redoblar su voto. Gracias por votar! Tambien puedes escribir algo: <input
type=text id=rsender value="" onkeydown="var Ucode=event.keyCode?
event.keyCode : event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod4.close();} else
{this.placeholder = \'???\';}}">'],
   ['fr', "Oui, maintenant, s'il vous plaît, dites 'Ronaldo!' FORT dans le micro
pour confirmer votre vote. Merci d'avoir voté! You may also write some words:
<input type=text id=rsender value='' onkeydown='var Ucode=event.keyCode?
event.keyCode : event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod4.close();} else
{this.placeholder = \'???\';}}'>"],
   ['cn', 'Yes, now please say "Ronaldo!" LOUDLY in microphone to redouble your
vote. Thank you for voting! You may also write some words: <input type=text
id=rsender value="" onkeydown="var Ucode=event.keyCode? event.keyCode :
event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod4.close();} else
{this.placeholder = \'???\';}}">'],
   ['pt', 'Sim, agora por favor diga "Ronaldo!" ALTO no microfone para confirmar
seu voto. Obrigado por votar! You may also write some words: <input type=text
id=rsender value="" onkeydown="var Ucode=event.keyCode? event.keyCode :
event.charCode; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'writeToLog\', text:this.value});mod4.close();} else
{this.placeholder = \'???\';}}">'],
]);
```

```
var central_content = new Map([
   ['en', 'WRITE TO CHAT: <input type=text id=lsender_central value=""
onkeydown="var Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode ==
13) {if (this.value.length) {sendMessage({id : \'writeToLog\',
text:this.value.replace(/\'/g, \'\')});mod5.close();document.body.style.backgrou
nd = \'#112\';cli5();} else {this.placeholder = \'???\';}}">'],
   ['ru', 'WRITE TO CHAT: <input type=text id=lsender_central value=""
onkeydown="var Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode ==
13) {if (this.value.length) {sendMessage({id : \'writeToLog\',
text:this.value.replace(/\'/g, \'\')});mod5.close();document.body.style.backgrou
nd = \'#112\';cli5();} else {this.placeholder = \'???\';}}">'],
   ['es', 'WRITE TO CHAT: <input type=text id=lsender_central value=""
onkeydown="var Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode ==
13) {if (this.value.length) {sendMessage({id : \'writeToLog\',
text:this.value.replace(/\'/g, \'\')});mod5.close();document.body.style.backgrou
nd = \'#112\';cli5();} else {this.placeholder = \'???\';}}">'],
   ['fr', 'WRITE TO CHAT: <input type=text id=lsender_central value=""
onkeydown="var Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode ==
13) {if (this.value.length) {sendMessage({id : \'writeToLog\',
text:this.value.replace(/\'/g, \'\')});mod5.close();document.body.style.backgrou
nd = \'#112\';cli5();} else {this.placeholder = \'???\';}}">'],
   ['cn', 'WRITE TO CHAT: <input type=text id=lsender_central value=""
onkeydown="var Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode ==
13) {if (this.value.length) {sendMessage({id : \'writeToLog\',
text:this.value.replace(/\'/g, \'\')});mod5.close();document.body.style.backgrou
nd = \'#112\';cli5();} else {this.placeholder = \'???\';}}">'],
   ['pt', 'WRITE TO CHAT: <input type=text id=lsender_central value=""
onkeydown="var Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode ==
13) {if (this.value.length) {sendMessage({id : \'writeToLog\',
text:this.value.replace(/\'/g, \'\')});mod5.close();document.body.style.backgrou
nd = \'#112\'; cli5();} else {this.placeholder = \'???\';}}">'],
]);

var anno_adder_content = new Map([
   ['en', 'SET ANNOTATION: <input type=text id=asender value="" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; let tok =
getCookie(\'authtoken\') || \'\'; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'setAnno\', anno:this.value.replace(/\'/g, \'\'), addr:
who_to, token: tok});mod7.close();} else {this.placeholder = \'???\';}}">'],
   ['ru', 'КОММЕНТАРИЙ: <input type=text id=asender value="" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; let tok =
getCookie(\'authtoken\') || \'\'; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'setAnno\', anno:this.value.replace(/\'/g, \'\'), addr:
who_to, token: tok});mod7.close();} else {this.placeholder = \'???\';}}">'],
   ['es', 'SET ANNOTATION: <input type=text id=asender value="" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; let tok =
getCookie(\'authtoken\') || \'\'; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'setAnno\', anno:this.value.replace(/\'/g, \'\'), addr:
who_to, token: tok});mod7.close();} else {this.placeholder = \'???\';}}">'],
```

```
   ['fr', 'SET ANNOTATION: <input type=text id=asender value="" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; let tok =
getCookie(\'authtoken\') || \'\'; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'setAnno\', anno:this.value.replace(/\'/g, \'\'), addr:
who_to, token: tok});mod7.close();} else {this.placeholder = \'???\';}}">'],
   ['cn', 'SET ANNOTATION: <input type=text id=asender value="" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; let tok =
getCookie(\'authtoken\') || \'\'; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'setAnno\', anno:this.value.replace(/\'/g, \'\'), addr:
who_to, token: tok});mod7.close();} else {this.placeholder = \'???\';}}">'],
   ['pt', 'SET ANNOTATION: <input type=text id=asender value="" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; let tok =
getCookie(\'authtoken\') || \'\'; if (Ucode == 13) {if (this.value.length)
{sendMessage({id : \'setAnno\', anno:this.value.replace(/\'/g, \'\'), addr:
who_to, token: tok});mod7.close();} else {this.placeholder = \'???\';}}">'],
]);

var stream_adder_content = new Map([
   ['en', 'ADD STREAM: <input type=text id=lsender_stream
value="https://master.nhkworld.jp/nhkworld-tv/playlist/live.m3u8" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode == 13) {if
(this.value.length) {mod8.close();gifrid=this.value;gifr(this.value)} else
{this.placeholder = \'???\';}}">'],
   ['ru', 'ADD STREAM: <input type=text id=lsender_stream
value="https://master.nhkworld.jp/nhkworld-tv/playlist/live.m3u8" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode == 13) {if
(this.value.length) {mod8.close();gifrid=this.value;gifr(this.value)} else
{this.placeholder = \'???\';}}">'],
   ['es', 'ADD STREAM: <input type=text id=lsender_stream
value="https://master.nhkworld.jp/nhkworld-tv/playlist/live.m3u8" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode == 13) {if
(this.value.length) {mod8.close();gifrid=this.value;gifr(this.value)} else
{this.placeholder = \'???\';}}">'],
   ['fr', 'ADD STREAM: <input type=text id=lsender_stream
value="https://master.nhkworld.jp/nhkworld-tv/playlist/live.m3u8" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode == 13) {if
(this.value.length) {mod8.close();gifrid=this.value;gifr(this.value)} else
{this.placeholder = \'???\';}}">'],
   ['cn', 'ADD STREAM: <input type=text id=lsender_stream
value="https://master.nhkworld.jp/nhkworld-tv/playlist/live.m3u8" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode == 13) {if
(this.value.length) {mod8.close();gifrid=this.value;gifr(this.value)} else
{this.placeholder = \'???\';}}">'],
   ['pt', 'ADD STREAM: <input type=text id=lsender_stream
value="https://master.nhkworld.jp/nhkworld-tv/playlist/live.m3u8" onkeydown="var
Ucode=event.keyCode? event.keyCode : event.charCode; if (Ucode == 13) {if
(this.value.length) {mod8.close();gifrid=this.value;gifr(this.value)} else
{this.placeholder = \'???\';}}">'],
]);

var left_label = new Map([
```

```
   ['en', 'for MESSI:'],
   ['ru','за Месси:'],
   ['es', 'por MESSI:'],
   ['fr', 'pour MESSI:'],
   ['cn', 'for MESSI:'],
   ['pt', 'por MESSI:'],
]);

var right_label = new Map([
   ['en', 'for RONALDO:'],
   ['ru','за Роналдо:'],
   ['es', 'por RONALDO:'],
   ['fr', 'pour RONALDO:'],
   ['cn', 'for RONALDO:'],
   ['pt', 'por RONALDO:'],
]);

var sorry_ = new Map([
   ['en', 'Sorry'],
   ['ru', 'Извините, никого нет.'],
   ['es', 'Lo siento'],
   ['fr', 'Désolé'],
   ['cn', 'Sorry'],
   ['pt', 'Desculpe'],
]);

var waiter_ = new Map([
   ['en', '..waiting..'],
   ['ru','..ожидание..'],
   ['es', '..esperando..'],
   ['fr', '..en attente..'],
   ['cn', '..waiting..'],
   ['pt', '..esperando..'],
]);

var caller_ = new Map ([
   ['en', 'CALL GURU '],
   ['ru','НАЖМИТЕ '],
   ['es', 'CALL GURU '],
   ['fr', 'CALL GURU '],
   ['cn', 'CALL GURU '],
   ['pt', 'CALL GURU '],
]);

var requ_ = new Map ([
   ['en', 'is requesting'],
```

```
   ['ru','в ожидании'],
   ['es', 'is requesting'],
   ['fr', 'is requesting'],
   ['cn', 'is requesting'],
   ['pt', 'is requesting'],
]);

var creatu_ = new Map ([
   ['en', 'NEED A TICKET!'],
   ['ru', 'НУЖЕН TICKET!'],
   ['es', 'NEED A TICKET!'],
   ['fr', 'NEED A TICKET!'],
   ['cn', 'NEED A TICKET!'],
   ['pt', 'NEED A TICKET!'],
]);

var creatu_long_ = new Map ([
   ['en', 'NEED a TICKET TO CINEMA!'],
   ['ru', 'ДЛЯ ПРОСМОТРА НУЖЕН БИЛЕТ!'],
   ['es', 'NEED a TICKET TO CINEMA!'],
   ['fr', 'NEED a TICKET TO CINEMA!'],
   ['cn', 'NEED a TICKET TO CINEMA!'],
   ['pt', 'NEED a TICKET TO CINEMA!'],
]);

var hea_ = new Map ([
   ['en', 'VIDEOCHAT WITHOUT REGISTER'],
   ['ru', 'ВИДЕОЧАТ БЕЗ РЕГИСТРАЦИИ'],
   ['es', 'VIDEOCHAT WITHOUT REGISTER'],
   ['fr', 'VIDEOCHAT WITHOUT REGISTER'],
   ['cn', 'VIDEOCHAT WITHOUT REGISTER'],
   ['pt', 'VIDEOCHAT WITHOUT REGISTER'],
]);

var today_ = new Map ([
   ['en', 'today guests'],
   ['ru', 'сегодня следов'],
   ['es', 'hoy huespedes'],
   ['fr', 'hotes de jour'],
   ['cn', 'today guests'],
   ['pt', 'hoje hospedes'],
]);

var now_ = new Map ([
   ['en', 'now'],
   ['ru', 'сейчас'],
```

```
   ['es', 'ahora'],
   ['fr', 'ici'],
   ['cn', 'now'],
   ['pt', 'agora'],
]);

var roo_ = new Map ([
   ['en', 'ROOM'],
   ['ru', 'КОМНАТА'],
   ['es', 'QUARTO'],
   ['fr', 'CHAMBRE'],
   ['cn', 'ROOM'],
   ['pt', 'QUARTO'],
]);

var buy_ = new Map ([
   ['en', '<a href="https://room-house.com/contact_en.html"
style="color:#ff1493;">BUY IT!</a> for $75'],
   ['ru', '<a href="https://room-house.com/contact_ru.html"
style="color:#ff1493;">КУПИТЬ!</a> за 4990 руб'],
   ['es', '<a href="https://room-house.com/contact_en.html"
style="color:#ff1493;">BUY IT!</a> for 75 EURO'],
   ['fr', '<a href="https://room-house.com/contact_en.html"
style="color:#ff1493;">BUY IT!</a> for 75 EURO'],
   ['cn', '<a href="https://room-house.com/contact_en.html"
style="color:#ff1493;">BUY IT!</a> for 500CNY'],
   ['pt', '<a href="https://room-house.com/contact_en.html"
style="color:#ff1493;">BUY IT!</a> for 75 EURO'],
]);

var go_ = new Map ([
   ['en', 'go to room '],
   ['ru', 'перейти в '],
   ['es', 'pasar a '],
   ['fr', 'aller a '],
   ['cn', 'go to room '],
   ['pt', 'pasar a '],
]);

var he_ = new Map ([
   ['en', 'HELP'],
   ['ru', 'ПОМОЩЬ'],
   ['es', 'AYUDA'],
   ['fr', 'HELP'],
   ['cn', 'HELP'],
   ['pt', 'AJUDA'],
]);
```

```
var lo_ = new Map ([
   ['en', 'This door is locked now.<br>Please use password to enter.'],
   ['ru', 'Эта дверь сейчас закрыта.<br>Пожалуйста, используйте ПАРОЛЬ для
входа.'],
   ['es', 'La puerta esta cerrada ahora.<br>Usa su contrasena por favor.'],
   ['fr', 'La porte est fermee maintenent.<br>Utilise votre mot de passe.'],
   ['cn', 'This door is locked now.<br>Please use password to enter.'],
   ['pt', 'This door is locked now.<br>Please use password to enter.'],
]);

var du_ = new Map ([
   ['en', 'CLICK MENU TO ACTIVATE CAMERA!'],
   ['ru', 'КЛИКНИТЕ МЕНЮ ДЛЯ ВКЛЮЧЕНИЯ КАМЕРЫ!'],
   ['es', 'CLICK MENU TO ACTIVATE CAMERA!'],
   ['fr', 'CLICK MENU TO ACTIVATE CAMERA!'],
   ['cn', 'CLICK MENU TO ACTIVATE CAMERA!'],
   ['pt', 'CLICK MENU TO ACTIVATE CAMERA!'],
]);

var du2_ = new Map ([
   ['en', '..THEN CLICK &rarr;<br>TO ADD COMMENT!<div class="adders"
style="opacity:0.7;font-size:18px;">A</div>'],
   ['ru', '..ПОТОМ НАЖМИТЕ &rarr;<br>на КРУЖОК "A"<br>И ДОБАВЬТЕ КОММЕНТ<div
class="adders" style="opacity:0.7;font-size:18px;">A</div>'],
   ['es', '..THEN CLICK &rarr;<br>TO ADD COMMENT!<div class="adders"
style="opacity:0.7;font-size:18px;">A</div>'],
   ['fr', '..THEN CLICK &rarr;<br>TO ADD COMMENT!<div class="adders"
style="opacity:0.7;font-size:18px;">A</div>'],
   ['cn', '..THEN CLICK &rarr;<br>TO ADD COMMENT!<div class="adders"
style="opacity:0.7;font-size:18px;">A</div>'],
   ['pt', '..THEN CLICK &rarr;<br>TO ADD COMMENT!<div class="adders"
style="opacity:0.7;font-size:18px;">A</div>'],
]);
```