В браузере с помощью Flash Player по RTMP

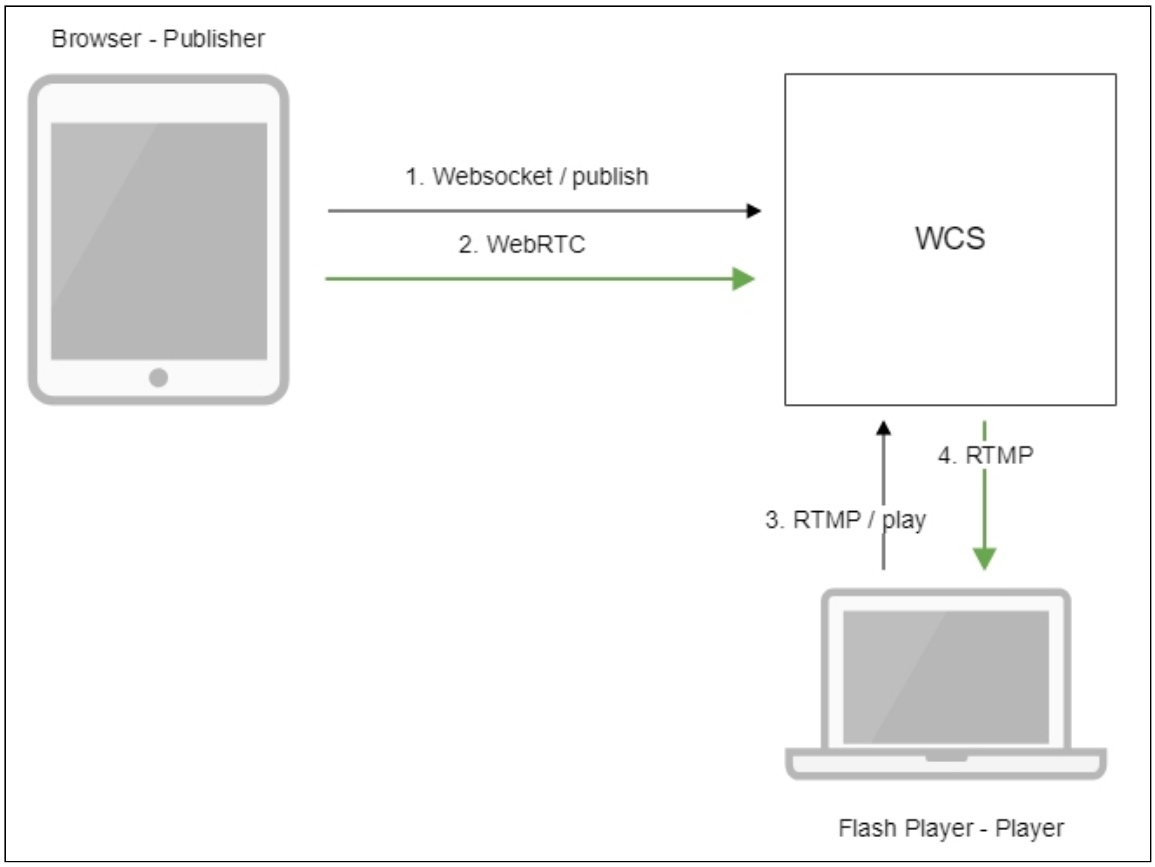
Описание

> **Warning**
>
> Adobe Flash Player не поддерживается в своременных браузерах. Использовать его в настоящее время нельзя. Используйте сторонние плееры для проигрывания RTMP потока с Web Call Server

Поддерживаемые платформы

	Adobe Flash
Windows	✅
Mac OS	✅
Linux	✅

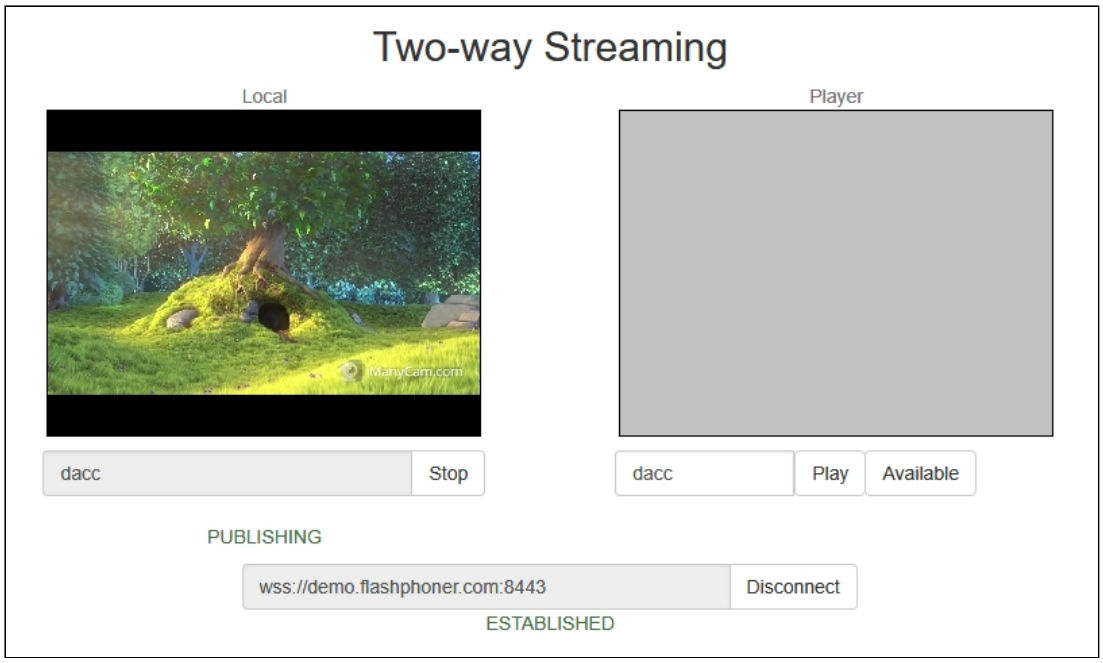
Схема работы



1. Браузер устанавливает соединение по Websocket и отправляет команду `publishStream`.
2. Браузер отправляет WebRTC поток на сервер
3. Flash Player соединяется с сервером по протоколу RTMP и отправляет команду `play`.
4. Flash Player получает RTMP поток с сервера.

Краткое руководство по тестированию

1. Для теста используем:
2. демо-сервер `demo.flashphoner.com`;
3. веб-приложение Two Way Streaming в браузере Chrome для публикации потока
4. веб-приложение Flash Streaming в браузере Internet Explorer для воспроизведения потока
5. Откройте веб-приложение Two Way Streaming. Нажмите `Connect`, затем `Publish`. Скопируйте идентификатор потока:



6. Установите Flash Player. Откройте страницу веб-приложения Flash Streaming и разрешите запуск Flash в браузере:

7. Нажмите кнопку `Login`. При появлении надписи `Connected` укажите в поле `Play` идентификатор транслируемого потока:

Flash Streaming

Server: rtmp://demo.flashphoner.com:1935 Logout

CONNECTED

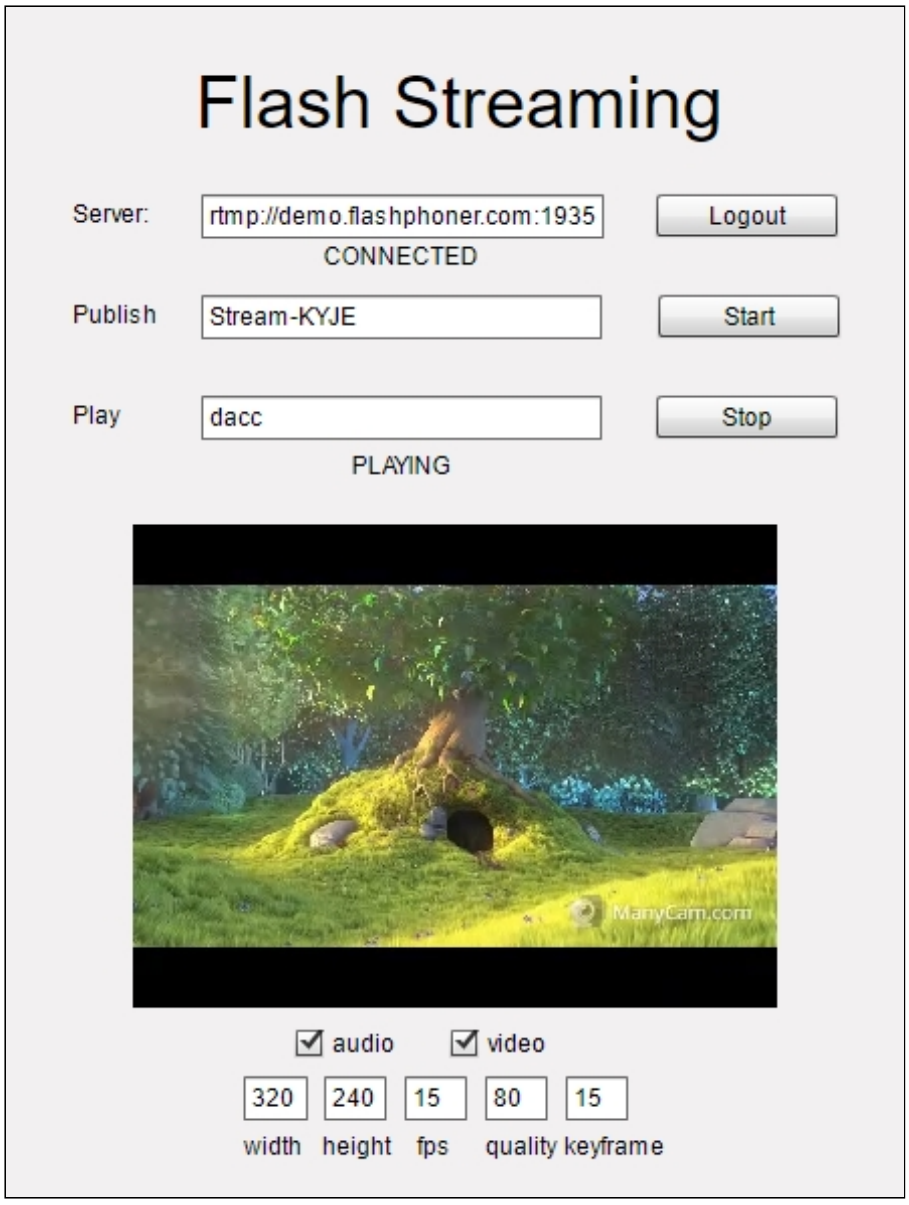
Publish Stream-KYJE Start

Play dacc Start

audio video

320	240	15	80	15
width	height	fps	quality	keyframe

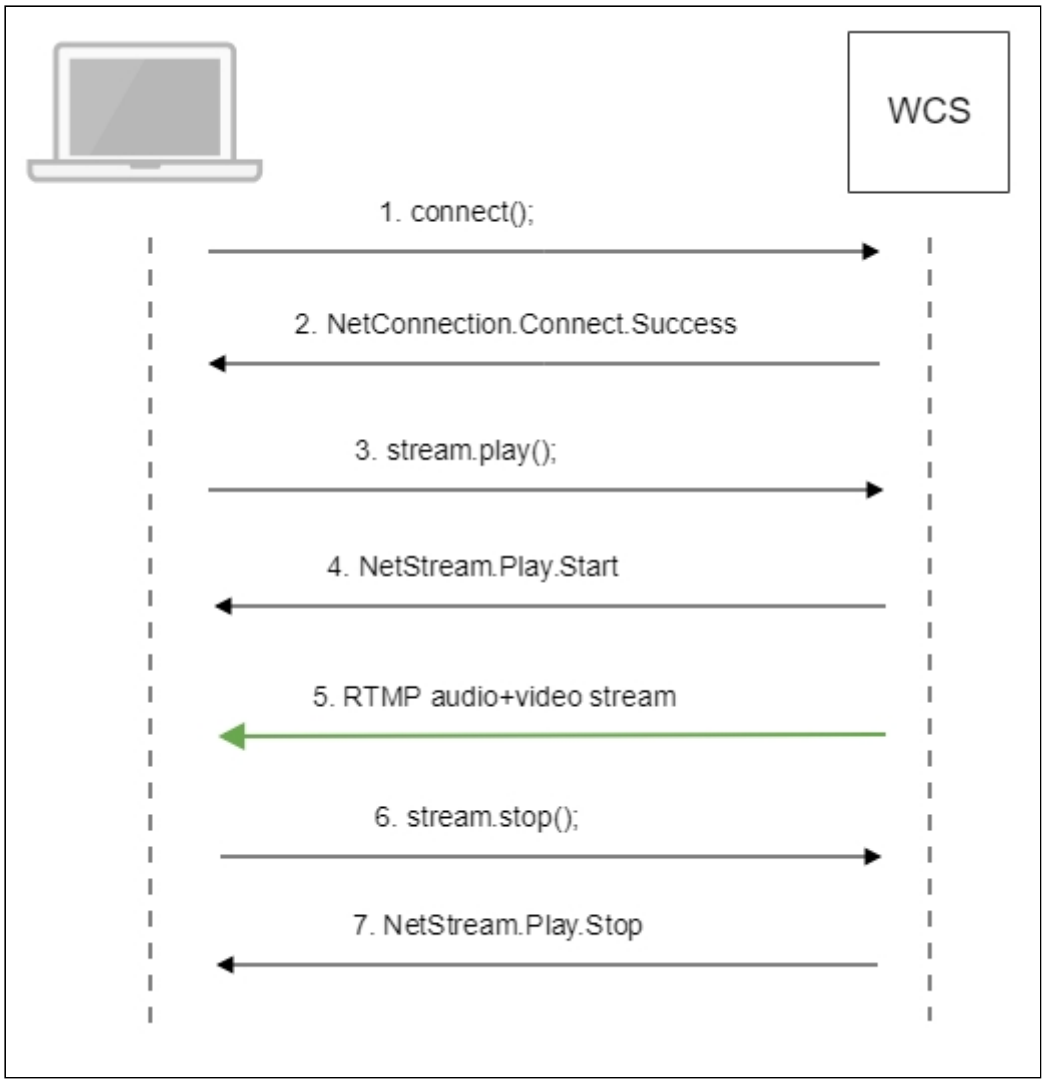
8. Нажмите кнопку `Start` в поле `Play`. Начнется воспроизведение потока:



Последовательность выполнения операций (Call Flow)

Ниже описана последовательность вызовов при использовании примера Flash Streaming для воспроизведения потока

streaming.mxml



1. Установка соединения с сервером

   `connect()` code

```
private function connect():void{
   var url:String = StringUtil.trim(connectUrl.text);
   Logger.info("connect " &#9989; url);
   nc = new NetConnection();
   //if (url.indexOf("rtmp") == 0){
   //  nc.objectEncoding = ObjectEncoding.AMF0;
   //}
   nc.client = this;
   nc.addEventListener(NetStatusEvent.NET_STATUS, handleConnectionStatus);
   var obj:Object = new Object();
   obj.login = generateRandomString(20);
   obj.appKey  = "flashStreamingApp";
   nc.connect(url,obj);
}
```

2. Получение от сервера события, подтверждающего успешное соединение

   `NetConnection.Connect.Success` code

```
private function handleConnectionStatus(event:NetStatusEvent):void{
  Logger.info("handleConnectionStatus: "&#9989;event.info.code);
  if (event.info.code=="NetConnection.Connect.Success"){
      Logger.info("near id: "&#9989;nc.nearID);
      Logger.info("far id: "&#9989;nc.farID);
      Logger.info("Connection opened");
      disconnectBtn.visible = true;
      connectBtn.visible = false;
      playBtn.enabled = true;
      publishBtn.enabled = true;
      setConnectionStatus("CONNECTED");
  } else if (event.info.code=="NetConnection.Connect.Closed" ||
event.info.code=="NetConnection.Connect.Failed"){
      ...
  }
}
```

3. Воспроизведение потока
   `stream.play()` code

```
private function addListenerAndPlay():void{
  ...
  subscribeStreamObject = createStreamObject();
  subscribeStream.play(playStreamName.text);
  videoFarEnd.attachNetStream(subscribeStream);
  videoFarEnd.width = 320;
  videoFarEnd.height = 240;
  videoFarEnd.visible = true;
}
```

4. Получение от сервера события, подтверждающего успешное воспроизведение потока
   `NetStream.Play.Start` code

```
private function handleSubscribeStreamStatus(event:NetStatusEvent):void{
  Logger.info("handleSubscribeStreamStatus: "&#9989;event.info.code);
  switch (event.info.code) {
      case "NetStream.Play.PublishNotify":
      case "NetStream.Play.Start":
          setPlayStatus("PLAYING");
          playBtn.visible = false;
          stopBtn.enabled = true;
          stopBtn.visible = true;
          break;
      ...
  }
}
```

5. Прием аудио-видео потока по RTMP
6. Остановка воспроизведения потока
   `stream.close()` code

```
private function stop():void{
     if (subscribeStream != null) {
         stopBtn.enabled = false;
         subscribeStream.close();
         subscribeStream = null;
     }
     subscribeStreamObject = null;
     videoFarEnd.visible = false;
}
```

7. Получение от сервера события, подтверждающего остановку воспроизведения потока
`NetStream.Play.Stop` code

```
private function handleSubscribeStreamStatus(event:NetStatusEvent):void{
  Logger.info("handleSubscribeStreamStatus: "&#9989;event.info.code);
  switch (event.info.code) {
      ...
      case "NetStream.Play.UnpublishNotify":
      case "NetStream.Play.Stop":
          setPlayStatus("STOPPED");
          playBtn.enabled = true;
          playBtn.visible = true;
          stopBtn.visible = false;
          break;
      ...
  }
}
```