CSCI 4140 – Tutorial 6

Assignment 2 Overview

Hints on Front-end Development

Matt YIU, Man Tung (mtyiu@cse)

SHB 118

Office Hour: Tuesday, 3-5 pm

2015.02.26

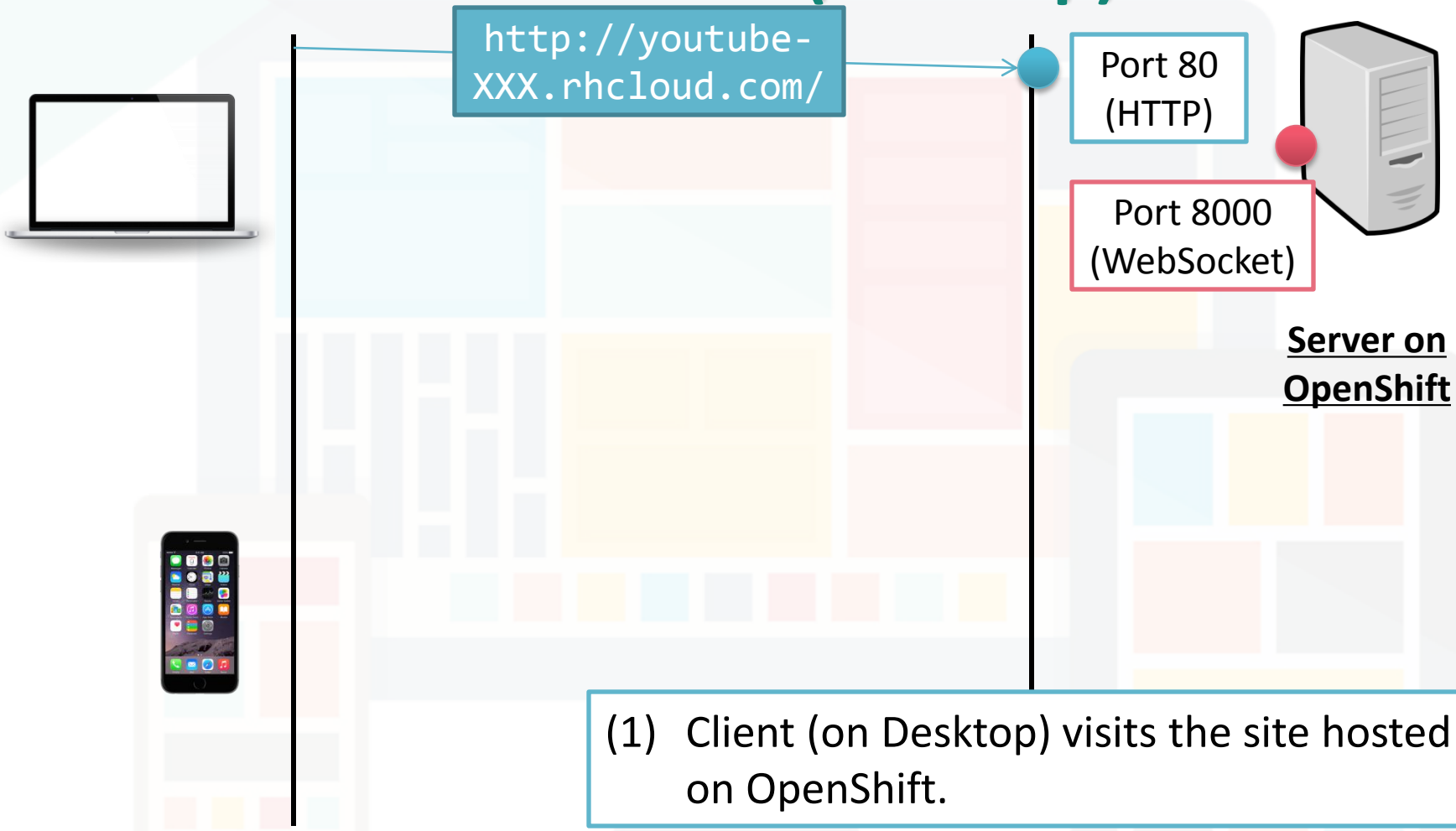
Outline

- Work flow
- Client side
  - Layout design
  - QR code display
  - Playlist management
  - YouTube player control
- Server side
  - Routing
  - Message forwarding
  - Retrieving video title

Work flow

Let's define what will happen between the clients and server…

Work flow: Initialization (Desktop)

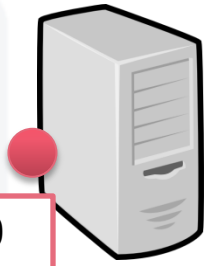


Work flow: Initialization (Desktop)

http://youtube-XXX.rhcloud.com/**session/4140**

Port 80 (HTTP)

Port 8000 (WebSocket)

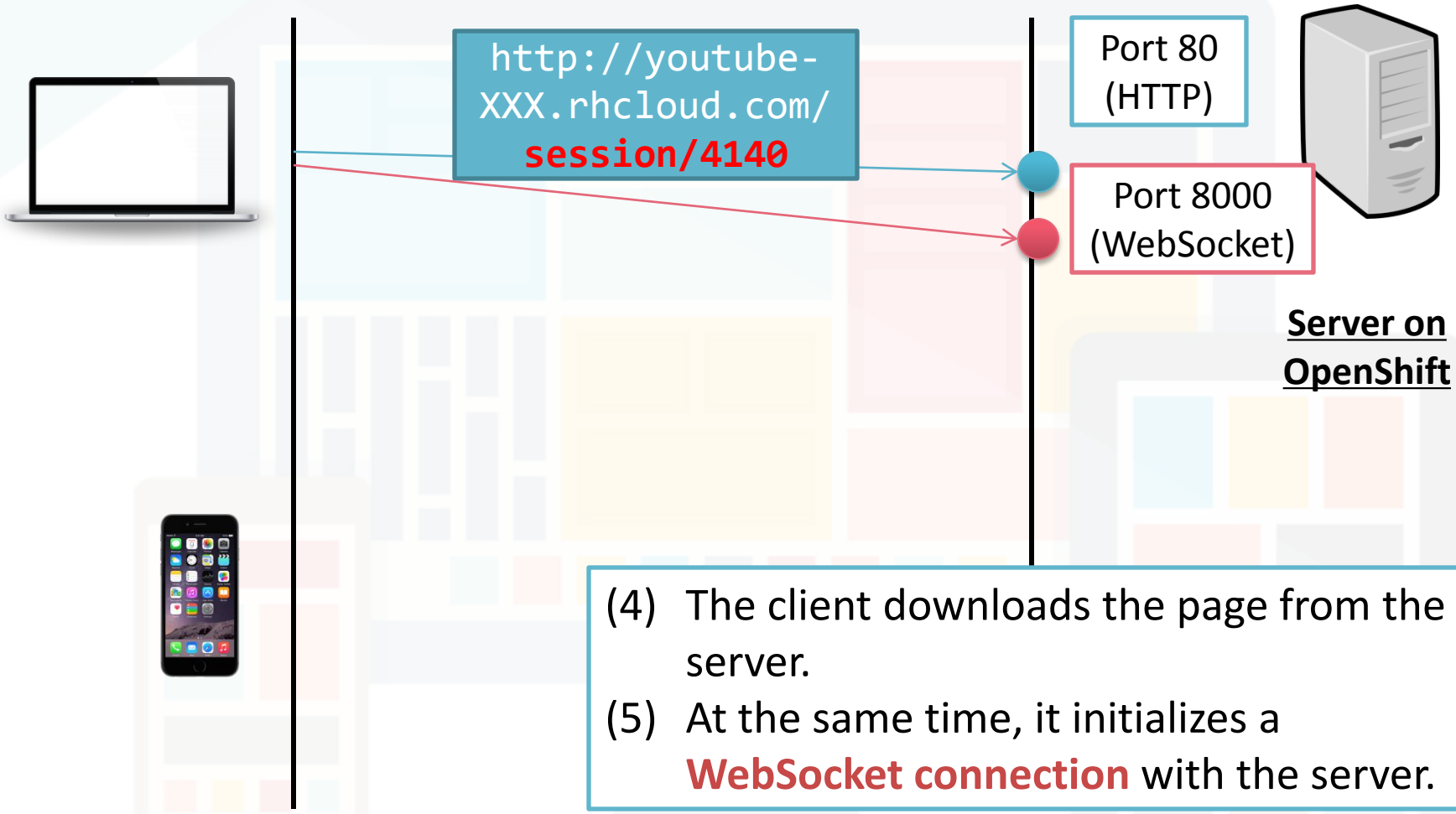


Server on OpenShift

(2) The server generate a **unique session number** to the client.

(3) The server **redirects** the client to the player page for this session.

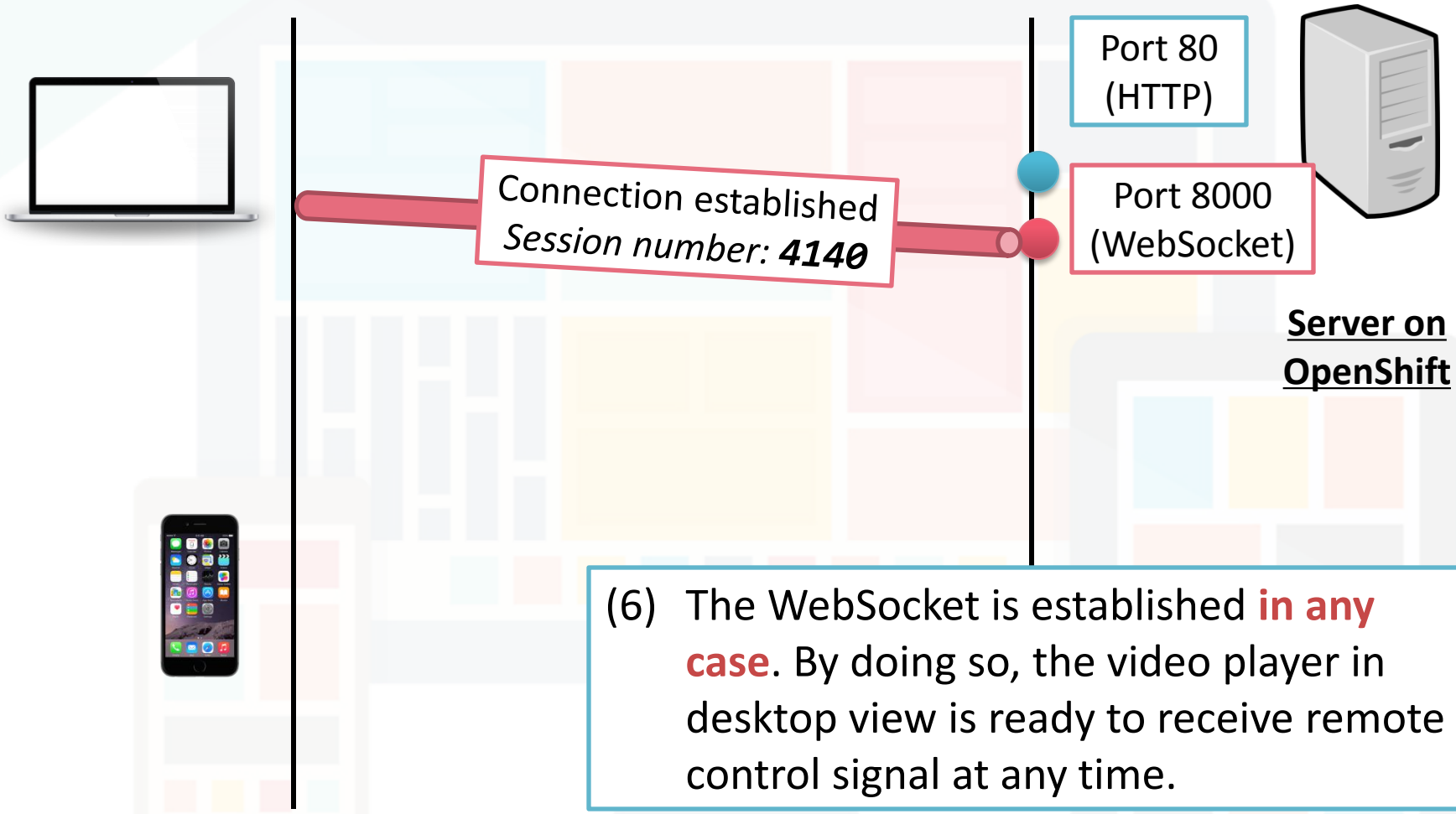
Work flow: Initialization (Desktop)



(4) The client downloads the page from the server.

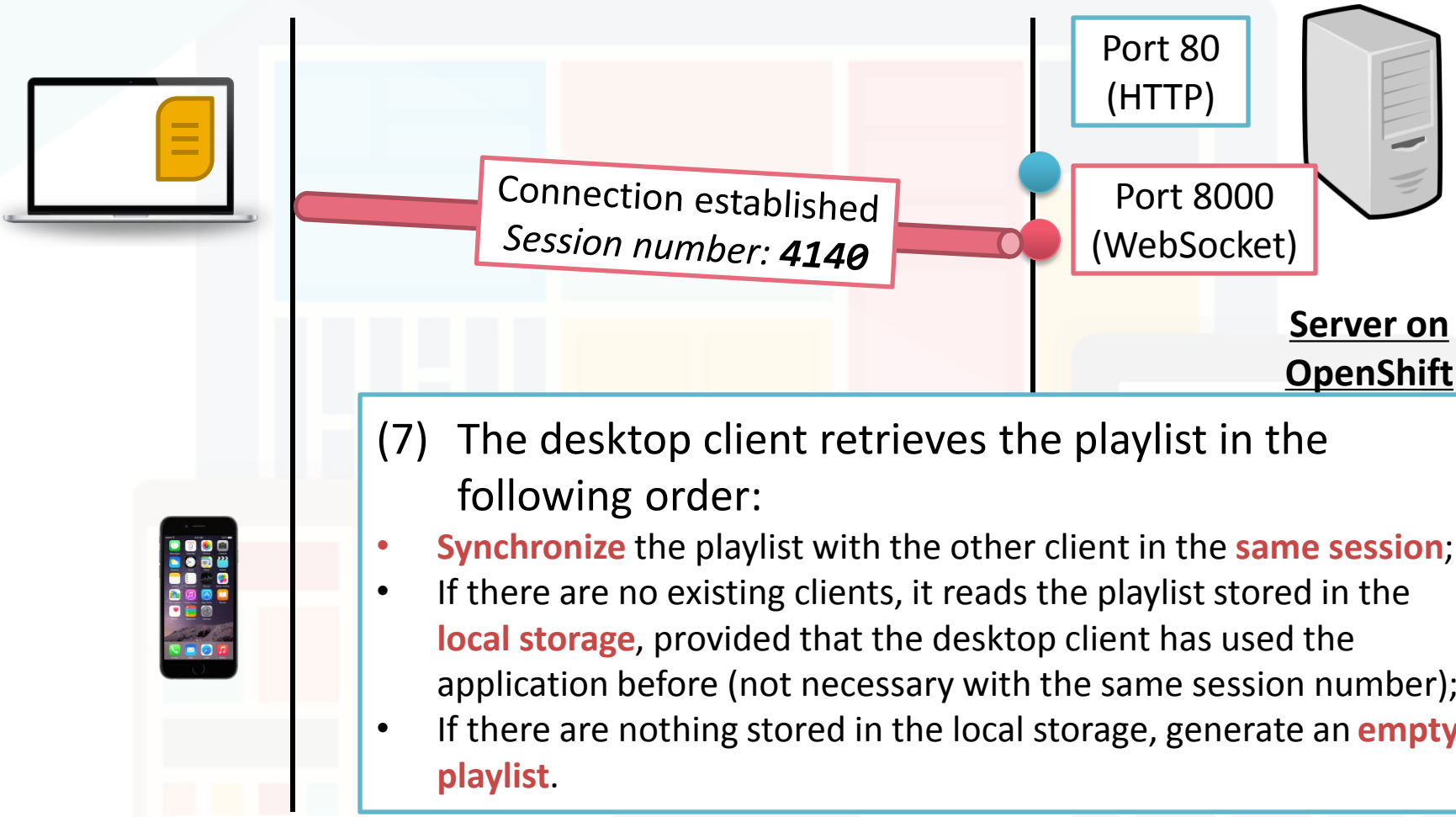
(5) At the same time, it initializes a **WebSocket connection** with the server.

Work flow: Initialization (Desktop)



(6) The WebSocket is established **in any case**. By doing so, the video player in desktop view is ready to receive remote control signal at any time.

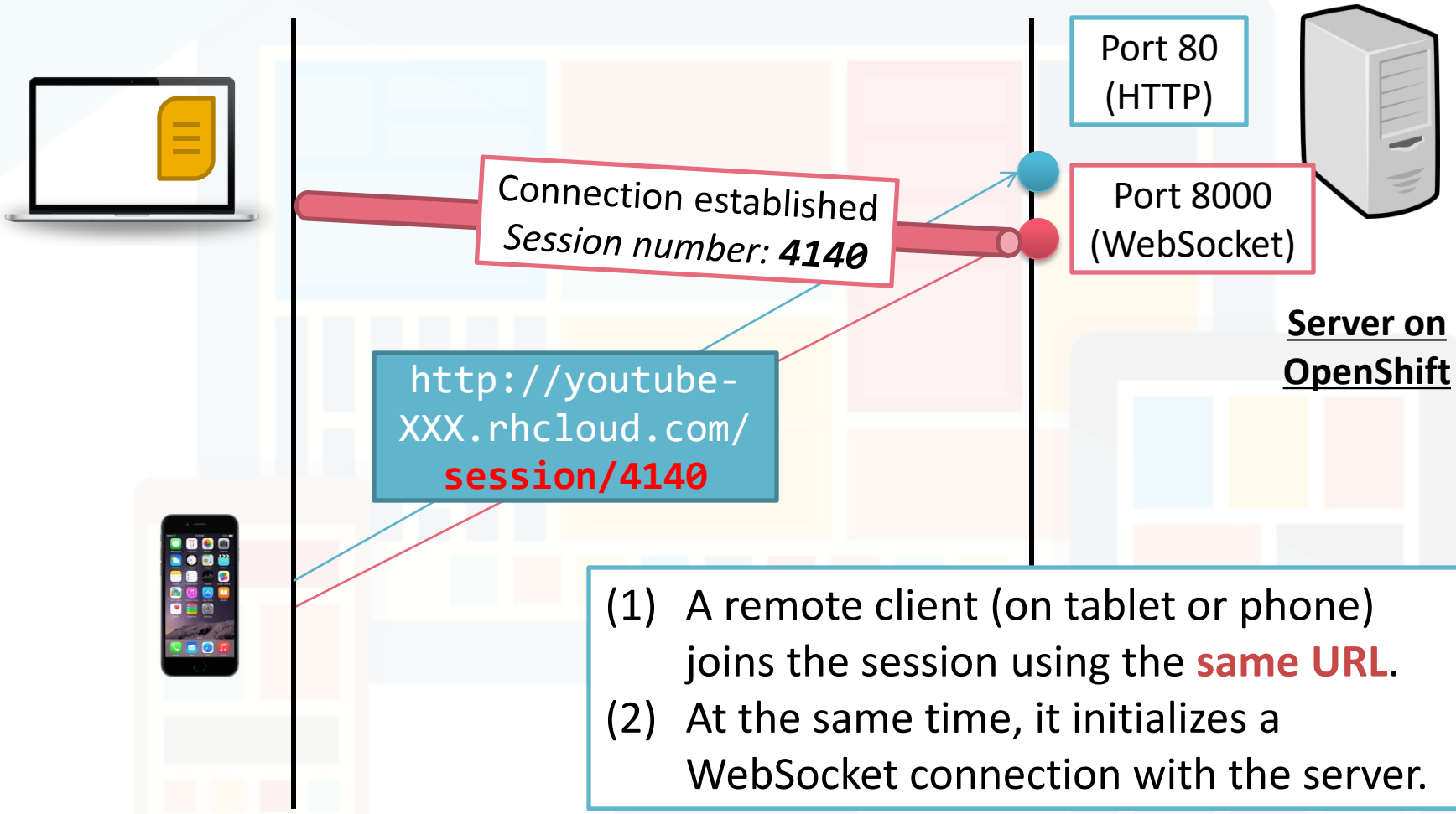
Work flow: Initialization (Desktop)



(7) The desktop client retrieves the playlist in the following order:

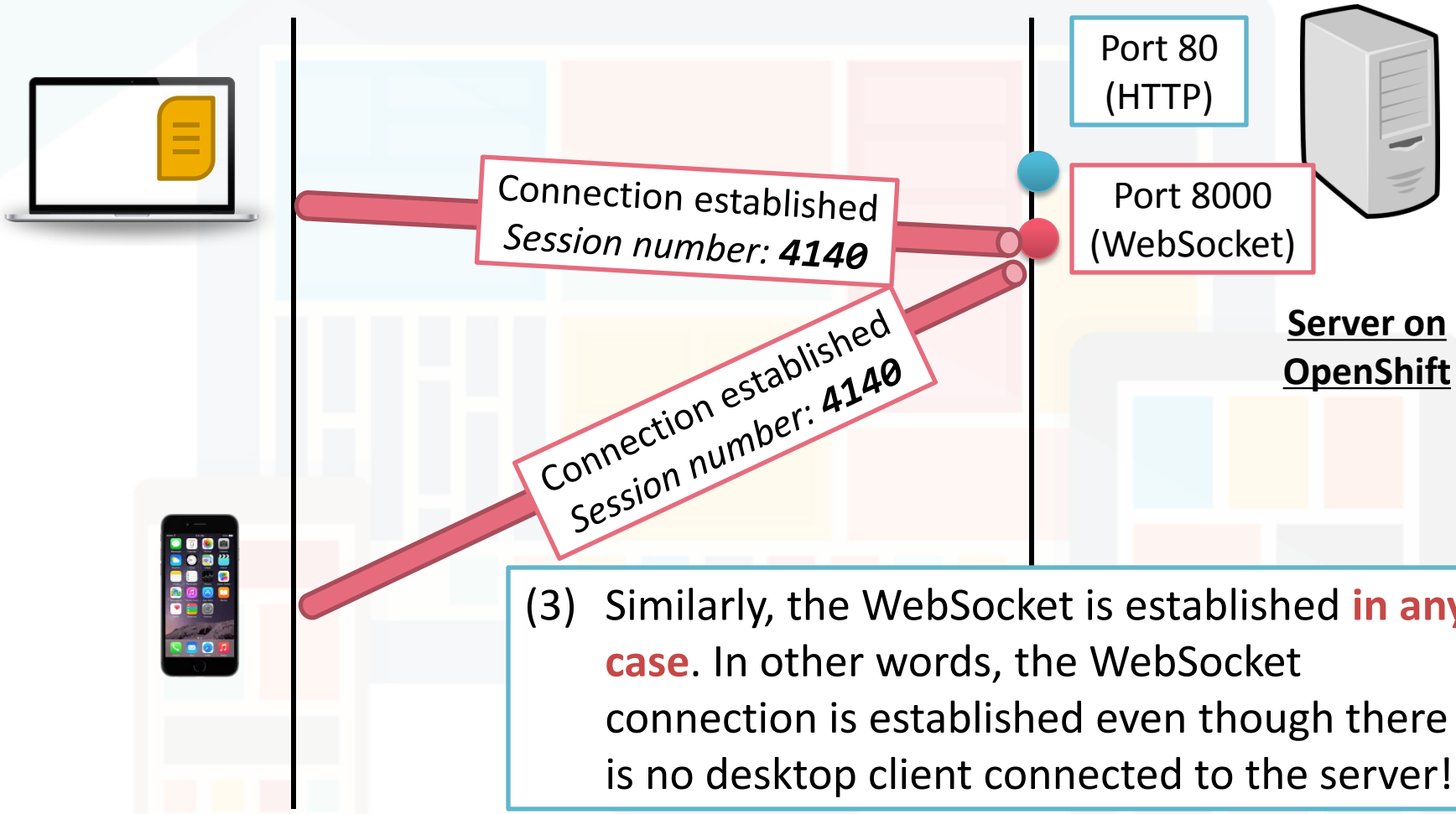
- **Synchronize** the playlist with the other client in the **same session**;
- If there are no existing clients, it reads the playlist stored in the **local storage**, provided that the desktop client has used the application before (not necessary with the same session number);
- If there are nothing stored in the local storage, generate an **empty playlist**.

Work flow: Initialization (Tablet or phone)



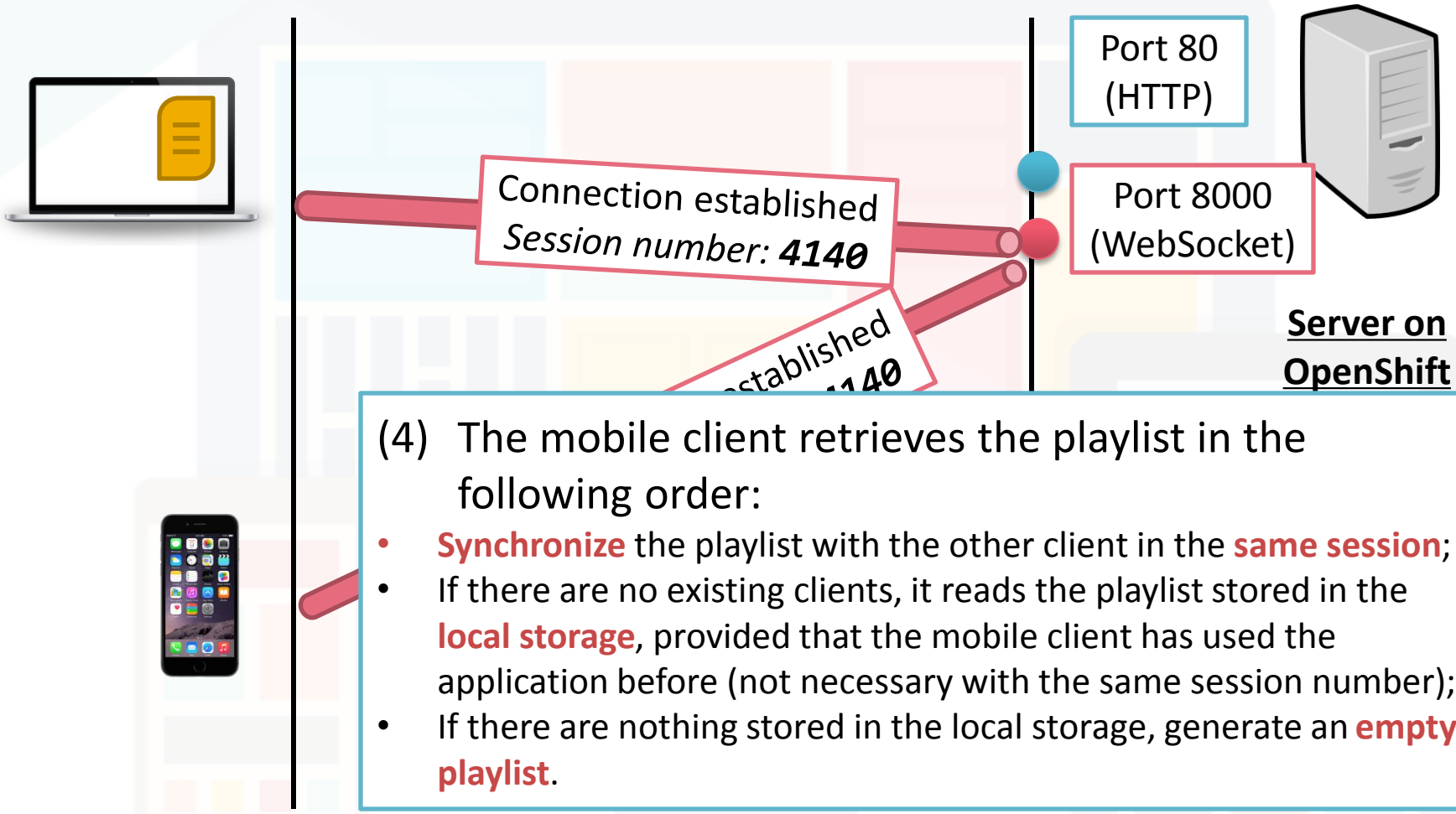
(1) A remote client (on tablet or phone) joins the session using the **same URL**.
(2) At the same time, it initializes a WebSocket connection with the server.

Work flow: Initialization (Tablet or phone)



(3) Similarly, the WebSocket is established **in any case**. In other words, the WebSocket connection is established even though there is no desktop client connected to the server!

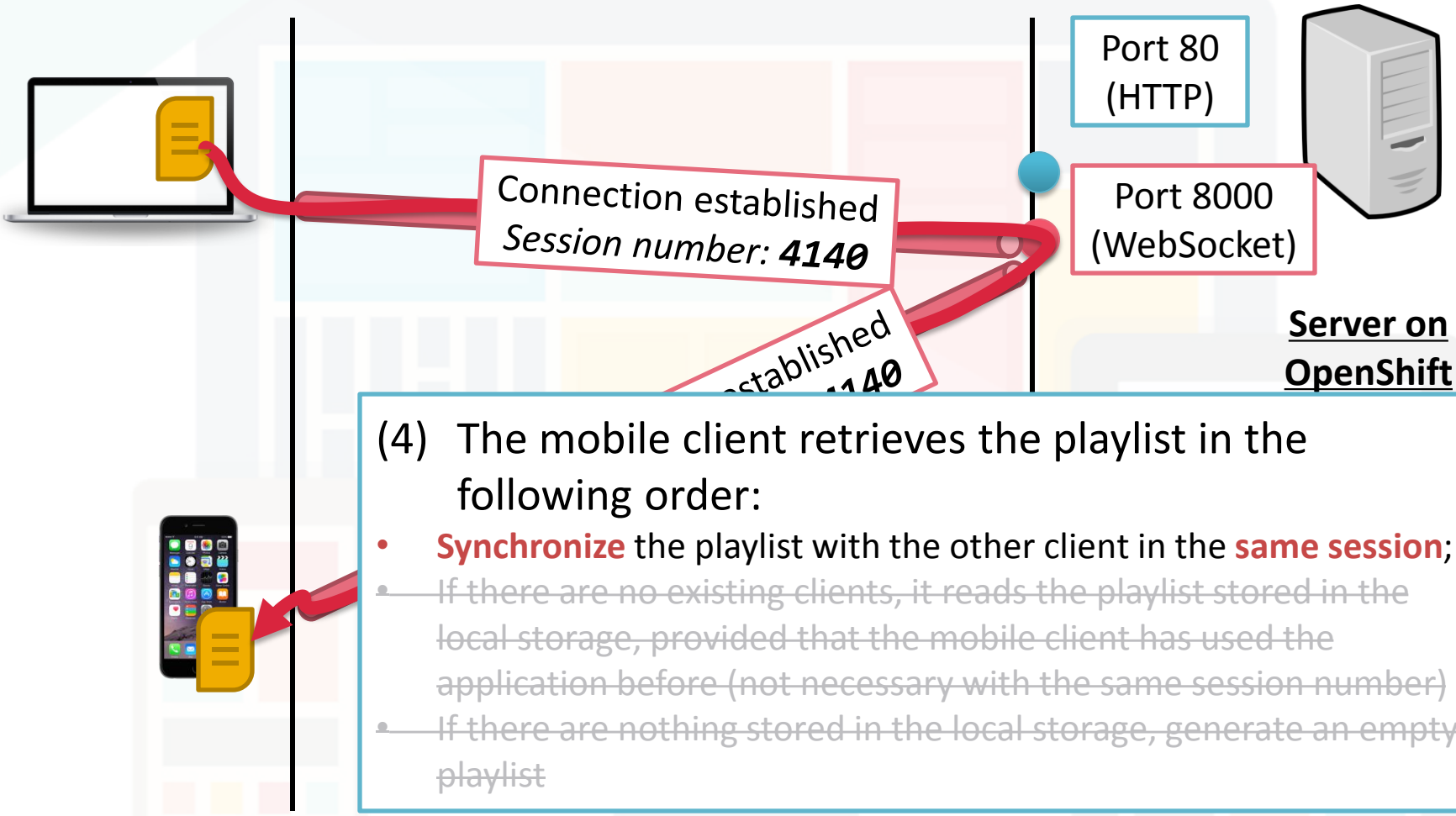
Work flow: Initialization (Tablet or phone)



(4) The mobile client retrieves the playlist in the following order:

- **Synchronize** the playlist with the other client in the **same session**;
- If there are no existing clients, it reads the playlist stored in the **local storage**, provided that the mobile client has used the application before (not necessary with the same session number);
- If there are nothing stored in the local storage, generate an **empty playlist**.

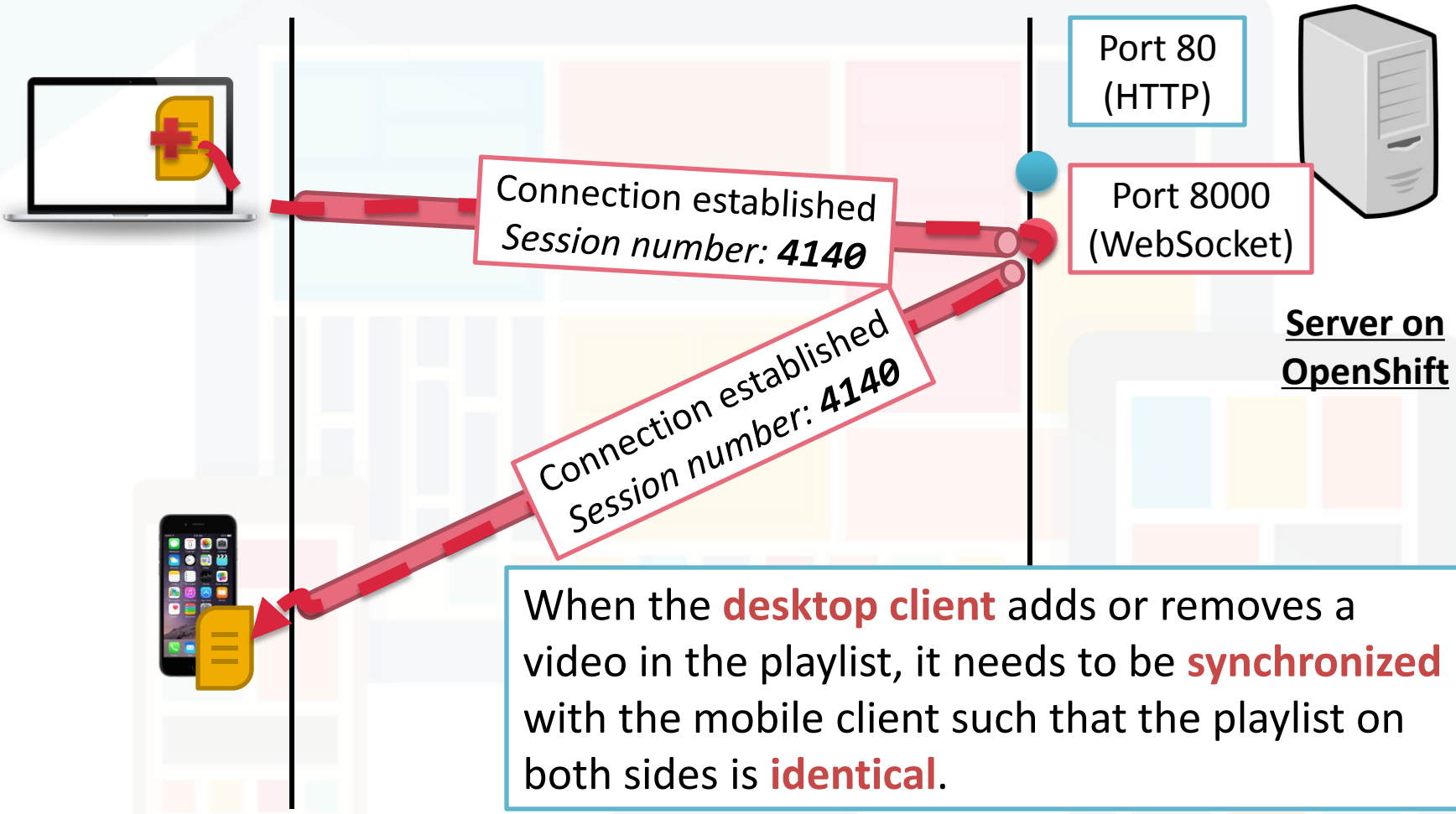
Work flow: Initialization (Tablet or phone)



(4) The mobile client retrieves the playlist in the following order:

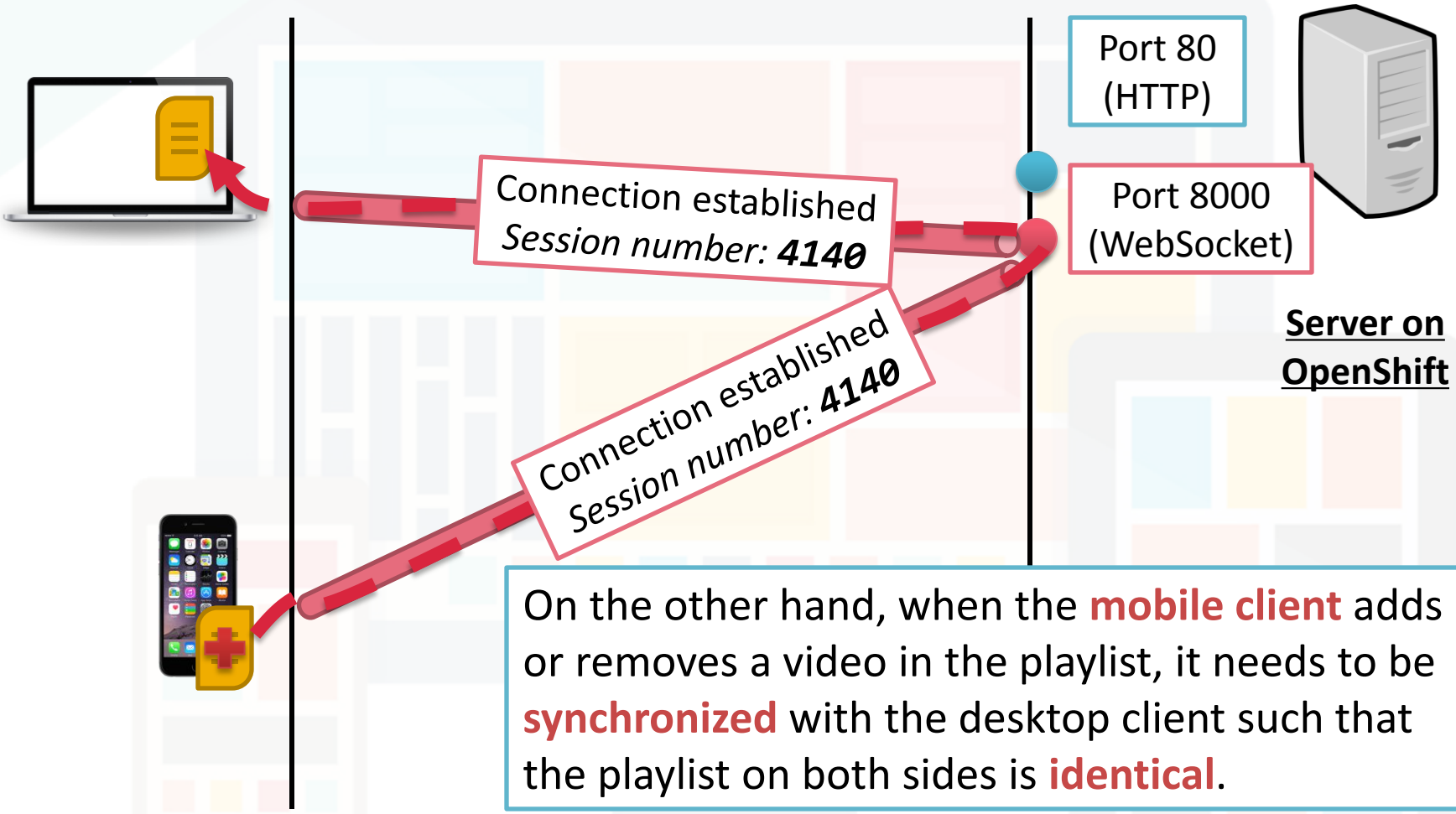
- **Synchronize** the playlist with the other client in the **same session**;
- ~~If there are no existing clients, it reads the playlist stored in the local storage, provided that the mobile client has used the application before (not necessary with the same session number)~~
- ~~If there are nothing stored in the local storage, generate an empty playlist~~

Work flow: Playlist synchronization



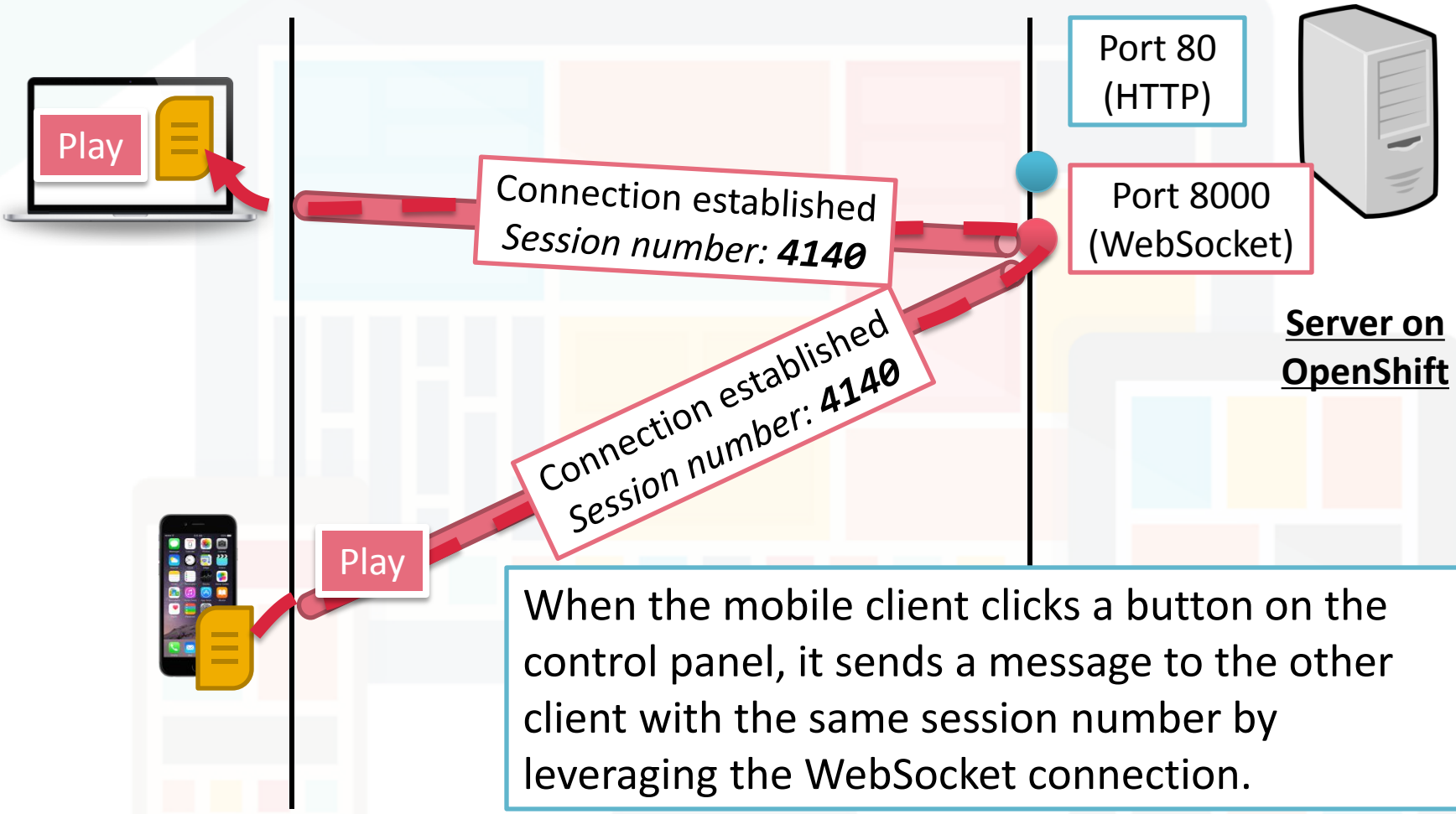
When the **desktop client** adds or removes a video in the playlist, it needs to be **synchronized** with the mobile client such that the playlist on both sides is **identical**.

Work flow: Playlist synchronization



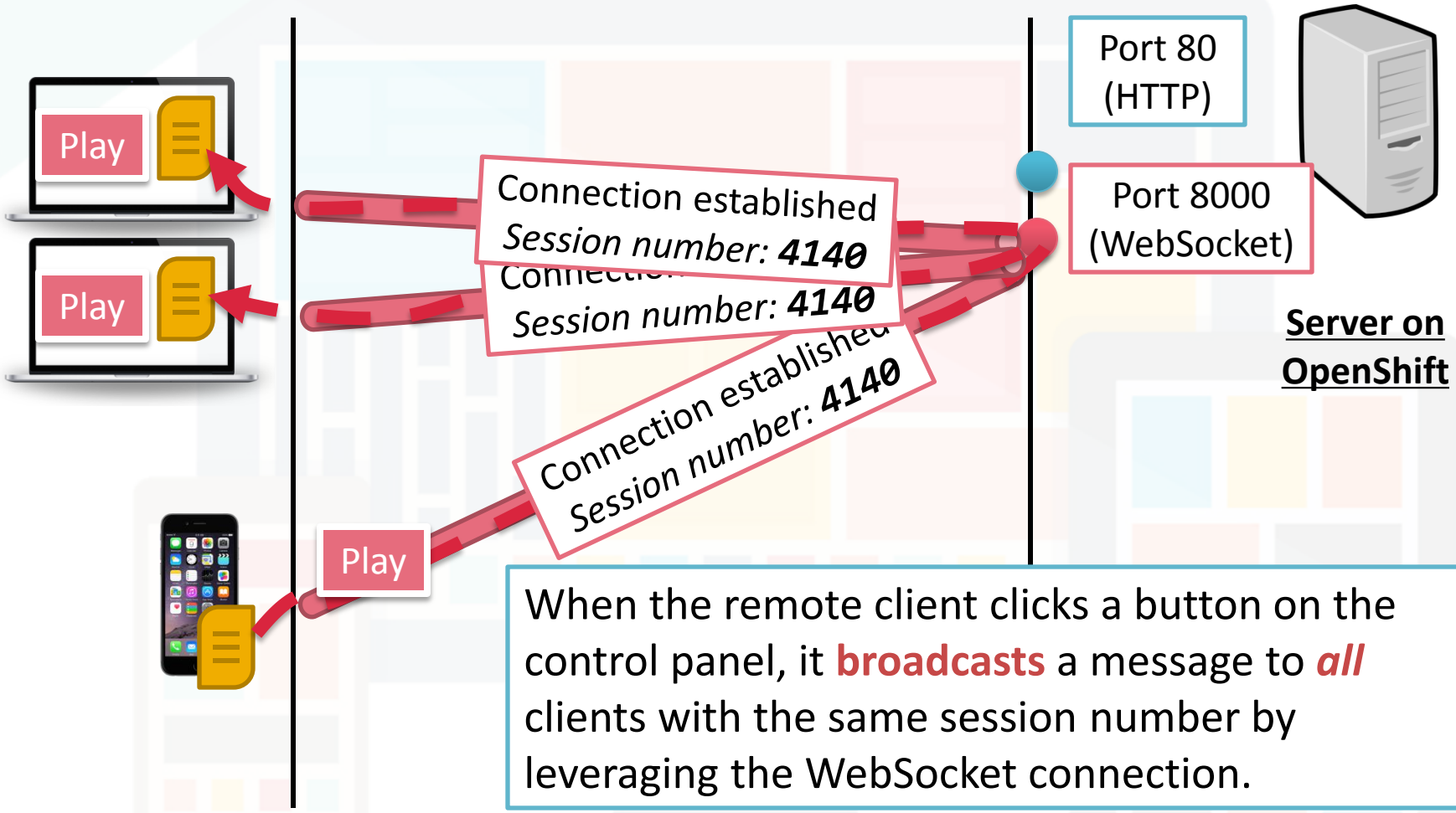
On the other hand, when the **mobile client** adds or removes a video in the playlist, it needs to be **synchronized** with the desktop client such that the playlist on both sides is **identical**.

Work flow: Remote control



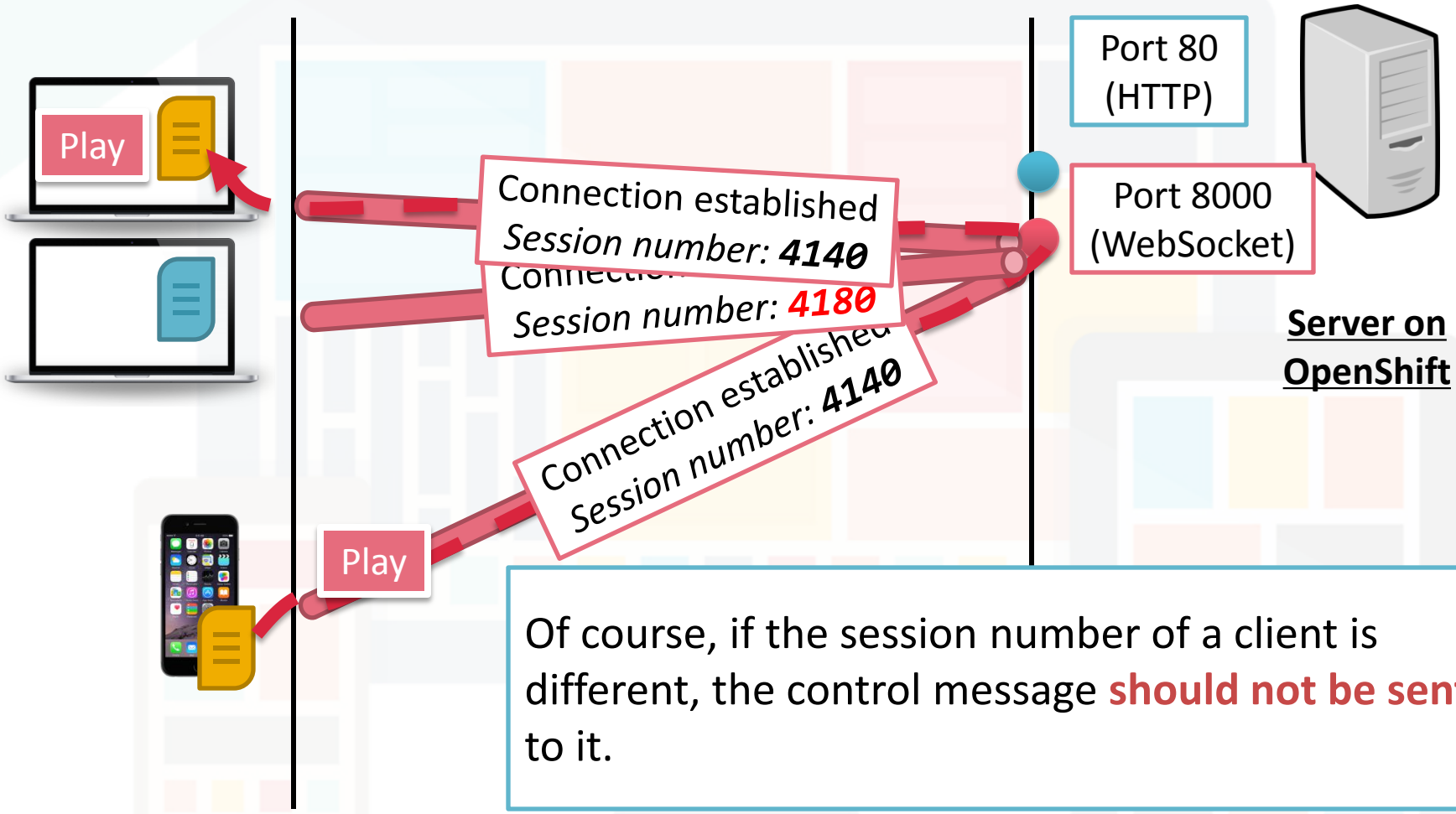
When the mobile client clicks a button on the control panel, it sends a message to the other client with the same session number by leveraging the WebSocket connection.

Work flow: Remote control



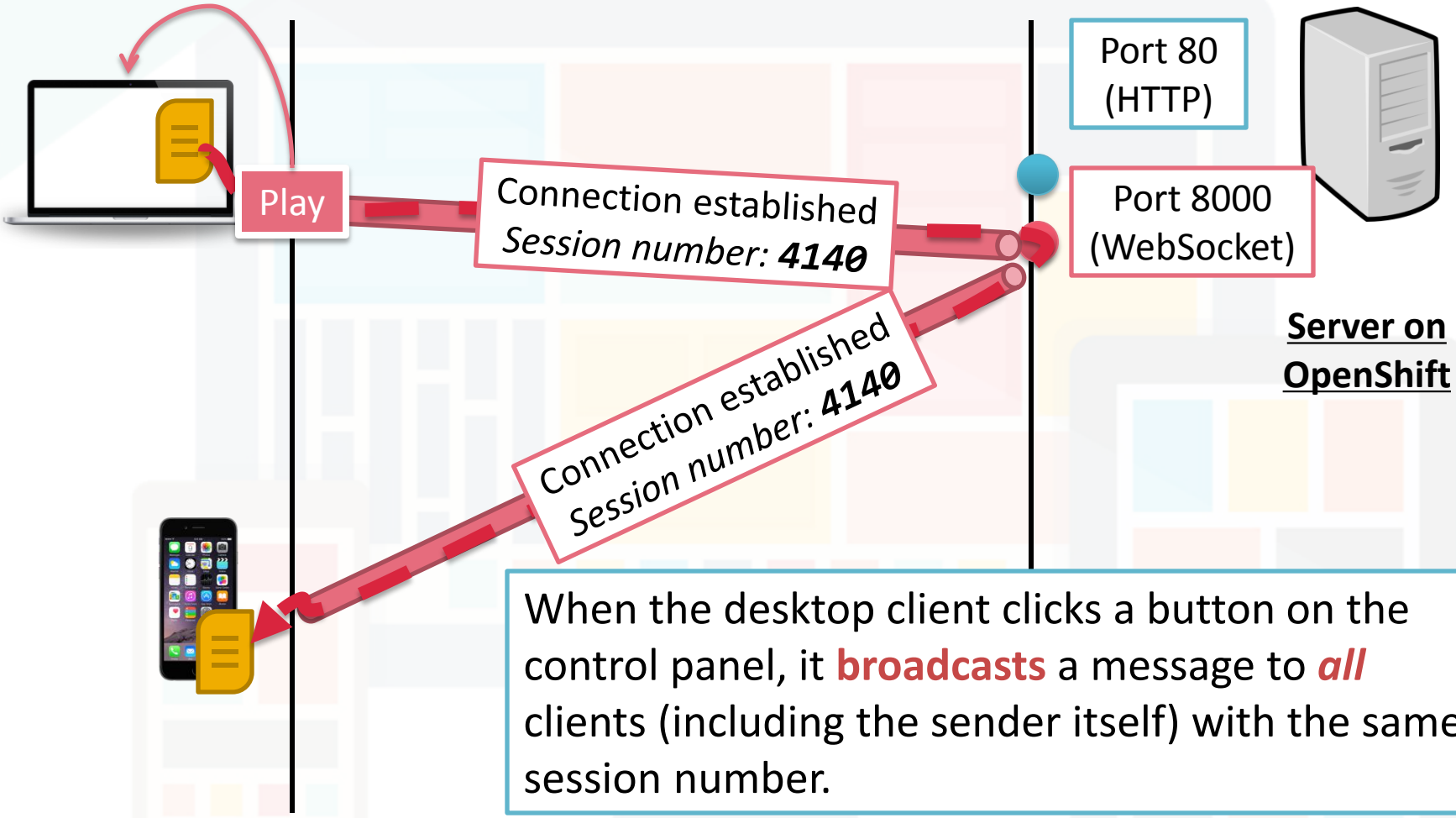
When the remote client clicks a button on the control panel, it **broadcasts** a message to ***all*** clients with the same session number by leveraging the WebSocket connection.

Work flow: Remote control



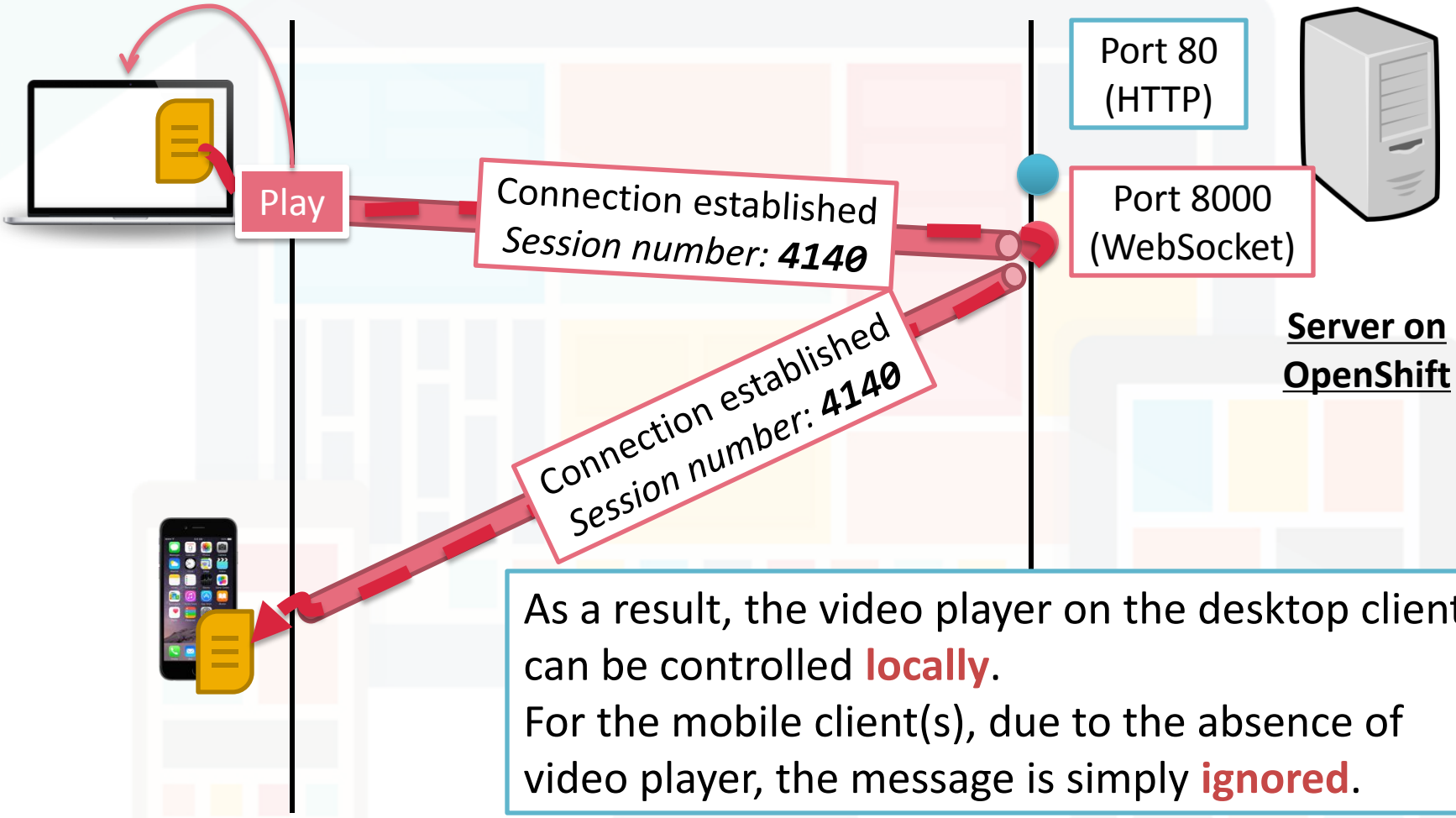
Of course, if the session number of a client is different, the control message **should not be sent** to it.

Work flow: Remote control

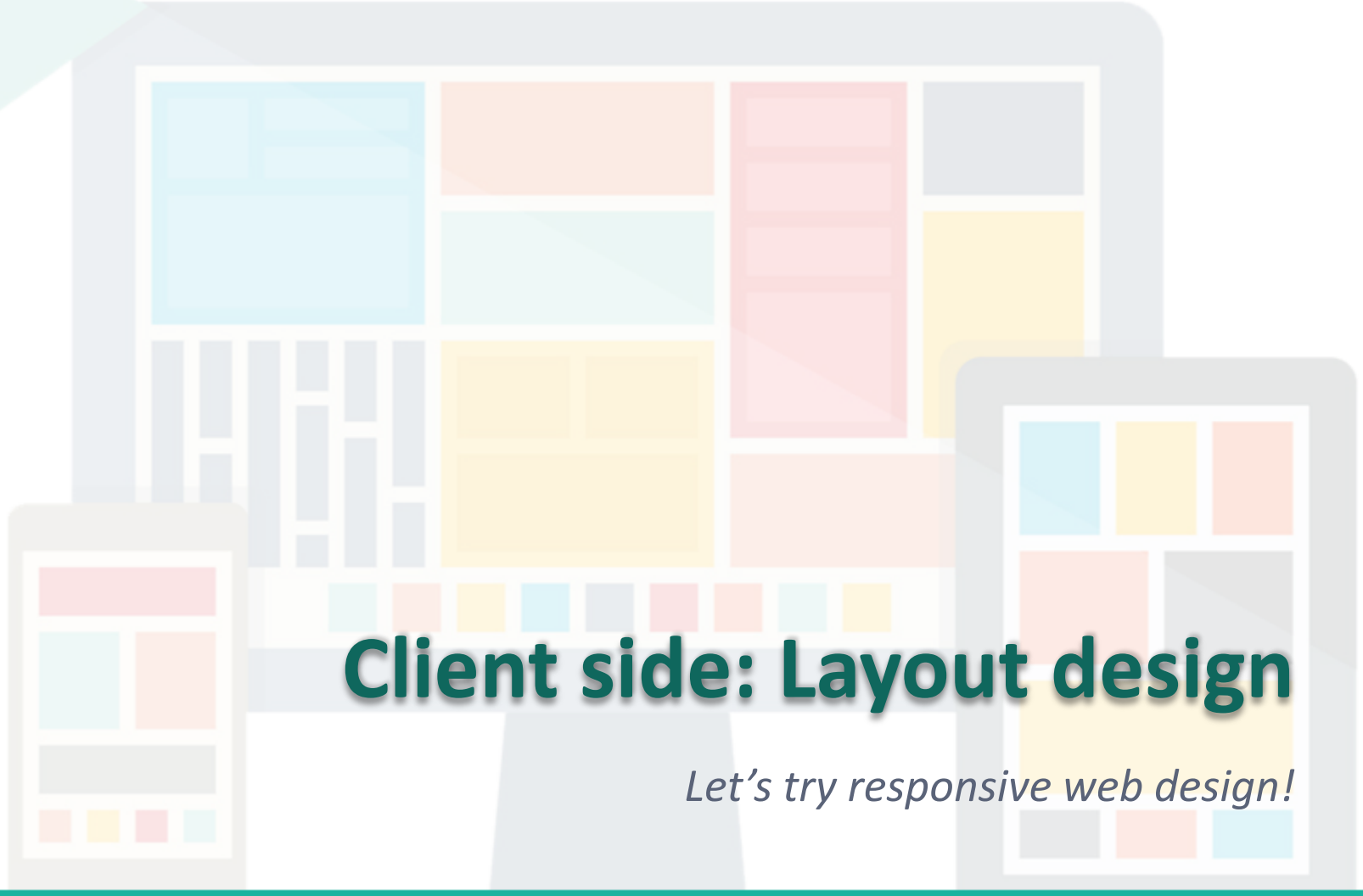


When the desktop client clicks a button on the control panel, it **broadcasts** a message to ***all*** clients (including the sender itself) with the same session number.

Work flow: Remote control

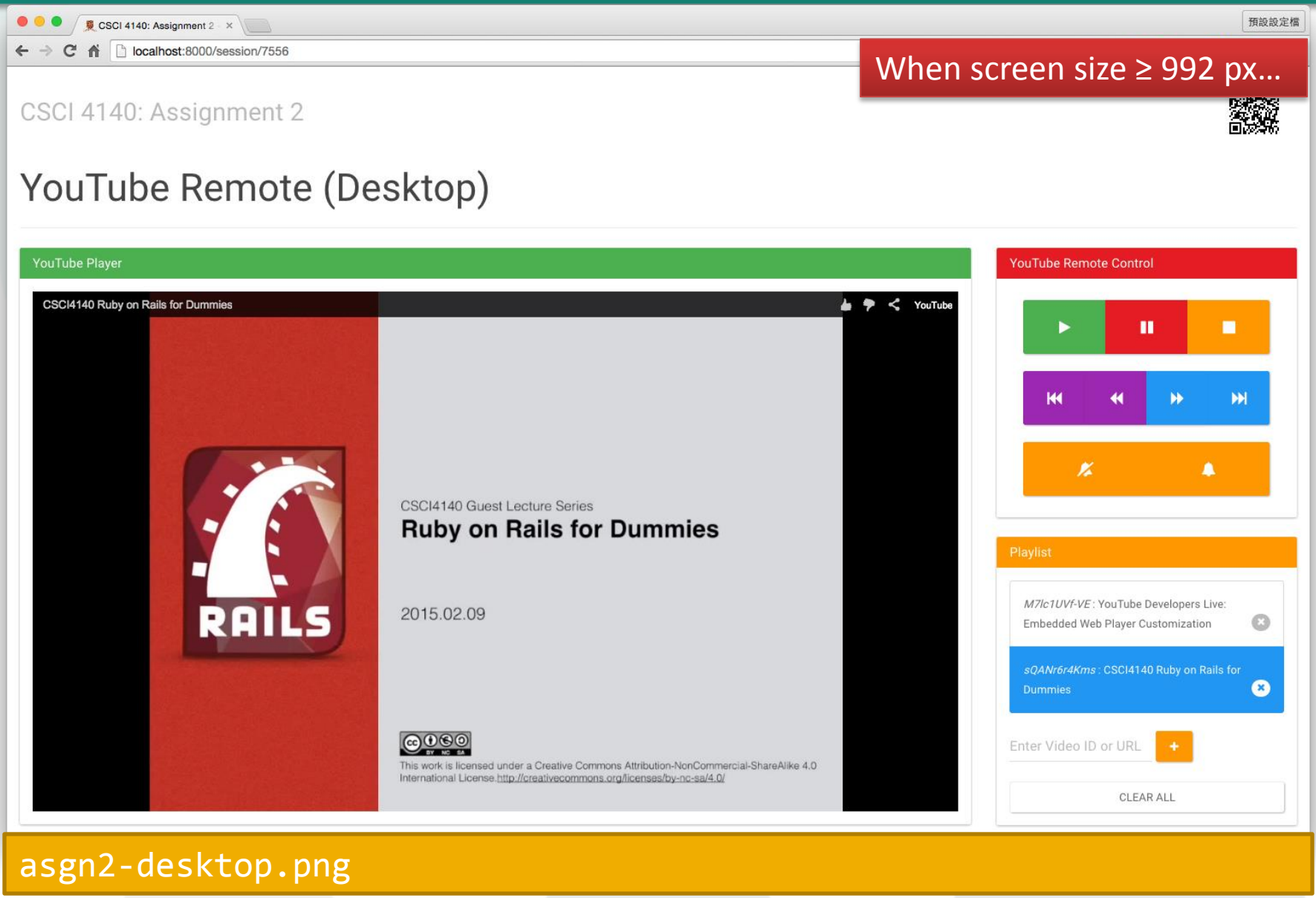


As a result, the video player on the desktop client can be controlled **locally**.
For the mobile client(s), due to the absence of video player, the message is simply **ignored**.

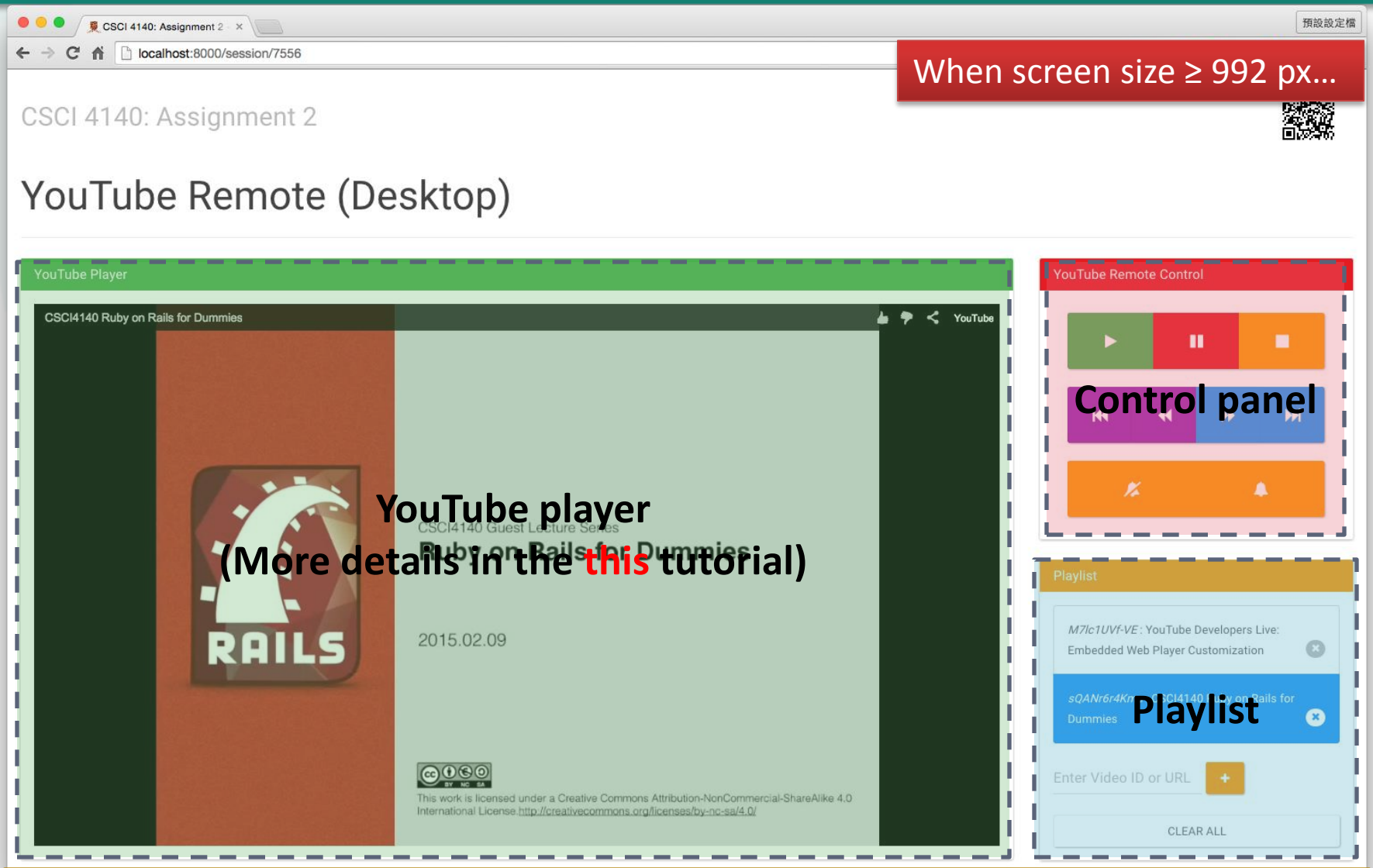


Client side: Layout design

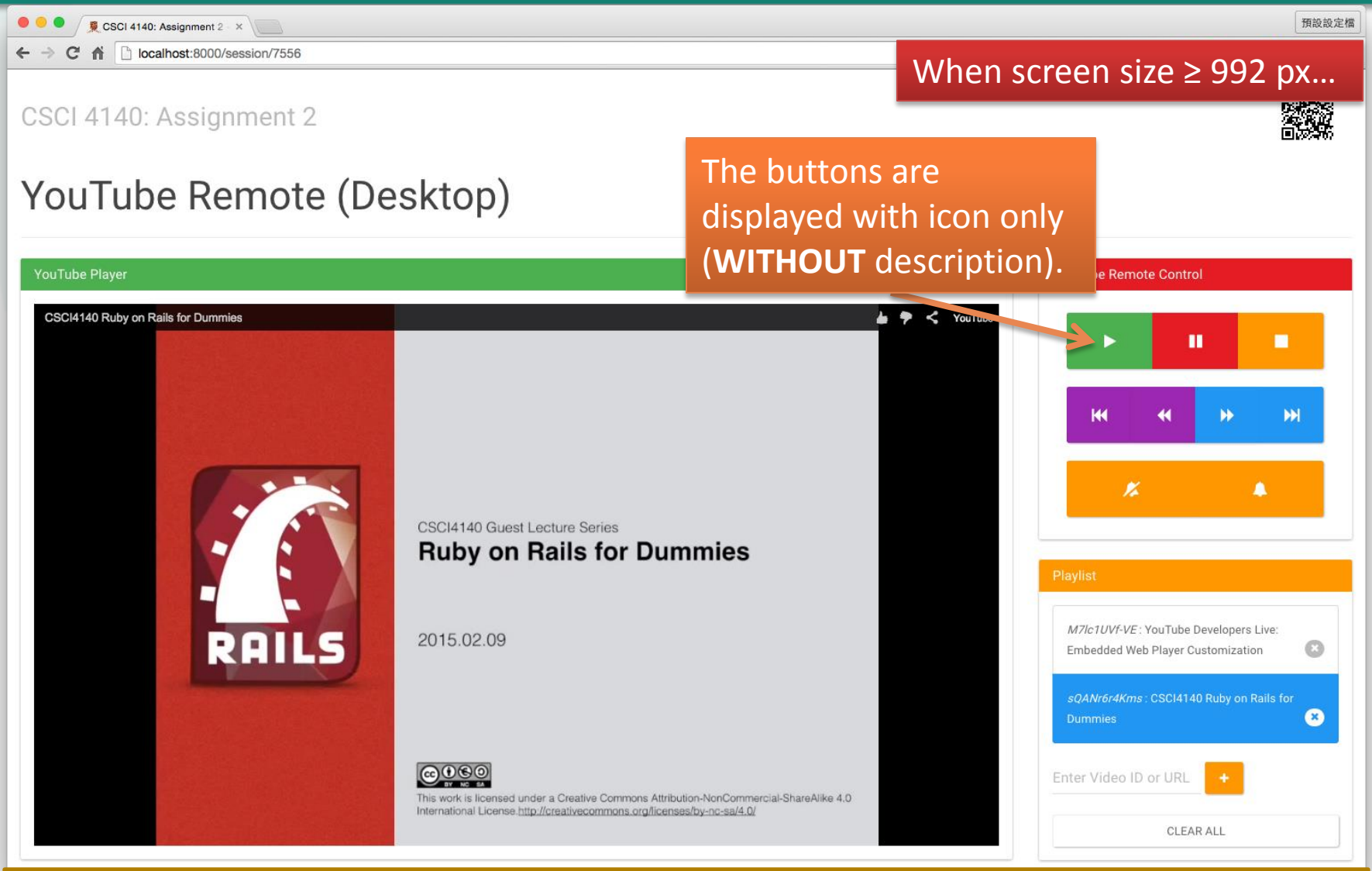
Let's try responsive web design!



asgn2-desktop.png

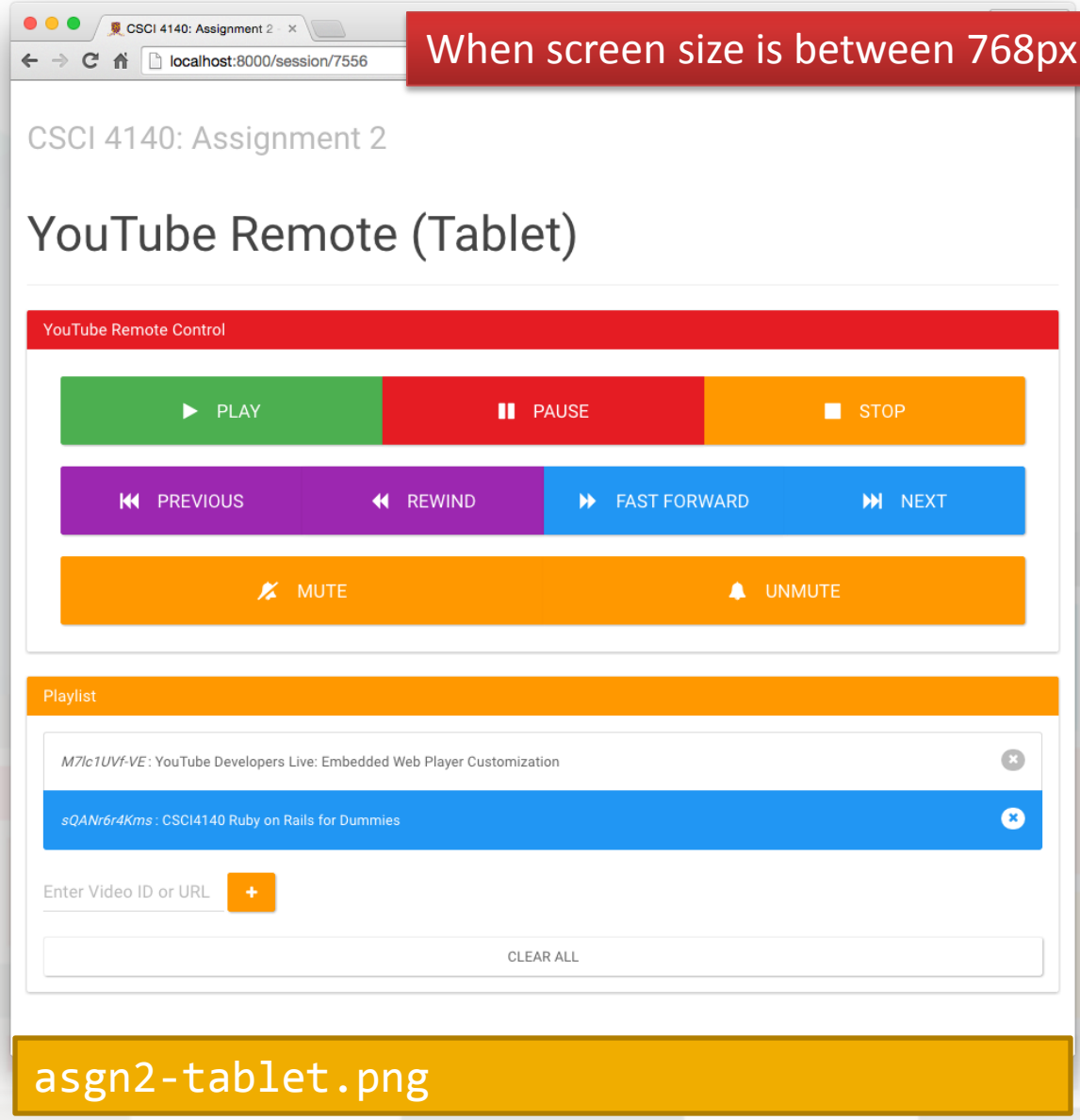


asgn2-desktop.png



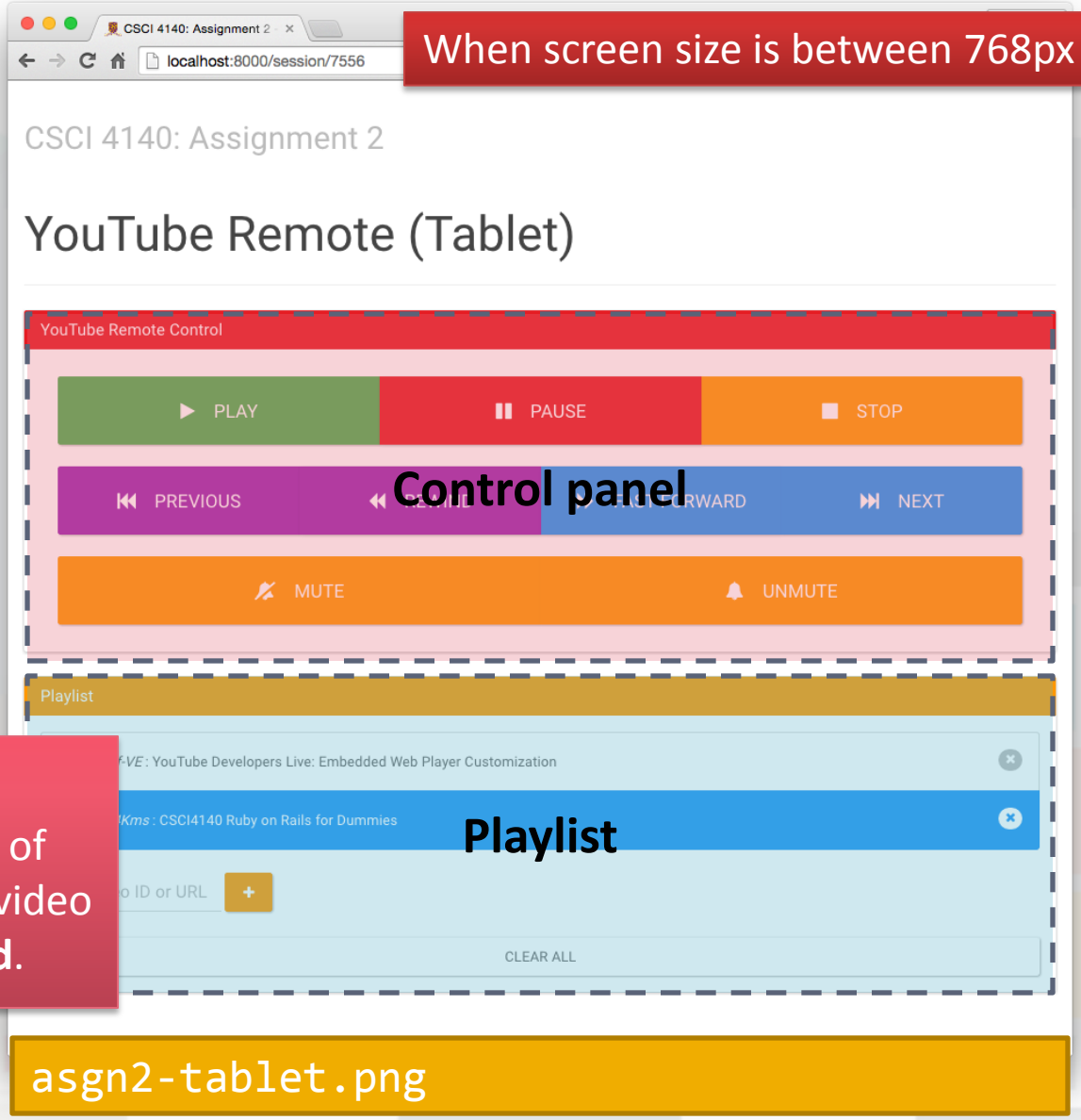
asgn2-desktop.png

When screen size is between 768px and 992 px…



asgn2-tablet.png

When screen size is between 768px and 992 px…

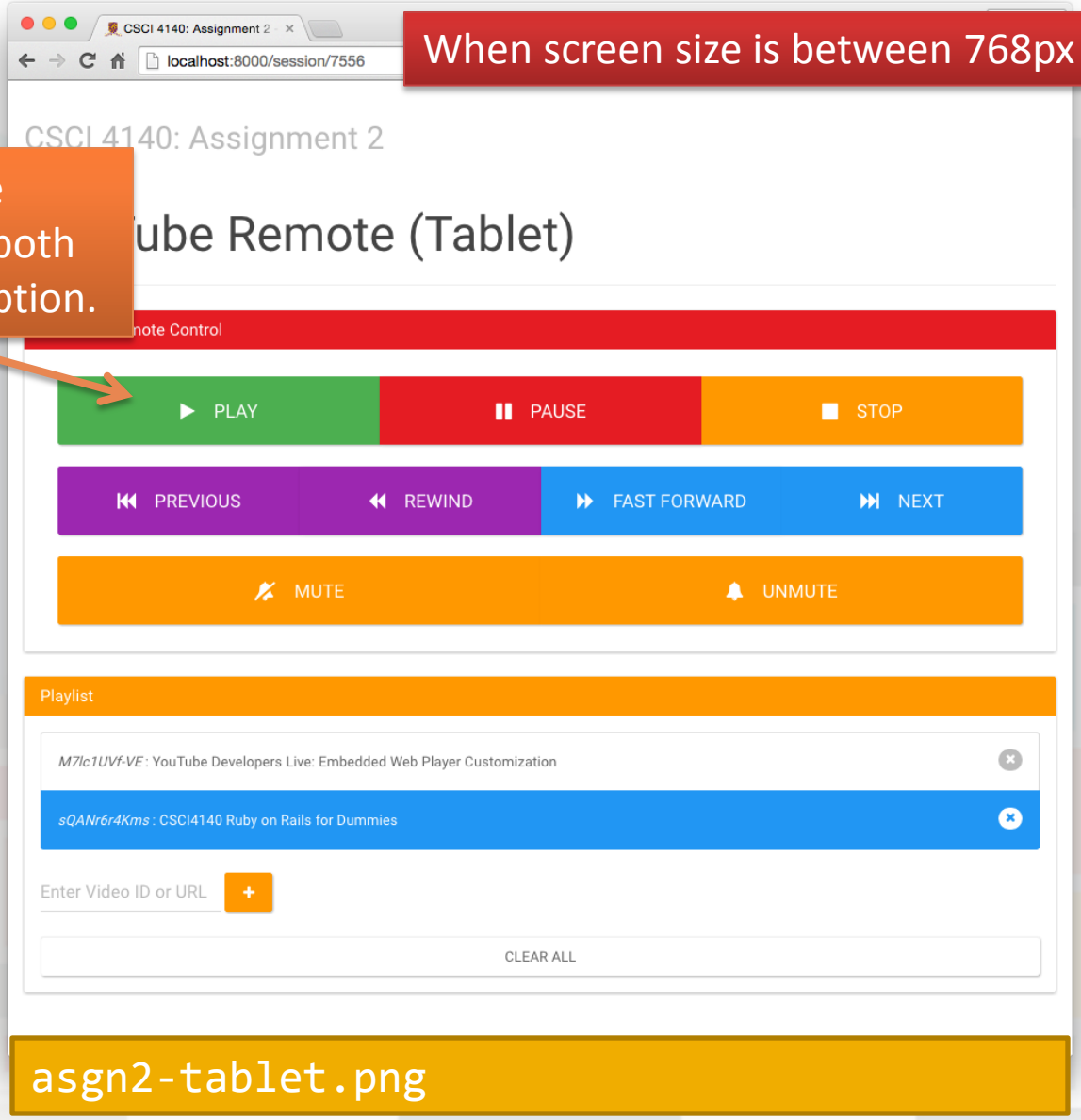


asgn2-tablet.png

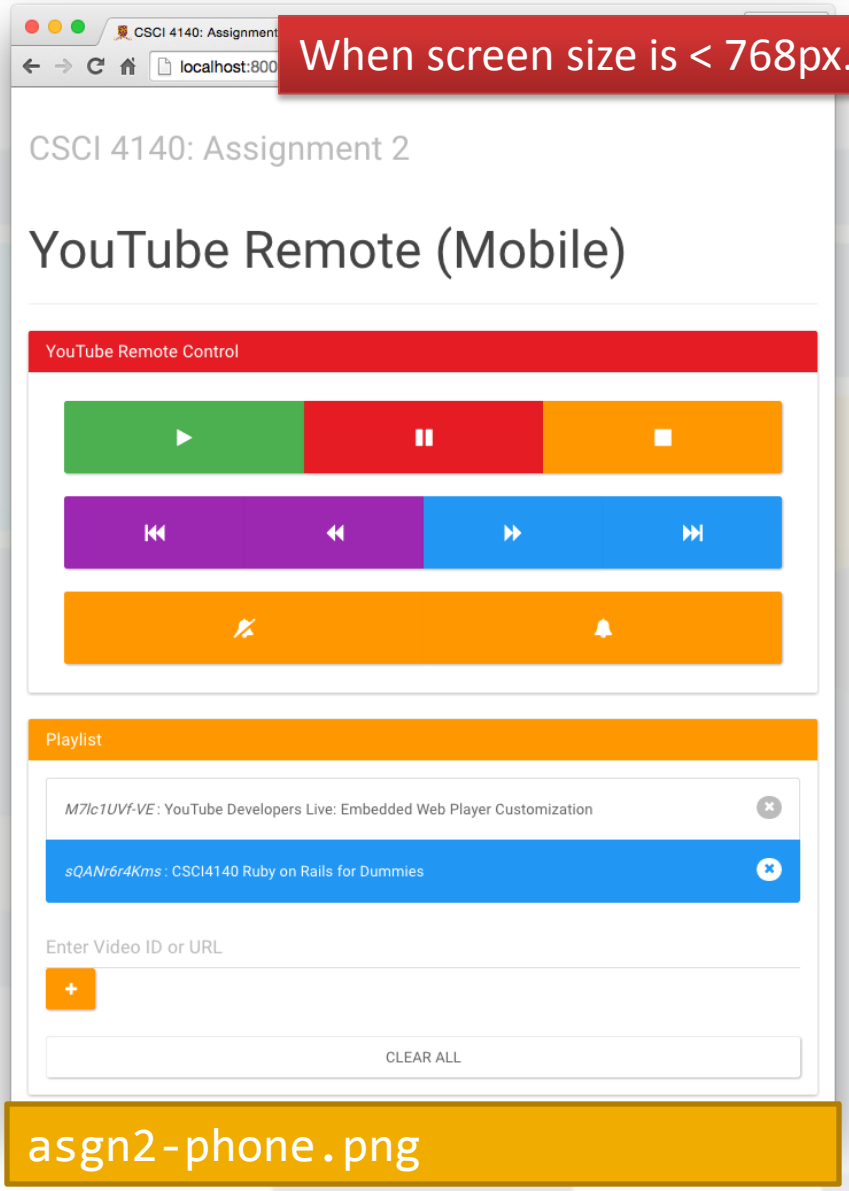
Note: When we reduce the width of the window, the video player is **removed**.

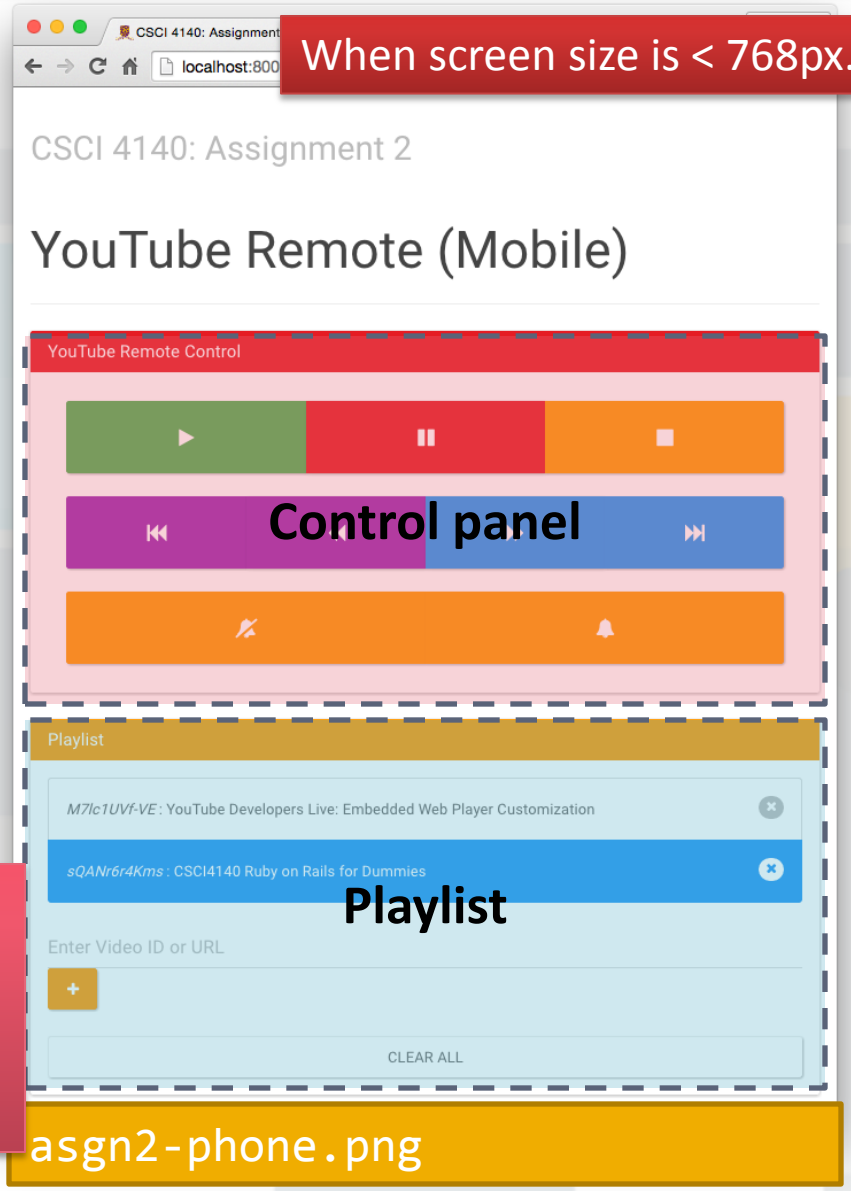
When screen size is between 768px and 992 px…

The buttons are displayed with both icon and description.



asgn2-tablet.png

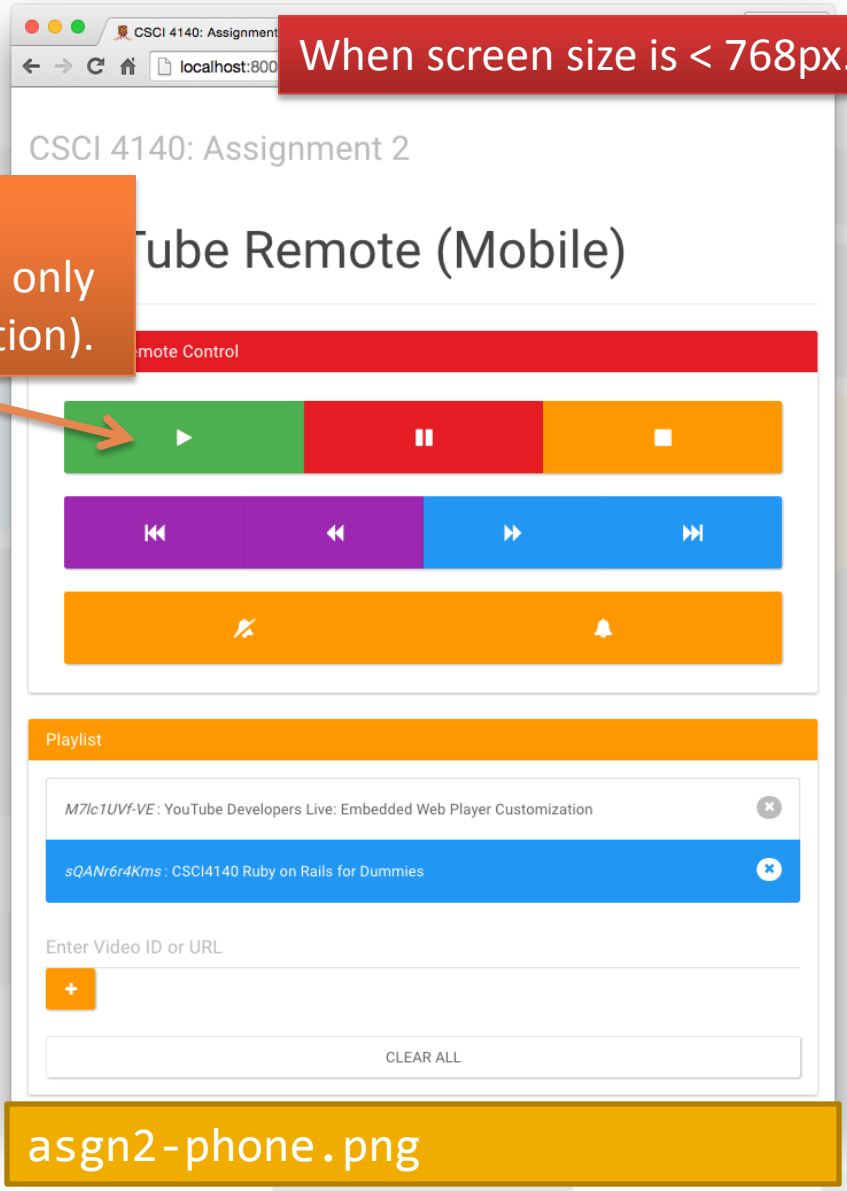

When screen size is < 768px...
CSCI 4140: Assignment 2
YouTube Remote (Mobile)
YouTube Remote Control
Playlist
M7lc1UVf-VE : YouTube Developers Live: Embedded Web Player Customization
sQANr6r4Kms : CSCI4140 Ruby on Rails for Dummies
Enter Video ID or URL
+
CLEAR ALL
asgn2-phone.png



Note: When we reduce the width of the window, the video player is hidden.

When screen size is < 768px...

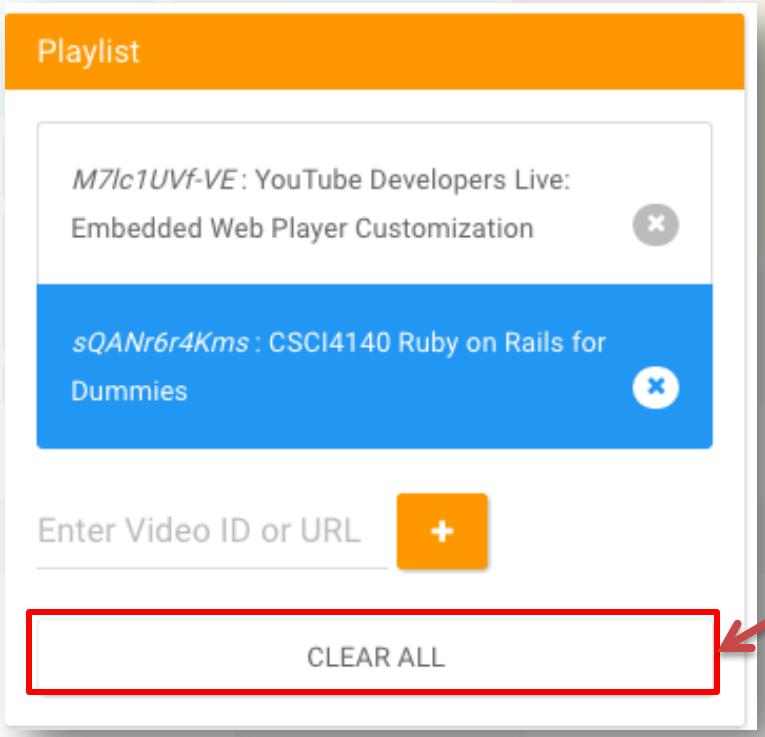
The buttons are displayed with icon only (**WITHOUT** description).



asgn2-phone.png

Client side: Layout design

- If you already implemented the layout by following the screen capture from last tutorial slides, please add a "**Clear All**" button for the playlist:



Sorry…this button is missing in last tutorial slides… Please add it back.

Client side: Layout design

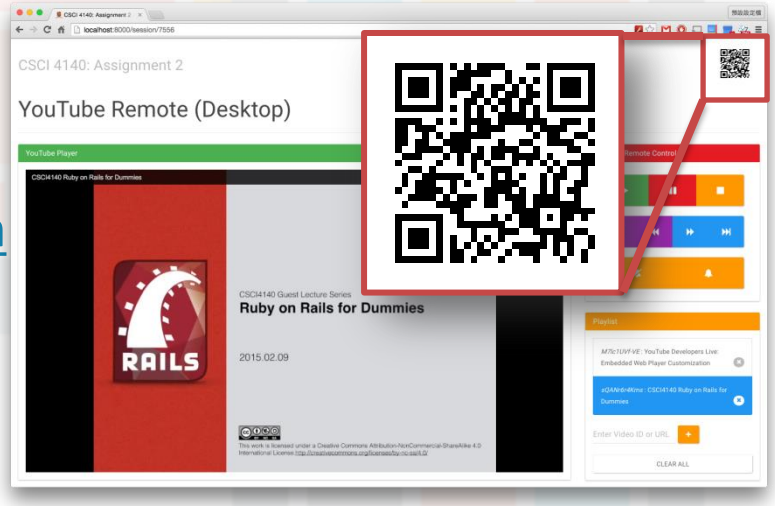
- You are asked to implement a **responsive UI**
- We will resize the window **WITHOUT refreshing the page**, so you are forced to do the screen width detection in **client side**
  - Suggested solution: Bootstrap / CSS media queries
  - Not recommended: JavaScript
- You are free to rearrange the components, but they must meet the requirements in the specification
- Read the slides from last tutorial if you didn't come

Note: If you want to clone the UI of my site, you can download the high quality screen capture from the resource page ☺

Client side: QR code display

Client side: QR code display

- You need to display a **QR code** on the page
- The QR code should only contain the URL of the current page
- You can generate the QR code on server side or client side
- Use "`location.href`" to get the URL of the current page in JavaScript
- Use **Google Chart** to generate QR code:
  - https://google-developers.appspot.com/chart/infographics/doc



Client side: QR code display

- Google Chart Example:

```
<html>
<head><title>QR Code Demo</title></head>
<body>
    <img
src="https://chart.googleapis.com/chart?cht=
qr&chs=500x500&chl=http%3A%2F%2Fwww.cse.cuhk
.edu.hk%2F"/>
</body>
```

qr_code/qr_code.html

Client side: QR code display

- Google Chart Example:

```
<html>
<head><title>QR Code Demo</t
<body>
    <img
src="https://chart.googleapis.com/chart?cht=
qr&chs=500x500&chl=http%3A%2F%2Fwww.cse.cuhk
.edu.hk%2F"/>
</body>
```

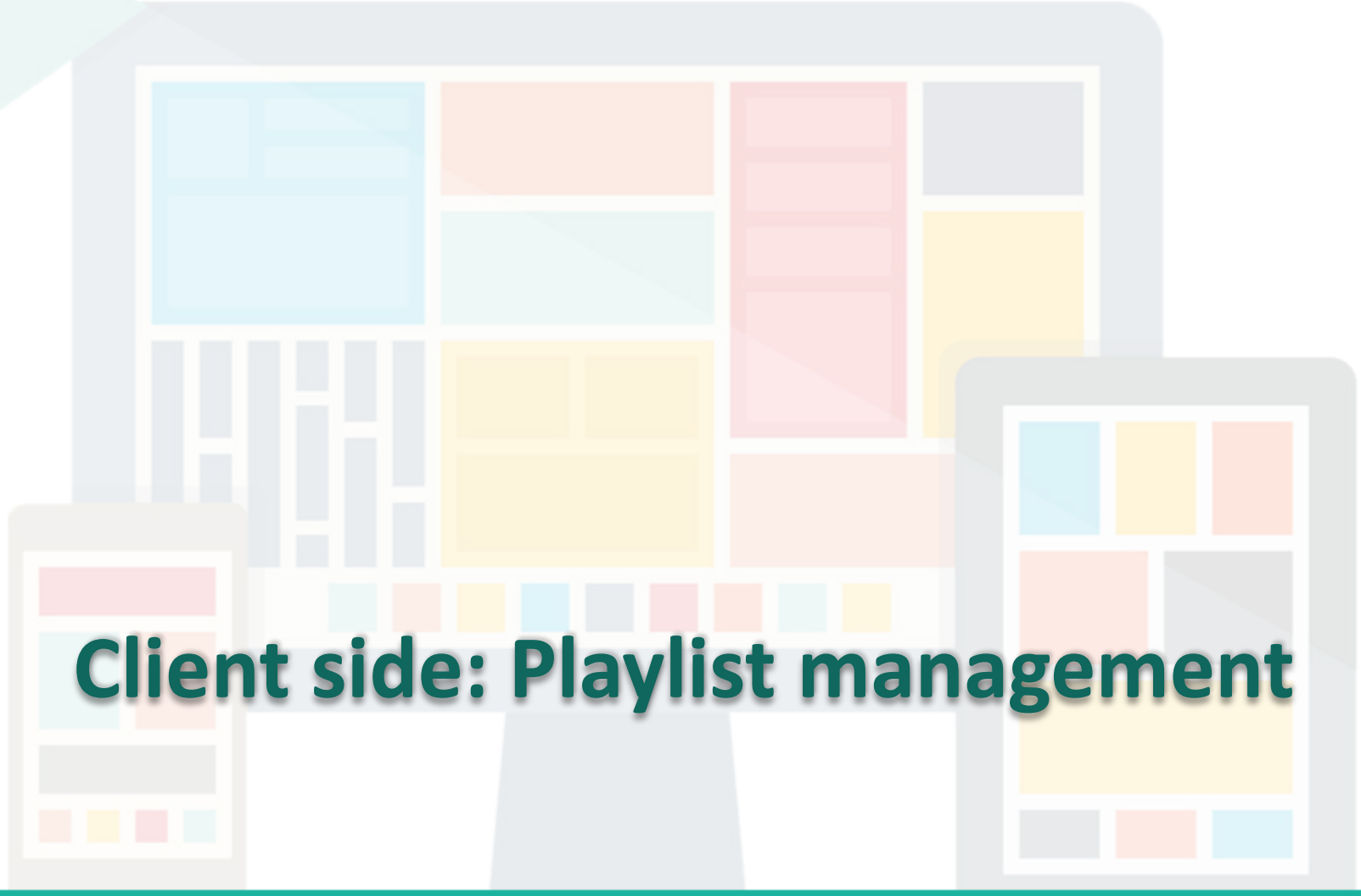
qr_code/qr_code.html

Specify a QR code.

Specify the image size.(cht=<*width*>x<*height*>)

The data to encode. If you need to specify a URL, it should be UTF-8 URL-encoded. The JavaScript `encodeURI()` function can do so.
Ref.: http://www.w3schools.com/jsref/jsref_encodeuri.asp

- How to combine it with "`location.href`"?
  - **Hint:** Generate the **`<img>`** tag in JavaScript (e.g. by **`document.write`**)



Client side: Playlist management

Client side: Playlist management

- You need to manage a playlist in the application
- Operations supported:
  - Add new video (with its video ID) to the playlist ***(1)***
  - Remove a existing video from the playlist ***(2)***
  - Clear all videos in the playlist ***(3)***
- Of course, the playlist should be displayed in your web page
- Suggested format:

```
<Video ID>: <Video title>
```

Client side: Playlist management

- When a video is added to or removed from the playlist, the change should be reflected in the UI
  - Use **DOM scripting**
  - Useful functions:
    - `document.getElementById()`
    - `document.querySelector()` (in HTML5)
    - `createElement()`
    - `appendChild()`
    - `removeChild()`
  - Read the lecture notes: “JavaScript (Part 1)”

Client side: Playlist management

- The playlist does not need to be stored on server side
- Storing it on client side is sufficient
  - But, let's forget about cookies
- Use HTML5 **localStorage**:
  - To get an item with a key:

```
var i = localStorage.getItem( <key> );
```

  - To set an item with a key

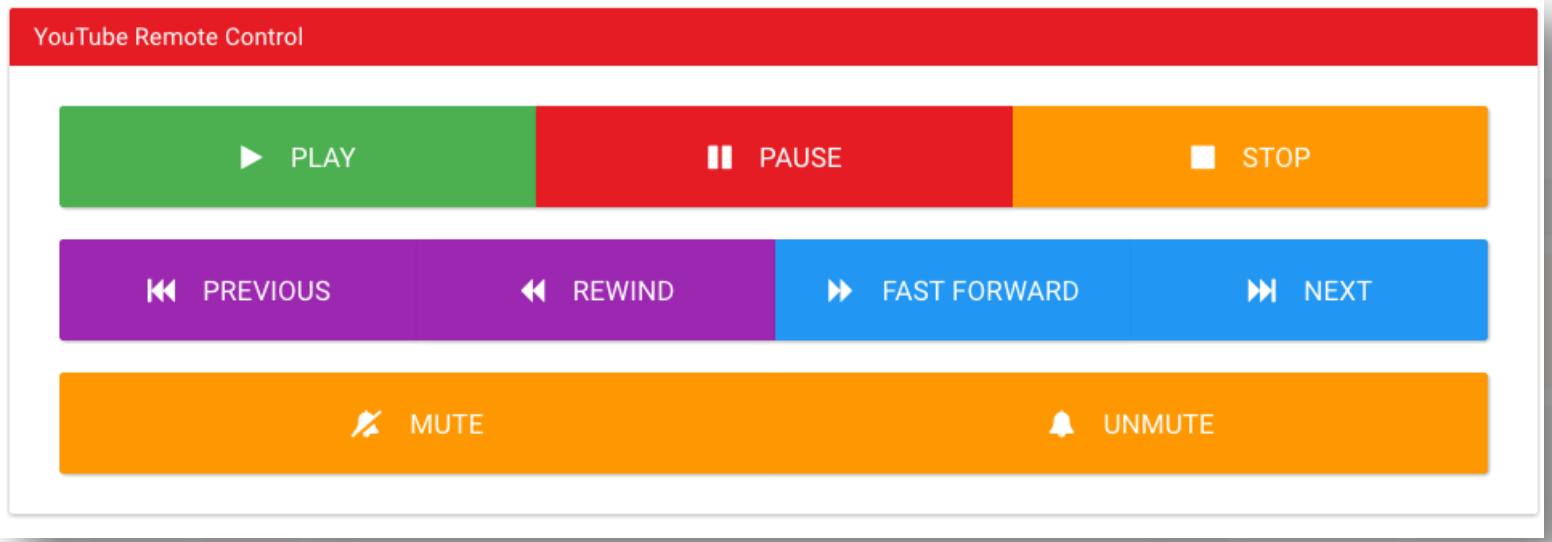
```
localStorage.setItem( <key>, <item> );
```

  - Note: The data type of the item should be **String**
    - Use "**JSON.stringify()**" to convert an array / object to a String

Client side: YouTube player control

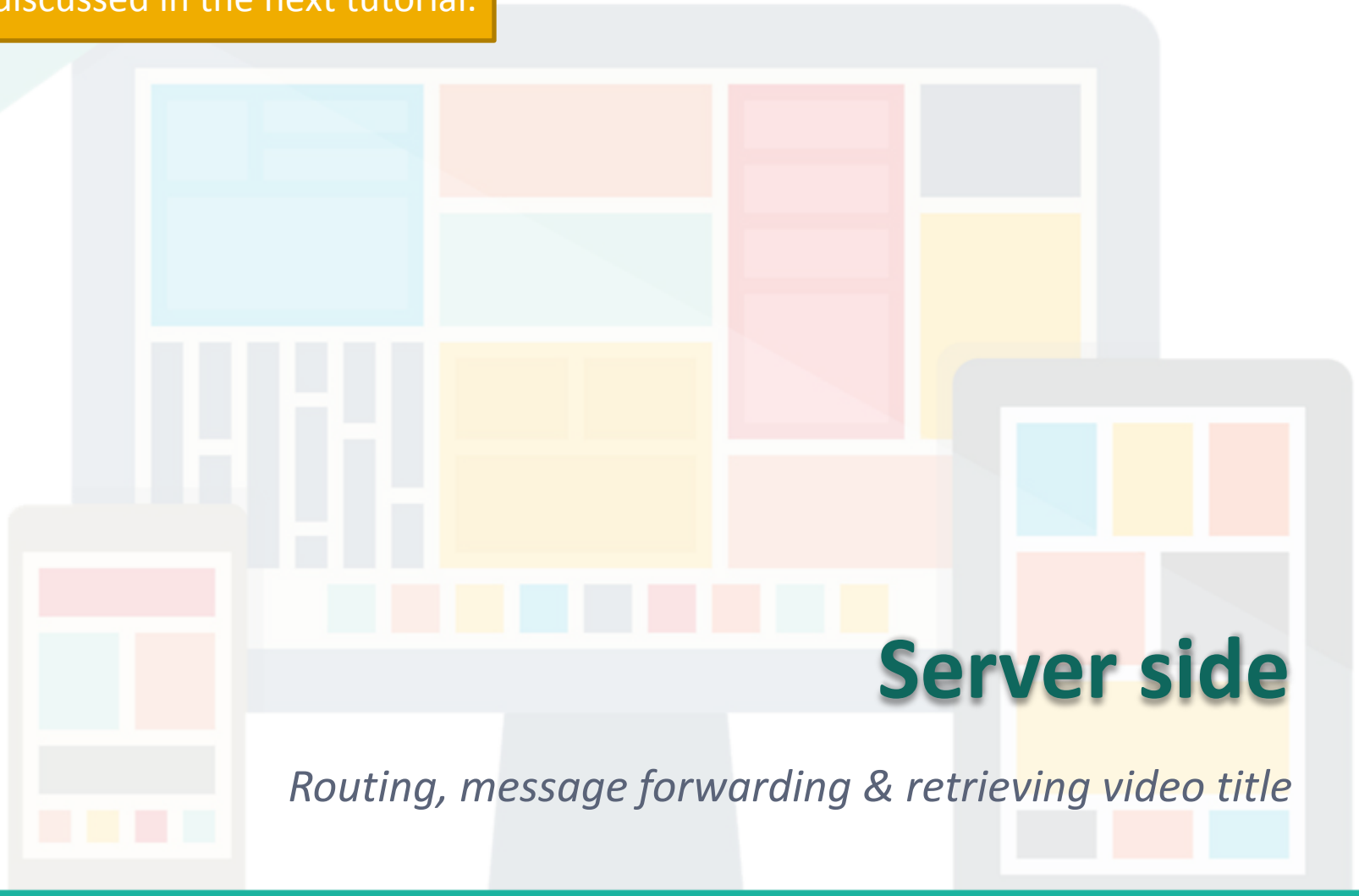
Client side: YouTube player control

- You need to implement the following functions with **YouTube IFrame Player API**



- Read the corresponding tutorial slides for more details

Note: Server-side implementation will be discussed in the next tutorial.



Server side

Routing, message forwarding & retrieving video title

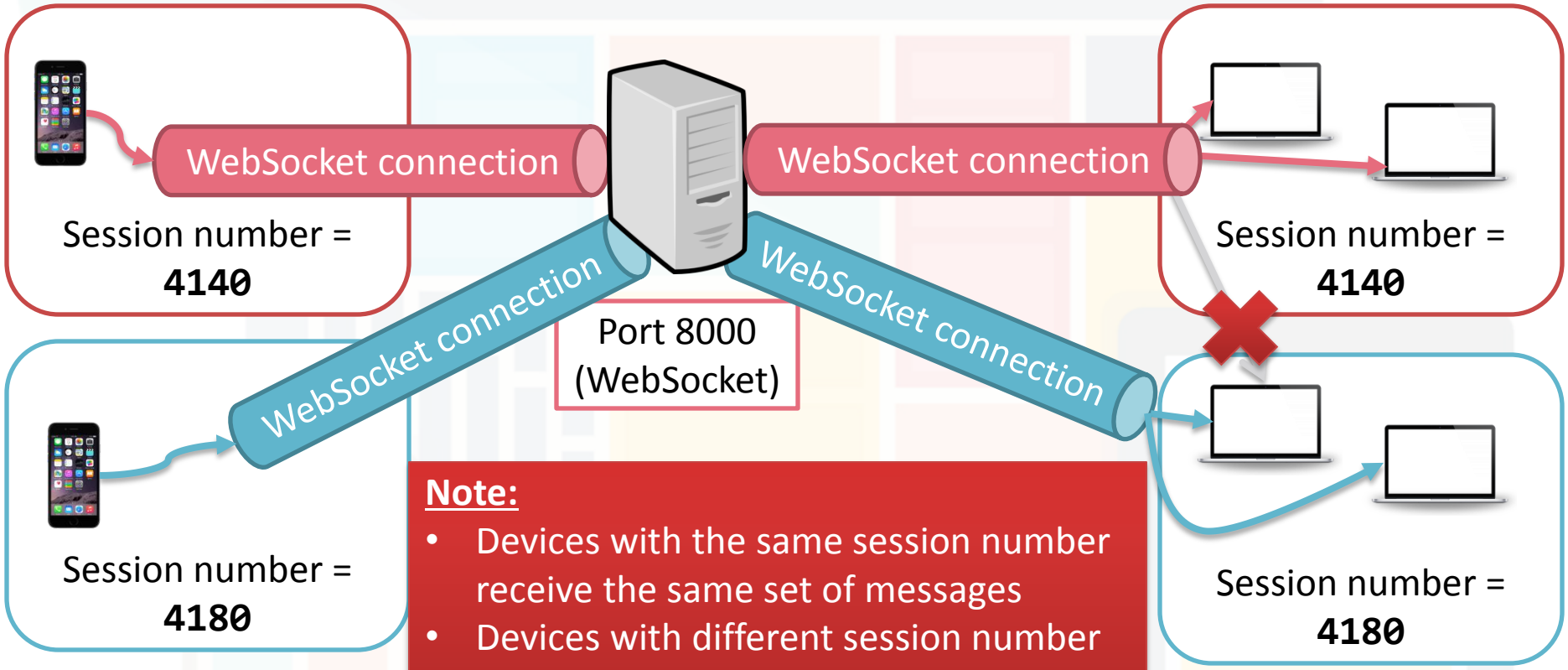
Server side: Routing

Client enters the page without session number
(e.g., `http://youtube-XXX.rhcloud.com/`)

Server generates a unique session number

Server redirects the client to the page representing this session
(e.g., `http://youtube-XXX.rhcloud.com/session/4140`)

Server side: Message forwarding



Note:

- Devices with the same session number receive the same set of messages
- Devices with different session number never receive the same message
- Different sessions work independent to each other

Server side: Retrieving video title

- You need to display the **video title** for each video in the playlist
- How to retrieve video title from an ID?

```
http://www.youtube.com/oembed?
url=http://www.youtube.com/watch?v=sQANr6r4Kms
```

Put the video ID here

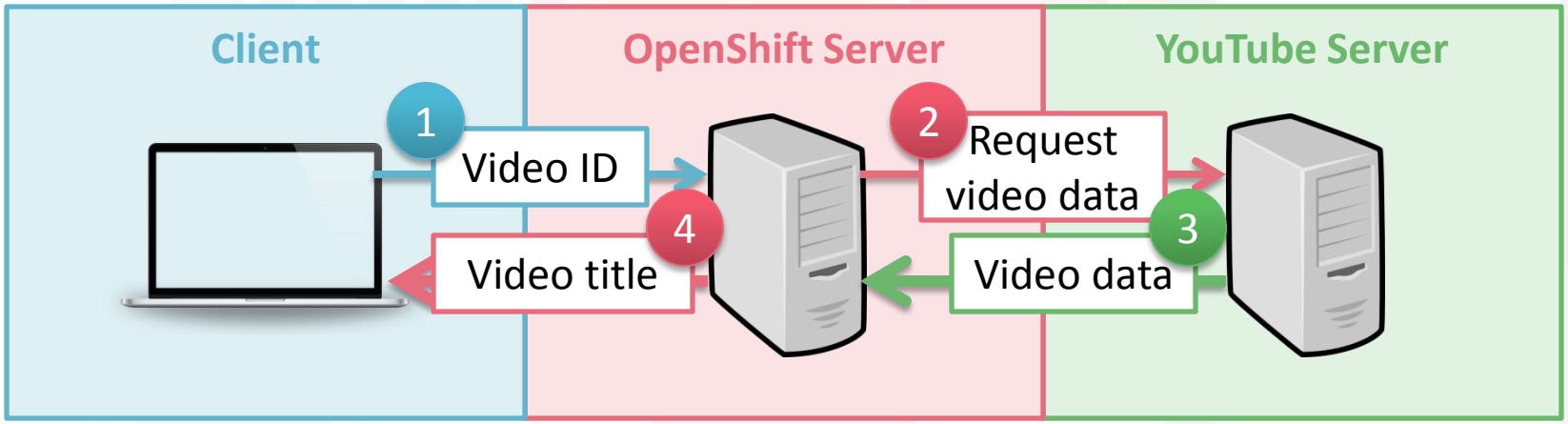
  - The server returns a **JSON object**
    - The video title is returned inside the "**title**" field

```
{"title": "CSCI4140 Ruby on Rails for Dummies",
"height": 344, "width": 459, ...
```

  - Can we use AJAX to get this JSON object directly from YouTube?
    - No because of the **same-origin policy (SOP)**
    - How to bypass this?

Server side: Retrieving video title

1. **Client** send a request with the video ID to **OpenShift server**
2. **OpenShift server** forwards the request to **YouTube server**
3. **YouTube server** responds with the video data
4. **OpenShift server** replies the client with the video title



Reminder

- You should finish the **UI design**
- You should start implementing the **player** and **control logic**
- Implement all **playback control functions** which controls the video player on the same page
  - **Hint:** Wrap all YouTube IFrame API calls in your functions
    - This is useful for extending to support remote control
- **Next week:** Back-end development
  - We will use **Node.js** and **Socket.IO** in server-side
  - Make sure you have installed Node.js and Express on your computer
    - Instructions are available in last week's tutorial slides

– End –