

CSCI 4140 – Tutorial 6

Assignment 2 Overview

Hints on Front-end Development

Matt YIU, Man Tung ([mtyiucse](mailto:mtyiucse@gmail.com))

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

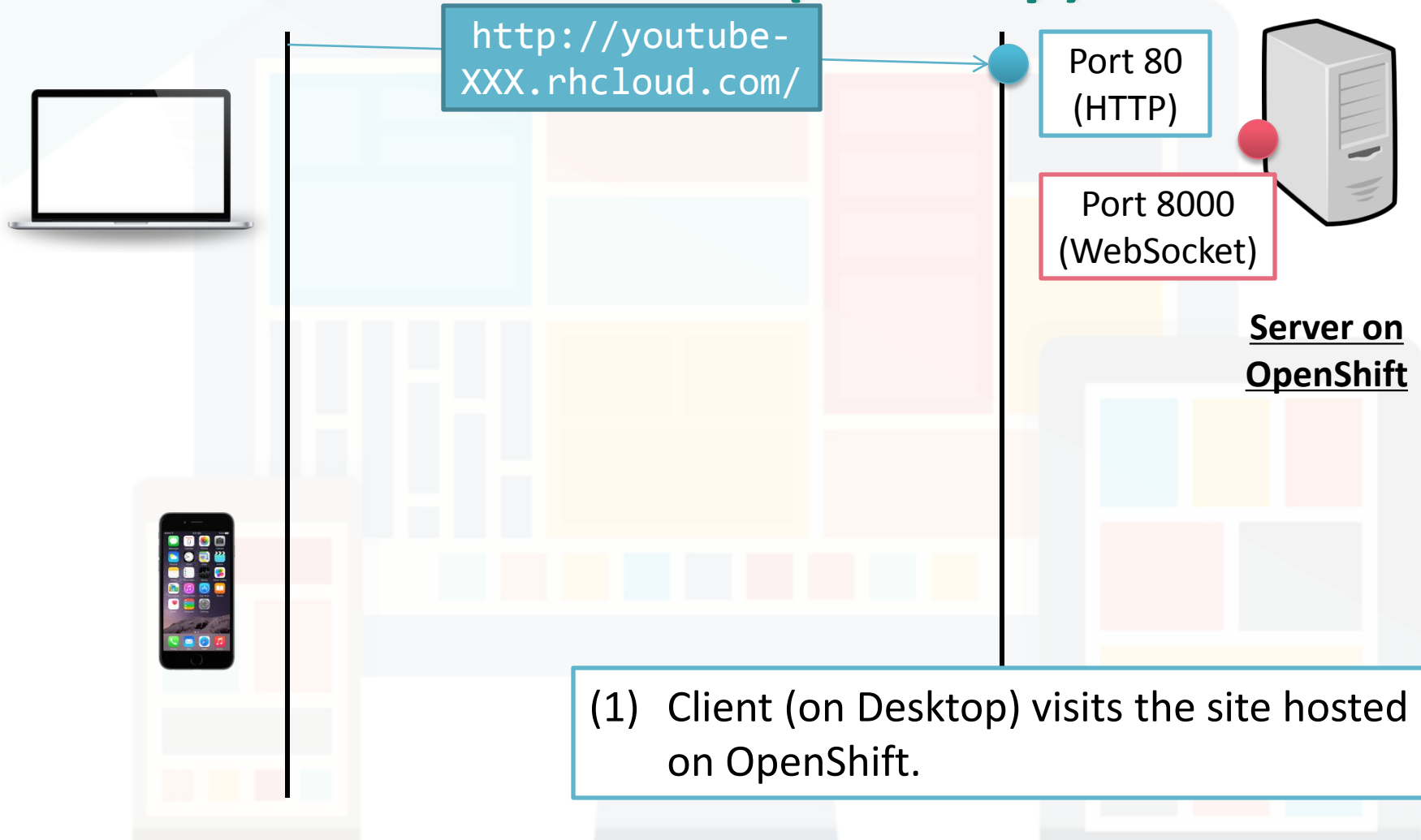




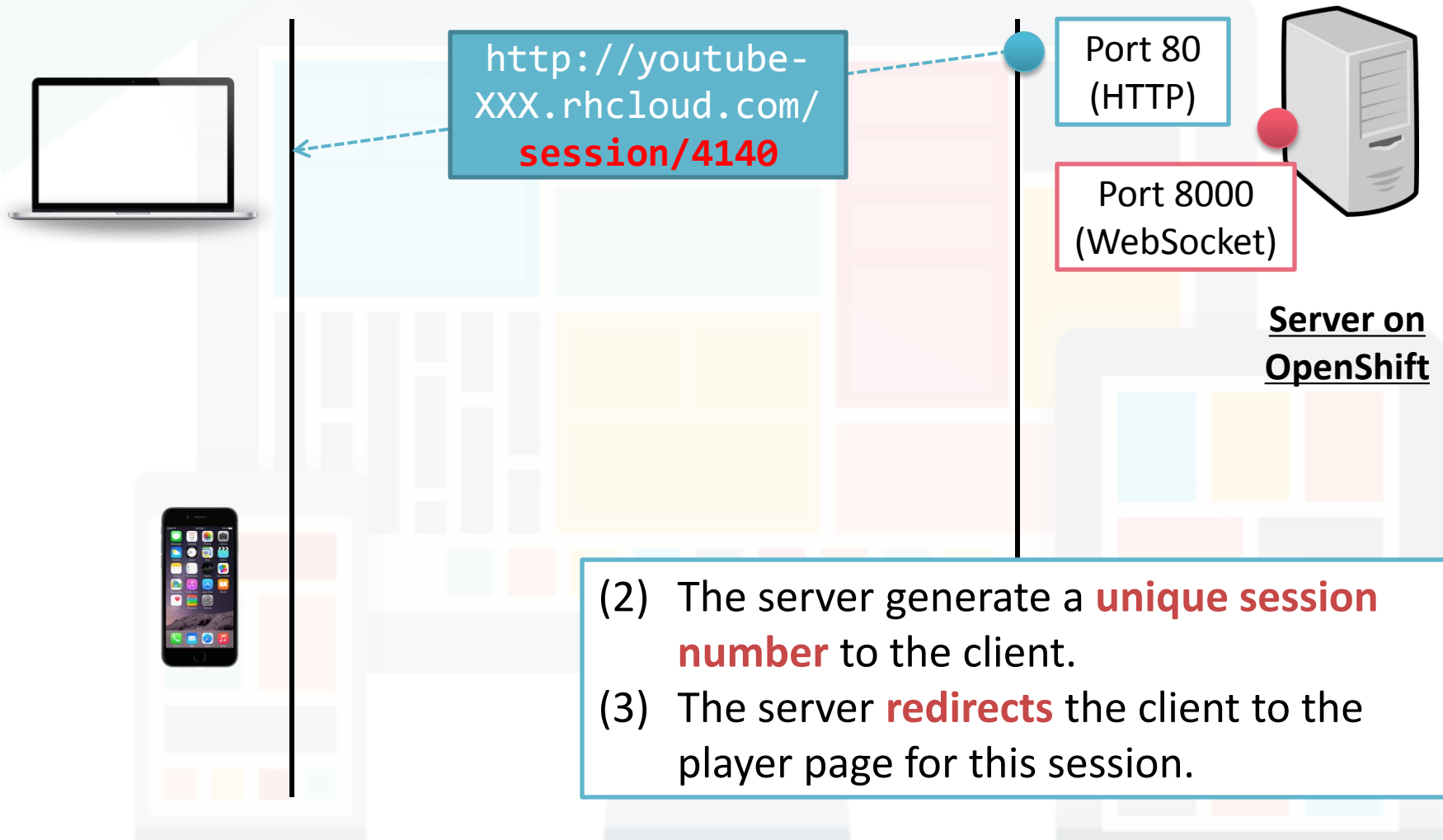
Work flow

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

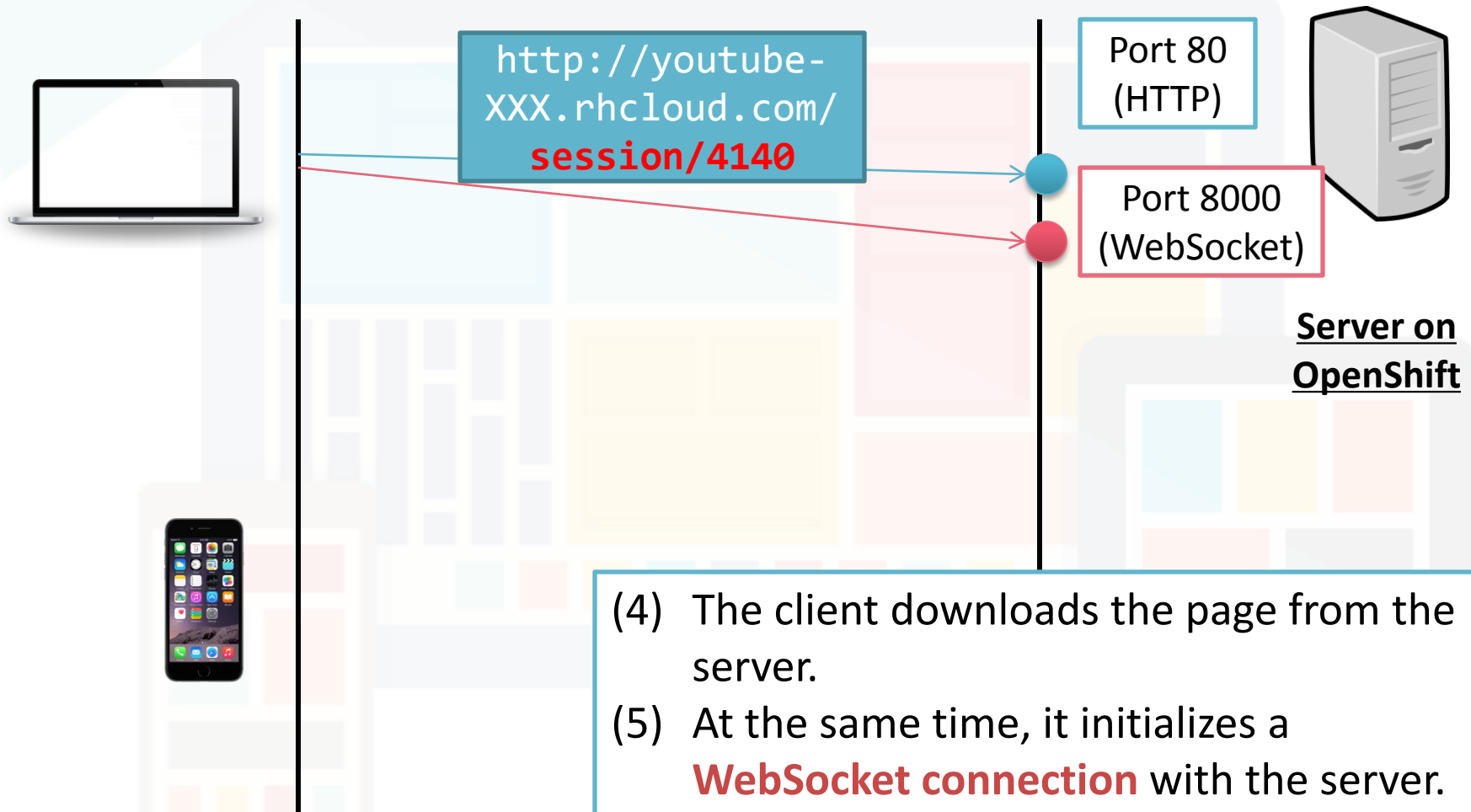
Work flow: Initialization (Desktop)



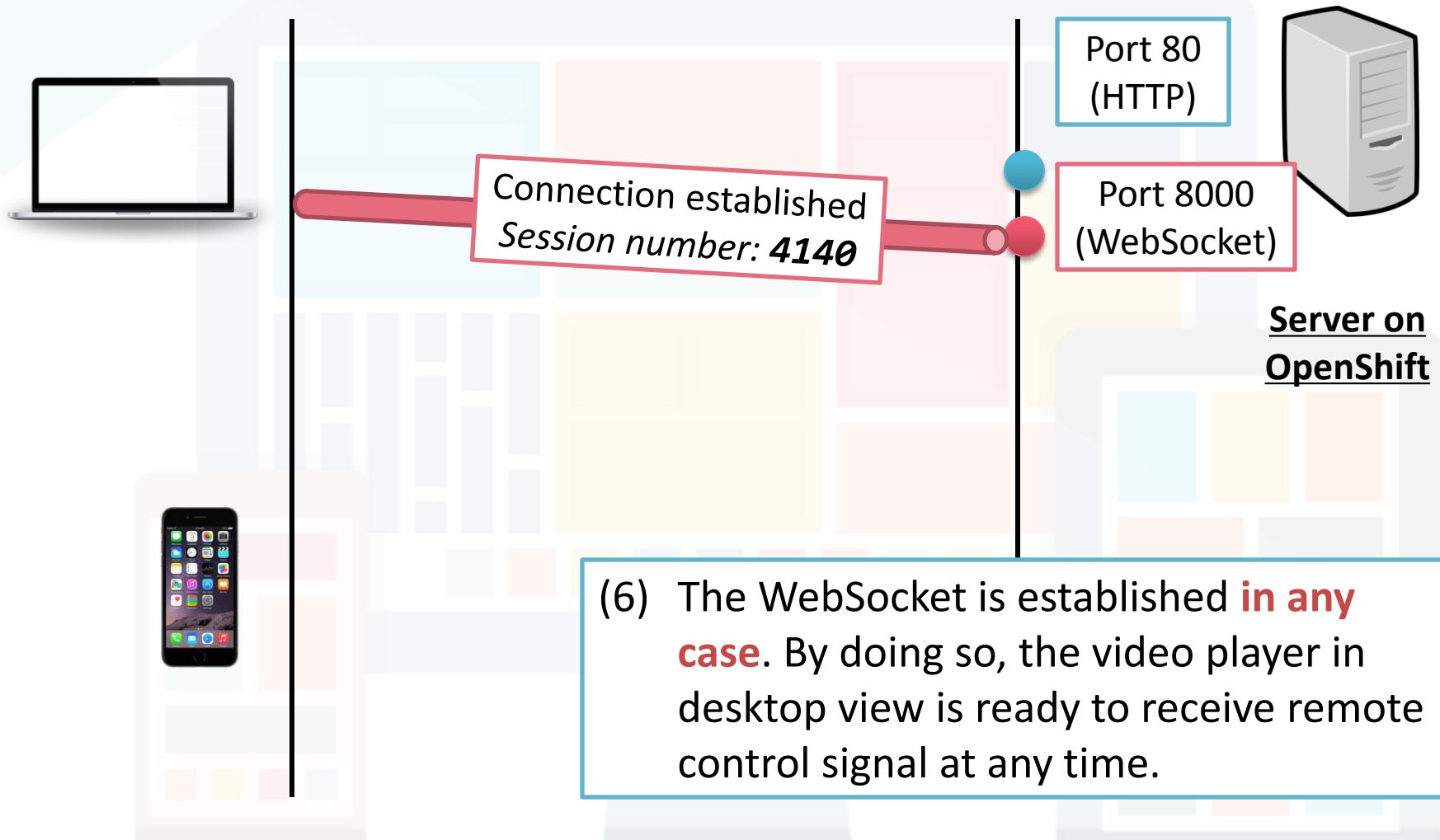
Work flow: Initialization (Desktop)



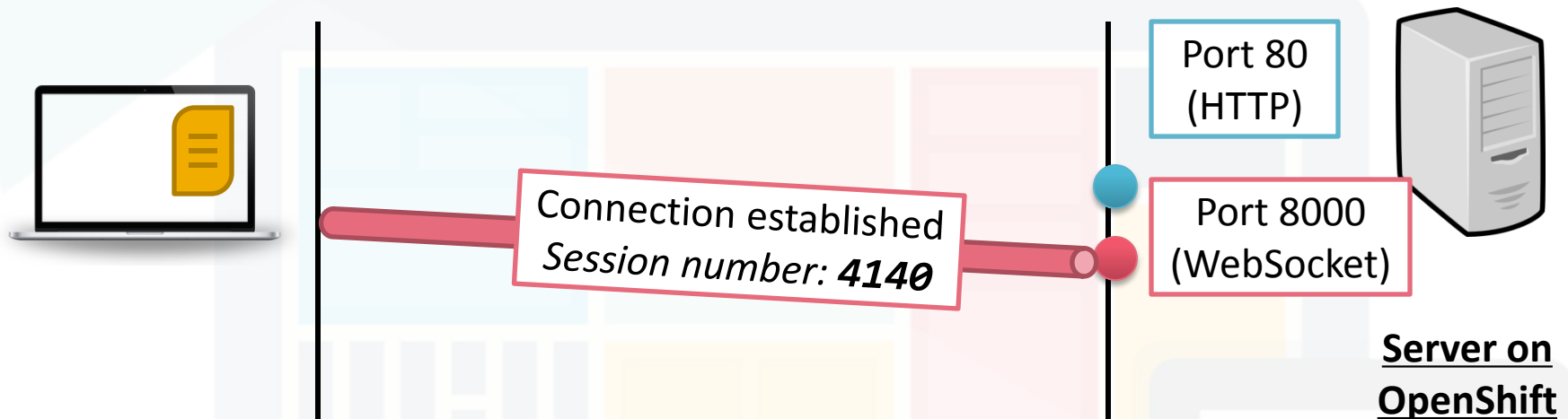
Work flow: Initialization (Desktop)



Work flow: Initialization (Desktop)



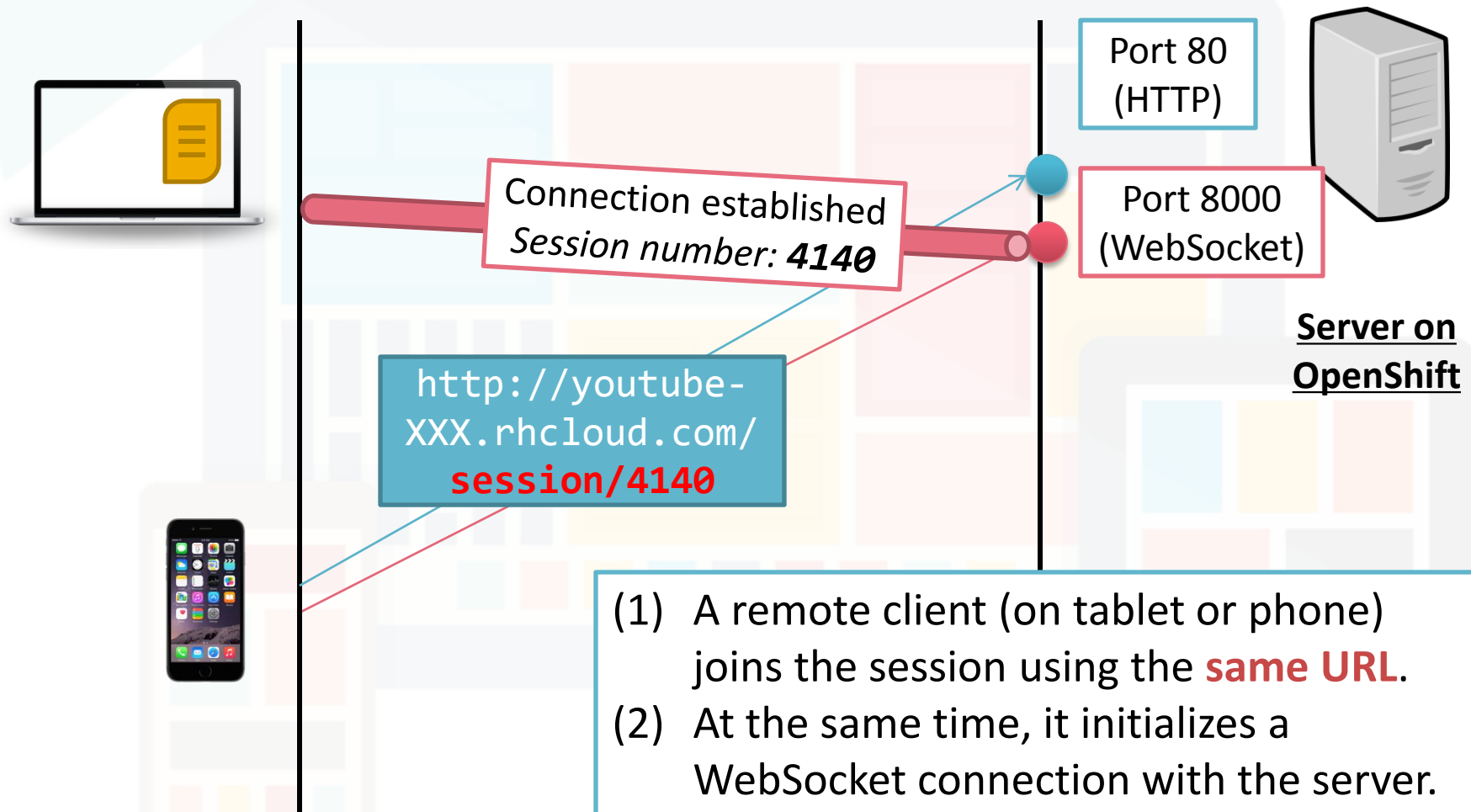
Work flow: Initialization (Desktop)



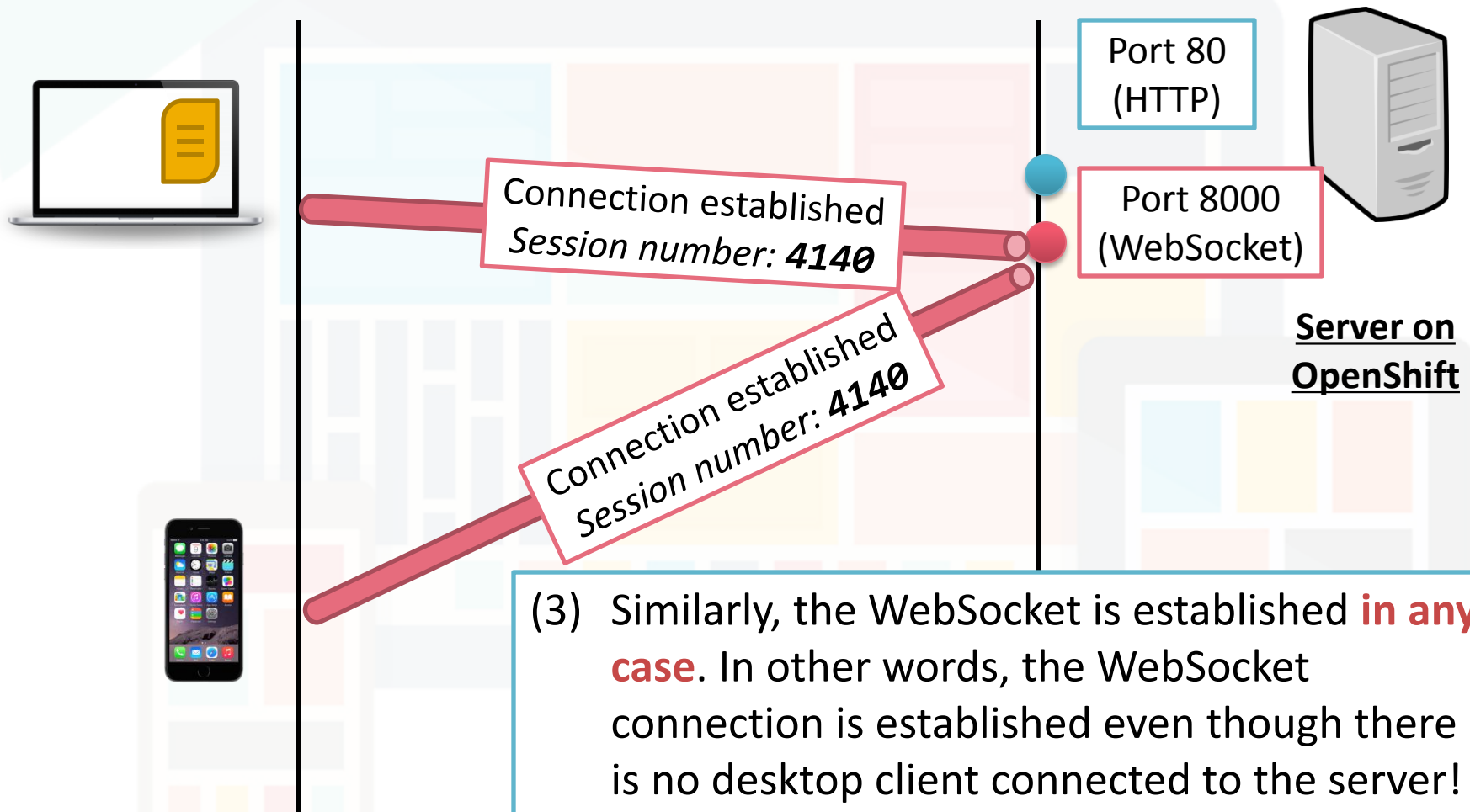
(7) The desktop client retrieves the playlist in the following order:

- **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**.

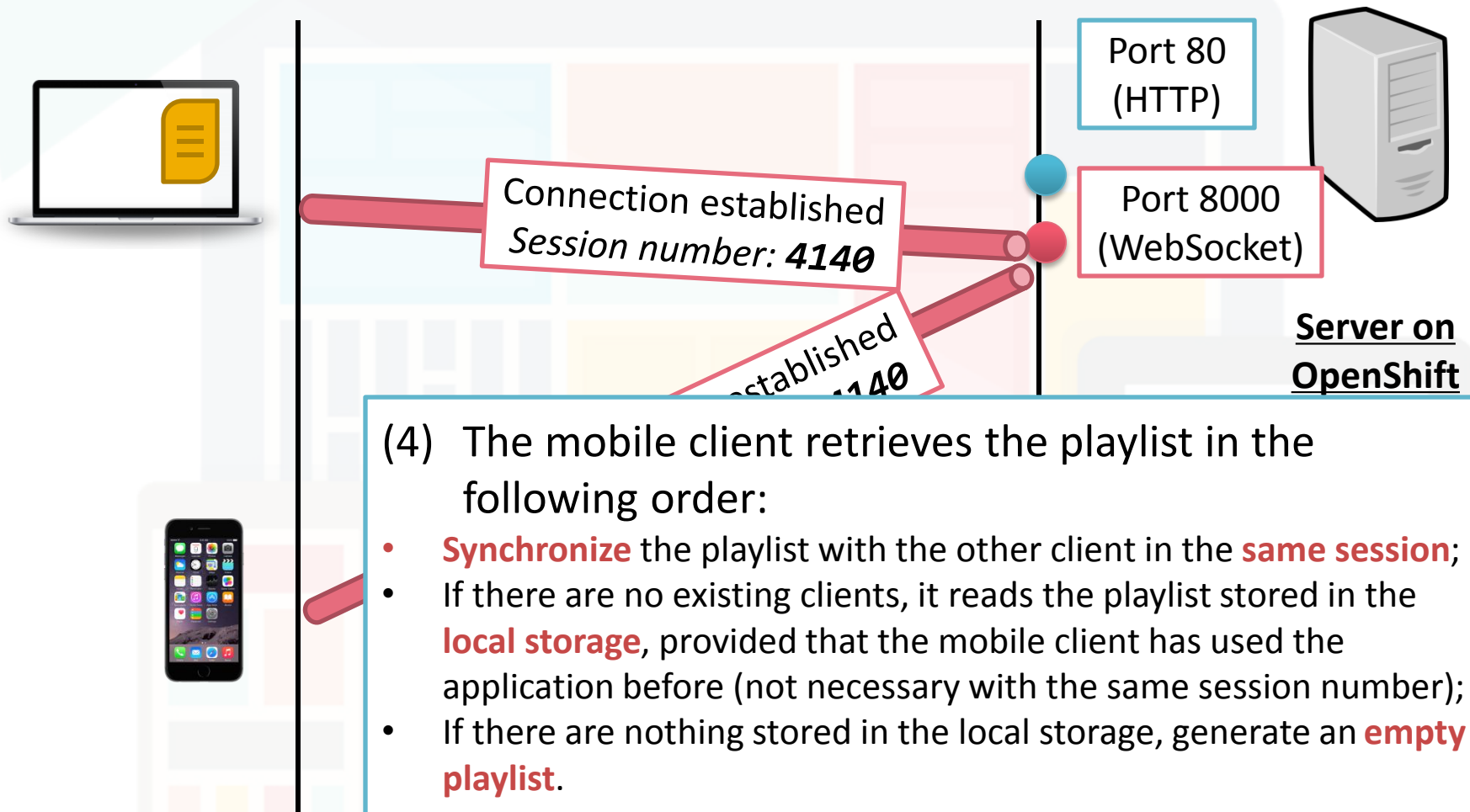
Work flow: Initialization (Tablet or phone)



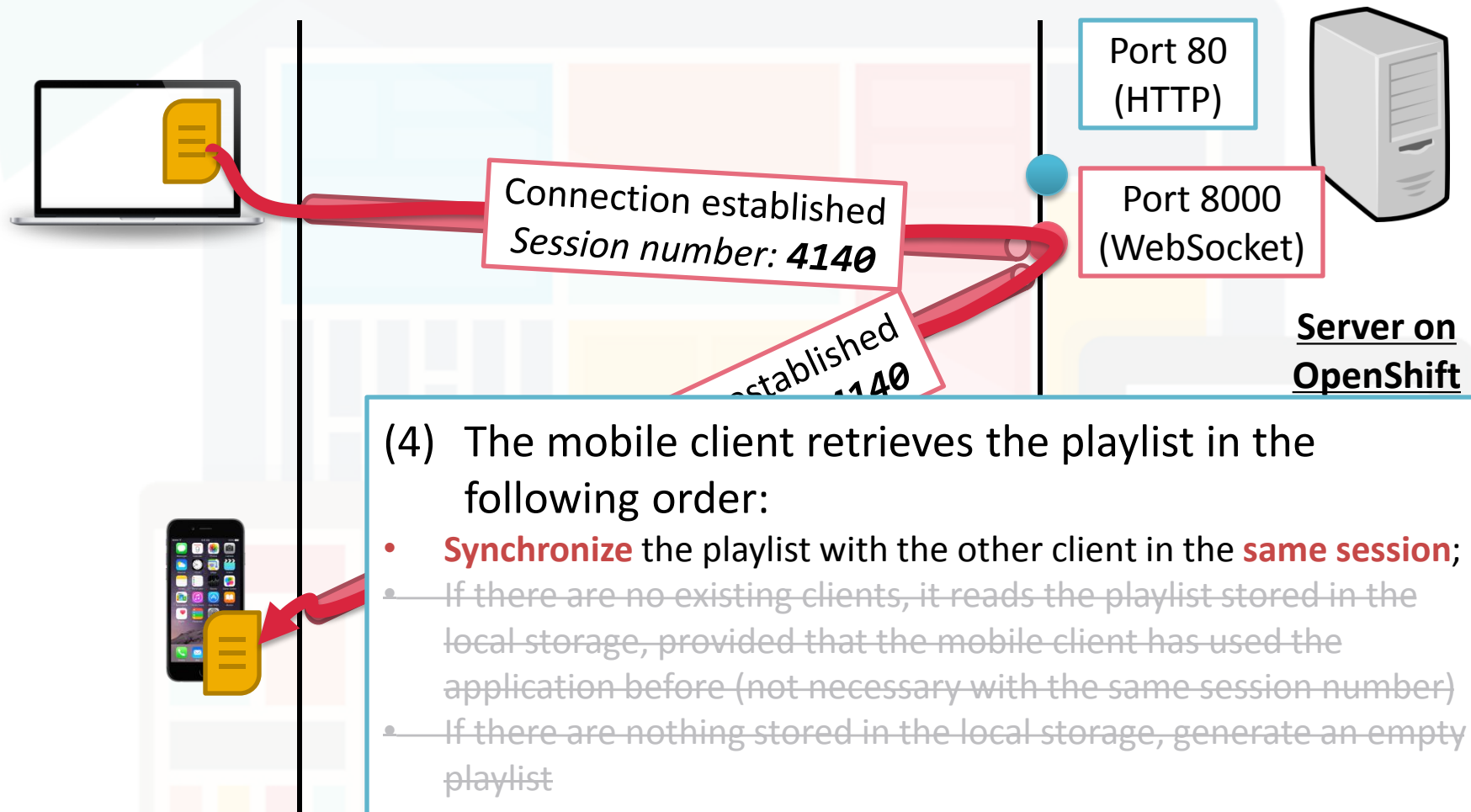
Work flow: Initialization (Tablet or phone)



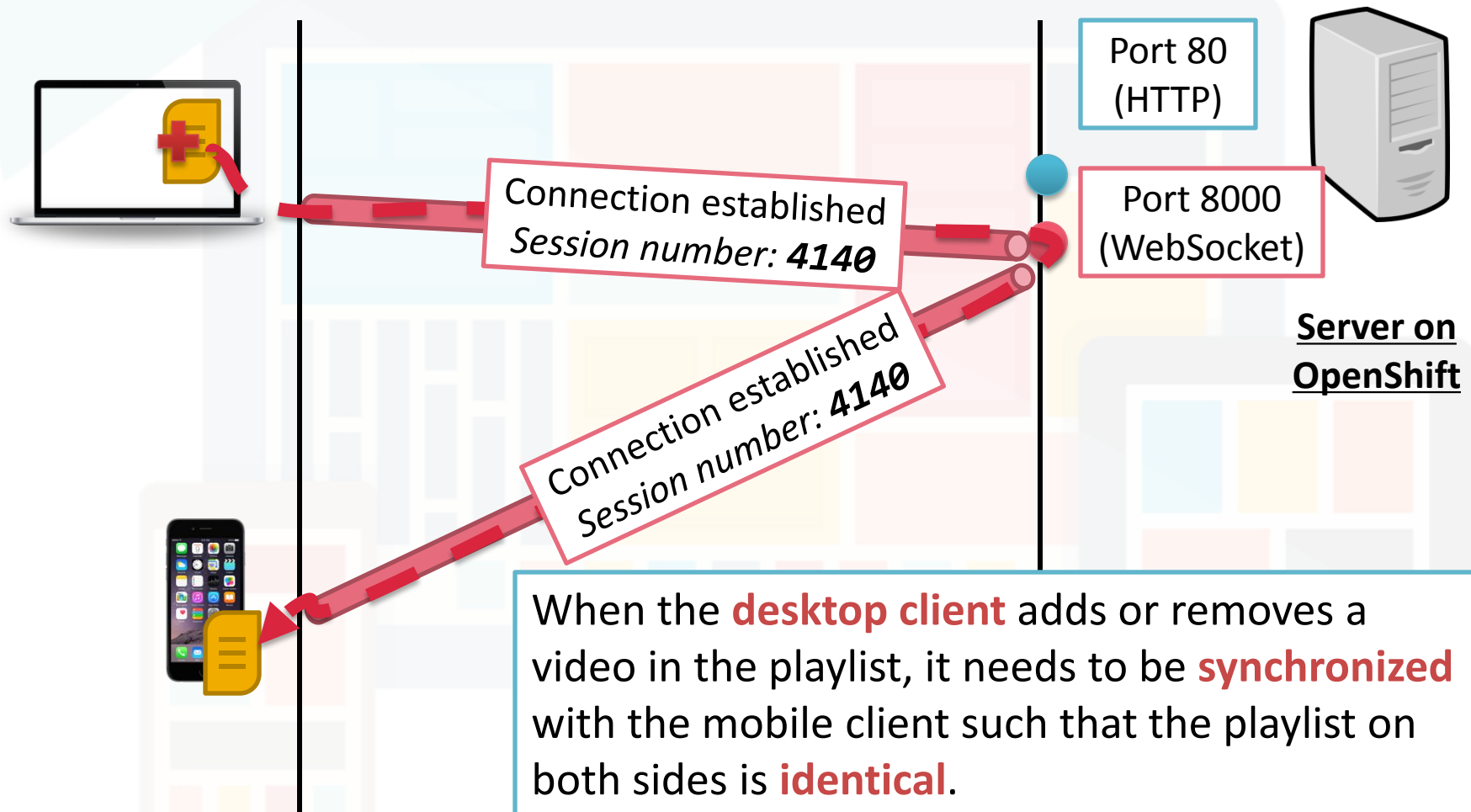
Work flow: Initialization (Tablet or phone)



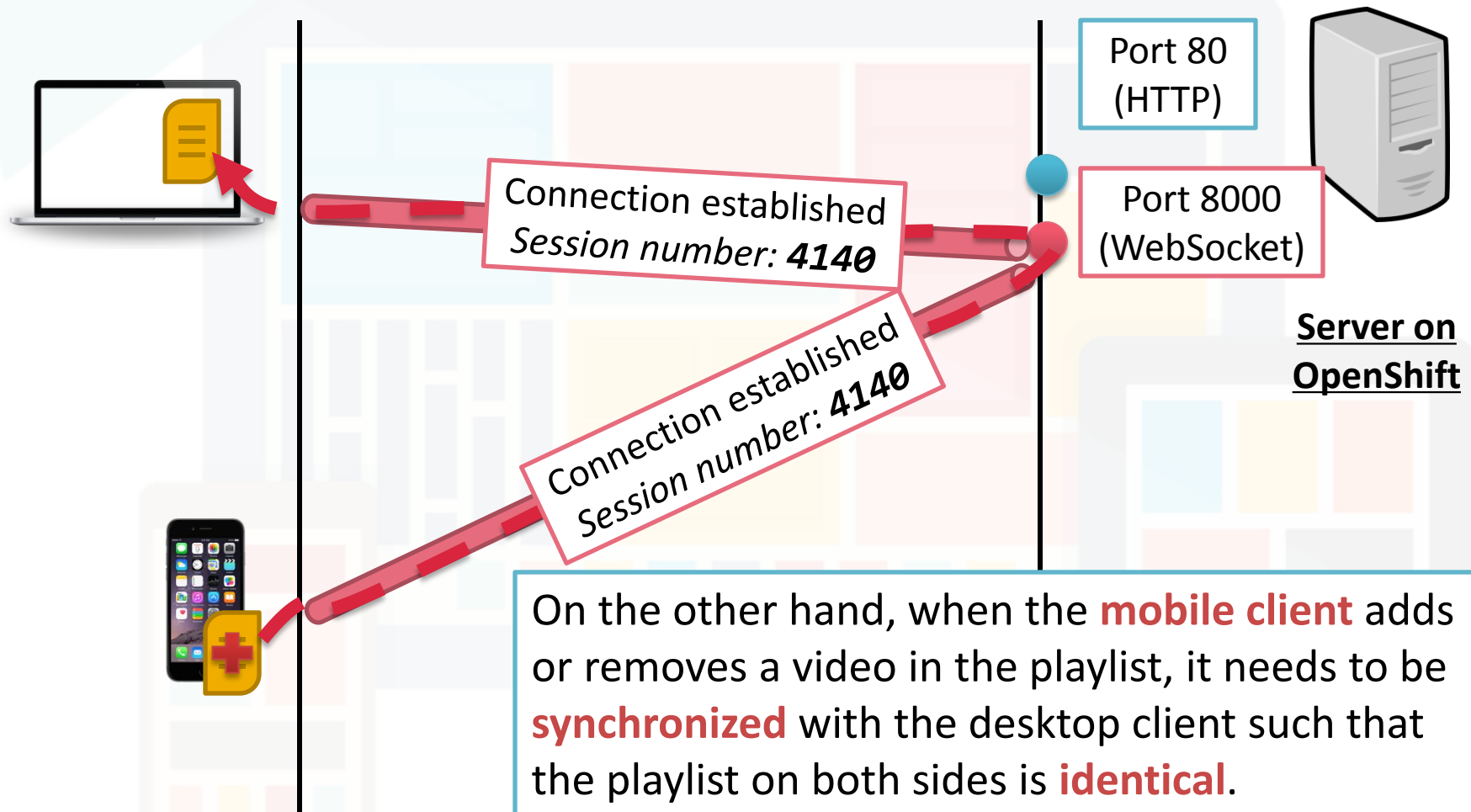
Work flow: Initialization (Tablet or phone)



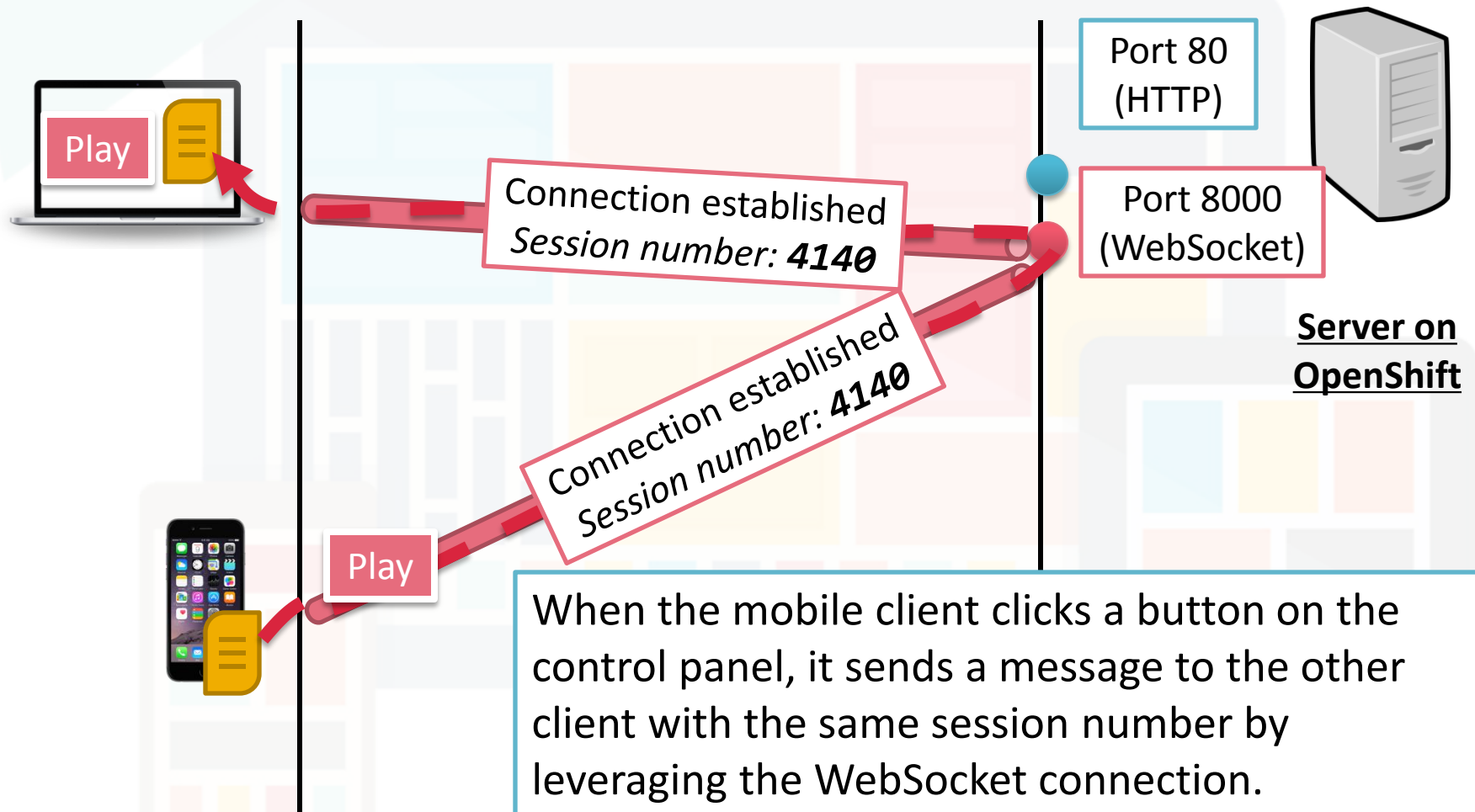
Work flow: Playlist synchronization



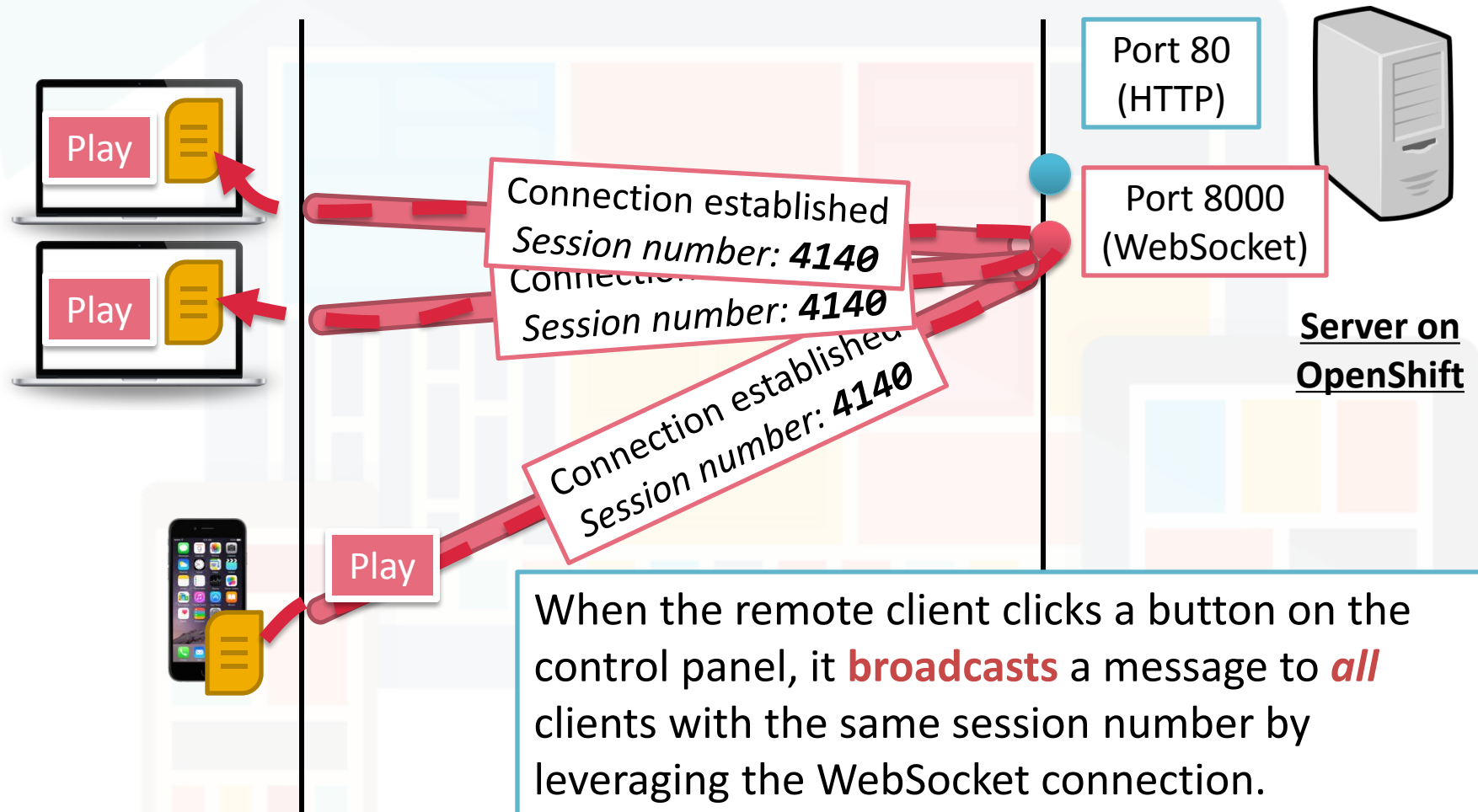
Work flow: Playlist synchronization



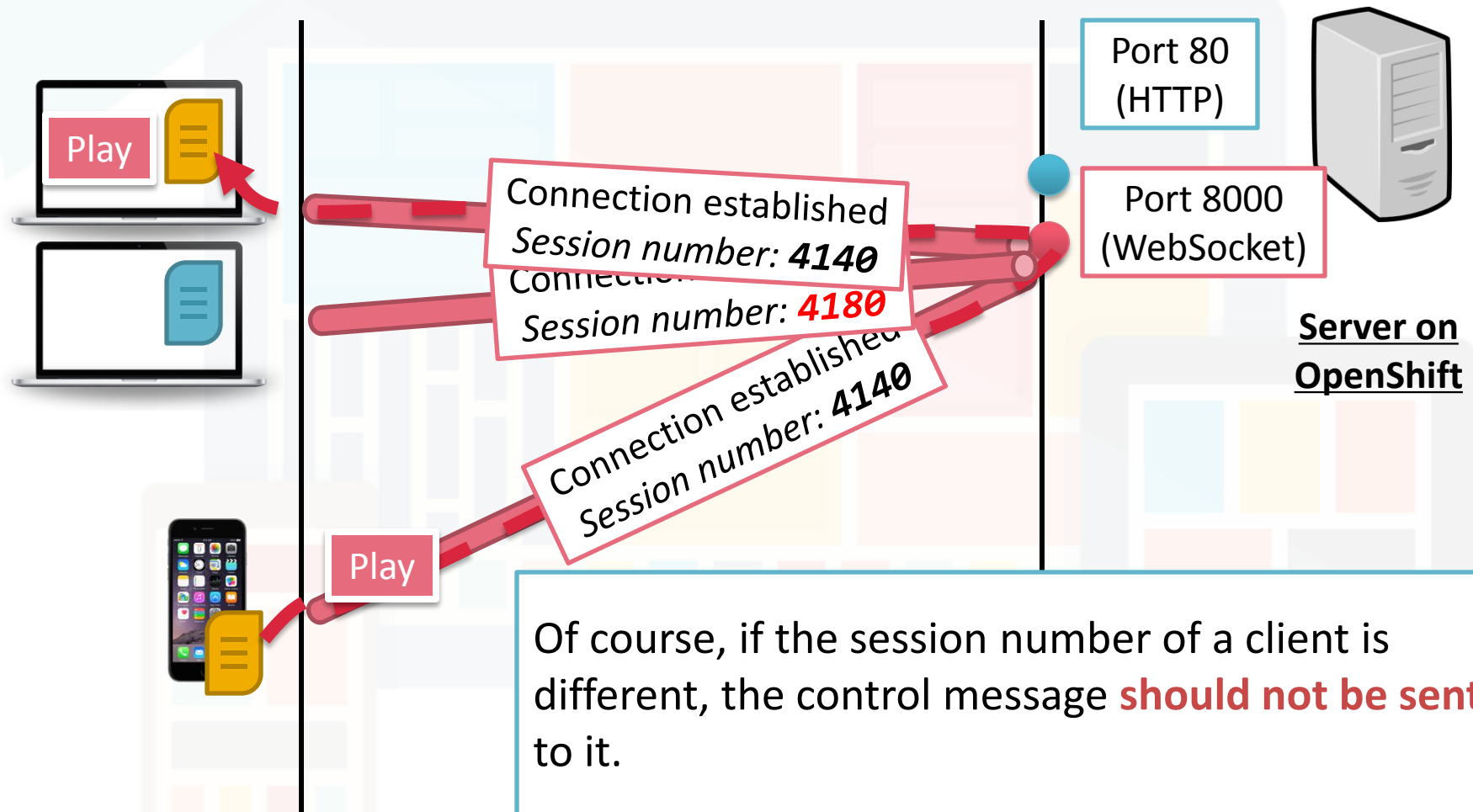
Work flow: Remote control



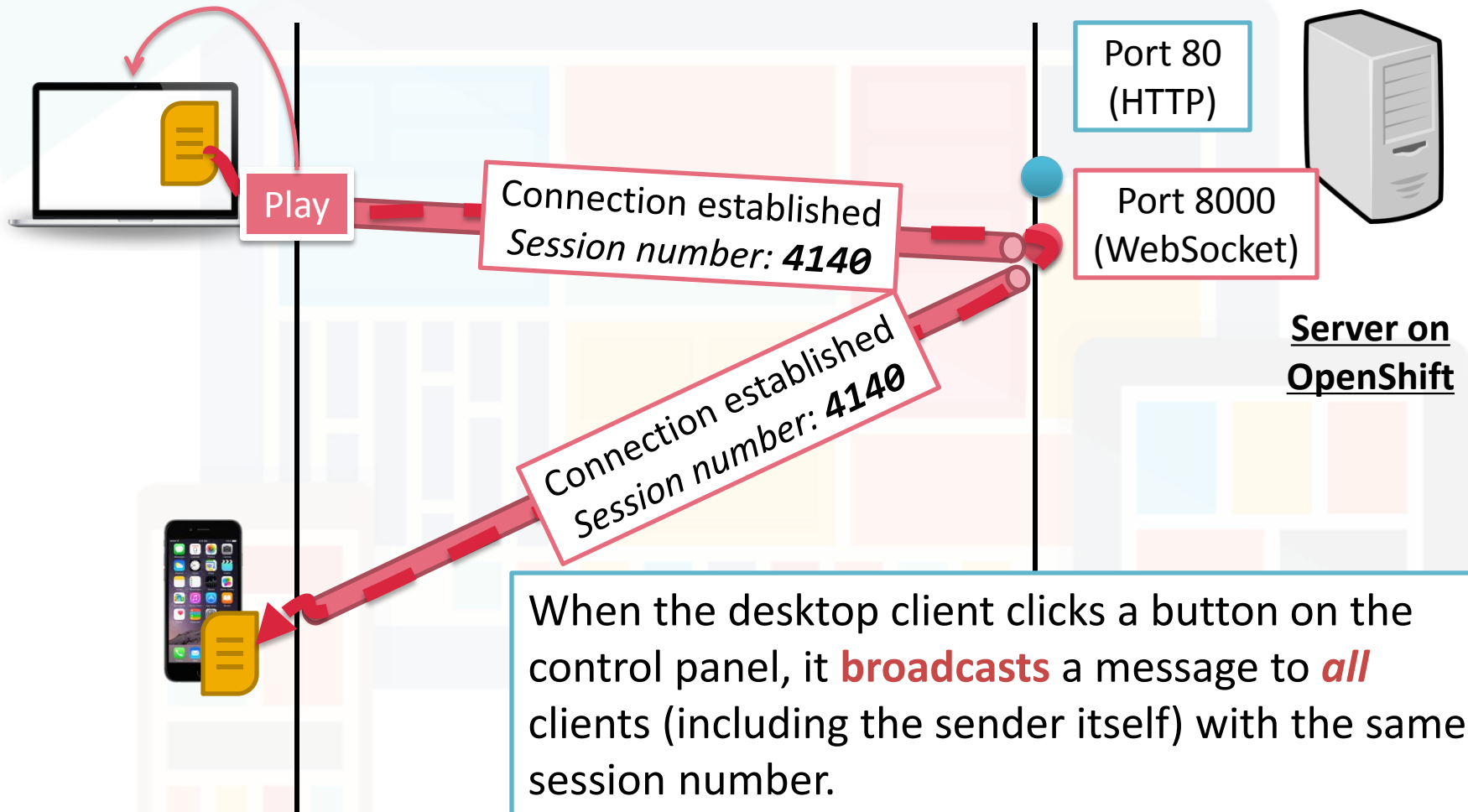
Work flow: Remote control



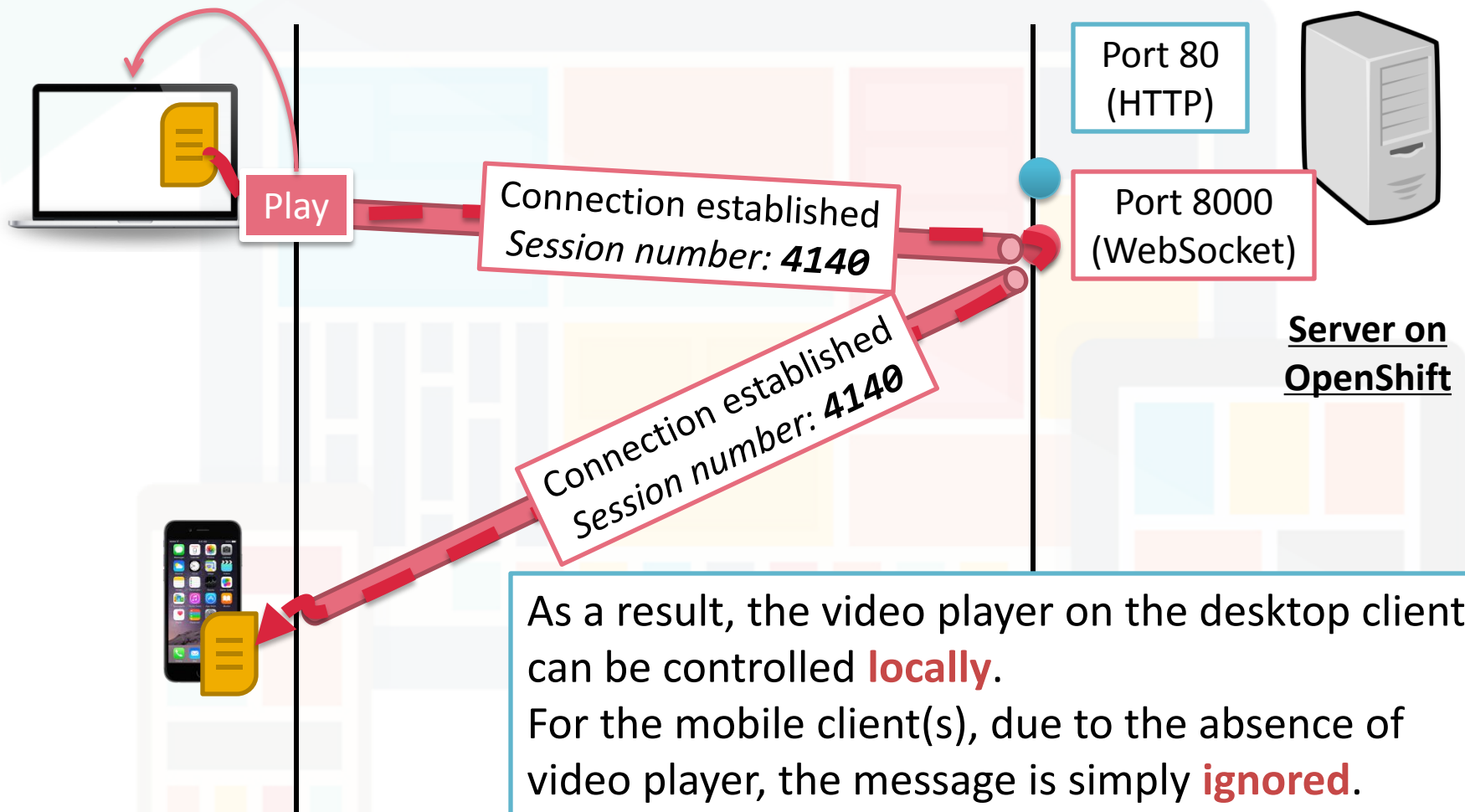
Work flow: Remote control



Work flow: Remote control



Work flow: Remote control





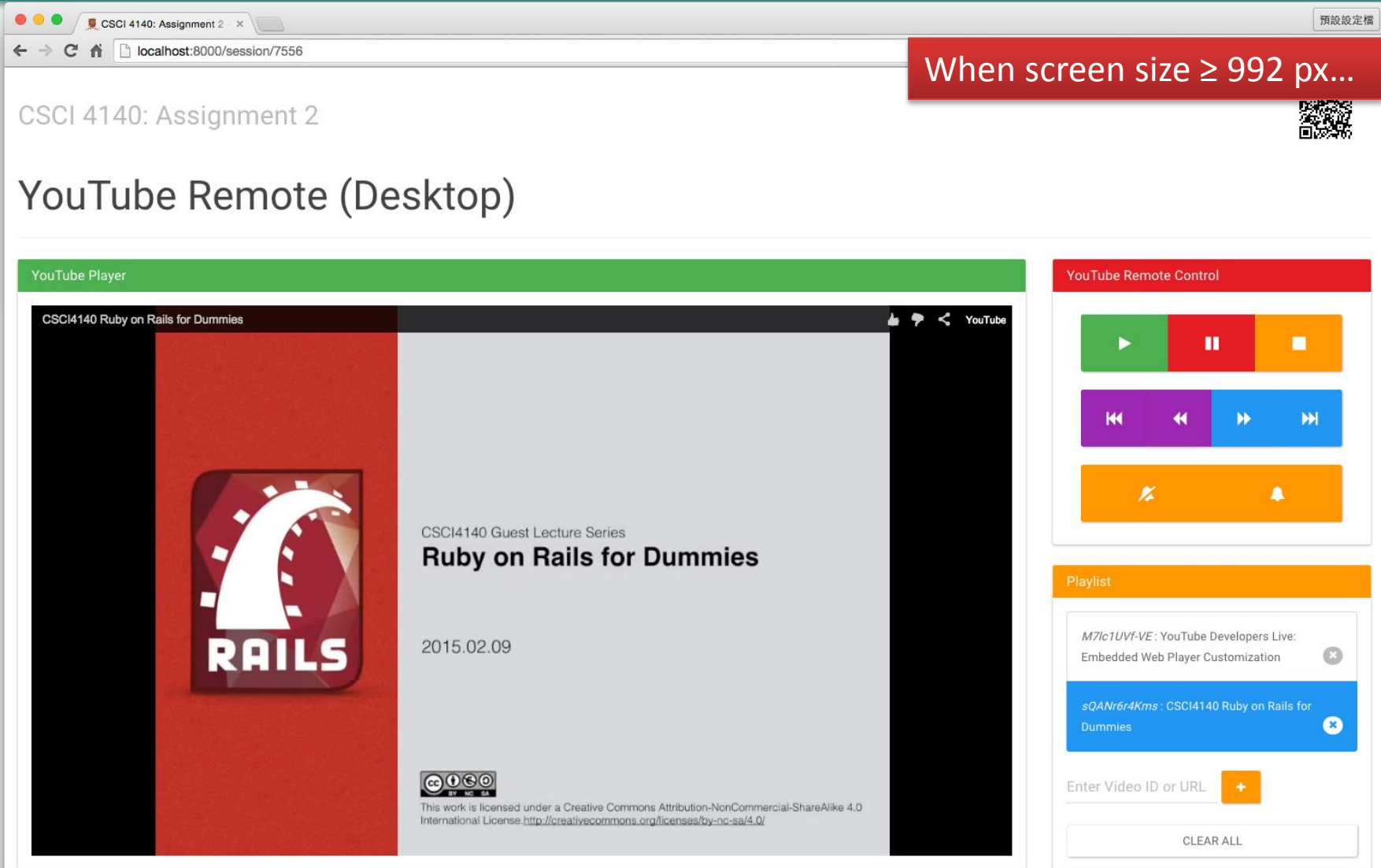
Client side: Layout design

Let's try responsive web design!

CSCI 4140: Assignment 2

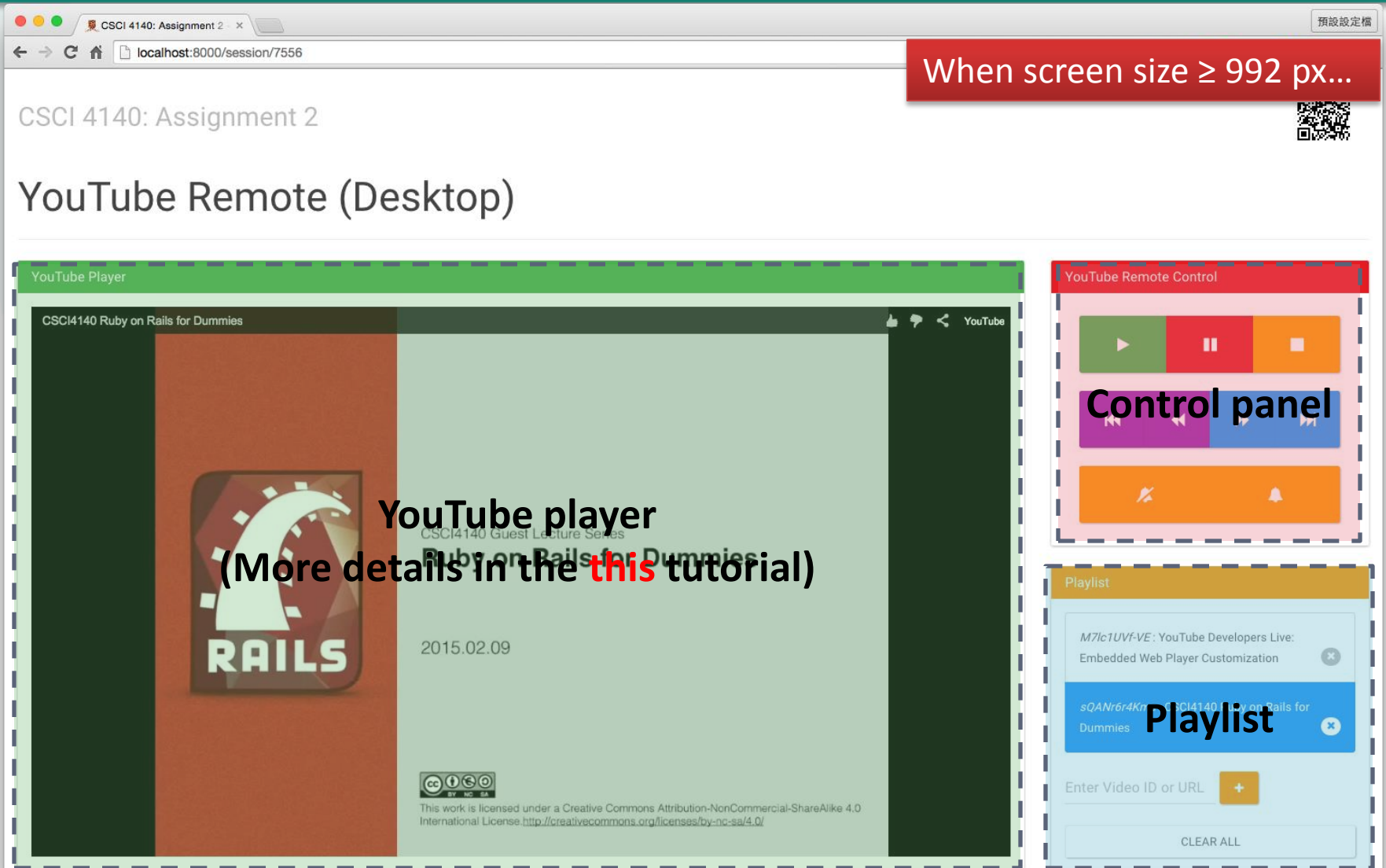
YouTube Remote (Desktop)

When screen size ≥ 992 px...



The screenshot displays a web browser window with the address bar showing 'localhost:8000/session/7556'. The main content area features a YouTube player with a green header 'YouTube Player'. The video player shows a thumbnail for 'CSCI4140 Ruby on Rails for Dummies' with a red background and a white 'RAILS' logo. The video title is 'CSCI4140 Guest Lecture Series Ruby on Rails for Dummies' and the date is '2015.02.09'. A Creative Commons license is visible at the bottom of the video player. To the right of the video player is a 'YouTube Remote Control' sidebar with a red header. It contains three rows of buttons: a green play button, a red pause button, and an orange stop button; a purple previous button, a blue next button, and a blue full screen button; and an orange volume button and an orange mute button. Below the remote control is a 'Playlist' section with an orange header. It shows two video entries: 'M7lc1UVf-VE: YouTube Developers Live: Embedded Web Player Customization' and 'sQANr6r4Kms: CSCI4140 Ruby on Rails for Dummies'. At the bottom of the playlist is a text input field for 'Enter Video ID or URL' with a plus button and a 'CLEAR ALL' button.

asgn2-desktop.png



asgn2-desktop.png

CSCI 4140: Assignment 2

YouTube Remote (Desktop)

YouTube Player

CSCI4140 Ruby on Rails for Dummies

CSCI4140 Guest Lecture Series
Ruby on Rails for Dummies

2015.02.09

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License <http://creativecommons.org/licenses/by-nc-sa/4.0/>

YouTube Remote Control

Play, Pause, Stop, Previous, Next, Full Screen, Volume

Playlist

- M7lc1UVf-VE: YouTube Developers Live: Embedded Web Player Customization
- sQANr6r4Kms: CSCI4140 Ruby on Rails for Dummies

Enter Video ID or URL

CLEAR ALL

When screen size ≥ 992 px...

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

asgn2-desktop.png

When screen size is between 768px and 992 px...

CSCI 4140: Assignment 2

YouTube Remote (Tablet)

YouTube Remote Control

▶ PLAY

⏸ PAUSE

■ STOP

⏮ PREVIOUS

⏮ REWIND

⏭ FAST FORWARD

⏭ NEXT

🔊 MUTE

🔊 UNMUTE

Playlist

M7lc1UVF-VE: YouTube Developers Live: Embedded Web Player Customization

sQANr6r4Kms: CSCI4140 Ruby on Rails for Dummies

Enter Video ID or URL



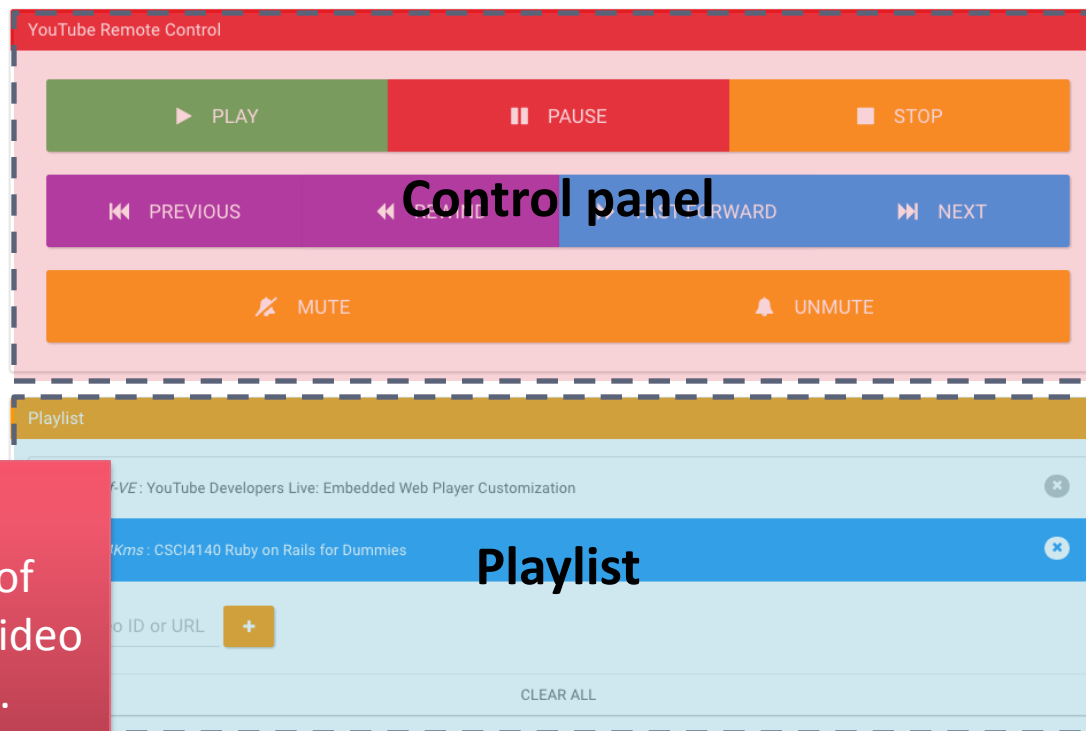
CLEAR ALL

asgn2-tablet.png

When screen size is between 768px and 992 px...

CSCI 4140: Assignment 2

YouTube Remote (Tablet)

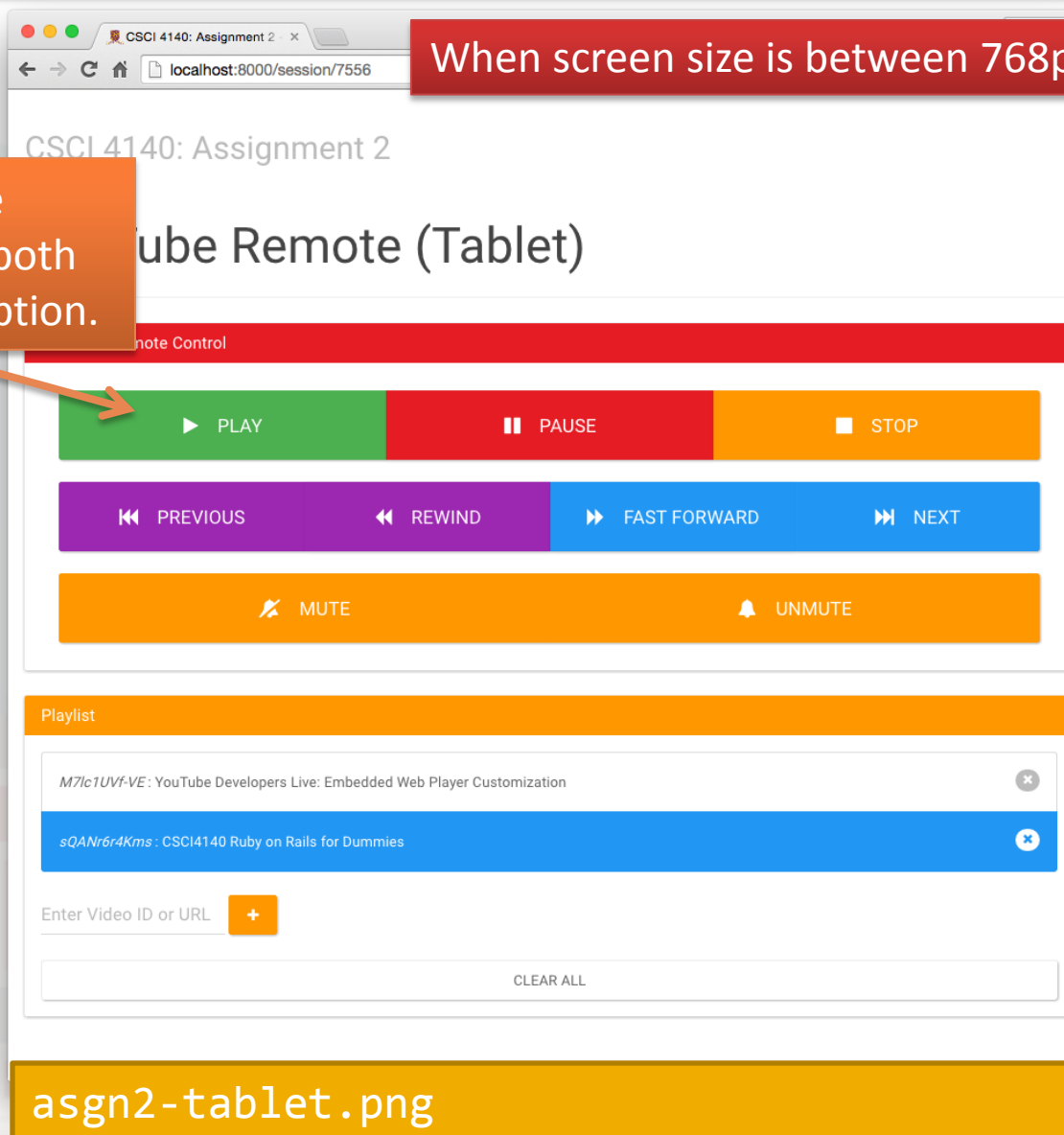


Note: When we reduce the width of the window, the video player is **removed**.

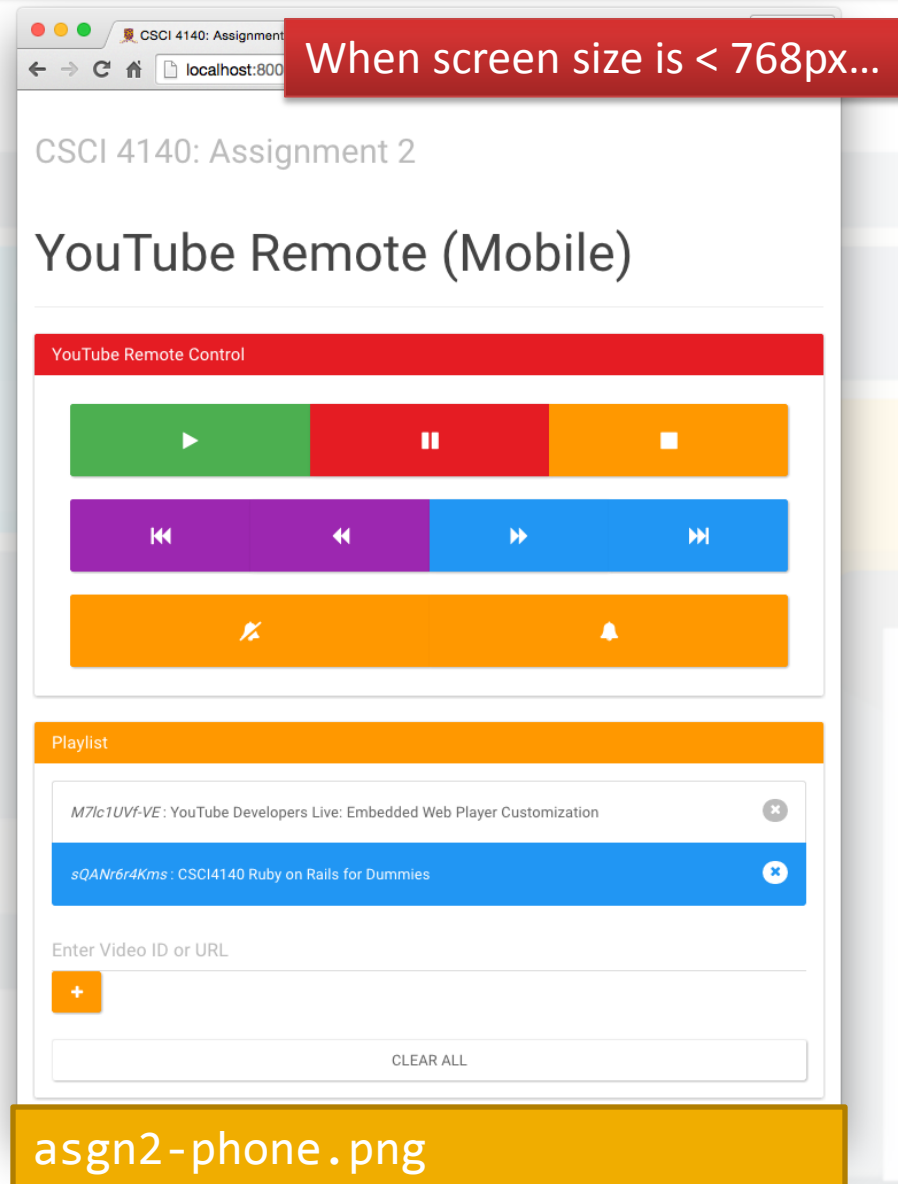
asgn2-tablet.png

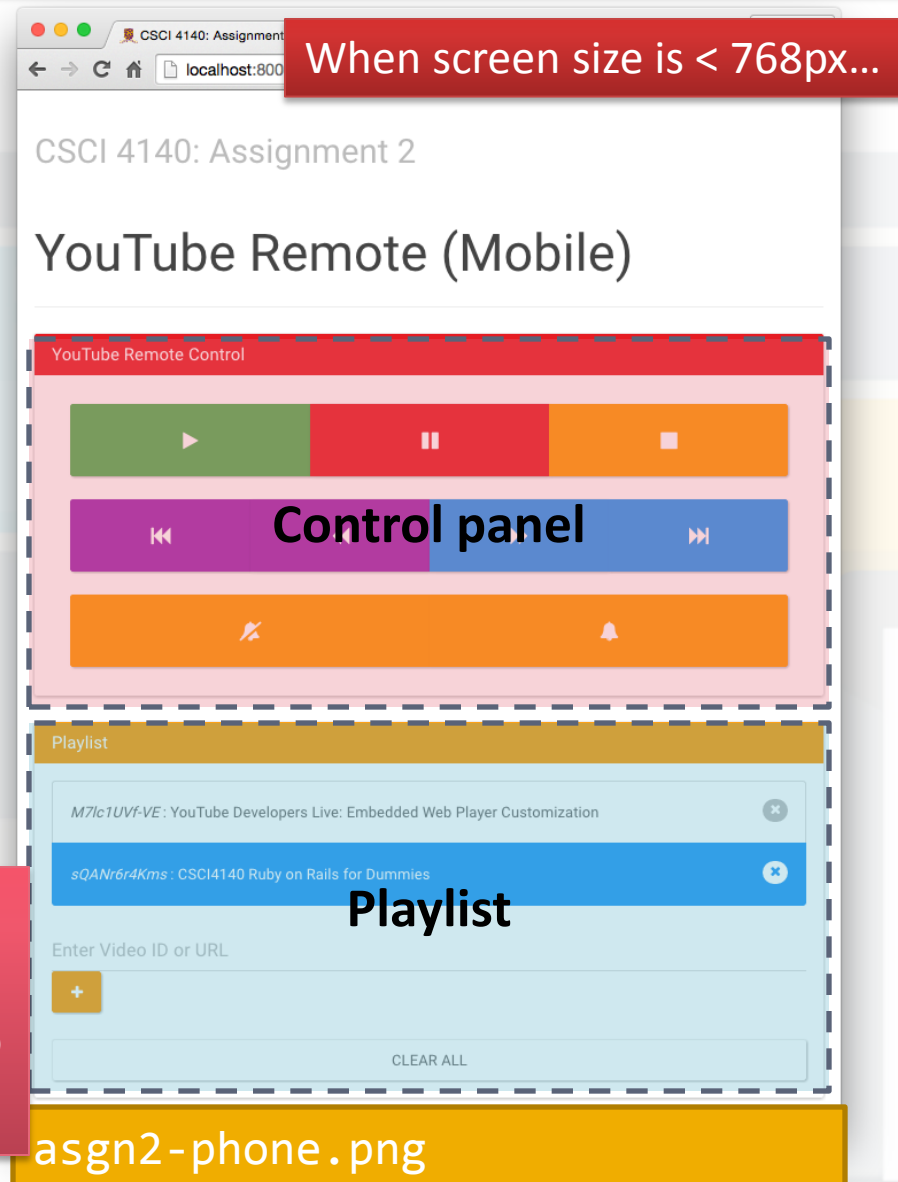
When screen size is between 768px and 992 px...

The buttons are displayed with both icon and description.



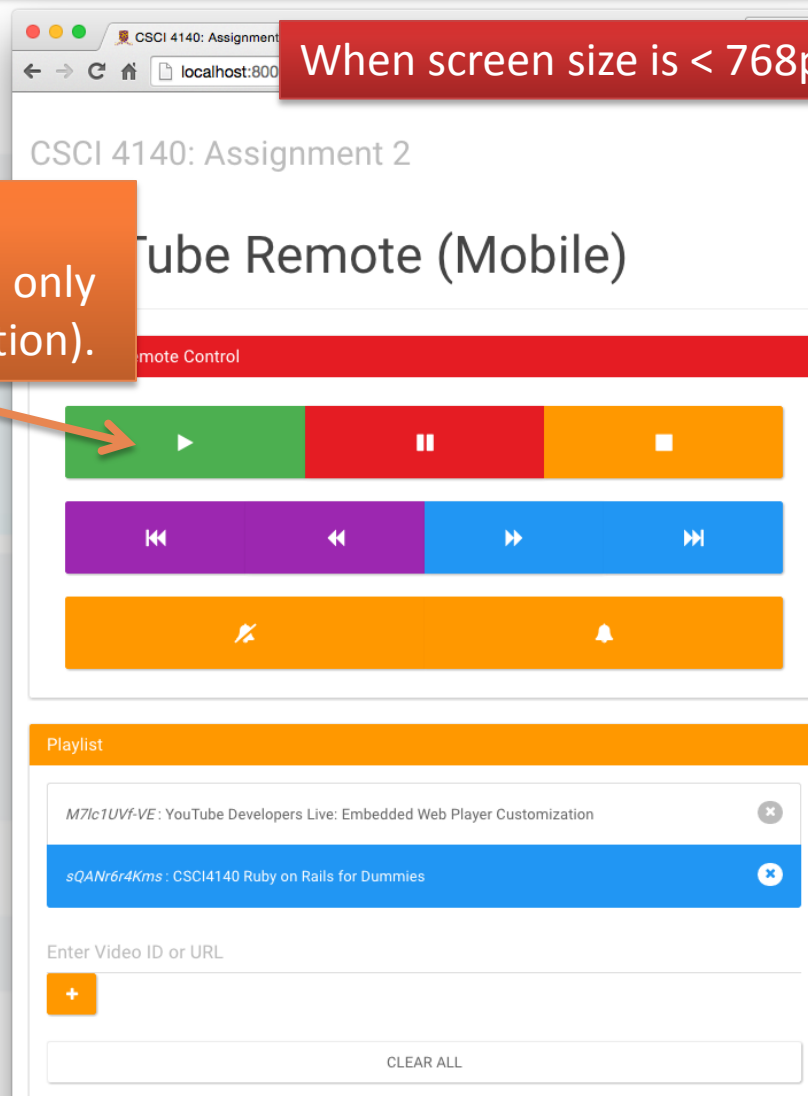
asgn2-tablet.png





When screen size is < 768px...

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:

Playlist

M7lc1UVf-VE: YouTube Developers Live:
Embedded Web Player Customization

sQANr6r4Kms: CSCI4140 Ruby on Rails for
Dummies

Enter Video ID or URL

CLEAR ALL

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

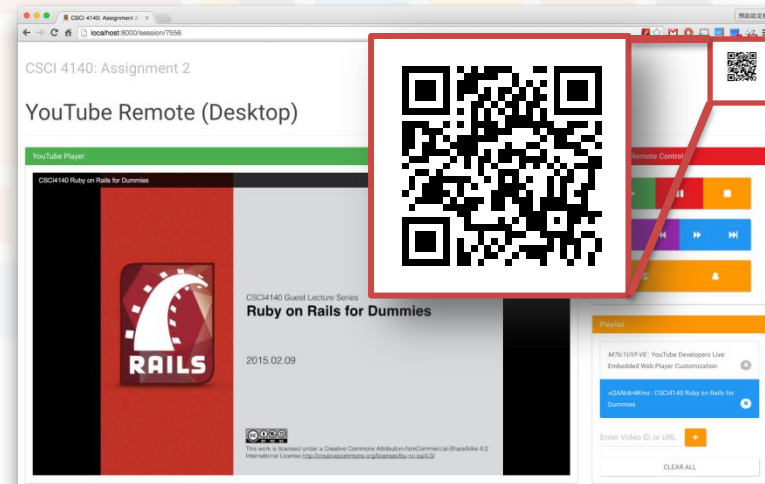
Note: If you want to clone the UI of my site, you can download the high quality screen capture from the resource page 😊



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://developers.google.com/chart/infographics/docs/qr_codes

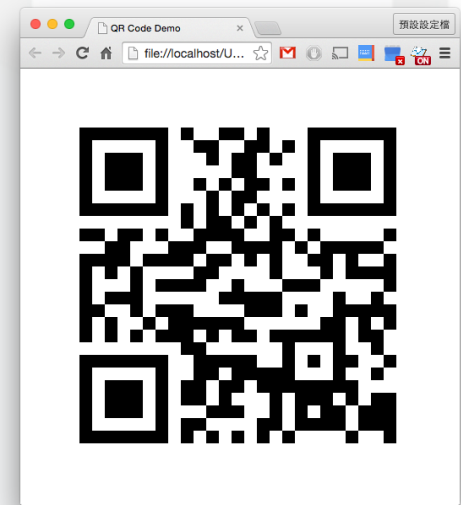


Client side: QR code display

- Google Chart Example:

```
<html>
<head><title>QR Code Demo</title></head>
<body>
  
</body>
</html>
```

qr_code/qr_code.html



Client side: QR code display

- Google Chart Example:

```
<html>
<head><title>QR Code Demo</title>
<body>
  
</body>
</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 `encodeURIComponent()` function can do so.

Ref.: http://www.w3schools.com/jsref/jsref_encodeuri.asp

- How to combine it with “`location.href`”?
 - Hint:** Generate the `` 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 an 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> );
```

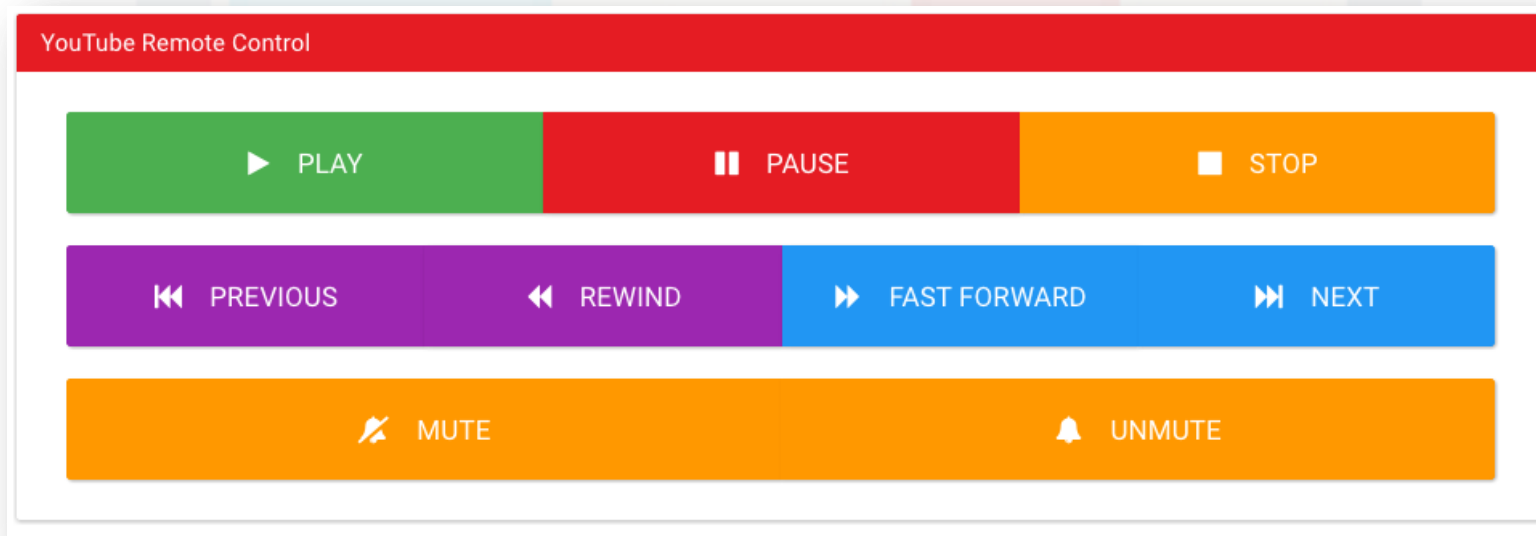
- Note: The data type of the item should be **String**
 - Use “**JSON.stringify()**” to convert an array / object to a String



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/>)

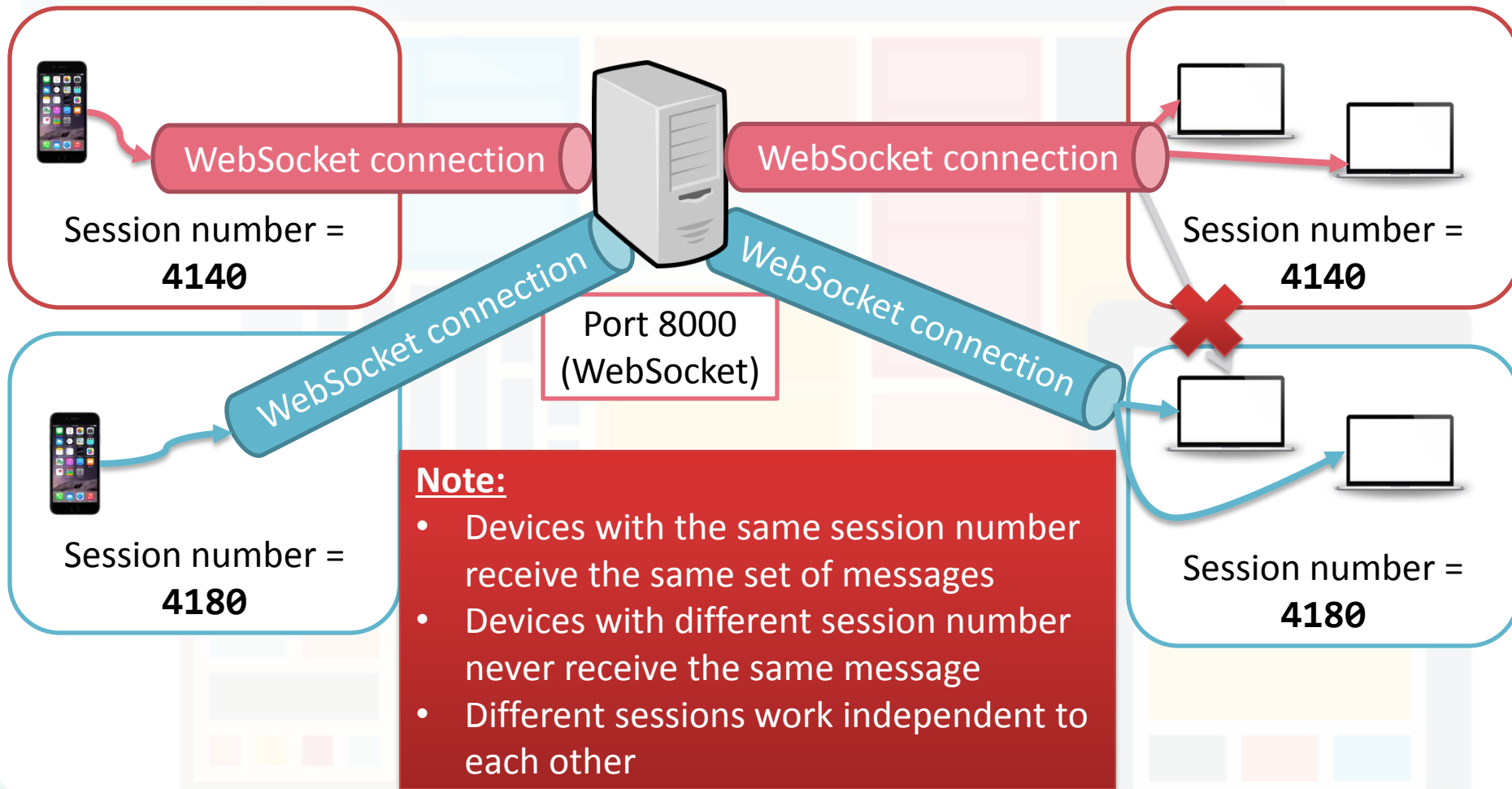


Server generates a unique session number



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

Server side: Message forwarding



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

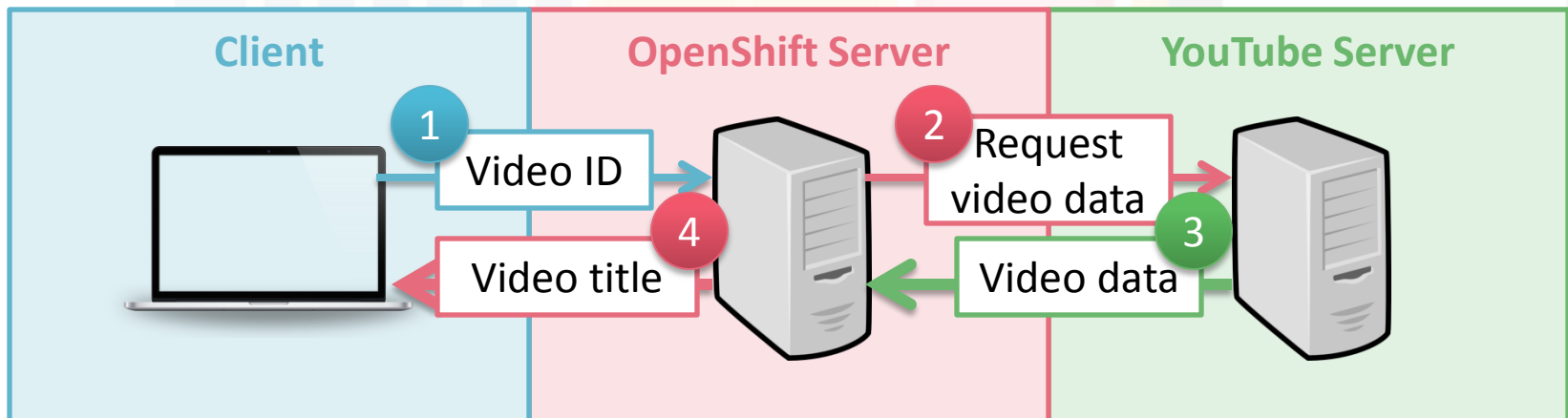
- The server returns a **JSON object**
 - The video title is returned inside the **"title"** field

```
{"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 –