Building Apps

Maxwell Da Silva - Video Software Architect at NYTimes
What platform do I use to develop my apps?

- iOS
- Android
- Chrome

In order to publish an application, you need to subscribe to Chromecast. There’s a $5 fee.
Design apps for Chromecast

- Your app is the remote
- Cross platform
  => (Design your app to be easy to use)
- Second screen experience
- Connect and Play or Play and Connect
- Don’t make your UI on TV look interactive
Settings for “Chromecast4212”

Wi-Fi Network: Maxwell’s iPhone
Wi-Fi Password: ******

Chromecast Name: NYTCast

Time Zone: Pacific Standard Time (Los Angeles)
Time Format: 12-Hour Time
Language: English (United States)

Send Chromecast usage data and crash reports to Google
Send this Chromecast’s serial number to Google when checking for updates

Reboot Chromecast  Factory reset Chromecast
Welcome to the Google Cast SDK Developer Console

The Google Cast Developer Console enables developers to register applications and authorize devices for testing.

### Applications

<table>
<thead>
<tr>
<th>Application ID</th>
<th>Application Name</th>
<th>Status</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>EF77BA29</td>
<td>customnyt</td>
<td>Unpublished</td>
<td>Edit</td>
</tr>
<tr>
<td>BF0CAE75</td>
<td>smwest</td>
<td>Unpublished</td>
<td>Edit</td>
</tr>
</tbody>
</table>

**ADD NEW APPLICATION**

### Devices

<table>
<thead>
<tr>
<th>Serial Number</th>
<th>Description</th>
<th>Status</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>nytimes cast</td>
<td>Ready For Testing</td>
<td>Remove</td>
</tr>
<tr>
<td></td>
<td>LUCAS</td>
<td>Ready For Testing</td>
<td>Remove</td>
</tr>
</tbody>
</table>

**ADD NEW DEVICE**
Receiver & Sender apps

Receiver

Senders

- iOS
- Android
- Chrome
Application lifecycle

1. Load the Application
   Initialize the API

2. Discover devices

3. Start a Session

4. Send/Receive messages

5. Disconnect
Types of receiver apps

Styled Media Receiver vs Custom Receiver
What is a Receiver app?

It’s an application written in Javascript/HTML/CSS that runs on the receiver (Chromecast)
It’s a pre-built app hosted by Google that was built to playback audio or video content.

You can provide a CSS file to customize the appearance of your app.
You can define a background style for your application:

```css
.background {
  background: #000000;
}
```
You can add your logo:

```css
.logo {
  background-image: url(timesvideo_logo.png);
}
```
You can add an transparent watermark image:

```css
.watermark {
    background-image: url(times_watermark.png);
    background-size: 57px 61px;
    opacity: 0.5;
}
```
You can style your progress bar:

```css
.progressBar {
    background-color: #6288a5;
}
```
You can also define a splash image:

```css
.splash {
  background-image: url(splash.png);
}
```
Live demo Styled Media Receiver
Debugging your Receiver app

Connecting to a non-secure WebSocket server from a secure origin is deprecated.

Failed to load resource: the server responded with a status of 404 (Not Found)

The page at 'https://www.gstatic.com/eureka/player/player.html?skin=https://smwest.herokuapp.com/chromecast-styled/skin4.css' was loaded over HTTPS, but displayed insecure content from 'http://graphics8.nytimes.com/images/2014/10/03/multimedia/36hours-tulum/36hours-tulum-thumbWide.jpg': this content should also be loaded over HTTPS.

The page at 'https://www.gstatic.com/eureka/player/player.html?skin=https://smwest.herokuapp.com/chromecast-styled/skin4.css' was loaded over HTTPS, but displayed insecure content from 'http://video1.nytimes.com/video/2014/11/05/29304_1_36hours-tulum_wg_720p.mp4': this content should also be loaded over HTTPS.

The page at 'https://www.gstatic.com/eureka/player/player.html?skin=https://smwest.herokuapp.com/chromecast-styled/skin4.css' was loaded over HTTPS, but displayed insecure content from 'https://smwest.herokuapp.com/chromecast-styled/splash.png': this content should also be loaded over HTTPS.
Custom Receiver

It’s also written in Javascript/HTML5/CSS.

You have to host all the assets.
Custom Receiver: Why?

- Build a second screen game
- Video application that require DRM
- Image gallery, Audio gallery
- All kinds of second screen experience
Basic structure for an audio/video custom receiver

```html
<!DOCTYPE html>
<html>
<head>
  <title>Your Custom Media Receiver</title>
  <script type="text/javascript" src="https://www.gstatic.com/cast/sdk/libs/receiver/2.0.0/cast_receiver.js"></script>
  <script type="text/javascript" src="https://www.gstatic.com/cast/sdk/libs/mediaplayer/0.7.0/mediaplayer.js"></script>
</head>
<body>
  <!-- Create your app elements here -->
</body>
```
Custom Receiver: DRM?

There is a Media Player Library to support the DRM
**Supported media types**

**Video Codecs:** H.264, VP8

**Audio Codecs:** MP3, HE/LC-AAC, Vorbis, WAV

**Subtitles:** WebVTT, TTML, CEA608

**DRM:** PlayReady, Widevine

**Adaptive Streaming:** HLS, DASH, Smooth Streaming
Receiver limitations

- Limited memory/CPU power
- Single concurrent playback experience
- No video compositions or manipulation such as, zooming or rotation.
How many developers in our audience?
Sender apps

- Android - Java
- Chrome - Javascript
- iOS - Objective C
Chrome sender apps

It’s also written in Javascript/HTML5/CSS.
// Initialize chromecast session
var applicationID = 'BF0CAE75';
var sessionRequest = new chrome.cast.SessionRequest(applicationID);
var apiConfig = new chrome.cast.ApiConfig(sessionRequest,
    function onSessionListener(session){
    },
    function onReceiverListener(device){
    });

chrome.cast.initialize(apiConfig,
    function onInitSuccess(e){
    console.log("success", e);
    },
    function onInitError(e){
    console.log("error", e);
    });
}
// Request Chromecast session

chrome.cast.requestSession(
    function onRequestSessionSuccess(s) {
        console.log("request session ", s);
    },
    function onRequestSessionSuccessError(e) {
        console.log("Error", e);
    });

Cast localhost to...

Chromecast3494

Options
Send feedback...
Help
function loadVideo(video) {
  var mediaInfo = new chrome.cast.media.MediaInfo(video.url);
  mediaInfo.metadata = new chrome.cast.media.GenericMediaMetadata(
    mediaInfo.metadata.metadataType = chrome.cast.media.MetadataTypeType.GENERIC;
    mediaInfo.contentType = 'video/mp4' // mp4/webm;
    mediaInfo.metadata.title = video.title;
    mediaInfo.metadata.images = [{url: video.poster}];

  var request = new chrome.cast.media.LoadRequest(mediaInfo);
  request.autoplay = true;
  request.currentTime = 0; // time in seconds
  session.loadMedia(request,
    function onMediaLoaded(media) {
      console.log("Media", media);
    },
    function onLoadMediaError(error) {
      console.log("Media Error", error);
    });
}

loadVideo({url: "my_video.mp4", title: "My Video SMW", poster: "my_video_poster.jpg"});
// Player controls functions

function seekTo(position) {
  var seekRequest = new chrome.cast.media.SeekRequest();
  seekRequest.currentTime = position; // value in seconds.
  media.seek(seekRequest,
             function onSuccess(s){},
             function onError(e){}
           );
}

function pausePlayback() {
  var pauseRequest = new chrome.cast.media.PauseRequest();
  media.pause(pauseRequest,
              function onSuccess(s){},
              function onError(e){}
            );
}

function playPlayback() {
  var playRequest = new chrome.cast.media.PlayRequest();
  media.play(playRequest,
             function onSuccess(s){},
             function onError(e){}
           );
}
Live demo Chrome Sender app
Resources

- https://developers.google.com/cast/
- http://chromecast.com/apps
Questions?

twitter:@dayvson
/email:dayvson@gmail.com