Brief Introduction to ITU-T H.762 (LIME)

ITU-T

LIME =Lightweight Interactive Multimedia Environment

- *Not a new "language" but a simple profile of HTML and Javascript for creating interactive content with tight integration with video and audio.*

- Based on common Web-practices like HTML, CSS, DOM and JavaScript (ECMAScript) - Just like very simple Web designing
- Primarily a multimedia language (rather than text)
- Easy integration of Video (both Linear streaming and Video-On-Demand)
- Suitable for different types of terminals: e.g. TV set, STB, Mobile phone, PC, etc
- Agnostic as to DTV standard:
  - Can work with DVB, ISDB, ATSC
  - Already used for cable TV, IPTV and mobile

“Hello world” in LIME

```
<?xml version="1.0" encoding="UTF-8" ?>
<?bml bml-version="100.0" ?>
<!DOCTYPE html>
<html >
<head>
    <title>Greetings</title>
</head>
<body>
    <p style="top:100px;left:100px;width:260px;height:72px;color-
    index:7;">Hello, world!!!</p>
    </body>
</html>
```

“Hello world” in LIME

```
<?xml version="1.0" encoding="UTF-8" ?>
<?bml bml-version="100.0" ?>
<!DOCTYPE html>
<html >
<head>
    <title>Greetings</title>
</head>
<body>
    <p style="top:100px;left:100px;width:260px;height:72px;color-
    index:7;" >Hello, world!!!</b>
    </body>
</html>
```

This is necessary for telling the STB the content is LIME.

This is for an HTML5 compliant browser. There are also other ways

An element to be shown has to have its “style” defined.

“Hello world” in LIME

```
<?xml version="1.0" encoding="UTF-8" ?>
<?bml bml-version="100.0" ?>
<!DOCTYPE html>
<html >
<head>
    <title>Greetings</title>
</head>
<body>
    <p style="top:100px;left:100px;width:260px;height:72px;color-
    index:7;">Hello, world!!!</p>
    </body>
</html>
```

This is necessary for telling the STB the content is LIME.

This is for an HTML5 compliant browser. There are also other ways

Color is specified by `color-index` attribute. Color-index 7 is white.

Monomedia Used in LIME content

<object> Element `type` attribute	Content
image/jpeg	JPEG
image/X-arib-png	cPNG*
image/X-arib-mng	MNG
audio/X-arib-mpeg2-aac	AAC sound
audio/X-arib-aiff	AIFF sound
application/X-arib-contentPlayControl	Video on Demand (H.264 or MPEG2)
application/X-arib-mpeg2-tts	Multicast video (if the server supports it)

*Contextualized PNG

AV File format used in LIME

- The audio/video file container used for ITU-T H.721, which LIME typically supports, is “Time-stamped MPEG2 Transport Stream”
- This typically has extensions such as “.m2ts”, “mts”, etc.
- This is the standard format also used in Blueray Disks, as well as in the high definition movie format for digital movie cameras (AVCHD).
- There are several open source tools to encapsulate an ITU-T H.264 file in an “.m2ts” file. (e.g., *tsMuxeR*)

How to use graphics in LIME (1)

- An embedded JPEG content is specified as follows:
- Javascript (LIME-Script) can control it various ways.
- A JPEG file is displayed ¼ size of the original file (½ width, ½ height), so that the JPEG file needs to be adjusted.
  - E.g. If the original JPEG file has the height 200 pixels and width 400pixels, the “style” attribute of the “object” element should be specified as “height:100px;width:200px”

```
<body>
<div>
<object id=“img" data=“path/graphics.jpg“ type="image/jpeg"
  ></object>
</div>
```

The path to the location of the jpeg file

MIMEType specifying JPEG

How to use graphics in LIME (2)

- An embedded cPNG content is specified as follows:
- cPNG is a contextual PNG, which may lack the palette information so that its size may be small. It also has fewer colors. A tool* is needed to reduce the number of colors.

```
<body>
<div>
<object id="png" type="image/x-arib-png" >
data="path/graphics.png"
</object>
</div>
```

MIMEType for cPNG

The path to the location of the jpeg file

* Available at ITU Application Challenge website

How to show video on LIME

- LIME can handle both multicast and VOD. In this document, we only deal with VOD.
- Two ways to view VOD:
- Embedded video is with Object tag
  - `<object type="application/X-arib-contentPlayControl" data="http://manifest" streamstatus="play" />`
- Full-screen, with the control to the terminal, can be launched by a Javascript API:
  - `browser.launchIPTVContent("http://manifest", './back.xml', 0);`

How to use VOD content in LIME

- An embedded VOD content is specified as follows.
- Javascript (LIME-Script) can control this in various ways
- It is easy to specify text and graphics overlaying the VOD content

```
<body>
<div>
<object id="vod" type="application/X-arib-contentPlayControl"
   data="http://server/video.cpc" streamstatus="play"
   ></object>
</div>
```

URL of the "manifest" file of the VOD content. Note that this is not the same as the actual location of the content.

MIMEType for VOD content

Sample using DOM and LIME-script

- The sample below shows a code snippet to change the width of VOD content upon loading.
- It is comparable to ordinary Javascript (Javascript v1.1)

```
<head>
<script>
function myload(){
    var video = document.getElementById(“vod");

    video.normalStyle.width = "540"
}
</script>
</head>
<body onload=“myload();”>
<div>
<object id="vod" type="application/X-arib-contentPlayControl"
    data=“http://server/video.cpc" streamstatus="play" ></object>
</div>
```

Get the element with the specified ID using DOM interface.

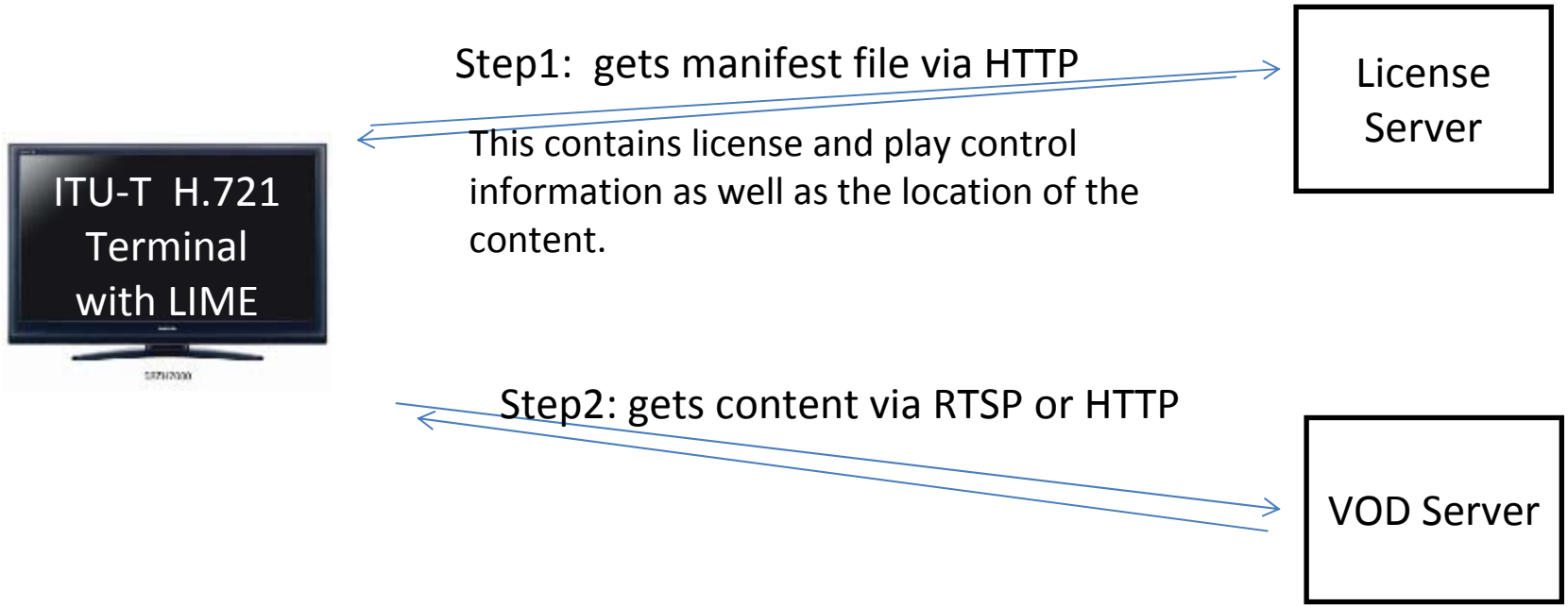
Change the style of the selected element. Note that the style attribute is accessed by `normalStyle` and not `style`

LIME Server

- A LIME server can be implemented using many open source HTTP (HTML) servers, such as *Apache* and *Nginx*.
- License server, which provides the manifest file for VOD content, can also be implemented as an HTTP web server.
- A LIME file can be treated as an ordinary HTML file or a web page.

Steps in calling VOD content from LIME

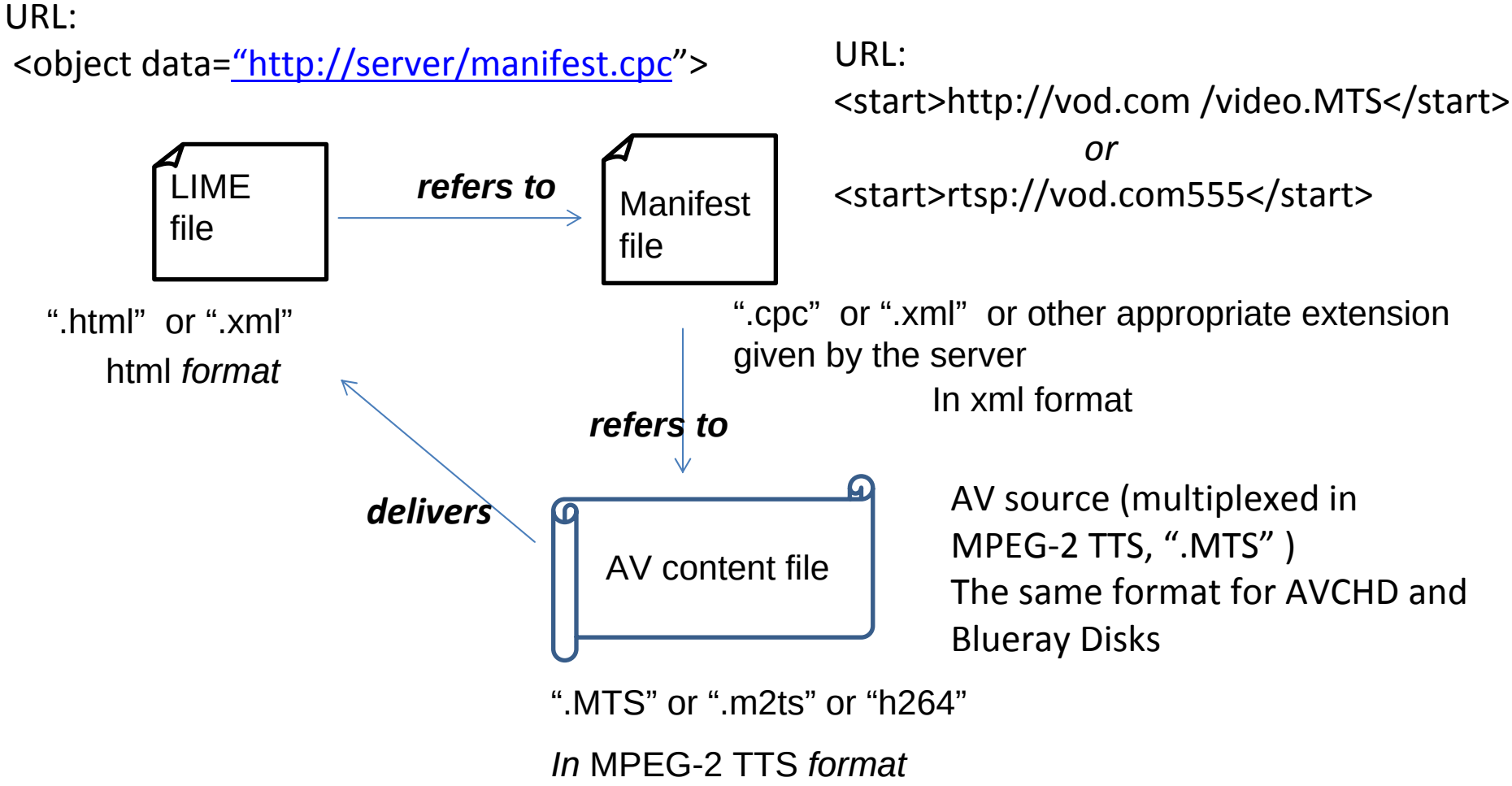
- LIME can show VOD content as embedded content.
- There are steps to receive actual content from the content server



The protocol (RTSP or HTTP) to be used in delivering the AV content is specified in the manifest file.

Relationship between Files

- The following diagram shows the relationship between the files involved in calling an AV content from LIME and to embed it as part of the LIME content.



H.762: Some Technical Details

- Static LIME Part
  - Simple XHTML (LIME-HTML)
  - Simple CSS (LIME-CSS)
- Dynamic LIME Part
  - Simple DOM (LIME-DOM)
  - Restricted JavaScript (LIME-Script)

Namespaces and Document type (1)

- Since H.762's LIME –HTML is first based on XHTML, it has many characteristics of XHTML.
- The following is a typical declaration of a LIME-HTML document. Note that it uses XHTML style of namespace.

```
<?xml version="1.0" encoding="UTF-8" ?>
<?bml bml-version="100.0" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

Note 1: In LIME, "bml" stands for "Basic Multimedia Language".

Namespaces and Document type (2)

- Note 2: Many implementations can understand the namespace, allowing both XHTML style and HTML. In this case, the following would also be allowable *a la* “HTML5”:

```
<?xml version="1.0" encoding="UTF-8" ?>
<?bml bml-version="100.0" ?>
<!doctype html>
<html xmlns="http://www.w3.org/1999/xhtml">
…
```

LIME-HTML: Simple HTML

- Only Structural Tags (`<head>`,`<style>`,`<script>`,`<body>`) and some limited number of Elements (`<p>`,`<div>`,`<object>`, `<input>`)
- All elements can have "`id`" and "`class`" attributes
- Any element should have its "`style`" defined, if to be displayed.

LIME-HTML: Elements

- About 10 elements, and the following Four Major elements in the `<body>`
- `<div>` :Universal “bag” element.
  - Defines a region that appears on the display
- `<p>`: Text element
  - Also used for defining e.g. a button, figure, form, etc.
- `<object>`: monomedia element
  - Used for graphics, audio and video
  - Always used within `<div>`
- `<input>` : Universal input element
  - <form> is not used
  - Used within and with <p> to create a form

LIME-HTML: some attributes

- The color-buttons are accessed by `accessKey` attribute on an element for a button (graphics or text)

- `onclick`, `onblur`, `onfocus` (and `onload` for `<body>`) can be defined for interactive events.

Use of Color Buttons

- Color buttons (Blue, Red, Green, Yellow) are accessed by `accessKey` attribute of an element.
- The values for this attribute are: B (blue), R (red), G (green), Y(Yellow).
- With this attribute and `onclick` attribute, an action can be associated with clicking of a color-button, where the action is defined by a function in Javascript (LIME-Script).
- The sample below associates an action to go back to another file to the blue button.
- ***Note that the defined action cannot take an argument.***

```
<script>
function goback(){browser.launchDocument("../index.xml", "cut");
        }
</script>
<body>
<p style="top:470px;left:17px;height:36px;width:64px;background-
color-index:4;" accesskey="B" onclick="goback();">Back</p>
</body>
```

LIME-CSS

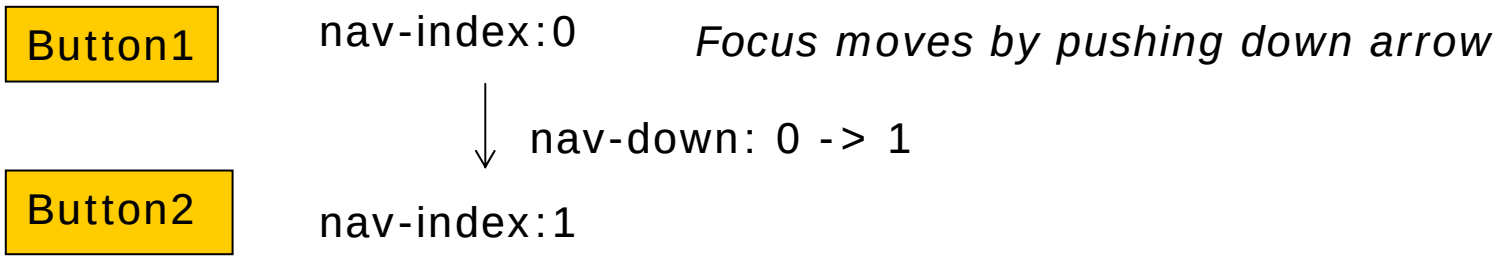
- Essential technology for presenting LIME (“without CSS, no LIME”)
- Restricted number of attributes (to reduce redundancy – “must haves” rather than “nice to haves”)
- Used for both <style> element and also for tag-element like <p> and
- Many restriction to guarantee the “same” look-and-feel across terminals
- Some extensions for broadcasting:
- define specific `color-index` to refer to color -> to reduce memory consumption and ambiguity
- **The details of LIME-CSS are also specified in ITU-T Rec. H.763.1.**

NAV-* attributes

- nav-* attributes control the focus of navigation. The default focus is nav-index:0
- Each focused element can have a nav-index, according its order of navigation.
- Up, down, left, right arrow buttons are used to navigate. `nav-up` , `nav-down` , `nav-left`, `nav-right` correspond to these.
- `nav-up:`*`i`* represents a function that moves the focus from the current position to the element indexed *i* by pushing the up arrow. Similarly for the others.

```
<p id="p01" class="button" style="top:10;left:0:nav-index:0;nav-up:1;nav-down:1;" onclick="fncA();">Button1</p>

<p id="p02" class="button"  style="top:50;left:0;nav-index:1;nav-up:0;nav-down:0" onclick="fncB();">Button2</p>
```



Navigation in LIME-CSS

- In order to control the focus navigation of the Remote controller, LIME-CSS has the following attributes
  - `nav-index:` indicates a focusable element with an index. Default focus is “0”.
  - `nav-up:` indicates the index of the element to be focused when UP arrow is pushed.
  - `nav-down:` indicates the index of the element to be focused when DOWN arrow is pushed.
  - `nav-left:` indicates the index of the element to be focused when LEFT arrow is pushed.
  - `nav-right:` indicates the index of the element to be focused when the RIGHT arrow is pushed.

Use of Color-index in LIME-CSS

- In the ordinary W3C-CSS1, the following are the ways to specify the color “yellow” (any of them can be used):
  - `{color: yellow}`
  - `{color: rgb(255,255,0)}`
  - `{color: #ffff00 }`
- In LIME-CSS, the following is the way to specify the same color:
  - `{color-index:3}`
- (ITU-T Rec. H.763.1, “CSS for IPTV”, Annex 1, gives the table that maps color index to RGB. And see the annex to this document.)

LIME-CSS: used-key-list

- an attribute of `body`
- specifies the keys to be used on the remote controller.
  - `used-key-list:basic data-button;`
  - `used-key-list:basic data-button numeric-tuning;`

Restricted JavaScript (LIME-Script)

- Few native Objects:
  - `Object,Array,Boolean,String,Date, Number (Function)`
- Limited number of methods: bare minimum: e.g. no `RegEx`, no `Real`, no `concat` for `String` or `Array`, etc. → for security and memory-management
- External files (libraries) are allowed
- Just sufficient for simple interactivity
  - HTTP GET and POST allowed
- Additional `browser` Pseudo-Object specially for LIME Provides some necessary functions and API.

Some Javascript API H.762 for IPTV

- API for launching VOD
  - `browser.launchIPTVContent()`
- parental control of content
  - API for DRM (different DRMs allowed)
- API for Service entry registration

Functions similar to normal ECMAScript

- `browser.setInterval(`*`fn`*`,`*`s`*`,`*`t`*`)`
  - Repeats function *`fn`* with the interval *`s`* miliseconds *`t`* times. If `t` is `0`, then it repeats itself forever.
  - Note `fn` cannot take an argument.
- `browser.random(`*`int`*`)`
  - Returns a random number between `1` and *`int`*.

```
var vid=document.getElementById(“video”);

function keepPlaying(){
        vid.streamStatus="play";
        browser.setInterval(“replay();",60000,0);
}
function replay(){
        vid.streamStatus="stop";
        vid.streamStatus="play";
}
```

Some Functions specific to LIME

- `browser.transmitTextDataOverIP (x,y,z)`
  - sends data *y* to URL *x* in code *z*, receiving an array of data
  - The actual data is in the third item in the array
  - This is an interface to HTTP POST
- `browser.launchDocument(x,y)`
  - Receives data from URL *x*. *y* is always *"cut"*, receiving the LIME document
  - This is an interface to HTTP GET

The following sample shows how the first one above is used:

```
var ret = browser.transmitTextDataOverIP(url,val,"EUC-JP");
        if( ret[0] == 1 ){/* if the returned data is not nil */
                do_something_with(ret[2]);
        /* ret[2] contains the returned data */
        }else{  do_other_thing(); }
```

Simple DOM in LIME

- Document object used for accessing nodes, rather than creating nodes (no createNode etc.)
- These are the main methods used in LIME-DOM
  - `document.getElementById("…")`
    - Same as in other DOM for HTML
  - `document.currentEvent`
    - Interface to the current event (of remote controller)

```
function handler(){ var evt = document.currentEvent;
        if(evt.type=='keydown'){  over1(); }else{  out1();}
} /* every time key is pushed down, function over1() is called */
</script>
..
<body>
<div class="button">
<object id="image" style="nav-index:0;" onkeydown="handler();"/>
</div>
```

Thank you!

- For more information on LIME, please visit the Application Challenge Website http://www.itu.int/challenges/iptv
- or write to: *iptvchallenge@itu.int*