kt olleh tv web application

개발 가이드

Version 1.0.9

2017/04/13

Table of Contents

List of Figures

List of Tables

Document Revision History

Ver.	Date	History Section	Author
0.1	2012/11/15	First draft	이상현
0.2	2012/11/30	- Overall chapters reordered - "2. 개발환경" revised - "3.1 웹 어플리케이션 소스 보호" added - "3.2 Single Thread Model" added - "4. Application UI Layering" revised - "7. IME/OSK" revised - "9. VOD" revised - "11. Closed Caption" added - "Appendix A. Compliance Table" added	이상현
0.3	2012/12/28	- "2.2 PC Browser 기반 개발환경" deprecated - "2.1.3 Remote Inspector" added - "2.5 Font" added - "2.6 User Agent" added - "3.5 HTML 작성 유의 사항" added - "3.6.1 Strict Mode, Appendix C Strict Mode" added - "5.1 How to Find Web Application" updated (application property key defined) - "5.3 How to Start Bound Application with Parameters" added - "8 Object Carousel" added - "9 Application Packaging (Widget)" added - "13 USB Storage Device" added - "Appendix A Compliance Table" updated - ClosedCaption moved to "Appendix E" - Extended API defined in "Appendix D, E"	곽창원
0.4	2013/01/29	- "3.7 Memory Usage" added - Default background color defined (4.1.3 Initial State) - "이전", "지우기" key code modified - "Appendix A.2 Web Specifications" added	곽창원
0.4.2	2013/02/08	-. "2.6 User Agent" updated -. Guide added in "3.2 Single Thread Model" -. Key name updated in "5.1. How to Find Web Application" -. "7. IME/OSK" updated -. Memory usage guideline added in "10. Unbound Application Guideline" -. "12. Configuration & Miscellaneous APIs" added -. "Appendix A" updated.	곽창원
0.5.2	2013/02/22	-. "1.3 Application Type" updated -. "2.1.1 어플리케이션 구동 방법" updated to support 2 different application type and resolution -. "2.2 PC Browser 기반 개발환경" removed -. "2.5 User Agent" updated -. "4 Application Basics" added -. "6.4 How to Start Unbound Application" added -. "11.3 API Guideline for Web Messaging" added -. "Appendix J. Sample Application" added	곽창원

0.5.3	2013/02/28	-. "Appendix C Comparison between OTS and OTV" added	곽창원
0.6.0	2013/03/04	-. "2.1.1 어플리케이션 구동 방법" updated to support 2 different application type and resolution -. "Appendix C Comparison between OTS and OTV" updated to modify model name of OTV	곽창원
0.6.1	2013/03/11	-. User Agent string updated in "2.5.1" -. Some code snippet fixed	곽창원
0.7.0	2013/03/19	-. Widget test page added in "2.1.1" -. "2.1.4 Mouse Event 사용법" added -. All orgId of app id changed to "4e30" -. Widget id format defined in "10.2" -. Memory limit of unbound app defined in "11.1" -. "11.3" changed to JSON format-. SAID key name changed in "13.1" -. "Appendix C Understanding Memory Usage" added	곽창원
0.7.1	2013/04/03	-. Application ID is assignable in test control page. ("2.1.1") -. "3.6.4 Indexed DB" added -. Comment added in "5.2.5"	곽창원
0.7.2	2013/04/16	-. Application ID is assignable for widget ("2.1.1") -. "16. Video Capture" added	곽창원
0.7.3	2013/04/30	-. New font added in "2.4" -. "13.3 PIN check" added -. "16 Video Caption" added to support SAMI format	곽창원
0.7.4	2013/05/20	-. "2.5 User Agent" updated -. "7.2 Keyboard Key" updated -. "8 OSK" updated with no IME, PIN mode, and initial input mode. -. "15.2 Widget Storage" added -. "16 Multiple Media Handling" added	곽창원
0.8.0	2013/06/07	-. "2.1.1 어플리케이션 구동 방법" updated -. "2.1.4 Mouse Event 사용법" removed -. "3.5.2 Margin, Padding" added -. "3.5.3 Single document" added -. "10.5 Widget Size" added -. "10.6 STB 내부에 위치한 어플리케이션의 CORS" added -. "13.4 가상 마우스 기능" added -. "15.1,1 How to List USB Files" updated using -. "16 Multiple Media Handling" updated -. "Appendix F. Extended APIs for video/mpeg, video/broadcast embedded object" added	곽창원
0.8.1	2013/06/14	-. "Appendix I. WebView APIs" added	곽창원
1.0.0	2013/08/25	-. "3.4.5 Event Handling Optimization" added -. "3.4.6. Avoiding Unnecessary Object Creation" added -. Configuration key "skylife_support" added in "13.1" -. "13.4 Mouse Control" revised -. "13.5 Channel Change" added -. "Appendix J. Mouse Control APIs" added	곽창원
1.0.1	2013/08/27	-. "5.3 Video Layering" added -. OTV/OTS model name changed in "Appendix D"	곽창원

1.0.2	2014/05/28	-. “2.3 Compile” added -. “3.4.7 반응성 좋은 UI 만드는 방법” added -. “13.6 Change to Promo Channel” added -. “19. Audio Clip” added -. “20 Media Metadata” added -. “21 Voice Data” added -. “Appendix K Audio Clip APIs” added -. “Appendix L Media Metadata APIs” added -. “Appendix M Voice Data APIs” added	박선규
1.0.3	2015/01/11	-. “2.1.5 How to Check Memory Usage” added	곽창원
1.0.4	2015/03/03	-. “3.8 Restriction on Application Destruction” added -. “16 Multiple Media Handling” modified -. “Appendix F Extended APIs for video/mpeg, video/broadcast embedded object” modified	박선규
1.0.5	2015/05/19	-. “1.3 Application Type” updated -. “2.1 STB 기반 실행환경” updated -. “2.1.1 어플리케이션 구동 방법” updated -. “2.1.4 How to Check Memory Usage” updated -. “4.2 Application Type” updated -. “4.3 Application Property” added -. “4.4 How to Start Application” updated -. “22 Speech Recognizer” added -. “23 Mosaic Window” added -. “Appendix N Mosaic Window APIs” added	박선규
1.0.6	2015/12/18	-. “7.1 Remote Controller Key” updated -. “22 Speech Recognizer” updated -. “24 Zip Extractor” added -. “25 TV Service Extension” added -. “Appendix N Speech Recognizer APIs” added -. “Appendix P Zip Extractor APIs” added -. “Appendix Q TV Service Extension APIs” added	박선규
1.0.7	2016/02/04	-. Configuration key “hdr_support” added in “13.1” -. “13.7 Video Resize for Channel” added -. “Appendix L Media Metadata APIs” modified	박선규
1.0.8	2016/4/4	-. Configuration Key “hdr_support_by_edid” in “13.1”	이우식
1.0.9	2017/04/13	-. Configuration key updated in “13.1”	안성민

1. Overview

1.1. Purpose

본 문서는 kt olleh tv web extension framework 기반에서 어플리케이션을 개발할 때 지켜야 할 개발 지침 사항이나 참고 사항을 명시하고 있다.

1.2. Target Audience

본 문서는 kt olleh tv web extension framework 기반에서 웹 어플리케이션 개발자를 대상으로 한다.

- 본 문서는 웹 서버 상에서의 서비스 개발 기술을 이해하는 자를 대상으로 한다.
- 본 문서는 웹 어플리케이션 개발 기술을 이해하는 자를 대상으로 한다.

1.3. Application Type

kt에서는 application 형태가 다양해서 그 형태에 대해서 각각 다음과 같이 정의한다.

Type	Description	useType[1]	Initial State	Example
Bound Application	AIT에 의해서 signaling되어 채널에 binding된 어플리케이션이다. 단 채널에 binding되지 않고 별도의 unicasting XML AIT로 제어되는 경우는 independent application으로 취급된다.	bound	invisible, inactive, unfocused	Red button application
Unbound Application	kt의 주된 서비스를 제공하기 위해서 XAIT로 signaling되는 어플리케이션. OIPF DAE widget의 형태를 띈다.	unbound	invisible, inactive, unfocused	Observer, Home Portal, AppStore[2]
User Application	kt AppStore를 통해서 설치한 개별 widget 형태의 web application을 말한다.	user	visible, active, focused	
Independent Application[3]	kt 채널을 통해서 송출하지 않고 AppStore에 의해서 설치되지도 않는 어플리케이션이다.	independent	visible, active, focused	
Host Application	STB firmware에 포함된 어플리케이션. 성격상 kt unbound application이 설치되지 않아도 동작해야 할 기능을 가진 어플리케이션이다.	host	invisible, inactive, unfocused	Navigator, HDS 인증 app 등

Table 1 Application Type

위 표에서 Initial state는 여기서 가장 중요한 유의해야 할 부분이다. 어플리케이션은 크게 초기에 visible이며 event 처리를 할 수 있는 active 이며 focused 상태로 시작되는 어플리케이션이 있고, 초기에 invisible이며 inactive 이며 unfocused 상태로 시작되어 show(), activateInput()을 호출해야

[1] Application Property 값 중 key 가 useType 에 해당하는 value 를 가리킵니다. 이를 이용해서 어떤 형태의 어플리케이션인지 구분할 수 있습니다. 방법은 "6.1 How to Find Web Application"를 참조.

[2] AppStore application 그 자체는 unbound application 으로 분류되어 서비스되지만, AppStore 에 등록되는 모든 어플리케이션은 user application 범주로 취급된다.

[3] 기존 ACAP 에서 독립형 어플리케이션과는 다르다. 독립형 어플리케이션도 엄밀하게는 독립형 채널에 bound 된 application 이기 때문에 bound application 이다. 여기서 말하는 independent application 은 채널에 bound 되지 않아야 한다.

각각 visible이며 event를 처리할 수 있는 상태로 되는 2가지의 형태를 가지고 있다. 동일한 어플리케이션이라고 해도 그 어플리케이션의 운용 방식에 따라서 그 초기 형태가 달라진다. 만약 어플리케이션이 띄우는 child application의 경우 정의된 type에 따른 initial state를 가지지 않고 application property를 이용하여 직접 initial state를 정의 할 수 있다.

자세한 사항은 "4.1 Application Identifier"을 참조한다.

1.4. Glossary

OIPF	Open IPTV Forum
OIPF DAE	OIPF Declarative Application Environment
NPAPI	Netscape Plugin Application Programming Interface
EPG	Electronic Program Guide
DOM	Document Object Model
CAS	Conditional Access System

2. 개발 환경

개발자가 웹 어플리케이션을 개발할 수 있는 방법에 대해서 크게 2가지 방법을 지원한다.

2.1. STB 기반 실행환경

개발자용 STB의 browser를 통해서 local 혹은 remote web page를 loading하여 테스트하는 방법이다. Debugging은 STB console을 통해서 로그로 확인하거나 PC browser를 통해 remote inspector를 사용할 수 있다.

2.1.1. 어플리케이션 구동 방법

어플리케이션을 구동할 수 있는 방법은 2가지가 있다.

WebApp Manager 구동방법

STB 가 동작 중인 상태에서 PC의 웹 브라우저를 통해 아래 주소로 접속을 한다.

```
http://<STB IP>:8090/sdk/appmanager.html
```

그러면 다음과 같은 WebApp Manager 페이지가 실행된다.

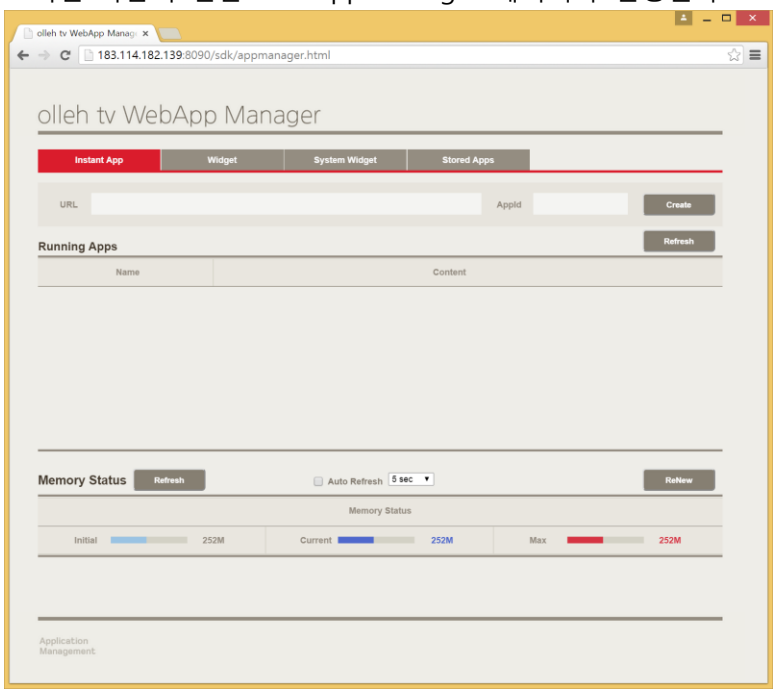


Figure 1 WebApp Manager

Instant App tab에서 URL에 테스트하기를 원하는 App의 주소를 입력하고 Create 버튼을 누르면 해당 App이 STB 에서 실행된다.
App 에 특정 ID를 지정하려는 경우 URL과 함께 AppId 에 application ID (<org ID>.<app ID>)를 입력하고 Create 버튼을 누른다. oipfApplicationManager의 findApplications("dvb.appId") 를 통해서 application proxy를 얻을 수 있게 된다.

App은 별도의 web server에 올려놓고 실행할 수도 있고, USB에 해당 App을 넣어놓고 실행할 수도 있다. USB는 "file:///mnt/usb1" prefix를 붙여서 접근 가능하다. 즉 USB root에 testApp.html 파일을 있다면 "file:///mnt/usb1/testApp.html" 로 실행할 수 있다.

예)
http://mywebserver.com/testApp.html
file:///mnt/usb1/testApp.html

Widget tab에서 URL에 설치하기를 원하는 widget의 주소를 입력하고 Install 버튼을 누르면 해당 widget이 STB에 설치된다.
Widget에 특정 ID를 지정하려는 경우 URL과 함께 AppId 에 application ID (<org ID>.<app ID>)를 입력하고 Install 버튼을 누른다. oipfApplicationManager의 findApplications("dvb.appId") 를 통해서 application proxy를 얻을 수 있게 된다.

Widget도 Instant App과 마찬가지로 별도의 web server에 올려놓고 설치할 수 있고, USB를 통해 설치할 수 있다.
설치가 실패한 경우 에러 팝업이 뜨며 설치 실패 사유를 알려준다.

설치 후에는 Installed Widget에서 실행하기를 원하는 widget을 선택 후 Start 버튼을 누르면 해당 widget이 STB 에서 실행된다.
마찬가지로 Uninstall 버튼을 눌러 해당 widget을 설치/제거 할 수 있다.

System Widget tab에서 설치된 unbound application을 교체하거나 삭제 할 수 있다. System Widget 리스트에 설치된 unbound application 리스트가 orgId_appId 형태로 나와 있다. 원하는 unbound application을 선택하고 URL에 교체하기 원하는 App의 주소를 입력하고 Replace 버튼을 누르면 교체된다. 교체가 완료되면 STB 재부팅을 하여야 교체된 application이 실행된다. 그리고 원하는 unbound application을 선택한 상태에서 Uninstall 버튼을 누르면 해당 application이 삭제된다.

Stored Apps tab에서 설치된 bound application을 교체하거나 삭제 할 수 있다. Stored Apps

리스트에 설치된 bound application 리스트가 orgId_appId 형태로 나와 있다. 원하는 bound application을 선택하고 URL에 교체하기 원하는 App의 주소를 입력하고 Replace 버튼을 누르면 교체된다. 교체가 완료되고 해당 채널로 진입하면 교체된 application이 실행된다. 그리고 원하는 bound application을 선택한 상태에서 Uninstall 버튼을 누르면 해당 application이 삭제된다.

실행된 이후에는 Instant App / Widget 모두 Running Apps 목록에 표시되고 각 App에 대해 원하는 동작을 할 수 있다.

- Refresh: 실행중인 어플리케이션 목록 및 상태를 갱신한다.
- Show/Hide: App의 Visibility를 강제 설정한다.
- Activate/Deactivate: App으로 Event전달을 강제 설정한다.
- Destroy: 실행중인 App을 종료시킨다.
- Debug: 실행중인 App을 디버깅한다. App에 대한 Remote Inspector로 연결된다.

WebApp Manager 하단의 Memory Staus를 통해 현재 STB의 memory 상태를 monitoring 할 수 있다. 이 기능을 통해 App이 동작 중 사용하는 메모리 크기를 추정할 수 있다.

- Renew: Initial, Current, Max 모두 메모리 사용량을 현재 사용량으로 update한다.
- Refresh: Current와 Max 메모리 사용량을 update한다. (Initial 값은 변하지 않는다.) Auto Refresh 설정된 주기마다 Current, Max 메모리 사용량을 자동 update한다.
- Initial: Renew버튼을 누른 시점의 메모리 사용량 (Refresh버튼을 눌러도 바뀌지 않는다.)
- Current: 현재 메모리 사용량
- Max: Renew버튼을 누른 시점 이후의 메모리 사용량 최고치

webmw.properties 사용하여 자동 실행시키는 방법

USB flash의 root에 아래와 같은 형식의 webmw.properties 파일을 넣고 부팅하면 설정된 app을 실행시켜 준다.

```
urlApp=<orgId>.<appId>;<url>
```

- orgId: 어플리케이션의 organization ID.
- appId: 어플리케이션의 application ID
- url: 실행할 어플리케이션의 URL

이 방법을 사용하면 박스 부팅과 함께 정해진 어플리케이션을 자동 구동시켜 준다는 장점이 있지만, 어플리케이션을 재구동하려면 박스를 매번 재부팅해야 하는 단점이 있어서 개발 단계에서는 다소 비효율적이다.

2.1.2. 로그 사용법

script내에서 console.log("Test Log") 로 출력하면 터미널 창에 아래와 같은 로그가 출력된다.

```
http://webmw.kt.com/test/test.html:13:Test Log
```

2.1.3. Remote Inspector 사용법

PC web browser에서 다음 형식의 주소로 접속을 한다.

```
http://<STB IP>:19233/inspector.html
```

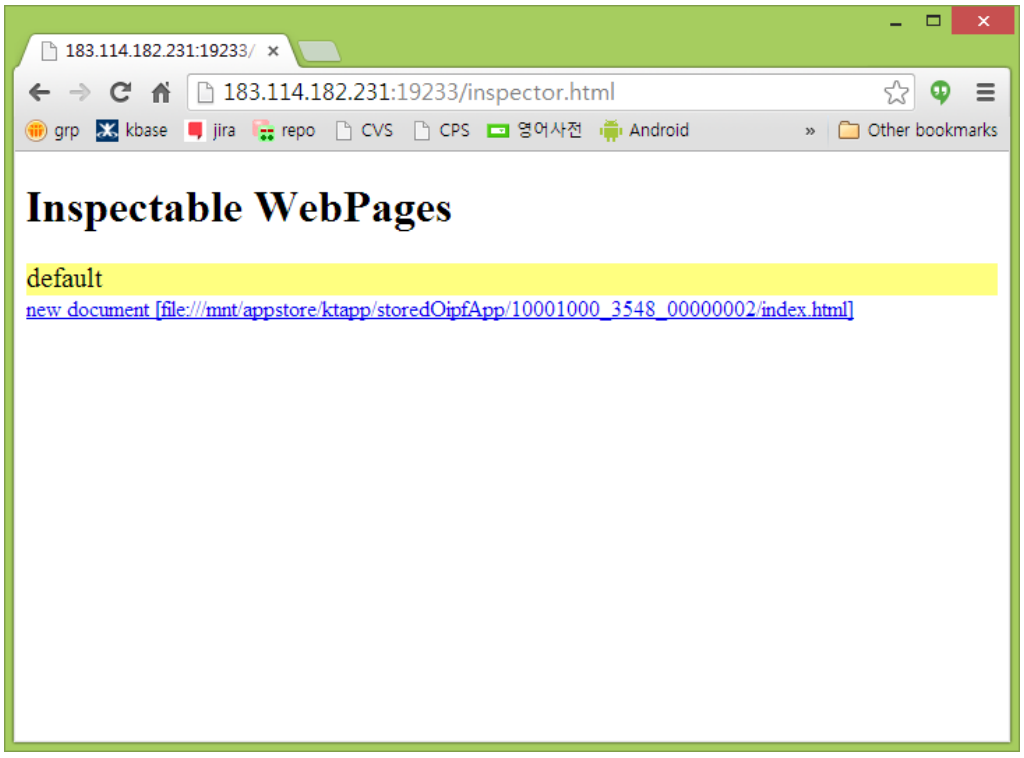


Figure 2 Remote Inspector – Application List

위와 같이 현재 STB에서 동작중인 application 항목이 나오고 이중 debugging할 application을 선택한다.

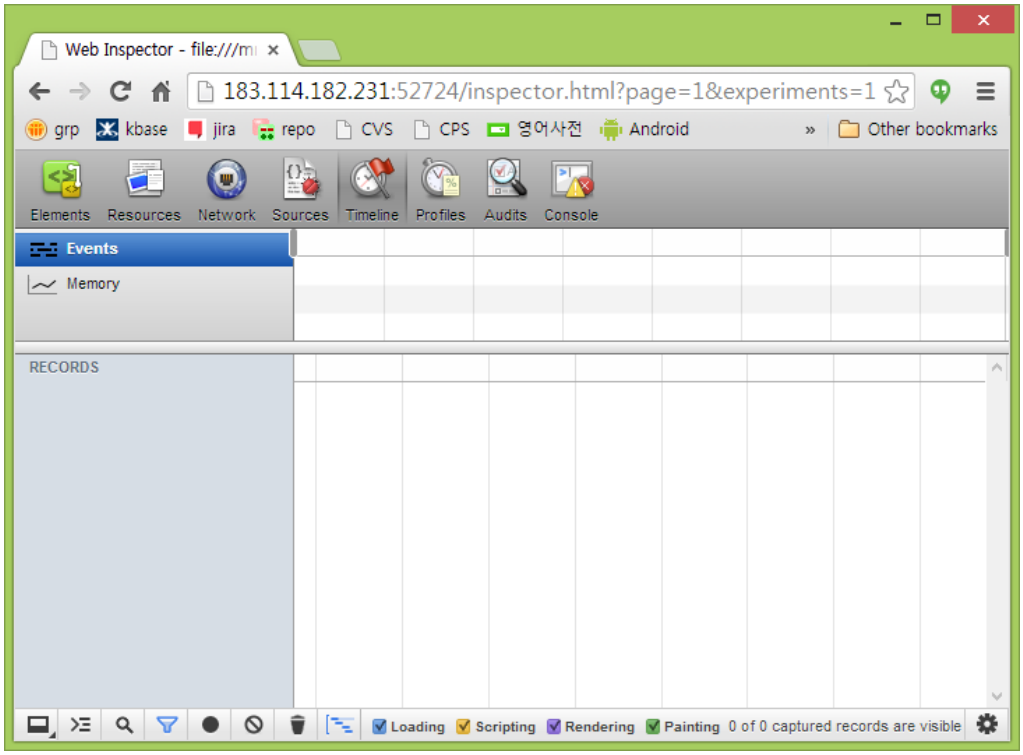


Figure 3 Remote Inspector – Inspector View

Safari나 Chrome의 inspector와 유사한 web inspector를 사용할 수 있다.

2.1.4. How to Check Memory Usage

하나 이상의 어플리케이션이 웹 브라우저 엔진과 같이 동작하기 때문에 정확하게 어플리케이션의 메모리 사용량을 측정할 수 있는 방법은 없다. 단 아래와 같은 방식으로 메모리 사용량 근사값을 측정할 수 있다.

기본

별도의 프로세스로 어플리케이션을 구동시킨 후, 해당 프로세스가 사용한 메모리 사용량을 측정하고, 여기서 기본적으로 웹 엔진이 사용하는 메모리를 제외한 값을 어플리케이션 메모리 사용량으로 간주하는 방식이다.

현재 웹 미들웨어는 기본적으로 다음과 같이 2개의 process로 동작한다.

- **master**: 모든 unbound/host app (예를 들어 navigator, observer와 같은 모든 host application과 home portal, app store와 같은 모든 unbound application)
- **default**: Bound / Independent / User application 등 unbound/host app을 제외한 모든 다른 어플리케이션 (예를 들어 full browser, 독립형 어플리케이션, createApplication()으로 구동되는 모든 child app)

이렇게 구성된 이유는 다음과 같은 위험성으로부터 master browser process에서 동작하는 기본 서비스를 보호하기 위해서이다.

- Bound 어플리케이션 등이 일부 잘못 구현되어 문제를 일으켜도 default browser process만 종료된다.
- 메모리가 부족하면 먼저 default browser process를 종료시켜서 기본 서비스를 계속 제공하면서 메모리 문제를 해결할 수 있다.

또한 이와 같이 process가 구분되어 있기 때문에 createApplication()으로 child application을 구동시키거나 bound app들이 구동되어도 기본적으로 master browser process의 메모리 사용량에는 큰 변화가 없다.

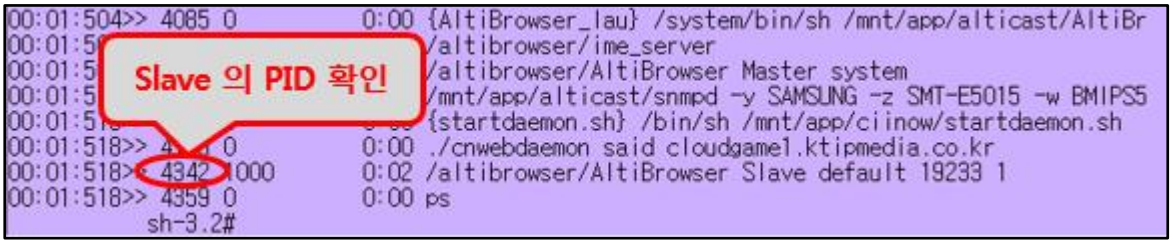
Unbound application의 메모리 사용량 측정 방법

먼저 특정 어플리케이션을 별도 프로세스로 구동하는 firmware[1]를 kt로부터 제공받아야 한다. 이후 다음과 같은 과정으로 메모리를 측정하면 된다.

1. USB flash를 준비한다.
2. Root directory에 slaveApp.properties라는 파일을 만들어서 아래와 같은 format으로 적고 자신의 어플리케이션 app id를 입력한다. 예를 들어 appId=4e30.3001과 같이 입력한다.

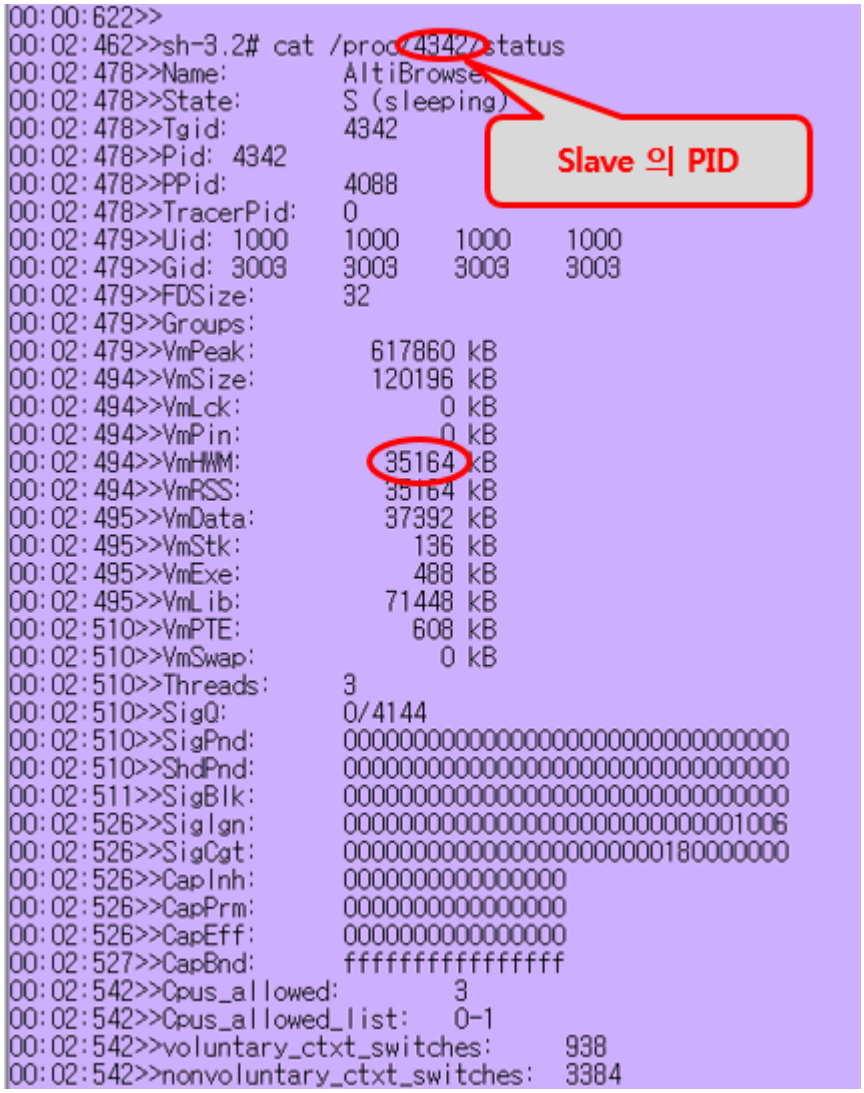
```
appId=<app id to test>
```

3. slaveApp.properties 파일을 저장한 USB flash를 STB에 꼽고 부팅한다.
4. 측정하고자 하는 어플리케이션을 구동시켜서 다양한 시나리오로 어플리케이션을 사용한다.
5. 이후, console을 통해서 "ps"를 입력하면 별도 "Slave"라는 이름의 별도 프로세스를 확인할 수 있으며, 그 PID를 확인한다



[1] 이 firmware 는 메모리 측정용으로 특별 제작된 것으로 unbound application 이 동일한 프로세스로 동작했을 때와는 달리 몇 가지 동작이 타이밍의 이슈로 다르게 동작할 수 있는 문제점이 있을 수 있습니다. 기능 점검용으로는 사용하지 마십시오.

6. console에서 확인된 PID 값을 이용해서 "cat /proc/<PID>/status"를 입력하면 메모리 사용량을 확인해 볼 수 있다.



7. 여기서 VmHWM이 최대 메모리 사용량이며, 어플리케이션이 사용한 메모리 사용량은 대략 "VmHWM – 20MB" 정도라고 보면 된다. 위 예제에서는 "35MB – 20MB"이기 때문에 15MB라고 생각하면 된다.

Bound application의 메모리 사용량 측정 방법

기본적인 측정 방법은 unbound application과 동일하다.

단, bound application은 기본적으로 별도 프로세스로 동작하는데, 다른 bound application이 있지 않은 이상 그 프로세스를 bound application이 독점적으로 사용한다고 보면 되며, 거의 대부분은 두 개 이상의 bound application이 동시에 구동되지 않는다. 그래서 별도의 firmware를 사용할 필요도 없고 USB flash를 사용하여 측정하고자 하는 bound application을 별도로 구동시키도록 할

필요도 없다. 그냥 개발용 firmware에서 직접 어플리케이션을 구동시킨 후, 동일하게 "Slave"라는 이름의 프로세스를 찾은 후 그 프로세스의 메모리 사용량을 확인하면 된다.

2.1.5. Sample Application

간단하게 kt olleh tv web extension framework에서 제공하는 API를 사용하는 어플리케이션을 참고삼아 "Appendix R. Sample Application"에 추가해 두었다.

2.2. API

별도의 "kt olleh tv web extension framework" 규격 문서를 참조한다.

2.3. Compile

어플리케이션 컴파일을 위해서 google closure compiler를 사용하려면 olleh tv WebApp SDK(http://app.tv.olleh.com/webappsdk/)를 참조한다.

Advanced mode로 컴파일하는 방법

Closure compiler를 이용해서 advanced mode로 컴파일을 하기 위해서는 먼저 olleh tv WebApp SDK에서 제공하는 extern.js를 이용해서 다음과 같이 command line으로 컴파일 가능합니다.

```
java –jar compiler.jar –compilation_level ADVANCED_OPTIMIZATIONS –js [source.js]
--externs extern.js
```

Closure compiler의 자세한 사용법은 google closure compiler 웹 페이지(http://code.google.com/p/closure-compiler/)를 참조한다.

2.4. Graphics Resolution

1280x720을 기본 해상도로 한다.

2.5. Font

다음의 font들을 지원하고 있다.

Font Face Name	Descrption
YGO 540	윤고딕 540
Korean iTV SanSerifD	산세리프
KMH	일본어 지원용 폰트

2.6. User Agent

Browser의 User Agent 문구는 다음과 같이 2가지로 나뉘며, 각각 OIPF user agent 문구 형식과 webkit 기반 browser의 user agent 문구 형식에 따라 조금 다르다.

2.6.1. User Agent for OIPF application

웹 미들웨어가 구동하는 웹 어플리케이션은 기본적으로 OIPF 규격에 따르는 user agent 문구를 제공받아야 하기 때문에 아래와 같은 user agent 문구를 얻게 된다.

Format

```
OIPF-OIP/2.2.0
(OITF_HD_UIPROF+TRICKMODE+IPTV_SDS+IPTV_URI+DVB_S2+CONTROLLED+SVG+POINTER+WIDGETS+HTML5_MEDIA; <venderName>; <modelName>; <SWVersion>; <HWVersion>; ) AltiBrowser/<AltiBrowserVersion> (olleh tv;Large Screen) AppleWebKit/<SafariTag>
```

Example

```
OIPF-OIP/2.2.0
(OITF_HD_UIPROF+TRICKMODE+IPTV_SDS+IPTV_URI+DVB_S2+CONTROLLED+SVG+POINTER+WIDGETS+HTML5_MEDIA; SAMSUNG; SMT-E5015; BAC.2012.05.12; 0x01;) AltiBrowser/3.0.0 (olleh tv; Large Screen) AppleWebkit/536.25
```

여기서 <AltiBrowserVersion>은 웹 미들웨어 내 브라우저 업데이트에 의해서 변경될 수 있으며, <SafariTag>는 webkit 업데이트에 따라서 변경될 수 있다.

2.6.2. User Agent of Web Browser

웹 미들웨어의 웹 브라우저를 통해서 진입한 일반 웹 어플리케이션이 얻게 되는 user agent 문구는 webkit 기반의 웹 브라우저의 user agent 문구 형식에 따라 아래와 같은 형식을 갖는다.

즉, 웹 브라우저를 통해서 naver에 진입하면, naver에서 얻은 user agent 문구는 아래와 같은 값이 된다.

Format

Mozilla/5.0 (Linux; olleh tv; U; <Locale>; <vendorName>; <modelName>; <SWVersion>; <HWVersion>;) AppleWebKit/<SafariTag> (KHTML, like Gecko) AltiBrowser/<AltiBrowserVersion> (olleh tv;Large Screen) Safari/<SafariTag>

Example

Mozilla/5.0 (Linux; olleh tv; U; ko-kr;; SMT-E5015;;) AppleWebKit/536.25 (KHTML, like Gecko) AltiBrowser/3.0.0 (olleh tv; Large Screen) Safari/536.25

여기서 <AltiBrowserVersion>은 웹 미들웨어 내 브라우저 업데이트에 의해서 변경될 수 있으며, <SafariTag>는 webkit 업데이트에 따라서 변경될 수 있다.

2.7. STB H/W Spec

현재 Web Middleware를 제공하고 있는 단말의 STB spec이다.

	Smart STB (BCM7356)	UHD STB (BCM7252)
CPU	1.3GHz	
RAM	DDR3 1GB	
Flash	4GB	
OpenGL	O	

Table 2 STB H/W Spec

3. 개발 유의 사항

기본적으로 본 문서에 언급되지 않은 사항은 kt olleh tv web extension framework를 따른다.

웹 어플리케이션 개발에 대한 기본적인 reference web site는 "Appendix B Reference Web Sites for Developer"를 참고한다.

3.1. 웹 어플리케이션 소스 보호

웹 어플리케이션 소스 보호를 위하여 Google Closure Compiler 사용을 권장한다. 컴파일 시 코드 압축 및 보안성 향상을 위해서 "Advanced" 옵션을 사용해야 하며, 컴파일 결과에 따른 Side effect 확인을 위해, 어플리케이션 개발 과정 중 주기적으로 컴파일 테스트 진행이 필요하다.

3.2. Single Thread Model

모든 JavaScript 작업은 single thread로 동작한다. 결국 특정 JavaScript에서 긴 작업을 수행하게 되면 event handling이 자칫 상당히 밀려서 수행이 될 수 있다. 그래서 기본적으로 모든 JavaScript 구현은 짧은 시간에 리턴될 수 있도록 구현해야 한다.

만약 긴 시간이 소요되는 작업을 수행해야 할 경우 보통 setTimeout()을 이용하여 짧게 끊어서 작업을 진행하도록 구현하여 오랜 시간 동안 key event 처리를 못하는 문제가 발생되지 않도록 해야 한다. 특히 이런 문제는 동시에 동작하는 모든 어플리케이션에 영향을 미치기 때문에, 자신의 어플리케이션뿐만 아니라 다른 모든 어플리케이션의 이벤트 반응 속도도 느리게 만든다.

다음은 긴 작업을 setTimeout()을 이용해서 끊어 처리하는 예제이다.

```
function doCalculation()
{
    //do your thing for a short time

    //figure out how complete you are
    var percent_complete=....

    return percent_complete;
}

function pump()
{
    var percent_complete=doCalculation();

    //maybe update a progress meter here!
```

```
    //carry on pumping?
    if (percent_complete<100)
    {
        setTimeout(pump, 50);
    }
}

//start the calculation
pump();
```

특히 여러 어플리케이션이 동시에 수행되는 상황이기 때문에 더욱 더 자신의 어플리케이션이 긴 시간 동안 JavaScript에서 리턴하지 않아서 다른 어플리케이션의 event 처리 등을 많이 지연시키는 일이 생기지 않도록 특별히 유의해야 한다.

그래서 하나의 javascript function이 최대 300ms 이내로 리턴될 수 있도록 만들어야 한다. 단, 어쩔 수 없이 다소 긴 작업을 해야 할 때도 1초 이내로 리턴될 수 있도록 구현해야 한다. 그래야 사용자가 key event에 대해서 1.5초 이내의 적절한 반응 속도를 얻을 수 있다.

3.3. 그래픽 메모리 사용 최적화

Web application이 새로 생성되는 것 때문에 추가적으로 사용되는 메모리는 수십 KB 수준이다. 그런데 문제는 window 크기에 따라 graphic buffer 메모리가 필요하게 되는데, 이게 1280X720 기준으로 8MB 정도 사용하게 된다. [1]

그래서 다음과 같은 지침을 준수하기를 권장한다.
가급적 child application이나 여러 어플리케이션으로 나누지 않는다.
각 어플리케이션은 display를 위한 최소 크기의 window를 할당한다. (Window.resizeTo() 이용)

- 항상 떠 있는 daemon형 어플리케이션의 경우에는 화면을 지울 때 window 사이즈도 "0 X 0"으로 만들어서 불필요하게 사용하는 graphic buffer를 해제한다. 다시 display 시작할 때는 window size를 원래 사이즈로 resize해야 한다. 이를 편하게 사용하게 하기 위해서 Application.hide(true)이라는 확장 JavaScript API가 제공되는데 이는 내부적으로 hide()한 후 window size를 "0 X 0"으로 줄이는 작업을 하며, Application.show()이 호출되면 window size를 원래 상태로 자동으로 복구한다.

[1] 4 byte color 를 사용하기 때문에 1280 X 720 X 4 = 4MB 정도를 graphic buffer 로 사용하게 된다. 그런데 깜박임 문제를 해결하기 위해서 double buffering 을 사용하기 때문에 graphic buffer 가 2 배로 필요하다. 그래서 1280 X 720 기준의 window 생성을 위해서는 총 8MB 가량의 메모리가 필요하게 된다.

이러한 사항을 준수한다면 많은 daemon형 unbound application을 기획한다고 해도 통상적으로 한 순간에 4개 이상의 어플리케이션이 뭔가를 보이는 경우가 드물다고 가정하면, bound application을 포함해서 32MB 정도의 메모리로 web application의 window에 필요한 graphic buffer memory를 해결할 수 있을 것으로 예상된다.

3.4. UI 개발 최적화

3.4.1. JavaScript UI Framework

일반적으로 UI 효과를 내기 위해서 가장 널리 사용되는 방법 jQuery나 Sencha인데, 이건 고성능 환경을 감안한 library이기 때문에 이것을 그대로 사용하는 것은 권장하지 않는다. 꼭 사용해야 한다면 jQuery Mobile을 사용하기를 권장한다.

3.4.2. Animation

Animation을 위해서 3D effect나 여러 CSS animation을 동시에 수행하는 것은 고성능을 요구하므로 권장하지 않는다. 한 순간에 하나의 CSS animation 정도를 동작시키는 것을 권장한다.

3.4.3. CSS Shadow

CSS에서 제공하는 text shadow 속성을 많이 사용하면 paint에 대한 성능이 저하될 수 있기 때문에 권장하지 않는다.

```
<html la”g=”ko">
<head>
    <meta chars”t="ut”-8">
    <title>CSS | Rounded Border </title>
    <style>
        p {
            font-size: 30pt;
            font-family: Georgia;
        }
        .shadow_one {
            text-shadow: 5px 5px 5px gray; // Text Shadow  속성
        }
    </style>
</head>
<body>
    <p cla”s="shadow_”ne">This is Text Shadow Effect!!</p>
</body>
</html>
```

3.4.4. Background Attachment

CSS에서 사용하는 background 속성 중에 background-repeat:repeat와 background-attachment:fixed는 페이지 스크롤시 repaint 영역이 전체 화면에 걸쳐 발생하기 때문에 가급적 사용하지 않도록 한다

```
body {
    background-image:u'l('smiley.'if');
    background-repeat:repeat;        // background에 이미지를 연속적으로 display
    background-attachment:fixed;   // 스크롤 시에도 Background는 고정
}
```

3.4.5. Event Handling Optimization

동시에 여러 어플리케이션이 동작하는 웹 미들웨어이기 때문에 하나 이상의 어플리케이션이 key event를 listening할 수 있다. 그런데 만약 자신의 어플리케이션이 처리하거나 혹은 key event를 다른 어플리케이션이 처리하지 못하도록 막을 목적이 아니라면 key event를 처리하지 않도록 구현해야 한다. 만약 그렇게 하지 않으면 불필요하게 항상 해당 어플리케이션에게 key event가 전달되어 전반적으로 단말의 서비스 성능이 저하된다. 특히 event 처리는 어플리케이션의 visibility와는 무관하기 때문에 invisible 상태에서도 KeySet을 해제하거나 필요한 최소한의 KeySet으로 설정하지 않으면 불필요하게 key event를 전달받게 되고 이로 인해서 전체적인 서비스 성능이 저하되게 된다. 반드시 필요한 경우에 한해서 필요한 key event만 전달받도록 KeySet을 설정하도록 한다.

특히 다음과 같은 점을 추가적으로 고려해야 한다.

- 나가기, 채널 up/down, 볼륨 up/down 등의 키는 기본적으로 설정하지 않아야 한다.
- 숫자 입력 화면을 제외하고는 숫자 키를 keyset에 설정하지 않아야 한다. Keyset에 설정하게 되면 DCA 동작이 되지 않는다.
- Unbound application이 아닌 일반 어플리케이션의 경우, 앱 최상위 화면에서는 이전 키를 처리하지 않아야 한다. 이전 키로 어플리케이션을 빠져 나가는 기능을 navigator가 처리하기 때문이다.

3.4.6. Avoiding Unnecessary Object Creation

기본적으로 object를 많이 만들게 되면 garbage collection으로 인해서 일시적인 성능 저하를 일으킬 수 있기 때문에 가급적 object를 재사용하거나 불필요하게 object를 생성하지 않도록 구현해야 한다.

Case Study: setInterval()

주기적인 작업 반복을 위해서 setInterval()을 사용할 경우 다음과 같이 호출하게 되면 timer object가 계속 쌓이게 됩니다. 그래서 불필요하게 garbage collection이 발생할 수 있습니다.

```
setInterval(function() { doSomething(); }, 1000);
```

이런 경우는 다음과 같이 clearInterval()를 호출하여 timer object가 쌓이는 것을 막아 줘야 합니다.

```
function a() {
	var t = setInterval(function() {
			doSomething();
			clearInterval(t); // 기존 timer 삭제!
			a();
		}, 1000);
}
```

3.4.7. 반응성 좋은 UI 만드는 방법

리모컨을 빠르게 누르거나 계속 누르고 있으면 늦은 화면 업데이트로 중간 과정이 생략되어 반응성이 안 좋게 느껴진다. 예를 들어 메뉴 이동 중 포커스를 빠르게 이동하면 가끔 일부 셀을 건너뛰는 경우가 있다. 이러한 현상을 requestAnimationFrame 메서드를 이용하여 반응성을 좋게 개선할 수 있다. requestAnimationFrame 메서드는 브라우저가 UI 업데이트를 한 후 callback으로 어플리케이션에 알리도록 되어 있다. 이에 키 이벤트를 전달 받은 후, requestAnimationFrame callback이 호출 되기 전에는 키 이벤트를 무시하도록 한다. 이렇게 하면 화면 업데이트 중 발생하는 키 이벤트는 무시되어 반응성이 좋아 보일 수 있다. 물론 이렇게 구현할 경우 사용자는 5번 아래로 이동했지만, 4번만 포커스 이동이 되는 문제가 있다. 단 이렇게 동작하는 게 사용자에는 더욱 자연스러워 보이는 경우가 있기 때문에 이렇게 구현하는 것도 고려해 볼 수 있다.

아래는 sample code 이다.

```
var frameReady = true;

function frameReadyCallback() {
	frameReady = true;
}

var requestAFrame = (function() {
		return window.requestAnimationFrame ||
			window.webkitRequestAnimationFrame;
})();
```

```
document.addEventListen“r("keyd”wn", function (event) {
      if(!frameReady) {
            event.preventDefault();
            event.stopPropagation();
            return ;
      }
      }
      frameReady = false;

      requestAFrame (frameReadyCallback);
      m
});
```

3.5. HTML 작성 유의 사항

3.5.1. DOCTYPE

HTML을 작성할 때,

- DTD를 표시하지 않거나
- <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">를 사용하는 경우가 종종 있다.

이와 같은 경우는 브라우저가 웹 페이지 렌더링을 Quirks Mode[1]로 처리하기 때문에 권장하지 않는다.

따라서 권고하는 DTD는 다음과 같다.

- **HTML 4.01**
  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" http://www.w3.org/TR/html4/loose.dtd>
- **XHTML 1.0**
  <!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd>
- **HTML5**
  <!DOCTYPE HTML>

그러나 요즘은 표준을 준수하는 브라우저도 다양해졌고, 이미 많은 웹 개발자들이 해당 내용을 잘 알고 있어서 큰 문제는 없을 것 같다. 다만 DOCTYPE이 없으면 브라우저는 해당 page를 비표준 모드로 해석하므로, 개발하는 HTML 페이지의 버전에 맞는 DOCTYPE의 선언은 브라우저가 해당 page를 표준 모드로 해석할 수 있도록 하기 위해 필요한 사항이다.

[1] Quirks mode 는 브라우저가 하위 호환성을 보장하기 위해서 처리하는 방식이다.

따라서,
오래된 웹 페이지들을 그대로 재사용하거나 해당 코드를 그대로 복사하면서 DOCTYPE을 신경 안 쓰게 되는 상황을 방지하고
브라우저가 비 표준 모드로 처리하게 않게 하기 위해서
HTML 코드 시작 부분에 HTML5의 DOCTYPE인 <!DOCTYPE HTML>을 선언하여 사용하는 것을 권고한다.

3.5.2. Margin, Padding

웹 페이지가 TV 화면의 크기와 동일할 때 웹페이지의 우측/하단에 스크롤 바가 디스플레이 될 수 있다. 이 경우를 방지 하기 위해 body의 margin, padding을 0으로 설정해야 한다.

```
<style>
body {
	maring: 0;
	padding: 0;
}
</style>
```

3.5.3. Single document

어플리케이션이 여러 개의 html 페이지로 구성 되어 사용자 동작에 의해 html 페이지 간 전환이 이루어지는 경우, 새로운 어플리케이션이 생성되는 것과 동일한 시간이 소요된다.
따라서 사용자 동작에 대해 빠른 응답을 주기 위해서는 하나의 html 페이지로 어플리케이션을 구성하고, div 사용하여 화면을 전환해야 화면 전환이 빨라진다.

3.6. JavaScript 작성 유의 사항

3.6.1. Strict Mode

Strict mode를 사용하면 더 나은 오류 검사를 코드에 적용한다. 예를 들어 암묵적으로 선언된 변수를 사용하거나 읽기 전용 속성의 값에 값을 할당하거나 할 수 없습니다.

Strict Mode 선언

파일, 프로그램 또는 함수의 시작 부분에 "use strict"를 추가하면 됩니다. 이 종류의 선언을 directive prolog (지시문 프롤로그)라고 합니다. 기본적으로 파일이나 프로그램 전체에 "use strict"를 추가하기를 권고하며 아래와 같이 추가합니다.

```
"use strict";          // Declare directive prolog

function testFunction() {
    var testvar = 4;
    return testvar;
}

// This causes a syntax error.
testvar = 5;
```

Strict Mode에서의 제약 사항

자세한 strict mode에서의 제약 사항에 대해서는 "Appendix E"를 참조한다.

3.6.2. XMLHttpRequest

JavaScript를 이용하여 서버와 직접 통신하기 위해서 XMLHttpRequest를 사용할 경우 synchronous 방식을 사용하지 않도록 한다. 동기화 방식을 사용하기 위해서는 open()을 통해서 async 여부에 true를 전달하면 되는데, 이렇게 될 경우 response가 올 때까지 JS가 정지하고 응답을 대기하게 되므로 그 동안 사용자의 입력을 처리하지 못하고 화면이 정지된 것 같은 현상이 발생한다. 그래서 synchronous XMLHttpRequest는 사용하지 않도록 한다.

open(string method, string ur22ooleanean async, string username, string password)

- **method**: HTTP 요청 방식을 지정하며 POST, GET, PUT 을 사용할 수 있다
- **url**: 요청 받는 서버측 URL.
- **async**: 요청을 비동기를 처리할지 여부를 지정. 지정하지 않으면 비동기 처리인 true. 비동기로 지정하면 요청이 처리되기 전에 다른 작업을 할 수 있다.
- username, password: 로그인 정보

3.6.3. Document.cookie

JavaScript 내에 cookie 값을 가져와 사용할 수 있는데, Document.cookie의 get()/set()은 모두 동기 방식으로 동작한다. 상황에 따라서는 response가 전달될 때까지 시간이 걸릴 수 있어서 document.cookie를 사용하지 않도록 한다.

3.6.4. IndexedDB

Indexed DB로는, webkit indexed DB만 제공되고 있다. webkit indexed DB는 모든 property에 webkit prefix가 붙는다. 그래서 현재 다음과 같은 property들이 window object 아래 property로 제공되고 있다.

- webkitIDBFactory
- webkitIDBDatabase
- webkitIDBRequest
- webkitIndexedDB
- webkitIDBCursor
- webkitIDBIndex

…

3.7. Memory Usage

일반적으로 하나의 어플리케이션은 메모리 사용량이 50MB 이하이기를 권고한다. 만약 이것보다 큰 경우에는 kt 담당자와 협의를 해야 한다.

왜냐하면 STB에서 bound application을 위해서 50MB 정도의 메모리만 확보할 예정이기 때문이다. 또한 하나 이상의 어플리케이션으로 구성하며, 이 어플리케이션이 동시에 구동될 수 있다면 그 어플리케이션 그룹 전체가 50MB 이하의 메모리를 사용하도록 구성되어야 한다.

어플리케이션의 메모리 사용량은 remote inspector의 timeline tab의 memory 항목을 이용하여 최대치가 50MB를 넘지 않도록 설계한다.

3.8. Restriction on Application Destruction

Application이 onApplicationDestroyRequest에서 수행하는 동작은 가끔 이후 다른 application이 수행한 동작에 잘못된 영향을 미칠 수 있다. 그래서 일부 기능이 동작하지 않도록 안전 장치를 제공한다.

Application이 종료될 때 video를 resize하면 타이밍상 다른 application이 설정한 video size를 변경시켜서 최종적으로 잘못된 결과를 낳을 수 있다. 그래서 onApplicationDestryRequest가 호출되는 시점 이후에는 video resize가 동작하지 않는다.

4. Application Basics

4.1. Application Identifier

여러 가지 이유로 특정 어플리케이션을 가리키는 방법이 필요하다. 송출하는 어플리케이션에 대해서 organization ID와 application ID를 이용해서 어플리케이션을 식별하도록 한다.

포맷은 다음과 같다.

```
<orgIdInHex>.<appIdInHex>
```

- 16진수로 표시하며 "0x"와 같은 값을 앞에 적지 않는다.
- 앞에 존재하는 "0"은 제거한다. 예를 들어 0x00004e30이면 그냥 4e30만 적는다.
- 영문자는 소문자로 한다.

이렇게 정확하게 제약하는 이유는 어플리케이션에서 단순히 문자열 비교만으로도 손쉽게 어플리케이션을 구분하게 하기 위해서이다. 또한 16진수를 사용하는 이유는 값이 10진수 기준으로는 너무 복잡한 값이기 때문이다.

그래서 organization ID가 0x00004e30이며, application ID가 0x3001인 경우에 전체 어플리케이션 식별자 문자열은 다음과 같다.

```
4e30.3001
```

Current Unbound Application ID

현재 정의된 unbound application ID는 다음과 같으며, 아직 송출 관련 정보가 확정되지 않았기 때문에 임시로 기존 ACAP의 application ID를 그대로 사용한다. 이 값은 추후 변경될 수 있다.

Application Name	Application ID
Observer	4e30.3000
Home Portal	4e30.3001

4.2. Application Type

OIPF에서는 다음의 3가지 어플리케이션 형태를 규정하고 있다.

- **Service provider related application**: service provider가 제공하는 어플리케이션으로써, kt에서 unbound application에 해당한다.
- **Broadcast related application**: broadcast channel에 AIT로 signaling되는 어플리케이션으로써, kt에서 bound application에 해당한다.

- **Broadcast independent application**: 위 2가지 경우에 포함되지 않는 어플리케이션으로, kt에서 independent application와 AppStore에 등록되는 user application에 해당한다.

여기서 어플리케이션 개발자에게 가장 중요한 부분은 초기 상태이다.

- Service provider related application & broadcast related application
  - Invisible 상태로 초기화된다. 어플리케이션이 Application.show()를 호출해야 visible 상태가 된다.
  - Inactive 상태로 초기화된다. 어플리케이션이 Application.activateInput()을 호출해야 event를 받는 active 상태로 된다.
  - Unfocused 상태로 초기화 된다.Application.activateInput(true)을 호출해야 focused 상태로 된다.
- Broadcast independent application
  - Visible 상태로 초기화된다.
  - Active 상태로 초기화된다.
  - Focused 상태로 초기화된다.

일반적인 웹 어플리케이션의 초기 상태와 동일하다.

결국 어플리케이션 개발자는 이 어플리케이션이 kt 서비스 내에서 어떤 형태로 운영될 것인지를 먼저 파악한 후, 그에 따른 초기 상태에 맞게 구현해야 한다.

- **Unbound / bound application**: 초기 상태가 invisible / inactive / unfocused가 된다.

Independent / user application: 초기 상태가 visible / active / focused가 된다.

4.3. Application Property

기본적으로 application type에 따라서 어플리케이션의 초기 상태가 정해진다. 하지만 새로운 어플리케이션 시나리오가 추가되어 기존 application type에 해당 되는 것이 없다면 application property을 전달하여 직접 초기 상태를 설정할 수가 있다. 아래 표는 지원하는 property이다.

Key	Value Type	Note
initWindow.visible	Boolean	true 이면 visible 상태로 초기화 되고 false이면 invisible 상태로 초기화 된다.
initWindow.active	Boolean	true이면 active 상태로 초기화 되고 false이면 inactive 상태로 초기화 된다.
initWindow.focused	Boolean	true이면 focused 상태로 초기화 되고 false이면 unfocused 상태로 초기화 된다. ※ 실행 시키는 어플리케이션이 어떠한 어플리케이션 위에서 뜰 수 있다면 focus를 가져가면 다양한 문제가

		발생 가능하다. 어플리케이션 위로 겹치게 실행이 된다면 이 값은 false로 설정되어야만 한다.

4.4. How to Start Application

어플리케이션을 새로 구동시키는 방법은 크게 2가지가 있으며, 이렇게 구동된 어플리케이션의 초기 상태는 다음과 같다.

Application.createApplication(String uri, Boolean createChild, Map properties)

- uri가 http, https로 어플리케이션 그 자체 혹은 AIT URL을 가리킬 경우에는 visible / active 상태로 초기화된 어플리케이션으로 구동된다. AIT URL인 경우에는 해당 요청의 MIME type이 "application/vnd.dvb.ait+xml"여야 한다.
- uri가 그 외 scheme으로 구동될 때는 invisible / inactive 상태로 초기화된 어플리케이션으로 구동된다. 대표적인 경우가 dvb URL을 사용할 경우이다.
- Properties는 정의된 OIPF API를 확장한 것 이다. 이것을 전달하지 않으면 application type에 따라 properties가 기본적으로 설정이 된다. Application type 외에 추가로 설정이 필요하다면 properties에 해당 값을 넣어서 전달하면 어플리케이션 초기화 시 적용이 된다.

Application.startWidget(WidgetDescriptor wd, Boolean createChild)

- visible / active 상태로 초기화된 어플리케이션으로 구동된다.

그래서 kt application type에 따라 다음과 같은 방법으로 초기화해야 해당 어플리케이션의 초기 상태에 맞게 구동될 수 있다.

Type of Application to Start	API	Note
Unbound Application	createApplication("dvb://atts.ait/4e30.3001")과 같이 ATTS에 해당하는 dvb URL을 사용하여 어플리케이션을 초기화해야 한다.	widget이지만, startWidget()으로 초기화하면 visible / active 상태로 초기화되기 때문에, 일부러 dvb URL 방식으로 createApplication()으로 구동해야 한다. 또한 kt에서는 observer만 이런 방식으로 unbound application을 구동해야 한다.
Bound	createApplication("dvb://current.ait/1000	

Application	a.3010")과 같이 dvb URL을 사용하여 어플리케이션을 초기화한다.	
User Application	startWidget()으로 구동한다.	
Independent Application	createApplication("http://...")과 같이 http나 https scheme을 사용하는 어플리케이션 혹은 AIT URL을 이용하여 어플리케이션을 구동한다.	

4.5. How to Manage Application

기본적으로 모든 어플리케이션은 해당 어플리케이션을 구동한 어플리케이션이 그 어플리케이션의 life cycle을 관리할 책임을 진다. 즉 어플리케이션 종료를 보장해야 한다. 손쉽게 어플리케이션 종료를 보장하는 방법은 child application으로 구동시키는 것이다. createApplication(uri, true, properties)나 startWidget(wd, true)와 같이 2번째 parameter를 true로 전달하면 child application으로 구동시키기 때문에 구동시킨 parent application이 종료될 때 자동적으로 같이 종료되게 된다.

또한 unbound application은 기본적으로 직접 구동하지 않는 것을 기본으로 한다. 그래서 observer를 통해서 구동해야 한다.

5. Graphics & Events

5.1. General

5.1.1. Window Layer

기본적으로 OIPF DAE 규격에 따라, 각 어플리케이션이 Application.activateInput(Boolean)을 호출하면 해당 window가 최상위로 이동하게 된다.

5.1.2. Event Handling

사용자 입력에 따른 키 이벤트는 현재 activateInput()을 호출하여 활성화되어 있는 어플리케이션에게 전달되며, 어플리케이션은 자신이 필요로 하는 키 이벤트 세트를 Application.privateData.keyset 프로퍼티에 미리 등록한다.
어플리케이션은 동작 상태에 따라 keyset 프로퍼티의 값을 변경하여 불 필요한 키 이벤트를 수신하거나 consume 하지 않아야 한다.

어플리케이션이 primary receiver이며 focus를 가지고 있으면, DOM Event 방식에 따른 event 가 전달 되며 그렇지 않을 경우 – primary receiver가 아니거나 focus를 가지지 못한 경우 – Application.onKeyXXX와 같은 형태의 핸들러를 등록하여 이벤트를 전달 받게 된다. 따라서 primary receiver가 아니거나 focus가 뺏겨도 처리하고자 하는 event에 대해서는 기존 DOM event 방식으로 등록한 핸들러 외에 Application object에도 동일한 핸들러를 추가로 등록하여야 키 이벤트를 전달 받을 수 있다. 그래서, 하나의 이벤트에 대하여 primary receiver이면서 focus를 가진 어플리케이션에게는 DOM Event로 전달되며 Application.onKeyXXX 방식으로는 전달되지 않는다. 반면 primary receiver가 아니거나 focus를 가지지 않은 어플리케이션에게는 DOM event로 전달되지 않고, Application.onKeyXXX를 통해서 event가 전달된다. 따라서 activateInput(false)로 focus를 얻지 않는다면 항상 Application.onKeyXXX에 의해서만 event를 전달받게 된다.

또한, 어플리케이션은 event의 stopPropagation()를 통해 해당 이벤트를 consume 할 수 있다. Consume 된 이벤트는 다른 어플리케이션으로 전달 되지 않는다.

```
function handleKeyEvent {
    switch (e.keyCode) {
        case VK_0:
        case VK_1:
            // do something
            ……
            // event consumed
```

```
            e.stopPropagation();
            return;
    }
}

document.addEventListener("keydown", handleKeyEvent);
var appMgr = oipfObjectFactory.createApplicationManagerObject();
var self = appMgr.getOwnerApplication( window.document );
self.onKeyDown = handleKeyEvent;

self.activateInput(true);      // This function makes this application receive events
```

5.1.3. Initial State

Visibility & Activeness

OIPF 규격에 의해서 service provider related application과 broadcast related application은 초기 상태가 invisible이며, activateInput()을 호출하기 전까지는 event도 받지 않는다. 반면 broadcast independent application은 초기 상태가 visible이며, activateInput()을 호출하지 않아도 event를 받게 된다. 위 규격을 준수하도록 어플리케이션을 구동하는 어플리케이션이 적절한 API를 사용하여 구동하게 된다. 그래서 구현하려고 하는 어플이 kt 내에서 어떤 application type[1]인지 확인하고 이에 따라 초기 상태가 결정된다.

Key Set

또한 KeySet을 특별히 등록하지 않으면, 아래의 key에 대해서 등록된 것으로 간주한다.

- **Color keys** – VK_RED, VK_GREEN, VK_YELLOW, VK_BLUE
- **Navigation keys** – VK_UP, VK_DOWN, VK_LEFT, VK_RIGHT, VK_ENTER, VK_BACK
- **VCR keys** – VK_PLAY, VK_PAUSE, VK_STOP, VK_NEXT, VK_PREV, VK_FAST_FWD, VK_REWIND, VK_PLAY_PAUSE
- **Scroll keys** – VK_PAGE_UP, VK_PAGE_DOWN
- **Info key** – VK_INFO
- **Numeric keys** – VK_0~VK_9
- **Alpha keys** – all alphabetic events

Background

Background color는 기본적으로 어플리케이션이 지정해야 한다. 만약 특별히 지정하지 않으면, transparent로 처리된다.

[1] "4.2 Application Type" 참조

5.2. Prioritized Graphics & Events

5.2.1. Limitation of OIPF

OIPF DAE 규격에서는 모든 어플리케이션이 대등한 관계를 형성하기 때문에 시스템 팝업이나 closed caption과 같이 항상 상위에 떠야 한다거나 하위에 떠야 하는 경우를 처리할 수 없다. 예를 들어 closed caption이 표기를 시작했을 때 그 때 해당 어플리케이션이 activate하게 되면 지금 표기하고 있는 홈 메뉴보다 위에 그려지는 것을 피할 수 없다. 반대로 network 오류가 발생해서 시스템 전체 오류를 표기하기 위해서 시스템 팝업을 띄워도, background로 실행 중이던 어플리케이션이 activate하게 되면 시스템 팝업을 가려버리게 된다.

이를 해결하기 위해서 OIPF 규격 기반 위에 특정 priority를 지정할 수 있는 기능을 추가하여 특정 priority의 window를 생성하거나 event를 처리하는 기능을 추가적으로 제공한다. OIPF DAE 규격대로 구현한 모든 어플리케이션은 priority 0의 우선순위로 간주되며 그들 간에는 OIPF DAE 규격대로 동작하게 된다. 그런데 특별하게 일반적인 어플리케이션이 동작하는 layer의 상위 혹은 하위에 그리거나, 일반적인 어플리케이션보다 먼저 event를 가로채고 싶다면 확장 API를 사용해야 한다.

5.2.2. Extended API

kt에서는 이 문제를 해결하기 위해서 다음과 같이 window와 event handler에 대해서 priority를 부여하는 방식을 추가적으로 지원한다.
일반적인 어플리케이션은 본 API를 사용하지 않고, activateInput(Boolean) 형태의 API를 사용해야 한다.

```
/* Window Layering */
Application Class {
    void activateInput(Boolean gainFocus, Integer eventHandlerPriority, Integer windowPriority )
}
```

- **eventHandlerPriority**: KeySet에 해당하는 event handler priority. 정수 값으로 큰 수일수록 먼저 event를 받게 된다. 0이면 표준과 동일하게 동작. 양수이면 0보다 먼저 받게 되고, 음수이면 0보다 나중에 event를 전달받게 된다. 만약 값을 주지 않으면 priority가 0인 것으로 간주한다.
- **windowPriority**: Window의 priority. 정수 값으로 큰 수일수록 사용자를 기준으로 위 쪽에 그려진다. 양수이면 0인 표준 어플리케이션보다 위에 그려지고, 음수이면 0인 표준 어플리케이션보다 하단에 그려진다. 이 값이 없으면 event handler priority와 동일한 priority로 동작한다.

5.2.3. Cross-application Event Handling

Priority 개념이 없는 표준 하에서는 나중에 activateInput()을 호출한 어플리케이션이 항상 키 이벤트를 먼저 받게 된다. 하지만 priority 개념이 도입된 이 시스템에서는 priority가 active 순서보다 선행되기 때문에 priority가 높은 어플리케이션이 event를 먼저 받게 된다. 단 동일한 priority 사이에는 늦게 activate된 어플리케이션이 먼저 event를 받게 된다.

5.2.4. Typical Usage

어플리케이션의 기능이 자연스럽게 동작하기 위해서 prioritized window나 event handler를 사용하게 되는 경우는 다음과 같다.
Home Portal에서 System Popup이나 예약 프로그램 알림 팝업은 일반적으로 상위에 그려져야 하고, 또한 event handling도 먼저 처리해야 하기 때문에 별도의 positive prioritized window와 event handler를 사용해야 한다.
Closed caption은 영상의 상위에 그려지지만, 다른 일반 graphics 보다는 항상 아래에 그려지는 것이 적절하기 때문에 negative prioritized windows를 사용해야 한다. 만약 event도 처리해야 한다면 negative prioritized event handler를 사용해야 한다.
Hot key를 처리하는 어플리케이션은 일반적으로 상당히 높은 positive prioritized event handler를 등록하여 다른 일반 어플리케이션이 특별한 처리를 하지 않아도 먼저 key를 가로채서 잘 처리하는 것이 적절하다.[1]

5.2.5. Remark

기본적으로 일반 어플리케이션이 priority를 설정하여 window를 생성하거나 event handler를 등록하는 경우는 크게 없는 것으로 간주한다. 만약 부득이하게 특정 priority로 window나 event handler를 생성해야 한다면, 주요 kt service app들이 사용하는 priority를 감안하여 적절한 priority에 대해서 kt와 협의를 해야 한다.

또한 어플리케이션이 구동 중에 일부 key event들을 못 받을 수 있다. 왜냐면 event 전달 우선순위에 의한 상위의 어플리케이션이 key event를 stopPropagation()을 통해서 하위의 어플리케이션에게 통상적으로 전달을 막도록 구현하기 때문이다.

[1] Event handling 만 하면 되는 경우에는 반드시 window size 를 0X0 로 설정하여 불필요한 그래픽 메모리 사용을 막아야 한다.

5.3. Video Layering

Video Layer Issue

기존 ACAP에서는 표준에 따라 background plane, video plane, graphic plane이 순서대로 위치하는 방식으로 설계되어 있다. 따라서 어플 A가 그린 graphics는 어플 B가 재생하는 video보다도 항상 위에 그려지게 된다.

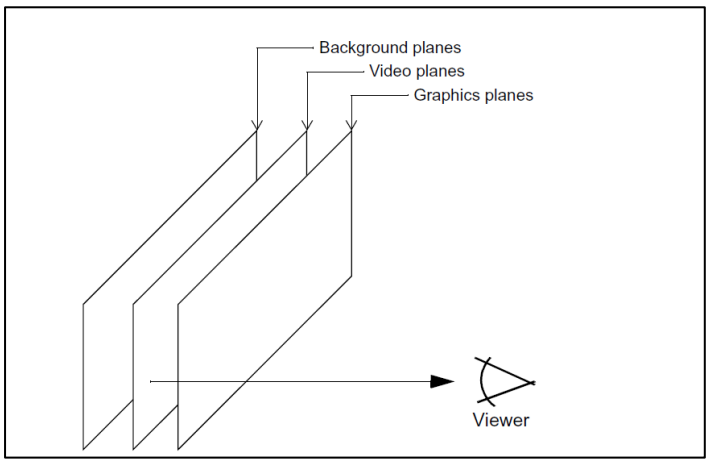


Figure 4 ACAP display planes

그런데 web middleware에서는 video가 어플리케이션에 embedded되는 형태이기 때문에 video도 graphics의 layer와 동일하게 처리된다. 즉 어플리케이션 A가 어플리케이션 B보다 graphics layer가 상위에 있으면 어플리케이션 A의 video는 어플리케이션 B의 graphics를 덮게 된다. 이는 실제 play되고 있는 video 영역 기반이 아니라 해당 video가 화면에서 차지하고 있는 공간을 기준으로 한다. 따라서 상위에 있는 어플리케이션이 재생하는 video는 하위에 있는 어플리케이션 그래픽스를 덮게 된다. 이로 인해서 다음과 같은 문제가 발생할 수 있다.

1. video/broadcast object를 사용하는 어플리케이션 A가 실시간 채널을 재생 중에 어플리케이션 B가 활성화되었다. 동일 priority이기 때문에 나중에 활성화된 어플리케이션 B가 어플리케이션 A의 video 위에 그래픽스를 그리고 있다.
2. 그런데 어플리케이션 A가 팝업을 그려야 하고, 그래서 key set을 변경해서 다시 활성화를 해야 한다. 어플리케이션 A가 다시 활성화하는 순간에 어플리케이션 A는 어플리케이션 B보다 나중에 활성화되었기 때문에 상위에 그려지게 된다.
3. 어플리케이션 A가 상위에 그려지면서 결국 video도 어플리케이션 B 위에 위치하게 된다.

사용자가 봤을 때는 위 3번에 의해서 어플리케이션 B가 죽었거나 hide했다고 생각할 수 있지만, 실제로는 어플리케이션 A의 영상 하위에 그래픽스를 그리고 있다.

위와 같은 문제는 실시간 채널을 재생하기 위해서 video/broadcast object를 사용하는 여러 어플리케이션 사이에서 흔히 나타날 수 있는 문제이다. 그런데 실제로 이런 기능의 어플리케이션은 video가 항상 다른 어플리케이션 그래픽스보다도 더 하위에 존재하기를 바라는 경우가 더욱 많다.

Solution

표준 기반에서 이 문제를 해결하기 위해서 CSS의 hidden 속성을 사용하는 방안을 마련했다. 즉 video의 visibility를 hidden 처리하면, video는 모든 graphics보다 하단에 표시되도록 되어 있다. 물론 이런 경우에도 video boundary는 여전히 영상의 위치 및 크기가 되기 때문에 제대로 설정을 해야 한다. 단지 graphics에 embedded된 것 같은 효과를 없애고, 모든 graphics의 하단에 위치하는 방식으로 보이게 된다.

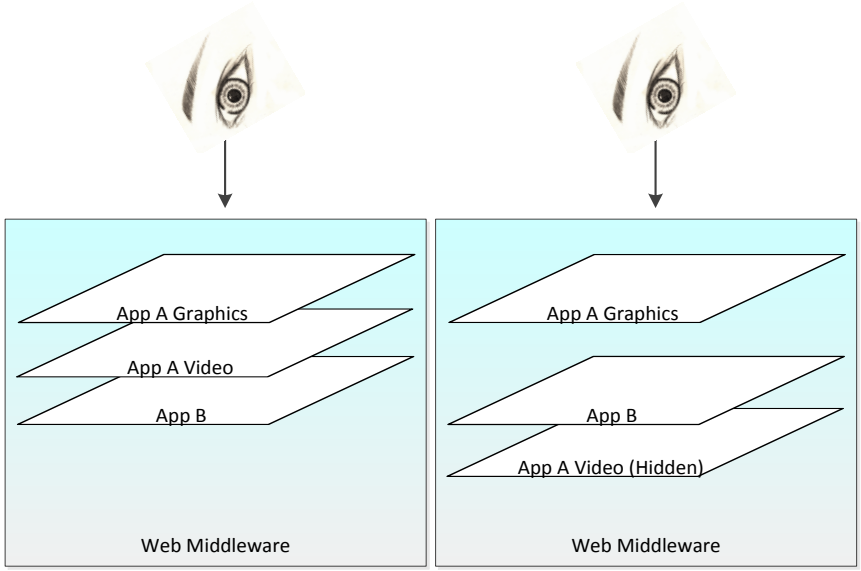


Figure 5 Effect of Hidden Video

위 그림에서 App A가 App B 위에 존재하며 App A가 전체화면으로 영상을 재생하는 중일 경우 video가 hidden 처리되어 있다면, App B의 그래픽스가 App A의 video 상위에 보이게 된다. (그림 우측) 그렇지 않은 경우는 기본적으로 App A의 그래픽스에 포함되어 있는 것처럼 보인다. (그림 좌측)

Hidden 처리하기 위해서는 다음과 같은 방법을 사용할 수 있다.

```
<javascript의 경우>
videobroadcast.style.visibility = hidden;
```

```
<css의 경우>
.videobroadcast {
	left: 0px;
	top: 0px;
	width: 1280px;
	height: 720px;
	z-index: 0;
	visibility: hidden;
}
```

6. Inter-Application Communication

6.1. How to Find Web Application

OIPF 표준 API에는 parent/child 관계가 아닌 application을 찾을 수 있는 방법이 없으므로 아래와 같이 추가된 interface를 제공한다.

```
oipfApplicationManager.findApplications(String key, String value)
```

제공하는 key는 다음과 같다.

Key	Description & Value	
useType	kt application type을 나타내는 문자열이며 각 application type에 따라 다음과 같은 문자열을 그 값으로 가진다.	
	Bound Application	"bound"
	Unbound Application	"unbound"
	User Application	"user"
	Independent Application	"independent"
	Host Application	"host"
dvb.appId	Unbound와 bound application를 가리킬 수 있는 ID로, "<orgId>.<appId>"의 형태를 띄고 있다. ID의 자세한 포맷은 "4.1 Application Identifier"를 참조하라. Unbound나 bound application에게 message 등을 전달하기 위해서 어플리케이션을 찾는데 사용할 수 있다. Host application에 대해서는 기본적으로 이 값이 존재하지 않으나, kt의 요청에 의해서 존재하는 host application이 있을 수도 있다.	

Table 3 Application Properties

User application의 proxy를 찾는 기능은 현재 지원되지 않는다. 필요한 부분을 확인하여 추후 정의 예정이다.

이 key 값을 이용하여 다음과 같이 어플리케이션 peer를 얻을 수 있다.

```
var appMgr = oipfObjectFactory.createApplicationManagerObject();

// Kills all bound applications
var boundApps = appMgr.findApplications("useType", "bound");
for (var i = 0; i < boundApps.length; i++) {
      boundApps[i].destroyApplication();
}
```

```
// Retrieves home portal application proxy
var apps = appMgr.findApplications("dvb.appId", "4e30.3001");
var homePortal = apps[0];
```

6.2. How to Send Message to Web Application

HTML5에 정의된 window 객"의 "void postMessage(any message, String targetOrig"n)" API를 통해서 event를 전달한다. message parameter에 string 혹은 JSON object를 사용할 수 있다.

예제)

```
// Message를 보내는 쪽
var peer = oipfApplicationManager.findApplications(key, value); // peer 검색
peer[0].window.postMessage(message, targetOrigin); // 찾아낸 peer로 message를 보냄.

// Message를 받는 쪽
function popupMessage(evt) {
      var message;
      message = 'data=' + event.data + ',origin=' + event.origin;
      alert(message);
}
window.addEventListen'r('mess'ge', popupMessage, false);
```

'message' 핸들러로 넘겨진 event object의 source property를 통해 event를 보낸 application에 리턴 메시지를 보낼 수 있으나, 이 경우 메시지를 전달 받은 application의 'message' 핸들러에서도 동일하게 리턴 메시지를 보내는 경우 메시지가 무한 전송되는 문제가 발생할 수 있으므로 주의해야 한다.

6.3. How to Start Bound Application with Parameters

현재 채널에 존재하는 bound application을 구동시키기 위한 URL은 "dvb://current.ait/<orgId>.<appId>"의 형태를 띈다. 이 때 반드시 orgId와 appId의 format은 "4.1 Application Identifier"을 참조한다.

```
// orgId가 0x100000a, appId가 0x3010인 어플리케이션을 구동하기
var appMgr = oipfObjectFactory.createApplicationManagerObject();
var self = appMgr.getOwnerApplication( window.document );
var child = self.createApplication( "dvb://current.ait/100000a. 3010", true );
```

또한 어플리케이션을 구동시킬 때 URL의 query string 부분에 넣을 수 있다. 예를 들어 위

예제에서 "user=hongkildong", "type=normal"이라는 2개의 input parameter를 전달하려고 하면 다음과 같이 하면 된다.

```
self.createApplication( "dvb://current.ait/100000a.3010?user=hongkildong&type=normal", true );
```

이렇게 query string을 전달받은 어플리케이션은 window.location.search property를 이용해서 이 값을 얻을 수 있다.

```
var queryString = window.location.search;
console.log( queryString );      // "?user=hongkildong&type=normal"
```

6.4. How to Start Unbound Application

기본적으로 모든 unbound application은 observer라는 별도의 unbound application이 관리하도록 되어 있기 때문에 특정 application이 직접 unbound application을 구동하면 안 된다. unbound application 구동이 필요하면 observer가 제공하는 API를 이용해서 활성화해야 한다.

7. Key Code Table

어플리케이션은 key code 처리시에 항상 VK_ 형식의 constants를 사용하여야 한다. 직접 integer형식의 code값을 사용해선 안 된다.

7.1. Remote Controller Key

Key Name	Key Code	비고
전원	VK_POWER	
영화	VK_RECALL_FAVORITE_0	
TV다시보기	VK_RECALL_FAVORITE_1	
편성표	VK_RESERVE_1	
메뉴	VK_HOME	
선호채널	VK_F3	
음량+	VK_VOLUME_UP	
음량-	VK_VOLUME_DOWN	
채널+	VK_CHANNEL_UP	
채널-	VK_CHANNEL_DOWN	
UP	VK_UP	
DOWN	VK_DOWN	
LEFT	VK_LEFT	
RIGHT	VK_RIGHT	
확인	VK_ENTER	
이전	VK_BACK	
음소거	VK_MUTE	
나가기	VK_ESCAPE	
REWIND	VK_REWIND	
PLAY	VK_PLAY	
STOP	VK_STOP	
FAST_FWD	VK_FAST_FWD	
0 ~ 9	VK_0 to VK_9	
*(획추가)	VK_F11	
#(쌍자음)	VK_F12	
검색창	VK_RECALL_FAVORITE_2	
지우기	VK_DELETE	

한영전환	VK_F10	
적색	VK_RED	
녹색	VK_GREEN	
황색	VK_YELLOW	
청색	VK_BLUE	
쇼핑	VK_CLEAR_FAVORITE_0	
스카이초이스	VK_F7	
스카이플러스	VK_F8	
위젯	VK_INFO	
캡쳐	VK_RESERVE_3	물리적으로 존재하는 키가 아니며 *키를 1.5초 이상 long press 해야 이벤트가 발생한다.

Table 4 Remote Controller Key Code Table

7.2. Keyboard Key

Key Name	Key Code	비고
ESC	VK_ESCAPE	
F1	VK_POWER	
F2	VK_HOME	
F3	VK_CHANNEL_UP	
F4	VK_CHANNEL_DOWN	
F5	VK_VOLUME_UP	
F6	VK_VOLUME_DOWN	
F7	VK_PLAY	
F8	VK_RECALL_FAVORITE_2	
F9	VK_RED	
F10	VK_GREEN	
F11	VK_YELLOW	
F12	VK_BLUE	
TAB	VK_BACK	
CTRL	VK_REWIND	
ALT	VK_FAST_FWD	
한영전환	VK_MODECHANGE	리모콘의 한영전환과 다르다. 리모콘은 VK_F10 이다.

Table 5 Keyboard Key Code Table

8. OSK

OSK는 기본적으로 input text field에 대해서 platform이 제공해 준다.

8.1. PIN Input Field

일반적으로 PIN을 입력하기 위한 input field를 많이 사용하게 된다. 이를 위해서 확장 attribute인 x-altibrowser-imemode를 제공하고 있다. 이 값이 'PIN'이면 PIN 입력을 위한 것으로 인지하여 OSK를 띄우지 않고 숫자 key로 숫자 입력만 가능하게 된다.

```
<input type="password" x-altibrowser-imemode='PIN'/>
```

8.2. How to Set Initial OSK Mode

OSK에서 초기 입력 모드를 설정할 수 있다. x-altibrowser-imemode라는 attribute를 사용하며, 다음과 같은 값을 설정할 수 있다.

- #KOR: 한글
- #eng: 영어 소문자
- #ENG: 영어 대문자
- #NUM: 숫자
- #SP: 기호
- PIN: PIN 입력 모드로써, 숫자만 입력 가능하며, OSK가 보이지 않는다.

기본적으로는 한글이 초기 언어로 설정된다.

다음은 숫자를 초기 입력 모드로 설정하는 예제이다.

```
<input x-altibrowser-imemode='#NUM"/>
```

8.3. How to Disable Platform OSK

어플리케이션에서 원하는 layout으로 OSK를 배치하거나 다른 key 배열을 사용하려고 하면 별도로 JavaScript로 개발하여야 한다. 이 때 web middleware에서 자체 OSK를 띄우지 않게 하기 위해서 <input>, <textarea> element에 다음과 attribute를 사용하면 된다.

- disabled: 값에 상관없이 이 값이 존재하면 IME가 자동 시작되지 않는다. Style을 따로 지정하지 않으면 disable의 기본 style이 적용되어서 background가 회색 처리된다. 또한 XHTML의 경우에는 disabled="disabled"와 같이 적어주면 된다.

- readonly: 값에 상관없이 이 값이 존재하면 IME가 자동 시작되지 않는다. Style에 따로 변경이 발생하지 않는다. 또한 XHTML의 경우에는 readonly="readonly"와 같이 적어주면 된다.

```
First name: <input type="text" name="fn"me"><br>
Last name: <input type="text" name="lname" disabled><br>
```

9. Object Carousel

기존에 object carousel을 사용하여 data를 전달받는 방법을 사용했던 경우에 웹 미들웨어에서도 그대로 사용할 수 있는 방법을 제공한다.

9.1. Carousel Objects Access with XMLHttpRequest

Object carousel의 file에 접근하는 기본적인 방식은 XMLHttpRequest object를 이용하는 것이며, 다음과 같이 동작한다.

- open()
  - method: "GET"이어야 함
  - url: URL scheme이 dvb인 URL임. ex) dvb://1.ff31.b4/img/vodList.xml
  - async: 반드시 "true"여야 함
  - user, password: 무시
  - status
    - 200: DSM-CC object를 정상적으로 찾았을 경우
    - 404: Carousel이 존재하지 않거나 carousel 안에 해당 object (file)이 존재하지 않는 경우
  - statusText: 빈 문자열
    - Header는 무시한다.
- setRequestHeader() 호출은 무시한다.
- getResponseHeader(), getAllResponseHeader()는 빈 문자열을 리턴한다.

responseText와 responseXML은 아래 표와 같다.

DSM-CC object	URL example	responseText	responseXML
File	/weather/data.xml	기본 XMLHttpRequest 동작과 동일	
Directory	/weather	쉼표로 구분된 하위 객체 목록	null

Table 6 XMLHttpRequest for DSM-CC Object Access

9.2. URL Format

기본적인 URL 형식은 다음과 같다.

```
[dvb://<onid>.<tsid>.<sid>[.<ctag>]]/<object_path>
```

다음과 같이 3가지 형태로 접근할 수 있다.

- Object carousel로 전송되는 어플리케이션의 내부 object에 접근할 때
  - 상대 경로만 지정하면 된다.
  - 예) /weather/data.xml

- Data만 별도의 object carousel로 전송되며, 해당 채널에서 object carousel이 하나뿐인 경우
  - "dvb://<onid>.<tsid>.<sid>/<object_path>"의 형태로 접근할 수 있다. 이 때 onid, tsid, sid는 16진수여야 한다.
  - 예) dvb://1.1fff.3e/weather/data.xml

- Data만 별도의 object carousel로 전송되며, 해당 채널에서 object carousel이 두 개 이상일 경우
  - 반드시 component tag를 추가해야 한다. ACAP에서는 carousel ID를 사용했지만, 표준에 의해서 URL에서는 component tag를 사용하는 방법만 지원한다.
  - 예) dvb://1.1fff.3e.b8/weather.data.xml (b8은 component tag임)

9.3. Change Notification

Video/broadcast embedded object에 object update listener를 등록할 수 있다. 이 방법을 이용해서 해당 object의 변경을 인지할 수 있다.

```
void addObjectChangeEventListener( String targetUrl, ObjectChangeEventListener lnr )
void removeObjectChangeEventListener( String targetUrl, ObjectChangeEventListener lnr )

interface ObjectChangeEvent {
      readonly attribute String sourceUrl;
      readonly attribute String status;
      readonly attribute Integer versionNumber;
}
```

만약 object update가 발생하면, status가 versionNumber와 함께 "update"인 ObjectChangeEvent가 발생하고, 아래와 같은 오류가 발생하면 status가 "error"이 ObjectChangeEvent가 발생한다.

- targetURL에 해당하는 object를 찾을 수 없거나
- Object carousel을 mount (attach) 할 수 없거나
- Object carousel이 여러 가지 이유로 unmounts (detach) 되었을 때

9.4. Remark

Object carousel을 attach하거나 detach하는 별도의 API 없이 file 접근 방식의 API나 object 변경 event handler 등록 기능만 제공하기 때문에, 최초로 해당 object carousel에 포함되는 file을 접근하려고 하거나 그 변경 event handler를 등록하려고 하면, 그 때 carousel attaching(mounting)을 시작하게 된다. 그래서 object carousel에 포함된 file을 최초로 접근할 때나 event handler를 등록할 때 다소 시간이 걸릴 수 있다. 이후 동일한 object carousel의 다른 파일에 접근할 때는 캐시된 파일에 바로 접근하기 때문에 빠른 시간 안에 결과를 받을 수 있다.

10. Application Packaging (Widget)

10.1. What is Widget?

Widget은 HTML, CSS(Cascading Style Sheet), JavaScript 파일 및 기타 자원(예: 이미지)으로 구성된 패키징된 웹 어플리케이션이다.

일반적인 웹 어플리케이션 대신 widget을 사용할 때의 장점 중 하나는 장치에 설치되는 일반 어플리케이션과 마찬가지로 한 번 다운로드된 후에는 여러 번 사용할 수 있다는 것이다. 이를 통해 사용자는 widget 파일 자체가 아니라 widget에서 사용하는 데이터만 전송하기 때문에 대역폭을 절약할 수 있다.

10.2. Widget Configuration

Widget을 build하려면 먼저 myWidget과 같은 widget 이름이 포함된 비어 있는 디렉토리를 작성한다. (파일 이름에 점을 사용할 수 있는 운영 체제의 경우에는 myWidget.wgt를 사용한다.) 새 widget 파일을 이 디렉토리에 저장한다. 새 HTML, CSS 및 JavaScript 파일을 작성하고 테스트하려면 widget을 패키지화하기 전에 브라우저에서 HTML 파일이 어떻게 작동하는지 확인할 수 있도록 웹 서버의 문서 루트에서 이 디렉토리를 작성한다.

패키지의 기본 디렉토리인 방금 작성한 비어 있는 디렉토리 바로 아래에 config.xml(대소문자 구분)이라는 구성 파일이 각 widget 패키지에 있어야 한다. config.xml이라는 구성 파일에는 이름, 작성자, 설명, license 등과 같은 widget에 대한 필수 정보가 들어 있다.

아래는 config.xml의 예제이며, 이 파일은 UTF-8으로 인코딩되어야 한다.

```
<?xml versi”n=””.0" encodi”g="UT”-8"?>
<widget xml”s="http://www.w3.org/ns/widg”ts"
”d="http://www.example.com/widgets/HelloWid”et"
versi”n=””.1" wid”h=””00" heig”t=””00
      <name>HelloWidget</nam
      <description>A very basic widget that say“, "He”lo"</descriptio
      <content s”c="index.h”ml"
      <icon s”c="images/icon.”ng"
      <access netwo”k="fa”se"
      <author>Nathan A. Good</autho
      <license>This is my license</license>
</widget>
```

config.xml에서 사용되는 속성(attribute)과 요소(element)에 대한 설명은 다음과 같다.

Attribute Name	Description
id	widget을 구분할 수 있는 unique URI 또는 application ID Application ID 형식을 사용하는 경우에는 "dvb.appId:" prefix를 붙여야 한다. 예) dvb.appId:4e30.3001
version	widget version
width, height	widget의 넓이와 높이 (unit: pixel)

Table 7 Attributes for Widget Parent Element

Element Name	Description
name	Widget의 축약 이름은 short 속성에 제공되고, 긴 이름은 XML element의 텍스트로 제공됨
description	Widget의 설명
author	작성자에 대한 정보
license	widget의 license. 단, license를 처리하는 기능은 추후 제공 예정이다.
icon	icon file의 상대 경로

Table 8 Widget Configuration Elements

10.3. The start File

config.xml 구성 파일을 제외하고 올바른 widget package에 필요한 유일한 다른 파일은 하나 이상의 시작 파일이다. 시작 파일이 지정되지 않은 경우 기본 파일은 index로 시작하는 이름을 가진 파일이다. 일반적으로 확장자가 html, htm 또는 xhtml이다.

아래는 시작 파일(index.html)의 예제이며, 일반적인 HTML 파일과 동일하다.

```
<!DOCTYPE HTML>
<html xml”s="http://www.w3.org/1999/xh”ml">
<hea
      <meta http-equ”v="pra”ma" conte”t="no-ca”he"
      <meta http-equ”v="Content-T”pe" conte”t="text/html; charset=us-as”ii"
      <title>HelloWorld</titl
      <link r”l="stylesh”et" hr”f="style/common.”ss" ty”e="text/”ss" />
</head>
<body>
<h1>Hello World...</h1>
</body>
</html>
```

10.4. Widget Packaging

Widget은 확장자가 wgt인 일반 zip 파일로 packaging하면 된다. Widget package는 압축 파일 유틸리티를 사용하여 만든다.

```
$ cd myWidget
$ zip myWidget.wgt *
```

혹은 일반 압축 프로그램을 사용하여 zip 파일을 생성한 후, 확장자를 wgt로 변경하면 된다.

Widget directory는 보통 다음과 같이 구성된다.

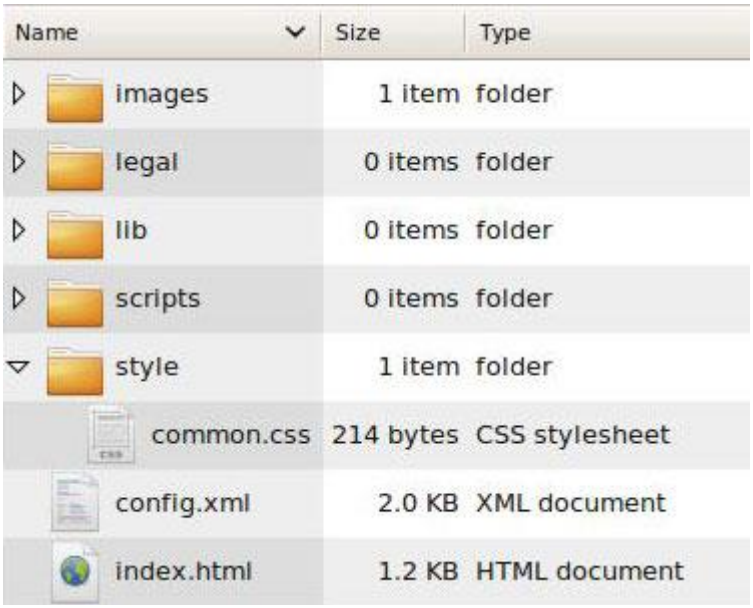


Figure 6 Widget Package Folder Structure

시작 파일에 있는 자원을 참조하는 경우 파일이 웹 서버에 전개된 것처럼 상대 경로를 사용한다. 예를 들어, style/common.css 파일과 style 폴더의 common.css 파일이 압축 파일 archive에 widget package로 포함되어 있는 경우라도 style/common.css를 사용하여 해당 common.css 파일을 참조한다.
widget package 내부의 파일은 파일명에 한글이나 특수문자를 사용하지 않도록 주의 한다.

10.5. Widget Size

Widget package는 압축 해제 시 10MB 미만으로 작성되어야 한다. 불가피하게 제한 크기를 초과하는 경우 kt 담당자와 협의해야 한다.

10.6. STB 내부에 위치한 어플리케이션의 CORS

Widget이 외부 웹 서버와 연동하는 경우, 웹 서버의 HTTP response header에 다음 정보를 추가해야 한다.

```
Access-Control-Allow-Origin: http://localhost:8090
```

11. Unbound Application Guideline

Unbound Application이기 때문에 추가적으로 필요한 개발 지침을 여기서 정리한다.

11.1. Memory Optimization

통상적으로 unbound application들은 daemon 형태로 항상 구동 중인 방식으로 설계될 가능성이 크다. 이러한 application일수록 메모리 사용량을 최적화해야 전체적인 메모리 활용도를 높일 수 있다.

자신이 화면에 무엇인가를 그리지 않을 때 graphic buffer 사용량을 줄일 수 있도록 반드시 고려해야 한다. 자세한 방법은 "3.3 그래픽 메모리 사용 최적화"을 참조한다.

또한 일반 어플리케이션과는 달리, daemon 형태로 항상 떠 있는 어플리케이션이기 때문에 항상 메모리 사용량이 최적이 되도록 사용량을 최소화하여야 한다. Daemon 형으로 계속 떠 있기 때문에 전체적인 메모리 사용량은 kt 담당자와 협의를 해야 하며, 단일 unbound application의 메모리 사용량 최대치는 다음과 같다. 메모리에 대한 설명은 "Appendix C Understanding Memory Usage (Informative)"를 참조한다.

- Kernel Memory: 20MB
- Driver Memory: 25MB

11.2. Window & Event Handling

Unbound application은 다른 일반적인 bound application과 충분히 연동될 수 있다. 그래서 OIPF DAE 규격에 맞게 작성된 bound application이 최대한 자유롭게 개발을 해도 unbound application이 자연스럽게 같이 동작되도록 작성하는 것이 바람직하다.

그래서 window layer와 event handling priority를 좀 더 신중하게 결정할 필요가 있다. 그래서 bound application과 같이 동작한다고 해도 bound application과 충돌이 생기지 않고 사용자가 보기에 자연스럽게 동작할 수 있도록 priority를 잘 정의하고 이에 맞게 잘 구현되어야 한다. Priority는 전체적인 서비스 시나리오에 맞게 잘 정의되어야 한다. 이는 kt와 협의하여야 한다.

"5.2 Prioritized Graphics & Events"을 참조한다.

또한 unbound application은 항상 떠 있기 때문에 불필요한 key event 처리를 하게 될 수 있다. 예를 들어 홈포털이 invisible 상태에서 거의 대부분의 key event를 처리할 필요가 없음에도

불구하고 visible 상태와 동일하게 KeySet을 유지하고 있다면 불필요하게 홈포털에게 key event가 전달되어 당시 visible한 어플리케이션의 event 처리 속도를 저하시킨다. 그래서 반드시 불필요한 event 처리가 일어나지 않도록 KeySet 설정을 잘 하도록 한다. "3.4.5 Event Handling Optimization" 참조

11.3. API Guideline for Web Messaging

Unbound application은 web messaging 방법을 이용하여 API를 제공하게 될 텐데, 이 때 여러 어플리케이션이 API를 제공하기 때문에 다음과 같이 JSON message format으로 정해야 한다.

<format>
{ 'method':[method_name], [param_1_name]:[param_1_value], ...
[param_n_name]:[param_n_value]}

<example>
Request: { 'method':'hp_getCurrentVODInfo', 'req_cd':'09', 'from':'4e30.3002' }
Response: { 'method':'hp_getCurrentVODInfo', 'const_id':'M0179BQLSGL070000100', 'title':'후르츠 칵테일', 'duration':'5160', 'viewpoint', '3260', 'from': '4e30.3001' }

- function name
  - prefix를 사용하여 다른 어플리케이션 사이에 동일한 이름의 API를 제공하는 것을 피하다. 예를 들어 home portal은 "hp_", observer는 "obs_" 등을 사용한다. 이와 같이 적절한 prefix를 kt 담당자와 협의하여 결정한다.
  - request에 해당하는 response의 경우에도 동일한 API function name을 사용한다. 위 예제에서도 hp_getCurrentVODInfo라는 request에 대해서 response에서도 동일한 function name을 유지했다.

- 어플리케이션 식별자
  - 위 예제처럼 어플리케이션 식별자가 필요한 경우에는 어플리케이션 ID를 "<org_id>.<app_id>"의 형태로 만들어진 문자열을 식별자로 한다. 예를 들어 org_id가 0x00004e30이고 app_id가 0x3001일 경우에 식별자는 "4e30.3001"과 같은 형태로 한다.

12. VoD

12.1. 소개

VOD 서비스는 CEA-2014A, OIPF 표준 API를 사용한다.

12.2. VOD through HTML tag

12.2.1. VOD Media control through HTML video object

VOD player object를 HTML tag로 생성하고 JavaScript로 이를 control할 수 있다. 예제는 다음과 같다

```
<script>
function playVideo() {
    var vodURL;     // play URL은 MoC로부터 갖고 오며 어플리케이션과 서버 간의 자체
                    // algorithm에 의해 처리된다.
var videoPlayer = document.getElementBy“d("vi”eo");
    videoPlayer.data = vodURL;

    videoPlayer.play(1); // 1배속으로 재생

    videoPlayer.play(2); // 2배속으로 재생

    videoPlayer.seek(60*1000); // 1분 position으로 이동.

    videoPlayer.stop();   // 정지
}
</script>

<body onlo”d="javascript:playVideo”);">
    <object ty”e="video/m”eg" wid”h=""00" heig”t=""00" ”d="vi”eo"/>
</body>
```

기타 세부적인 기"은 "12.3 VOD through JavaScript" 를 참조한다.

12.3. VOD through JavaScript

12.3.1. VOD Media Control through JavaScript

VOD play 및 control은 oipfObjectFactory.createVideoMpegObject()를 통해 생성한 object를 통해

지원한다. 예제는 다음과 같다.

```
var vodURL; // play URL은 MoC로부터 갖고 오며 어플리케이션과 서버 간의 자체
                    // algorithm에 의해 처리된다.
var videoPlayer = window.oipfObjectFactory.createVideoMpegObject();

document.getElementById('playerDiv').appendChild(videoPlayer);

videoPlayer.data = vodURL;

videoPlayer.play(1); // 1배속으로 재생

videoPlayer.play(2); // 2배속으로 재생

videoPlayer.seek(60*1000); // 1분 position으로 이동.

videoPlayer.stop();   // 정지
```

12.3.2. 중간 광고 play 중 VOD play(), seek() 처리

VOD 광고는 VOD 재생 전/후 또는 중간에 삽입되는 광고 동영상 콘텐트를 의미하는데, 광고 콘텐트가 재생되는 경우에는 kt 광고 서비스 정책에 따라 Fast-Forward, Rewind, Seek등의 트릭모드 플레이를 할 수 없다. (광고 중 PAUSE는 지원)

이 경우 videoPlayer.seek(), videoPlayer.play()는 false를 리턴한다.

현재 광고를 play하고 있는지 알 수 있는 API는 별도의 MediaTimeLineControl을 참고하면 된다.

12.3.3. Event 전달

kt olleh tv web extension framework 규격 문서를 따른다.

12.3.4. 서버 다중화에 대한 지침

Play 중에 session error가 발생한 경우는 서버와의 연결에 문제가 발생한 것으로 간주하고 시나리오에 따라서 다른 서버 URL로 다시 play 요청하는 서버 다중화 처리를 어플리케이션이 직접 해야 한다.

12.4. VOD 광고 및 PTS Media Time 정보

어플리케이션은 VOD가 전달하는 광고 정보 등 현재 VOD session과 관련된 정보를 MediaTimeLineControl를 통해서 얻어올 수 있다. MediaTimeLineControl이 반환하는 값은 VOD

서버에 전적으로 의존하므로 VOD측에 문의해야 한다.

VOD API에서 제공하는 playPosition 정보는 재생 화면과 오차가 발생할 수 있기 때문에 현재 재생 화면에 일치하는 시간 정보를 얻을 수 있도록 PTS(Presentation Time Stamp) 기반의 좀 더 정확한 값을 전달해 주고 성능도 빠른 API가 추가 됐다. 추가된 API는 MediaTimeLineControl을 참고하면 된다. 참고로 PTS는 ISO13818-1에 정의된 용어로 encoding될 당시에 video packet 에 기록된 값이다.

12.4.1. VOD 광고 정보 및 PTS Media Time API

아래는 VOD 광고 정보와 PTS media time을 얻어오는 간략한 예제이다.

```
var mtlc = oipfObjectFactory.createMediaTimeLineControl();

var _onContentChanged = function(event) {
      // MediaContentChanged callback.
      ...
};

mtlc.onMediaContentChanged = _onContentChanged;

var ptsTime = mtlc.getMediaTime();
var isTrickPlayPossible = mtlc.trickPlayPossible();
var adsInfo = mtlc.getAdsInfo();
```

자세한 사용법은 "Appendix G. Media Timeline Control APIs" 를 참고한다.

13. API Extensions

표준을 간단하게 확장하여 필요한 기능들을 제공하고 있는 기능들은 다음과 같다.

13.1. Configuration Keys

Configuration.getText()를 이용해서 얻을 수 있으며 얻을 수 있는 key 값은 다음과 같다.

Key Name	Description	Note	Read only
SAID	SAID		O
bdi_version	채널정보 버전(BDI)		O
pdi_version	채널정보 버전(PDI)		O
adi_version	채널정보 버전(ADI)		O
spdi_version	채널정보 버전(SPDI)		O
di.updatePeriod	채널정보 업데이트 날짜		O
version.xait	XAIT 버전		O
info.attsAddress	ATTS 주소		O
Info.attsLocator	ATTS Locator 정보		O
id.bouquet	Bouquet ID		O
info.productCode	상품코드		O
bmailNotify.ots	긴급메시지 알림 설정		
restrictionTime	시청시간제한 설정		
mbs.ots	DCS 설정		
hdmiCEC	HDMI 전원 동기화		
hdmiCECFunctionMasc	HDMI 전원 동기화	"STB_ON_BY_TV" "STB_OFF_BY_TV" "TV_ON_BY_STB" "TV_OFF_BY_STB"	
hdmi.isConnected	HDMI 연결 확인 여부		O
id.smartcard	Skylife 스마트카드 ID		O
id.chip	Chip ID		O
CC_ONOFF	자막 설정		
CC_FONT_SIZE	자막 글자 크기 설정		
CC_LANGUAGE	자막 언어 설정		
Visual Impaired	화면 해설 방송 설정		

Audio Language	화면 해설 방송 언어 설정		
User Language	기본 언어 설정		
info.softwareUpdateDate	업데이트 날짜		O
version.firmware	펌웨어 버전		O
version.mainSoftware	Main 소프트웨어 버전		O
version.middleware	미들웨어 버전		O
version.navigator	네비게이터 버전		O
version.caAdaptor	CA Adaptor 버전		O
version.verifier	KT-CAS Library 버전		O
version.software	Glue 버전		O
version.loader	로더 버전		O
network.macAddress	MAC 주소		O
version.driver	Driver 버전		O
networkStatus	네트워크 연결 상태		O
channelInfoOutput	채널정보 출력 설정		O
downloadStatus	S/W Download Status 정보		O
serialNumber	시리얼 번호		O
tuneStatus	Tune Status		O
id.manufacturer	Manuufacturer ID		O
version.hardware	하드웨어 버전		O
version.caAPI	CAK 버전		O
modelNumber	모델 넘버		O
model_number			O
version.card	Skylife 스마트카드 버전		O
caECMMetadata	Skylife ECM meta data		O
caZipCode	Skylife zip code		O
caSegment	Skylife segments		O
caNUID	Skylife STB chipset NUID		O
caSerialNumber	Skylife STB CA 시리얼 번호		O
caProjectInfo	Skylife project information		O
version.chipset	Skylife chipset version		O
info.chipsetRevision	Skylife chipset revision		O
stb.standbyLevel	저전력 설정	"0" : 능동 대기모드 "1" : 수동 대기모드	
version.observer	서비스 관리자 버전		
version.appstore	앱스토어 버전		

version.oam	OAM 버전		
version.pbroadcast	개인방송 버전		
version.mashup	매시업 매니저 버전		
version.smartpush	Smart push 버전		
kidsCareMode	자녀안심 설정		
kidsCareModeAuto	자동 자녀안심 설정		
live_vod_support	성질 급한 VOD 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
default_channel	기본 채널 설정	DVB locator	
ble_rcu_support	BLE RCU 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
mosaic_support	mosaic 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
mosaic_count	지원하는 mosaic 개수		O
tvpay_support	TV pay 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
longpress_sharp	Sharp long-press 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
support.uhd	UHD 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
mbs_support	DCS 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
skylife_support	Skylife 기능 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
hdr_support	HDR 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
hdr_support_by_edid	HDR by EDID 지원 여부	"true" : 지원 "false" : 미지원	O

		"undefined" : 미지원	
lcw_booting_support	시작채널 LCW 지원 여부	"true" : 지원 "false" : 미지원 "undefined" : 미지원	O
stb.type	STB Type	"SMART" "UHD1" "UHD2" "UHD3"	O
castis_sdk_version	Castis SDK version		O
beacon_interval	비콘 주기		
beacon_tx_power_level	비콘 신호 세기 레벨		
long_distance_voice	원거리 음성 인식 지원 여부	"ON" : 지원 "OFF" : 미지원	
use_kids_mode	키즈 모드 사용 여부	"ON" : 사용 "OFF" : 미사용	
auto_standby	자동 대기 모드 사용 여부	"ON" : 사용 "OFF" : 미사용	
Adult Menu Display	성인 메뉴 노출 여부	"true" : 노출 "false" : 미노출	

이에 따라 said를 get/set 방법은 다음과 같다.

```
var config = oipfObjectFactory.createConfigurationObject();
var said = config.configuration.getText("SAID");
config.configuration.setText("stb.standbyLevel", "0");
```

추가적으로 predefined 되지 않은 key에 대해서도 get/set이 가능한데, home portal과 타 bound app간 parameter를 전달할 목적으로 기능이 구현되었다.

1. home portal이 setText()로 parameter를 저장하고
2. bound app은 getText()로 읽은 후, 다시 setText()로 null로 만든다. (반드시 지켜야할 mandatory는 아지나, 다른 언바운드와 연동 고려시 필요하다)
3. 서로 정보를 주고 받은 규약은 각 앱간 협의를 통해 정의를 하는 것으로 한다.

13.2. Capabilities

여러 종류의 STB가 지원하는 기능은 조금씩 다를 수 있다. 이에 따라 지원하는 기능의 차이가 생길 경우 Capabilities.hasCapability(String profileName)의 profileName을 통해서 지원 여부를 확인할 수 있다.

현재는 1종의 STB이기 때문에 큰 의미가 없으며, 추후 필요에 따라 추가 정리될 예정이다.

13.3. PIN Check

PIN을 확인하기 위해서는 kt olleh tv web extension framework에서 제공하는 ParentalControlManager object를 사용하면 되며, 간단한 사용법은 다음과 같다.

```
var p = oipfObjectFactory.createParentalControlManagerObject();
if (p.verifyParentalControlPIN('0000')) {
      // success
}
```

13.4. Mouse Control

자세한 API 설명은 "Appendix J Mouse Control APIs"을 참조한다.

13.4.1. How to Enable Mouse Function

USB mouse가 연결되었다고 해도 mouse pointer가 보이게 되면 사용자가 mouse 기능이 동작될 것을 기대하게 된다. 그런데, 현재 많은 어플리케이션이 아직 마우스 이벤트 처리를 지원하지 않을 것이다. 그래서 기본적으로 mouse가 연결되었다고 해도 mouse pointer가 보이지 않도록 개발되어 있다.

그래서 MouseControlObject.requestMouseControl()을 통해서 마우스 기능을 활성화시키고, 비활성화시키기 위해서는 MouseControlObject.releaseMouseControl()을 호출한다. 비록 requestMouseControl()을 호출했다고 해도 연결되어 있는 마우스가 없다면 여전히 마우스 포인터는 보이지 않게 된다.

13.4.2. IR Virtual Mouse

기존 리모컨은 마우스 포인팅 기능이 없어서, "상", "하", "좌", "우", "확인"을 이용해서 가상 마우스 기능을 제공하고 있다. 이 가상 마우스 기능을 사용하고 싶다면 MouseControlObject.setIRMouseEmulation(true)을 이용하여 활성화시키면 된다. 이것은 개념적으로 마우스 디바이스 하나를 연결하는 것과 동일하다. 그래서 추후 setIRMouseEmulation(false)를 호출하여 가상 마우스 디바이스를 연결 해제하도록 호출해 주어야 한다.

이렇게 가상 마우스 디바이스를 연결하게 되면 리모컨의 "상", "하", "좌", "우", "확인" 키는 mouse event 생성에 사용될 뿐 key event를 발생하지 못하게 된다. 단 RF 리모컨은 이런 상태라고 해도 위 화살표 키와 확인 키에 대해서 key event로 항상 발생시키게 된다.

이 기능을 활성화한 어플리케이션이 반드시 비활성화시켜야 한다. 만약 그렇지 못하면 가상 마우스 기능이 계속 활성화된 채로 남아 있기 때문에 mouse pointer가 보이고, "상", "하", "좌", "우", "확인" 키를 기반으로 mouse event가 생성되게 된다. 단 비활성화시키지 않았다고 해도 호출했던 어플리케이션이 죽게 되면 STB에서 알아서 연결 해체 처리해서 비활성화시키게 된다.

13.4.3. How to Enable Virtual Mouse only for IR Remote Controller

만약 RF 리모컨을 사용하는 사용자라면 굳이 가상 마우스 기능을 켜지 않는 것이 좋다. 왜냐면 RF 리모컨은 터치패드가 포함되어 있기 때문에 굳이 화살표 키로 마우스 포인터를 움직이게 하는 기능이 필요없기 때문이다. 그래서 화살표 키 이벤트의 이벤트 소스를 확인하여 IR 리모컨이라면 가상 마우스 기능을 활성화하는 것을 권장한다.

```
function keyHandler(e) {
    // e.source 상수 값은 kt olleh tv web extension framework 참조
    if (e.source == 1) { // IR 리모컨
        switch (e.keyCode) {
            case VK_UP: case VK_DOWN: case VK_LEFT: case VK_RIGHT:
                var m = oipfObjectFactory.createMouseControlObject();
                m.setIRMouseEmulation(true);  // 마우스 에뮬레이션 활성화
                break;

        }
    }

...
```

13.5. Channel Change

Background

video/broadcast object의 setChannel(Channel channel)에 의해서 채널 변경을 할 경우에는 다음과 같이 채널 전환의 전체 과정이 모두 수행된다.

- Observer가 채널 변경 때문에 수행할 어플리케이션 종료를 비롯한 다양한 기능을 수행한다.
- Navigator가 이동할 채널 OSD를 보여준다.
- 실제 채널 변경을 처리한다.

채널 변경으로 인해서 bound application이 AIT signaling의 결과로 죽거나 뜨는 것은 어쩔 수

없지만, 어플리케이션 내부적으로 영상 전환용으로 채널 전환을 하고 싶은 경우가 있다. 이런 경우에는 채널 OSD를 표시하거나 홈 포털이 채널 변경 때문에 사라지는 기능들이 처리되지 않기를 원할 것이다.

예를 들어

- 프로야구 어플리케이션은 편파해설방송 채널로 이동하지만, 이 때 채널 OSD를 표시하고 싶지 않다.
- 홈포털이 스티칭 채널을 재생하고자 할 때 채널 전환으로 인해서 홈 메뉴가 사라지거나 채널 OSD를 표시하고 싶지 않다.

Solution

이 문제를 해결하기 위해서 setChannel()에 확장 parameter를 추가하였다.

setChannel(Channel channel, Boolean disableChannelEvent, Boolean disableOsdDisplay)

- channel: 이동하고자 하는 채널
- disableChannelEvent: true이면 채널 변경을 observer에게 알리지 않는다. observer에게 알리지 않으면, 채널 변경으로 인해서 어플리케이션을 종료하는 작업을 처리하지 않게 된다. 기본값은 false이다.
- disableOsdDisplay: true이면 채널 OSD를 보이지 않는다. 기본값은 false이다.

확장 parameter를 이용해서 프로야구 어플리케이션이 편파해설방송으로 진입할 때는 다음과 같이 호출하면 된다.

```
// 채널 변경으로 인해서 홈포털이 사라지기를 원하지 않으므로 disableChannelEvent를 true로
// 준다. 단, 프로야구 어플이 떠 있을 때는 홈포털이 뜨지 않기 때문에 disableChannelEvent는
// false로 줘도 사실 문제없다.
// 편파방송채널의 채널 번호를 표시하지 않기를 원하기 때문에 disableOsdDisplay를 true로 전달
// 한다.
vbo.setChannel(channel, true, true);
```

13.6. Change to Promo Channel

Background

독립형 어플리케이션이 자체 오류로 죽게 되거나 오류 상황을 극복하기 위해서 promo channel로 이동하는 기능이 필요할 수 있다. 독립형 어플리케이션의 경우 promo channel로 이동하면 자연스럽게 현재 상황이 잘 종료될 수 있기 때문이다.

그런데 video/broadcast object로 setChannel()을 사용할 경우에는 일부 OTS STB의 요구사항을 준수할 수 없게 된다. Middleware의 입장에서는 setChannel()로 채널이 변경되는 것은 정상적인 채널 변경이기 때문에 이전 채널 및 채널 up/down에 있어서 그 기준이 된다. 그런데 독립형 어플리케이션이 종료되면서 이동하는 promo channel의 경우에는 실제 의도적인 채널 변경으로

인한 채널로 인정되지 않아서 channel up/down이 독립형 채널 진입 이전 채널 기준으로 동작해야 한다. 그래서 특별한 API를 통해서 promo channel로 이동해야 한다.

Solution

Observer에게 promo channel로 이동 요청을 하면, observer가 promo channel로 이동시키면서 여러 가지 상황을 정리하는 방식으로 정리되어 있다. 그래서 observer에게 obs_setPromoChannel이라는 web messaging을 이용해서 promo channel로 이동하면 된다.

Observer에게 메시지 보내는 방법은 "6 Inter-Application Communication"을 참고한다. Observer의 dvb.appId는 4e30.3000 이며, obs_setPromoChannel의 message format은 다음과 같다.

```
message = {
      'method' : 'obs_setPromoChannel'b}
```

13.7. Video Resize for Channel

Background

어플리케이션에서 특정 서비스를 위해 영상 크기를 변경을 한 상태에서 다른 어플리케이션이 video/broadcast object을 생성하면 재생 중인 영상의 크기에 영향을 준다. 이유는 video/broadcast object 생성 시, 기본으로 설정된 영상 크기 혹은 사용자가 설정한 영상 크기로 설정을 시도하기 때문이다. Middleware의 입장에서는 정상적인 영상 크기 변경 요청이라서 무시할 수가 없다. 그렇기 때문에 기존 설정된 영상 크기를 유지하면서 서비스하려는 어플리케이션의 요구를 충족할 수 없다.

Solution

이 문제를 해결하기 위해서는 video/broadcast 에 'allocationMethod'라는 값을 parameter로 추가해야 한다.

'allocationMethod' 가 "DYNAMIC_ALLOCATION"으로 설정된 parameter가 전달되면 영상 크기 변경은 채널 변경 관련 operation(bindToCurrentChannel, setChannel, prevChannel, nextChannel) 이후에 가능하다. 해당 parameter를 전달하여 video/broadcast object을 생성하면 채널 변경 관련 operation 이전에는 영상 크기를 변경하지 않고 값을 내부적으로 가지고 있는다. 그리고 채널 변경 관련 operation이 일어나는 순간 최종적으로 반영한 크기에 맞춰서 영상 크기를 변경시킨다.

```
<object type="video/broadcast">
<param name="allocationMethod" value="DYNAMIC_ALLOCATION">
</object>
```

```
var v = oipfObjectFactory.createVideoBroadcastObject();
var param = document.createElement("param");
```

```
param.setAttribute("name", "allocationMethod");
param.setAttribute("value", "DYNAMIC_ALLOCATION");
v.appendChild(param);
document.getElementById('content').appendChild(v);
```

14. Metadata

14.1. Application/oipfSearchManager

EPG 관련 검색 기능만 제공한다. createSearch() 시에 항상 searchTarget:1 을 사용하여야 한다. 메타데이터의 변경 관련 onMetadataUpdate(Action:1) 이벤트만 지원하고 있으며 어플리케이션에서는 해당 이벤트 발생 시 관련 EPG 데이터를 새로 업데이트 하여야 한다.

14.2. MetadataSearch

addRatingConstraint(), orderBy() 등은 지원 하지 않는다. 단 orderBy()의 경우 기본적으로 channel no, starttime 순으로 반영된다.

14.3. SearchResults

getResult()는 항상 동기적으로 처리 한다. 메타데이터 변경으로 인한 문제는 onMetadataUpdate() 이벤트 발생 시 어플리케이션에서 다시 데이터를 조회하여 처리해야 한다. 내부 리소스 관리나 어플리케이션의 문제로 발생 할 수 있는 leak을 방지 하기 위해서 실제 페이징 모델을 지원 하지 않는다. 어플리케이션은 많은 데이터를 조회하게 될 경우 채널 별 혹은 시간 별로 적절히 조회하여 처리해야 한다.

14.4. Programme

Programme 정보에서는 기본적으로 DVB EIT-S 에서 확인 할 수 있는 정보들만 조회가 가능하다. 따라서 programmeIDType은 항상 ID_DVB_EVENT 이다. Programme은 name, description, startTime, duration, channelID, programmeID, channel properties 정보 들만 지원 되며 그 외에 필요로 하는 데이터 들은 어플리케이션에서 getSIDescriptors()를 통해 각 descriptor 정보를 가지고 처리하면 된다.

14.5. Example

```
var vb = oipfObjectFactory.createVideoBroadcastObject();
document.getElementById('playerDiv').appendChild(vb);

var sm = oipfObjectFactory.createSearchManagerObject();
var search = sm.createSearch(1);
```

```
var cc = vb.getChannelConfig();
var curtime = parseInt(new Date().getTime()/1000);

var qry = search.createQuery("programme.startTime", 5, curtime);
qry = qry.and(search.createQuery("(programme.startTime + programme.duration)", 2, curtime));
search.setQuery(qry);

for (var c = 0; c < cc.channelList.length && c < 3; c++) {
	search.addChannelConstraint(cc.channelList[c]);
}

search.result.getResults(0, 1000);

for (var i = 0; i < search.result.length; i++) {
	console.log (search.result[i].channelID + ":" + search.result[i].startTime);
}
```

15. Storage

15.1. USB Storage Device

USB storage device에 저장된 파일에 접근하기 위해서 StorageManager를 사용한다.

15.1.1. How to List USB Files

다음과 같은 방법으로 USB storage의 최상위 파일들을 나열할 수 있다.

```
var storageManager = oipfObjectFactory.createStorageManagerObject();
var storageList = storageManager.storages;


for(var i = 0; i < storageList.length; i++) {
console.log(storageList[i].vendorName + ":" + storageList[i].modelName + ":" + storageList[i].type);
console.log(storageList[i].free + ":" + storageList[i].total);

var rootfolder = storageList[i].rootFolder;
console.log("The # of children: " + rootfolder.totalSize);
storageList[i].onStorageAction =
function(action, result, progress, file) {
    console.log(file.name + ":" + file.uri + ":" + file.type);
};

rootfolder.getPage(0, rootfolder.totalSize);
}}
```

15.1.2. How to Play USB Media File

다음과 같은 방법으로 MPEG file을 play할 수 있다.

```
var storageManager = oipfObjectFactory.createStorageManagerObject();
var storageList = storageManager.storages;

var rootfolder = storageList[0].rootFolder;
rootfolder.getPage(0, 5);

var videoPlayer = window.oipfObjectFactory.createVideoMpegObject();
document.getElementById('playerDiv').appendChild(videoPlayer);
videoPlayer. data = rootfolder.item(0).uri;
videoPlayer.play(1);
```

15.2. Widget Storage

Widget storage의 현재 저장 공간 상태를 확인하기 위해서도 Storage object를 이용해서 그 기능을 제공한다. Storage 중에서 type이 STORAGE_TYPE_FLASH (1)이고, rootFolder의 이름이 "WIDGET_STORAGE"인 것이 widget storage이다. 다음과 같은 방법으로 전체 size와 현재 남은 size를 구할 수 있다. Size는 MB 단위이다.

```
var storages = oipfObjectFactory.createStorageManagerObject().storages;
for (var i = 0; i < storages.length; i++) {
if (storages[i].type == 1 && storages[i].rootFolder.name == 'WIDGET_STORAGE') {
console.log('total=' + storages[i].total + 'MB;free=' + storages[i].free + 'MB');
}
}
```

16. Multiple Media Handling

현재 2개 이상의 video를 동시에 재생하거나 video와 별도의 audio를 재생하는 방법은 현재 완벽하게 자유롭지는 않고 다소 제한된 형태로 제공되고 있다.

2nd Video Play

현재 2개의 video를 동시에 재생할 수 있지만, main video와 달리 sub video (main video보다 상위에서 재생됨)의 경우에는 VOD를 재생할 수 없다.

기본적으로 video object를 사용하여 재생하게 되면, main screen에 재생되게 된다. 특별히 sub video로 재생하여 main screen 상위에 재생되게 하려면 'changeMainService', 'videoPlaneId'라는 값을 parameter로 추가해야 한다.

먼저 'videoPlaneId'는 "1"로 설정한다. 그리고 'changeMainService'를 "true"로 설정하면 main screen 자체의 video를 변경하는 것이기 때문에 현재 채널이 변경되게 되며, 이로 인해서 bound application이라면 종료되게 된다. "false"로 설정하면 main screen의 채널은 유지된 상태에서 main screen 상위에 지정한 크기로 video를 재생하게 된다. 단 main screen의 채널이 유지된다고 의미가 main screen의 채널 video가 유지된다는 뜻은 아니다. 그래서 main screen의 video가 재생되는 경우라면 비록 'changeMainService'를 'false'로 했다고 해도 sub video 재생을 위해서 자원이 뺏기게 되므로 main screen의 video는 꺼지게 된다.

```
<object type="video/broadcast" width=”300” height=”300”>
<param name="changeMainService" value="false">
<param name="videoPlaneId" value="1">
</object>
```

```
var v = oipfObjectFactory.createVideoBroadcastObject();
v.width=”300”;
v.height=”300”;
var param = document.createElement("param");
param.setAttribute("name", "changeMainService");
param.setAttribute("value", "false");
param.setAttribute("name", " videoPlaneId ");
param.setAttribute("value", "1");
v.appendChild(param);
document.getElementById('content').appendChild(v);
```

Audio Selection between 2 video objects

아직 audio mixing 기능을 제공하지 못하여 한 순간에 하나의 audio만 최종 음성 출력으로 나가게 된다. 그래서 하나 이상의 video object가 audio를 활성화하는 경우에는 가장

videoPlaneId가 낮은 video object의 audio가 출력되게 된다. 단 모든 video object가 audio를 비활성화되어 있는 상태라면 videoPlaneId가 0인 video의 audio는 활성화된 것처럼 처리된다. 이유는 기본적으로 audio가 모두 비활성화된 경우에도 하나의 audio는 출력 audio가 되는 것을 기본 개념으로 하기 때문이다.

다음은 PIP를 위한 video/broadcast object를 생성하는 예제이다. 또한 PIP audio를 오디오 출력으로 내 보내고 싶은 경우이다.

```
<object type="video/broadcast" width=”300” height=”300”>
   <param name="changeMainService" value="false">
   <param name=”videoPlaneId” value=”1”>
<param name= “audio" value="true">
</object>
```

```
var v = oipfObjectFactory.createVideoBroadcastObject();
v.width=”300”;
v.height=”300”;
var param = document.createElement(“param”);
param.setAttribute("name", "changeMainService");
param.setAttribute("value", "false");
param.setAttribute("name", " videoPlaneId ");
param.setAttribute("value", "1");
v.appendChild(param);
document.getElementById('content').appendChild(v);
v.enableAudio();
```

만약 PIP를 종료하는 경우에는 일부러 audio를 비활성화시켜야 한다. 그래서 이런 경우 아래와 같이 명시적으로 다시 audio를 비활성화시켜야 한다.

```
v.disableAudio();
```

현재 audio selection 기능은 video/broadcast object와 video/mpeg object에 대해서만 지원하며, video tag나 audio tag로 지원하지 않는다. 그래서 만약 어떤 어플리케이션이 AV 위에 별도 audio file의 audio를 출력하려고 한다면 audio tag를 사용할 수 없다. 현재 video/mpeg object로 audio 재생이 가능하기 때문에 임시로 다음과 같은 방법으로 재생하여 AV 위에 별도 audio를 재생하며 그 audio를 최종 오디오 출력으로 내보낼 수 있다.

```
// create main video/broadcast object
var v = oipfObjectFactory.createVideoBroadcastObject();
v.width=”300”;
v.height=”300”;
document.getElementById('content').appendChild(v);

// create audio object using video/mpeg
var a = oipfObjectFactory.createVideoMpegObject();
```

```
a.width=”300”;
a.height=”300”;
var param = document.createElement(“param”);
param.setAttribute("name", "changeMainService");
param.setAttribute("value", "false");
param.setAttribute("name", " videoPlaneId ");
param.setAttribute("value", "1");
a.appendChild(param);
document.getElementById(‘content’).appendChild(a);
a.enableAudio();
a.data = song.mp3
```

API는 “Appendix F Extended APIs for video/mpeg, video/broadcast embedded object” 참조.

17. Video Caption

Video의 자막을 처리하기 위해서 <track> tag를 사용한다. W3C 표준에 따라서 <track> tag는 WebVTT만 지원한다. 하지만 기존에 많이 사용되어 오고 있었던 SAMI format을 지원하기 위해서 별도로 WebVTT file로 변환해 주는 웹 서비스를 제공하고 있다.

SAMI-to-WebVTT conversion

다음과 같이 호출하여 Sami에 해당하는 WebVTT byte stream을 얻을 수 있다.

```
http://localhost:8092/vtt/conv?uri=[URL of SAMI file]&type=text/sami&lang=[language]
```

- uri: 실제 SAMI 파일이 있는 경로. (http, file만 지원)
- lang: SAMI에서 추출해야 하는 언어. ISO639-ISO3166 format을 따른다. 예) "en-US"

위와 같이 요청하면 SAMI 파일에서 지정한 language에 해당하는 언어를 기준으로 WebVTT file을 byte stream으로 제공해 준다.

결국 이 web service API를 이용해서 다음과 같이 video에 자막 처리를 할 수 있다.

```
<video width="1280" height="720">
<source src="forrest_gump.mp4" type="video/mp4">
<track
src="http://localhost:8092/vtt/conv?uri=http://subtitle.com/subtitles_en.smi&type=text/sami&lang=en-US"
kind="subtitles" srclang="en">
<track
src="http://localhost:8092/vtt/conv?uri=http://subtitle.com/subtitles_ko.smi&type=text/sami&lang=ko-KR"
kind="subtitles" srclang="ko">
</video>
```

SAMI language list

현재 SAMI 파일이 제공하고 있는 자막 언어의 목록을 모를 경우에는 이 값을 얻어낼 수 있는 방법은 다음과 같다.

```
http://localhost:8092/vtt/sami/lang?uri=[URL of SAMI file]
```

- uri: 실제 SAMI 파일이 있는 경로. (http, file만 지원)

위와 같이 요청하면 다음과 같이 JSON object 형태로 응답을 준다. 이 때 key는 "language"가 되고 그 value가 지원 언어가 되는 array를 되돌려 준다.

{"language", [lang1, lang2, ...]}

18. Video Capture

HTML5에서는 canvas의 drawImage()를 이용해서 동영상의 한 장면을 캡쳐할 수 있다. 다음은 동영상을 캡쳐하는 예제이다.

```
<!DOCTYPE html>
<title>Video/Canvas Demo 1</title>
<script src=jquery.min.js></script>

<script>
function draw(v,c,w,h) {
if(v.paused || v.ended)         return false;
c.drawImage(v,0,0,w,h);
}

function capture() {
var v = document.getElementById('v');
var canvas = document.getElementById('c');
var context = canvas.getContext('2d');

draw(v, context, canvas.width, canvas.height);
}
</script>

<table id="t">
<tr>
<td>
<video id=v height="270" width="480" controls loop autoplay>
<source src="http://html5doctor.com/demos/video-canvas-magic/video.mp4" type="video/mp4" />
</video>
</td>
<td width="50" align="center">
<input type="button" value="->" id="i" onClick="capture()" />
</td>
<td width="480" align="center">
<canvas id=c height="270" width="360" />
</td>
</table>

<style>
body {
background: white;
}

#t {
position: absolute;
top: 25%;
left: 10%;
}
</style>
```

19. Audio Clip

작은 크기의 오디오 데이터를 재생중인 AV와 mixing 되는 효과음으로 재생되는 기능을 제공한다.

자세한 API 설명은 "Appendix K Audio Clip APIs"을 참조한다.

19.1. 제약 사항

- MP3 포맷 파일만 재생이 가능하다.
- 파일 하나의 크기는 60KB 이하로 제한된다.

19.2. 유의 사항

XMLHttpRequest을 이용하여 오디오 데이터를 가져오는데 로컬 파일의 경우 XMLHttpRequest.status는 0으로 반환된다.

19.3. Example

```
var mp3Data; // XMLHttpRequest로 데이터를 가져왔다고 가정
var volume;

var audioClipService = window.oipfObjectFactory.createAudioClipServiceObject();

volume = 0.5;
audioClipService.play(mp3Data, volume);

volume = 1.0;
audioClipService.play(mp3Data, volume, true); // loop audio stream

audioClipService.stop();
```

20. Media Metadata

미디어 파일에서 이미지 추출 및 메타데이터를 제공하는 기능을 제공한다.

자세한 API 설명은 "Appendix L Media Metadata APIs"을 참조한다.

20.1. 제약 사항

이미지를 추출할 수 있는 미디어의 종류는 다음과 같다.

- 비디오
- Thumbnail 정보를 가지고 있는 오디오
- 이미지 (그림 파일)

메타데이터는 파일 포맷에 따라 제공되는 데이터가 나뉘어 진다. Property는 제공되나 존재하는 데이터만 전달해준다.

음악 파일인 경우 저장된 ID3 정보를 기반으로 제공되며 데이터는 아래와 같다.

- Album
- Artists
- Title
- Track

이미지 파일인 경우 제공되는 데이터는 아래와 같다.

- Rotate

20.2. Example

```
function byteArrayToString(bArray) {
	var result = "";
	for(var i = 0; i < bArray.length; i++) {
		result += String.fromCharCode((bArray[i]&0xFF));
	}
	return result;
}

function imageExtracted(result, imageData) {
	var image = document.createElement('img');
	var imagedata = btoa(byteArrayToString(imageData));
	image.src = '' + imagedata;
```

```
        document.body.appendChild(image);
}

var mediaMetadataUtil = oipfObjectFactory.createMediaMetadataUtilObject();

var mediaMetadata;
mediaMetadata = mediaMetadataUtil.getMediaMetadata("file:///gem_storage/2/music/aaa.mp3");
console.log("mediaMetadata.album = " + mediaMetadata.album);
console.log("mediaMetadata.artists = " + mediaMetadata.artists);
console.log("mediaMetadata.title = " + mediaMetadata.title);
console.log("mediaMetadata.track = " + mediaMetadata.track);
mediaMetadata.extractImage(50, 50, imageExtracted);
....중략....
mediaMetadata.close(); // mediaMetadata 사용이 끝나면 반드시 close() 호출하여 리소스를 해제한다.

mediaMetadata = mediaMetadataUtil.getMediaMetadata("file:///gem_storage/2/music/bbb.mp3");
....중략....
mediaMetadata.close();


mediaMetadataUtil.close(); // MediaMetadataUtil 의 사용이 끝나면 이 역시 close() 호출하여 리소스를 해제한다.
```

21. Voice Data

리모콘의 녹음 기능을 이용하여 어플리케이션에게 Voice Data를 제공한다.

자세한 API 설명은 "Appendix M Voice Data APIs"을 참조한다.

21.1. Data Format

어플리케이션에 전달되는 Voice Data Format은 아래와 같다.

- File Format: wave
- Encoding : PCM signed
- Bits : 16
- Endian : little
- Channels : 1 (mono)
- Sample rate : 16000 Hz

21.2. 유의 사항

리모콘에 있는 녹음기능이 초기화가 되어야 하기에 API 호출만으로는 Voice Data를 전달 받을 수가 없다. 리모콘 "음성검색"키 (VK_WINK) 를 눌러야 녹음 기능이 초기화 되므로 어플리케이션은 "음성검색" 키를 받아서 API를 이용하고 키 이벤트는 stopPropagation()를 통해 consume 해야 한다. 만약 키 이벤트를 consume하지 않는다면 STB에서 제공하는 음성인식 어플리케이션이 구동된다.

21.3. Example

```
var voiceDataManager = oipfObjectFactory.createVoiceDataManagerObject();

voiceDataManager.onstart = function voiceStart() {
    // 시작 처리
console.log(ager.onstart = function v
}

voiceDataManager.onend = function voiceEnd(aborted) {
    // 종료 처리
console.log("voice recording end - " + " aborted);
}
```

```
voiceDataManager.onresult = function voiceResult(data) {
	// 데이터 처리
console.log(“voice recording result - data length = ” + data.length);
}

voiceDataManager.onerror = function voiceError(code) {
	// 오류 처리
console.log(“voice recording error - code = ” + code);
}

document.addEventListener("keydown", keydown);

function keydown(event) {
	switch(event.keyCode) {
		case VK_WINK :
voiceDataManager.start();
event.preventDefault();
			event.stopPropagation();
			break;
	}
}
```

22. Speech Recognizer

음성검색 키를 이용하여 어플리케이션에게 음성 인식 기능을 제공한다.

22.1. How to recognize speech

음성 인식은 2가지 방법으로 제공이 된다. Input field에서 직접 받는 방식과 API를 이용하여 처리하는 방식이 있다.

22.1.1. Input Field

Input field에 음성 인식 결과를 전달할 수 있다. 이를 위해서 확장 attribute인 x-kt-voice를 제공하고 있다. Input field에 포커스가 있는 상태에서 음성검색 키를 누르면 음성 인식이 진행되며 인식된 결과를 화면에 노출하여 사용자가 입력하기 원하는 적절한 단어를 선택하게 한다.

```
<input type=”input” x-kt-voice/>
```

22.1.2. API

Input field에 전달되는 것 외에 API를 이용하여 음성 인식 결과를 이벤트로 전달 받을 수 있다.

자세한 API 설명은 “Appendix N Speech Recognizer APIs”을 참조한다.

22.1.2.1. 유의사항

리모콘에 있는 녹음기능이 초기화가 되어야 하기에 API 호출만으로는 음성 인식 결과를 전달 받을 수가 없다. 리모콘 “음성검색”키 (VK_WINK) 를 눌러야 녹음 기능이 초기화 되므로 어플리케이션은 “음성검색” 키를 받아서 API를 이용하고 키 이벤트는 stopPropagation()를 통해 consume 해야 한다. 만약 키 이벤트를 consume하지 않는다면 STB에서 제공하는 음성인식 어플리케이션이 구동된다.

22.1.2.2. Example

```
var speechRecognizer = oipfObjectFactory.createSpeechRecognizerObject();

speechRecognizer.onrecognized = function recognized (input) {
    // 결과 처리
```

```
        console.log(“speech recognized – input = “ + input);
}

document.addEventListener("keydown", keydown);

function keydown(event) {
        switch(event.keyCode) {
                case VK_WINK :
                        speechRecognizer.start();
                        event.preventDefault();
                        event.stopPropagation();
                        break;
        }
}
```

23. Mosaic Window

4개의 화면에 채널을 동시 재생할 수 있는 기능을 제공한다.

자세한 설명은 "Appendix O Mosaic Window APIs"을 참조한다.

23.1. 제약사항

아래와 같은 제약사항이 있으며 STB 환경[1]에 따라서 지원이 가능할 수도 있다.

- 화면의 개수는 4개여야만 한다.
- 구성한 화면의 영역이 겹치면 안 된다.
- 화면의 크기에 제약 조건이 존재한다.
- 생성된 화면의 비디오 크기는 실행 중에 변경되지 않는다.

23.2. Example

```
// index0 에 해당하는 화면 크기 및 위치 설정
var screenPosition0 = new ScreenPosition(0, 0, 320, 240);
...중략...
// index3 에 해당하는 화면 크기 및 위치 설정
var screenPosition3 = new ScreenPosition(350, 300, 320, 240);

// 배열로 만들어둔다.
var screenPositions = [screenPosition0, screenPosition1, screenPosition2, screenPosition3];

// 4개의 화면 위치를 넘겨주면서 MosaicWindow 를 생성한다.
var mosaicWindow = oipfObjectFactory.createMosaicWindow(screenPositions);

// 각 화면에 채널을 재생한다. channel0,3은 임의의 channel object이다.
mosaicWindow.setChannel(0, channel0);
...중략...
mosaicWindow.setChannel(3, channel3);

// 특정 화면의 오디오가 나오도록 설정한다.
mosaicWindow.selectMosaicAudio(0);

// MosaicWindow 가 들고 있는 VideoBroadcast Object 를 직접 컨트롤 하고 싶을 때에는
```

[1] 특정 STB 만 고려한다면 별도로 제약사항 문의가 필요하다.

```
// 아래 API 로 VideoBroadcast 객체를 얻어와 직접 컨트롤 할 수 있다. (단, video size 관련 operation 불가)
var mvbos = mosaicWindow.getVideoBroadcastObjects();

// 특정 화면의 채널 재생을 중지한다.
mosaicWindow.stop(2);

// 모든 화면의 채널 재생을 중지한다.
mosaicWindow.stopAll();

// 생성한 MosaicWindow 를 종료 한다.
mosaicWindow.close();
```

24. Zip Extractor

Zip 파일을 추출하는 기능을 제공한다. Javascript library를 사용하는 경우 메모리를 많이 사용하는 경우가 존재하여 플랫폼에서 제공하는 기능이다.

자세한 설명은 "Appendix P Zip Extractor APIs"을 참고한다.

24.1. 유의사항

압축 해제/실패/취소 시 압축을 풀려는 디렉토리의 모든 파일을 삭제한다. 즉, 압푹을 풀려는 디렉토리는 항상 삭제가 되므로 다른 파일이 존재하는 디렉토리를 대상으로 사용하지 않도록 주의해야 한다.

어플리케이션에서 동일 디렉토리를 반복해서 사용한다면 크게 문제가 되지는 않겠지만 매번 다른 디렉토리를 사용하는 경우 확률은 적지만 중간에 박스가 꺼지거나 하는 경우 삭제되지 않는 경우가 발생할 수 있기 때문에 어플리케이션에서 주의해서 디렉토리를 관리해야 한다.

24.2. Example

```
var ze = oipfObjectFactory.createZipExtractorObject();

var callback =    {
      "notifyZipExtractorState" : function (handle, state) {
            console.log("handle = " + handle + " , state = " + state);
      }
};

var handle = ze.requestExtraction("/data/unzip/test.zip", "/data/unzip/test_unzip", callback);
```

25. TV Service Extension

"kt olleh tv web extension framework"에 정의되지 않은 TV 서비스 관련된 확장 기능을 제공한다.

자세한 설명은 "Appendix Q TV Service Extension APIs"을 참고한다.

Appendix A. Specification Conformance Table

A.1. kt Specification

"kt olleh tv web extension framework v.1.0.9"를 기준으로 현재 web middleware가 지원하지 못하는 부분이나 지원 일정은 다음과 같다.

- 3.3 Country/Language code
  - 한국어만 지원
- 3.4 Power state
  - Off, On, Active Standby만 지원
- 4.16 Garbage collector
  - Asynchronous GC 형태로 추후 지원 예정
- 4.19 Dialog box
  - window.prompt()는 추후 지원 예정
- 4.20 Automatic Login
  - 추후 지원 예정
- 4.22.5 Pointing event
  - 전체 어플리케이션이 pointing event를 지원하지는 않을 것이므로 지원 방안 협의 필요
- 4.25 Widgets
  - Packaging feature 중에서 "Internationalization"과 "localization"은 미지원
  - Digital Signature는 추후 지원 예정
- 24.7 The WebSocket class
  - Binary data 전송 기능 미지원
- 26. NPAPI
  - 화면을 직접 그리는 방식 미지원

A.2. Web Specifications

기본적인 표준 지원 사항은 다음과 같다.

- HTML5
- DOM Level 3
- CSS3
- javascript 1.8

A.2.1. HTML5

자세한 지원 사항은 http://www.html5test.com을 참조한다.

Section	Subsection	Supported	Comment
Semantics	Richer tag sets	O	
	microdata	O	
	microformat	O	
Offline and storage	App cache	O	
	Local storage	O	
	IndexedDB	O	*webkit indexed DB만 지원
	File API	X	*File reader : 1Q, 2013
Device access	Geolocation	X	*Can support if device supports.
	STB device access	O	
Connectivity	Web sockets	O	
	Server-Sent events	O	
Multimedia	video	O	
	audio	O	
3D, Graphics and effects	SVG	O	*Inline SVG supported
	Canvas	O	
	WebGL	O	
	CSS3 3D	O	
Performance and integration	Web Workers	O	
	XMLHttpRequest 2	O	
CSS3 styling	CSS3	△	

A.2.2. DOM Level 3

DOM Level 3에 대해서 기본적인 지원 사항은 다음과 같다.

- DOM Level 3 Core (Partial)
- DOM Level 3 Events (Partial)
- DOM Level 3 XPath (Partial)
- DOM Level 3 Load and Save (Not support)
- DOM Level 3 Validation (Not support)
- DOM Level 3 Views and Formatting Specification (Not support)

- DOM Level 3 Abstract Schemas (Not support)

다음은 세부적인 DOM level 3 지원 사항이다.

A.2.2.1. DOM Level 3 Core (Partial)

규격 세부 사항은 http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/idl-definitions.html을 참조한다.

DOMImplementationRegistry interface (X)

Type	Name	Supported
Method	DOMImplementation getDOMImplementation(String features)	X
	DOMImplementationList getDOMImplementationList(String features)	X

DOMException interface

Type	Name	Supported
Constant	VALIDATION_ERR	O
	TYPE_MISMATCH_ERR	O

DOMStringList interface

Type	Name	Supported
Property	readonly attribute unsigned long length	O
Method	DOMString item(in unsigned long index)	O
	boolean contains(in DOMString str)	O

NameList interface (X)

Type	Name	Supported
Property	readonly attribute unsigned long length	X
Method	DOMString getName(in unsigned long index)	X
	DOMString getNamespaceURI(in unsigned long index)	X
	boolean contains(in DOMString str)	X
	boolean containsNS(in DOMString namespaceURI, in DOMString name)	X

DOMImplementationList interface (X)

Type	Name	Supported
Property	readonly attribute unsigned long length	X
Method	DOMImplementation item(in unsigned long index)	X

DOMImplementationSource interface (X)

Type	Name	Supported
Method	DOMImplementation getDOMImplementation(in DOMString features)	X
	DOMImplementationList getDOMImplementationList(in DOMString features)	X

DOMImplementation interface

Type	Name	Supported
Method	DOMObject getFeature(in DOMString feature, in DOMString version)	X

Node interface

Type	Name	Supported
Property	readonly attribute DOMString baseURI	O
Method	Node insertBefore(in Node newChild, in Node refChild) raises(DOMException)	O
	Node replaceChild(in Node newChild, in Node oldChild) raises(DOMException)	O
	Node removeChild(in Node oldChild) raises(DOMException)	O
	Node appendChild(in Node newChild) raises(DOMException)	O
	unsigned short compareDocumentPosition(in Node other) raises(DOMException)	O
	attribute DOMString textContent // raises(DOMException) on setting	O
	boolean isSameNode(in Node other)	O
	DOMString lookupPrefix(in DOMString namespaceURI)	O

	boolean isDefaultNamespace(in DOMString namespaceURI)	O
	DOMString lookupNamespaceURI(in DOMString prefix)	O
	boolean isEqualNode(in Node arg)	O
	DOMObject getFeature(in DOMString feature, in DOMString version)	X
	DOMUserData setUserData(in DOMString key, in DOMUserData data, in UserDataHandler handler)	X
	DOMUserData getUserData(in DOMString key)	X

Attr interface

Type	Name	Supported
Property	readonly attribute boolean isId	O

Element interface

Type	Name	Supported
Property	readonly attribute TypeInfo schemaTypeInfo	X
Method	void setIdAttribute(in DOMString name, in boolean isId) raises(DOMException)	X
	void setIdAttributeNS(in DOMString namespaceURI, in DOMString localName, in boolean isId) raises(DOMException)	X
	void setIdAttributeNode(in Attr idAttr, in boolean isId) raises(DOMException)	X

Text interface

Type	Name	Supported
Property	readonly attribute boolean isElementContentWhitespace	X
	readonly attribute DOMString wholeText	O
Method	Text replaceWholeText(in DOMString content) raises(DOMException)	O

TypeInfo interface (X)

Type	Name	Supported
Constant	DERIVATION_RESTRICTION	X
	DERIVATION_EXTENSION	X
	DERIVATION_UNION	X
	DERIVATION_LIST	X
Property	readonly attribute DOMString typeName	X
	readonly attribute DOMString typeNamespace	X
Method	boolean isDerivedFrom(in DOMString typeNamespaceArg, in DOMString typeNameArg, in unsigned long derivationMethod)	X

UseDataHandler interface (X)

Type	Name	Supported
Constant	NODE_CLONED	X
	NODE_IMPORTED	X
	NODE_DELETED	X
	NODE_RENAMED	X
	NODE_ADOPTED	X
Method	void handle(in unsigned short operation, in DOMString key, in DOMUserData data, in Node src, in Node dst)	X

DOMError interface (X)

Type	Name	Supported
Constant	SEVERITY_WARNING	X
	SEVERITY_ERROR	X
	SEVERITY_FATAL_ERROR	X
	readonly attribute unsigned short severity	X
	readonly attribute DOMString message	X
	readonly attribute DOMString type	X
	readonly attribute DOMObject relatedException	X
	readonly attribute DOMObject relatedData	X
	readonly attribute DOMLocator location	X

DOMErrorHandler function (X)

Type	Name	Supported
Method	boolean handleError(in DOMError error)	X

DOMLocator interface (X)

Type	Name	Supported
Property	readonly attribute long lineNumber	X
	readonly attribute long columnNumber	X
	readonly attribute long byteOffset	X
	readonly attribute long utf16Offset	X
	readonly attribute Node relatedNode	X
	readonly attribute DOMString uri	X

DOMConfiguration interface (X)

Type	Name	Supported
Property	readonly attribute DOMStringList parameterNames	X
Method	void setParameter(in DOMString name, in DOMUserData value) raises(DOMException)	X
	DOMUserData getParameter(in DOMString name) raises(DOMException)	X
	boolean canSetParameter(in DOMString name, in DOMUserData value)	X

Entity interface

Type	Name	Supported
Property	readonly attribute DOMString inputEncoding	X
	readonly attribute DOMString xmlEncoding	X
	readonly attribute DOMString xmlVersion	X

Document interface

Type	Name	Supported
Property	readonly attribute DocumentType doctype	O
	readonly attribute DOMString inputEncoding	O
	readonly attribute DOMString xmlEncoding	O
	attribute boolean xmlStandalone // raises(DOMException) on setting	O
	attribute DOMString xmlVersion //	O

	raises(DOMException) on setting	
	attribute boolean strictErrorChecking	X
	attribute DOMString documentURI	O
	readonly attribute DOMConfiguration domConfig	X
Method	Node adoptNode(in Node source) raises(DOMException)	O
	void normalizeDocument()	X
	Node renameNode(in Node n, in DOMString namespaceURI, in DOMString qualifiedName) raises(DOMException)	X

A.2.2.2. DOM Level 3 Events (Partial)

규격 세부 사항은 http://www.w3.org/TR/2011/WD-DOM-Level-3-Events-20110531/#webidl-definitions를 참조한다.

Event interface

Type	Name	Supported
Property	readonly attribute boolean defaultPrevented	O
	readonly attribute boolean isTrusted	X
Method	void stopImmediatePropagation()	O

CustomEvent interface

Type	Name	Supported
Property	readonly attribute any detail	O
Method	void initCustomEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in any detailArg)	O

EventTarget interface

Type	Name	Supported
Method	boolean dispatchEvent(in Event evt) raises(EventException)	O

EventException interface

Type	Name	Supported

Constant	DISPATCH_REQUEST_ERR = 1	O
Property	unsigned short code	O

DocumentEvent interface

Type	Name	Supported
Method	Event createEvent(in DOMString eventInterface) raises(DOMException)	O

FocusEvent interface (X)

Type	Name	Supported
Property	readonly attribute EventTarget relatedTarget	X
Method	void initFocusEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in AbstractView viewArg, in long detailArg, in EventTarget relatedTargetArg)	X

MouseEvent interface

Type	Name	Supported
Method	boolean getModifierState(in DOMString keyArg)	X

MouseEvent interface

Type	Name	Supported
Constant	DOM_DELTA_PIXEL = 0x00	X
	DOM_DELTA_LINE = 0x01	X
	DOM_DELTA_PAGE = 0x02	X
	Property readonly attribute float deltaX	X
	readonly attribute float deltaY	X
	readonly attribute float deltaZ	X
	readonly attribute unsigned long deltaMode	X
Method	void initWheelEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in AbstractView viewArg, in long detailArg, in long screenXArg, in long screenYArg, in long clientXArg, in long clientYArg, in unsigned short buttonArg, in EventTarget relatedTargetArg, in DOMString modifiersListArg, in float deltaXArg, in float	X

	deltaYArg, in float deltaZArg, in unsigned long deltaMode)	

TextEvent interface

Type	Name	Supported
Constant	DOM_INPUT_METHOD_UNKNOWN = 0x00	X
	DOM_INPUT_METHOD_KEYBOARD = 0x01	X
	DOM_INPUT_METHOD_PASTE = 0x02	X
	DOM_INPUT_METHOD_DROP =0x03	X
	DOM_INPUT_METHOD_IME = 0x04	X
	DOM_INPUT_METHOD_OPTION = 0x05	X
	DOM_INPUT_METHOD_HANDWRITING = 0x06	X
	DOM_INPUT_METHOD_VOICE = 0x07	X
	DOM_INPUT_METHOD_MULTIMODAL = 0x08	X
	DOM_INPUT_METHOD_SCRIPT = 0x09	X
Property	readonly attribute DOMString data	O
	readonly attribute unsigned long inputMethod	X
	readonly attribute DOMString locale	X
Method	void initTextEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in AbstractView viewArg, in DOMString dataArg, in unsigned long inputMethod, in DOMString localeArg)	X
	void initTextEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in AbstractView viewArg, in DOMString dataArg)	O

KeyboardEvent interface

Type	Name	Supported
Constant	DOM_KEY_LOCATION_STANDARD = 0x00	O
	DOM_KEY_LOCATION_LEFT = 0x01	O
	DOM_KEY_LOCATION_RIGHT = 0x02	O
	DOM_KEY_LOCATION_NUMPAD = 0x03	O
	DOM_KEY_LOCATION_MOBILE = 0x04	X
	DOM_KEY_LOCATION_JOYSTICK = 0x05	X

Property	readonly attribute DOMString char	X
	readonly attribute DOMString key	X
	*readonly attribute DOMString keyIdentifier	O
	readonly attribute unsigned long location	X
	*readonly attribute DOMString keyLocation	O
	readonly attribute boolean ctrlKey	O
	readonly attribute boolean shiftKey	O
	readonly attribute boolean altKey	O
	readonly attribute boolean metaKey	O
	readonly attribute boolean repeat	X
	readonly attribute DOMString locale	X
	*readonly attribute boolean altGraphKey	O
	*readonly attriubte long keyCode	O
	*readonly attribute long charCode	O
	Method boolean getModifierState(in DOMString keyArg)	X
	void initKeyboardEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in AbstractView viewArg, in DOMString charArg, in DOMString keyArg, in unsigned long locationArg, in DOMString modifiersListArg, in boolean repeat, in DOMString localeArg)	X
	void initKeyboardEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in AbstractView viewArg, in DOMString keyIdentifier, in unsigned long keyLocation, in boolean ctrlKey, in boolean altKey, in boolean shiftKey, in boolean metaKey, in boolean altGraphKey)	O

CompositionEvent interface

Type	Name	Supported
Property	readonly attribute DOMString data	O
	readonly attribute DOMString locale	X
Method	void initCompositionEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in AbstractView viewArg, in DOMString dataArg, in DOMString localeArg)	X

	*void initCompositionEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in AbstractView viewArg, in DOMString dataArg)	O

MutationNameEvent interface (X)

Type	Name	Supported
Property	readonly attribute DOMString prevNamespaceURI	X
	readonly attribute DOMString prevNodeName	X
	void initMutationNameEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in Node relatedNodeArg, in DOMString prevNamespaceURIArg, in DOMString prevNodeNameArg)	X

A.2.2.3. DOM Level 3 XPath (Partial)

규격 세부 사항은 http://www.w3.org/TR/2004/NOTE-DOM-Level-3-XPath-20040226/idl-definitions.html를 참조한다.

Supported interfaces (All constants, properties, and all methods)

XPathEvaluator

XPathExpression

XPathNSResolver

XPathResult

Not supported interfaces

XPathNamespace

A.2.2.4. DOM Level 3 Load and Save (Not support)

규격 세부 사항은 http://www.w3.org/TR/2004/REC-DOM-Level-3-LS-20040407/idl-definitions.html을 참조한다.

A.2.2.5. DOM Level 3 Validation (Not support)

규격 세부 사항은 http://www.w3.org/TR/2004/REC-DOM-Level-3-Val-20040127/idl-definitions.html을 참조한다.

A.2.2.6. DOM Level 3 Views and Formatting Specification (Not support)

규격 세부 사항은 http://www.w3.org/TR/2004/NOTE-DOM-Level-3-Views-20040226/idl-definitions.html을 참조한다.

A.2.2.7. DOM Level 3 Abstract Schemas (Not support)

규격 세부 사항은 http://www.w3.org/TR/2002/NOTE-DOM-Level-3-AS-20020725/idl-definitions.html을 참조한다.

A.2.3. CSS3

자세한 지원 사항은 http://css3test.com을 참조한다.

Section	Subsection	Supported	Comment
(Recommendation)	Media Queries	O	
	CSS Namespaces Module	O	
	Selectors Level 3	O	
	CSS Color Module Level 3	O	
(Candidate Recommendation)	CSS Values and Units Module Level 3	O	
	CSS Backgrounds and Borders Module Level 3	O	
	CSS Image Values and Replaced Content Module Level 3	O	
	CSS Speech Module	X	
	CSS Multi-column Layout Module	O	
	CSS Marquee Module Level 3	X	
(Last call drafts)	CSS Basic User Interface Module Level 3 (CSS3 UI)	O	

(Working drafts)	CSS Conditional Rules Module Level 3	X	
	CSS Fonts Module Level 3	O	
	CSS Regions Module Level 3	O	
	CSS Fragmentation Module Level 3	X	
	CSS Text Level 3	X	
	CSS Box Alignment Module Level 3	X	
	CSS Exclusions and Shapes Module Level 3	X	
	CSS Writing Modes Module Level 3	X	
	CSS Animations	O	
	CSS Transitions	O	
	CSS Positioned Layout Module Level 3	O	
	CSS Template Layout Module	X	
	CSS3 Ruby Module	X	
	CSS Lists and Counters Module Level 3	△	
	CSS Grid Positioning Module Level 3	X	

Appendix B. Reference Web Sites for Developer

웹 어플리케이션 개발을 위해서 필요한 참고 사이트는 아래와 같다.

설 명	참고 URL
HTML5 규격	http://www.w3.org/TR/2012/CR-html5-20121217/
W3C HTML 문법 확인	http://validator.w3.org/
JavaScript 코드 최적화	https://developers.google.com/speed/pagespeed/
Chrome 브라우저를 이용한 개발자 툴 사용 방법	https://developers.google.com/chrome-developer-tools/docs/overview?hl=ko-KR
Safari 브라우저를 이용한 개발자 툴 사용 방법	http://www.apple.com/kr/developer/technologies/safari/developer-tools.html

Table 9 References for Developer

Appendix C. Understanding Memory Usage (Informative)

Smart STB에는 크게 kernel memory 영역과 driver memory 영역으로 구분 관리한다. 그리고 웹 어플리케이션이 구동되면 다음과 같이 메모리를 사용하게 된다.

- **kernel memory**: 어플리케이션 리소스 (HTML, CSS, image file), compiled javascript, data, DOM tree, rendering tree, 그 외 어플리케이션이 사용하는 javascript API에 따라서 일시적으로 사용되는 web middleware memory 등
- **driver memory**: 주로 그래픽스를 그리기 위한 graphic buffer 및 각 decoded image가 차지하는 메모리

1280X960의 4byte color-depth 기준으로 했을 때 웹 어플리케이션 하나에 대해서 대략 다음과 같은 기본 메모리가 사용된다.

- kernel memory: 2MB
- driver memory: 8MB

현재까지 개발된 STB 기준으로 대략적인 예상 memory 할당량은 다음과 같다. 이는 추후 STB 기능 개발 상황에 따라 다소 변경될 수 있다.

구분	항목	메모리 할당량	
		Smart STB	UHD STB
KERNEL	기본(OS, TV Core, Host app)	100 MB	600 MB
	Unbound App	200 MB	200 MB
	Full Browser	168 MB	100 MB
	안정성 buffer	50 MB	832 MB
	계	518 MB	1,732 MB
BMEM	Driver + PL graphics buffer	486 MB	1,307 MB
합계		1,004 MB	3,039 MB

Appendix D. Comparison between OTS and OTV

어플리케이션 개발 환경은 service 형상을 기준으로 OTS (ollehtv skylife)와 OTV (ollehtv)의 2가지로 나뉜다. 두 service 형상에 대해서 차이점은 다음과 같다.

	OTS	OTV
모델명	SMT-E5015	OTV-SMT-E5015
채널 목록	위성 채널과 IPTV 채널이 모두 존재한다. 따라서 idType이 ID_DVB_S인 채널과 ID_IPTV_SDS인 채널이 모두 존재한다.	IPTV 채널만 존재한다. 모든 채널의 idType이 ID_IPTV_SDS이다.
부팅 후 초기 채널	스카이라이프 프로모 채널이 초기 채널이 되며, 위성 신호 연결이 없을 때에는 black 화면에 오류 팝업이 뜬다.	kt의 프로모 채널이 초기 채널이 된다.
채널번호 표기	9번의 경우 "9"와 같이 표기된다.	9번의 경우 "009"와 같이 표기된다.

Appendix E. Strict Mode

다음 표에는 strict 모드에서 적용되는 주요 제한 사항들을 설명하고 있다.

언어 요소 / 제한	오류	예제
변수 선언하지 않고 변수 사용.	SCRIPT5042: strict 모드에서 변수가 정의되지 않았습니다.	testvar = 4;

읽기 전용 속성
읽기 전용 속성에 쓰기.
	SCRIPT5045: strict 모드에서는 읽기 전용 속성에 할당할 수 없습니다.	var testObj = Object.defineProperties({}, {
		prop1: {
		value: 10,
		writable: false // by default
		},
		prop2: {
		get: function () {
		}
		}
		});
		testObj.prop1 = 20;
		testObj.prop2 = 30;

확장할 수 없는 속성 extensible 특성이 false로 설정된 개체에 속성 추가.	SCRIPT5046: 확장 가능하지 않은 개체에 대해 속성을 만들 수 없습니다.	var testObj = new Object(); Object.preventExtensions(testObj); testObj.name = "Bob";

delete
변수, 함수 또는 인수 삭제.
configurable 특성이 false로 설정된 속성 삭제.
	SCRIPT1045: strict 모드에서는 <식>에 대해 delete를 호출할 수 없습니다.	var testvar = 15;
		function testFunc() {};
		delete testvar;
		delete testFunc;
		
		Object.defineProperty(testObj, "testvar", {
		value: 10,
		configurable: false
		});
		delete testObj.testvar;

속성 중복	SCRIPT1046: strict	var testObj = {

개체 리터럴에서 속성을 두 번 이상 정의.	모드에서는 한 속성을 여러 번 정의할 수 없습니다.	prop1: 10, prop2: 15, prop1: 20 };

매개 변수 이름 중복
함수에서 매개 변수 이름을 두 번 이상 사용.
	SCRIPT1038: strict 모드에서는 정식 매개 변수 이름이 중복될 수 없습니다.	function testFunc(param1, param1) {
		return 1;
		};

다음에 사용하기 위한 예약 키워드 다음에 사용하기 위한 예약 키워드를 변수 또는 함수 이름으로 사용.	SCRIPT1050: 식별자에 대해 다음에 사용하기 위한 예약어를 잘못 사용했습니다.strict 모드에서는 식별자 이름이 예약됩니다.	implements interface let package private protected public static yield
8진수 숫자 리터럴에 8진수 값을 할당하거나 8진수 값에 이스케이프를 사용하려고 함.	SCRIPT1039: strict 모드에서는 8진수 숫자 리터럴 및 이스케이프 문자를 사용할 수 없습니다.	var testoctal = 010; var testescape = ₩010;

this
this 의 값이 null 또는 undefined인 경우 전역 개체로 변환되지 않습니다.
		function testFunc() {
		return this;
		}
		var testvar = testFunc();
		
		strict 모드가 아닌 경우 testvar 값은 전역 개체이지만 strict 모드에서 이 값은 undefined입니다.

식별자인 eval 문자열 "eval"은 식별자 (변수 또는		var eval = 10;

함수 이름, 매개 변수 이름 등)로 사용할 수 없습니다.		

문 또는 블록 내에서 선언된 함수
문 또는 블록 내에서 함수를 선언할 수 없습니다.
	SCRIPT1047: strict 모드에서는 함수 선언을 문이나 블록 내에 중첩할 수 없습니다. 함수 선언은 함수 본문 내에 직접 나오거나 최상위 수준에만 나와야 합니다.	var arr = [1, 2, 3, 4, 5];
		var index = null;
		for (index in arr) {
		function myFunc() {};
		}

eval 함수 내에서 선언된 변수
변수가 eval 함수 내에서 선언되는 경우 해당 함수 밖에서 사용할 수 없습니다.
	SCRIPT1041: strict 모드에서 'eval'을 잘못 사용했습니다.	eval("var testvar = 10");
		testvar = 15;
		간접 계산이 가능하지만 eval 함수 외부에 선언된 변수를 여전히 사용할 수 없습니다.
		
		var indirectEval = eval;
		indirectEval("var testvar = 10;");
		document.write(testVar);
		이 코드는 SCRIPT5009: 'testVar'이 정의되지 않았습니다. 오류를 발생시킵니다.

식별자인 Arguments 문자열 "arguments"는 식별자 (변수 또는 함수 이름, 매개 변수 이름 등)로 사용할 수 없습니다.	SCRIPT1042: strict 모드에서 'arguments'를 잘못 사용했습니다.	var arguments = 10;

함수 내 arguments
로컬 arguments 개체의 멤버 값을 변경할 수 없습니다.
		function testArgs(oneArg) {
		arguments[0] = 20;
		}
		strict 모드가 아닌 경우 `arguments[0]` 값을 변경하여 `oneArg` 매개 변수의 값을 변경할 수 있습니다. 그러면 `oneArg` 및 `arguments[0]`의 값이 모두 20이 됩니다.strict

		모드에서 `arguments[0]` 값을 변경해도 `oneArg` 값에 영향을 주지 않습니다. arguments 개체가 로컬 복사본이기 때문입니다.

arguments.callee
허용되지 않습니다.
		function (testInt) {
		if (testInt-- == 0)
		return;
		arguments.callee(testInt--);
		}

with
허용되지 않습니다.
	SCRIPT1037: strict 모드에서는 'with' 문을 사용할 수 없습니다.	with (Math){
		x = cos(3);
		y = tan(7);
		}

Table 10 Restrictions on Code in Strict Mode

Appendix F. Extended APIs for video/mpeg, video/broadcast embedded object

F.1. Common APIs

아래 properties와 methods는 video/mpeg와 video/broadcast embedded object에 공통적으로 제공되는 기능이다.

Properties

readonly Integer zindex [Deprecated]	
Description	Video의 z-index 값으로 가장 하위에 존재하는 main video가 0이 된다. 만약 sub video가 하나 더 재생 가능하다면 그 값이 1이 되며, 그 외 3개 이상 video 동시 재생이 가능하다면 이에 따라 zindex는 동시 재생 가능한 video 개수까지 사용할 수 있다. 현재는 2개까지 지원되므로 main video는 0, sub video는 1이다. Default 값은 0이다. [Deprecated] zindex="0"은 changeMainService="true", videoPlaneId="0"으로 zindex="1"은 changeMainService="false", videoPlaneId="0"으로 해석되어 동작한다.

readonly Integer audio	
Description	오디오를 출력으로 내 보낼 것인지 여부 값. True면 이 오디오가 출력 오디오가 된다. 단 동시에 2개 이상이 audio가 true이면 가장 videoPlaneId가 높은 video의 audio가 출력으로 내보내지게 된다. 반면 모든 audio가 false라면 videoPlaneId가 0인 main video의 audio가 무조건 출력으로 내보내지게 된다. videoPlaneId가 0이면 default 값이 true이고, videoPlaneId가 1 이상이면 default가 false이다.

readonly Integer changeMainService	
Description	Main screen의 서비스 자체를 변경 여부 값. True면 main screen의 서비스 자체를 변경하려는 목적으로 사용된다. 그래서 현재 채널에 bound된 어플리케이션이면 종료되게 된다. False면 main screen의 서비스는 유지된다.

readonly Integer videoPlaneId	
Description	재생하려는 video의 plane을 설정하기 위한 값. Main video에 재생하기 위해서는

	0으로 설정이 되어야 하며 sub video에 재생하기 위해서는 1로 설정이 되어야 한다. changeMainService가 true인 경우에는 main screen의 video를 변경하는 것이기 때문에 videoPlaneId의 값이 반드시 0이어야 한다. 그래서 0이 아닌 값을 설정하더라도 0으로 설정된 것처럼 동작한다. (main screen 변경하는 것이므로 무조건 main video에 재생이 된다.)

Methods

void enableAudio()	
Description	오디오를 활성화시킨다.

void disableAudio()	
Description	오디오를 비활성화시킨다.

F.2. Video/broadcast Object APIs

Video/broadcast object만 제공하는 기능이다.

Methods

readonly Boolean channelControl	
Description	일반적인 sub video로 사용할 경우 false이지만, PIP와 같이 특정 key event로 채널 변경 등이 가능하도록 요청된 경우에는 true이다.

void SIDescriptorCollection getPMTDescriptos(Integer descriptorTag)	
Description	Descriptor tag에 해당하는 PMT descriptor 정보를 얻는다. SIDescriptorCollection 참조
Argument	descriptorTag: descriptor 정보를 filtering하기 위한 descriptor tag. -1인 경우에는 모든 descriptor를 리턴한다.
Return	요청한 tag에 해당하는 descriptor 정보를 SIDescriptorCollection 형태로 리턴한다. 해당하는 descriptor가 없으면 null을 리턴한다.

Events

function onPMTChanged()	
Description	PMT가 변경되었을 경우 발생한다. PMT descriptor를 조회할 때 해당 이벤트를 이용하여 update 처리를 할 수 있다.

DOM2	PMTChanged

F.3. The SIDescriptor class

Properties

readonly Integer tag	
Description	Descriptor tag

readonly Integer length	
Description	Descriptor content 전체 길이 (단위: 바이트)

readonly Array<Integer> contents	
Description	Content data

F.4. The SIDescriptorCollection class

SIDescriptor object의 콜렉션이다.

Typedef Collection<SIDescriptor> SIDescriptorCollection	
Description	SIDescriptor collection

Appendix G. Media Timeline Control APIs

MediaTimeLineControl은 VOD player에 대한 확장 기능을 제공한다. 이 API는 VOD player에 대해서만 지원하며, 일반 media play나 video/broadcast object에 대해서는 지원하지 않는다. oipfObjectFactory.createMediaTimeLineControl()를 호출하여 생성한다.

G.1. The MediaTimeLineControl class

Methods

Integer getMediaTime()	
Description	PTS 기반의 media time 값을 millisecond 단위로 리턴한다.
Return	Millisecond 단위의 PTS media time

Boolean trickPlayPossible()	
Description	현재 재생 중인 position에서 rate, position변경이 가능한지 확인한다. 이 기능을 이용해서 VOD의 본편과 광고를 구분하는 용도로 사용할 수 있다.
Return	Rate나 position 변경이 가능하면 true를 리턴하고, 변경이 불가능하면 false를 리턴한다.

String getAdsInfo()	
Description	현재 play되고 있는 광고에 대한 정보를 리턴한다.
Return	Return 하는 String의 형태와 return 하는 시점 등은, firmware 변경 없이도 VOD 서버 측에서 변경이 가능한 사항이므로 여기에 명시하지 않으며, castis에 문의하도록 한다.

Events

function onMediaContentChanged(Integer event)	
Description	Play중인 content에 변동이 있을 때 이를 알려준다.
DOM2	MediaContentChanged
Argument	event: Castis event code

Appendix H. Closed Caption APIs

ClosedCaptionControl은 실시간 채널에 포함되는 closed caption을 처리할 때 사용되는 API이며, 아래와 같이 oipfObjectFactory.createClosedCaptionControl()에 video/broadcast object를 넘겨주어 생성한다.

```
<script>
function setupVideo() {
   var chPlayer = document.getElementById("video");
   var closedCaptionControl = oipfObjectFactory.createClosedCaptionControl(chPlayer);
  ...
}
</script>

<body>
   <object type="video/broadcast"  width="300"  height="300"  id="video"/>
</body>
```

H.1. The ClosedCaptionControl class

Properties

readonly Integer state	
Description	ClosedCaptionControl의 state. 아래와 같은 값을 가진다. 0: OFF 1: ON 2: ON_MUTE

Methods

ClosedCaptionInfoCollection getClosedCaptionInfo()	
Description	사용 가능한 closed caption 서비스 목록을 리턴한다.
Return	사용 가능한 서비스 목록

ClosedCaptionInfo getCurrentClosedCaptionInfo()	
Description	현재 설정되어 있는 closed caption 서비스의 정보를 리턴한다. Closed caption 서비스가 설정되어 있지 않으면 undefined를 리턴한다.
Return	현재 설정되어 있는 ClosedCaption 서비스의 정보

Boolean requestClosedCaption(ClosedCaptionInfo ccInfo, Boolean turnOnMute)	
Description	새로운 closed caption 서비스를 요청한다.
Argument	ccInfo: 설정할 closed caption 서비스 turnOnMute: 음소거 모드일 때만 closed caption을 표시할지에 대한 flag. true이면 음소거 모드일 때만 closed caption을 표시한다.
Return	요청이 성공한 경우 true를 리턴하고 실패한 경우 false를 리턴한다.

void cancelClosedCaption()	
Description	현재 사용중인 closed caption 서비스를 중지한다.

Events

function onClosedCaptionInfoChanged(Integer event)	
Description	Closed caption 정보가 변경된 것을 알려준다.
DOM2	ClosedCaptionInfoChanged
Argument	event: 변경 내용을 나타내며 아래와 같다. 1: PMT update에 의해서 closed caption 정보가 변경되었음 2: 채널이 변경되어 closed caption 정보가 변경되었음

function onClosedCaptionDataReceived(Array rawData, String extractedString)	
Description	Closed caption data를 전달한다
DOM2	ClosedCaptionDataReceived
Argument	rawData: 원 자막 데이터인 caption stream의 byte 배열. rawData에는 자막 문자열과 control code들이 포함되어 있다. extractedString: stream으로부터 화면에 출력될 문자열 부분만을 추출한 값. Middleware가 문자열 추출 작업을 하지 않을 경우 null일 수 있다.

H.2. The ClosedCaptionInfo class

Properties

readonly Integer closedCaptionServiceNumber	
Description	Closed caption의 service number

readonly String language	
Description	Language는 3글자의 언어코드이다. (ISO 639.2/B 참조)

readonly Boolean easyReader	
Description	초보자용 자막 서비스인지에 대한 flag true: 초보자용 자막 서비스 false: 초보자용으로 맞춰지지 않은 자막 서비스

readonly Boolean wideAspectRatio	
Description	자막 서비스가 16:9 화면용인지 4:3 화면용인지에 대한 flag true: 16:9 화면용 자막 서비스 false: 4:3 화면용 자막 서비스

readonly Boolean unicodeKorean	
Description	한글 코드를 리턴한다. true: KS X ISO/IEC 10646 (유니코드) 한글 코드 false: KS X 1001:2004 완성형 한글 코드

H.3. The ClosedCaptionInfoCollection class

typedef Collection<ClosedCaptionInfo> ClosedCaptionInfoCollection	
Description	ClosedCaptionInfo의 collection

H.4. Examples

일부 채널의 경우 Closed Caption 서비스를 사용할 수 있다. 아래는 Closed Caption service를 사용하는 간략한 예제이다.

```
var v = oipfObjectFactory.createVideoBroadcastObject();
document.getElementById('video').appendChild(v);

var chlist = v.getChannelConfig().channelList;
v.setChannel(chlist[0]);

var ccc = oipfObjectFactory.createClosedCaptionControl(v);

var _onClosedCaptionInfoChanged = function(event) {
    var ccInfo = ccc.getClosedCaptionInfo();

    if(event == 1) {
        // PMT update.
        ccc.requestClosedCaption(ccInfo[0].closedCaptionServiceNumber);
    } else if(event == 2) {
```

```
            // New service selected.
            ccc.requestClosedCaption(ccInfo[0].closedCaptionServiceNumber);
        }
};

var _onClosedCaptionDataReceived = function(rawData, extractedString) {
        ...
};

var ccInfo = ccc.getClosedCaptionInfo();

ccc.requestClosedCaption(ccInfo[0], false);

ccc.onClosedCaptionInfoChanged = _onClosedCaptionInfoChanged;
ccc.onClosedCaptionDataReceived = _onClosedCaptionDataReceived;
```

Appendix I. WebView APIs

I.1. Application/x-alticast-webview object APIs

Type 이 "application/x-alticast-webview" 인 오브젝트를 통해 웹 브라우저 기능을 제공한다.

Using HTML tag

```
<object type="application/x-alticast-webview"></object>
```

Using Javascript

```
var webview = document.createElement("object");
webview.type = "application/x-alticast-webview";
parentNode.appendChild(webview);
```

Properties

Integer scrollX	
Description	

Integer scrollY	
Description	

Boolean focused	
Description	포커스를 가지면 true로 설정해야 함

Boolean enableFlash	
Description	

Boolean mouseCursorVisible	
Description	true: cursor visible false : cursor invisible

Readonly Integer previousHistorySize	
Description	

Readonly Integer forwardHistorySize

Description	

Readonly Integer allHistorySize	
Description	

Methods

void load(String url)	
Description	

void cancel()	
Description	

void reload()	
Description	

void goToHistory(HistoryItem historyItem)	
Description	

HistoryItem historyItem(Integer index)	
Description	
Argument	0: Current Negative: Back Pository: Forward

Collection<HistoryItem> historyItemAll()	
Description	History object array. Null if no history item.

Boolean captureScreen(String path, Integer width, Integer height)	
Description	
Argument	Path: Path of the captured image Width: Optional. Width of captured image Height: Optional. Height of captured image

Boolean dispatchKeyDown(KeyboardEvent event)	
Description	Return true if event is processed
Argument	event: keyboard event

Boolean dispatchKeyPress(KeyboardEvent event)	
Description	Return true if event is processed
Argument	event: keyboard event

Boolean dispatchKeyUp(KeyboardEvent event)	
Description	Return true if event is processed
Argument	event: keyboard event

void editText(String text)	
Description	Update whole text of the input editor
Argument	text: text

void editEnd()	
Description	End input state

void evaluate(String script)	
Description	Evaluate javascript
Argument	script: String of javascript code

void show()	
Description	Show application/x-alticast-webview plugin contents. Invalidate automatically.

void hide()	
Description	Hide application/x-alticast-webview plugin contents. Invalidate automatically.

Events

function onTitle()	
Description	
DOM2	Title

function onEditBegin()	
Description	
DOM2	EditBegin

function onEditEnd()

Description	
DOM2	EditEnd

function onURL()	
Description	
DOM2	URL

function onAlert(String msg)	
Description	
DOM2	Alert
Argument	msg: Message string to display

function onConfirm(String msg) Boolean	
Description	
DOM2	Confirm
Argument	msg: Message string to display
Return	true: confirm false: not confirm

function onLoadStarted()	
Description	
DOM2	LoadStarted

function onLoadProgress(Number progress)	
Description	
DOM2	LoadProgress
Argument	progress: 0.0 ~ 1.0

function onLoadFinished(Boolean ok, Number reason, String desc)	
Description	
DOM2	LoadFinished
Argument	ok: Success or fail reason: The reason code of failure. 0 if ok is true desc: Description of failure. Empty if ok is true

I.2. HistoryItem class

Properties

Readonly String title	
Description	

Readonly String url	
Description	

Appendix J. Mouse Control APIs

J.1. The MouseControlObject class

마우스 기능을 제어하는 API를 제공한다.

Properties

readonly Boolean irMouseEmulation	
Description	IR 가상 마우스가 활성화되어 있으면 true이다. true인 경우에는 방향키, "확인" 키를 가상 마우스 에뮬레이션을 위해서 사용하여 key event로 발생되지 않는다.

Methods

void setIRMouseEmulation(Boolean enable)	
Description	IR 가상 마우스 기능을 켜거나 끈다.

void requestMouseControl()	
Description	Mouse event를 사용하도록 요청한다. Mouse가 disableMouseControl()에 의해서 disable되어 있으면 requestMouseControl()을 호출하여도 mouse event가 발생하지 않는다.

void releaseMouseControl()	
Description	Mouse event를 사용하지 않도록 요청한다. 단 이 해제 요청은 기존의 사용 요청을 취소하는 것이므로, 다른 하나 이상의 어플리케이션이 여전히 requestMouseControl()으로 사용 요청한 상태이면, 여전히 mouse event는 사용할 수 있는 상태로 남아 있게 된다.

void disableMouseControl()	
Description	Mouse event를 사용하지 않는다. 이 설정은 모든 설정에 우선하여 disable되기 때문에 requestMouseControl()을 호출한 어플리케이션이 있다고 해도 mouse event를 사용할 수 없는 상태가 된다. 이 API는 일반 어플리케이션이 사용할 필요는 없고, 시스템 팝업과 같이 다른 어플리케이션과 같이 뜰 수 있는 어플리케이션이 mouse event 처리를 할 수 있도록 구현된 것이 아니라면 일시적으로 시스템 팝업을 띄울 때 mouse event를 처리하지 않도록 만들고 싶을 것이다. 이 때 사용하는 API이고 반드시 restoreMouseControl()을 사용하여 원래 상태로 돌려 놓아야 한다.

void restoreMouseControl()	
Description	Disable된 mouse 상태를 되돌려서 mouse event를 사용할 수 있도록 한다.

Appendix K. Audio Clip APIs

oipfObjectFactory.createAudioClipServiceObject()를 호출하여 생성한다

K.1.The AudioClipServiceObject class

Audio clip을 재생하고 정지하는 API를 제공한다.

Methods

void play(Blob audioClipData, Float gain, Boolean isLoop)	
Description	Audio clip을 재생한다. 동시에 하나씩만 재생이 가능하며 중복 요청 시 마지막 요청한 audio clip만 재생된다.
Argument	audioClipData : audio clip raw data gain : 음량 (0.0 ~ 1.0) isLoop : 반복 여부 (true - 반복 , false - 한번)

void stop()	
Description	Audio clip 재생을 정지한다.

Appendix L. Media Metadata APIs

oipfObjectFactory.createMediaMetadataUtilObject()를 호출하여 생성한다

L.1. The MediaMetadataUtilObject class

Methods

MediaMetadata getMediaMetadata(String path)	
Description	미디어 파일에서 MediaMetadata 객체를 가져온다.
Argument	path : file path
Return	MediaMetadata

void close()	
Description	MediaMetadataUtil이 사용하는 리소스를 해제한다.

L.2. The MediaMetadata class

Constants

Name	Value
RS_OK	0
RS_NOT_FOUND	1
RS_NOT_ACCEPTABLE	2
RS_ABORTED	3
RS_FAILED	4

Properties

readonly String album	
Description	음악파일의 메타데이터에 저장된 앨범명

readonly String artists	
Description	음악파일의 메타데이터에 저장된 참여 음악가

readonly String title	
Description	음악파일의 메타데이터에 저장된 제목

readonly String track	
Description	음악파일의 메타데이터에 저장된 트랙번호

readonly String rotate	
Description	이미지파일의 메타데이터에 저장된 방향정보

Methods

void extractImage(Integer width, Integer height, Callback func)	
Description	미디어 파일에서 이미지를 가져온다. 동시에 하나씩만 추출이 가능하며 중복 요청 시 Exception이 발생할 수 있다.
Argument	width : image width (pixel) height : image height (pixel) func : callback function. Function(Integer result, Uint8Array image)

void abort()	
Description	이미지 추출 작업 취소

void close()	
Description	MediaMetadata 객체가 사용하는 리소스를 해제한다.

Appendix M. Voice Data APIs

oipfObjectFactory.createVoiceDataManagerObject()를 호출하여 생성한다

M.1. The VoiceDataManagerObject class

Constants

Name	Value
ERROR_NOT_READY	1
ERROR_NO_DATA	2

Methods

void start()	
Description	음성 입력을 활성화 한다. 음성 입력 데이터가 존재하면 onresult로 데이터가 전달된다.

void abort()	
Description	활성화된 음성 입력을 중단시킨다.

Events

function onstart()	
Description	음성 입력 시작을 전달한다.
DOM2	start

function onend(Boolean aborted)	
Description	음성 입력 종료를 전달한다.
DOM2	end
Argument	aborted : 중단 여부 (true : 다른 app에 의해 종료, false : 그 외 경우)

function onresult(Byte[] data)	
Description	음성 입력 결과를 전달한다.
DOM2	result
Argument	data : 입력된 음성 데이터(byte[])를 전달한다.

function onerror(Integer errorcode)	
Description	음성 입력 오류를 전달한다.

DOM2	error
Argument	errorcode : 오류코드 (1 : ERROR_NOT_READY(입력준비 실패) , 2: ERROR_NO_DATA(음성데이터 없음))

Appendix N. Speech Recognizer APIs

oipfObjectFactory.createSpeechRecognizerObject()를 호출하여 생성한다.

N.1. The SpeechRecognizerObject class

Methods

void start(Boolean select)	
Description	음성 인식을 실행한다.
Argument	select : true이면 인식된 인식어 리스트가 표시되어 사용자가 선택할 수 있게 한다. false이면 첫번째에 있는 인식어가 자동으로 선택되어 전달한다.

void stop()	
Description	음성 인식 실행을 중단한다.

Events

function onrecognized(String input)	
Description	음성 인식 결과를 전달한다.
DOM2	recognized
Argument	input : 선택한 인식어

function onended()	
Description	음성 인식 종료를 전달한다. 명시적으로 stop()을 호출하거나 음성 인식 처리 중에 다른 application이 start() 요청을 하여 이전 요청이 유효하지 않는 경우 전달된다.
DOM2	ended

Appendix O. Mosaic Window APIs

O.1. The ScreenPosition class

화면의 위치 및 크기를 지정할 때 사용하는 클래스이다.
new ScreenPosition(Integer x, Integer y, Integer width, Integer height)를 통해 생성가능하며 MosaicWindow 생성 시 4개의 화면을 배열 형태로 전달하게 된다.

Properties

Integer x	
Description	화면의 x 좌표

Integer y	
Description	화면의 y 좌표

Integer width	
Description	화면의 폭

Integer height	
Description	화면의 높이

O.2. The MosaicWindow class

oipfObjectFactory.createMosaicWindow(ScreenPosition[] screenPosition)를 호출하여 생성한다.

Methods

void setChannel(Integer index, Channel channel)	
Description	전달받은 채널을 index에 해당하는 화면에 재생한다.
Argument	index : 제어하려는 화면의 접근 번호 channel : 재생할 channel object

void stop(Integer index)	
Description	index에 해당하는 화면의 채널 재생을 중지한다.
Argument	index : 제어하려는 화면의 접근 번호

void stopAll()	
Description	모든 화면의 채널 재생을 중지한다.

void selectMosaicAudio(Integer index)	
Description	오디오가 나올 화면을 지정한다.
Argument	index : 제어하려는 화면의 접근 번호

VideoBroadcast[] getVideoBroadcastObjects()	
Description	MosaicWindow가 가지고 있는 화면들의 VideoBroadcast object를 array 형태로 반환한다. 단, video size 관련 operation은 불가능하다.
Return	화면 개수에 해당하는 VideoBroadcast 배열

void close()	
Description	MosaicWindow를 종료한다.

Appendix P. Zip Extractor APIs

P.1. The ZipExtractor class

oipfObjectFactory.createZipExtractorObject()를 호출하여 생성한다.

Constants

Name	Value
ZIP_EXTRACTING_STARTED	0
ZIP_EXTRACTING_SUCCEEDED	1
ZIP_EXTRACTING_CANCELED	2
ZIP_EXTRACTING_FAILED	3

Methods

Integer requestExtraction(String path, String destDir, ZipExtractorCallback callback)	
Description	압축 파일 추출을 요청한다.
Argument	path : 압축을 풀려는 파일의 경로 destDir : 압축이 풀리는 디렉토리 callback : 압축 관련 이벤트를 전달 받기 위한
Return	handle : 압축을 요청한 handle 값을 전달한다. Callback에서 이 값을 이용하여 어떤 요청의 상태인지 확인 할 수 있다.

void cancelRequest(Integer handle)	
Description	압축 파일 추출 요청을 취소한다.
Argument	handle : 압축 파일 추출 요청 시 전달 받은 handle 값

P.2. The ZipExtractorCallback object

ZipExtractor를 이용하여 압축 해제 시 결과를 전달 받기 위한 callback object 이다. 생성한 Object에서 구현해야 할 Method는 아래와 같다.

Methods

void notifyZipExtractorState(Integer handle, Integer state)	
Description	압축 파일 추출 상태를 전달하는 callback method 이다.
Argument	handle : 압축 파일 추출 요청 시 전달 받은 handle 값 state : 압축 파일 추출 상태

Appendix Q. TV Service Extension APIs

Q.1. The TVServiceExtension class

oipfObjectFactory.createTVServiceExtensionObject()를 호출하여 생성한다.

Methods

void pauseChannel()	
Description	메인 채널 A/V를 정지시킨다. VOD 재생 시, 채널 multicast ip를 leave하여 traffic 확보하기 위해 사용된다.

void resumeChannel()	
Description	정지한 메인 채널 A/V를 다시 재생한다.

Channel getCloudGameChannel()	
Description	Wiz Game이 실행될 가상 채널을 전달한다. Wiz Game 실행 전에 이 API로 채널을 가져와서 VideobroadcastObject의 setChannel을 이용하여 채널 전환을 하면 된다.
Return	Channel : cloud game을 위한 가상 데이터 채널

Channel getActualChannel()	
Description	현재 채널이 hidden service인 경우 VideobroadcastObject의 currentChannel을 이용하면 이전 채널 정보를 가져오게 된다. 이 API는 hidden service 여부와 상관없이 현재 채널을 전달하게 된다.
Return	Channel : 서비스 타입에 상관 없이 현재 서비스 되고 있는 실제 채널

Appendix R. Sample Application

참고를 위해서 2개의 sample application의 source를 포함한다. 이 어플리케이션은 실제로 동작하는 어플리케이션으로써 적절한 위치에 두고 실행시키면 된다.

R.1.Program Metadata Viewer (program_viewer.html)

이 어플리케이션은 3개의 채널 정보를 화면에 출력하는 어플리케이션이다.

- 첫번째 prev, next라는 버튼을 이용하여, 각각 현재 화면에 출력된 채널을 기준으로 이전 3개 채널 혹은 다음 3개의 채널 정보를 보여준다.
- 화면에는 "프로그램 이름(시작시간)"을 형태로 정보를 표시한다.
- setChannel 버튼으로 해당 채널로 이동한다.

소스 코드는 다음과 같다.

```
<html>
<head>
<script>
var vb;
var sm;
var search;
var cc;
var ui;

function view(s) {
	var nitem = 3;
	var e;
	var c;
	var qry;
	var panel, bpanel;
	var prevbtn, nextbtn, sbtn;

	s = (s < 0)? 0: s;
	e = s + nitem;
	// clear panels
	document.getElementById("panelContainer").innerHTML = "";

	prevbtn = createButton("prev");
	nextbtn = createButton("next");
	prevbtn.onclick = function() { view( (s - nitem) ) }
	nextbtn.onclick = function() { view( (s + nitem) ) }
	bpanel = createPanel("button");
	bpanel.appendChild(prevbtn);
	bpanel.appendChild(nextbtn);

	// for test..
```

```
        qry = search.createQuery("programme.startTime", 5, new Date().getTime()/1000);
        search.setQuery(qry);

        for (c = s; c < cc.channelList.length && c < e; c++) {
                search.addChannelConstraint(cc.channelList[c]);
                search.result.getResults(0, 100);

                for (i = 0; i < search.result.length; i++) {
                        if (i == 0) {
                                panel = createPanel(search.result[i].channelID);
                                sbtn = createButton("setChannel");
                                sbtn.onclick = new function("console.log(cc.channelList[" + c + "]);
annelID);
                                sbtn = createButton("setChannel");
                                sbtn.onclick = new function("console.log(cc.channelList[" + c + "]);

                ccc.requestClosedCaption(ccInfo[0].closedCaptionServiceNumber);
        } else if(event == 2) {
                // New service selected.
                ccc.requestClosedCaption(ccInfo[0].closedCaptionServiceNumber);
        }
};

var _onClosedCaptionDataReceived = function(rawData, extractedString) {
        ...
};

var ccInfo = ccc.getClosedCaptionInfo();

ccc.requestClosedCaption(ccInfo[0], false);

ccc.onClosedCaptionInfoChanged = _onClosedCaptionInfoChanged;
ccc.onClosedCaptionDataReceived = _onClosedCaptionDataReceived;
```

```
ccInfo = ccc.getClosedCaptionInfo();

ccc.requestClosedCaption(ccInfo[0], false);

ccc.onClosedCaptionInfoChanged = _onClosedCaptionInfoChanged;
ccc.onClosedCaptionDataReceived = _onClosedCaptionDataReceived;
```

R.2.Channel Controller (channel_control.html)

이 어플리케이션은 채널 정보를 보여주고 채널 변경을 하는 어플리케이션이다.

- 좌상단에는 현재 재생 중인 채널의 major number, minor number, name, CCID, idType을 보여준다.
- 우상단에는 현재 재생 중인 채널 화면을 보여준다.
- 좌하단에는 "CH+", "CH-" 버튼이 있어서 채널 변경을 한다.

➢ 우하단에는 여러 가지 동작 결과의 로그를 출력한다.

소스코드는 다음과 같다.

```
<html>
<head>
	<title>Channel control Test</title>
</head>
<body>
	<table style="align:center;width:940px;height:520px">
		<tr style="height:80%">
			<td id="chinfo" style="background-color:#C8FFFF;width:20%"></td>
			<td style="width:80%">
				<object type="video/broadcast" id="content" width="100%" height="100%">
				</object>
			</td>
		</tr>
		<tr style="height:20%">
			<td id="menu" style="text-align:center;vertical-align:middle;background-
color:#FFFFC8;width:20%">
				<input type='button' id='prevChannel' value='CH-' onclick=onPrevChannel()>
				<input type='button' id='nextChannel' value='CH+' onclick=onNextChannel()>
			</td>
			<td id="chmessage" style="overflow-y:scroll;background-color:#C8FFFF;width:80%"></td>
		</tr>
	</table>

	<script type="text/javascript">
		var chdiv = document.getElementById('chinfo');
		var chmsg = document.getElementById('chmessage');

		messagelog = function(msg) {
			chmsg.innerHTML += msg + '<br>';
			console.log(msg);
		}

		messageclear = function(msg) {
			chmsg.innerHTML = '';
		}

		messagelog("Loading...");
		messagelog("Channel control Test start.");
		messagelog("Get video/broadcast object");

		var v = document.getElementById('content');

		messagelog("getChannelConfig()");
		var cc = v.getChannelConfig();

		messagelog("getChannelList()");
		var chlist = cc.channelList;
```

```
	channelChangeError = function(evt) {
		messagelog("ChannelChangeError, channel="+evt.channel+", error="+evt.error);
	}

	playStateChange = function(evt) {
		messagelog("PlayStateChange, state="+evt.state+", error="+evt.error+",
v.state="+v.playState);
	}

	channelChangeSucceeded = function(evt) {
		messagelog("ChannelChangeSucceeded, channel="+evt.channel+", v.state="+v.playState);
		updateChannelInfo();
	}

	fullScreenChange = function(evt) {
		messagelog("fullScreenChange, v.fullScreen="+v.fullScreen);
	}

	dumpProgrammes = function() {
		var p = v.programmes;
		messagelog("v.programmes.length="+p.length);
		for(var i = 0; i < p.length; i++) {
			messagelog('Programme['+i+']='+p[i].name+', '+p[i].startTime+', '+p[i].duration+','+
p[i].channelID);
		}
	}

	onNextChannel = function() {
		messageclear();
		messagelog('Next channel');
		v.nextChannel();
		updateChannelInfo();
	}

	onPrevChannel = function() {
		messageclear();
		messagelog('Prev channel');
		v.prevChannel();
		updateChannelInfo();
	}

	updateChannelInfo = function() {
		var curch = v.currentChannel;

		console.log('updateChannelInfo, curch='+curch+
',CC.currentChannel='+v.getChannelConfig().currentChannel);

		if(curch != null) {
			console.log('updateChannelInfo, curch='+curch.majorChannel+curch.minorChannel+
' '+curch.name);
chdiv.innerHTML = '<h2>'+curch.majorChannel+curch.minorChannel+
' '+curch.name+'</h2>'+'<h3>CCID : '+curch.ccid+'</h3>'+
```

```
'<h3>idType : '+curch.idType+'</h3>';
                    } else {
                        chdiv.innerHTML = '<h2>Unknown channel</h2>';
                    }
                }

                v.addEventListener('ChannelChangeError', channelChangeError);
                v.addEventListener('PlayStateChange', playStateChange);
                v.addEventListener('ChannelChangeSucceeded', channelChangeSucceeded);
                v.addEventListener('FullScreenChange', fullScreenChange);
                v.addEventListener('ProgrammesChanged', dumpProgrammes);

                v.bindToCurrentChannel();
                v.setFullScreen(false);

                updateChannelInfo();
        </script>
</body>
</html>
```