사용설명서 ver. 1.0K

C2_Flash UCI(User Creative Interface) 제작 지침

DIGITAL PRIDE.
COWON

변경 이력

Revision	Date	Description

1. C2 Flash UCI(User Creative Interface) 개요

C2는 Full UCI를 지원하는 것을 목적으로 개발되었습니다. 배경이나 아이콘 변경만 가능한 UCI가 아닌, 고급 Flash 개발자가 자유롭게 UI를 만들고 그에 따른 기기 제어가 가능한 Platform 성격의 기기입니다. 임베디드와 Flash라는 한계로 인해 기기의 모든 기능을 UCI에서 제어할 수는 없지만, 최대한 UCI 개발자가 자유롭게 개발할 수 있도록 구현되어 있습니다. 또한 개발 및 유지 보수를 원활하기 위해 각 기능을 모듈화 형태로 구현되어 있으며, 각 기능 제어에 필요한 FS Command들은 간단하고 명료하도록 구현되어 있습니다.
그렇기 때문에 기존 Flash ActionScript 개발자들은 C2의 '입력장치'와 그에 따른 'FS Command'처리만 기존 Flash ActionScript에서 추가적으로 이해한다면 UCI 개발에 큰 어려움이 없습니다.

원활한 UCI 개발 환경을 제공하기 위해서 C2는 각 동작 별로 Mode를 구분하여 각 Mode별로 모듈화 형태로 구현되었습니다. 즉 음악, 라디오, 비디오, 문서, 사진 등 C2 기능들이 하나의 Mode로 구분되어 하나의 Flash 컨텐츠로 구현되었습니다. 이에 따라 각 Mode의 GUI들은 독립적으로 구현 가능합니다. 하지만 반대로 한 Flash 컨텐츠 안에 다양한 Mode를 처리하는 방식도 역시 구현 가능합니다.
C2에서는 다양한 기능을 제어할 목적으로 'FS Command'들도 많이 존재합니다. 이 많은 'FS Command'들을 효율적으로 정리하기 위해 C2에서는 Tree 구조로 같은 성격을 띤 'FS Command'들끼리 묶여 있습니다.
Mode와 'FS Command' 그룹핑을 통해 C2에서는 모든 FS Command를 알지 못해도 UCI 개발이 가능합니다. 즉 Music에 관련된 FS Command들이 한 그룹으로 묶여 있기 때문에 Music에 관련된 FS Command만 이해를 한다면 Music UCI를 만들 수 있습니다.

2. C2 Flash Engine Specification

1). Flash Version : Flash Player 7
2). ActionScript Version : ActionScript 2.0
3). Display Resolution : 320 * 240
4). Frame Rate : 24 fps, launcher.swf에 따라 변경 가능
5). Sound : UI에서는 미지원
6). Font : Vector Font 지원
임베디드 기계 특성으로 인해 Action Script에서 폰트를 지정해도 '시스템 폰트' 혹은 '사용자 폰트'로만 출력
7). Rotation : Text Field를 제외하고는 제한 없음. Text Field 경우 'Single line'만 Rotation을 지원
8). Memory : 15MB

3. C2 소프트웨어 구조

1). 소프트웨어 구조 설명

C2의 Flash는 Full UCI를 구현하기 위해 '그림 3 – 1'과 같이 네 계층으로 구현되어 있습니다.

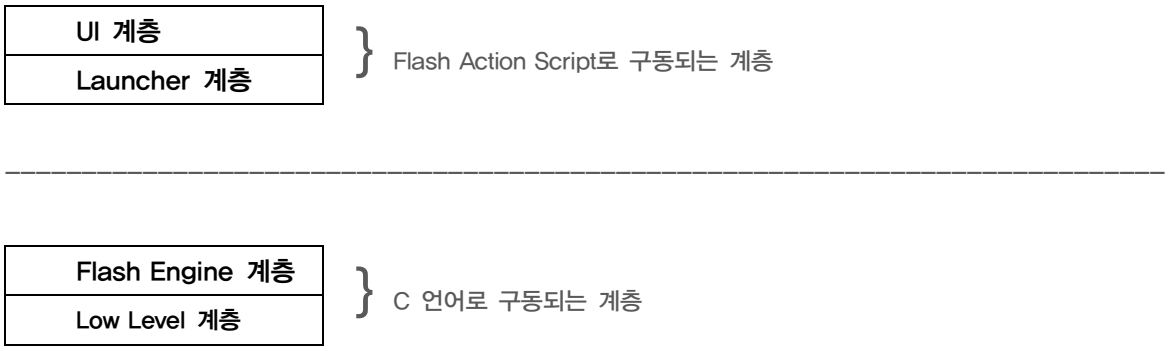

그림 3 – 1

- Low Level 계층 : Low Level Device(CPU, LCD, Codec 등) 제어
- Flash Engine 계층 : Flash 컨텐츠를 구동하며, Low Level 계층 이벤트 등을 Flash 컨텐츠로 전달
- Launcher 계층 : Flash에서 'Level0'에 위치하여 UI를 담당하는 Flash 컨텐츠를 Load, Unload 처리
- UI 계층 : Flash 'Level1'에 위치하며 일반 UI를 담당

이 구조는 일반 동작에서만 적용되며, 'USB' / '충전 화면' / '펌웨어 업그래이드', '시작 화면', '종료 화면' 등에서는 'Low Level 계층'에서 직접 화면을 제어하기 때문에 UCI가 적용되지 않습니다. 또한 LCD Off 한 상태에서는 구조적인 이유로 'Low Level 계층'만 동작하도록 구현되어 있습니다.

Launcher 계층과 UI 계층을 담당하는 Flash 컨텐츠 파일들은 펌웨어에 포함되어 가상으로 실행됩니다. 다만 기기 특정 폴더('System₩Flash UI')에 파일 형태로 존재하면 그 파일이 먼저 실행됩니다. 즉 음악을 실행하기 위해선 'loadMovieNum("System₩₩Flash UI₩₩music.swf",1)'를 사용하여 'music.swf'를 Load하는데, 해당 폴더에 파일이 있을 경우 폴더에 있는 파일이 먼저 실행되며, 파일이 없을 경우에는 펌웨어 내부에 있는 가상의 'music.swf' 파일을 실행됩니다.

2). Low Level 계층

CPU, LCD, Codec 등 Low Level Device를 제어하는 가장 하위 계층입니다. Flash Script에서는 FS Command를 통하여 미리 서로 정의되어 있는 Command를 통해서만 Low Level Device 제어 가능합니다. FS Command는 기본적으로 Synchronous Command이며 소수의 Command만 Asynchronous 하게 동작합니다. Asynchronous FS Command는 FS Command 설명시 별도로 언급되며, 언급이 없는 FS Command는 Synchronous로 생각하고 사용하면 됩니다.

3). Flash Engine 계층

Flash 컨텐츠를 해석하여 컨텐츠에 있는 Action Script에 따라 Flash 컨텐츠를 구동합니다. 그리고 Action Script에 따라서 FS Command를 호출하여 Low Level Device를 제어하거나, Low Level Device에서 발생한 Key, Touch, 중력 센서 등의 Event를 Action Script로 전달합니다.

Flash Engine에는 Flash 컨텐츠를 구동하기 위해 15MB(펌웨어 버전에 따라 다를 수 있습니다.)를 할당하고 있으며, 컨텐츠의 메모리가 15MB 이상을 사용할 경우 외부 Dynamic Memory를 자동으로 할당하여 사용합니다. 다만 Dynamic Memory는 음악 재생 및 동영상 재생 등에서도 사용하기 때문에 가능하면 Flash 컨텐츠를 15MB 이내로 구현하기를 권장합니다.

Flash Player(사용자가 브라우저를 통해서 Flash 컨텐츠를 재생하는 경우)는 별도의 메모리로 구동되는데 이 때 메모리의 크기는 5MB입니다.

4). Launcher 계층

Launcher 계층은 'launcher.swf'로 구현되어 있으며, UI를 담당하는 Flash 컨텐츠 파일을 Load, Unload하는 역할을 합니다. 이 파일에는 Global 함수로 LoadSWF() 함수가 구현되어 있는데, 이 함수를 통해 개발자들은 'Level1'에 원하는 Flash 컨텐츠를 올릴 수 있습니다.

LoadSWF() 함수를 실행하면 함수 내부에서 'unloadMovieNum(1)' 함수가 먼저 호출되어 이전에 LoadMovie 되었던 Flash 컨텐츠를 UnloadMoive합니다. 그 이후 입력된 ItemNum 값에 따라서 'loadMovieNum("System₩₩Flash UI₩₩music.swf",1)'가 호출되어 Flash 컨텐츠를 실행합니다.

- 함수 원형

  _global.LoadSWF = function(itemNumber:Number, fileName:String):Void

- 입력 값

  ItemNum : 실행할 Flash 컨텐츠의 Index. 입력값은 아래와 같습니다. 아래 Global 변수는 'launcher.swf'에 정의되어 있습니다. 다만 값 5, 9는 아무런 동작을 하지 않으며, 값 '_global.MODE_MAIN', '_global.MODE_MAIN2', '_global.MODE_MAIN3'은 모두 같은 MainMenu를 Load합니다. 만약 다른 MainMenu를 실행하기 위해서는 'EtcUsrSetMainmenu' FS Command를 사용해야합니다. 즉 '_global.LoadSWF(_global.MODE_MAIN)', '_global.LoadSWF(_global.MODE_MAIN2)', '_global.LoadSWF(_global.MODE_MAIN3)'로 다르게 호출해도 모두 동일한 MainMenu가 실행되지만, 'EtcUsrSetMainmenu'으로 '0 ~ 2'까지 값을 설정한 다음 '_global.LoadSWF(_global.MODE_MAIN)'를 호출하면 설정한 값에 따라 해당 MainMenu를 Load합니다.

  Text, Flash에서는 단일 파일이 실행되는 구조이기 때문에 브라우저가 먼저 실행되어 브라우저에서 직접 해당 컨텐츠를 선택했을 때에만 정상적으로 동작됩니다. 그렇기 때문에 '_global.MODE_TEXT'으로 'document.swf'를 Load하는 것은 브라우저에서 '문서' 파일을 선택했을 때만 Load해야합니다. Flash 컨텐츠는 별도로 UI가 필요 없기 때문에 펌웨어에서 바로

해당 컨텐츠를 Play합니다. '_global.MODE_FLASHBROWSER', '_global.MODE_TEXTBROWSER'가 존재하는 이유는 브라우저 실행 후 브라우저 초기화 시에 가장 최근에 실행한 문서나 Flash 파일을 지정하기 위한 역할입니다.

_global.MODE_MUSIC : '0' 값을 갖고 있는 Number형 변수이며 'music.swf' 파일 실행
_global.MODE_VIDEO : '1' 값을 갖고 있는 Number형 변수이며 'movie.swf' 파일 실행
_global.MODE_RADIO : '2' 값을 갖고 있는 Number형 변수이며 'radio.swf' 파일 실행
_global.MODE_RECORD : '3' 값을 갖고 있는 Number형 변수이며 'record.swf' 파일 실행
_global.MODE_DMB : '4' 값을 갖고 있는 Number형 변수이며 'record.swf' 파일 실행
_global.MODE_TEXT : '6' 값을 갖고 있는 Number형 변수이며 'text.swf' 파일 실행
_global.MODE_PICTURE : '7' 값을 갖고 있는 Number형 변수이며 'picture.swf' 파일 실행
_global.MODE_DICTIONARY : '8' 값을 갖고 있는 Number형 변수이며 'powerdicrun.swf' 파일 실행
_global.MODE_ETC : '10' 값을 갖고 있는 Number형 변수이며 입력 받은 'FileName'에 따라 파일 실행
_global.MODE_MAIN : '11' 값을 갖고 있는 Number형 변수
_global.MODE_MAIN2 : '12' 값을 갖고 있는 Number형 변수
_global.MODE_MAIN3 : '13' 값을 갖고 있는 Number형 변수
_global.MODE_SETTING : '14' 값을 갖고 있는 Number형 변수이며 'setting.swf' 파일 실행
_global.MODE_BROWSER : '15' 값을 갖고 있는 Number형 변수이며 'browser_total.swf' 파일 실행
_global.MODE_FLASHBROWSER : '17' 값을 갖고 있는 Number형 변수이며 'browser_total.swf' 파일 실행
_global.MODE_TEXTBROWSER : '18' 값을 갖고 있는 Number형 변수이며 'browser_total.swf' 파일 실행

FileName : ItemNum으로 'MODE_ETC'이 입력될 때만 처리되는 값으로, FileName을 가진 Flash 컨텐츠를 실행할 수 있습니다. 예를 들어 입력값이 'calculator.swf'이면 'System₩Flash UI₩calculator.swf' 파일이 실행됩니다.

– 반환 값

Void

5). UI(User Interface) 계층

실제 사용자가 기기를 사용하는데 필요한 UI가 구현되어 있는 계층입니다. UCI는 이 계층에서 구현됩니다.

4. Flash ActionScript 구조

1). Flash ActionScript 구조 설명

'2-1'에서 설명한 것과 같이 Flash ActionScript는 Launcher 계층에서 사용자가 선택한 Flash UI 파일을 실행하여 UI 계층이 동작하도록 되어있습니다. 그런데 음악, 비디오 UI 같이 단순 GUI처리 뿐만 아니라 파일 DB 및 재생 리스트

등 'Low Level 계층'에 대한 제어가 필요한 것들이 존재합니다. 만약 각각 FS Command를 정의하여 사용한다면 복잡도가 높아지고 그에 따른 기기 안정성이 떨어지는 문제가 발생됩니다. 따라서 C2에서는 각 Mode를 정의하여 음악에서 비디오로 전환시, 혹은 비디오에서 라디오 전환시 하나의 FS Command를 통해 Mode를 전환하도록 구현되어 있습니다. 이를 통해 Mode 초기화시에 발생하는 'Low Level 계층'에 대한 제어를 한 FS Command로 처리합니다.

2). Mode FS Command 설명

Mode 관련 FS Command는 'EtcModChangeMode'입니다. 이 FS Command를 통해 각 Mode에 맞는 'Low Level 계층'을 제어하도록 되어 있기 때문에 'music.swf', 'radio.swf' 등 GUI를 담당하는 Flash의 Action Script 맨 앞부분에 이 FS Command가 각 Mode에 맞게 호출되어야 합니다. 단 'Flash'와 'Text' Mode는 별도의 재생 리스트가 있는 것이 아닌 개별 파일이 재생되는 형태이기 때문에 브라우저 Action Script 초기 부분에서 'EtcModChangeMode' FS Command를 호출해야 합니다.

현재 Flash Script에서는 한 Flash UI 파일에서 하나의 Mode만 처리되도록 되어 있는데, 한 Flash UI 파일에서 다양한 Mode가 구현 가능합니다. 즉 'EtcModChangeMode' FS Command만 적절하게 호출하면 Text Viewer 동작 중 Music 이나 Radio 제어가 가능합니다. 또한 한 화면에서 Music을 듣다 Radio 청취 역시 가능하도록 구현되어 있습니다.

각 Mode 별 동작은 아래와 같습니다.

- 'Music' : 이전 재생 중이던 Mode를 중지한 후 최근 재생한 음악 파일로 DB 설정 및 재생 리스트 갱신합니다.
- 'Video' : 이전 재생 중이던 Mode를 중지한 후 최근 재생한 동영상 파일로 DB 설정 및 재생 리스트 갱신합니다.
- 'Radio' : 이전 재생 중이던 Mode를 중지한 후 최근 설정한 라디오 주파수로 라디오 청취되도록 설정합니다.
- 'Record' : 이전 재생되었던 Mode를 중지한 후 최근 녹음한 파일로 DB 설정합니다.
- 'MobileTV' : DMB가 지원되는 기기에서만 지원되는 Mode이며, 이전 재생되었던 Mode를 중지한 후 최근 시청한 DMB 채널을 시청되도록 설정합니다.
- 'Flash' : 이전 재생되었던 Mode를 중지한 후 최근 재생한 Flash 파일로 DB 설정합니다.
- 'Text' : 멀티 Mode으로 동작되며 Background로 이전 Mode를 유지한 채 최근에 읽은 Text 파일로 DB 설정합니다.
- 'Picture' : 멀티 Mode으로 동작되며 Background로 이전 Mode를 유지한 채 최근에 본 Picture 파일로 DB 설정합니다.
- 'Dictionary' : 멀티 Mode으로 동작되며 Background로 이전 Mode를 유지한 채 최근에 본 단어로 전자사전을 설정합니다.

3). 각 Mode에 따른 FS Command

FS Command는 '표 4 – 3'와 같은 구조로 구현되어 있습니다.

Key	Common Mode, Music Mode, Video Mode, FM Radio Mode, Mobile TV Mode, Record Mode
Get Parameter	JetEffect 3.0, Display, Timer, System, Music Mode, Video Mode, Record Mode, FM Radio Mode, Mobile TV Mode, Bluetooth, Etc
Set Parameter	JetEffect 3.0, Display, Timer, System, Music Mode, Video Mode, Record Mode, FM Radio Mode, Mobile TV Mode, Bluetooth, Etc
Etc	Browser, Mode, Text, User Data

'표 4 – 3'

FS Command 중 Key 관련 FS Command들이 각 Mode별로 구현되어 있습니다. 그런데 기기 동작의 안정성을 위해 'Music' Mode로 동작 중 'Video', 'Radio' 등의 다른 Mode의 FS Command가 입력이 되면 에러로 처리됩니다. 즉 현재 설정된 Mode와 다른 Mode의 Key FS Command 입력이 되면 'Low Level 계층'에서는 Key 관련 처리를 하지 않으며, ActionScript에 에러를 반환합니다. 자세한 FS Command 설명은 별도의 FS Command 문서를 참조하시기 바랍니다.

4). UI를 구현하다보면 'Low Level 계층'을 직접 제어할 필요는 없지만 저장이 필요한 Parameter가 있습니다.

브라우저 'Flip' 기능 같은 경우를 말합니다. 이런 경우 아래와 같은 방법으로 ActionScript에서 필요한 Parameter를 저장할 수 있습니다. 여기서는 개요만 언급합니다. 자세한 것은 FS Command 문서에서 참조하시기 바랍니다.

- 각 Mode 별 할당되어 있는 저장 공간을 사용하는 방법 : 'GetEtcUIConfig', 'SetEtcUIConfig' FS Command로 구현되어 있습니다.
- ActionScript 개발자에게 개방된 저장 공간을 사용하는 방법 : 'EtcUsrGetNumber', 'EtcUsrSetNumber', 'EtcUsrGetString', 'EtcUsrSetString' FS Command로 구현되어 있습니다. 다만 사용자가 수시로 UCI를 바꿀 수 있기 때문에 이 FS Command를 사용하기 위해선 별도의 간단한 Header를 붙여 고유한 정보를 확인해야 합니다. 또한 사용자가 한 개발자가 만든 UCI를 사용하는 것이 아닌, 여러 버전의 UCI를 사용할 경우 이 FS Command 값을 보장할 수 없습니다. 따라서 가급적 각 Mode 별 할당된 저장 공간을 사용하는 것을 권장합니다.
- SharedObject를 사용하여 저장 공간을 사용하는 방법

5. 입력 장치

1). 상단 Key

C2의 상단 Key는 'Power/Hold', 'M', '–', '+' Key가 배치되어 있습니다. 'Power/Hold' Key 동작은 아래와 같습니다.

- 'Hold' Key : Key Code는 145(ScrLk)입니다. 'Hold' Key 반복 동작을 할 필요가 없기 때문에 Hold On 및 Off 동작 시 한 번만 순차적으로 Key Down Event, Key Up Event가 발생됩니다. 만약 'Hold' Key Event가 발생되면 플래시 개발자는 'GetSysHoldKey'와 'GetSysCtrlHoldState' 값을 읽어 각 상태에 맞는 GUI를 표시하면 됩니다.

– 'Power' Key : Low Level 계층에서 처리하기 때문에 Key Event가 발생되지 않습니다.

'M', '–', '+' Key 동작은 반복적으로 이벤트를 발생하도록 구현되어 있습니다. 키를 누르고 있으면 35ms마다 Key Down Event가 반복적으로 발생됩니다. 반대로 사용자가 누르고 있던 키를 떼면 Key Up Event가 발생됩니다. Flash 엔진에서 Action Script로 전달되는 Key Code는 아래와 같습니다.

– 'M' Key : 32

– '+' Key : 40

– '–' Key : 38

2). Touch

Flash에서는 Touch 이벤트를 처리하는 기능이 별도로 없기 때문에 C2에서 터치가 눌리면 Flash Engine 계층에서는 Mouse Event로 대체되어 이벤트를 발생시켜 처리합니다. Touch 동작별 처리는 아래와 같습니다.

– 사용자가 LCD On 상태에서 Touch를 할 경우 : Mouse Up Move Event를 먼저 발생한 다음 Mouse Down Event가 발생됩니다. 이후부터는 한 Frame마다 Mouse Down Move Event가 발생됩니다. 사용자가 터치한 위치는 Mouse X, Y 값을 통해서 알 수 있습니다.

– 사용자가 LCD On 상태에서 Touch를 뗄 경우 : Mouse Down Move Event가 발생한 다음 Mouse Up Event가 발생됩니다.

3). 가상 Key

C2에서는 LCD Off가 된 상태에서는 Flash Engine이 동작하지 않습니다. 따라서 LCD Off 상태에서 On 상태로 변경되면 Off된 상태의 GUI가 남아 있는 현상이 발생합니다. 또한 음악 GUI에서 재생 중 다음 곡으로 넘어갈 때 펌웨어 내부에서 처리되기 때문에 Flash에서는 다음 곡으로 넘어갔는지 여부를 알 수 없어 GUI는 이전 곡으로 남아있는 현상이 발생합니다. 이런 현상을 해결하기 위해서 C2에서는 Flash GUI를 업데이트하기 위한 목적으로 가상 Key인 'F12' Key를 정의하여 사용합니다. 따라서 Action Script에서는 'F12' Key가 발생하면 GUI를 갱신하면 됩니다. 'F12' Key가 발생하는 상황은 아래와 같습니다.

– LCD Off 상태에서 On 상태로 변경시

– 음악이나 비디오에서 현재 재생중인 컨텐츠가 완료되어 다음 컨텐츠 재생으로 변경시

– 음악이나 비디오에서 현재 재생중인 컨텐츠가 완료되어 'Stop' 상태로 변경시

– Bluetooth Key가 발생되어 기기가 'Play', 'Stop' 등의 동작을 할 경우

– 기타 펌웨어에서 Flash GUI가 업데이트 되어야 하는 상황이 발생 할 경우

6. Flash Player

C2의 Flash 재생은 브라우저에서 Flash 컨텐츠 선택시 'Low Level 계층'에서 재생 Flag를 설정한 후 해당 Flash GUI Frame이 끝날 때 Flash Engine에서는 별도의 Flash Player를 생성하여 선택된 Flash 컨텐츠를 재생합니다. Flash Player는 일반 Flash UI와 아래와 같은 차이가 있습니다.

1). Display Resolution : 320 * 240

2). Sound : Mp3 및 Stream Mp3 지원

3). Memory : 5MB

4). 'M' Key : 'Low Level 계층'에서 Flash Player를 종료한 후 Flash Engine으로 전환 기능

5). '+', '–' Key : 볼륨 + – Key로 동작합니다.

7. 앨범아트, JPEG Thumbnail 등의 가상 이미지 Load

C2에서는 Movie Clip Load(loadMovie(), loadClip() 함수 등)를 통해 가상의 파일을 Load할 수 있습니다. C2는 Tag 및 Folder DB를 사용하여 제어를 하고, 또한 파일 내부에 JPEG 파일이 존재하는 경우가 있어 Flash Spec에 없는 별도의 기능을 제공합니다.

1). Music Mode에서 앨범아트 Load

– '*.MUS', '*.MU0', '*.MU1' 확장자로 Load하면 해당 파일의 앨범아트가 Load됩니다. '*.MUS'는 앨범아트를 원본 이미지 크기로 Load하는 것이며, '*.MU0'는 앨범아트 이미지의 크기가 'SetAudAlbumArtMCSize' FS Command를 통해 미리 설정된 Pixel 값보다 크다면, 가로 혹은 세로 중의 최대 크기가 해당 Pixel 값에 맞도록 축소하여 Load합니다. '*.MU1'은 앨범아트에 Thumbnail 이미지가 있을 경우 Thumbnail이 Load되지만 Thumbnail이 없을 경우 일반 이미지가 Load됩니다. 이 때 이미지의 크기가 150 Pixel보다 크다면, 가로 혹은 세로 중의 최대 크기가 150 Pixel에 맞도록 축소하여 Load합니다. Load Movie 및 Flash에서 이미지 처리하는 속도는 '*.MU1' 〉 '*.MU0' 〉 '*.MUS' 순으로 빠릅니다.

예) loadClip("0.MUS" ,"MC_AlbumArt")

– 앞에 붙는 숫자는 앨범아트를 Load할 파일의 인덱스 입니다. 즉 재생 목록에 10개의 파일이 있을 경우 Load할 수 있는 인덱스는 '0 ~ 9' 까지 입니다.

– 재생 목록의 총 파일 개수는 'GetEtcTotalPLNum' FS Command를 통해서 알 수 있습니다.

– 음원 파일이 멀티 앨범아트를 지원할 경우 'GetAudAlbumArtTotalNum'와 'SetAudAlbumArtIndex'를 통해 최대 6장까지 멀티 앨범아트를 Load할 수 있습니다. 만약 파일에는 앨범아트가 3개 밖에 없는데도 불구하고 4번째 앨범아트를 Load할 경우 펌웨어 내부에서 첫번째 앨범아트를 Load하도록 처리됩니다.

2). Album Browser에서 앨범아트 Load

– 현재 Mode가 음악인 상태에서 Browser 초기화를 'Album'으로 했을 경우에만 정상 동작합니다.

– '*.AB0', '*.AB1' 확장자로 Load하면 해당 앨범의 앨범아트가 Load됩니다. '*.AB0'는 앨범아트 이미지의 크기가 'SetAudAlbumBMCSize' FS Command를 통해 미리 설정된 Pixel 값보다 크다면, 가로 혹은 세로 중의 최대 크기가 해당 Pixel 값에 맞도록 축소하여 Load합니다. '*.AB1'은 해당 앨범의 앨범아트에 Thumbnail 이미지가 있을 경우 Thumbnail이 Load되지만 Thumbnail이 없을 경우 일반 이미지가 Load됩니다. 이 때 이미지의 크기가 150 Pixel보다 크다면, 가로 혹은 세로 중의 최대 크기가 150 Pixel에 맞도록 축소하여 Load합니다.

예) loadClip("0.AB0" ,"MC_BR_AlbumArt")

– 앞에 붙는 숫자는 앨범아트를 Load할 앨범의 인덱스 입니다. 즉 브라우저 앨범 갯수가 10개일 경우 Load할 수 있는 인덱스는 '0 ~ 9' 까지 입니다.

– 브라우저의 앨범 개수는 브라우저 FS COmmand인 'EtcBrwSetInitialization', 'EtcBrwSetPrevStage' 호출 시에 알 수 있습니다.

3). Video Mode에서 Preview Load

– '*.VID' 확장자로 Load하면 해당 파일의 Preview가 Load됩니다.

예) loadClip("0.VID" ,"MC_Preview")

– 앞에 붙는 숫자는 Preview를 Load할 파일의 인덱스 입니다. 즉 재생 목록에 10개의 파일이 있을 경우 Load할 수 있는 인덱스는 '0 ~ 9' 까지 입니다.

– 재생 목록의 총 파일 개수는 'GetEtcTotalPLNum' FS Command를 통해서 알 수 있습니다.

4). Video Mode에서 Story Board Load

– '*.STR' 확장자로 Load하면 해당 파일의 Story Board가 Load됩니다.

예) loadClip("0_11.STR" ,"MC_Story")

– 앞에 붙는 숫자는 Story Board 파일의 인덱스이며 뒤에 붙는 숫자는 그 파일의 Story Board의 인덱스 입니다. 즉 '2_10.STR' 파일을 Load할 경우 재생 목록의 2번째 파일에서 10번째 Story Board 내용을 Load하라는 뜻입니다.

– 재생 목록의 총 파일 개수는 'GetEtcTotalPLNum' FS Command를 통해서 알 수 있습니다.

한 파일에 15개(0 ~ 14)의 Story Borad 이미지가 존재합니다.

5). Picture Mode(Picture Viewer)에서 이미지 파일 Load

– '*.PM0', '*.PM1' 확장자로 Load하면 해당 인덱스의 파일이 Load됩니다. '*.PM0'은 일반 사진을 Load하는 것이며, '*.PM1'은 JPEG Thumbnail을 Load합니다. JPEG Thumbnail은 사진 Load 속도를 빨리 할 때 사용하면 됩니다. Thumbnail로 Load할 때 이미지의 크기가 150 Pixel보다 크다면, 가로 혹은 세로 중의 최대 크기가 150 Pixel에 맞도록 축소하여 Load합니다.

예) loadClip("11.PM0" ,"MC_Viewer")

– 앞에 붙는 숫자는 Load할 이미지 파일의 인덱스 입니다. 즉 이미지 목록에 10개의 파일이 있을 경우 Load할 수 있는 인덱스는 '0 ~ 9' 까지 입니다.

– 이미지 목록의 총 파일 개수는 'GetEtcTotalPLNum' FS Command를 통해서 알 수 있습니다.

6). Picture Mode(Picture Browser)에서 이미지 파일 Load

– '*.PB0', '*.PB1' 확장자로 Load하면 해당 인덱스의 파일이 Load됩니다. '*.PB0'은 일반 사진을 Load하는 것이며, '*.PB1'은 JPEG Thumbnail을 Load합니다. 사진 Browser에서는 여러 파일이 Load되기 때문에 Thumbnail로 Load하는 것을 권장합니다. Thumbnail로 Load할 때 이미지의 크기가 150 Pixel보다 크다면, 가로 혹은 세로 중의 최대 크기가 150 Pixel에 맞도록 축소하여 Load합니다.

예) loadClip("11.PB0" ,"MC_PBrowser")

– 앞에 붙는 숫자는 Load할 브라우저상의 리스트 인덱스입니다. 즉 리스트에 10개의 파일이 있을 경우 Load할 수 있는 인덱스는 '0 ~ 9' 까지 입니다.

- 브라우저의 리스트의 총 개수는 'EtcBrwSetInitialization', 'EtcBrwSetNextStage', 'EtcBrwSetPrevStage' 등 브라우저 관련된 FS Command를 사용하면 알 수 있습니다.

7). Notepad 파일 Load

- '*.NPD' 확장자로 Load하면 해당 인덱스의 Notepad 파일이 Load됩니다. 자세한 것은 Notepad FS Command를 참고하시기 바랍니다.

  예) loadClip("1.NPD" ,"MC_Notepad")
- 앞에 붙는 숫자는 Load할 Notepad 파일의 리스트 인덱스입니다.

8). 임시 Notepad 이미지 데이터 Load

- '*.NPM' 확장자로 Load하면 해당 인덱스의 임시 Notepad 이미지 데이터가 Load됩니다. 자세한 것은 Notepad FS Command를 참고하시기 바랍니다.

  예) loadClip("0.NPM" ,"MC_MemoryNotepad")
- 앞에 붙는 숫자는 Load할 임시 Notepad 이미지 데이터의 리스트 인덱스입니다.

9). 바탕화면 Load

- 'WALLPAPER.BNG' 이름으로 Load하면 바탕화면 이미지 데이터가 Load 됩니다. Load된 이미지 데이터는 가로 320 Pixel, 세로 240 Pixel 입니다. 바탕 화면 이미지 데이터가 없거나 바탕 화면 설정이 'Off'('SetDisWallpaper' FS Command를 통해)로 되어 있을 경우 Load되지 않습니다.

  예) loadClip("WALLPAPER.BNG" ,"MC_Wallpaper")