NextGen TV Run3TV Implementation Guidelines: RSS Feed Specification

Version: 1.2 (Release)

Date: Dec 11th, 2023

Doc: R3TV-IG-0231

Revision history

Version	Date	Update
1.0 (Beta Release)	13 March 2023	Initial draft
1.0 (Beta Release)	1st June 2023	Moved to a different specification version.
1.0 (Beta Release)	15th June 2023	Added support for launching the RUN3TV Settings
1.1 (Released)	30th Aug 2023	Supported native JSON RSS format
1.2 (Released)	11 December 2023	Added support to disable/enable playback controls and content looping

Copyright notice

Contents

1. Glossary

Glossary of unfamiliar words and acronyms.

Term	Definition
ESG	Electronic Schedule Guide: A schedule of broadcast events transmitted as part of the ATSC 3.0 standard
MRSS	Media RSS: An extension of the RSS standard that allows the inclusion of rich media metadata within an RSS feed
RSS	Really Simple Syndication: An XML formatted feed of content items. In the context of this document, RSS files contain one or more items of content that will be presented to the viewer by the Framework

2. References

ID	Publisher	Document
[RUN3TV-IG-0204]	A3FA	Run3TV Application Management
[RUN3TV-IG-0232]	A3FA	AEAT and OSN Feed Specification
[RUN3TV-IG-0233]	A3FA	ESG (OMA) Feed Specification
[RUN3TV-IG-0221]	A3FA	Q-Bar Application
[RUN3TV-IG-0201]	A3FA	IOP for Receivers

3. Introduction

The Run3TV framework can accept RSS feeds to drive viewer-facing application functionality. For example, rails containing links to VOD content, Live TV or advertisements can be dynamically updated using RSS.

For the complete list of content types that can be offered to viewers, see the Content Items section.

For a summary of how to signal these Content Item types, see the media:Content Content types section.

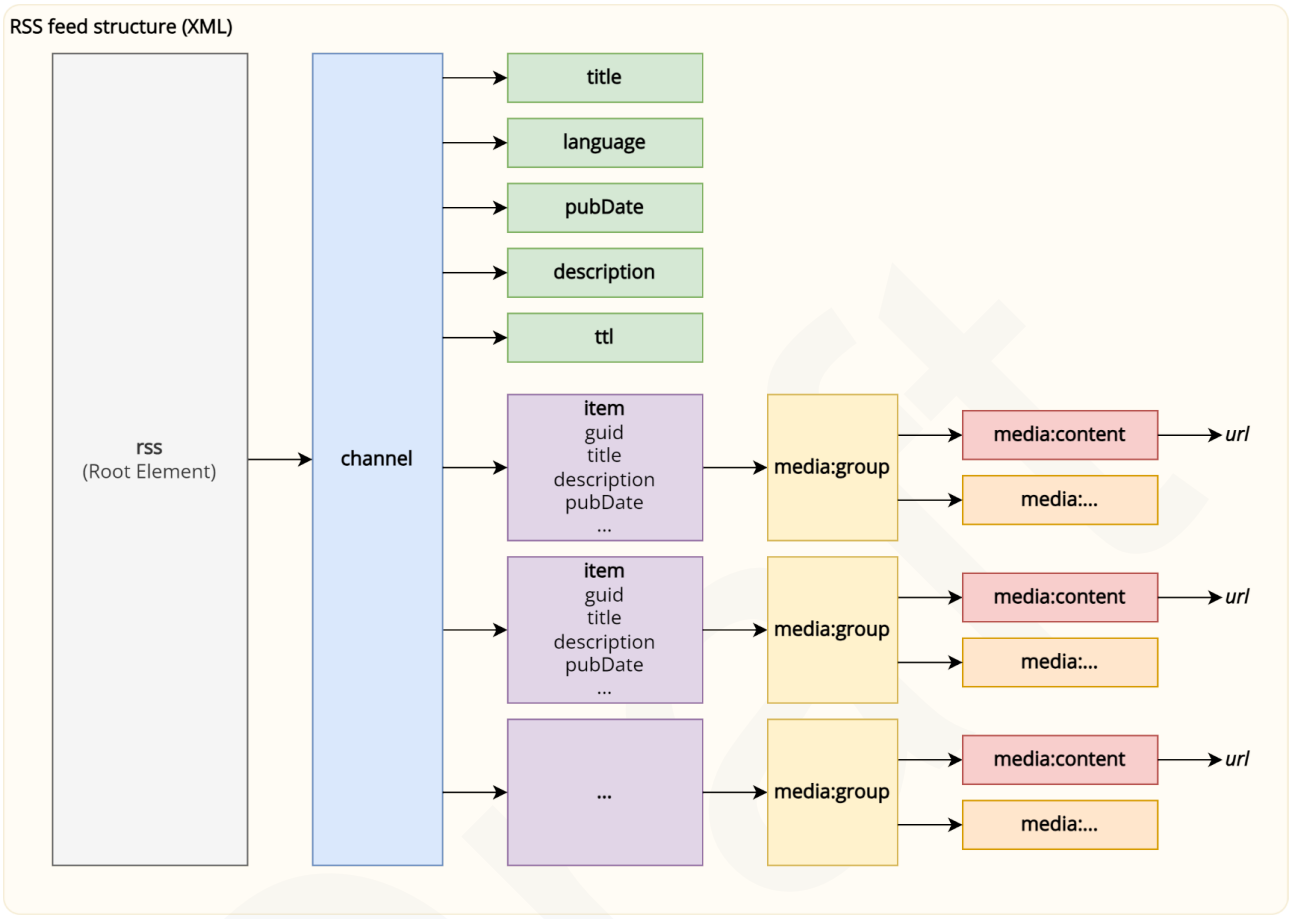
This document describes the Run3TV framework's RSS feed profile.

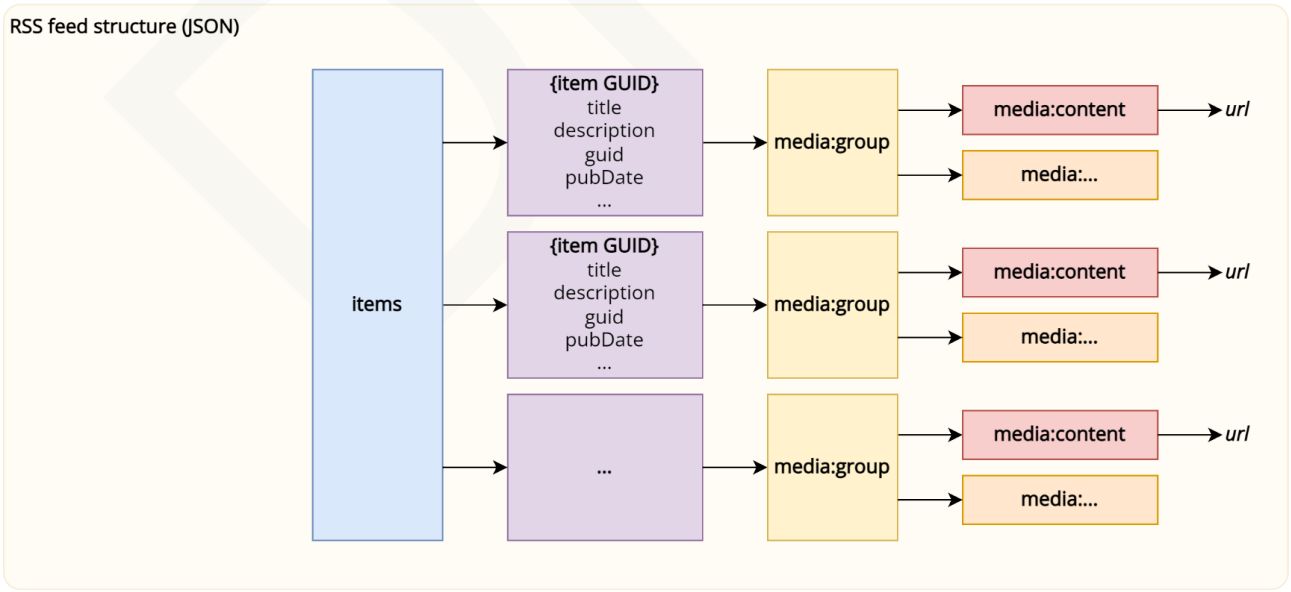
Feeds can be supplied either as .atom files (XML format) or .json (JSON format) files. Both data formats share predominantly the same data structures. Throughout this document, data structures are referred to in XML format, however examples of both XML and JSON are provided.

The Framework accepts feeds that are well formed to the following formats:

RSS Feed Type	Defined by	<Media:...> Definitions Defined by
Atom XML v1.0	https://datatracker.ietf.org/doc/html/rfc4287	https://www.rssboard.org/media-rss
JSON Feed v1.1	https://www.jsonfeed.org/	

The general structure of the RSS feed data in the two formats (XML and JSON) is shown below:





The two structures are near-identical, with the key exceptions of the root node and channel-level description.

A simple one-item feed containing the equivalent data is shown below in both XML and JSON formats. The equivalent lines are aligned to provide context.

XML Format

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:atom="http://www.w3.org/2005/Atom"
     xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
    <channel>
        <title>Test streams</title>
        <language>en-us</language>
        <pubDate>Mon, 02 Apr 2018 16:19:56 -0700</pubDate>
        <description>Contains short videos.</description>
        <ttl>40</ttl>
        <item>
            <guid>3c60712e36d549329f49fd5c6db16191</guid>
            <title>Big Buck Bunny MPEGTS</title>
            <description>An HLS stream.</description>
            <pubDate>Mon, 01 Jul 2019 04:28:14 GMT</pubDate>
            <keywords>Local,News,Top Stories</keywords>



            <media:group>
                <media:content

                               duration="597"
                               height="720"
                               width="1280"
                               medium="video"
                               type="application/vnd.apple.mpegurl"
                               url="https://example.com/file.m3u8"/>


                <media:thumbnail

                       url="https://example.com/thumb.jpg"/>

            </media:group>
        </item>
    </channel>
</rss>
```

JSON format

```json
{
    "items": [
        {
            "guid": "3c60712e36d549329f49fd5c6db16191",
            "title": "Big Buck Bunny MPEGTS",
            "description": "An HLS stream.",
            "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
            "category": [
                "Local",
                "News",
                "Top Stories"
            ],
            "media:group": {
                "media:content": [
                    {
                        "duration": "597",
                        "height": "720",
                        "width": "1280",
                        "medium": "video",
                        "type": "application/vnd.apple.mpegurl",
                        "url": "https://example.com/file.m3u8"
                    }
                ],
                "media:thumbnail": [
                    {
                        "url": "https://example.com/thumb.jpg"
                    }
                ]
            }
        }
    ]
}
```

These identical feeds contain a reference to a single video item with the title “Big Bug Bunny MPEGTS” and an associated thumbnail.

3.1. Referencing feeds in the framework

> ***Note*** This section provides a high level introduction to referencing feeds. For more details, see [RUN3TV-IG-0204] and the documents referenced below.

Within the config.json file of your Q-Bar application, the `feeds` key can be populated with one or more feed objects.

The permitted Feed Types are listed below.

Feed Type	Description	Defined by	Profiled by
`mrss`	XML-formatted RSS feeds containing one or more items of content	https://datatracker.ietf.org/doc/html/rfc4287	**This document**
`mrss-json`	JSON-formatted RSS feeds containing one or more items of content	https://www.jsonfeed.org/	**This document**
`esg`	Electronic Schedule Guide data, sourced from the receiver's ESG database	[RUN3TV-IG-0233]	

3.2. Content Items

The A3FA framework can accept a number of different *content types* in the RSS feeds that it receives.

Different items within the same RSS feed may have the same or different content types.

The permitted types of content are listed below. For a summary on detailed implementation points see media:Content Content types and the appropriate annex for each content type.

Content Type	Description	Example
Live TV Service	Links to a live Over-The-Air TV service. For example, service changing from Univision to Unimas	ANNEX B
Live Radio Service	Links to live Over-The-Air radio service. For example, service changing from KACL to KBBL	ANNEX C
Current Live TV Program	Decorates the current TV broadcast's Now event with additional content	ANNEX D
Next program on Live TV	Decorates the current TV broadcast's Next event with additional content	ANNEX E
Current Live Radio Program	Decorates the current Radio broadcast's Now event with additional content	ANNEX D
Next Radio content live	Decorates the current Radio broadcast's Next event with additional content	ANNEX E
Live Streaming Service	Provides a live OTT Streamed service	ANNEX F
Live Radio	Provides a live Radio (audio only) Streamed service	ANNEX G
Video on Demand	Provides a live VOD service	ANNEX H
On-Demand Audio	Provides on-demand audio content (for example, a podcast)	ANNEX I
Imagery	Provides Imagery such as JPEG and GIFs	ANNEX J
Web Display (non-interactive)	Provides content from an non-interactive web page	ANNEX K
Application Links	Provides a link to another interactive application within the Run3TV Framework or to an external Smart TV application	ANNEX L
Animated Content	Provides content that is animated using the Lottie format[1]	ANNEX M
Advertisement	Provides content that is an advertisement	<media:category> Element
Promotional Video	Provides promotional and trailer content	<media:category> Element
Alerts	Provides text and/or video based alerts (TBD)	ANNEX N - Alert Content Examples

[1] For further details, see https://github.com/lottie-animation-community/docs/blob/main/Lottie_Specification.md

4. HTTP Requests

Feed data is downloaded by each ATSC 3.0 receiver as and when it requires the data. Both the scheduling of applications (using appSchedule.json as described in [RUN3TV-IG-0203]) and on-screen promotions of the application may lead to upticks in feed requests to the server. As such, it is advised to ensure that any servers providing feeds are scaled appropriately.

4.1. CORS (Cross-Origin Resource Sharing)

The method of delivering the application to the ATSC 3.0 receiver determines the browser origin of the application, and as such, the required CORS headers to be configured on the server. The table below summarizes these CORS configuration requirements.

Source of application	Application's browser origin	CORS headers
Broadband-delivered	The https domain that supplied the application	`Access-Control-Allow-Origin: [application's domain]`
Broadcast-delivered (NRT)	http://localhost or http://127.0.0.1	`Access-Control-Allow-Origin: http://localhost:* http://127.0.0.1:*`

No matter the source of the application, the following server CORS headers must also be included:

```
Access-Control-Allow-Methods: GET
Access-Control-Allow-Headers: X-PINGOTHER, Content-Type
```

4.2. Caching: ETAGs and Cache Control Headers

In order to minimize re-requesting RSS data that hasn't changed, the Framework will honor any ETAGs (Entity Tags) provided by the server in the server's HTTP response headers.

In addition, the Framework will include an `If-Modified-Since` header in any re-request for data.

Note It is encouraged to ensure that the server honors the `If-Modified-Since` header, in order to minimize the amount of data sent to the ATSC 3.0 receiver base.

Server operators should ensure that their `Cache-Control` headers are correctly configured. It is recommended that these headers be aligned with the `<ttl>` tag (where present), described later in this document.

4.3. User Agent Identifier

ATSC 3.0 receivers will present their User Agent Identifiers in the form described by [RUN3TV-IG-201].

> ***Note*** During application development, the server may receive requests from regular PC-based user agents, such as Chromium or Firefox.

5. Data Structures

This section describes and profiles the various data structure elements seen in XML and JSON RSS files to be used with the A3FA Framework.

Example snippets of data are provided for each element where at least one sub-element is required by the Framework. Where possible, complete samples are provided, with the elements under discussion highlighted in bold.

Further complete examples can be found in the Annexes at the end of this document.

Throughout this section, cardinality is noted using the form {1+} for one or more items.

5.1. <rss>

The `<rss>` tag is the root note of the XML RSS format. This is not required for JSON RSS.

It has the following attributes:

Tag / Attribute	Description	Required	Comment
xmlns:media	Provides the XML namespace	M	Always set to: `http://search.yahoo.com/mrss`
version	Provides the supported version of the RSS	M	Always set to: `2.0`
<channel>	The Channel tag contains a description of the contents of the feed and the list of items.	M	

Example	

XML	`<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">`
	`  <channel> ... </channel>`
	`</rss>`

JSON	*Not applicable*

5.2. <channel>

The `<channel>` tag provides a high level description of the feed and contains the individual feed items.

This tag is not required for JSON RSS.

Tag / Attribute	Description	Required	Comment
<title>	The title of the channel	M-Run3TV	Example: `Local News`
<link>	The URL of the HTML website corresponding to the channel.	M	
<description>	Information describing the channel	M	Example: `Everything local - updated every 15 minutes from the News Center`
<language>	Indicates the language the channel is written in using BCP-47 notation	M-Run3TV	Example: `en-us`
<copyright>		0	
<managingEditor>		0	
<webMaster>		0	
<pubDate>	Date and time of the original publication of the content (in RFC 822 format)	M-Run3TV	Example: `Sat, 07 Sep 2002 0:00:01 EST`
<lastBuildDate>	Date and time of the last time the content of the channel changed (in RFC 822 format)	M-Run3TV	Example: `Sat, 07 Sep 2002 9:42:31 EDT`
<category>		0	
<generator>		0	
<docs>		0	
<cloud>		0	
<ttl>	The number of minutes that the feed can be cached for before re-downloading it	M-Run3TV	Example: `30`
<image>		0	
<rating>		0	
<textInput>		0	
<skipHours>		0	
<hour> {1+}		0	
<skipdays>		0	
<day> {1+}		0	
<item> {1+}	The content of a specific item	M-Run3TV	

Example channel element

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <!-- ... -->
    </item>
  </channel>
</rss>
```

JSON

```json
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [ ]
}
```

5.2.1. <image>

Tag / Attribute	Description	Required	Comment
<url>	URL of the image Permitted formats are: • GIF • JPEG • PNG	M-Run3TV	Example: `https://server.com/Company_A_logo.jpg`
<title>	Defines the text to display if the image can not be shown	M-Run3TV	Example: `Sports`
<link>		0	

Example

XML

```xml
<image>
   <url>https://example.com/logo.png</url>
   <title>Company A</title>
</image>
```

JSON

<TODO>

5.2.2. <cloud>

Tag / Attribute	Description	Required	Comment
domain		0	
port		0	
path		0	
registerProcedure		0	
protocol		0	

6. <item> tag / “items” list

<item> tags are used to describe individual items of content to present to viewers.

In XML format feeds, one or more <item> tags sit within the <channel> tag.

In JSON format feeds, a single `items` array of item objects sits as the root of the file.

Tag / Attribute	Description	Required	Comment
<title>	The title of the content	M-Run3TV	
<link>	The URL of the content item	M-Run3TV	Run3TV expanded Schema:
<description>	Information describing the content	M-Run3TV	
<author>		0	
<category>		0	
<comments>		0	
<enclosure>		0	
<guid>	A string to uniquely identify the content item	M-Run3TV	Note: The Framework ignores the *isPermaLink* attribute. *isPermaLink* is always set to false. Example: `1b81c0f7ac70941e75bf17a429db3ae737a443bc`
<pubDate>	RFC 822 date and time of the publication of the content	O-Run3TV	Sat, 07 Sep 2002 0:00:01 +0000
<source>		0	
<media:group>	Allows grouping of <media:content> effectively of the same content, yet different representations	M-Run3TV	
<media:content>	Allows media that is not the same content format	M-Run3TV	
<thumbnail>		0	

Example: An rss file containing a single `item`

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Big Buck Bunny with MPEGTS based segments</title>
      <link></link>
      <guid>d093e73c-67cc-41b5-a072-4d640ff7a03</guid>
      <description>A stream using HLS.</description>
      <pubDate>Mon, 01 Jul 2019 04:28:14 GMT</pubDate>
      <media:group>
        <media:content>
          <!-- Media content details -->
        </media:content>
        <media:thumbnail>
          <!-- Media thumbnail details -->
        </media:thumbnail>
      </media:group>
    </item>
  </channel>
</rss>
```

JSON

```json
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Big Buck Bunny with MPEGTS based segments",
      "link": "",
      "description": "Description text",
      "guid": "d093e73c-67cc-41b5-a072-4d640ff7a03f",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            // Media content details
          }
        ],
        "media:thumbnail": [
          {
            // Medial thumbnail details
          }
        ]
      }
    }
  ]
}
```

6.1.1. <enclosure>

Tag / Attribute	Description	Required	Comment
url		0	
length		0	
type		0	

6.1.2. <media:content>

This tag is used to define the content available to be viewed or listened to when the viewer chooses to access the item. This tag may be placed within an <item> or within a <media:group> tag.

Tag / Attribute	Description	Required	Comment
@url	The URL of the media	M-Run3TV	
@filesize		0	
@type	Standard MIME types	M-Run3TV	Accepted MIME types for each type of content are listed in media:Content Content types For further details on `run3tv/remote`, `run3tv/local` and `run3tv/frwk`, see [RUN3TV-IG-0221]
@medium	Describes the type of object; • `image` • `audio` • `video` • `document` • `executable`	M-Run3TV	Informs the Framework of the type of content available via the supplied @url value. For further details, see Media:Content Content types
@isdefault		0	
@expression	Informs if the media is one of the following; • sample • full • nonstop • ota • loop	M-Run3TV	Informs the Framework of the type of content available via the supplied @url value. For further details, see Media:Content Content types
@bitrate		0	
@framerate		0	
@samplingrate		0	
@channels		0	
@duration	The number of seconds the media object plays. For a live stream, this should be omitted	M-Run3TV	Informs the Framework of the duration of the content. For further details, see Media:Content Content types
@height	The height of the content (in pixels), where appropriate	0	
@width	The height of the content (in pixels), where appropriate	0	
@lang	The primary RFC 3066 language of the media object	O-Run3TV	This will override the language value of the <channel> object, if provided

Example: A complete rss file containing a media:content entry within a media:group

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Big Buck Bunny with MPEGTS based segments</title>
      <link></link>
      <guid>d093e73c-67cc-41b5-a072-4d640ff7a03</guid>
      <description>A stream using HLS.</description>
      <pubDate>Mon, 01 Jul 2019 04:28:14 GMT</pubDate>
      <media:group>
        <media:content
          duration="185"
          height="720"
          width="1280"
          type="video/mp2t"
          url="https://example.com/master.m3u8" />
      </media:group>
    </item>
  </channel>
</rss>
```

JSON

```json
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "guid": "d093e73c-67cc-41b5-a072-4d640ff7a03f",
      "title": "Big Buck Bunny with MPEGTS based segments",
      "description": "A stream using HLS.",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:group": {
          "media:content": [
            {
              "duration": "597",
              "height": "720",
              "width": "1280",
              "type": "video/mp2t",
              "url": "https://example.com/master.m3u8"
            }
          ]
        }
      }
    }
  ]
}
```

6.1.3. media:Content Content types

The Framework can link to various types of content supplied within the `<media:content>` tag.

In order for the Framework to correctly present the content, it must be informed of the content type using the following fields within the `<media:content>` tag:

- `expression`
- `medium`
- `type`

Additionally, the presence or absence of other tags (such as the `<media:content>` tag's `duration` and the `<item>` tag's `pubDate` field are used to further inform the Framework of the content to be presented.

The table below provides a summary of these types, along with the required additional `media:content` (and other) fields as applicable.

Content Type	Expression value	Medium value(s)	Permitted MIME types	URL value	Other values	Details
ATSC 3.0 broadcast TV channel	ota	video	application/dash+xml	The GSID of a TV channel	The duration must not be present. The item/pubDate must not be present	See ANNEX B - Example Live TV Service Content
ATSC 3.0 broadcast radio station		audio		The GSID of a radio station		See ANNEX C - Example Live Radio Service Content
ATSC 3.0 Now event		video		The GSID of a TV channel	duration must be non-zero. The item/pubDate must be set to the start time of the program. Therefore "Now" event start times will be in the past	See ANNEX D - Example Live TV with Now Program Information
ATSC 3.0 Next event		video		The GSID of a TV channel	duration must be non-zero. The item/pubDate must be set to the start time of the program. Therefore "Next" event start times will be in the future	See ANNEX E - Example Live TV with Next Program Information
Live OTT TV stream	nonstop	video	application/dash+xml application/vnd.apple.mpegurl video/mp2t video/mp4	The location of a manifest file	duration must be non-zero	See ANNEX F - Example Live TV Streaming Content
Live OTT Radio stream	nonstop	audio	application/dash+xml application/vnd.apple.mpegurl	The location of a manifest file	duration must be non-zero	See ANNEX G - Example Live Radio Streaming Content
Video On Demand (VOD) asset	full or loop	video	application/vnd.apple.mpegurl application/dash+xml audio/mp4 audio/mp3	The location of a manifest file	–	See ANNEX H - Example VOD Content
Audio On Demand (AOD) asset	full or loop	audio	application/vnd.apple.mpegurl application/dash+xml audio/mp4 audio/mp3	The location of a manifest file	–	See ANNEX I - Example Audio On Demand Content
Image	*not used*	image	image/jpeg image/gif image/png	The location of an image	–	See ANNEX J - Example Image Content

Content Type	Expression value	Medium value(s)	Permitted MIME types	URL value	Other values	Details
Website	*not used*	`document`	`text/html`	The location of a web page	–	See ANNEX K - Web Display Content
Local Application	*not used*	`document`	`run3tv/local`	An appName (as defined in the service's appsList.json file)	–	See ANNEX L - Application Linking Content
Remote Application	*not used*	`document`	`run3tv/remote`			
Manufacturer-specific Smart TV Application	*not used*	`executable`	`run3tv/<manufacturer>`	A manufacturer-specific application name	–	See ANNEX L - Application Linking Content
Animation	*not used*	`document`	`application/json+lottie`	The location of a lottie animation file	–	See ANNEX M - Animation Content
Alert / Advertisement / Promotional Video	*Any Permitted*	*Any Permitted*	*Any Permitted*	*Any Permitted*	`item/media:category` must be set correctly	See ANNEX N - Alert Content Examples
Settings	*not used*	*not used*	`run3tv/frwk`	`preference settings` These may include additional attributes	–	See ANNEX O - Settings

6.1.4. Looping and autoplaynext

It is possible to configure VOD and AOD assets (including VOD and AOD assets that are Alerts, Advertisements or Promotional Videos) to play in a loop. This can be achieved in one of two ways:

- By setting the `<media:content>` `expression` value to be "`loop`".
- By including a `<media:embed>` element with a `loopPlayback` parameter set to `true`.

To avoid an undefined state, if the `<media:content>` `expression` value to be "`loop`", this cannot be overridden using the `<media:embed>` `loopPlayback` parameter.

> ***Note*** RSS feeds can be configured in the **config.json** file to automatically play the next piece of content when the current piece of content completes playing. This is achieved using the `autoplaynext` field. Content set to loop will never complete playback (until the viewer stops playback of this looping content manually).

6.2. Optional Tags within <Item>

The following optional elements may appear as sub-elements of `<channel>`, `<item>`, `<media:content>` and/or `<media:group>`. If the element is provided higher in the XML/JSON stack then it applies to all media below that level.

6.2.1. <media:rating>

Tag / Attribute	Description	Required	Comment
scheme	Provides the scheme	M-Run3TV	The ratings scheme to use. One of: • `urn:simple` • `urn:icra` • `urn:mpaa` • `urn:v-chip` If this attribute is omitted, the default scheme `urn:simple` is used
<text> / rating	The rating of the object(s) contained in the same elements.	M-Run3TV	A valid rating value, as defined by the ratings scheme selected by the `scheme` attribute

Example: A complete rss file containing a media:rating entry within a media:group

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Big Buck Bunny with MPEGTS based segments</title>
      <link></link>
      <guid>d093e73c-67cc-41b5-a072-4d640ff7a03</guid>
      <description>A stream using HLS.</description>
      <pubDate>Mon, 01 Jul 2019 04:28:14 GMT</pubDate>
      <media:group>
        <media:content duration="185" height="720" width="1280"
            type="video/mp2t" url="https://example.com/master.m3u8"/>
        <media:rating scheme="urn:v-chip">tv-y7-fv</media:rating>
      </media:group>
    </item>
  </channel>
</rss>
```

JSON

```json
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "guid": "d093e73c-67cc-41b5-a072-4d640ff7a03f",
      "title": "Big Buck Bunny with MPEGTS based segments",
      "description": "A stream using HLS.",
      "category": ["Local", "News", "Top Stories"],
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:group": {
          "media:content": [
            {
              "duration": "597",
              "height": "720",
              "width": "1280",
              "type": "video/mp2t",
              "url": "https://example.com/master.m3u8"
            }
          ]
        },
        "media:rating": [
          {
            "scheme": "urn:v-chip",
            "rating": "tv-y7-fv"
          }
        ]
      }
    }
  ]
}
```

Only one `<media:rating>` element is permitted per `<media:group>`.

A number of rating schemes are permitted, as summarized below. Please refer to the appropriate ratings system documentation for details of permitted values.

Rating	XML example	JSON example
Simple ratings	`<media:rating scheme="urn:simple">adult</media:rating>`	`"media:rating": [ { "scheme": "urn:simple", "rating": "adult" } ]`
ICRA Grades	`<media:rating scheme="urn:icra" >r (cz 1 lz 1 nz 1 oz 1 vz 1)</media:rating>`	`"media:rating": [ { "scheme": "urn:icra", "rating": "r (cz 1 lz 1 nz 1 oz 1 vz 1)" } ]`
MPAA ratings	`<media:rating scheme="urn:mpaa">pg</media:rating>`	`"media:rating": [ { "scheme": "urn:mpaa", "rating": "pg" } ]`
V-Chip ratings	`<media:rating scheme="urn:v-chip">tv-y7-fv</media:rating>`	`"media:rating": [ { "scheme": "urn:v-chip", "rating": "tv-y7-fv" } ]`

6.2.2. <media:title>

This tag will override the `<item>` tag's `<title>` attribute.

Tag / Attribute	Description	Required	Comment
type	Provides the type title; • plain (default) • html	M-Run3TV	HTML must be entity-encoded

Example: A complete rss file containing a media:title entry within a media:group

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Big Buck Bunny with MPEGTS based segments</title>
      <link></link>
      <guid>d093e73c-67cc-41b5-a072-4d640ff7a03</guid>
      <description>A stream using HLS.</description>
      <pubDate>Mon, 01 Jul 2019 04:28:14 GMT</pubDate>
      <media:group>
        <media:content
          duration="185"
          height="720"
          width="1280"
          type="video/mp2t"
          url="https://example.com/master.m3u8" />
        <media:title type="plain">Big Buck Bunny</media:title>
      </media:group>
    </item>
  </channel>
</rss>
```

Example: A complete rss file containing a media:title entry within a media:group

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Big Buck Bunny with MPEGTS based segments",
      "link": "",
      "description": "A stream using HLS.",
      "guid": "d093e73c-67cc-41b5-a072-4d640ff7a03f",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:group": {
          "media:content": [
            {
              "duration": "597",
              "height": "720",
              "width": "1280",
              "type": "video/mp2t",
              "url": "https://example.com/master.m3u8"
            }
          ],
          "media:title": [
            {
              "type": "plain",
              "title": "Big Buck Bunny"
            }
          ]
        }
      }
    }
  ]
}
```

6.2.3. <media:description>

Tag / Attribute	Description	Required	Comment
type	Provides the type of description; • plain (default) • html	M-Run3TV	HTML must be entity-encoded

Example: A complete rss file containing a media:description entry within a media:group

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Big Buck Bunny with MPEGTS based segments</title>
      <link></link>
      <guid>d093e73c-67cc-41b5-a072-4d640ff7a03</guid>
      <description>A stream using HLS.</description>
      <pubDate>Mon, 01 Jul 2019 04:28:14 GMT</pubDate>
      <media:group>
        <media:content
          duration="185"
          height="720"
          width="1280"
          type="video/mp2t"
          url="https://example.com/master.m3u8" />
        <media:description type="plain">Big Buck Bunny goes on an adventure!</media:description>
      </media:group>
    </item>
  </channel>
</rss>
```

Example: A complete rss file containing a media:description entry within a media:group

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Big Buck Bunny with MPEGTS based segments",
      "link": "",
      "description": "A stream using HLS.",
      "guid": "d093e73c-67cc-41b5-a072-4d640ff7a03f",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:group": {
          "media:content": [
            {
              "duration": "597",
              "height": "720",
              "width": "1280",
              "type": "video/mp2t",
              "url": "https://example.com/master.m3u8"
            }
          ],
          "media:description": [
            {
              "type": "plain",
              "title": "Big Buck Bunny goes on an adventure!"
            }
          ]
        }
      }
    }
  ]
}
```

6.2.4. <media:keywords>

Tag / Attribute	Description	Required	Comment
	Provides a maximum of 10 keywords which are comma-delimited.	0	

6.2.5. <media:thumbnail>

Tag / Attribute	Description	Required	Comment
url	Defines the URL of the thumbnail.	M-Run3TV	
isDefault		0	
width		0	
height		0	
time		0	

Example: A complete rss file containing a media:thumbnail entry

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Big Buck Bunny with MPEGTS based segments</title>
      <link></link>
      <guid>d093e73c-67cc-41b5-a072-4d640ff7a03</guid>
      <description>A stream using HLS.</description>
      <pubDate>Mon, 01 Jul 2019 04:28:14 GMT</pubDate>
      <media:group>
        <media:content
          duration="185"
          height="720"
          width="1280"
          type="video/mp2t"
          url="https://example.com/master.m3u8" />
        <media:thumbnail url="https://example.com/thumb1.jpg"/>
      </media:group>
    </item>
  </channel>
</rss>
```

Example: A complete rss file containing a media:thumbnail entry

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Big Buck Bunny with MPEGTS based segments",
      "link": "",
      "description": "A stream using HLS.",
      "guid": "d093e73c-67cc-41b5-a072-4d640ff7a03f",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:group": {
          "media:content": [
            {
              "duration": "597",
              "height": "720",
              "width": "1280",
              "type": "video/mp2t",
              "url": "https://example.com/master.m3u8"
            }
          ]
        },
        "media:thumbnail": [
          {
            "url": "https://example.com/thumb1.jpg"
          }
        ]
      }
    }
  ]
}
```

6.2.6. <media:category>

The `<media:category>` tag is used to provide additional context about the content to the Framework. For example, it can be used to mark the content as an alert, or a promotion.

Tag / Attribute	Description	Required	Comment
scheme	Defines the URI of the categorization scheme being used. https://run3tv.org/mrss/category_schema	M-Run3TV	The default scheme is "http://search.yahoo.com/mrss/category_schema". Added custom support for a media category to promote internal content types.
label	Human readable labels that can be displayed to end users.	M-Run3TV	Permitted values are: • `Advertisement` • `Sponsorship` • `Promotion` • `Alert`
<text> / "category"	Specifies the categories the object(s) belong to. Run3TV has altered the use of this field in the context of the schema.	M-Run3TV	Used in conjunction with the label Alert. The string is constructed as follows: `<A331-AEA-CATEGORIES>/<A331-AEA-PRIORITY>` Supported values for <A331-AEA-CATEGORIES> but may be expanded by the specification. • `ADVISORY` • `HEALTH` • `WEATHER` • `EMERGENCY` • `SCHOOL` • `COMMUNITY` • `TRANSIT` • `OTHER` Supported values for <A331-AEA-PRIORITY> but may be expanded by the specification. • 4 (Maximum Priority) • 3 (High Priority) • 2 (Moderate Priority) • 1 (Low Priority) • 0 (Minor Priority)

Example: A complete rss file containing a media:category entry

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
   <item>
      <title>Alert example</title>
      <description>An alert example</description>
      <guid>202</guid>
      <link></link>
       <media:group>
        <media:content
          medium="video"
          type="video/mp4"
          url="https://example.com/alertVid.mp4"
          duration="300"
          width="1280"
          height="720"/>
         <media:category
          scheme="https://run3tv.org/mrss/category_schema"
          label="Alert">Advisory/2</media:category>
       </media:group>
       <media:thumbnail url="/london/img/WebDisplay.png" width="300"
height="169"></media:thumbnail>
    </item>
  </channel>
</rss>
```

JSON

```json
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Alert example",
      "description": "An alert example",
      "guid": "201",
      "link" : "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "video",
            "type": "video/mp4",
            "url": "https://example.com/alertVid.mp4",
            "duration": "300",
            "width" : "1280",
            "height" : "720"
          }
        ]
      },
      "media:category": [
        {
          "scheme": "https://run3tv.org/mrss/category_schema",
          "label": "Alert",
          "category": "Advisory/2"
        }
      ]
    }
  ]
}
```

6.2.7. <media:hash>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
algo	Indicates the algorithm used in creating the hash.	0	default algorithm is MD5
<text>	The string value of the hash	0	

6.2.8. <media:player>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
url	Indicates the URL of a player that plays the media.	0	
height	The height of the player	0	
width	The width of the player	0	

6.2.9. <media:credit>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
role		0	
scheme		0	
width		0	

6.2.10. <media:copyright> Tag

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
url		0	

6.2.11. <media:text> Tag

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
type		0	
lang		0	
start		0	
end		0	

6.2.12. <media:restriction>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
relationship		0	
type		0	

6.2.13. <media:community>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
<media:starRating>		0	
<media:statistics>		0	
<media:tags>		0	

6.2.14. <media:comments>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
<media:comment>		0	

6.2.15. <media:embed>

This tag can be included to provide additional playback configuration options.

Tag / Attribute	Description	Required	Comment
url	The URL of the media player	0	As of the current version of the Framework, this value must either be omitted or be one of the following: **Value** / **Description** `run3tv://player/AMP` *(default)* — The content shall be played using the AMP (Application Media Player) Future releases may include the ability to reference the RMP (Receiver Media Player)
width	Not used	0	
height	Not used	0	
<media:param>	Parameters to configure the presentation of media playback	M-Run3TV	The complete list of parameters is detailed in the table below

Parameters

The following `media:param` options are permitted. Any other supplied values will be ignored.

name	Description	Permitted values	Default value
showPlaybackControls	Informs the Framework to show or hide the playback (trickplay) controls. By default, these controls are enabled	`true, false`	`true`
loopPlayback	Informs the Framework to start playback again once the end of the content is reached. By default, this is not enabled. **Note** Looping of content can also be enabled by setting the <media:content> expression to be “loop”. The loopPlayback parameter is provided as an alternate means to enable looping where the RSS author is unable to set the expression value to “loop”. The loopPlayback parameter will have no effect if the expression value is set to “loop”. For more details, see Looping and autoplaynext	`true, false`	`true`

Example: A complete rss file containing a media:embed entry

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Video_On_Demand</title>
      <description>A Video on Demand content</description>
      <guid>202</guid>
      <link></link>
       <media:group>
        <media:content
          medium="video"
          type="application/vnd.apple.mpegurl"
          expression="full"
          url="https://example.com/trailer.m3u8"
          duration="300"/>
        <media:embed url="run3tv://player/AMP" >
          <media:param name="showPlaybackControls">false</media:param>
          <media:param name="loopPlayback">true</media:param>
        </media:embed>
       </media:group>
       <media:thumbnail url="/london/img/VOD.png" width="300" height="169"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: A complete rss file containing a media:embed entry

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Video",
      "description": "VOD event description",
      "guid": "201",
      "link" : "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "video",
            "type": "application/vnd.apple.mpegurl",
            "expression": "full",
            "url": "https://example.com/trailer.m3u8",
            "duration": "300"
          }
        ],
        "media:embed" : [
          { "url" : "run3tv://player/AMP",
            "media:param" :
            {
              "showPlaybackControls" : false,
              "loopPlayback" : true
            }
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/VOD.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

6.2.16. <media:responses>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
<media:response>		0	

6.2.17. <media:backLinks>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
<media:backLink>		0	

6.2.18. <media:status>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
state		0	
reason		0	

6.2.19. <media:price>

If omitted then the media object is assumed free. This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
type		0	
info		0	
price		0	
currency		0	

6.2.20. <media:license>

Provides a machine-readable license. This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
type		0	
href		0	

6.2.21. <media:subTitle>

Provides a link to subtitle (caption) data for the content. This will be used in place of any subtitle (caption) data supplied in an MPD. For more details see ANNEX H - Example VOD Content.

Tag / Attribute	Description	Required	Comment
type	Provides the type of subtitle being provided.	M	The Framework currently supports `text/vtt` type subtitles.
lang	The language the subtitle/cc is being provided using RFC 3066	M	
href	The URL to the subtitle definition	M	

Example: A complete rss file containing a media:subTitle entry

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Big Buck Bunny with MPEGTS based segments</title>
      <link></link>
      <guid>d093e73c-67cc-41b5-a072-4d640ff7a03</guid>
      <description>A stream using HLS.</description>
      <pubDate>Mon, 01 Jul 2019 04:28:14 GMT</pubDate>
      <media:group>
        <media:content
          duration="185"
          height="720"
          width="1280"
          type="video/mp2t"
          url="https://example.com/master.m3u8" />
        <media:subTitle
          type="text/vtt"
          lang="en-us"
          href="http://example.com/subs/subs.vtt"/>
      </media:group>
    </item>
  </channel>
</rss>
```

Example: A complete rss file containing a media:subTitle entry

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",

  "items": [
    {
      "title": "Live TV",
      "link": "",
      "description": "Live TV",
      "guid": "d093e73c-67cc-41b5-a072-4d640ff7a03f",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "video",
            "type": "application/vnd.apple.mpegurl",
            "expression": "nonstop",
            "url": "https://example.com/playlist.m3u8",
            "duration": "0"
          }
        ],
        "media:subTitle": [
          {
            "type": "text/vtt",
            "lang": "en-us",
            "href": "http://example.com/subs/subs.vtt"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/Streaming_TV.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

6.2.22. <media:peerLink>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
type		0	
href		0	

6.2.23. <media:location>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
description		0	
start		0	
end		0	
<georss>		0	

6.2.24. <media:rights>

This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
status		0	

6.2.25. <media:scenes>

<media:scenes> can contain multiple sub-elements <media:scene>. This tag can be supplied, but will not be acted upon by the Framework.

Tag / Attribute	Description	Required	Comment
<sceneTitle>		0	
<sceneDescription>		0	
<sceneStartTime>		0	
<sceneEndTime>		0	

ANNEX A - Example RSS Channel Block

The example below describes a fully populated channel element, with a space for item elements to be added.

Example: Channel

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <!-- ... -->
    </item>
  </channel>
</rss>
```

JSON

```json
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [ ]
}
```

ANNEX B - Example Live TV Service Content

The example below shows how to add a content item that points to a Live TV service. This is achieved by setting the:

- the `expression` value of the `<media:content>` tag to “`ota`”
- the `url` value of the `<media:content>` tag to be the GSID of the service to link to
- the `medium` value value of the `<media:content>` tag to be “`video`”
- the `duration` value of the `<media:content>` tag is **not** set
- the `pubDate` value of the `item` tag is **not** set

The channel can be any other service that the broadcaster carries via ATSC 3.0 in the market. This may be on the same transmitter or on a different one.

Example: Linking to a Live TV service. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Live TV</title>
      <link></link>
      <description>Live TV Channel Description</description>
      <guid>201</guid>
      <media:group>
        <media:content
          medium="video"
          type="application/dash+xml"
          expression="ota"
          url="https://doi.org/10.5239/BCC1-49C1"/>
      </media:group>
      <media:thumbnail url="/london/img/Live_TV.png" width="300"
        height="169" attr="Live_TV"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to a Live TV service. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Live TV",
      "description": "Live TV Channel Description",
      "guid": "201",
      "media:group": {

          "media:content": [
            {
              "medium": "video",
              "type": "application/dash+xml",
              "expression": "ota",
              "url": "https://doi.org/10.5239/BCC1-49C1"
            }
          ],
        "media:thumbnail": [
          {
            "url": "https://example.com/thumb1.jpg"
          }
        ]
      }
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Tag / Attribute Required	Value Required	Comment
item/media:content/@medium	video	This informs the receiver to expect a video service (rather than, for example, a radio service)
item/media:content/@expression	ota	This informs the receiver to expect a GSID in the url attribute, and to use that url to tune to the ATSC 3.0 service as referenced by that GSID
item/media:content/@url	<globalserviceid_value> For example: https://doi.org/10.5239/BCC1-49C1	The Global Service ID of the service to tune to when the viewer selects the item of content on the rail
item/media:content/@type	application/dash+xml	This value must be set to "application/dash+xml", in order to match the data provided by the ATSC 3.0 receiver
item/media:content/@duration	*<not defined in the media:content>*	This attribute MUST NOT be defined
item/pubdate	*<not defined in the item>*	This attribute MUST NOT be defined

ANNEX C - Example Live Radio Service Content

The example shows how to add a content item that points to a Live Radio service. This is achieved by setting the:

- the `expression` value of the `<media:content>` tag to “ota”
- the `url` of the `<media:content>` tag to be the GSID of the service you wish to link to
- the `medium` value of the `<media:content` tag> to be “`audio`”
- the `duration` value of the `<media:content>` tag is **not** set
- the `pubDate` value of the `<item>` tag is **not** set

The channel can be any other service that the broadcaster carries via ATSC 3.0 in the market. This may be on the same transmitter or on a different one.

This example includes a full screen image (https://example.com/RadioFullScreenImg.jpg) that is presented on screen whilst the viewer listens to the radio station.

Example: Linking to a Live Radio service. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Live Radio</title>
      <description>Live Radio Channel Description</description>
      <guid>201</guid>
      <link></link>
      <media:group>
        <media:content
          medium="audio"
          type="application/dash+xml"
          expression="ota"
          url="https://doi.org/10.5239/BCC1-49C1"/>
      </media:group>

      <media:group>
        <media:content
          medium="image"
          type="image/jpeg"
          url="https://server.com/RadioFullScreenImg.jpg"
          width="1920"
          height="1080"/>
      </media:group>

      <media:thumbnail url="/london/img/Live_Radio.png" width="300"
          height="169" attr="Live_Radio"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to a Live Radio service. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Live Radio",
      "description": "Live Radio Channel Description",
      "link" : "",
      "guid": "201",
      "media:group": {
          "media:content": [
            {
              "medium": "audio",
              "type": "application/dash+xml",
              "expression": "ota",
              "url": "https://doi.org/10.5239/BCC1-49C3"
            }
          ]
      },
      "media:group" : {
        "media:content" : [
          {
          "medium" : "image",
          "type" : "image/jpeg",
          "url" : "https://example.com/RadioFullScreenImg.jpg",
          "width" : 1920,
          "height" : 1080
          }
        ]

      },
      "media:thumbnail": [
        {
          "url": "https://example.com/thumb1.jpg"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Tag / Attribute Required	Value Required	Comment
item/media:content/@medium	audio	This informs the receiver to expect a radio service (rather than, for example, a video service)
item/media:content/@expression	ota	This informs the receiver to expect a GSID in the url attribute, and to use that url to tune to the ATSC 3.0 service as referenced by that GSID
item/media:content/@url	<globalserviceid_value> For example: https://doi.org/10.5239/BCC1-49C3	The Global Service ID of the service to tune to when the viewer selects the item of content on the rail
item/media:content/@type	application/dash+xml	This value must be set to "application/dash+xml", in order to match the data provided by the ATSC 3.0 receiver
item/media:content/@duration	<not defined in the media:content>	This attribute MUST NOT be defined.
item/pubdate	<not defined in the item>	This attribute MUST NOT be defined.

ANNEX D - Example Live TV with Now Program Information

The example shows how to add a content item that points to a currently running event on an ATSC 3.0 service. This is achieved by setting the:

- the `expression` value of the `<media:content>` tag to “ota”
- the `url` value of the `<media:content>` tag to be the GSID of the service you wish to link to
- the `medium` value of the `<media:content>` tag to be “`video`”
- the `duration` value of the `<media:content>` tag to the length of the event
- the `pubDate` value of the `<item>` tag to the start time of the event, such that the Framework can verify that the event is currently running

The channel can be any other service that the broadcaster carries via ATSC 3.0 in the market. This may be on the same transmitter or on a different one.

It is possible to add content items that point to an ATSC 3.0 service’s future events. This is achieved by setting the `duration` and `Item/pubDate` to match those of the future event.

Example: Linking to the Now Program. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Live_TV_Program_Now</title>
      <description>Live TV Streaming Content Description</description>
      <guid>202</guid>
      <Link></Link>
      <pubDate>Tue, 27 Dec 2022 01:00:08 +0000</pubDate>
      <media:group>
        <media:content
          medium="video"
          type="application/dash+xml"
          expression="ota"
          url="https://doi.org/10.5239/BCC1-49C1"
          duration="7200"/>
       </media:group>
       <media:thumbnail url="/london/img/Program_Now.png"
        width="300" height="169" attr="Live_TV_Program_Now"/>
    </item>
  </channel>
</rss>
```

Example: Linking to the Now Program. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Live TV",
      "description": "Live TV Channel Description",
      "guid": "201",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
          "media:content": [
            {
              "medium": "video",
              "type": "application/dash+xml",
              "expression": "ota",
              "url": "https://doi.org/10.5239/BCC1-49C1",
              "duration" : "7200"
            }
          ],
        "media:thumbnail": [
          {
            "url": "https://example.com/thumb1.jpg"
          }
        ]
      }
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	video	This informs the receiver to expect a video service (rather than, for example, a radio service)
item/media:content/@expression	ota	This informs the receiver to expect a GSID in the `url` attribute, and to use that url to tune to the ATSC 3.0 service as referenced by that GSID
item/media:content/@url	<globalserviceid_value> For example: https://doi.org/10.5239/BCC1-49C1	The Global Service ID of the service to tune to when the viewer selects the item of content on the rail
item/media:content/@type	application/dash+xml	This value must be set to "application/dash+xml", in order to match the data provided by the ATSC 3.0 receiver
item/media:content/@duration	For example: 7200	The number of seconds is the length of the program and MUST be greater than 0.
item/pubdate	For example: Tue, 27 Dec 2022 01:00:08 +0000	The start time of the program. As this is the "Now" (current) program, this will be in the past

ANNEX E - Example Live TV with Next Program Information

It is possible to link to future events in an ATSC 3.0 service's schedule by following the same structure as described in ANNEX E - Example Live TV with Next Program Information and ensuring that the `duration` and `Item/pubDate` values correctly reflect those of the event.

The Framework will detect that the event is not currently live, and will act accordingly. As time passes and the event becomes the current (Now) event, the Framework will offer it as a viewable content item.

The following table provides the required fields in order for the Run3TV platform to determine the type of content that is being presented.

Element / Attribute Required	Value Required	Comment
`item/media:content/@medium`	`video`	This informs the receiver to expect a video service (rather than, for example, a radio service)
`item/media:content/@expression`	`ota`	This informs the receiver to expect a GSID in the `url` attribute, and to use that url to tune to the ATSC 3.0 service as referenced by that GSID

`item/media:content/@url`	`<globalserviceid_value>`
	For example:
	`https://doi.org/10.5239/BCC1-49C1`
		The Global Service ID of the service to tune to when the viewer selects the item of content on the rail

`item/media:content/@type`	`application/dash+xml`	This value must be set to "`application/dash+xml`", in order to match the data provided by the ATSC 3.0 receiver
`item/media:content/@duration`	For example: `7200`	The number of seconds is the length of the program and MUST be greater than 0
`item/pubdate`	For example: `Tue, 27 Dec 2022 01:00:08 +0000`	The start time of the program. As this is a "Next" program, this start time will be in the future

ANNEX F - Example Live TV Streaming Content

The example below shows how to add a content item that points to a Live streamed TV service over the internet. This is achieved by setting the:

- the `expression` value of the `<media:content>` tag to "`nonstop`"
- the `url` value of the `<media:content>` tag to be the location of the live streamed content
- the `medium` value of the `<media:content>` tag to be "`video`"
- the `duration` value of the `<media:content>` tag to be set to either the length of the content (if known) or "0".

The supported formats are HLS and DASH Live.

Example: Linking to a Live TV stream. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Live_TV_Streaming</title>
      <description>Live TV Streaming Content Description</description>
      <guid>202</guid>
      <link>News</link>
       <media:group>
        <media:content
          medium="video"
          type="application/vnd.apple.mpegurl"
          expression="nonstop"
          url="https://example.com/playlist.m3u8"
          duration="0"></media:content>
         <media:subTitle
          type="text/vtt"
          lang="en-us"
          href="http://example.com/subs/subs.vtt"></media:subTitle>
       </media:group>
        <media:thumbnail url="/london/img/Streaming_TV.png" width="300" height="169"
></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to a Live TV stream. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Live TV",
      "description": "Live TV Channel Description",
      "guid": "201",
      "link" : "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "video",
            "type": "application/vnd.apple.mpegurl",
            "expression": "nonstop",
            "url": "https://example.com/playlist.m3u8",
            "duration": "0"
          }
        ],
        "media:subTitle": [
          {
            "type": "text/vtt",
            "lang": "en-us",
            "href": "http://example.com/subs/subs.vtt"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/Streaming_TV.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	video	This informs the receiver to expect a video service (rather than, for example, a radio service)
item/media:content/@expression	nonstop	This informs the receiver to expect a livestream MPD in the url attribute
item/media:content/@type	application/vnd.apple.mpegurl or application/dash+xml	Only HLS and DASH are supported for live streaming
item/media:content/@url	<url of the content>	The URL of the content. Replaceable Field Tags (see [RUN3TV-IG-0204]) may be used
item/media:content/@duration	For example: 7000	This attribute may be provided to indicate the length of the live streaming channel (in seconds). If not known, this can be set to 0

If subtitles (captions) are not signaled as part of the live stream, it is possible to associate a subtitle (caption) track using the `<media:subTitle>` tag, as summarized below.

Element / Attribute Required	Value Required	Comment
`item/media:subTitle/@type`	At the time of publication, only webVTT subtitles/captions are supported: • `text/vtt`	Future subtitling types that may be supported include: • `application/x-subrip (SRT)` • `application/ttml+xml (TTML / IMSC1)` • `application/x-mp4-vtt (MP4 WebVTT)` • `application/cea-608 (CEA-608)` • `application/x-mp4-cea-608 (MP4 CEA-608)` • `application/cea-708 (CEA-708)`

ANNEX G - Example Live Radio Streaming Content

The example below shows how to add a content item that points to a Live streamed radio service over the internet. This is achieved by setting the:

- the `expression` value of the `<media:content>` tag to "`nonstop`"
- the `url` value of the `<media:content>` tag to the location of the live streamed content
- the `medium` value of the `<media:content>` tag to "`audio`"
- the `duration` value of the `<media:content>` tag to either the length of the content (if known) or "0".

The supported formats are HLS and DASH Live.

Example: Linking to a Live Radio stream. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Live_Radio_Streaming</title>
      <description>Live Radio Streaming Content Description</description>
      <guid>202</guid>
      <category>News</category>
       <media:group>
        <media:content
          medium="audio"
          type="application/vnd.apple.mpegurl"
          expression="nonstop"
          url="https://example.com/playlist.m3u8"
          duration="0"/>
       </media:group>
       <media:thumbnail url="/london/img/Streaming_Radio.png" width="300"
                        height="169" ></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to a Live Radio stream. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Live Radio",
      "description": "Live Radio Channel Description",
      "guid": "201",
      "category": ["News"],
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "audio",
            "type": "application/vnd.apple.mpegurl",
            "expression": "nonstop",
            "url": "https://example.com/playlist.m3u8",
            "duration": "0"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/Streaming_TV.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	audio	This informs the receiver to expect an audio service
item/media:content/@expression	nonstop	This informs the receiver to expect a livestream MPD in the url attribute
item/media:content/@type	application/vnd.apple.mpegurl or application/dash+xml	Only HLS and DASH are supported for live streaming
item/media:content/@url	<url of the content>	The URL of the content. Replaceable Field Tags (see [RUN3TV-IG-0204]) may be used
item/media:content/@duration	For example: 7000	This attribute may be provided to indicate a live radio streaming channel.

ANNEX H - Example VOD Content

The example below shows how to add a content item that points to a piece of VOD content. This is achieved by setting the:

- the `expression` value of the `<media:content>` tag to either:
  - "`full`" to play the content once
  - "`loop`" to play the content in a loop (see Looping and autoplaynext)
- the `url` value of the `<media:content>` tag to be the location of the VOD content
- the `medium` value of the `<media:content>` tag to be "`video`"
- the `duration` value of the `<media:content>` tag to be set to the length of the content.

The supported formats are MP4, HLS and DASH containers.

Example: Linking to VOD content. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Video_On_Demand</title>
      <description>A Video on Demand content</description>
      <guid>202</guid>
      <link></link>
       <media:group>
        <media:content
          medium="video"
          type="application/vnd.apple.mpegurl"
          expression="full"
          url="https://example.com/trailer.m3u8"
          duration="300"/>
       </media:group>
       <media:thumbnail url="/london/img/VOD.png" width="300" height="169"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to VOD content. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Video",
      "description": "VOD event description",
      "guid": "201",
      "link" : "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "video",
            "type": "application/vnd.apple.mpegurl",
            "expression": "full",
            "url": "https://example.com/trailer.m3u8",
            "duration": "300"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/VOD.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Taq / Attribute Required	Value Required	Comment
item/media:content/@medium	video	This informs the receiver to expect a video service (rather than, for example, a radio service)
item/media:content/@expression	full	This informs the receiver to expect an on-demand asset MPD in the url attribute. When the content completes playback, it stops
	loop	This informs the receiver to expect an on-demand asset MPD in the url attribute. When the content completes playback, it starts playback again[2]
item/media:content/@type	application/vnd.apple.mpegurl or: application/dash+xml or: video/mp4	HLS, DASH and MP4 containers are supported
item/media:content/@url	<url of the content>	The URL of the VOD content. Replaceable Field Tags (see [RUN3TV-IG-0204]) may be used
item/media:content/@duration	For example: 300	This attribute shall be provided to indicate the length of the program

If subtitles (captions) are not signaled as part of the live stream, it is possible to associate a subtitle (caption) track using the <media:subTitle> tag, as summarized below.

Tag / Attribute Required	Value Required	Comment
item/media:subTitle/@type	At the time of publication, only webVTT subtitles/captions are supported: • text/vtt	Future subtitling types that may be supported include: • application/x-subrip (SRT) • application/ttml+xml (TTML / IMSC1) • application/x-mp4-vtt (MP4 WebVTT) • application/cea-608 (CEA-608) • application/x-mp4-cea-608 (MP4 CEA-608) • application/cea-708 (CEA-708)

[2] An alternate means to enable looping playback is with the <media:embed> loopPlayback parameter value.

The example below describes VOD content where the player controls are hidden:

Example: Linking to VOD content with hidden player controls. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Video_On_Demand</title>
      <description>A Video on Demand content</description>
      <guid>202</guid>
      <link></link>
       <media:group>
        <media:content
          medium="video"
          type="application/vnd.apple.mpegurl"
          expression="full"
          url="https://example.com/trailer.m3u8"
          duration="300"/>
        <media:embed url="run3tv://player/AMP" >
          <media:param name="showPlaybackControls">false</media:param>
        </media:embed>
       </media:group>
       <media:thumbnail url="/london/img/VOD.png" width="300" height="169"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to VOD content with hidden player controls. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Video",
      "description": "VOD event description",
      "guid": "201",
      "link" : "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "video",
            "type": "application/vnd.apple.mpegurl",
            "expression": "full",
            "url": "https://example.com/trailer.m3u8",
            "duration": "300"
          }
        ],
        "media:embed" : [
          { "url" : "run3tv://player/AMP",
            "media:param" :
            {
              "showPlaybackControls" : false
            }
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/VOD.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

ANNEX I - Example Audio On Demand Content

The example below shows how to add a content item that points to a piece of Audio On Demand content. This is achieved by setting the:

- the `expression` value of the `<media:content>` tag to either:
  - "`full`" to play the content once
  - "`loop`" to play the content in a loop (see Looping and autoplaynext)
- the `url` value of the `<media:content>` tag to be the location of the on demand content
- the `medium` value of the `<media:content>` tag to be "`audio`"
- the `duration` value of the `<media:content>` tag to be set to the length of the content.

The supported formats are MP3, MP4, HLS and DASH containers.

Example: Linking to Audio On Demand content. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Theme Tune</title>
      <description>Audio On Demand event description</description>
      <guid>202</guid>
      <link>News</link>
       <media:group>
        <media:content
          medium="audio"
          type="application/vnd.apple.mpegurl"
          expression="full"
          url="https://example.com/theme-tune.m3u8"
          duration="300"/>
       </media:group>
      <media:group>
        <media:content medium="image" type="image/jpeg"
          url="https://example.com/audio.jpg" width="1920" height="1080"></media:content>
       </media:group>
       <media:thumbnail url="/london/img/AOD.png" width="300" height="169"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to Audio On Demand content. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Theme Tune",
      "description": "Audio On Demand event description",
      "guid": "201",
      "link": "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "audio",
            "type": "application/vnd.apple.mpegurl",
            "expression": "full",
            "url": "https://example.com/theme-tune.m3u8",
            "duration": "300"
          }
        ],
        "media:content": [
          {
            "medium": "image",
            "type": "image/jpeg",
            "url": "https://example.com/audio.jpg",
            "width": "1920",
            "height": "1080"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/AudioOnDemand.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	audio	This informs the receiver to expect an audio service
item/media:content/@expression	full	This informs the receiver to expect an on-demand asset MPD in the url attribute. When the content completes playback, it stops
	loop	This informs the receiver to expect an on-demand asset MPD in the url attribute. When the content completes playback, it starts playback again[3]
item/media:content/@type	One of the following: • application/vnd.apple.mpegurl • application/dash+xml • audio/mp4 • audio/mp3	HLS, DASH, MP4 and MP3 containers are supported.
item/media:content/@url	<url of the content>	The URL of the content. Replaceable Field Tags (see [RUN3TV-IG-0204]) may be used
item/media:content/@duration	For example: 300	This attribute shall be provided to indicate the length of the program

[3] An alternate means to enable looping playback is with the <media:embed> loopPlayback parameter value.

ANNEX J - Example Image Content

The example below shows how to add a content item that points to an image. This is achieved by setting the:

- the `url` value of the `<media:content>` tag to be the location of the image
- the `medium` value of the `<media:content>` tag to be "`image`"

The supported formats are PNG, JPEG and GIFs. The content in the `media:group` can contain the main image to render, whilst the thumbnail provides an indicator on what content will be accessible when selected.

Images may be a maximum of 1920×1080 pixels.

Example: Linking to Image content. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Weather Map</title>
      <description>Today's high temperatures</description>
      <guid>202</guid>
      <link></link>
       <media:group>
        <media:content
          medium="image"
          type="image/jpeg"
          url="https://example.com/weather.jpg"
          width="1920"
          height="1080"/>
       </media:group>
       <media:thumbnail url="/london/img/24Weather-report.png"
        width="300" height="169" attr="Weather_Report"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to Image content. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Weather Map",
      "description": "Today's high temperatures",
      "guid": "201",
      "link": "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "image",
            "type": "image/jpeg",
            "url": "https://example.com/weather.jpg",
            "width": "1920",
            "height" : "1080"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/24Weather-report.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	image	This informs the receiver to expect an image
item/media:content/@type	One of the following: • image/jpeg • image/gif • image/png	
item/media:content/@url	<url of the content>	The URL of the image. Replaceable Field Tags (see [RUN3TV-IG-0204]) may be used

ANNEX K - Web Display Content

The example below shows how to add a content item that points to web content. This is achieved by setting the:

- the `url` value of the `<media:content>` tag to be the location of the web content
- the `medium` value of the `<media:content>` tag to be "`document`"

HTML and JavaScript are used to render the page into an IFRAME.

Authors of the Web Display page should size the page to 1920×1080px or 1280×720px and the Run3TV Framework shall resize accordingly depending on the viewer request.

Example: Linking to web content. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
   <item>
      <title>Dynamic Weather Map</title>
      <description>See the latest weather map, dynamically updated every 15 mins</description>
      <guid>202</guid>
      <link></link>
       <media:group>
        <media:content
          medium="document"
          type="text/html"
          url="https://example.com/weather.html"/>
       </media:group>
       <media:thumbnail url="/london/img/WebDisplay.png" width="300"
height="169"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to web content. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Dynamic Weather Map",
      "description": "See the latest weather map, dynamically updated every 15 mins",
      "guid": "201",
      "link": "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "document",
            "type": "text/html",
            "url": "https://example.com/weathermap.html"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/WebDisplay.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	document	This informs the receiver to expect a web page
item/media:content/@type	text/html	Web Page mime type
item/media:content/@url	<url of the content>	The URL of the image. Replaceable Field Tags (see [RUN3TV-IG-0204]) may be used

ANNEX L - Application Linking Content

The example below shows how to add a content item that points to an application. This is achieved by setting the:

- the `medium` value of the `<media:content>` tag to be either:
  - "`document`" for Run3TV Framework applications
  - "`executable`" for receiver-native applications
- the `type` value of the `<media:content<` tag to be:
  - "`run3tv/local`" for local applications
  - "`run3tv/remote`" for remote applications
  - "`run3tv/{manufacturer}`" for SmartTV receiver-native applications
- the `url` value of the `<media:content>` tag to be the identifier of the application
  - for Run3TV Framework applications, this should refer to the application's `appName` as defined in appsList.json
  - for receiver-native applications, these will be unique to the manufacturer

It is possible (as shown in the example below) to include multiple applications to launch. The Framework will launch the first application that is available to the Framework.

This is useful when attempting to launch receiver-native applications, as all manufacturers' applications can be listed, and the Framework will choose the correct application for the receiver.

Note For details of SmartTV receiver-native application names, please refer to your Run3TV representative.

Example: Linking to applications. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Application Linking</title>
      <description>Application Linking Content Description</description>
      <guid>202</guid>
      <Link></Link>
      <media:group>
       <!-- Ordered list of applications to launch -->
        <media:content
          medium="document"   type="run3tv/local"          url="pac-man"/>
        <media:content
          medium="document    type="run3tv/remote"         url="work.yotta.pac-man?params=1"/>
        <media:content
          medium="executable" type="run3tv/samsung"      url="work.yotta.pac-man"/>
        <media:content
          medium="executable" type="run3tv/lge"
          url="work.yotta.pac-man?params=1&amp;params=2"/>
      </media:group>
      <media:thumbnail url="/london/img/Application.png" width="300" height="169"
attr="Application_Display"></media:thumbnail>
    </item>
  </channel>
</rss>
```

Example: Linking to applications. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Dynamic Weather Map",
      "description": "See the latest weather map, dynamically updated every 15 mins",
      "guid": "201",
      "link": "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "document",
            "type": "run3tv/local",
            "url": "pac-man"
          },
          {
            "medium": "document",
            "type": "run3tv/remote",
            "url": "work.yotta.pac-man?params=1"
          },
          {
            "medium": "executable",
            "type": "run3tv/samsung",
            "url": "work.yotta.pac-man"
          },
          {
            "medium": "executable",
            "type": "run3tv/lge",
            "url": "work.yotta.pac-man?params=1&amp;params=2"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/WebDisplay.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the tables below in detail, and are summarized in media:Content Content types.

Local Run3TV applications

The table below summarizes the fields required for the Framework to launch a local RUN3TV application.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	document	This informs the receiver to expect a local Run3TV application
item/media:content/@type	run3tv/local	The mime type defines linking to a local run3tv application as defined in the local appslist.json
item/media:content/@url	<application-name>[?<parameters>]	The application-name is one that is listed in the appsList.json.

Remote RUN3TV Applications

The table below summarizes the fields required for the Framework to launch a remote RUN3TV application.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	document	
item/media:content/@type	run3tv/remote	The mime type defines linking to a local run3tv application as defined in the local appslist.json file
item/media:content/@url	Of the form: <a344-appId>[?<parameters>]	The a344-app-id shall correspond to the signaling information of the remote application. An optional set of parameters can be added for the remote application to receive. Using a ? will delimit the parameters. E.g. work.yotta.pac-man?params=1&params=2 If no parameters are supplied, the default parameters in appsList.json shall be used

SmartTV (Receiver-native) Applications

The following table provides the required fields in order for the Run3TV platform to determine the type of content that is being presented when linking to a SmartTV Application. Note that there may be more than one <media:content> tag to support linking on different platforms for the same application.

The order of the applications listed determines the order of preference. The example above shows that Pac-man will launch the samsung link on a Samsung TV, a LGE link on a LGE TV and all others shall launch the local RUN3TV version.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	executable	
item/media:content/@type	run3tv/<manufacturer-name>	For example: run3tv/samsung For more information, please contact your RUN3TV representative
item/media:content/@url	<application-information>[?<parameters>]	Bespoke information provided by the manufacturers to support linking to 3rd party smart tv applications along with any optional parameters.

ANNEX M - Animation Content

The example below shows how to add a content item that points to an animation. This is achieved by setting the:

- the `url` value of the `<media:content>` tag to be the location of the Lottie image
- the `medium` value of the `<media:content>` tag to be "`document`"
- the `type` value of the `<media:content>` tag to be "`application/json+lottie`"

Example: Linking to animations. Key points indicated in bold

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
    <item>
      <title>Animation</title>
      <description>Animation</description>
      <guid>202</guid>
      <link></link>
      <media:group>
        <media:content
          medium="document"
          type="application/json+lottie"
          url="https://server.com/example-lottie.lottie"/>
       </media:group>
       <media:thumbnail url="/london/img/animation.png"
        width="300" height="169"/>
    </item>
  </channel>
</rss>
```

Example: Linking to animations. Key points indicated in bold

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Animation",
      "description": "Animation",
      "guid": "201",
      "link": "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "document",
            "type": "application/json+lottie",
            "url": "https://example.com/example-lottie.lottie"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/animation.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Element / Attribute Required	Value Required	Comment
item/media:content/@medium	document	This informs the Framework to expect a document file. Lottie files are of such type
item/media:content/@type	application/json+lottie	
item/media:content/@url	<Location of the Lottie animation file>	The URL of the Lottie file.

ANNEX N - Alert Content Examples

The example below shows how to add a content item that is marked as an alert.

Any of the content types described in this document can be marked as an alert by including a `<media:category>` tag within the same `<media:group>` as the `<media:content>` to be presented.

Note The `<media:category>` tag can also be used to indicate the following:

Type	Label	Text
Advertisement	`Advertisement`	
Sponsorship	`Sponsorship`	
Promotion	`Promotion`	

In the example below, the alertVid.mp4 video is marked as an alert with an alert category of “Advisory” and a priority of “2”.

Example: Linking to an alert. Key points indicated in bold.

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:media="http://search.yahoo.com/mrss/" version="2.0">
  <channel>
    <title>News</title>
    <link>https://example.com</link>
    <description>Description</description>
    <language>en-us</language>
    <pubDate>Sat, 07 Sep 2002 0:00:01 EST</pubDate>
    <lastBuildDate>Sat, 07 Sep 2002 9:42:31 EDT</lastBuildDate>
    <ttl>50</ttl>
   <item>
      <title>Alert example</title>
      <description>An alert example</description>
      <guid>202</guid>
      <link></link>
       <media:group>
        <media:content
          medium="video"
          type="video/mp4"
          url="https://example.com/alertVid.mp4"
          duration="300"
          width="1280"
          height="720"/>
         <media:category
          scheme="https://run3tv.org/mrss/category_schema"
          label="Alert">Advisory/2</media:category>
       </media:group>
       <media:thumbnail url="/london/img/WebDisplay.png" width="300"
height="169"></media:thumbnail>
    </item>
  </channel>
</rss>
```

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Alert example",
      "description": "An alert example",
      "guid": "201",
      "link" : "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "video",
            "type": "video/mp4",
            "url": "https://example.com/alertVid.mp4",
            "duration": "300",
            "width" : "1280",
            "height" : "720"
          }
        ]
      },
      "media:category": [
        {
          "scheme": "https://run3tv.org/mrss/category_schema",
          "label": "Alert",
          "category": "Advisory/2"
        }
      ]
    }
  ]
}
```

The fields used by the Framework to determine the type of alert to present are described in the table below.

Element / Attribute Required	Value Required	Comment
item/media:content	<valid_content_item>	A media:content object must exist within the same media:group as the media:category object that defines the alert. In other words, an alert within a media:category on its own does not itself contain a reference to any content to present
item/media:category/@scheme	https://run3tv.org/mrss/cateogry_schema	
item/media:category/@label	Alert	This must always be set to "Alert"
item/media:category	<A331-AEA-CATEGORIES>/<A331-AEA-PRIORITY> For example: WEATHER/4	This defines the category and priority of the alert. See <media:category> for details of how to configure this value

The following examples show a different type of alert using different media to provide that messaging.

Example: Image and alert. Key points indicated in bold.

XML

```
    <item>
       <title>Alert example</title>
       <description>An alert example</description>
       <guid>202</guid>
       <link></link>
        <media:group>
        <media:content
         medium="image"
         type="image/jpeg"
         url="https://example.com/weatherWarning.jpg"
         width="1920"
         height="1080"/>
        <media:category
         scheme="https://run3tv.org/mrss/category_schema"
         label="Alert">WEATHER/4</media:content>
        </media:group>
        <media:thumbnail url="/london/img/WebDisplay.png" width="300"
height="169"></media:thumbnail>
     </item>
```

Example: Image and alert. Key points indicated in bold.

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Alert example",
      "description": "An alert example",
      "guid": "201",
      "link" : "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "image",
            "type": "image/jpeg",
            "url": "hhttps://example.com/weatherWarning.jpg",
            "width" : "1920",
            "height" : "1080"
          }
        ]
      },
      "media:category": [
        {
          "scheme": "https://run3tv.org/mrss/category_schema",
          "label": "Alert",
          "category": "WEATHER/4"
        }
      ]
    }
  ]
}
```

ANNEX O - Settings

The example below shows how to add a content item that points to a settings page. This is achieved by setting the:

- the `medium` value of the `<media:content>` tag to be "`document`"
- the `type` value of the `<media:content>` tag to be "`run3tv/frwk`"
- the `url` value of the `<media:content>` tag to be either "`settings`", "`properties`", or sub-section of properties

Example: Linking to a settings page. Key points indicated in bold

XML

```
   <item>
      <title>Settings page</title>
      <description>Change settings</description>
      <guid>202</guid>
      <link></link>
       <media:group>
        <media:content
          medium="document"
          type="run3tv/frwk"
          url="settings"/>
       </media:group>
       <media:thumbnail url="/london/img/WebDisplay.png" width="300"
height="169"></media:thumbnail>
    </item>
```

JSON

```
{
  "title" : "Title",
  "link" : "https://example.com",
  "description" : "Description",
  "language" : "en-us",
  "pubDate" : "Sat, 07 Sep 2002 0:00:01 EST",
  "lastBuildDate" : "Sat, 07 Sep 2002 9:42:31 EDT",
  "ttl" : "50",
  "items": [
    {
      "title": "Settings page",
      "description": "Change settings",
      "guid": "201",
      "link" : "",
      "pubDate": "Mon, 01 Jul 2019 04:28:14 GMT",
      "media:group": {
        "media:content": [
          {
            "medium": "document",
            "type": "run3tv/frwk",
            "url": "settings"
          }
        ]
      },
      "media:thumbnail": [
        {
          "url": "/london/img/settings.png",
          "width": "300",
          "height": "169"
        }
      ]
    }
  ]
}
```

The following `url` values are supported:

URL	Description
`settings`	The root of the Framework's settings section
`properties`	The root of the Framework's properties section

As described above, the Run3TV Framework uses a number of fields to determine the type of content that is to be presented. These fields are described in the table below in detail, and are summarized in media:Content Content types.

Element / Attribute Required	Value Required	Comment
`item/media:content/@medium`	`document`	Combined, these values inform the framework that it should expect a framework-specific url
`item/media:content/@type`	`run3tv/frwk`	
`item/media:content/@url`	`<section>?params=1&params=2`	The section shall define an area that the framework can pass the params that have been called. Using a ? will delimit the parameters. E.g. `settings?params=1&params=2` NOTE: settings for top level settings settings?submenu=location