IVP 6.0
User Manual
written by Cad Delworth CEng MBCS CITP

Introduction

Thanks for downloading **IVP**, the **'Intelligent' Voicetrack Processor** script for version 4.x of the unparalleled **mAirList** radio playout software written and maintained by Torben Weibert in Germany.

Please note that IVP 6.0 will *only* work with version 4.x of mAirList; for earlier versions of mAirList, use IVP 5.11 instead.

I wrote IVP to automate the task of adding 'talkovers' (and now 'talkouts' or 'talk*under*s' as well) to voicetracked Playlists. IVP is a mAirList script which I provide as **freeware** under the terms of the GNU Public Licence.

If you have any problems with IVP, find any bugs, or have suggestions for further improvements or features, please post your thoughts in the mAirList user forum, where I am an active member (username: **cad**).

Installation

UnZIP the **mls** files into your mAirList *Scripts* folder (usually, this is **c:\Program Files\mAirList\scripts**). Er ... that's it!

The main IVP script is **IVP-6.0.mls**.
The 'Playlist markup' scripts are **IVP-6.0-MarkEndingsByType.mls** and **IVP-6.0-MarkEndings.mls**.

To uninstall IVP, delete the scripts from your PC.

NOTE: You will find a *fully-commented* version of each IVP script in the IVP ZIPfile, in case you wish to study or amend the code.

TIP: If you find IVP useful, use the mAirList *configuration program* to add IVP scripts to the *Action* menu on mAirList computers in production offices and areas (wherever your producers and presenters 'prep' their Playlists!).

What Does IVP Do?

IVP processes the current Playlist and looks for items with special *Ending* codes. IVP uses these as 'commands' to modify the current Playlist to achieve 'talkover/talkunder' by changing cue points in the items and adding Volume Envelopes to 'duck' or 'fade' music tracks 'under' your voicetracks. You then save the amended Playlist for automated playout or mixdown.

Or if you prefer: sequence your music and voicetracks into a Playlist, 'mark' the items using *Ending* codes, then run IVP to do the chore of setting deadly accurate cue points to create 'killer' perfect talkovers, talkunders, and even music beds under voicetracks with **no** 'crashes'—**guaranteed!**

The **IVP-MarkEndingsByType** and **IVP-MarkEndings** scripts automate the process of adding *Ending* codes to the items in a Playlist.

NOTE: IVP **does not** change the audio files and **does not** change the MMD files associated with the audio files. The **only** changes IVP makes are to data *within the Playlist*: exactly the same process as manually changing cue points and envelopes for a Playlist item using the *Mix Editor* dialog.

What Do I Need To Do First?

- I ***strongly recommend*** that you use a Playlist with **at least** three Players for automated playout of a voicetracked Playlist.
  With only two Players, items can sometimes 'wait' for a free Player.
- IVP **totally relies** on a correctly tagged music library. If all your music has at least a *Ramp* and a *FadeOut* point set, you can use IVP to create talk*over*s ('overlays'). Run-on-*under*s ('underlays') work best if your music has an *Outro* point set. Of course, not every music track *has* a sensible *Ramp* or *Outro* point, so IVP does its best to make 'intelligent' decisions if any 'expected' cue points are not set.
- IVP also assumes that voicetracks are correctly 'topped and tailed (!),' or at the very least have *CueIn* and/or *CueOut* points set to eliminate any silence at the start or end.
- IVP also checks the durations of items you mark up. 'Songs' must be 90 seconds or more in duration; voicetracks must be 90 seconds or **less** in duration. (You can change these two settings, or remove the tests completely if you wish, by editing the IVP script.)

'Overlay' And 'Underlay'

In IVP, the terms **'overlay'** and **'underlay'** have these specific meanings:

- **OVERLAY** means:
  'this item (usually a **voicetrack**) ends by "talking **over**" the next item (usually a song), up to the next item's *Ramp* point.' To 'tighten' or 'loosen' this, change the global ramp offset setting in the IVP script.
- **UNDERLAY** means:
  'this item (usually a **song**) ends by "running on **under**" the next item (usually a voicetrack), starting from this item's *Outro* point.' In other words, the following voicetrack starts at this item's *Outro* point.

Hence, in most cases:

- you use **overlay** codes on **voicetracks**,
- and you use **underlay** codes on **songs**.

Overlay ending codes begin with the letter **o**.
Underlay ending codes begin with the letter **u**.

The simplest way you could mark up a voicetracked Playlist would be this:

Item Type	*Ending*
VOICETRACK	**o**
Song	**u**
VOICETRACK	**o**
Song	**u**
VOICETRACK	**o**

IVP provides a few extra Ending codes to allow you more control over segues. For example, you can talk up to a specific *Ramp* (1, 2, or 3) instead of up to the first *Ramp*; or start the next song in sync with the start of the voicetrack; or apply a long fadeout to a song as the voicetrack begins.

The full list of IVP Ending codes begins on the next page.

IVP Ending Code Reference

IVP uses the *Ending* codes below as its 'commands.'
You can type *Ending* codes in UPPERCASE or lowercase.

To add an *Ending* code to an item in the Playlist, either:

- type the code in the *End Type* box in the Playlist *Edit Bar*, or
- right-click a Playlist item, click *Properties*,
  and type the code in the *Ending* text box on the *General* tab.

The Playlist *Edit Bar* is easier to use, because you don't need to close and re-open a dialog for each item you want to edit.

OVERLAY Ending Codes

Ending code	On a ...	What it does:
o	**voicetrack**	(**o**verlay) Sets *StartNext* in the voicetrack so that it ends at the first *Ramp* of the next item.
o1	**voicetrack**	Same as **o**, but voicetrack ends at *Ramp1*.
o2	**voicetrack**	Same as **o**, but voicetrack ends at *Ramp2*.
o3	**voicetrack**	Same as **o**, but voicetrack ends at *Ramp3*.
os	**voicetrack**	(**o**verlay **s**ync) Sets *StartNext* in the voicetrack so the next item starts ***in sync with**** the start of the voicetrack.
ob	**voicetrack**	(**o**verlay **b**ed) Same as **os**, but also **fast fades** the next item (usually a music bed) at the end of the voicetrack, so that both **end** in sync.

* The voicetrack's *StartNext* is set to 250 mS (by default).

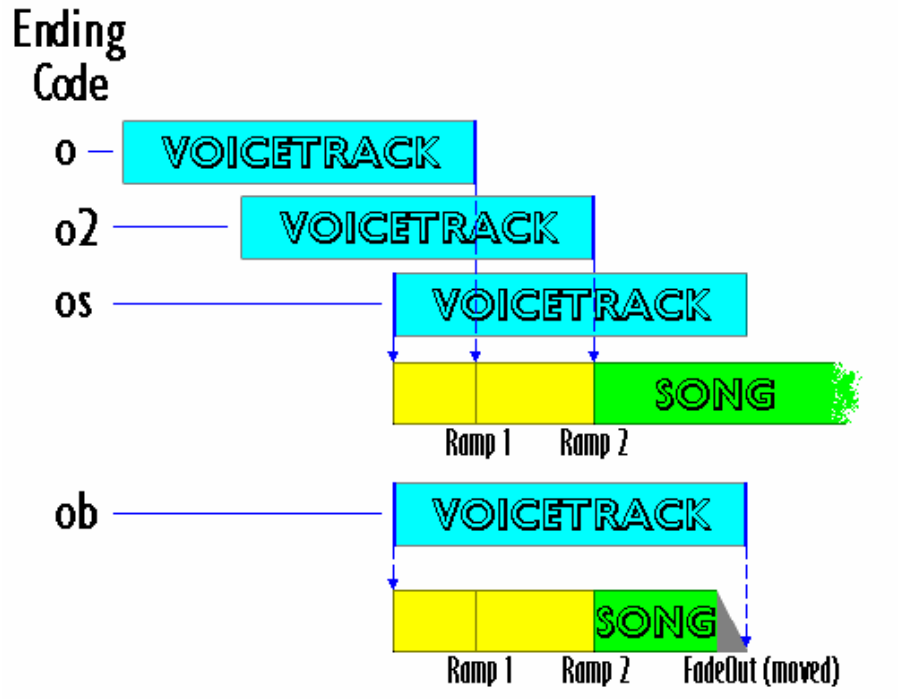

Figure 1: Examples of OVERLAY Ending codes

UNDERLAY Ending Codes

Ending code	On a …	What it does:
u	**song**	(**u**nderlay) Sets the song's *StartNext* to its *Outro* point, so the next item starts at the song's *Outro* point. The song's *FadeOut* point is moved if it would 'crash' the start of the item ***after*** the following item.
ulf	**song**	(**u**nderlay **l**ong **f**ade) Same as **u**, but executes a long fade from the song's *Outro* point to its (moved) *FadeOut* point.

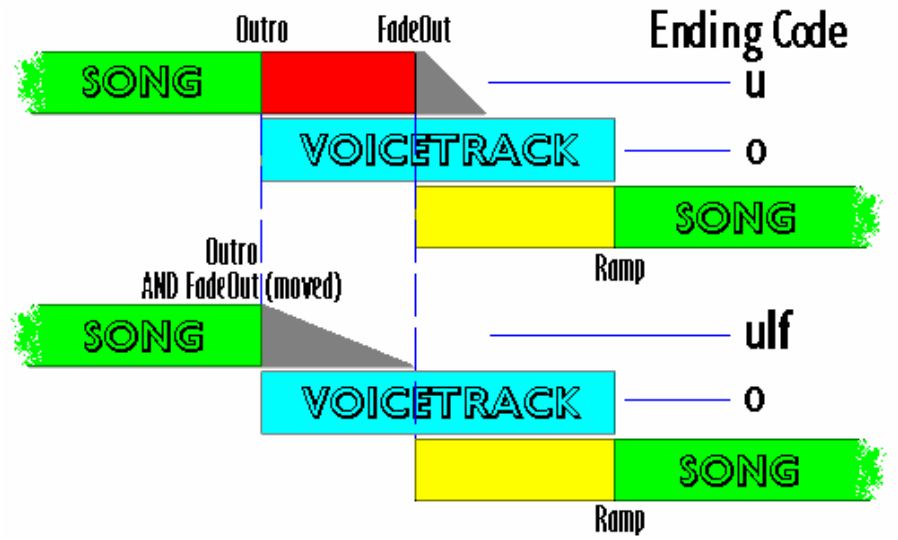

Figure 1: Examples of UNDERLAY Ending Codes

Volume Envelopes

IVP 'ducks' the level of music tracks to make their intros and outros play out 'under' voicetracks. IVP does this using the **volume envelope** for music tracks; adding points to drop and re-raise the track's level.

The default 'ducked' level is **11 dB** (on each channel) below peak level, but you can easily change this setting in the script.
The **same** 'ducked' level is used for intros, outros, and 'beds.'

To 'tighten' or 'loosen' the 'ducking' of intros/outros, change the duck intro/outro offset settings in the IVP script. (Think of this as globally 'moving' all duck in/out points backwards/forwards by these amounts.)

Note that by default, IVP **clears** the envelope before adding the 'duck' envelope points. You *can* change this if you wish, but I **strongly recommend** that you don't. Existing envelope points almost always 'interfere' with the points added by IVP, producing unexpected results.

Processing A Playlist With IVP

Step One: Prepare Your Playlist

1. Sequence your Playlist of voicetracks, songs, etc. as normal and **save** it.
2. Set the *Ending* codes on the items you want to overlay and underlay. Do this manually, or run the **IVP-6.0-MarkEndingsByType** or **IVP-6.0-MarkEndings** script. **Save** the Playlist again.

Some tips on setting Ending codes to achieve specific effects:

- **Voicetrack over music bed:**
  Mark the **voicetrack** with an **ob** code.
  Don't mark the bed!
- **Sweeper or jingle playing over start of song:**
  Mark the **sweeper/jingle** with an **os** code.
  Don't mark the item *before* the sweeper/jingle.

Step Two: Run The IVP Script

1. Run the **IVP-6.0** script.
2. Check the *mAirList* system log for any messages. In most cases, there will be no failures or warnings, meaning: everything worked.

   If there *were* any problems, the IVP warning or failure messages in the *mAirList* system log describe the problems.
3. Right-click the Playlist, then click **This Playlist...**, **PFL** to audition the segues IVP created. Alternatively, use the *Mix Editor*.
4. If you are happy with the results, **save** the Playlist again.
   TIP: Save the Playlist with a **different** name.

If anything didn't work as planned, **re-load** the Playlist you saved after Step One, make changes as needed, then **save** the Playlist again. You can now re-run the IVP script.
TIP: Save the changed Playlist with a **different** name.

Step Three: Test The Results

It's always a good idea to **test** a Playlist's segues after IVP has amended the cue points.

If you need to change anything, **re-load** the 'coded' Playlist you *should* have saved at the end of Step One above, change any or all of the *Ending* codes as needed, **save** the amended Playlist, then run IVP-6.0 again.

For example, if you find that an overlay doesn't work properly because the song it overlays has a very long *Ramp*, you might decide to change the code from **o** to **os** to start the voicetrack and song in sync instead. You may also need to clear the *Ending* code of the song *before* the voicetrack.

You may also need to change some *Ending* codes to allow for Break commands, ad. breaks, etc.

You can run IVP as many times as necessary to get the results **you** want, **but** if you need to re-run IVP against the same Playlist, you must re-load the Playlist—with the *Ending* codes added, but *before* IVP has altered anything—before **each** re-run of IVP.

Why Re-Load The Playlist Each Time?

IVP uses the **existing** cue point values in Playlist items to work out which cue point values need to be 'moved.'

After you run IVP, a number of cue points in your Playlist will have been 'moved' and quite probably some cue points will have been deleted as well.

This means that if you simply run IVP again, IVP will base its calculations on the **amended** cue points, and will invariably 'get things wrong.'

For this reason, if you need to change anything (such as changing an Ending code), you must re-load the **original** marked-up Playlist, make your changes, and save the changed Playlist **before** you run IVP again.

Advanced IVP

Amending IVP

You can change the default IVP setup by changing the **const**ant values at the start of the script. For example, you can change the Ending codes, or whether you want by default to 'talk over' up to the first or last *Ramp*.

In particular, you may want to change the default maximum duration of voicetracks (90 seconds).

Everything you need to know is in the comments within IVP. The only point to re-emphasise here is that all **overlay** codes **must** begin with the **same** first letter, because IVP relies on that to identify overlay codes.

TIP: Open the *fully commented* version of each script in your text editor to fully understand how each IVP script works.

IVP-6.0-MarkEndingsByType Script

The **IVP-6.0-MarkEndingsByType** script automates the process of setting *Ending* codes on the items in a Playlist. For most users, this script is a better choice than the **IVP-6.0-MarkEndings** script (see next page).

The script requires that every item in your Playlist has a **Type**, which is set when your files are imported into *mAirList*, or manually at a later time. If your items/files do **not** all have a *Type*, use the **IVP-6.0-MarkEndings** script (see next page) instead.

By default, the following *Types* are recognised:

- A **Music** or **Instrumental** item is marked as a **song** (*Ending* is set to **u**).
- A **Voicetrack** item is marked as a **voicetrack** (*Ending* is set to **o**) *unless* it is the final item in the Playlist.
- A **Sweeper** item is marked as a **'start in sync'** (*Ending* is set to **os**) *unless* it is the final item in the Playlist.

You can easily change the *Types* recognised, the *Ending* code set for each *Type*, etc. by changing the script.

Mark As 'Special Item:' A Note To Existing IVP Users

The *Special Items* option in *mAirList* 3.x does not exist in *mAirList* 4.x, hence this function has been removed from the **IVP 6.0** scripts.

IVP-6.0-MarkEndings Script

The **IVP-6.0-MarkEndings** script automates the process of setting *Ending* codes on the items in a Playlist. (Note that if all your items have *Types* set, the **IVP-6.0-MarkEndingsByType** script on the previous page is a better choice than the **IVP-6.0-MarkEndings** script.)

NOTE: You **will** need to amend the script to match the voicetrack file naming standards used at your station, or to change any of the defaults. You should find this very easy to do, even if you are not a programmer.

The script identifies items as songs or voicetracks using the following simple rules:

1. If the item's effective duration is at least **90 seconds** (you can amend this value), it is marked as a **song** (*Ending* is set to **u**).
2. Otherwise, the item's Title and Artist are checked. If the item's Title or Artist match any of the checks the script performs (see below), the item is marked as a **voicetrack** (*Ending* is set to **o**).

Is It A Voicetrack?

Many stations use a standard Title or Artist—or some standard letters at the start or end of Title or Artist—to identify an audio file as a voicetrack.

IVP-6.0-MarkEndings can search for strings which match:

- the **entire** Title and/or Artist, and/or
- the **start** (**prefix**) of the Title and/or Artist, and/or
- the **end** (**suffix**) of the Title and/or Artist.

For example, you could test for a Title starting **INTRO-**, or an Artist of **VT**, or a Title ending **VoiceTrack**, all at the same time.

If **any** strings match, the item is identified as a voicetrack and marked.

To enable these tests, change the appropriate **const**s within the script to a ***lowercase*** value. **NOTE:** You **must** type the strings in ***lowercase***!

Mark As 'Special Item:' A Note To Existing IVP Users

The *Special Items* option in *mAirList* 3.x does not exist in *mAirList* 4.x, hence this function has been removed from the **IVP 6.0** scripts.