StormCat User Guide

Table of Contents

Index

0

1 Introduction

StormCat

Last reviewed: 2019-11-01
Application Version: 1.11.1
Author: Jesús A. Mora

What is StormCat?

StormCat is a little tool for:

1. Creating and managing multiple independent catalogues of Moviestorm addons, either installed or present as addon files.
   Catalogues can be searched for specific assets according to a series of criteria.
   Further, asset lists created as the output of a search can be exported as Microsoft Excel 2007 workbooks.
2. Checking the validity and, optionally, listing the contents and details of your valid Moviestorm addons and content packages, either installed or present as files.
3. Checking the version of the Sketchup files to be imported in Moviestorm with the Modder's Workshop
4. Optionally restoring Moviestorm addon files disguised as archive files

StormCat is the natural successor to **MSAddonChecker**, a previous application conceived essentially for checking purposes (entries 2 to 4 in the previous list). While **StormCat** completely preserves (and eventually fixes some problems still present in the last version of **MSAddonChecker**) the functionalities of its predecessor, its main focus is on cataloguing the Moviestorm addons in your collection.

Although many of the repositories of free addons for Moviestorm have gone down over the last few years, there are still a few of them out there with a good deal of nice stuff available for downloading.

So if you happen to be a veteran Stormer (or an avid addon collector) you've probably managed to gather a few hundreds of addons for your collection. Most probably, you've used only a fraction of them along your lifetime as a Stormer and even most probably you can't say whether a specific asset you need for the project you're working right now is already readily available somewhere in your addon collection. Here is where **StormCat** comes in help, providing you with a handy tool for cataloguing all the addons and the individual assets in your collection.

Please see also:
StormCat Terminology

Moviestorm Stuff

1.1 StormCat Terminology

In this section, a few terms used throughout this guide will be introduced and defined.

- A ***Moviestorm Addon*** (or simply, ***Addon***) is a set of assets recognized by **Moviestorm** so they can be used by the application during the production of movies. Those assets are described more thoroughly in another section.
  Sometimes the term ***Content Package*** will be used. From every point of view they are undistinguishable from common addons, but I reserve this term for the official addons created by *ShortFuze/Moviestorm*.
  Addons can be present in two "states":
  - As addon package files: they are files with a file name followed by an extension `.addon`. This is the usual format addons are distributed. However, they need to be installed, either manually or by using the option in the **Moviestorm** application to this purpose.
    From a technical point of view addon files are simply .zip archives with their extension changed.
    NOTE: addon files for every official content package acquired by the user are stored inside the folder `{Moviestorm_Installation_Folder}\download`, that is, for a typical installation of Moviestorm on a Windows 64-bit, the folder at `C:\Program Files (x86)\Moviestorm\download`
  - As installed addons: they appear as folders inside one of the two addon installation folders, ie:
    Official content packs: `{Moviestorm_Installation_Folder}\Addons` (for a typical installation of Moviestorm on a Windows 64-bit, the folder at `C:\Program Files (x86)\Moviestorm\Addons`)
    User/Third-party addons: `{Moviestorm_User_Data_Folder}\Addons` (for a typical installation of Moviestorm on a Windows 64-bit, the folder at `C:\Users\[Your_Username]\Moviestorm\Addons`)
- An ***Addon Catalogue*** (or simply, ***Catalogue***) is a collection of addons created with **StormCat**, including information about every addon:
  - General information about the addon, such as name, publisher, datatime of last compilation, size of the mesh, statistics about the assets
  - Detailed information about every asset inside the addon, such as name, type and subtype, tags, etc.

  Catalogues are locally stored as files with a extension `.scat` and located in the same folder than the application executable.

  It might be a good idea to create a number of different catalogues for different purposes. For example, you may create a catalogue for gathering all the information about the currently installed addons, and another one for every available addon in your collection.
- The ***Catalogue Index*** is a special file automatically created and managed by **StormCat**

with very minimalistic information (its name and description) about every Catalogue created by the user and currently existing, plus about the name of the current Catalogue loaded by default. The Catalogue Index is stored as a file named `CataloguesIndex.xml` in the same folder that the application executable.

- ***Duplicate Groups***: if your collection of Moviestorm addons is very large (say, amounting a few hundred of entries) quite probably you've got a few of them by duplicate. StormCat will try to identify these duplicities by assigning a numerical value which identifies the addons belonging to the same group of duplicate addons, in order to deal with them. The identification process is customizable by the user and is discussed in an specific section.

2 Installing and Launching the Application

Installing the Application.

StormCat is distributed as an .ZIP archive.

To get it installed simply extract the content of the archive inside in an folder created for allocating the files that make up the application. Also be sure, the application will be able to create and/or delete files inside the folder it has been installed, which are required for its operation. These files include:

- Temporary folder and files
- Configuration files
- Catalogue files and index

Also it's advisable creating a direct link on your desktop (or in any other convenient place) to the application executable (*StormCat.exe*)

Pre-requisites

Microsoft .NET Framework 4.0, 4.5 or later must be previously installed in order to the application to work properly.

Launching the Application

There are a few ways to start the application:

1. Simply clicking on the application executable or in any link to it you've previously created. This way, the page by default (**Catalogue Content**) will we showed.
2. By dragging any number of files and/or folders and dropping then on the program's executable (or link). This way, the **Checking** page will become active and every recognized item passed to the application will be checked to determine whether they will work properly or not with the current version of Moviestorm (1.7.1). Any possible cause of problems will identified by the token string *'#!?'*
   Please note that the program only checks and reports abouth the following types of files:
   - Moviestorm addon files (extension .addon)
   - Moviestorm addons and content packages already installed (folders)
   - Sketchup files (extension .skp)
   - Archive files (extensions .zip .rar .7z): will look for addon and Sketchup files inside
   - Folders: will be scanned recursively

First time execution and Version changes

Please take into account that the first time the application is launched it must performs some extra chores:

- Determining the location of the Moviestorm installation and user data folders. Usually it gets them identified without a problem, unless you've installed Moviestorm in any location other than that defined by default. In this case, you'll need to provide this information in the Setup form.
- Creating a configuration file
- Creating a default Addon Catalogue and the Catalogue Index.

Also, there will be times when some significant changes have been introduced between versions in the definition of some relevant data structures, so **StormCat** will have to update on-the-fly some of those files (configuration, catalogues, index).

Command-line arguments

StormCat can be invoked also passing arguments as part of its command line. These arguments can be, for example, included, in the command line specified in the direct link to the application.

The accepted syntax for calling the program this way is:

`StormCat [options...] [source-specifications...]`

Options:

`-onlyissues | -i` : report only files with issues

`-showcontents | -c` : show the contents of valid addon files

`-listallanimations | -laa` : list all animation files

`-listgestureanimations | -lga` : list all gesture and gait animation files

`-listweirdgestures | -lwg` : list improper gesture verb definitions

`-compactdupverbs | -cdv` : compact duplicate verb names, even if their animation file names differ

`-correctdisguisedaddons | -cda` : correct valid addon files disguised as archives, keeping the original archive file

`-correctdisguisedaddons+ | -cda+` : correct valid addon files disguised as archives and delete the original archive file if succeeds.

`source-specifications`: any mix of files and/or folders you'd like to check.

PLEASE NOTE: options specified in the command line take precedence over and nullify processing options saved by default.

3 Application Main Form

When **StormCat** is launched, it will display its main form, which contains four selectable tabs, each one of them designed for a very specific purpose.

Those tabs or views are:

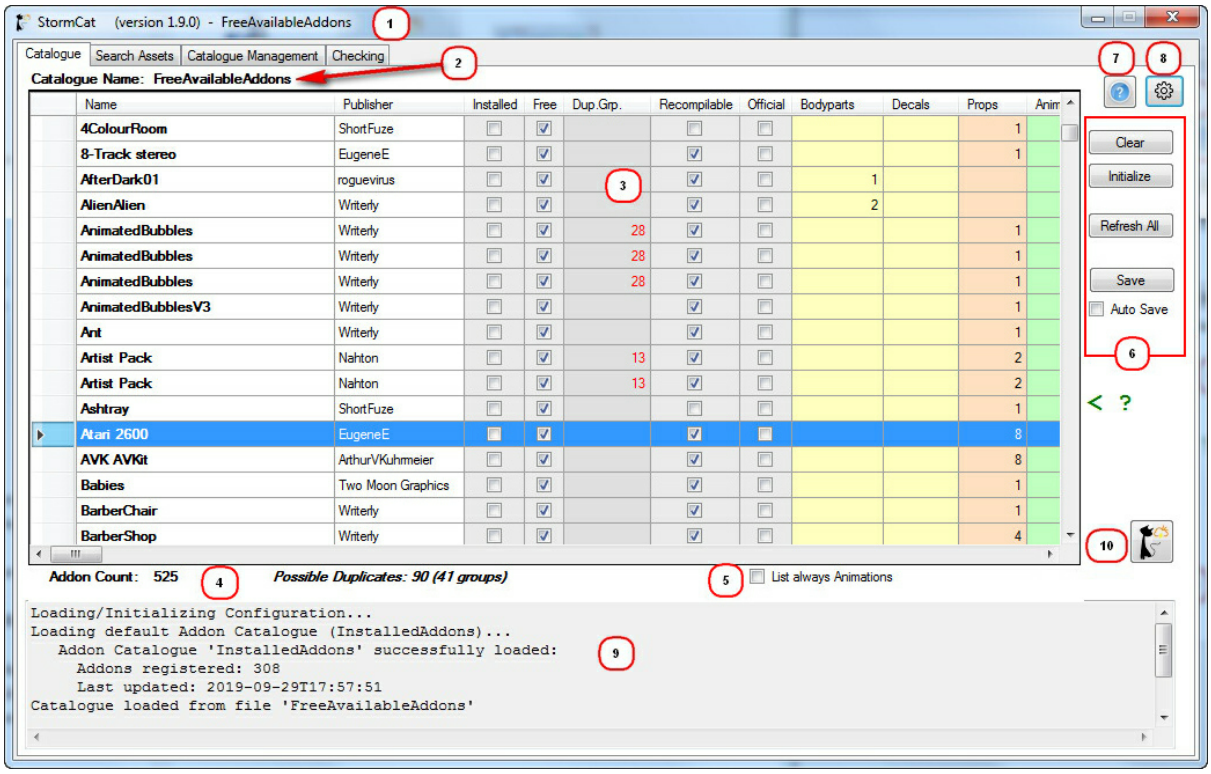
- Catalogue View: it shows information about the currently loaded and active Catalogue. It's the view selected by default most of the time, except when **StormCat** is launched by dragging and dropping some files and/or folders on the application executable or direct link.
- Search Assets View: it allows searching for assets in the addons included in the Catalogue collection, by specifying a wide range of different criteria.
- Catalogue Management View: it allows to perform a series of operations on Catalogues, such as creating new catalogues, or making copies, deleting, renaming, etc, existing Catalogues.
- Checking View: here's where files and folders can be checked and validated for their use as Moviestorm addons or source Sketchup files that can be imported with the Modder's Workshop.

3.1 Catalogue View

Catalogue View is the default view shown when **`StormCat`** is launched.

In essence, it displays information about the currently loaded and active Catalogue.

Its general appearance will be something like this:



The following features can be found:

1. The application title (common to every other view in the main form of the application) displays the ***name and version of the application***, followed by the ***name of the currently active Catalogue***.
2. The ***name of the current Catalogue*** is also shown on the very first line.
   PLEASE NOTE: the name of the current Catalogue will be displayed in red whenever there are any changes in its content not yet saved to file. As soon those pending changes are saved (either manually or automatically), the name will be displayed again with black characters.
3. A large data grid fills most of the view, with a ***list of the addons*** included in the Catalogue. Please refer to the section below for a complete reference about its content and use.
4. Below the data grid the total ***count of addons*** in the Catalogue is shown.
   Also, if one or more possible duplicate addons are detected according to the detection criteria currently established, the ***number of possible duplicates*** will be displayed. (Note: by clicking on this number, the current dup detection criteria will show up).
5. A checkbox to ***force the listing of Animations*** in the Asset List view, in addition to the Verbs. By default, only Verbs are listed.
6. A few controls which affect the current Catalogue as a whole:
   - ***Clear***: deletes every addon included in the current Catalogue. It will ask for confirmation before proceeding.
   - ***Initialize***: deletes every addon included in the current Catalogue and then will populate it with information about every official content pack and third-party addon currently installed. It will ask

for confirmation before proceeding.
- ***Refresh All***: updates the information for every addon in the current catalogue. NOTE: it can't take a while and it will only work for those addons that have not been moved or deleted from their registered location.
- ***Save***: saves the current Catalogue to disk, so changes made are preserved. Please note: it won't ask for confirmation.
- ***Auto Save*** option: if checked, every time a change is made to the current Catalogue, it will be automatically saved to disk, without asking for confirmation.

7. Invokes the help system.
8. Launches the Setup form, for modifying the general configuration of the application. The changes made to the configuration, if confirmed, will be saved so they will be apply in successive sessions until changed again.
9. A ***log box***, reporting relevant events.
10. Credits information.

Using the Addon List data grid

The Addon List data grid is the main control in the Catalogue page. As stated before, it lists every addon in the current Catalogue, along with many important items of summary information about each one of them, including:

- ***Name*** of the addon
- ***Publisher***
- A flag indicating the addon is currently ***installed*** and readily available for its use (although it might be disabled)
- A flag indicating whether is a ***free*** addon or requires a valid license
- A number indicating the ***Duplicate Group ID***, if according to the current duplicate detection criteria the addon is a possible duplicate.
- A flag indicating the addon, as provided, ***can be re-published***. For this, an addon must either:
  - include no asset requiring a mesh at all (body parts and/or props), or
  - include the source Cal3D mesh files (`.cmf`) for the assets included

  Please take into account this flag offers, at its best, only a guess.
- ***Official*** flag means is an official expansion developed by Moviestorm/ShortFuze; otherwise, it's a third-party addon
- ***Count*** of the different types of assets found inside the addon, including:
  - Body parts
  - Decals
  - Props
  - Animations and Verbs
  - Materials
  - Sounds
  - Special Effects
  - Cutting Room Assets
  - Sky Textures
  - Other Assets
  - Stocks
  - Demo Movies (Starting and Background movies)
- ***Description*** of the addon
- ***Location*** of the addon (folder or file)

Sorting

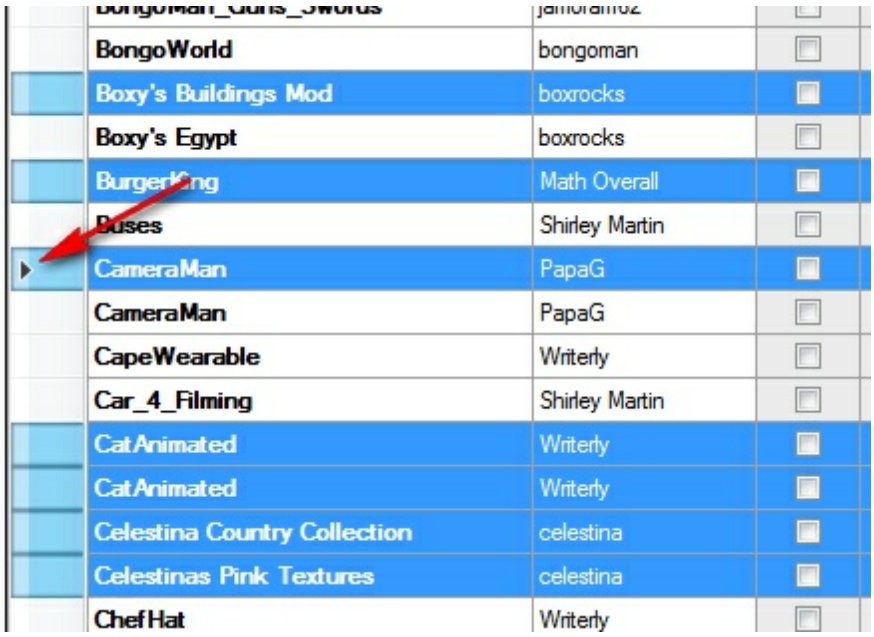
The addons are sorted by their name by default, but they can be sorted by most of the columns just by clicking on the column header.

Selecting addons

An addon in the list can be selected by simply clicking on its row. Clicking on another addon, will automatically de-select the previously selected.

It's possible to mark a number of addons as selected by keeping the Ctrl key pressed while clicking on each of them. Also a range of contiguous addons can be selected by clicking on the first (or last of them) and then, while keeping the Shift key pressed, click on the last (or the first) of the series.

However, take into account that, regardless multiple addons have been selected, some operations will only work with the one with the "current" index in its leftmost column (see image below):



Adding new addons to the Catalogue

It can be done by two procedures:

- Simply, dragging one or more addon files (or archives containing them) and/or folders and dropping them on the data grid.
- By copying a number of addons from a source Catalogue and pasting them into a destination Catalogue.

Working with addons

By double-clicking on any of the addons listed, a new window will pop up, with a list of the assets included in the selected addon.

Also, by right-clicking on the data grid, a menu will pop up with lots of options for displaying the contents

of the addons and performing a series of operations on them, such as:

- Deleting addons
- Refreshing addons
- Copying a number of addons from a source Catalogue and pasting them into a destination Catalogue.

About Duplicate Addons

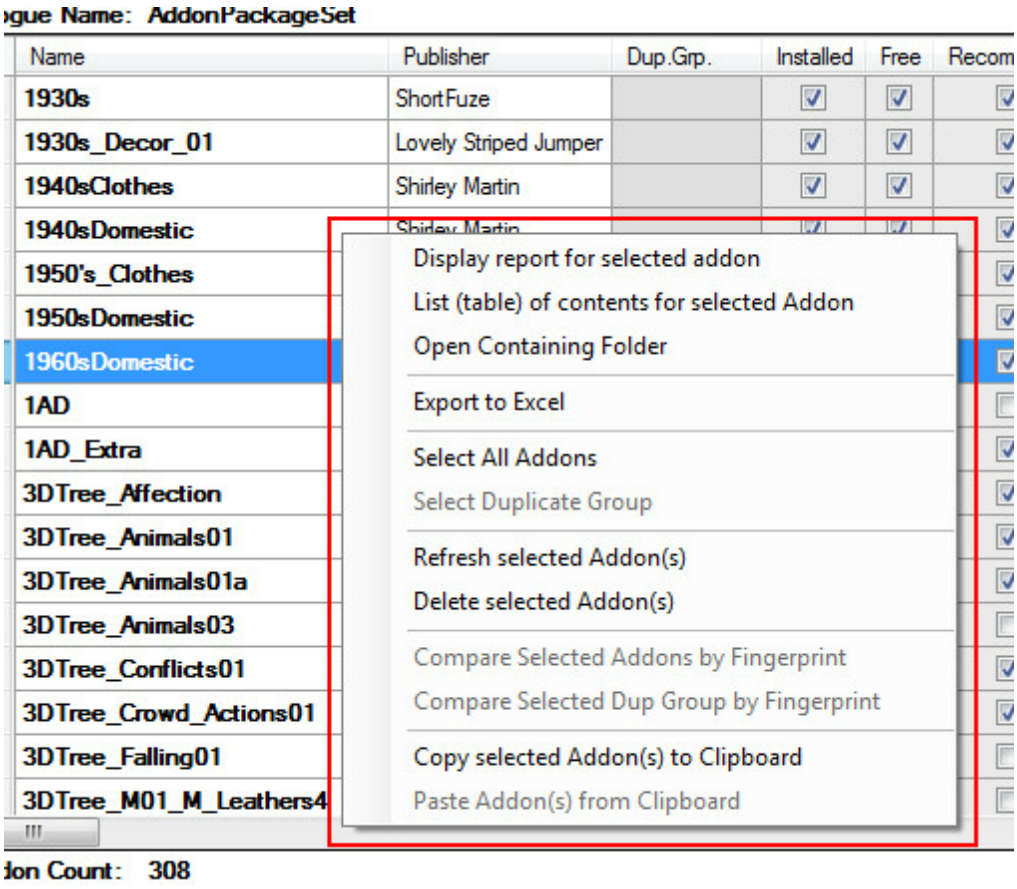
Addons labeled as possible duplicate according to the current duplicate detection criteria are assignated a Duplicate Group Id, so the list of addons may be sorted by this value.

Once a number of potentially duplicate addons are identified, it's advisable to carefuly compare the contents of each addon in a duplicate group, for confirming they're duplicates in fact, before proceeding to delete.

For a discussion about duplicate addons and their detection, please refer to this section.

3.1.1 Addon List Popup Menu

By right-clicking on the Addon list in the Catalogue view, a menu will pop up.



The options available might vary, depending on the number of the addons currently in the Catalogue and other circumstances. If the Catalogue contains no addons at all, the menu won't be displayed.

The options in the menu are these:

Display report for selected addon.
A new window will pop up, with a comprehensive report about the selected addon.
This option only works with the most currently selected addon (denoted by the index mark in its leftmost column).

List (table) of contents for selected Addon
A new window will pop up, with a list of the assets in the selected addon. The same effect will result by double-clicking on a addon.
This option only works with the most currently selected addon (denoted by the index mark in its leftmost column).

Open Containing Folder
A new instance of Windows File Explorer will be created displaying the contents of the folder containing the selected addon, according to its registered location.
If the addon is not installed, the file containing it will also be selected in the Explorer file list.

Export to Excel
The information in the addon list can be saved into a MS Excel workbook.

Select All Addons
Select all addons in the current Catalogue

Select Duplicate Group
If the currently selected addon is part of a Duplicate Group, all addons with the same Duplicate Group ID will also be selected.

Refresh selected Addon(s)
The information about the selected addons will be updated by analyzing their source file or folder, as specified in their Location.

Delete selected Addon(s)
Those addons selected will be deleted from the Catalogue, provided the user confirms the operation.

Compare Selected Addons by Fingerprint
If more than one addon is currently selected, a new form will open, comparing them by their fingerprints (provided they have been generated).

Compare Selected Duplicate Group by Fingerprint
If the currently selected addon is part of a Duplicate Group, a new form will open, comparing all the addons in the same Duplicate Group by their fingerprints (provided they have been generated).

Copy selected Addons(s) to Clipboard
The information regarding the selected addons will be copied into the clipboard, so it can be pasted later

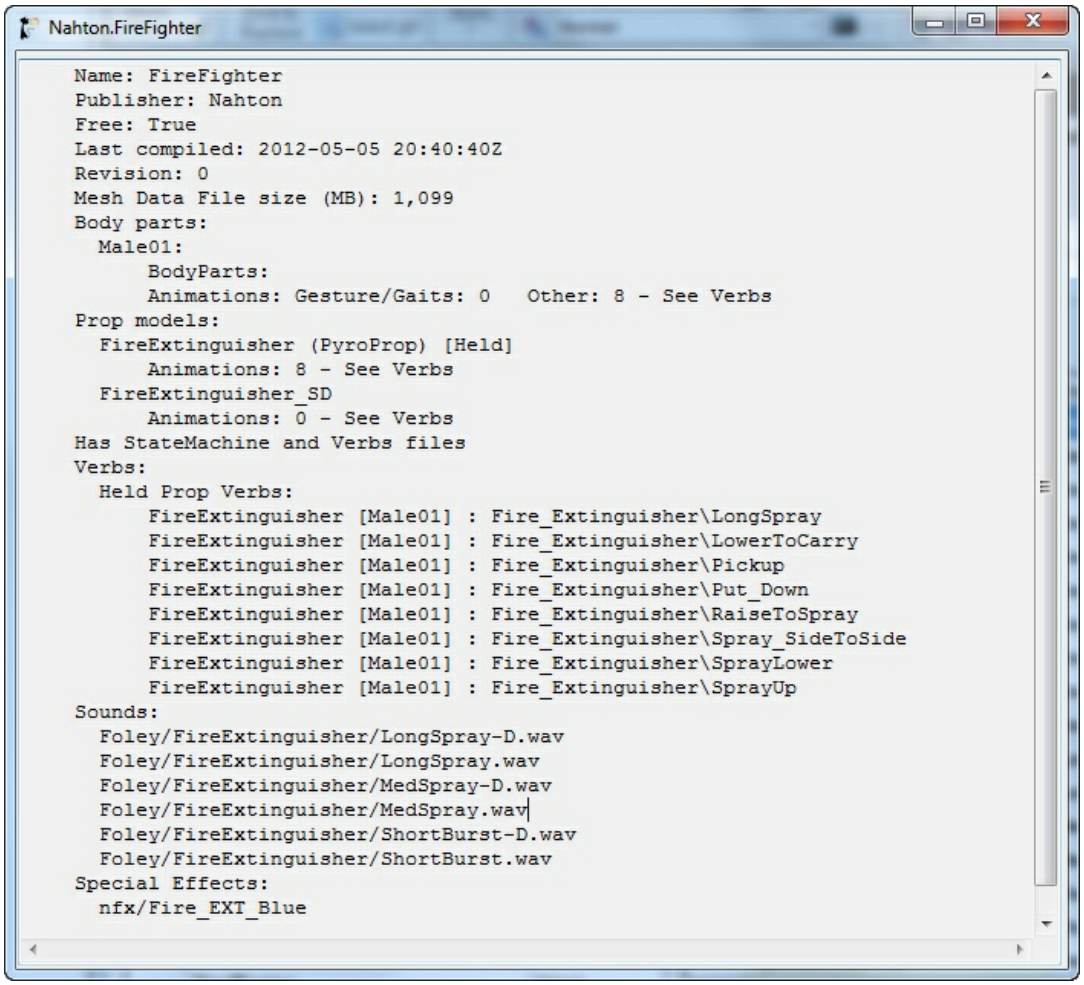
into another Catalogue (see next option).

Paste Addon(s) from Clipboard
If information about one or more addons has been previously copied into the clipboard (see previous option), with this option those addons will be added to the current Catalogue.

By using the last two options, addons included in one Catalogue can be easily copied into a second Catalogue, specially if they are loaded on different instances of the program, by using the ***'Load Child'*** command (see the Catalogue Management View commands)

3.1.2 Addon Report Form

This text report includes a most complete information about the addon it refers to.



By right-clicking on the text box, a menu will pop up with a few hopefully convenient options.

For a complete reference of the information in this report please refer to this page.

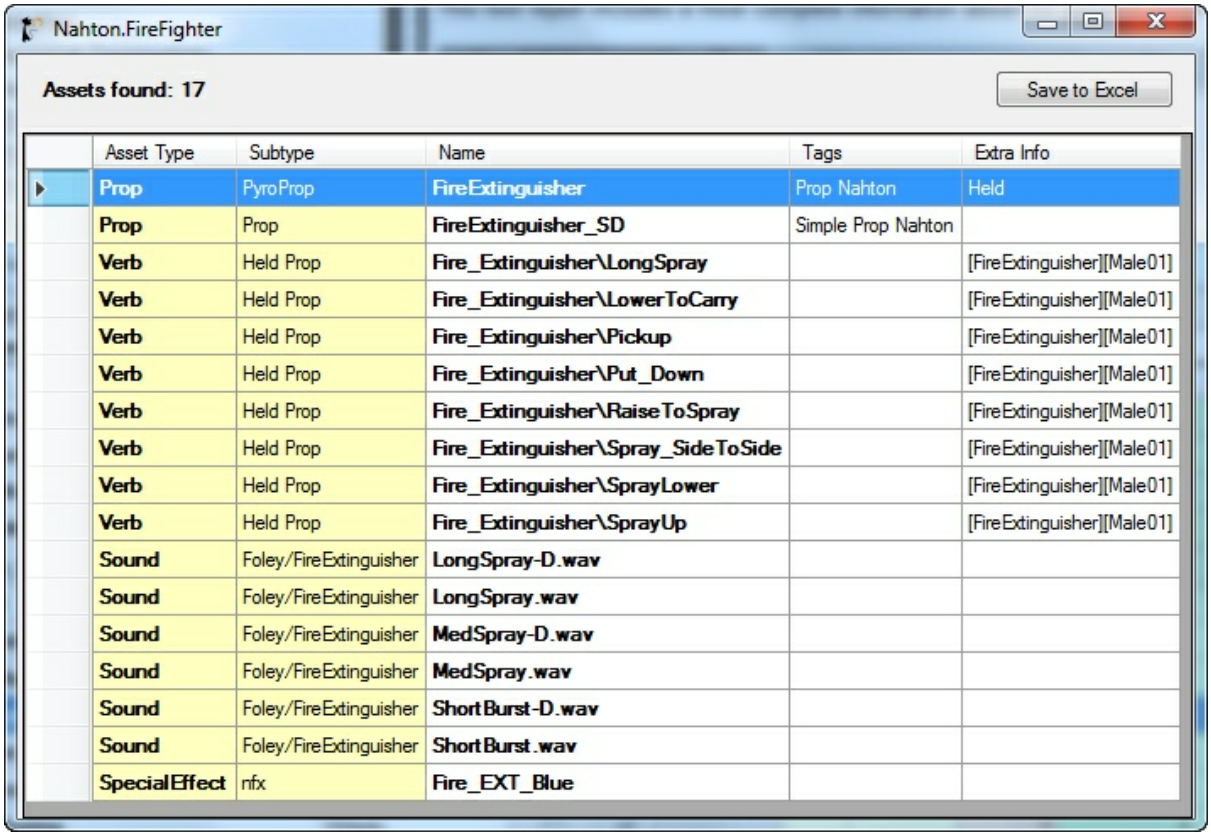
3.1.3 Asset List View

This form offers a list of the assets found in an addon.

The qualified name of the addon is shown on the title bar of the window. Also, a total count of the assets is displayed.

The main feature in the form is a table with detailed information about the assets in the addon, including:
- Asset Type
- Asset Subtype
- Name of the asset
- Tags
- Additional information about the asset, depending on the type of the asset.

For a complete reference about the information offered in this listing, please refer to this page.

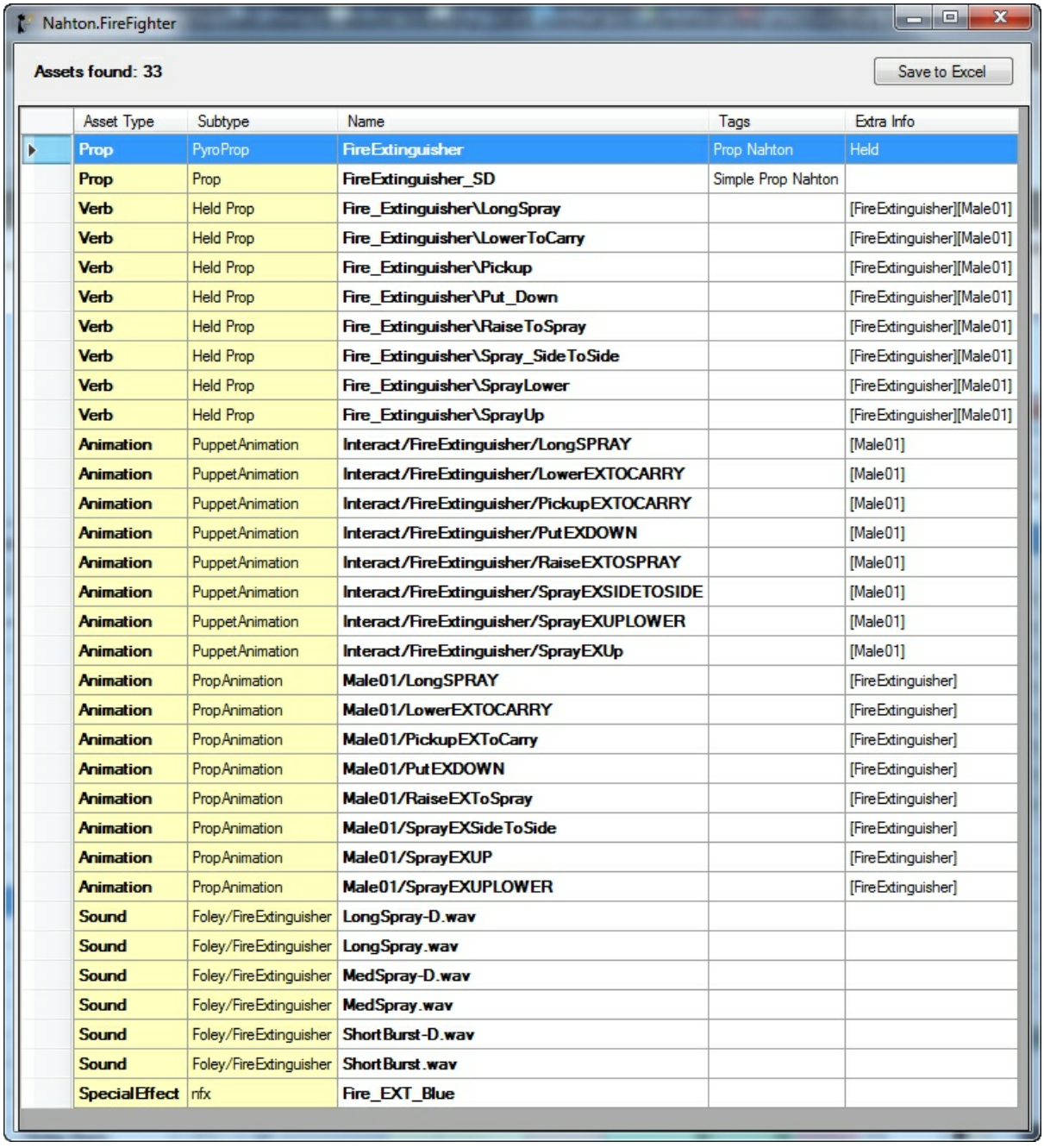


Asset Type	Subtype	Name	Tags	Extra Info
Prop	PyroProp	FireExtinguisher	Prop Nahton	Held
Prop	Prop	FireExtinguisher_SD	Simple Prop Nahton	
Verb	Held Prop	Fire_Extinguisher\LongSpray		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\LowerToCarry		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\Pickup		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\Put_Down		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\RaiseToSpray		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\Spray_SideToSide		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\SprayLower		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\SprayUp		[FireExtinguisher][Male01]
Sound	Foley/FireExtinguisher	LongSpray-D.wav		
Sound	Foley/FireExtinguisher	LongSpray.wav		
Sound	Foley/FireExtinguisher	MedSpray-D.wav		
Sound	Foley/FireExtinguisher	MedSpray.wav		
Sound	Foley/FireExtinguisher	ShortBurst-D.wav		
Sound	Foley/FireExtinguisher	ShortBurst.wav		
SpecialEffect	nfx	Fire_EXT_Blue		

By default, assets are sorted by their type, but they can be sorted by any of the columns by clicking on the corresponding column header.

Also, by clicking the ***'Save to Excel'*** button, the information in the asset table can be saved to a MS Excel Workbook.

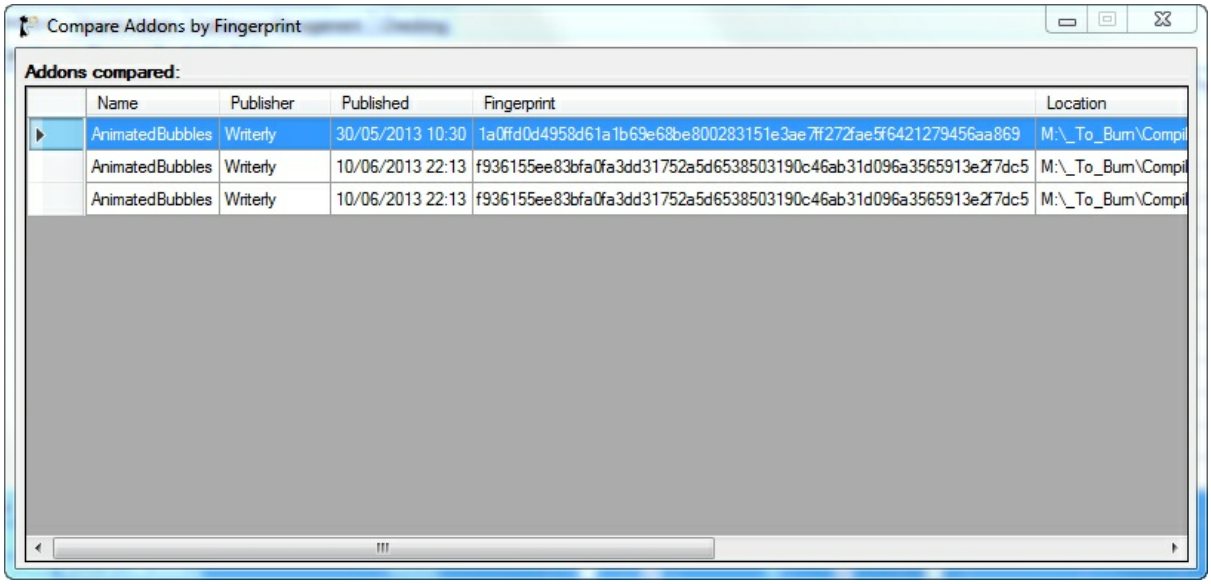
Finally, if the option 'List always animations' had been checked on the Catalogue view page, the animations in the addon will also be listed:



Asset Type	Subtype	Name	Tags	Extra Info
Prop	PyroProp	FireExtinguisher	Prop Nahton	Held
Prop	Prop	FireExtinguisher_SD	Simple Prop Nahton	
Verb	Held Prop	Fire_Extinguisher\LongSpray		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\LowerToCarry		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\Pickup		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\Put_Down		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\RaiseToSpray		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\Spray_SideToSide		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\SprayLower		[FireExtinguisher][Male01]
Verb	Held Prop	Fire_Extinguisher\SprayUp		[FireExtinguisher][Male01]
Animation	PuppetAnimation	Interact/FireExtinguisher/LongSPRAY		[Male01]
Animation	PuppetAnimation	Interact/FireExtinguisher/LowerEXTOCARRY		[Male01]
Animation	PuppetAnimation	Interact/FireExtinguisher/PickupEXTOCARRY		[Male01]
Animation	PuppetAnimation	Interact/FireExtinguisher/PutEXDOWN		[Male01]
Animation	PuppetAnimation	Interact/FireExtinguisher/RaiseEXTOSPRAY		[Male01]
Animation	PuppetAnimation	Interact/FireExtinguisher/SprayEXSIDETOSIDE		[Male01]
Animation	PuppetAnimation	Interact/FireExtinguisher/SprayEXUPLOWER		[Male01]
Animation	PuppetAnimation	Interact/FireExtinguisher/SprayEXUp		[Male01]
Animation	PropAnimation	Male01/LongSPRAY		[FireExtinguisher]
Animation	PropAnimation	Male01/LowerEXTOCARRY		[FireExtinguisher]
Animation	PropAnimation	Male01/PickupEXToCarry		[FireExtinguisher]
Animation	PropAnimation	Male01/PutEXDOWN		[FireExtinguisher]
Animation	PropAnimation	Male01/RaiseEXToSpray		[FireExtinguisher]
Animation	PropAnimation	Male01/SprayEXSideToSide		[FireExtinguisher]
Animation	PropAnimation	Male01/SprayEXUP		[FireExtinguisher]
Animation	PropAnimation	Male01/SprayEXUPLOWER		[FireExtinguisher]
Sound	Foley/FireExtinguisher	LongSpray-D.wav		
Sound	Foley/FireExtinguisher	LongSpray.wav		
Sound	Foley/FireExtinguisher	MedSpray-D.wav		
Sound	Foley/FireExtinguisher	MedSpray.wav		
Sound	Foley/FireExtinguisher	ShortBurst-D.wav		
Sound	Foley/FireExtinguisher	ShortBurst.wav		
SpecialEffect	nfx	Fire_EXT_Blue		

3.1.4 Compare Addons by Fingerprint Form

This form simply displays a data grid (table) with information about a number of selected addons, for detecting duplicate addons in the current catalogue with a higher degree of certainty.



The information for each addon in the list includes:

- Name
- Publisher
- Date and time the addon was published
- Fingerprint
- Fingerprint2: a hash calculated for every addon in the list. This value is much more reliable for comparing addons, as it based on the actual contents of the relevant files of the files in the package.
- Location of the folder or file containing the addon

Addons in the list can be sorted by any criteria just by clicking on the column headers.

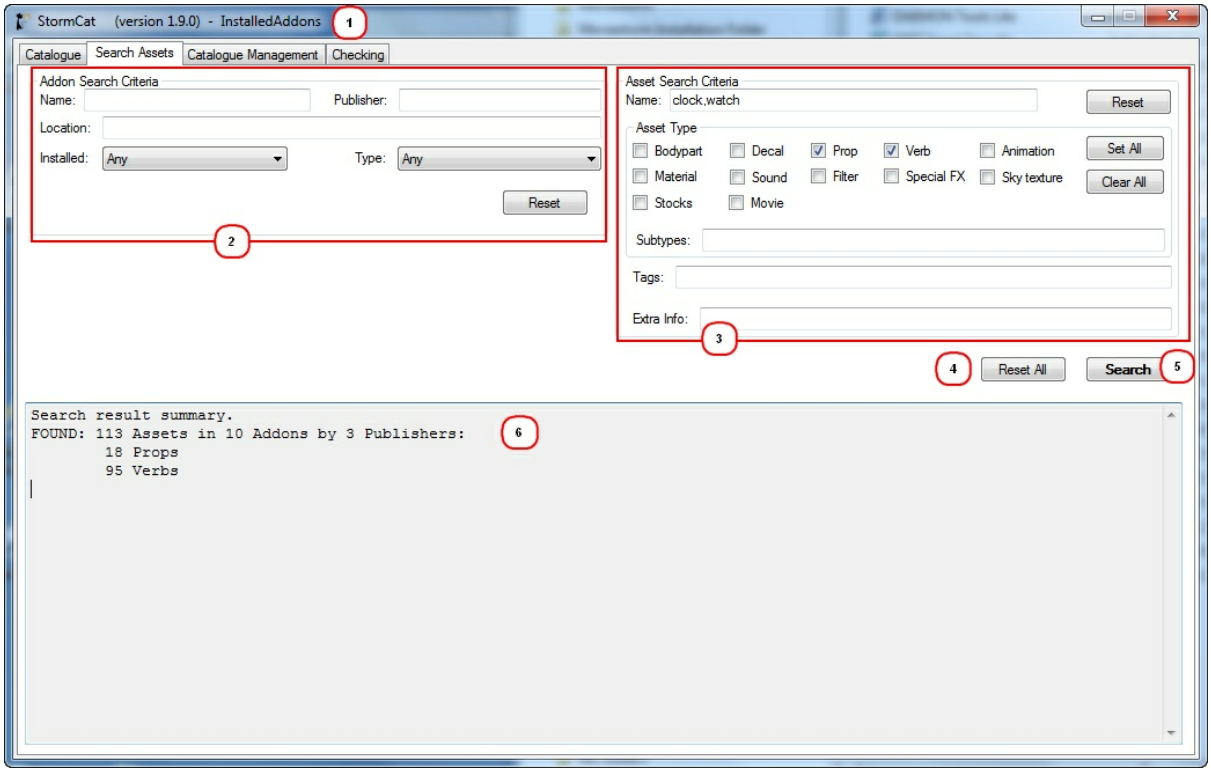
By double-clicking on any of the addons in the list, a new form with a full list of its assets will appear.

Right-clicking on the table, a contextual menu will pop-up, with a few options:

- Display Report for the Addon
- List contents of the Addon
- ***Open Containing Folder***: a new instance of Windows File Explorer will be created displaying the contents of the folder containing the selected addon, according to its registered location. If the addon is not installed, the file containing it will also be selected in the Explorer file list.

3.2 Search Assets

By using the controls in this tab, assets in the current Catalogue can be searched for according to a number of criteria.



1. The name of the current Catalogue is shown on the title bar of the application
2. A number of search criteria regarding the addon as a whole:
    - Name of the addon: a list of strings separated by commas. There will be a match if the name of an addon contains any of the strings specified. Eg: "*domestic,animal*" will match the addons named *1940sDomestic*, *1950sDomestic*, *1960sDomestic*, *3DTree_Animals01*, *DomesticDrama01*, etc
    - Publisher of the addon: a list of strings separated by commas. There will be a match if the name of addon's publisher contains any of the strings specified. Eg.: "*sh,nah*" will match the addons published by *Shirley Martin*, *ShortFuze*, *Nahton*, etc
    - Location: will try to match the location of the addon's file or folder
    - Installed: will search in installed addons, not installed ones, or every addon in the Catalogue.
    - Type: will search in official content packs only, third-party addons or every addon in the Catalogue

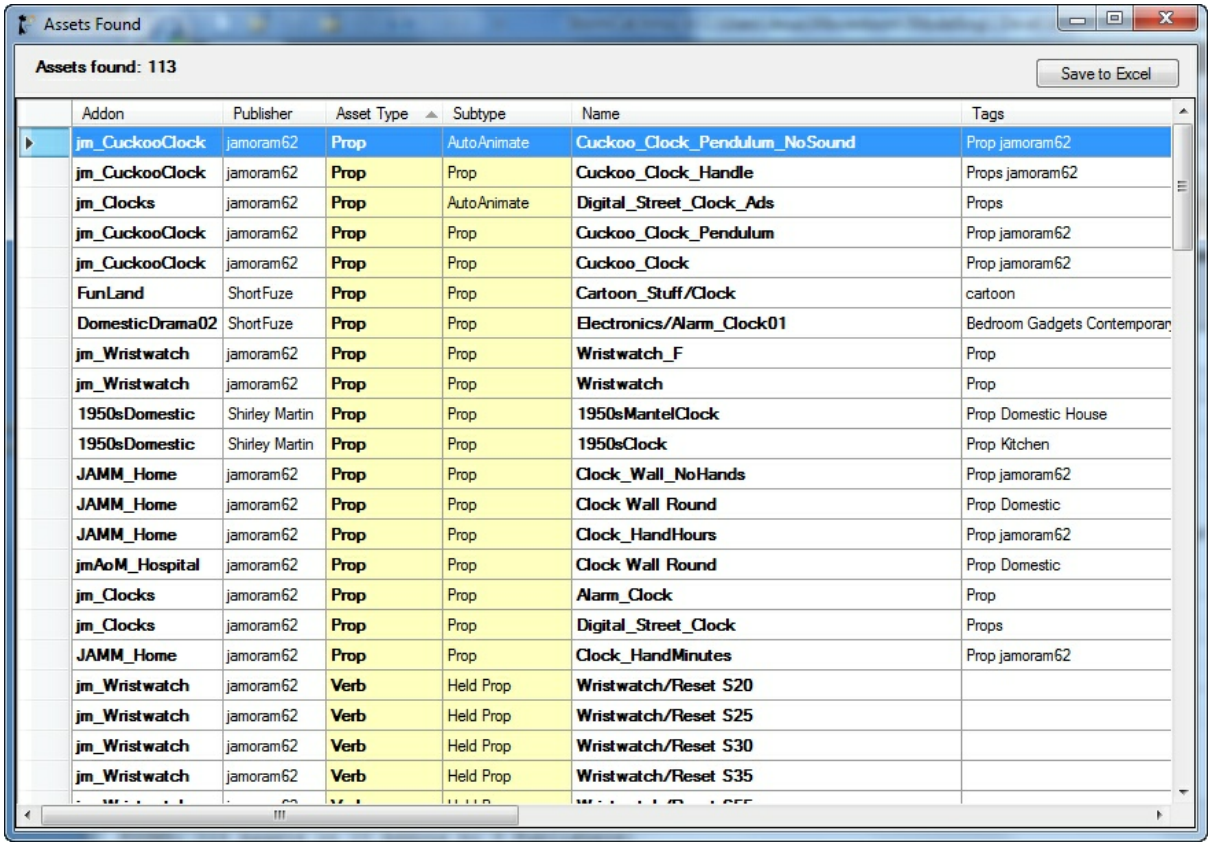
    By clicking on the **Restore** button, the addon criteria will be reset.
3. Search criteria specific for the addon(s) we're looking for:
    - Name of the asset: a list of strings separated by commas. There will be a match if the name of the asset contains any of the strings specified. Eg: "*motor,hover*" will match the assets named *Motorbike Leather*, *Road_Sign_Motor_Vehicle*, *HoverSlider05*, etc
    - Asset Type: filters the types of assets we're looking for. Clicking on the Set All button will search for every asset in the Catalogue. On the other hand, clicking on the Clear All will de-select all types.

- Asset Subtypes: a list of strings separated by commas. There will be a match if the name of the asset's subtype contains any of the strings specified. Eg:"*ceil,floor*" will return only those Materials that can be applied to the ceiling or floor surfaces on the set, plus the props of the subtype *DanceFloor*.
- Asset Tags: a list of strings separated by commas. There will be a match if the name of any of the asset's tags contains any of the strings specified. Please note that only ***BodyParts***, ***Props*** and ***FiIters*** (a subtype of Cutting Room Assets) have tags associated.
- Extra Info: a list of strings separated by commas. There will be a match if the extra info about the asset tags contains any of the strings specified.

4. **Reset All**: clicking this button the addon and asset criteria will be cleared or reset to their default valued.
5. **Search** button: will start the search of assets according to the criteria established.
   When the search is complete, a new window with the search result will appear and a summary report will be written to the text box below.
6. Text box with a summary report of the assets found, with the number of addons and publishers they belong to.

3.2.1 Asset Search Result View

Once the Search for assets is complete, a new window will appear, showing the results of the search, according to the criteria specified:



Addon	Publisher	Asset Type	Subtype	Name	Tags
jm_CuckooClock	jamoram62	Prop	AutoAnimate	Cuckoo_Clock_Pendulum_NoSound	Prop jamoram62
jm_CuckooClock	jamoram62	Prop	Prop	Cuckoo_Clock_Handle	Props jamoram62
jm_Clocks	jamoram62	Prop	AutoAnimate	Digital_Street_Clock_Ads	Props
jm_CuckooClock	jamoram62	Prop	Prop	Cuckoo_Clock_Pendulum	Prop jamoram62
jm_CuckooClock	jamoram62	Prop	Prop	Cuckoo_Clock	Prop jamoram62
FunLand	ShortFuze	Prop	Prop	Cartoon_Stuff/Clock	cartoon
DomesticDrama02	ShortFuze	Prop	Prop	Electronics/Alarm_Clock01	Bedroom Gadgets Contemporar
jm_Wristwatch	jamoram62	Prop	Prop	Wristwatch_F	Prop
jm_Wristwatch	jamoram62	Prop	Prop	Wristwatch	Prop
1950sDomestic	Shirley Martin	Prop	Prop	1950sMantelClock	Prop Domestic House
1950sDomestic	Shirley Martin	Prop	Prop	1950sClock	Prop Kitchen
JAMM_Home	jamoram62	Prop	Prop	Clock_Wall_NoHands	Prop jamoram62
JAMM_Home	jamoram62	Prop	Prop	Clock Wall Round	Prop Domestic
JAMM_Home	jamoram62	Prop	Prop	Clock_HandHours	Prop jamoram62
jmAoM_Hospital	jamoram62	Prop	Prop	Clock Wall Round	Prop Domestic
jm_Clocks	jamoram62	Prop	Prop	Alarm_Clock	Prop
jm_Clocks	jamoram62	Prop	Prop	Digital_Street_Clock	Props
JAMM_Home	jamoram62	Prop	Prop	Clock_HandMinutes	Prop jamoram62
jm_Wristwatch	jamoram62	Verb	Held Prop	Wristwatch/Reset S20	
jm_Wristwatch	jamoram62	Verb	Held Prop	Wristwatch/Reset S25	
jm_Wristwatch	jamoram62	Verb	Held Prop	Wristwatch/Reset S30	
jm_Wristwatch	jamoram62	Verb	Held Prop	Wristwatch/Reset S35	

The most remarkable feature on this window is a data grid (table) with a number of rows, one for each asset found matching the search criteria. For each asset in the list some information will be displayed:

- Name of the addon that contains the asset

- Name of the Publisher of the addon
- Asset Type
- Asset Subtype
- Name of the asset
- Tags
- Additional information about the asset, depending on the type of the asset.
- Additional information about the addon containing the asset, including:
    - Free flag
    - Installed flag
    - Official content pack flag
    - Location of the addon

For a complete reference about the information offered in this listing, please refer to this page.

Assets can be sorted by most of the columns, by just clicking on the header of the column.

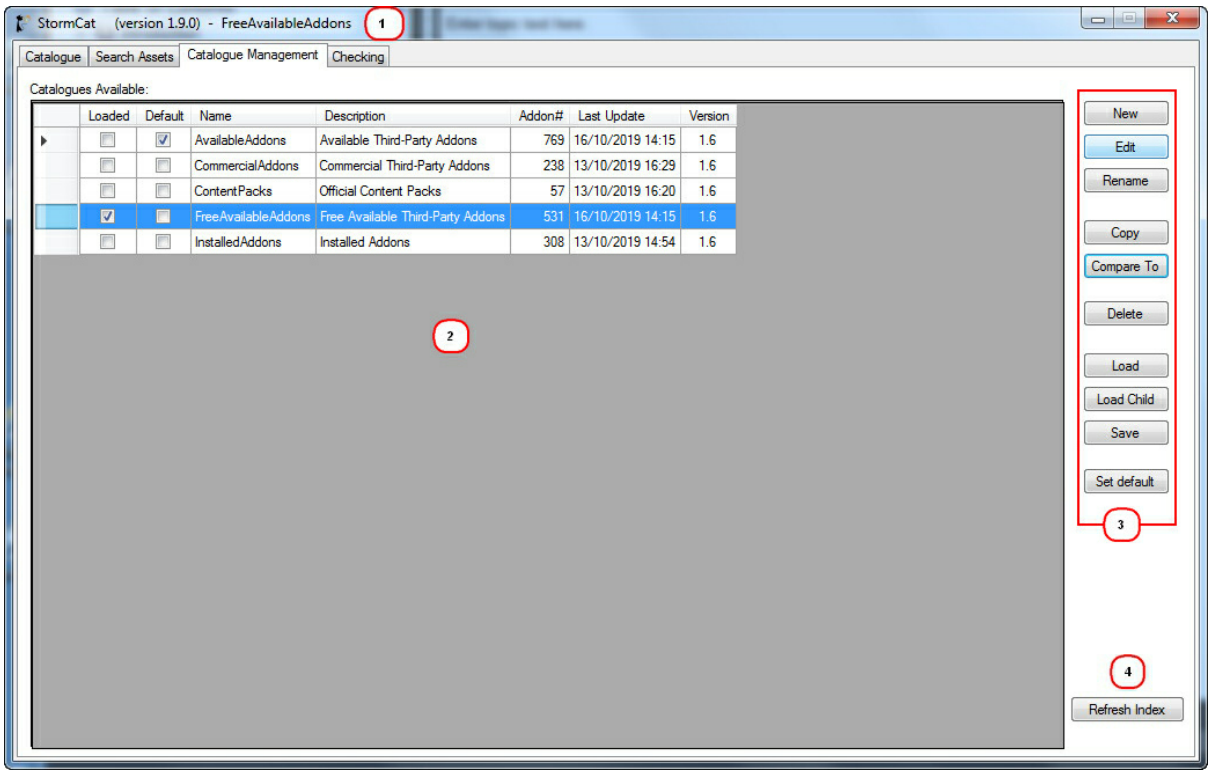
The table can be saved to a MS Excel workbook by clicking on the **Save to Excel** button.

Double-clicking on any of the asset rows, a new form will appear listing the contents of the addon the asset belongs to.

Also, by right-clicking on the table, a menu will pop-up with the following options:
- ***Display Report for the Addon***: a new window will appear with a full report about the addon in which the asset is included.
- ***List (table) contents of the Addon***: a new form will appear listing the contents of the addon the asset belongs to (ie, just like if you double-click on the asset row)
- ***Save To Excel File***: save the table to a MS Excel workbook.

3.3 Catalogue Management

In this tab of the main form, we'll be allowed to perform a series of operations on the Catalogues.



There a few remarkable interface items on this view:

1. As usual, the name of the current Catalogue is displayed on the title bar of the application
2. A data grid (table) with information about the Catalogues available and registered in the Catalogue Index. For each one of them some information will be displayed:
    - ***Loaded***: a flag indicating the Catalogue that's currently loaded. Of course, only one Catalogue can be loaded at any time.
    - ***Default***: a flag indicating the Catalogue that will loaded by default every time the application is started. Again, only one Catalogue can be designed as the default Catalogue.
    - ***Name*** of the Catalogue file
    - ***Description***: a brief description of the Catalogue contents or purporse
    - ***Number of addons*** in the Catalogue.
    - ***Date and time the Catalogue was updated*** for the last time.
    - ***Version*** of the Catalogue.
3. A series of buttons for performing a number of management actions on the Catalogue currently selected in the table:
    - ***New***: a new Catalogue, with no contents at all, will be created. A form will appear for setting the new Catalogue's name and description. The new Catalogue will become the active (loaded) one. BEWARE: any changes made on the previously loaded Catalogue won't be saved automatically.
    - ***Edit***: allows to modify the description of the Catalogue selected on the table. A form will appear for editing its description.
    - ***Rename***: for modifying the file name of the selected Catalogue. Again, a form will appear for editing its file name. In case the renamed Catalogue is the currently loaded, its name will be updated also in other items of the user interface

- ***Copy***: creates a copy (with its contents) of the selected Catalogue, under a new file name and with its own description. As usual, a form will appear for specifying the file name and description of the new Catalogue.
- ***Compare To***: will compare the addons contained in the current Catalogue with those in any other Catalogue available and selected in the table.
  If any difference is detected, a new form will be displayed, showing the details in the differences between the Catalogues compared.
  IMPORTANT: the comparison process will be made by matching the name and the publisher of the addons, ignoring any other data. Also duplicated addons in a Catalogue according to these criteria (ie, those with their names and publishers are identical) will be dropped before proceeding to the comparison between Catalogues.
- ***Delete***: will delete the currently selected Catalogue. It will ask for confirmation before proceeding.
- ***Load***: the selected Catalogue will be loaded, replacing the one currently loaded.
  The same result can be obtained by double-clicking on a Catalogue row in the table.
  BEWARE: any changes made on the previously loaded Catalogue won't be saved automatically.
- ***Load Child***: a new instance (Child) of the application will be launched, with the selected Catalogue loaded.
  Please note on the title bar of the newly launched copy of the application, will appear the text '[Child]', for denoting is a child instance, which comes with some limitations:
  - It can't modify the configuration of the application
  - Only the Catalogue and the Search Assets views are available (not Catalogue management nor Checking view).

  Despite their limitations, child instances can come in handy, for example, for comparing side-by-side two Catalogues or for copying contents between different Catalogues
- ***Save***: simply saves the Catalogue currently loaded.
- ***Set Default***: labels the selected Catalogue as the new Default Catalogue, ie, the one that will be loaded by default whenever the application is launched.

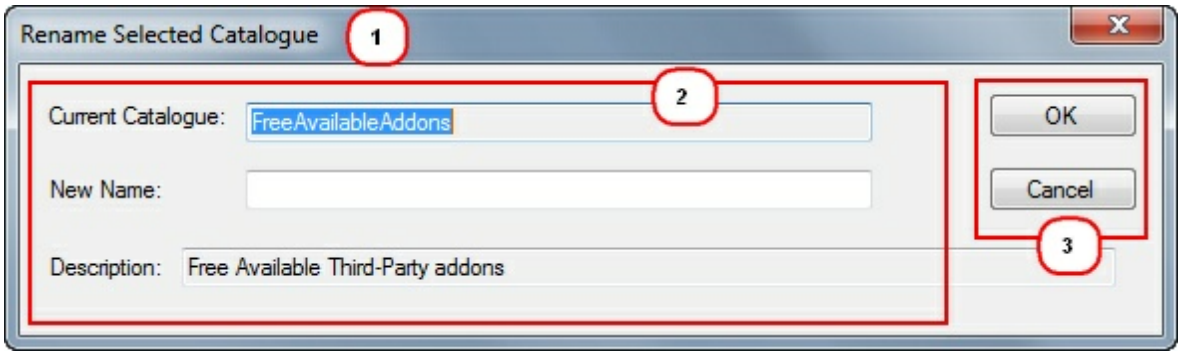
PLEASE NOTE: all these operations can be started also by right-clicking on the Catalogue table, via a pop up menu.

4. Finally, a button labeled **Refresh Index**, which will recreate the Catalogue Index, a special file which contains summary information about the Catalogues recognized by the application. This operation might eventually be necessary, in case that, for whatever reason, the Catalogue Index would become corrupt or if we manually added to our collection a new Catalogue that has not been created by any of the procedures described before.

3.3.1 Catalogue Operation Form

This form will appear whenever some of the Catalogue operations on the Catalogue management view is started.

It will looks something like that, although its actual aspect will vary depending on the operation invoked:



1. Name of the operation to be performed
2. A number of fields with information about the involved Catalogue(s). The actual fields and their associated labels will vary depending on the specific operation:

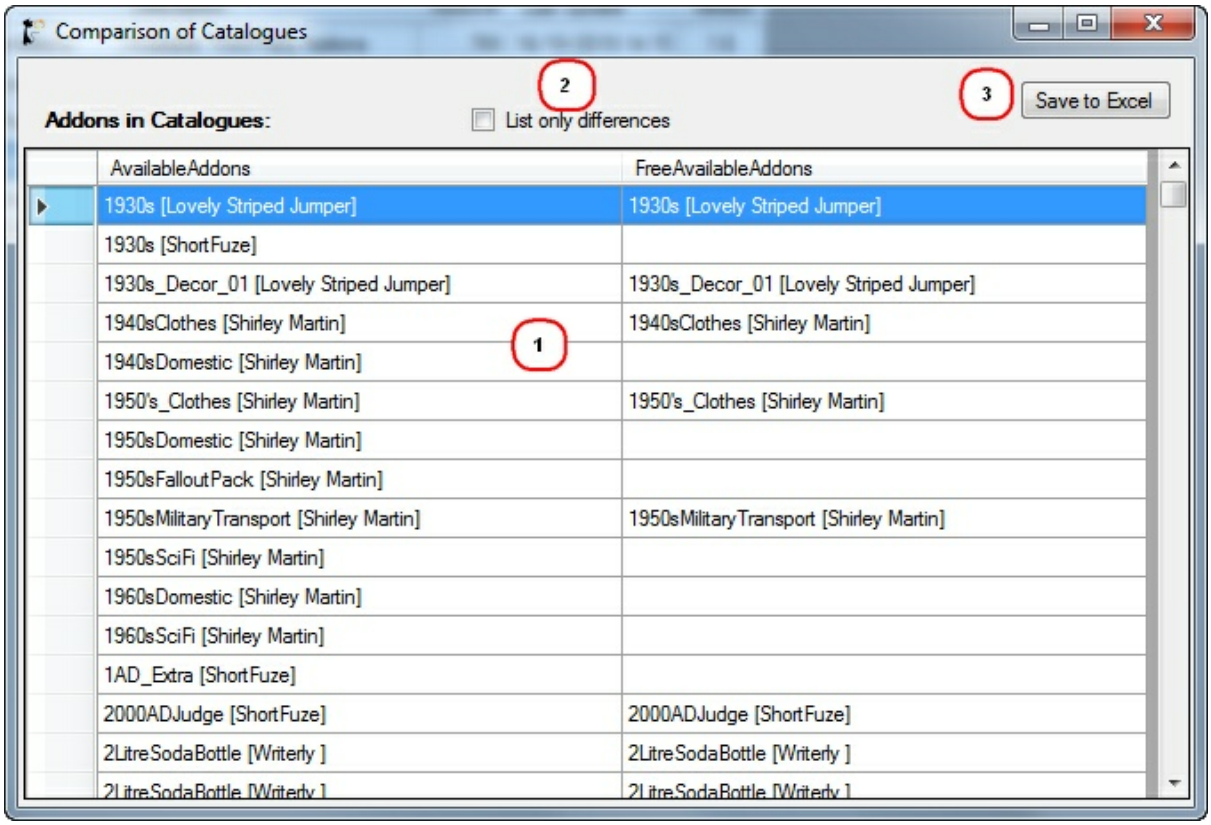
Operation	**Fields**		
New		New Catalogue	Description
Edit	*Current Catalogue*		Description
Rename	*Current Catalogue*	New Name	*Description*
Copy	*Current Catalogue*	New Catalogue	Description

NOTE: a field in *gray and italics* means it cannot be modified for this operation.

3. By clicking on the **Ok** button, the changes will be confirmed and the operation will executed. **Cancel**, on the other hand, will discard any change and the operation will be cancelled.

3.3.2 Comparison of Catalogues Form

This little form will display the differences between two Catalogues in relation to the addons contained in each one of them.

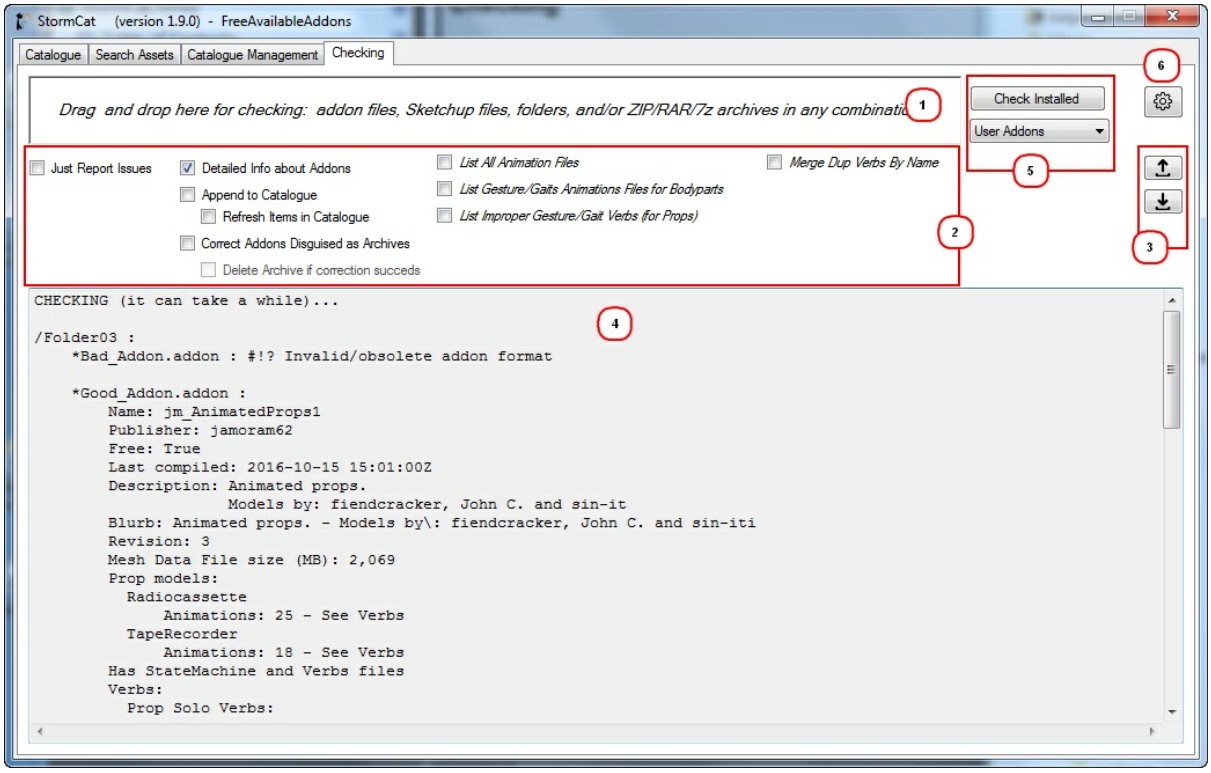


1. Differences between the Catalogues compared (their names appear at the header of each column) are listed on a **table** or data grid. Identical addons are listed paired together on the same row; addons present in one of the Catalogues but not found in the another one, will appear on its own row under the column corresponding to the Catalogue that contains it, while the other cell on the same row will appear blank.
2. Optionally, by checking this **checkbox** only differences found will be listed, ignoring the matches between the Catalogues compared.
3. The **Save To Excel** button will allow to save the current data in the table to a MS Excel workbook.

3.4 Checking

The controls on this tab allows the user to perform essentially checking operations on files and folders, plus some additional informative and corrective functionalities.

It looks like this:



1. A rectangular panel on which files and/folders on any combination can be dragged in and dropped onto for being checked.
2. A large number of options for directing the work of the checking engine. They will be discussed later
3. A couple of buttons for saving (below) the current configuration checking options to a file, so they will be restored automatically whenever the application is run again or manually whenever the restore button (above) is clicked.
   NOTE: if the application is passed some options in the comman line when launched, these options will take precedence on the saved option configuration.
4. A text box with the actual of output of the checking engine, including the issues found, a more or less description of the addons identified as valid and the result of some ancillary operations.
5. By clicking the **Check Installed** button the addons specified in the category(ies) selected (User Addons, Official content packs, everything) on the combo box below will be checked.
   PLEASE NOTE: for this operation be functional, it's required that the Moviestorm folders have been correctly established in the configuration of the application.
6. A button for editing the configuration of the application.

How the Checking Engine works

There are three ways to pass to the Checking Engine an arbitrary number of file and/folder specifications as its input:

1. as part of the command line

2. by dragging and dropping them on the application executable or on a direct link to the application executable.
3. by dragging and dropping them onto the rectangular panel provided for this purpose on the Checking tab.

The Checking Engine will act upon every one of the items passed, depending on its type:

- **Addon files** (files with an .addon extension): will check if it has a valid format, so it will work with the last version of Moviestorm (1.7.1). For being considered a valid addon it must contain a pair of specific meta-info files in its root folder.
  If the addon file doesn't meet the minimal validation criteria a message like this will be shown:
  ```
  *DoctorWho.addon : #!? Invalid/obsolete addon format
  ```
  In this case, was an addon published for running on old versions of Moviestorm
  The same problem has been observed in the case of addons that were published for the right version of Moviestorm, but that were incorrectly packaged.
  If the addon is valid, a description (more or less complete, depending on the options) of it and its contents will be written to the output:
  ```
  *jm_Dripping.addon : OK (incl. Movies)    [jamoram62]
  ```
  That's a very terse one-line report, but a very comprehensive report can be obtained by selecting the right options (see later)
  For a more complete discussion about the report for valid addons please refer to this.

- **Sketchup files** (files with a .skp extension): will be checked for their version. As of this date the Sketchup importing library of Moviestorm's Modder's Workshop only accept Sketchup files version 6.x (other Sketchup file versions will make the Modder's Workshop crash without a warning or message).
  An example of a valid Sketchup file:
  ```
  Good_Sketchup_File.skp : OK [6.4.112]
  ```
  And one that will lead to problems:
  ```
  Bad_Sketchup_File.skp : #!? Format not importable [17.0.1]
  ```

- **Archive files** (extensions: .zip, .rar or .7z): they will be inspected in the search of possible addon or Sketchup files contained inside them, which will be checked as described before. For example, a case of an archive containing two valid addons inside:
  ```
  +Writerly_DrWho Helmets.rar :
      *DoctorWhoDiver.addon : OK   [Writerly ]
      *WhoHelmet.addon : OK   [Writerly ]
  ```
  Archives will also be checked for the presence of some files which might lead to think thery actually are nothing but addon files "disguised" as archive files, ie, unorthodoxically package. In these cases, two possible messages can be written to the output.
  First, the case of possibly valid addon file with the wrong extension:
  ```
  +jm_CuckooClock.zip :
          #!? Possibly an Addon file disguised as an archive
  ```
  And now, a case a bit more pathological, in which the addon files are placed inside a folder in the archive:
  ```
  +jm_CuckooClockRooted.rar :
          #!? Possibly an Addon file disguised as an archive, with a
  root directory
  ```
  These cases can be easily corrected manually by the user, or automatically, by checking the correct option.
  PLEASE NOTE: a known limitation of the Checking Engine is that thus far it doesn't recursively inspect archive files, ie, should it find an archive inside the archive file is currently inspecting, it will be ignored.

- **Folders**: the very first thing the Checking Engine will do whenever it enters a new folder is checking

for the existence of a couple of files which could denote the current folder is actually an installed addon:

- If it happens to be a valid addon already installed, it will write a report about it, just like in the case of addon files:
  ```
  *1AD_Extra (installed) : OK   [ShortFuze]
  ```
- Otherwise, the folder will be scanned recursively in search of the types of files describe above.

Checking Engine Options

There are aplenty of options for modifying and directing the working of the Checking Engine:

1. **Just Report Issues**: if checked, the Checking Engine will report only possible issues it finds while validating the items passed as input, providing absolutely no information about valid items it could have find in the process. It's the most terse of the Engine's operating modes.
   In this mode, the only other options that can be activated is those related to the correction "disguised" addons (see 4.)
   Example:

```
CHECKING...

/Folder01 :
    +jm_FlamingHead_20170920.zip :
    +jm_FlamingHead_20170920.rar :

    /Folder11 :

/Folder03 :
    *Bad_Addon.addon : #!? Invalid/obsolete addon format
Bad_Sketchup_File.skp : #!? Format not importable [17.0.1]
+jm_AnimatedProps1.rar :
+jm_AnimatedProps1_20161015.zip :
+jm_AnimatedProps1_dir.rar :
+PapaG_Hat Hair ADDON_AS_ZIP.zip :
       #!? Possibly an Addon file disguised as an archive
+PapaG_Hat Hair ADDON_AS_ZIP_ROOTED.zip :
       #!? Possibly an Addon file disguised as an archive, with a
root directory
*PapaG_Hat Hair BAD.addon : #!? Invalid/obsolete addon format

/CPacks :

*** PROCESSING FINISHED ****...
```

2. **Detailed Info about Addons**: if unselected, the report provided for a valid addon is pretty terse, eg:
   ```
   *jm_AnimatedProps1.addon : OK   [jamoram62]
   ```
   However, if this option is checked, a quite complete report is generated about the addon and its contents:

```
*jm_AnimatedProps1.addon :
    Name: jm_AnimatedProps1
    Publisher: jamoram62
    Free: True
    Last compiled: 2016-10-15 15:01:00Z
    Total files: 123
    Description: Animated props.
```

```
                Models by: fiendcracker, John C. and sin-it
    Blurb: Animated props. - Models by\: fiendcracker, John C. and
sin-iti
    Revision: 3
    Mesh Data File size (MB): 2,069
    Prop models:
      Radiocassette
          Animations: 25 - See Verbs
      TapeRecorder
          Animations: 18 - See Verbs
    Has StateMachine and Verbs files
    Verbs:
      Prop Solo Verbs:
          [Radiocassette] : Radiocassette/Fast Forward Loop (no
foley)
          [Radiocassette] : Radiocassette/Fast Forward Loop
          [Radiocassette] : Radiocassette/Fast Forward Start (no
foley)
          [Radiocassette] : Radiocassette/Fast Forward Start
          [Radiocassette] : Radiocassette/Fast Forward Stop (no
foley)
          [Radiocassette] : Radiocassette/Fast Forward Stop
          [Radiocassette] : Radiocassette/Hide tape
          [Radiocassette] : Radiocassette/Lid Close (no tape)
          [Radiocassette] : Radiocassette/Lid Close
          [Radiocassette] : Radiocassette/Lid Open (no foley)
          [Radiocassette] : Radiocassette/Lid Open (no tape)
          [Radiocassette] : Radiocassette/Lid Open (no tape, no
foley)
          [Radiocassette] : Radiocassette/Lid Open
          [Radiocassette] : Radiocassette/Play Loop
          [Radiocassette] : Radiocassette/Play Start (no foley)
          [Radiocassette] : Radiocassette/Play Start
          [Radiocassette] : Radiocassette/Play Stop (no foley)
          [Radiocassette] : Radiocassette/Play Stop
          [Radiocassette] : Radiocassette/Rewind Loop (no foley)
          [Radiocassette] : Radiocassette/Rewind Loop
          [Radiocassette] : Radiocassette/Rewind Start (no foley)
          [Radiocassette] : Radiocassette/Rewind Start
          [Radiocassette] : Radiocassette/Rewind Stop (no foley)
          [Radiocassette] : Radiocassette/Rewind Stop
          [Radiocassette] : Radiocassette/Show tape
          [TapeRecorder] : TapeRecorder/Lid Close
          [TapeRecorder] : TapeRecorder/Lid Open
          [TapeRecorder] : TapeRecorder/Play Loop
          [TapeRecorder] : TapeRecorder/Play Start (no foley)
          [TapeRecorder] : TapeRecorder/Play Start
          [TapeRecorder] : TapeRecorder/Play Stop (no foley)
          [TapeRecorder] : TapeRecorder/Play Stop
          [TapeRecorder] : TapeRecorder/Rec Loop
          [TapeRecorder] : TapeRecorder/Rec Start (no foley)
          [TapeRecorder] : TapeRecorder/Rec Start
          [TapeRecorder] : TapeRecorder/Rec Stop (no foley)
```

```
            [TapeRecorder] : TapeRecorder/Rec Stop
            [TapeRecorder] : TapeRecorder/Rewind Loop (no foley)
            [TapeRecorder] : TapeRecorder/Rewind Loop
            [TapeRecorder] : TapeRecorder/Rewind Start (no foley)
            [TapeRecorder] : TapeRecorder/Rewind Start
            [TapeRecorder] : TapeRecorder/Rewind Stop (no foley)
            [TapeRecorder] : TapeRecorder/Rewind Stop
      Sounds:
        Foley/Props/Tape/Tape_click.mp3
        Foley/Props/Tape/Tape_click2.mp3
        Foley/Props/Tape/Tape_double_click.mp3
        Foley/Props/Tape/Tape_fast.mp3
```

Please refer to the Addon Report Reference for more detail about the structure and meaning of the addon detailed report.
As soon the detailed reports are enabled, a new series of additional options become available for more finely tuning the output of the description report:

- ***List All Animation Files***: list all animation files declared in the Manifest file of the addon. By default, only animation files which are not referred to in the StateMachine or Verbs files will be listed.
- ***List Gesture/Gaits Animations for Bodyparts***: list every animation file for the puppets' Gesture and Gaits declared in the Manifest file of the addon, regardless they are referred to in the StateMachine files will be listed.
- ***List Improper Gesture/Gait Verbs (for Props)***: list 'verbs' for Gestures or Gaits associated to props (gestures and gaits should be properly associated only to puppets). Most probably they are spurious entries left behind by the developer of the addon and can be ignored, as far as they don't lead to problems.
- ***Merge Dup Verbs By Name***: merge in the report those Verbs with the same name, even if the names of their animation files differ.

3. **Append to Catalogue**: will append those addons which are checked as valid to the current Catalogue, provided it's not already registered.
   If the related option **'Refresh items in Catalogue'** is checked, in case the addon has been already registered, will be updated.
4. **Correct Addons Disguised as Archives**: if this option is active (checked), whenever the Cheking Engine finds an archive which has proved to be a valid addon "disguised" (incorrectly packed as an archive, with the wrong file extension; see above), the correct addon file will be created.
   If the related option **'Delete Archive if correction succeeds'** is active, in case the addon file is created, the old archive file will be deleted.
   For example, in the case described before:

```
+jm_CuckooClock.zip :
        #!? Possibly an Addon file disguised as an archive
```

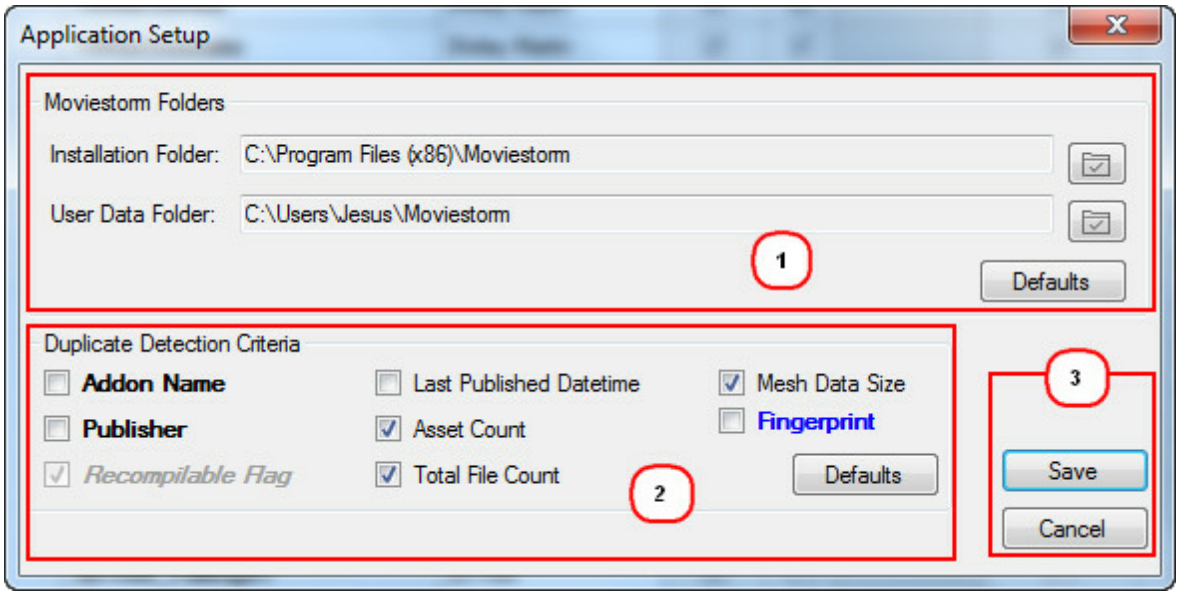
   if the **Correct Addons Disguised as Archives** is active, the output will read:

```
+jm_CuckooClock.zip :
        Addon file disguised as an archive. Restored: jm_CuckooClock.
addon
```

4 Setup Form

This form allows the user to configure the behaviour of the application.

You'll find these features on it (they may vary between different versions):



1. ***Moviestorm folders***. They are needed to search for the official content packs (*Installation folder*) and third-party addons (*User Data folder*) currently installed. By clicking the **Defaults** button, the application will try to find those folders by itself, but if you installed Moviestorm on a location other than the default, it may fail to work. In any case, each of these folders can be selected manually by clicking on the little button with the folder icon.

2. ***Duplicate Detection Criteria***. A series of checkboxes are provided for the user to choose the criteria used to detect possible duplicates among the addons in her Catalogues. Take into account that by selecting a larger number of criteria, the lesser the probability of a false positive (ie, erroneously identifying two addons as the same addon), but it may increase the number of false negatives (ie, erroneously not identifying two addons which are in fact duplicate). By clicking on the **Defaults** button, the default criteria will be restored.
   The criteria available as of this date are:
   - Addon name
   - Publisher name
   - Recompilable flag (can't be deselected)
   - Last published datetime
   - Asset count
   - Total file count
   - Mesh data size
   - Fingerprint *(introduced in version 1.10.0)*: whenever this item is selected, any other one else will be deselected and unavailable.

   For a discussion about duplicate addons and their detection, please refer to this section.

3. Button **Save**: confirms the changes made to the configuration and saves them to file to be used in the future, and then return to the main form.
   Button **Cancel**: discard any change made to the configuration and return to the main form.

5 Appendixes

A few topics are covered in the following sections, including:

- Structure of a `Moviestorm` Addon
- `Moviestorm` Stuff
- Addon Report Reference
- Asset List Reference
- Duplicate Addons
- `Notes.txt` file

5.1 Structure of a Moviestorm Addon

A typical Moviestorm Addon is made of a number of files and folders with very well established and mostly rigid structure, which can be abstracted something like this:

```
\ (Addon root folder)
|   .AddOn
|   .properties
|   assetData.jar
|   meshData.data
|   meshData.index
|   thumbnail.jpg
|   version.xml
|
+---Data/
|
+---Movies/
|
\---Stock/
```

Inside the root folder of the addon there are a few files that bear more or less important meta-information about the addon. The two most important of these files are:

- **`.AddOn`** or signature file: which contains information about the:
  - Name of the addon
  - Publisher of the addon
  - Whether the addon is free or a license is required to use it
  - List of files in the addon
  - A cryptographic signature specific for the publisher, validating the addon as genuine.

- **`assetData.jar`** or Manifest archive: which contains:
  - An asset manifest, with information about every bodypart and prop in the addon, together with its associated elements (skeletons, animations, parts, puppet, parts, etc)
  - A collection of files describing the bodyparts and props included in the addon (DESCRIPTOR, templates, parts)

Along with these major meta-files, there are other few which also convey meta-information, although they're of lesser importance:

- **`.properties`**
- **`version.xml`**
- **`thumbnail.jpg`**: which is completely optional and it's not present in most of the third party addons
- **`notes.txt`**: an unofficial file that can be inserted in the the root folder of the addon for conveying additional information, not included in any of the official files. For a complete description of this file please refer to this section.

Also in the root folder there a couple of files which will present whenever the addon include some "physical" assets, ie, assets with a mesh, such as props and bodyparts: `meshData.data` and `meshData.index`. These file contain information about the meshes of every one of these assets, compiled in a compact way. They may be absent if, for example, an addon only defines new animations.

Finally a number of folders may be found:

- **`Data/`**: by far, the most important of them (it will rarely be absent), it's the root for the actual contents of the addon. You may find a couple of files and lots of folder inside the Data folder:

```
\(Addon root folder)
|...
|
+---Data/
    |   StateMachine
    |   Verbs
    |
    +---fx/
    |
    +---Materials/
    |
    +---Props/
    |
    +---Puppets/
    |
    +---Sky/
    |
    +---Sound/
```

- `StateMachine`: defines a number of machine states and the transitions for every animation file in the addon (except of the animation for AutoAnimate props). It also defines the Verbs for gestures and gaits.
- `Verbs`: defines the Verbs for most of the animations (except of gestures and gaits).
  NOTE: generally the `StateMachine` and `Verbs` files can be found side by side in an addon, along with the animation files they refer to. However sometimes this is not the case:
  - `StateMachine` file exists, no `Verbs` file: that uses to be the case when only puppet Gestures and/or Gaits are contained in the addon, for these types of animations don't need their verbs defined in the `Verbs` file.
  - Most of the verbs for animations in the official content packs can be found in a huge `Verbs` file inside the **Core** pack. Whatever scenario/combination you might think of can be found in the official content packs:
    - sometimes you'll find packs with lots of animation files and no `StateMachine` or `Verbs` file at all
    - some packs have a `Verbs` file, but no `StateMachine` file.
    - etc
- A series of folders where the individual assets and the files they are made from are located. The exact folder and the arrangement of the files and subfolders inside them, will depend on the asset type. The most relevant are these:
  - `Puppets`/ : contains all the assets related to the puppets (characters), separated by models (`Female01`, `Male01`, etc), ie:
    Bodyparts (along with their texture files, and possibly their Cal3D mesh and material files)
    Decals
    Animations, of different types: gaits, gestures, puppet solo postures, held props and interactive animations (with wither props or other puppets)
  - `Props`/: contains the prop models and their related files:
    Prop description files, along with their texture files, and possibly their Cal3D mesh,skeleton and material files.
    Animations, of different types: prop solo, held props and interactive prop animations
  - `Materials`: textures which can be applied to the different surfaces on the set (walls, floor, ceiling, etc)

- `Sound`: sound effects, music.
- `fx`: particle visual effects
- `Sky`: textures that can be applied to the sky in the Set Workshop

There exist other folder for some more exotic asset types.

- `Movies/`: here is where demo movies live, if any.
- `Stock/`: the home of the Stock assets.

5.2 Moviestorm Stuff

A wide range of different types of assets can be found inside Moviestorm addons, which can be used for producing our movies.

In this section will describe every kind of Moviestorm asset and what relevant information is associated to each one of them. But, before delving into each of them, I think a short introduction about the most definitely essential item in any movie.

Moviestorm Puppets

Puppets are the actors in our Movistorm movies. Currently, there are only two official models of puppets available:

- **Female01** - the base on which every female actual character is modeled or built on
- **Male01** - the base for male characters

Every Puppet model is defined essentially by its own specific ***Puppet Skeleton***, made up from a number of bones and which allows a puppet to be animated.

Puppets are complex entities, for they are made from a few types of assets we'll see later (Body Parts and Decals), which are associated to a specific model of puppet in such a way so there's not a straightforward way to use an asset specifically created for a puppet model with another different model.

Also, animations other that pure solo prop animations are somehow associated as well to a specific model of puppet, so each and every puppet model comes with its own set of animations, which can't be easily applied to another model (for their skeletons are different).

Types of Assets

So let's see what are the different types of assets a movie is made from in Moviestorm:

Body Parts

Decals

Props

Animations and Verbs

Materials

Sounds

Special Effects

Filters

Sky Textures

Stocks

Demo Movies

Other (not part of Addons)

5.2.1 Body Parts

Body Parts are the main components our actual characters are made of.

There are a few subtypes of Body Parts:

- **Heads**: they're are... well the heads of our characters. They can be (and use to be) morphable.
- **Bodies**: they represent not only the torso and limbs of the puppets themselves, but also the garments they're wearing, including the footwear and possibly gloves and other complements. Although they can be made morphable, most of the bodies available aren't.
- **Hair**: most diverse hairstyles, combined or not with hats, caps and possibly other hair ornaments. Many of them are morphable.
- **Accessories**: everything else the actors can wear. An actor can be applied an undefined number of Accessories. Too bad, Accessories can be made neither morphable nor animated.

A few considerations to take into account in relation to Body Parts:

1. Every puppet model comes with its own set of body parts, which are rigged (bound) to the bones of the model's skeleton, so there's not an easy way to use a body part: trying to do so would require a very involved and time-consuming process of re-rigging the mesh of the body part to the skeleton of the destination puppet model (only Accessories use to be easily copied to another puppet model, for they usually are bound to a very low number of bones, mostly just one).
2. Most of subtypes of Body Parts, with the only exception of Accessories can be made morphable, that is, their dimensions and/or positions can be adjusted inside some limits defined by the creator of the part.

5.2.2 Decals

Decals are essentially texture maps which are applied to some parts of the puppet's body surface (ie, head and/or body)

There are the following Decal subtypes:

- ***Beards***: mostly found applied to male puppets, but there's no
- ***Effects***: ie, scar, bullet holes, wounds, blood, etc
- ***Eyebrows*** (female and male)
- ***Makeup***, ***Makeup Presets*** and ***Makeup Tints***
- ***Skin***: base skin texture for a specific gender, age and race. They are applied to both the head and the body of the puppet.
- ***Skin Detail***: such as freckles, blush spots, sunburns, red eyes, etc

Some of these textures can be resized, repositioned, rotated and/or applied different colors.
Apart from the diffuse map (ie, color), Decals may come with a other types of texture maps (specular

and normal, but no emissive) for better effect, but UV animations won't work with them.

5.2.3 Props

Props are any object used on stage screen by our actors during their performance. They are probably the most important category of assets, second only to the characters themselves.

There are number of subtypes of Props, according to some special characteristics or capabilities:

- ***Prop***: these are ordinary props, nothing special about them. They can be static or animated. They can be animated autonomously and/or interact with the characters.
- ***AutoAnimate***: props which, as soon they're placed on the set, start to execute a predefined animation that is repeated in an endless loop as long the prop stays on the set.
- ***Car***: mobile props with lights that can be turned on and off and whose wheels do spin and veer automatically while the prop is moving on the set.
- ***Door***: a very special type of prop that can only be placed on a section of wall and that, unfortunately to a somehow limited extend, can interact with characters.
- ***Window***: another type of props that can only be placed on a section of wall
- ***Doorway***: ditto.
- ***Seat***: a category of props characters can sit on by means of a predefined. They can be animated and/or mobile (eg, wheelchair)
- ***DoubleBed***: a more specialized and extended variant of the Seat subtype, with additional specific interactive animations.
- ***Lamp***: props which can act as a configurable light source that can be turned on and off.
- ***FollowSpot***: a variant of Lamp which can act as a source of a directed beam of light that can be instructed to follow a character or another prop.
- ***LightingRig***: a very specialized variant of Lamp, with multiple sources of light, which can be programmed to create very complex lighting patterns.
- ***PyroProp***: props with one or more Special Effects bound to it.
  In the case of PyroProps designed as held props, their bound SFXs are turned on and off by using specific animations.
  On the other hand, no-held PyroProps, as designed by Moviestorm's developers, trigger their bound SFXs as soon they are placed on the set, although their SFXs can be turned on and off by means of animations to this effect. However, their handling on the set can become a real annoyance, because Moviestorm use to lose track of the SFX's state too often. Worse, if a PyroProp is happened to be hidden, as soon it's made visible again the associated SFXs will be turned on so it won't be possible to turn them off!
- ***Weather***: a complex variant of PyroProp which allows to configure the characteristics of particles falling on the set, simulating rain, snow falling, etc. It also pretends to simulate the effect of wind on those particles, although, as a matter of fact, there's no physics implemented in Moviestorm.
- A series of other special variants of Pyroprops:
  Explosive
  Fireplace
  FlameBar
  FlashPot
  SmokeBar
  RocketLauncher
- ***LoudSpeaker***: props that can play audio files, acting as sound sources on the set.
- ***DanceFloor***: a subtype of props which attaches to the floor, with sections of different colours, flashing at a configurable rate.
- ***DrumKit***: well, a specific type of interactive prop.

5.2.4 Animations and Verbs

Animations are... You must know very well by now what animations are: changes more or less transient in the conformation (shape or relation between the parts) and/or position of a character or prop. In Moviestorm animations come actually implemented as Cal3D animation (`.caf`) files. Moviestorm does only supports skeleton animations, ie, based on the application of transformations on the bones of a skeleton a character's body part or prop mesh is rigged to.
(Pedantic note: there's something called UV Animation, but that's a completely different beast and has nothing to do with the animations we're interested in).

Verbs, on the other hand, are the diverse menu items on the user interface the user select to trigger those animations. Verbs are contained in the most diverse menus, depending on the kind of animation they are associated to, ie, postures (solo puppet animations), gestures, solo prop animations, held prop animations, mutual animations (between two puppets), walking gaits, etc. Verbs are defined essentially in two special files: `StateMachine` and `Verbs`, depending on the kind of animations.

Disagreement between Animation and Verb counts

There is not a one-to-one correspondence between verbs and animations:
- Every Held prop verb use to be associated to two animations: one for the puppet and another for the prop
- The same goes for animations of interactive props, although there are times that one verb is only bound to a prop or a puppet animation. And this is the case also for mutual verbs (interactions between two puppets)
- Autoanimate props have a predefined animation associated, but there's no verb at all, for in this case the animation is started as soon the prop in placed on the set.

As a rule of thumb, verbs are defined in the same addon along with their related animations, but that's not necessarily the case. The most remarkable exception to this rule can be found in the official content packs. For whatever reason, the designers of Moviestorm decided to put the definition of most of the verbs (or, at least, the `StateMachine` part of their definition) for every official content pack inside the `Core` pack, while keeping the actual animation files as part of the packs they belong to.

So, for the reasons stated above, you must not get puzzled if you the counts of animations and verbs in an addon don't match at all.

5.2.5 Materials

Materials are textures applied to different surfaces on the set, ie:
- Ceiling
- Floor
- Walls
- Terrain

Apart from a diffuse map (ie, color), they can include other kinds of texture maps (normal, specular and emissive).
NOTE: so far, I don't know whether UV animations would work with Materials

5.2.6 Sounds

Sounds are... well, sound files, which are played under different circumstances, either:
- as part of an animation or
- explicitly at the movie producer's discretion, for creating a specific ambience, as foley sounds or music.

5.2.7 Special Effects

In the context of Moviestorm, Special Effects (SFX) refer to particle-emitting visual effects made of particles (actually 2D images), whose colour, shape, density and other parameters are defined by the SFX definition.

SFX are not autonomous assets, ie, they can't be put directly on the set by the user, but they're associated or bound to a series of subtypes of PyroProps, which in some cases allow the user to control some of the characteristics of those particles. They also use to be associated to a lighting effect, whose quality and intensity can also be modified by the user in some cases.

5.2.8 Cutting Room Assets

A series of assets which are used or applied during the cutting process. Three subtypes:
- ***Filters*** are applied in the Cutting Room on specific parts or to the full movie clip as a whole for creating a special visual effect.
- ***Text styles***: ie, titles, subtitles, credits, etc
- ***Transitions***: it seems they were planned as a featured, but AFAIK they are not actually supported in the last version of Moviestorm.

5.2.9 Sky Textures

Diffuse texture maps specially designed to be applied to the sky dome on the set.

5.2.10 Stocks

Stocks are made of a collection of other assets which are saved to disk, into a Stock library, in order to be used in a number of different movies.

There are basically two kinds of stocks:
- Stock characters
- Stock sets

5.2.11 Demo Movies

The title says it all. Actually, Moviestorm puts them into (at least, AFAIK) subcategories:
- Starter Movies
- Background Movies

5.2.12 Other (not part of Addons)

In this section a series of predefined assets which usually are not to be found as part of a content pack or addon and that play a major role in the production of a movie:

- Set elements: walls, ceiling, terrain, general lightning
- Other: cameras, cutting room assets

Some of those assets (walls, terrain) can be modified by the user, to her own risk, by editing the movie definition file.

Finally, there are other kinds of assets that might be part of new addons (although I haven't seen any of them so far), such as:

- ***Terrain Masks***. The very only actual example of this kind of asset can be found as part of the `Core` official content pack.
- ***Scenarios***, which define a series of elements of the different views of the GUI of Moviestorm. So far, you can only found some of them in the `Core` and `Base01` official content packs.

5.3 Addon Report Reference

Reporting Bad Addons

In the case a checked addon file does not meet the requirements to be used with Moviestorm 1.7, a brief error message will be written to the output:

```
*BongoMan_Guns_Swords.addon : #!? Invalid/obsolete addon format
```

They most probably are very old addons that were created for versions 1.4 or prior (not sure about when took place the change of format), so they need to be republished, preferably by their original author.

There are other possibilities while checking for the validity of addons, such as:

```
+jm_CuckooClock.zip :
        #!? Possibly an Addon file disguised as an archive
```

or

```
+jm_CuckooClock.rar :
        #!? Possibly an Addon file disguised as an archive, with a root
directory
```

In both cases, while checking the contents of the archives (`jm_CuckooClock.zip` and `jm_CuckooClock.rar`), **`StormCat`** has found a hierarchy of files and folders strongly suggesting there's a valid addon inside, but they have improperly been archived instead of being packaged into an `.addon` file (the container format Moviestorm recognizes).
These cases can be easily corrected by hand by the user or by using the options in the Checking page:

- *Correct Addons Disguised as Archives*
- *Delete Archive if correction succeeds*

Brief Report

In the Checking Page, if the ***Detailed Info about Addons*** is not selected (checked), a quite terse one-liner with only the most basic information about the addon will be generated.

The most basic example, only the name of the addon and the publisher will be written to the output, and will look something like this:

```
*1AD_Extra : OK   [ShortFuze]
```

`1AD_Extra` is the official name of the addon
`[ShortFuze]` is the name of the publisher
`OK` meaning the addon has the right format to be used with Moviestorm 1.7

Sometimes, some additional information will be also written:

```
*3DTree_Animals03  (installed) : OK (incl. Movies)   [3DTree  NOT
FREE!]
```

`(installed)` means the addon is currently installed
`(incl. Movies)` denotes the addon comes with one or more demo movies (Starting or Background)
`NOT FREE!` implies that a valid license is required for the addon being used

Other possible information items you might find are:

`no meshes` means the addon does not contain assets with a mesh (ie, props and body parts)
`(incl. Stock assets)` : the addon includes some ready-to-use stock set or character, which can be imported and used in your own movies.

Detailed Report

The structure and contents of a detailed addon report includes the following items:

1. **Name:** official name of the addon, as found in the addon signature file, regardless of the name of the folder containing the addon when installed.

Name: 1950sDomestic

2. ***Issue warning***: in the case of some issue or problem has happened while scanning and analyzing the addon, a line will appear with this text:

```
!Has Problems - refer to the end of the report...
```

Seeing this line means bad news, for it uses to indicate that the application has met something unexpected which it can't deal with. Blame the stupid developer!

3. **Publisher**: name of the account of the author of the addon (or the last user publishing it). It's also obtained from the addon signature file.
Please note that all official content packs are published by **`ShortFuze`**, but there are also a good deal of third-party addons which are published under that account (eg, addons created by Chris Ollis and available for purchasing on its site http://www.moddingstorm.com)

Publisher: Shirley Martin

4. **Free**: indicates whether the addon containing the asset is free () or requires a license to be used ().

```
Free: True
```

5. **Last Compiled**: Date and time the addon was published for the last time. It's the modified timestamp of the addon's manifest archive

```
Last compiled: 2012-03-08 12:46:58Z
```

6. **Total Files**: total count of files in the addon

```
Total files: 337
```

7. **Description**: of the addon, if any

```
Description: 1AD Extra
             Copyright Chris Ollis 2011
             www.moddingstorm.co.uk
```

8. **Blurb**: additional descriptive notes

9. **Revision**: number of version of the addon

```
Revision: 5
```

10. **Fingerprint**: a hash code computed by **`StormCat`** (version 1.10.0 and later) while generating the detailed report, which can be come very handy for identifying duplicate addons

```
* Fingerprint:
a2c1d646aaa35ebc10475005848bebc17839127644010d44f9bf0eb29abf0fd4
```

11. **Mesh Data File size (MB)**: the size in megabytes of the `meshData.data` file, which contains the data for the meshes of every body part and prop in the addon. Eg:

```
Mesh Data File size (MB): 18,439
```

If an addon does not includes any asset with a mesh (body part or mesh), no `meshData.data` file will be created when publishing the addon, so a message like this will be displayed instead:

```
No meshes
```

Finally, sometimes there addons in which the Cal3D (mesh) files for the props have been deleted and the information about their meshes are contained exclusively in the `meshData.data` file (which it's just what Moviestorm needs to work). This way, the `meshData.data` file can't be rebuild and the addon can't be republished, which is a common occurrence with official content packs and some third-party addons (especially those requiring a licence):

```
NOTE: Addon can't be republished (no Cal3D mesh files found)!
```

12. **Thumbnail image**: it does simply mean that the addon comes with a little thumbnail image.

```
Has Thumbnail image
```

13. **Demo Movies**: should the addon include one or more starting or background movie, they will be listed here:

```
* Includes Demo Movies:
    'Downtown Background Day'
    'Downtown Background Night'
    'Downtown Surprise Day'
    'Downtown Surprise Night'
```

14. **Stock assets**: if the addon comes with some stock items ready to be used, they are listed, indicating whether they're stock characters or sets

```
* Includes Stock assets:
    'Character:Barack Obama'
    'Character:George W Bush'
    'Character:Joe Biden'
    'Set:Oval Office'
    'Set:Political Podiums'
    'Set:Press Room'
```

15. **Body parts**: for each model of puppet found in the addon, information about their associated body parts, decals and animations will be listed, eg:

```
Body parts:
  Female01:
      BodyParts:
         Fembot Head MkII (HEAD)
         Fembot Head (HEAD)
      Decals: Only extern Decals referenced
      Animations (non-gesture/gaits): 7 - See Verbs
  Male01:
      BodyParts:
         Malebot Head Mk II (HEAD)
         Malebot Head (HEAD)
      Decals: Only extern Decals referenced
      Animations (non-gesture/gaits): 7 - See Verbs
```

Body parts are grouped by the puppet model they can be applied to (usually `Female01` or `Male01`, ie the two only official models, but other models created by third-party modders can also be found - most remarkable of them is 3DTree, Trevor, and his `3DT_Chicken01`)
Let's see each one of these type of assets:

- BodyParts: parts puppets are made of (please refer to this section for a description and discussion about BodyParts).
  For every BodyPart found, it will be listed its name and its type. Also if the part is morphable, it will be indicated:

```
Female01:
    BodyParts:
```

```
AVK Happy Head (HEAD) [MORPHABLE]
Monster Head Alien (HEAD)
Toga Undershirt (BODY)
Morph Short Sleeve Shirt n Jeans (BODY) [MORPHABLE]
AVK Hair'n'Hat Asian Parting (WIG) [MORPHABLE]
Hair Curly Long (WIG)
Bowler Hat (ACCESSORY)
Cigarette Smoked (ACCESSORY)
Cigarette (ACCESSORY)
```

- Decals: Decals are listed after the Body parts (if any), with its name and its group/subtype:

```
Female01:
    Decals:
        Bullet Hole HQ (Effects)
        Eyebrows 01 HQ (Female Eyebrows)
        Facepaint Cat HQ (Makeup Presets)
        Blush HQ (Skin Detail)
        Afro Old 1 HQ (Skin)
```

  In the event that only external decals are referenced (ie, decals whose texture files are not part of the addon itself), this circumstance will be noted (otherwise, they will be ignored):

```
Decals: Only extern Decals referenced
```

- Animations: should a puppet model in the addon come with one or more animations file associated, they will be referrenced (for a discussion about the relation between verbs and animation, please refer to this section).
  - By default, actual names of the animation are not listed, but a simple count of the them is shown instead:

```
Body parts:
  Female01:
      Animations: Gesture/Gaits: 10   Other: 39 - See Verbs
  Male01:
      Animations: Gesture/Gaits: 12   Other: 39 - See Verbs
```

    Please note that for puppet models, gesture and gaits animations are tallied up separate of other types of animations (puppet solo, held and interactive prop, and mutual interaction animations).
  - If you choose to check the ***List Gesture/Gaits Animations Files for Bodyparts*** option, the actual names of the gait and gesture (with their actual path in the Gestures menu) will be listed, but any other type of animation will only be summarized:

```
Body parts:
  Female01:
      Animations:
         Gestures/Salutes/Female_Fascista_Salute_Start
         Gestures/Salutes/Female_Fascista_Salute_Stop
         Gestures/Salutes/Female_Marxist_Salute_Start
         ...
         Gestures/Salutes/Female_Vulcan_Salute_Start
         Gestures/Salutes/Female_Vulcan_Salute_Stop
         Non-gesture/gaits animations: 39 - See Verbs
```

  - Finally, if the ***List All Animation Files*** option is checked, the names of each animation file associated to the puppet model will be listed, regardless of their types:

```
Body parts:
  Female01:
      Animations:
         Chair/Sit_Legs_Crossed
```

```
Chair/Sit_Legs_Crossed_Recover
...
Gestures/Salutes/Female_Fascista_Salute_Start
Gestures/Salutes/Female_Fascista_Salute_Stop
Gestures/Salutes/Female_Marxist_Salute_Start
...
JammMisc/Female_Snapping_Heels
JammMisc/Kneel_Both2Right_start
SideStep/sidestepleft
Walking_In_Place/walk_loop
...
```

16. **Prop models**: prop models defined in the addon are listed identified by their model name (including their path inside the `Props/` folder), eg:

```
Prop models:
  Furniture/Sofa_Twoseater_Classic_01
  1930s_pelmet_blind
  1930s_coal_skuttle
  Dressing/Rugs_Lounge/Rug_01
  Flora/PotPlants/LargePotPlant/LargePotPlant_Narrow
```

Should the prop have any other special property, it will be displayed, including:

- Prop subtypes, other than ordinary props (AutoAnimate, PyroProp, etc):

```
Hover_Bike_01 (PyroProp) [Mobile Navigable] [Multipart]
Cuckoo_Clock_Pendulum_NoSound (AutoAnimate)
Dressing/Decor/Fireplaces/Fireplace02 (Fireplace)
Vehicles/Land/Indian_Taxi (Car) [Multipart]
```

- Some relevant attributes, such as Navigable, Mobile, Held:

```
Hover_Bike_01 (PyroProp) [Mobile Navigable] [Multipart]
 Donut_Held [Held]
```

- Multipart props are explicitly noted as such

```
Hover_Bike_01 (PyroProp) [Mobile Navigable] [Multipart]
Vehicles/Land/Indian_Taxi (Car) [Multipart]
```

- Prop variants and, if defined, default variant:

```
Cuckoo_Clock [Multipart]
    Variants (default: Cuckoo_Clock_Base):
        Cuckoo_Clock
        Cuckoo_Clock_Base
        Cuckoo_Clock_NoHandles
        Cuckoo_Clock_NoPendulum
```

Animation Props: should a prop in the addon come with one or more animations file associated, they will be referrenced (for a discussion about the relation between verbs and animation, please refer to this section).

- By default, actual names of the animation are not listed, but a simple count of the them is shown instead:

```
Hover_Bike_01 (PyroProp) [Mobile Navigable] [Multipart]
    Animations: 33 - See Verbs
Donut_Held [Held]
    Animations: 13 - See Verbs
```

- By checking the ***List All Animation Files*** option, the names of the animation files associated to the prop are listed:

```
Donut_Held [Held]
```

```
Animations:
    Male01/eat02
    Male01/eat24
    Male01/eat46
    Male01/eat_finish
    Male01/get0
    Male01/get2
    Male01/get4
    Male01/get6
    soloeat2
    soloeat3
    soloeat4
    soloeat5
    soloeat6
```

- Only in the case of the AutoAnimate props, it will be displayed the name of the animation which will be automatically triggered as soon as the prop is placed on the set:

```
Drinking_Bird (AutoAnimate)
    Autoanimation: DrinkingLoop
```

17. **StateMachine and Verbs files**: the presence of a StateMachine or a Verb files is indicated. These files define the animations of props and/or puppets, how are they related, and how they are shown in the UI. Please refer to this discussion about these files and the related animation files.

```
Has StateMachine and Verbs files
Has StateMachine file
```

18. **Verbs**: verbs are the way animations appear in the diverse menus of the UI of Moviestorm's Director's View (Postures, Gestures, Mutual Interactions, Edit walking gait, props' own menu, etc). Only animations for Autoanimate props don't need an explicit verb to be triggered.
Verbs appear listed by their type and the information listed for each one of them will depend on their type. When referring to puppet model(s) a verb applies to, '`*`' means both `Female01` and `Male01`), and `?` means it couldn't be determined (either because the animation file couldn't be found in the same addon or because there was some error in the definition of the verb, eg:

```
Gestures:
    [*] : Magic/Spell04Right_Stop
    [?] : Geatures/Magic/Spell01_Start
```

In the preceding example, the first gesture verb was correctly defined, but the second contained an error (`Geatures` instead of the mandatory `Gestures` gestures' menu path have to start with)

- Gaits and Gestures: the name of the animation will appear preceded by the puppet model(s) the verb applies:

```
Gaits:
    [*] : Confident_loop
    [*] : Depressed_loop
    [Female01] : Middle_aged_loop
    [Female01] : Skip_loop
    [Male01] : Hop_loop
    [Male01] : Run_strident_loop
Gestures:
    [*] : KVs/Eyebrow/As_if_brow
    [*] : KVs/Eyebrow/As_if_brow_end
    [Female01] : KVs/Eyebrow/Flirty_brow
    [Female01] : KVs/Eyebrow/Flirty_brow_end
```

- Puppet Solo verbs: the name (full path: note they must start with `Posture/`) of the verb, preceded by the puppet model(s) the verb applies:

```
Puppet Solo Verbs (Postures):
    * : Posture/Superhero/Float 01 Start
    * : Posture/Superhero/Float 01 Stop
    * : Posture/Superhero/Float 01 to Float 02
    * : Posture/Superhero/Float 01 to Fly Along
```

- Prop Solo verbs: name of the verb preceded by the prop model

```
Prop Solo Verbs:
    [Cuckoo_Clock] : Cuckoo Clock/Close door (no sound)
    [Cuckoo_Clock] : Cuckoo Clock/Close door
    [Cuckoo_Clock] : Cuckoo Clock/Cuckoo (no sound)
    [Cuckoo_Clock] : Cuckoo Clock/Cuckoo
```

- Held Prop and Interactive Prop verbs: name of the verb preceded by the model of the prop and the puppet model(s) the verb can be applied to.

```
Held Prop Verbs:
    Held_Prop/Flowers/Roses [*] : Flowers/Back to idle
    Held_Prop/Flowers/Roses [*] : Flowers/Carry Flowers
    Held_Prop/Flowers/Roses [*] : Flowers/Hide Flowers
    Held_Prop/Flowers/Roses [*] : Flowers/Present Flowers
Interactive Prop Verbs:
    Railing_Interactive [Female01] : Railing/Lean 01 Start
    Railing_Interactive [Male01] : Railing/Lean 01 Look Down
Left Start
    Railing_Interactive [Male01] : Railing/Lean 01 Look Down
Left Stop
    Railing_Interactive [Male01] : Railing/Lean 01 Look Down
Right Start
```

- Puppet Interaction (Mutual) verbs: name of the verb, preceded by both puppet models the verb applies to.

```
Puppet Interaction Verbs:
    * -> * : Hold Hands
    * -> * : Interact/Control/Frisk Person on Ground
    * -> Female01 : Interact/Fight/Punch up/(A) Clip around
ear ... (B) Unaffected
    * -> Male01 : Interact/Fight/Punch up/(A) Flurry of
punches ... (B) Falls
    Female01 -> Female01 : Interact/Have head Kissed
    Male01 -> Female01 : Interact/Have head Kissed
```

Please refer to this section for a discussion about the relation between verbs and animations.

19. **Sounds**: sound files are listed including the folder hierarchy they can be found, usually matching their function as assets of the sound track:

```
Sounds:
  Ambient/Indoor/party01.mp3
  Ambient/Outdoor/church_bells01.mp3
  Ambient/Vocal/chanukah.mp3
  Foley/Character/Contact/snowball_impact02.wav
  Foley/Footsteps/snow_barefoot_run02.wav
  Foley/Wall Fittings/Fires/fire_crackle01.wav
```

20. **Cutting Room Assets**: they are listed by categories, with their names along with a description and associated tags (only Filters), if available

```
Cutting Room Assets:
  Filters:
    Negative  :  Negates the colours in the scene
       Tags: colour
    Old Movie  :  Adds sepia toning and a 'scratch' effect
       Tags: colour,scratch
    Op Art  :  An effect resembling "optical art" circa 1960
       Tags: colour,abstract
  Text Styles:
    Simple credits list  :  Credits paired as [role][name] are
presented one at a time
    Overlaid simple credits  :  Credits paired as [role][name]
are presented one at a time
    Centred title with fade  :  Displays the movie title in
the centre of the screen. Fades in and out over the duration
of the clip
  Transitions:
    Long cross-blend  :  Adds a long cross-blend between
adjacent clips
    Medium cross-blend  :  Adds a medium-length cross-blend
between adjacent clips
    Short cross-blend  :  Adds a short cross-blend between
adjacent clips
```

21. **Special Effects**: listed with their names preceded by the folder hierarchy they can be found (usually `fx/`)

```
Special Effects:
  fx/Barrel_Fire_Medium
  fx/Barrel_Fire_Small
  fx/Barrel_Green_Heavy
  fx/Barrel_Green_Lite
```

22. **Materials**: they are listed with their names preceded by the folder hierarchy they are placed into, depending the type of surface on the set they can be applied to.

```
Materials:
  Ceiling/Basic_Ceiling_Textured Tintable
  Ceiling/Basic_Ceiling Tintable
  Floor/Carpet/Floor_Carpet_01 Tintable
  Floor/Ground/Ground_Grass_01
  Floor/Lino/Floor_Lino_01 Tintable
  Wall/Wall_Paint_Efffect_01 Tintable
  Wall/Bricks 01
  Wall/Emissive (Tintable)
```

23. **Sky Textures**: the name the texture files are listed:

```
Sky Textures:
   Sky_SciFi_01.dds
   Sky_SciFi_02.dds
   Sky_SciFi_03.dds
   Sky_SciFi_04.dds
```

24. **Other Assets**: this type of assets has been only found so far in the **Core** and **Base01** official content packs. They are listed with their name preceded by the subtype they belong to:

```
Other Assets:
  Scenario:GeneralScenario
  Scenario:IntroScenario
  Scenario:VidIt
  TerrainMask:grass_mask
```

25. ***Full description of problems***: in the case one or more problems were found while analyzing the addon, a more complete description of the issues will be given in this section.

5.4 Asset List Reference

This list is actually a data grid (table) with detailed information about the assets in either one specific addon or the output of an asset search. The information displayed in both cases is almost exactly the same, except of a couple of columns.

The information for each one of the addons in the list includes:

1. ***Name of the Addon***. It's the official name of the addon, as found in the addon signature file, regardless of the name of the folder containing the addon when installed.
2. ***Publisher of the Addon***. The name of the account of the author of the addon (or the last user publishing it). It's also obtained from the addon signature file.
   Please note that all official content packs are published by **`ShortFuze`**, but there are also a good deal of third-party addons which are published under that account (eg, addons created by Chris Ollis and available for purchasing on its site http://www.moddingstorm.com)
3. **Type of the asset**. This is one of the categories we've classified assets into.
   By large, the most numerous assets you'll find will be Body Parts, Props and Verbs and Animations, but there are other few types of assets out there.
   Regarding animations, by default they won't be listed in the list of assets in a specific addon, unless the checkbox for forcing it is checked. Also, please refer to the discussion about the disagreement between the number of animations and verbs.
4. **Subtype of the asset**. Most of the types of assets can be placed into one from a range of subtypes. The number of subtypes differ widely from one type of assets to other.
   NOTE: there's a very special case (albeit rather frequent) in which **`StormCat`** does not specify the subtype of an asset, and this happens for Decals that are reference in the definition of a **BodyPart** (especially Heads and Bodies) but are not included in the same addon; they are collectively listed as **External Decals Referenced** and they are listed with a subtype '?'
5. **Name of the asset**. The name specified depends on the type of asset:
   - Body Parts: the name of the body part is preceded by the puppet's model it can be applied to (ie, Male01 or Female01)
   - Decals: name the decal, preceded by the puppet's model it can be applied to
   - Props: what is displayed is the name of the prop's model, not properly its name as shown on the Set Workshop.
   - Verbs: for animations other than Gaits: menu path for the verb. For Gaits animations: name of the animation file
   - Sounds: name of the sound file.
   - Sky textures: name of the texture file
6. **Tags**: only a few types of assets have tags associated:
   - Body Parts
   - Props
   - Cutting Room Assets (Filters)
7. **Extra Information**: the structure and meaning of the additional information for an asset will depend on the type of asset:
   - Body Parts: morphable bodyparts (mainly heads, bodies and hairstyles) will be labelled as such
   - Props: lots of information can be placed here:
     - Variants
     - Flag indicating the prop is `[Multipart]`
     - Flag indicating the prop has animations associated (`[Animated]`). NOTE: however, take into account that a prop may be animated despite not sporting this flag (it might be a variant of a prop whose "parent" counterpart is part of another addon).
     - Some attributes: Held, Mobile, Navigable

- Verbs: basically specifies what assets and puppets' models apply to, depending on the type of Verb:
    - Gait and Gesture Animations: puppet model. [*] means both Female01 and Male01. [?] meaning it couldn't be determined
    - Held and Interactive Props: model of prop followed by model of the puppet (*=both Female and Male)
    - Prop Solo: model of the prop
    - Puppet Solo: model of the puppet (*=both Female and Male)
    - Puppets Interaction (Mutual Verbs): models of both puppets interacting
- Animations: name of the model of prop or puppet they're associated to
- Cutting Room Assets: description, when available

8. ***Free flag of the addon***, indicating whether the addon containing the asset is free or requires a license to be used.
9. ***Addon installed***: denotes whether the addon is installed or not
10. ***Official content pack***: for telling official content packs and third-party mods apart.
11. ***Location***: of the folder (installed addons) or file on the user's system disk(s)

NOTE: Columns in italics are displayed in the output of a search for assets, but are omitted in the list of the assets in a specific addon.

5.5 Duplicate Addons

The detection of duplicate addons in a Catalogue is a pretty complex subject, especially if you've managed to gather a respectable amount of them for your collection from very different sources. By no means is something as easy as checking by the names of the files they come in (many a time you'll find the same adon contained in files with very different names), not even the names of the addons themselves and their publishers (too often old addons created for previous versions of Moviestorm need to be republished to make them compatible with the last version of MS, and, too bad, its original author is not anymore around, so the republishing process must be undertaken by another user).

So a number of criteria is offered for customizing the identification of possible duplicates, including so far:

- Addon name
- Publisher name
- Recompilable flag (can't be deselected)
- Last published datetime
- Asset count
- Total file count
- Mesh data size
- Fingerprint (new in version 1.10.0)

Take into account there's not a magic combination of those criteria with a sensitivity of 100% (ie, it will detect every real duplicate in a collection) and a specificity of 100% (ie, it will reject every false duplicate).

An absolutely foolproof procedure for identifying true duplicates with both perfect sensitivity and 100% specificity would be, at the best, very costly from the computational point of view, so the procedure implemented try to do an acceptable job at an affordable cost.

The sensitivity and specificity, as defined above, of the duplicate detection procedure can vary to a great extent, depending on the criteria established. You'd probably go rather for a combination with a high specificity and an acceptable sensitivity than another with a high sensitivity but a poor specificity (ie, with a high rate of false positives).

As a starting point a combination of ***Name, Publisher and Last Published Datetime*** is offered by default. It should work most of the time, and it should render a detection with a very high specificity but not a sensitivity that great.
Another interesting combination of criteria is that made of ***Asset count, Total File count and Mesh Data Size***, but again is not perfect, for it offers a better sensitivity but a lower specificity.
Finally, in version 1.10.0 a new criterium for detecting duplicates was introduced for the first time: the addon's fingerprint. By itself, it's probably the most reliable item for identifying duplicate addons in a highly reliable way. However, it only will work if the fingerprint has previously been generated for the addons in a catalogue, which means you'll need either refreshing the information for the addons or recreating the catalogue (as an alternative, you can easily select the addons in a Duplicate Group, refresh the information just for them and then compare their fingerprints).

So you're free to experiment.

In any case, the output rendered by the duplicate detection process, as of this date, must be take with a grain of salt and is merely orientative. So it's advisable, before proceeding to discard (and, eventually, physically deleting) the offending duplicate from your library of addons, that you analyze and compare carefully the information for every of the addons in a duplicate group.

5.6 Notes.txt file

The **notes.txt** is an unofficial file that can be inserted in the the root folder of the addon by the user for conveying additional information, not included in any of the official files.
Take into account that this file will be recognized and read only by **StormCat** and will be ignored by Moviestorm.

Format of the file
notes.txt is a text file, with a very simple structure:

- Each line will contain an unique information item
- The format of every line will be:

  name = value

  Where **name** is the identifier of the information item (it can't be null or blank) and value is its actual value (it might be blank, but if so it will be ignored).
- The item identifiers currently recognized by **StormCat** are:
  - **original publisher** : for specifying the original author of a republished addon. The specified value will be displayed instead of the account name in the addon's signature file (which will be listed in the **Republished by** line of the addon report).
    Synonims: **original_publisher**, **publisher**
  - **comments** : reserved, not used as of this date.

Example:
Pak1234_Flight_Deck is an addon original created by Pak1234 for an very old version of Moviestorm, in a format not recognized by the current version of the program. To make it work again, it was republished by jamoram62.
So to give credit where credit is due, a notes.txt file was inserted into the addon's root folder. The file reads so:

```
original publisher = Pak1234
```

And the addon report output will show:

```
Name: Pak1234_Flight_Deck
Publisher: Pak1234
Republished by: jamoram62
Free: True
Last compiled: 2016-07-31 22:16:10Z
Total files: 57
Description: Flight Deck. By Pak1234
Revision: 0
* Fingerprint:
f8dc1d6cafe411c6300f57bceb05582817cdb68235f4ad619ea3dba3ea47f693
Mesh Data File size (MB): 1,486
Prop models:
  Cockpit
```

Back Cover