Add-on settings

See also: **Python development**

Contents

1 Description

settings.xml is a XML file that contains the current configuration for the addon and should be placed in the resources direcory (my.addon.id/resources/settings.xml). If your addon has configurable items that are set by the user, put them here. This file defines what the user sees when they click on Addon settings for your addon. You don't need to do any coding to utilise this functionality. The format for the settings file is relatively straightforward as can be seen in the following example:

Example code:

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<settings>
    <category label="32001">
        <setting label="32011" type="text"   id="username" default=""/>
        <setting label="32012" type="text"   id="password" option="hidden"  enable="!eq(-1,)" default=""/>
        <setting label="32053" type="slider" id="limit" subsetting="true" default="20" range="5,5,100" option="int" />
        <setting type="sep"/>
        <setting id="debug" type="bool" label="32013" default="false"/>
    </category>
    <category label="32010">
        <setting label="32032" type="action" action="RunScript(my.addon.id, downloadreport)"/>
    </category>
</settings>
```

You need to supply at least one category element. The label attribute of both categories and settings should be the id of a language string in your language files or the one of the main Kodi language file.

There are some functionality not cover by this wiki page. See C++ source:

```
https://github.com/xbmc/xbmc/tree/master/xbmc/addons/GUIDialogAddonSettings.cpp
```

2 Elements

2.1 Separators

Separators are purely user-interface elements that do not allow user input and therefore have no meaningful value as a setting.

2.1.1 type="sep"

A line separator, it adds a horizontal line that is useful to separate groups of settings.

Example code:

```
    <setting type="sep"/>
```

2.1.2 type="lsep"

A label separator, it adds a heading text that can be used to display on top of a group of settings.

Attributes:

- **label="*id*"** (required) - an id from the language file that indicates which text to display.

Example code:

```
    <setting label="32032" type="lsep"/>
```

2.2 Text input

Text input elements allow a user to input text in various formats. The "label" attribute must contain an id from the language file that indicates which text to display for the input field.

2.2.1 type="text"

Allow a user to enter one line of text.

Attributes:

- **id="*string*"** (required) - the name of the setting.
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- option="hidden"|"urlencoded" (optional)

  if set to "hidden", each characters entered by the user will be obfuscated with an asterisks ("*"), as is usual for entering passwords.
  if set to "urlencoded", each non-alphanumeric characters entered by the user will be "escaped" using %XX encoding.

- default="*value*" (optional) - the default value.

Example code:

```
    <setting label="32033" type="text" id="username" />
```

The value entered by the user is saved as a string value on disk:

```
<setting id="username" value="john.doe" />
```

Note: Security consideration: when using option="hidden", the value entered by the user is still saved as an unencrypted string value on disk.

2.2.2 type="ipaddress"

Allow a user to enter an ip address as text.

Attributes:

- **id="*string*"** (required) - the name of the setting.
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- default="*value*" (optional) - the default value.

Example code:

```
<setting label="32035" type="ipaddress" id="ipaddress"/>
```

The value entered by the user is saved as a string value on disk:

```
<setting id="ipaddress" value="127.0.0.1" />
```

2.3 Numeric input

Numeric input elements allow a user to enter a number. The "label" attribute must contain an id from the language file that indicates which text to display for the input field.

2.3.1 type="number"

Allows the user to enter an integer using up/down buttons.

Attributes:

- **id="*string*"** (required) - the name of the setting.
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- default="*value*" (optional) - the default value.

Example code:

```
<setting label="32036" type="number" id="code"/>
```

The value entered by the user is saved as a string value on disk:

```
<setting id="code" value="127000" />
```

Note: In Python, calling xbmcplugin.getSetting will return the number as a string value, which must be converted to an integer value if needed

2.4 Date and time input

2.4.1 type="date"

Attributes:

- **id="*string*"** (required) - the name of the setting
- **label="*id*"** (required) - an id from the language file that indicates which text to display or string
- default="2015-03-12" (optional) - the default value.

Example code:

```
<setting id="record_date" type="date" label="32030" default="2015-03-12"/>
```

2.4.2 type="time"

Attributes:

- **id="*string*"** (required) - the name of the setting
- **label="*id*"** (required) - an id from the language file that indicates which text to display or string
- default="13:13" (optional) - the default value.

Example code:

```
<setting id="record_time" type="time" label="32031" default="13:13"/>
```

2.5 Boolean

Boolean input elements allow a user to switch a setting on or off.

2.5.1 type="bool"

Attributes:

- **id="*string*"** (required) - the name of the setting
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- default="true"|"false" (optional) - the default value.

Example code:

```
<setting label="32033" type="bool" id="background" default="false"/>
```

Note: "true" and "false" are case sensitive!

The value entered by the user is saved as a string "true" or "false":

```
<setting id="background" value="false" />
```

Note: In Python, calling xbmcplugin.getSetting will return a string "true" or "false", which must be converted to a boolean value if needed

2.6 Select dialog

Will open separate selection window

2.6.1 type="select"

- **id="*string*"** (required) - the name of the setting
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- **values="*value1*[|*value2*[...]]" - or -**
- **lvalues="*id1*[|*id2*[...]]" (required): A list of values or language file ids from which the user can choose.**

```
<setting id="id_select" type="select" label="32000" lvalues="32001|32002|32003|32004" />
```

2.6.2 type="addon"

- **id="*string*"** (required) - the name of the setting
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- **addontype="*xbmc.metadata.scraper.movies*"** (required) - type of addon.
- multiselect="*true*|*false*" (optional) - allows to select several addons

```
<setting id="id_addon" type="addon" label="32111" default="" addontype="xbmc.metadata.scraper.movies"  multiselect="true" />
```

2.7 Spinner

A rotary selector allows the user to selected from a list of predefined values.

2.7.1 type="enum" or "labelenum"

Both element types work the same except that the "enum" type will use the index of the chosen value, wheras the "labelenum" will use the actual value.

Arguments:

- **id="*string*"** (required) - the name of the setting

- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- **values="*value1*[|*value2*[...]]" - or -**
- **lvalues="*id1*[|*id2*[...]]" (required):**

  A list of values or language file ids from which the user can choose.

- values="$HOURS" is special case to select hour
- sort="yes" - sorts labels (works only for type="labelenum")

Example code:

```
<setting label="32018" type="enum" id="service1" lvalues="32021|32022|32023|32024"/>
<setting label="32020" type="labelenum" id="service3" lvalues="32021|32022|32023|32024"/>
<setting label="32022" type="enum" id="default_enumhours" values="$HOURS" />
```

- Selecting the value "One" will return the int value "0", selecting the value "Two" will return the int value "1", etc.

The value entered by the user is saved as a string. For "enum"-type settings the index of the value is saved, starting at 0 for the first value. For "labelenum"-type settings the value is saved or, in case the lvalues attribute is used, the label translated in the language of the user.

```
<setting id="service1" value="0" />
<setting id="service3" value="SomethingTranslated" />
```

2.8 Slider

Will display a slider control

2.8.1 type="slider"

Allows the user to select a numberic value using a horizontal sliding bar.

Example:

```
<setting label="32053" type="slider" id="limit" default="20" range="5,5,100" option="int" />
<setting label="32053" type="slider" id="limit" default="20" range="0,100"/>
```

Attributes:

- **id="*string*"** (required) - the name of the setting.
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- **range="*min*[,*step*],*max*"** (required) - specify the range of valid values.
- **option="int"|"float"|"percent"** (required) - specifies whether to allow the user to choose between integers, floating point numbers or a percentage.
- default="*value*" (optional) - the default value.

The value entered by the user is saved as a string representation of a floating point value on disk:

```
<setting id="limit" value="5.000000" />
```

Note: In Python, calling xbmcplugin.getSetting will return a string, which must be converted to an int, float or percentage value if needed

2.9 Browser dialog

File and folder browsing elements allow a user to select a file or folder by browsing the local disks or network. You can specify a media type to show only files of the chosen type to the user. The "label" attribute must contain an id from the language file that indicates which text to display for the input field.

2.9.1 type="file", "audio", "video", "image" or "executable"

Allow the user to browse for and select a file. When using type="audio", "video", "image" or "executable", only files of those types are displayed to the user.

Attributes:

- **id="*string*"** (required) - the name of the setting
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- default="*value*" (optional) - the default value.

Example code:

```
<setting label="32033" type="file" id="file" default="path_to_files"/>
```

The full path to the file selected by the user is saved as a string value on disk:

```
<setting id="file" value="smb://path_to_files/file.ext" />
```

2.9.2 type="folder"

Allow the user to browse for and select a folder.

- **id="*string*"** (required) - the name of the setting
- **label="*id*"** (required) - an id from the language file that indicates which text to display.
- source="video", "music", "pictures", "programs", "files", "local" or blank - will show the respective folders from sources.xml in the browse dialog. source="" will list both local drives and network shares.
- default="*value*" (optional, default="") - the default value.
- option="writeable" (optional, default="") - the user can be allowed to create and select new folders by setting this argument.

Example code:

```
<setting label="32033" type="folder" id="folder" source="auto" option="writeable"/>
```

The full path to the folder selected by the user is saved as a string value on disk:

```
<setting id="folder" value="smb://path_to_files/"/>
```

2.9.3 type="fileenum"

Display files in a folder as an enum selector.

Example: List sub-folders of the 'resources' folder for this add-on in the settings.

```
<setting label="32000" id="myenum" type="fileenum" values="resources" mask="/" />
```

2.9.4 Extra attributes

- mask="*value*" - filter selectable files. Examples: "/" - display only folders. "*.txt" - display only .txt files.
- option="hideext" - hide file extensions.

2.10 Action execution

2.10.1 type = "action"

Adds line to configuration windows which allow to execute action

Example code:

```
<setting label="32032" type="action" action="RunScript(my.addon.id, downloadreport)"/>
```

List of actions that can be used:

- List of built-in functions

Attributes:

- option="close" (close the settings dialog before executing the action)

3 Conditions

Settings can optionally have a visible and/or enable attribute:

visible=""	"true", "false" or conditional. Determines if the setting is displayed in the settings dialog (default = 'true')"
	"true", "false" or conditional.

enable=""	Allows you to determine whether this setting should be shown as greyed-out and unable to be changed.
conditions	3 comparators are available to allow a setting to be made visible or enabled based on other settings: ▪ eq() Equal to ▪ gt() Greater than ▪ lt() Less than One can AND the comparators using the + symbol, and OR the comparators using the | symbol. A combination of AND and OR is not currently supported. One can negate a comparator using the ! symbol (e.g. !eq()). Each comparator takes 2 operands: ▪ Relative position of setting to compare ▪ Value to compare against Thus if we place settings in our file in this order: myFirst setting mySecondSetting myThirdSetting for the third setting we might add the option: enable="gt(-2,3) + lt(-2,8)" You can also use a conditional statement: enable="System.HasAddon(plugin.video.youtube)" Comparisons CANNOT be made across category/page boundaries. If one needs to compare to a setting on a different page, a copy of the setting with the same id can be placed on the page where the comparison is to take place and then hidden by setting visible to 'false'. The value is automatically updated when the setting on the other page is updated.

4 Subsettings

You can create a subsetting by adding **subsetting="true"** to a setting. This will add dash in front of the setting.