

TEMPESTWIKI



User Guide for

TEMPEST

**The Most Advanced
Programmable GUI XMLTV
EPG Generator**

Created by Kivanc Altug (Kvanc)

Document Date/Revision : 21.02.2022 / Rev.1

Tempest Version : v1.3.2

Website : <https://github.com/K-vanc/Tempest-EPG-Generator.git>

Website : <https://k-vanc.github.io/Tempest-EPG-Generator/>

Contact Information : tempestpg@gmail.com

Table of Content

- 1) Introduction**
- 2) Siteconfig Maker & Elements**
- 3) Tempest Variables & Operators**
- 4) Tempest Commands & Syntax**
- 5) Siteconfig Debugger**
- 6) EPG Generator**
- 7) Channel Generator**
- 8) Siteconfig Editor**
- 9) Tempest Configurator & Settings**
- 10) Hints & Tips**

1) Introduction

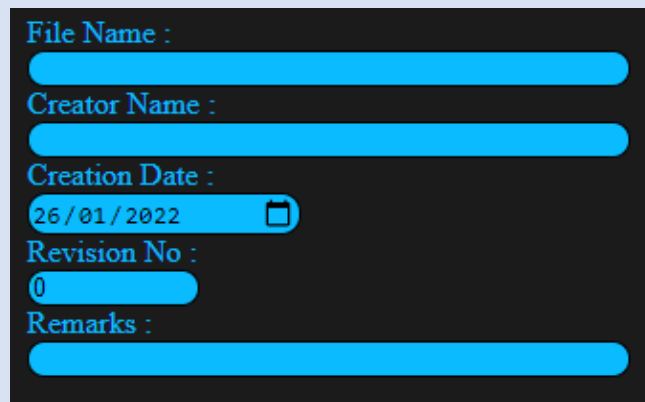
Tempest is a fully programmable XMLTV formatted EPG(Electronic Program Guide) generator bundled in a single php file with several advanced features such as;

- Graphical User Interface (GUI) & Optimization For Mobile Devices
- Multiple Website Configuration Support
- The Highest Generating Speed
- The Lowest Memory Consumption
- Multiple Platform Support (Windows/Unix etc.)
- Multiple Async URL Support Up to 10
- Full Support For XMLTV Standart Elements
- Daylight Saving Time(DST) Support For Automatic Time Offset Changes
- Easy Configuration Maker & Editor
- Powerful & Simple Command Syntax For Combo-Operations/Modifications
- Scrapping Engine Built with Regex (Regular Expression) For Powerful Operations
- Built-In Cyclone Module to Add Alternative For Failed Channel
- Built-In Time Converter to Change Times of All Shows Into Requested Time Offset (DST Support)
- Built-In Inverter Module to Transfer XMLTV Elements Into Description With Requested Order & Style
- Built-In Fusion Module to Keep Previous Dated Shows Up to 15 Days For Catch-Up Service
- Multiple Configuration File Support Up to 5000 Channels For Each
- Remote Access, Operations & Downloading (port-forwarding required)
- Scheduled Operations (such as crontab etc.)
- Support For Scrapping From HTML, XML, CSV, TXT, XLS, PDF (Experimental) & More..


2) Siteconfig Maker & Elements

You may easily prepare your own siteconfig files by using Tempest's Siteconfig Maker with hundreds of predefined element as below;

a) Config Info



The screenshot shows a dark-themed form with the following fields:

- File Name :
- Creator Name :
- Creation Date : 
- Revision No :
- Remarks :

File Name: Name which will be used to create siteconfig file
(Required)

Creator Name: Name of siteconfig maker

Creator Date: Date of siteconfig file created

Revision No: Revision number of siteconfig **(Required)**

Remarks: Explanations about siteconfig file, changes etc.

b) Site Options



The screenshot shows a dark-themed form with two columns of options:

Left Column:

- Timezone [±XX:XX]: (for reference: Timezones)
- Culture Info :
- Charset Info :
- Maximum Days :
- Rating System :
- Star Rating System :

Right Column:

- Episode System :
- Custom Episode System 1 :
- Custom Episode System 2 :
- First Show No :
- First Day No [Mon:0/Sun:6] :
- Enable Keep Index Page : ☐

Enable Past Day Remover : <input type="checkbox"/>	Channel Delay Time [sec] : <input type="text"/>
Compression System : --select--	Index Page Delay Time [sec] : <input type="text"/>
User-Agent : <input type="text"/>	Detail Page Delay Time [sec] : <input type="text"/>
Enable Proxy [http/socks] : <input type="checkbox"/>	
Proxy Address : <input type="text"/>	

Timezone: Timezone of source to set time offset and make necessary date/time calculations etc. It can be in $\pm XX:XX$ format such as "+03:00" or with referenced patterns such as "Europe/Istanbul". To enable DST(Daylight Saving Time) option, referenced name pattern shall be set. It will be considered as "+00:00" if no value is given. It supports;

##usertime## (Special Variable**)**

If timezone parameter is set as "##usertime##" , time offset of user will be applied. This means same siteconfig file will give different time offsets for a user in Europe and a user in America as an example. This is very useful for some websites which converts their data based on geolocation of visitor's IP address.

Culture Info: 2 letters culture code of source language. This is necessary for correct text formatting such as converting upper cases to lower cases etc. It will be considered as "xx" if no value is given.

Charset Info: Encoding type of source such as UTF-8 or WINDOWS-1252 etc. This is necessary for correct text scrapping. It will be considered as "UTF-8" if no value is given.

Maximum Days: Maximum number of days that source contains. It can be in 2 different types as "X" or "X.1" which "X" means each day in separate page/request and "X.1" means all days in single page/request such as "7" (for 7 days) and "7.1" (for 7 days in same page).

Rating System: Name of rating system that source uses. As example "TVPG" or "PEGI" etc.

Star Rating System: Name of star rating system that source uses. As example "IMDB" or "TMDB" etc.

Episode System: Name of episode system that source uses. 2 possible options are available as "**on-screen**" and "**xmltv_ns**".

Custom Episode System 1: Name of first custom episode system. It can be anything or a user-defined special system.

Custom Episode System 2: Name of second custom episode system. It can be anything or a user-defined special system.

First Show No: Number of first show which belongs today in source. This is necessary to skip shows at the beginning of source from previous day if no show date is available.

First Day No: Number of show blocks which associated with week days in source . This is **mandatory** for "**Maximum days 7.1**" sites to capture correct day block from source. "0" always represents "Monday" and "6" always represents "Sunday". If first show block belongs to Monday and continues up to Sunday, day order shall be "0123456". If first show block belongs to Sunday and continues as Monday, Tuesday etc till Saturday; day order shall be "6012345". It supports also shuffled block orders such as "0421635".

Enable Keep Index Page: Indicator for Tempest to store source data to use for other channels. This is necessary to avoid repeated source request when source also contains data for other channels.

Enable Past Day Remover: Indicator for Tempest to auto-delete shows belongs to previous day(s) in source. This is necessary to delete useless data from fixed source such as monthly updating websites which always starts with first day of month.

Compression System: Name of compression system that source uses. 2 possible options are available as "**gzip**" and "**deflate**". Do not mix it with "Accept-Encoding" header which auto decompression is already done by your browser/server. Setting it on a decompressed content, will result malformed byte sequences due to twice decompression. Basically do not use it if your source is in a human-readable format.

User-Agent: String of user-agent that will be used during your web requests to emulate a web browser. Tempest will try to use

your server's user-agent string if no value is given. If it is not set in your server, Tempest will use its own predefined user-agent string.

Enable Proxy: Indicator/Check Point for Tempest to connect a proxy address for web requests of applied siteconfig.

Proxy Address: IP address/port of proxy that will be used by Tempest for tunnelling web requests of applied siteconfig. It supports http, https, socks4 and socks5 proxy types. String shall be as below examples formats;

Example:

Proxy => "<https://ipaddress:port>"

or (with username & password)

Proxy => "[socks5://username:password@ipaddress:port](https://username:password@ipaddress:port)"

Channel Delay Time: Number of delay as second between different channel requests of applied siteconfig.

Index Page Delay Time: Number of delay as second between a channel's index page requests.

Detail Page Delay Time: Number of delay as second between a channel's detail page requests.

c) Page Options

Note: "All (A)" defines that information is valid for all "Index (I)", "Detail (D)" and "Channel Creation (C)" sections.

Url setting are same from Url1 to Url4 as below;

Url1 :	<input type="text"/>
Url1 Request Method :	<input type="text"/>
	GET
Url1 Post Data :	<input type="text"/>
Url1 Accept Header :	<input type="text"/>
Url1 Content-Type Header :	<input type="text"/>
Url1 Referer Header :	<input type="text"/>

Url1 Host Header :	<input type="text"/>
Url1 Origin Header :	<input type="text"/>
Url1 Cookie Header :	<input type="text"/>
Url1 Custom Header :	<input type="text"/>
Enable Url1 XMLHttpRequest :	<input type="checkbox"/>
Enable Url1 Cookie Transfer :	<input type="checkbox"/>
Enable Url1 Header Out :	<input type="checkbox"/>

Url1 Urldate Format : (for reference: Date Formats) <input type="text"/> Url1 Stopdate Format : <input type="text"/> Url1 Subpage Format : <input type="text"/> Enable Grabber1 : <input type="checkbox"/> Grabber1 Pattern : <input type="text"/> Enable Grabber2 : <input type="checkbox"/> Grabber2 Pattern : <input type="text"/>	Expand Url2 Parameters : <input checked="" type="checkbox"/> Enable Url2 Parameters : <input type="checkbox"/> Url2 : <input type="text"/> Url2 Request Method : <input type="text" value="GET"/> Url2 Post Data : <input type="text"/> Url2 Accept Header : <input type="text"/>
--	--

Url(1-4) (A): Url address that will be used by Tempest

Url(1-4) Request Method (A): Url request option that will be used by Tempest. Available options are "GET", "POST", "HEAD", "PUT", "DELETE" and "OPTIONS".

Url(1-4) Post Data (A): Request post data that will be used by Tempest. It may be used together with supported request methods except "GET" and "HEAD".

Url(1-4) Accept Header (A): Accept header string that will be used by Tempest to emulate web browser.

Url(1-4) Content-Type Header (A): Content-Type header string that will be used by Tempest to emulate web browser.

Url(1-4) Referer Header (A): Referer header string that will be used by Tempest to emulate web browser.

Url(1-4) Host Header (A): Host header string that will be used by Tempest to emulate web browser.

Url(1-4) Origin Header (A): Origin header string that will be used by Tempest to emulate web browser.

Url(1-4) Cookie Header (A): Cookie header string that will be used by Tempest to emulate web browser. This is necessary to send requests with user-defined or modified cookie strings. Sometimes it may be needed to send more than 1 cookie header. In this case, header strings may be separated with;

|#| (Special Operator**)**

Tempest will split cookie headers given which combined with "|#|", in single cookie header string and send them all individually for request. If you don't need to manipulate cookie to be sent, use "Enable Cookie Transfer" option to let Tempest automatically capture, store and submit required cookies for the current and following requests.

Url(1-4) Custom Header (A): User-defined or non-predefined header string that will be used by Tempest to emulate web browser. It is necessary to send some special header/data which is not available in predefined parameters. Sometimes it may be needed to send more than 1 custom header. In this case, header strings may be separated with "|#|" special operator to let Tempest split and send them all individually for request.

Enable Url(1-4) XMLHttpRequest (A): Indicator for Tempest to send "XMLHttpRequest" header to emulate web browser.

Enable Url(1-4) Cookie Transfer (A): Indicator for Tempest to capture, store and send back cookies to emulate web browser. This is necessary for some website which requires session and/or private cookies to authenticate your request. Tempest will capture and store cookies on the first "Url" which this option enabled. For the following "Url", Tempest will only send back if "Cookie Transfer" option is not enabled or send back, capture and store new cookies for the following "Url" if "Cookie Transfer" option is enabled. As example;

Case 1

- Cookie transfer is enabled for Url1 and not for Url2

Tempest will store the cookies which are received from Url1 and submit to Url2

Case 2

- Cookie transfer is enabled for both Url1 and Url2

Tempest will store the cookies which are received from Url1, submit to Url2, capture and store new cookies of Url2 for possible request of Url3

Enable Url(1-4) Header Out (A): Indicator for Tempest to add website's header response to website's normal response. This is necessary for some rare websites to capture data and use in the following requests.

Url(1-4) Urldate Format (I/C): Format pattern for Tempest to calculate needed date/time value and transfer to its associated special variable with;

##urldateX## (Special Variable**)**

Note: Highlighted "X" will be number of urldate format

Example:

Url1 => "<https://www.test.page?date=##urldate1##>"

Url1 Urldate Format => "Y-m-d"

Result => "<https://www.test.page?date=2022-01-29>"

All urldate formats may be set differently and individually at the same time and they could output different date results. They can be in referenced date/time formats or Tempest's special date/time operator formats as below;

(Special Date/Time Operators**)**

"Unix" => Returns Unix timestamp of current day

"Java" => Returns Java timestamp of current day

"#weekdayname#(n1|n2|n3|n4|n5|n6|n7)" => Returns corresponding weekday name based on given user list as n1(Monday) to n7(Sunday)

"#weekdaynumber#X" => Returns corresponding weekday number based on given user Sunday number as "X" which could be "0", "6" or "7". It will be considered as "7", If no "X" value is given.

"#daycounter#X" => Returns corresponding day number based on given user counter start point as "X". It will be considered as "0", If no "X" value is given.

"#yeardaycounter#X" => Returns corresponding year day number based on given user counter start point as "X". It will be considered as "0", If no "X" value is given.

"#daylist#n1|n2|...|nX" => Returns corresponding day name based on given user list as n1(Start) to nX(End). It does not have to be 7 days. It is necessary for sites which use fixed date labels such as "today", "tomorrow" etc.

"#range#format" => Returns date string from start to end based on maximum days or timespan of given format separated with comma(,)

Sometimes, it may be needed to use parsed date strings. In this case, date string may be splitted into sub variables as

"###urldateX_N###" which "N" supports 1 to 5, with Tempest command **"||#split#"** as below;

||#split#(pattern) (Tempest Command**)**

Note: Highlighted **"pattern"** must be valid regex pattern

Example:

Url1 Urldate Format => "Y-m-d||#split#(-)"

Results =>

"###urldate1##" => "2022-01-29"

"###urldate1_1##" => "2022"

"###urldate1_2##" => "01"

"###urldate1_3##" => "29"

Also it may be needed to modify/increase or decrease calculated date from referenced patterns or "Unix/Java" operators with Tempest command **"||#date#"** as below;

||#date#string (Tempest Command**)**

Note: Highlighted **"string"** may be in "day", "hour", "minute", "second", "month", "year" and their combination formats

Example: (As today "2022-01-29")

Url1 Urldate Format => "Y-m-d||#date#+1day-2year"

Results => "##urldate1##" => "2020-01-30"

Urldate variables may be used in all corresponding Url headers unlimitedly.

Url(1-4) Stopdate Format (I/C): Format pattern for Tempest to calculate needed date/time value and transfer to its associated special variable with;

##stopdateX## (Special Variable**)**

Note: Highlighted "X" will be number of stopdate format

Example:

Url1 => "<https://www.test.page?date=##stopdate1##>"

Url1 Stopdate Format => "Y-m-d" (As today "2022-01-29")

Result => "<https://www.test.page?date=2022-01-30>"

All stopdate formats may be set differently and individually at the same time and they could output different date results. They can be in referenced date/time formats or Tempest's special operator formats as given on "**Urldate Format**". Differently from urldate format, stopdate format will be calculated as "+1 day" of urldate calculated for sites which have "**Maximum Days**" in "X" format and as "+1 day + timespan" of urldate calculated for sites which have "**Maximum Days**" in "X.1" format. Stopdate format may be parsed or modified same as urldate format with same Tempest commands as explained on "**Urldate Format**".

Example:

Url1 Stopdate Format => "Y-m-d||#split#(-)"

Results => "##stopdate1##" => "2022-01-30"

"##stopdate1_1##" => "2022"

"##stopdate1_2##" => "01"

"##stopdate1_3##" => "30"

Stopdate variables may be used in all corresponding Url headers unlimitedly.

Url(1-4) Subpage Format (I/C): String or format pattern for Tempest to calculate needed loop value and transfer to its associated special variable with;

##subpageX## (Special Variable**)**

Note: Highlighted "X" will be number of subpage format

Each values shall be separated with "|" as string or as format pattern Tempest special operator "#counter#X#Y#Z" shall be used as below;

String1|String2|....|StringN (Separator**)**

#counter#X(#Y)#Z (Special Operator** / (#Y) Optional)**

Example:

Url1 Subpage Format => "1|2|3|4"

Results => For each page request

"##subpage1##" => "1"

"##subpage1##" => "2"

"##subpage1##" => "3"

"##subpage1##" => "4"

Url2 Subpage Format => "#counter#1#2#5"

X => Start Indice

Y => Step Interval (Optional)

Z => Destination

Results => For each page request

"##subpage2##" => "1"

"##subpage2##" => "3"

"##subpage2##" => "5"

Url4 Subpage Format => "#counter#1#5"

X => Start Indice

Z => Destination

Results => For each page request

"##subpage4##" => "1"

"##subpage4##" => "2"

"##subpage4##" => "3"

"##subpage4##" => "4"

"##subpage4##" => "5"

Subpage variables may be transferred to following url parameters in order to built nested loops which each subpage format will loop inside following url subpage format's loop.

Subpage variables will create a loop from first value to last value for each page request.

Enable Grabber(1-8) (I/C): Indicator for Tempest to enable pattern box of Tempest's page data grabbers.

Grabber(1-8) Pattern (I/C): Regex pattern(auto-delimited with "/" by default) to be used by Tempest to capture essential page data and transfer to its associated special variable which may be used in following requests as below;

##grabber_X## (Special Variable**)**

Note: Highlighted "X" will be number of grabber pattern

2 individual grabbers for each url has been predefined by Tempest and they will capture data as per given regex pattern on their associated url response data. But in case of need, **higher numbered** unused grabbers may also be called for additional help on needed url with "**||#urlX#**" special command;

Grabber_7pattern||#url2# (Special Command**)**

On above example, this command will send Grabber7 which is normally associated with Url4 to grab on page response of Url2 and transfer its value to "**##grabber_7##**" variable.

Grabbers may be enabled independent from their corresponding Url parameters.

Expand Url(2-4) Parameters (I/C): Indicator for Tempest to expand additional url parameters.

Enable Url(2-4) Parameters (I/C): Indicator for Tempest to enable expanded additional url parameters.

d) Show Options

Note: All regex patterns have been pre-delimited with "/"(slash) and shall be escaped by "\"(backslash) together with other regex tokens.

Show Pattern :	Showicon Encode/Decode :
<input type="text"/>	1: <input type="text"/> 2: <input type="text"/>
Sort Pattern :	Season Pattern :
<input type="text"/>	<input type="text"/>
Sort Flag :	Episode Pattern :
<input type="text"/>	<input type="text"/>
Sort Order :	Total Episode Pattern :
<input type="text"/>	<input type="text"/>
Start Pattern :	Custom Episode 1 Pattern :
<input type="text"/>	<input type="text"/>
Start Format Pattern :	Custom Episode 2 Pattern :
<input type="text"/>	<input type="text"/>
Stop Pattern :	Channel Logo Pattern :
<input type="text"/>	<input type="text"/>
Stop Format Pattern :	Channel Logo Encode/Decode :
<input type="text"/>	1: <input type="text"/> 2: <input type="text"/>
Duration Pattern :	Production Date Pattern :
<input type="text"/>	<input type="text"/>
Title Pattern :	Actor Pattern :
<input type="text"/>	<input type="text"/>
Original Title Pattern :	Role(for Actor) Pattern :
<input type="text"/>	<input type="text"/>
Sub-title Pattern :	Director Pattern :
<input type="text"/>	<input type="text"/>
Description Pattern :	Presenter Pattern :
<input type="text"/>	<input type="text"/>
Category Pattern :	Producer Pattern :
<input type="text"/>	<input type="text"/>
Language Pattern :	Composer Pattern :
<input type="text"/>	<input type="text"/>
Original Language Pattern :	Editor Pattern :
<input type="text"/>	<input type="text"/>
Showicon Pattern :	Commentator Pattern :
<input type="text"/>	<input type="text"/>

Writer Pattern :	Keyword Pattern :
Adapter Pattern :	Subtitles Pattern :
Guests Pattern :	Rating Pattern :
Country Pattern :	Rating Icon Pattern :
Video Colour Pattern :	Rating Icon Encode/Decode :
Video Aspect Pattern :	1: --select-- 2: --select--
Video Quality Pattern :	Star Rating Pattern :
Premiere Pattern :	Star Rating Icon Pattern :
Audio Pattern :	Star Rating Icon Encode/Decode :
Length Pattern :	1: --select-- 2: --select--
Previously-Shown Pattern :	New Pattern :
	Last Chance Pattern :
	Review Pattern :

Tempest has several predefined xmltv element in accordance with XMLTV Standard and all may be set from "**Show Options**" tab as below;

Show Pattern (I): This is the regex pattern which tells Tempest how epg information containing part will be separated from unnecessary data and parsed as individual small blocks that each contains single show. Since regex is not the best tool for parsing irregular sources such as html, Tempest has a special pattern format which will be automatically recognized and will apply 2 layers of parsing to make it significantly easy, as below;

MBS.*?(?:SBS)(.*?)(?:SBE).*?MBE (Special Pattern**)**

MBS => Main-Block Start

MBE => Main-Block End

SBS => Sub-Block Start

SBE => Sub-Block End

It shall be considered as each sub-block shall contain 1 single show and main blocks may be single or multiple based on page structure.

Special pattern is mandatory to use with “**First Day No**” required sites for “**Maximum days 7.1**” to parse data based on correct day block. Otherwise, most probably results will not match with correct dates. For these kind of sites, 7 main blocks (1 for each day) need to be captured and Tempest will parse them based on given “**First Day No**” order. Special pattern is not mandatory for other type websites but it is highly recommended to use whenever possible since it is safer for new users and it will make Tempest to parse faster with lower memory consumption.

Example:

```
<div role="tab-\d.*?>.*?(?:<p>)(.*?)(?:</p>).*?</div>
```

Tempest will auto convert unicode characters, clean show blocks, fix malformed bytes and perform json decoding after parsing.

Note: Tempest will auto clean trailing spaces, line breaks, html tags etc. for all captured data of xmltv elements.

Sort Pattern (I): This is the regex pattern which tells Tempest to search for a sorting value inside of sub-blocks. This is necessary if your data blocks are provided without date order or mixed order.

Sort Flag (I): This is the indicator for Tempest which tells how sorting operation will be done. Available options are “**regular**”, “**numeric**”, “**string**”, “**natural**”, “**locale**”, “**case-string**” and “**case-natural**”.

Sort Order (I): This is the indicator for Tempest which tells order of sorting. Available options are “**ascending**” and “**descending**”.

Start Pattern (I/D): This is the regex pattern for capturing start time value of each show.

Start Format Pattern (I/D): This is the string which tells Tempest format of captured start time. It shall be like in referenced date formats on “**Urldate Formats**”.

Example:

Captured Start Time => “2022-01-29 02:30:00”

Required Start Format => “Y-m-d H:i:s”

Stop Pattern (I/D): This is the regex pattern for capturing stop time value of each show. If not set, next show's start time will be set as stop time of current show.

Stop Format Pattern (I/D): This is the string which tells Tempest format of captured stop time. It shall be like in referenced date formats on "**Urldate Formats**".

Duration Pattern (I/D): This is the regex pattern for capturing duration value of each show. If "**Stop Pattern**" is not set or could not capture any data, Tempest may use duration value by adding up to start time. When used, format of data shall be defined with "**||#format#X**" Tempest command.

||#format#X (Tempest Command**)**

Note: Highlighted "**X**" shall be one of the below values;

"ms" => "microsecond"

"s" => "second"

"m" => "minute" (by default)

"h" => "hour"

"t" => "time pattern such as 01:30 as hour:minute"

Title Pattern (I/D): This is the regex pattern for capturing title value of each show.

Original Title Pattern (I/D): This is the regex pattern for capturing original title value of each show. When used, language code shall be defined with "**||#culture#xx**" special command.

||#culture#xx (Special Command**)**

Note: Highlighted "**xx**" shall be 2 letters culture code of data

If not set, "xx" value will be used as culture code of original title element.

Sub-title Pattern (I/D): This is the regex pattern for capturing sub-title value of each show.

Description Pattern (I/D): This is the regex pattern for capturing description value of each show.

Category Pattern (I/D): This is the regex pattern for capturing category value of each show.

Language Pattern (I/D): This is the regex pattern for capturing language value of each show.

Original Language Pattern (I/D): This is the regex pattern for capturing original language value of each show.

Showicon Pattern (I/D): This is the regex pattern for capturing showicon value of each show.

Showicon Encode/Decode (1-2) (I/D): This is the indicator for Tempest to tell which encode/decode operation(s) will be performed on captured showicon value(s). Available options are "urlencode", "urldecode", "rawurlencode" and "rawurldecode". When both are set, operations will be performed from 1 to 2.

Season Pattern (I/D): This is the regex pattern for capturing season value of each show.

Episode Pattern (I/D): This is the regex pattern for capturing episode value of each show.

When "Episode System" is set for "on-screen", **season** and **episode** prefix which is "S" and "E" by default, may be set by user with "**||#name#X**" Tempest command.

||#name#X (Tempest Command**)**

Note: Highlighted "X" shall be user defined string as season/episode prefix

Example:

Default Season No => "S3"

With "||#name#Temporada " => "Temporada 3"

Default Episode No => "E14"

With "||#name#Episodio " => "Episodio 14"

Default Season/Episode => "S3 E14"

With "||#name#X" => "Temporada 3 Episodio 14"

Total Episode Pattern (I/D): This is the regex pattern for capturing total episode value of each show.

Custom Episode 1 Pattern (I/D): This is the regex pattern for capturing first custom(user-defined) season/episode value of each show. Corresponding “**Custom Episode System**” shall be defined.

Custom Episode 2 Pattern (I/D): This is the regex pattern for capturing second custom(user-defined) season/episode value of each show. Corresponding “**Custom Episode System**” shall be defined.

Channel Logo (I/D): This is the regex pattern for capturing channel logo value of each channel. Different from other elements, “**channel logo**” element is in global mode which will search in unparsed source data instead of main or show blocks.

Channel Logo Encode/Decode (1-2) (I/D): This is the indicator for Tempest to tell which encode/decode operation(s) will be performed on captured channel logo value. Available options are “**urlencode**”, “**urldecode**”, “**rawurlencode**” and “**rawurldecode**”. When both are set, operations will be performed from 1 to 2.

Production Date Pattern (I/D): This is the regex pattern for capturing production date value of each show.

Actor Pattern (I/D): This is the regex pattern for capturing actor value(s) of each show.

Role Pattern (I/D): This is the regex pattern for capturing role value of each actor. This will not work if no actor value is captured or unequal number of actor-role values captured. Values will be set to attribute of actor tag as per XMLTV standard

Director Pattern (I/D): This is the regex pattern for capturing director value(s) of each show.

Presenter Pattern (I/D): This is the regex pattern for capturing presenter value(s) of each show.

Producer Pattern (I/D): This is the regex pattern for capturing producer value(s) of each show.

Composer Pattern (I/D): This is the regex pattern for capturing composer value(s) of each show.

Editor Pattern (I/D): This is the regex pattern for capturing editor value(s) of each show.

Commentator Pattern (I/D): This is the regex pattern for capturing commantator value(s) of each show.

Writer Pattern (I/D): This is the regex pattern for capturing writer value(s) of each show.

Adapter Pattern (I/D): This is the regex pattern for capturing adapter value(s) of each show.

Guest Pattern (I/D): This is the regex pattern for capturing guest value(s) of each show.

Country Pattern (I/D): This is the regex pattern for capturing country value(s) of each show.

Video Colour Pattern (I/D): This is the regex pattern for capturing video colour value of each show.

Video Aspect Pattern (I/D): This is the regex pattern for capturing video aspect value of each show.

Video Quality Pattern (I/D): This is the regex pattern for capturing video quality value of each show.

Premiere Pattern (I/D): This is the regex pattern for capturing premiere value of each show. It may be as text description or self-closing tag format. If final value is "true" or "yes", Tempest will convert it to self-closing tag.

Audio Pattern (I/D): This is the regex pattern for capturing audio value of each show.

Length Pattern (I/D): This is the regex pattern for capturing length value of each show. When used, unit of data shall be defined with "**||#unit#X**" Tempest command.

||#unit#X (Tempest Command**)**

Note: Highlighted "X" shall be "s" (second), "m" (minute by default) or "h" (hour)

Previously-Shown Pattern (I/D): This is the regex pattern for capturing previously shown value of each show. Value may be shown as attribute of tag if it is in date format without separators or Tempest will convert it to self-closing tag.

Keyword Pattern (I/D): This is the regex pattern for capturing keyword value of each show.

Subtitles Pattern (I/D): This is the regex pattern for capturing subtitle value of each show. When used, type of data shall be defined with "**||#type#X**" Tempest command.

||#type#X (Tempest Command**)**

Note: Highlighted "**X**" shall be "**t**" (teletext by default), "**o**" (onscreen) or "**d**" (deaf-signed)

Rating Pattern (I/D): This is the regex pattern for capturing rating value of each show.

Rating Icon Pattern (I/D): This is the regex pattern for capturing rating icon value(s) of each show.

Rating Icon Encode/Decode (1-2) (I/D): This is the indicator for Tempest to tell which encode/decode operation(s) will be performed on captured rating icon value(s). Available options are "**urlencode**", "**urldecode**", "**rawurlencode**" and "**rawurldecode**". When both are set, operations will be performed from 1 to 2.

Star Rating Pattern (I/D): This is the regex pattern for capturing star rating value of each show. When used, max rating limit shall be defined to the end of value as "**/X**" such as "6.2/10" etc.

Star Rating Icon Pattern (I/D): This is the regex pattern for capturing star rating icon value(s) of each show.

Star Rating Icon Encode/Decode (1-2) (I/D): This is the indicator for Tempest to tell which encode/decode operation(s) will be performed on captured star rating icon value(s). Available options are "**urlencode**", "**urldecode**", "**rawurlencode**" and "**rawurldecode**". When both are set, operations will be performed from 1 to 2.

New Pattern (I/D): This is the regex pattern for capturing new value of each show. It is self closing tag without any attribute. Tempest will generate new tag if anything captured with given pattern.

Last Chance Pattern (I/D): This is the regex pattern for capturing last chance value of each show. It may be as text description or self-closing tag format. If final value is "true" or "yes", Tempest will convert it to self-closing tag.

Review Pattern (I/D): This is the regex pattern for capturing review value of each show.

e) Detail Page Options

Note: Tempest gives priority order on page elements which "**Detail Page 2**" have the highest priority while "**Index Page**" have the lowest. If both elements are set and captured, Tempest will use value of "**Detail Page 2**". If "**Detail Page 2**" fails, Tempest will look for "**Detail Page 1**" and then for "**Index Page**".

Expand Detail Page Parameters : <input type="checkbox"/>	Enable Detail Page Grabber1 : <input type="checkbox"/>
Enable Detail Page Parameters : <input type="checkbox"/>	Detail Page Grabber1 Pattern :
Detail Page Key Pattern :	<input type="text"/>
Detail Page Key Link Encode/Decode :	Enable Detail Page Grabber2 : <input type="checkbox"/>
1: <input type="text"/> 2: <input type="text"/>	Detail Page Grabber2 Pattern :
Detail Page Url :	<input type="text"/>

Expand Detail Page (1-2) Parameters (D): Indicator for Tempest to expand detail page url parameters.

Enable Detail Page (1-2) Parameters (D): Indicator for Tempest to enable expanded detail page url parameters.

Detail Page (1-2) Key Pattern (D): This is the regex pattern for capturing detail page key value of each show and transfer to its associated special variable which may be used in detail page url/parameter requests as below;

##pagekeyX## (Special Variable**)**

Note: Highlighted "X" will be number of page key pattern

"Detail Page Key" will always search inside show block while "Detail Page2 Key" will search in response of "Detail Page Url". In case of need, "Detail Page2 Key" may be directed to show block with "||#index#" special command;

Detail_Page2_Keypattern||#index#(Special Command**)**

On above example, this command will send Detail Page2 Key which is normally associated with Detail Url1 to grab on show block and transfer its value to "##pagekey2##" variable.

Detail Page Key Encode/Decode (1-2) (I/D): This is the indicator for Tempest to tell which encode/decode operation(s) will be performed on captured detail page key value. Available options are "urlencode", "urldecode", "rawurlencode" and "rawurldecode". When both are set, operations will be performed from 1 to 2.

Note: "Detail Page Url" parameters/headers are same as explained in "Page Options".

Enable Detail Page Grabber(1-4) (D): Indicator for Tempest to enable pattern box of Tempest's detail page data grabbers.

Detail Page Grabber(1-4) Pattern (D): Regex pattern(auto-delimited with "/" by default) to be used by Tempest to capture essential detail page data and transfer to its associated special variable which may be used in following requests as below;

##dgrabber_X## (Special Variable**)**

Note: Highlighted "X" will be number of grabber pattern

2 individual grabbers for each url has been predefined by Tempest and they will capture data as per given regex pattern on their associated url response data. But in case of need, **higher numbered** unused grabbers may also be called for additional help on needed url with "||#durl1#" special command;

Detail_Grabber_4pattern||#durl1#(Special Command**)**

On above example, this command will send Detail Grabber4 which is normally associated with Detail Url2 to grab on page response of Detail Url1 and transfer its value to "##dgrabber_4##" variable.

Also they may be set to capture data from corresponding show block with "**||#index#**" special command;

Detail_Grabber_2pattern||#index# (Special Command**)**

On above example, this command will send Detail Grabber2 which is normally associated with Detail Url1 to grab on show block and transfer its value to "**##dgrabber_2##**" variable.

Grabbers may be enabled independent from their corresponding Url parameters.

In case of, detail pages have different values for same element and both needed to be kept such as parted description that first part in show block and second part is in detail page, captured value from detail page may be sent to merge with show block value of same element with "**##detail##**" special variable;

##detail## (Special Variable**)**

Example:

Desc Pattern => desc1:"(.*?)"||#addend###detail##

Detail Page Desc Pattern => desc2:"(.*?)"

Result Desc Element => Value of "desc1" + Value of "desc2"

or

Result Desc Element => Value of "desc1" (if desc2 is empty)

Note: Highlighted "**||#addend#**" is a Tempest command to insert given user input to end of requested element value

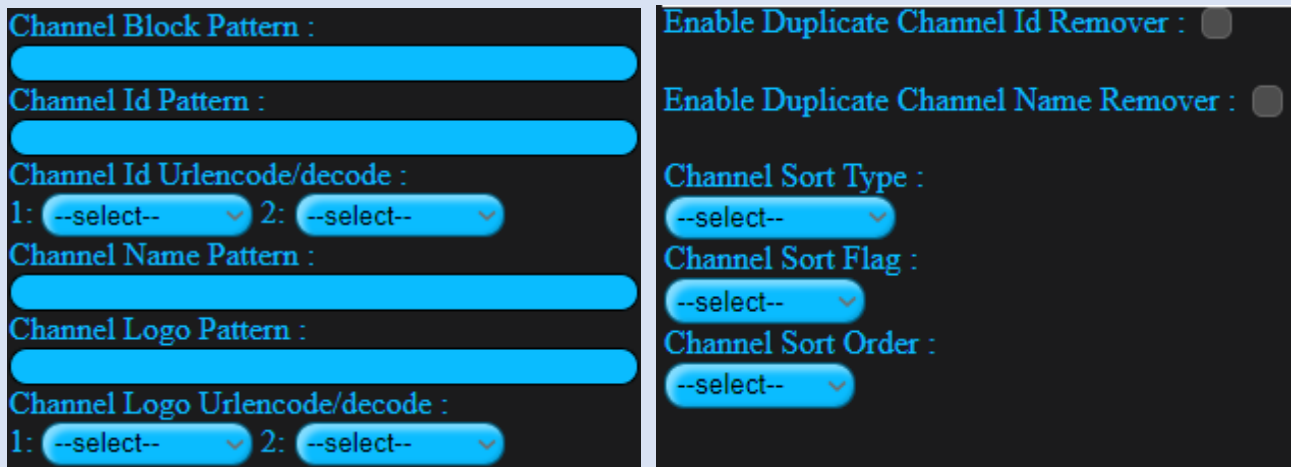
f) Channel Page Options

This section is used to set Url parameters which will be used by Tempest to achieve source data for channel creation.

Note: "**Channel Page Option**" parameters/headers are same as explained in "**Page Options**" except that resulted index will be for parsing channel ids, channel names, channel logos etc. instead of parsing show blocks. Also separate proxy address may be set differently from "**Site Options**" to be used only during channel creation requests in case of need.

g) Channel Options

Note: All regex patterns have been pre-delimited with "/"(slash) and shall be escaped by "\"(backslash) together with other regex tokens.



Channel Block Pattern (C): This is the regex pattern which tells Tempest how channel information containing part will be separated from unnecessary data and parsed as individual small blocks that each contains single channel data(id, name, logo etc.). "**Channel Block**" is an **optional** solution to make channel data parsing easier with higher accuracy. "**Channel Block**" pattern may be ordinary regex or special pattern which will be automatically recognized and will apply 2 layers of parsing to make it significantly easy, as below;

MBS.*?(?:SBS)(.*?)(?:SBE).*?MBE (Special Pattern**)**

MBS => Main-Block Start

MBE => Main-Block End

SBS => Sub-Block Start

SBE => Sub-Block End

When used, "**Channel Id**", "**Channel Name**" & "**Channel Logo**" patterns will search inside captured sub-blocks instead of source data. "**Channel Block**" will search in loose capture mode when ordinary regex string is set. This means no need to completely parse small channel blocks with very accurate capture string but you can just mention Tempest that how channel blocks should be look like and it will parse it accordingly.

Channel Id Pattern (C): This is the regex pattern for capturing id value of each channel. If "**Channel Block**" is not set, "**Channel Id**" result will be used as reference by Tempest for alignment of other channel datas. Its value will be transferred in to "**##channel##**" special variable;

##channel## (Special Variable**)**

Beside regex pattern, numerical values may be calculated with "**#counter#X#Y#Z**" operator as below;

#counter#X(#Y)#Z (Special Operator** / (#Y) Optional)**

Example:

Channel Id => "#counter#1#2#5"

X => Start Indice

Y => Step Interval (Optional)

Z => Destination

Results => Channel Id1 => 1

Channel Id2 => 3

Channel Id3 => 5

if "Channel Block" set and not empty, both Y(Step Invertal) and Z(Destination) may be optional and Tempest increase with 1 or with given interval till it reaches total number of channel blocks as below;

Count of Channel Blocks Captured => 4

Channel Id => "#counter#1"

Results => Channel Id1 => 1

Channel Id2 => 2

Channel Id3 => 3

Channel Id4 => 4

Channel Id => "#counter#1#2"

Results => Channel Id1 => 1

Channel Id2 => 3

Channel Id3 => 5

Channel Id4 => 7

Also "**Channel Id**" value(s) may be directly set as user-input with "**||#set#**" Tempest command as below;

||#set#X (Tempest Command**)**

Note: Highlighted "**X**" will be string for Channel Id(s). If multiple values will be set, they shall be separated with "|" in string.

Example:

Channel Id => "||#set#ab|cd|ef"

Results => Channel Id1 => ab

Channel Id2 => cd

Channel Id3 => ef

Channel Id Encode/Decode (C): This is the indicator for Tempest to tell which encode/decode operation(s) will be performed on captured channel id values. Available options are "**urlencode**", "**urldecode**", "**rawurlencode**" and "**rawurldecode**". When both are set, operations will be performed from 1 to 2.

Channel Name Pattern (C): This is the regex pattern for capturing name value of each channel. Command and operators explained in "**Channel Id**" section, may be used as same. Its value will be transferred in to "**##xmltv_id##**" special variable;

##xmltv_id## (Special Variable**)**

Channel Logo Pattern (C): This is the regex pattern for capturing logo value of each channel. Command and operators explained in "**Channel Id**" section, may be used as same. Its value will be transferred in to "**##cclogo##**" special variable;

##cclogo## (Special Variable**)**

Channel Logo Encode/Decode (C): This is the indicator for Tempest to tell which encode/decode operation(s) will be performed on captured channel logo values. Available options are "**urlencode**", "**urldecode**", "**rawurlencode**" and "**rawurldecode**". When both are set, operations will be performed from 1 to 2.

Enable Duplicate Channel Id Remover (C): Indicator for Tempest to compare and delete duplicated "**Channel Id**" values and their corresponding "**Channel Name**" / "**Channel Logo**" values.

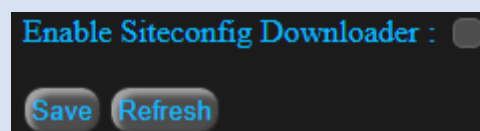
Enable Duplicate Channel Name Remover (C): Indicator for Tempest to compare and delete duplicated "**Channel Name**" values and their corresponding "**Channel Id**" / "**Channel Logo**" values.

Channel Sort Type (C): This is the indicator for Tempest to sort and reindex captured values based on selected source. Available options are "**Channel Id**" and "**Channel Name**".

Channel Sort Flag (C): This is the indicator for Tempest to sort and reindex captured values based on selected style. Available options are "**regular**", "**numeric**", "**string**", "**natural**", "**locale**", "**case-string**" and "**case-natural**".

Channel Sort Order (C): This is the indicator for Tempest which tells order of channel sorting. Available options are "**ascending**" and "**descending**".

h) Save & Exit



Enable Siteconfig Downloader (C): Indicator for Tempest to start downloading of created siteconfig file after saving completed.

Save (C): Completion of prepared siteconfig file. Tempest will create the file with inputted content if all required information fulfilled.

3) Tempest Variables & Operators

Tempest uses 2 types of string format called as "**variables**" and "**operators**" in order to generate dynamic url requests and even regex patterns. The main difference between these string formats is, "**variables**" may store static or dynamic data and return their recent stored data on time of call while "**operators**" are able to perform some modifications/operations on their stored static or dynamic data and return their recent stored data on time of call. The list and brief explanations of "**variables**" are as below;

a) Tempest Variables

##channel## : Stores channel id value

##xmltv_id## : Stores channel name value

##cclogo## : Stores channel logo value

##urldateX## : Stores calculated urldate value. **X** may change from **1** to **4** based on enabled url no and may be used in channel creation with same string.

##urldateX_Y## : Stores calculated splitted("**||#split#**") urldate value based on user input. **X** may change from **1** to **4** based on enabled url no while **Y** may change from **1** to **5** and may be used in channel creation with same string.

##stopdateX## : Stores calculated stopdate value. **X** may change from **1** to **4** based on enabled url no and may be used in channel creation with same string.

##stopdateX_Y## : Stores calculated splitted("**||#split#**") stopdate value based on user input. **X** may change from **1** to **4** based on enabled url no while **Y** may change from **1** to **5** and may be used in channel creation with same string.

##subpageX## : Stores calculated/given subpage value(s). **X** may change from **1** to **4** based on enabled url no and may be used in channel creation with same string.

##grabber_X## : Stores captured value as per given regex pattern. **X** may change from **1** to **8** based on enabled url no and may be used in channel creation with same string.

##dgrabber_X## : Stores captured value from detail pages as per given regex pattern. **X** may change from **1** to **4** based on enabled detail page url no.

##pagekey1## : Stores captured value as per given regex pattern for detail page request.

##pagekey2## : Stores captured value as per given regex pattern on detail or index("||**#index#**") page for detail page request.

##cctag## : Stores captured or set channel tag value from channel creation.

##ccsubpage## : Stores captured or set channel subpage value from channel creation.

##ckey## : Stores given "**User Key**" value for channel creation.

##usertime## : Stores timezone settings of user for Timezone element

##detail## : Stores captured the highest available detail page result of a XMLTV element to transfer its desired lower counter partner. Single elements results will be transferred as it is while multiple element results will be converted to string with " , " separation among them. This separation is for making it easier to reconvert final result into array if needed.

b) Tempest Operators

In addition to special date/time operators which are already explained in "**Url(1-4) Urldate Format**" section, below special operators may be used for Tempest;

##now#format# : Calculates and replace itself with recent date/time value in requested format on time of call.

Example:

`##now#Y-m-d# => 2022-01-31` (as today's date)

`##now#H:i:s# => 10:57:46` (as current time)

##showtime#format# : Calculates and replace itself with date/time value of show's start time in requested format on time of call. It is useful for detail page requests.

Example:

Show's start time => 20220131091500 +0300

`##showtime#Y-m-d# => 2022-01-31`

`##showtime#H&Y# => 09&2022`

#counter#X(#Y)#Z : Calculates and replace itself with resulting array based on given start, stop and interval. It may be used for subpages and channel creation elements as explained in previous chapter.

Note: From this point, combo operators of Tempest will be explained. Combo operators allow you to perform multiple search in a single line regex pattern and even keep/modify/combine these results as final. I named them as "Shikai" and "Bankai" with different levels (Yes, I am an old Bleach fan..)

| (Special Operator) (Shikai Lvl1) : Special operator of Tempest to set multiple and unlimited regex patterns for XMLTV elements. Tempest will parse regex patterns and search synchronously till any of them found.

Example:

Data => show:[title:"asd",shortdesc:"fgh",desc:"jkl"]

Desc pattern => desc:"(.*)"|shortdesc:"(.*)"

Result => fgh (since 2nd pattern is the first found in data)

|>| (Special Operator) (Shikai Lvl2) : Special operator of Tempest to set multiple and unlimited regex patterns for XMLTV elements based on left(high) to right(low) priority order. Tempest will parse regex patterns and search based on priority till find first non-empty result.

Example:

Data => show:[title:"asd",shortdesc:"fgh",desc:"jkl"]

Desc pattern => desc:"(.*?)"|>|shortdesc:"(.*?)"

Result => jkl (1st pattern is high priority and not empty)

Note: From this point, it is recommended to try/use after gaining some experience with regex patterns and Tempest.

|#| (Special Operator) (!Bankai! Lvl1) : This is the most powerful operator of Tempest. It has been briefly explained for "Custom Headers" and "Cookie Headers" to separate string header into multiple headers but its ability is not limited with this. It can also search and combine unlimited of regex patterns as a final result.

Example:

Data => show:[title:"asd",shortdesc:"fgh",desc:"jkl"]

Desc pattern => desc:"(.*?)"|#|shortdesc:"(.*?)"

Result => jklfgh (Bankai result added to end of first result)

For multiple xmltv elements

Data => show:[title:"asd",genre1:"fgh",genre2:"jkl"]

Category pattern => genre1:"(.*?)"|#|genre2:"(.*?)"

Result => fgh

jkl (Bankai result added to end of array result)

|#|[[X]] (Special Operator) (Bankai Lvl2) : This is the first power-up for bankai. In this level the strings given as highlighted "X", will be added to start of bankai result if it is not empty. It will add new line between results if "\n" value is given.

Example:

Data => show:[title:"asd",shortdesc:"fgh",desc:"jkl"]

Desc pattern => desc:"(.*?)"|#|[[string]]shortdesc:"(.*?)"

Result => jklstringfgh

For multiple xmltv elements

Data => show:[title:"asd",genre1:"fgh",genre2:"jkl"]

Cat. pattern => genre1:"(.*)"|#|**string**]genre2:"(.*)"

Result => fgh

stringjkl

|#|[[~X~]] (Special Operator) (Bankai Lvl3) : This is the second power-up for bankai. In this level the strings given as highlighted "**X**", shall be a regex pattern and result will be added as header to bankai result. This lets Tempest to capture dynamic header values.

Example:

Data => show:[title:"asd",shortdesc:"fgh",desc:"jkl"]

Desc pattern => desc:"(.*)"|#|[[~**title:"(.*)"**~]] shortdesc:"(.*)"

Result => jkl**asd**fgh

For multiple xmltv elements

Data => show:[title:"asd",genre1:"fgh",genre2:"jkl"]

Cat. pattern => genre1:"(.*)"|#|[[~**title:"(.*)"**~]] genre2:"(.*)"

Result => fgh

asdjkl

|#|[[Y~X~Z]] (Special Operator) (Bankai Lvl4) : This is the last power-up for bankai. In this level the strings given as highlighted "**X**", shall be a regex pattern and result will be added as header to bankai result with prefixed by "**Y**" string and suffixed by "**Z**" user defined strings. This lets Tempest to edit captured dynamic header values. This is the most powerful form of bankai.

Example:

Data => show:[title:"asd",shortdesc:"fgh",desc:"jkl"]

Desc pattern => desc:"(.*)"|#|[[**123**~**title:"(.*)"**~**456**]] shortdesc:"(.*)"

Result => jkl**123asd456**fgh

For multiple xmltv elements

Data => show:[title:"asd",genre1:"fgh",genre2:"jkl"]

Cat. pattern => genre1:"(.*)"#|123~title:"(.*)"~ 456|genre2:"(.*)"

Result => fgh

123asd456jkl

4) Tempest Commands & Syntax

regexpattern || #comm1# || #comm2#.. || #commN#

Tempest have several commands which may be used globally or limited with some dedicated elements/headers to achieve desired changes/modifications on result output. Tempest commands are may be repeated/linked each other unlimitedly and each command will modify the output of previous command based on its own ability. The list and description of built-in Tempest commands as below;

|| #replace#(X) || Y : Replace command finds the required portion of element based on given regex pattern(**X**) and replace finding(s) with given user string **Y**. It supports unlimited replace operations in a single command with "##" separation. Also Tempest variables both in regex pattern and replace string will be converted to their dynamic results prior to operations. It is used for deletion operations as well with given empty replace string.

Example:

Data => show:[title:"asd123fgh",desc:"jkl456QWE"]

Title pattern => "title:"(.*)"#||#replace#(\d+)|| K_"

Title Result => asd K_fgh

(\d+) regex will capture all recursive numbers as a block and replace them with string " K_"

Title pattern => "title:"(.*)"#||#replace#(s)##(23)||S##"

Title Result => aSd1fgh

First, (s) regex will capture all "s" letters and replace them with string "S" and then second regex will capture all "23" strings and delete them while second string is empty

||#add#X : Add command assigns given user string "X" as value of element. It **stops element's regex search**, means regex pattern given before will be interpreted as string, when used which is useful for setting channel logo with **##cclogo##** variable.

Example:

Data => show:[title:"asd123fgh",desc:"jkl456QWE"]

Desc pattern => desc:"(.*)" ,||#add#789

Desc Result => desc:"(.*)" ,789

(Above will not be interpreted as regex pattern)

Ch. logo pattern =>

||#add#http://example/##cclogo##.png

Ch. logo Result => http://example/123.png

||#addstart#X : Addstart command inserts given user string "X" to the beginning of element value.

Example:

Data => show:[title:"asd123fgh",desc:"jkl456QWE"]

Desc pattern => desc:"(.*)" ,||#addstarts#789

Desc Result => 789jkl456QWE

||#addend#X : Addend command inserts given user string "X" to the end of element value.

Example:

Data => show:[title:"asd123fgh",desc:"jkl456QWE"]

Desc pattern => desc:"(.*)" ,||#addend#789

Desc Result => jkl456QWE789

||#include#X#N : Include command filters element value based on given user string "X" and only keep value(s) which contains

string "**X**". Multiple search key may be added separated with "**#**". Search keys/words shall be carefully selected on multiple keys due to filtering will be done recursively on included results. It is also case-sensitive mode by default.

Example:

Data => show:[title:"asd123fgh",desc:"jkl456QWE"]

Desc pattern => desc:"(.*?)"||#include#asd

Desc Result => "" (No "asd" exists in desc value, result will be empty)

Title pattern => title:"(.*?)"||#include#as#gh

Title Result => "asd123fgh" (Both "as" and "gh" exists in title value, result will not filtered)

||#exclude#X#N : Exclude command filters element value based on given user string "**X**" and only keep value(s) which not contains string "**X**". Multiple search key may be added separated with "**#**". Search keys/words shall be carefully selected on multiple keys due to filtering will be done recursively on excluded results. It is also case-sensitive mode by default.

Example:

Data => show:[title:"asd123fgh",desc:"jkl456QWE"]

Desc pattern => desc:"(.*?)"||#exclude#asd

Desc Result => "jkl456QWE" (No "asd" exists in desc value, result will not be filtered)

Title pattern => title:"(.*?)"||#exclude#as#jk

Title Result => "" ("as" exists but "jk" not exists in title value, result will be filtered due to "as" match)

||#max#X : Max command applies quantity limit to multiple value elements based on given user number "**X**". If element has more values than requested, excessive ones will be deleted.

Example:

Data => show_detail:[cat:"asd",cat:"jkl",cat:"fgh",cat:"123"]

Category pattern => cat:"(.*?)"||#max#2

Desc Result => "asd"

"jkl"

Captured "fgh" and "123" results deleted due to excess quantity

||#split#X : Split command will explode element value based on given regex pattern "**X**" and treat remaining portions as multiple element values. Since it is the most powerful and dangerous command which is able to change data structure, its usage is limited with multiple value elements. When used on already multiple value containing element, split pattern will be applied each value individually and remaining portions will create new values.

Example:

Data => cast:[actor:{"asd","jkl","fgh","cvb"}]

Actor pattern => actor:{"(.*)"}||#split#(",")

Actor Result => "asd"

"jkl"

"fgh"

"cvb"

(Regex pattern captured value as a string 'asd','jkl','fgh','cvb' and ||#split# command exploded all '"','"' and converted remainings to value)

If it is used on already multiple value containing element, will check all values for match and explode them while keeping remaining as new values as below;

Actor pattern => "(.*)"/>||#split#(k|v)

Actor Result => "asd"

"j" ("jkl" has been splitted from "k")

"k"

"fgh"

"c" ("cvb" has been splitted from "v")

"b"

||#date#X : Date command will calculate element date value based on given user input "X". User input may be in "day", "hour", "minute", "second", "month", "year" and their combination formats. It can be used on **all object date/time operations** such as start time, stop time, urldate formats, stopdate formats and parameters which are already explained in "Urldate format"

Example:

Data => start:"2022-02-01 02:30:00"

Start pattern => start:"(.*)"||#date#+2hour-5minutes

Start Format pattern => "Y-m-d H:i:s"

Start Result => "2022-02-01 04:25:00"

||#calc#(Y)||X : Calc command will calculate element numeric value based on given user input "X". User input may be arithmetic operators "+", "-", "*", "/", "%" and their combination formats. Also with optional "(Y)" addition which "Y" indicates valid regex pattern, only the location of calculation may be targeted. It may be used for all numeric values.

Example:

Data => date:"2014"

Production Date pattern => date:"(.*)"||#calc#+6*2/5

Production Date Result => "808" (Following operations done as;

2014 + 6 = 2020 => 2020 * 2 = 4040 => 4040 / 5 = 808)

Production Date pattern => date:"(.*)"||#calc#(\d{2})\$||+6*2/5

Production Date Result => "208" (Regex pattern addressed the last 2 digits of capture which is "14" then following operations done as;

14 + 6 = 20 => 20 * 2 = 40 => 40 / 5 = 8 => 14 has been replaced with result 8 in main element value 2014)

||#hash#method#(Y) : Hash command will calculate element's hash value in given method. Also with optional "(Y)" addition which

"Y" indicates valid regex pattern, only the location of calculation may be targeted. It may be used for any element.

Example:

Data => title:"Hello"

Title pattern => title:"(.*?)"||#hash#md5#

Title Result => 8b1a9953c4611296a827abf8c47804d7 (md5 hash of Hello)

Title pattern => title:"(.*?)"||#sha1#(||)

Title Result => He110c8a30c16070bf2813480d9492a1a170a7d80ao (sha1 hash of captured "||" replaced in element value)

||#roman#(Y) : Roman command will convert element's roman letter value to its numeric sum value. Also with optional "(Y)" addition which "Y" indicates valid regex pattern, only the location of calculation may be targeted. It may be used for any element. If element value or targeted value is not a valid roman number, it will skip operation. Roman command works in case-insensitive mode.

Data => season:"XIV"

Season pattern => season:"(.*?)"||#roman#

Season Result => 14 (numeric value of "XIV")

Season pattern => season:"(.*?)"||#roman#(.)\$ (indicates last digit)

Season Result => XI5 (converted numeric value of captured "V" replaced in element value)

||#convert#X#Y : Convert command will convert element's date/time value which given in "X" format into requested "Y" format. It may be used on XMLTV element's date/time operations such as "start time", "stop time", "production date" and "previously-shown". If no value is given for request format "Y", it will be considered as "Y" (year).

Data => date:" 1099872000" (Unix value of "2004-11-08 00:00:00 GMT")

Production Date pattern => date:"(.*?)"||#convert#Unix#Y-m-d

Production Date Result => 2004-11-08 (date value of unix timestamp)

Production Date pattern => date:"(.*?)"||#convert#Unix

Production Date Result => 2004 (year value of unix timestamp)

||#upper# : Upper command will convert all letters to uppercase. It may be used in any element.

Example:

Data => title:"Hello"

Title pattern => title:"(.*)"||#upper#

Title Result => HELLO

||#lower# : Lower command will convert all letters to lowercase. It may be used in any element.

Example:

Data => title:"Hello"

Title pattern => title:"(.*)"||#lower#

Title Result => hello

||#word# : Word command will convert all the first letters to uppercase and other letters of each word to lowercase. It may be used in any element.

Example:

Data => title:"HELLO WORLD"

Title pattern => title:"(.*)"||#word#

Title Result => Hello World

||#sentence# : Sentence command will convert all the first letters of a sentence to uppercase and other letters to lowercase based on punctuation marks. It may be used in any element.

Example:

Data => title:"HELLO WORLD. this is TEMPEST!"

Title pattern => title:"(.*)"||#sentence#

Title Result => Hello world. This is tempest!

||#repeat#X#Y : Repeat is a limited with "**show**" element command to tell Tempest from which block it will start capturing as

"**X**" (channel position) and with which interval it will be repeated as "**Y**" (next start position). When "special pattern" applied, interval value "**Y**" may be omitted.

||#format#X : Format is a limited with "**duration**" element command to tell Tempest format of captured duration as "**X**". Format may be set as "ms" (microsecond), "s" (second), "m" (minute by default), "h" (hour) or "t" (time pattern).

||#culture#X : Culture is a limited with "**original title**" element command to tell Tempest 2 letter culture code of captured value as "**X**".

||#unit#X : Unit is a limited with "**length**" element command to tell Tempest unit of captured value as "**X**". Unit may be set as "**s**" (second), "**m**" (minute by default) or "**h**" (hour).

||#name#X : Name is a limited with "**season**" and "**episode**" element command to tell Tempest change the on-screen prefix of captured value as "**X**".

||#type#X : Type is a limited with "**subtitles**" element command to tell Tempest type of captured value as "**X**". Type may be set as "**t**" (teletext by default), "**o**" (onscreen) or "**d**" (deaf-signed)

||#urlX# : This is a limited with page and channel page "**grabbers**" element command to tell Tempest higher level unused grabbers to forward on given url no as "**X**".

||#durlX# : This is a limited with detail page "**grabbers**" element command to tell Tempest higher level unused grabbers to forward on given url no as "**X**".

||#index# : This is a limited with detail page "**page2key**" and "**grabbers**" element command to tell Tempest forward them on show blocks.

||#set#X: This is a limited with channel creation elements command to tell Tempest values of "**channel id**", "**channel name**" and/or "**channel logo**" will be assigned as given user input "**X**". Multiple values may be set separated by "**|**".

||#merge#X||Y: This is a limited with show and channel blocks command to tell Tempest to use regex pattern "**X**" on given source

"Y" and merge their results with corresponding show or channel block results if total numbers are equal.

||#source#X: This is a limited with channel creation elements command to tell Tempest values of "**channel id**", "**channel name**" and/or "**channel logo**" will be searched in given user input "X". With this command, each channel creation element may be searched in a different data source contained in "**grabbers**". Please note that Tempest cannot make any data alignment so different sources must have aligned datas with each other for a proper result.

Example:

2 Urls Channel Creation site with channel ids in first url response and channel name and/or channel logos in last url response

Enable Channel page grabber_1 with "(.*)" pattern so full page response will be stored in ##grabber_1##

Channel Id Pattern => regex||#source###grabber_1##

With this command and method, channel id(s) will be searched in first url response while other elements will be searched in last url response.

||#sameas#X: This is a limited with "**Tempest Configuration**" command for "**xmltv_id**" element to tell Tempest grabbing will be skipped for the containing channel and previous channel xml data will be directly set for it. Additionally, time offset may be applied as given user input "X".

Example:

Channel1 => Example Tv HD

Channel2 => Example Tv SD||#sameas#

In above example, grabbing operation will be directly skipped for Channel2 and data of Channel1 will be assigned for it.

Channel1 => Example Tv HD

Channel2 => Example Tv SD||#sameas#+2

In above example, grabbing operation will be directly skipped for Channel2 and data of Channel1 will be assigned for it with +2 hours offset for each show.

||#offset#X: This is a limited with “**Tempest Configuration**” command for “**xmltv_id**” element to tell Tempest grabbing will be completed with user given time offset as “**X**” for each show. This is useful if site has some different timezone channel from the rest of its data.

Example:

Channel1 => Example Tv HD||#offset#-1

In above example, grabbing operation will be completed for Channel1 with -1 hours offset for each show, different from the other channels in the same siteconfig

|| (Multiple Display Name) : This is a limited with “**Tempest Configuration**” command for “**display_name**” element to tell Tempest split given value with “**||**” and create separate display names for each value.

Example:

Channel1 => Channel1||Channel 1 HD||Channel 1 SD

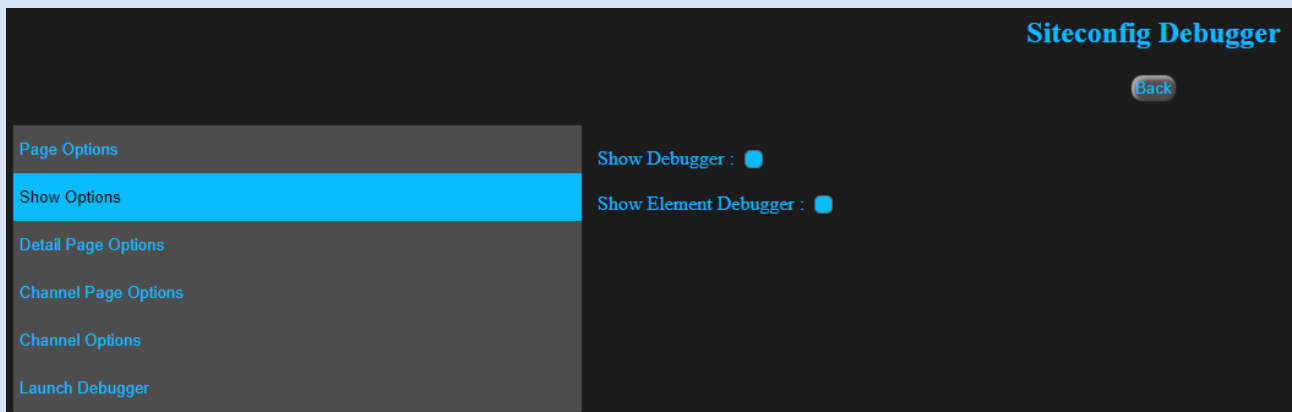
Result => <display-name lang="xx">Channel1</display-name>

<display-name lang="xx">Channel 1 HD</display-name>

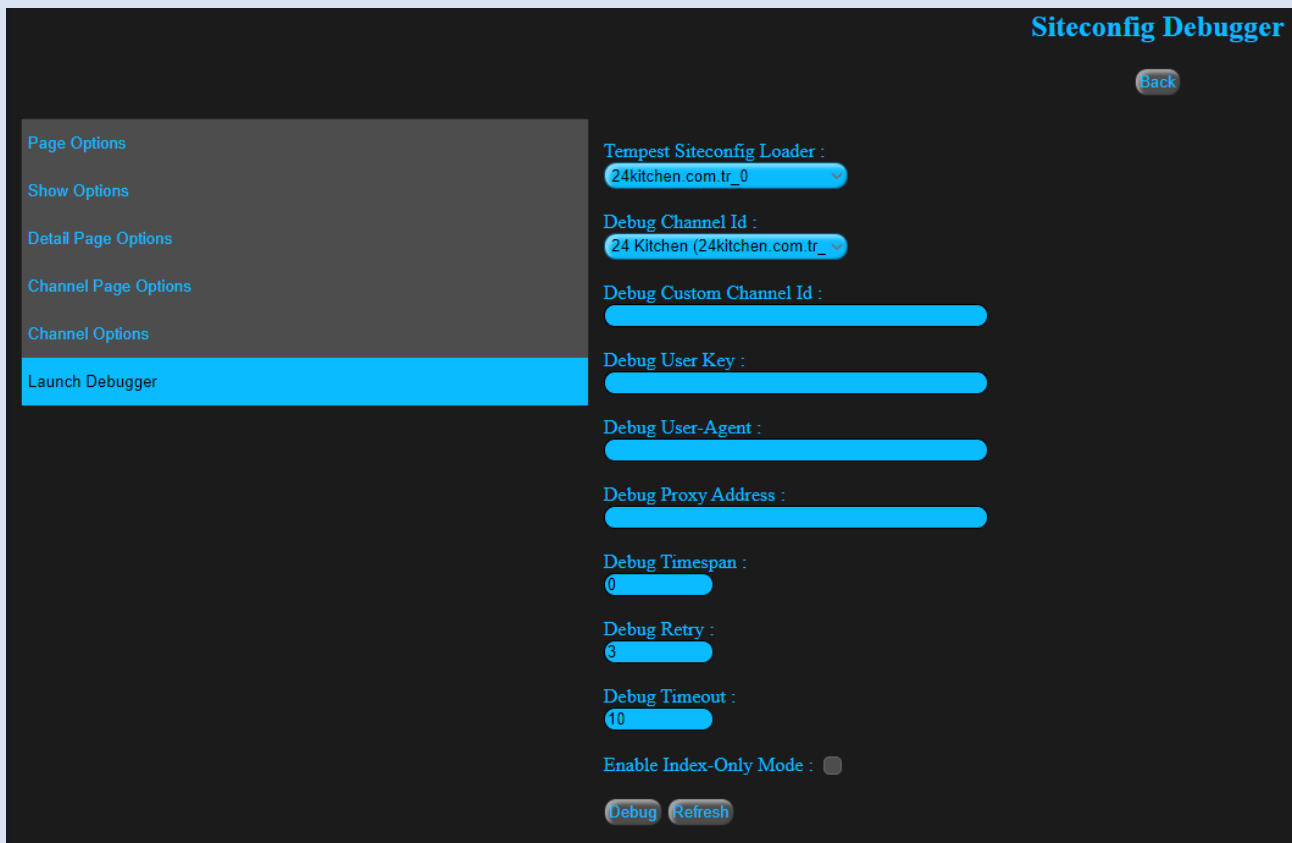
<display-name lang="xx">Channel 1 SD</display-name>

5) Siteconfig Debugger

Tempest has built-in on-screen debugger which you may test url responses (index, detail, channel), grabbers (index, detail, channel), captured show blocks, captured XMLTV elements and captured all channel elements. In addition to normal debugging, you may test your siteconfigs with custom timespans, channel ids, user keys, user agents, timeout settings and proxy addresses. It also supports “**Index-Only**” grabbing mode to temporarily disable detail pages for faster debugging.



Debugger settings may be adjusted as below;



Siteconfig Debugger

Back

Initializing...

Checking required modules...

Debugging started for 1 day

(1/1) Site: 24Kitchen.com.tr_0

Channel id: 24Kitchen

Xmltv id: Debug_24Kitchen

Show Debug

[1] => is-today" data-datetime-timestamp="1643778880" data-end-timestamp="1643773280" data-datetime-date="2022-02-02" data-id="365947278"><div class="row accordionToggle acilia-click-access"><div class="large-12 column new acilia-now-tag" style="display: none">\$ID</div><div class="large-3 medium-3 small-3 column"><h5>06:00</h5></div><div class="large-8 medium-8 small-8 column"><h3 >Masterchef</h3><!-- pg rating classification --></div><div class="large-1 medium-1 small-1 column controller tCenter">+</div></div><div class="row accordionContent acilia-schedule-event-content"><div class="large-3 medium-3 small-1 column left"></div><div class="large-8 medium-8 small-12 column description left"><h4>Ceviz, Sezon 9 | Bölüm 7 </h4><p>Gizem Kutusu Etabı'nda amatör aşçılar, cevizleri kullanarak bir yemek yapmaya çalışırlar.</p>Programa git</div></div>

[2] => is-today" data-datetime-timestamp="1643773580" data-end-timestamp="1643774880" data-datetime-date="2022-02-02" data-id="365947278"><div class="row accordionToggle acilia-click-access"><div class="large-12 column new acilia-now-tag" style="display: none">\$ID</div><div class="large-3 medium-3 small-3 column"><h5>06:45</h5></div><div class="large-8 medium-8 small-8 column"><h3 >Jamie'nin 30 Dakikalık Yemekleri</h3><!-- pg rating classification --></div><div class="large-1 medium-1 small-1 column controller tCenter">+</div></div><div class="row accordionContent acilia-schedule-event-content"><div class="large-3 medium-3 small-1 column left"></div><div class="large-8 medium-8 small-12 column description left"><h4>Doğulu Focaccia, Sezon 1 | Bölüm 13 </h4><p>30 dakikada birbirinden lezzetli yemekler yapıp sofrayı kurmaya hazır olun. Jamie Oliver, lezzetinden ödün vermeyen çabuk yemekler hazırlamanız için yanınızda</p>Programa git</div></div>

Show Element Debug

Start Time :

[1] => 20220202030000 +0000

[2] => 20220202034500 +0000

[3] => 20220202041500 +0000

[4] => 20220202044000 +0000

[5] => 20220202050700 +0000

[6] => 20220202053500 +0000

Title :

[1] => Masterchef

[2] => Jamie'nin 30 Dakikalık Yemekleri

[3] => Topraktan Sofraya

[4] => Donal'ın Aile Mutfağı

[5] => Donal'ın Aile Mutfağı

[6] => Martha'nın Fırını

[7] => Yemek Aşkına

[8] => Yemek Aşkına

[9] => Sara La Fountain ile İstanbul'un En İyileri

[10] => à la şerife

[11] => Topraktan Sofraya

[39] => 6

[40] => 4

[41] => 5

[42] => 9

[43] => 12

[44] => 6

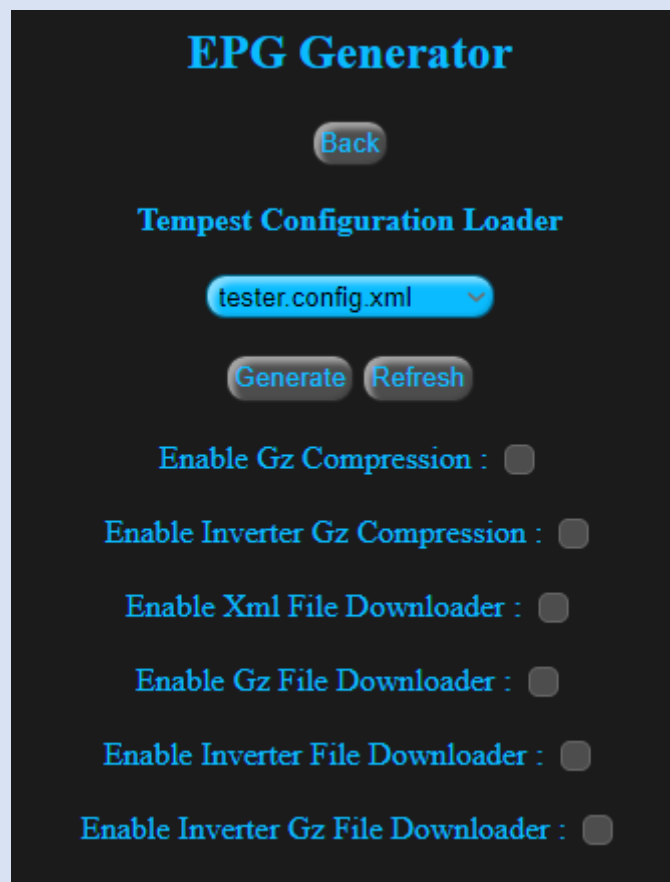
[45] => 16

Grabbing completed...
40 shows grabbed in 2.62 seconds

Tempest Overall Memory Usage: 2001KB
Tempest Peak Memory Usage: 5341KB

6) EPG Generator

Tempest's "**EPG Generator**" section is the place where you start generating your XMLTV EPG files based on selected "Tempest Configuration" file ("**tempest.config.xml**" is set by default) with additional compression and file download(only for GUI) options. Tempest may run this module both from GUI(Graphical User Interface) and CLI(Command Line Interface) to allow scheduled and headless XMLTV EPG file generation.



The screenshot shows the "EPG Generator" window with a dark background and red text. At the top is a "Back" button. Below it is the "Tempest Configuration Loader" section, which contains a dropdown menu showing "tester.config.xml". Underneath the dropdown are "Generate" and "Refresh" buttons. At the bottom, there are six toggle switches, all currently turned off, labeled: "Enable Gz Compression", "Enable Inverter Gz Compression", "Enable Xml File Downloader", "Enable Gz File Downloader", "Enable Inverter File Downloader", and "Enable Inverter Gz File Downloader".

Tempest Configuration Loader : Selection box which as drop-down list of created Tempest Configuration files, to specify configuration file to be used. If not set, "**tempest.config.xml**" will be used as default

Enable Gz Compression : Indicator for Tempest to create ".gz" (gunzip) compressed copy of generated ".xml" file.

Enable Inverter Gz Compression : Indicator for Tempest to create “.gz” (gunzip) compressed copy of generated inverted “.xml” file.

Enable Xml File Downloader : Indicator for Tempest to initialize downloading of generated “.xml” file.

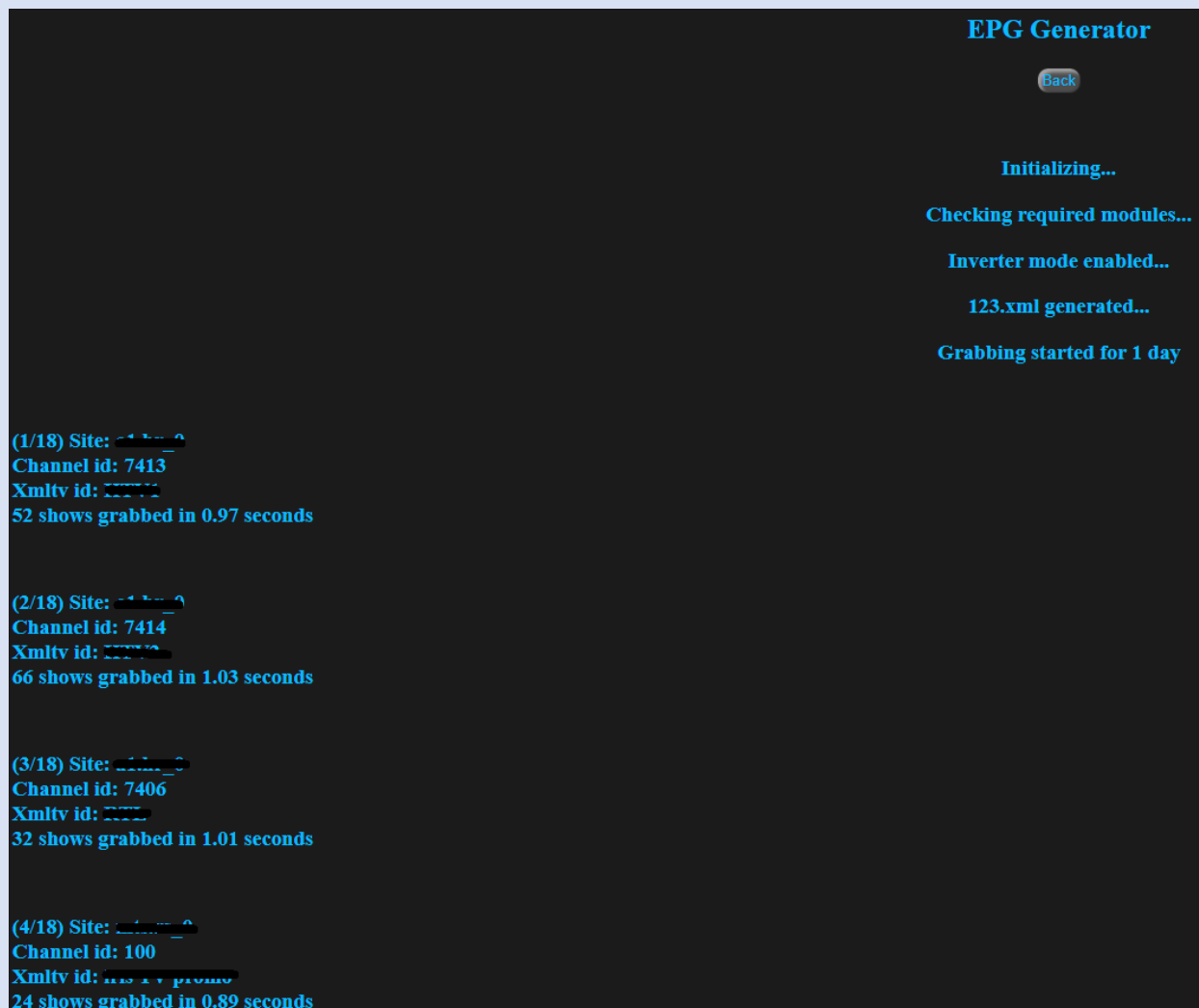
Enable Gz File Downloader : Indicator for Tempest to initialize downloading “.gz” (gunzip) copy of generated “.xml” file.

Enable Inverter File Downloader : Indicator for Tempest to initialize downloading of generated inverted “.xml” file.

Enable Inverter Gz File Downloader : Indicator for Tempest to initialize downloading “.gz” (gunzip) copy of generated inverted “.xml” file.

Note: Downloading options will only work on GUI mode.

Note: If multiple downloading option is enabled, a single “.zip” file which contains all requested files, will be downloaded.




```
(15/18) Site: [REDACTED]_3
Channel id: 88
Xmltv id: [REDACTED]
10 shows grabbed in 1.87 seconds

(16/18) Site: [REDACTED]_3
Channel id: 89
Xmltv id: [REDACTED]
9 shows grabbed in 1.50 seconds

(17/18) Site: [REDACTED]_1
Channel id: 2105
Xmltv id: [REDACTED]
31 shows grabbed in 0.85 seconds

(18/18) Site: [REDACTED]_1
Channel id: 2006
Xmltv id: [REDACTED]
31 shows grabbed in 0.62 seconds

Grabbing completed...
563 shows grabbed in 26.24 seconds

Tempest Overall Memory Usage: 2028KB
Tempest Peak Memory Usage: 5338KB
```

For using Tempest's "**EPG Generator**" on **CLI** (Command Line Interface) mode;

Note: Below explanations done as considering php defined in \$PATH enviroment. Otherwise use absolute file paths.

**" php /file/path/of/tempest.php engine=Generate
tempconfig=tester.config.xml createxmlgz=on "**

engine=Generate (Mandatory parameter)

tempconfig=tester.config.xml (Optional parameter / Name of configuration file to be used. If not set, default config will be used)

createxmlgz=on (Optional parameter / Indicator for ".gz" compressed file creation)

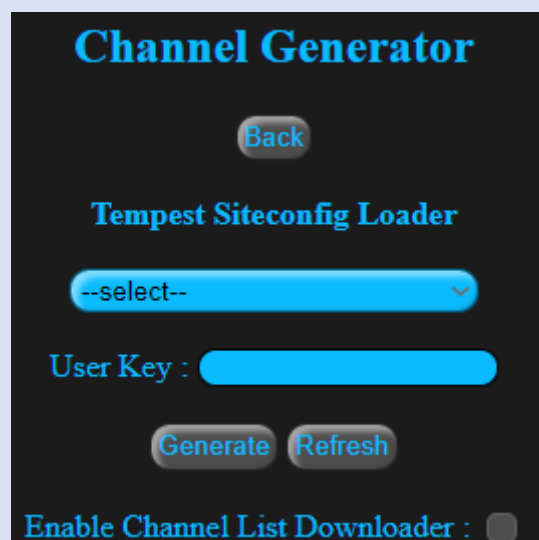
createinvxmlgz=on (Optional parameter / Indicator for ".gz" compressed inverted file creation)

Note: Due to user permissions, it may be required to call with "sudo" on UNIX.

```
[ ] Tempest EPG Generator
[ ]
[ ] The Most Advanced Programmable
[ ] GUI XMLTV EPG Generator
[ ] by Kivanc Altug aka "Kvanc"
[ ] -----
[02-02-2022 11:34:10] Tempest Version : 10.0
[02-02-2022 11:34:10] System Info : Windows NT WDS4JL3144 10.0 build 18363 (Windows 10) AMD64
[02-02-2022 11:34:10] PHP Version : 7.3.8
[02-02-2022 11:34:10] Server Info : Command Line Interface (Cli)
[02-02-2022 11:34:10] Initializing...
[02-02-2022 11:34:10] Checking required modules...
[02-02-2022 11:34:10] Inverter mode enabled...
[02-02-2022 11:34:10] 123.xml generated...
[02-02-2022 11:34:10] Grabbing started for 1 day
[ ]
[02-02-2022 11:34:10] (1/18) Site: [redacted]
[02-02-2022 11:34:10] Channel id: 7413
[02-02-2022 11:34:10] Xmltv id: [redacted]
[02-02-2022 11:34:11] 52 shows grabbed in 1.01 seconds
[ ]
[02-02-2022 11:34:11] (2/18) Site: [redacted]
[02-02-2022 11:34:11] Channel id: 7414
[02-02-2022 11:34:11] Xmltv id: [redacted]
[02-02-2022 11:34:12] 66 shows grabbed in 0.92 seconds
[ ]
[02-02-2022 11:34:12] (3/18) Site: [redacted]
[02-02-2022 11:34:12] Channel id: 7406
[02-02-2022 11:34:12] Xmltv id: [redacted]
[02-02-2022 11:34:13] 32 shows grabbed in 0.89 seconds
[ ]
[02-02-2022 11:34:13] (4/18) Site: [redacted]
[02-02-2022 11:34:13] Channel id: 100
[02-02-2022 11:34:13] Xmltv id: [redacted]
[02-02-2022 11:34:13] 24 shows grabbed in 0.67 seconds
[ ]
```

7) Channel Generator

Tempest's "**Channel Generator**" section is the place where you can create channel list for selected siteconfig file with additional file download(only for GUI) options.

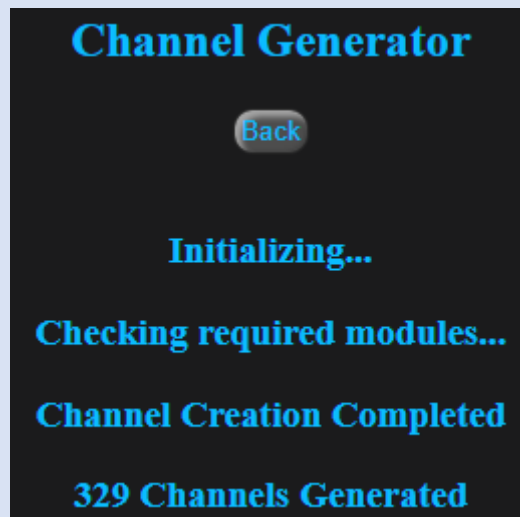


The screenshot shows the 'Channel Generator' window with a dark background and blue text. At the top is the title 'Channel Generator'. Below it is a 'Back' button. The main section is titled 'Tempest Siteconfig Loader'. It features a blue drop-down menu with '--select--' and a small blue arrow. Below the menu is a 'User Key :' label followed by a blue input field. At the bottom are two buttons: 'Generate' and 'Refresh'. At the very bottom is a checkbox labeled 'Enable Channel List Downloader :' which is currently unchecked.

Tempest Siteconfig Loader : Selection box which as drop-down list of available siteconfig files, to specify the file to be used for channel list creation.

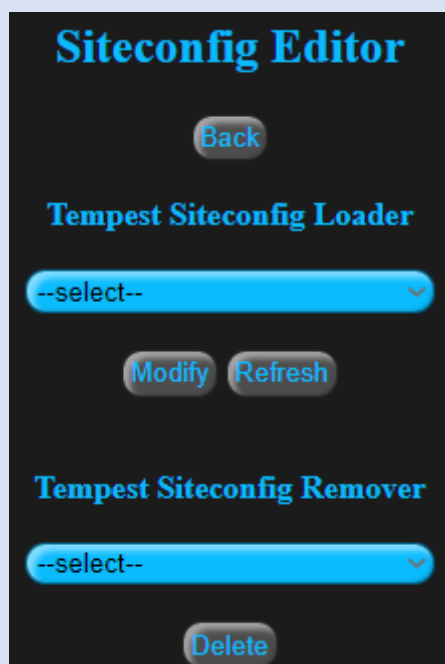
User Key : Channel creation key value as user input. It is useful for sites which are using postal/zip code etc. for channel creation. Input value may called with “**##cckey##**” variable.

Enable Channel List Downloader : Indicator for Tempest to initialize downloading of generated “channel.xml” file.



8) Siteconfig Editor

Tempest has built-in siteconfig editor to modify created and/or downloaded siteconfig files in a GUI form which is identical with “**Siteconfig Maker**”. Since the created siteconfig data is encoded by php’s itself, it is highly recommended to modify siteconfig files from built-in “**Siteconfig Editor**” to avoid any syntax error.

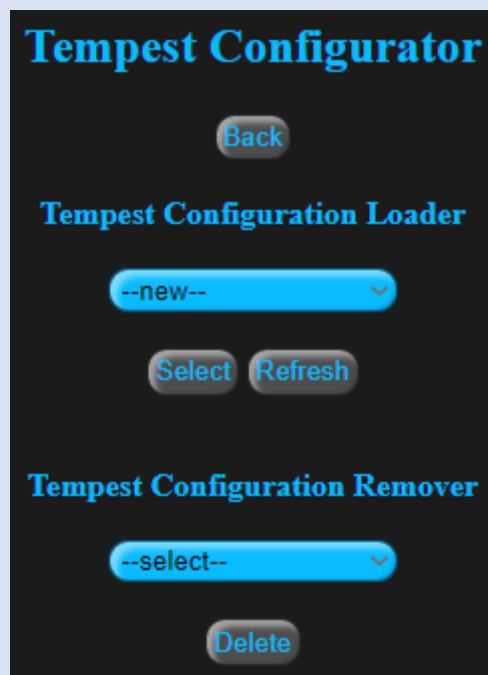


Tempest Siteconfig Loader : Selection box which as drop-down list of available siteconfig files, to specify the file to be modified.

Tempest Siteconfig Remover : Selection box which as drop-down list of available siteconfig files, to specify the file to be deleted **permanently**. When used, corresponding "channel.xml" will also be automatically deleted.

9) Tempest Configurator & Settings

Tempest has built-in configurator to create, modify or delete Tempest Configuration files which contains your selected channel(s) and EPG generating details.



Tempest Configuration Loader : Selection box which as drop-down list of available Tempest configuration files, to specify the file to be created or modified.

Tempest Configuration Remover : Selection box which as drop-down list of available Tempest configuration files, to specify the file to be deleted **permanently**.

a) File Configuration

You can adjust main settings of configuration files from "**File Configuration**" tab as below;

Configuration Name :

EPG File Name :

User Agent :

Timespan :

Retry :

Timeout :

Channel Delay :

Index Page Delay :

Detail Page Delay :

Time Converter :

Proxy :

Enable Index-Only Mode : ☐

Enable Fusion : ☐

Fusion Level :

Enable Inverter : ☐

Inverter File Name :

Inverter Path :

Enable Logger : ☐

Configuration Name : Name of the configuration file to be created. It is only available when “**new**” is selected.

EPG File Name : Name of the EPG “.xml” file to be created.

User-Agent : String of user-agent that will be used during web requests of all channels included in configuration, to emulate a web browser. “**User-Agent**” string which is set inside siteconfig, will take precedence over configuration string.

Timespan : Numeric timespan value to specify how many days of request(s) will be performed. “**0**” means 1 day of grabbing and increases in the same order.

Retry : Numeric retry value to specify how many times of requests will be performed in case of Url errors.

Timeout : Numeric timeout value to specify how many seconds of waiting is allowed for Url response. Note that too low values may cause timeout errors.

Channel Delay : Numeric delay value to specify how many seconds of waiting is required for the request of next channel. “**Channel Delay**” value which is set inside siteconfig, will take precedence over configuration value. Supports is upto 0.01 seconds

Index Page Delay : Numeric delay value to specify how many seconds of waiting is required for the request of next index page. "**Index Page Delay**" value which is set inside siteconfig, will take precedence over configuration value. Supports is upto 0.01 seconds.

Detail Page Delay : Numeric delay value to specify how many seconds of waiting is required for the request of next detail page. "**Detail Page Delay**" value which is set inside siteconfig, will take precedence over configuration value. Supports is upto 0.01 seconds.

Time Converter : String of time offset to specify resulting time offset of all shows generated by configuration file. It can be in "±XX:XX" format such as "+03:00" or with referenced patterns such as "Europe/Istanbul". To enable DST(Daylight Saving Time) option, referenced name pattern shall be set. This is useful for some PVRs which ignores time offset values or accepts only single type of time offset from EPG file.

Example:

Time Converter Value => Europe/Istanbul (equals +03:00)

Channel 1 Show Start => 20220202011000 +0100

Channel 2 Show Start => 20220202030000 +0300

Channel 3 Show Start => 20220202053000 -0200

Result in Generated Xml File=>

Channel 1 Show Start => 20220202031000 +0300

Channel 2 Show Start => 20220202030000 +0300

Channel 3 Show Start => 20220202103000 +0300

Proxy : String of proxy address that will be used for web requests. It supports http, https, socks4 and socks5 proxy types. String shall be as below examples formats with(out) username/password;

Example:

Proxy => "<https://ipaddress:port>"

Proxy => "[socks5://username:password@ipaddress:port](https://username:password@ipaddress:port)"

"Proxy" address which is set inside siteconfig, will take precedence over configuration string.

Enable Index-Only Mode : Indicator for Tempest to skip detail page request of all channels included in configuration. This is useful for quick EPG generation with less detail data(Only available in index page elements will be captured).

Enable Fusion : Indicator for Tempest to enable fusion module which protects existing day's data and allows to keep/merge previous day(s) data as specified in **"Fusion Level"** upto 15 days. This is useful for some PVRs and catch-up services. It is also highly recommended to enable **"Fusion Mode"** with **"Fusion Level"** as **"0"** to running Tempest in incremental mode.

Fusion Level : Numeric value to specify how many days of previous dated data will be kept/merged with newly captured data. **"0"** means current day shows and increases in the same order as previous day(s).

Enable Inverter : Indicator for Tempest to enable inverter module which creates modified copy of the generated EPG ".xml" file as per user defined sequence/style by **"Inverter Path"**. This is useful for some PVRs which is not supporting all XMLTV elements so requested elements may be transferred to "description" in specified location/style etc.

Inverter File Name : Name of the inverted EPG ".xml" file to be created. If not set and **"Inverter"** is enabled, **"EPG File Name"** will be used with "_inv.xml" suffix.

Inverter Path : String of user defined sequence/style for fusion module to perform requested modifications. Supported elements with keywords;

Element Name	Keyword	Element Name	Keyword
Original Title	titleoriginal	Original Language	original-lang
Sub-title	subtitle	Custom Episode 1	customepisode1
Description	desc	Custom Episode 2	customepisode2
Episode	episode	Actor	actor
Production Date	date	Director	director
Category	category	Presenter	presenter
Language	language	Producer	producer

Element Name	Keyword	Element Name	Keyword
Composer	composer	Premiere	premiere
Commentator	commentator	Audio	audio
Writer	writer	Length	length
Editor	editor	Previously-Shown	prev_shown
Adapter	adapter	Keyword	keyword
Guest	guest	Rating	rating
Country	country	Star Rating	star-rating
Video Aspect	aspect	Last Chance	lastchance
Video Quality	quality	Review	review
Video Colour	colour		

Elements to be inverted shall be written between “##” and separated from other elements with pipes “||” as given below;

##Element1keyword## || ... || ##ElementNkeyword##

Example:

Inverter Path => ##titleoriginal## || ##desc## || ##date##

Above example will move “**Original Title**” value (if exists) to the beginning of “**Description**” while moving “**Production Date**” value (if exists) to the end of “**Description**”. Both “**Original Title**” and “**Production Date**” tags will be automatically removed from inverted copy of generated EPG file.

Since “**Description**” tag is the reference point for inverter module, if “##desc##” is not defined in “**Inverter Path**”, all sequence will be added to the end of “**Description**” value.

User defined input for each keyword may be added between header and footer “#” for modify element value. Also newlines may be added with “\n”.

Example:

Inverter Path => #Episode title: #subtitle#\n# || ##desc## || #\nDate: #date##

Above example will add “Episode title: ” prefix to “**Subtitle**” value and add a new line before “**Description**” value. Also “**Production Date**” value will be prefixed with a new line and “Date: ” word.

Multiple value elements such as "**Category**", "**Actor**", "**Director**" etc. may be separated with string defined in "(")" which will be added to keyword.

Example:

Inverter Path => ##desc#\n# ||#Directors: #director(,)##

Above example, if "Director" has multiple values, they will be converted into string which each value separated with "," and final string will be prefixed with "Directors: " string.

Element systems such as "**Rating**", "**Star Rating**", "**Season**", "**Episode**", "**Custom Episodes**" etc. may be added to inverted data with "{system}" syntax.

Example:

Inverter Path => ##desc#\n# ||#Rating[{system}]: #rating##

Above example, "Rating System" value will be added into prefix string such as "Rating[PEGI]: ".

If "**Episode System**" of siteconfig set to "xmltv_ns" and "episode" keyword is set in "**Inverter Path**", value will be automatically converted to "**on-screen**" system format.

If "**Role**" element is not empty, its value will be automatically added to linked "**Actor**" value followed by " (**Role**)" suffix.

Real Application Example:

Inverter Path =>

```
#Episode Title: #subtitle()\n# ||#Season/Episode:  
#episode#\n# ||##desc## ||#\nGenre:  
#category(,)## ||#\nActors: #actor(,)## ||#\nDirectors:  
#director(,)## ||#\nDate: #date## ||#\nRating[{system}]:  
#rating## ||#\nVideo Quality: #quality## ||#\nAudio: #audio##
```

Above "**Inverter Path**" applied a channel for some XMLTV elements and results for comparison for the same show as below;

Result of Generated Xml =>

```
<title lang="en">Judge Judy</title>
<sub-title lang="en">How Do You Support Yourself?!; Don't Cross the
Yellow Line!</sub-title>
<desc lang="en">A young couple asked to account for their income have a
colorful history; an uninsured motorist is questioned.</desc>
<credits>
  <actor>Judy Sheindlin</actor>
</credits>
<date>2017</date>
<category lang="en">Reality</category>
<category lang="en">Law</category>
<icon src="
http://tvty.tmsimg.com/assets/p184382\_b\_v12\_ak.jpg?w=240&h=360"/>
<episode-num system="xmltv_ns">20.116.</episode-num>
<video>
  <quality>HDTV</quality>
</video>
<audio>
  <stereo>Stereo</stereo>
</audio>
<previously-shown start="20170201000000"/>
<subtitles type="teletext"/>
<rating system="TVPG">
  <value>TVPG</value>
</rating>
```

Result of Inverted Xml =>

```
<title lang="en">Judge Judy</title>
<desc lang="en">Episode Title: How Do You Support Yourself?!; Don't
Cross the Yellow Line!
Season/Episode: S21 E117
A young couple asked to account for their income have a colorful history; an
uninsured motorist is questioned.
Genre: Reality, Law
Actors: Judy Sheindlin
Date: 2017
Rating[TVPG]: TVPG
Video Quality: HDTV
Audio: Stereo</desc>
  <icon src="
http://tvty.tmsimg.com/assets/p184382\_b\_v12\_ak.jpg?w=240&h=360"/>
  <previously-shown start="20170201000000"/>
  <subtitles type="teletext"/>
```

Enable Logger : Indicator for Tempest to enable logger in order to record real-time grabbing operation details in "**Tempest_log.txt**" file.

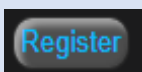
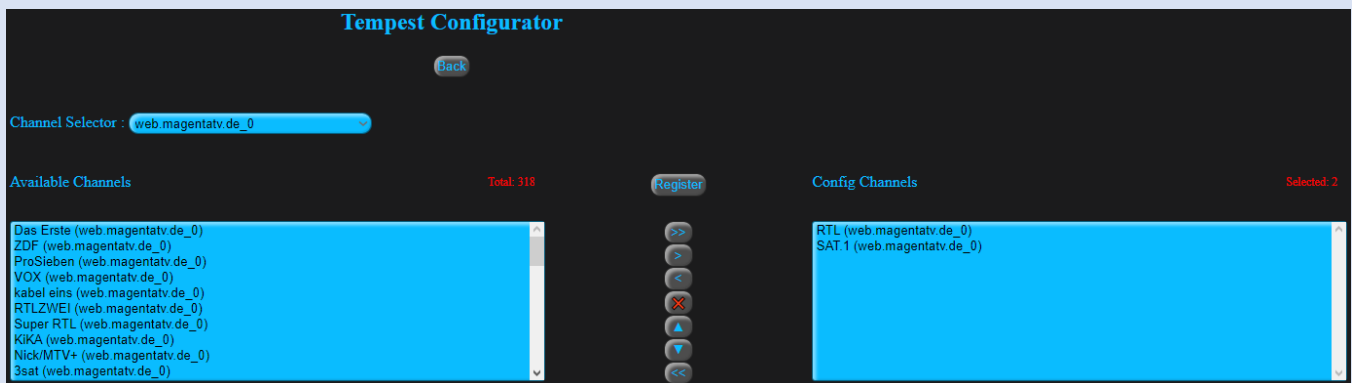
b) Channel Configuration

You can multiply add, delete and/or change locations of selected channels from "**Channel Configuration**" tab as below;

Channel Selector : Selection box which as drop-down list of available siteconfig files, to list their available channels in "**Available Channels**" box.

Available Channels : List of available channels belongs to chosen siteconfig file on "**Channel Selector**" and its content will be automatically updated in each change.

Config Channels : List of added/selected channels belongs to chosen Tempest configuration file.



Register will send selected channels to "**Channel Indexer**" tab. Without clicking it, changes will not take effect



Send all to "**Config Channels**"

Send selected channel(s) to "**Config Channels**"

Send selected channel(s) to "**Config Channels**"

Delete selected channel(s)

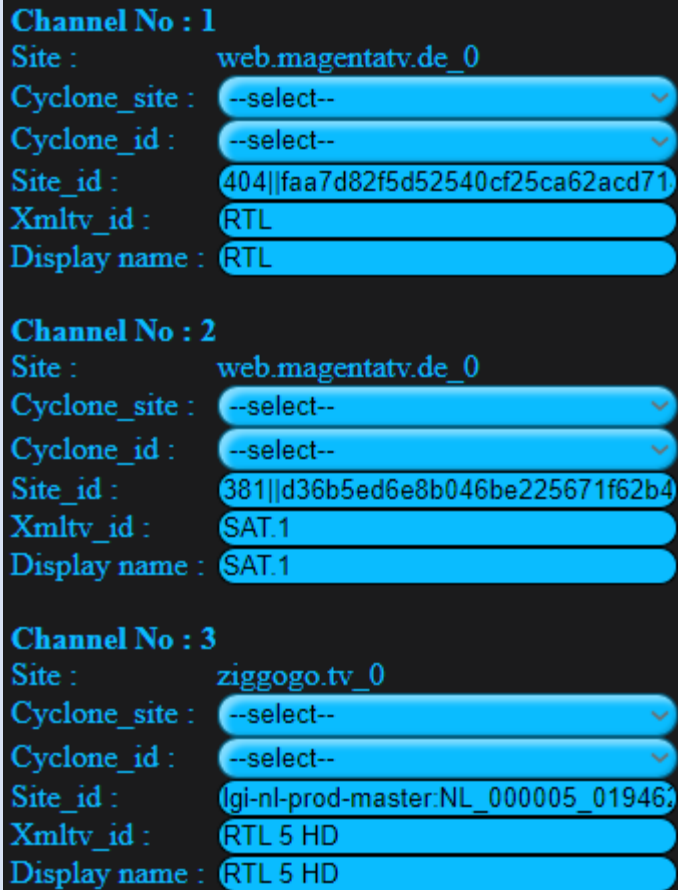
Move selected channel(s) to up

Move selected channel(s) to down

Send all to "**Available Channels**"

c) Channel Indexer

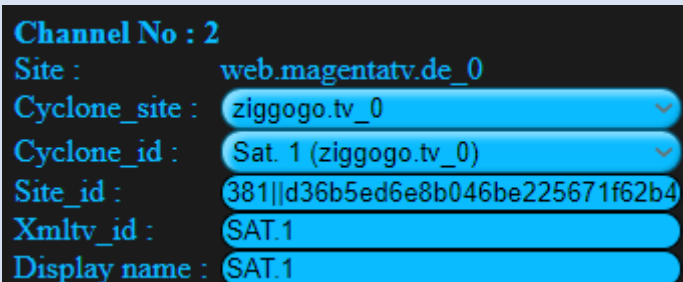
You can manually edit “**site_id**”, “**xmltv_id**” and “**display_name**” values of selected channels from “**Channel Indexer**” tab as below;



The screenshot displays the 'Channel Indexer' interface with three channels listed. Each channel has a set of input fields for configuration. Channel 1 is 'web.magentatv.de_0' with 'RTL' as the XMLTV ID and display name. Channel 2 is also 'web.magentatv.de_0' with 'SAT.1' as the XMLTV ID and display name. Channel 3 is 'ziggogo.tv_0' with 'RTL 5 HD' as the XMLTV ID and display name. The 'Cyclone' fields are currently set to '--select--'.

Channel No	Site	Cyclone_site	Cyclone_id	Site_id	Xmltv_id	Display name
1	web.magentatv.de_0	--select--	--select--	404 faa7d82f5d52540cf25ca62acd71	RTL	RTL
2	web.magentatv.de_0	--select--	--select--	381 d36b5ed6e8b046be225671f62b4	SAT.1	SAT.1
3	ziggogo.tv_0	--select--	--select--	lgi-nl-prod-master:NL_000005_01946	RTL 5 HD	RTL 5 HD

“**Cyclone Mode**” which allows users to assign alternative channel in case failure, may be set for each channel.



This screenshot shows 'Channel No : 2' with 'Cyclone Mode' activated. The 'Cyclone_site' is set to 'ziggogo.tv_0' and the 'Cyclone_id' is set to 'Sat. 1 (ziggogo.tv_0)'. The 'Site_id', 'Xmltv_id', and 'Display name' remain the same as in the previous screenshot.

Channel No	Site	Cyclone_site	Cyclone_id	Site_id	Xmltv_id	Display name
2	web.magentatv.de_0	ziggogo.tv_0	Sat. 1 (ziggogo.tv_0)	381 d36b5ed6e8b046be225671f62b4	SAT.1	SAT.1

In above example, If “**Channel No : 2**” fails on grabbing data, “**Cyclone Mode**” will be activated and will be started to grab from “**Cyclone_site**” with “**Cyclone_id**”. Resulted shows will be stored with same “**Xmltv_id**” and “**Display_name**”.

Previously explained "**xmltv_id**" commands' "**||#sameas#X**", "**||#offset#X**" and multiple "**display_name**" command "**||**"; may be set from "**Channel Indexer**" as below;

Channel No : 1	
Site :	a1.hr_0
Cyclone_site :	--select--
Cyclone_id :	--select--
Site_id :	3667 /hbo_hd/33380-7-cro-HR/hbo_h
Xmltv_id :	HBO HD
Display name :	HBO HD
Channel No : 2	
Site :	a1.hr_0
Cyclone_site :	--select--
Cyclone_id :	--select--
Site_id :	920 /hbo/9041-4-cro-HR/hbo.png
Xmltv_id :	HBO #sameas#
Display name :	HBO

Grabbing operations will be skipped for "**Channel No : 2**" and grabbed content of "**Channel No : 1**" will be assigned to "**Channel No : 2**" but with "**xmltv_id**", "**display_name**", "**channel logo**" etc. of "**Channel No : 2**"

Channel No : 1	
Site :	a1.hr_0
Cyclone_site :	--select--
Cyclone_id :	--select--
Site_id :	3667 /hbo_hd/33380-7-cro-HR/hbo_h
Xmltv_id :	HBO HD
Display name :	HBO HD
Channel No : 2	
Site :	a1.hr_0
Cyclone_site :	--select--
Cyclone_id :	--select--
Site_id :	920 /hbo/9041-4-cro-HR/hbo.png
Xmltv_id :	HBO #sameas#+1
Display name :	HBO

Grabbing operations will be skipped for "**Channel No : 2**" and grabbed and **+1 hour added for all show times** content of "**Channel No : 1**" will be assigned to "**Channel No : 2**" but with "**xmltv_id**", "**display_name**", "**channel logo**" etc. of "**Channel No : 2**"

Channel No : 1
 Site : a1.hr_0
 Cyclone_site : --select--
 Cyclone_id : --select--
 Site_id : 3667||/hbo_hd/33380-7-cro-HR/hbo_h
 Xmltv_id : HBO HD||#offset#+2
 Display name : HBO HD

Channel No : 2
 Site : a1.hr_0
 Cyclone_site : --select--
 Cyclone_id : --select--
 Site_id : 154||/cinemax/1151-10-cro-HR/cinema
 Xmltv_id : CINEMAX
 Display name : CINEMAX

Grabbed content of "**Channel No : 1**" will be modified as +2 hours for all show times while "**Channel No : 2**" is being grabbed regularly despite both channels are assigned to same siteconfig file.

Channel No : 1
 Site : a1.hr_0
 Cyclone_site : --select--
 Cyclone_id : --select--
 Site_id : 3667||/hbo_hd/33380-7-cro-HR/hbo_h
 Xmltv_id : HBO HD
 Display name : HBO HD||HBO||HBO SD||HBO FHD

Multiple "**display_name**"s have been set with "||" command and result is as below;

```
<channel id="HBO HD">
  <display-name lang="hr">HBO HD</display-name>
  <display-name lang="hr">HBO</display-name>
  <display-name lang="hr">HBO SD</display-name>
  <display-name lang="hr">HBO FHD</display-name>
  <icon src="https://www.a1.hr/var/ezflow_site/storage/images/televizija/hbo_hd/33380-7-cro-HR/hbo_hd.png"/>
  <url>http://www.a1.hr</url>
</channel>
```

d) Save & Exit

You can save your created/updated Tempest configuration file and download from "**Save & Exit**" tab as below;

Enable Configuration Downloader : ☐

Save Refresh

Enable Configuration Downloader : Indicator for Tempest to initialize downloading of generated "config.xml" file.

10) Hints & Tips

Tempest is coded to run on the highest possible speed with the lowest possible memory consumption by default but there are some user tricks which may make it run faster/smoothier. You can find some of these user tips as below;

- Tempest uses a siteconfig recognition algorithm to decide caching or deleting key parameters based on actual and following channel. This effect will be maximized with **"keepindexpage"** siteconfigs. Because of this, it is not mandatory but recommended to list channels belong to same siteconfig as groups in Tempest configuration.
- Cyclone mode also uses the same siteconfig recognition algorithm and supports **"keepindexpage"** as well. This means when cyclone mode is activated for a **"keepindexpage"** siteconfig and next normal channel is also from the same siteconfig, grabbing operation will proceed without any data or time loss.
- Tempest supports unlimited combined command syntax but each command needs extra time and memory to being executed. So it is recommended to avoid useless commands. Also when possible it is recommended to call a command with multiple request instead of calling same command multiple times with single request for each. For example, **"||#replace#req1##req2||str1##str2"** will be executed faster and will need less memory than **"||#replace#req1||str1||#replace#req2||str2"** command block.
- Tempest coded to assign highest priority for the furthest url/page/element(if you want to go that far, it should be important) and as soon as it is reached, Tempest will proceed to next stage. This means, it is to be avoided assignment of useless urls/detail pages or elements. For example, if you really don't need the value of a detail page 2 element or it is already available in index or detail page 1, do not add pattern for this detail page 2 element otherwise it will be assigned as the most needed by Tempest.