- [ ] Neue Sektiion für apt ergänzen (zugewiesen an Michael Schlenstedt)
- [ ] Neue Sektiion für backup ergänzen (zugewiesen an Michael Schlenstedt)

general.json

General

The general.json is the central configuration object of LoxBerry beginning with LoxBerry 2.2. In all versions before LoxBerry 2.2, the general.cfg (ini-style config file) was the primary configuration object of LoxBerry (and is automatically generated also by LoxBerry 2.2+).

The general.json includes all the configuration done in the widgets, including the Miniservers and it's network settings and credentials. For Perl and PHP users, it is recommended to use the library functions from LoxBerry::System (Perl) and loxberry_system.php (PHP), as they provide on-the-fly calculated information like your plugin directory and resolved network settings of the Miniservers. With other languages, like Python or Node.js, read the general.json.

The general.json (and general.cfg too) by design is thought to be **read-only** for plugin developers. Reading of the file is allowed, changes are forbidden.

Configuration settings

This is a mapping table of all general.json settings and general.cfg settings. Not all settings of general.json are present in general.cfg (and vice versa) because of new features. Key names in general.json are always *first-letter-uppercase* (e.g. `Base.Sendstatistic`, or `Ipv4address`), in general.cfg everything is *capital letters* (e.g. `BASE.SENDSTATISTIC`). Key names are not equal in all cases. Note, that LoxBerry::System/loxberry_system.php functions like get_miniservers use other casing for the values.

general.json introduced hierarchy of objects, that were not possible in the ini format. The general.json - like general.cfg - still maintains main sections as objects,, e.g. `Base`, `Miniserver`, `Network`.

Settings and mapping

general.json	LB version	general.cfg	Configured by	Possible values	Description
==== Base ====	**2.2+**	**BASE**			
Base.Clouddnsuri		CLOUDDNS	Image		
Base.Country		n/a	My LoxBerry		2 letter country code ISO 3166-1 alpha2, lower-case (most commonly the Internet TLD) Non-existant, or string "undef" if not configured by the user
Base.Lang		LANG	My LoxBerry		ISO-Code of users selected language
Base.Sendstatistic		SENDSTATISTIC	My LoxBerry		
Base.Startsetup		STARTSETUP	Wizard		
Base.Systemloglevel		SYSTEMLOGLEVEL	Image		

Base.Version		VERSION	Image & LoxBerry Update		
Base.ImageVersion		n/a	Image		The inital version LoxBerry was installed.
n/a		MINISERVERS	Miniserver widget	1-x	The number of Miniserver entries as [MINISERVERx]. Not required in general.json
n/a		INSTALLFOLDER	Image	<Path>	LoxBerrys base folder /opt/loxberry Not required anymore, use $ENV{'LBHOMEDIR'}
==== Healthcheck ====	**2.2+**	HEALTHCHECK	no UI		(no UI, read by healthcheck.cgi)
Disable_all		DISABLE_ALL			Disables all healthcheck to not send notifications/emails
Disable_<checkname>		DISABLE_<CHECKNAME>			<checkname> is the name of the specific check to disable.
==== Log2Ram ====	**2.0.0+**	**n/a**			
Log2ram.Compalg					
Log2ram.Logdisksize					
Log2ram.Manualconfigured					
Log2ram.Ramlog					
Log2ram.Size					
Log2ram.Zl2r					
==== Mqtt ====	**2.0+**	**n/a**			**Only applies if MQTT Gateway 2.0.4+ is installed**
Brokerhost			MQTT Gateway	String	Hostname/IP of the broker (e.g. "localhost" or "loxberry-partyraum")
Brokerpass			MQTT Gateway	String	Broker Password - may be empty if no password is set
Brokerport			MQTT Gateway	String	MQTT Port - "1883" or a specific port. The field is always set.
Brokerpsk			MQTT Gateway	String	TLS-PSK Pre-shared key. *Depreciated*. *Dropped with LB3.0.*
Brokeruser			MQTT Gateway	String	Broker User - may be empty if anonymous access is set at the broker
Udpinport			MQTT Gateway	String	The UDP port where MQTT Gateway is listening for incoming UDP messages
Uselocalbroker			MQTT Gateway	0/1	1 ... Broker settings are managed by MQTT Gateway plugin 0 ... Broker settings are managed by the user
Websocketport			MQTT Gateway	String	Default Port is "9001", or a specific port set by the user.
Finderdisabled	3.0+		-	0/1	Default is unset. If set, MQTT Finder will stop after initial start.
==== Network ====	**2.2+**	**NETWORK**			
Network.Friendlyname		FRIENDLYNAME			
Network.Interface		INTERFACE			
Network.Ssid		SSID			
Network.Wpa		WPA			
Network.Ipv4.Dns		DNS			
Network.Ipv4.Gateway		GATEWAY			
Network.Ipv4.Ipaddress		IPADDRESS			
Network.Ipv4.Mask		MASK			
Network.Ipv4.Type		TYPE			
Network.Ipv6.Dns		DNS_IPv6			
Network.Ipv6.Ipaddress		IPADDRESS_IPv6			
Network.Ipv6.Mask		MASK_IPv6			
Network.Ipv6.Privacyext		PRIVACYEXT_IPv6			
Network.Ipv6.Type		TYPE_IPv6			
==== Remote ====	**1.4.0+**	**n/a**			
Remote.Autoconnect					
Remote.Httpport					
Remote.Httpproxy					
==== Ssdp ====	**2.2+**	**SSDP**			
Ssdp.Disabled		DISABLED			
Ssdp.Uuid		UUID			
==== Timeserver ====	**2.2+**	**TIMESERVER**			
Timeserver.Method		METHOD			
Timeserver.Ntpserver		SERVER			
Timeserver.Timemsno		n/a			
Timeserver.Timezone		ZONE			
==== Update ====	**2.2+**	**UPDATE**			

Update.Branch		BRANCH			Default branch is 'master'. To update from a specific branch, set the branch name here.
Update.Dryrun		DRYRUN			No changes are made, no update scripts are run. (DEBUGGING)
Update.Failedscript		FAILED_SCRIPT			If one or more update scripts fail during LoxBerry Update, this value stores the lowest version number (the version number of the first failed update script)
Update.Installtype		INSTALLTYPE			
Update.Interval		INTERVAL			
Update.Keepinstallfiles		KEEPINSTALLFILES			Does not delete the downloaded and unzipped files. They stay in the /tmp/... directory. (DEBUGGING)
Update.Keepupdatefiles		KEEPUPDATEFILES			Updates LoxBerry, but does not overwrite loxberryupdatecheck.pl and loxberryupdate.pl (DEBUGGING)
Update.Latestsha		LATESTSHA			If installtype is Latest, stores the last commit sha that was installed. Required to detect if a new commit is present.
Update.max_version	2.0.0		ajax-config-handler.cgi (updates.html)		This value limits the maximal version LoxBerry Update will update to. User has to accept a warning in LoxBerry Update to update to a new major release.
Update.Releasetype		RELEASETYPE			
==== Watchdog ====	**1.4.0+**	**n/a**			
Watchdog.Enable					
Watchdog.Logging					
Watchdog.Ping					
Watchdog.Maxload1					
Watchdog.Maxload5					
Watchdog.Maxload15					
Watchdog.Minmem					
Watchdog.Maxtemp					
Watchdog.Tempsensor					
==== Webserver ====	**2.2+**	**WEBSERVER**			
Webserver.Port		PORT			
Webserver.Oldport		OLDPORT			During port change, this temporary saves the old port for rollback situations.
Webserver.Oldsslport	**3.0+**	OLDSSLPORT	services.php		During port change, this temporary saves the old port for rollback situations.
==== Miniserver ====	**2.2+**	**MINISERVERx**	Miniserver widget		
"x" is a string label with the number, The "x" object holds the data of each Miniserver					In general.json, each Miniserver is an object, named by the number ("1", "2",...) In general.cfg, each Miniserver has a section [MINISERVERx], where x is starting at 1, maximal count in BASE.MINISERVERS
Miniserver.x.Admin		ADMIN			Miniserver user (URI-encoded)
Miniserver.x.Admin_raw		ADMIN_RAW			Miniserver user (not encoded)
Miniserver.x.Cloudurl		CLOUDURL			MAC address of the Miniserver for Cloud DNS access
Miniserver.x.Cloudurlftpport		CLOUDURLFTPPORT			FTP port of the Miniserver
Miniserver.x.Credentials		CREDENTIALS			Combined Miniserver user:password string (URI-encoded)
Miniserver.x.Credentials_raw		CREDENTIALS_RAW			Combined Miniserver user:password strin (not encoded)
Miniserver.x.Encryptresponse		ENCRYPTRESPONSE			Internal, *Depreciated* since Miniserver https support. On token-based authentication, request encryption of Miniservers response. Since Miniserver Gen. 2 supports https, Loxone-proprietary encryption won't be implemented in LoxBerry.
Miniserver.x.Fulluri		n/a			This includes the full uri to connect to the Miniserver, e.g. https:%%//%%user:pass@miniserveradress:port/ The field respects to use http oder https, and to use IPv4 or IPv6, depending to users settings.
Miniserver.x.Fulluri_raw		n/a			Like Fulluri, but user:pass is not uri-encoded
Miniserver.x.Ipaddress		IPADDRESS			Hostname, IPv4 or IPv6 of the Miniserver
Miniserver.x.Ipv6format		n/a			This flag indicates, that the Ipaddress fields holds an IPv6 address. Some libraries need special escaping for IPv6:Port combinations, but not for hostnames or IPv4 addresses. This helps your code to decide.
Miniserver.x.Name		NAME			Miniserver name shown in the UI, e.g. within Miniserver dropdowns or in logfiles
Miniserver.x.Note		NOTE			A user-defined note
Miniserver.x.Pass		PASS			Miniserver password (URI-encoded)
Miniserver.x.Pass_raw		PASS_RAW			Miniserver password (not encoded)

Miniserver.x.Port		PORT			Miniserver webport for http
Miniserver.x.Porthttps		n/a			Miniserver webport for https (Miniserver does not let users change the port away from 443, but using port forwarding the port may be different)
Miniserver.x.Preferhttps		n/a			Flag (true/false) to indicate for http requests, if your library should use http or https to connect. Use this to generate the uri. Using *Fulluri* instead, will give you the correct full URI including credentials and protocol considering user settings.
Miniserver.x.Securegateway		SECUREGATEWAY			Internal, *Depreciated* since Miniserver https support. LoxBerry will not implement Loxones proprietary encryption.
Miniserver.x.Transport		n/a		"http" or "https"	
Miniserver.x.Useclouddns		USECLOUDDNS			Don't use the local *ipaddress*, but ip needs to be requested from Loxone Cloud DNS using the *Cloudurl* (MAC address). To support Cloud DNS, you need to query Loxone Cloud DNS for the ip of this Miniserver. The LoxBerry SDK (Perl, PHP), e.g. get_miniservers() does this automatically, but for other languages, you need to implement this by yourself.

Access json properties from different languages

This examples show, how to access the above listed properties

- Webserver.Port, and
- Miniserver.1.Name

Perl

- Use `decode_json` to parse the content to a `$cfg` variable
- To access Webserver.Port: `$cfg->{Webserver}→{Port}`
- To access the Miniserver.x.Name of the first Miniserver: `$cfg->{Miniserver}->{1}->{Name}`

PHP

- Use `json_decode` to parse the content to a `$cfg` object (without the assoc = true parameter, so the content is parsed to be objects
- To access Webserver.Port: `$cfg->Webserver->Port`
- To access the Miniserver.x.Name of the first Miniserver: `$cfg->Miniserver->{1}->Name`

JavaScript / Node.js

- Use `JSON.Parse` to parse the content to a `cfg` object
- To access Webserver.Port: `cfg.Webserver.Port`
- To access the Miniserver.x.Name of the first Miniserver: `cfg.Miniserver[1].Name`

Python

- Use `import json` and `cfg = json.loads` to parse the content to a `cfg` object
- To access Webserver.Port: `cfg['Webserver']['Port']`
- To access the Miniserver.x.Name of the first Miniserver: `cfg['Miniserver']['1']['Name']`

Bash

Use the pre-installed tool `jq` to assign the result of every needed line to a variable:

- To access Webserver.Port: `PORT=$(jq -r '.Webserver.Port' "$LBSCONFIG/general.json")`
- To access the Miniserver.x.Name of the first Miniserver: `MSNAME=$(jq -r '.Miniserver["1"].Name' "$LBSCONFIG/general.json")`
- Use the variables, e.g. `echo $MSNAME $PORT`

Other articles

Perl Modul LoxBerry::System::General