**Thermostat.cpp**

**Thermostat(String cal, String s, Application \*c, SensorManager \*se, IRManager \*ir,DomoticzBroadcaster \*domo)**

Constructeur d’un thermostat :

Initialise les paramètres lconfig, lsensor, lIR, file, fileCal, ldomo avec les attributs passés en paramètre.

**uint8\_t IsStarted(void)**

Retourne la variable checkConfort

**void init(void)**

Si tableau Cal est vide désactivation, libération mémoire et suppression de AlarmTab

Lecture du fichier file si échec création de file avec des valeurs par défaut sinon assotions des valeurs aux variables.

Appel de load\_Cal() .

Création ou mise à NULL du tableau AlarmTab

Appel de setAlarm en fonction des valeurs du tableau Cal.

Fait une action en fonction du mode de fonctionnement.

**void setAlarm(uint8\_t i,char mode, uint8\_t nbday,uint8\_t h,uint8\_t s)**

Réglage de l’alarme :

Celons les différent mode Off = O, Eco = E, Comfort = C, Frostfree = F dans un switch

Pour toute la semaine si nbday = 0

Sinon pour le jour indiquer

Default erreur alarme

// Alarm.alarmRepeat(17,45,0,EveningAlarm); // 5:45pm every day

//Alarm.alarmRepeat(dayofweek, hours, minutes, seconds, function);

//"dayofweek" can be dowSunday, dowMonday, dowTuesday, dowWednesday, dowThursday, dowFriday, or dowSaturday.

//1..7 sunday to saturday

//0 ALL

**void load\_Cal()**

Chargement du fichier fileCal

//1,10:00,ON,A

//1,10:00,ON,DAYOFWEEK (entre 1,...,7)

S’il n’existe pas création

Sinon chargement dans un tableau de string Cal

**void save\_Cal(int l)**

Sauvegarde le tableau Cal dans le fichier fileCal sans la ligne l correspondant au tableau Calidx.

Si le tableau Cal n’est pas vide on le supprime puis on appelle load\_cal et enfin on appel init().

**void save\_Cal(String n)**

Sauvegarde le tableau Cal dans le fichier fileCal plus une ligne n.

On appel init() puis si le tableau Cal n’est pas vide on le supprime et enfin on appelle load\_cal

**void save()**

Création d’une chaine de string formaté des valeurs Eco, Frostfree, Comfort, hysteresis, triggerHeat, prestime, presentbool, presmin, presmax.

Suppression de l’ancien fichier file.

Sauvegarde de la chine de caractères dans le fichier file.

**void globalEco()**

Enveloppe le texte « Eco » dans la variable modeCal pour setSyncProvider.

Puis si ldmo n’est pas NULL appel la fonction SendMode(de DomoticzBrodcaster.cpp) avec modeCal pour paramètre.

**void globalOff()**

Enveloppe le texte « Off » dans la variable modeCal pour setSyncProvider.

Puis si ldmo n’est pas NULL appel la fonction SendMode(de DomoticzBrodcaster.cpp) avec modeCal pour paramètre.

**void globalComfort()**

Enveloppe le texte « Comfort » dans la variable modeCal pour setSyncProvider.

Puis si ldmo n’est pas NULL appel la fonction SendMode(de DomoticzBrodcaster.cpp) avec modeCal pour paramètre.

**void globalFrostfree()**

Enveloppe le texte « Frostfree » dans la variable modeCal pour setSyncProvider.

Puis si ldmo n’est pas NULL appel la fonction SendMode(de DomoticzBrodcaster.cpp) avec modeCal pour paramètre.

**void run() //main method**

Récupération de la température (via capteur ou Domoticz, l’état et la température de la pompe.

Récupération du mode.

Récupération de la température attendue celons le mode.

Si la température réelle est < = 0 : on return.

Si une présence est détectée et que toutes les conditions sont réunies passage de la température attendue en Comfort.

Si le mode est Off que l’état de la pompe est 0(éteinte) et (température actuelle <= température attendu – hystérésis) : on envoie la commande IR pour mettre en route la pompe.

Si le mode est Off que l’état de la pompe est 1 : on envoie la commande IR pour éteindre la pompe.

Si la température réelle >= température attendue + trigerHeat : on envoie la commande IR pour éteindre la pompe.

Si l’état de la pompe est 1 et (tempPompe > tempAttendue+hysterésis) : on réduit la température de la pompe.

Si l’état de la pompe est 1 et (tempPompe < tempAttendue-hysterésis) : on augmente la température de la pompe.

**void off()**

Eteint la pompe et envoie le mode à Domoticz.

**void on()**

Envoie le mode à Domoticz.

**void Autochecking(void)**

Si toutes les conditions sont réunies : Récupération des valeurs du tableau puis définition du mode en fonction des valeurs.

|  |
| --- |
| Thermostat |
| - getThermosObject : Thermostat\*  - modeCal : String  - lconfig : Application \*  - lsensor : SensorManager \*  - lIR : IRManager \*  - ldomo : DomoticzBroadcaster \*  - file : String  - fileCal : String  - AlarmTab : AlarmID\_t \*  - Cal : String \*  - checkConfort : uint8\_t  - Calidx : uint8\_t  + Eco : float  + Frostfree : float  + Comfort : float  + hysteresis : float  + triggerHeat : float  + prestime : uint8\_t  + presentbool : bool  + presmin : uint8\_t  + presmax : uint8\_t |
| - load\_Cal()  - Autochecking()  - globalOff()  - globalComfort()  - globalEco()  - globalFrostfree()  - setAlarm(uint8\_t i, char mode, uint8\_t nbday,uint8\_t h,uint8\_t s)  +Thermostat(String s, String fcal, Application \*c, SensorManager \*se, IRManager \*ir,DomoticzBroadcaster \*domo) : Thermostat  (Thermostat(Application \*c, SensorManager \*se, IRManager \*ir,DomoticzBroadcaster \*domo : Thermostat)  + run()  + off()  + on()  + save()  + save\_Cal(int l)  + save\_Cal(String n)  + init();  + getnbCal() {return Calidx;} : uint8\_t  + getCal() {return Cal;} : String \*  + IsStarted() : uint8\_t |