

# Documentation SDK Dll .Net pour Zibase

Proposé par <http://www.planete-domotique.com>

Retrouvez nos produits, et les différents accessoires Zibase directement dans notre boutique

Téléchargez la dernière version de la dll ZibaseDll.dll et la dernière documentation à jour à l'adresse suivante : <http://www.planete-domotique.com/zibase>

La Zibase est un produit développé par la société Zodianet : <http://www.zodianet.com>



Version 1.0

## 1. Principe de développement

Le SDK Zibase vous permettra de communiquer avec votre Zibase grâce à un composant .Net facile à intégrer dans vos projets.

Pour développer vos projets, vous pouvez utiliser les produits Visual Studio Express disponibles à l'adresse suivante :

<http://msdn.microsoft.com/fr-fr/express/aa975050.aspx>

Le composant permet de gérer une ou plusieurs Zibase sur votre réseau local. Les Zibase sont détectées automatiquement dès le lancement de l'initialisation.

Vous pourrez gérer différents type d'événements déclenchés par la Zibase :

- 1) Détection d'une nouvelle Zibase
- 2) Détection d'un nouveau capteur
- 3) Mise à jour de la valeur d'un capteur

Vous pourrez aussi gérer les fonctions de contrôle de la Zibase :

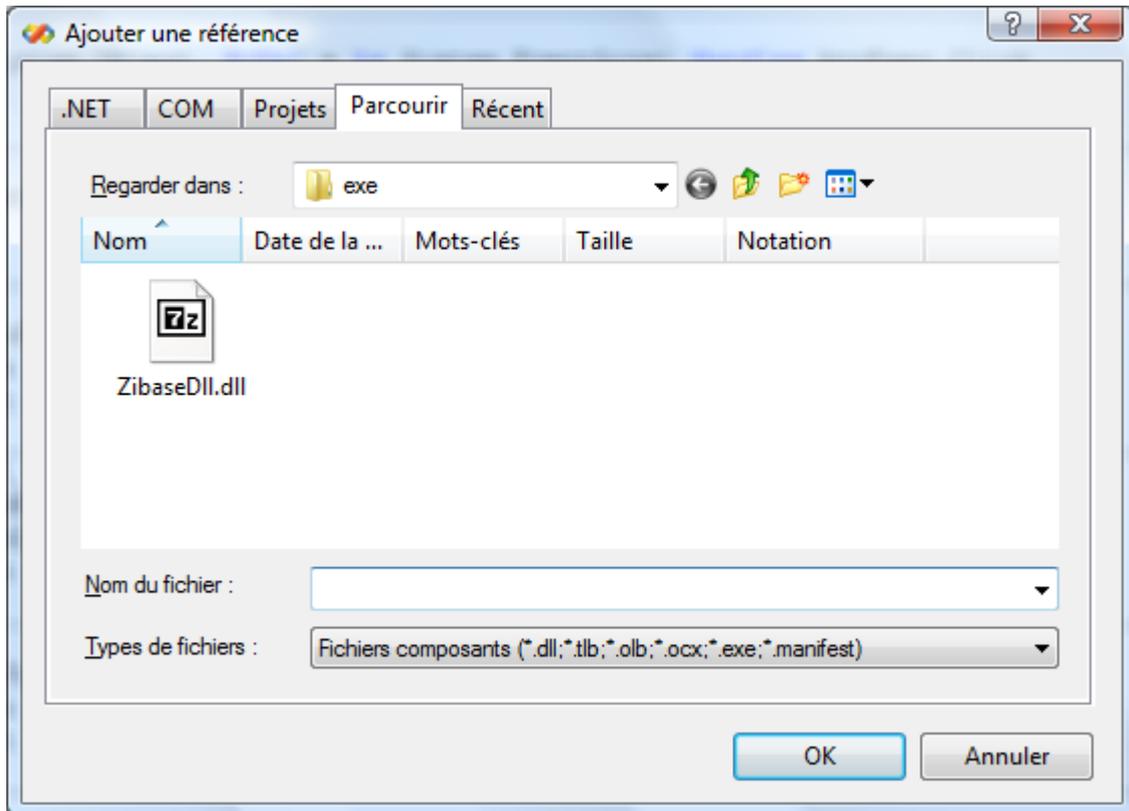
- 1) Envoie d'un ordre sur un périphérique domotique (exemple A1 ON) dans les différents protocoles reconnus par la Zibase (X10, Visonic, Chacon, Domia)
- 2) Exécution d'un scénario
- 3) Exécution d'un script Zibase

Le plugin dispose d'une sauvegarde locale de l'état des capteurs permettant à tout moment d'interroger la valeur de celui-ci.

## 2. Utilisation du module

Pour pouvoir utiliser le composant, il faut d'abord insérer la référence dans votre projet .Net.

Choisissez Projet/Ajouter une référence...



Sélectionnez la Dll ZibaseDll.dll et validez pour pouvoir désormais accéder aux fonctions et événements du module.

Il suffit ensuite d'importer le namespace correspondant :

```
Imports ZibaseDll
```

Les différentes fonctions seront accessible depuis un objet de type Zibase avec gestion d'événement :

```
Dim WithEvents zba As ZiBase = New ZiBase
```

Quatre événements sont disponibles :

- NewSensorDetected
- NewZibaseDetected
- UpdateSensorInfo
- WriteMessage

Les fonctions suivantes sont disponibles :

- StartZB : Permet de lancer l'initialisation des fonctions Zibase
- StopZB : Permet d'arrêter la gestion des fonctions Zibase
- RestartZibaseSearch : Permet de relancer une recherche automatique
- GetSensorInfo : Permet de récupérer la valeur courante d'un capteur
- SendCommand : Permet d'envoyer une commande à la Zibase
- ExecScript : Permet d'exécuter une commande de script Zibase
- RunScenario : Permet de lancer un scénario disponible sur la Zibase

### 3. Les évènements

#### a. NewSensorDetected

Cet évènement est déclenché dès que la Zibase détecte un nouveau capteur. Chaque capteur est identifié par un ID unique et par un type de valeur (hum, tem).

Pour un même ID, on pourra donc avoir plusieurs type (exemple : Température et Humidité pour une sonde THGR228)

Le type de valeur est une chaîne de 3 ou 4 caractères définissant le type de valeur mesurée :

Type de valeur (Tag)	Correspondance
Bat	Niveau de batterie (Ok / Low)
lev	Niveau de réception RF (1 à 5)
tem	Température (°C)
hum	Humidité (%)
Lnk	Etat de la connexion Zibase
sta	Status pour un switch (ON/OFF)
temc	Température de consigne (Thermostat : °C)
kwh	Mesure d'énergie totale (CM119)
Kw	Mesure d'énergie instantanée (CM119)
tra	Niveau de pluie total (Total Rain)
cra	Niveau de pluie courant (Currant Rain)
awi	Mesure de la vitesse du vent
drt	Direction du vent
uvl	Niveau d'UV

Le prototype de l'évènement est le suivant :

```
Private Sub zba_NewSensorDetected(ByVal seInfo As ZibaseDll.ZiBase.SensorInfo) Handles zba.NewSensorDetected
```

Les données sur le capteur sont transmises grâce à une structure de type Zibase.SensorInfo

Cette structure contient les informations sur le capteur ainsi que la dernière valeur mesurée :

```
Public Structure SensorInfo
    Dim sName As String
    Dim sType As String
    Dim sID As String
    Dim dwValue As Long
    Dim sValue As String
End Structure
```

Pour les valeurs de type réelles, la variable dwValue contient la valeur multipliée par 100. La variable sValue contient la valeur formatée avec l'unité de mesure (exemple : 23.5°C)

#### b. UpdateSensorInfo

Cet événement est géré comme l'événement NewSensorDetected, mais il est déclenché à chaque réception d'une nouvelle valeur par la Zibase.

```
Private Sub zba_UpdateSensorInfo(ByVal seInfo As
ZibaseDll.ZiBase.SensorInfo) Handles zba.UpdateSensorInfo
```

#### c. NewZibaseDetected

Cet événement est déclenché sur détection d'une nouvelle Zibase sur le réseau. La détection des Zibase s'effectue à l'initialisation (commande *StartZB*) ou après appel de la fonction *RestartZibaseSearch*

Les informations sur la Zibase sont transmises sous forme d'une structure :

```
Public Structure ZibaseInfo
    Dim sLabelBase As String
    Dim lIpAddress As Long
End Structure
```

Cette structure contient le nom logique de la Zibase (sLabelBase) et l'adresse IP de celle-ci.

Les différentes fonctions permettant de déclencher des actions sur la Zibase (SendCommand, RunScenario, ExecScript...) peuvent ensuite être exécutées sur une Zibase particulière en précisant son nom en premier paramètre ou sur toutes les Zibase détectée.

Dans le cas où une seule Zibase est présente sur le réseau, il n'est donc pas nécessaire de spécifier son nom dans les appels des fonctions.

#### d. WriteMessage

La fonction WriteMessage est un événement qui permet de récupérer les informations de Debug de la dll. Elle permet entre autre d'afficher les différentes données qui sont affichées dans le log sur la console de la Zibase (<http://www.zibase.net>)

```
Private Sub zba_WriteMessage(ByVal sMsg As String, ByVal level As Integer)  
Handles zba.WriteMessage
```

Le message transmit est contenu dans la variable sMsg avec une indication de niveau (INFO, DEBUG...)

### 4. Les méthodes

#### a. StartZB

La méthode StartZB permet d'initialiser le module et de démarrer la communication avec les Zibase. La première phase consiste à démarrer une recherche des Zibase.

```
Public Sub StartZB()
```

Exemple d'utilisation :

```
Dim WithEvents zba As ZiBase = New ZiBase  
zba.StartZB()
```

#### b. StopZB

La méthode StopZB permet d'arrêter la communication avec les Zibase et de libérer les ressources utilisées.

L'appel à cette méthode est indispensable avant de fermer votre application

```
Public Sub StopZB()
```

Exemple d'utilisation :

```
Dim WithEvents zba As ZiBase = New ZiBase  
zba.StopZB()
```

#### c. RestartZibaseSearch

La méthode RestartZibaseSearch permet de relancer à tout moment la recherche des Zibase du réseau. Par défaut, la recherche des bases s'effectue au démarrage de l'application. Cette méthode permet de relancer le processus à tout moment.

Exemple d'utilisation :

```
Dim WithEvents zba As ZiBase = New ZiBase  
zba.RestartZibaseSearch()
```

#### d. GetSensorInfo

La fonction `GetSensorInfo` permet de récupérer à tout moment la valeur courante d'un capteur.

En paramètre, l'ID du capteur et le type de mesure. La structure renvoyée est la dernière transmise par l'événement `UpdateSensorInfo`

```
Public Function GetSensorInfo(ByVal sID As String, ByVal sType As String) As SensorInfo
```

Exemple d'utilisation :

```
Dim WithEvents zba As Zibase = New Zibase
Dim sei As Zibase.SensorInfo

sei = zba.GetSensorInfo(TextBox4.Text, TextBox3.Text)
MsgBox(sei.sID & "(" & sei.sType & ") = " & sei.sValue)
```

#### e. SendCommand

La méthode `SendCommand` permet d'envoyer une commande à partir de la Zibase.

Les paramètres de la méthode sont :

- `sZibaseName` : Nom de la Zibase (facultatif)
- `sAddress` : Adresse du périphérique (format X10)
- `iState` : Etat à utiliser (ON/OFF/DIM)
- `iDim` : Valeur à utiliser pour une commande DIM (uniquement Chacon) : 0 à 100%
- `iProtocol` : Le protocole à utiliser (VISONIC, CHACON, X10, DOMIA)
- `iNbBurst` : Nombre d'émission de la commande à envoyer (par défaut 1)

```
Public Sub SendCommand(ByVal sZibaseName As String, ByVal sAddress As String, ByVal iState As State, Optional ByVal iDim As Integer = 0, Optional ByVal iProtocol As Protocol = Protocol.PROTOCOL_CHACON, Optional ByVal iNbBurst As Integer = 1)
```

Pour l'état, il est possible de choisir une valeur parmi les états suivants :

```
Public Enum State
    STATE_OFF = 0
    STATE_ON = 1
    STATE_DIM = 3
End Enum
```

Pour le protocole, les valeurs suivantes sont possibles :

```
Public Enum Protocol
    PROTOCOL_BROADCAST = 0
    PROTOCOL_VISONIC433 = 1
    PROTOCOL_CHACON = 3
    PROTOCOL_DOMIA = 4
    PROTOCOL_X10 = 5
End Enum
```

La valeur iDim est utilisée uniquement dans le cas du protocole Chacon et si l'état est positionné à DIM : iState = STATE\_DIM

Exemple d'utilisation :

```
Dim WithEvents zba As ZiBase = New ZiBase
zba.SendCommand("A1", ZiBase.State.STATE_OFF, 0,
ZiBase.Protocol.PROTOCOL_CHACON, 1)
```

#### f. ExecScript

La méthode ExecScript permet d'exécuter une instruction sur la Zibase.

Le langage de script de la Zibase n'étant pas public, cette fonction a un intérêt limitée.

```
Public Sub ExecScript(ByVal sZibaseName As String, ByVal sScript As String)
```

Le script est passé directement en paramètre.

Exemple d'utilisation :

```
Dim WithEvents zba As ZiBase = New ZiBase
Zba.ExecScript("Zibase00026d", "ON A1")
```

#### g. RunScenario

La méthode RunScenario permet de lancer un scénario construit sur la Zibase directement à partir de son nom.

Afin d'éviter les problèmes de compatibilité, nous vous conseillons de ne pas utiliser de caractères accentués dans vos noms de scénarios.

```
Public Sub RunScenario(ByVal sZibaseName As String, ByVal sCmd As String)
```

Exemple d'utilisation :

```
Dim WithEvents zba As ZiBase = New ZiBase
Zba.RunScenario("Zibase00026d", "Allumer Lampes")
```