

Composant Serveur API REST pour WinDev®

Léger.

Rapide.

Gratuit & Open Source.

Sans serveur IIS/Apache/...

Sans licence Serveur WebDev®.

Pas de SAAS, pas d'abonnement.

Windows 32/64 & Linux

Présentation

NEW : LightREST V3

La Documentation



Téléchargements

Un coup de main ?

Classe IrSession



Role

Gestion de sessions persistantes : création, lecture, écriture, vérification, rafraîchissement, nettoyage, et codes d'erreur.

A la création de la session, un timeout est défini. Si ce timeout est atteint sans aucun accès en lecture ou écriture, la session est détruite automatiquement.

A chaque lecture, écriture ou vérification de la validité d'une session, le compteur de timeout est réinitialisé. Il est possible de

forcer la réinitialisation de la session en utilisant la méthode `IrSession:Refresh`.

Les données de session peuvent être stockées, selon le paramètre passé à la méthode `IrSession:Create` :

- sous forme de paires Clé / Valeur
- sous forme d'un variant (plus souple et puissant)

Signature de l'ID de session : L'identifiant de session est signé numériquement, ce qui permet de détecter la réception d'un ID illégal et de bloquer toute exécution de requête. Cette signature est calculée en utilisant un buffer stocké dans le membre `:Salt` (partagé entre toutes les instances de la classe `IrSession`).

⚠ Si le membre `:Salt` est modifié en cours d'exécution (par n'importe quelle instance de `IrSession`), toutes les sessions en cours deviendront illégales. De même, si le serveur est redémarré avec un `:Salt` différent, les sessions en cours seront inaccessibles.

La valeur par défaut du membre `:Salt` est définie dans la constante `::DEFAULT_SALT`. Il est conseillé de la personnaliser (soit lors de l'exécution soit en modifiant sa valeur dans le code du composant).

Chiffrement des données de session Il est possible (et recommandé), lorsque la sécurité le nécessite, de chiffrer les données de la session (stockées dans le fichier dont le nom = `:ID` de la session). Pour activer le chiffrement, il faut que le membre

:CIPHERINGKEY soit défini.

⚠ Si le membre :CIPHERINGKEY est modifié en cours d'exécution (par n'importe quelle instance de *IrSession*), toutes les données des sessions en cours deviendront illisibles. De même, si le serveur est redémarré avec une :CIPHERINGKEY différent, les données des sessions en cours seront indéchiffrables.



Membres et propriétés

Nom	Type	Usage
CipheringKey	Buffer	Clé de chiffrement utilisée pour protéger les données de session. Intervient lors de la signature et du chiffrement/déchiffrement des informations stockées.
ID	chaîne	Identifiant de session.
IsCheckingSession	boolean	Active ou désactive la vérification de la signature de session. Permet de détecter les sessions falsifiées ou

tur e		altérées.
Last tEr ror Cod e	en tie r	Dernier code d'erreur rencontré lors d'une opération. Correspond à une constante ErrorSession*.
Sal t	Bu ffe r	Sel utilisé lors des opérations de signature et de chiffrement. Renforce la sécurité des données de session.
Ses sio nsD ire cto ry	ch aî ne	Répertoire contenant les fichiers de session. Chaque session est généralement persistée dans ce dossier.

▼ C Constantes		
Nom	Type	Usage
Err	en	0x1000 – Session

orSessionExpired	en	0x1000 – Session expirée.
orSessionIllegal	en	0x1001 – Session illégale ou identifiant invalide.
orSessionCannotLoad	en	0x1002 – Impossible de charger la session.
orSessionCannotSave	en	0x1003 – Impossible de sauver la session.

Err orS ess ion Can not Set Val ue	en tie r	0x1004 – Impossible d'écrire une valeur dans la session.
Err orS ess ion Can not Des tro y	en tie r	0x1005 – Impossible de détruire la session.
Err orS ess ion Can not Dec iph er	en tie r	0x1006 – Impossible de déchiffrer les données de session.
Err orS ess	en tie r	0x1007 – Impossible de lire une valeur de session.

ion
Can
not
Get
Val
ue

Méthodes

+

Create(Durée)

Présentation

Crée une session avec un timeout de type Durée.

Prototypage

Create(pTimeout Durée = 3600s)

Paramètres

No m	T y p e	Usage
pT	D	Durée de vie de

```
im      u      session.  
eo      ré  
ut      e
```

Valeurs retournées

Ty pe	Usage
bo olé en	Vrai si la session est créée.

Exemple

Création d'une session avec un timeout exprimé en Durée.

```
FUNCTION GetSession(pRe  
objet lrRequest <utile>,  
poResponse objet lrResponse)  
  
oSess      est objet  
lrSession  
duTimeout  est Durée  
  
duTimeout..EnSecondes = 3600  
  
SI oSess:Create(duTimeout)  
ALORS  
  
poResponse:SetHeaderValue("S  
SSION_ID", oSess:ID)  
    poResponse:Status  
= lrResponse::StatusOK
```

[Copier](#)

```
    poResponse:Body
    = "Session [%oSess:ID%]
    created."
    poResponse:ContentType
    = lrResponse::ContentTXT
    SINON
    poResponse:Status
    =
    lrResponse::StatusInternalSe
    rverError
    poResponse:Body
    =
    oSess:GetErrorMessage(oSess:
    LastErrorCode)
    poResponse:ContentType
    = lrResponse::ContentTXT
    FIN
```

▼ + Create(secondes)

📖 Présentation

Crée une session et initialise ses métadonnées (création, accès, timeout).

i Fonction dépréciée, utiliser plutôt la syntaxe avec un paramètre de type Durée

🧩 Prototypage

```
Create(pTimeout entier = 3600)
```

🔧 Paramètres

No m	T y p e	Usage
---------	------------------	-------

pT im eo ut	e n ti e r	Durée de vie en secondes.
----------------------	------------------------	------------------------------

Valeurs retournées

Ty pe	Usage
----------	-------

bo olé en	Vrai si la session est créée.
-----------------	-------------------------------

Exemple

Création d'une session avec un timeout exprimé en secondes.

```
FUNCTION GetSession(pRe  
objet lrRequest <utile>,  
poResponse objet lrResponse)  
  
oSess est objet lrSession  
  
SI oSess:Create(3600) ALORS
```

[Copier](#)

```
poResponse:SetHeaderValue("S  
SESSION_ID", oSess:ID)  
    poResponse:Status  
= lrResponse::StatusOK  
    poResponse:Body  
= "Session [%oSess:ID%]  
created."  
    poResponse:ContentType  
= lrResponse::ContentTXT  
SINON  
    poResponse:Status  
=  
lrResponse::StatusInternalSe  
rverError  
    poResponse:Body  
=  
oSess:GetErrorMessage(oSess:  
LastErrorCode)  
    poResponse:ContentType  
= lrResponse::ContentTXT  
FIN
```



Check

Présentation

Vérifie la validité de la session (existence, expiration, signature si activée).

Prototypage

```
Check() : (booléen, entier)
```

Paramètres

(aucun)

Valeurs retournées

Ty pe	Usage
bo olé en	Vrai si session valide.
ent ier	0 si OK, sinon un code ErrorSession*.

Exemple

Vérification de la validité d'une session
avant traitement.

```
FUNCTION
GetCustomer(poRequest objet
lrRequest, poResponse objet
lrResponse)

oSess est objet lrSession

oSess:ID =
poRequest:GetHeaderValue("SE
SSION_ID")

SI oSess:ID = "" ALORS
    poResponse:Status
    =
```

Copier

```
lrResponse::StatusBadRequest
    poResponse:Body
= "Missing SESSION ID"
    poResponse:ContentType
= lrResponse::ContentTXT
    RETOUR
FIN

SI pas oSess:Check() ALORS
    poResponse:Status
=
    lrResponse::StatusUnauthorized
    poResponse:Body
=
    oSess:GetErrorMessage(oSess:
    LastErrorCode)
    poResponse:ContentType
= lrResponse::ContentTXT
    RETOUR
FIN

//On rafraîchit la session
pour éviter un timeout
d'inactivité
oSess:Refresh()

////////// Suite du
traitement //////////
```

Refresh

Présentation

Rafraîchit la session en mettant à jour l'horodateur de dernier accès de session (prolonge l'activité jusqu'à atteinte du timeout d'inactivité).

Prototypage

`Refresh()` : (booléen, entier)

Paramètres

(aucun)

Valeurs retournées

Ty pe	Usage
bo olé en	Vrai si succès.
ent ier	0 si OK, sinon un code ErrorSession*.

Exemple

Exemple : prolonger l'activité d'une session.

```
(bOK, eErr) =  
oSess:Refresh()
```

Copier

  **GetData(clé)**

 **Présentation**

Récupère une valeur de session par sa clé préalablement stockée avec la méthode `SetData(pTag, pValue)`.

 **Prototypage**

```
GetData (pTag chaîne) : (booléen
```

 **Paramètres**

No m	T y p e	Usage
pT ag	c h aî n e	Nom du tag (clé).

 **Valeurs retournées**

Ty pe	Usage
----------	-------

bo Vrai si succès.
olé
en

Var Valeur si succès sinon un
ian code d'erreur.
t

Exemple

Exemple : lecture d'une valeur stockée en session.

```
(bOK, vVal) =  
oSess :GetData ("user")
```

Copier

GetData()

Présentation

Récupère le variant des données de la session, préalablement stocké avec la méthode :SetData(pVariant).

Prototypage

```
GetData() : (booléen, Variant)
```

 Paramètres

(aucun)

 Valeurs retournées

Ty pe	Usage
bo olé en	Vrai si succès.
Var ian t	Valeur si succès (sinon, selon implémentation, un code d'erreur).

 Exemple

Exemple : lecture du variant contenant les données de session.

```
vSessionData est varian
```

 Copier

```
(bOK, vSessionData) =  
oSess:GetData ()
```



GetAllData

Présentation

Retourne toutes les données de session (clé/valeur) dans un tableau associatif, préalablement stockées avec la méthode `SetData(pTag, pValue)`.

Prototypage

```
GetAllData() : (booléen, tableau)
```

Paramètres

(aucun)

Valeurs retournées

Ty pe	Usage
bo olé en	Vrai si succès.
tab lea u ass oci atif de	Toutes les données applicatives.

Var
ian
t

Exemple

Exemple : récupération de toutes les données d'une session.

```
vSessionData est varian  
  
(bOK, vSessionData) =  
oSess: GetAllData ()
```

Copier

SetData(clé, valeur)

Présentation

Mémoire une paire clé/valeur dans les données de session

Prototypage

```
SetData(pTag chaine, pValue var
```

Paramètres

No m	T y p e	Usage
ptag	Chaine	Nom de la clé

Valeurs retournées

Ty pe	Usage
bo olé en	Vrai si succès.
ent ier	0 si OK, sinon un code ErrorSession*.

Exemple

Ecriture d'une donnée clé/valeur dans le données de la session

```
(bOK, eErr) =  
oSess:SetData("nom", "toto")
```

[Copier](#)

  **SetData(variant)**

 **Présentation**

Mémoire des données de session sous forme de variant.

Permet de récupérer l'ensemble des données de session en une seule lecture, et donc un seul déchiffrement (au lieu d'effectuer plusieurs accès avec la syntaxe `SetData(pTag, pValue)`, donc meilleure performance).

 **Prototypage**

```
SetData(pValues Variant) : (bo
```

 **Paramètres**

No m	T y p e	Usage
pV al ue s	V a ri a	Objet Variant dont les propriétés sont les clés et valeurs à enregistrer.

nt

 Valeurs retournées

Ty pe	Usage
bo olé en	Vrai si succès.
ent ier	0 si OK, sinon un code ErrorSession*.

 Exemple

Ecriture de plusieurs valeurs dans la session.

```
v est Variant  
v.user = 123  
v.role = "admin"  
(bOK, eErr) =  
oSess:SetData(v)
```

 Copier

Destroy



Présentation

Détruit la session et libère les ressources associées, utilisé principalement à la fermeture de la session client.

Prototypage

```
Destroy() : (booléen, entier)
```

Paramètres

(aucun)

Valeurs retournées

Ty pe	Usage
bo olé en	Vrai si succès.
ent ier	0 si OK, sinon un code ErrorSession*.

Exemple

Suppression explicite d'une session.

```
(bOK, eErr) =  
oSess:Destroy()
```

Copier

 **Cleanup**

 **Présentation**

Nettoie les sessions expirées (dont le timeout d'inactivité a été dépassé).

 Cette méthode est appelée automatiquement chaque minute par la classe IrSession, il est donc inutile (mais possible) de l'appeler depuis le code du serveur LightREST.

 **Prototypage**

```
Cleanup ( )
```

 **Paramètres**

(aucun)

 **Valeurs retournées**

(aucune)

 **Exemple**

Nettoyage des sessions expirées.

```
oSession:Cleanup()
```

Copier

GetErrorMessage

Présentation

Retourne le message à un code
:ErrorSession*.

Prototypage

```
GetErrorMessage (pErrorMessage e
```

Paramètres

No m	T y p e	Usage
pE rr or Me ss	e n ti e r	Code d'erreur ErrorSession*.

ag

e

Valeurs retournées

**Ty
pe****Usage**ch
aîn
e

Message correspondant.

Exemple

Conversion d'un code d'erreur en message lisible.

```
(bOK, eErr) =  
oSession:Check()  
SI PAS bOK alors  
    poResponse:Status  
    =  
    lrResponse::StatusUnauthoriz  
    ed  
    poResponse:Body  
    =  
    oSess:GetErrorMessage(eErr)  
    poResponse:ContentType  
    = lrResponse::ContentTXT  
  
FIN
```

Copier

  **GetLastAccess**

 **Présentation**

Renvoie la date/heure du dernier accès à la session.

 **Prototypage**

```
GetLastAccess ( )
```

 **Paramètres**

(aucun)

 **Valeurs retournées**

Ty pe	Usage
Ch aîn e	Horodateur du denier accès au format AAAAMMJJHHMMSSCC

 **Exemple**

Récupération de l'horodateur du dernier accès à la session.

```
sLastAccess =  
oSess:GetLastAccess ()
```

[Copier](#)

GetTimestamp



Présentation

Récupère la date/heure de création de la session.



Prototypage

```
GetTimestamp ()
```



Paramètres

(aucun)



Valeurs retournées

Ty pe	Usage
Ch aîn e	Horodateur dz création de la session au format AAAAMMJJHHMMSSCC

 Exemple

Récupération de l'horodateur de création de la session.

```
sCreationTS =  
oSess:GetTimeStamp()
```

Copier **SetCIPHERingKey** **Compatibilite v2**

Signature conservée pour compatibilité LightREST v2.

 Deprecie

Conserve pour compatibilité. Préférer l'alternative recommandée.

 Présentation

Définit la clé de chiffrement.

 Prototypage

```
SetCIPHERingKey(pKey Buffer)
```



Paramètres

No m	T y p e	Usage
pK ey	B u ff er	Clé de chiffrement.



Valeurs retournées

(aucune)



Exemple

Définition de la clé de chiffrement

```
oSess:SetCipheringKey('cle-de-chiffrement')
```

Copier