Documentation : Briques applicatives

Principe

Les briques applicatives visent à faciliter et optimiser le développement grâce à des modules prédéveloppés réutilisables quelque soit le projet. Chaque module est composé d'un service qui va répondre aux demandes de l'utilisateur ainsi que d'un composant adapté à l'environnement qui va permettre la communication entre une application cliente et le service.

Structure

Tous les composants et services de socket fonctionnent de la même manière. On utilise une classe (CEchange) composée d'une classe (CInfoClient) et d'un buffer. La classe comporte les informations de l'utilisateur en cours, les informations de connexion à sa base de données, un tableau comportant les informations de connexion aux différents sockets auxquels il peut être amené à se connecter ainsi que le type de demande qu'il souhaite faire au service (Création d'un élément, recherche d'un élément, etc…) et un message informant des actions réalisées et des erreurs. Cette partie est quasiment invariable sauf pour le type de demande qui va varier à chaque action de l'utilisateur. Le buffer, lui, permet l'échange de données qui est spécifique à la demande du client. Il permet de récupérer les données renvoyées par le service dans le cas d'une recherche par exemple et il permet d'envoyer des données comme dans le cas d'un ajout. De plus, les données sont sérialisées en JSON et encodées en Base64.

Fonctionnement du composant

Le composant permet la communication entre l'application cliente et le service. Quelque soit la demande, nous avons en entrée et en sortie un CEchange. Ce qui nous permet de n'avoir qu'une seule procédure pour communiquer le socket dans notre composant. Dans cette procédure, on commence par se connecter au socket souhaité dans le cas où la connexion ne serait pas déjà ouverte. Ensuite, si la connexion est ouverte, on sérialise les données de notre échange en JSON et on l'encode dans un buffer en base64. Puis, on écrit au socket et on lit ce qu'il nous renvoie, on le décode et on le désérialise. Enfin, on renvoie l'échange dans l'application cliente. Dans le composant, on appelle également des procédures de filtrage qui permettent de filtrer les données que l'on a déjà récupéré au préalable sans avoir à refaire une requête.

Fonctionnement du service de socket

Le service fonctionne comme une application qui tourne à l'arrière-plan, en écoute sur un port, attendant les requêtes clientes. Lors de la réception d'une demande cliente, e service de socket va lire, décoder et désérialiser les données puis, grâce aux données récupérées il va se connecter à la base de données du client en cours afin de répondre à sa demande. Les connexions à la base de données et au socket restent ouvertes jusqu'à la déconnexion de l'utilisateur.

Les classes

Nous générons les classes à partir de nos fichiers de données et pour chaque fichier de données nous avons deux classes. Une classe modèle qui est la génération du fichier de données avec ses rubriques qui sont transformées en variable et la deuxième classe est composée d'un tableau de notre classe modèle. De plus, cette classe comporte des méthodes permettant de récupérer une liste d'éléments, un élément suivant son identifiant et une méthode de création/mise à jour.

MCOUtilisateurs = classe modèle CCOUtilisateurs = classe objet

Procédures TestWebSocket

AttenteSocket : procédure qui boucle à l'infini en attendant une requête.

LectureMessage : procédure qui va lire et répondre au composant qui l'a sollicité.

Entrée : Canal de communication (STCanaux)

MessageConnexion : procédure qui va traiter la requête

Entrée : STCanaux

ConnexionHFSQL : procédure de connexion à la base de données du client

Entrée : Informations du client (CInfoClient), STCanaux

Connexion : procédure d'authentification du client

Entrée : CInfoClient

Sortie : CInfoClient

ChiffreMdp : procédure pour chiffrer un mot de passe

Entrée : mot de passe en clair (chaine), identifiant de l'utilisateur (entier sur 8 octets)

Sortie : mot de passe chiffré (chaine)

GestionRole : procédure permettant de créer un rôle

Entrée : CInfoClient, MCORole

Sortie : message informatif (CInfoClient)

SupprRole : procédure permettant de supprimer un rôle

Entrée : CInfoClient, MCORole

Sortie : message informatif (CInfoClient)

CreeUtilisateur : procédure permettant de créer un utilisateur

Entrée : CInfoClient, MCOUtilisateurs

Sortie : message informatif (CInfoClient)

ModifUtilisateur : procédure permettant de modifier un utilisateur

Entrée : CInfoClient, MCOUtilisateurs

Sortie : message informatif (CInfoClient)

SupprUtilisateur : procédure permettant de supprimer un utilisateur

Entrée : CInfoClient, MCOUtilisateurs

Sortie : message informatif (CInfoClient)

Procédures SocketTier

RecupConstantesParType : procédure permettant de récupérer la liste des constantes selon le type

Entrée : Filtre (MCOListesConstantes)

Sortie : Liste des constantes filtrés (Tableau associatif de MCOListesConstantes)

ListePays : procédure permettant de lister les pays

Sortie : Liste des pays (Tableau associatif de MCOPays)

GestionTypeTiers : procédure permettant de créer un type de tiers

Entrée : CInfoClient, MCOTypeTiers

Sortie : message informatif (CInfoClient)

SupprTypeTiers : procédure permettant de supprimer un type de tiers

Entrée : CInfoClient, MCOTypeTiers

Sortie : message informatif (CInfoClient)

ListeCommune : procédure permettant de lister les communes

Sortie : Liste des communes (Tableau associatif de MCCommune)

ListeInfoCommunes : procédure permettant de lister les communes selon les filtres

Entrée : CInfoClient, MCCommune

Sortie : Liste des communes filtrés (Tableau associatif de MCCommune)

RecupCommuneParID : procédure permettant de récupérer une commune selon un identifiant

Entrée : MCCommune

Sortie : Commune recherché (Tableau associatif de MCCommune)

CreeCommune : procédure permettant de créer une commune

Entrée : CInfoClient, MCCommune

Sortie : message informatif (CInfoClient)

ModifCommune : procédure permettant de modifier une commune

Entrée : CInfoClient, MCCommune

Sortie : message informatif (CInfoClient)

SupprCommune : procédure permettant de supprimer une commune

Entrée : CInfoClient, MCCommune

Sortie : message informatif (CInfoClient)

GestionFamille : procédure permettant de créer une famille

Entrée : CInfoClient, MCOFamilles

Sortie : message informatif (CInfoClient)

SupprFamille : procédure permettant de supprimer une famille

Entrée : CInfoClient, MCOTypeTiers

Sortie : message informatif (CInfoClient)

RecupTiers : procédure permettant de récupérer les tiers

Sortie : Liste des tiers (Tableau associatif de MCOTiers)

ListeInfoTiers : procédure permettant de lister les tiers selon les filtres

Entrée : CInfoClient, Filtre (MCOTiers)

Sortie : Liste des tiers filtrés (Tableau associatif de MCOTiers)

ListeContactParTiers : procédure permettant de lister les contacts d'un tiers

Entrée : CInfoClient, numéro interne du tiers (MCOContact)

Sortie : Liste des contacts d'un tiers (Tableau associatif de MCOContact)

ListeNouveauContact : procédure permettant de lister les éléments qui ne sont pas contact du tiers choisi

Entrée : CInfoClient, numéro interne du tiers (MCOTiers)

Sortie : Liste des éléments qui ne sont pas en contact avec le tiers choisi (Tableau associatif de MCOTiers)

RecupTiersParID : procédure permettant de récupérer un tiers selon un identifiant

Entrée : MCOTiers

Sortie : Tiers recherchée (Tableau associatif de MCOTiers)

CreeTiers : procédure permettant de créer une famille

Entrée : CInfoClient, MCOTiers

Sortie : message informatif (CInfoClient)

ModifTiers : procédure permettant de modifier un tiers

Entrée : CInfoClient, MCOTiers

Sortie : message informatif (CInfoClient)

SupprTiers : procédure permettant de supprimer un tiers

Entrée : CInfoClient, MCOTiers

Sortie : message informatif (CInfoClient)

SupprContact : procédure permettant de supprimer un contact

Entrée : CInfoClient, MCOContact

Sortie : message informatif (CInfoClient)

AjouterContact : procédure permettant d'ajouter un contact

Entrée : CInfoClient, MCOContact

Sortie : message informatif (CInfoClient)

ListeNaf : procédure permettant de lister les APE naf

Sortie : Liste des APE naf (Tableau associatif de MCOApeNaf)

ListeCatJuridique : procédure permettant de lister les catégories juridiques

Sortie : Liste des catégories juridiques (Tableau associatif de MCOCatJuridiques)

SupprCompte : procédure permettant de supprimer un compte

Entrée : CInfoClient, MCOCompte

Sortie : message informatif (CInfoClient)

CreeCompte : procédure permettant de créer un compte

Entrée : CInfoClient, MCOCompte

Sortie : message informatif (CInfoClient)

ListeCompteTiers : procédure permettant de lister les comptes d'un tiers

Entrée : numéro interne du tiers (MCOCompte)

Sortie : Liste des comptes d'un tiers (Tableau associatif de MCOCompte)

RecupCompteParID : procédure permettant de récupérer un compte selon un identifiant

Entrée : MCOCompte

Sortie : Compte recherché (Tableau associatif de MCOCompte)

ModifCompte : procédure permettant de modifier un compte

Entrée : CInfoClient, MCOCompte

Sortie : message informatif (CInfoClient)

Procédures SocketDocument

ListeDocument : procédure permettant de lister des documents selon les filtres

Entrée : CInfoClient, Filtres (CFiltreDocument)

Sortie : Liste des documents filtrés (Tableau associatif de MCODocument)

ListeDocumentParNum : procédure permettant de récupérer un document selon un numéro interne

Entrée : CInfoClient, numéro interne du document (MCODocument)

Sortie : Tiers recherchée (Tableau associatif de MCODocument)

ListeTypeLiaison : procédure permettant de lister les types de liaison

Sortie : Liste des types de liaison (Tableau associatif de MCOTypeDeLiaison)

SupprDocument : procédure permettant de supprimer un document avec ses liaisons

Entrée : CInfoClient, MCODocument

Sortie : message informatif (CInfoClient)

ListeCodeLiaison : procédure permettant de lister les codes de liaison selon le type de liaison

Entrée : CInfoClient, type de liaison (MCOTypeDeLiaison)

Sortie : Liste des codes de liaison selon le type (Tableau associatif de MCOTypeDeLiaison)

SupprLiaisonDocument : procédure permettant de supprimer la liaison d'un document

Entrée : CInfoClient, MCOLiaisonDocument

Sortie : message informatif (CInfoClient)

ListeLiaisonDocumentParID : procédure permettant de récupérer un la liaison d'un document avec son type par l'identifiant

Entrée : CInfoClient, numéro interne du document (MCOLiaisonDocument)

Sortie : Liaison document recherchée (Tableau associatif de MCOLiaisonDocument)

CreeDocument : procédure permettant de créer un document avec sa liaison à un type de document

Entrée : CInfoClient, MCODocument

Sortie : message informatif (CInfoClient)

ModifDocument : procédure permettant de modifier un document

Entrée : CInfoClient, MCODocument

Sortie : message informatif (CInfoClient)

CreeTypeLiaison : procédure permettant de créer un type de liaison

Entrée : CInfoClient, MCOTypeDeLiaison

Sortie : message informatif (CInfoClient)

CreeLiaisonDocument : procédure permettant d'ajouter une liaison à un document

Entrée : CInfoClient, MCODocument

Sortie : message informatif (CInfoClient)

ListeDocumentParCodeLiaison : procédure permettant de lister les documents selon son code de liaison

Entrée : CInfoClient, code de liaison (MCODocument)

Sortie : Liste des documents filtrés (Tableau associatif de MCODocument)

Procédures communes aux composants

AppelComposantNomComposant **:** procédure permettant de faire le lien entre l'application cliente et le service de socket. Se connecte au socket, envoie la demande de l'utilisateur et récupère le résultat renvoyé par le service de socket.

Entrée : Informations du client + buffer (CEchange)

Sortie : Informations du client + buffer (CEchange)

Procédures Composant_TierSocket

GetCommuneByCP : procédure permettant de filtrer les communes selon le code postal

Entrée : liste des communes (MCCommune), code postal (chaine)

Sortie : Liste des communes filtrées (Tableau de MCCommune)

Pareil pour les autres éléments (famille, type tiers , etc..)