Les Services Diapason Web Génériques

Les Services Web Génériques

Principe

Diapason propose une plateforme de services web paramétrables REST en mode POST seulement.

Le service web est paramétrable par requête REB et déployable simplement. Disponible en mode synchrone et asynchrone.

Il permet :

- De réceptionner des informations de type simple structurées (au format JSON).
- De retourner des informations de type simple structurées et des listes de données (au format JSON).

Il est par exemple possible d'obtenir des listes de données de fabrication relatives à un ordre de fabrication, ou d'obtenir la liste des commandes en cours d'un client.

L'utilisation d'un canal basé sur des Services Web permet une communication inter-applicative interactive basée sur le modèle de service : demande => réponse.

Les différents types de communication

La communication est toujours basée sur le principe Client/Serveur : Le **client** est celui qui **déclenche une demande** sur un **serveur** qui est **en attente pour traiter toutes les demandes**.

> **Attention ! : Ne pas confondre le rôle de « Client et Serveur » avec flux principal des informations échangées.**

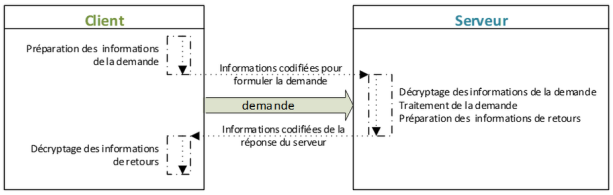
Le rôle du serveur n'est pas forcément de restituer une information, son service peut se limiter à réceptionner et prendre en compte les informations. Dans ce cas, c'est le client qui transmet l'information principale sans rien attendre en retour. (C'est par exemple le cas lorsque le « Client » ElciaOnline envoi le détail d'une configuration au « Serveur » Diapason).

- Ce n'est pas le sens de circulation de l'information principale qui importe. Le client est toujours le déclencheur (à l'origine de la demande) à destination d'un Serveur (en attente pour traiter les demandes).
- Il y a toujours de l'information qui circule dans un sens puis dans l'autre.

Dans le cas ou Diapason a le rôle du **Client**, on parlera de **« Service Web Externe »** et dans le cas ou Diapason a le rôle du **Serveur**, on parlera de **« Service Web Diapason »**.

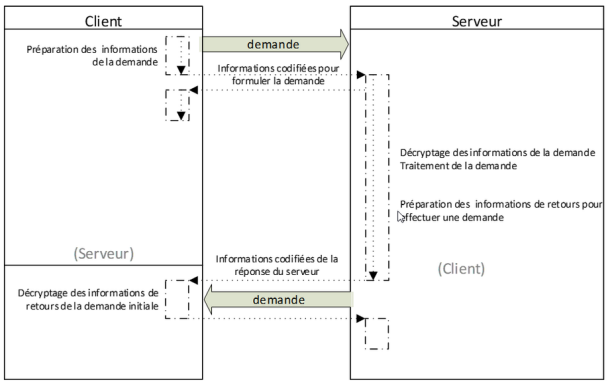
La communication synchrone

Dans une communication synchrone le client reste en attente pendant le traitement de sa demande sur le serveur.



La communication asynchrone

La communication asynchrone est principalement utilisée dans le cas des traitements serveurs longs. Dans ce cas, le client n'attend pas que sa demande soit traitée. Le serveur lui retourne juste une information pour indiquer au client qu'il va effectivement bien traiter la demande « plus tard ». Une fois la demande traitée, le serveur peut éventuellement retourner la réponse : il aura alors le rôle du « Client ».



Dans ce type de communication chaque partie à le rôle du client et du serveur. (Service Web Externe et service Web Diapason).

Périmètre des Services Web DIAPASON Génériques

Diapason propose une plateforme de services web paramétrables de type REST en Méthode POST seulement.

Deux modes d'exécution sont disponibles :

- un mode asynchrone qui poste le traitement du service dans les traitement BATCH de Diapason.

- un mode synchrone qui exécute le traitement immédiatement et qui retourne la réponse immédiatement.

Le traitement d'un service web générique est totalement assuré par une requête REB qui a pour objectif :

- La récupération des données en entrée du SW
- Le traitement du service
- La restitution des données de sortie.

Structuration des données

Les données échangées en entrée et en sortie sont transmises dans le body dans une structure au format JSON imposé par Diapason. Une autre partie du contenu dépendra des paramètres attendus ou généré par la requête.

La structure générale

Le body est au format JSON en entrée comme en sortie. Seul l'élément DIAP_DATA contiendra les paramètres spécifiques.

Voici la liste des éléments standard de premier niveau :

- "DIAP_METHODE": "Nom de la méthode de service web Diapason générique (défini dans Diapason)",
- "DIAP_IDENTIFIANT": "<<Identifiant unique de la requête de Service Web dans DIAPASON (GUID)>>",
- "DIAP_STATUTS": "<<Statuts de sortie de la requête SUCCES ou ERROR>>",
- "DIAP_DATA": { << Objet Json contenant les informations du service>> }

Exemple de structure en entrée :

```
{
   "DIAP_METHODE": "SWDiapason1",
   "DIAP_DATA": {
      "ParamCar1": "Valeur 1",
   }
}
```

Remarque :

La structure minimale en entrée est la suivante

```
{
   "DIAP_METHODE": "SWDG_DOC",
```

```
  "DIAP_DATA": {
  }
}
```


Exemple de structure en sortie :


```
{
  "DIAP_METHODE": "SWDiapason1",
  "DIAP_IDENTIFIANT": "86dfce14-8fc6-c3b1-7c14-e0c9d875c471",
  "DIAP_STATUTS": "SUCCES",
  "DIAP_DATA": {
    "ParamResult": "Valeur Resultat",
    }
}
```


Remarque :

La structure minimale en sortie est la suivante


```
{
  "DIAP_METHODE": "SWDiapason1",
  "DIAP_IDENTIFIANT": "86dfce14-8fc6-c3b1-7c14-e0c9d875c471",
  "DIAP_STATUTS": "SUCCES",
  "DIAP_DATA": {
    }
}
```


Les paramètres spécifiques

Toute manipulation de paramètres passe par la fonction DIALOG « SW-DIAPASON » avec :

- [MODE= PUT-*] pour transformer les variables et structures DIALOG en informations structurées dans l'élément « DIAP_DATA » du JSON dans le body du message en retour.
- [MODE= GET-*] pour transformer informations structurées dans l'élément « DIAP_DATA » du JSON dans le body du message en variables et structures DIALOG.

> ⚠ **Important :**
>
> **Les noms des paramètres (donnés après PARAM=) sont case-sensitive ce qui implique que les Majuscules et minuscules sont différenciées. Le nom du paramètre devra être donné de manière exacte.**

Voici la liste des possibilités offertes par la fonction DIALOG avec la structure générée ou attendue dans le JSON.

- **MODE= PUT-C**

Permet d'écrire un paramètre de type caractère dans le « DIAP_DATA » de retour.

Exemple de REB :

SW-DIAPASON(MODE= PUT-C , PARAM= CLO."Exemple_PUT-C" , VALEUR= CLO."Valeur du paramètre")

Résultat dans le JSON :

…

"DIAP_DATA": {

…

"Exemple_PUT-C": "Valeur du paramètre"

…

- **MODE= PUT-I**

Permet d'écrire un paramètre de type entier dans le « DIAP_DATA » de retour.

Exemple de REB :

SW-DIAPASON(MODE= PUT-I , PARAM= CLO."Exemple_PUT-I" , VALEUR= CLO.1234)

Résultat dans le JSON :

…

"DIAP_DATA": {

…

"Exemple_PUT-I": 1234,

…

- **MODE= PUT-R**

Permet d'écrire un paramètre de type réel dans le « DIAP_DATA » de retour.

Exemple de REB :

SW-DIAPASON(MODE= PUT-R , PARAM= CLO."Exemple_PUT-R" , VALEUR= CLO.3,141)

Résultat dans le JSON :

…

"DIAP_DATA": {

…

"Exemple_PUT-R": 3.141,

…

Remarque : Le séparateur décimal est le « . »

- **MODE= PUT-D**

Permet d'écrire un paramètre de type date dans le « DIAP_DATA » de retour.

Exemple de REB :

```
SW-DIAPASON( MODE= PUT-D , PARAM= CLO."Exemple_PUT-D" , VALEUR= CLO.21/12/2021 )
```

Résultat dans le JSON :

```
…
"DIAP_DATA": {
…
    "Exemple_PUT-D": "2021-12-21",
…
```

Remarque : La date est dans un format « inversé ».

- **MODE= PUT-L**

Permet d'écrire un paramètre de type logique dans le « DIAP_DATA » de retour.

Exemple de REB :

```
SW-DIAPASON( MODE= PUT-L , PARAM= CLO."Exemple_PUT-L" , VALEUR= CGL.OUI )
```

Résultat dans le JSON :

```
…
"DIAP_DATA": {
…
    "Exemple_PUT-L": true ,
…
```

Remarque :

Le logique est au format true/false.

- **MODE= PUT-LISTE**

Permet d'écrire un paramètre de type liste dans le « DIAP_DATA » de retour.

Exemple de REB :

```
CREATION Liste LST.ListeElements :
  PRENDRE ListeElements RefElement = CLO."Ref1"
  PRENDRE ListeElements DesElement = CLO."Description élément 1"
  PRENDRE ListeElements ValNum = CLO.1
FIN_BLOC
CREATION Liste LST.ListeElements :
  PRENDRE ListeElements RefElement = CLO."Ref2"
  PRENDRE ListeElements DesElement = CLO."Description élément 2"
```

```
  PRENDRE ListeElements ValNum = CLO.2
FIN_BLOC
CREATION Liste LST.ListeElements :
  PRENDRE ListeElements RefElement = CLO."Ref3"
  PRENDRE ListeElements DesElement = CLO."Description élément 3"
  PRENDRE ListeElements ValNum = CLO.3
FIN_BLOC
SW-DIAPASON( MODE= PUT-LISTE , PARAM= CLO."Exemple_PUT-LISTE" , VALEUR= ListeElements )
```

Résultat dans le JSON :

```
…
"DIAP_DATA": {
…
      "Exemple_PUT-LISTE": [
        {
          "RefElement": "Ref1",
          "ValNum": 1.0,
          "DesElement": "Description élément 1"
        },
        {
          "RefElement": "Ref2",
          "ValNum": 2.0,
          "DesElement": "Description élément 2"
        },
        {
          "RefElement": "Ref3",
          "ValNum": 3.0,
          "DesElement": "Description élément 3"
        }
      ],
…
```

Remarque : Choisir impérativement le format de liste Temp-Table **[TT] Liste Optimisé**.

- **MODE= PUT-TEXTE**

Permet d'écrire un paramètre de type « contenu de fichier texte » (liste WfFicContenu) dans le « DIAP_DATA » de retour.

Exemple de REB :

VLO.NomFichier = CLO."/tmp/ExempleFichierTexte.txt"

FICHIER(NOM-FICHIER= VLO.NomFichier , CDE-SYS-OK= CLO."" , CDE-SYS-PB= CLO."")

SW-DIAPASON(MODE= PUT-TEXTE , PARAM= VLO.NomFichier , VALEUR= WFFicContenu)

Résultat dans le JSON :

…

"DIAP_DATA": {

…

"/tmp/ExempleFichierTexte.txt": "Tm90ZSBkb250IGxhIGZy6XF1ZW5jZSBzZXJ0IGRlIHLpZulyZW5jZSBwb3VyIGwnYWNjb3JkIGRlcyB2b2l4IGV0IGRlcyBpbnN0cnVtZW50cy4KQXBwYXJlaWwgcHJvZHVjdGV1ciBkZSBjZXR0ZSBub3RlLgpPc2NpbGxhdGV1ciDgIGRldXggYnJhbmNoZXMgZW4gZm9ybWUgZGUgVSBkJ3VuIGFwcGFyZWlsIGhvcmFpcmUsIHZpYnJhbnQg4CB1bmUgZnLpcXVlbmNlIHBsdXMgc3RhYmxlIHF1ZSBsZXMgZGlzcG9zaXRpZnMg4CBiYWxhbmNpZXItc3BpcmFsLgpDb3JyZXNwb25kYW5jZSBjb252ZW50aW9ubmVsbGUg6XRhYmxpZSwgZGFucyB1biBkaWFncmFtbWUgb3UgdW4gY2FydG9ncmFtbWUsIGVudHJlIGxlIHR5cGUgZGUgaGFjaHVyZXMgb3UgbGEgY291bGV1ciBldCBsZXMgdGVudGF0aXN0aXF1ZXMgcmVwcmVzZW506WVzLgpWT1VTIENIRVJDSEVaIFBFVVQtylRSRQpkaWFwYXNvbiBuLm0uCk5vdGUgZG9udCBsYSBmculxdWVuY2Ugc2VydCBkZSBy6WbpcmVuY2UgcG91ciBsJ2FjY29yZCBkZXMuLi4KRVhQUkVTU0lPTlMgOgrKdHJlLCBzZSBtZXR0cmUgYXUgZGlhcGFzb24sIGRhbnMgdW5lIGRpc3Bvc2l0aW9uIGQnZXNwcml0IGNvbmZvcm1lIGF1eCBjaXJjb25zdGFuY2VzLCBlbiBoYXJtb25pZSwgZW4gYWNjb3JkIGF2ZWMgbCdhdHRpdHVkZSBvdSBsZXMgb3BpbmlvbnMgZCdhdXRydWku",

…

> Remarque :
>
> La valeur du paramètre est encodé en base 64 et le séparateur de ligne est le séparateur UNIX.
>
> Le nom du paramètre (PARAM=) doit correspondre à la valeur contenue dans WFFicContenu.NomFichier.

- **MODE= PUT-XML**

Permet d'écrire un paramètre de type « contenu de fichier XML » (liste WFEIAXmlMes) dans le « DIAP_DATA » de retour.

Exemple de REB :

VLO.NomFichier = CLO."/tmp/ExempleFichierXml.xml"

VLO.BalDep = CLO.""

XML-LECMESFIC(REF-MESSAGE-XML= CLO."XML2Test" , NOM-FICHIER= VLO.NomFichier , BALISE-DEPART= VLO.BalDep)

SW-DIAPASON(MODE= PUT-XML , PARAM= CLO."XML2Test" , VALEUR= WFEIAXmlMes)

Résultat dans le JSON :

…

"DIAP_DATA": {

…

"XML2Test": "PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iSVNPLTg4NTktMSIgPz4KPHZvaXR1cmVzPjx2b2l0dXJlPjxpbW1hdHJpY3VsYXRpb24+MTI0NVRSOTM8L2ltbWF0cmljdWxhdGlvbj48bW9kZWxlPkxhZ3VuYTwvbW9kZWxlPjxjb3VsZXVyPmJsYW5jaGU8L2NvdWxldXI+PG5iS20+MTI1MDAwPC9uYkttPjxkYXRlU29ydGllPjEyLzEyLzE5OTY8L2RhdGVTb3J0aWU+PC92b2l0dXJlPjx2b2l0dXJlPjxpbW1hdHJpY3VsYXRpb24+NDU3OEdlOTM8L2ltbWF0cmljdWxhdGlvbj48bW9kZWxlPkdvbGY8L21vZGVsZT48Y291bGV1cj5ibGV1ZTwvY291bGV1cj48bmJLbT4xMjQ1MDwvbmJLbT48ZGF0ZVNvcnRpZT4wMS8xMi8yMDA0PC9kYXRlU29ydGllPjwvdm9pdHVyZT48L3ZvaXR1cmVzPg=="

…

> Remarque :
>
> La valeur du paramètre est encodé en base 64 et le séparateur de ligne est le séparateur UNIX.
>
> Le nom du paramètre (PARAM=) doit correspondre à la valeur contenue dans WFEIAXmlMes.MesIde.

- **MODE= PUT-FICHIER**

Permet d'écrire un paramètre de type « contenu de fichier binaire » (images, pdf ou autres) dans le « DIAP_DATA » de retour.

Exemple de REB :

SW-DIAPASON(MODE= PUT-FICHIER , PARAM= CLO."ParamImage.jpg" , VALEUR= CLO."/tmp/Image2Test.jpg")

Résultat dans le JSON :

…

"DIAP_DATA": {

…

"ParamImage.jpg":
"/9j/4AAQSkZJRgABAQEAYABgAAD/2wBDAFA3PEY8MlBGQUZaVVBfeMiCeG5uePWvuZHI//2wBDAVVaWnhpeOuCguv//wAARCAFAAgADAREAAhEBAxEB/8QAHwAAAQUBAQEBAQEAAAAAAAAAAAECAwQFBgcICQoL/8QAtRAAAgEDAwIEAwUFBAQAAAF9AQIDAAQRBRIhMUEGE1FhB

...

7UAZ/nSf32/OgBUlkLrl26jvQBqUAf/Z"

…

Remarque :

La valeur du paramètre est le contenu binaire du fichier encodé en base 64.

- **MODE= GET-C**

Permet de lire un paramètre de type caractère dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :

…

"DIAP_DATA": {

…

"ParamCar1": "Valeur 1",

…

Exemple de lecture dans la REB du Service Web :

SW-DIAPASON(MODE= GET-C , PARAM= CLO."ParamCar1" , S:VALPARAM= VLO.VarCar)

Résultat :

La variable VarCar contient "Valeur 1"

- **MODE= GET-I**

Permet de lire un paramètre de type entier dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :

…

"DIAP_DATA": {

…

"ParamInt1": 9876,

…

Exemple de lecture dans la REB du Service Web :

SW-DIAPASON(MODE= GET-I , PARAM= CLO."ParamInt1" , S:VALPARAM= VLO.VarInt)

Résultat :

La variable VarInt contient 9876

- **MODE= GET-R**

Permet de lire un paramètre de type décimal dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :

…

"DIAP_DATA": {

…

"ParamNum1": 12345.6789,

…

Exemple de lecture dans la REB du Service Web :

SW-DIAPASON(MODE= GET-R , PARAM= CLO."ParamNum1" , S:VALPARAM= VLO.VarNum)

Résultat :

La variable VarNum contient 12345,6789

- **MODE= GET-D**

Permet de lire un paramètre de type décimal dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :

…

"DIAP_DATA": {

…

"ParamDate1": "2021-11-29",

…

Exemple de lecture dans la REB du Service Web :

SW-DIAPASON(MODE= GET-D , PARAM= CLO."ParamDate1" , S:VALPARAM= VLO.VarDate)

Résultat :

La variable VarDate contient la date 29/11/21

- **MODE= GET-L**

Permet de lire un paramètre de type décimal dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :

…

"DIAP_DATA": {

…

"ParamLog1": false

…

Exemple de lecture dans la REB du Service Web :

SW-DIAPASON(MODE= GET-L , PARAM= CLO."ParamLog1" , S:VALPARAM= VLO.VarLog)

Résultat :

La variable VarLog contient le logique NO

- **MODE= GET-LISTE**

Permet de lire un paramètre de type liste dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :

…

"DIAP_DATA": {

…

"Exemple_de_liste": [

{

"RefElement": "Elem10",

```
            "ValNum": 10,
            "DesElement": "élément 10"
        },
        {
            "RefElement": "Elem20",
            "ValNum": 20,
            "DesElement": "élément 20"
        },
        {
            "RefElement": "Elem30",
            "ValNum": 30,
            "DesElement": "élément 30"
        }
    ],
…
```


Exemple de lecture dans la REB du Service Web :

```
SW-DIAPASON( MODE= GET-LISTE , PARAM= CLO."Exemple_de_liste" , VALEUR= ListeElements )
```

Résultat :

La liste « ListeElements » est alimentée avec le contenu du JSON. Seuls les champs ayant exactement le même nom seront alimentés.

- **MODE= GET-TEXTE**

Permet de lire un paramètre de type « contenu texte » dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :


```
…
"DIAP_DATA": {
…

 "ParamTexte": "Tm90ZSBkb250IGxhIGZy6XF1ZW5jZSBzZXJ0IGRlIHLpZulyZW5jZSBwb3VyIGwnYWNjb3JkIGRlcyB2b2l4IGV0IGRlcyBpbnN0cnVtZW50cy4KQXBwYXJlaWwgcHJvZHVjdGV1ciBkZSBjZXR0ZSBub3RlLgpPc2NpbGxhdGV1ciDgIGRldXggYnJhbmNoZXMgZW4gZm9ybWUgZGUgVSBkJ3VuIGFwcGFyZWlsIGhvcmFpcmUsIHZpYnJhbnQg4CB1bmUgZnLpcXVlbmNlIHBsdXMgc3RhYmxlIHF1ZSBsZXMgZGlzcG9zaXRpZnMg4CBiYWxhbmNpZXItc3BpcmFsLgpDb3JyZXNwb25kYW5jZSBjb252ZW50aW9ubmVsbGUg6XRhYmxppZSwgZGFucyB1biBkaWFncmFtbWUgb3UgdW4gY2FydG9ncmFtbWUsIGVudHJlIGxlIHR5cGUgZGUgaGFjaHVyZXMgb3UgbGEgY291bGV1ciBldCBsZXMgdHJhbmNoZXMgdmFsZXVyIGRlcyBzdGF0aXN0aXF1ZXMgcmVwculzZW506WVzLgpWT1VTIENIRVJDSEVaIFBFVVQtylRSRQpkaWFwYXNvbiBuLm0uCk5vdGUgZG9udCBsYSBmculxdWVuY2Ugc2VydCBkZSBy6WbpcmVuY2UgcG91ciBsJ2FjY29yZCBkZXMuLi4KRVhQUkVTU0lPTlMgOgrKdHJlLCBzZSBtZXR0cmUgYXUgZGlhcGFzb24sIGRhbnMgdW5lIGRpc3Bvc2l0aW9uIGQnZXNwcml0IGNvbmZvcm1lIGF1eCBjaXJjb25zdGFuY2VzLCBlbiBoYXJtb25pZSwgZW4gYWNjb3JkIGF2ZWMgbCdhdHRpdHVkZSBvdSBsZXMgb3Bpbmlvbnn MgZCdhdXRydWku"

,
```

…

Exemple de lecture dans la REB du Service Web :

SW-DIAPASON(MODE= GET-TEXTE , PARAM= CLO."ParamTexte" , VALEUR= WFFicContenu)

ECRIT-FICHIER(IDENTIFIANT= CLO."ParamTexte" , NOM= CLO."FichierTexte.txt" , REPERTOIRE= CLO."/tmp/" , PARAMETRES= CLO."")

Résultat :

La liste « WFFicContenu» est alimentée avec le contenu du texte et la fonction DIALOG « ECRIT-FICHIER » permet d'écrire ce contenu dans un fichier.

Remarque :

La valeur du paramètre est encodée en base 64 et le séparateur de ligne est le séparateur UNIX.

- **MODE= GET-XML**

Permet de lire un paramètre de type « contenu XML » dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :

…

"DIAP_DATA": {

…

"ParamXML": "PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iSVNPLTg4NTktMSIgPz4KPHZvaXR1cmVzPjx2b2l0dXJlPjxpbW1hdHJpY3VsYXRpb24+MTI0NVRSOTM8L2ltbWF0cmljdWxhdGlvbj48bW9kZWxlPkxhZ3VuYTwvbW9kZWxlPjxjb3VsZXVyPmJsYW5jaGU8L2NvdWxldXI+PG5iS20+MTI1MDAwPC9uYkttPjxkYXRlU29ydGllPjEyLzEyLzE5OTY8L2RhdGVTb3J0aWU+PC92b2l0dXJlPjx2b2l0dXJlPjxpbW1hdHJpY3VsYXRpb24+NDU3OEdIOTM8L2ltbWF0cmljdWxhdGlvbj48bW9kZWxlPkdvbGY8L21vZGVsZT48Y291bGV1cj5ibGV1ZTwvY291bGV1cj48bmJLbT4xMjQ1MDwvbmJLbT48ZGF0ZVNvcnRpZT4wMS8xMi8yMDA0PC9kYXRlU29ydGllPjwvdm9pdHVyZT48L3ZvaXR1cmVzPg=="

,

…

Exemple de lecture dans la REB du Service Web :

SW-DIAPASON(MODE= GET-XML , PARAM= CLO."ParamXML" , VALEUR= WFEIAXmlMes)

VLO.BalDep = CLO."voitures"

XML-ECRMESFIC(REF-MESSAGE-XML= CLO."ParamXML" , NOM-FICHIER= CLO."/tmp/ExempleFichierXml1.xml" , BALISE-DEPART= VLO.BalDep)

Résultat :

La liste « WFEIAXmlMes» est alimentée avec le contenu du XML et la fonction DIALOG « XML-ECRMESFIC » permet d'écrire ce contenu dans un fichier.

Remarque :

La valeur du paramètre est encodée en base 64 et le séparateur de ligne est le séparateur UNIX.

- **MODE= GET-FICHIER**

Permet de lire un paramètre de type « contenu Fichier » dans le « DIAP_DATA » de la demande.

Soit un JSON avec le contenu suivant :

…

"DIAP_DATA": {

…

"ParamFichierJpg":
"/9j/4AAQSkZJRgABAQEAYABgAAD/2wBDAFA3PEY8MlBGQUZaVVBfeMiCeG5uePWvuZHI//2wBDAVVa
WnhpeOuCguv//wAARCAFAAgADAREAAhEBAxEB/8QAHwAAAQUBAQEBAQEAAAAAAAAA
AAECAwQFBgcICQoL/8QAtRAAAgEDAwIEAwUFBAQAAAF9AQIDAAQRBRIhMUEGE1FhB

...

7UAZ/nSf32/OgBUlkLrl26jvQBqUAf/Z" ,

…

Exemple de lecture dans la REB du Service Web :

SW-DIAPASON(MODE= GET-FICHIER , PARAM= CLO."ParamFichierJpg" , VALEUR= CLO."/tmp/ExempleImage.jpg")

Résultat :

Ecriture d'un fichier /tmp/ExempleImage.jpg avec le contenu du fichier d'origine.

Remarque :

La valeur du paramètre est encodée en base 64.

Gestion des erreurs

L'instruction LC-Erreur permet de mettre le Service Web en erreur. Il retourne alors un code 500 avec un DIAP_STATUTS à « ERROR » est le message du LC-Erreur.


```
{
   "DIAP_METHODE": "SWDG_DOC",
   "DIAP_IDENTIFIANT": "d00cf424-a341-7390-7d14-3791c8839f6a",
   "DIAP_STATUTS": "ERROR",
   "messages": "\n\nExemple d'erreur !"
}
```


Remarque :

Il conviendra donc de tester le retour des fonctions utilisées dans la REB pour éventuellement déclencher des erreurs comme dans l'exemple suivant.

VLO.RetourFonction = SW-DIAPASON(MODE= GET-C , PARAM= CLO."ParamCar1" , S:VALPARAM= VLO.VarCar)

SI VLO.RetourFonction EXISTE ET <> ""

  LC-Erreur = CLO."Erreur de lecture du paramètre :" + " " + VLO.RetourFonction

FIN_BLOC

Analyse du besoin : structure des échanges

La structure des données d'échange entre la REB et l'application cliente du service, devra être transmise à l'équipe qui développera cette application (par le concepteur de la REB). Le « DOCUMENT MODELE D'UNE API DE SERVICES WEB PARAMETRABLES » disponible plus loin en annexe permet de normaliser la description de l'API du service.

Chapitre infrastructure

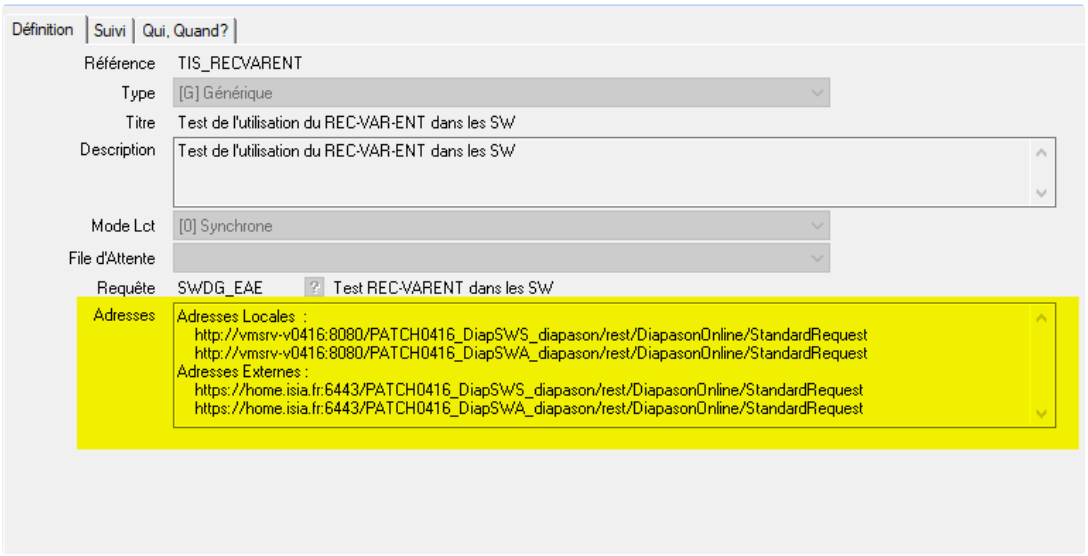
DIAPASON doit être configuré et doit pouvoir accéder aux Services Web (voir documentation d'installation des Services Web). Le document INS_Services_Web_Progress.pdf (Installation des services WEB dans Diapason), décrit l'infrastructure et le paramétrage de Diapason à mettre en place par le service exploitation.

Tester l'infrastructure des Services Web

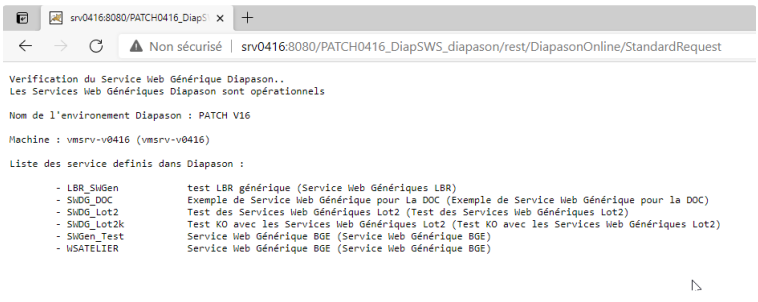
Un canal de test a été ajouté pour permettre de contrôler que les Services Web sont bien configurés pour un Diapason.

Ce test peut être effectué dans un navigateur Web à l'aide de l'une des l'URLs paramétrées dans Diapason.

La liste des adresses est affiché dans la fiche de définition des Services Web Génériques.



Le résultat du test retourne la liste des Services Web disponibles pour ce DIAPASON comme dans l'exemple ci-dessous :



Définition Service Web

Pour mettre en place un Service Web Générique il est nécessaire de définir le point d'entrée en renseignant une référence d'accès (une méthode de service web) associé à la **requête REB** de traitement dans l'application de Définition des Services Web Génériques.

Accès : Exploitation > Services Web Diapason > Services Web Génériques.

Requête REB de traitement

La requête REB associée à un service Web générique permet de gérer l'alimentation et la structuration des fichiers JSON, notamment grâce aux fonctions dédiées SW-EXTERNE et SW-DIAPASON.

Accès : Studio > DIALOG : Requêtes > Requêtes par type > REB.

Test des services Web

Les tests des services Web mis en place avec Diapason peuvent être réalisés facilement grâce à des outils dédiés type PostMan ou Swagger

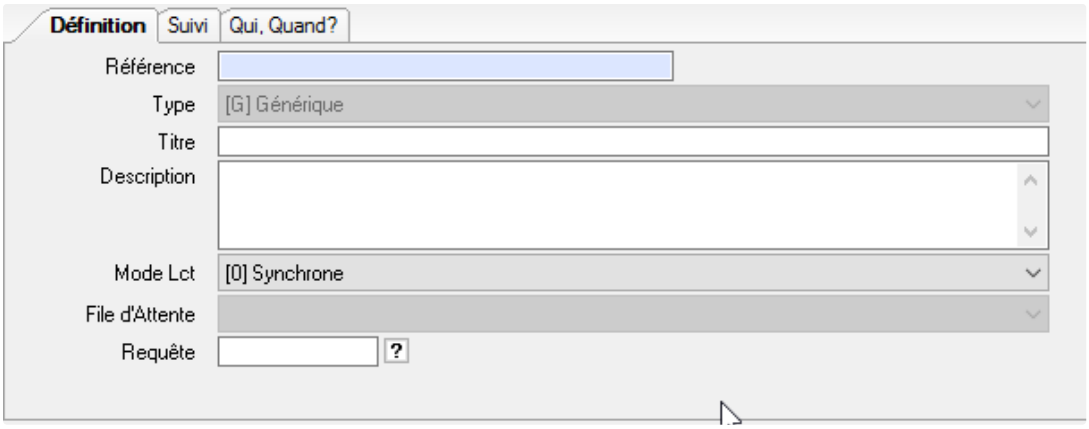
https://www.postman.com/

https://swagger.io/

Exemple détaillé de mise en place à consulter ici : Exemple de mise en place : interaction avec des écrans Atelier.

Définition des Services Web Génériques

Description de la fiche :



Onglet « Définition »

- **Référence**

Référence du service Web. Zone obligatoire. Saisie possible uniquement en création.

Cette référence doit commencer par une lettre (sont autorisés les lettres, les chiffres et le caractère « _ »)

- **Type**

Type de service Web. Contient « Générique ». Zone non saisissable.

- **Titre**

Titre du service Web. Zone facultative.

- **Description**

Description du service Web. Zone facultative.

- **Mode Lct**

Mode de lancement du service Web. Zone obligatoire. Doit contenir « Synchrone » ou « Asynchrone ».

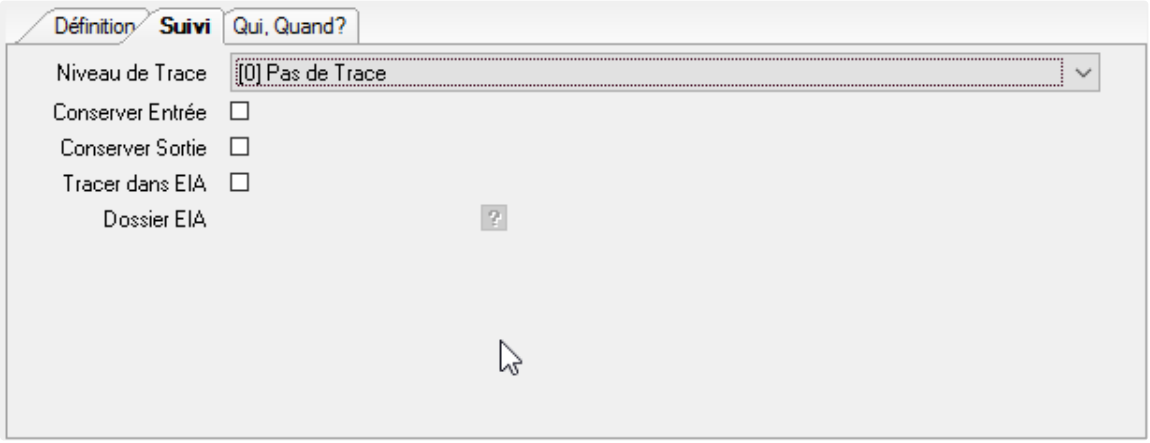
- **File d'Attente**

File d'attente à utiliser si le mode de lancement est « Asynchrone ». Zone obligatoire dans ce cas et non saisissable si le mode de lancement est « Synchrone ».

- **Requête**

Requête DIALOG de type « REB » à exécuter pour le service Web. Zone obligatoire.

> ⚠ **Attention : Cette requête doit exister dans TOUTES les sociétés de Diapason.**



Onglet « Suivi »

- **Niveau de Trace**

Niveau de trace affiché lors de l'utilisation du service Web

- **Conserver Entrée**

Indique si on souhaite ou non conserver les fichiers reçus lors de l'utilisation du service Web

- **Conserver Sortie**

Indique si on souhaite ou non conserver les fichiers générés par Diapason lors de l'utilisation du service Web

- **Tracer dans EIA**

Indique si on souhaite ou non générer un événement dans la boite aux lettres de l'EIA lors de l'utilisation du service Web

- **Dossier EIA**

Dossier de stockage de l'événement généré dans la boite aux lettres de l'EIA si on trace l'utilisation du service Web. Zone obligatoire dans ce cas et non saisissable si on ne trace pas dans l'EIA.

Fonctions de configuration d'un service WEB

Fonction SW-EXTERNE : Service WEB Externe

BUT

Cette fonction permet de configurer, exécuter et récupérer les paramètres d'un service WEB externe, à partir d'une requête DIALOG.

Disponible dans tous les types de requête.

SYNTAXE

Variable = SW-EXTERNE (MODE : action sur le service WEB
SW : Nom du service WEB à exécuter
IDENT : identifiant du service WEB
PARAM : Nom du paramètre en entrée ou sortie du service WEB
VALEUR : Valeur du paramètre en entrée du service WEB
E-S :LISTE : Liste DIALOG en entrée/sortie du service WEB
S :ID : identifiant retournée lors de l'initialisation du service WEB
S :VALPARAM : Valeur du paramètre en sortie)

PARAMETRES

Paramètre	E/S	O	Type	Description
MODE	E	Oui	Caractère	Action sur le service WEB externe. Cette référence peut prendre l'une des valeurs suivantes : • INIT : initialisation du service WEB externe • PUT-C : passage d'un paramètre de type caractère • PUT-I : passage d'un paramètre de type entier • PUT-D : passage d'un paramètre de type date • PUT-R : passage d'un paramètre de type décimal • PUT-L : passage d'un paramètre de type logique • PUT-LISTE : passage d'un paramètre de type Liste DIALOG • GET-C : récupération d'un paramètre de type caractère • GET-I : récupération d'un paramètre de type entier • GET-D : récupération d'un paramètre de type date • GET-R : récupération d'un paramètre de type décimal • GET-L : récupération d'un paramètre de type logique • GET-LISTE : récupération d'un paramètre de type Liste DIALOG • EXE : exécution service WEB externe
SW	E	Oui	Caractère	Référence service WEB externe à lancer. Cette référence peut être donnée par une variable locale, une constante globale ou une constante locale. Ce paramètre n'est disponible que dans le cas où le MODE est égal à EXE.
IDENT	E	Oui	Caractère	Identifiant du service WEB renvoyé par la fonction en MODE « INIT ». Peut être donné par une variable locale, une constante globale ou une constante locale.
PARAM	E	Oui	Caractère	Nom du paramètre à envoyer au service WEB ou à récupérer en sortie du service WEB. Peut être donné par une variable locale, une constante globale ou une constante locale. Ce paramètre n'est pas utilisé dans le cas où le MODE est égal à INIT ou EXE.
VALEUR	E	Oui	Caractère Numérique Logique	Valeur du paramètre à envoyer au service WEB. Paramètre utilisé seulement dans le cas où le MODE est égal à l'une des options de passage de paramètres (PUT-*). Ce paramètre peut être donné par une variable locale, une constante globale, une constante locale ou par une sélection de Liste DIALOG dans le cas PUT-LISTE.

			Date	
E-S :LISTE	E	Oui	Caractère	Paramètre utilisé dans le cas où le MODE est égal à GET-LISTE. Il contient la référence de la liste à récupérer en retour d'exécution du service WEB.
S :ID	E	Oui	Caractère	Identifiant service WEB retourné dans une variable de type caractère, dans le cas où le MODE est égal à INIT.
S :VALPARAM	E	Oui	Caractère Numérique Logique Date	Valeur du paramètre après exécution du service WEB. Paramètre utilisé seulement dans le cas où le MODE est égal à l'une des options de récupération de paramètres (GET-* sauf GET-LISTE).

NOTES

La fonction retourne dans la variable résultat :

- Le code erreur DIAPASON et le libellé associé à l'erreur, lors de toute anomalie d'exécution
- Vide si tout s'est bien passé

EXEMPLE

```
VLO.RES = SW-EXTERNE( MODE= INIT , S:ID= VLO.ID )

VLO.NumLig = CLO.214888

VLO.RES = SW-EXTERNE( MODE= PUT-I , IDENT= VLO.ID , PARAM= CLO."idLigne" , VALEUR= VLO.NumLig )

VLO.RES = SW-EXTERNE( MODE= EXE , SW= CLO."CFG_WEB_ELCIA_POST" , IDENT= VLO.ID )

VLO.RES = SW-EXTERNE( MODE= GET-LISTE , IDENT= VLO.ID , PARAM= CLO."Param-liste" , E-S:LISTE= WFEIAXmlMes )
```

Cette requête permet de lancer le service WEB qui attend en entrée un numéro de ligne et qui renvoie en résultat la liste DIAPASON WFEIAXmlMes.

Fonction SW-DIAPASON

BUT

Cette fonction permet de configurer, exécuter et récupérer les paramètres d'un service WEB Diapason, à partir d'une requête DIALOG.

Disponible dans tous les types de requête.

SYNTAXE

```
Variable =  SW-DIAPASON ( MODE          : action sur le service WEB
                          PARAM         : Nom du paramètre en entrée ou sortie du service WEB
                          VALEUR        : Valeur du paramètre en entrée du service WEB
                          S :VALPARAM   : Valeur du paramètre en sortie )
```

PARAMETRES

Paramètre	E/S	O	Type	Description
MODE	E	Oui	Caractère	Action sur le service WEB Diapason. Cette référence peut prendre l'une des valeurs suivantes : • PUT-C : passage d'un paramètre de type caractère • PUT-I : passage d'un paramètre de type entier • PUT-D : passage d'un paramètre de type date • PUT-R : passage d'un paramètre de type décimal • PUT-L : passage d'un paramètre de type logique • PUT-LISTE : passage d'un paramètre de type Liste DIALOG • PUT-TEXTE : passage d'un paramètre de type contenu de fichier texte (liste WfFicContenu) • PUT-XML : passage d'un paramètre de type contenu de fichier XML (liste WFEIAXmlMes) • PUT-FICHIER : passage d'un paramètre de type contenu de fichier binaire • GET-C : récupération d'un paramètre de type caractère • GET-I : récupération d'un paramètre de type entier • GET-D : récupération d'un paramètre de type date • GET-R : récupération d'un paramètre de type décimal • GET-L : récupération d'un paramètre de type logique • GET-LISTE : récupération d'un paramètre de type Liste DIALOG • GET-TEXTE : récupération d'un paramètre de type contenu de fichier texte (liste WfFicContenu) • GET-XML : récupération d'un paramètre de type contenu de fichier XML (liste WFEIAXmlMes) • GET-FICHIER : récupération d'un paramètre de type contenu binaire pour générer un fichier
PARAM	E	Oui	Caractère	Nom du paramètre à envoyer au service WEB ou à récupérer en sortie du service WEB. Peut être donné par une variable locale, une constante globale ou une constante locale.
VALEUR	E	Oui	Caractère Numérique Logique Date	Valeur du paramètre à envoyer au service WEB. Paramètre utilisé seulement dans le cas où le MODE est égal à l'une des options de passage de paramètres (PUT-*). Ce paramètre peut être donné par une variable locale, une constante globale, une constante locale ou par une sélection de Liste DIALOG dans le cas PUT-LISTE.
S :VALPARAM	E	Oui	Caractère Numérique Logique Date	Valeur du paramètre après exécution du service WEB. Paramètre utilisé seulement dans le cas où le MODE est égal à l'une des options de récupération de paramètres (GET-*).

NOTES

La fonction retourne dans la variable résultat :

- Le code erreur DIAPASON et le libellé associé à l'erreur, lors de toute anomalie d'exécution
- Vide si tout s'est bien passé

EXEMPLE

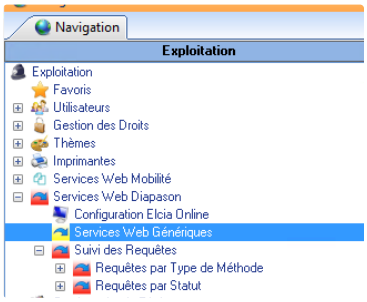
```
VLO.REs = SW-DIAPASON( MODE= GET-C , PARAM= CLO."ParamCar1" , S:VALPARAM= VLO.ValeurParam )

VLO.REs = SW-DIAPASON( MODE= GET-D , PARAM= CLO."ParamDate2" , S:VALPARAM= VLO.ValeurDate )

VLO.REs = SW-DIAPASON( MODE= GET-R , PARAM= CLO."ParamNum3" , S:VALPARAM= VLO.ValeurNum )

VLO.REs = SW-DIAPASON( MODE= PUT-C , PARAM= CLO."ListeParametresEntrees" , VALEUR= VLO.LstParamEntree )

VLO.REs = SW-DIAPASON( MODE= PUT-C , PARAM= CLO."Param_CHARACTER" , VALEUR= CLO."Test CAR" )

VLO.REs = SW-DIAPASON( MODE= PUT-D , PARAM= CLO."Param_DATE" , VALEUR= CLO.29/06/2020 )

VLO.REs = SW-DIAPASON( MODE= PUT-L , PARAM= CLO."Param_LOGICAL" , VALEUR= CGL.OUI )

VLO.REs = SW-DIAPASON( MODE= PUT-R , PARAM= CLO."Param_DECIMAL" , VALEUR= VLO.ValR )

CREATION Liste LST.Liste2Test :

  PRENDRE Liste2Test ParCar1 = CLO."Valeur1"

  PRENDRE Liste2Test ParCar2 = CLO."Valeur2"

  PRENDRE Liste2Test ParNum1 = CLO.3

FIN_BLOC

VLO.REs = SW-DIAPASON( MODE= PUT-LISTE , PARAM= CLO."NomDeLaListeEnRetour" , VALEUR= Liste2Test )
```

Exemple de mise en place : interaction avec des écrans Atelier

Ce document présente le paramétrage effectué pour la mise en œuvre des webservices un atelier ligne automatique.

Méthode service web

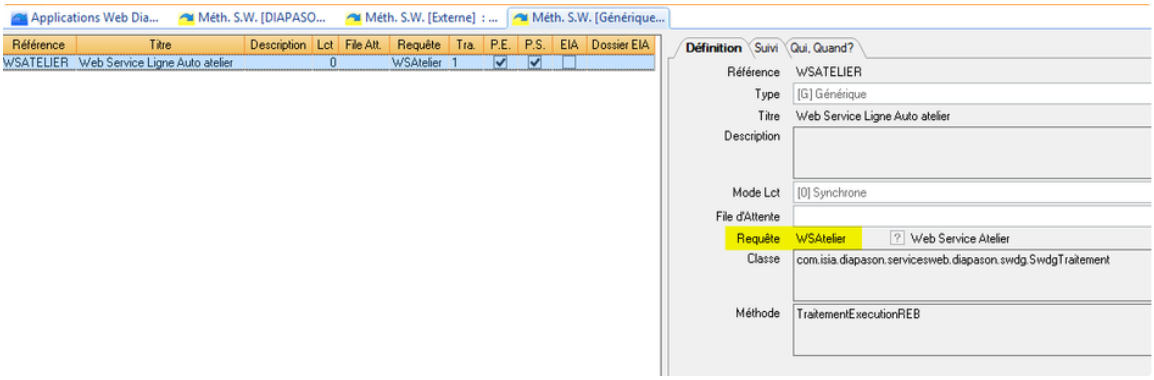
Dans le menu « Exploitation », déclaration de la méthode utilisée pour le web service :



C'est cette référence méthode qui devra être « déclarée » dans le json d'appel de diapason, dans la balise « DIAP_METHODE » :

```
1  {"DIAP_METHODE":"WSATELIER","DIAP_DATA":{"Contexte":"LigneAuto","IdEtape":"VTOU1","IdPiece":"1044800072","Version":"1.0"}}
```

D'autre part, sur la méthode, on indique la requête utilisée sur la méthode :



Requête

Il s'agit ici de la requête WSAtelier.

Dans cette requête, on déclare d'abord les éléments récupérés du json :

```
VLO.Res = SW-DIAPASON( MODE= GET-C , PARAM= CLO."Contexte" , S:VALPARAM= VLO.Contexte )
VLO.Res = SW-DIAPASON( MODE= GET-C , PARAM= CLO."IdEtape" , S:VALPARAM= VLO.Etape )
VLO.Res = SW-DIAPASON( MODE= GET-C , PARAM= CLO."IdPiece" , S:VALPARAM= VLO.NumPiece )
VLO.Res = SW-DIAPASON( MODE= GET-C , PARAM= CLO."Version" , S:VALPARAM= VLO.Version )
```

Correspondant au json :


```
{"DIAP_METHODE":"WSATELIER","DIAP_DATA":{"Contexte":"LigneAuto","IdEtape":"VTOU1","IdPiece":"1044800072","Version":"1.0"}}
```


La requête permet à partir de ces éléments de renvoyer les numéros de pièces « en liste » dans le json.

Dans la requête :

```
CREATION Liste LST.Pieces :
   PRENDRE Pieces IdPiece = DTD LAFatCT.LanSerCTAI1
   PRENDRE Pieces Reference = DTD LAFatCT.LanSerCTA
   PRENDRE Pieces Dimension = VLO.LongueurC
   PRENDRE Pieces Contenant_Type = CLO.""
   PRENDRE Pieces Contenant_Id = VLO.Contenant
   PRENDRE Pieces Contenant_Case = DTD LAFatCT.LanS
   PRENDRE Pieces Contenant_Emplac = VLO.Emplac
FIN  BLOC
```

Puis :

```
COMMENTAIRE : mise en forme pour le service web
VLO.Res = SW-DIAPASON( MODE= PUT-LISTE , PARAM= CLO."Pieces" , VALEUR= Pieces )
```

Résultat dans le json :

```
{
    "DIAP_METHODE": "WSATELIER",
    "DIAP_IDENTIFIANT": "afecefeb-e3dd-8c90-7014-99bb18b238d3",
    "DIAP_STATUTS": "SUCCES",
    "DIAP_DATA": {
        "Pieces": [
            {
                "IdPiece": "",
                "Reference": "4/20/4R",
                "Dimension": "1799.00x340.00",
                "Contenant_Type": "",
                "Contenant_Id": "",
                "Contenant_Case": "",
                "Contenant_Emplac": ""
            },
            {
                "IdPiece": "1044800034",
                "Reference": "PM28B",
                "Dimension": "348.00",
                "Contenant_Type": "",
                "Contenant_Id": "",
                "Contenant_Case": "2",
                "Contenant_Emplac": ""
            },
            {
                "IdPiece": "1044800035",
                "Reference": "PM28B",
                "Dimension": "1807.00",
                "Contenant_Type": "",
                "Contenant_Id": "",
                "Contenant_Case": "2",
                "Contenant_Emplac": ""
            },
            {
```


Outil de test

ISIA s'est servi de l'outil POSTMAN pour tester les paramétrages réalisés.

Méthodologie : DOCUMENT MODELE D'UNE API DE SERVICES WEB PARAMETRABLES

Préambule : Comment utiliser ce document

Ce document est un modèle pour permettre de fournir au client une description des échanges proposés par l'implémentation d'un Service Web Diapason Générique dans une REB.

La partie de la description « standard » est parfois optionnelle car elle peut dépendre des choix du paramétreur. (Exemple : c'est le paramétreur qui décide si une authentification est nécessaire).

Le texte de couleur rouge ne devra pas apparaitre dans le document final de description de l'API !

Il conviendra donc de garder et de compléter le texte en vert ou dans des cadres vert.

Introduction

Faire une petite présentation du SW : son rôle et son cadre d'utilisation.

> *Ce document présente et documente le Service Web qui permet …*

Historique des modifications

Le tableau ci-dessous permet de tracer les modifications de l'API.

> *Le tableau ci-dessous liste l'ensemble des modifications apportées au document.*

Date	***Version***	***Éléments modifiés***	***Auteur***
29/01/2020	*1.0*	*Création du document*	*…*

….	*1.1*	*Ajout de ….*	*…*

Authentification

L'authentification avec un identifiant et un mot de passe n'est pas imposé dans l'outil SWDG. Vous pouvez donc choisir de ne pas demander d'informations d'authentification ou de les rendre obligatoire. S'il n'y a pas d'information d'authentification, l'outil SWDG utilisera l'utilisateur par défaut « isia_swdg ». Le login et mot de passe présent dans ces informations seront ceux d'un utilisateur Diapason.

Dans tous les cas, vous recevrez les informations concernant le type d'authentification suivantes dans la REB sous la forme de paramètre :

SWDG_LOGIN : Paramètre (caractère) qui indique l'identifiant utilisé pour l'aut

SWDG_ACCES_AUTORISE : Paramétré (Logique) qui indique si le client du service c'est authentifié avec succès.

C'est donc à la requête REB de déclencher un LC-ERREUR en cas de SWDG_ACCES_AUTORISE = NO

Si vous ne souhaitez pas utiliser l'authentification, vous pouvez enlever ce chapitre du document.

Si vous demandez au client de s'authentifier, voici la description de son formalisme qu'il faudra mettre dans la description de l'API :

> *L'authentification s'effectue en méthode « Basic » définie par la spécification RFC 2617 : https://www.ietf.org/rfc/rfc2617.txt avec le login et le mot de passe d'un compte Diapason valide.*
>
> *En résumé, pour s'authentifier, la requête cliente doit spécifier l'en-tête HTTP « Authorization ». Celle-ci doit contenir la méthode utilisée (Basic) suivie de la représentation en Base64 du nom de l'utilisateur et du mot de passe séparés par le caractère « : » (deux-points).*
>
> *Par exemple, pour authentifier l'utilisateur « Aladdin » avec le mot de passe « open sesame », le client envoie :*
>
> Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
>
> QWxhZGRpbjpvcGVuIHNlc2FtZQ== *étant la représentation en Base64 du texte Aladdin:open sesame.*

Web Service

Formulation de la demande

- **URL**

Sur une architecture standard, l'URL permettant d'utiliser le Service Web est celle du serveur HTTP frontal que le client doit fournir et administrer. Pensez à spécifier au client qu'ISIA n'a pas la responsabilité des évolutions ou des problèmes qui pourraient survenir.

Donnez l'URL permettant de lancer le service (sur le serveur frontal).

Exemple :

> *Voici l'URL que <<<<< Nom du client >>>>> met à disposition pour l'utilisation du Service Web :*
>
> ***https://NomClient.com/api/nomservice***

Si vous avez prévu d'utiliser des paramètres à la suite de l'URL, donnez la description complète de l'URL et de ces paramètres comme dans l'exemple suivant :

> *Voici l'URL que <<<<< Nom du client >>>>> met à disposition pour l'utilisation du Service Web :*
>
> ***https://NomClient.com/api/nomservice?{Param1}=...t&{Param2}=...&{Param3}=...***
>
> ***Paramètres URL requis :***

> *Param1[entier] : Description du paramètre 1*
>
> *Param2[caractère] : Description du paramètre 2*
>
> ***Paramètres URL facultatifs :***
>
> *Param3[caractère] : Description du paramètre 3*

- **Méthode**

> ***POST***

- **Corps du message**

Le corps du message est au format Json. Il doit impérativement contenir les informations "DIAP_METHODE" et « DIAP_DATA » sur le premier niveau tel que :

DIAP_METHODE (caractère)

Doit contenir la référence de la méthode de service web Diapason générique qui a été créé lors du paramétrage.

DIAP_DATA (Objet Json)

Doit contenir l'ensemble des paramètres d'entrée qui ne sont pas passés dans l'URL.

Pensez à préciser si les paramètres sont facultatifs ou optionnels ainsi que le type des paramètres.

Voici un modèle :

> *Le corps du message est au format Json. Il présentera les données d'entrée du service avec le formalisme suivant :*
>
> *{*
>
> *"DIAP_METHODE": "<<<Nom de la méthode de service web Diapason générique (défini dans Diapason)>>>",*
>
> *"DIAP_DATA": {*
>
> *"<<<Paramètre1>>>": "<<<Valeur du Paramètre1>>>",*
>
> *"<<<Paramètre2>>>": "<<<Valeur du Paramètre2>>>",*
>
> *"<<<Paramètre3>>>": <<<Valeur numérique du Paramètre3>>>*
>
> *}*
>
> *}*

Paramètres requis :

Param1[caractère] : Description du paramètre 1

Param2[caractère] : Description du paramètre 2

Paramètres facultatifs :

Param3[décimal] : Description du paramètre 3

Si le Service Web n'a pas de paramètre ou si tous les paramètres sont donnés dans l'URL, le corps du message aura sa forme la plus simple comme ci-dessous :

> *Le corps du message est au format Json. Il présentera les données d'entrée du service avec le formalisme suivant :*

> *{*
>
> *"DIAP_METHODE": "<<<Nom de la méthode de service web Diapason générique (défini dans Diapason)>>>",*
>
> *" DIAP_DATA": {}*
>
> *}*

Paramètres requis :

Aucun

Paramètres facultatifs :

Aucun

- **Exemple de contenu**

Donnez un exemple de contenu du corps du message (avec différents cas si possible).

Le retour

Réponse succès

- **Codes de retour**

Codes : 200

- **Corps du message**

Le corps du message est au format Json. Le premier niveau de paramètre sera toujours le même pour une même version de Diapason. Les paramètres provenant de la REB avec la fonction DIALOG « SW-DIAPASON » seront sur le premier niveau de l'objet DIAP_DATA.

Voici la description de la partie standard du corps du message :

> *Le corps du message est au format Json. Il présentera les données de retour du service avec le formalisme suivant :*
>
> *Paramètres du premier niveau*
>
> *Au premier niveau, l'objet « DIAP_DATA » contiendra toutes les informations de retour propre au service. Les autres paramètres de premier niveau sont des informations techniques pour la gestion et le suivi. La liste de ces autres paramètres pourra être agrémenté dans les versions futures de Diapason.*
>
> *Voici la liste des éléments de premier niveau :*
>
> *"DIAP_METHODE": "Nom de la méthode de service web Diapason générique (défini dans Diapason)",*
>
> *"DIAP_IDENTIFIANT": "<<Identifiant unique de la requête de Service Web dans DIAPASON (GUID)>>",*
>
> *"DIAP_STATUTS": "<<Statuts de sortie de la requête SUCCES ou ERROR>>",*
>
> *"DIAP_DATA": { << Objet Json contenant les informations du service>> }*
>
> *L'objet "DIAP_DATA"*
>
> *Voici la liste des paramètres de retour de l'objet "DIAP_DATA" :*

Le début de la description du corps du message est donc fixe. Le contenu de l'objet DIAP_DATA est à décrire par vous avec les informations suivantes :

- **Pour un paramètre simple**

Présentez les paramètres fixes et facultatif avec pour chaque élément :

Nom [type] : Description

Exemple :

Paramètres fixes :

Param1[caractère] : Description du paramètre 1

Param2[caractère] : Description du paramètre 2

Paramètres facultatifs :

Param3[décimal] : Description du paramètre 3

Pour un paramètre de type « liste »

Donnez le nom de la liste puis la description de son contenu :

NomDuChamp[type] : Description

Pour chaque liste, pensez à ajouter les informations suivantes :

> *Les paramètres d'une liste sont fixe d'un élément de la liste à l'autre.*
>
> *L'ordre des éléments de la liste ne doit pas être pris en compte.*

- **Exemple**

Donnez un exemple de contenu du corps du message (avec différents cas si possible).

Voici un exemple d'exemple :


```
{
  "DIAP_METHODE": "Nom de la méthode de service web Diapason générique (défini dans Diapason)",
  "DIAP_IDENTIFIANT": "<<Identifiant unique de la requête de Service Web dans DIAPASON (GUID)>>",
"DIAP_STATUTS":" SUCCES ",

  "DIAP_DATA": {
    "<<<Paramètre1>>>": "<<<Valeur du Paramètre1>>>",
    "<<<Paramètre2>>>": "<<<Valeur du Paramètre2>>>",
    "<<<Paramètre3>>>": "<<<Valeur numérique du Paramètre3>>>",
    "<<<ExempleListe>>>": [
      {
        "ChampC": "ValCar1",
        "ChampN": 45.0,
        "ChampD": "2020-06-11",
        "ChampL": false
      },
      {
```

```
      "ChampC": "ValCar2",
      "ChampN": 22.5,
      "ChampD": "2020-06-10",
      "ChampL": true
    },
    {
      "ChampC": "ValCar3",
      "ChampN": 15.0,
      "ChampD": "2020-06-09",
      "ChampL": false
    }
  ]
 }
}
```


Réponse Erreur

L'utilisation de l'instruction LC-ERREUR dans la REB génèrera un code 500 avec la description du LC-ERREUR dans l'élément « message ».

Vous devez donc ajouter et décrire la liste des erreurs gérées dans ce tableau.

Voici la liste des codes d'erreurs issus de Diapason avec le message associé :

Code	Message

401	Authentification incorrecte :
	{
	"DIAP_METHODE": "Nom de la méthode de service web Diapason générique (défini dans Diapason)",
	"DIAP_IDENTIFIANT": "<<Identifiant unique de la requête de Service Web dans DIAPASON (GUID)>>",
	"DIAP_STATUTS":"ERROR",
	"message": "Unauthorized"
	}

500	Erreur interne : { "DIAP_METHODE": "Nom de la méthode de service web Diapason générique (défini dans Diapason)", "DIAP_IDENTIFIANT": "<<Identifiant unique de la requête de Service Web dans DIAPASON (GUID)>>", "DIAP_STATUTS":"ERROR", "message": "Internal Server Error"

	}

501	le serveur ne prend pas en charge la fonction demandée. :
	{
	"DIAP_METHODE": "Nom de la méthode de service web Diapason générique (défini dans Diapason)",
	"DIAP_IDENTIFIANT": "<<Identifiant unique de la requête de Service Web dans DIAPASON (GUID)>>",
	"DIAP_STATUTS":"ERROR",
	"message": "Not Implemented"
	}

Caractéristiques complémentaires

Les caractéristiques complémentaires suivantes doivent impérativement être dans le document final.

- **Encodage**

> *Encodage : UTF-8*

- **Format des données**

> *Pour les valeurs de type numériques, le séparateur de décimale est le « . ».*
>
> *Les valeurs logiques seront décrites par « true » ou « false ».*
>
> *Les dates auront la forme suivante : <année>-<mois>-<jour> (exemple : "2020-06-30")*

Les Services WEB EXTERNES PARAMETRABLES →