WebServices w.c.s.

Cette page décrit les modes de communication de w.c.s. avec l'extérieur.

Généralités

Protocole général : REST, données en JSON.

Gestion des accès via des API Users et des ACLs (sur les URLs REST).

API w.c.s. Objects

Permet de créer, lire, mettre à jour ou effacer des formulaires (CRUD sur formdata).

- Accès aux formdef en lecture seule : **/api/v1/formdef/X**
- Accès aux formdata :
  - **/api/v1/formdef/X/formdata/Y** : CRUD
  - **/api/v1/formdef/X/formdata/Y/data** uniquement sur les données, read/update
  - **/api/v1/formdef/X/formdata/Y/wf** sur le workflow : read/update. En read on obtient le statut actuel et les suites possibles (dont les données attendues) ; en write on envoie les éventuelles données demandées pour un changement de statut choisi
  - **/api/v1/regie/<régie>/invoice/Y** : CRUD

Cas d'usage

- Orléans: connecteur facturation
- Noyelles-Godault: externalistion du connecteur Abelium compte famille et facturation

API w.c.s. Form (réponse à distance)

Permet de remplir un formulaire pas à pas, en suivant éventuellement les pages et en obtenant une réponse à la fin (première étape du workflow)

POST sur **/api/v1/formdef/X/page/Y**

Retourne des codes 200, 300, 400, 500... selon ce qui doit se passer pour continuer ou reprendre la saisie, selon le cas.

API w.c.s. DataSources

Utilisé par w.c.s. :

- pour renseigner les valeurs possibles d'un select
- pour le préremplissage d'un champ en fonction d'autres données

w.c.s. interroge la source distance avec une requête de type "JSONSQL" (à trouver ou à inventer)
Du genre :

```
{
  "columns": [ "a", "b", "c"],
  "where": ['or', ['a', '<', 1], ['b', '=', 'xyz']],
  "offset": 5,
  "limit": 2
}
```

Retourne un JSON :

```
[
 { "a": ..., "b": ..., "c": ... },
 { "a": ..., "b": ..., "c": ... }
]
```

Cas d'usages

- CG14 recherche d'homonymie
- Montpellier & CG14 complétion des adresses

API w.c.s. Notify

Utilisé par w.c.s. pour envoyer des messages. Peut recevoir une réponse en synchrone ou asynchrone (via une URL de retour).

Requête :

```
{ "data": ...,
  "replyto": url
}
```

Réponse attendue : (selon l'URL de replyto ?)

```
{ "data": ...,
}
```

Cas d'usage

- Vincennes: faire interagir les workflows W.C.S. avec autre chose genre K2