

<b>Sujet</b>	Mode opératoire des scripts .php et .sh	<b>Rédacteur</b>	Arnaud AUZOLAT	<b>Statut</b>	Validé
<b>Date de création</b>	26/03/10	<b>Date de modification</b>	10/09/10	<b>Diffusé le</b>	

## SOMMAIRE

<b>1. Importation des fichiers pôle emploi .....</b>	<b>2</b>
<b>2. Importation des fichiers APRE de la CAF (CG93).....</b>	<b>2</b>
<b>3. Affichage des anomalies de la base de données (dossiers en doublon, ..).....</b>	<b>2</b>
<b>4. Calcul de la moyenne des ressources, de la pré-orientation et de la notion de soumis à droits et devoirs de l'allocataire.....</b>	<b>3</b>
<b>5. Importations des cantons (CG66).....</b>	<b>4</b>
<b>6. Importations des domiciliations bancaires.....</b>	<b>4</b>
<b>7. Détection du parcours de l'allocataire (module équipe pluridisciplinaire).....</b>	<b>5</b>
<b>8. Liste des révisions de fichier (matrice des changements).....</b>	<b>5</b>
<b>9. Modification de la casse des allocataires insérés manuellement dans WebRSA.....</b>	<b>5</b>
<b>10. Suppression du cache.....</b>	<b>5</b>
<b>11. Mise à jour des modèles .odt.....</b>	<b>6</b>
<b>12. Création de divers dump de base de données.....</b>	<b>6</b>
<b>13. Exportation d'un dossier RSA à partir d'un fichier XML.....</b>	<b>7</b>
<b>14. Algorithme de préorientation (gestion en cohorte).....</b>	<b>7</b>
<b>15. Opérations de maintenance sur la base de données Postgresql.....</b>	<b>8</b>
<b>16. Enregistrement des orientations en fichiers pdf.....</b>	<b>9</b>

Les scripts sont soit de type sh, soit de type php. Les scripts php sont placés dans le répertoire /var/www/webrsa/app/vendors/shells/ et les scripts sh sont placés dans le répertoire /var/www/webrsa/app/.

## 1 Importation des fichiers pôle emploi

Ce script permet l'importation des fichiers .csv inscriptions, cessations et radiations transmis par Pôle Emploi.

- *Nom du script* : [importcsvinfope.php](#)
- *Commande à taper* : `cake/console/cake importcsvinfope -type <TYPE>  
<CHEMIN>/<NOM_FICHER.csv>`
- *Paramètres possibles* :
  - `-type` : type de fichier csv à importer : [inscription](#), [cessation](#), [radiation](#) ;
  - `-help` : pour obtenir de l'aide sur le script.
- *Exemple* :  
`cake/console/cake importcsvinfope -type inscription /tmp/RSA_INSCRIPTION_201002_DPT34.csv`

## 2 Importation des fichiers APRE de la CAF (CG93)

Ce script permet, au CG93, d'importer, via des fichiers .csv, des APREs forfaitaires transmis par la CAF.

- *Nom du script* : [importcsvapres.php](#)
- *Commande à taper* : `cake/console/cake importcsvapres -parametre  
<CHEMIN>/<NOM_FICHER.csv>`
- *Paramètres possibles* :
  - `-headers` : valeurs possibles: [true](#) ou [false](#) (true par défaut) – précise si le fichier à importer commence par une colonne d'en-tête ou s'il commence directement par des données à intégrer ;
  - `-separator` : le caractère utilisé comme séparateur (par défaut ";" ) ;
  - [chemin et nom du fichier à importer](#) ;
  - `-?` : pour obtenir de l'aide sur le script.
- *Exemple* :  
`cake/console/cake importcsvapres -separator ";" /tmp/APRE_201003_DPT34.csv`

## 3 Affichage des anomalies de la base de données (dossiers en doublon, ...)

Le script a pour but de détecter certaines anomalies dans la base de données (doublons,...)

- *Nom du script* : [anomaliesbdd.php](#)
- *Commande à taper* : `cake/console/cake anomaliesbdd -parametre true/false`
- *Paramètres possibles* :
  - `-type` : type de rapport (soit full, soit short). Le type est short par défaut

- **short** : permet de comptabiliser les différentes erreurs ;
  - **full** : permet de lister toutes les entrées posant problème.
  - **-dossiersvides** : dossiers ne contenant aucune personne liée
    - **true** (par défaut) ;
    - **false**.
  - **-dossierssansdemandeur** : dossiers contenant des personnes, mais sans demandeur RSA
    - **true** (par défaut) ;
    - **false**.
  - **-nirsdoublonsfoyersdiff** : personnes avec NIR en doublon dans des foyers différents
    - **true** (par défaut)
    - **false**
  - **-nirsdoublonsmemefoyer** : personnes avec NIR en doublon dans le même foyer
    - **true** (par défaut)
    - **false**
  - **-personnesdoublonsfoyersdiff** : personnes avec au moins un NIR manquant en doublon dans des foyers différents
    - **true** (par défaut)
    - **false**
  - **-personnesdoublonsmemefoyer** : personnes avec au moins un NIR manquant en doublon dans le même foyer
    - **true** (par défaut)
    - **false**
  - **-demcjtmultiples** : personnes demandeurs ou conjoints multiples au sein d'un foyer
    - **true** (par défaut)
    - **false**
  - **-help** : pour obtenir de l'aide sur le script
- *Exemple* :

`cake/console/cake anomaliesbdd -dossierssansdemandeur false -nirsdoublonsfoyersdiff false -nirsdoublonsmemefoyer false -personnesdoublonsfoyersdiff false -personnesdoublonsmemefoyer false`

#### 4 Calcul de la moyenne des ressources, de la pré-orientation et de la notion de soumis à droits et devoirs de l'allocataire

Le script refresh a pour but : de recalculer les ressources, de déterminer si l'allocataire est soumis à droits et devoirs et de calculer la pré-orientation (via des règles de gestion basées sur les DSP de l'allocataire).

- *Nom du script* : **refresh.php**
- *Commande à taper* : `cake/console/cake refresh -parametre true/false`
- *Paramètres possibles* :
  - **-ressources** : doit-on recalculer la moyenne des ressources mensuelles des demandeurs et des conjoints ?
    - **true** (par défaut)
    - **false**
  - **-soumis** : doit-on recalculer si les demandeurs et les conjoints sont soumis à droits et devoirs ?
    - **true** (par défaut)

- **false**
  - **-preorientation** : doit-on calculer et sauvegarder une préorientation pour les demandeurs et les conjoints qui ne sont pas encore orientés ni préorientés ?
    - **true** (par défaut)
    - **false**
  - **-force** : doit-on forcer le calcul de la préorientation même si une préorientation a déjà été calculée ?
    - **true** (par défaut)
    - **false**
  - **-help** : pour obtenir de l'aide sur le script
- *Exemple* :

`cake/console/cake refresh -ressources true -soumis true -preorientation true -force true`

## 5 Importations des cantons (CG66)

Ce script permet, au CG66, d'importer, via des fichiers .csv, la liste des cantons du département des Pyrénées Orientales.

- *Nom du script* : **importcsvcantons.php**
- *Commande à taper* : `cake/console/cake importcsvcantons <CHEMIN>/<NOM_FICHER.csv>`
- *Paramètres possibles* :
  - **chemin et nom du fichier à importer.**
- *Exemple* :

`cake/console/cake importcsvcantons /tmp/CANTONS_DPT66.csv`

## 6 Importations des domiciliations bancaires

Ce script permet d'importer la liste complète (nationale) des domiciliations bancaires. La table "domiciliationsbancaires" est composée des champs code banque, code agence et libellé de la domiciliation.

- *Nom du script* : **importcsvdomiciliationsbancaires.php**
- *Commande à taper* : `cake/console/cake importcsvdomiciliationsbancaires <CHEMIN>/<NOM_FICHER.csv>`
- *Paramètres possibles* :
  - **-headers** : valeurs possibles: **true** ou **false** (true par défaut) – précise si le fichier commence par une colonne d'en-tête ou s'il commence directement par des données à intégrer ;
  - **-separator** : le caractère utilisé comme séparateur (par défaut ";" ) ;
  - **chemin et nom du fichier à importer** ;
  - **-?** : pour obtenir de l'aide sur le script.
- *Exemple* :

`cake/console/cake importcsvdomiciliationsbancaires -headers true -separator ";" /tmp/domiciliations_bancaires.csv`

## 7 Détection du parcours de l'allocataire (module équipe pluridisciplinaire)

Ce script est lié au module équipe pluridisciplinaire. Il permet de lister les allocataires qui ont été orientées vers le social ou le socioprofessionnel depuis plus de 6 mois et qui n'ont pas signé de contrat d'engagement réciproque ou dont la date de fin du contrat d'engagement réciproque est dépassée.

- *Nom du script* : [detectionparcours.php](#)
- *Commande à taper* : [cake/console/cake detectionparcours](#)
- *Paramètres possibles* : pas de paramètres
- *Exemple* :

[cake/console/cake detectionparcours](#)

## 8 Liste des révisions de fichier (matrice des changements)

Ce script génère un fichier "[changematrix.html](#)" qui présente la date de dernière modification des fichiers controllers, models et views pour chacun des modules. L'équipe RSA génère systématiquement ce fichier à chaque montée de version. Le fichier "[changematrix.html](#)" s'enregistre dans le répertoire racine à savoir : [/var/www/webrsa/](#).

- *Nom du script* : [changematrix.php](#)
- *Commande à taper* : [cake/console/cake changematrix <NUMÉRO\\_DE\\_VERSION>](#)
- *Paramètres possibles* :
  - [numéro de la version de WebRSA](#) ;
- *Exemple* :

[cake/console/cake changematrix 1.2](#)

## 9 Modification de la casse des allocataires insérés manuellement dans WebRSA

Ce script permet de modifier la casse des personnes qui ont le nom ou le prénom en minuscule. Ces personnes sont intégrées dans la base de données via le formulaire d'ajout de dossiers ou de personnes.

Ce script permet également de fusionner les personnes en doublon. Cette fonctionnalité n'est pas complètement opérationnelle et sera corrigée ultérieurement.

- *Nom du script* : [personnescase.php](#)
- *Commande à taper* : [cake/console/cake personnescase](#)
- *Paramètres possibles* : ce script ne prend pas de paramètres
- *Exemple* :

[cake/console/cake personnescase](#)

## 10 Suppression du cache

Ce script permet de supprimer le cache dans les répertoires [var/www/webrsa/app/tmp/](#) et [var/www/webrsa/app/logs/](#).

Il permet également de préparer les packages, les patches et le fichier changelog.txt lors des montées de versions. Certains paramètres {[changelog](#) | [package](#) | [patch](#)} ne sont utilisés que par l'équipe de développement.

Ce script peut être exécuté soit depuis la racine de l'application (en ajoutant "[app](#)") soit depuis le répertoire `app` (il faut ajouter dans ce cas `./`).

- *Nom du script* : [webrsa.sh](#)
- *Commande à taper* : [app/webrsa.sh](#) ou [./webrsa.sh](#)
- *Paramètres possibles* :
  - [clear](#) : efface le cache dans les répertoires `var/www/webrsa/app/tmp/cache/` et `var/www/webrsa/app/tmp/logs/`
  - [clearcache](#) : efface le cache tous les fichiers dans le répertoire `var/www/webrsa/app/tmp/cache/`
  - [clearlogs](#) : efface le cache tous les fichiers dans le répertoire `var/www/webrsa/app/tmp/logs/`
  - [package](#) : paramètre utilisé uniquement par l'équipe webrsa
  - [patch](#) : paramètre utilisé uniquement par l'équipe webrsa
  - [changelog](#) : paramètre utilisé uniquement par l'équipe webrsa
- *Exemple* :

[app/webrsa.sh clearcache](#)

## 11 Mise à jour des modèles .odt

Ce script permet, lors de montée de version, de remplacer les modèles `.odt` par les modèles `.odt.default` qui sont plus récents. Ce script est utilisé par le CG93 dans une procédure de remplacement des différents modèles de courriers, contrats,... Les fichiers remplacés sont ceux placés dans le répertoire (et sous-répertoire) `/var/www/webrsa/app/vendors/modelesodt/`.

Ce script peut-être exécuté soit depuis la racine de l'application (en ajoutant "[app](#)") soit depuis le répertoire `app` (il faut ajouter dans ce cas `./`).

- *Nom du script* : [majmodeles.sh](#)
- *Commande à taper* : [app/majmodeles.sh](#) ou [./majmodeles.sh](#)
- *Paramètres possibles* : ce script ne prend pas de paramètres.
- *Exemple* :

[app/majmodeles.sh](#)

## 12 Création de divers dump de base de données

Ce script permet d'exporter des dump sql de tout ou partie de la base de données webrsa.

Ce script peut-être exécuté soit depuis la racine de l'application (en ajoutant "[app](#)") soit depuis le répertoire `app` (il faut ajouter dans ce cas `./`).

- *Nom du script* : [exportdb.sh](#)
- *Commande à taper* : [app/exportdb.sh](#) ou [./exportdb.sh](#)
- *Paramètres possibles* :

- [alldatabase](#) : exporte la structure et les données de la base de données ;
- [allBD\\_struct](#) : exporte uniquement la structure de la base de données ;
- [allBD\\_data](#) : exporte uniquement les données de la base de données ;
- [tablesmetiers](#) : exporte les données des tables métiers uniquement ;
- [tablesmetiers\\_struct](#) : exporte la structure des tables métiers uniquement ;
- [tablesapplicatives](#) : exporte les données des tables applicatives uniquement ;
- [tablesapplicatives\\_struct](#) : exporte la structure des tables applicatives uniquement.

- *Exemple* :

[app/exportdb.sh tablesmetiers](#)

### 13 Exportation d'un dossier RSA à partir d'un fichier XML

Ce script permet d'extraire un fichier XML contenant un dossier RSA particulier à partir d'un fichier XML du flux CAF. Le fichier exporté contient les informations (blocs XML) entre les balises <InfosFoyerRSA> et </InfosFoyerRSA> à partir du fichier passé en premier paramètre.

Ce script peut-être exécuté soit depuis la racine de l'application (en ajoutant "**app**") soit depuis le répertoire app (il faut ajouter dans ce cas ".").

- *Nom du script* : [extractdossier.sh](#)
- *Commande à taper* : [app/extractdossier.sh](#) ou [./extractdossier.sh](#)
- *Paramètres possibles* :
  - [le chemin et le nom du fichier de flux XML](#)
  - [le numéro de dossier RSA à exporter](#)

- *Exemple* :

[app/extractdossier.sh /tmp/NRSACGIR\\_IRSACG\\_20090929\\_I2921245.RCV 01043441093](#)

### 14 Algorithme de préorientation (gestion en cohorte)

Ce script permet de calculer la préorientation des allocataires. L'algorithme de préorientation se base sur quatre données socioprofessionnelles. Les données socioprofessionnelles sont alimentées par les fichiers XML CAF instructions. Voici les règles qui composent cet algorithme :

- **règle 1** (prioritaire) : code XML : **NATLOG** (nature du logement). **3** cas de figure :
  - si la valeur est " 0904 " = *Logement d'urgence* : CHRS alors orientation vers le **social** ;
  - si la valeur est " 0911 " = *Logement précaire : résidence sociale* alors orientation vers le **social** ;
  - sinon on passe à la **règle 2**.
- **règle 2** (prioritaire) : code XML : **DIFLOG** (difficultés logement). **2** cas de figure :
  - si la valeur est " 1006 " = *Fin de bail, expulsion* alors orientation vers le **social** ;
  - sinon on passe à la **règle 3**.
- **règle 3** (prioritaire) : code XML : **SITPERSDEM RSA** (motif de la demande RSA). **5** cas de figure :
  - si la valeur est " 0102 " = *Fin de droits AAH* alors orientation vers le **social** ;
  - si la valeur est " 0105 " = *Attente de pension vieillesse ou invalidité, ou d'allocation handicap* alors orientation vers le **social** ;
  - si la valeur est " 0109 " = *Fin d'études* alors orientation vers l'**emploi** ;

- si la valeur est " 0101 " = *Fin de droits ASSEDIC* alors orientation vers l'emploi ;
- sinon on passe à la **règle 4**.
- **règle 4** (prioritaire) : codes XML : **DTNAI** (date de naissance) et **CESSDERACT** (cessation de dernière activité). **5** cas de figure :
  - si la valeur de DTNAI est " <57 " et CESSDERACT est " 2701 " (*encore en activité ou cessation depuis moins d'un an*) alors orientation vers l'emploi ;
  - si la valeur de DTNAI est " <57 " et CESSDERACT est " 2702 " (*cessation d'activité depuis plus d'un an*) alors orientation vers le socioprofessionnel ;
  - si la valeur de DTNAI est " >=57 " et CESSDERACT est " 2701 " (*encore en activité ou cessation depuis moins d'un an*) alors orientation vers le socioprofessionnel ;
  - si la valeur de DTNAI est " >=57 " et CESSDERACT est " 2702 " (*cessation d'activité depuis plus d'un an*) alors orientation vers le social ;
  - sinon on passe à la **règle 5**.
- **règle 5** : code XML : **HISPRO** (passé professionnel). **2** cas de figure :
  - si la valeur est " 1901 " (*vous avez toujours travaillé*) alors orientation vers l'emploi ;
  - si la valeur est " 1902 " (*vous avez travaillé par intermittence*), ou " 1903 " (*vous avez déjà exercé une activité professionnelle*), ou " 1904 " (*vous n'avez jamais travaillé*) alors orientation vers le socioprofessionnel.

Ce script doit être exécuté depuis la racine de l'application (" /var/www/webrsa/ ").

- *Nom du script* : [preorientation.php](#)
- *Commande à taper* : [cake/console/cake preorientation <parametre>](#)
- *Paramètres possibles* :
  - **-force** : valeurs possibles: **true** ou **false** (true par défaut) – Doit-on forcer le calcul de la préorientation même si une préorientation a déjà été calculée ? ;
  - **-help** : pour obtenir de l'aide sur le script
- *Exemple* : [cake/console/cake preorientation -force false](#)

## 15 Opérations de maintenance sur la base de données Postgresql

Ce script permet de réaliser certaines opérations de maintenance sur une base de données PostgreSQL dont les paramètres de connexion sont définis dans app/config/database.php.

Ce script doit être exécuté depuis la racine de l'application (" /var/www/webrsa/ ").

- *Nom du script* : [postgresql.php](#)
- *Commande à taper* : [cake/console/cake postgresql <connection> <commande>](#)
- *Paramètres / commandes possibles* :
  - **-help** : pour obtenir de l'aide sur le script
  - **connection** : paramètres de connexion définis dans le fichier de configuration de BDD " app/config/database.php ". Ce paramètre peut prendre une des valeurs suivantes : *default*, *test* (*default étant le tableau dans lequel sont définis les paramètres de connexion de la base principale*).



- **sequences** : cette commande donne le prochain numéro du champ auto-incrémenté (" id " = clé primaire) de la table. Le script met à jour les séquences de toutes les tables de la base de données (boucle).
- *Exemple* : `cake/console/cake postgresql default sequences`

## 16 Enregistrement des orientations en fichiers pdf

Ce script se charge de créer une impression d'orientation en *.pdf* en base de données pour toutes les personnes ayant été orientées avant la version 2.0rc2. Ces fichiers doivent être enregistrés en base pour voir les personnes orientées depuis le menu « Cohorte → Orientation → Demandes orientées » ou le menu « Recherches → Par Orientation ». Ce script n'est à exécuter qu'une seule fois.

Ce script doit être exécuté depuis la racine de l'application (" /var/www/webrsa/ ").

- *Nom du script* : `cohortepdfs.php`
- *Commande à taper* : `cake/console/cake cohortepdfs <parametre>`
- *Paramètres / commandes possibles* :
  - `-help` : pour obtenir de l'aide sur le script
  - `-limit` : Nombre d'enregistrements à traiter. Doit être un nombre entier positif. Par défaut: 10. Utiliser `0` ou `null` pour ne pas avoir de limite et traiter tous les enregistrements.
  - `-order` : Permet de trier les enregistrements à traiter par date de validation de l'orientation (" date\_valid ") en ordre ascendant ou descendant. Valeurs possibles: `asc` ou `desc`. Par défaut: `asc`.
- *Exemple* : `cake/console/cake cohortepdfs -limit 100 -order asc`