Code Analysis Helper

BlueCompany a développé un composant qui répond aux problématiques suivantes :

1. Je dois migrer une base d'une version X de 4D vers une version récente
2. Mon applicatif est lent, y-a-t-il des recommandations pour agir rapidement ?
3. J'ai un programme sur lequel j'ai moins investi depuis quelques années, par où commencer pour supprimer les archaïsmes ?
4. Je dois reprendre un programme que je ne connais pas du tout, dans quel état technique est ce programme sur lequel je vais devoir intervenir ?
5. Je dois m'affranchir de la bibliothèque d'image
6. Je dois supprimer les liens automatiques de ma structure
7. ...

Pour tout cela Code Analysis Helper va vous aider à répondre aux questions suivantes :

1. Dans quoi est-ce que je mets les pieds ?
2. Quelles sont les tâches à mener et à déléguer ?
3. Quelles sont les méthodes à risques qui doivent être inspectées ?
4. Combien de fois est ou sont utilisé(es) la(es)commande(s) ?
5. Où sont utilisés des Plugins et quelle partie du plugin est utilisée ?
6. Où sont utilisées les images de la bibliothèque ?

1. Evolution

Release notes du composant "Code Analysis Helper"

Version 2.0 de fin janvier en cours de développement

ATTENTION CETTE VERSION NÉCESSITE 4D V16

Interface

1. Création de la palette
2. Analyse des attributs utilisés dans les objets dans le fichier de données

Outils :

1. Possibilité d'ajouter dans votre base des méthodes générant par programmation les barres de menus équivalentes à celle que vous avez créé en dur dans votre base..
2. Ajout dans la liste des commandes la version dans laquelle la commande est devenu obsolète ainsi que la date.

Nouveaux contrôles :

.

Version 1.5.0 du 24 janvier 2019

CETTE VERSION EST LA DERNIÈRE EN 4D V15

Interface

1. Réorganisation de l'interface pour un meilleur accès aux différentes fonctions.

Outils :

1. Génération de la liste des attributs utilisés dans les objets et détection d'anomalie
2. Fonction de beautification du code

Cette fonction remet en ordre l'apparence du code.

Elle supprime les doubles sauts de ligne Elle supprime les \ suivi de retour chariot inutiles (4D en insérait lors du déplacement de méthode dans certaines version)

Le code suivant :

```
OB FIXER($objet;"attribut1";15;"attribut2";!00/00/0000!;"attribut3";?00:00:00?)
```

Sera transformé dans le code suivant

```
OB FIXER($objet;\
"attribut1";15;\
"attribut2";!00/00/0000!;\
"attribut3";?00:00:00?)
```

De manière à en faciliter la lecture.

Les commandes 4D traitées sont les suivantes :

1. Sélection vers tableau
2. Sélection limitée vers tableau
3. Tableau vers sélection
4. Trier tableau

1. ob fixer
2. Créer objet
3. Créer collection
4. Créer collection partagée
5. Tableau vers collection

1. ST fixer attributs
2. ST lire attributs

1. Exécuter méthode

3. Installation et retrait des méthodes de contrôle de verrouillage des enregistrements

L'objectif de cet outil est de vous permettre de déterminé les verrouillages d'enregistrements mal gérés dans une base 4D.

Pour ce faire le CAH va installer avant ou après l'appel de commandes enregistrant des données une méthode qui va contrôler si l'enregistrement c'est bien déroulé. Si ce n'est pas le cas la méthode va inscrire dans le log (LockingRecordLog.txt) qui sera créer dans le dossier des logs de la base l'heure et le problème rencontré.

Les commandes suivies sont les suivantes :

1. Appliquer à sélection
2. Stocker enregistrement
3. Supprimer enregistrement
4. Supprimer sélection
5. Tableau vers sélection
6. JSON vers sélection
7. Vider table

Pour utiliser insertion ou le retrait, vous devez avoir effectué l'analyse de la structure au préalable.

Si vous demandez au composant d'insérer à nouveau le code, il n'ajoutera bien sur le code que pour les nouveaux appels aux commandes. Dans le cas où vous avez beaucoup modifié votre code, vous pouvez demander une nouvelle insertion, cela mettra les N° de ligne à jour.

Le retrait retire TOUS les appels à la méthode de contrôle.

Exemple de log :

```
2018-12-15T13:09:05Z : Création du log
2018-12-15T13:09:05Z : [Société] LockedSet (3 records) - Method : Test_Lock (Line 11)
2018-12-15T13:09:05Z : [Société] Record Lock - Method : Test_Lock (Line 16)
2018-12-15T13:09:05Z : [Société] Record Lock - Method : Test_Lock (Line 16)
2018-12-15T13:09:05Z : [Société] Record Lock - Method : Test_Lock (Line 16)
```

Le code avant les insertions de contrôle

```
TOUT SÉLECTIONNER([Société])
```

```
APPLIQUER À SÉLECTION([Société];[Société]Nom:=[Société]Nom)

DÉBUT SELECTION([Société])
Tant que (Non(Fin de sélection([Société])))
        [Société]Nom:=[Société]Nom

        STOCKER ENREGISTREMENT([Société])
        ENREGISTREMENT SUIVANT([Société])
Fin tant que
```

Le code après les insertions de contrôle

```
TOUT SÉLECTIONNER([Société])

APPLIQUER À SÉLECTION([Société];[Société]Nom:=[Société]Nom)
ANACAH_DebugLockingRecord ("Selection";"Test_Lock (Line 11)";->[Société])  // insert by CAH 2018-12-04

DÉBUT SELECTION([Société])
Tant que (Non(Fin de sélection([Société])))
        [Société]Nom:=[Société]Nom

        STOCKER ENREGISTREMENT([Société])
        ANACAH_DebugLockingRecord ("record";"Test_Lock (Line 16)";->[Société])  // insert by CAH 2018-12-04
        ENREGISTREMENT SUIVANT([Société])
Fin tant que
```

le CAH vous permet bien sur de retirer tout ce code de contrôle.

Il vous permet également d'obtenir des statistiques sur les enregistrements dans votre base.

4. Comptage des commandes d'enregistrement et répartition par table
5. Ajout d'un écran de comptage et d'affichage des fonctions utilisées comme des commandes (accès par l'écran "Commands"), les fonctions suivies sont pour le moment :
    1. Créer fenêtre
    2. Créer fenêtre formulaire
    3. Imprimer ligne
    4. Nous ajouterons d'autres fonctions dans les versions à venir.

Fonctions :

1. Gestion des feuilles de styles
    1. Liste des feuilles de styles et de leurs utilisations
    2. Détection des objets sans feuilles de style

Améliorations

1. Ajout de la fonction "Fixer recherche et verrouillage" dans les commandes à suivre
2. Ajout de la fonction "Fixer recherche et verrouillage" dans les commandes ouvrantes fermantes
3. Dans les patterns : détection des "charger enregistrement" inutiles après un tri
4. Détections des liens entrants sur les clefs primaires pour en mesurer l'utilisation (écran structure)
5. Modification du code pour aller a une ligne dans le code pour visualiser les erreurs plus facilement

Corrections

1. Correction des détections de la commande appeler 4D
2. Les fonctions d'export et de rechargement d'une analyse traitent bien l'intégralité des données
3. Correction de beaucoup de petits bugs...
4. Correction d'un bug lors de l'impression d'une liste.
5. Correction de l'import et de l'export des analyses.

.

Version 1.3.0.1 (Hot fix) du 20 décembre 2017

1. Déclaration d'une variable dans la méthode ANACAH_GetInfoMenu
2. Correction de la détection du fichier de symbole

.

Version 1.3.0 du 19 décembre 2017

Fonctions :

1. Affichage dans les metrics de la taille des variables process et interprocess de la base
2. Détection des méthodes n'utilisant que des commandes thread safe
3. Détection des appels de méthodes via les menus créés par la boite à outils
4. Début d'analyse du fichier des symboles et du fichier de build
5. Formatage des commentaires ajoutés par le CAH (dans les préférences)
6. Ajout de graphes dans 'Sort & Query'
    1. Ajout de graphe de répartition des recherches
    2. Ajout de graphe de répartition des tris
7. Ajout de nombreuses préférences
8. Ajout de la commande 'créer fenêtre' dans les commandes à surveiller
9. Ajout d'un écran de comptage et d'affichage des utilisations de sémaphores (accès par l'écran "Commands")

. Pattern

1. Détection des sémaphores non temporisés

. Commands to check

1. Détection des appels à "créer fenêtre" qui devraient être remplacés "Créer fenêtre formulaire"
2. Ajout du contrôle de l'utilisation de l'attribut d'objet 'length' qui entrera en conflit avec l'argument virtuel de même nom http://doc.4d.com/4Dv16R4/4D/16-R4/VALEURS-DISTINCTES-ATTRIBUT.301-3317238.fr.html#2998555

. Outils :

1. Ajout d'un outil pour analyser et migrer les menus
2. Ajout d'un outil pour analyser les pop-ups menus
3. Ajout d'un outil pour aider aux migrations de write vers Write Pro

    1. Première fonction : le comptage pour connaitre le nombre d'appels aux commandes 4D write et le nombre de formulaires et de méthodes concernés

4. Ajout d'un outil de détection des erreurs de formatage des directives de compilation qui ne sont plus tolérées et ajout d'un bouton de correction.

. Améliorations :

1. Sécurisation de la recherche dans le code (erreur de lecture de la liste des objets par 4D)
2. Sécurisation de la recherche par regex dans le code (Regex invalide, erreur de lecture de la liste des objets par 4D)

. Bugs

1. Correction de l'ascenseur de progression d'analyse
2. Correction de l'affichage des formulaires dans l'onglet 'Methods' dans l'écran 'Dependancies'
3. Correction du comptage d'appel des méthodes qui ne tenait pas compte des appels multiples au seins d'une même méthode
4. Correction de la sauvegarde et du chargement des analyses qui "oubliaient" des données

.

Version 1.2.4 du non publiée

.

Version 1.2.3 du 15 novembre 2017

Bugs

1. Problème de détection des images de la bibliothèque dans les boutons image
2. Problème d'affiche de HTML lors de la mise en couleur de certaines chaines (Il doit en rester encore, merci de nous les signaler)
3. Problèmes de détection des formulaires non utilisés
4. Problèmes de détection des formulaires non existants
5. Suppression de faux positifs liés à des commentaires

6. Nombreuses corrections sur la lecture des constantes, en particulier sur Mac. Ces erreurs de lecture provoquaient des bugs collatéraux.

Fonctions

1. Capsule de recherche dans les commandes
2. Capsule de recherche dans les méthodes
3. Ajout d'un sélecteur dans les formats de chaîne pour ne voir que les formats à corriger
4. Affichage du nombre d'utilisation des formulaires

.

Version 1.2.2 du 3 novembre 2017

Debug

1. Correction de bugs d'interface sur les bases avec des tables effacées
2. Suppression de faux positifs sur les commandes LECTURE ECRITURE - LECTURE SEULEMENT
3. Suppression de faux positifs sur le pattern sur les LIBÉRER ENREGISTREMENT inutiles
4. Suppression de faux positifs sur le pattern sur les SÉLECTION VERS TABLEAU sur plusieurs lignes
5. Erreur de 4D dans l'onglet des ressources lors de la lecture des métadonnées non trouvées
6. Ajout dans le comptage d'utilisation des ressources de l'utilisation des images dans les boutons

.

Version 1.2.1 du 1 novembre 2017

Debug

1. Correction de la disparition de nombreux ascenseurs dans les listes
2. Suppression de faux positifs (cde 'remplacer chaîne')
3. Les méthodes bases étaient analysées 2 fois : ce n'est plus le cas
4. Plus un ou deux petits trucs

. Relookage du composant

. Fonctions

1. Possibilité de filtrer les recherches pour ne voir que celles portant sur des champs non indexés et/ou celles ne portant que sur un champ
2. Surlignage en rouge des "anciens formats" de la commande chaîne dans la liste des formats de chaînes
3. Détection d’ouvrantes-fermantes
    1. Correction d’ouvrantes-fermantes qui ne fonctionnaient plus
    2. Mise en place des commentaires d'inactivations
    3. WR Hors ecran - WR DETRUIRE HORS ECRAN
4. Détection du nombre d'utilisation d'un formulaire et donc également des formulaires non utilisés
5. Détection des formulaires utilisés mais qui n'existent pas
6. Ajout de graphes sur les dates de modifications de méthodes afin de savoir sur quoi on a travaillé ces dernières années.
7. Ajout de graphe sur la répartition du code par Type de méthodes

.

Version 1.2 du 25 octobre 2017

Dans cette version nous avons réalisé un gros travail sur la structure du code afin de pouvoir encore accélérer l'analyse et de pouvoir vous proposer de nouvelles fonctions sans détériorer les performances du composant. En effet nous ajoutons des fonctions à un rythme soutenu et plus de détections implique plus de traitements et donc un ralentissement global de l'analyse. Vous devriez constater une forte amélioration pour les analyses qui duraient plus de 10 mn

. Debug

1. Correction de divers bugs
2. Suppression de certains faux positifs

Fonctions

1. Détection des paramètres de la commande "chaîne" et de l'utilisation des formats dans les formulaires et affichage dans l'onglet commands

2. Détection des énumérations et de leurs usages et affichage dans l'onglet commands
3. Export et import d'une analyse
4. Ajout des nombres d’occurrences dans les listes de pattern et d'ouvrantes/fermantes

Optimisations

1. Amélioration des temps d'analyse
2. Amélioration de l’algorithme d'analyse (nombreuses retombées et suppression de faux positif dans les versions à venir)
3. Optimisation du code de détection des "Commands to check"
4. Activation des commentaires d'inactivation sur toutes les "Commands to check"
5. Modification de l'affichage de la progression de l'analyse
6. Passage de l'analyse dans un process indépendant en vue du passage aux workers
7. Optimisation du code : zéro variables process

Version 1.1.2 du 17 octobre 2017

Correction de bug

1. Correction de divers bugs
    1. Détection des Evénements sur la forme en elle-même ne détectait pas les bons
    2. Détection de fermant suivi de commentaires
    3. Erreur sur clic d'une image statique

. Suppression de certains faux positifs

1. Fixer destination de recherche
2. SMTP New/Clear
3. Position
4. Remplacer chaine

Fonctions

1. Détection des ensembles de Listbox

.

Version 1.1.1 du 16 octobre 2017

Correction de bug

1. Correction de divers bugs

. Fonctions

1. Détections des ressources appelées dans le code
2. Récupération des images statiques et possibilité de les enregistrer
3. Visualisation de ressource de type text
4. Ajout du nombre d'image de la bibliothèque

. Commandes à vérifier

1. Correction de la détection des Majusc , Minusc et remplacer chaine sans *

. Commandes ouvrantes fermantes

1. ré-écriture du code de détection de
    1. Fixer destination Recherche
    2. Fixer limite Recherche
    3. EMPILER / DEPILER
    4. SMTP_New /Clear

.

Version 1.1.0 du 15 octobre 2017

Correction de bug

1. Les commentaires sur une ligne fermant un ouvrante/fermante empêchaient une détection correcte de la fermante
2. La détection des 'si(faux)' dans un code en anglais pouvait provoquer un bug en raison d'un texte resté en dur en français
3. Suppression de faux positifs
4. Contournement d'un problème général avec le composant dans un 4D 32 bits du a problème gestion des parenthèses par 4D en 32bits compilé

5. Amélioration de l'analyse des recherches pour l'analyse de l'utilisation des index

. Fonctions

1. Analyse des ressources utilisées ou doublonnées
2. Surlignage en rouge des liens récursifs
3. Détection des méthode appelées via
    1. 'APPELER SUR ERREUR'
    2. 'EXÉCUTER SUR CLIENT'
    3. 'EXÉCUTER MÉTHODE'
    4. 'EXÉCUTER MÉTHODE DANS SOUS FORMULAIRE'
    5. 'FIXER METHODE LIGNE MENU'
4. Détection des images statiques dans les formulaires

. Pattern

1. Détection de 'Charger enregistrement' inutiles
2. 'TOUT SÉLECTIONNER' suivi de 'SUPPRIMER SÉLECTION' à remplacer par un 'VIDER TABLE'

Divers

1. Mise en place d'un code défensif pour les formulaires comportant des objets sans nom.

.

Version 1.0.10 du 06/10/2017

Nouvelles fonctions :

1. Possibilité de visualiser les commandes inutilisées

Détection des patterns :

1. LIBÉRER ENREGISTREMENT suivi de RÉDUIRE SÉLECTION(0)

Détection d'ouvrante fermante

1. Correction : Certaines ouvrantes/fermantes n'étaient plus détectées
2. Détection de "ajouter a document" non suivi de "FERMER DOCUMENT"

3. Détection de "DOM Analyser source XML" non suivi de "DOM FERMER XML"
4. Détection de "DOM Analyser variable XML" non suivi de "DOM FERMER XML"

Commands to check

1. Détection de "AJOUTER ENREGISTREMENT"
2. Détection de "MODIFIER ENREGISTREMENT"
3. Détection de "AFFICHER ENREGISTREMENT"
4. Détection de "AJOUTER ENREGISTREMENT"# Détection de "VISUALISER SÉLECTION"
5. Détection de "MODIFIER SÉLECTION"
6. Détection de minusc non suivi de *
7. Détection de majusc non suivi de *
8. Détection de l'utilisation de "LISTE VERS TABLEAU"
9. Détection de l'utilisation de "TABLEAU VERS LISTE"

.

Version 1.0.9 du 26/09/2017

1. Correction de faux positifs
2. Détection du 'lecture seulement (*)' pour la fermeture des ouvrantes fermantes
3. Amélioration de la détection des méthodes récursives
4. Création de la première version de la fonction 'Beautify'
5. Mise en place de la recherche des méthodes par regex
6. Détection et comptage des 'Caractère(xx)' dans l'onglet 'Commands' choisir l'écran 'count'
7. Détection des tris sur des champs non indexés dans l'analyse des tris
8. Correction d'un message d'erreur qui indique qu'on n'utilise pas la bonne version de 4D alors qu'en fait on n'a pas autorisé l'exécution de la méthode "sur événement base hôte" dans l'onglet sécurité de la base à analyser.

.

Version 1.0.8 du 16/09/2017

1. Amélioration de la détection des tris séquentiels
2. Détections des patterns suivants liés a LIBÉRER ENREGISTREMENT :
    1. LIBÉRER ENREGISTREMENT suivi de TRIER
    2. LIBÉRER ENREGISTREMENT suivi de ENREGISTREMENT PRÉCÉDENT
    3. LIBÉRER ENREGISTREMENT suivi de ENREGISTREMENT SUIVANT
    4. LIBÉRER ENREGISTREMENT suivi de ALLER À DERNIER ENREGISTREMENT
    5. LIBÉRER ENREGISTREMENT suivi de CHERCHER PAR ATTRIBUT
    6. LIBÉRER ENREGISTREMENT suivi de CHERCHER PAR EXEMPLE
    7. LIBÉRER ENREGISTREMENT suivi de CHERCHER PAR SQL
    8. LIBÉRER ENREGISTREMENT suivi de CHERCHER PAR TABLEAU
    9. LIBÉRER ENREGISTREMENT suivi de CHERCHER PAR TABLEAU DANS SÉLECTION
3. Détections du pattern : TRIER suivi de VALEURS DISTINCTES

.

Version 1.0.7 du 15/09/2017

1. Détection des tris séquentiels
2. Corrections de bugs d'interface
3. Correction d'un bug sur la commande date du jour quand il y en a plusieurs sur une ligne
4. Correction d'un bug sur les lignes scindées

.

Version 1.0.6 du 11/09/2017

1. Détection de la commande "position" sans utilisation de "*"
2. Détection de la commande "remplacer chaine" sans utilisation de "*"

3. Typage complet des variables des méthodes ajoutées dans la base hôte
4. Ajout d'une recherche des méthodes comportant certaines chaines
5. Ajout d'une recherche des méthodes par regex
6. Création d'une nouvelle méthode dans vos bases permettant la récupération des cycles d'exécution d'un objet dans la base hôte

.

Version 1.0.4 du 07/09/2017

1. Mise en place de la synthèse des patterns
2. Corrections mineures

.

Version 1.0.0 du 01/09/2017

La première version complète mais on a encore + de 100 idées d'améliorations

.

Version 0.9.5 du 30/07/2017

1. Réduction très importante de la durée d'analyse
2. Ajout dans les metrics de la durée de l'analyse
3. Ajout d'une fonction de copier dans les listes permettant de copier au choix :
    1. La liste entière
    2. Les lignes sélectionnées
4. Correction de bugs sur les analyses en anglais
5. Correction des insertions des méthodes d'analyse en anglais
6. Suppression de message en cours d'analyse et ajout d'un écran récapitulatif en fin d'analyse

.

Version 0.7.2

1. Modification pour améliorer la "résistance" aux sous-tables
2. Correction d'un bug dans le remplacement de certains "tableau alpha"
3. Bug sur clic dans la liste des commandes sur une ligne vide provoque une erreur
4. Correction du déplacement du bouton d'action dans la gestion des images
5. Correction d'un bug lors des remplacements des "inactiver bouton" qui remplaçait par un objet fixer activation a vrai au lieu de le mettre à faux
6. Les ensembles définit via les pointeurs ne sont plus considérés comme erreur
7. Ajout de la commande "Modifie" dans les commandes obsolètes

.

Version 0.7.1

1. Bug fix
    1. Problème d'affichage après des tris dans différents écrans
    2. Problème avec des composant ayant des méthodes qui ne font parties d'aucun thème
2. Ajout d'un pattern sélection vers tableau après un trier
3. Gestion des lignes inactivées avec un si (faux)
4. Suppression de la fonction des recherches de textes qui sera développée pour la première release (1.1 prévue en août 2017)

2. Informations diverses concernant le composant

1. Comment installe-t-on le composant ?

Le composant Blue Code Analysis Helper s’installe simplement en copiant le dossier du composant dans votre base.

Cependant pour pouvoir faire une analyse compléte de votre base le composant va devoir créer des méthodes dans votre base.

Ces méthodes sont au nombre de 4 (à la date du 30 août 2017) et sont préfixée par 'ANACAH', pour pouvoir faire cela le composant a besoin d'un réglage particulier dans la base hôte.

- Allez dans les propriétés de la base
- Allez sur l'onglet sécurité
- Cocher "Exécuter la méthode "Sur événement base hôte" des composants
- Relancer la base

Les méthodes installées par le composant sont les suivantes (au 25 janvier 2019)

2. Comment démarre-t-on le composant ?

Créer une méthode comportant la Ligne de code suivante :

ANA_Start ("XX6XXCXX9XXBXX4XX5XX0XX7XX8XX6XX")

Le composant s'ouvre, cela peut prendre quelques secondes en raison du contrôle du numéro de licence sur nos serveurs

Si le numéro de licence est valide vous aurez accès à l'ensemble des fonctionnalités, dans le cas contraire, vous n'aurez accès qu'aux écrans de metrics et de synthèse

3. Les préférences
4. Le code inactif c'est quoi ?

Le code inactif c'est le code qui est

1. Dans un si (faux)
2. Dans un si (1=2)
3. Dans un si (2=1)

Ce code n'est pas analysé par le composant pour les recherches, tris, etc qu'il pourrait comporter.

5. Description succincte du composant

Le Code Analysis Helper a été conçu pour répondre aux problématiques suivantes :

1. Je dois migrer une base d'une version ancienne de 4D vers une version récente.
2. Mon applicatif est lent, y-a-t-il des recommandations pour agir rapidement ?
3. J'ai un programme sur lequel j'ai moins investi depuis quelques années, par où commencer pour supprimer les archaïsmes ?
4. Je dois reprendre un programme que je ne connais pas du tout, dans quel état technique est ce programme sur lequel je vais devoir intervenir ?
5. Je dois m'affranchir de la bibliothèque d'image
6. Je dois supprimer les liens automatiques de ma structure
7. ...

Pour tout cela Code Analysis Helper va vous aider à répondre aux questions suivantes :

1. Dans quoi est-ce que je mets les pieds ?
2. Quelles sont les tâches à mener et à déléguer ?
3. Quelles sont les méthodes à risques qui doivent être inspectées ?

4. Combien de fois est ou sont utilisé(es) la(es)commande(s) ?
5. Où sont utilisés des Plugins et quelle partie du plugin est utilisée ?
6. Où sont utilisées les images de la bibliothèque ?

6. Palette de l'inspecteur de méthode

Cette fonction qui apparaît en version 2.0 permet en cours de programmation de :

1. Voir les anomalies concernant votre méthode
    1. Pattern
    2. Méthodes ouvrantes fermantes
    3. Commandes à surveiller
    4. ...
2. Voir les informations concernant votre méthode
    1. Méthodes appelées
    2. Formulaires utilisés
    3. Plug in
    4. Query
    5. Sort
    6. Visualisation et modification des attributs de la méthode depuis la palette
    7. ...
3. Parameters
    1. Déclaration des paramètres
    2. Utilisation des paramètres
    3. Utilisation directe des paramètres
    4. Visualisation et modification des commentaires de la méthode
    5. ...

7. Informations Techniques

Comment est calculée la criticité

La criticité est la somme des éléments suivants

- Nombre d'appelant / 5
- Nombre de méthodes appelées / 5
- Nombre de ligne de la méthode / 100
- Nombre de tables impactées par la méthode
- Nombre de jours d'ancienneté de la méthode / 180
- Niveau de complexité de la méthode
- +10 si la méthode est récursive

Nous sommes ouverts à une discussion sur l'ajustement de cette formule, sur un paramétrage de cette formule ou même sur un calcul complètement différent si vous avez des idées.

Comment inactiver une alerte

8. Beautification du code

Cette fonction remet en ordre l'apparence du code. Elle supprime les doubles sauts de ligne Elle supprime les \ suivi de retour chariot inutiles (4D en insérait lors du déplacement de méthode dans certaines version) Le code suivant :
OB FIXER($objet;"attribut1";15;"attribut2";!00/00/0000!;"attribut3";?00:00:00?)
Sera transformé dans le code suivant
OB FIXER($objet;\ "attribut1";15;\ "attribut2";!00/00/0000!;\

"attribut3";?00:00:00?)

De manière à en faciliter la lecture.

Les commandes 4D traitées sont les suivantes :

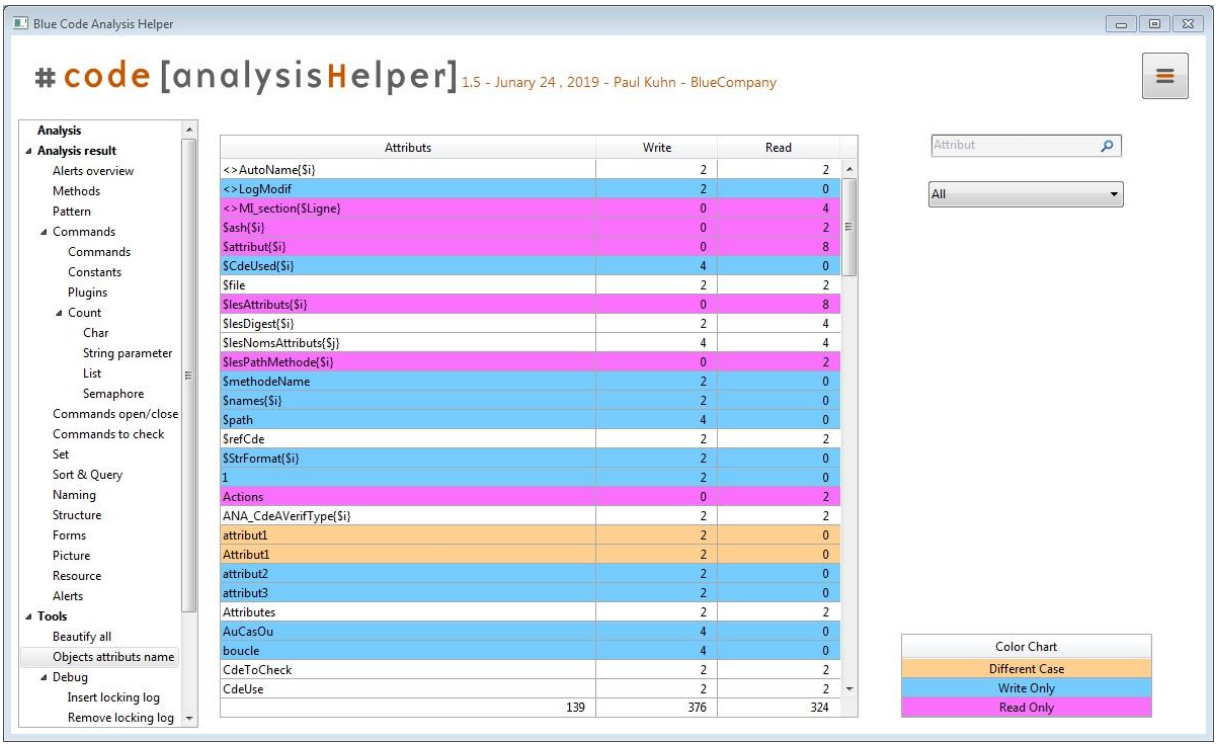
5. Sélection vers tableau
6. Sélection limitée vers tableau
7. Tableau vers sélection
8. Trier tableau

6. ob fixer
7. Créer objet
8. Créer collection
9. Créer collection partagée
10. Tableau vers collection

3. ST fixer attributs
4. ST lire attributs

2. Exécuter méthode

9. Génération de méthodes de barres de menus
10. Contrôle des noms d'attributs



11. Installation et retrait des méthodes de contrôle de verrouillage des enregistrements

L'objectif de cet outil est de vous permettre de déterminé les verrouillages d'enregistrements mal gérés dans une base 4D.

Pour ce faire le CAH va installer avant ou après l'appel de commandes enregistrant des données une méthode qui va contrôler si l'enregistrement c'est bien déroulé. Si ce n'est pas le cas la méthode va inscrire dans le log (LockingRecordLog.txt) qui sera créer dans le dossier des logs de la base l'heure et le problème rencontré.

Les commandes suivies sont les suivantes :

1. Appliquer à sélection
2. Stocker enregistrement
3. Supprimer enregistrement
4. Supprimer sélection
5. Tableau vers sélection
6. JSON vers sélection
7. Vider table

Pour utiliser insertion ou le retrait, vous devez avoir effectué l'analyse de la structure au préalable.

Si vous demandez au composant d'insérer à nouveau le code, il n'ajoutera bien sur le code que pour les nouveaux appels aux commandes. Dans le cas où vous avez beaucoup modifié votre code, vous pouvez demander une nouvelle insertion, cela mettra les N° de ligne à jour.

Le retrait retire TOUS les appels à la méthode de contrôle.

Exemple de log :

```
2018-12-15T13:09:05Z : Création du log
2018-12-15T13:09:05Z : [Société] LockedSet (3 records) - Method : Test_Lock (Line 11)
2018-12-15T13:09:05Z : [Société] Record Lock - Method : Test_Lock (Line 16)
2018-12-15T13:09:05Z : [Société] Record Lock - Method : Test_Lock (Line 16)
2018-12-15T13:09:05Z : [Société] Record Lock - Method : Test_Lock (Line 16)
```

Le code avant les insertions de contrôle

```
TOUT SÉLECTIONNER([Société])

APPLIQUER À SÉLECTION([Société];[Société]Nom:=[Société]Nom)

DÉBUT SELECTION([Société])
Tant que (Non(Fin de sélection([Société])))
	[Société]Nom:=[Société]Nom

	STOCKER ENREGISTREMENT([Société])
	ENREGISTREMENT SUIVANT([Société])
Fin tant que
```

Le code après les insertions de contrôle

```
TOUT SÉLECTIONNER([Société])

APPLIQUER À SÉLECTION([Société];[Société]Nom:=[Société]Nom)
ANACAH_DebugLockingRecord ("Selection";"Test_Lock (Line 11)";->[Société])  // insert by CAH 2018-12-04
```

```
DÉBUT SELECTION([Société])
Tant que (Non(Fin de sélection([Société])))
	[Société]Nom:=[Société]Nom

	STOCKER ENREGISTREMENT([Société])
	ANACAH_DebugLockingRecord ("record";"Test_Lock (Line 16)";->[Société])  // insert by CAH 2018-12-04
	ENREGISTREMENT SUIVANT([Société])
Fin tant que
```

---

le CAH vous permet bien sur de retirer tout ce code de contrôle.

Il vous permet également d'obtenir des statistiques sur les enregistrements dans votre base.

3. Les modules du composant

Pour accéder aux différents écrans il vous suffira de cliquer sur l'onglet correspondant, vous devrez bien sûr avoir effectué au préalable l'analyse de votre structure.

1. Analisys

    1. La liste des metrics
    2. Les outils

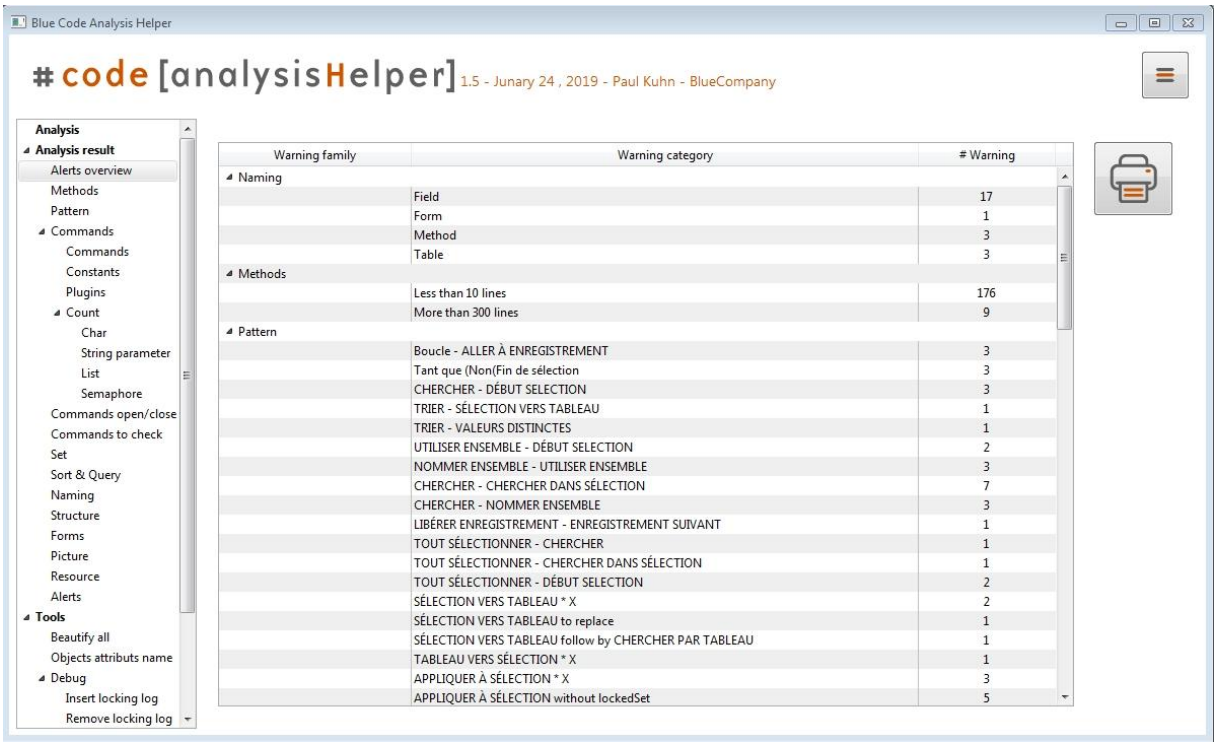
        1. Outils pour barres de menus
        2. Outils pour pop menu
        3. Outil Write
        4. Outil pour directive de compilations

    1. Les préférences
    2. Les recherches

        1. D'une chaine
        2. Par regex

        1. Sauver une analyse
        2. Charger une analyse

2. Alerts overview



3. Methods

Informations & criticité

Attributes

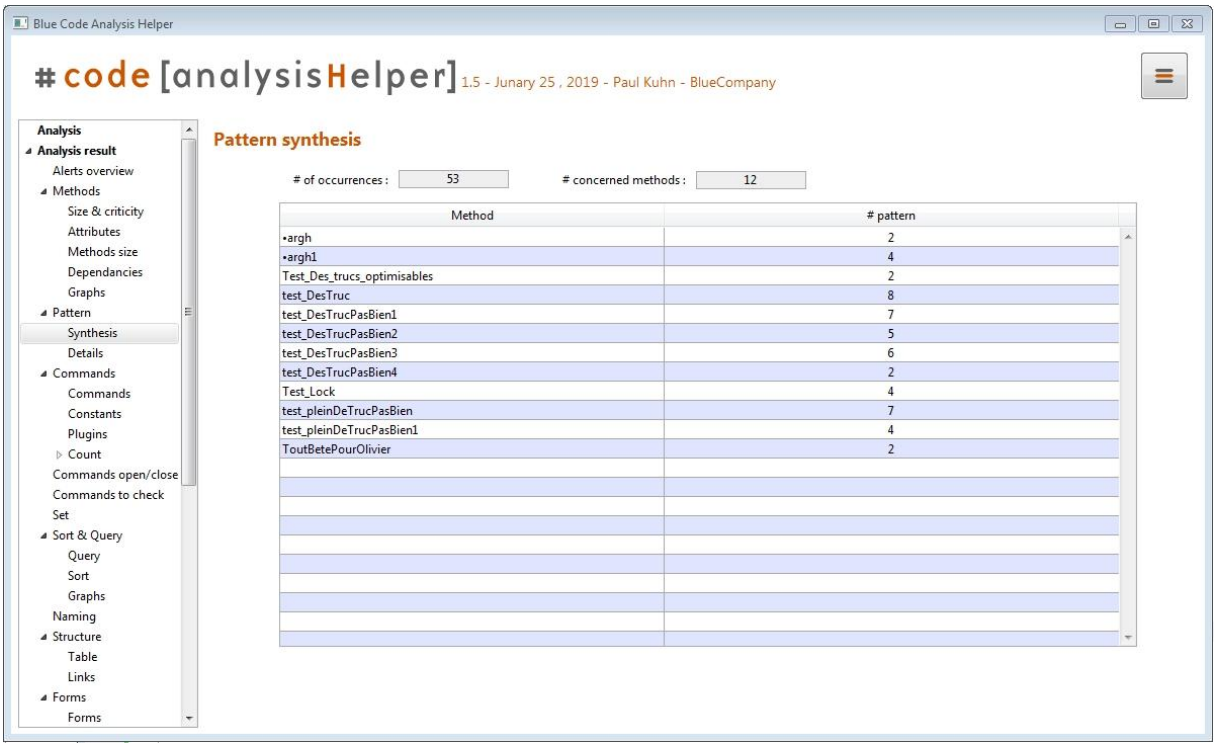
Grandes et petites méthodes

Dépendances

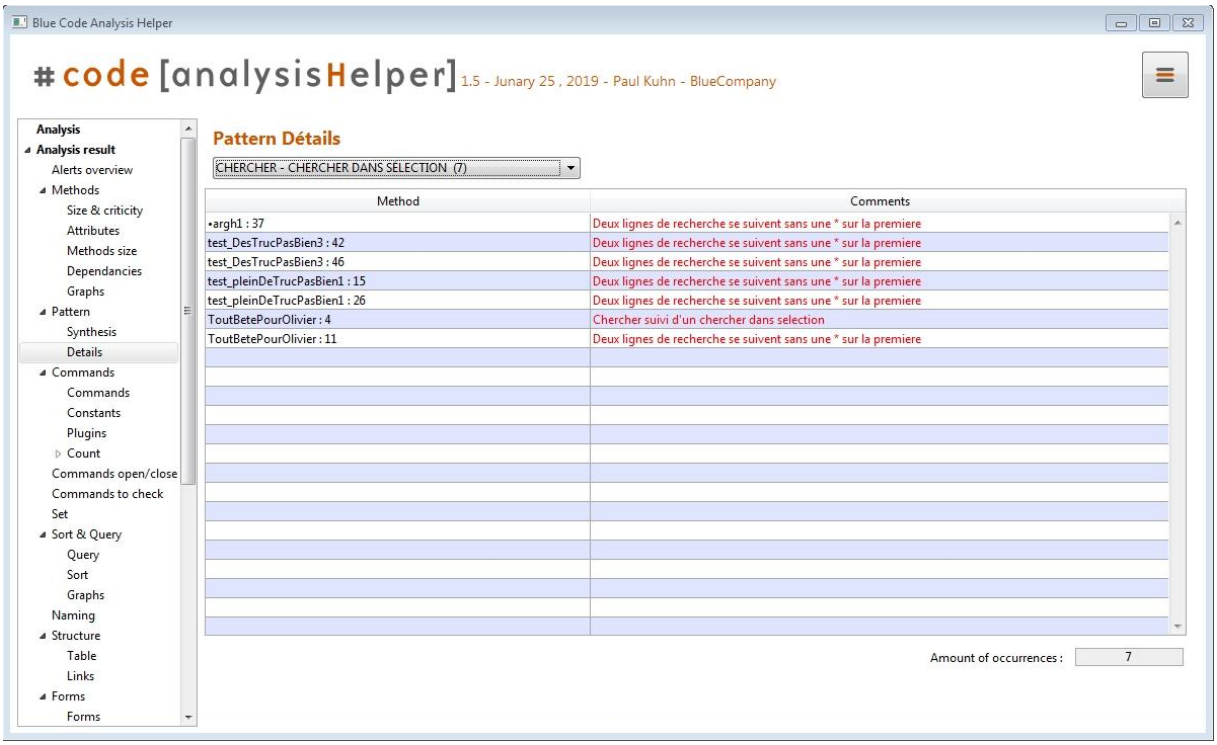
Graphes

4. Pattern

Synthesis


Blue Code Analysis Helper
#code [analysisHelper] 1.5 - Junary 25 , 2019 - Paul Kuhn - BlueCompany
Analysis
Analysis result
Alerts overview
Methods
Size & criticity
Attributes
Methods size
Dependancies
Graphs
Pattern
Synthesis
Details
Commands
Commands
Constants
Plugins
Count
Commands open/close
Commands to check
Set
Sort & Query
Query
Sort
Graphs
Naming
Structure
Table
Links
Forms
Forms
Pattern synthesis
# of occurrences : 53
concerned methods : 12
Method
pattern
•argh 2
•argh1 4
Test_Des_trucs_optimisables 2
test_DesTruc 8
test_DesTrucPasBien1 7
test_DesTrucPasBien2 5
test_DesTrucPasBien3 6
test_DesTrucPasBien4 2
Test_Lock 4
test_pleinDeTrucPasBien 7
test_pleinDeTrucPasBien1 4
ToutBetePourOlivier 2

Details


Blue Code Analysis Helper
#code [analysisHelper] 1.5 - Junary 25 , 2019 - Paul Kuhn - BlueCompany
Analysis
Analysis result
Alerts overview
Methods
Size & criticity
Attributes
Methods size
Dependancies
Graphs
Pattern
Synthesis
Details
Commands
Commands
Constants
Plugins
Count
Commands open/close
Commands to check
Set
Sort & Query
Query
Sort
Graphs
Naming
Structure
Table
Links
Forms
Forms
Pattern Détails
CHERCHER - CHERCHER DANS SÉLECTION (7)
Method
Comments
•argh1 : 37 Deux lignes de recherche se suivent sans une * sur la premiere
test_DesTrucPasBien3 : 42 Deux lignes de recherche se suivent sans une * sur la premiere
test_DesTrucPasBien3 : 46 Deux lignes de recherche se suivent sans une * sur la premiere
test_pleinDeTrucPasBien1 : 15 Deux lignes de recherche se suivent sans une * sur la premiere
test_pleinDeTrucPasBien1 : 26 Deux lignes de recherche se suivent sans une * sur la premiere
ToutBetePourOlivier : 4 Chercher suivi d'un chercher dans selection
ToutBetePourOlivier : 11 Deux lignes de recherche se suivent sans une * sur la premiere
Amount of occurrences : 7

5. Commands

Commands

constants

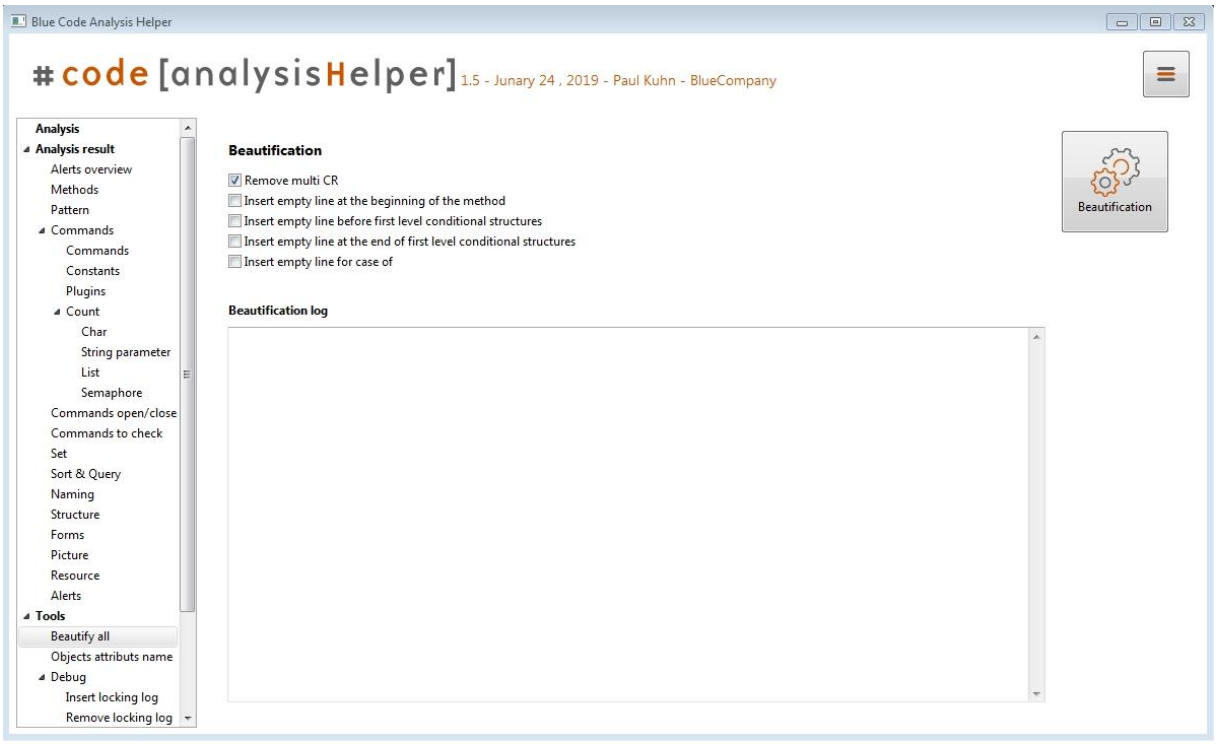
Plugins

Count

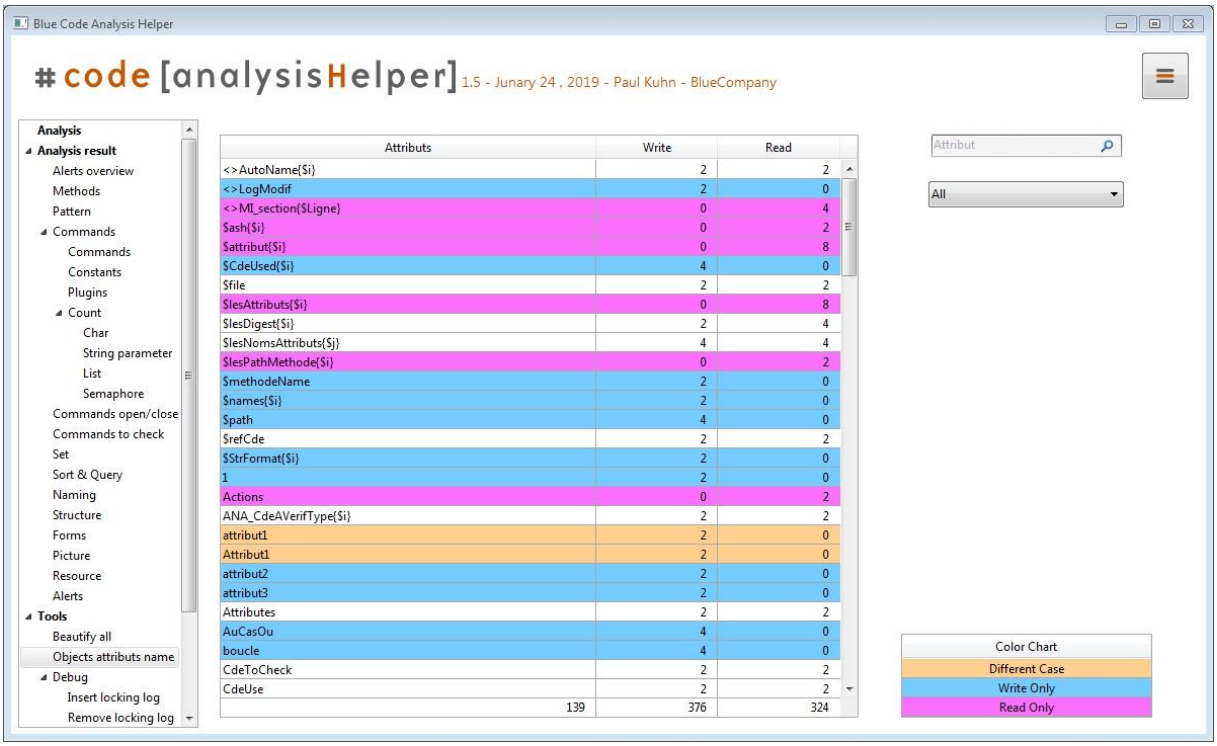
- Char
- String parameter
- List
- Semaphore

6. Commands open/close
7. Commands to check
8. Set

Les ensembles de même nom sur des tables différentes peuvent suivant les cas et les versions faire crasher 4D (Retour directe au bureau).

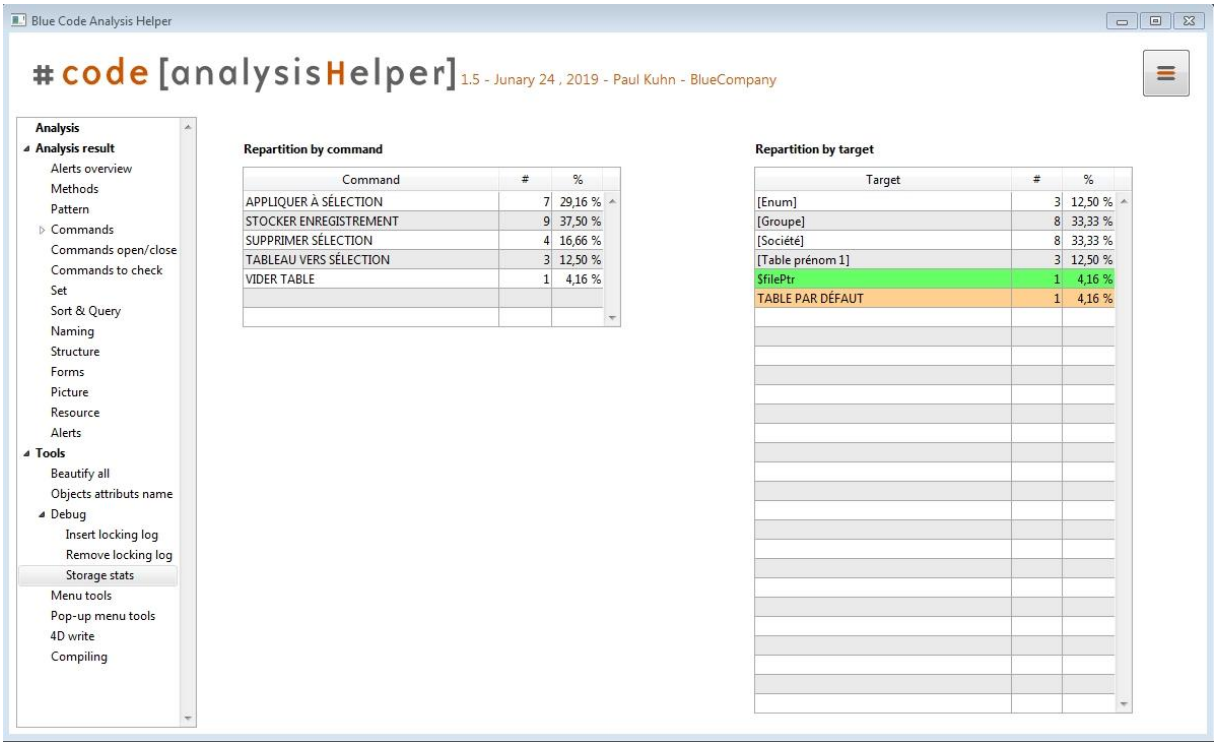


Objects attributs name



Outils de debug

- Insert locking log
- Remove locking Log
- Storage statistics



Cet écran vous permet de visualiser la manière dont sont stockées les données dans la base sur laquelle vous êtes en train de travailler.

Vous avez les statistiques par commandes et par tables.

Si vous constatez l'apparition de la notion de "Table par défaut", nous vous recommandons de modifier votre code afin de plus utiliser cette commande. En effet table par défaut provoque plusieurs problèmes :

1. Vous ne pouvez plus utiliser la commande dialogue avec des formulaires projet
2. Le code devient plus difficile à lire et à maintenir
3. Certaines commandes n'en tiennent pas ou plus compte... (cf notes de compatibilités)

Menu tools

Pop-up menu tools

4D Write

Compiling

Depuis toujours 4D tolère les erreurs dans les commandes de typage.

Les commandes suivantes sont tolérées aussi bien à l'exécution, à la compilation et à l'exécution en compilé.

Depuis la version 16.xx ou 16.Rx 4D tolère toujours ces écritures à l'exécution qu'à la compilation mais génère des erreurs à l'exécution en compilé.

Le composant vous détecte donc les endroits comportant des problèmes dans les directives de typage et vous propose de la corriger pour vous.

4. La palette de l'explorateur de méthode

Cette fonction est disponible à partir de la version 2.0 du CAH qui nécessite la version 16 de 4D. .

1. Les warnings
2. Les attributs de méthode
3. Les comptages
4. Les commentaires
5. Le typage