ENTERPRISE ARCHITECT

Série de Guides d'Utilisateur

Construire et Déboguer

Author: Sparx Systems
Date: 23/11/2023
Version: 16.1

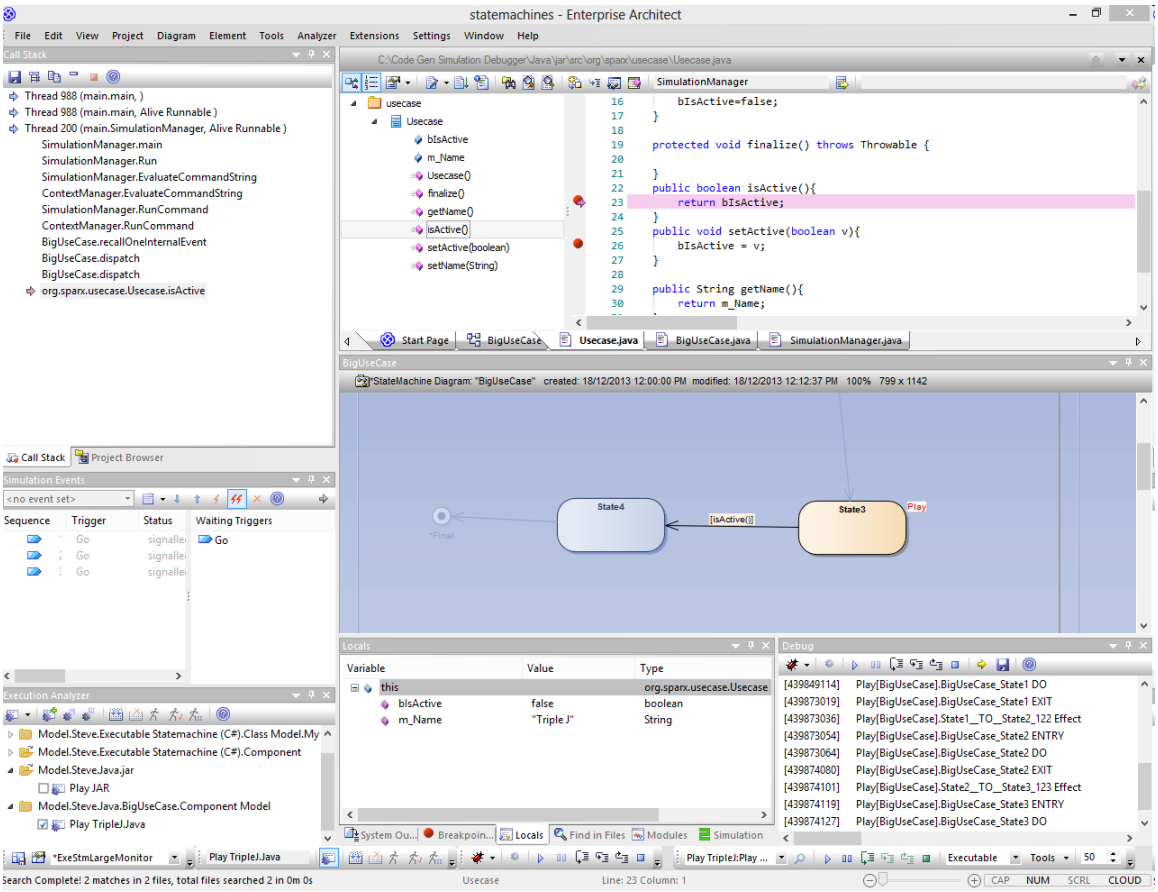
CRÉÉ AVEC ENTERPRISE ARCHITECT

Table des Matières

Construire et Déboguer 4
Prestations de service 6
Fenêtre Services de l'analyseur 10
Débogage 12
Exécuter le Débogueur 14
Gestion des Point d'Arrêt et Marqueurs 17
Définition Points d'Arrêt du code 20
Instructions de trace 21
Interrompre lorsqu'une Variable Change de Valeur 23
Trace lorsque Variable Change de Valeur 26
Détection des Opérations d'Adresse Mémoire 27
Point d'Arrêt Propriétés 29
Défaut de lier Point d'Arrêt 31
Déboguer une Application en Cours 32
Voir les Variables Locales 33
Voir le Contenu de Longues Chaînes 36
Variables Vue Déboguer dans Éditeurs de Code 38
Instantanés variables 39
Points d'Action 41
Voir Variables dans d'Autres Portées 45
Voir Éléments du Réseau 46
Voir la Pile d'Appel 47
Créer Diagramme de Séquence de Pile d'Appel 49
Inspecter Mémoire de Processus 51
Afficher Modules Chargés 52
Traiter Exceptions à Première Chance 53
Débogueur juste à temps 55
Créer une application 56
Localiser les erreurs Compilateur dans le code 57
Scripts d'Analyseur 58
La fenêtre de la file d'attente des tâches 60
Script Code Miner 64
Script de services 67
Fusionner le script 68
Scripts Enregistrement 69
Déployer le script 71
Exécuter Script 73
Script Débogage 74
Exigences Spécifiques au Système d'Exploitation 75
Systèmes d'Exploitation compatibles UAC 76
Débogage WINE 77
Java 79
Configuration générale pour Java 80
Techniques avancées 82
Attacher à Virtual Machine 83
Applet Java Navigateur Internet 84
Travailler avec Serveurs Web Java 85

Serveur JBOSS 87
Serveur Apache Tomcat 88
Apache Tomcat Windows Service 89
.NET 90
Configuration générale pour .NET 91
Débogage d'une application non gérée 93
Déboguer COM Interop 94
Déboguer ASP .NET 95
Le Mono Débogueur 96
Configuration du débogage Linux 97
Windows de configuration de débogage 99
Le Débogueur PHP 101
Débogueur PHP - Exigences système 104
Liste de contrôle Débogueur PHP 105
Le Débogueur GNU (GDB) 107
Le Débogueur Android 109
Débogueur Java JDWP 112
Sortie Point de Trace 115
Établi établie 116
Microsoft C++ et Natif (C, VB) 117
Configuration générale 118
Symboles Déboguer 120
Sortie des Testpoints 121
Scripts de Test 123
Script Nettoyage 125
Créer Scripts 127
Éditeur de Script Analyseur 129
Gestion Scripts d'Analyseur 134

Construire et Déboguer



Enterprise Architect s'appuie sur ses capacités déjà exceptionnelles de génération de code, de création de diagrammes et de conception avec une suite complète d'outils pour créer, déboguer, visualiser, enregistrer, tester, profiler et autrement construire et vérifier des applications logicielles. L'ensemble d'outils est intimement lié aux capacités modélisation et de conception et fournit un moyen unique et pratique de construire un logiciel à partir d'un modèle et de maintenir le modèle et le code synchronisés.

Enterprise Architect vous aide à définir des « Scripts d'Analyseur » liés aux Paquetages Modèle qui décrivent comment une application sera compilée, quel débogueur utiliser et d'autres informations connexes telles que les commandes de simulation. Le script Analyzer est l'élément de configuration principal qui relie votre code aux fonctionnalités de création, de débogage, de test, de profilage et de déploiement Enterprise Architect .

Pour mesurer la compétence de l'ensemble d'outils, il convient de noter qu'Enterprise Enterprise Architect est en fait construit, débogué, profilé, testé et autrement construit entièrement dans l'environnement de développement Enterprise Architect . De nombreux outils de débogage avancés tels que Action Points ont été développés pour résoudre des problèmes inhérents à la construction d'applications logicielles volumineuses et complexes (telles que Enterprise Architect) et sont régulièrement utilisés quotidiennement par l'équipe de développement Sparx Systems .

Il est recommandé aux nouveaux utilisateurs de prendre le temps de bien comprendre l'utilisation des Scripts d'Analyseur et la manière dont ils lient le modèle au code, aux compilateurs et aux autres outils nécessaires à la construction de logiciels.

Intégration Modèle et Code

Modèle Driven Engineering est une approche moderne du développement de logiciels et promet une plus grande productivité et un code de meilleure qualité, ce qui permettra aux systèmes d'être commercialisés plus rapidement et avec moins de défauts. Ce qui rend cette approche convaincante est la possibilité pour l'architecture et la conception d'un système d'être décrites et maintenues dans un modèle, puis générées pour programmer du code et des schémas qui

peuvent être synchronisés et visualisés dans le modèle.

L'environnement de développement piloté Modèle (MDDE) d' Enterprise Architect supporte cette approche et fournit un ensemble d'outils flexibles pour augmenter la productivité et réduire les erreurs. Celles-ci incluent la possibilité de définir l'architecture et la conception des modèles, de générer du code à partir de ces modèles, de synchroniser le code avec les modèles et de maintenir le code dans des éditeurs de code sophistiqués. Le code source ou les binaires peuvent également être importés, et les utilisateurs peuvent enregistrer et documenter le code préexistant ou récemment développé. L'outil Analyzer Script vous aide à décrire comment créer, déboguer, tester et déployer une application.

Facilité	Description
Développement basé Modèle	Modèle Driven Development offre un cycle de développement plus robuste, accessible et plus rapide que les cycles traditionnels basés sur le codage. Un modèle bien construit, intimement lié aux capacités de création, exécuter , de débogage, de test et de déploiement du code source, fournit une architecture cible riche, facile à naviguer et à comprendre. La traçabilité, la liaison avec les cas d'utilisation, les composants et autres artefacts de modèle, ainsi que la possibilité d'enregistrer et de documenter facilement le code préexistant ou récemment développé, rendent l'environnement de développement d' Enterprise Architect particulièrement efficace. Enterprise Architect intègre des langages d'édition, de débogueur et modélisation intelligents conformes aux normes de l'industrie.

L'environnement de développement piloté Modèle (MDDE)	Le MDDE fournit des outils pour concevoir, visualiser, construire et déboguer une application :
	• Technologies et outils UML pour modéliser des logiciels
	• Outils de génération de code pour générer/effectuer de l'ingénierie inverse du code source
	• Outils pour importer du code source et des binaires
	• Éditeurs de code support différents langages de programmation
	• Intelli-sense pour faciliter le codage
	• Scripts d'analyseur qui permettent à un utilisateur de décrire comment créer, déboguer, tester et déployer l'application
	• Intégration avec des compilateurs tels que Java, .Net, Microsoft C++
	• Capacités de débogage pour Java, .NET , Microsoft C++ et autres
	• Capacités avancées de visualisation, d'enregistrement, d'inspection, de test et de profilage
	`pApp = new CBCGPAppointmentDemo`
	`(`
	2 of 2 CBCGPAppointmentDemo::CBCGPAppointmentDemo(**COleDateTime& dtStart**, COleDateTime& dtFinish, CString& strText, COLORREF clrBackground, COLORREF clrForeground, COLORREF clrDuration)
	`RGB (165, 222, 99),`
	`CLR_DEFAULT,`
	`RGB(128, 0, 128)`
	`);`

Prestations de service

Enterprise Architect fournit deux services pour faciliter l'exécution de scripts à distance et le débogage à distance. Les services prennent principalement support Enterprise Architect fonctionnant sous Linux pour permettre aux utilisateurs d' exécuter des scripts shell Linux natifs et de déboguer des programmes Linux. Le service Satellite supporte Scripts d'Analyseur tandis que le service Agent supporte le débogage.

Accéder

Ruban	Exécuter > Outils > Services

Le Service Satellite

Le service Satellite se charge d'exécuter Scripts d'Analyseur sur la machine sur laquelle il s'exécute. Cette fonctionnalité peut aider les utilisateurs de Linux à exécuter directement des programmes Linux natifs et des commandes shell, en contournant Wine. Le service peut être géré depuis le ruban, et il peut également être exécuter indépendamment depuis un terminal.

Le shell Linux

Le shell par défaut utilisé par Enterprise Architect est « bash ». Pour remplacer le Linux Shell utilisé par Enterprise Architect , ouvrez un terminal Linux, exécuter 'wine regedit ' et ajoutez une string valeur à cette clé de registre :

HKEY_CURRENT_USER\Software\ Sparx Systems \EA400\EA\Options

où:

- nom de la clé : "LINUX"
- valeur clé : *chemin*

et *path* est le chemin Linux vers le programme shell " /bin/bash", par exemple.

Autorisations

Sous Linux, vous devez vérifier que les programmes de service disposent des autorisations appropriées. Les programmes se trouvent dans le dossier d'installation Enterprise Architect , dans le sous-répertoire 'VEA/x86/linux'. Vérifiez que chacun des programmes de ce répertoire dispose de l'autorisation d'exécution définie pour le propriétaire.

Notes

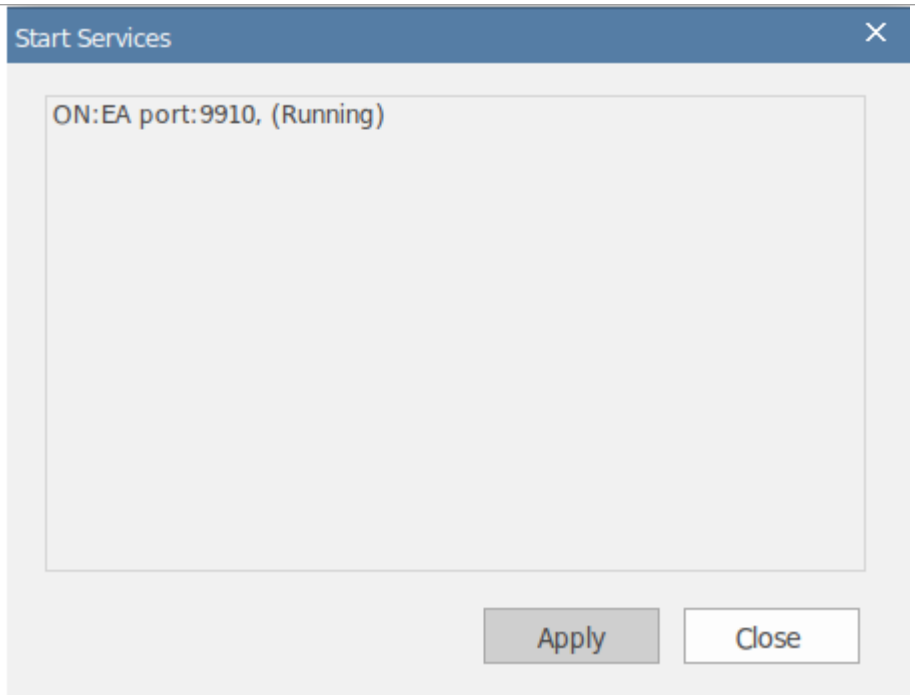
- Les services Satellite sont activés dans les éditions Unified et Ultimate d' Enterprise Architect

Le Service d'Agent

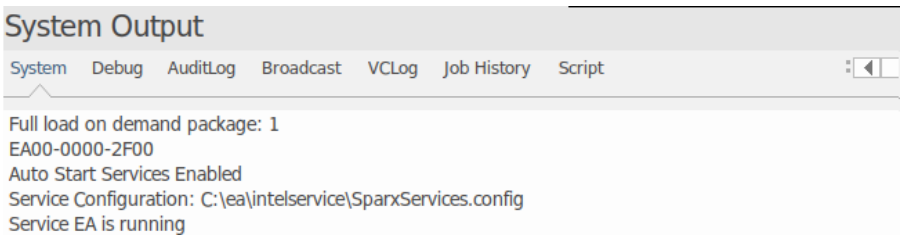
Le service Agent est responsable de la gestion des sessions de débogage pour le débogueur GDB d' Enterprise Architect . Le service permet aux utilisateurs Enterprise Architect de déboguer les programmes Linux. Le service peut être géré à partir du ruban. Il peut également être exécuter indépendamment d'un terminal.

Le menu des services

Option	Description
Statut Vue de tous les services	Affiche une Vue qui répertorie l'état de chaque service Enterprise Architect nommé dans le fichier de configuration et son état.
Service Satellite	
Démarrer	Démarre le service. Le service écoute sur le port satellite configuré dans n'importe quelle page des services de script Analyzer.
Arrêt	Arrête le service.
Test	Teste l'état du Service Satellite , qu'il soit en cours d'exécution ou non.
Service d'Agent	
Démarrer	Démarre le service. Le service écoute sur le port d'agent configuré dans une page des services de script Analyzer.
Arrêt	Arrête le service.
Test	Teste l'état du Service d'Agent , qu'il soit en cours d'exécution ou non.
Service Code Miner	
Démarrer	Cette option lit le fichier de configuration de service actuel et démarre les services configurés pour exécuter , et arrête l'exécution des services qui ne sont pas configurés pour exécuter . Un service est configuré si : 1. Il est nommé dans le fichier de configuration. 2. Il a le statut d'attribut : ON.

<table>
<tr><td></td><td> 
</td></tr>
<tr><td>Arrête tout</td><td>Cette option arrête tous les services en cours d'exécution.</td></tr>
<tr><td>Modifier le fichier de configuration</td><td>Cette option prompts à utiliser le fichier de configuration du service, puis ouvre ce fichier dans un éditeur de texte Enterprise Architect . Le système se souvient de l'endroit où se trouve le fichier.

```
31 # <number>   - digits
32 # ----------------------------------------------------------
33 #
34 {
35     name=project1,
36     status=ON,
37     lazyload=true,
38     port=9910,
39     allow=localhost,
40     network=local,
41     autoupdate=true,
42     show=true,
43     logoutput=c:\My Documents\project1.txt,
44     loglevel=information|warning|error,
45     database=c:\My Documents\project1\project1.cdb
46 }
47
```
</td></tr>
<tr><td>Démarrer automatique avec EA</td><td>Cette option démarre automatiquement les services ayant l'attribut « status:ON » à l'ouverture du modèle.



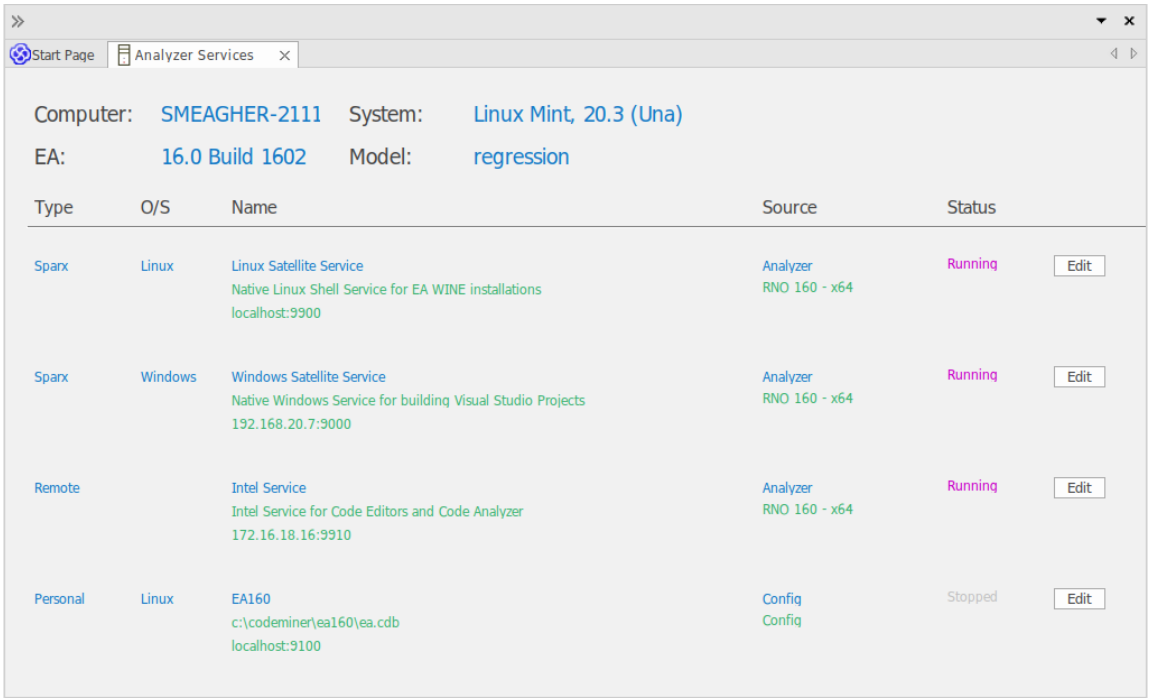
Les messages enregistrés dans la fenêtre Sortie système ici lorsque le modèle est</td></tr>
</table>

	ouvert indiquent que le service était déjà en cours d'exécution.
Arrêt automatique à la fermeture	Cette option arrête automatiquement l'exécution des services lorsque Enterprise Architect est fermé.

Fenêtre Services de l'analyseur

La fenêtre Analyzer Services affiche l'état de chacun des éléments suivants :

- Les services gérés par le script Analyzer actif (maintenus via l'option de ruban « Exécuter > Outils > Analyser »)
- Tous les services locaux gérés séparément à l'aide du fichier de configuration du service (disponible à partir de l'option de ruban « Exécuter > Outils > Services > Service Code Miner > Modifier le fichier de configuration »).



La fenêtre vous permet de voir en un coup d'œil quel service Intel vous utilisez, par exemple, ou que le service Windows que vous utilisez pour travailler dans Enterprise Architect est en cours d'exécution.

Toutes les données de la fenêtre sont en lecture seule, mais vous pouvez modifier le service en cliquant sur le bouton Modifier. Cela affiche la page 'Options privées - Services' de la fenêtre de l'Editeur Scripts d'Analyseur , que vous pouvez mettre à jour selon vos besoins.

Accéder

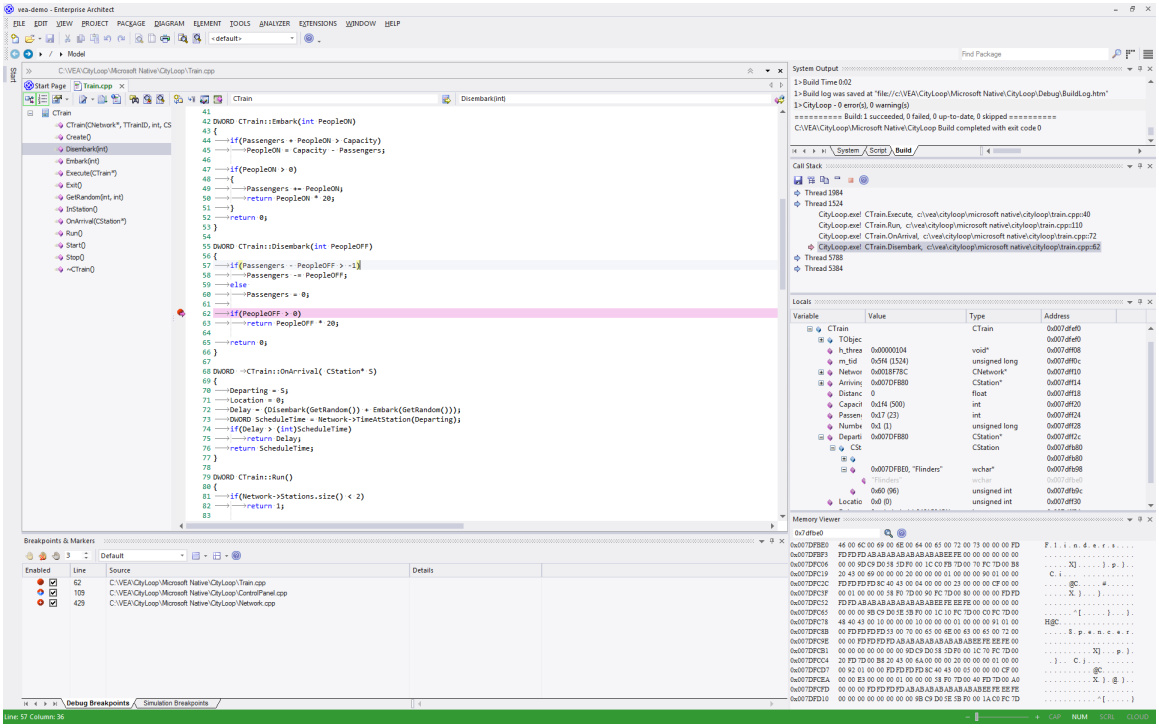
Ruban	Exécuter > Outils > Services > Statut Vue de tous les services Démarrer > Toutes Windows > Exécuter > Analyser | Services d'analyse
Raccourcis Clavier	Alt+4 | Analyser | Services d'analyse

Champs de la fenêtre des services d'analyseur

Champ	Description
Ordinateur	Le poste de travail sur lequel les services sont exécutés.

Système	Le système d'exploitation sous lequel le poste de travail est exécuté.
EA	Le numéro de version et le numéro de build de la version d' Enterprise Architect que vous utilisez.
Modèle	Le nom du modèle dans lequel vous travaillez actuellement.
Type	Le type de prestation. Les services répertoriés comme type « Sparx » sont des services Enterprise Architect .
Système d'exploitation	Le système d'exploitation sous lequel le service s'exécute.
Nom	Le nom et la description du service. • Le Linux Service Satellite est géré automatiquement par Enterprise Architect lorsque l'application s'exécute sous Linux sous WINE • Le Windows Service Satellite est un service facultatif généralement utilisé dans les installations WINE pour aider à créer des projets Visual Studio sur une machine Windows distante telle qu'une VM (Virtual Machine) ; le service fonctionne uniquement sous Windows et est géré localement à l'aide des services Windows sur la machine elle-même
Source	Le groupe de scripts ou le fichier de configuration définissant le service.
Statut	Le statut du service. Lorsque vous ouvrez la fenêtre pour la première fois, chaque service a initialement le statut « Test » lorsque le système évalue le service. Le statut passe alors à la valeur appropriée, telle que « En cours d'exécution » ou « Arrêté ».
Modifier	Cliquez sur ce bouton pour afficher ou modifier la définition du service.

Débogage



Enterprise Architect est plus qu'un outil de dessin : il fournit également toutes fonctionnalité que vous pouvez attendre d'un IDE. Des environnements et outils de débogage complets pour de nombreuses plates-formes majeures sont mis à disposition. L'intégration de la capacité de débogage au sein de l'outil modélisation permet au code d'être développé, construit et géré par ses auteurs. Travailler et collaborer dans un modèle intégré a permis de faire en sorte que les actions comptent et que chaque action soit responsable d'une manière qui n'est tout simplement pas possible avec d'autres chaînes d'outils.

Fonctionnalités

Vitesse

Les débogueurs d' Enterprise Architect sont rapides ! Parcourir les programmes ne prendra pas toute la journée.

Le programme Enregistrement peut être exécuté sans pas manuel.

Support

- C++, C et Visual Basic
- Microsoft .NET , ASP.NET WCF
- Java, utilisant le transport de socket (JDWP) ou le modèle en mémoire (JVMIT)
- Android sur un émulateur ou un appareil
- JavaScript , VBScript et JScript
- Scripts PHP sur les serveurs Web Apache
- Processus Linux GDB distants utilisant Enterprise Architect sous Windows
- Simulation - débogage des simulations en UML et BPMN
- Statemachines Exécutables - déboguer une Statemachine en cours d'exécution

Isolement

Les débogueurs fonctionnent hors processus d' Enterprise Architect , l'isolant des effets secondaires.

Efficacité

Le démarrage et l'arrêt du débogueur sont rapides et indolores. Cela ne vous retient pas. Conçu pour être une UI réactive, le fil principal UI est isolé des tâches qui ne relèvent pas de sa responsabilité.

Productivité

Passez de modélisation aux exigences, de l'envoi d'une demande de modification au suivi des modifications de code dans un modèle partagé au sein d'une organisation, en passant par le profilage des modifications de code récentes. Le tout dans un seul outil.

Notes

- Les fonctionnalités de débogage et d'enregistrement de l' Analyseur d'Exécution Visuelle ne sont pas disponibles pour la plateforme serveur Java 'Weblogic' d'Oracle.

Exécuter le Débogueur

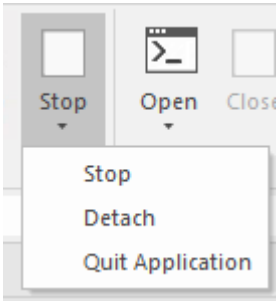
Enterprise Architect propose plusieurs façons de démarrer et de contrôler une session de débogage. Il y a la fenêtre principale Déboguer , ainsi qu'une barre d'outils Déboguer et le panneau ' Exécuter ' dans le ruban 'Exécuter'. Il est toujours préférable d'afficher la fenêtre Déboguer chaque fois que vous exécutez une session de débogage, car c'est là que toutes les sorties de débogage sont capturées.

Accéder

Ruban	Exécuter > Exécuter > Démarrer Exécuter > Outils > Débogueur > Démarrer le débogage
Raccourcis Clavier	Alt+8 (affiche la fenêtre Déboguer) F6 (commence l'exécution de l'application en cours de débogage)
Barre d'outils	Explorer > Portails > Afficher la barre d'outils > Déboguer

Utiliser la fenêtre Déboguer

Action	Détail
Démarrer le Débogueur	Lorsqu'un script Analyzer a été configuré pour support le débogage, vous pouvez démarrer le débogueur de ces manières : • Dans le ruban, sélectionnez 'Exécuter > Exécuter > Démarrer > Exécuter '. • Depuis le ruban, sélectionnez 'Exécuter > Outils > Débogueur > Démarrer le débogage'. • Sur la barre d'outils ' Déboguer ', cliquez sur le bouton , ou • Appuyez sur F6 Vous pouvez également lancer le débogueur pour n'importe quel script via son menu contextuel dans la « Fenêtre de script de l'analyseur », ou appuyer sur Maj+F12. Si vous n'avez pas de script Analyzer, il est toujours possible de déboguer une application en cours d'exécution en l'attachant directement à ce processus : • Dans le ruban, sélectionnez « Exécuter > Outils > Débogueur > Attacher au processus », ou • Sur la barre d'outils ' Déboguer ', cliquez sur le bouton (Attach) et choisissez manuellement la plateforme de débogage
Pause/Reprise du débogage	Vous pouvez suspendre une session de débogage ou reprendre la session après une pause de ces manières : • Dans le ruban, sélectionnez « Exécuter > Exécuter > Pause ». • Sur la barre d'outils ' Déboguer ', cliquez sur le bouton

Stop au Débogueur	Le débogueur se termine normalement lorsque le processus de débogage en cours se termine ; cependant, certaines applications et services (tels que Java Virtual Machine) peuvent nécessiter l'arrêt manuel du débogueur. Pour arrêter le débogage, soit : • Sur la barre d'outils ' Déboguer ', cliquez sur le bouton (Stop) • Appuyez sur Ctrl+Alt+F6 • Sélectionnez la flèche déroulante sur l'option du ruban "Exécuter > Exécuter > Arrêter". L'option du ruban affiche un menu court proposant trois façons de terminer le débogage de l'application.  • Stop - arrête le Débogueur et arrête le processus en cours de débogage (valeur par défaut lorsque vous cliquez simplement sur l'icône du ruban) • Détacher - arrête le Débogueur mais laisse le processus en cours d'exécution • Quitter l'application - arrête le Débogueur et publie un message WM_QUIT dans la fenêtre principale du processus, s'il en a une
Enjamber les lignes de code	Pour passer à la ligne de code suivante : • Dans le ruban, sélectionnez « Exécuter > Exécuter > Step Over », ou • Dans la barre d'outils ' Déboguer ', cliquez sur le bouton (Step Over), ou • Appuyez sur Alt+F6
Entrez dans Appels de Fonction	Pour accéder à un appel de fonction : • Dans le ruban, sélectionnez 'Execute > Exécuter > Entrer ', ou • Sur la barre d'outils ' Déboguer ', cliquez sur le bouton (Entrer), ou • Appuyez sur Maj+F6 Si aucune source n'est disponible pour la fonction cible alors le Débogueur revient immédiatement à l'appelant.
Sortir les fonctions	Pour quitter une fonction : • Dans le ruban, sélectionnez 'Exécuter > Exécuter > Sortir '. • Sur la barre d'outils ' Déboguer ', cliquez sur le bouton (Sortir), ou • Appuyez sur Ctrl+F6 Si le débogueur entre dans une fonction sans code source, il continuera à sortir jusqu'à ce qu'un point contenant du code source soit trouvé.
Afficher le point d'exécution	Pendant que le Débogueur est en pause, pour revenir au fichier source et à la ligne de code que le Débogueur s'apprête à exécuter : • Dans le ruban, sélectionnez 'Exécuter > Exécuter > Démarrer > Afficher le point d'exécution'.

	• Sur la barre d'outils ' Déboguer ', cliquez sur le bouton (Afficher le point d'exécution). La ligne appropriée est mise en évidence, avec une flèche rose dans la marge gauche de l'écran.
Sortir	Lors d'une session de débogage, des messages s'affichent dans la fenêtre Déboguer détaillant : • Démarrage de la séance • Fin de séance • Des exceptions • les erreurs • Messages de trace, tels que ceux générés à l'aide de Java System.out ou .NET System.Diagnostics. Déboguer Si vous double-cliquez sur un message de débogage, soit : • Une fenêtre contextuelle s'affiche avec un texte de message plus complet, ou • S'il y a eu une fuite de mémoire, le fichier est affiché au point où l'erreur s'est produite
Enregistrer la sortie (et la sortie Effacer)	Vous pouvez enregistrer l'intégralité du contenu de la sortie Déboguer dans un fichier .txt externe ou enregistrer les lignes sélectionnées de la sortie dans le presse-papiers Enterprise Architect . Pour enregistrer toute la sortie dans un fichier, cliquez sur le bouton (Enregistrer la sortie dans un fichier). Pour enregistrer les lignes sélectionnées dans le presse-papier, cliquez-droit sur la sélection et sélectionnez l'option 'Copier la sélection dans le presse-papier'. Lorsque vous avez sauvegardé la sortie ou que vous ne souhaitez plus l'afficher, cliquez-droit sur la sortie actuelle et sélectionnez l'option ' Effacer Results'.
Passer au profileur	Si vous exécutez une session de débogage sur du code, vous pouvez arrêter la session de débogage et passer immédiatement à une session de profilage. Pour passer du Débogueur au Profiler : • Dans le ruban, sélectionnez « Exécuter > Outils > Débogueur > Passer au profileur ». • Sur la fenêtre Déboguer , cliquez sur l'icône ' | Basculer vers l'option Profiler, ou • Sur la barre d'outils Déboguer , cliquez sur l'icône ' | Passer à l'option Profiler Le profileur s'attache au processus en cours d'exécution. Cette facilité n'est pas disponible pour les débogueurs Java.

Gestion des Point d'Arrêt et Marqueurs

Points d'Arrêt fonctionnent dans Enterprise Architect de la même manière que dans n'importe quel autre Débogueur . Les marqueurs sont similaires aux points d'arrêt, mais dans Enterprise Architect , ils ont des pouvoirs spéciaux. En termes simples, les marqueurs effectuent des actions - telles que l'enregistrement de l'exécution et de l'analyse - que les points d'arrêt ne font pas. L'action d'un point d'arrêt est toujours d'arrêter le programme.

Vous définissez n'importe quel marqueur ou point d'arrêt dans l'éditeur de code source, où ils sont visibles dans la marge de gauche. Cliquer dans cette marge ajoutera un point d'arrêt sur cette ligne. Points d'Arrêt et les marqueurs sont interchangeables - vous pouvez transformer un point d'arrêt en marqueur et vice versa, en utilisant sa dialogue ' Propriétés '. Vous pouvez rapidement visualiser et modifier les propriétés d'un point d'arrêt ou d'un marqueur en utilisant Ctrl+clic soit sur son icône dans la marge de l'éditeur, soit dans la fenêtre Points d'Arrêt et marqueurs.

Points d'Arrêt sont maintenus par ensembles. Il existe un ensemble par défaut pour chaque modèle et chaque point d'arrêt y réside généralement, mais vous pouvez enregistrer la configuration actuelle du point d'arrêt en tant qu'ensemble nommé, créer un nouvel ensemble et basculer entre eux. Les postes Point d'Arrêt sont partagés ; c'est-à-dire qu'ils sont disponibles pour la communauté modèle. L'exception est l'ensemble par défaut qui est un ensemble privé et personnel alloué à chaque utilisateur de n'importe quel modèle.

Accéder

Ruban	Exécuter > Windows > Points d'Arrêt Simuler > Simulation Dynamique > Points d'Arrêt

Options Point d'Arrêt et de marqueur

Option	Détail
Supprimer un point d'arrêt ou un marqueur	Pour supprimer un point d'arrêt spécifique : • Si le point d'arrêt est activé, cliquez sur le cercle rouge du point d'arrêt dans la marge gauche de la Source Éditeur de Code , ou • Cliquez-droit sur le point d'arrêt ou le marqueur dans le Source Éditeur de Code , le dossier *Points d'Arrêt* ou la fenêtre Points d'Arrêt & Markers et sélectionnez l'option 'Supprimer', ou • Sélectionnez le point d'arrêt dans l'onglet ' Déboguer Points d'Arrêt ' et appuyez sur la touche Suppr
Supprimer tous les points d'arrêt	Cliquez sur le bouton Supprimer tous les points d'arrêt ( ).
Propriétés Point d'Arrêt	Dans la fenêtre Points d'Arrêt ou dans l'éditeur de code, utilisez le menu contextuel du marqueur pour faire apparaître les propriétés. Ici, vous pouvez changer le type de marqueur, ajouter ou modifier des contraintes et saisir des instructions de trace. (Raccourci utile : maintenez la touche Ctrl enfoncée tout en cliquant sur le marqueur pour afficher rapidement ses propriétés.)

Désactiver un point d'arrêt	Décochez la case en regard du point d'arrêt ou du marqueur.
Activer un point d'arrêt ou un marqueur	Cochez la case en regard du point d'arrêt ou du marqueur.
Désactiver tous les points d'arrêt	Cliquez sur le bouton
Activer tous les points d'arrêt	Cliquez sur le bouton Activer tous les points d'arrêt ( ).
Pause lorsque l'adresse mémoire est modifiée	Cliquez sur le bouton Point d'arrêt des données ( ).
Identifier ou modifier le jeu de marqueurs	Vérifiez le champ Default dans la barre d'outils de la fenêtre Points d'Arrêt & Événements . Si nécessaire, cliquez sur la flèche déroulante et sélectionnez un autre jeu de marqueurs. L'ensemble Default est normalement utilisé pour le débogage et est personnel à votre ID utilisateur ; d'autres jeux de marqueurs sont partagés entre tous les utilisateurs du modèle.
Modifier la façon dont les points d'arrêt et les marqueurs sont regroupés dans la fenêtre Points d'Arrêt et Événements	Les points d'arrêt et les marqueurs peuvent être regroupés par classe ou par fichier de code. Pour regrouper les éléments, cliquez sur la flèche vers le bas de l'icône dans la barre d'outils, puis cliquez sur l'option appropriée. Si vous ne souhaitez pas regrouper les éléments, cliquez sur l'option sélectionnée pour la désélectionner ; les points d'arrêt et les marqueurs sont ensuite répertoriés par numéro de ligne.

States Point d'Arrêt

State	Remarques
	Déboguer Running: Lié *Déboguer ne fonctionne pas :* activé
	Déboguer en cours d'exécution : Désactivé *Déboguer ne fonctionne pas :* désactivé
	Déboguer Running : Non lié - cela signifie généralement qu'un module n'a pas encore été chargé. De plus, les DLL sont déchargées de temps en temps. *Déboguer ne fonctionne pas :* N/a
	Déboguer en cours d'exécution : échec - cela signifie que le débogueur n'a pas pu faire correspondre cette ligne de code à une instruction dans l'un des modules chargés. Peut-être que la source provient d'un autre projet ou que la configuration du projet est obsolète. Note que si la date du module est antérieure à la date du code source du point d'arrêt, vous verrez une notification dans la fenêtre du débogueur. Le texte est de couleur rouge pour qu'ils se démarquent. C'est un signe clair que le

	projet nécessite une construction. *Déboguer ne fonctionne pas :* N/a

Définition Points d'Arrêt du code

Points d'Arrêt normaux sont généralement définis sur une ligne de code source. Lorsque le Débogueur atteint la ligne indiquée pendant l'exécution normale, le Débogueur arrête l'exécution et affiche les variables locales, la pile d'appels, les threads et autres informations d'exécution.

Définir un point d'arrêt sur une ligne de code

Étape	Action
1	Ouvrez le code source à déboguer dans l'éditeur de code source intégré.

2	Recherchez la ligne de code appropriée et cliquez dans la colonne de marge de gauche. Un cercle rouge plein dans la marge indique qu'un point d'arrêt a été défini à cette position.
	`12 CTest::CTest(LPCTSTR name, TTestType type)`
	`13 {`
	`14 m_Name = name;`
	`15 m_Type = type;`
	`16 theTest = this;`
	`17 }`
	Si le code est actuellement arrêté à un point d'arrêt, ce point est indiqué par une flèche bleue à côté du marqueur.
	`6 int _tmain(int argc, _TCHAR* argv[])`
	`7 {`
	`8 CTest Test(_T("Model"), CTest::Regression);`
	`9 return Test.Run();`
	`10 }`
	Alternativement, vous pouvez définir le marqueur de Point d'Arrêt (ou autre marqueur) en cliquant avec le bouton droit sur la marge gauche de la ligne souhaitée, pour afficher le menu contextuel du point d'arrêt/marqueur ; sélectionnez le type de marqueur approprié.

Instructions de trace

Une instruction Trace est un message généré lors de l'exécution d'une session de débogage. Les instructions de trace peuvent être définies dans Enterprise Architect sans nécessiter de modification du code source de votre application.

Les marqueurs Point de Trace sont définis dans l'éditeur de code. Comme les points d'arrêt, ils sont placés sur une ligne de code. Lorsque cette ligne de code s'exécute, le débogueur évalue l'instruction, dont le résultat est enregistré dans la fenêtre Déboguer (ou dans un fichier s'il est remplacé par le script Analyzer).

Accéder

Toutes les instructions Trace existantes peuvent être visualisées et gérées dans la fenêtre Points d'Arrêt et marqueurs. La fenêtre Points d'Arrêt et marqueurs peut être affichée en utilisant l'une des méthodes décrites ici.

Ruban	Exécuter > Windows > Points d'Arrêt

Ajouter un marqueur Point de Trace

Étape	Action
1	Ouvrez le code source à déboguer dans l'éditeur de code source.
2	Recherchez la ligne de code appropriée, cliquez-droit dans la marge de gauche et sélectionnez l'option 'Ajouter un marqueur Point de Trace '. Si un marqueur est déjà là, appuyez sur Ctrl+clic pour afficher la fenêtre Propriétés Point d'Arrêt .
3	Assurez-vous que la case « Trace Statement » est cochée.
4	Dans le champ de texte sous la case à cocher « Instruction Trace », saisissez l'instruction Trace requise.
5	Cliquez sur le bouton OK . Un marqueur Point de Trace est affiché dans la marge gauche de l'éditeur de code. 55 DWORD CTrain::Disembark(int PeopleOFF) 56 { 57 if(Passengers - PeopleOFF > -1) 58 Passengers -= PeopleOFF; 59 else 60 Passengers = 0; 61 62 if(PeopleOFF > 0) 63 return PeopleOFF * 20; 64 65 return 0; 66 }

Spécification d'une instruction de trace

Une instruction trace peut être n'importe quel texte de forme libre. La valeur de toutes les variables actuellement dans la portée peut également être incluse dans une instruction trace en préfixant le nom de la variable avec un jeton spécial.

Les jetons disponibles sont :

- $ - lorsque la variable doit être interprétée comme une string
- @ - lorsque la variable est un type primitif (int , double, char)

En utilisant notre exemple dans l'image, nous pourrions afficher le nombre de personnes descendant d'un train en utilisant cette instruction :

Il y avait des @Passengers avant que @PeopleOFF ne descende du train à la station $Arriving.Name

En plus de tracer les valeurs des variables à partir de votre code, vous pouvez utiliser les mots-clés $stack et $frame dans votre instruction Trace pour imprimer la trace actuelle de la pile ; utiliser:

- $stack - pour imprimer toutes les images, ou
- $frame[start](count) - imprime un nombre spécifique d'images de la pile en commençant à une image donnée ; par exemple, $frame[0](5) imprimera le cadre actuel et 4 ancêtres

Notes

- Les instructions de trace peuvent être incluses sur n'importe quel type de point d'arrêt ou de marqueur.

Interrompre lorsqu'une Variable Change de Valeur

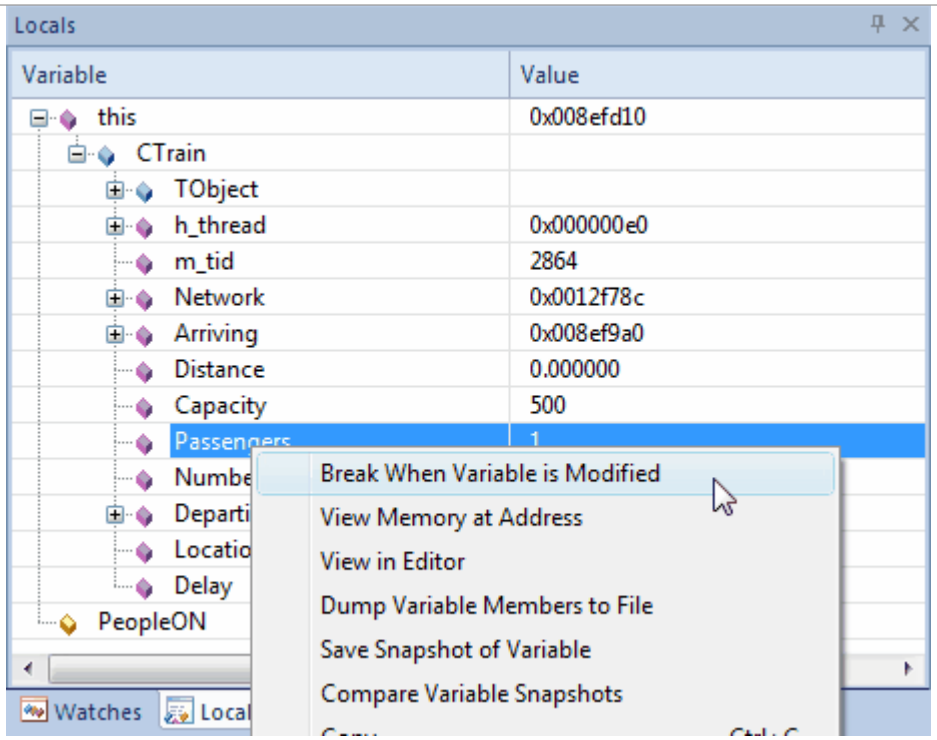
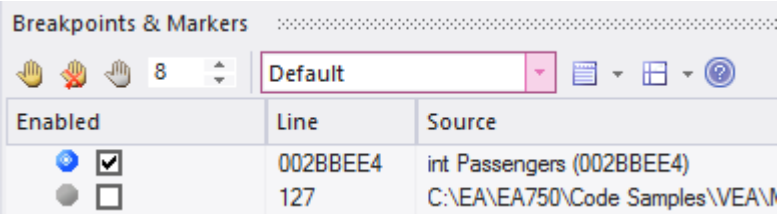
Des points d'arrêt de données peuvent être définis sur une variable de mémoire prédéterminée pour amener le débogueur à arrêter l'exécution à la ligne de code qui vient de provoquer la modification de la valeur de la variable. Cela peut être utile lorsque vous essayez de localiser le point auquel une variable est modifiée pendant l'exécution du programme, en particulier s'il n'est pas clair comment l'exécution du programme affecte un état object particulier.

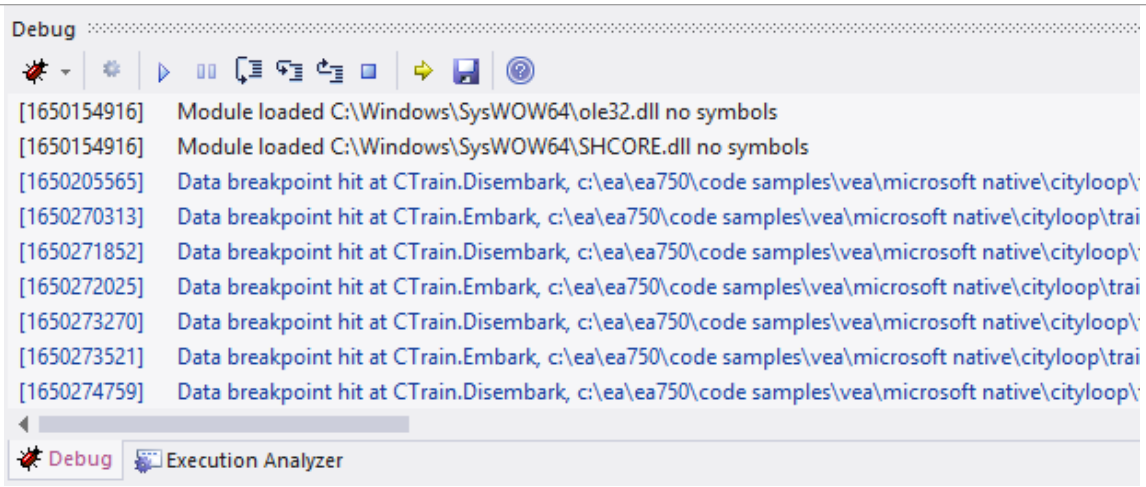
Accéder

Ruban	Exécuter > Windows > Variables Locales : Cliquez-droit sur la variable > Break When Variable is Modified ou Exécuter > Windows > Observateurs : Cliquez-droit sur la variable > Break When Variable is Modified
Autre	Dans une fenêtre d'éditeur de code : Cliquez-droit sur la variable qui vous intéresse | Casser lorsque l'élément est modifié

Capturer les modifications apportées à une variable à l'aide de points d'arrêt de données

Pas	Détail
1	Définissez un point d'arrêt normal dans le code afin de pouvoir choisir une variable. exécuter ensuite le débogueur (F6).
2	Lorsque le programme a atteint le point d'arrêt, sélectionnez la variable qui vous intéresse et dans son menu contextuel, sélectionnez l'option « Break When Variable is Modified ».

	
3	Il n'y a pas d'indicateurs de points d'arrêt dans le code, mais les points d'arrêt des données sont facilement reconnaissables dans la fenêtre Points d'Arrêt & Événements , sous la forme d'une icône bleue avec un losange blanc. Enterprise Architect affiche le nom de la variable et son adresse au lieu d'un numéro de ligne. 
4	Une fois le point d'arrêt des données défini, vous pouvez désactiver tous les autres points d'arrêt dont vous disposez. Le programme s'arrêtera à toute ligne de code qui modifie valeur de cette variable. exécuter maintenant votre programme.
5	Lorsque cette variable est modifiée, le débogueur s'arrête et affiche la ligne de code en cours dans l'éditeur. Il ne s'agit pas de la ligne qui a provoqué la rupture, mais de la ligne de code suivant l'événement. L'événement est journalisé dans la fenêtre Déboguer .



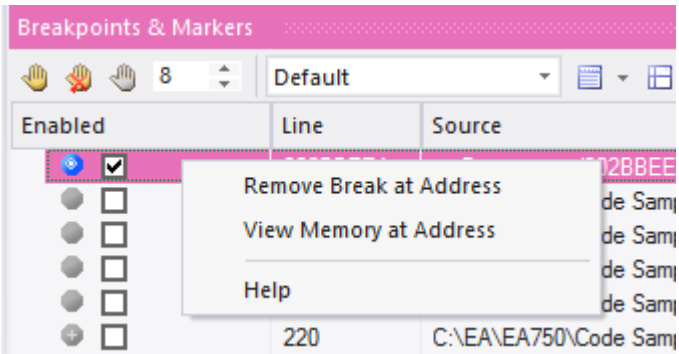
Nous savons maintenant comment et où cette valeur (son State) a changé. Par exemple, le relevé de la ligne 58 vient de mettre à jour le nombre de passagers.

```
55 DWORD CTrain::Disembark(int PeopleOFF)
56 {
57     if(Passengers - PeopleOFF > -1)
58         Passengers -= PeopleOFF;
59     else
60         Passengers = 0;
61
62     if(PeopleOFF > 0)
63         return PeopleOFF * 20;
64
65     return 0;
66 }
```

6

Après avoir découvert cet endroit et d'autres où cette valeur est modifiée, assurez-vous de vous débarrasser de la notification avant de continuer. Vous pouvez supprimer rapidement le point d'arrêt des données en le sélectionnant dans la fenêtre Points d'Arrêt et en appuyant sur la touche Suppr.

Vous pouvez également utiliser le menu contextuel cliquez-droit pour ce faire.



Notes

- Cette fonctionnalité n'est actuellement pas prise en charge par la plateforme Microsoft .NET

Trace lorsque Variable Change de Valeur

Lorsque votre code s'exécute, il peut modifier la valeur d'une variable. Il est possible de capturer ces changements ainsi que la nouvelle valeur de la variable, sur la fenêtre Déboguer . Vous pouvez ensuite double-cliquer sur la fiche de modification pour afficher la ligne de code qui a provoqué la modification, dans l' Éditeur de Code .

Accéder

Ruban	Exécuter > Windows > Variables Locales : Cliquez-droit sur la variable > Trace lorsque la variable est modifiée ou Exécuter > Windows > Observateurs : Cliquez-droit sur la variable > Trace lorsque la variable est modifiée
Autre	Dans Éditeur de Code | Cliquez-droit sur variable | Trace lorsque la variable est modifiée

Configurer la trace

La variable que vous tracez doit être dans la portée, donc pour l'identifier et la sélectionner, définissez un point d'arrêt normal sur la ligne de code où vous savez que la variable existera. Lorsque le débogueur atteint ce point d'arrêt, localisez la variable et utilisez son menu contextuel pour activer la trace.

Pour localiser une variable :

- Si vous voyez la variable dans le code source, survolez-la, cliquez-droit et sélectionnez l'option 'Afficher la variable' ; Enterprise Architect le localisera
- Si la variable est dans la portée (un local, ou 'this' ou un membre de 'this'), recherchez-la dans la fenêtre Locals (« Exécuter > Windows > Variables locales »).
- Si la variable est globale (C, C++), affichez la fenêtre Observateurs ('Execute > Windows > Observateurs ') et recherchez-la par son nom
- Si la variable est un membre statique de Classe, affichez la fenêtre Observateurs ('Execute > Windows > Observateurs ') et saisissez son nom complet

Une fois la trace activée, vous pouvez désactiver tous les autres points d'arrêt et laisser le programme exécuter . Chaque fois que la variable change valeur , elle sera enregistrée dans l'onglet 'Sortie' du débogueur. Vérifiez l'évolution de valeur et double-cliquez sur la ligne pour afficher le code dans l' Éditeur de Code .

Notes

- Le débogueur ne s'arrête pas lorsque l'événement de changement se produit, il enregistre uniquement le changement
- Cette facilité est disponible sur les plateformes Microsoft Native et Java
- Microsoft .NET ne prend pas support les points d'arrêt sur les valeurs

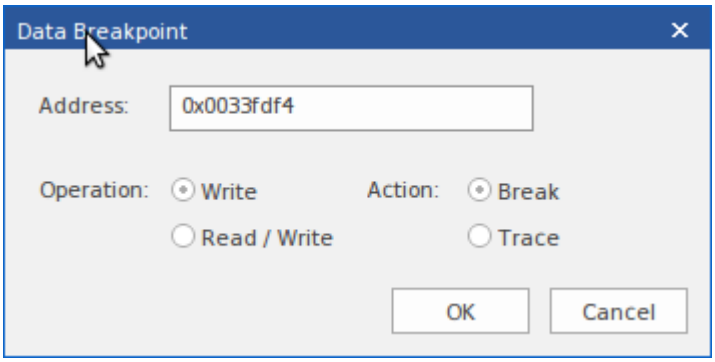
Détection des Opérations d'Adresse Mémoire

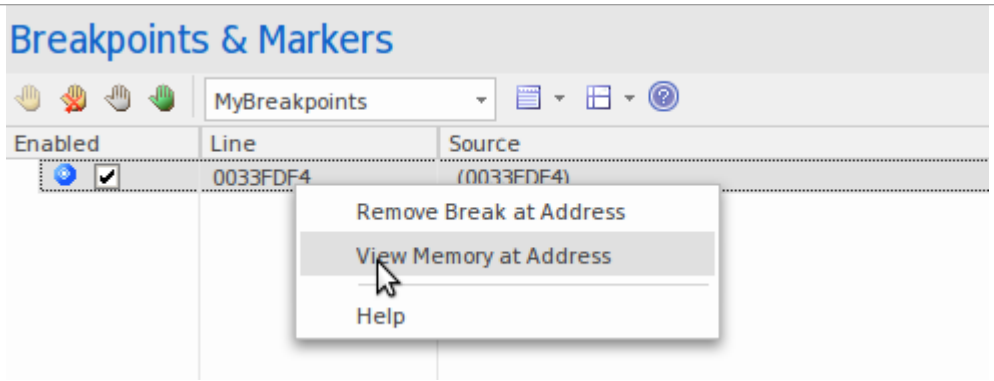
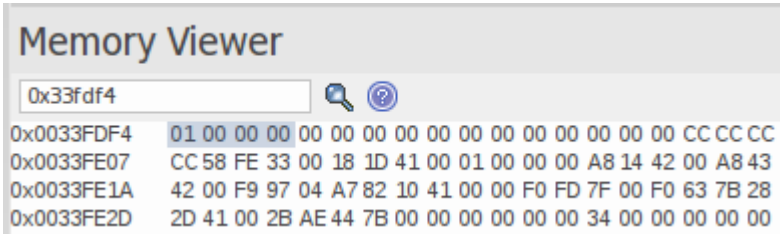
Être capable de détecter où et quand une zone de mémoire est lue ou écrite peut être d'une grande aide pour les enquêteurs, même lorsque la base de code est bien comprise. Sans cet outil, un développeur C++ pourrait avoir une tâche potentiellement ardue consistant à suivre où et quand une variable globale est accédée et à déboguer ces fonctions. Les points d'arrêt de données permettent à un programmeur C++ de suivre le moment où une variable/un emplacement mémoire est lu ou lorsqu'il est écrit. Lorsque l'opération est détectée, le débogueur arrêtera l'exécution et la ligne de code suivant l'opération sera affichée dans l'éditeur de code.

Accéder

Ruban	Exécuter > Windows > Points d'Arrêt

Détecter l'opération sur l'adresse mémoire

Étape	Action
1	Cliquez sur le bouton .
2	Entrez l'adresse mémoire à surveiller. Vous pouvez copier une adresse à partir de la fenêtre Locales (Local Variables). 
3	Sélectionnez l'opération à détecter. Si vous sélectionnez « Écrire », le débogueur s'arrêtera lorsque l'adresse sera écrite. Si vous choisissez 'Lecture/Ecriture', le débogueur vous avertira lorsque l'adresse sera lue ou écrite.
4	Sélectionnez l'action à effectuer. Si vous choisissez 'Break', le débogueur arrêtera le programme et la ligne de code sera affichée dans l'éditeur. Si vous choisissez « Trace », le débogueur n'arrêtera pas l'exécution, mais log toute opération sur l'adresse au fur et à mesure qu'elle se produit. Cette sortie est affichée dans la fenêtre Débogueur .
5	Le point d'arrêt des données est ajouté à la fenêtre Points d'Arrêt et marqueurs.

	
6	Vous pouvez utiliser le menu contextuel sur le point d'arrêt des données pour vérifier la valeur à l'adresse mémoire. 
7	Pour supprimer un point d'arrêt de données, sélectionnez-le dans la fenêtre Points d'Arrêt et marqueurs et appuyez sur la touche Suppr. Vous pouvez également décocher la case à côté. Les points d'arrêt de données sont supprimés lorsqu'ils sont désactivés ; ils ne persistent pas comme le font les autres points d'arrêt.

Exigences du système

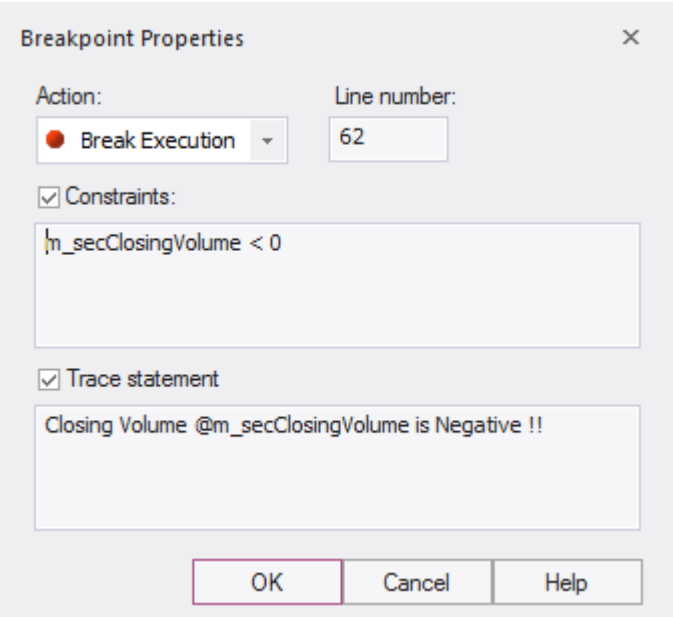
Les points d'arrêt d'adresse mémoire sont pris en charge dans le débogueur natif C/C++.

Point d'Arrêt Propriétés

Points d'Arrêt possèdent un certain nombre de propriétés supplémentaires qui déterminent ce qui se produit lors de l'exécution de la ligne de code à laquelle le point d'arrêt s'applique.

Ces propriétés définissent :

- L'action à réaliser
- La ligne de code à laquelle le point d'arrêt s'applique
- Contraintes qui déterminent si l'action est effectuée ou non lorsque le point d'arrêt est atteint
- Informations de trace à afficher lorsque le point d'arrêt est atteint



Accéder

Il existe plusieurs manières d'afficher la dialogue ' Point d'Arrêt Propriétés ' :

Éditeur de Code	• Cliquez-droit sur un marqueur de point d'arrêt | Propriétés ou • Ctrl+Clic sur le marqueur de point d'arrêt ou • Cliquez-droit sur le code comportant un marqueur de point d'arrêt | Point d'Arrêt | Propriétés
Fenêtre Points d'Arrêt et marqueurs	• Cliquez-droit sur le point d'arrêt | Propriétés

Possibilités

Champ	Détails
Action	Le comportement lorsque le point d'arrêt est atteint.

Doubler	Ligne de code source à laquelle ce point d'arrêt s'applique.
Hauteur de la pile	Pour les marqueurs Stack Capture, le nombre d'images de l'appelant à enregistrer. Pour enregistrer la pile entière, définissez la valeur sur 0.
Contraintes	Définit la condition dans laquelle l'action du point d'arrêt sera effectuée. Pour les points d'arrêt normaux, ce serait la condition qui interrompt l'exécution. Dans cet exemple, pour un point d'arrêt normal, l'exécution s'arrêterait à cette ligne lorsque la condition est évaluée à True. Les contraintes sont évaluées à chaque fois que la ligne de code est exécutée. (this.m_FirstName="Joe") AND (this.m_LastName="Smith")
Instruction de trace	Un message affiché dans la fenêtre Déboguer lorsque le point d'arrêt est atteint. Les variables actuellement dans la portée peuvent être incluses dans la sortie d'une instruction trace en préfixant le nom de la variable avec un jeton $ pour les variables string , ou un jeton @ pour les types primitifs tels que int ou long. Par exemple: Le compte $pAccount->m_sName a un solde de @pAccount->m_fBalance

Défaut de lier Point d'Arrêt

Un échec de point d'arrêt se produit s'il y a un problème lors de la liaison du point d'arrêt. Les échecs Point d'Arrêt sont le plus souvent causés par la modification des fichiers sources sans que l'application ne soit reconstruite. Points d'Arrêt peuvent parfois se lier à une ligne différente, provoquant leur déplacement. Si un point d'arrêt ne peut pas être lié au binaire sur cette ligne ou les trois lignes qui la suivent, il est affiché avec un point d'interrogation.

Un message d'avertissement s'affiche dans la colonne 'Détails' de la fenêtre Points d'Arrêt & Événements , identifiant le type de problème :

- Le fichier source du point d'arrêt ne correspond pas au fichier source utilisé pour créer l'image de l'application.
- L'horodatage du fichier est supérieur à celui de l'image

Un message d'avertissement est également affiché dans la fenêtre Déboguer .

Déboguer une Application en Cours

Plutôt que de démarrer un processus explicitement à partir d' Enterprise Architect , vous souhaiterez peut-être déboguer une application (processus) déjà en cours d'exécution sur votre système.

Dans ce cas, vous pouvez utiliser la fonctionnalité de débogage pour vous attacher au processus déjà en cours d'exécution. À condition que les informations de débogage appropriées soient écrites dans le processus en cours d'exécution et/ou dans les fichiers de débogage associés (tels que les fichiers .PDB), le débogueur se lie à ce processus et lance une session de débogage.

Vous pouvez également vous « détacher » du processus après avoir terminé votre inspection et laisser le processus se exécuter normalement.

Accéder

Ruban	Exécuter > Exécuter > Démarrer > Attacher au processus
Fenêtre Débogueur	La barre d'outils de la fenêtre du débogueur comporte un bouton Attacher

Étapes

Scène	Description
Afficher Processus	Lorsque vous choisissez de déboguer un autre processus, la dialogue « Attacher au processus » s'affiche. Vous pouvez limiter les processus affichés à l'aide des boutons radio en haut de le dialogue ; pour rechercher un service tel qu'Apache Tomcat ou ASP.NET, sélectionnez le bouton radio Système.
Sélectionnez Débogueur	Lorsque vous sélectionnez un processus, vous devrez peut-être choisir le débogueur dans la liste déroulante Débogueur ; cependant, si le Paquetage sélectionné a déjà été configuré dans un script Analyzer, alors le débogueur répertorié dans le script est prédéfini sur le dialogue .
Sélection du processus	Une fois que vous double-cliquez sur un processus contenant des informations de débogage, Enterprise Architect est attaché au processus : • Tous les points d'arrêt rencontrés sont détectés par le débogueur • Le processus est arrêté lorsqu'un point d'arrêt est rencontré, et • Les informations sont disponibles dans la fenêtre Déboguer
Se détacher du processus	Pour vous détacher d'un processus, cliquez sur le bouton (Arrêt Débogage).

Voir les Variables Locales

La fenêtre Locals affiche les variables du système d'exécution. Que vous enregistriez C# , déboguez Java, C++ ou VBScript, déboguez un Statemachine Exécutable ou exécutez une simulation, cette fenêtre est l'endroit où se trouvent les variables du système. Les valeurs actuelles ne sont affichées que lorsqu'un programme est arrêté. Cela se produit lorsqu'un point d'arrêt est rencontré lors du débogage, lorsque vous parcourez une ligne de code ou lorsque vous passez d' States à l'autre dans une simulation.

Accéder

Ruban	Exécuter > Windows > Variables locales Simuler > Simulation Dynamique > Variables locales
Menu Contexte	Dans Éditeur de Code | Cliquez-droit sur n'importe quel identifiant de variable > Afficher la variable

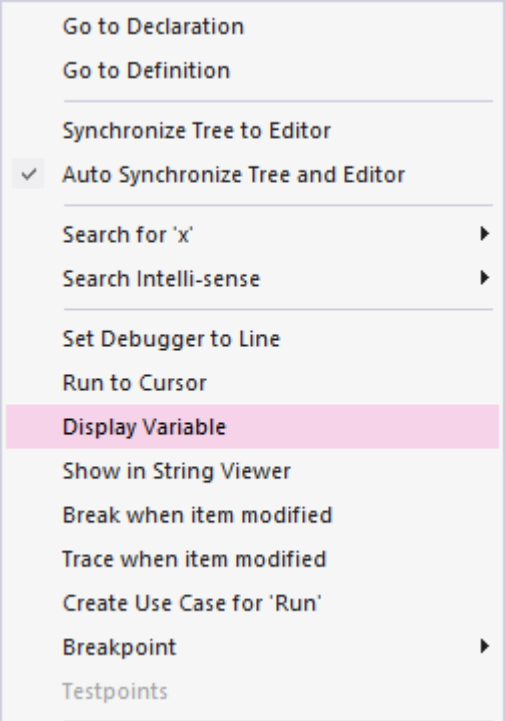
Icônes

La valeur et le type de toute variable dans la portée sont affichés dans une arborescence ; chaque variable possède une icône en forme de boîte colorée qui identifie le type de variable :

- Bleu - Object avec membres
- Vert – Tableaux
- Rose – Types élémentaires
- Jaune - Paramètres
- Rouge - Instance Établi

Trouver des variables

Le moyen le plus simple de trouver une variable est de la localiser d'abord dans l'éditeur de code et d'utiliser le menu contextuel cliquez-droit sur la variable, en sélectionnant 'Afficher la variable'. Enterprise Architect trouvera et révélera toute variable de portée, y compris les membres profondément imbriqués. Si la variable se trouve dans une portée différente (globale, fichier, module, statique), elle sera affichée dans la fenêtre Observateurs (voir *Voir Variables dans d'Autres Portées*).

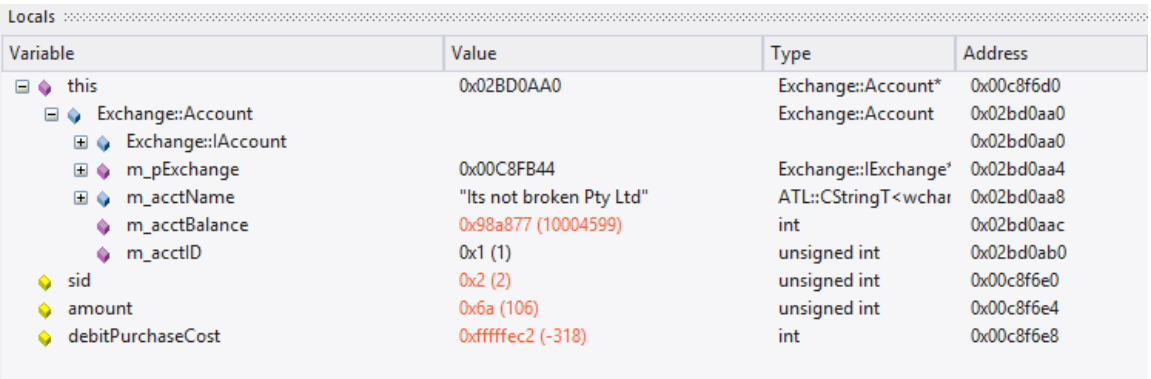


Vue persistante

L'examen des variables implique généralement de creuser dans l'arbre pour exposer les valeurs d'intérêt. Il peut alors être ennuyeux, après avoir traversé ces problèmes, de passer à la ligne de code suivante, pour ensuite voir ces variables à nouveau cachées à cause d'un changement de contexte. La fenêtre Locals a une vue persistante qui persiste pendant un certain temps après une commande exécuter ou step. Lorsque vous parcourez une fonction dans Enterprise Architect , la structure des variables persiste ligne après ligne. Cela permet de parcourir une fonction rapidement et facilement.

Qu'est ce qui a changé

Dans le cadre de la vue persistante, la fenêtre Locals suit les modifications apportées aux valeurs et les met en surbrillance.

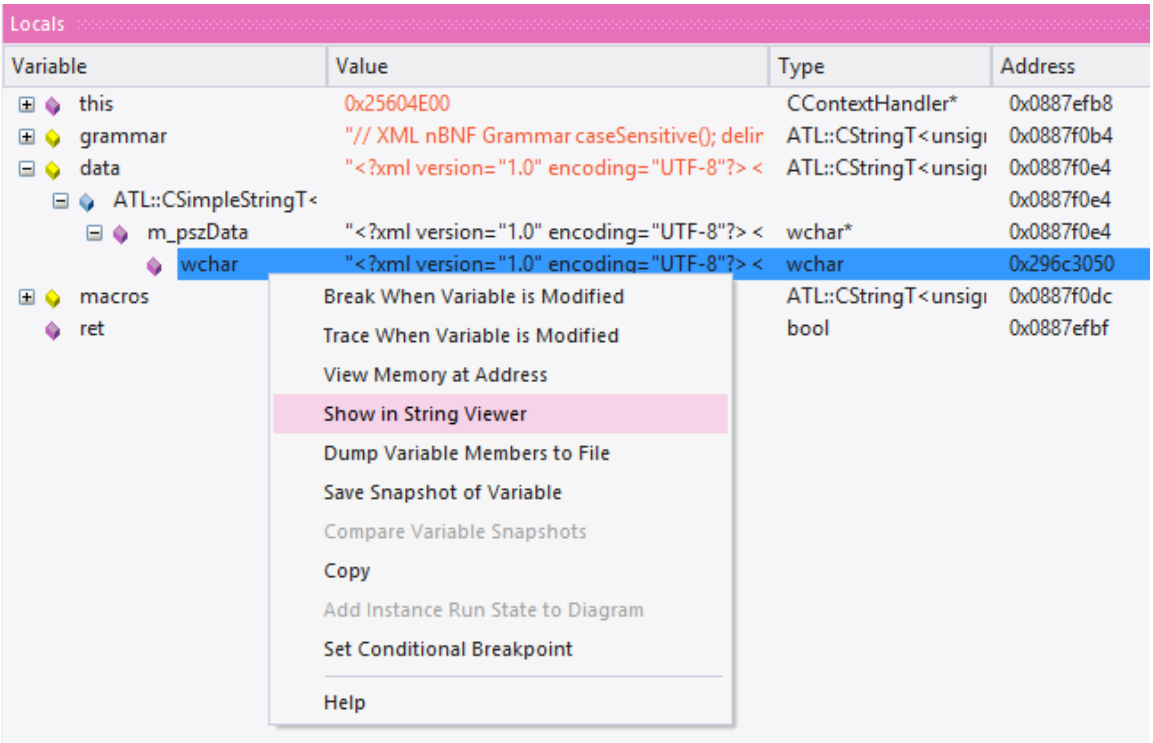


Menu Contexte

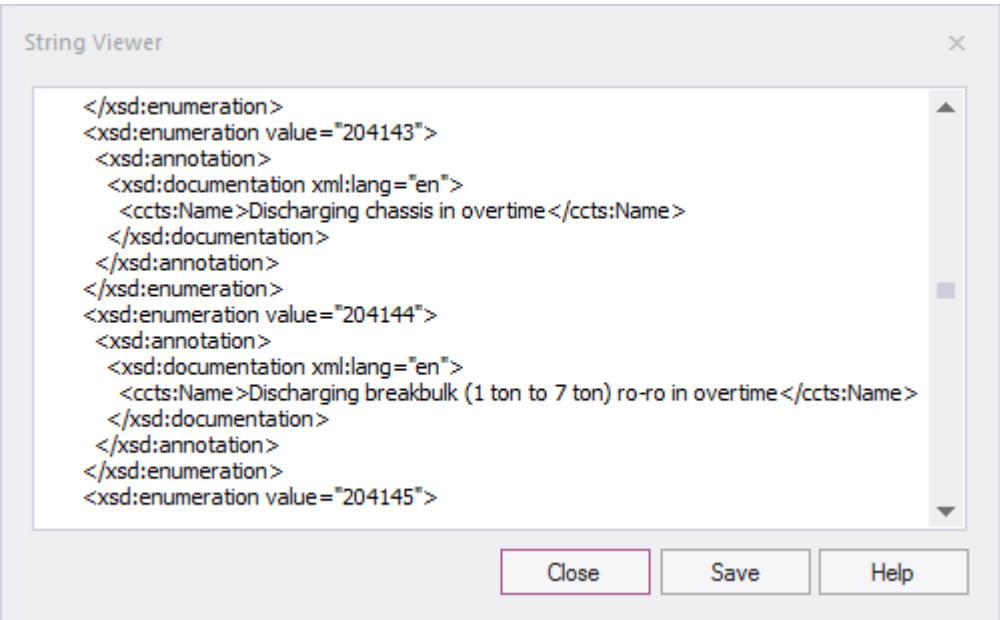
Facilité	Détail
Pause lorsque la variable est modifiée	Définissez des points d'arrêt de données sur la variable de mémoire sélectionnée pour arrêter l'exécution du débogueur à la ligne de code qui vient de provoquer la modification de la valeur de la variable.
Mémoire Vue à l'adresse	Afficher les valeurs brutes en mémoire à l'adresse sélectionnée, en hexadécimal et ASCII.
Afficher dans la visionneuse String	Affichez la string variable dans la dialogue ' String Viewer'.
Vider les membres variables dans un fichier	Capturez et stockez les variables sélectionnées dans un emplacement distinct ; un navigateur s'affiche pour sélectionner le nom du fichier .txt et le chemin du fichier appropriés.
Enregistrer un instantané de la variable	Capturez la valeur d'une variable à un moment précis de la vie de cette variable.
Comparer les instantanés variables	Comparez les valeurs d'une variable à différents moments de la vie de cette variable.
Copie	Copiez la variable sélectionnée dans le presse-papiers Enterprise Architect .
Ajouter State Exécuter de l'instance au Diagramme	Si vous avez ouvert un diagramme modèle contenant un Object de la Classe pour laquelle le code source est en cours de débogage, cette option met à jour cet Object avec l' State Exécuter représenté par la variable valeur .
Définir Point d'Arrêt conditionnel	Ajoutez un point d'arrêt à la position d'exécution actuelle avec une contrainte pour cette variable correspondant à sa valeur actuelle.

Voir le Contenu de Longues Chaînes

Pour des raisons d'efficacité, la fenêtre Locals affiche uniquement les chaînes partielles. Cependant, vous pouvez afficher l'intégralité du contenu d'une variable string à l'aide du ' String Viewer'.



Cet exemple montre la valeur d'une variable contenant le contenu d'un fichier de schéma XML.



Accéder

Depuis la fenêtre Éditeur de Code ou Locals	Cliquez-droit sur la variable string | Afficher dans la visionneuse String

Variables Vue Déboguer dans Éditeurs de Code

Lorsqu'un point d'arrêt se produit, vous verrez toutes les variables locales dans cette fenêtre. Vous pouvez également inspecter les variables dans le Source Éditeur de Code en passant votre souris sur la référence. Voici quelques exemples.

```
public void Print()
{
    int n = 0;
    while(names[n].Length > 0)
    {
        Document d = new Document(names[n++]);
        d.Print();
    }
}
```

names = {[4] names[0]=book, names[0]=book, names[1]=novel, names[2]=film}, ...}

```
public void Print()
{
    int n = 0;
    while
    {
        Document d = new Document(names[n++]);
        d.Print();
    }
}
```

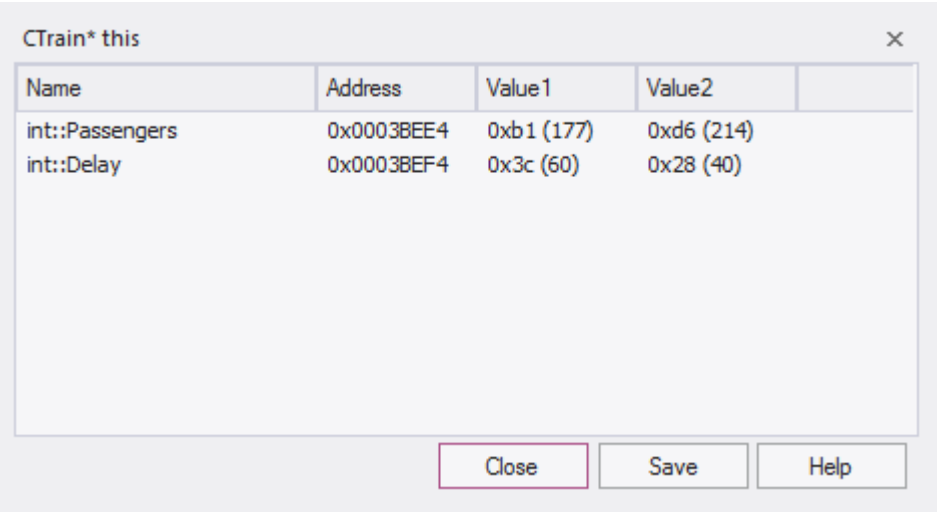
32-bit signed integer n=0

Note : La variable ne doit pas nécessairement faire partie des variables locales. Il peut avoir une portée de fichier ou de module.

Instantanés variables

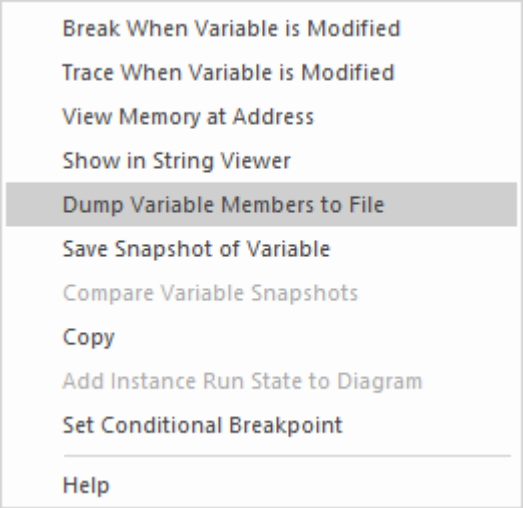
Il est possible de prendre un « instantané » d'une variable lorsque votre programme atteint un point d'arrêt et d'utiliser cet instantané pour voir comment la valeur de la variable change à différents moments de sa vie. Le débogueur ne copie pas uniquement la valeur de la variable sélectionnée ; pour les variables complexes, il copie les valeurs de la variable sélectionnée et de chacune de sa hiérarchie de membres jusqu'à ce qu'il ne puisse plus trouver d'informations de débogage pour un membre ou qu'aucun autre membre ne puisse être trouvé.

Capturer un instantané variable

Étape	Action
1	Dans l' Éditeur de Code , définissez deux points d'arrêt : un au début d'une fonction et un autre à la fin de la fonction.
2	Au point d'arrêt de départ, cliquez-droit sur une variable dans la fenêtre Locals et sélectionnez l'option de menu 'Save Variable Snapshot'.
3	Exécuter l'application.
4	Lorsque le point d'arrêt de fin est atteint, cliquez-droit sur la variable dans la fenêtre Locals et sélectionnez l'option 'Comparer les instantanés de variables'. Une dialogue s'affiche qui montre la valeur d'origine du premier instantané et la valeur actuelle du deuxième instantané, comme illustré dans ce diagramme tiré du modèle EA.Example. 

Enregistrer un instantané de variable dans un fichier

Vous pouvez enregistrer l'état d'une variable dans un fichier en utilisant son menu contextuel cliquez-droit .



Ceci est un extrait du contenu du fichier.

```
00000006|0x00731F00|name|TObjectType::Type  |value|TypeIsStation|
00000005|0x00731F08|name|wchar::Name    |value|"Treasury"|
00000005|0x00731F0C|name|unsigned::Location |value|0x40 (64)|
00000003|0x0003BED8|name|float::Distance    |value|0|
00000003|0x0003BEE0|name|int::Capacity  |value|0x1f4 (500)|
00000003|0x0003BEE4|name|int::Passengers    |value|0xd6 (214)|
00000003|0x0003BEE8|name|unsigned::Number   |value|0x3 (3)|
00000003|0x0003BEF0|name|unsigned::Location |value|0x0 (0)|
00000003|0x0003BEF4|name|int::Delay |value|0x28 (40)|
```

Points d'Action

Points d'Action sont des points d'arrêt qui peuvent effectuer des actions. Lorsqu'un point d'arrêt est atteint, le script Point d'Action est invoqué par le débogueur et le processus continue d' exécuter . Points d'Action sont des outils de débogage sophistiqués et fournissent aux développeurs experts une suite de commandes supplémentaire. Avec eux, un développeur peut modifier le comportement d'une fonction, capturer le point auquel un comportement change et modifier/détecter l'état d'un object . Pour support ces fonctionnalités , Points d'Action peut modifier la valeur des variables primitives locales et membres, définir ses propres "variables définies par l'utilisateur" et modifier l'exécution du programme.

Variables définies par l'utilisateur dans Points d'Action et Points d'Arrêt

Variables définies par l'utilisateur (UDV) :

- Fournir les moyens de définir une primitive ou string UDV dans les instructions Point d'Action
- Peut être utilisé dans les instructions de condition de plusieurs marqueurs/points d'arrêt
- Peut être vu facilement dans la même fenêtre Variables locales
- Les valeurs finales de tous les UDV sont enregistrées à la fin du débogage.

Dans la syntaxe UDV, le nom UDV :

- Doit être précédé d'un caractère # (dièse)
- N'est pas sensible à la casse

Énoncés Point d'Action

Les instructions Point d'Action peuvent contenir des commandes set, des commandes goto et des commandes jmp.

définir la commande

Ensembles de valeurs variables. Une instruction Point d'Action peut contenir plusieurs commandes « set », qui doivent toutes précéder toute commande « goto ».

La syntaxe de la commande 'set' est la suivante :

définir LHS = RHS

Où:

- **LHS** = le nom de la variable sous la forme :
  - variable définie par l'utilisateur (UDV) telle que #myval
  - variable locale ou membre telle que strName ou this.m_strName
- **RHS** = la valeur à attribuer :
  - Comme variable littérale ou locale
  - S'il s'agit d'un littéral, parmi : integer , booléen, virgule flottante, caractère ou string

commande set - Exemples de variables

Exemples d'UDV	Exemples de variables locales
définir #mychar = 'a'	définir this.m_nCount=0

set #mystr = "une string "	définir bSuccès = faux
définir #monint = 10	
définir #myfloat = 0,5	
définir #mytrue = vrai	

aller à la commande

Cette commande bascule l'exécution vers un numéro de ligne différent dans une fonction. Une instruction Point d'Action ne peut contenir qu'une seule commande goto, comme commande finale de l'instruction.

La syntaxe de la commande goto est :

aller à L

Où **L** est un numéro de ligne dans la fonction actuelle.

La commande goto utilise des points d'arrêt pour atteindre son objectif, ce qui entraîne un léger retard dans l'exécution du code. Cela peut être perceptible dans les régions de code qui sont exécutées très fréquemment, vous préférerez donc peut-être utiliser la commande jmp dans un tel code, pour obtenir le même détournement d'exécution mais avec moins de délai.

commande jmp

La commande jmp est en fait la même que la commande goto .

jmp125

aller à 125

Ces deux commandes entraînent le changement de l'exécution à la ligne 125.

Cependant, l'instruction jmp utilise en interne une instrumentation pour demander au programme de déplacer l'exécution, tandis que l'instruction goto utilise des points d'arrêt pour ce faire, ce qui entraîne un retard dans le traitement. La différence réside donc dans les performances supérieures de l'instruction jmp , en particulier lorsque des régions de code sont exécutées très fréquemment.

Opérateurs Integer

Lorsqu'une variable définie par l'utilisateur (UDV) existe et qu'elle est de type int , elle peut être incrémentée et décrémentée à l'aide des opérateurs ++ et --. Par exemple:

1. Créez un UDV et définissez sa valeur et son type sur une variable integer locale.
   AP1 : définir #myint = nTotalSoFar
2. Incrémentez l'UDV.
   AP2 : #monint++
3. Décrémentez l'UDV.
   AP3 : #monint--

Opérations de minuterie

Points d'Action peut signaler le temps écoulé entre deux points. Il n'y a qu'un seul minuteur disponible, qui est réinitialisé ou démarré avec la commande startTimer. Le temps écoulé actuel peut ensuite être imprimé avec la commande printTimer. Enfin, le temps total écoulé est imprimé et le chronomètre se termine avec la commande endTimer.

Exemples de conditions Point d'Action

Avec des littéraux et des constantes :

- (#mychar='a')
- (#mystr <> " ")
- (#monint > 10)
- (#monflottant > 0.0)

Avec des variables locales :

- (#myval == this.m_strValue)
- (#myint <> this->m_nCount)
- (#myint != this->m_nCount)

Enregistrement des instructions

L'enregistrement des instructions peut être utile pour détecter les modifications apportées à un comportement connu ; le point de l'exécution (B) qui s'écarte d'une ou plusieurs exécutions précédentes (A). Les commandes sont :

- recStart - démarre l'enregistrement ou commence à comparer si un enregistrement précédent existe
- recStop - termine l'enregistrement
- recPause - suspendre l'enregistrement
- recResume - reprend l'enregistrement

La commande **recStart** commence l'enregistrement des instructions. Les instructions exécutées sont ensuite stockées. Lorsqu'une commande **recStop** est rencontrée, l'enregistrement est sauvegardé. Il ne peut y avoir qu'un seul enregistrement sauvegardé à la fois entre deux Points d'Action . Lorsqu'un **recStart** est rencontré et qu'un enregistrement précédent existe, le débogueur commencera à comparer chaque instruction suivante avec son enregistrement. Il pourrait effectuer de nombreuses comparaisons. Si et quand une différence est détectée, le débogueur s'arrêtera et la ligne de code où le comportement a changé sera affichée dans l'éditeur de code. L'itération de la comparaison est également imprimée.

L'enregistrement est stocké en mémoire par défaut, mais il peut également être stocké dans un fichier avec la syntaxe de commande :

spécifications des fichiers recStart

Par exemple:

recStart c:\mylogs\onclickbutton.dat

Lorsqu'une commande **recStart** spécifiant un fichier est rencontrée et que ce fichier existe, il est chargé en mémoire et le débogueur passe immédiatement en mode comparaison.

Expressions

Il n'y a pas de préséance implicite dans les expressions conditionnelles Point d'Arrêt , Point d'Action et Testpoint . Dans les expressions complexes, l'utilisation des parenthèses est obligatoire. Voir ces exemples :

Type	Exemple
Exemple de Point d'Action UDV	(#myint=1) AND (#mystr="Allemagne")
Exemples de variables locales	(this.m_nCount > 10) OR (nCount%1) (this.m_nCount > 10) OR (bForce)
Opérateurs d'égalité dans les expressions conditionnelles	<> - Pas égal != - Pas égal == - Égal = - Égal
Opérateur d'affectation chez Point d'Action	= - Attribue RHS à LHS
Opérateurs Arithmétique dans les expressions conditionnelles	/ - division + - plus - - moins * - multiplication % - module
Opérateurs logiques dans les expressions conditionnelles	AND - les deux doivent être vrais OR - il faut être vrai && - les deux doivent être vrais || - il faut être vrai ^ - OR exclusif (un seul doit être vrai)

Voir Variables dans d'Autres Portées

Accéder

Ruban	Exécuter > Windows > Observateurs
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution :	Observateurs

Vues

Vue	Description
Observateurs	La fenêtre Observateurs est particulièrement utile pour le code natif (C, C++, VB) où elle peut être utilisée pour évaluer des éléments de données qui ne sont pas disponibles en tant que variables locales - éléments de données avec portée de module ou de fichier et éléments membres de classe statiques. Vous pouvez également utiliser la fenêtre pour évaluer les éléments statiques des membres de classe en Java et .NET Pour ajouter une surveillance, saisissez le nom de la variable à surveiller dans la barre d'outils et appuyez sur la touche Entrée. Pour examiner une variable membre de classe statique en C++, Java ou Microsoft .NET , entrez son nom complet : CMyClass ::MaVarStatique Pour examiner un symbole de données C++ avec une portée de module ou de fichier, entrez simplement son nom. Les variables sont évaluées en examinant la portée actuelle ; c'est-à-dire le module du frame de pile actuel (vous pouvez modifier la portée à un point d'arrêt en double-cliquant sur le frame dans la Pile d'Appel). Si la variable globale existe dans un autre module, vous pouvez examiner la variable en préfixant le nom du module à la variable nom du module ! nom_de la variable Il est assez facile de mal saisir les noms d'éléments de données, donc si vous faites ou trouvez une erreur, mettez en surbrillance la string , appuyez sur F2 et retapez le texte. Cela accélère également la résolution des éléments nommés dans le débogueur, en interrompant une recherche lorsque des éléments correspondants sont découverts.
Histoire	L'historique des éléments saisis est conservé. Les noms ou expressions précédemment saisis peuvent être à nouveau sélectionnés à l'aide de la touche fléchée vers le haut et de la touche fléchée vers le bas dans la zone de texte de la barre d'outils. L'historique sera également conservé pour l'utilisateur sur n'importe quelle instance d' Enterprise Architect ou modèle sur la même machine.

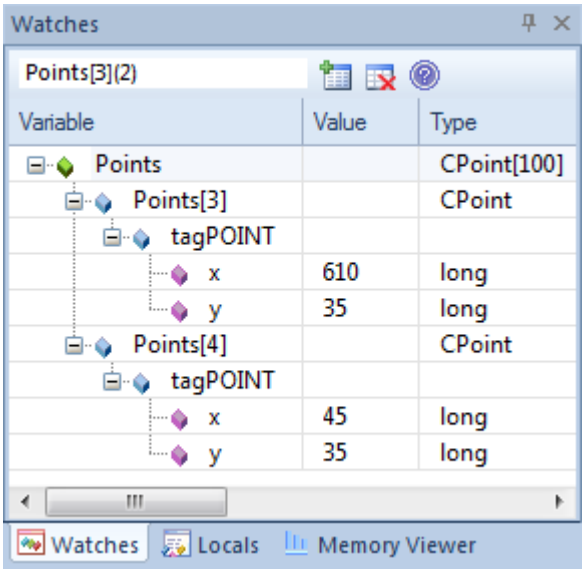
Voir Éléments du Réseau

Vous pouvez utiliser la fenêtre Observateurs pour inspecter un ou plusieurs éléments spécifiques d'un tableau.

Dans le champ à gauche de la barre d'outils de la fenêtre Observateurs , saisissez le nom de la variable du tableau suivi de l'élément de départ et du nombre d'éléments à afficher. L'élément de départ est placé entre crochets et le nombre d'éléments est placé entre parenthèses ; c'est-à-dire:

variable[start_element](count_of_elements)

Par exemple, Points[3](2) affiche les quatrième et cinquième éléments du tableau Points, comme illustré.



Si vous avez entré Points[3], la fenêtre Observateurs affichera uniquement le troisième élément du tableau.

Accéder

Ruban	Exécuter > Windows > Observateurs
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution :	Observateurs

Voir la Pile d'Appel

La fenêtre Pile d'Appel est utilisée pour afficher tous les threads en cours d'exécution dans un processus. Il peut être utilisé pour identifier quel thread est opérationnel, immédiatement avant que l'échec du programme ne se produise.

Lorsqu'une Simulation est active, la Pile d'Appel affichera le contexte d'exécution actuel de la simulation en cours. Cela inclura une pile de contexte distincte pour chaque « thread » de simulation simultanée.

Une trace de pile s'affiche chaque fois qu'un thread est suspendu, via l'une des actions d'étape ou lors de la rencontre d'un point d'arrêt. La fenêtre Pile d'Appel peut enregistrer un historique des changements de pile et vous permet de générer diagrammes Séquence à partir de cet historique.

Accéder

Ruban	Exécuter > Windows > Pile d'Appel
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution : | Pile d'Appel

Avoir l'habitude de

- Historique de la pile Vue pour comprendre l'exécution d'un processus
- Sujets Vue
- Enregistrer une pile d'appels pour une utilisation ultérieure
- Enregistrer les modifications de la pile d'appels pour la génération diagramme Séquence
- Générer un diagramme Séquence à partir de la pile d'appels
- Vue la ligne de code associée dans le Source Éditeur de Code

Facilités

Facilité	Description
Indicateurs	• Une flèche rose met en évidence le cadre de pile actuel • Une flèche bleue indique un thread en cours d'exécution • Une flèche rouge indique un thread pour lequel un historique de trace de pile est en cours d'enregistrement
Enregistrer une Pile d'Appel dans un fichier .TXT	Non disponible actuellement.
Enregistrer un fil de discussion dans une session Déboguer	Pour enregistrer l'exécution d'un thread et diriger l'enregistrement vers la fenêtre Enregistrer & Analyser, cliquez-droit sur le thread dans la Pile d'Appel et sélectionnez l'option de menu contextuel appropriée : • 'Enregistrer' - pour enregistrer manuellement le thread actuel pendant la session

	de débogage Utilisé conjointement avec les boutons « étape » du débogueur ; chaque fonction appelée en raison d'une commande d'étape est enregistrée dans la fenêtre Enregistrer et analyser • 'Auto-Record' - pour effectuer un enregistrement automatique pendant une session de débogage Lorsque vous sélectionnez cette icône, l'analyseur commence l'enregistrement et ne s'arrête que lorsque le programme se termine, que vous arrêtez le débogueur ou que vous cliquez sur l'icône 'Stop'.
Arrête d'Enregistrer	Si vous avez démarré un enregistrement manuel ou automatique d'un fil de discussion, vous pouvez l'arrêter avant la fin ; sélectionnez le fil (indiqué par une flèche rouge) et soit : • Cliquez sur le bouton (Arrête d'Enregistrer) dans la barre d'outils ou • Cliquez-droit et sélectionnez l'option 'Stop'
Générer un Diagramme de Séquence à partir de la Pile d'Appel	Pour générer diagramme Séquence à partir de la trace Pile d'Appel , soit : • Cliquez sur le bouton (Générer Diagramme de Séquence of Stack), ou • Cliquez-droit et sélectionnez l'option ' Générer Diagramme de Séquence '
Copier la pile dans Historique d'Enregistrement	Pour ajouter immédiatement les détails de la pile à la fenêtre Enregistrer et analyser (pour une génération ultérieure de diagrammes Séquence), soit : • Cliquez sur le bouton , ou • Cliquez-droit et sélectionnez l'option 'Copier la pile pour enregistrer l'historique'
Toggle Profondeur de Pile	Pour basculer entre l'affichage de la pile complète et l'affichage uniquement des images avec source, cliquez sur le bouton (Toggle Profondeur de Pile).
Afficher le code associé dans Source Éditeur de Code	Double-cliquez sur un thread/frame pour afficher la ligne de code associée dans le Source Éditeur de Code ; les variables locales sont également actualisées pour le cadre sélectionné.

Créer Diagramme de Séquence de Pile d'Appel

La fenêtre Pile d'Appel enregistre un historique des changements de pile à partir duquel vous pouvez générer diagrammes Séquence .

Accéder

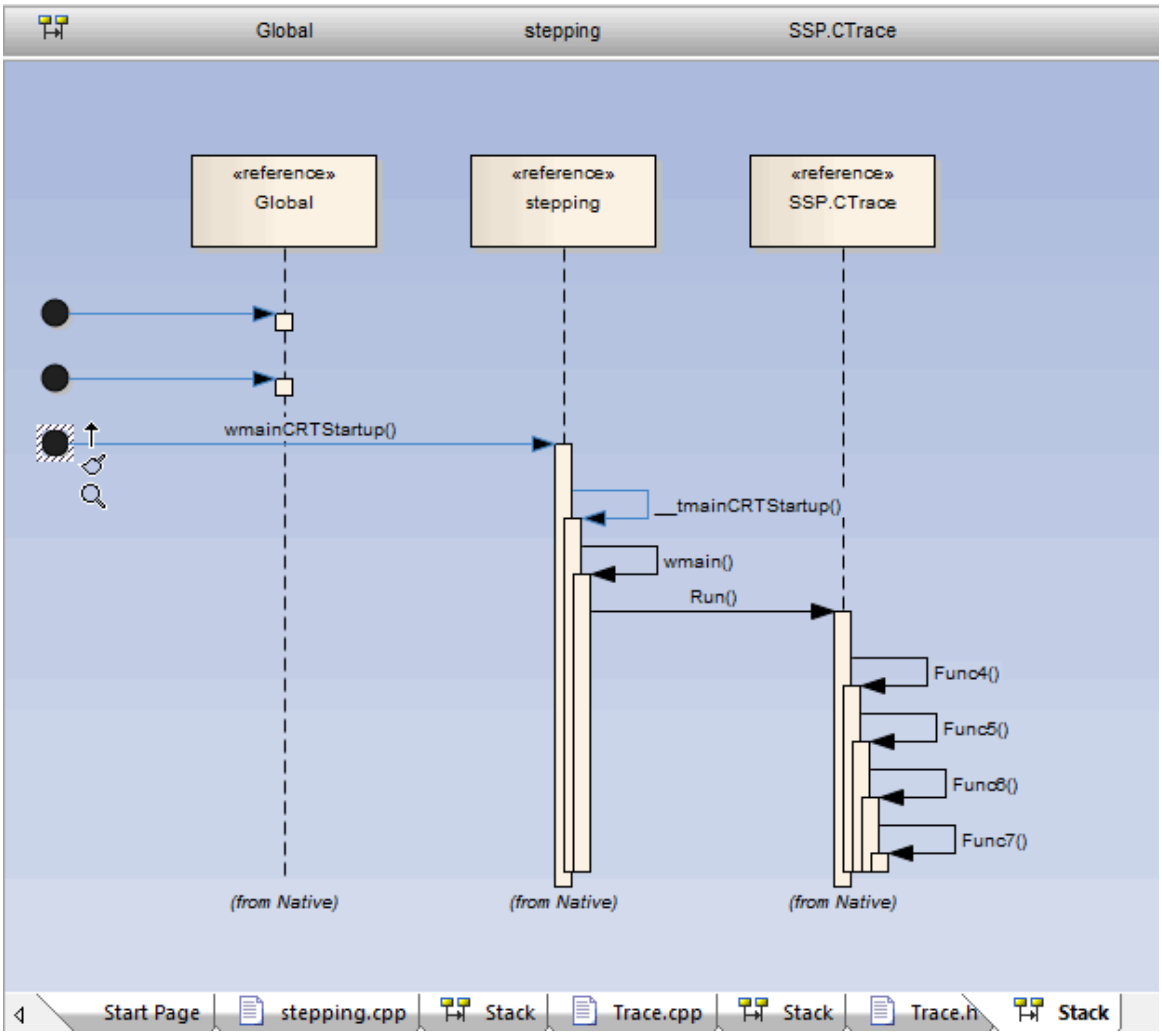
Ruban	Exécuter > Windows > Pile d'Appel
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution :	Pile d'Appel

Avoir l'habitude de

- Enregistrer les modifications Pile d'Appel pour la génération diagramme Séquence
- Générer un diagramme Séquence à partir de la Pile d'Appel

Pour générer un diagramme Séquence à partir de la Stack actuelle, cliquez sur le bouton (Générer Diagramme de Séquence of Stack) de la barre d'outils de la fenêtre Pile d'Appel .

Cela génère immédiatement un diagramme Séquence dans le Diagramme Vue .


Global
stepping
SSP.CTrace
«reference»
Global
«reference»
stepping
«reference»
SSP.CTrace
wmainCRTStartup()
__tmainCRTStartup()
wmain()
Run()
Func4()
Func5()
Func6()
Func7()
(from Native)
(from Native)
(from Native)
Start Page
stepping.cpp
Stack
Trace.cpp
Stack
Trace.h
Stack

Inspecter Mémoire de Processus

À l'aide de Memory Viewer, vous pouvez afficher les valeurs brutes de la mémoire en hexadécimal et ASCII. Vous pouvez définir manuellement l'adresse mémoire dans le champ 'Adresse' (en haut à droite), ou cliquez-droit sur une variable dans la fenêtre Locals ou Observateurs et sélectionner l'option ' Vue Memory at Address'.

Accéder

Ruban	Exécuter > Windows > Visionneuse de mémoire
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution : | Visionneuse de mémoire Depuis la fenêtre Locals ou Observateurs : Cliquez-droit sur une variable | Mémoire Vue à l'adresse

Notes

- Le Memory Viewer est disponible pour le débogage des applications Microsoft Native Code (C, C++, VB) exécutées sous Windows ou dans WINE sous Linux.

Afficher Modules Chargés

Pour les applications .NET et Windows natives, vous pouvez lister les DLL chargées par le processus débogué, à l'aide de la fenêtre Modules. Cette liste peut également inclure les fichiers symboliques associés (fichiers PDB) utilisés par le débogueur.

Accéder

Ruban	Exécuter > Windows > Modules

Affichage de la fenêtre des modules

Colonne	Description
Chemin	Affiche le chemin du fichier du module chargé.
Charger l'adresse	Affiche l'adresse mémoire de base du module chargé.
Date modifiée	Affiche la date du fichier local et l'heure à laquelle le module a été modifié.
Symboles Déboguer	Montre: • Le type de symboles de débogage • Si les informations de débogage sont présentes dans le module, et • Si les informations de ligne sont présentes pour le module (obligatoire pour le débogage)
Correspondance du fichier de symboles	Indique la validité du fichier de symboles ; si la valeur est fausse, le fichier de symboles est obsolète.
Chemin du symbole	Affiche le chemin du fichier de symboles, qui doit être présent pour que le débogage fonctionne.
Date modifiée	Affiche la date et l'heure du fichier local de création du fichier de symboles.

Traiter Exceptions à Première Chance

Accéder

Ruban	Exécuter > Outils > Débogueur > Traiter Exceptions à Première Chance
Autre	Barre d'outils de la fenêtre Déboguer : | Traiter Exceptions à Première Chance

Éléments de traitement

Élément	Description
Processus Déboguer	Lorsqu'une application est en cours de débogage et que le débogueur est informé d'une exception, l'application est mise en pause et le débogueur répond de la manière pour laquelle il est configuré ; soit : • Reprend l'application et laisse l'exception à l'application à gérer, ou • Maintient l'application suspendue et transmet l'exception aux routines appropriées pour une résolution automatique ou une intervention manuelle
Exceptions de la deuxième chance	Le débogueur Enterprise Architect utilise par défaut le premier comportement répertorié. Si l'application peut gérer l'exception, elle continue le traitement ; s'il ne peut pas gérer l'exception, le débogueur est à nouveau averti et cette fois il doit suspendre l'application et résoudre la condition d'exception. Dans ce comportement, étant donné que le débogueur a rencontré l'exception à deux reprises, il s'agit d'une exception de seconde chance ; dans ce cas, si l'exception n'interrompt pas l'exécution, elle est ignorée et vous évitez de passer du temps sur des conditions qui n'impactent pas le résultat global du traitement. Vous pouvez travailler de cette façon sur des systèmes vastes ou complexes qui impliquent invariablement des conditions d'exception quelque part dans les chemins de traitement.
Exceptions de première chance	Toutefois, si vous souhaitez examiner chaque exception qui se produit dès qu'elle se produit, vous pouvez configurer le débogueur pour qu'il adopte le deuxième comportement. Étant donné que le débogueur répond à l'exception au premier contact, on parle d'exception de première chance. Vous pouvez travailler de cette façon avec des fonctions ou des routines individuelles qui doivent fonctionner proprement ou pas du tout.
Sélection	Sélectionnez l'option ' Traiter Exceptions à Première Chance ' pour déboguer les exceptions au premier contact. Désélectionnez l'option permettant de traiter les exceptions uniquement si l'application échoue lorsqu'elles se produisent.

Débogueur juste à temps

Vous pouvez enregistrer le débogueur Enterprise Architect en tant que débogueur juste-à-temps du système d'exploitation, à appeler lorsqu'une application exécutée en dehors Enterprise Architect sur le système rencontre une exception ou se bloque. Lorsque vous le faites, un crash d'application entraînera l'ouverture Enterprise Architect et l'affichage de la source et de la raison du crash.

Accéder

Ruban	Exécuter > Outils > Débogueur > Définir comme Débogueur JIT

Créer une application

Cette rubrique explique comment exécuter un script Build sur votre application, dans Enterprise Architect .

Accéder

Ruban	Exécuter > Source > Construire > Construire
Raccourcis Clavier	Ctrl+Maj+F12
Autre	Barre d'outils 'Construire' > Fenêtre Analyseur d'Exécution |

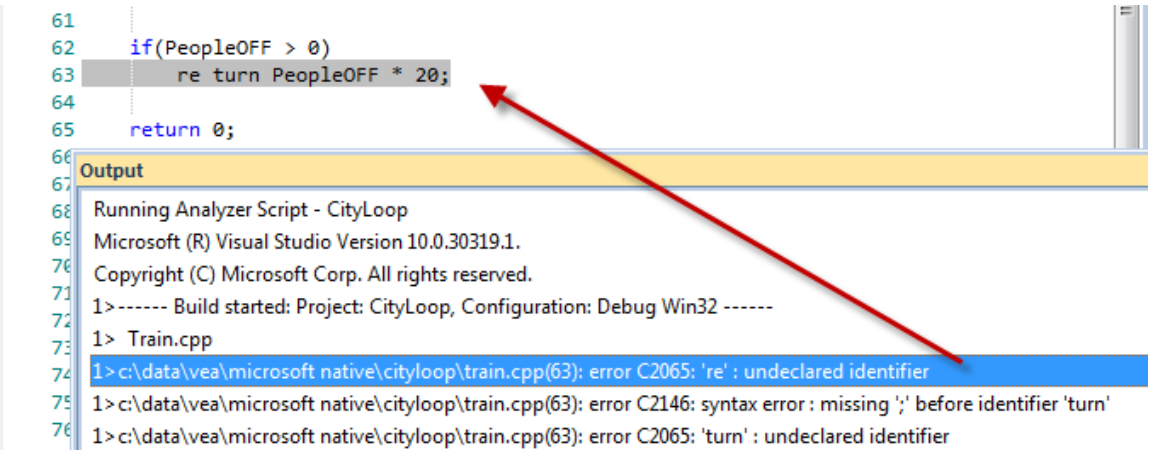
Action

Lorsque vous sélectionnez l'option 'Build', il exécute la commande 'Build' dans le script sélectionné dans la fenêtre Analyseur d'Exécution . La progression et le résultat de l'opération de génération sont affichés dans l'onglet « Construire » de la fenêtre Sortie système.

Vous pouvez rapidement visiter la ligne de code pour toute erreur de compilation apparaissant en double-cliquant sur l'erreur.

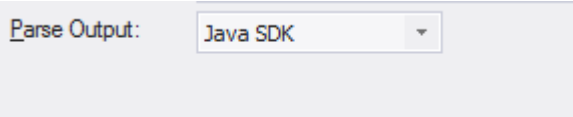
Localiser les erreurs Compilateur dans le code

Lorsque vous créez une application à l'aide d'un script Analyzer, la sortie du compilateur est enregistrée dans la fenêtre Sortie système. Vous pouvez double-cliquer sur n'importe quel message d'erreur qui apparaît ici et accéder au code source. Lorsque vous le faites, le curseur se positionne sur la ligne contenant l'erreur.



Conseil

Si la sortie est manquante, vérifiez qu'un analyseur de langage est mentionné dans le script de l'analyseur (Maj+F12).

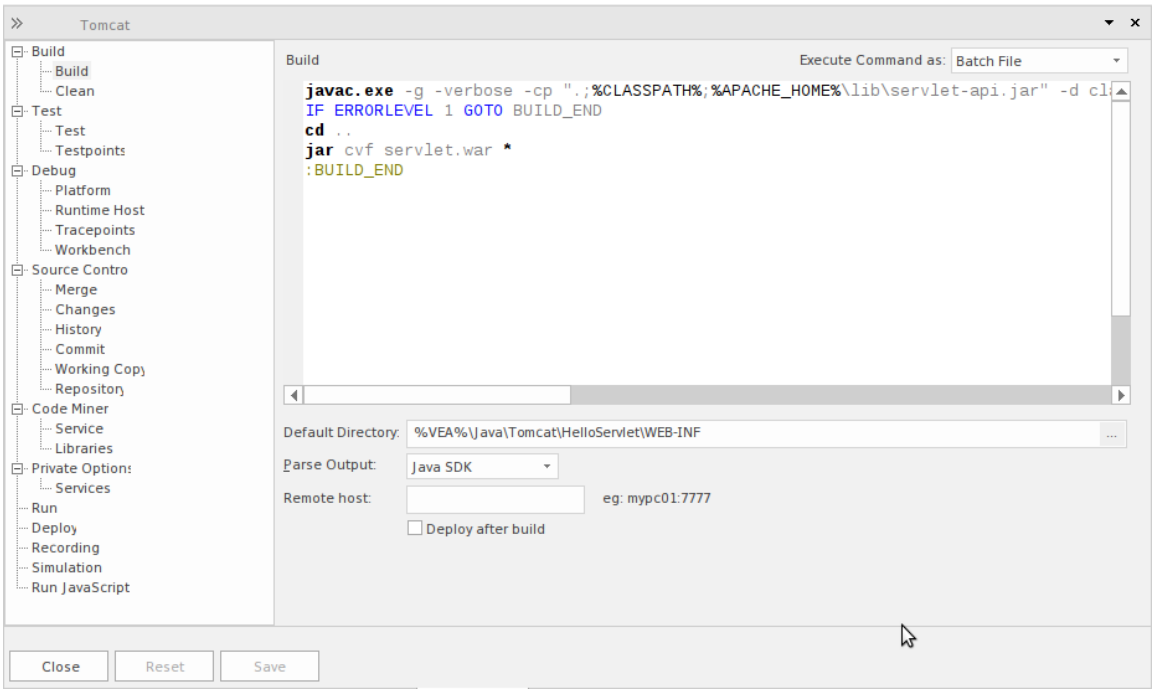


Accéder

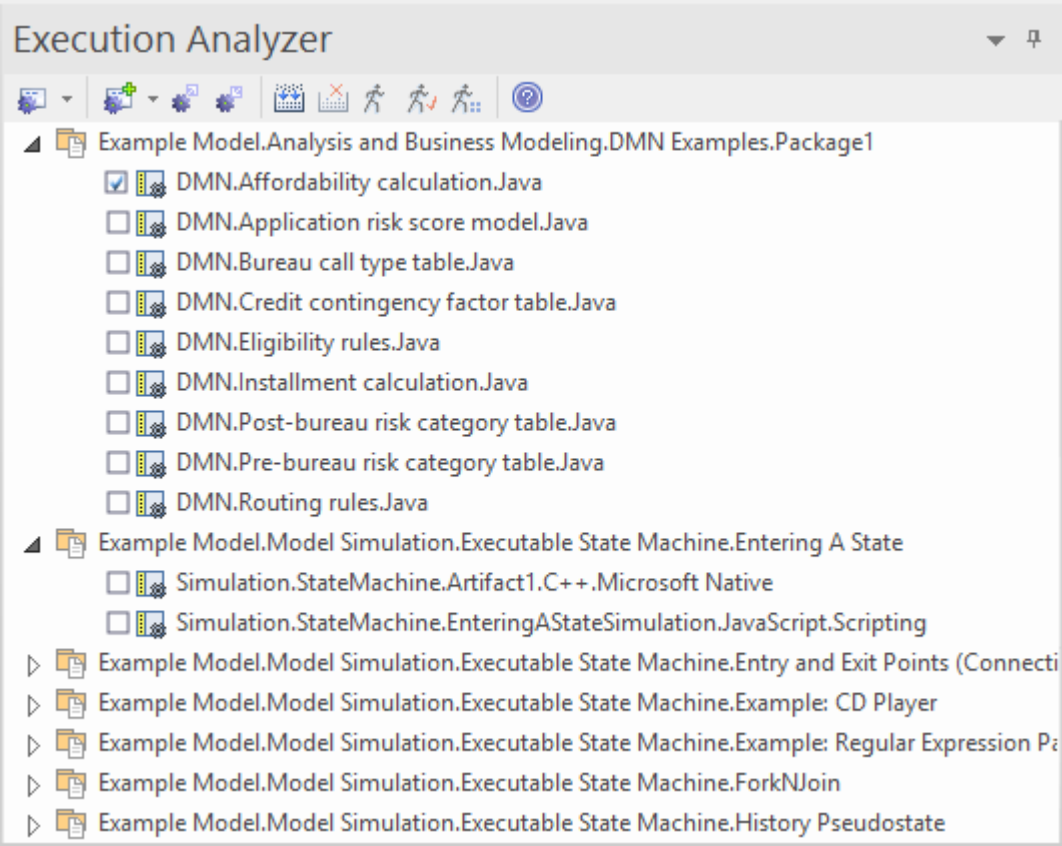
Ruban	Démarrer > Toutes Windows > Conception > Explorer > Système
Raccourcis Clavier	Ctrl+Maj+8

Scripts d'Analyseur

Scripts d'Analyseur sont utilisés par l' Analyseur d'Exécution . Vous n'avez pas à vous soucier de leur création. Il ne s'agit pas du même type de script que JavaScript ou PHP, mais ils sont gérés à l'aide d'une interface utilisateur familière - une arborescence - et vous pouvez rapidement localiser la fonctionnalité à modifier. Scripts d'Analyseur peuvent être partagés par les utilisateurs d'un modèle communautaire et sont facilement importés et exportés sous forme de fichiers XML.



Un même projet peut avoir plusieurs configurations et celles-ci peuvent être trouvées regroupées dans la fenêtre Analyzer.



Chaque script d'analyseur est défini pour un Paquetage , les projets peuvent donc coexister sans problème. Dans de nombreuses organisations, les procédures de gestion des systèmes sont distribuées et varient d'un individu à l'autre et d'un groupe à l'autre. Scripts d'Analyseur dans un modèle Enterprise Architect peuvent apporter une certaine tranquillité d'esprit à ces organisations, en faisant confiance à une procédure unique, partagée et responsable pour créer et déployer toute variété de configurations. Tous les aspects d'un script sont facultatifs. Vous pouvez, par exemple, déboguer sans un ; cependant, en quelques lignes, ils peuvent activer ces fonctionnalités utiles :

- Bâtiment
- Tester
- Débogage
- Enregistrement
- Exécution
- Déploiement
- Simulation

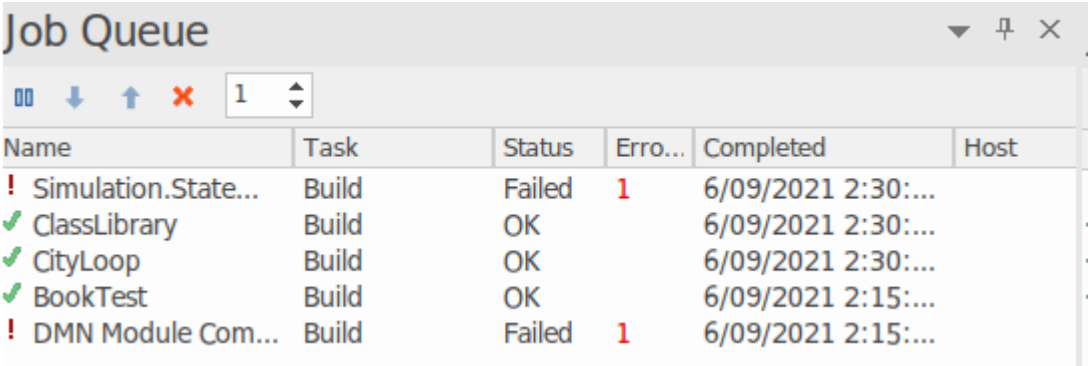
Exécution de scripts à distance

Diverses sections d'Analyzer Script telles que Build et Exécuter fournissent un champ « Hôte distant ». Ce champ permet de décrire l'ordinateur sur lequel le script doit exécuter . Pour utiliser cette fonctionnalité , le service Sparx Satellite doit être exécuté sur la machine. Le format de ce champ est *nom d'hôte:port,* où *nom d'hôte* est l'adresse IP ou le nom de réseau d'une machine Windows ou Linux et *port* est le numéro de port sur lequel le service Satellite écoute. L'objectif principal de cette fonctionnalité est de permettre à un utilisateur d' Enterprise Architect fonctionnant sous Linux d'exécuter des commandes natives de Linux.

La fenêtre de la file d'attente des tâches

La fenêtre Job Queue rationalise le processus de travail avec Scripts d'Analyseur , qui étaient à l'origine travaillés individuellement et si aucun autre script n'était en cours d'exécution. Dans les versions d' Enterprise Architect à partir de la version 16.0, lorsqu'une option du menu contextuel d'Analyzer Script est exécutée (par exemple, « Build »), elle est placée dans une file d'attente de tâches ; plusieurs tâches peuvent être mises en file d'attente et d'autres travaux effectués pendant le traitement des tâches.

Le nom du script Analyzer est utilisé comme nom de la tâche. Un script Analyzer peut avoir plusieurs sections, telles que Build, Test , Exécuter et Deploy, et chaque section est affectée en tant que tâche du travail.



La sortie de chaque tâche exécutée à partir de la fenêtre File d'attente des tâches est capturée dans l'onglet « Historique des tâches » de la fenêtre Sortie système.

Accéder

Ruban	Exécuter > Outils > Analyseur > File d'attente des tâches Vue Démarrer > Toutes Windows > Conception > Explorer > Système > Historique des tâches

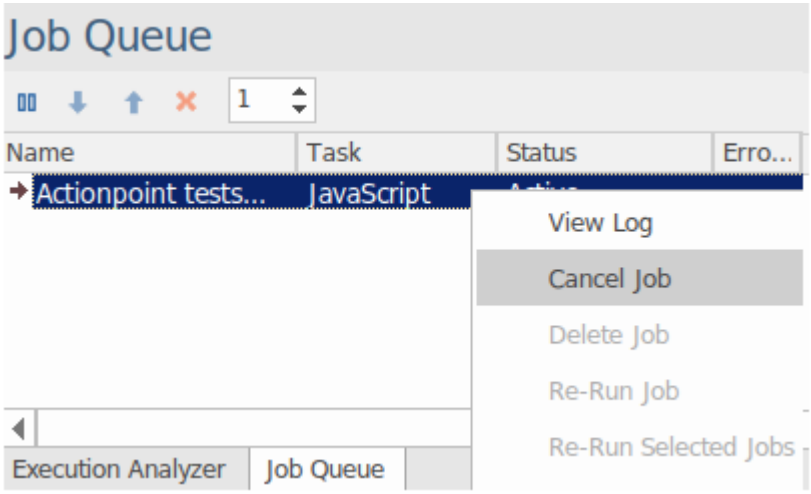
Colonnes de la file d'attente des travaux

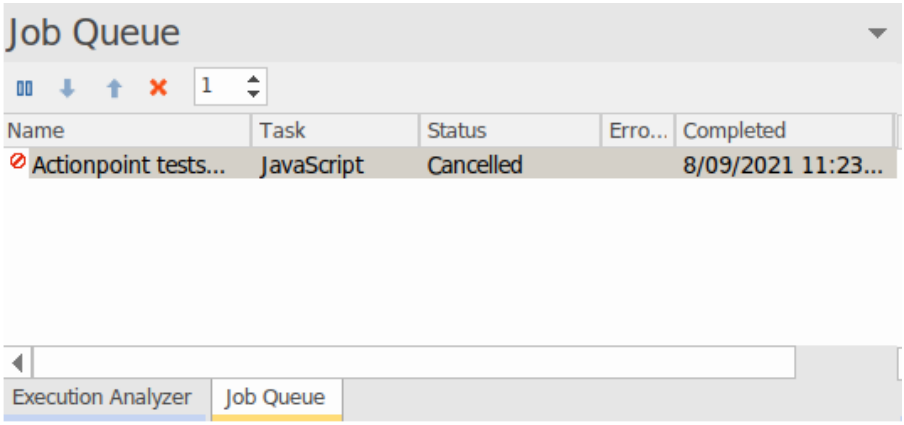
Colonne	Description
Nom	Nom du script Analyzer pour lequel la tâche a été ajoutée à la file d'attente des tâches. Si le travail a été exécuté, le nom sera précédé d'une coche (pour une réussite) ou d'un point d'exclamation (pour un échec).
Tâche	Section du script Analyzer que la tâche est en train d'exécuter - par exemple, Build ou Deploy.
Statut	L'état d'achèvement du travail - si le travail s'est terminé avec succès (« OK ») ou a échoué.
les erreurs	Si le travail a été exécuté et a échoué, le nombre d'erreurs survenues.
Complété	La date et l'heure auxquelles le travail s'est terminé.

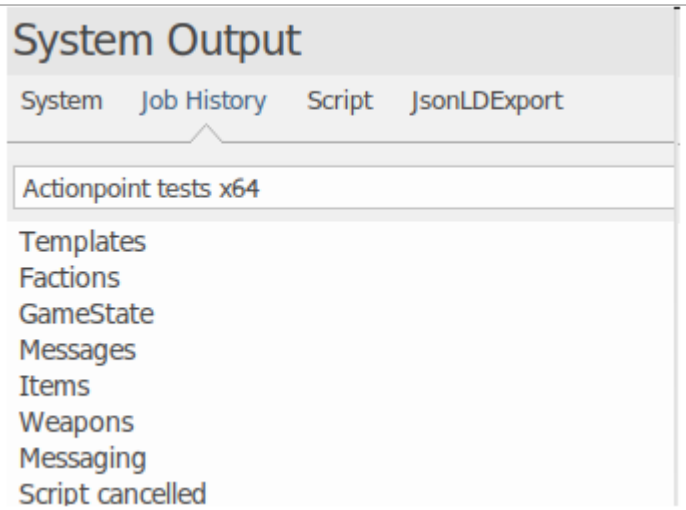
Hôte	Si le travail est exécuter à distance, l'adresse IP ou le nom d'hôte de la machine distante.
Reçu	Si le travail est exécuter à distance, tout message de retour de la machine hôte.

Options Menu Contexte

Cliquez-droit sur le nom d'un travail, ou sur le fond de la fenêtre, pour afficher le menu contextuel 'Job Queue'.

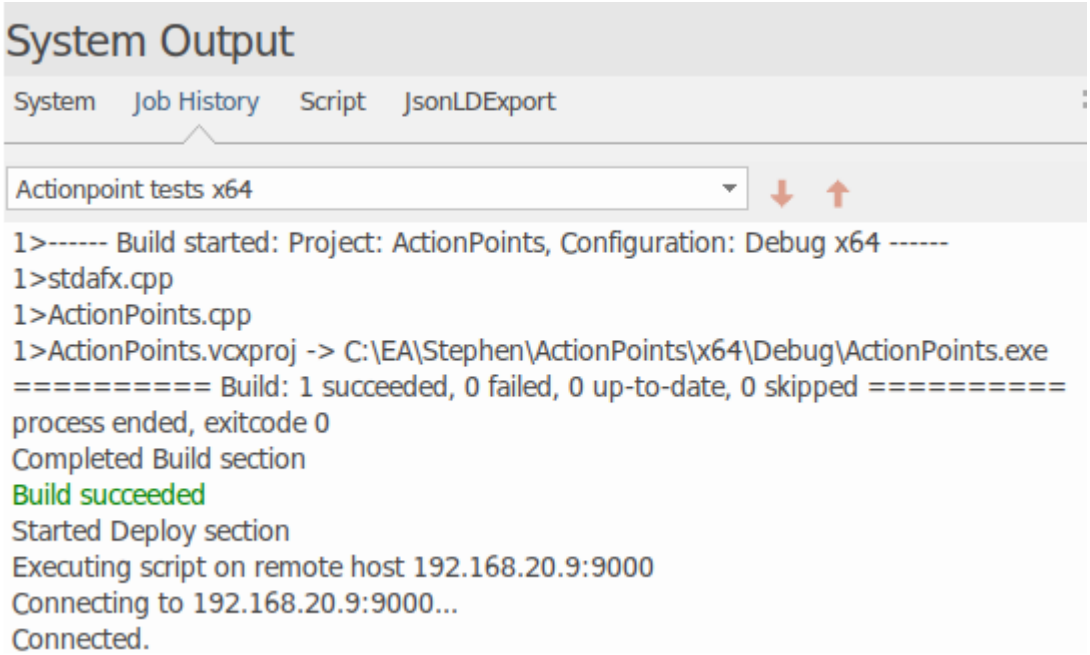


Option	Description
Journal Vue	Sélectionnez cette option pour afficher l'onglet « Historique des tâches » de la fenêtre Sortie système, pour une tâche qui a été exécutée.
Annuler le travail	Cliquez-droit sur un nom de tâche et cliquez sur cette option, si nécessaire et à tout moment, pour annuler la tâche JavaScript sélectionnée. L'annulation est indiquée dans la fenêtre File d'attente des travaux par une icône « aucune entrée » à côté du nom du travail.  Dans l'onglet « Historique des tâches » de la fenêtre Sortie système, la sortie se termine par un message « Script annulé ».

	
Supprimer le travail	Cliquez sur une tâche qui n'a pas encore démarré et sélectionnez cette option pour la supprimer de la file d'attente des tâches.
Réexécuter le travail	Sélectionnez cette option pour exécuter à nouveau le travail terminé sélectionné.
Réexécuter les tâches sélectionnées	(Ctrl+clic sur un certain nombre de tâches requises.) Sélectionnez cette option pour exécuter à nouveau toutes les tâches terminées sélectionnées.
Réexécuter les tâches terminées	Sélectionnez cette option pour exécuter à nouveau tous les travaux terminés dans la liste actuelle.

L'onglet Historique des tâches

La sortie de chaque tâche exécutée à partir de la fenêtre File d'attente des tâches est capturée dans l'onglet « Historique des tâches » de la fenêtre Sortie système. À partir de là, n'importe quel log de travail peut être consulté à votre discrétion, en le sélectionnant dans la liste déroulante de la barre d'outils. Si le travail a échoué et qu'il y a des messages d'erreur, vous pouvez passer d'un message à l'autre à l'aide des icônes en forme de flèche rouge. Ces icônes sont désactivées s'il n'y a aucun message d'erreur actif.



Options de la barre d'outils de la file d'attente des travaux

Option	Description
	Cliquez sur le nom d'un travail et cliquez sur cette icône pour suspendre ou reprendre le travail sélectionné.
	Cliquez sur le nom d'un travail et sur l'une de ces deux flèches pour déplacer le travail vers le haut ou vers le bas dans la file d'attente des travaux, le rendant ainsi plus tôt ou plus tard dans l'ordre de traitement.
	Cliquez sur le nom d'un travail qui n'a pas encore démarré, puis sur cette icône pour supprimer ce travail de la fenêtre File d'attente des travaux.
1	Cliquez sur la flèche vers le haut ou vers le bas pour définir le nombre de tâches pouvant être exécuter simultanément, jusqu'à un maximum de 8. Le nombre est par défaut de 1 afin que la file d'attente des tâches traite les tâches une par une, premier entré, premier sorti.

Script Code Miner

Le système Code Miner utilise un ensemble de bases de données pour fournir un accès rapide et complet aux informations dérivées du code source existant. Les fonctionnalités Intelli-sense des éditeurs de code d' Enterprise Architect et ses outils de recherche peuvent utiliser les informations extraites de ces bases de données.

Grâce aux pages de script Code Miner , vous pouvez spécifier les bases de données Code Miner à utiliser avec un projet particulier et vous pouvez créer, mettre à jour et ajouter de nouvelles bases de données à la bibliothèque Code Miner . La page « Services » vous permet de spécifier une bibliothèque Code Miner locale ou que vous souhaitez accéder à la bibliothèque disponible via le service Sparx Intel.

Différents détails Code Miner peuvent être spécifiés pour chaque script Analyzer, de sorte que les bibliothèques Code Miner utilisées sont déterminées par le script Analyzer *actif* .

Accéder

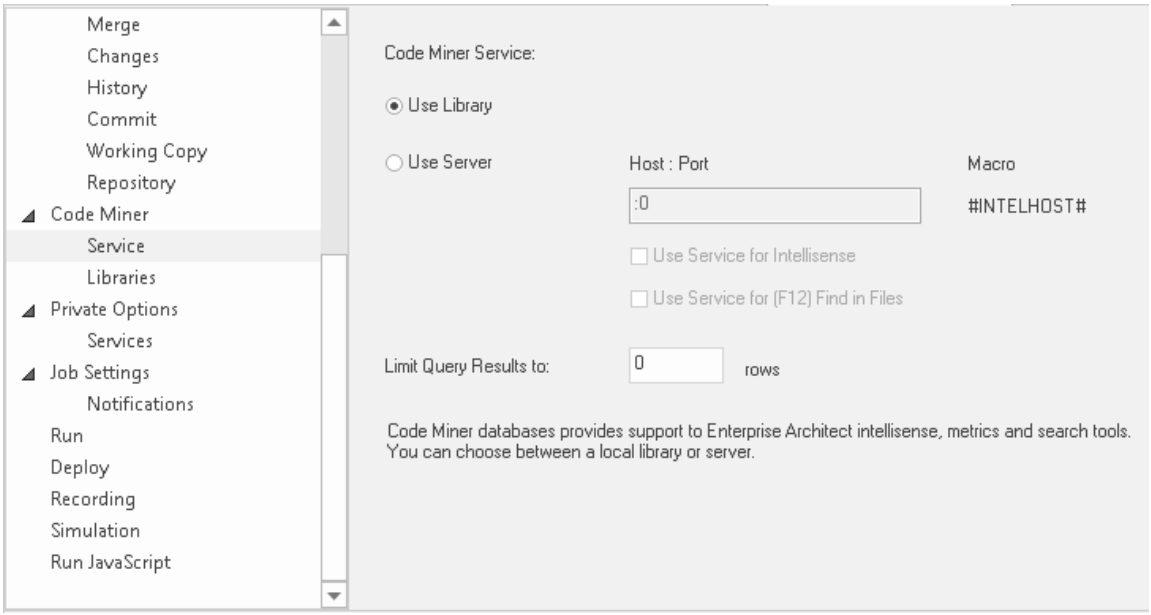
Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page ' Code Miner ' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, puis sélectionnez la page ' Code Miner '

Ruban	Exécuter > Outils > Analyser > Vue Scripts d'Analyseur > Double-cliquez sur Nom du script > Code Miner > Service Développer > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur > Double-cliquez sur Nom du script > Code Miner > Service
Raccourcis Clavier	Maj+F12

Service Intel Sparx

Une Bibliothèque Code Miner peut être utilisée localement ou elle peut être déployée sur un emplacement de serveur où elle peut desservir plusieurs clients. Vous sélectionnez le scénario à utiliser sur la page « Code Miner Service » du script Analyzer.

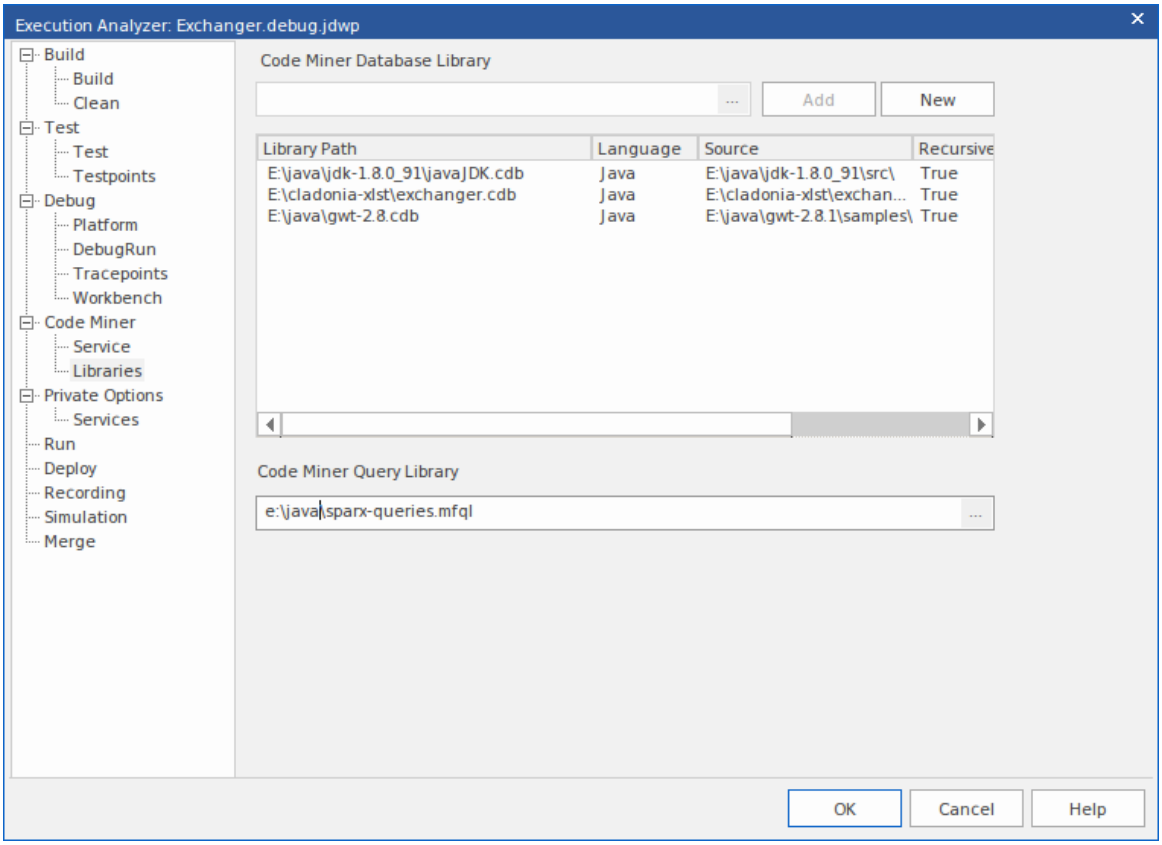


Bibliothèques Code Miner

Les bibliothèques de bases de données Code Miner sont une collection de bases de données contenant des informations dérivées du code source. Ensemble, ces bases de données forment la Bibliothèque Code Miner utilisée par les fonctionnalités Intelli-sense d' Enterprise Architect . Habituellement, une bibliothèque est créée pour chaque framework ou projet. La page Bibliothèques Code Miner permet de créer de nouvelles bases de données et d'ajouter, de mettre à jour ou de supprimer des bases de données existantes d'une bibliothèque.

Les bibliothèques Query Code Miner sont un ensemble de fonctions, écrites dans le langage mFQL de Code Miner , regroupées dans un seul fichier source.

La Bibliothèque de base de données Code Miner et Bibliothèque Query pour un script d'analyseur donné sont spécifiées dans le fichier ' Code Miner | Page des bibliothèques de l' Éditeur de Script .


Execution Analyzer: Exchanger.debug.jdwp
Build
Build
Clean
Test
Test
Testpoints
Debug
Platform
DebugRun
Tracepoints
Workbench
Code Miner
Service
Libraries
Private Options
Services
Run
Deploy
Recording
Simulation
Merge
Code Miner Database Library
Add
New
Library Path
Language
Source
Recursive
E:\java\jdk-1.8.0_91\javaJDK.cdb
Java
E:\java\jdk-1.8.0_91\src\
True
E:\cladonia-xlst\exchanger.cdb
Java
E:\cladonia-xlst\exchan...
True
E:\java\gwt-2.8.cdb
Java
E:\java\gwt-2.8.1\samples\
True
Code Miner Query Library
e:\java\sparx-queries.mfql
OK
Cancel
Help

Script de services

La page 'Services' d'un Analyzer Script décrit les ports par défaut utilisés lorsque des scripts sont créés par différentes fonctions Analyseur d'Exécution Visuelle (Import project, Générer Statemachine Exécutable). Vous pouvez mettre à jour n'importe quelle spécification du port sur cette page.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez « Options privées | Page des services ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, puis sélectionnez « Options privées | Page des services

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Fusionner le script

Une commande de fusion dans un script Analyzer donne aux utilisateurs une commande supplémentaire pour effectuer une action. L'action de fusion dépend de vos besoins.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page « Fusionner » ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, puis sélectionnez la page 'Fusionner'

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Source > Fusionner
Raccourcis Clavier	Ctrl+Alt+M

Scripts Enregistrement

La beauté de l'enregistrement ne réside pas vraiment dans le fait que nous obtenons toujours une vue d'ensemble, mais plutôt dans la possibilité de voir une image plus petite qui a une part de vérité à dire. Nous avons tous vu diagrammes Séquence qui ne sont pas très utiles. (*Le même message apparaissant 100 fois de suite sur un diagramme nous dit effectivement quelque chose, mais pas grand-chose.*) Heureusement Enterprise Architect s'occupe de ce premier point grâce à l'utilisation de fragments. Les comportements répétitifs sont identifiés comme Motifs et représentés une fois sous forme de fragment sur le diagramme Séquence . Le fragment est étiqueté en fonction du nombre d'itérations. Bien entendu, l'historique des enregistrements montre toujours l'intégralité de l'historique. Nous avons également besoin d'outils pour nous aider à concentrer l'enregistrement sur des domaines d'intérêt particuliers et à réduire le bruit des autres. Nous pouvons utiliser des filtres pour ce faire. Avec les filtres, vous pouvez exclure toutes les classes, fonctions ou même modules de tout enregistrement. Vous pouvez créer plusieurs ensembles de filtres et les utiliser avec des ensembles de marqueurs pour cibler différents cas d'utilisation.

Accéder

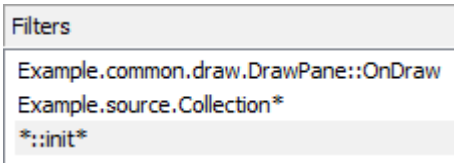
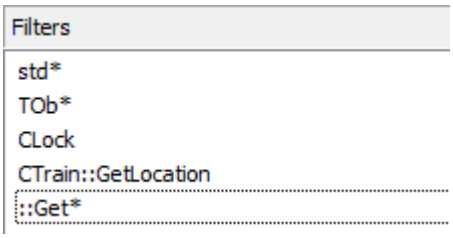
Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script souhaité et sélectionnez la page ' Enregistrement ' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Enregistrement '

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Filtrer les chaînes

Élément	Discussion
Filtration	Si la case 'Enable Filter' est cochée sur la page ' Enregistrement ' de l'Execution Éditeur de Script Analyseur , le débogueur exclut de l'enregistrement les appels aux méthodes correspondantes. La comparaison est sensible à la casse. Pour ajouter une valeur , cliquez sur l'icône 'Nouveau' ("Insérer") dans le coin droit de la zone 'Filtres d'exclusion', et saisissez la string de comparaison ; chaque string de filtre prend la forme : class_name_token ::method_name_token Le class_name_token exclut les appels à toutes les méthodes d'une ou plusieurs classes dont le nom correspond au jeton ; la string peut contenir le caractère générique * (astérisque). Le method_name_token exclut les appels aux méthodes ayant un nom qui correspond au jeton ; là encore, la string peut contenir le caractère générique *. Les deux jetons sont facultatifs ; si aucun jeton de classe n'est présent, le filtre est appliqué uniquement aux fonctions globales ou publiques (c'est-à-dire aux méthodes n'appartenant à aucune classe).
Exemple	Dans cet exemple Java, le débogueur exclurait :

	• Appels à la méthode OnDraw pour la classe Exemple.common.draw.DrawPane • Appels à n'importe quelle méthode de n'importe quelle classe ayant un nom commençant par Sample.source.Collection • Appels à n'importe quel constructeur pour n'importe quelle classe (comme <clint> et <init>)  Dans cet exemple de code natif, le débogueur exclurait : • Appels effectués vers l'espace de noms Standard Gabarit Bibliothèque • Appels vers n'importe quelle classe commençant par TOb • Appels à n'importe quelle méthode de Class CLock • Appels à la méthode GetLocation pour la classe CTrain • Appels à n'importe quelle fonction globale ou publique dont le nom commence par Get 

Filtres

Utiliser l'entrée de filtre	À filtrer
::Get*	Toutes les fonctions publiques ayant un nom commençant par « Get » de la session d'enregistrement (par exemple, GetClientRect dans l'API Windows).
::Get	Toutes les méthodes commençant par « Get » dans n'importe quelle classe.
CClass::Get*	Toutes les méthodes commençant par Get pour la classe CClass.
Classe C : *	Toutes les méthodes pour la classe CClass.
ATL* norme*	Toutes les méthodes pour les classes appartenant aux bibliothèques Standard Gabarit et Actif Gabarit .
CClass::GetName	La ou les méthodes spécifiques GetName pour la classe CClass.

Déployer le script

Ces sections expliquent comment créer un script de commande pour déployer le Paquetage actuel. Le script peut être exécuté en sélectionnant l'option de ruban « Exécuter > Source > Construire > Déployer » ou en appuyant sur Ctrl+Maj+Alt+F12.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page « Déployer » ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, puis sélectionnez la page 'Déployer'

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Ctrl+Maj+Alt+F12

Actions

Action	Détail
Exécutez la commande en tant que :	**Process** If the deployment is handled externally, enter the path to the program or batch file to run, followed by any parameters; the program is launched in a separate process. Example: C:\apache-ant-1.7.1\bin\ant.cmd myproject deploy **Batch File** When using this option, you can enter multiple commands that are then executed as a single script in a command console; you have access to any environment variables available in a standard command console. Example: @echo on IF NOT EXIST "%1%" GOTO DEPLOY_NOWAR IF "%APACHE_HOME%" == "" GOTO DEPLOY_NOAPACHE xcopy /L "%1%" "%APACHE_HOME%\webapps" GOTO DEPLOY_END rem rem NO WAR FILE rem :DEPLOY_NOWAR echo "%1% WAR file not found"

	GOTO DEPLOY_END rem rem NO APACHE ENVIRONMENT VARIABLE rem :DEPLOY_NOAPACHE echo "APACHE_HOME environment variable not found" :DEPLOY_END pause
Analyser la sortie	La sélection d'un Parser dans la liste entraîne la capture de la sortie du script de déploiement ; la sortie est analysée selon la syntaxe sélectionnée dans la liste. Pour afficher la fenêtre Sortie système, sélectionnez l'option de ruban « Démarrer > Toutes Windows > Conception > Explorer > Système ».

Exécuter Script

Cette section décrit comment créer une commande pour exécuter votre code exécutable.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script souhaité et sélectionnez la page ' Exécuter ' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Exécuter '

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Ctrl+Alt+N

Éléments de script

Élément	Description
Commande	Il s'agit de la commande qui est exécutée lorsque vous sélectionnez l'option de ruban 'Execute > Exécuter > Démarrer ' ; dans sa forme la plus simple, le script contiendrait l'emplacement et le nom du fichier à exécuter .
Exemples	Ces deux exemples montrent des scripts configurés pour exécuter une application .Net et Java dans Enterprise Architect . .Filet: C:\benchmark\cpp\example_net_1\release\example.exe Java: client La commande répertoriée dans ce champ est exécutée comme si elle provenait de l' prompt de commande ; par conséquent, si le chemin de l'exécutable ou des arguments contiennent des espaces, ils doivent être entourés de guillemets.

Notes

- Enterprise Architect offre la possibilité de démarrer votre application normalement OR avec le débogage à partir du même script ; le menu 'Analyzer' propose des options distinctes pour démarrer un exécuter normal et un exécuter de débogage

Script Débogage

Le processus de configuration de la section Déboguer d'un script Analyzer est généralement une affaire ponctuelle qui doit rarement être revisitée. Ainsi, une fois que votre script fonctionnera, vous n'aurez probablement plus à y penser. Les détails que vous fournissez ne sont pas compliqués, mais définir un script donne accès à de nombreux avantages tels que :

- Débogage
- Enregistrement diagramme Séquence
- Statemachine Exécutable exécution et simulation
- Test la création et l'enregistrement de domaines
- Profilage Comportementale des processus sur une variété d'environnements d'exécution

Tout ce que vous avez à faire est de sélectionner la plateforme appropriée et de saisir quelques détails de base. Les plates-formes de débogage que vous pouvez utiliser incluent :

- Java
- Protocole Java Déboguer Wire (JDWP)
- Débogueur Microsoft .NET
- Débogueur Microsoft Native Code (C++, C, VB)
- Mono
- Le Débogueur PHP
- Le Débogueur GNU (GDB)

Accéder

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	F6

Notes

- Un script Analyzer n'est pas nécessaire pour déboguer les scripts du modèle Enterprise Architect tels que JavaScript ou VBScript.

Exigences Spécifiques au Système d'Exploitation

Le débogueur Enterprise Architect est capable de fonctionner sur un certain nombre de plates-formes différentes. Ce tableau décrit les exigences individuelles pour le débogage sur chaque plateforme.

Plateformes

Plate-forme	Détail
.NET	• Microsoft™ .NET Frameworks 4.0, 3.5 et 2.0 • Langage support : C, C# , C++, J#, VB.NET
Java	• Kit de développement Java SE d'Oracle™ (version 5.0 minimum) (JDK 32 bits ou 64 bits) L'architecture Java Platform Débogueur (JPDA) a été introduite dans Java SE version 5.0. Le JPDA fournit deux protocoles pour le débogage ; l'interface Java Virtual Machine Tools (JVMTI) et le protocole Java Déboguer Wire (JDWP). Le débogueur d' Enterprise Architect supporte les deux protocoles.
Débogueur GNU (GDB)	Enterprise Architect supporte le débogage à l'aide du Débogueur GNU, qui vous permet de déboguer vos applications sous Linux localement ou à distance. Nécessite GDB version 7.0 ou supérieure. Le chemin du fichier de code source ne doit pas contenir d'espaces.
Windows pour les applications natives	Enterprise Architect supporte le débogage du code natif (C, C++ et Visual Basic) compilé avec le compilateur Microsoft™ où un fichier PDB associé est disponible.
PHP	Enterprise Architect vous permet d'effectuer le débogage local et à distance des scripts PHP sur les serveurs Web. Nécessite que le serveur Web soit configuré pour support PHP. Nécessite que PHP soit configuré pour support XDebug PHP (extension PHP tierce).

Notes

- La facilité de débogage est disponible dans toutes les éditions d' Enterprise Architect

Systèmes d'Exploitation compatibles UAC

Le système d'exploitation Microsoft Windows 7 fournit un contrôle de compte d'utilisateur (UAC) pour gérer la sécurité des applications.

L' Analyseur d'Exécution Visuelle Enterprise Architect est conforme à l'UAC et les utilisateurs de systèmes compatibles UAC peuvent effectuer des opérations avec l' Analyseur d'Exécution Visuelle et facilités associées sous des comptes qui sont membres uniquement du groupe Utilisateurs.

Toutefois, lors de l'attachement à des processus exécutés en tant que services sur un système d'exploitation compatible UAC, il peut être nécessaire de log en tant qu'administrateur.

Connectez-vous en tant qu'administrateur

Étape	Action
1	Avant d' exécuter Enterprise Architect , cliquez-droit sur l'icône Enterprise Architect sur le bureau et sélectionnez l'option ' Exécuter en tant qu'administrateur'.

Alternativement

Modifiez ou créez un lien vers Enterprise Architect et configurez le lien pour exécuter en tant qu'administrateur.

Étape	Action
1	Cliquez-droit sur l'icône Enterprise Architect et sélectionnez l'option ' Propriétés '. La dialogue « Propriétés » Enterprise Architect s'affiche.
2	Cliquez sur le bouton Avancé. La dialogue ' Propriétés avancées' s'affiche.
3	Cochez la case « Exécuter en tant qu'administrateur ».
4	Cliquez sur le bouton OK , puis à nouveau sur la dialogue ' Enterprise Architect Propriétés '.

Débogage WINE

Configurer Enterprise Architect pour déboguer sous WINE

Étape	Action
1	En ligne de commande, exécuter $ winecfg.
2	Sélectionnez l'onglet « Applications ». Ajoutez l'exécutable Enterprise Architect « EA.exe » à partir du dossier d'installation Enterprise Architect . Ajoutez ensuite ces programmes depuis les sous-répertoires VEA : • SSampler32.exe • SSampler64.exe • SSProfiler32.exe • SSProfiler64.exe
3	Sélectionnez chaque programme tour à tour, puis passez à l'onglet « Bibliothèques ». Assurez-vous que ces valeurs sont répertoriées avec une priorité (native et intégrée) : • dbghelp • msxml4 • msxml6
4	Copiez le code source de l'application ainsi que les exécutables dans votre bouteille. Le chemin doit être le même que celui de la version compilée ; c'est-à-dire: Si la source Windows = C:\Source\SampleApp, sous Crossover, il doit s'agir de C:\Source\SampleApp.
5	Copiez tous les assemblys côte à côte utilisés par l'application.

Autorisations

Une installation d' Enterprise Architect contient des programmes Linux natifs qui fournissent des services de création et de débogage à Enterprise Architect sous Wine. Ces programmes doivent être vérifiés à l'aide du système de fichiers ou du shell Linux pour garantir qu'ils disposent de l'autorisation « Exécuter » définie de manière appropriée. Les programmes se trouvent dans le sous-répertoire "VEA/x86/linux" de l'installation Enterprise Architect .

Exceptions de violation d'accès

En raison de la manière dont WINE gère le dessin direct et l'accès aux données DIB, une option supplémentaire est fournie dans le menu déroulant de la barre d'outils de la fenêtre Déboguer pour ignorer ou traiter les exceptions de violation d'accès émises lorsque votre programme accède directement aux données DIB.

Sélectionnez cette option pour détecter les violations d'accès réelles (inattendues) ; désélectionnez-le pour ignorer les violations attendues.

Comme le débogueur ne peut pas faire la distinction entre les violations attendues et inattendues, vous devrez peut-être recourir à des essais et des erreurs pour capturer et inspecter les véritables plantages du programme.

Notes

- Si WINE plante, les traces arrière peuvent ne pas être correctes
- Si vous utilisez MFC, n'oubliez pas de copier les assemblys de débogage côte à côte dans le répertoire C:\window\winsxs.
- Pour ajouter un chemin Windows à WINE , modifiez l'entrée de registre : HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\Environment

Java

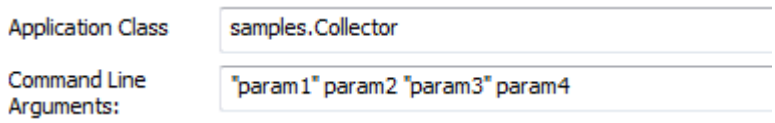
Cette section décrit comment configurer Enterprise Architect pour le débogage des applications Java et des serveurs Web.

Configuration générale pour Java

La configuration générale pour le débogage des applications Java supporte deux options :

- Déboguer une application
- Attacher à une application en cours d'exécution

Option 1 - Déboguer une application

Champ	Action
Débogueur	Sélectionnez Java.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.
Mode	Sélectionnez Exécuter .
Répertoire par défaut	Ce chemin est ajouté à la propriété du chemin de classe lors de la création de la Virtual Machine Java.
Classe d'application	Identifiez le nom de classe complet à déboguer ; la Classe doit avoir une méthode déclarée avec cette signature : public static void main(String ()); 
Arguments de ligne de commande	Spécifiez tous les paramètres à transmettre à la méthode principale de la classe d'application. Les paramètres contenant des espaces doivent être entourés de guillemets doubles.
Options Virtual Machine Java	Spécifiez les options de ligne de commande pour la création Virtual Machine . Vous devez également fournir un paramètre pour Java Runtime Environment (JRE) comme chemin d'accès au fichier jvm.dll ; il s'agit de la DLL fournie dans le cadre de l'environnement d'exécution ou du JDK de Sun Microsystems ™ . Le paramètre JRE peut être : • Un chemin local défini par Enterprise Architect • Un chemin de fichier absolu (sans guillemets doubles) vers le dossier d'installation du JDK Java à utiliser pour le débogage Le paramètre JRE doit pointer vers le dossier d'installation du JDK Java. Une installation JDK est nécessaire pour que le débogage réussisse. Le JRE ne doit pas pointer vers l'installation de l'environnement d'exécution Java public, si celui-ci est installé. Les variables d'environnement peuvent être utilisées lors de la spécification des options de démarrage de la VM, telles que le chemin de classe. Par exemple, en utilisant : • Un chemin local JAVA Enterprise Architect et un chemin de classe de variable d'environnement :

	Java Virtual Machine Options: JRE=%JAVA%,-Djava.class.path=%classpath%;.; • Ou un chemin absolu vers le répertoire d'installation du JDK et un chemin de classe de variable d'environnement : Java Virtual Machine Options: JRE=C:\Program Files (x86)\Java\jdk1.7.0,-Djava.class.path=%classpath%;.; Dans ces deux exemples, le débogueur va créer une machine virtuelle en utilisant le JDK situé à la valeur du paramètre JRE. Si aucun chemin de classe n'est spécifié, le débogueur crée toujours la machine virtuelle avec une propriété de chemin de classe égale à n'importe quel chemin contenu dans la variable d'environnement plus le chemin entré dans le répertoire de travail par défaut de ce script. Si les fichiers source et les fichiers .class se trouvent sous des arborescences de répertoires différentes, la propriété classpath DOIT inclure à la fois le(s) chemin(s) racine vers la source et le(s) chemin(s) racine vers les fichiers de classe binaire.

Option 2 - Attacher à Virtual Machine

Il y a très peu de choses à spécifier lors de la connexion à une VM ; cependant, l'agent de débogage Sparx Systems doit être chargé sur la machine virtuelle.

Champ	Action
Débogueur	Sélectionnez Java
Mode	Sélectionnez Attacher à Virtual Machine

Techniques avancées

En plus des techniques de débogage Java standard, vous pouvez :

- Attach to Virtual Machine
- Internet Browser Java Applets

Attacher à Virtual Machine

Vous pouvez déboguer une application Java en vous connectant à un processus qui héberge une Virtual Machine Java ; vous souhaiterez peut-être faire cela pour vous connecter à un serveur Web tel que Tomcat ou JBOSS.

L'interface Java Virtual Machine Tools de Sun Microsystems est l'API utilisée par Enterprise Architect ; il permet de spécifier un agent de débogage lors de la création de la JVM.

Pour déboguer une JVM en cours d'exécution à partir Enterprise Architect , l'agent de débogage de Sparx Systems doit avoir été spécifié comme option de démarrage de la JVM lors de son démarrage ; La manière dont cela est accompli pour des produits tels que Tomcat et JBOSS doit être fournie par la propre documentation de ce produit.

Pour java.exe, l'option de ligne de commande permettant de charger l'agent de débogage Enterprise Architect pourrait être (en fonction de votre environnement) :

- -agentpath : "c:\program files\sparx Systems\ea\VEA\x86\SSJavaProfiler32"
- -agentpath : "c:\program files (x86)\sparx Systems\ea\VEA\x86\SSJavaProfiler32"
- -agentpath : "c:\program files (x86)\sparx Systems\ea\VEA\x64\SSJavaProfiler64"

L'option appropriée dépendra de votre système d'exploitation et si vous travaillez sur une application 32 bits ou une application 64 bits.

Alternativement, si vous ajoutez le répertoire VEA approprié à votre variable d'environnement PATH, vous pouvez choisir d'utiliser :

- -agentlib:SSJavaProfiler32
- -agentlib:SSJavaProfiler64

Il n'est pas nécessaire de configurer un script Analyzer lorsque vous vous connectez à une Virtual Machine ; vous pouvez simplement utiliser le bouton Attacher sur l'une des barres d'outils de l'analyseur.

Si vous configurez un script Analyzer, seules deux choses doivent être sélectionnées :

- Sélectionnez « Java » comme plate-forme de débogage
- Choisissez l'option ' Attacher à Virtual Machine '

Applet Java Navigateur Internet

Cette rubrique décrit les exigences de configuration et la procédure de débogage des applets Java exécutées dans un navigateur à partir Enterprise Architect .

Attacher au processus du navigateur hébergeant la Virtual Machine Java (JVM) d' Enterprise Architect

Étape	Action
1	Assurez-vous que les binaires du code de l'applet à déboguer ont été créés avec les informations de débogage.
2	Configurez la JVM à l'aide du panneau de configuration Java.
3	Dans le panneau « Paramètres d'exécution de l'applet Java », cliquez sur le bouton Vue .
4	Sur la version installée à utiliser, incluez l'une de ces options dans le champ 'Paramètres d'exécution', selon votre environnement et si vous travaillez sur une application 32 bits ou une application 64 bits : -agentpath : "c:\program files\sparx Systems\ea\VEA\x86\SSJavaProfiler32" -agentpath : "c:\program files (x86)\sparx Systems\ea\VEA\x86\SSJavaProfiler32" -agentpath : "c:\program files (x86)\sparx Systems\ea\VEA\x64\SSJavaProfiler64"
5	Dans ce champ, ajoutez les chemins de classe requis. Au moins un de ces chemins doit inclure le chemin racine des fichiers source à utiliser pour le débogage.
6	Définissez des points d'arrêt.
7	Lancez le navigateur.
8	Attachez-vous au processus du navigateur depuis Enterprise Architect .

Travailler avec Serveurs Web Java

Si vous déboguez des serveurs Web Java tels que JBOSS et Apache Tomcat (configuration du serveur et configuration du service Windows) dans Enterprise Architect , appliquez ces exigences et procédures de configuration.

Note : Les fonctionnalités de débogage et d'enregistrement de l' Analyseur d'Exécution Visuelle ne sont pas supportées pour la plateforme serveur Java 'Weblogic' d'Oracle.

Attacher au processus hébergeant la Virtual Machine Java d' Enterprise Architect

Étape	Action
1	Créez des binaires pour le code du serveur Web à déboguer, avec des informations de débogage.
2	Lancez le serveur avec l'option ' Démarrage Virtual Machine ', décrite dans *Configuration du serveur* .
3	Importez le code source dans Enterprise Architect Modèle ou synchronisez le code existant.
4	Définissez des points d'arrêt.
5	Lancez le client.
6	Attachez-vous au processus depuis Enterprise Architect .

Configuration du serveur

La configuration nécessaire pour que les serveurs web interagissent avec Enterprise Architect doit aborder ces deux points essentiels :

- Toute VM à déboguer, créer ou hébergée par le serveur doit avoir l'option de ligne de commande Sparx Systems Agent spécifiée ou dans l'option de démarrage de la VM (c'est-à-dire : -agentlib:SSJavaProfiler32 ou -agentlib:SSJavaProfiler64)
- Le CLASSPATH, quelle que soit la manière dont il est transmis à la VM, doit spécifier le chemin racine des fichiers sources du Paquetage .

Le débogueur Enterprise Architect utilise la propriété java.class.path dans la VM en cours de débogage pour localiser le fichier source correspondant à un point d'arrêt se produisant dans une classe lors de l'exécution ; par exemple, une Classe à déboguer s'appelle :

abc

Celui-ci se trouve dans le répertoire physique :

C:\source\a\b

Ainsi, pour que le débogage réussisse, CLASSPATH doit contenir le chemin racine :

c:\source

Configuration du script de l'analyseur

A l'aide de l'onglet ' Déboguer ' de la boîte dialogue 'Build Script', créez un script pour le code que vous avez importé et :

- Sélectionnez le bouton radio « Attacher au processus » et, dans le champ en dessous, tapez « Attacher ».
- Dans le champ 'Utiliser Débogueur ', cliquez sur la flèche déroulante et sélectionnez 'Java'

Tous les autres champs sont sans importance ; le champ 'Répertoire' est normalement utilisé en l'absence de toute propriété de chemin de classe.

Exécuter le Débogueur

Les points d'arrêt pourraient afficher un point d'interrogation. Dans ce cas, la classe n'a peut-être pas encore été chargée par la VM. Si le point d'interrogation persiste même après que vous êtes sûr que la classe contenant le point d'arrêt a été chargée, alors soit :

- Les binaires exécutés par le serveur ne sont pas basés sur le code source
- Le débogueur ne peut pas réconcilier le point d'arrêt avec un fichier source (vérifiez les chemins de classe), ou
- La JVM n'a pas chargé l'agent Sparx Systems

Étape	Action
1	Exécuter le serveur et vérifier que le processus serveur a chargé l'agent Sparx Systems : DLL SSJavaProfiler32.DLL ou SSJavaProfiler64 Utilisez « Process Explorer » ou des outils similaires pour prouver que le processus serveur a chargé l'agent.
2	Dans Enterprise Architect , ouvrez le code source et définissez quelques points d'arrêt.
3	Cliquez sur le bouton Exécuter Déboguer dans Enterprise Architect . La dialogue « Attacher au processus » s'affiche.
4	Sélectionnez le processus serveur hébergeant l'application.
5	Cliquez sur le bouton OK . Un message de confirmation s'affiche dans la fenêtre Déboguer , indiquant que le processus a été attaché.

Serveur JBOSS

Dans cet exemple JBoss, pour une application 32 bits, le code source d'un simple servlet se trouve à l'emplacement du répertoire :

C:\Benchmark\Java\JBOSS\Inventaire

Les binaires exécutés par JBOSS se trouvent dans le fichier JAW.EAR à cet emplacement :

C:\JBOSS\03b-dao\build\distribution

Le débogueur Enterprise Architect doit être capable de localiser les fichiers sources pendant le débogage ; pour ce faire, il utilise également le CLASSPATH, recherchant dans n'importe quel chemin répertorié un fichier source JAVA correspondant, donc le CLASSPATH doit inclure un chemin vers la racine du Paquetage pour Enterprise Architect pour trouver la source pendant le débogage.

Ceci est un extrait du fichier de commandes qui exécute le serveur JBOSS ; la classe à déboguer se trouve à :

com/inventaire/dto/carDTO

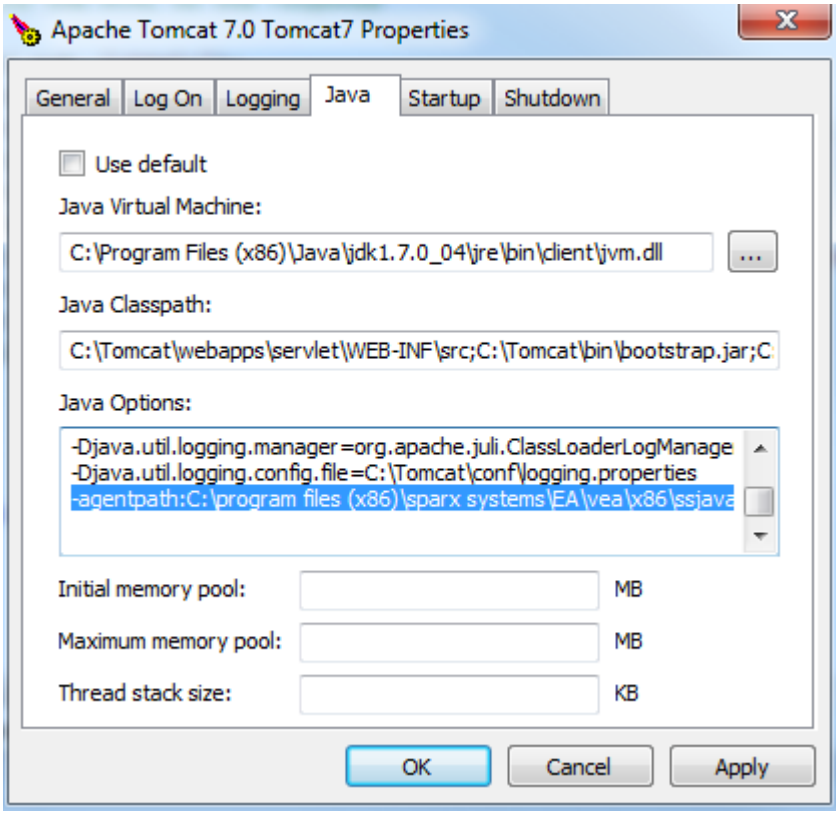
Par conséquent, la racine de ce chemin est incluse dans JBOSS_CLASSPATH.

Exemple de code

```
RUN.BAT
-------------
set SOURCE=C:\Benchmark\Java\JBOSS\Inventory
set JAVAC_JAR=%JAVA_HOME%\lib\tools.jar
if "%JBOSS_CLASSPATH%" == ""
(
set JBOSS_CLASSPATH=%SOURCE%;%JAVAC_JAR%;%RUNJAR%;
)
else
(
set JBOSS_CLASSPATH=%SOURCE%;%JBOSS_CLASSPATH%;%JAVAC_JAR%;%RUNJAR%;
)
set JAVA_OPTS=%JAVA_OPTS% -agentpath:"c:\program files\sparx systems\vea\x86\ssjavaprofiler32"
```

Serveur Apache Tomcat

Le serveur Apache Tomcat peut être configuré pour le débogage à l'aide du débogueur Java dans Enterprise Architect . Cet exemple montre la dialogue de configuration pour Apache Tomcat 7.0 sur un PC exécutant Windows 7 .



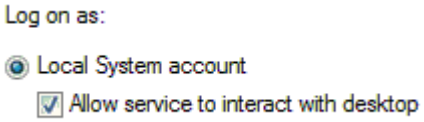
Ces trois points sont importants :

- La 'Java Virtual Machine ' spécifie le runtime d'une installation du Java JDK
- Le chemin source de tout servlet à déboguer est ajouté au chemin de classe Java ; dans ce cas nous ajoutons le chemin vers la servlet Tomcat :
  c:\tomcat\webapps\servlet\WEB-INF\src
- Les « Options Java » incluent le chemin d'accès à l'agent de débogage Sparx Systems :
  -agentpath:c:\program files (x86)\sparx Systems\vea\x86\ssjavaprofiler32

Apache Tomcat Windows Service

Configuration

Pour les utilisateurs exécutant Apache Tomcat en tant que service Windows ™ , il est important de configurer le service pour permettre l'interaction avec le bureau ; si vous ne le faites pas, le débogage échoue dans Enterprise Architect .



Cochez la case « Autoriser le service à interagir avec le bureau ».

.NET

Cette section décrit comment configurer Enterprise Architect pour le débogage des applications .NET . Il comprend:

- General Setup for .NET
- Debugging an Unmanaged Application
- Debug COM Interop
- Debug ASP .NET

Configuration générale pour .NET

Il s'agit de la configuration générale pour le débogage des applications Microsoft .NET . Vous disposez de deux options lors du débogage :

- Déboguer une application
- Attacher à une application en cours d'exécution

Option 1 – Déboguer une application

Champ	Action
Débogueur	Sélectionnez Microsoft .NET comme plate-forme de débogage.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.
Mode	Sélectionnez le bouton radio Exécuter .
Répertoire par défaut	Ceci est défini comme répertoire par défaut pour le processus en cours de débogage.
Chemin d'accès à la candidature	Sélectionnez et entrez le chemin complet ou relatif de l'exécutable de l'application. • Si le chemin contient des espaces, spécifiez le chemin complet ; ne pas utiliser de chemin relatif • Si le chemin contient des espaces, il doit être entouré de guillemets
Arguments de ligne de commande	Paramètres à transmettre à l'application au démarrage.
Afficher la console	Créez une fenêtre de console pour le débogueur ; ne s'applique pas à l'attachement à un processus.
Chemins de recherche de symboles	Spécifiez tous les chemins supplémentaires pour localiser les symboles de débogage pour le débogueur ; séparez les chemins par un point-virgule.

Option 2 – Connecter à une application en cours d'exécution

Champ	Action
Débogueur	Sélectionnez Microsoft .NET comme plate-forme de débogage.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.
Mode	Sélectionnez le bouton radio Attacher au processus.

Débogage d'une application non gérée

Si vous déboguez du code managé à l'aide d'une application non managée, le débogueur peut ne pas réussir à détecter la version correcte du Common Language Runtime (CLR) à charger.

Vous devez spécifier un fichier de configuration si vous n'en avez pas déjà un pour l'application de débogage spécifiée dans la commande Déboguer de votre script.

Le fichier de configuration doit résider dans le même répertoire que votre application et avoir le format :

name.exe.config

où « nom » est le nom de votre application.

La version du CLR que vous spécifiez doit correspondre à la version chargée par le code managé invoqué par le débogué.

Dans l'exemple de code qui suit, « clr_version » représente la version du CLR ciblée par votre plug-in ou votre code COM.

Exemple de fichier de configuration

```
<configuration>
    <startup>
        <requiredRuntime version="clr_version"/>
    </startup>
</configuration>
```

Déboguer COM Interop

Enterprise Architect vous permet de déboguer le code managé .NET exécuté à l'aide de COM sur un serveur local ou en cours.

Cette fonctionnalité est utile pour déboguer les plug-ins et les composants ActiveX.

Déboguer Code managé .NET exécuté à l'aide de COM

Étape	Action
1	Créez un Paquetage dans Enterprise Architect et importez le code à déboguer.
2	Assurez-vous que le composant COM est construit avec des informations de débogage.
3	Créez un script pour le Paquetage .
4	Dans le ' Déboguer | Plateforme', vous pouvez choisir de vous attacher à un processus non géré ou de spécifier le chemin d'accès à une application non gérée pour appeler votre code managé.
5	Ajoutez des points d'arrêt dans le code source pour déboguer.

Attacher à un processus non géré

Si vous utilisez :

- Un serveur COM In-Process, attaché au processus client
- Un serveur COM local, attaché au processus serveur

Cliquez sur le bouton Exécuter de la fenêtre Déboguer (ou appuyez sur F6) pour afficher une liste de processus parmi lesquels vous pouvez choisir.

Notes

- Le détachement d'un processus COM Interop que vous avez débogué met fin au processus ; il s'agit d'un problème connu pour Microsoft .NET Framework et des informations à ce sujet peuvent être trouvées sur de nombreux blogs MSDN .NET .

Déboguer ASP .NET

Le débogage des services Web tels qu'ASP nécessite que le débogueur Enterprise Architect puisse se connecter à un service en cours d'exécution.

Commencez par vous assurer que le répertoire contenant le projet de service ASP .NET a été importé dans Enterprise Architect et, si nécessaire, le dossier Web contenant les pages Web du client.

Si le répertoire de votre projet Web réside dans le répertoire d'hébergement du site Web, vous pouvez importer à partir de la racine et inclure simultanément le code ASP et les pages Web.

Il est nécessaire de lancer d'abord le client, car le processus du service ASP .NET n'est peut-être pas déjà en cours d'exécution ; chargez le client à l'aide de votre navigateur - cela garantit que le serveur Web est en cours d'exécution.

Dans la configuration du débogage, vous sélectionnerez ensuite le bouton radio « Attacher ». Lorsque ce choix est sélectionné, le débogueur vous prompt à chaque fois que le processus soit à déboguer.

Cliquez sur le bouton Exécuter de la fenêtre Déboguer pour démarrer le débogueur ; la dialogue « Attacher au processus » s'affiche.

Le nom du processus varie selon les systèmes d'exploitation Microsoft, comme expliqué dans le *SDK ASP .NET* ; par exemple, sous Windows XP, le nom du processus ressemble à aspnet_wp.exe, bien que le nom puisse refléter la version du framework .NET qu'il prend en charge.

Plusieurs processus ASP.NET peuvent être exécutés sous XP ; vous devez vous assurer que vous vous attachez à la bonne version, qui serait celle hébergeant la version du framework .NET sur laquelle votre application s'exécute ; vérifiez le fichier web.config de votre service Web pour vérifier la version du framework .NET auquel il est lié.

Le bouton Stop de la fenêtre Déboguer doit être activé et tous les points d'arrêt doivent être rouges, indiquant qu'ils ont été liés.

Vous pouvez définir des points d'arrêt à tout moment dans le code du serveur Web. Vous pouvez également définir des points d'arrêt dans la ou les pages Web ASP si vous les avez importées.

Notes

Certains points d'arrêt n'ont peut-être pas été liés avec succès, mais si aucun n'est lié (indiqué par un rouge foncé avec des points d'interrogation), quelque chose n'est plus synchronisé ; essayez de reconstruire et de réimporter le code source

Le Mono Débogueur

Mono est une plate-forme logicielle sponsorisée par la .NET Foundation pour faciliter le développement multiplateforme. Il est populaire auprès des développeurs de jeux pour ses riches fonctionnalités de jeu, basées sur des API et de portabilité.

Enterprise Architect fournit support à la communauté Mono en fournissant un environnement moderne pour modélisation et le développement de logiciels. Les projets existants peuvent être importés, construits et débogués de manière native sous Linux et Windows .

Aperçu

Le débogage sous Mono implique la coopération de trois processus. Le runtime Mono gère l'application et communique à l'aide d'un protocole socket avec Enterprise Architect Débogueur , qui à son tour communique avec Enterprise Architect agissant en tant que frontal. Lorsque vous lancez Mono, vous devez lui demander de support le débogage, ce que vous réalisez à l'aide d'une directive de ligne de commande dans laquelle vous nommez l'hôte et le numéro de port sur lesquels Mono doit écouter. L'hôte peut être omis, auquel cas Mono acceptera les connexions à partir de n'importe quelle adresse IP. L'hôte peut avoir la valeur 'localhost' pour restreindre les connexions à la même machine. Le numéro de port est un numéro de votre choix.

L'hôte et le numéro de port sont des informations importantes, car ils sont utilisés lors de la configuration du script Analyzer.

Exigences pour Windows

- Enterprise Architect (version 14 minimum)
- Mono pour Windows (version 5.4 minimum)

Exigences pour Linux

- Enterprise Architect (version 14 minimum)
- Mono pour Linux (version 5.4 minimum)
- Vin pour Linux

La page hôte d'exécution

Cette page est facultative et n'est utile que lorsque Mono et Enterprise Architect s'exécutent sur la même machine. Il offre la possibilité d' exécuter Mono d'abord avec les directives de débogage requises, avant le démarrage du débogueur Enterprise Architect . Une fois le débogueur connecté, il reprend le runtime Mono, qui a été démarré comme suspendu. Si l'application s'exécute sur une machine différente de celle d' Enterprise Architect que vous utilisez, vous devez effacer cette section.

Configuration du débogage Linux

Configuration Débogueur

Cette section décrit la section Déboguer d'un script Analyzer concernant le débogage de Mono sous Linux. Les champs qui ne sont pas répertoriés ici ne sont pas obligatoires.

Débogueur	Sélectionnez « Mono ».
Répertoire par défaut	Il s'agit du chemin Linux natif complet où se trouve l'application au format Unix.
Connexion	• port : le port de débogage • host : le nom ou l'adresse IP de la machine sur laquelle Mono s'exécute ("localhost" si la machine est la même) • localpath : le chemin racine du code source au format Windows ; il s'agit du chemin d'accès aux fichiers sources que vous utilisez pour définir les points d'arrêt dans l'éditeur de code d' Enterprise Architect • remotepath : le chemin racine du code source au format Unix, celui-ci étant le chemin vers les fichiers sources utilisés pour construire le programme sous Linux Ces chemins sont renvoyés lors des événements de débogage, puis mappés sur le chemin local, afin Enterprise Architect puisse afficher le fichier source lors d'un point d'arrêt ou d'une étape. Les deux paramètres peuvent spécifier la même racine de fichier source physique, mais doivent utiliser Windows ou Unix. format pour chaque champ • arrêt : (vrai ou faux) ; lorsque vrai, la VM est terminée lorsque le Débogueur est arrêté • timeout : le délai d'attente en millisecondes pour les appels de socket • sortie : le chemin Wine/ Windows du fichier log dans lequel écrire • journalisation : (vrai ou faux) ; lorsque c'est vrai, les messages supplémentaires sont enregistrés dans la fenêtre Déboguer et les messages de socket sont enregistrés dans le fichier de sortie spécifié

Démarrage automatique de Mono

Vous pouvez configurer Enterprise Architect pour démarrer Mono pour vous lorsque vous démarrez le débogueur. Pour ce faire, configurez la page « Runtime Host » de votre script Analyzer. Le format des commandes est décrit ici :

chemin d'accès au programme sur CD

/usr/bin/mono --debug --debugger-agent=transport=dt_socket,address= *host:port* ,server=y,suspend=y *programme*

où:

- *chemin d'accès au programme* est le chemin du répertoire où se trouve le programme

- *l'hôte* est l'un d'entre eux :

- hôte local
- une adresse IP

- un nom de machine en réseau

- *port* est le port du socket

- *programme* est le nom de l'application (comme MonoProgram.exe)

Démarrage manuel de Mono à l'aide de la ligne de commande

Vous pouvez démarrer Mono manuellement à partir d'une console. Localisez le programme dans votre explorateur de fichiers, puis ouvrez une console à cet emplacement. Le format de la ligne de commande est décrit ici :

/usr/bin/mono --debug --debugger-agent=transport=dt_socket,address= *host:port* ,server=y,suspend=y *programme*

où *host* est l'un de ces éléments :

- hôte local
- une adresse IP
- un nom de machine en réseau

port est le port du socket et *program* est le nom de l'application (par exemple, MonoProgram.exe).

Windows de configuration de débogage

Configuration Débogueur

Cette section décrit la section Déboguer d'un script Analyzer en ce qui concerne le débogage de Mono sous Windows . Les champs qui ne sont pas répertoriés ici ne sont pas obligatoires.

Champ	Description
Débogueur	Sélectionnez « Mono ».
x64	Sélectionnez si le programme à déboguer est un exécutable 64 bits.
Exécuter ou Attacher	Choisissez ' Exécuter ' pour nommer le programme à lancer. Choisissez « Attacher » si vous vous attacherez toujours à un processus en cours d'exécution.
Répertoire par défaut	Le répertoire par défaut que le programme occupera lors de son exécution.
Chemin d'application	Le chemin complet de l'application Mono.
Arguments de ligne de commande	Tous les paramètres à transmettre au programme. Si les paramètres contiennent des espaces, entourez-les de guillemets doubles (")

Démarrage automatique de Mono

Vous pouvez configurer Enterprise Architect pour démarrer Mono pour vous lorsque vous démarrez le débogueur. Pour ce faire, configurez la page « Runtime Host » de votre script Analyzer. Le format des commandes est décrit ici :

chemin d'accès au programme sur CD

mono --debug --debugger-agent=transport=dt_socket,address= *host:port* ,server=y,suspend=y *programme*

où:

- *chemin d'accès au programme* est le chemin du répertoire où se trouve le programme

- *l'hôte* est l'un d'entre eux :

- hôte local
- une adresse IP
- un nom de machine en réseau

- *port* est le port du socket

- *programme* est le nom de l'application (comme MonoProgram.exe)

Démarrage manuel de Mono à l'aide de la ligne de commande

Vous pouvez démarrer Mono manuellement à partir d'une console. Localisez le programme dans votre explorateur de fichiers, puis ouvrez une console à cet emplacement. Le format de la ligne de commande est décrit ici :

mono --debug --debugger-agent=transport=dt_socket,address= *host:port* ,server=y,suspend=y *programme*

où *host* est l'un de ces éléments :

- hôte local
- une adresse IP
- un nom de machine en réseau

port est le port du socket et *program* est le nom de l'application (par exemple, MonoProgram.exe).

Le Débogueur PHP

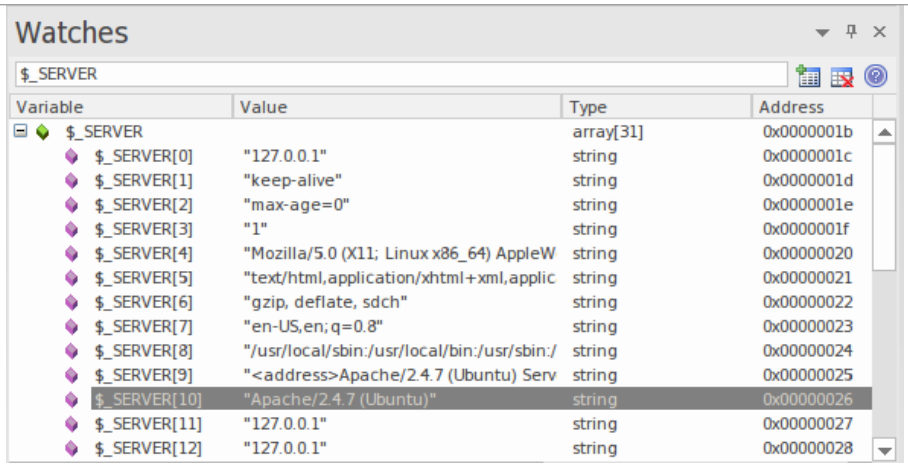
Le Débogueur PHP Enterprise Architect vous permet de déboguer les scripts PHP.exe. Cette section traite de la configuration de base et des différents scénarios de débogage couramment rencontrés ; les scénarios concernent le mappage des chemins de fichiers, ce qui est essentiel au succès d'une session de débogage à distance.

- Configuration des scripts
- Machine Windows locale (serveur Apache)
- Machine Windows locale (PHP.exe)
- Machine Linux distante (serveur Apache)
- Machine Linux distante (PHP.exe)

Configuration et scénarios

Scénario	Détails
Configuration des scripts	Un script Analyzer est une exigence de base pour le débogage dans Enterprise Architect ; vous créez un script à l'aide de la barre d'outils de l' Analyseur d'Exécution . Sélectionnez PHP.XDebug comme plateforme de débogage ; lorsque vous sélectionnez cette plateforme, la page de propriétés affiche ces paramètres de connexion : • host - localhost - L'adaptateur sur lequel Enterprise Architect écoute les connexions entrantes depuis PHP • localpath - %LOCAL% - Spécifie le chemin du fichier local à mapper à un chemin de fichier distant ; il s'agit d'un paramètre de débogage à distance - pour le débogage local, effacez la valeur , la valeur est un espace réservé et vous devez la modifier pour l'adapter à votre scénario particulier • remotepath - %REMOTE% - Spécifie le chemin de fichier distant auquel un chemin de fichier local doit être mappé ; il s'agit d'un paramètre de débogage à distance - pour le débogage local, effacez la valeur , la valeur est un espace réservé et vous devez la modifier pour l'adapter à votre scénario particulier • logging - Entrez true ou false pour activer la journalisation des communications depuis le serveur XDebug • sortie - nomme le chemin du fichier sur la machine distante à utiliser avec l'option de journalisation ; ce fichier sera toujours écrasé
Serveur Apache de machine locale	Dans cette situation, considérez cette configuration : • S/E: Windows 7 • Nom de l'ordinateur du réseau : MyPC • Partage réseau MyShare mappé sur c:\myshare • Les fichiers sources dans Enterprise Architect ont été importés depuis c:\myshare\apache\myapp\scripts • La racine du document Apache est définie sur //MyPC/MyShare/apache Dans ce scénario, un script Analyzer pour les paramètres de connexion peut être configuré comme : • hôte : hôte local • port : 9000 • chemin local : c:\myshare\apache\

	• chemin distant : MonPC/MonPartage/apache/
Machine locale PHP.EXE	Dans ce scénario, un script Analyzer pour les paramètres de connexion peut être configuré comme indiqué, car les chemins de fichiers correspondent toujours au même chemin physique : • hôte : hôte local • port : 9000 • chemin local : • chemin distant :
Machine Linux distante Serveur Apache	Dans cette situation, considérez cette configuration : Machine locale: • S/E: Windows 7 • Les fichiers sources dans Enterprise Architect ont été importés depuis c:\myshare\apache\myapp\scripts Appareil distant : • S/E: Linux • La racine du document Apache est définie sur home/apache/htdocs • Les fichiers sources dans Apache se trouvent dans home/apache/htdocs/myapp/scripts Dans ce scénario, un script Analyzer pour les paramètres de connexion peut être configuré comme : • hôte : hôte local • port : 9000 • chemin local : c:\myshare\apache\ • chemin distant : home/apache/htdocs/
Machine Linux distante PHP.exe	Dans cette situation, considérez cette configuration : • Machine locale • S/E: Windows 7 • Les fichiers sources dans Enterprise Architect ont été importés depuis c:\myshare\apache\myapp\scripts • Machine distante • S/E: Linux • Fichiers sources dans Apache situés dans home/myapp/scripts Dans ce scénario, un script Analyzer pour les paramètres de connexion peut être configuré comme : • hôte : hôte local • port : 9000 • chemin local : c:\myshare\apache\ • chemin distant : accueil/
Variables globales PHP	Lorsque vous êtes à un point d'arrêt, vous pouvez examiner les valeurs des globales PHP à l'aide de la fenêtre Analyser Observateurs . Pour répertorier tous les globaux, tapez « globaux » ou « superglobaux » dans le champ. Pour afficher un élément individuel, entrez son nom. Cette image montre la valeur de la variable d'environnement PHP $_SERVER affichée.


Watches
$_SERVER
Variable
Value
Type
Address
$_SERVER array[31] 0x0000001b
$_SERVER[0] "127.0.0.1" string 0x0000001c
$_SERVER[1] "keep-alive" string 0x0000001d
$_SERVER[2] "max-age=0" string 0x0000001e
$_SERVER[3] "1" string 0x0000001f
$_SERVER[4] "Mozilla/5.0 (X11; Linux x86_64) AppleW string 0x00000020
$_SERVER[5] "text/html,application/xhtml+xml,applic string 0x00000021
$_SERVER[6] "gzip, deflate, sdch" string 0x00000022
$_SERVER[7] "en-US,en;q=0.8" string 0x00000023
$_SERVER[8] "/usr/local/sbin:/usr/local/bin:/usr/sbin:/ string 0x00000024
$_SERVER[9] "<address>Apache/2.4.7 (Ubuntu) Serv string 0x00000025
$_SERVER[10] "Apache/2.4.7 (Ubuntu)" string 0x00000026
$_SERVER[11] "127.0.0.1" string 0x00000027
$_SERVER[12] "127.0.0.1" string 0x00000028

Débogueur PHP - Exigences système

Cette rubrique identifie la configuration système requise et les systèmes d'exploitation pour le débogueur PHP Enterprise Architect .

Exigences du système :

- Enterprise Architect version 9
- PHP version 5.3 ou supérieure
- Extension PHP zend XDebug 2.1 ou supérieur
- Pour les serveurs web tels qu'Apache, une version serveur supporte la version PHP

Systèmes d'Exploitation supportés :

- Client (Enterprise Architect)
- Microsoft Windows XP et supérieur
- Linux exécutant Crossover Office
- Serveur (PHP)
- Microsoft Windows XP et supérieur
- Linux

Liste de contrôle Débogueur PHP

Cette rubrique fournit un guide de dépannage pour le débogage des scripts PHP dans Enterprise Architect .

Points de contrôle

Point de contrôle	Détails
Exigences du système	• Serveur Web HTTP Apache version 2.2 • PHP version 5.3 ou supérieure • XDebug version 2.1.1
Enterprise Architect	• Le modèle dispose d'un script Analyzer configuré pour utiliser la plateforme PHP XDebug • Le code source PHP a été importé dans le modèle (pour l'enregistrement et les points de test) • Lorsque la plateforme PHP XDebug est sélectionnée dans la dialogue « Script d'analyseur », les paramètres d'exécution par défaut sont répertoriés dans le champ « Connexion » : chemin local :%LOCAL% chemin distant :%REMOTE% Définissez des chemins locaux pour ces variables par défaut ou modifiez le script pour fournir les chemins réels. Par exemple : source locale, source distante chemin local : c:\exemples de code\vea\php\sample chemin distant : serveur Web/échantillon • « serveur Web » est un réseau ou un partage local • 'sample' est un dossier sous le partage
PHP	Afin de déboguer les scripts PHP dans Enterprise Architect , il est nécessaire que PHP soit correctement configuré pour charger l'extension XDebug. Des paramètres similaires à ceux-ci doivent être utilisés (pour XDebug version 3 ou supérieure) : • [xdebug] • zend_extension=xdebug.so • xdebug.mode=débogage • xdebug.mode=débogage • xdebug.start_with_request=oui • xdebug.client_host=localhost • xdebug.client_port=9003 Pour les versions de Xebug inférieures à 3, utilisez les anciens paramètres tels que : • [xdebug] • xdebug.extended_info=1 • xdebug.idekey=ea • xdebug.remote_enable=1 • xdebug.remote_handler=dbgp

	• xdebug.remote_autostart=1 • xdebug.remote_host=XXXX • xdebug.remote_port=9000 • xdebug.show_local_vars=1 L'adresse IP XXXX fait référence et doit correspondre à l'hôte spécifié dans le script d'analyseur du modèle. L'adresse IP est l'adresse à laquelle XDebug se connecte et la même adresse sur laquelle l'agent PHP Enterprise Architect écoute.
Apache	Pour le débogage avec Apache, ces lignes doivent être présentes dans le fichier de configuration Apache, httpd.conf : LoadModule php5_module "php_home/php5apache2_2.dll" Application AddHandler/x-httpd-php .php PHPIniDir "php_home" La valeur "php_home" est le chemin d'installation de PHP (le chemin où existent php.ini et apache dll).
Dépannage	Pour éviter les délais d'attente PHP et Apache lors d'une session de débogage, ces paramètres peuvent nécessiter une modification. Les paramètres ont été utilisés lors du développement de l'agent de débogage PHP dans Enterprise Architect .
PHP	Fichier : php.ini ; Enterprise Architect empêche les délais d'attente PHP lors du débogage des extensions PHP max_execution_time = 0 ; Enterprise Architect empêche les délais d'attente du serveur Web lors du débogage des extensions PHP max_input_time = -1 ; Enterprise Architect enregistre les erreurs display_errors = Activé ; Enterprise Architect affiche les erreurs de démarrage display_startup_errors = Activé
Apache	Fichier : httpd.conf ; Enterprise Architect empêche les délais d'attente lors du débogage des extensions PHP Délai d'expiration 60 000

Le Débogueur GNU (GDB)

Lors du débogage de vos applications, vous pouvez utiliser le Débogueur GNU (GDB), qui est portable et fonctionne sur des systèmes de type Unix tels que Linux, ainsi que sur Windows . Le GDB fonctionne pour de nombreux langages de programmation, notamment Ada, Java, C, C++ et Objective-C. Grâce au GDB, vous pouvez déboguer vos applications localement ou à distance.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script souhaité et sélectionnez la page ' Déboguer > Plateforme' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Déboguer > Plateforme'

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Menu Contexte	Fenêtre Navigateur | Cliquez-droit sur Paquetage | Analyseur d'Exécution
Raccourcis Clavier	Maj+F12

Configurer le Débogueur GNU

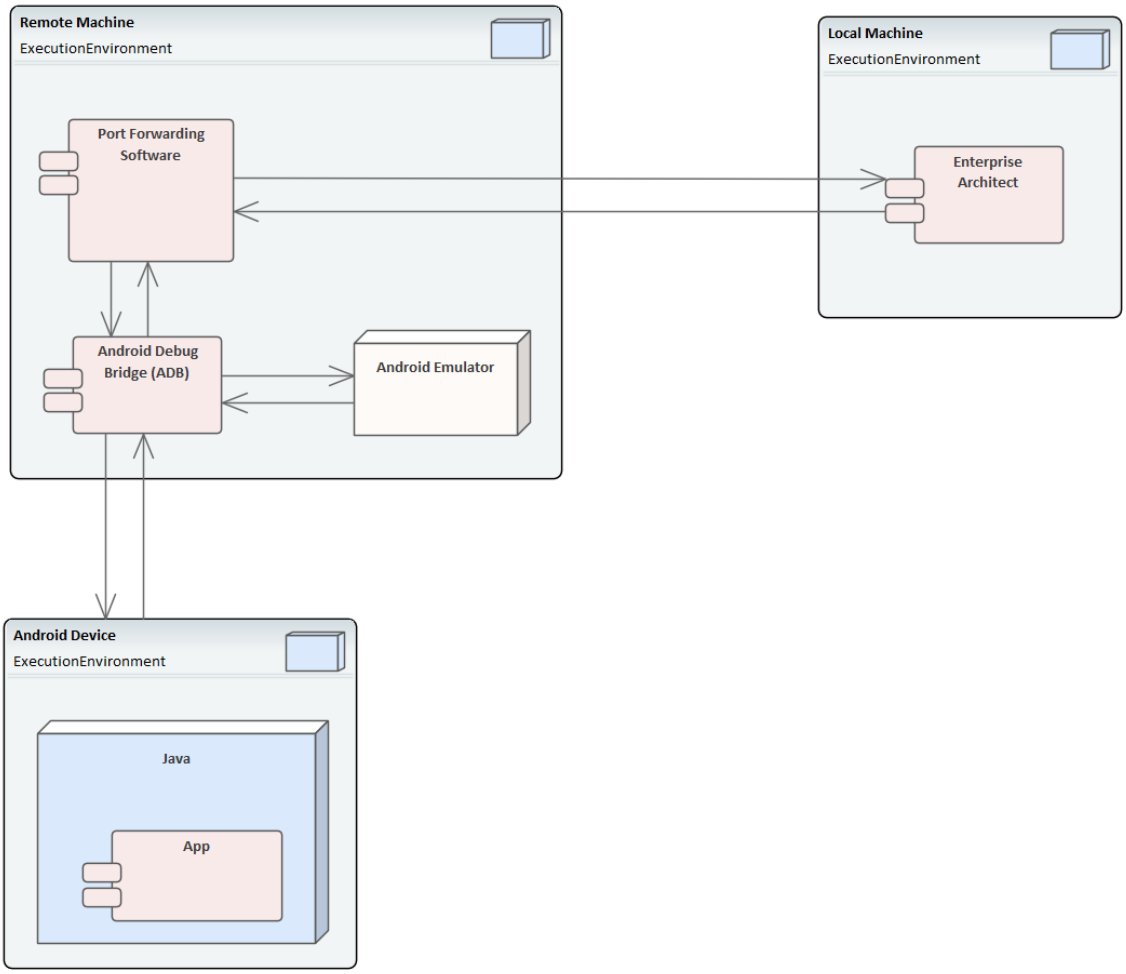
Tâche	Détails
Configurer le script	Un script Analyzer est une exigence de base pour le débogage dans Enterprise Architect ; vous créez un script à l'aide de la barre d'outils Analyseur d'Exécution . Sur la page 'Plateforme' de l'Execution Éditeur de Script Analyseur , dans le champ ' Débogueur ' cliquez sur la flèche déroulante et sélectionnez 'GDB'.
Définir les paramètres de connexion	Le panneau de propriétés affiche un certain nombre de paramètres de connexion pour lesquels vous fournissez des valeurs. • path - <path> - Le chemin complet du fichier de l'exécutable GDB ; vous ne le spécifiez que si le GDB est introuvable dans le chemin du système • source - <path>, <path> - Le chemin dans lequel le débogueur recherchera les fichiers source, s'ils ne résident pas dans le répertoire exécutable • distant - F - Définir pour le débogage à distance ; sinon laisser vide • port - <nnnnn> - Le port auquel se connecter sur le serveur distant • host - localhost - Le nom d'hôte auquel se connecter • fetch - T - Définir pour récupérer le binaire du système distant • dumpgdb - <path> - Le nom de fichier dans lequel écrire la sortie GDB • initpath - <path> - Le chemin complet du fichier gbinit

Notes

- Une exigence du GDB est que le chemin de votre fichier de code source ne contienne pas d'espaces ; le débogueur ne exécuter pas correctement avec des espaces dans le chemin du fichier

Le Débogueur Android

Si vous développez des applications Java exécutées sur des appareils ou des émulateurs Android, vous pouvez également les déboguer. Les machines locales et distantes peuvent être sur une plate-forme 32 bits ou une plate-forme 64 bits.



Exigences du système

Sur la machine distante, ce logiciel est requis :

- SDK Android, qui inclut le pont de débogage Android, ADB (vous devez être familier avec le SDK et ses outils)
- Java JDK (support 32 et 64 bits)
- Logiciel de redirection de port (tiers)

Sur la machine locale, ce logiciel est requis :

- Enterprise Architect version 10 ou supérieure

Paramètres du script de l'analyseur

Champ/Bouton	Action

Débogueur	Cliquez sur la flèche déroulante et sélectionnez Java (JDWP).
Exécuter	Cliquez sur ce bouton radio.
Répertoire par défaut	Sans objet – laissez vide.
Chemin de candidature	Sans objet – laissez vide.
Arguments de ligne de commande	Sans objet – laissez vide.
Construire d'abord	Sans objet – laissez vide.
Afficher la console	Sans objet – laissez vide.
Afficher les messages de diagnostic	Sans objet – laissez vide.
Connexion	Sans objet – laissez vide.
Port	Il s'agit du port d'application, attribué en avant à l'aide d'adb ou d'autres moyens, par lequel Enterprise Architect et la Virtual Machine (VM) Android peuvent communiquer.
Hôte	Ordinateur hôte (par défaut sur localhost) Si Android s'exécute sur un émulateur sur un appareil connecté à un ordinateur en réseau, saisissez ici le nom du réseau. Par défaut, le débogage tentera de se connecter au port que vous spécifiez sur la machine locale.
Source	Il s'agit de l'équivalent source du paramètre classpath en Java. La racine de chaque arborescence source doit être répertoriée. Si plusieurs sont spécifiés, ils doivent être séparés par un point-virgule ; c'est-à-dire: c:\myapp\src;c:\myserver\src Vous devez spécifier au moins un chemin source racine. Lorsqu'un point d'arrêt se produit, le débogueur recherche la source Java dans chacune des arborescences sources répertoriées ici.
Enregistrement	Permet de consigner des informations supplémentaires à partir du débogueur valeurs possibles : vrai, faux, 1,0, oui, non
Sortir	Spécifie le nom complet du fichier log local à écrire. Le dossier doit exister sinon aucun log ne sera créé. Le fichier log contient généralement un vidage des octets envoyés entre le débogueur et la VM.
Plate-forme	Si vous déboguez Java exécuté dans un scénario Android, sélectionnez Android. Pour tous les autres scénarios, sélectionnez Java.

Configurer les ports pour le débogage - Redirection de port (local)

Le débogueur ne peut déboguer qu'une seule VM à la fois ; il utilise un seul port pour communiquer avec la VM. Le port de l'application à déboguer peut être attribué à l'aide d'ADB, fourni avec le SDK Android.

Avant le débogage, démarrez l'application une fois dans l'appareil. Lorsque l'application démarre, découvrez son identifiant de processus (pid) :

adb jdwp

Le dernier numéro indiqué est le pid de la dernière application lancée ; note le pid et utilisez-le pour permettre au débogueur de se connecter à la VM :

- adb transférer tcp:port jdwp:pid
  - port = numéro de port répertorié dans le script de l'analyseur
  - pid = identifiant de processus de l'application sur l'appareil

Configurer les ports pour le débogage - Redirection de port (à distance)

Pour déboguer à distance, la même procédure doit être suivie que pour la machine locale, mais la communication nécessite un transfert supplémentaire car le socket créé à l'aide de la commande adb forward n'écoutera que sur l'adaptateur local. Le socket est lié à l'hôte local et les tentatives de connexion à ce port recevront des messages de « connexion refusée ».

Afin de réaliser le débogage à distance, il est nécessaire d'avoir un proxy exécuté sur la machine distante qui écoute toutes les connexions entrantes et transfère tout le trafic vers le port adb ; il existe de nombreux logiciels disponibles pour ce faire.

Le débogage à distance avec Enterprise Architect ne fonctionnera que si vous avez configuré un redirecteur de port proxy.

Débogueur Java JDWP

Java fournit deux technologies de débogage principales : un système basé sur un agent en cours appelé Java Virtual Machine Tools Interface (JVMTI) et un paradigme basé sur un socket appelé Java Déboguer Wire Protocol (JDWP). Une Virtual Machine Java peut nommer l'un ou l'autre mais pas les deux, et la fonctionnalité doit être configurée au démarrage de la JVM.

Exigences du système

1. Le débogueur Enterprise Architect JDWP ne pourra communiquer qu'avec une JVM démarrée avec l'option 'JDWP'. Voici un exemple d'option de ligne de commande :
   java -agentlib:jdwp=transport=dt_socket,address=localhost:9000,server=y,suspend=n -cp "c:\java\myapp;%classpath%" demo.myApp "param1" "param2"
2. La Virtual Machine ne doit pas être actuellement attachée à un débogueur.
3. Il n'est pas possible qu'une VM soit déboguée par Enterprise Architect et Eclipse en même temps.

Paramètres du script de l'analyseur

Champ/Bouton	Action
Débogueur	Cliquez sur la flèche déroulante et sélectionnez Java (JDWP).
Exécuter	Cliquez sur ce bouton radio pour exécuter le débogueur lorsque le script est exécuté.
Répertoire par défaut	Sans objet – laissez vide.
Chemin de candidature	Sans objet – laissez vide.
Arguments de ligne de commande	Sans objet – laissez vide.
Construire d'abord	Sans objet – laissez vide.
Afficher la console	Sans objet – laissez vide.
Afficher les messages de diagnostic	Sans objet – laissez vide.
Connexion	Sans objet – laissez vide.
Port	Définissez le port d'application attribué au processus de la VM lors du démarrage, dans les options de ligne de commande Java.
Hôte	Définir l'ordinateur hôte (par défaut sur localhost) Si la VM s'exécute sur un ordinateur en réseau, entrez le nom ou l'URL du réseau ici. Par défaut, le débogage tentera de se connecter au port que vous spécifiez sur la

	machine locale.
Source	Il s'agit de l'équivalent source du paramètre *classpath* en Java. Répertoriez la racine de chaque arborescence source ; spécifiez au moins un chemin source racine. Si vous en spécifiez plusieurs, séparez-les par un point-virgule ; Par exemple: c:\monapplication\src **;** c:\monserveur\src Lorsqu'un point d'arrêt se produit, le débogueur recherche la source Java dans chacune des arborescences sources répertoriées ici.
Enregistrement	Activez ou désactivez la journalisation des informations supplémentaires du débogueur. Les valeurs possibles incluent : • vrai • FAUX • 1 • 0 • Oui • Non
Sortir	Spécifiez le nom complet du fichier log local à écrire. Si le dossier n'existe pas déjà, aucun log ne sera créé. Le fichier log contient généralement un vidage des octets envoyés entre le débogueur et la VM.
Plate-forme	Sélectionnez Java.

Configurer les ports pour le débogage

Le débogueur ne peut déboguer qu'une seule VM à la fois ; il utilise un seul port pour communiquer avec la VM. Le port de l'application à déboguer est attribué lors de la création de la VM.

Débogage local

Lorsque Enterprise Architect et la machine virtuelle Java s'exécutent sur la même machine, vous pouvez effectuer un débogage local. Il est nécessaire de lancer la VM avec le transport JDWP activé - voir la documentation sur *Java Platform Débogueur Architecture (JPDA)* chez Oracle pour les spécifications des options de ligne de commande. Par exemple:

java -agentlib:jdwp=transport=dt_socket,address=localhost:9000,server=y,suspend=n -cp "c:\samples\java\myapp;%classpath%" samples.MyApp "param1" "param2"

Dans cet exemple, les valeurs du script Analyzer seraient « host : localhost » et « port : 9000 ».

Débogage à distance

Lorsque Enterprise Architect s'exécute sur la machine locale et que la machine virtuelle Java s'exécute sur une machine

distante, vous pouvez effectuer un débogage à distance. Il est nécessaire de lancer la VM avec le transport JDWP activé - voir la documentation sur JPDA chez Oracle pour les spécifications des options de ligne de commande. Voici un exemple, où l'ordinateur distant porte le nom de réseau testmachine1 :

```
java -agentlib:jdwp=transport=dt_socket,address=9000,server=y,suspend=n -cp "c:\samples\java\myapp;%classpath%" samples.MyApp "param1" "param2"
```

Note l'absence de nom d'hôte dans l'adresse. Cela signifie que la VM écoutera une connexion depuis n'importe quelle machine. Dans cet exemple, les valeurs du script Analyzer seraient « host : testmachine1 » et « port : 9000 ».

Sortie Point de Trace

La page Tracepoints du script Analyzer vous permet d'indiquer où va la sortie de toute instruction Trace au cours d'une session de débogage.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script souhaité et sélectionnez la page ' Déboguer > Tracepoints' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Déboguer > Tracepoints'

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur > sélectionner et exécuter le script
Menu Contexte	Fenêtre Navigateur | Cliquez-droit sur Paquetage | Analyseur d'Exécution
Raccourcis Clavier	Maj+F12

Propriétés Point de Trace

Champ	Détail
Sortir	Vous pouvez choisir parmi deux options : • 'Ecran' (valeur par défaut) - La sortie est dirigée vers la fenêtre Déboguer • 'Fichier' - La sortie est dirigée vers un fichier
Dossier	Entrez le dossier à utiliser pour les fichiers log de l'instruction Trace.
Nom de fichier	Entrez le nom à utiliser pour les fichiers log de l'instruction Trace.
Écraser	Si cette option est sélectionnée, le fichier spécifié est écrasé à chaque démarrage d'une session de débogage.
Numéro automatique	Si cette option est sélectionnée, le fichier log de trace est composé du nom de fichier que vous spécifiez et d'un numéro. Chaque fois que vous démarrez une session de débogage, le numéro est incrémenté.
Sortie de trace de préfixe avec fonction	Si cette option est sélectionnée, toutes les instructions Trace exécutées pendant la session exécuter débogage sont préfixées par l'appel de fonction en cours.

Établi établie

Cette rubrique décrit la configuration requise pour la configuration d' Object Établi sur Java et Microsoft .NET .

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script souhaité et sélectionnez la page ' Déboguer > Établi ' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Déboguer > Établi '

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Plateformes

Plate-forme	Détail
Plateformes prises en charge	L' Établi supporte ces plateformes : • Microsoft .NET (version 2.0 ou ultérieure) • Java (JDK 1.4 ou version ultérieure)
Microsoft .NET Établi	L'atelier .NET nécessite un assembly, qui est utilisé pour créer les éléments de l'atelier. Vous spécifiez le chemin d'accès à l'assembly sur la page ' Établi ' du script Analyzer. Il existe deux contraintes liées à l'utilisation de l'atelier .NET : • Les membres définis comme struct dans le code managé ne sont pas pris en charge • Les classes définies comme internes ne sont pas prises en charge
Java Établi	Le workbench Java utilise les paramètres Virtual Machine configurés dans la page ' Déboguer ' d'Analyzer Script pour créer la JVM.

Microsoft C++ et Natif (C, VB)

Vous pouvez déboguer le code natif uniquement s'il existe un fichier PDB correspondant pour l'exécutable. Un fichier PDB est créé à la suite de la création de l'application.

La version doit inclure des informations de débogage complètes et aucune optimisation ne doit être définie.

Le script doit spécifier deux éléments pour support le débogage :

- Le chemin vers l'exécutable
- Microsoft Native comme plateforme de débogage

Configuration générale

Il s'agit de la configuration générale pour le débogage des applications natives Microsoft (C++, C, Visual Basic). Vous disposez de deux options lors du débogage :

- Déboguer une application
- Attacher à une application en cours d'exécution

Option 1 – Déboguer une application

Champ	Action
Débogueur	Sélectionnez Microsoft Native comme plate-forme de débogage.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.
Mode	Sélectionnez le bouton radio Exécuter .
Répertoire par défaut	Ceci est défini comme répertoire par défaut pour le processus en cours de débogage.
Chemin d'application	Sélectionnez et entrez le chemin complet ou relatif de l'exécutable de l'application. • Si le chemin contient des espaces, spécifiez le chemin complet ; ne pas utiliser de chemin relatif • Si le chemin contient des espaces, il doit être entouré de guillemets
Arguments de ligne de commande	Paramètres à transmettre à l'application au démarrage.
Afficher la console	Créez une fenêtre de console pour le débogueur ; non applicable pour le rattachement à un processus.
Chemins de recherche de symboles	Spécifiez tous les chemins supplémentaires pour localiser les symboles de débogage pour le débogueur ; séparez les chemins par un point-virgule.

Option 2 – Connecter à une application en cours d'exécution

Champ	Action
Débogueur	Sélectionnez Microsoft Native comme plate-forme de débogage.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.
Mode	Sélectionnez le bouton radio Attacher au processus.

Chemins de recherche de symboles	Spécifiez tous les chemins supplémentaires pour localiser les symboles de débogage pour le débogueur. Vous pouvez spécifier un serveur de symboles ici si vous préférez ; séparez les chemins par un point-virgule ou une virgule.

Symboles Déboguer

Pour les applications créées à l'aide du SDK Microsoft Platform, les symboles Déboguer sont écrits dans un fichier PDB d'application lors de la génération de l'application.

Les outils de débogage pour Windows , une API utilisée par le Débogueur d'exécution visuelle , utilisent ces symboles pour présenter des informations significatives aux contrôles Analyseur d'Exécution .

Ces symboles peuvent facilement devenir obsolètes et provoquer un comportement aberrant : le débogueur peut mettre en évidence la mauvaise ligne de code dans l'éditeur alors qu'il se trouve à un point d'arrêt ; il est donc préférable de s'assurer que l'application est créée avant toute session de débogage ou d'enregistrement.

Le débogueur doit indiquer à l'API comment réconcilier les adresses dans l'image en cours de débogage ; pour ce faire, il spécifie un certain nombre de chemins vers l'API qui lui indiquent où rechercher les fichiers PDB.

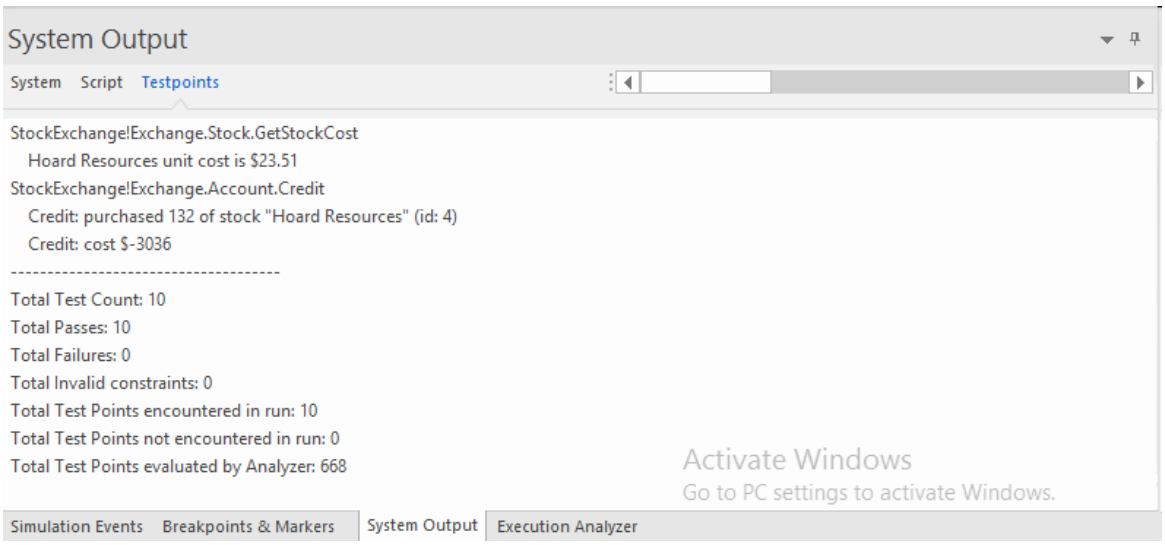
Pour les DLL système (kernel32, mfc90ud) pour lesquelles aucun symbole de débogage n'est trouvé, la Pile d'Appel affiche certaines trames avec uniquement les noms et adresses de module.

Vous pouvez compléter les symboles traduits en transmettant des chemins supplémentaires vers l'API ; vous transmettez des chemins de symboles supplémentaires dans une liste séparée par des points-virgules dans l'onglet ' Déboguer '.

Sortie des Testpoints

La page ' Testpoints ' du script Analyzer vous aide à configurer la sortie d'un Testpoint exécuter .

Par défaut, la sortie est enregistrée dans la fenêtre Sortie système, comme dans cet exemple.



Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page ' Test > Testpoints ' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, puis sélectionnez la page ' Test > Testpoints '

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur > Vue Scripts d'Analyseur
Raccourcis Clavier	Maj+F12

Possibilités

Option	Description
Sortir	Vous pouvez choisir parmi deux options : • « Écran » (valeur par défaut) : la sortie est dirigée vers l'onglet « Testpoints » de la fenêtre Sortie système. • 'Fichier' - La sortie est dirigée vers un fichier
Dossier	Cliquez sur le dossier à utiliser pour les fichiers log Testpoint .
Nom de fichier	Entrez le nom à utiliser pour les fichiers log Testpoint .

Écraser	Lorsque cette option est sélectionnée, le fichier spécifié est écrasé à chaque fois qu'un Testpoint exécuter est exécuté.
Numéro automatique	Lorsque cette option est sélectionnée, la sortie Testpoint est composée du nom de fichier que vous spécifiez et du numéro du Test exécuter ; chaque fois que vous effectuez un Test exécuter le nombre est incrémenté.
Sortie de trace de préfixe avec fonction	Lorsque cette option est sélectionnée, toutes les instructions de trace exécutées pendant l' exécuter de Testpoint sont préfixées par l'appel de fonction en cours.

Scripts de Test

Ces sections expliquent comment configurer la page « Test » d'un script Analyzer pour effectuer des tests unitaires sur votre code. La plupart des utilisateurs l'appliqueront aux scénarios de test NUnit et JUnit. Enterprise Architect accepte les résultats de ces systèmes et peut automatiquement ajouter et gérer l'historique de chaque cas de test unitaire. Pour afficher l'historique du cas, vous devez sélectionner l'élément Class du scénario de test et appuyer sur Alt+2 > Tester .

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page ' Test > Test ' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, puis sélectionnez la page ' Test > Test '

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Menu Contexte de l'analyseur	Test
Raccourcis Clavier	Maj+F12

Actions

Test

Type la commande ou le script Test dans ce champ. Par exemple:

- NUnit - "C:\Program Files\NUnit\bin\nunit-console.exe" "bin\debug\Calculator.exe"
- JUnit - java junit.textui.Testrunner %N

La commande répertoriée dans ce champ est exécutée comme si elle provenait de l' prompt de commande ; par conséquent, si le chemin de l'exécutable ou des arguments contiennent des espaces, ils doivent être entourés de guillemets.

Si vous incluez la string %N dans votre script de test, elle est remplacée par le nom complet qualifié par l'espace de noms de la classe actuellement sélectionnée lorsque le script est exécuté.

Exécuter la commande en tant que

Cliquez sur la flèche déroulante et sélectionnez l'option appropriée :

- Fichier batch - Utilisez cette option pour créer un script shell qui est exécuté dans une fenêtre de commande système ; les variables d'environnement sont accessibles par les commandes de ce script
- Processus - Utilisez cette option pour exécuter un seul programme - la commande doit spécifier le chemin d'accès au programme, ainsi que tous les arguments de ligne de commande ; si le chemin ou les arguments contiennent des espaces, entourez le chemin de guillemets - par exemple : "c:\program files (x86)\java\bin\javac.exe"

Répertoire par défaut

La valeur par défaut est la valeur entrée pour le script Build. Si aucune valeur n'a

été définie pour le script Build, recherchez ou saisissez le chemin du répertoire par défaut dans lequel le processus du script Clean exécuter .

Analyser la sortie Lorsqu'un analyseur est sélectionné, la sortie des tests nUnit et jUnit peut être analysée, enregistrée et gérée à partir du modèle ; (Alt+2 > Tester). Sachez que la sortie n'est capturée que lorsqu'un analyseur est sélectionné.

Hôte distant Type l' ID du système hôte distant et son port ; par exemple, monpc01:7777.

Si vous définissez cette propriété sur #SYSTEMHOST#, le script est envoyé au Windows Service Satellite lors de son exécution sous Windows et au Linux Service Satellite lors de son exécution sous Wine. Les ID de service et les Ports sont définis dans la section 'Options privées - Services' de l'Editeur Scripts d'Analyseur .

Construire d'abord Sélectionnez pour vous assurer que le Paquetage est compilé à chaque fois que vous exécuter le test.

Script Nettoyage

Les constructions incrémentielles consistent à créer uniquement les actifs qui ont changé d'une manière ou d'une autre. Il y a cependant des moments où il est nécessaire de tout reconstruire à partir de zéro. Cette commande est utilisée à ces occasions, pour supprimer les binaires et les fichiers intermédiaires associés à une version ou une configuration particulière. Le projet peut alors être reconstruit. Lorsque vous exécutez l'option de menu « Reconstruire » sur un script, la ou les commandes que vous spécifiez dans ce champ sont exécutées, suivies immédiatement par la commande « Construire » du même script Analyzer. Certains compilateurs proposent des options pour le faire à votre place. Visual Studio, par exemple, dispose du commutateur de ligne de commande " /clean".

Ceci est un exemple de script : devenv.com /Clean Déboguer MyProject.sln

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page « Construire > Nettoyer » ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, puis sélectionnez la page 'Build > Clean'.

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Source > Construire > Nettoyer
Menu Contexte de l'analyseur	Faire le ménage
Raccourcis Clavier	Maj+F12

Aspects

Aspect	Détail
Faire le ménage	Entrez la commande à exécuter lorsque vous sélectionnez « Nettoyer » dans le menu contextuel du script.
Exécuter la commande comme	Cliquez sur la flèche déroulante et sélectionnez l'option appropriée : • Fichier batch - Utilisez cette option pour créer un script shell qui est exécuté dans une fenêtre de commande système ; les variables d'environnement sont accessibles par les commandes de ce script • Processus - Utilisez cette option pour exécuter un seul programme - la commande doit spécifier le chemin d'accès au programme, ainsi que tous les arguments de ligne de commande ; - si le chemin ou les arguments contiennent des espaces, entourez-les de guillemets - par exemple : "c:\program files (x86)\java\bin\javac.exe"
Répertoire par défaut	La valeur par défaut est la valeur entrée pour le script Build. Si aucune valeur n'a été définie pour le script Build, recherchez ou saisissez le chemin du répertoire par défaut dans lequel le processus du script Clean exécuter .

Analyser la sortie	Cliquez sur la flèche déroulante et sélectionnez une méthode pour analyser automatiquement la sortie du compilateur. Si vous sélectionnez cette option, la sortie du script est enregistrée dans la fenêtre Sortie système ; Enterprise Architect analyse la sortie selon la syntaxe que vous spécifiez.
Hôte distant	Type l' ID du système hôte distant et son port ; par exemple, monpc01:7777. Si vous définissez cette propriété sur #SYSTEMHOST#, le script est envoyé au Windows Service Satellite lors de son exécution sous Windows et au Linux Service Satellite lors de son exécution sous Wine. Les ID de service et les Ports sont définis dans la section 'Options privées - Services' de l'Editeur Scripts d'Analyseur .

Créer Scripts

La page 'Build' vous permet de saisir des commandes pour construire votre projet. Vous pouvez utiliser les chemins locaux et les variables d'environnement Enterprise Architect pour composer votre ou vos lignes de commande. Vous pouvez choisir de créer votre propre script de construction, en entrant diverses commandes shell. Vous pouvez également choisir d' exécuter simplement un programme externe ou un fichier batch tel qu'un script Ant.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page 'Build > Build' ou
- Cliquez sur dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, puis sélectionnez la page 'Build > Build'.

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Menu Contexte de l'analyseur	Construire
Raccourcis Clavier	Ctrl+Maj+F12

Construire

Écrivez votre script dans cette zone de texte, à l'aide des commandes du shell Windows ; le format et le contenu de cette section dépendent du compilateur réel que vous utilisez pour créer votre projet. Si le chemin ou les arguments contiennent des espaces, entourez-les de guillemets ; par exemple : "c:\program files (x86)\java\bin\javac.exe".

Voici quelques exemples:

Visual Studio:

"C:\Program Files (x86)\Microsoft Visual Studio 9.0\Common7\IDE\devenv.com" /Reconstruire Déboguer RentalSystem.sln

Visual Studio utilisant un chemin local :

"%VsCompPath%\devenv.exe" /build Déboguer Subway.sln

Java:

C:\Program Files (x86)\Java\jdk1.6.0_22\bin\javac.exe" -g -cp "%classpath%;. " %r*.java

Java utilisant un chemin local :

"%JAVA%\bin\javac.exe" -g -cp "%classpath%;. " %r*.java

Constructions Java génériques (%r) : les fichiers source dans les sous-dossiers peuvent être créés à l'aide du jeton %r. Le jeton a pour effet de provoquer une exécution récursive de la même commande sur tous les fichiers de tous les sous-dossiers, comme le montre l'exemple.

Exécuter la commande comme

Cliquez sur la flèche déroulante et sélectionnez le mode d'exécution du script :

- Fichier batch - Utilisez cette option pour exécuter un script shell dans une fenêtre de commande système ; les variables d'environnement sont accessibles par les commandes de ce script
- Processus - Utilisez cette option pour exécuter la commande en tant que programme unique ; la commande doit spécifier le chemin d'accès au programme, ainsi que tous les arguments de ligne de commande

Répertoire par défaut

Recherchez ou saisissez le chemin du répertoire par défaut dans lequel le processus de script de construction exécuter .

Analyser la sortie

Cliquez sur la flèche déroulante et sélectionnez une méthode pour analyser automatiquement la sortie du compilateur.

Si vous sélectionnez cette option, la sortie du script est enregistrée dans la fenêtre Sortie système ; Enterprise Architect analyse la sortie selon la syntaxe que vous spécifiez.

Hôte distant

Type l' ID du système hôte distant et son port ; par exemple, monpc01:7777.

Si vous définissez cette propriété sur #SYSTEMHOST#, le script est envoyé au Windows Service Satellite lors de son exécution sous Windows et au Linux Service Satellite lors de son exécution sous Wine. Les ID de service et les Ports sont définis dans la section 'Options privées - Services' de l'Editeur Scripts d'Analyseur .

Déployer après la construction

Cochez cette case pour que le script de déploiement soit exécuté immédiatement après la fin de ce script de génération.

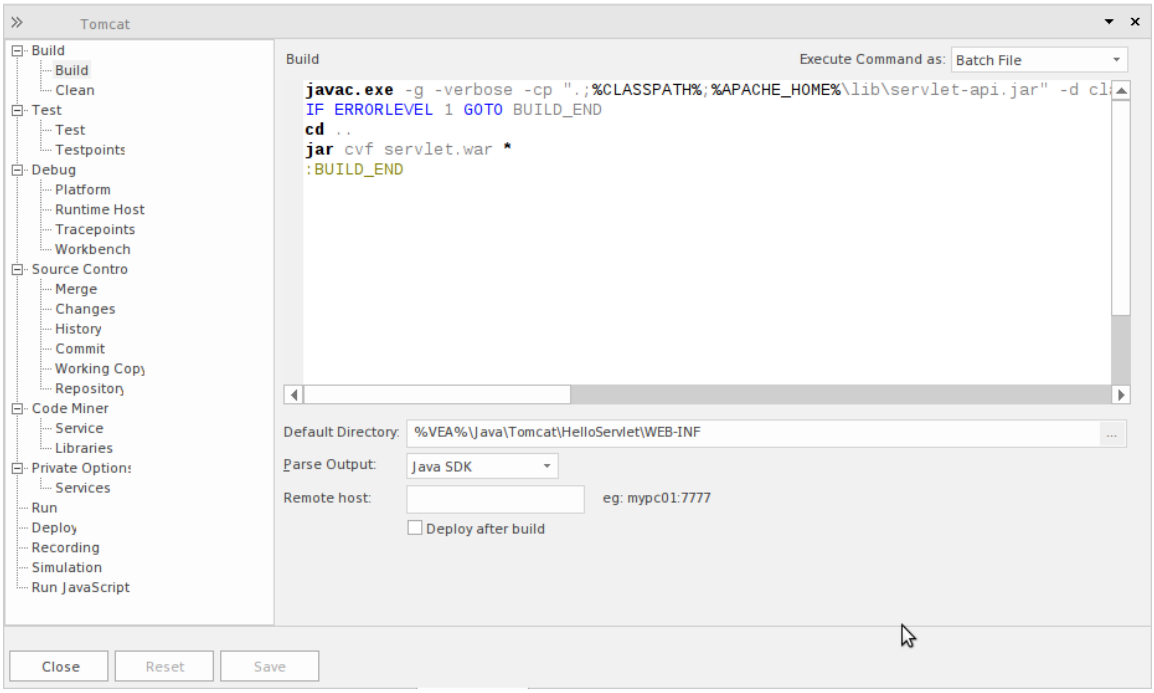
Notes

Pour exécuter le script Build, cliquez sur le Paquetage dans la fenêtre Navigateur et soit :

- Cliquez-droit sur n'importe quelle barre d'outils et sélectionnez 'Analyzer Barres d'Outils | Construire', ou
- Appuyez sur Ctrl+Maj+F12 ou
- Sélectionnez l'option de ruban « Exécuter > Source > Construire > Construire ».

Éditeur de Script Analyseur

L' Éditeur de Script Analyseur dispose d'une interface utilisateur simple, avec une arborescence des scripts à gauche permettant une navigation facile dans les groupes de scripts, et une vue de contenu à droite dans laquelle vous définissez et configurez les scripts.



Accéder

Depuis la fenêtre ' Analyseur d'Exécution ', soit :

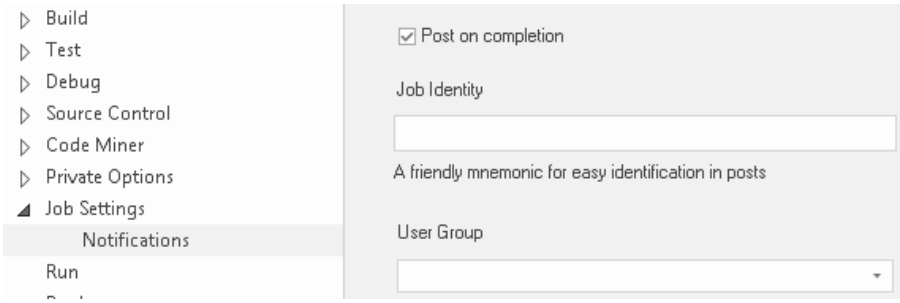
- Double-cliquez sur un script pour le modifier ou
- Cliquez-droit sur un script et sélectionnez l'option 'Modifier'

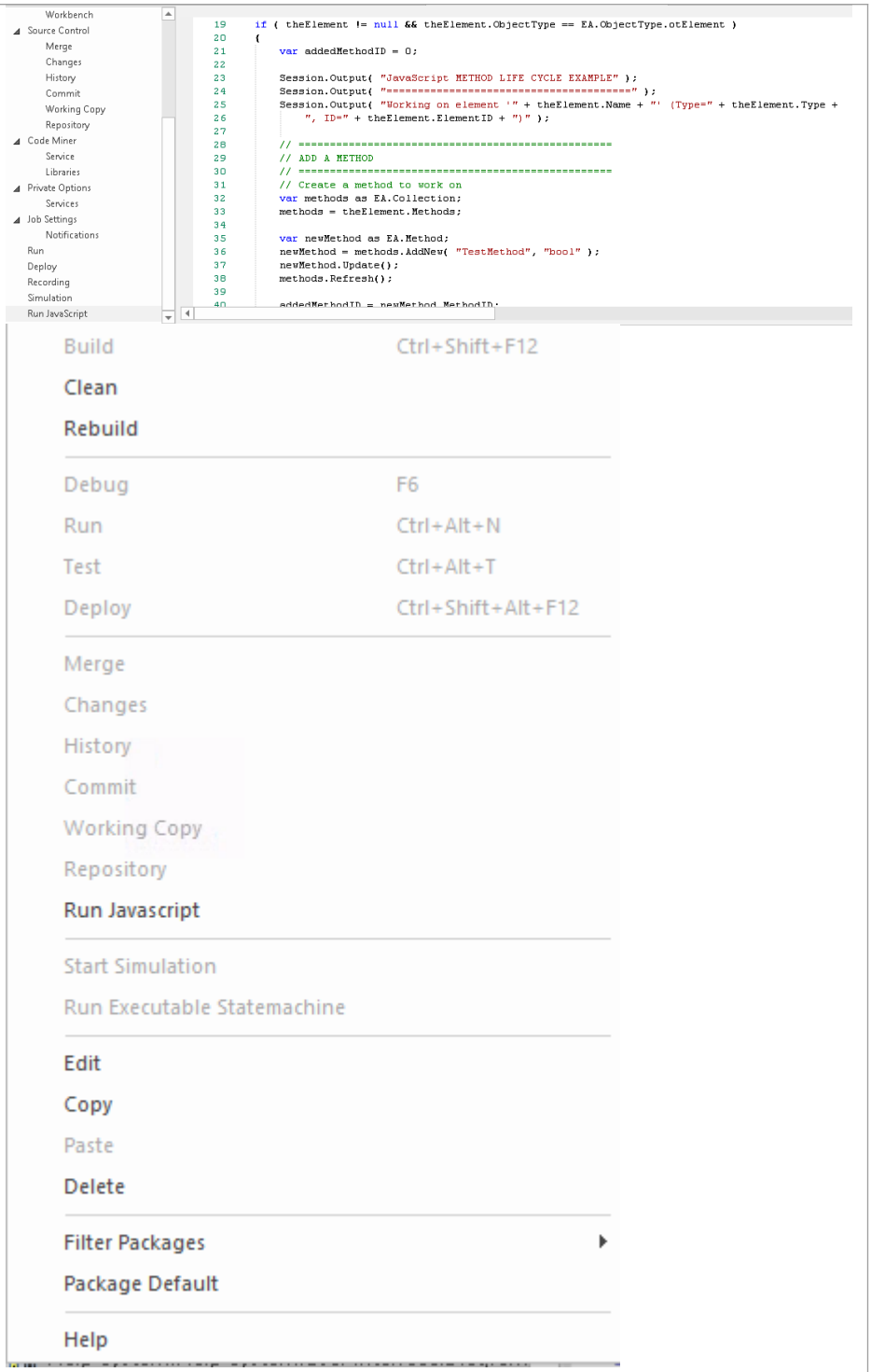
Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Analyseur d'Exécution Scripts

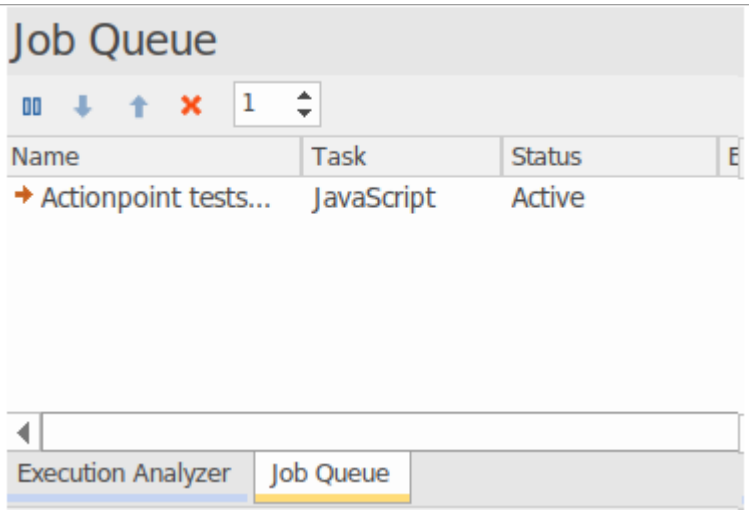
Tâche - Page	Action
Construire - Construire	Entrez le script ou la commande pour créer l'application. Il peut s'agir d'une commande Apache Ant ou Visual Studio, mais peut également être personnalisée en fonction de votre environnement de développement. Note : Pensez à sélectionner un analyseur pour accéder directement au code source en cas d'erreur. Le champ de l'analyseur se trouve sur la même page et offre support pour de nombreuses langues.

Construire - Nettoyer	Entrez le script ou la commande pour nettoyer la version précédente. Il s'agit de la ligne de commande que vous émettez normalement pour construire votre système. Il peut s'agir d'une commande Apache Ant ou Visual Studio en fonction de votre environnement de développement.
Test - Test	Entrez le script ou la commande pour tester l'application. C'est généralement là qu'un appel nUnit ou jUnit peut être configuré, mais il peut tout aussi bien s'agir de n'importe quelle procédure ou programme.
Test - Testpoints	Spécifiez où la sortie d'un Testpoint exécuter est envoyée.
Déboguer - Plateforme	Spécifiez la plateforme de débogage, l'application à déboguer et le mode de débogage (attacher au processus ou exécuter).
Déboguer - Tracepoints	Spécifiez où sont envoyées les sorties des points de trace rencontrés lors d'une session de débogage.
Déboguer - Établi	Pour les projets .NET , l'assembly à charger. Non requis pour Java.
Déboguer - Hôte d'exécution	Permet à Enterprise Architect de lancer le programme à déboguer à l'aide d'une ligne de commande. Ceci est généralement utilisé pour les programmes Mono ou Java qui utilisent un transport socket pour le débogage. Cette commande est exécutée avant que le Débogueur soit exécuter . Le numéro de port spécifié dans cette commande doit être la même valeur passée à l'option 'Port' dans la page Déboguer . Au démarrage du Débogueur , il tentera de se connecter au runtime sur ce Port. En cas de succès, il lie ensuite tous les points d'arrêt et reprend le programme, qu'il suppose suspendu. Java et Mono ont tous deux des options de ligne de commande sur le transport de débogage pour suspendre initialement le processus jusqu'à ce que le Débogueur se connecte.
Contrôle de source - Fusionner	Il s'agit du script qui s'exécute lorsque l'option « Fusionner » est choisie dans le menu contextuel d'un script Analyzer. Il fournit un endroit pour exécuter un programme ou un script shell pour examiner les différences entre les fichiers sources.
Contrôle de source – Modifications	Il s'agit du script qui s'exécute lorsque l'option « Modifications » est choisie dans le menu contextuel d'un script Analyzer. Il fournit un endroit pour exécuter un programme de contrôle de source tel que « svn » qui peut répertorier les modifications actuelles apportées à un référentiel de contrôle de source.
Contrôle de source - Historique	Il s'agit du script qui s'exécute lorsque l'option « Historique » est choisie dans le menu contextuel d'un script Analyzer. Il fournit un endroit pour exécuter un programme de contrôle de source tel que « svn » qui peut répertorier un historique des modifications apportées à un référentiel de contrôle de source.
Contrôle de source - Validation	Il s'agit du script qui s'exécute lorsque l'option « Commit » est choisie dans le menu contextuel d'un script Analyzer. Il fournit un endroit pour exécuter un programme de contrôle de source tel que « svn » qui pourrait valider les modifications apportées à une copie de travail de contrôle de source.
Contrôle de source - Copie de travail	Il s'agit du script qui s'exécute lorsque l'option « Copie de travail » est choisie dans le menu contextuel d'un script Analyzer. Il fournit un endroit pour exécuter un programme de contrôle de source tel que « svn » pour effectuer des actions sur la copie de travail actuelle d'un référentiel source.

Contrôle des sources - Référentiel	Il s'agit du script qui s'exécute lorsque l'option ' Référentiel ' est choisie dans le menu contextuel d'un script Analyzer. Il fournit un endroit pour exécuter un programme de contrôle de source tel que « svn » pour effectuer des actions sur un référentiel source.
Code Miner - Service	Dans cette section, vous pouvez choisir le fonctionnement du service Code Miner . Vous pouvez choisir soit un serveur distant, soit utiliser les bibliothèques localement.
Code Miner - Bibliothèques	Cette section fournit un espace pour la gestion des bibliothèques Code Miner . Ici, vous pouvez créer des bibliothèques basées sur une base de code ou un référentiel de projet. Les bibliothèques Code Miner créées ici peuvent être recherchées à l'aide de requêtes mFQL. Les requêtes composées en mFQL peuvent être utilisées pour rechercher une ou plusieurs bibliothèques en une seule opération.
Options privées - Services	C'est ici que sont configurés l'adresse IP et le port des services Enterprise Architect Satellite pour Linux et Windows . Ces services fournissent support à l'échelle de l'entreprise pour les fonctions de gestion du système et les scénarios de débogage à distance.
Paramètres du travail	La plupart des commandes contenues dans un script Analyzer sont exécutées en tant que tâches dans la file d'attente des tâches. Chaque script peut être configuré pour publier une notification à un groupe d'utilisateurs spécifié lorsqu'une tâche spécifiée est terminée. Le groupe 'Paramètres du travail' propose l'option 'Notifications', qui affiche les champs dans lesquels vous saisissez l'identité du travail dans le cadre du texte à afficher aux membres du groupe d'utilisateurs Mail de Modèle , ainsi que le nom du groupe d'utilisateurs.  • Publier une fois terminé : cochez cette case pour activer les deux autres champs. • Identité du travail : Type le texte à afficher, qui doit également identifier le travail ; par exemple : « Installateurs terminés » • Groupe d'utilisateurs - Cliquez sur la flèche déroulante et sélectionnez le groupe d'utilisateurs Mail de Modèle approprié.
Exécuter JavaScript	Dans cette section, vous pouvez créer et stocker un JavaScript que vous pouvez exécuter en sélectionnant l'option ' Exécuter JavaScript ' dans le menu contextuel d'un script d'analyseur.



Lorsque cette option est sélectionnée, une tâche est créée dans la fenêtre File d'attente des tâches.

	
Exécuter	Entrez une commande pour exécuter une application.
Déployer	Entrez un script ou une commande pour déployer le projet. Construisez votre fichier jar. Déployez sur votre appareil, un émulateur ou un serveur Tomcat. Publier un site Web. C'est à vous.
Enregistrement	Votre diagramme Séquence ressemble-t-il au quadrillage national ? Réduisez l'encombrement avec des filtres. Les filtres définissent des zones d'exclusion dans votre base de code qui peuvent réduire considérablement tout « bruit » enregistré. Même un bruit précis n'est pas toujours utile.
Simulation	Terminez la configuration de Simulation Control.

Gestion Scripts d'Analyseur

La fenêtre Analyseur d'Exécution vous permet de gérer tous les scripts Analyseur du modèle. Vous pouvez utiliser les boutons de la barre d'outils de la fenêtre ou les options du menu contextuel du script pour contrôler les tâches de script. Scripts sont répertoriés par Paquetage ; la liste affiche uniquement Paquetages pour lesquels des scripts Analyzer sont définis. Chaque utilisateur peut définir son propre script actif, indépendant des autres utilisateurs du même modèle ; l'activation d'un script par un utilisateur n'a pas d'impact sur les scripts actuellement actifs des autres utilisateurs ni sur les scripts disponibles pour eux. Le script actif régit le comportement de l' Analyseur d'Exécution ; en choisissant la commande build dans un menu, par exemple, ou en cliquant sur le bouton Déboguer d'une barre d'outils.

Il peut y avoir de nombreux Paquetages avec des scripts répertoriés dans la fenêtre. Pour vous aider à localiser et à isoler un Paquetage particulier, utilisez les options du menu contextuel « Filtrer Paquetages », décrites dans le tableau *Options Menu Contexte* de cette rubrique.

Note que lorsqu'un script est mis en surbrillance dans la fenêtre de l' Analyseur d'Exécution , les options du menu contextuel de la fenêtre et les boutons de la barre d'outils de la fenêtre Analyseur d'Exécution fonctionneront sur le script en surbrillance. Cependant, les options Analyseur d'Exécution sur n'importe quel ruban ou barre d'outils flottante, ou dans le Débogueur (qui ne sont pas sur la fenêtre) utiliseront toujours le script Analyzer par défaut - celui dont la case est cochée à côté du nom du script.

Accéder

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Options barre d'outils

Bouton de la barre d'outils	Action
	Accès rapide aux fenêtres principales de l'Analyseur telles que Pile d'Appel ou Variables Locales, ainsi qu'aux fonctionnalités avancées : • Profilage • Enregistrement • Testpoints • Simulation
	Créez et modifiez un nouveau script d'analyseur pour le Paquetage sélectionné, sous Linux ou Windows ™ .
	Exporter Scripts . Exportez un ou plusieurs Scripts d'Analyseur vers un fichier XML, qui peut être utilisé pour importer les scripts dans un autre modèle. La dialogue « Analyseur d'Exécution : Exporter » s'affiche à partir de laquelle vous sélectionnez le ou les scripts à exporter, suivie d'une prompt pour le nom et l'emplacement du fichier cible.

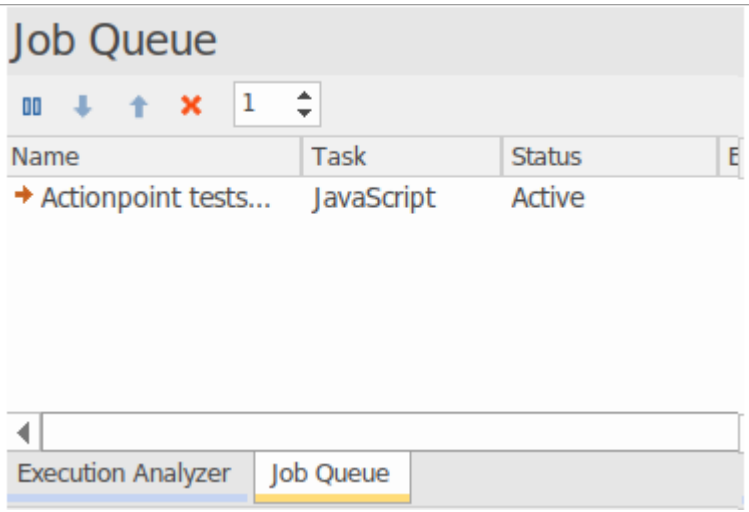
	Importer Scripts . Importez un ou plusieurs Scripts d'Analyseur dans le modèle courant à partir d'un fichier XML préalablement exporté. La dialogue « Rechercher Paquetage » s'affiche, dans laquelle vous sélectionnez le Paquetage dans lequel importer les scripts, suivie d'une prompt indiquant le nom et l'emplacement du fichier source.
	Exécutez la commande 'Build' du script actif.
	Annulez la commande 'Build' actuellement en cours.
	Exécutez la commande ' Exécuter ' du script actif.
	Exécutez la commande ' Test ' du script actif.
	Exécutez la commande 'Déployer' du script actif.

Options Menu Contexte

Cliquez-droit sur le script souhaité ou Paquetage pour afficher les menus contextuels.

Option	Action
Ajouter un nouveau script	Ajoutez un nouveau script au Paquetage sélectionné. La fenêtre Analyseur d'Exécution s'affiche, affichant la page 'Build'.
Coller le script	Collez un script copié du presse-papiers d' Enterprise Architect dans le Paquetage sélectionné. Vous pouvez coller le script copié plusieurs fois ; chaque copie porte le suffixe « Copie ». Pour renommer le script copié, appuyez sur F2 et remplacez le nom du script.
Exporter Scripts	Exportez les scripts du Paquetage sélectionné. La dialogue « Analyseur d'Exécution : Exporter » s'affiche, à partir de laquelle vous sélectionnez le ou les scripts à exporter, suivie d'une prompt pour le nom et l'emplacement du fichier cible.
Importer Scripts	Importez des scripts à partir d'un fichier .XML dans le Paquetage sélectionné. Une prompt s'affiche pour le nom et l'emplacement du fichier source.
Filtrer Paquetages	Affiche un sous-menu d'options pour : • Entrez le chemin Paquetage par rapport auquel filtrer la liste des Paquetages . Lorsque vous sélectionnez l'option « Filtrer Paquetages », une prompt s'affiche pour accepter le chemin Paquetage actuellement sélectionné, ou vous pouvez supprimer les éléments *de fin* du chemin pour spécifier un ensemble plus grand. des Paquetages ; lorsque vous cliquez sur le bouton OK , le premier Paquetage de la liste est développé pour lister les scripts qu'il contient • Basculez entre l'affichage de la liste complète des Paquetages et le masquage

	de la liste complète des Paquetages pour exposer uniquement le Paquetage actuellement sélectionné. • Supprimez le filtre actuellement actif pour afficher la liste complète des Paquetages
Sélectionnez dans Navigateur de projet	Mettez en surbrillance le Paquetage sélectionné dans la fenêtre Navigateur . Affichez la fenêtre Navigateur , qui est maintenant développée pour afficher le Paquetage en surbrillance.
Construire	Exécutez la commande 'Build' du script sélectionné.
Faire le ménage	Exécutez la commande 'Clean' du script sélectionné.
Reconstruire	Exécutez les commandes 'Clean' et 'Build' du script sélectionné.
Déboguer	Exécutez la commande ' Déboguer ' du script sélectionné.
Exécuter	Exécutez la commande ' Exécuter ' du script sélectionné.
Test	Exécutez la commande ' Test ' du script sélectionné.
Déployer	Exécutez la commande 'Déployer' du script sélectionné.
Fusionner	Exécutez la commande de contrôle de source « Fusionner » du script sélectionné.
Changements	Exécutez la commande de contrôle de source « Modifications » du script sélectionné.
Histoire	Exécutez la commande de contrôle de source 'Historique' du script sélectionné.
Commettre	Exécutez la commande de contrôle de source 'Commit' du script sélectionné.
Copie de travail	Exécutez la commande de contrôle de source « Working Copy » du script sélectionné.
Référentiel	Exécutez la commande de contrôle de source ' Référentiel ' du script sélectionné.
Exécuter JavaScript	Exécutez la commande ' Exécuter JavaScript ' du script sélectionné. Lorsque cette option est sélectionnée, une tâche est créée dans la fenêtre File d'attente des tâches.

	
Démarrer Simulation	Démarrer la simulation référencée par la page 'Analyzer Script Simulation '.
Exécuter Exécutable Statemachine	Démarrer une simulation de l'artefact Executable Statemachine sélectionné.
Modifier	Ouvrez le script sélectionné dans l'Editeur Scripts d'Analyseur .
Copie	Copiez le script sélectionné dans le presse-papiers Enterprise Architect .
Pâte	Collez le script copié le plus récemment dans le même Paquetage que le script sélectionné. Vous pouvez coller le script copié plusieurs fois ; chaque copie porte le suffixe « Copie ». Pour renommer le script copié, appuyez sur F2 et remplacez le nom du script.
Supprimer	Supprimez le script sélectionné ; il n'y a aucune prompt de confirmation. Pour supprimer un Paquetage de la fenêtre Analyseur d'Exécution , supprimez les scripts du Paquetage . Lorsque le dernier script est supprimé, le Paquetage n'est plus répertorié.
Paquetage par défaut	Définissez le script sélectionné comme script par défaut pour le Paquetage . L'icône à gauche du script change de couleur ; tout défaut Paquetage précédent revient à la normale.