

## Scénarios de tests d'intégration/fonctionnels



Partenaire : Triskell

Nom de l'outil : Kermeta Samples

Date test : 23 avril 2009

Version de l'outil : 1.3.0

Plateforme de test (OS + processeur) : Linux

Organisme testeur : cellule

Nom du testeur : vmahe

## Récapitulatif des tests

Commentaires du testeur.....	1
Scénario A : installation des exemples.....	2
Scénario B : exemple arabic2roman.....	3
Scénario C : exemple class2RDBMS.....	3
Scénario D : exemple fileIO.....	4
Scénario E : exemple kmlogo.....	4
Scénario F : exemple kunit.....	4
Scénario G : exemple prettyprinter.....	5
Scénario H : exemple kermeta2docbook.....	5
Scénario I : aide en ligne.....	5

## Commentaires du testeur

## Scénario A : installation des exemples

Remarques : à partir de la version 1.2.0 de Kermeta

Action	Résultat attendu	OK	Commentaires
Dans le Package Explorer : - clic droit puis « New » - « Example... » - choisir le dossier « Kermeta samples » puis « Kermeta samples » - clic sur « Finish »	Le projet <i>fr.irisa.triskell.kermeta.samples</i> doit être créé	<b>Oui</b>	
Dans le Package Explorer, développer le projet <i>fr.irisa.triskell.kermeta.samples</i>	Les dossiers suivants doivent apparaître : - arabic2roman - class2RDBMS - fileIO - kmlogo - kunit_sample - <del>prettyprinter</del> (non implanté)	<b>Oui</b> <b>et</b> <b>Non</b>	Pretty Printer toujours présent (mais pas la version aspects faite récemment)
Dans le menu « Run » choisir « Open Run Dialog »	Les launchers suivants doivent apparaître en tant que Kermeta Applications : - Arabic to Roman translation - Class to RDBMS transformation - Files operations in Kermeta - KmLogo turtle - <del>Kmt pretty printer</del> (non implanté) - KUnit unit testing	<b>Oui</b> <b>et</b> <b>Non</b>	Pretty Printer toujours présent (mais pas la version aspects faite récemment)

## Scénario B : exemple arabic2roman

Remarques : --

Action	Résultat attendu	OK	Commentaires
Dans le menu « Run » / « Open Run Dialog », choisir le launcher « Arabic to Roman translation » puis cliquer sur « Run »	Dans la console s'affichent un nombre décimal (3999) et son équivalent en chiffres romains	Oui	

## Scénario C : exemple class2RDBMS

Remarques : --

Action	Résultat attendu	OK	Commentaires
Dans le Package Explorer, développer le répertoire <i>class2DBMS</i> puis le sous-répertoire <i>metamodels</i> . Sur le méta-modèle <b>ClassMM.core</b> , clic droit puis « EPackages Registration » / « Register EPackages »	Eclipse affiche la vue « EMF Registered Packages » avec une ligne pour <b>ClassMM</b>	Oui	
Sur le méta-modèle <b>RDBMSMM.core</b> , clic droit puis « EPackages Registration » / « Register EPackages »	Eclipse affiche la vue « EMF Registered Packages » avec une ligne pour <b>RDBMSMM</b>	Oui	
Dans le menu « Run » / « Open Run Dialog », choisir le launcher « Class to RDBMS transformation » puis cliquer sur « Run »	Dans la console s'affichent les <b>create</b> des tables et colonnes	Oui	

## Scénario D : exemple fileIO

Remarques : --

Action	Résultat attendu	OK	Commentaires
Dans le Package Explorer, développer le répertoire <i>fileIO</i> . Sur le fichier <b>TextFileIOSample.kmt</b> , double clic	L'éditeur Kermeta s'affiche, avec la coloration syntaxique et sans erreurs.	Oui	
Dans le menu « Run » / « Open Run Dialog », choisir le launcher « File operations in Kermeta » puis cliquer sur « Run »	Dans le répertoire « fileIO » apparaît un fichier « myFile.txt ». Dans la console s'affiche un message « Hello World! ».	Oui	

## Scénario E : exemple kmlogo

Remarques : --

Action	Résultat attendu	OK	Commentaires
Dans le menu « Run » / « Open Run Dialog », choisir le launcher « KmLogo turtle » puis cliquer sur « Run »	Dans la console s'affichent des commandes pour tracer un carré de 100	Oui	

## Scénario F : exemple kunit

Remarques : --

Action	Résultat attendu	OK	Commentaires
Dans le Package Explorer, développer le répertoire <i>kunit_sample</i> . Sur le fichier <b>my_testfile.kmt</b> , double clic	L'éditeur Kermeta s'affiche, avec la coloration syntaxique et sans erreurs.	Oui	
Dans le menu « Run » / « Open Run Dialog », choisir le launcher « KUnit unit testing » puis cliquer sur « Run »	Dans la console s'affichent : - des stack traces - 2 failures - 1 error - le summary	Oui	

## Scénario G : exemple prettyprinter

Remarques : l'exemple PrettyPrinter n'étant pas finalisé, il ne doit pas être packagé

Action	Résultat attendu	KO	Commentaires
Ouvrir le menu « Run » / « Open Run Dialog »	Il ne doit pas y avoir de launcher « prettyprinter »	Non	Encore présent

## Scénario H : exemple kermeta2docbook

Remarques : l'exemple Kermeta2docbook n'étant pas finalisé, il ne doit pas être packagé

Action	Résultat attendu	OK	Commentaires
Ouvrir le menu « Run » / « Open Run Dialog »	Il ne doit pas y avoir de launcher « kermeta2docbook »	Oui	

## Scénario I : aide en ligne

Remarques : --

Action	Résultat attendu	KO	Commentaires
Dans la barre de menu Eclipse, cliquer sur « Help » puis « Help Contents »	Un item « Kermeta » doit apparaître	Oui	
Sélectionner l'item « Kermeta » et le déplier	Des rubriques et/ou une première page de documentation doivent apparaître	Oui	
Passer en revue chaque rubrique et page de documentation	Les pages ne doivent pas comporter d'images absentes (marquées par un cadre vide) et les liens doivent être fonctionnels	Non	Trop nombreux liens cassés