

Comment appeler un WebService depuis une application générée avec FormPublisher

1. Introduction

Ce tutorial-cookbook montre comment appeler un WebService simple, depuis une application générée avec FormPublisher. Par "simple", il est question d'un WebService qui prend quelques paramètres en entrée et qui fournit quelques données en résultats (à la différence d'un WebService plus sophistiqué qui fournirait des listes de réponses de tailles dynamiques).

L'exemple porte sur un service de conversion monétaire : à partir de deux devises, le système tiers fournit le taux de change. Pour suivre ce cookbook, il faut au moins avoir suivi "<u>Comment faire un formulaire</u>".

2. Initialiser un nouveau projet/document

Rappel : les WebServices sont fournis par le module CommunicationPack.

Initialiser un nouveau projet/document pour cette démonstration ou ouvrir un projet/document existant.

Dans le document <u>JForm</u>, ajouter une section "Input" pour recevoir les paramètres et une section "Output" pour recevoir les résultats. Ces noms de <u>Section</u> restent au libre choix de l'auteur FormPublisher.

3. Recueillir les paramètres

Dans la Section "Input", placer deux champs pour le choix des devises :

Devise de départ

Devise d'arrivée

Ces champs portent ici les noms "deviseDepart" et "deviseArrivee".

Accessoirement, le premier est une <u>ListBox</u> et le second une <u>ComboBox</u>, mais cela n'a aucune importance dans le cadre de cet exemple de <u>WebService</u>.

4. Préparer le résultat

Dans la Section "Output", placer un champ pour recevoir le résultat :

Taux de conversion / :

5. Poser le WebService

Dans la Section "Output", avant le réceptacle du résultat, placer un élément WebService.

Insert Before	0	Paragraph
Insert Inside		List
Insert After		Variable
Replace by		Object
Cut		Object
cut	0	WebService
Сору		Table
Paste Before	<u> </u>	Table
Paste Inside	۲	Include
Paste After	0	QuestionSet

Cet élément <u>WebService</u> constitue le centre de cette démonstration. Il provoquera un appel au WebService tiers lors de l'affichage de la page sur laquelle il est posé.



6. Pointer le WSDL

Un WebService expose son offre à ses consommateurs à travers un document WSDL (WebService Definition Language). Cette norme WSDL permet aux consommateurs du WebService de s'y "brancher" automatiquement. Le document WSDL peut être fourni sous la forme d'un fichier ou d'un lien web. Le Studio va extraire automatiquement les informations nécessaires publiées dans le WSDL.

Ouvrir les propriétés de l'élément WebService :

Ż	"WebService" Properties	
	Definition	
	Wsdl	
4	Service	
	Service	
4	Port	
	Port	
4	Data Mappings	
	InputMapping	
	OutputMapping	
4	Operation	
	Operation	
4	Comments	
	Comments	-
Co	mments mments	Close

Pointer le WSDL :

ŗ	"WebService" Properties	
	Definition	
	Wsdl	http://www.webservicex.net/CurrencyCo
4	Service	
	Service	
4	Port	
	Port	
4	Data Mappings	
	InputMapping	
	OutputMapping	
4	Operation	
	Operation	
4	Comments	
	Comments	-
W: We	sdl eb Service Definition	Clara
		Close

Par ce "branchement" direct du projet FormPublisher au WSDL du WebService, le projet va pouvoir à présent spécifier sa requête et ses paramètres.

7. Sélectionner l'action

way

Comme une offre WSDL peut comporter plusieurs choses, les propriétés "Service", "Port" et "Operation" servent à sélectionner l'action retenue (ici, l'obtention d'un taux de conversion monétaire). Sélectionner le "Service" :

Ż	"WebService" Properties	
4	Definition	
	Wsdl	http://www.webservicex.net/CurrencyConve
4	Service	
	Service	CurrencyConvertor 🔹
▲	Port	CurrencyConvertor
	Port	
4	Data Mappings	
	InputMapping	
	OutputMapping	
4	Operation	
	Operation	
4	Comments	
	Comments	
Se	rvice	
We	b Service Binding	
		Close

Sélectionner le "Port" :

Ż	"WebService" Properties	
	Definition	
	Wsdl	http://www.webservicex.net/CurrencyConve
⊿	Service	
	Service	CurrencyConvertor
4	Port	
	Port	CurrencyConvertorSoap 🔹
4	Data Mappings	CurrencyConvertorSoap
	InputMapping	CurrencyConvertorSoap12
	OutputMapping	CurrencyConvertorHttpGet
4	Operation	CurrencyConvertorrapPost
	Operation	
4	Comments	
	Comments	_
Po We	rt eb Service Binding	Close

Sélectionner l'"Opération" :

way

Ż	"WebService" Properties	
	Definition	▲
	WsdI	http://www.webservicex.net/CurrencyConve
4	Service	
	Service	CurrencyConvertor
⊿	Port	
	Port	CurrencyConvertorSoap
4	Data Mappings	
	InputMapping	
	OutputMapping	
4	Operation	
	Operation	ConversionRate 🔹
4	Comments	ConversionRate
	Comments	-
		-
Ор	eration	
We	b Service Operation	
		Close
L		

A l'issue de ces sélections, l'action du WebService est définie et le paramétrage va pouvoir être réalisé.

8. Brancher les paramètres

La mise en correspondance entre les champs de saisie du formulaire et les paramètres d'appel du WebService est appelée "Input Mapping".

Ouvrir les propriétés du WebService et sélectionner "InputMapping" :

ŗ	"WebService" Properties	
4	Definition	<u> </u>
	Wsdl	http://www.webservicex.net/CurrencyConver
4	Service	
	Service	CurrencyConvertor
4	Port	
	Port	CurrencyConvertorSoap
⊿	Data Mappings	
	InputMapping	
	OutputMapping	
4	Operation	
	Operation	ConversionRate
4	Comments	
	Comments	
In In	outMapping out Parameters	
		Close

way

L'assistant de mise en correspondance s'ouvre pour relier aisément les champs de saisie du formulaire aux paramètres attendus par le WebService.

	🎢 Input Ma	pping			
	Template	Header			
J	Standard	Advanced			
	FromCurre	ency	-	\$(Data1)	-
	ToCurrenc	У	-	\$(deviseDepart)	
I				\$(deviseArrivee)	
I				\$(taux)	
	ОК	Cancel	Test Web Service		

A gauche figurent les paramètres attendus, à droite, des listes déroulantes alimentées par le dictionnaire de données du formulaire. Remplacer le premier paramètre par le champ "deviseDepart" et le second par "deviseArrivee".

	🎢 Input Ma	pping				
	Template	Header				
J	Standard	Advanced				
I	FromCurre	ency	-	\$(deviseDepart)		•
	ToCurrenc	у	-	\$(Data2)		•
I				\$(deviseDepart)		
I				\$(deviseArrivee)		
ŀ				\$(taux)		
	ОК	Cancel	Test Web Service			



9. Tester les paramètres

Le Studio permet de tester l'InputMapping du WebService directement depuis l'assistant, sans encore générer l'application. Une fois les correspondances renseignées, cliquer sur "Test Web Service".

	🎢 Input Ma	pping					
	Template	Header					
	Standard	Advanced					
I	FromCurre	ncy	-	\$(deviseDepar	t)		•
I	ToCurrenc	У	-	\$(deviseArrive	e)		•
I							
ŀ							
	ОК	Cancel	Test Web Service				
);=	$_{\rm g} \frac{{\sf D}{\sf ata}}{{\sf T}} = Y_{\rm H} \frac{{\sf D}}{{\sf T}}$	Port	ta Mappings	Neb Service	CurrencyConvertorSoa	р	

Un autre assistant apparaît pour tester le WebService depuis le Studio :

📝 WebSen	vice Test						
Server Url:	http://www.webse	ervicex.net/CurrencyConvertor.a	asmx				
Standard	Advanced						
[Input -					Output		
FromCur	rency	EUR	*		ConversionRateResult	1.4242	
ToCurrer	псу	USD		Run			
			-				
				· · · ·	s		
ОК							

Saisir deux valeurs dans "Input", cliquer sur "Run" et constater le résultat dans "Output". Cet "Output" va maintenant être raccordé au réceptacle de résultat, comme l'ont été les champs de saisie aux paramètres.

10. Brancher le résultat

Ouvrir les propriétés du WebService pour sélectionner cette fois-ci l'Output Mapping :

Ż	"WebService" Properties	
	Definition	▲
	Wsdl	http://www.webservicex.net/CurrencyConver
4	Service	
	Service	CurrencyConvertor
4	Port	
	Port	CurrencyConvertorSoap
⊿	Data Mappings	
	InputMapping	
	OutputMapping	
4	Operation	
	Operation	ConversionRate
4	Comments	
	Comments	
Ou	tputMapping	
Ou	tput Data	
		Close

L'assistant affiche à gauche les données renvoyées par le WebService et permet des mises en correspondance avec le dictionnaire de données via les listes déroulantes à droite :

🖍 Output Mapping			
Standard Advanced			
ConversionRateResult	->	{Data1}	-
		{deviseDepart} {deviseArrivee} {taux}	
OK Cancel			

Le paramétrage "in/out" du WebService est achevé.

11. Gérer les anomalies

way

A l'intérieur de l'élément <u>WebService</u>, il est possible de poser des <u>OnErrorAction</u>. Leur @ActionType définit le type d'anomalie (pas de réponse du tout ou réponse incorrecte). Leur @ErrorType définit le degré de gravité (simple avertissement ou erreur bloquante).

Web	Service
\odot	OnErrorAction
	OnErrorMessage
	Le WebService appelé ne répond pas (no answer).
	OnErrorAction
0	OnErrorMessage
	Le WebService n'a pas fourni de résultat (no data).

Chaque OnErrorAction peut contenir un OnErrorMessage pour renseigner l'utilisateur sur le déroulement des événements.

12. Générer et lancer l'application

Générer, lancer, tester, empaqueter, déployer l'application s'effectue comme à l'ordinaire par Build, Run, Package, etc. Un premier test d'intégration avec le WebService peut être fait depuis le Studio, via Build+Run. Un test plus proche de l'environnement de production sera obtenu par Build+Package, suivi du déploiement de l'application générée (fichier.war) sur un serveur de test. Si les essais sont concluants, l'application est déjà prête au passage en production.

13. Conclusion

En suivant pas à pas ce tutorial-cookbook, un utilisateur (non informaticien) formé à FormPublisher peut **concevoir et** réaliser une application web qui utilise des WebServices de type SOAP.

Le fournisseur du WebService n'a qu'à publier son offre au format standard WSDL. A partir du WSDL, le Studio automatise les branchements des paramètres et du résultat.

A l'élément <u>WebService</u> de FormPublisher sont adjoints <u>OnErrorAction</u> et <u>OnErrorMessage</u> pour traiter dynamiquement les éventuelles anomalies du WebService lors de l'exécution de l'application générée.