# Plugin esup-utils-channels-mag-message

Ce plugin permet d'afficher un message à l'utilisateur.

Comme tout plugin, son utilisation n'est possible qu'après l'avoir enregistré dans MainChannel :

```
Message.register(this);
```

Quatre méthodes sont accessibles aux développeurs pour réaliser différents types de messages :

## Message.message(MainChannel, ChannelRuntimeData, MessageBean);

Cette méthode permet d'afficher un message simple associé à un type d'évênement (erreur, warning, information). Ce message est bloquant est permet de traiter une erreur fatale :

```
Message.message(mainChannel, runtimeData, new MessageBean("Erreur    fatale"));
```


Par défaut le type d'évênement est une erreur mais il possible de forcer un type différent :

```
Message.message(mainChannel, runtimeData, new MessageBean("Opération    terminée", Message.INFO));
```


```
Message.message(mainChannel, runtimeData, new MessageBean("Pensez à    effectuer les modifications", Message.
WARNING));
```


```
Message.message(mainChannel, runtimeData, new MessageBean("Erreur fatale",    Message.ERROR));
```

# Message.message(MainChannel, ChannelRuntimeData, MessageBean, String);

Cette méthode est identique à la précédente sauf qu'elle n'est pas bloquante. Un bouton 'OK' apparait dans la feuille et redirige vers l'action définie comme quatrième paramètre :

```
Message.message(mainChannel, runtimeData, new MessageBean("Cliquez    sur OK pour continuer", Message.INFO),
"actionSuivante");
```

# Message.message(MainChannel, ChannelRuntimeData, MessageBean, String, Hashtable);

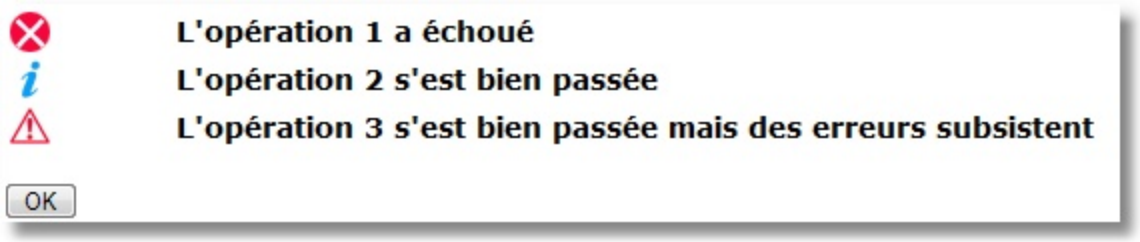
Cette méthode est identique à la précédente sauf que cette fois il est possible de transmettre des paramètres à l'action suivante. Tous les couples clé / valeur contenus dans la Hashtable sont transmis à l'action suivante et pourront être récupérés dans les ChannelRuntimeData. Ces paramètres doivent être des chaînes de caractères :

```
Hashtable parameters = new Hashtable();

parameters.put("att1", "val1");

parameters.put("att2", "val2");

Message.message(mainChannel, runtimeData, new MessageBean("Cliquez sur    OK pour continuer", Message.INFO),
"actionSuivante", parameters);
```

# Message.message(MainChannel, ChannelRuntimeData, Vector, String, Hashtable);

Cette méthode permet d'afficher une liste de messages de types différents. Comme la précédente il est possible de spécifier des paramètres qui seront récupérés par l'action suivante :

```
Vector messages = new Vector();

messages.add(new MessageBean("L'opération 1 a échoué"));

messages.add(new MessageBean("L'opération 2 s'est bien passée",    Message.INFO));

messages.add(new MessageBean("L'opération 3 s'est bien passée    mais des erreurs subsistent", Message.
WARNING));

Hashtable parameters = new Hashtable();

parameters.put("att1", "val1");

parameters.put("att2", "val2");

Message.message(mainChannel, runtimeData, messages, "actionSuivante",    parameters);
```




Dans le cas où on ne souhaite pas passer de paramètres à l'action suivante, il suffit de passer une Hashtable vide comme paramètre.