Formation JAVA EE

Marseille, *du 6 juin au 2 août 2012*

Thomas Bailet : bailet.thomas@gmail.com

Struts

Le framework Struts (charpente) a été conçu pour faciliter la réalisation de la couche présentation pour les applications Web J2EE.

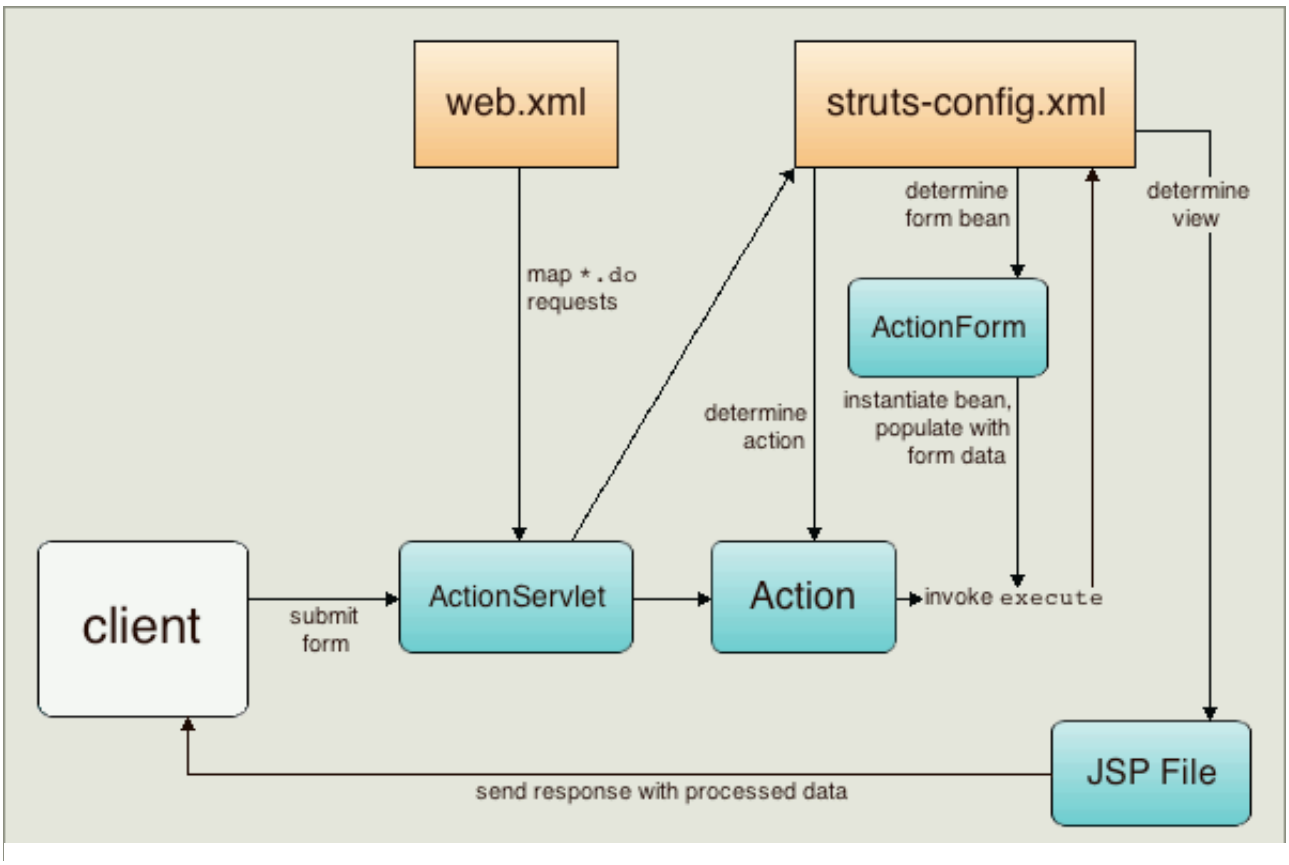
Il a été initialisé par le concepteur principal de Tomcat, Craig R. McClanahan et de nombreux développeurs bénévoles y ont collaboré.

Le projet Struts est actuellement hébergé par l'ASF (Apache Software Foundation)

Principes de Struts

Struts propose une architecture MVC II inspirée du design pattern Front Controller. Dans une telle architecture, il n'existe qu'un seul et unique contrôleur réceptionnant toutes les requêtes clientes. Le lien entre les requêtes, les actions déclenchées par celles-ci et la formalisation des réponses est décrit de manière déclarative, dans un fichier de configuration XML. Les vues sont implémentées sous forme de JSP et composées à l'aide de balises spécifiques (customs tags) fournies par Struts.

Workflow



workflow

La servlet `ActionServlet` joue un rôle central dans l'architecture de **Struts**. C'est elle qui reçoit les requêtes et exploite les informations fournie dans le fichier de configuration `struts-config.xml` pour connaître le détail des éléments qu'elle va gérer dans l'application et définir comment ceux-ci vont interagir lors des traitements.

A la réception d'une requête, la servlet invoque un objet de classe `Action` qui exécute les traitements. En fonction de la réponse de l'objet Action fournie sous forme d'un objet `ActionForward`, la servlet détermine grâce à un objet `ActionMapping` la JSP qui affichera le résultat des traitements à l'utilisateur.

Des informations sont échangées entre la servlet et les objets `Action` sous forme d'objets `ActionForm`, qui sont initialisés par la servlet à partir des formulaires transmis dans les requêtes

Les informations fournies dans le fichier `struts-config.xml` sous forme d'éléments XML sont analysées par la servlet à l'aide d'un objet `Digester` importé du framework `commons-digester`.

L'analyse d'un élément XML donne lieu à l'instanciation d'un objet de la classe correspondante du package Struts config.

Des objets `ActionMessage` sont utilisés pour la gestion internationalisée des messages

---

Classes structurantes de Struts

Différences entre Struts 1 et Struts 2

Struts 2 est arrivé à la suite d'un rapprochement de Struts dans sa version 1, de WebWorks et de XWorks dans un objectif de résolution des différents problèmes de Struts1.

Pour effectuer une comparaison, nous listerons les différents problèmes relevés sur Struts1 et comment ils ont été résolus dans la version 2 :

Classes d'actions

Struts1 oblige à hérite de classes abstraites dont nos classes actions doivent hériter, ce qui créé une dépendance forte avec Struts. Dans la version 2, Struts utilise des interfaces que nous pouvons implémenter ou non. Struts2 fournit une interface « ActionSupport » permettant de profiter de la plupart des fonctionnalités nécessaires. Mais ces implémentations ne sont pas obligatoires et n'importe quelle classe POJO qui contient une méthode à exécuter respectant la signature définie dans Struts conviendra.

Gestion multitâches

Les actions dans Struts1 sont des singletons , il faut donc qu'elles soient «

thread-safe » parce qu'une seule instance de la classe gère toutes les requêtes entrantes pour cette action. La stratégie d'instance adoptée par Struts1 oblige le développeur de l'action Struts à faire attention à ce que les ressources soient thread-safe.

Struts2 n'a pas de problème de « thread-safe » puisque une instance de l'action est créée pour chaque requête. Un gestionnaire de Servlets génère tous les objets à créer pour chaque requête et les supprime aussitôt terminés. Créer plus d'objets n'affecte pas les performances et n'a pas d'impacts sur le ramasse miettes (garbage collector), c'est donc une meilleure solution.

Dépendances aux Servlets

De manière globale, dans tout projet Java, l'objectif est de réduire au minimum les dépendances du projet à des frameworks ou autre classes et API. Les actions dans Struts1 sont dépendantes de l'API Servlet puisque leurs méthodes reçoivent en paramètre les objets HttpServletRequest et HttpServletResponse. Le servlet d'applications ne considère pas les actions Struts comme une Servlet. Les contextes de Servlet sont le plus souvent représentées par de simples Map dans une action. Les actions Struts2 peuvent tout de même avoir accès aux objets de requêtes et de réponses originaux si besoin. Cette architecture permet de s'affranchir de la dépendances aux classes HttpServletRequest et HttpServletResponse.

Tests

Le plus grand défaut de Struts1 sur le test est que sa classe a besoin de l'API Servlet. Afin de pouvoir tester ses méthodes, Struts1 propose des objets bouchons (Mocks Objects). Afin de tester ses actions, Struts2 instancie l'objet, définit les attributs et invoque les méthodes. Ceci est appelé l'injection de dépendance et permet une plus grande capacité aux tests.

Gestion des saisies

Struts 1 gère les saisies via un objet « ActionForm » contenant toutes les informations saisies par l'utilisateur dans le formulaire. Comme les classes action, toutes les classes de gestion de formulaires doivent héritées la classe ActionForm. Tout autre classe de type JavaBean ne peut être utilisé en tant que ActionForm alors que les développeurs pourraient créer des classes personnalisées pour gérer les différentes saisies. DynaBeans est la meilleure alternative pour gérer les classes ActionForm conventionnelles. Dans Struts2, ce sont les attributs de la classe action qui sont utilisés pour les saisies, ils sont définis directement par Struts en utilisant les accesseurs. Ces attributs peuvent être des objets riches puisque ceux-ci peuvent avoir leur propres attributs (utilisateur.identifiant par exemple). Struts2 supporte aussi les ActionForm.

Expression Langage

Struts1 s'intègre avec JSTL donc il utilise le langage d'expression de la JSTL : JSTL EL. Ce dernier utilise une gestion des arbres d'objets très basique et une gestion faible des collections et propriétés indexées. Struts2 peut lui aussi utiliser la JSTL mais le framework supporte aussi un langage bien plus puissant et plus flexible appelé OGNL : Object Graph Notation Language.

Liaison des valeurs avec les vues

Struts1 associe les objets à la page en utilisant le mécanisme classique des JSP. Struts2, lui, utilise une technologie de ValueStack permettant de lier les vues et données et les rendre accessible via les taglibs et ce sans créer des dépendances. Cette technologie permet de réutiliser des vues en accédant à des couches de données différentes.

Conversion de types

Les ActionForm de Struts1 ne supportent quasiment que des String. Les classes de BeanUtils sont utilisées pour les conversions de types. Ces dernières sont donc faites par classe, à la main. Struts2 utilise OGNL

pour les conversions de type et les converteurs sont capables de convertir tous les types qu'ils soient primitifs ou complexes.

Validation

Struts 1 ne propose qu'une validation manuelle qui est faite via une méthode de validation dans la classe ActionForm. Les classes peuvent avoir des contextes de validation différents pour une même classe et l'enchainement de validation de sous-objets n'est pas autorisé. Struts 2 utilise la validation manuelle via la méthode de validation du framework X Work Validation. Ce framework permet l'enchainement de validations successives et celles-ci peuvent être définies sur des propriétés de classes des contextes de validation.

Contrôle du flux d'exécution

Dans Struts1, chaque module a son propre cycle de vie mais toutes les actions partagent le même cycle de vie. Dans Struts2, des cycles de vie différents sont créés par action avec des piles d'intercepteurs. Des piles personnalisées peuvent être créées et utilisées avec différentes actions, en fonction de vos besoins.

Mise en route

Installation

Pour commencer à utiliser Struts, vous devez tout d'abord avoir un environnement de développement Java EE. Pour cela, merci de vous référer au tutoriel JEE que nous avons déjà écrit pour vous. L'installation de Struts2 est des plus simple, il suffit : Télécharger les différents .jar de Struts dans le répertoire WebContent/WEB-INF/lib Editer un fichier struts.xml à la racine (src/). Vous pouvez créer ce fichier en faisant Bouton Droit sur src > Nouveau > Fichier. Pour pouvoir commencer à

Bouton Bienvenue. Nouveau client pour pouvoir commencer à utiliser l'application, vous devez donc avoir l'arborescence suivante dans Eclipse (ou tout autre I.D.E) :

L'architecture Struts

Dans cette section, nous allons vous expliquer l'architecture de Struts. Struts est connu pour son architecture robuste et est utilisé pour développer des projets de petites et grandes tailles. Struts est un projet opensource utilisé pour développer des applications web JEE en utilisant le modèle d'architecture MVC. Struts utilise l'API JEE qu'il étend pour permettre aux développeurs d'adopter une architecture MVC. Le framework Struts comporte trois composantes :

- Le gestionnaire de requêtes fourni par le développeur de l'application qui permet de lier un comportement à une URL
- Le gestionnaire de réponses qui permet de transférer le contrôle vers une autre ressource qui va s'occuper du rendu de la réponse (vers le client)
- Une librairie de « tags » permettant au développeur de créer des formulaires interactifs avec des pages web.

Struts fournit toute l'architecture pour implémenter un MVC dans vos applications et ainsi permettre aux développeurs de se concentrer sur la partie « métier ».

L'architecture MVC

Le principal objectif de l'architecture MVC est de séparer la logique « métier » de l'affichage de données (appelée « présentation ») de l'application. Voici quelques-unes des raisons pour lesquelles nous devons utiliser une architecture MVC :

- Vos couches sont réutilisables : Quand nous avons un besoin, vous n'avez pas besoin de développer une nouvelle solution, vous avez juste besoin d'adapter la solution actuelle à vos nouveaux besoins.
- Elles sont expressives : En utilisant une architecture MVC, vos couches

sont plus expressives : vous comprenez mieux ce que fait chacune des briques de votre application et où elles sont utilisées.

Le modèle

L'objet modèle a la main sur les données à afficher. C'est lui qui est mis au courant de toutes les modifications à effectuer sur vos données . Il représente uniquement les données de votre application mais ces données sont les données de l'entreprise ainsi que les règles métier qui gère les accès et les droits de modifications à ces données.

La vue

La vue représente ce qui est affiché dans l''application. L'objet de vue réfère au modèle pour connaitre les données à afficher. La vue n'est pas dépendante de l'application, afin d'être réutilisable (Par exemple elle n'appelle jamais explicitement une autre partie de l'application, elle indique un code de retour et le contrôleur se chargera d'appeler la bonne vue). La vue est donc chargée de modifier l'affichage lorsque des données sont modifiées.

Le contrôleur

Chaque fois que l'utilisateur fait une requête à l'application, celle-ci passe par le contrôleur, il est le point d'entrée unique obligatoire. Le contrôleur est responsable d'intercepter les requêtes de la vue et de les rediriger vers le modèle. Après que l'action ait pris les données appropriées, le contrôleur doit appeler la vue appropriée pour l'affichage à l'utilisateur.

Aperçu du framework Struts

Le framework Struts est composé de plus de 300 classes et interfaces qui sont organisés dans 12 packages. Le cycle de vie d'une application (d'une page) Struts est le suivant :

Les composants Contrôleurs

Chaque fois que l'utilisateur fait une requête, la requête est gérée par la Servlet de Struts appelée « Struts Action Servlet ». Quand « Action Servlet » reçoit une requête, elle intercepte l'URL et en fonction des fichiers de configuration de Struts, elle appelle une classe d'action pour qu'elle gère la requête. La classe dite « Action » est une partie du contrôleur et est responsable de communiquer avec la couche modèle.

Les composants Vue de Struts

Les composants de vue sont responsables de l'affichage d'informations aux utilisateurs ainsi que de la récupération des données qu'il a saisies. Les vues doivent afficher les informations que fournit la couche modèle. La plupart du temps, nous utilisons les Java Server Pages (JSP) pour afficher la vue. Afin d'étendre les possibilités d'une page, nous pouvons des tags personnalisés, du Java Script, etc.

Les composants Modèle de Struts

Les composants de modèle fournissent les données issues de la couche « métier ». Ils fournissent des interfaces génériques afin d'accéder à des bases de données ou des systèmes tiers. Les composants modèle sont généralement des classes Java. Il n'y a pas de format défini pour le modèle, donc il est possible d'utiliser des couches de modèle réutilisables sur différents projets.