

Modèle d'exécution : le rôle comme contrainte

Référence d'article

Article complet

<https://b-ziq.com/fr/blog/filemaker-modele-execution/>

Doctrine fondamentale

- Chaque script occupe un rôle défini dans le modèle d'exécution.
- Un rôle est une contrainte sur le comportement autorisé, et non une convention de nommage ou d'organisation.
- Un script qui effectue une action hors de son rôle est invalide, quel qu'en soit le résultat.
- Le rôle d'un script doit être visible depuis la structure seule par le nommage et l'organisation de l'espace de travail.
- L'exécution des méthodes est atteinte par le modèle de routage; les chemins d'accès directs sont contraints par le rôle.
- Le modèle d'exécution ne permet que des positions d'exécution et des chemins d'exécution définis.

Définitions

Rôle de script

Contrainte qui définit quelles opérations un script est autorisé à effectuer, ce qu'il a l'interdiction de faire, et où il est autorisé à apparaître dans le modèle d'exécution.

Script d'interface

Script attaché au comportement de modèle qui gère l'état de présentation et l'interaction d'interface sans implémenter de comportement métier.

Appelant

Script qui initie l'exécution d'une méthode en assemblant l'enveloppe de requête et en acheminant l'appel par la passerelle.

Script d'entrée de méthode

Script désigné comme cible d'exécution d'une méthode après résolution de l'adresse `module.method` par le routeur.

Script d'infrastructure

Script qui soutient la mécanique d'exécution — routage, acheminement et application de politique — sans implémenter de comportement métier.

Script de frontière de session

Script invoqué par les événements du cycle de vie de la plateforme pour gouverner le démarrage et l'arrêt hors du modèle d'exécution actif.

Responsabilités

Scripts d'interface

- Gérer l'état de présentation.
- Déclencher l'exécution par l'intermédiaire d'un appelant.
- Afficher les résultats retournés.

Appelants

- Assembler les enveloppes de requête.
- Acheminer les appels par la passerelle.
- Recevoir les enveloppes de réponse.
- Brancher selon l'issue de l'exécution.
- Effectuer le traitement post-exécution.

Scripts d'entrée de méthode

- Recevoir les enveloppes de requête.
- Exécuter la logique, la validation ou la mutation constituant la méthode.
- Retourner les enveloppes de réponse par les assistants contractuels.

Scripts d'infrastructure

- Soutenir le routage, l'acheminement et l'application de politique.
- Fournir des services transversaux d'exécution.

Scripts de frontière de session

- Établir l'état de session.
- Appliquer les politiques de démarrage.
- Gouverner l'arrêt et la fermeture.
- S'exécuter aux frontières du cycle de vie de la session.

Contraintes

Visibilité du rôle

- Le rôle d'un script doit pouvoir être inféré de son nom et de son emplacement dans l'espace de travail sans lire son corps.
- Si le rôle ne peut pas être inféré du nom et de l'emplacement seuls, le nommage ou l'organisation est invalide.

Les scripts d'interface ne doivent pas

- Implémenter des règles métier.
- Muter des données directement.
- Appeler un routeur directement.
- Appeler un script d'entrée de méthode directement.
- Contourner la passerelle pour invoquer de la logique.

Les appelants ne doivent pas

- Contenir des règles métier.
- Muter des données directement.
- Appeler un routeur directement.
- Appeler un script d'entrée de méthode directement.
- Contourner la passerelle.

Les scripts d'entrée de méthode ne doivent pas

- Dépendre d'un modèle actif, d'une sélection active, d'un état de fenêtre ou de variables globales.
- Lire une entrée non déclarée depuis l'environnement.
- Afficher des dialogues.
- Effectuer de la navigation.
- Être appelés directement par un appelant ou un script d'interface.

Les scripts d'infrastructure ne doivent pas

- Contenir de logique métier.
- Être confondus avec des méthodes.
- Être appelés directement par des appelants comme s'ils étaient des scripts d'entrée de méthode.

Les scripts de frontière de session ne doivent pas

- Implémenter des règles métier.
- Muter des données métier canoniques hors d'une méthode.
- Participer à une chaîne d'acheminement de méthode.

Organisation de l'espace de travail

- Chaque rôle d'exécution doit posséder une section dédiée de l'espace de travail.
- EDITOR doit exister comme section dédiée à la chorégraphie de session d'édition.
- Aucun rôle ne peut être absorbé silencieusement dans une autre section.

- L'organisation par fonctionnalité qui mélange plusieurs rôles dans un même dossier est invalide.

Regroupements de scripts

- Un regroupement de scripts demeure une seule méthode.
- La frontière du regroupement est interne à l'implémentation.
- Le contrat et l'identité logique de la méthode ne changent pas.
- Un membre du regroupement autre que le script d'entrée ne doit pas être accessible hors de la frontière de la méthode.
- L'état partagé à l'intérieur d'un regroupement doit être établi par la méthode, appartenir à une seule exécution, demeurer inaccessible hors du regroupement et être effacé à la fin de l'exécution.

Invariants

- Le comportement appartient aux méthodes, pas aux scripts.
- Les rôles sont définis par le comportement autorisé et la position dans le modèle d'exécution.
- L'autorité de routage est liée à l'identifiant de script.
- Un script d'entrée de méthode reçoit un contrat, pas un contexte d'exécution.
- Les scripts d'infrastructure gouvernent comment le comportement est atteint, et non le comportement lui-même.
- Les scripts de frontière de session opèrent à l'extérieur du modèle d'exécution actif.
- Deux chemins d'exécution seulement sont valides : l'invocation directe des scripts d'interface par les déclencheurs ou boutons, et l'invocation routée par la chaîne d'acheminement pour tout le reste.
- L'espace de travail constitue une surface d'application de la typologie.
- EDITOR est une section d'espace de travail et non un rôle d'exécution.
- Les regroupements de scripts ne créent pas d'unités architecturales supplémentaires.

Validation

- Déterminer le rôle d'un script à partir de son nom et de son emplacement sans ouvrir le script.
- Vérifier que le comportement d'un script demeure dans les permissions de son rôle déclaré.
- Vérifier que toute exécution de méthode entre par le modèle de routage.
- Vérifier que tout effet métier initié depuis l'interface réintègre le modèle par un appelant.

- Vérifier que les scripts de frontière de session ne sont utilisés qu’aux limites du cycle de vie.
- Vérifier que chaque rôle possède une section dédiée dans l’espace de travail.
- Vérifier que les regroupements de scripts demeurent invisibles à l’extérieur de la frontière de la méthode.
- Vérifier que les violations de rôle sont détectables depuis la structure seule.

Conséquences des violations

- Le comportement apparaît hors de son propriétaire déclaré.
- La séparation entre orchestration, logique, infrastructure et interface est perdue.
- L’exécution contourne les contrôles architecturaux.
- Des mutations surviennent hors de la propriété contractuelle des méthodes.
- Les violations de rôle deviennent invisibles depuis la structure.
- Le système ne peut plus être raisonné à partir du modèle d’exécution seul.
- L’applicabilité de la typologie est compromise.
- Un membre de regroupement accessible hors de la frontière de la méthode devient une violation structurelle.
- Un script correctement typé mais hébergé dans un fichier non autorisé demeure architecturalement invalide.