Model-View-Controller-Service-Paradigma/Service

Wechseln zu:Navigation, Suche

Dieser Artikel erfüllt die GlossarWiki-Qualitätsanforderungen **nur teilweise**:

Korrektheit: 4 (großteils überprüft)	**Umfang**: 3 (einige wichtige Fakten fehlen)	**Quellenangaben**: 4 (fast vollständig vorhanden)	**Quellenarten**: 4 (sehr gut)	**Konformität**: 4 (sehr gut)

Diese Bewertungen beziehen sich auf alle im nachfolgenden Menü genannten Artikel gleichermaßen.

MVC-Paradigma:	Model (Data) | View | Controller	MVC-Pattern:	Singletons | Dependency Injection | Observers
MVCS-Paradigma:	Service	MVCS-Pattern:	Singletons | Dependency Injection | Observers
LDVCS-Paradigma:	Logic	VCLSD-Pattern:	Singletons | Dependency Injection | Observers

Inhaltsverzeichnis

1 Definition (Kowarschick (MMProg))
- 1.1 Eigenschaften eines Servicemoduls

2 Anmerkungen
3 Quellen
4 Siehe auch

1 Definition (Kowarschick (MMProg))

Ein Service (engl. service) einer MVCS-Anwendung oder einer LDVCS-Anwendung dient zur Kommunikation mit der Außenwelt, d. h. mit Dienste-Anbietern wie Web-Servern, Datenbanksystemen oder auch Dateisystemen. Die Kommunikation kann in beide Richtungen erfolgen: Services können sowohl Daten aus Modellen bzw. Datenmodulen auslesen und in ein externes Repository schreiben, als auch Daten aus einem externen Repository lesen und in in ein Modell bzw. Datenmodul einfügen.

Service-Dienste können i. Allg. nur von MVCS-Controllern bzw. LDVCS-Logik-Modulen sowie von anderen Services angefordert werden.

Einem Service stehen in speziellen MVCS-Modellen bzw. LDVCS-Datenmodulen i. Allg. Container (wie z. B. Arrays, ein Hash Maps, Objekte oder Ähnliches) zur Verfügung, aus denen er Daten auslesen kann oder in die er Daten einfügen kann, sobald diese verfügbar sind. Eventuell stehen ihm auch noch spezielle Status-Modelle bzw. -Datenmodule zur Verfügung, in denen er Statusmeldungen über den aktuellen Stand und den Verlauf der Bearbeitung der Serviceanfrage eintragen kann.

Der MVCS-Prozess

Der LDVCS-Prozess

1.1 Eigenschaften eines Servicemoduls

Ein Service bearbeitet eine Anfrage im Allgemeinen asynchron. Sobald ein Ergebnis in einem Modell oder Daten-Modul eingetragen wurde, kann dieses Modul den Aufrufer und beliebig viele andere Module mit Hilfe eines Signals darüber informieren. Sollte die aktuelle Anfrage nicht erfolgreich bearbeitet werden können, so kann der Service eine entsprechende Fehlermeldung in ein spezielles Fehler-Modell bzw. -Datenmodul eintragen, das dann die übrigen Module über die aufgetretenen Probleme informiert. Ebenso können anderen Module über die erfolgreiche Übertragung von Daten in die Außenwelt mit Hilfe von speziellen Status-Modellen bzw. -Datenmodulen informiert werden.

Ein Service kann mehrere Anfragen hintereinander bearbeiten und dabei auch auf frühere Ergnisse (die er z. B. in privaten Zustandsvariablen abgelegt hat) zugreifen. In manchen Situation ist es vorteilhaft, wenn dem Aufrufer spezielle Services zur Verfügung stehen, die eingehende Service-Anfragen nicht parallel, sondern nacheinander abarbeiten.

2 Anmerkungen

Die Idee, Services als eigene Module zu betrachten, d. h. das MVC-Paradigma zu einem MVCS-Paradigma zu erweitern, geht auf Joe Berkovitz zurück.[1]

MVCS-Services und LDVCS-Service unterscheiden sich nicht, wenn man davon absieht, dass MVCS-Services auf Modelle zugreifen und LDVCS-Services auf Datenmodule.

Die Aufrufer unterscheiden sich allerdings: MVCS-Services werden von MVCS-Controllern aufgerufen und LDVCS-Services werden von LDVCS-Logik-Modulen aufgerufen.

In MVC-Anwendungen übernehmen i. Allg. MVC-Modelle die Kommunikation mit der Außenwelt. Es ist aber auch denkbar, dass MVC-Controller diese Aufgabe übernehmen.

3 Quellen

Berkovitz (2006): Joe Berkovitz; An architectural blueprint for Flex applications; https://web.archive.org/web/20070105004310/http://www.adobe.com:80/devnet/flex/articles/blueprint.html; 2006; Quellengüte: 2 (Web)
Kowarschick (MMProg): Wolfgang Kowarschick; Vorlesung „Multimedia-Programmierung“; Hochschule: Hochschule Augsburg; Adresse: Augsburg; Web-Link; 2018; Quellengüte: 3 (Vorlesung)

4 Siehe auch

Model (MVC)
Controller (MVC)
Service (VCLSD)

Kategorien:
MVC