



V43 1062

MÁV



Inhalt

1. Einführung
2. Warum Rails?
3. Wichtige Prinzipien
4. Model View Controller
5. Rails und AJAX
6. Die Sicherheit
7. Test Driven Development

Einführung

- erstmals im Juli 2004 vorgestellt
- OpenSource Web Framework auf Basis von Ruby
- Version 1.0 am 13. Dezember 2005
- Version 2.3 am 15. März 2009
- geplant: Verschmelzung mit merb
→ Version 3

Warum Rails?

- einfache und schnelle Entwicklung
- MVC
- Generatoren, insbesondere Scaffolding
- Unterstützung verschiedener DBMS
- REST
- interaktive Konsole

Wichtige Prinzipien

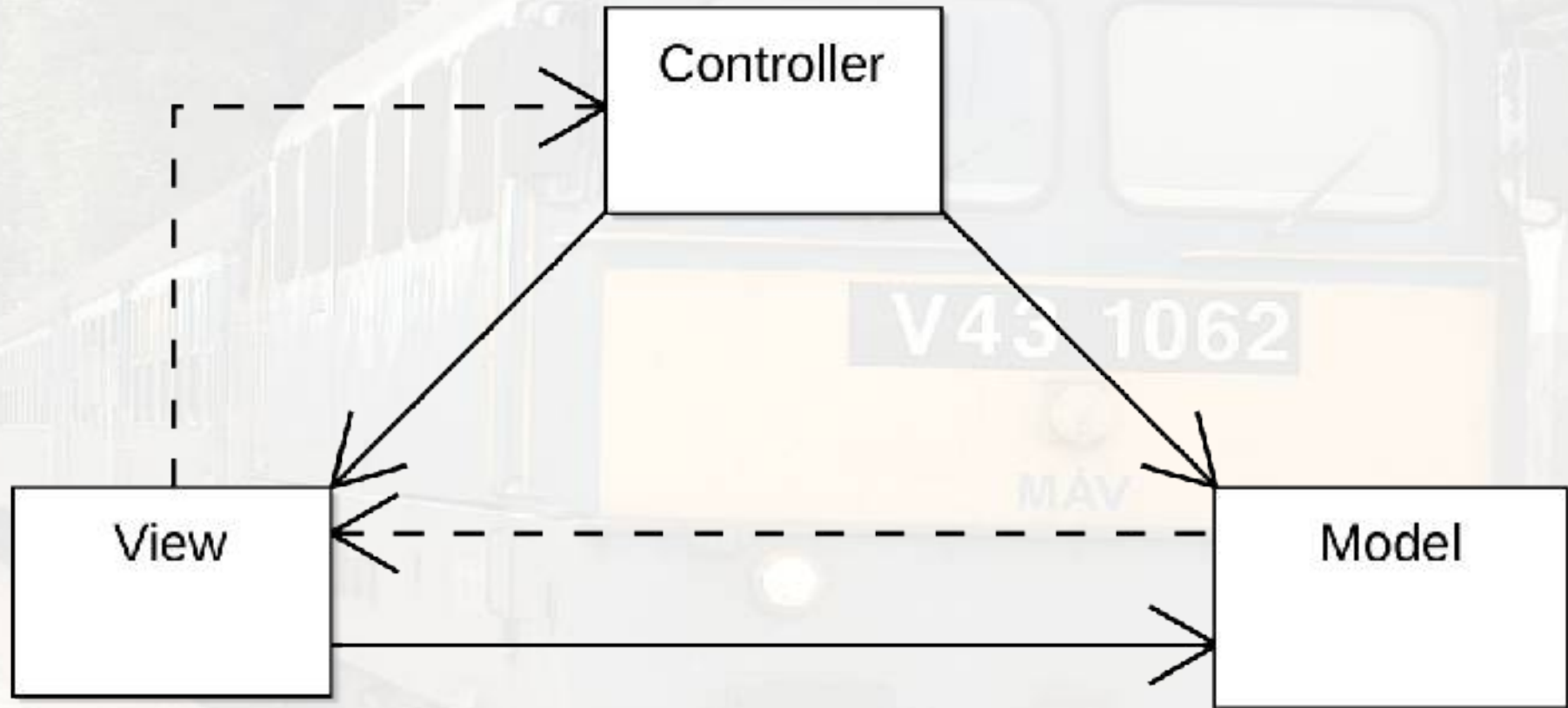
- "Don't repeat yourself"

Jede Information und Logik sollte nur einmal vorhanden sein.

- Convention over Configuration

Klare Vorgaben der Struktur ersparen Programmier- und Konfigurationsarbeiten.

Model View Controller



Model

Datenschicht

- durch ORM und Prepared Statements:
 - Schutz vor SQL-Injection
- Relationales Datenbankmodel → Objektstruktur
 - einfache Arbeit mit den Daten
 - durch durchgehende Objektorientierung Ruby's einfach verarbeitbar
- einfache Validierung von Daten

Controller

Steuerschicht

- Zugriff auf das Model
- Verarbeitung der Nutzeraktionen
- enthält die Logik
- verschiedene Ausgabeformate (Views) mit einem Controller

View

Ausgabeschicht

- Nutzung von ERB
- Layouts
- Einsatz von Helfern
- Partialas

Rails und AJAX

- feste Integration von Prototype und `script.aculo.us`
 - Prototype entstand aus dem Rails-Umfeld
- Einfache Umsetzung von AJAX-Anwendungen durch Helper
 - `link_to_remote`
 - `submit_to_remote`
- View-Funktionen: Bau von JavaScript-Code durch Ruby on Rails

Die Sicherheit

- SQL Injections
 - Schutz durch ORM und Prepared Statements
- Cross-Site Request Forgery (XSRF/CSRF)
 - seit Rails 2.0 integrierter Schutz durch Token
- Cross-Site-Scripting (XSS)
 - Helper zum einfachen entschärfen gefährdeter Ausgabevariablen

Test Driven Development

Aufgabe eines Tests

Prüfen der Ergebnisse einer Funktion bzw. eines Ablaufes

Vorgehen

- Schreiben eines Tests
- Implementierung bis der Test erfolgreich verläuft
- Entwurf des nächsten Tests

Test Driven Development

Test-Arten

- **Unit**
Prüfen eines Model
- **Functional**
Prüfen eines Controllers und des Views
- **Integration**
Prüfen eines Ablaufes
Beispiel: **Abruf der Bestellungen eines Webshop**
Zugriffsversuch → Login erforderlich → Login
→ Anzeige der Bestellungen

Demonstration von Rails

- Hier stelle ich kurz die Arbeit mit Rails anhand einiger kurzer Beispiele vor
 - erstellen einer Rails Applikation
 - kurzer Blick auf die Ordnerstruktur
 - erstellen eines Scaffold

Das Ende

**Vielen Dank fürs Zuhören
Noch Fragen?**

Quellen:

http://de.wikipedia.org/wiki/Ruby_on_Rails

Galileo Computing: Ruby on Rails

Ausführliche Quellenliste:

<http://www.ingmars-bastelecke.net/de/web/ruby-on-rails.html>