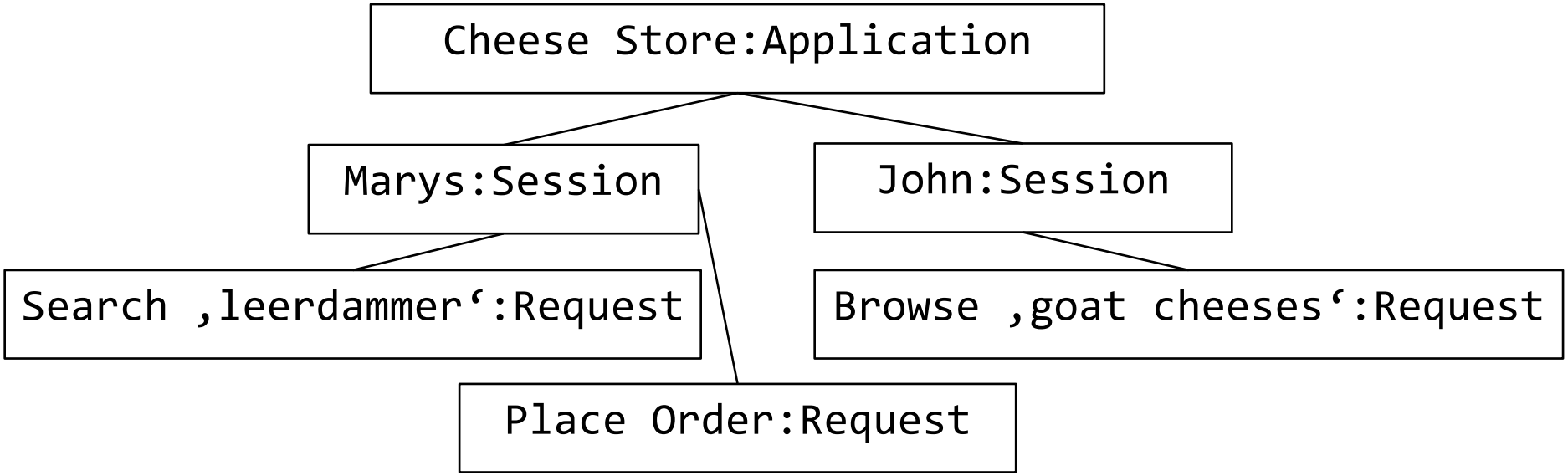
Agenda

- **Tag 1 – Einführung**
  - Installation
  - Erste Anwendung
  - Architektur
- **Tag 2 – Entwicklung**
  - Models
  - Darstellung
  - Formulare
  - Ajax
- **Tag 3 – Fortgeschrittene Themen**
  - Tests
  - Security & Deployment
  - Lokalisierung & Internationalisierung
  - Performance
  - Best Practices

Nach M.Dashorst, E.Hillenius, *Wicket in Action*, Manning 2008

DIE WICKET-ARCHITEKTUR – WIE WIRD EIN REQUEST VERARBEITET?

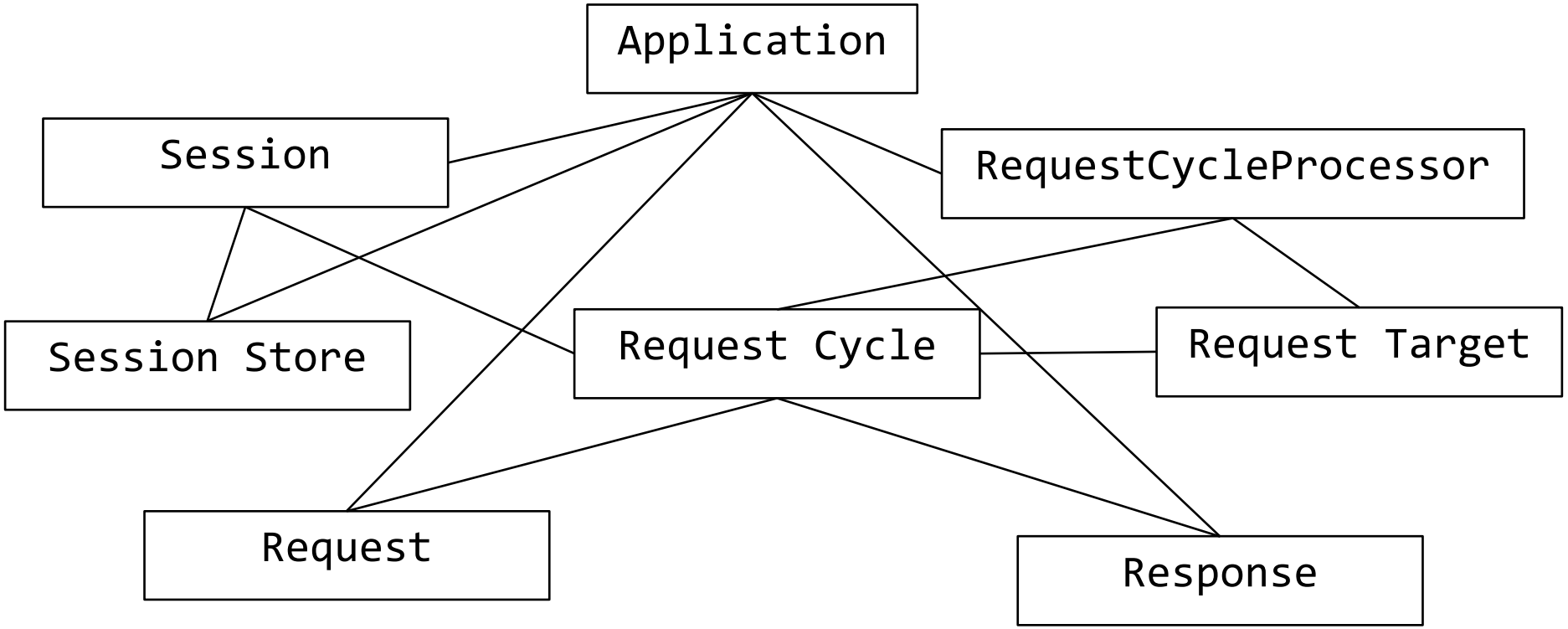
Request Handling: Vogelperspektive



- `Application`: Oberster Container – Singleton
- `Session`: „State of user“
- `Request`: Kapselt den HTTP-Request des Users
- `Response`: „Schreiboperationen“ – Output des Requests

Nach M.Dashorst, E.Hillenius, *Wicket in Action*, Manning 2008

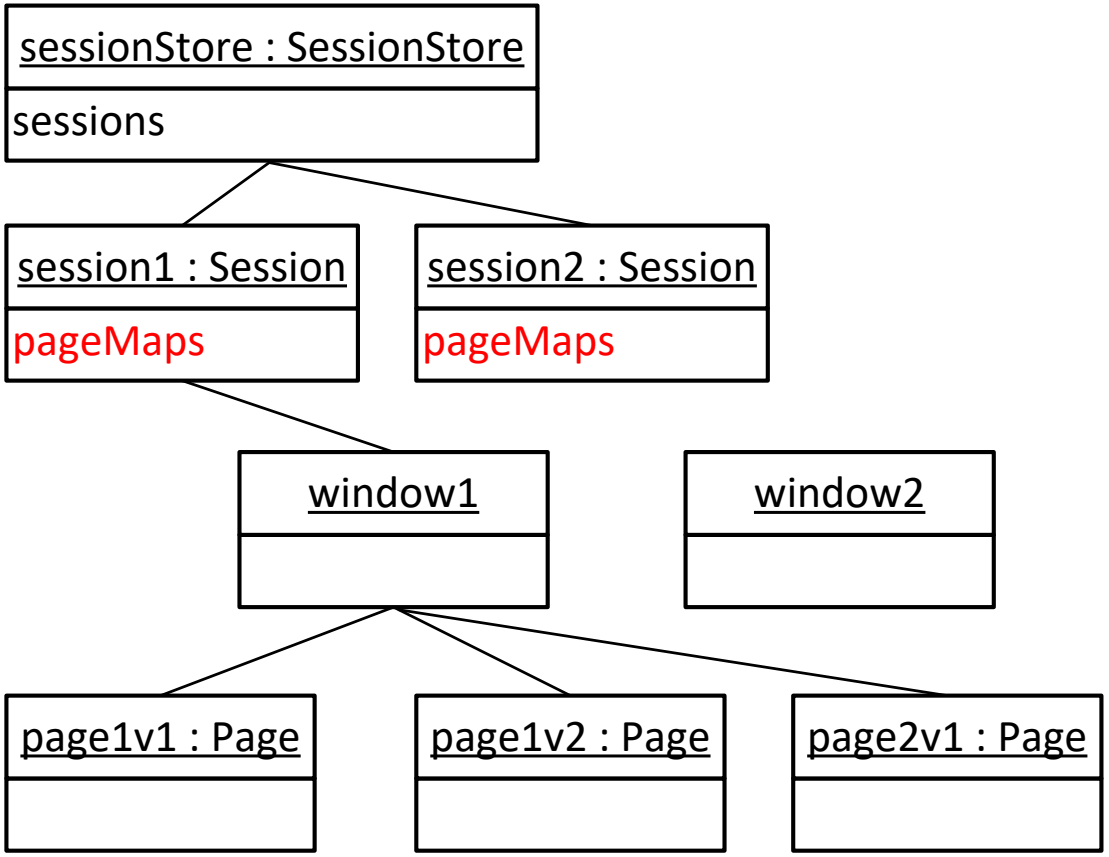
Klassen für das Request-Handling



- `RequestCycle`: Wickelt den Request ab - Delegiert
- `RequestCycleProcessor`: Schritte / Events im Request
- `RequestTarget`: Aufrufbare Seite, AjaxTarget, etc.

Nach M.Dashorst, E.Hillenius, *Wicket in Action*, Manning 2008

Request-Handling: Sessions



Nach M.Dashorst, E.Hillenius, *Wicket in Action*, Manning 2008

Architektur: Thread-Safety

- Idee: Single-Threaded soweit möglich
- Components: synchronized auf Pages
- Achtung:
  - Instanz einer Komponente nur auf einer Page
  - **Nicht Thread-Safe:**
    - Application
    - Session
    - SessionStore

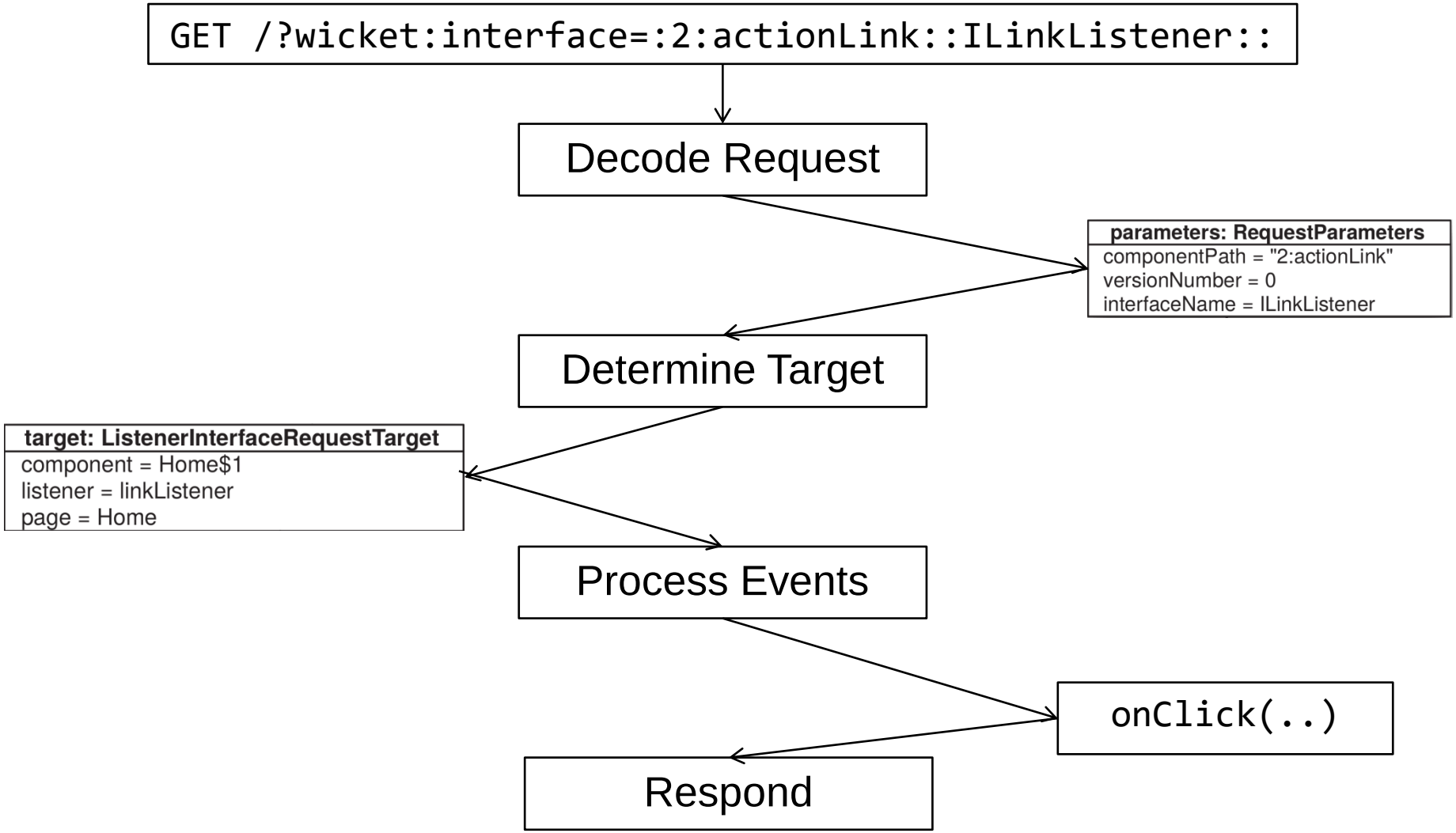
Nach M.Dashorst, E.Hillenius, *Wicket in Action*, Manning 2004

Request-Handling: Eigenschaften

Request Handling

- Kontrolliert durch Framework
- Abstraktion der Servlet API
  - ServletWebRequest
  - ServletWebResponse
- RequestCycle
  - Statischer Zugriff: RequestCycle.get()
  - Verarbeitung von Request & Response
  - Bestimmung konkreter URLs für Pages
  - Hook-Methoden und Listener
  - Ein Thread pro Cycle

Request-Handling: Ablauf



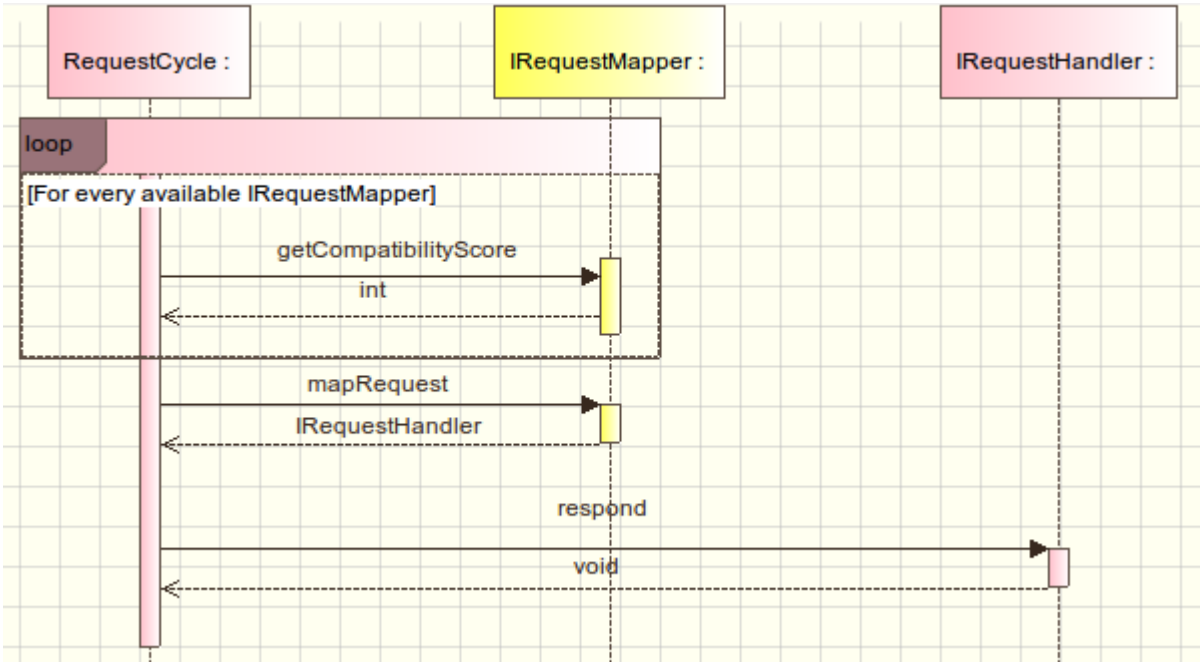
Quelle: M.Dashorst, E.Hillenius, *Wicket in Action*, Manning 2008

Request-Handling: Ablauf

anderScore trust in competence

GFU Cyrus AG

Request Handling



Quelle: Wicket User Guide – The Reference Documentation

Architektur: Sessions

Sessions

- Statischer Zugriff: Session.get()
- Aktueller Zustand des Komponentenbaums
- Eigene Implementierungen üblich
  - Beispiel: Durchführung des Logins
  - Registrierung in Application-Klasse
- Einklinken von Listenern
- Stateless Page: temporäre Session

Architektur: Sessions

anderScore trust in competence

GFU Cyrus AG

```
public class BasicAuthenticationSession extends AuthenticatedWebSession {

    private String username;

    public BasicAuthenticationSession(Request request) {super(request);}

    @Override
    public boolean authenticate(String username, String password) {
        boolean success = false;

        // Dies ist nur ein Beispiel!
        if (username.equals("Wicket") && password.equals("Training")){
            this.username = username;
            success = true;
        }

        return success;
    }

    @Override
    public Roles getRoles() {return null;}

    public String getUsername() {return username;}
}
```

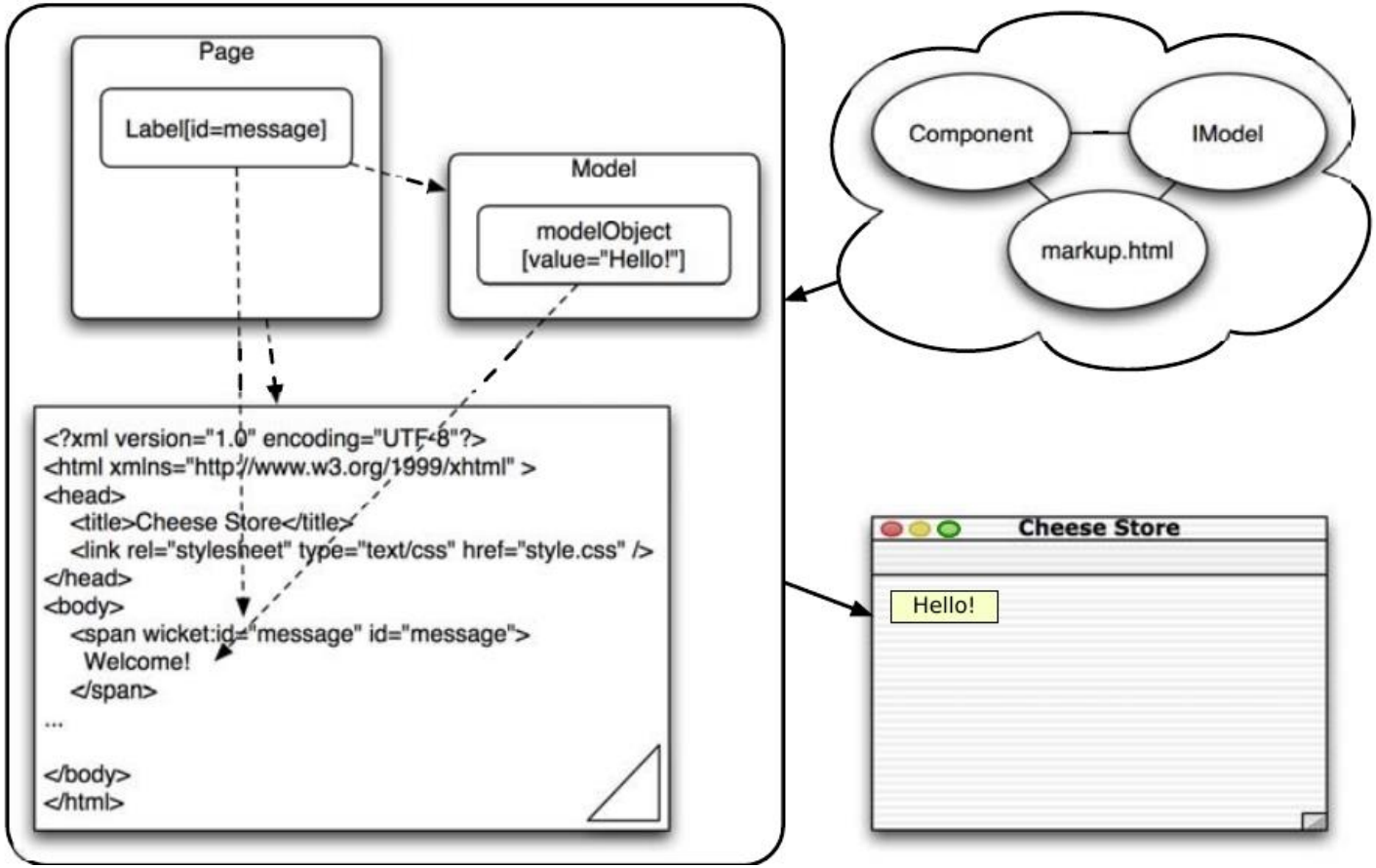
Quelle: Wicket User Guide – The Reference Documentation

Komponenten - Eigenschaften

Komponenten

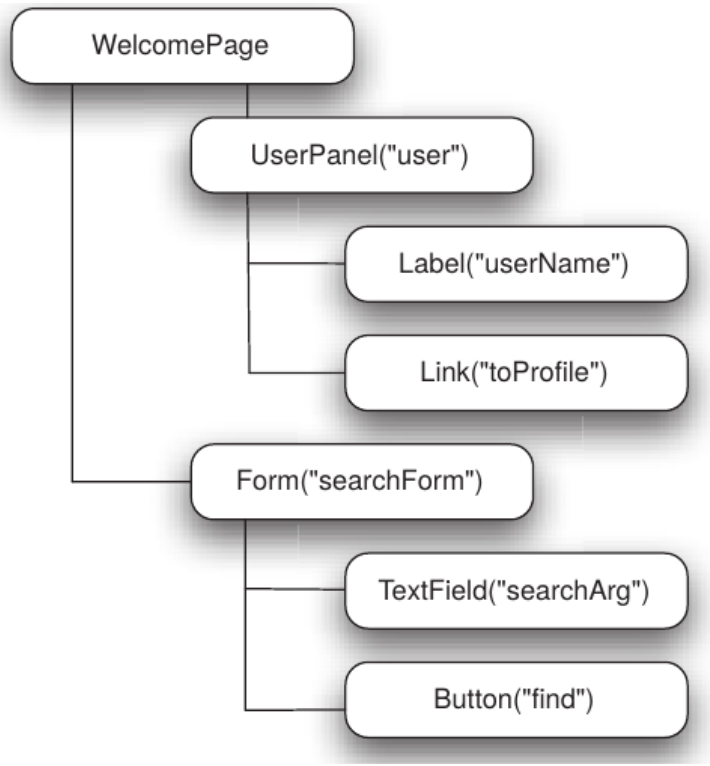
- Wiederverwendbare UI-Bausteine
- Self-contained
- HTML + Java (ggf. JavaScript…) – Strukturen sind ähnlich
- Stateful vs. stateless
- Vorgefertigte Komponenten
  - Labels
  - Forms
  - Links
  - Buttons

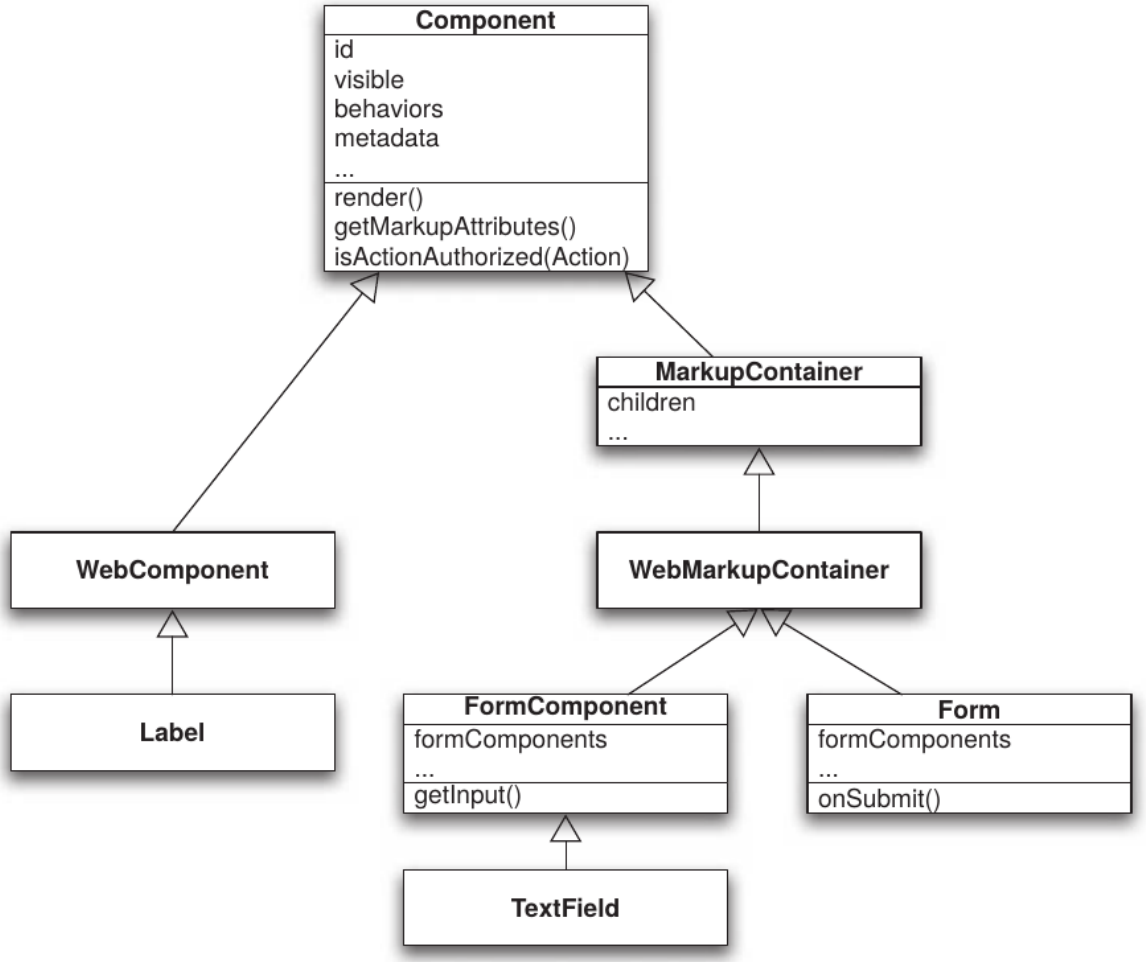
Komponenten - MVC



Quelle: M.Dashorst, E.Hillenius, *Wicket in Action*, Manning 2008

Komponentenbaum & - struktur





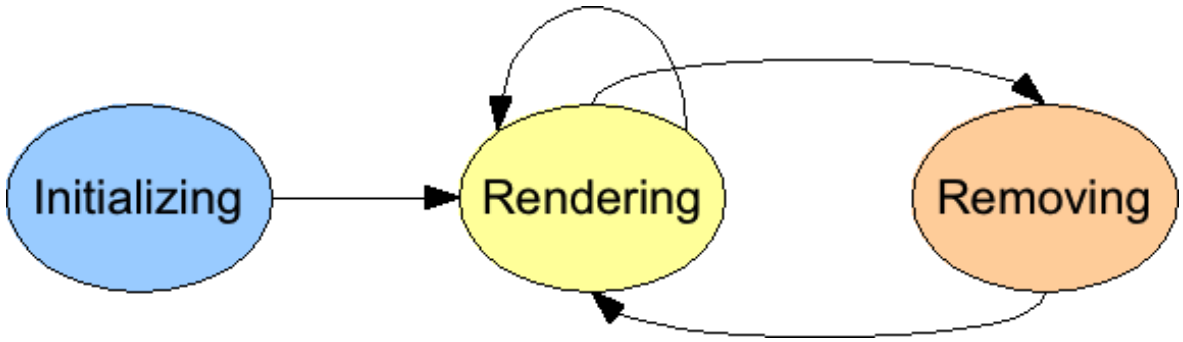
Quelle: M.Dashorst, E.Hillenius, *Wicket in Action*, Manning 2008

Verwendung der Komponenten

anderScore trust in competence
GFU Cyrus AG

Komponenten

- Lebenszyklus
  - Initializing — Instanziierung durch Wicket
  - Rendering — Generierung von Markup
  - Removing — Entfernung aus Komponentenbaum
- Hook-Methoden
  - onInitialize()
  - onConfigure(), onRender(), …
  - onRemove()



Quelle: Wicket User Guide – The Reference Documentation

Pages

- Navigation über Link-Komponenten
  - Definition eines onClick-Handlers in Java
- Deeplinks: Mountpoints in der Application-Klasse definieren:
  - `mountPage("/pageMount/${foo}", ProductInfo.class);`
- Wie viele Pages braucht eine Anwendung?
  - Wie viele abstrakte Pages?
  - Events

Page Stateful vs. Stateless

anderScore trust in competence

GFU Cyrus AG

Stateful Pages

- Verwendung über mehrere Requests hinweg
- Zustand in der Session (Java Serialization)
- Versionierung (→ Back-Button)

Quelle: Wicket User Guide – The Reference Documentation

Architektur

Stateless Pages

- Einsparung von Ressourcen
- Beispiel: Login Page
- Voraussetzungen
  - Instanziierung durch das Framework
  - Alle enthaltenen Komponenten ebenfalls stateless

Übungsaufgabe

- Modifizieren Sie die Echo-Anwendung
- Erstellen Sie eine eigene Submit-Button Komponente (abgeleitet von *Button*)
- Geben Sie im Server Antwort Bereich aus, in welcher Sequenz folgende Methoden durchlaufen werden:
  - `onInitialize()`
  - `onConfigure()`
  - `onRender()`
  - `onRemove()`