

OOD

Object Oriented Design

Markus Gehrig

# Inhaltverzeichnis

[Inhaltverzeichnis 1](#_Toc9752976)

[1 Tabelle 2](#_Toc9752977)

[2 Grafik 2](#_Toc9752978)

[3 Überschrift 2](#_Toc9752979)

[3.1 Überschrift 2](#_Toc9752980)

[3.1.1 Überschrift 2](#_Toc9752981)

# MVC – Pattern

Für die Umsetzung der Anwendung wird das MVC Pattern benutzt, es ist sehr weit verbreitet und bei Webanwendungen das meist benutzte Pattern.

## Dispatcher

Der Dispatcher nimmt alle Anfragen entgegen und leitet diese an die richtigen Stellen weiter. Er ist also vergleichbar mit dem Telefonisten aus vergangenen Tagen.

Die für seine Aufgabe nötigen Informationen bekommt durch GET-Parameter. Ein normaler Aufbau der URL sieht wie folgt aus:

index.php?**module**=dashboard&**controller**=dashboard&**action**=show&**data**=my-cooldata

|  |  |
| --- | --- |
| **module** | **Modul welches aufgerufen werden soll.** |
| **controller** | **Controller des Moduls welches aufgerufen wird.** |
| **action** | **Aufzurufende Action des Controllers** |
| **data** | **Daten welchen an den Controller weitergegeben werden sollen.** |

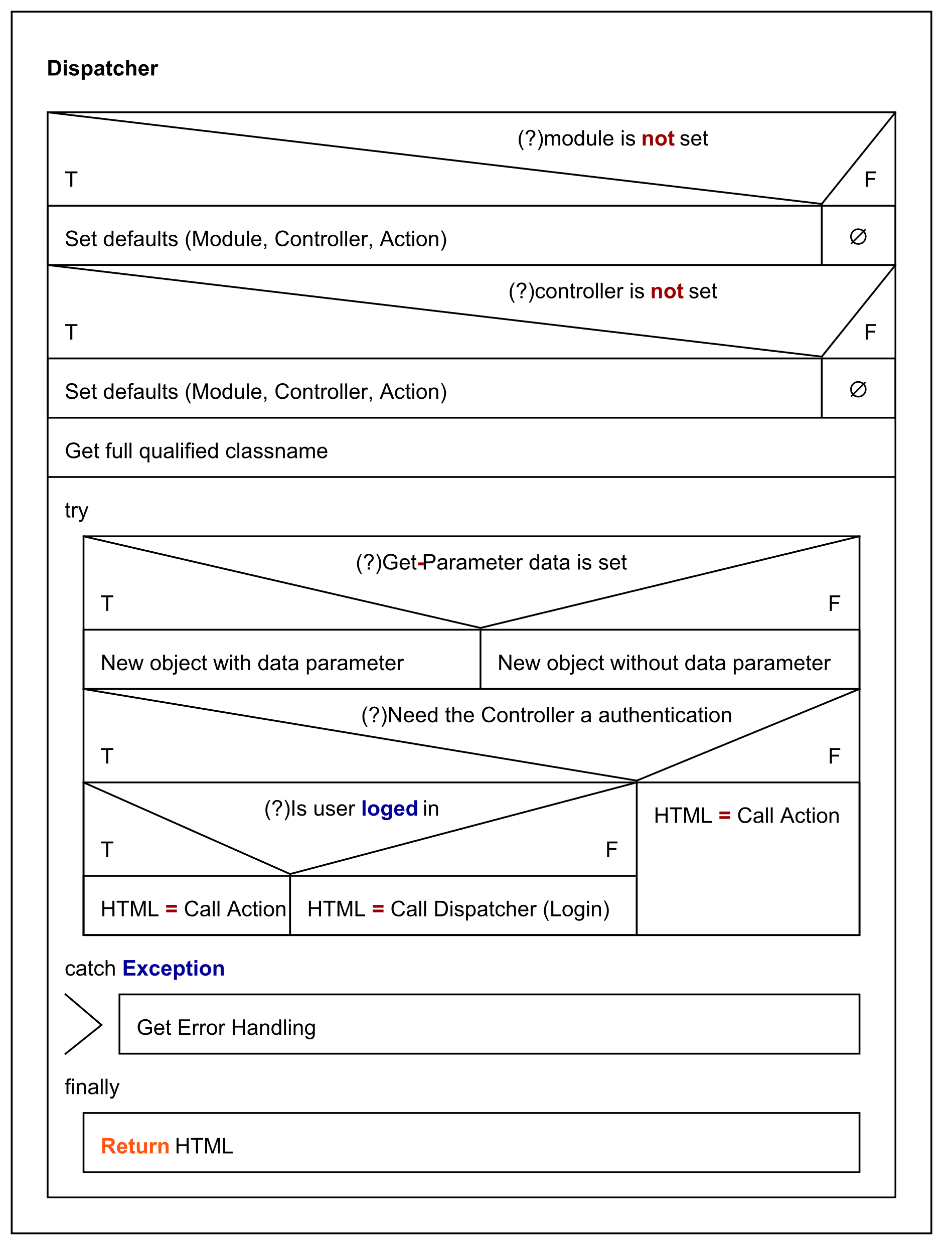
## Globale Variablen

|  |  |  |
| --- | --- | --- |
| module | private | string |
| controller | private | string |
| action | private | string |
| data | private | undefined |

## Methoden Dispatcher

|  |  |  |  |
| --- | --- | --- | --- |
| setModule | public | module:string |  |
| getModule | public |  | string |
| setController | public | controller:string |  |
| getController | public |  | string |
| setAction | public | action:string |  |
| getAction | public |  | string |
| setData | public | data |  |
| getData | public |  | data |
| dispatch | public | module:string controller:string action:string data | string |
| redirect | public | module:string controller:string action:string data |  |
| getClassName | private | module:string controller:string action:string | string |

### Nassi Schneidermann Dispatcher



# Ordnerstruktur

Die Ordnerstuktur ist sehr einfach aufgebaut. Es unterscheidet zwischen Core und Modulen, welche die einzelnen Ansichten darstellen.

* root
  + core
    - view
    - dispatcher
    - database
  + module
    - dashboard
      * controller
    - …
  + utilities
    - pdf
    - …

|  |  |
| --- | --- |
| **core** | **Globale Klassen, welche häufig gebraucht werden.** |
| view | Klassen welche für das Frontend-Rendering verantwortlich sind. |
| dispatcher | Klassen welche für das Routing der Anfragen verantwortlich sind. |
| **module** | **In sich geschlossene Module (FE-Ansichten)** |
| **utitlities** | **Klassen welche spezielle Funktionen zur Verfügung stellen** |