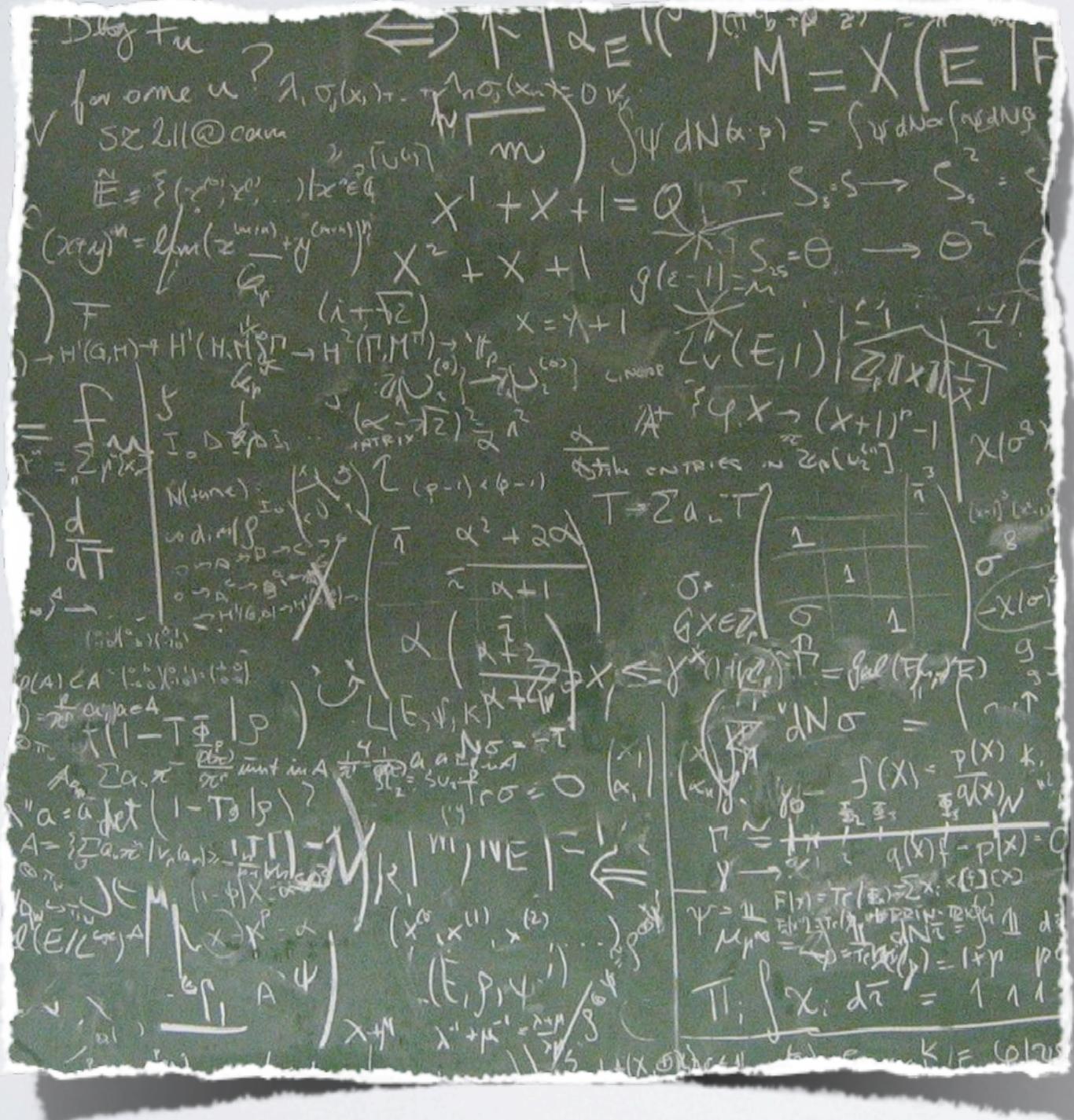


THE M PROJECT

Robin Vinzenz // rv012

Matthias Nagel // mn023

Agenda



- Was ist The-M-Project
- Entstehung
- Warum The-M-Project
- Architektur
- Ausblick

Was ist THE M PROJECT

- kostenloses OpenSource HTML5-JavaScript Framework
- MIT-Lizenz
- Entwicklung von webbasierten, plattformunabhängigen mobilen Anwendungen
- Für alle Plattformen, die HTML5, JavaScript & CSS unterstützen

<http://the-m-project.org/>

iOS



„WRITE ONCE, RUN ANYWHERE“

 BlackBerry

 Windows phone

Entstehung

- Entwickler Sebastian Werler und Dominik Laubach
- Masterthesen bei M-Way Solutions GmbH
- Ursprünglich Anpassung von SproutCore-Framework
- Gründung Panacoda, mit Ausgliederung von The-M-Project
- Weiterentwicklung durch Community



[The-M-Project on Google Groups](#)



<https://github.com/mwaylabs/The-M-Project>



https://twitter.com/@_themproject



<http://www.facebook.com/pages/The-M-Project/155037587882581>



<http://webchat.freenode.net/#themproject>

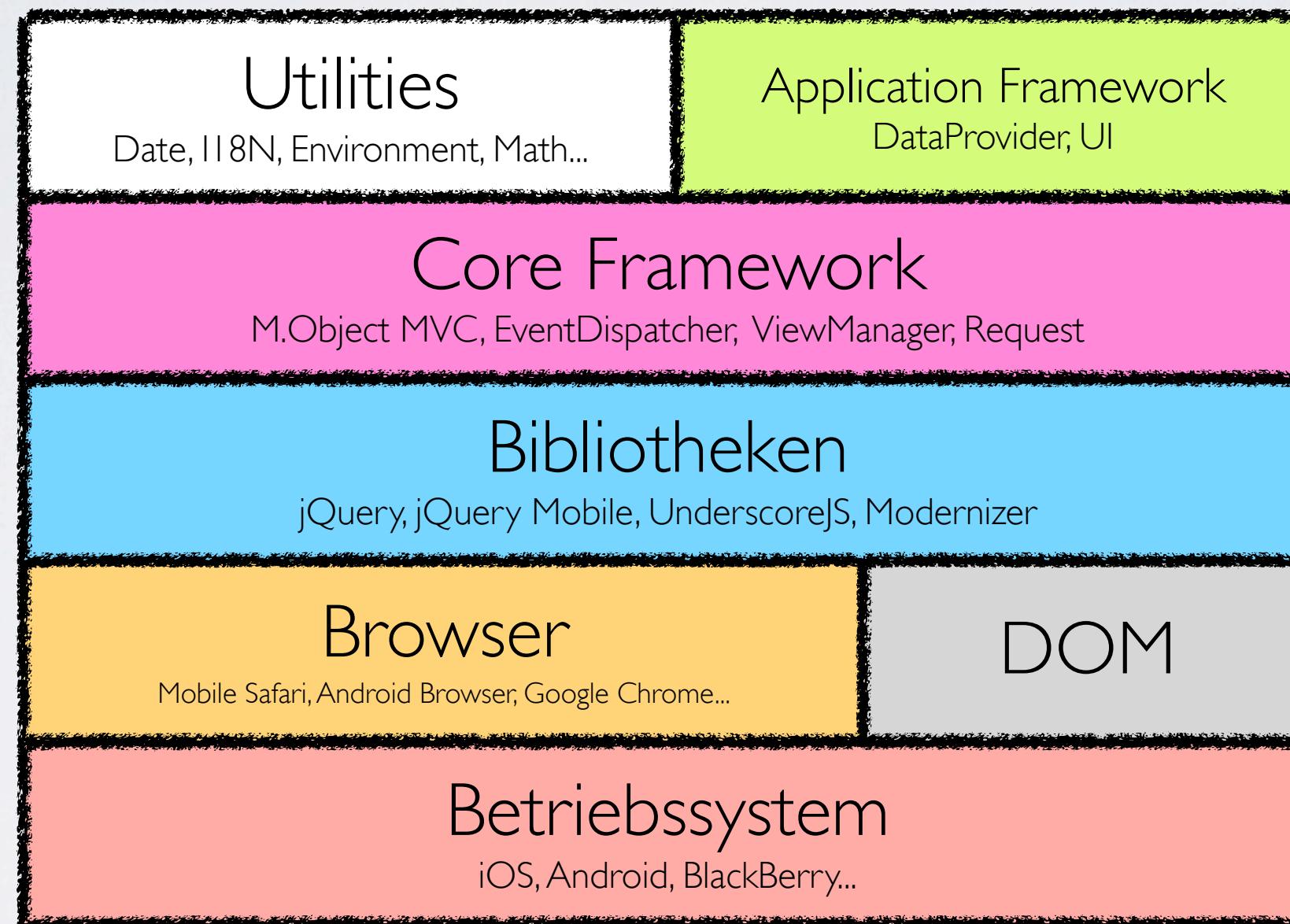
Warum THE M PROJECT

- Strukturierte Entwicklung mit MVC
- Vorgefertigte UI Komponenten
- Automatisierte Aufgaben
- Integrierter Test Server und Buildwerkzeug
- Funktioniert auch mit Nicht-Webkit Browsern

Architektur



Framework als Schichtenmodell



Was ist **ESPRESSO**

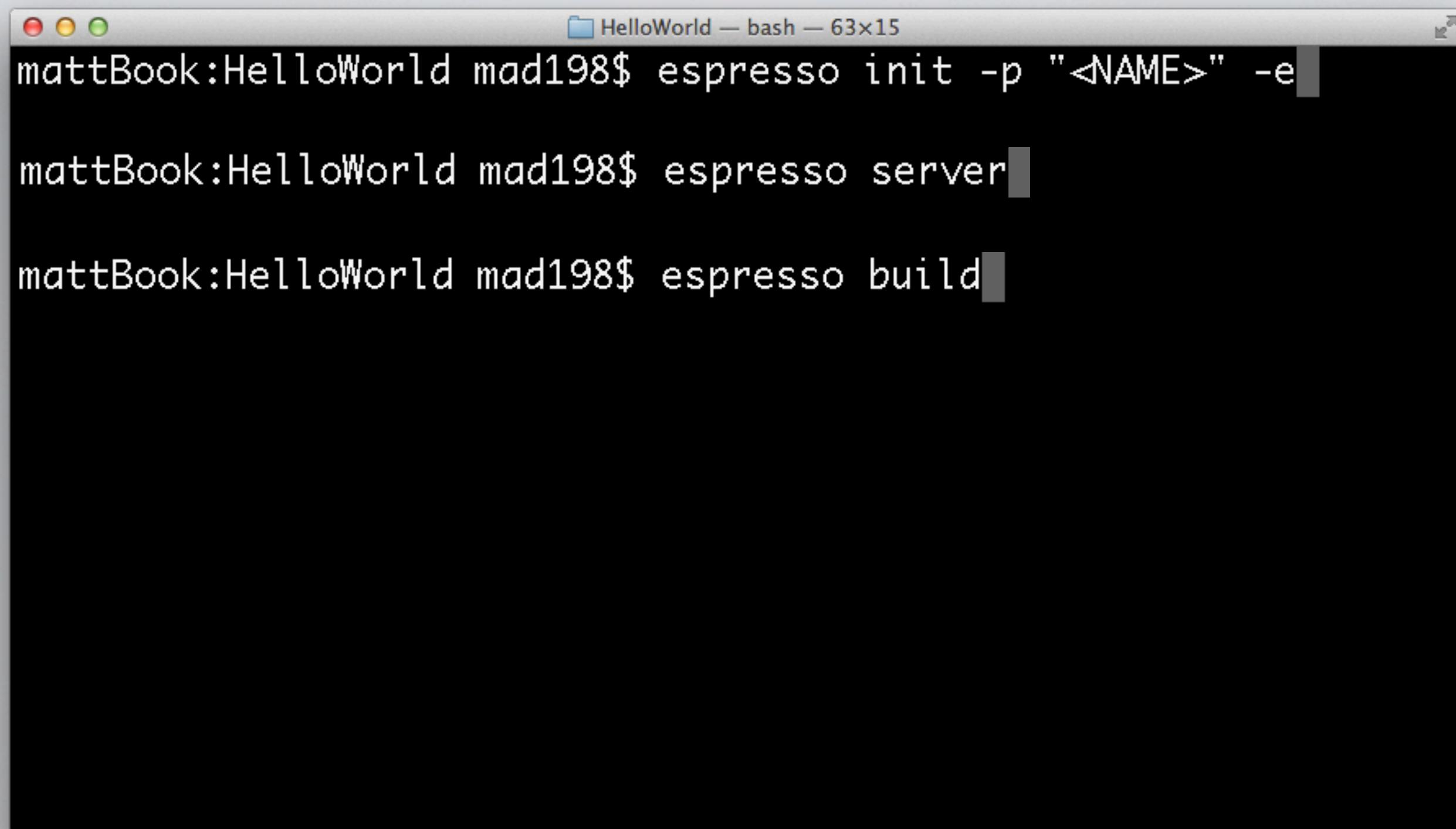
- Build-Prozess-Tool mit integriertem Webserver
- Node-basiertes Build Werkzeug
- Pack & Eliminate & Minify
- Deploy-Prozess
- Generator für The-M-Project
- Natives Packaging
- wird nicht für die Ausführung benötigt

<http://the-m-project.org/friends/espresso/>

Hands-on



ESPRESSO - Befehle



A screenshot of a macOS terminal window titled "HelloWorld — bash — 63x15". The window contains the following text:

```
mattBook:HelloWorld mad198$ espresso init -p "<NAME>" -e  
mattBook:HelloWorld mad198$ espresso server  
mattBook:HelloWorld mad198$ espresso build
```

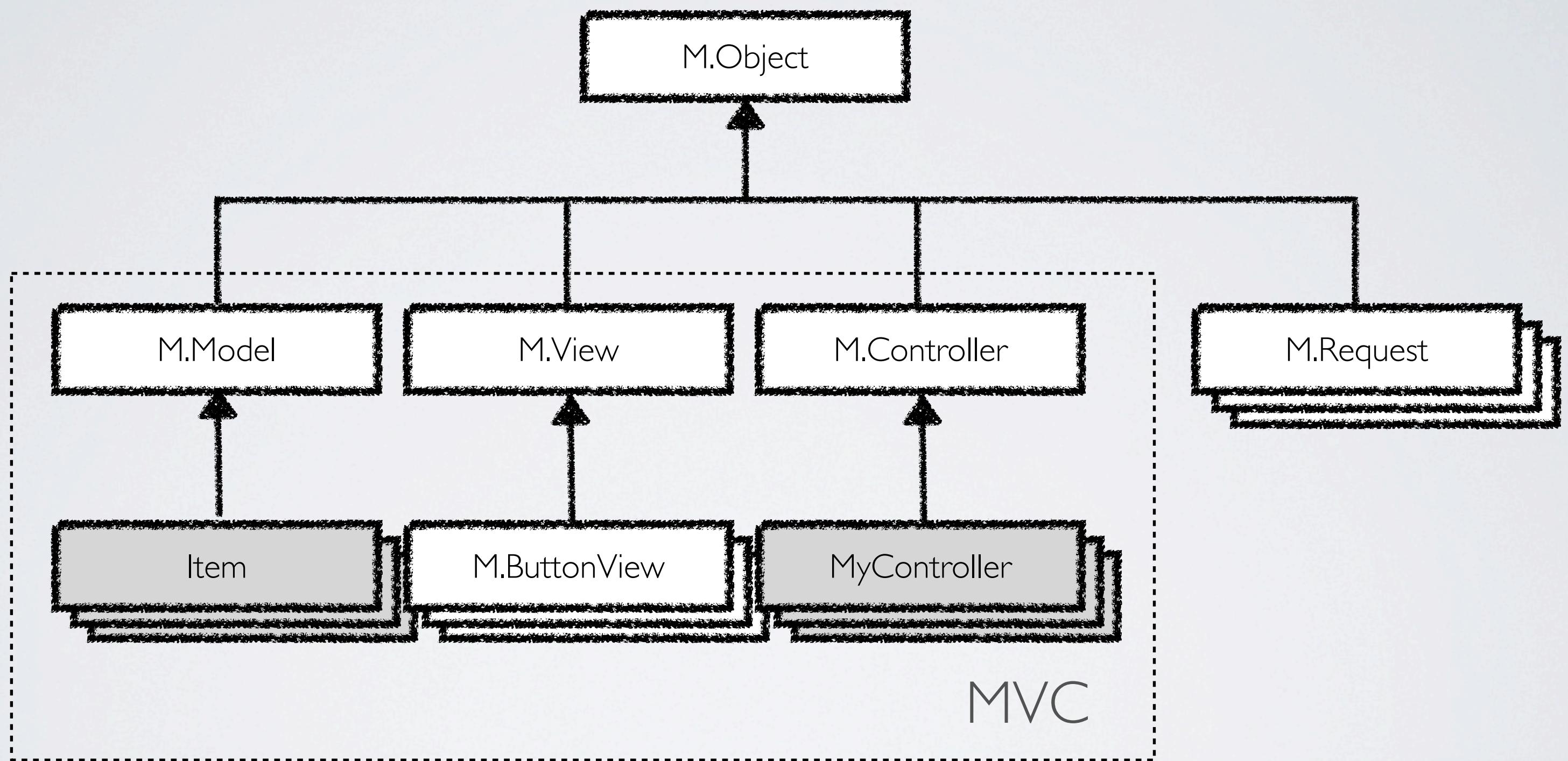
Core Concepts



- MVC
- Content Binding
- Dynamic Value Computing
- Event Handling

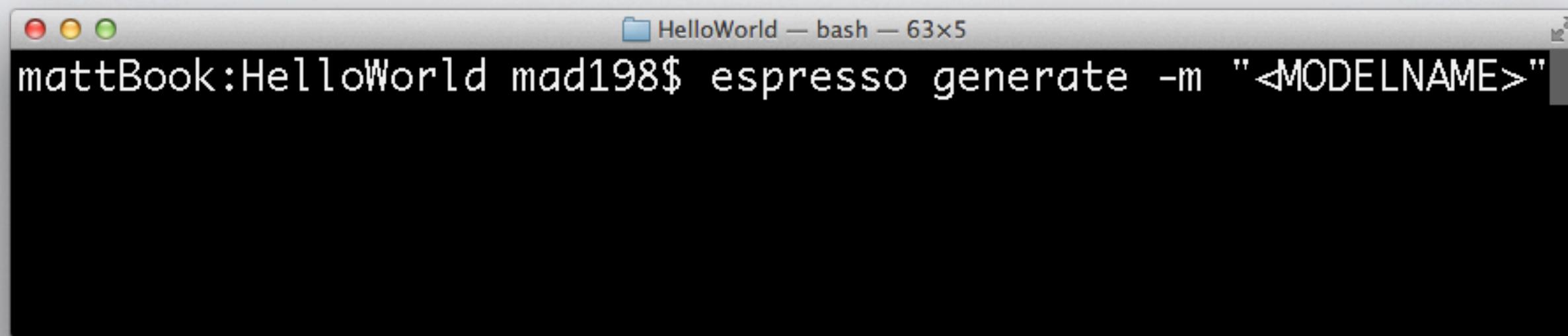
<http://panacodalabs.github.com/The-M-Docs/#home>

Objekt-Vererbung



Model

- Datenhaltung - Kapselung der Businessdaten- und logik
- Schnittstelle zur Persistenz
- zentrale Funktionen: **find()**
save()
del()



A screenshot of a Mac OS X terminal window titled "HelloWorld — bash — 63x5". The window shows a command being typed in: "mattBook:HelloWorld mad198\$ espresso generate -m <MODELNAME>". The background of the slide is a light gray color.

View

- Kapselung der UI in JavaScript-Objekten

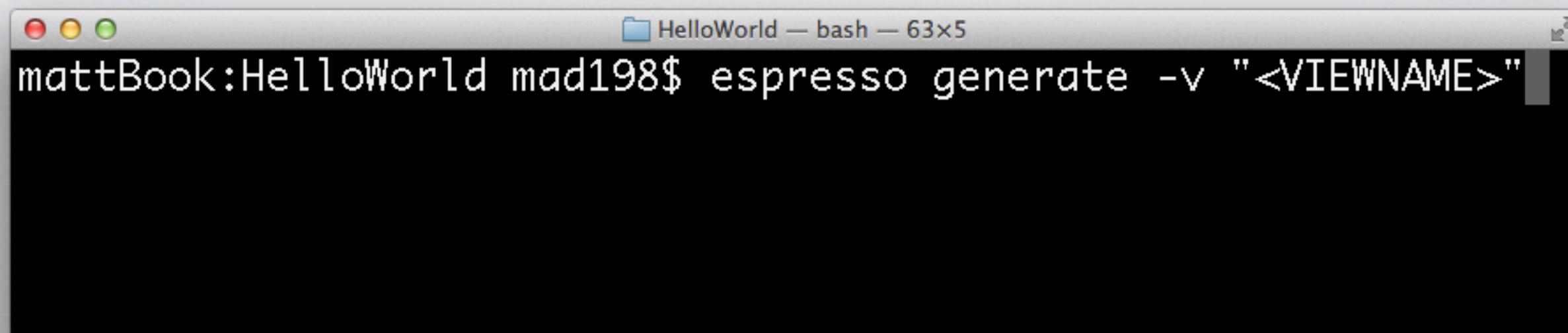
M. ButtonView

M. LabelView

M. ToolbarView

M. TextfieldView

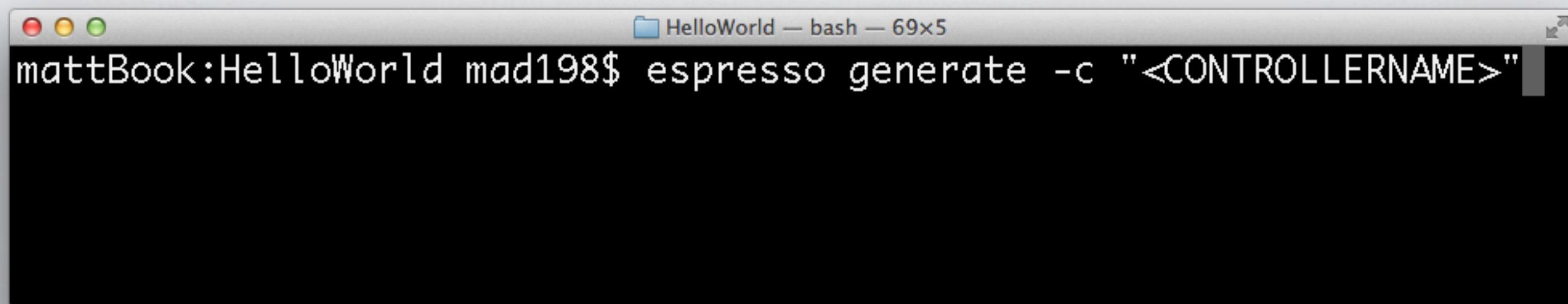
M. TabbarView



```
mattBook:HelloWorld mad198$ espresso generate -v "<VIEWNAME>"
```

Controller

- Die Kommandozentrale
- Steuerungseinheit: Bindeglied zwischen Model und View
- zentrale Rolle für das Content Binding
- Objekt der Wahl für die Programmlogik
- zentrale Funktionen: `switchToPage()` // Wechsel zu anderem View
`set()` // `notify()` in Observer-Pattern



```
mattBook:HelloWorld mad198$ espresso generate -c "<CONTROLLERNAME>"
```

Data Provider

- Abstraktion verschiedener Speichertechnologien
- Mapping Model <> Speicher
- Objektorientierte Schnittstelle
- Lesender Data Provider: **M.DataConsumer**

M. ViewManager

- verwaltet die Views einer Anwendung
- vergibt eindeutige ids
- bietet Methoden an, um Views zu finden

`getCurrentView()`

`getViewById(id)`

`getView(page, viewname)`

`getPage()`

M. Request

- Kapselung von jQuery AJAX Request
- Proxy Espresso für Requests an andere Domains (umgeht same origin policy)
- Parameter:
 - `url`
 - `method` // eg. POST, default GET
 - `isAsync`
 - `isJSON`
 - `timeout`
 - `beforeSend`, `onError`, `onSuccess`

Content Binding

- Observer Pattern: Bindeglied zwischen Controller und View
- Grundlage dynamischer Programme
- View reagiert auf Datenänderung
- Reverse Content Binding => Controller reagiert auf Viewänderung

Dynamic Value Computing

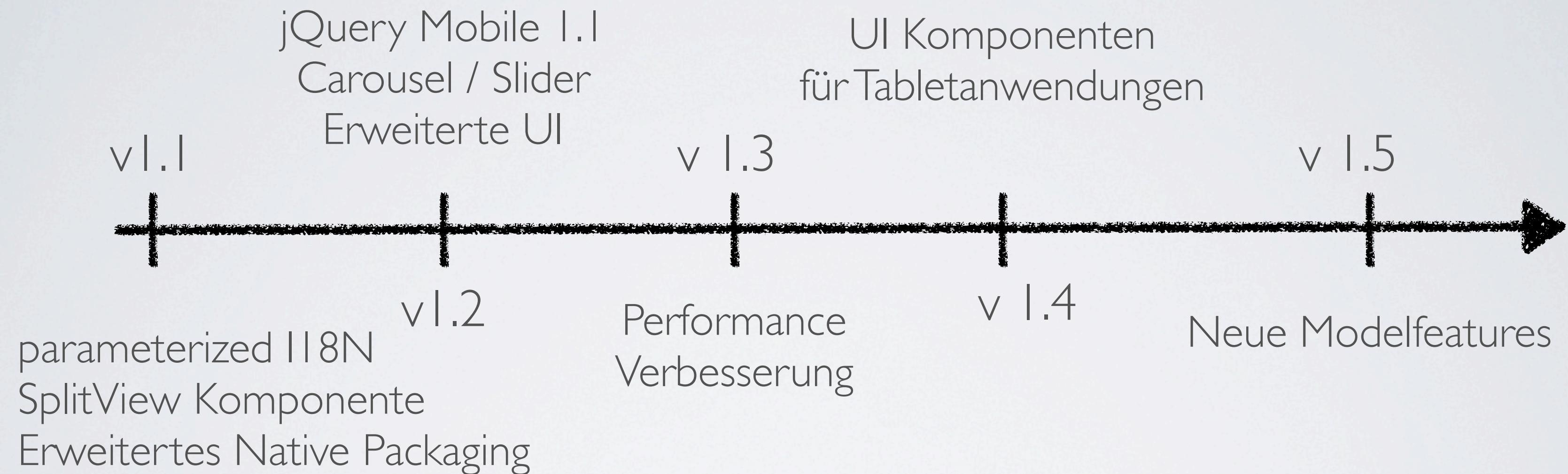
- Special Feature, um Content Binding zu bereichern
- Benutzerdefinierte Wertbearbeitung vor Darstellung

computedValue:

Event Handling

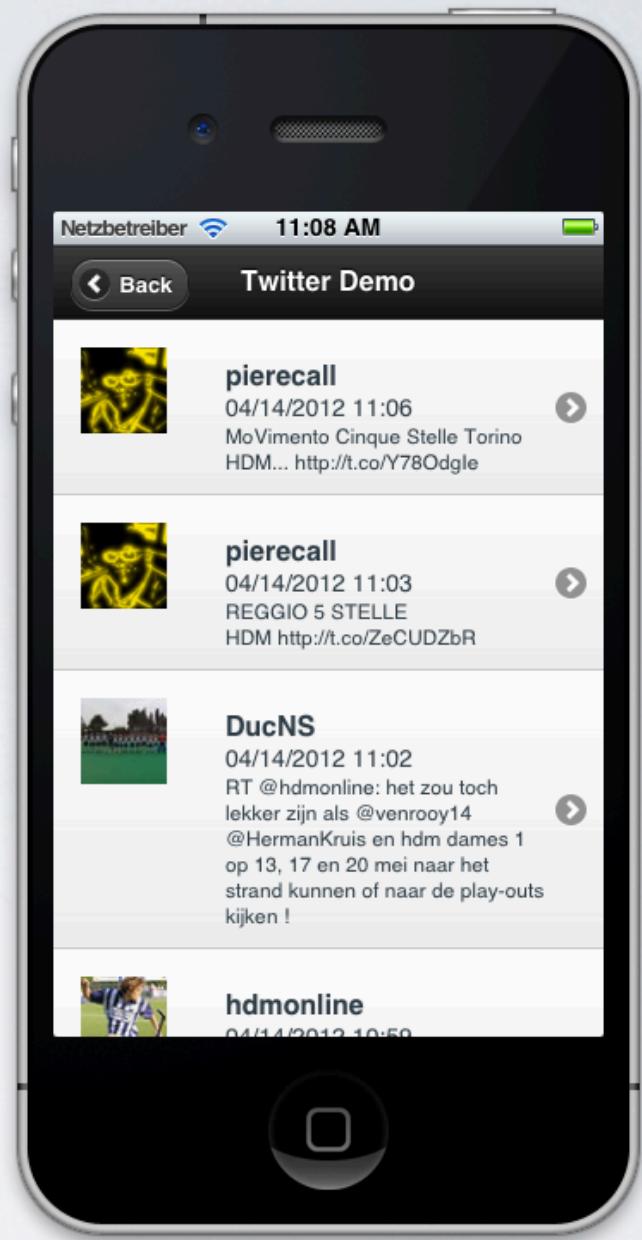
- Events werden zentral verwaltet über **M.EventDispatcher**
- **M.EventDispatcher** leitet Events an Listener weiter
- Registrierung im View über Property **events**:

Zukunft





<https://github.com/gsuslol/TwitterDemo>



goo.gl/qwLyn