

# Liste offener Punkte [LoP]

---

## Grundstruktur & Organisatorisches

- Sameln von Satzketzen die thematische Tangenten anschneiden. [kontinuierlich]
- Farbschema für Diagramme festlegen & diese vereinheitlichen [WIP]

## Bearbeitung

- Händische Tests (Dart-DevTools):
  - Test-Vorgehen aufschreiben [WIP]

## Ergebnisse

- Daten sammeln
  - Dart-DevTools
- Datenauswertung:
  - Entsprechendes Kapitel verfassen
  - aussagekräftige Graphiken erstellen [WIP]

## Layout & Diagramme

- ☐ Messergebnisse ordentlich darstellen
- ☐ Inhaltlicher Feinschliff (Grammatik, Rechtschreibung, etc.)
- ☐ Finale Überprüfung des Layouts, nachdem alle Texte geschrieben sind

---

## Abgearbeitete Punkte:

- Die jeweiligen ZVM implementieren (201021-1155)
  - ☒ setState
  - ☒ Provider
  - ☒ BLoC
  - ☒ Redux (201020-1717)
  - ☒ MobX (201021-1155)
- Einleitung/Exposé überarbeiten (201024-1540)
- Stand der Wissenschaft/Technik beleuchten (201025-1912)
- Integrationstests zur Messung der Renderzeiten für jede ZVM implementiert (201028-1630)
- Kapitel über die Applikation, unabhängig von Zustandsverwaltungsmethoden (folgend ZVM), verfassen
  - Welche Eigenschaften des Frameworks werden genutzt?(201102-1740)
  - Wie interagiert die Softwarearchitektur mit den untersuchten Kriterien (201103-1520)

- App anpassen, um beim Zustandswechsel einen kleineren Teils des Widget-Trees auszutauschen. (Optimierung der Performanz!)(201107-1425)
- Kapitel über die Applikation, unabhängig von Zustandsverwaltungsmethoden erweitern und anpassen:
  - Datenmodell beschreiben (201110-1345)
  - Schaubilder erstellen (201111-1220)
  - Verwendete Tools beschreiben [Rohfassung] (201112-1515)
- Diagrammsoftware wählen (201116- 1000)
- In der Providerimplementierung Selector-Widgets statt Consumer-Widgets verwenden (201117-1650)
- Kapitel über verwendete Tools
  - Schaubilder zur Applikation erstellen (201119-1400)
  - ZVM (201123-1845)
    - Datenfluss der einzelnen ZVM veranschaulichen
  - ZVM APIs untersuchen (201124-1000)
- Für jedes Kriterium jede ZVM abarbeiten (jeweils Code Snippet erstellen und beschreiben)(20121-1915)
  - ☒ Definition, Deklaration und Initialisierung eines Globalen Zustands (201124-1815)
    - ☒ Store, Notifier, BLoC, Actions und Events (201125-16:00)
    - ☒ StoreProvider, BlocProvider, ChangeNotifierProvider etc. (201126-1330)
  - ☒ Codeausschnitt der Abfrage zur Darstellung (201130-1130)
  - ☒ Codeausschnitt der Zustandsänderung (201130-1430)
  - ☒ Beschreibung der Abfrage zur Darstellung (20121-1530)
  - ☒ Beschreibung der Zustandsänderung (20121-1915)
- buildWhen-Bedingung für alle BlocBuilder implementieren (201202-1515)
- ☒ Bisherige "Methodik" aufteilen (in Methodik und Bearbeitung/Implementierung) (201201-1820)
- ☒ Beschreibung der Methodik überarbeiten (201210-0100)
- Kapitel über verwendete Tools
  - ☒ Flutter Driver (201214-1600)
  - ☒ Dart DevTools (201214-1900)
- Automatisierte Tests (Flutter Driver):
  - ~~CPU Taktfrequenz vereinheitlichen (CPU Governor überbrücken, am Beispiel von Filip Hracek)~~  
[evtl. irrelevant geworden!]
- Daten sammeln

- Flutter Driver (201212-1400)
  - Daten als Datei exportieren (201212-1400)
  - Test-Code beschreiben (201227-1530)

Darstellung der gesammelten Daten:

- Gelesene Daten verstehen/ordnen (201221-1920)
- Entsprechende Software finden (PyPlot Bibliothek & Libre Office Calc) (201226-2230)
- Sicherstellen, dass die CPU immer im gleichen takt arbeitet (201228-1400)
- Zu viele Videospiele die mich ablenken (24/7)