**Textadventure – The Little Mermaid**

|  |  |
| --- | --- |
| IPA-Daten |  |
| Firma | enersis suisse AG |
| Abteilung | Development/ IT |
| Autor | Nicole Sager |
| Ausgabedatum | 10.05.2017 |
| Version | V1.0 |
| Status | Genehmigt, zur Nutzung |

|  |  |
| --- | --- |
| Beteiligter Personenkreis | |
| In der Genehmigung | -- |
| Valid-Experte | Ramun Hofmann |
| In der Durchführung | Nicole Sager, Mansur Esmann, Berkay Üner |
| Fachvorgesetzter | (Nora Kleisli) |
| Tech. Fachvorgesetzter | -- |
| Hauptexperte | Ramun Hofmann |
| Zweit-Experte | Vanessa Meister |
| Berufsbilder | Roland Dardel |

IPA 2017, Kt. Bern

**Dokumentinformationen**

Änderungskontrolle, Prüfung, Genehmigung

|  | | | |
| --- | --- | --- | --- |
| Version | Datum | Name | Beschreibung |
| Vorlage | 23.06.2013 | A. Mueller | Dokumentvorlage QV2013, Version V1.0 |
| X0.01 | 26.04.2017 | N. Sager | Teil 1: Aufgabenstellung, Projektorganisation, Organisation der IPA |
| X0.02 | 27.04.2017 | N. Sager | Teil 2: Initialisierungsphase, Ist-Analyse |
| X0.03 | 28.04.2017 | N. Sager | Teil 2: Initialisierungsphase, ISDS, Lösungssuche, Risikoanalyse, Variantenentscheid |
| X0.04 | 01.05.2017 | N. Sager | Teil 3: Initialisierungsphase, Optimierungen  Teil 2: Konzeptphase, Systemarchitektur |
| X0.05 | 02.05.2017 | N. Sager | Teil 2: Konzeptphase, Systemarchitektur, Klassendiagramm, Machbarkeitsprüfung, |
| V1.0 | 10.05.2017 | N. Sager | Abnahme des Projektes |

**Verwendete Abkürzungen**

| Abkürzung | Bedeutung |
| --- | --- |
| API | Application Programming Interface |
| bzw. | beziehungsweise |
| CSS | Cascading Style Sheets |
| D3 | Data Driven Documents |
| FV | Fachvorgesetzter |
| HTML | Hypertext Markup Language |
| IDE | integrated development environment |
| IPA | Individuelle praktische Arbeit |
| ISDS | Informationssicherheit und Datenschutz |
| JSON | JavaScript Object Notation |
| MODX | Name der MODX Firma |
| OdA | Organisation der Arbeitswelt |
| QV | Qualifikationsverfahren |
| SAP HANA | Systeme Anwendungen Produkte, High Performance Analytic Appliance |
| SVG | Scalable Vector Graphics |
| vgl. | vergleiche |
| VZ | Verzeichnis |

**Inhaltsverzeichnis**

[1. Aufgabenstellung 2](#_Toc482180126)

[1.1 Titel der Facharbeit 2](#_Toc482180127)

[1.2 Thematik 2](#_Toc482180128)

[1.3 Ausgangslage 2](#_Toc482180129)

[1.4 Detaillierte Aufgabenstellung 2](#_Toc482180130)

[1.5 Mittel und Methoden inklusive Projektmethode 3](#_Toc482180131)

[1.6 Vorkenntnisse 3](#_Toc482180132)

[1.7 Vorarbeiten 4](#_Toc482180133)

[1.8 Neue Lerninhalte 4](#_Toc482180134)

[1.9 Arbeiten des Kandidaten im Schwerpunkt während dem 3. und 4. Lehrjahr 4](#_Toc482180135)

[2. Detaillierte Projektmethode 5](#_Toc482180136)

[2.1 Projektmethode 5](#_Toc482180137)

[2.2 Szenario 5](#_Toc482180138)

[2.3 Phasen 7](#_Toc482180139)

[2.4 Module 7](#_Toc482180140)

[2.5 Projektorganisation 9](#_Toc482180141)

[2.5.1 Projektrollen 9](#_Toc482180142)

[3. Zeitplan 10](#_Toc482180143)

[4. Organisation der IPA 12](#_Toc482180144)

[4.1 Infrastruktur 12](#_Toc482180145)

[4.2 Datensicherung der IPA 12](#_Toc482180146)

[4.3 Ordnerstruktur 13](#_Toc482180147)

[5. Firmenstandards 15](#_Toc482180148)

[6. Arbeitsjournal 16](#_Toc482180149)

[6.1 Erster halber Tag: Mittwoch, 26.04.2017 17](#_Toc482180150)

[6.2 Zweiter Tag: Donnerstag, 27.04.2017 19](#_Toc482180151)

[6.3 Dritter Tag: Freitag, 28.04.2017 21](#_Toc482180152)

[6.4 Vierter Tag: Montag, 01.05.2017 23](#_Toc482180153)

[6.5 Fünfter Tag: Dienstag, 02.05.2017 25](#_Toc482180154)

[6.6 Sechster Tag: Mittwoch, 03.05.2017 27](#_Toc482180155)

[6.7 Siebter Tag: Donnerstag, 04.05.2017 29](#_Toc482180156)

[6.8 Achter Tag: Freitag, 05.05.2017 30](#_Toc482180157)

[6.9 Neunter Tag: Montag, 06.05.2017 32](#_Toc482180158)

[6.10 Zehnter Tag: Dienstag, 07.05.2017 34](#_Toc482180159)

[7. Projektjournal 35](#_Toc482180160)

[8. Abschlussbericht 36](#_Toc482180161)

[8.1 Vergleich Ist/Soll 36](#_Toc482180162)

[8.2 Mittelbedarf 37](#_Toc482180163)

[8.3 Realisierungsbericht 37](#_Toc482180164)

[8.3.1 Ungeplante Ereignisse 37](#_Toc482180165)

[8.3.2 Erkenntnisse der Realisierung 37](#_Toc482180166)

[8.4 Testbericht 38](#_Toc482180167)

[8.5 Fazit zum IPA (Projekt) 38](#_Toc482180168)

[8.6 Persönliches Fazit 38](#_Toc482180169)

[8.7 Schlussreflexion 39](#_Toc482180170)

[9. Unterschriften Teil 1 40](#_Toc482180171)

[Teil 2: Projektdokumentation 41](#_Toc482180172)

[10. Initialisierung 42](#_Toc482180173)

[10.1 Studie; Ist-Zustand 42](#_Toc482180174)

[10.1.1 Das greenited Projekt 42](#_Toc482180175)

[10.1.2 Barchart 43](#_Toc482180176)

[10.1.3 Donutchart 47](#_Toc482180177)

[10.2 Meilensteine 50](#_Toc482180178)

[10.2.1 SMART 50](#_Toc482180179)

[10.2.2 Persönliche Vorgehensziele 50](#_Toc482180180)

[10.2.3 Projektziele 51](#_Toc482180181)

[10.3 Anforderungen 52](#_Toc482180182)

[10.3.1 Funktionale Anforderungen 52](#_Toc482180183)

[10.3.2 Nicht funktionale Anforderungen 52](#_Toc482180184)

[10.4 Informationssicherheit und Datenschutz (ISDS) 53](#_Toc482180185)

[10.5 Machbarkeitsprüfung Initialisierungsphase 53](#_Toc482180186)

[10.6 Risikoanalyse 57](#_Toc482180187)

[10.7 Risikographen 58](#_Toc482180188)

[10.7.1 Risikograph vor den Massnahmen 58](#_Toc482180189)

[10.7.2 Risikograph nach den Massnahmen 58](#_Toc482180190)

[10.7.3 Kurze Stellungnahmen zu den Risiken 58](#_Toc482180191)

[10.8 Lösungen suchen 59](#_Toc482180192)

[10.8.1 Aufsetzen der Chart Library 59](#_Toc482180193)

[10.8.2 Aufruf 59](#_Toc482180194)

[10.9 Varianten 60](#_Toc482180195)

[10.9.1 Gewichtung der Kriterien 60](#_Toc482180196)

[10.9.2 React Redux Starter Kit 61](#_Toc482180197)

[10.9.3 ES6 Karma Jasmine Webpack Boilerplate 61](#_Toc482180198)

[10.9.4 Webpack library starter 62](#_Toc482180199)

[10.10 Variantenentscheid 63](#_Toc482180200)

[10.10.1 Begründung 63](#_Toc482180201)

[11. Konzept 64](#_Toc482180202)

[11.1 Allgemein 64](#_Toc482180203)

[11.1.1 Grundlagen 64](#_Toc482180204)

[11.1.2 ISDS 64](#_Toc482180205)

[11.2 Architektur 65](#_Toc482180206)

[11.2.1 Vom Source Code zur fertigen Library 65](#_Toc482180207)

[11.2.2 Einbindung der Charts auf einer Webseite 66](#_Toc482180208)

[11.2.3 Klassendiagramm 67](#_Toc482180209)

[11.2.4 Datenstruktur 68](#_Toc482180210)

[11.3 Anwendungsfall 68](#_Toc482180211)

[11.4 Machbarkeitsprüfung Konzeptphase 69](#_Toc482180212)

[11.5 Testkonzept 70](#_Toc482180213)

[11.5.1 Grundlagen 70](#_Toc482180214)

[11.5.2 Test Szenario 70](#_Toc482180215)

[11.5.3 Testrahmen 70](#_Toc482180216)

[11.5.4 Testvorgehen 70](#_Toc482180217)

[11.5.5 Testmethode 70](#_Toc482180218)

[11.5.6 Testfälle 71](#_Toc482180219)

[11.5.7 Testziele 73](#_Toc482180220)

[12. Realisierung 74](#_Toc482180221)

[12.1 Template 74](#_Toc482180222)

[12.1.1 ECMAScript 6 74](#_Toc482180223)

[12.1.2 Babel 75](#_Toc482180224)

[12.1.3 ESLint 75](#_Toc482180225)

[12.1.4 Unit Testing 76](#_Toc482180226)

[12.1.5 Vom Source Code zur fertigen Library 77](#_Toc482180227)

[12.2 Klassendiagramm 79](#_Toc482180228)

[12.3 Entwicklung 80](#_Toc482180229)

[12.3.1 ensChart Klasse 80](#_Toc482180230)

[12.3.2 Ausgelagerte Klassen 82](#_Toc482180231)

[12.4 Testprotokoll 83](#_Toc482180232)

[12.4.1 Constructor 84](#_Toc482180233)

[12.4.2 Title 85](#_Toc482180234)

[12.4.3 Legend 86](#_Toc482180235)

[12.4.4 DurationTime 87](#_Toc482180236)

[12.4.5 Color Klasse 88](#_Toc482180237)

[12.4.6 Ease Klasse 90](#_Toc482180238)

[12.4.7 Scale Klasse 91](#_Toc482180239)

[12.4.8 Size Klasse 92](#_Toc482180240)

[12.4.9 Typ Klasse 94](#_Toc482180241)

[12.5 Vergleich der Anforderung und der Umsetzung 95](#_Toc482180242)

[12.5.1 Funktionale Anforderungen 95](#_Toc482180243)

[12.5.2 Nicht funktionale Anforderungen 95](#_Toc482180244)

[12.6 API Dokumentation vorbereiten 96](#_Toc482180245)

[12.6.1 Struktur der API Dokumentation 96](#_Toc482180246)

[13. API documentation ensChart 97](#_Toc482180247)

[General Information 97](#_Toc482180248)

[Getting started 97](#_Toc482180249)

[Initializing the Library 97](#_Toc482180250)

[The different possible parameters you can set 97](#_Toc482180251)

[Lib.title(arg) 97](#_Toc482180252)

[Lib.data(arg) 98](#_Toc482180253)

[Lib.legend(arg) 98](#_Toc482180254)

[Lib.color(arg) 98](#_Toc482180255)

[Lib.ease(arg) 98](#_Toc482180256)

[Lib.durationTime(arg) 99](#_Toc482180257)

[Lib.scale(arg) 99](#_Toc482180258)

[Lib.width(arg) 99](#_Toc482180259)

[Lib.height(arg) 99](#_Toc482180260)

[Lib.type(arg) 99](#_Toc482180261)

[If something doesn’t work 100](#_Toc482180262)

[If you want to include a new Chart into the Library 100](#_Toc482180263)

[14. Literatur und Quellenverzeichnis 101](#_Toc482180264)

[15. Glossar 102](#_Toc482180265)

[16. Unterschriften für Abnahme 104](#_Toc482180266)

[17. Anhang 105](#_Toc482180267)

[17.1 Entwickler Richtlinien 105](#_Toc482180268)

[1. Einleitung 107](#_Toc482180269)

[1.1. Ziel und Zweck dieses Dokuments 107](#_Toc482180270)

[1.2. Vorschriften und Standards 107](#_Toc482180271)

[2. Allgemeine Richtlinien und Verantwortlichkeiten 107](#_Toc482180272)

[2.1. Zuständigkeiten 107](#_Toc482180273)

[2.2. Technologische Verantwortlichkeiten 108](#_Toc482180274)

[3. Projektmanagement 108](#_Toc482180275)

[3.1. Methodik 108](#_Toc482180276)

[3.2. Projektmanagement Tool 109](#_Toc482180277)

[4. Softwareumgebung 110](#_Toc482180278)

[5. Entwicklungsrichtlinien 110](#_Toc482180279)

[5.1. Allgemein 110](#_Toc482180280)

[5.2. Grids 111](#_Toc482180281)

[5.3. Browserkompatibilität 111](#_Toc482180282)

[5.4. Coding Style 111](#_Toc482180283)

[5.4.1. Allgemein 111](#_Toc482180284)

[5.4.2. C# 112](#_Toc482180285)

[5.4.3. JavaScript 113](#_Toc482180286)

[5.4.4. PL-SQL 114](#_Toc482180287)

[6. Dokumentation 116](#_Toc482180288)

[17.2 Protokolle 117](#_Toc482180289)

[17.2.1 Erster Besuchstag 117](#_Toc482180290)

[17.2.2 zweiter Besuchstag 120](#_Toc482180291)

[1 Ablauf 121](#_Toc482180292)

[2 Wichtige Punkte und Entscheidungen 121](#_Toc482180293)

[17.3 Source Code 122](#_Toc482180294)

[17.3.1 index.js 122](#_Toc482180295)

[17.3.2 color.config.js 128](#_Toc482180296)

[17.3.3 ease.config.js 129](#_Toc482180297)

[17.3.4 scale.config.js 130](#_Toc482180298)

[17.3.5 size.config.js 131](#_Toc482180299)

[17.3.6 type.config.js 131](#_Toc482180300)

[17.3.7 main.chart.js 133](#_Toc482180301)

[17.3.8 donut.chart.js 135](#_Toc482180302)

[17.3.9 pie.chart.js 135](#_Toc482180303)

[17.3.10 bar.chart.js 138](#_Toc482180304)

[17.3.11 horizontal.js 139](#_Toc482180305)

[17.3.12 vertical.js 140](#_Toc482180306)

[17.3.13 index.html 141](#_Toc482180307)

[17.3.14 library.spec.js 142](#_Toc482180308)

[17.3.15 color.spec.js 145](#_Toc482180309)

[17.3.16 ease.spec.js 148](#_Toc482180310)

[17.3.17 scale.spec.js 149](#_Toc482180311)

[17.3.18 size.spec.js 150](#_Toc482180312)

[17.3.19 type.spec.js 154](#_Toc482180313)

[17.3.20 ensChart.js & ensChart.js.map 155](#_Toc482180314)

**Abbildungsverzeichnis**

[Abbildung 1: Detaillierte Projektmethode 5](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180315)

[Abbildung 2: Szenarioleitfaden 6](#_Toc482180316)

[Abbildung 3: Module die für das Projekt verwendet wurden 8](#_Toc482180317)

[Abbildung 4: Projektorganisation 9](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180318)

[Abbildung 5: Zeitplan 10](#_Toc482180319)

[Abbildung 6: Legende Zeitplan 11](#_Toc482180320)

[Abbildung 7: Arbeitsplatz 12](#_Toc482180321)

[Abbildung 8: Ordnerstruktur 13](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180322)

[Abbildung 9: Ordner 02\_Dokumentation 14](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180323)

[Abbildung 10: Inhalt des ersten Tagesordners 14](#_Toc482180324)

[Abbildung 11: Unterschrift Teil 1 40](#_Toc482180325)

[Abbildung 12: greenited Projekt - So funktioniert es 42](#_Toc482180326)

[Abbildung 13: greenited Projekt - Webseite 43](#_Toc482180327)

[Abbildung 14: greenited Projekt - Stacked Bar Chart 43](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180328)

[Abbildung 15: HTML DIV-Element Barchart 44](#_Toc482180329)

[Abbildung 16: Donutchart «Energiemix regional» 47](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180330)

[Abbildung 17: HTML-DIV-Element Donutchart 47](#_Toc482180331)

[Abbildung 18: Risikograph 58](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180332)

[Abbildung 19: Risikograph 58](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180333)

[Abbildung 20: Bewertungsmöglichkeiten 60](#_Toc482180334)

[Abbildung 21: Einbindung der Charts auf einer Webseite 66](#_Toc482180335)

[Abbildung 22: Klassendiagramm 67](#_Toc482180336)

[Abbildung 23: Datenstruktur 68](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180337)

[Abbildung 24: Anwendungsfälle 69](#_Toc482180338)

[Abbildung 25: ecma international 74](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180339)

[Abbildung 26: BABEL 75](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180340)

[Abbildung 27: Babel Beispiel 75](#_Toc482180341)

[Abbildung 28: ESLint 75](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180342)

[Abbildung 29: mocha und chai 76](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180343)

[Abbildung 30: Testing des Titels 76](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180344)

[Abbildung 31: Klassendiagramm 79](#_Toc482180345)

[Abbildung 32: testing the Constructor 84](#_Toc482180346)

[Abbildung 33: Testing the Title 85](#_Toc482180347)

[Abbildung 34: Testing the Legend 86](#_Toc482180348)

[Abbildung 35: Testing the DurationTime 87](#_Toc482180349)

[Abbildung 36: Testing the Color Class 88](#_Toc482180350)

[Abbildung 37: Testing the Color Class 89](#_Toc482180351)

[Abbildung 38: Testing the Ease Class 90](#_Toc482180352)

[Abbildung 39: Testing the Scale Class 91](#_Toc482180353)

[Abbildung 40: Testing the Size Class (width) 92](#_Toc482180354)

[Abbildung 41: Testing the Size Class (height) 93](#_Toc482180355)

[Abbildung 42: Testing Type Class 94](#_Toc482180356)

[Abbildung 43: Unterschrift für Abnahme 104](#_Toc482180357)

**Tabellenverzeichnis**

[Tabelle 1: Projektrollen 9](#_Toc482180358)

[Tabelle 2: Inhalte der Ordnerstruktur 13](#_Toc482180359)

[Tabelle 3: Projektjournal 35](#_Toc482180360)

[Tabelle 4: Mittelbedarf 37](#_Toc482180361)

[Tabelle 5: Funktionen Barchart 46](#_Toc482180362)

[Tabelle 6: Funktionen Donutchart 49](#_Toc482180363)

[Tabelle 7: Funktionen des Barcharts 49](#_Toc482180364)

[Tabelle 8: SMART 50](#_Toc482180365)

[Tabelle 9: Risikoanalyse 57](#_Toc482180366)

[Tabelle 10: Risikobeschreibung 58](#_Toc482180367)

[Tabelle 11: Risikobeschreibung 58](#_Toc482180368)

[Tabelle 12: Kriterien 60](#_Toc482180369)

[Tabelle 13: React Redux Starter Kit 61](#_Toc482180370)

[Tabelle 14: ES6 Karma Jasmine Webpack Boilerplate 62](#_Toc482180371)

[Tabelle 15: Webpack library starter 62](#_Toc482180372)

[Tabelle 16: Variantenentscheid 63](#_Toc482180373)

[Tabelle 17: Vom Source Code zur fertigen Library 65](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180374)

[Tabelle 18: Testfälle 72](#_Toc482180375)

[Tabelle 19: Hauptbausteine von Chai 77](#_Toc482180376)

[Tabelle 20: Vom Source Code zur fertigen Library 77](file:///D:\nicol\Dropbox%20(enersis)\55%20Shared%20Services\Office%20IT\03_Projekte\Projekt_IPA_SNI\IPA_Nicole_2017\02_Dokumentation\11_Tag\Entwicklung%20einer%20Chart%20Library%20zur%20internen%20Verwendung_V1.0.docx#_Toc482180377)

[Tabelle 21: Was ist webpack 78](#_Toc482180378)

[Tabelle 22: Glossar 103](#_Toc482180379)

**Kurzfassung des IPA-Berichtes**

Ausgangssituation

Umsetzung

Ergebnis

2017

Textadventure – the little mermaid

AutoriN: Nicole Sager

Teil 1: Ablauf und Umfeld

# Aufgabenstellung

Dieses Kapitel beschreibt die Aufgabenstellung gemäss Originaleingabe des Fachvorgesetzten auf PKORG.

## Titel der Facharbeit

Textadventure – The Little Mermaid

## Thematik

Ein simples Textadventure soll erstellt werden. Dieses soll von der Originalgeschichte der kleinen Meerjungfrau (geschrieben von Hans Christian Andersen) inspiriert werden. Für die Umsetzung soll Python verwendet werden.

## Ausgangslage

Um sich für die IPA entsprechend vorzubereiten wird eine Probe-IPA durchgeführt. Im Gegensatz zu der „richtigen“ IPA, welche 10 Tage dauert, wird diese erste IPA 5 Tage dauern. Python wurde vorgegeben da dies die Programmiersprache ist welche im normalen Arbeitsumfeld verwendet wird.

## Detaillierte Aufgabenstellung

Gefordert wird ein Textbasiertes Spiel. Es werden mindestens 3 verschiedene Auswahlmöglichkeiten gefordert welche jeweils eine Auswirkung auf das ende des Spiels haben müssen. Die visuellen Eigenschaften des Projektes stehen nicht im Vordergrund. Die ausgegebene Sprache des Spiels ist Englisch.

Das Projekt soll mit Unit-test getestet werden.

## Mittel und Methoden inklusive Projektmethode

HERMES 5.1 IPA soll als Projektmethode verwendet werden.

Zu benutzten ist das von der Firma zur Verfügung gestellte Macbook Pro.

Das Programm soll mit Python realisiert werden.

## Vorkenntnisse

Eine Probe IPA, eine offizielle IPA. Ein 1-wöchentliches training bezüglich Python.

## Vorarbeiten

Die teilweise Wiederverwendung bereits erstellter Diagramme.

## Neue Lerninhalte

Ein eigenes Projekt mit Python umzusetzen.

## Arbeiten des Kandidaten im Schwerpunkt während dem 3. und 4. Lehrjahr

Die Lernende absolviert den vierjährigen Bildungsgang der Informatikmittelschule in der Fachrichtung Applikationsentwicklung und kaufmännische Berufsmaturität.

Auf die dreijährige schulische Vollzeitausbildung am bwd folgt für die Lernende ein Praktikumsjahr im Betrieb der enersis suisse AG und am 02.08.2017 startete die lernende beim Betrieb 89grad GmbH. Die Ausbildungszeit "on-the-job" ist daher relativ kurz.

# Detaillierte Projektmethode

In diesem Teil der Arbeit werden die einzelnen Phasen und Module dokumentiert.

Im ersten Abschnitt wird die vordefinierte Projektmethode genauer erklärt. Wie in Kapitel 1.5 Projektmethode beschrieben, wird Hermes 5.1 IPA angewendet.

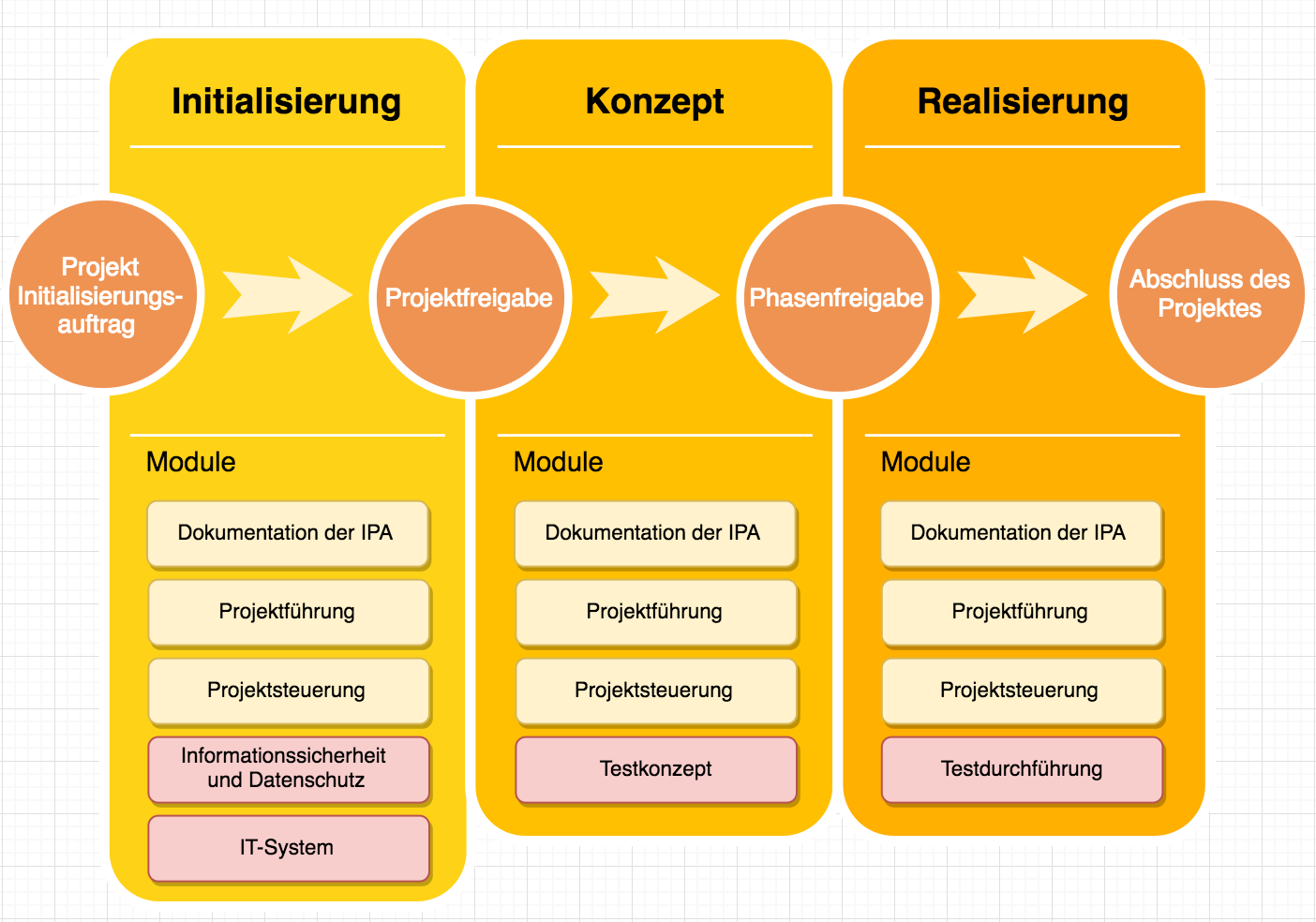


Abbildung : Detaillierte Projektmethode

Das ebengezeigte Diagramm Zeigt in welchen Phasen welche Module durchgeführt werden.

## Projektmethode

Hermes 5.1 IPA ist für die Durchführung von IPAs entwickelt worden. Es ist eine stark vereinfachte Form von Hermes 5.1.[[1]](#footnote-1)

## Phasen

Das Phasenmodell gliedert den Lebenszyklus des Projekts und schafft die Voraussetzung für das gemeinsame Verständnis der Projektbeteiligten zum Projektablauf. Das Phasenmodell bildet unter anderem ebenfalls die Grundlage für die finanzielle Steuerung des Projekts.

Bei Phasenfreigabe werden die Mittel (finanziell, personell, infrastrukturell) für die jeweils anschliessende Phase durch den Auftraggeber freigegeben.

Die vier Phasen einer Hermesplanung sind die Initialisierungsphase, die Konzeptphase, die Realisierungsphase und zum Schluss die Einführungsphase

Es ist zu beachten das bereits im Vorfeld festgelegt wurde, dass es keine eigentliche Phasenfreigabe geben wird, da dies zu zeitintensiv wäre. Ausserdem wird in dieser Probe IPA keine Einführungsphase durchlaufen.

## Module

Folgende Module wurden in dieser IPA verwendet:

|  |  |
| --- | --- |
| **Modul** | **Beschreibung** |
| **Dokumentation der IPA** | * Der Verlauf der IPA wird dokumentiert und stets aktuell gehalten * Das Arbeitsjournal wird täglich geschrieben |
| **Projektführung** | * Das Projekt planen, führen und in den definierten Rahmenbedingungen von Zeit und Kosten mit dem geforderten Ergebnis zum Ziel bringen * Die Interessen der Stakeholder kennen, die Kommunikation führen und Entscheide sicherstellen * Risiken managen, Probleme bewältigen und Erfahrungen berücksichtigen * Qualitätssicherung führen gemäss IPA Richtlinien und Firmenrichtlinien |
| **Projektsteuerung** | * Das Projekt initialisieren, kontinuierlich steuern und mit den übergeordneten Zielen und Vorgaben der Stammorganisation in Übereinstimmung halten * Anliegen der Stakeholder berücksichtigen und integrieren, Risiken managen und Entscheide treffen * Das Projekt abschliessen |
| **Informationssicherheit und Datenschutz** | • Anforderungen der Sicherheit und des Datenschutzes ermitteln, Risiken bewerten und Massnahmen zur Erfüllung der Anforderungen konzipieren und umsetzen   * Das ISDS-Konzept erstellen und die Ergebnisse laufend dokumentieren * Datenschutz gemäss Firmenrichtlinien umsetzten |
| **IT-System** | * Das IT-System realisieren und dokumentieren * Die Systemanforderungen verfeinern, die Systemarchitektur erarbeiten und die Machbarkeit überprüfen * Die Detailspezifikation erarbeiten sowie das System und die Integration realisieren |
| **Testkonzept** | * Konzeption des Testings * Szenarien erstellen * Vorgehen definieren wie Strategie und Methodiken * Tests vorbereiten (Szenario) |
| **Testdurchführung** | * Tests durchführen und dokumentieren * Testfazit erstellen * Erkenntnisse ableiten |

Abbildung 2: Module die für das Projekt verwendet wurden

## Szenario

Das Szenario legt ein grundsätzliches Schema für die Inhalte der Phasen fest. Bei dieser Probe IPA wurde das Szenario „IT-Individualanwendung“ ausgewählt.

|  |  |
| --- | --- |
| **Szenario** | **Beschreibung** |
| **IT-Individualanwendung** | Für die spezifischen Bedürfnisse eines Fachbereichs einer IT-Anwendung entwickeln und technisch und organisatorisch integrieren |

## Projektorganisation

Die Projektorganisation stellt die hierarchische Struktur des Projektes dar.

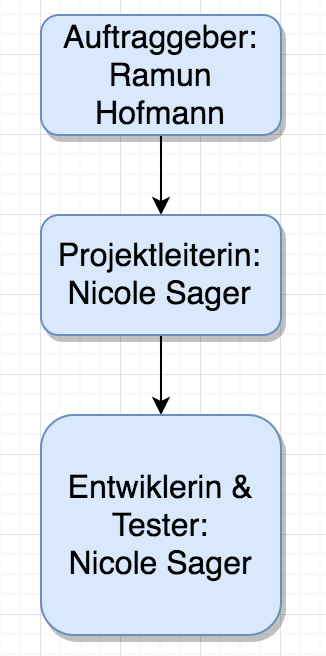


Abbildung 4: Projektorganisation

### Projektrollen

Die IPA Projektrollen werden in der folgenden Tabelle kurz erläutert:

|  |  |
| --- | --- |
| Rolle | Rollenbeschreibung |
| Auftraggeber | Erteilt den Auftrag mit den gewünschten Anforderungen. |
| Projektleiter | Ist für die Planung und Steuerung des Projektes verantwortlich |
| Entwickler | Setzt das System um. |

Tabelle 1: Projektrollen

# Zeitplan!!!

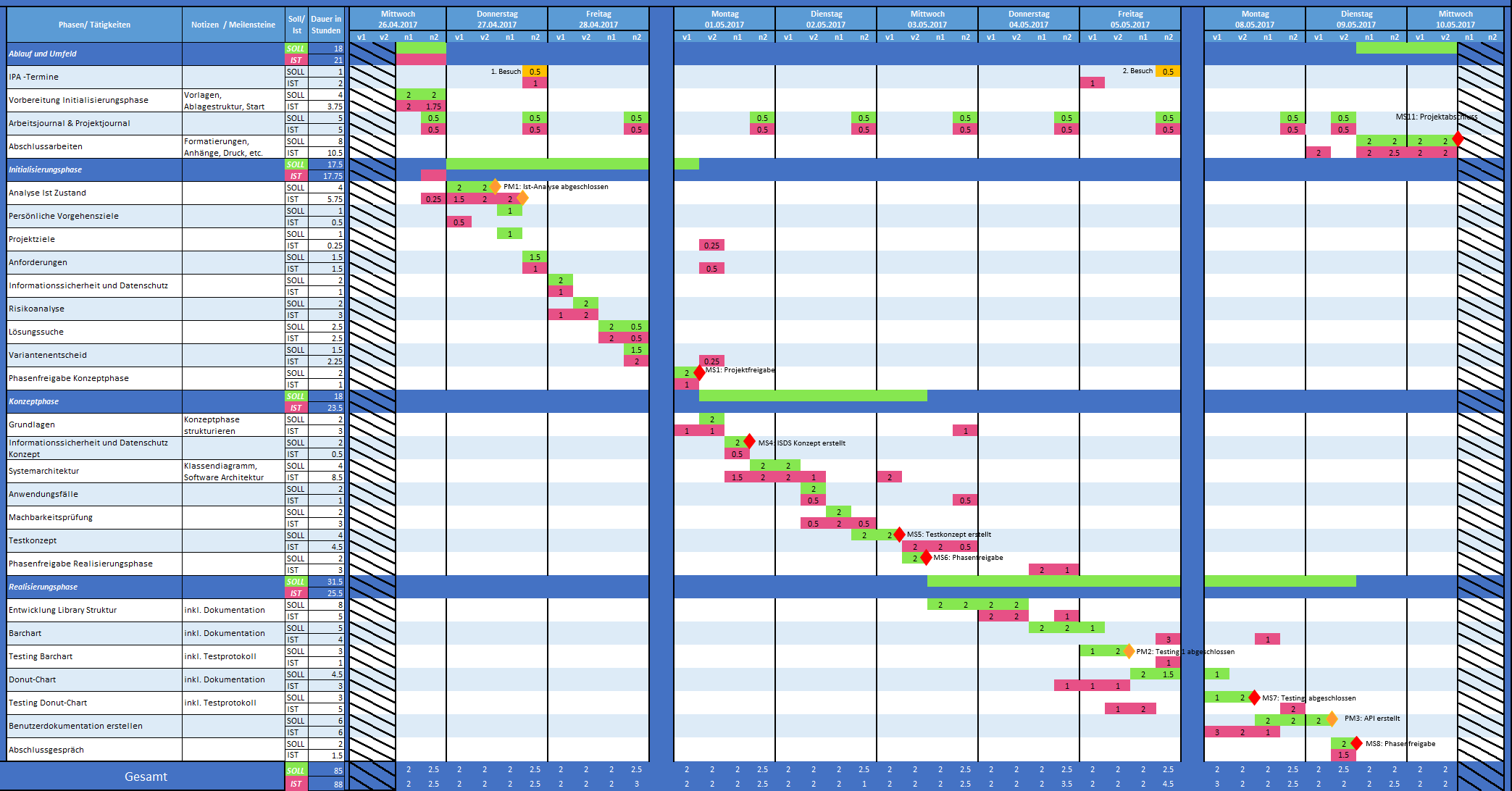


Abbildung 4: Zeitplan

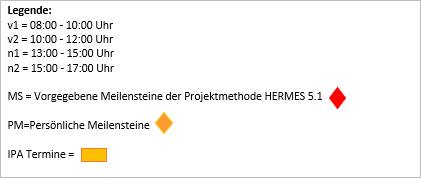


Abbildung 5: Legende Zeitplan

Die Aufgezeigten roten Meilensteine sind die von Hermes 5.1 IPA vorgegebenen Meilensteine. Hierbei wurden bewusst die jeweiligen Phasenfreigaben ausgelassen.

# Organisation der Probe IPA !!!



Abbildung 6: Arbeitsplatz

Der Arbeitsplatz besteht aus einem Macbook Pro einem verfügbaren Stuhl/Tisch.

Zusätzlich stehen verschiedene Büromaterialien zur Verfügung.

## Infrastruktur

Wie bereits im Kapitel 1 vermerkt erfolgt die Realisierung im Rahmen der lokalen Infrastruktur.

* Einen eigenen PC
* Entwicklungsumgebung: Visual Studio Code- als Texteditor

Zusätzlich werden verwendet:

* Draw.io – für das Erstellen Diverser Diagramme

## Datensicherung der IPA

Die Sicherung erfolgt einmaltäglich auf Github sowie während des normalen Arbeitens auf der Lokalen Maschine. Eine Umstellung der zentralen Ablage ist geplant, dies kann jedoch für die IPA nicht mehr berücksichtigt werden.

Die Dokumente, die während dieser Arbeit erstellt werden, werden laufend gespeichert. Zudem erfolgt eine Zweitspeicherung einmal pro Tag auf einem USB-Stick.

## Ordnerstruktur



Abbildung : Ordnerstruktur

Die vorherige Abbildung zeigt die für die Projektorganisation erstellte Ordnerstruktur auf.

In der folgenden Liste wird kurz erläutert, welche Inhalt die jeweiligen Ordner besitzen bzw. besitzen werden.

|  |  |  |
| --- | --- | --- |
| **Nr.** | **Name** | **Inhalte** |
| **00** | Unterlagen Pkorg | Alle Vorlagen, die auf der Pkorg Seite zur Verfügung stehen, sowie eine PDF Version der Kandidatenseiten und dem dazugehörigen Detailbeschrieb werden darin abgespeichert. |
| **01** | Vorlagen und Vorarbeiten | Die bereits als Vorarbeit erstellten Dokumente & die als Übung durchgeführten IPAs welche als Hilfestellung verwendet werden |
| **02** | Dokumentation | Enthält die Ordnerstruktur für die einzelnen Tage |
| **07** | Bilder | Die in diesem Dokument verwendeten Bilder |
| **98** | Sonstiges | Alles Dokumente, welche keiner der Hauptkategorien zugeordnet werden können |
| **99** | SrcCode[[2]](#footnote-2) | Der für das Projekt erstellte Quelltext |

Tabelle 2: Inhalte der Ordnerstruktur

In einem der einzelnen Tage ist die Dokumentation und der Zeitplan enthalten.

# Firmenstandards

Die Firma 89grad besitzt momentan keine relevanten Firmenstandards.

# Arbeitsjournal

Gemäss Art. 5 Absatz 2 der Wegleitung über die individuelle praktische Arbeit (IPA) an Lehrabschlussprüfungen des BBT vom 27. August 2001 gilt:

*„Die zu prüfende Person führt ein Arbeitsjournal. Sie dokumentiert darin täglich das Vorgehen, den Stand der Prüfungsarbeit, sämtliche fremde Hilfestellungen (auch das Internet ist eine Hilfestellung) und besondere Vorkommnisse wie z.B. Änderungen der Aufgabenstellung, Arbeitsunterbrüche, organisatorische Probleme, Abweichungen von der Soll-Planung.“*

Das Arbeitsjournal zur IPA ist zwingend zu führen und den Experten und Fachvorgesetzten vorzulegen. Das Arbeitsjournal ist täglich sinngemäss und korrekt auszufüllen.

Das Arbeitsjournal dient der Nachvollziehbarkeit der von den Lernenden ausgeführten Arbeiten und wird als Teil der IPA in die Bewertung mit einbezogen.

Die Tätigkeit war für heute geplant.

Die Tätigkeit war für einen späteren Tag geplant.

Die Tätigkeit war für einen bereits vergangenen Tag geplant.

## Erster Tag: Montag, 14.08.2017

|  |  |  |  |
| --- | --- | --- | --- |
| Tätigkeiten | Beteiligte  Personen | SOLL  (Std) | IST  (Std) |
| Vorbereitung Initialisierungsphase | Nicole Sager | 4 | 4.5 |
| Arbeitsjournal | Nicole Sager | 0.5 | 0.5 |
| Analyse Ist Zustand | Nicole Sager | 1 | 0.25 |
| Persönliche Vorgehensziele | Nicole Sager | 0.5 | 0.25 |
| Projektziele | Nicole Sager | 0.5 | 0.25 |
| Anforderungen | Nicole Sager | 1 | 1 |
| Informationssicherheit und Datenschutz | Nicole Sager | 0.5 | 0.25 |
| Machbarkeitsprüfung | Nicole Sager | 0 | 1.5 |
| **Total**: | - | 8 | 8.5 |
| Tagesablauf |  |  |  |
| Heute habe ich das Projekt gestartet. Es waren vor allem Vorbereitungen für die Initialisierungsphase und bereits einzelne Bereiche der Initialisierungsphase. Unter anderem habe ich die Ordnerstruktur erstellt und die allgemeinen Informationen ausgefüllt & die bestätigte Aufgabenstellung übernommen.  Der vorbereitete Zeitplan habe ich noch ergänzt die Kapitel Detaillierte Projektmethode, Organisation der Probe IPA & Firmenstandards habe ich abgeschlossen.  Ich hatte sogar noch etwas Zeit, um erste Arbeiten an der Machbarkeitsprüfung umzusetzen. | | | |
| Hilfestellungen |  |  |  |
| www.hermes.admin.ch(14.08.2017)  Leitfaden\_Hermes5\_1.pdf  Hermes 5.1 – Projektmanagement für alle Projekte, REFERENZHANDBUCH (Eidgenössisches Finanzdepartement EFD, Informatiksteuerungsorgan des Bundes ISB, 2014)  https://de.wikipedia.org/wiki/SMART\_(Projektmanagement)(26.04.2017) | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Reflexion |  |  |  |
| **Was lief gut?**  Ich bin mit der Dokumentation gut vorangekommen und sogar leicht vor meinem Zeitplan.  Es war eine sehr interessante Erfahrung so oft um Hilfe gefragt zu werden, aber da ich nunmal die einzige mit etwas Erfahrung bin war das wohl vorhersehbar.  Auch ist es schön das sich meine bisherige Arbeit schlussendlich doch auszahlt.  **Was lief weniger gut?**  Da ich bereits wusste was auf mich zukommt eigentlich nichts.  **Meine Erkenntnisse von heute:**  Je öfter man eine IPA macht desto einfacher wird es. | | | |
| Nächste Schritte |  |  |  |
| Morgen möchte ich die Initialisierungsphase abschliessen und mit der Konzeptphase beginnen. | | | |

# Abschlussbericht

Der Abschlussbericht dient der Zusammenfassung des gesamten durchgeführten Projektes.

## Vergleich Ist/Soll

In der folgenden Auflistung wird überprüft, ob die gestellten Anforderungen erfüllt worden sind. Hierbei gilt: ✓ als erfüllt und 🗶 als nicht erfüllt oder nicht vollständig

* Die Chart Library soll folgende zwei Diagrammtypen zur Verfügung stellen:
  + Donut-Chart
  + Barchart

Wie zu sehen ist, wurden beinahe alle Punkte erfüllt. Einzig und allein das Barchart konnte aufgrund aufgetretener Probleme nicht vollständig abgeschlossen werden.

## X Mittelbedarf

Die Mittel, welche verwendet wurden sind nachfolgend aufgelistet. Anzumerken ist, das keinerlei zusätzliche Mittel besorgt werden mussten.

|  |  |
| --- | --- |
| Mittel | Bedarf für |
| Macbook Pro | Ist die von der Firma zur Verfügung gestellte Arbeitsgerät. |
| Draw.io | Zur professionellen Visualisierung von Diagrammen |
| Office 365 | Zur allgemeinen Bearbeitung der Dokumentation |
| Visual Sudio Code | Zur Bearbeitung und Erstellung des Souce Codes und dessen Testing. |
| Schreibzeug und Papier | Wurde für allgemeine Notizen, zum Strukturieren und Planen des Projektes verwendet |
| Harvest | Wird für die allgemeine Zeiterfassung verwendet |

Tabelle 4: Mittelbedarf

## Realisierungsbericht

### Ungeplante Ereignisse

Es gab zu Beginn einige Probleme mit dem Erstellen der Unit Tests. Es stellte sich heraus, dass einige Konfigurationen bezüglich des Source Codes falsch gewesen waren. Das Problem hierbei war, dass das Tool Mocha, welches für das Unit Testing verwendet wurde, direkt auf den Source Code zugegriffen hat. Da dieser jedoch in ECMAScript 6 geschrieben wurde, gab es den Fehler unexpected Token. Dadurch, dass Mocha auf die von Babel formatierte Version, welche in ECMAScript 5 geschrieben ist, zugriff konnte dieser Fehler behoben werden.

Zudem hatte die Projektleiterin mit dem ihr unbekannten ECMAScript 6 und dessen klassenorientierte Schreibweise keinerlei Erfahrung wodurch die Umsetzung erschwert wurde.

### Erkenntnisse der Realisierung

Anderweitige grössere Probleme Traten nicht auf. Jedoch war vor allem die Umsetzung des Testing und der Umgang mit dem für die Projektleiterin noch unbekannten ECMAScript 6, sehr zeitintensiv.

## Testbericht

Die Unit Tests verliefen allesamt gut. Es ist zwar nicht so, dass alles auf Anhieb funktioniert hat, jedoch wurde dies auch nicht erwartet. Durch die stetig kontrollierten Tests wurden alle Fehler und möglichen Fehler schnell erkannt und umgehend behoben.

Die Tests wurden solange wiederholt, bis alle Testdurchläufe erfolgreich verliefen. Dies wurde bei allen implementierten Testfällen erreicht.

## Fazit zum IPA (Projekt)

## Persönliches Fazit

Da es sich um ein persönliches Fazit handelt, wurde dieses Unterkapitel in der Ich-Person verfasst.

Ich persönlich bin sehr zufrieden mit dem Endergebnis meiner IPA.

## Schlussreflexion

Für mich persönlich war das Dokumentieren am schwierigsten. Da ich mir weder Wörter noch Buchstaben oder Zahlen bildlich vorstellen kann, war das formulieren der Sätze besonders schwer. Was ich jedoch bei der Erstellung der API Dokumentation feststellte, war, dass es für mich einfacher war auf Englisch zu schreiben als auf Deutsch.

Trotz dieses persönlichen Mankos habe ich alles versucht, um eine so aussagekräftige Dokumentation wie irgend möglich zu erstellen. Am Ende der Arbeit habe ich gelernt was es wirklich heisst, ein Projektleiter zu sein und wie schwer es sein kann, das grosse Ganze nicht aus den Augen zu verlieren. Dies ist ein Punkt, den ich besonders für die nächsten Arbeiten mitnehmen möchte. Dass es wichtig ist zuerst ein gutes Konzept und anschliessend eine solide Grundstruktur zu erstellen da man sich sonst in den Details verheddert.

# Unterschriften Teil 1

Die lernende Person bestätigt mit ihrer Unterschrift diese IPA aus Eigenleistung erbracht und nach den Vorgaben der Prüfungskommission Informatik Kanton Bern erstellt zu haben. Die Angaben im Arbeitsjournal entsprechen dem geleisteten Arbeitsaufwand.

Da die Phasenfreigabe während dieser ProbeIPA nicht durchgeführt werden gibt es entsprechend auch keine Unterschrift.

Teil 2: Projektdokumentation

# Initialisierung

„Die Initialisierung schafft eine definierte Ausgangslage für das Projekt/IPA und stellt sicher, dass die Projektziele mit PkOrg übereinstimmen. Die Projektgrundlagen und der Projektauftrag werden erarbeitet. Es wird ein Variantenentscheid getroffen oder es wird begründet warum es keinen gibt. Der Variantenentscheid wird schlussendlich vom Lernenden erarbeitet, geprüft und eingeführt.“[[3]](#footnote-3)

## Studie; Ist-Zustand

## X Meilensteine

Nach Hermes 5.1 IPA wurden bereits sieben Meilensteine vorgegeben. (Im Normalfall wären es acht jedoch wird in dieser Probe IPA die Einführungsphase nicht durchgeführt)

Diese wären wie Folgt:

1. Projekt Initialisierungs-auftrag
   1. ISDS Konzepterstellt
2. Projektfreigabe
   1. Testkonzept
3. Phasenfreigabe
   1. Test abgeschlossen
4. Abschluss des Projektes

### X SMART

Die persönlichen Ziele und die Projektziele wurden mithilfe des SMART-Prinzips erstellt.

|  |  |  |
| --- | --- | --- |
| **Buchstabe** | **Bedeutung** | **Beschreibung** |
| **S** | Spezifisch | Ziele müssen eindeutig definiert sein (nicht vage, sondern so präzise wie möglich). |
| **M** | Messbar | Ziele müssen messbar sein (Messbarkeitskriterien). |
| **A** | Ansprechend | Die Ziele müssen für die Person ansprechend bzw. erstrebenswert sein, zum Teil auch "achievable" oder "attainable", also erreichbar, dann steht das 'R' für 'relevant', ursprünglich "assignable", also - einem bestimmten Verantwortlichen - zuweisbar. |
| **R** | Realistisch | Das gesteckte Ziel muss möglich und realisierbar sein. |
| **T** | Terminiert | Das Ziel muss mit einem fixen Datum festgelegt werden können. |

Tabelle 8: SMART[[4]](#footnote-4)

### X Persönliche Vorgehensziele

In diesem Abschnitt werden die persönlichen Ziele der Absolventin definiert. Dabei ist zu beachten, dass die Ziele unabhängig vom Variantenentscheid sowie der Realisierung gestellt sind.

Die Ziele wurden nach dem SMART Prinzip, welches im vorherigen Kapitel erklärt wurde, erstellt.

1. Bis spätestens 14.08.2017 ist die Ist-Analyse abgeschlossen.
2. Bis spätestens 17.08.2017 ist der Realisierungsteil abgeschlossen und die Ergebnisse wurden dokumentiert.
3. Bis spätestens 18.05.2017 ist die Dokumentation, einschliesslich der zusätzlichen Dokumente wie Arbeitsjournal, Projektjournal, Handbuch etc. abgeschlossen.

### X Projektziele

In diesem Abschnitt werden die allgemeinen Ziele definiert. Dabei ist zu beachten, dass die Ziele unabhängig vom Variantenentscheid sowie der Realisierung gestellt sind.

Die Ziele wurden nach dem SMART Prinzip, welches im Kapitel 9.2 erklärt wurde, erstellt.

1. Das ISDS-Konzept muss bis spätestens dem 15.08.2017 fertiggestellt werden.
2. Das Test-Konzept muss bis spätestens dem 16.08.2017 fertiggestellt werden.
3. Bis spätestens 18.08.2017 ist das Testing abgeschlossen und die Ergebnisse wurden dokumentiert.
4. Bis spätestens 18.08.2017 ist die Dokumentation, einschliesslich der zusätzlichen Dokumente wie Arbeitsjournal abgeschlossen und die ausgedruckten Exemplare wurden abgegeben und eine digitale Version wurde via Mail verschickt.

## Anforderungen

### Funktionale Anforderungen

Funktionale Anforderungen beschreiben WAS das System bzw. Produkt leisten soll. Die funktionalen Anforderungen beschreiben somit die gewünschten Funktionalitäten des Systems, aber auch dessen Daten oder Verhalten.

* Es sollen min. 3 verschiedene Eingabeparameter vom Benutzerverlangt werden welche eine Auswirkung auf den Verlauf des Spiels haben
* Es sollen Pausen integriert werden welche den lese Fluss erleichtern sollen
* Das Spiel Soll mit Eingabe des eigenen Namens Personalisiert werden

### Nicht funktionale Anforderungen

Nicht funktionale Anforderungen beschreiben WIE GUT das System etwas leisten soll und decken im Prinzip die qualitativen Aspekte und Bedingungen des Systems oder Funktionalitäten ab (z.B. Erweiterbarkeit).

* Das Produkt soll mittels whitebox-testing getestet werden
  + Jede dieser Inputparameter soll via Unittesting getestet werden

## Informationssicherheit und Datenschutz (ISDS)

Da dieses Projekt keinerlei sensitive Daten enthält, wird von einer zusätzlichen Sicherung, wie z.B. mittels Eingaben von Anmeldeinformationen, zur Verwendung des Codes abgesehen.

Wie bereits im Kapitel 1 und im Kapitel 4 vermerkt erfolgt die Dokumentenablage während des normalen workflows auf der Lokalen Maschine eine weitere Sicherung erfolgt jeweils einmal täglich auf Github.

## Machbarkeitsprüfung Initialisierungsphase

Die folgende Machbarkeitsprüfung wurde von dem Dokument «14.\_checkliste\_machbarkeitspruefung» übernommen und teilweise angepasst. Dieses Dokument wird von der Webseite der Finanzdirektion des Kantons Bern [[5]](#footnote-5)zur Verfügung gestellt.

Die Machbarkeitsprüfung wird zuerst hier in der Initialisierungsphase umgesetzt und anschliessend noch einmal in der Konzeptphase aufgenommen. Die Machbarkeitsprüfung wird zur Sicherstellung eines realistischen Projektziels, sprich ob mit den zur Verfügung gestellten Ressourcen etc. das Projekt in der geforderten Qualität umgesetzt werden kann.

Durch die Durchführung einer Machbarkeitsprüfung wird das Risiko vermeidbarer Aufwände aufgrund Nichtdurchführbarkeit enorm gesenkt, da man bereits vor der eigentlichen Umsetzung mögliche Risikofaktoren erkennt und dadurch Projektanforderungen anpasst oder im schlimmsten Fall das Projekt abbrechen kann.

Für dieses Projekt werden folgende Aspekte geprüft und bewertet:

1. Technische und fachliche Machbarkeit
2. Ressourcen und Verfügbarkeit
3. Wirtschaftliche Machbarkeit
4. Zeitliche Umsetzung
5. **Technische und fachliche Machbarkeit**

Die Grundlage der technischen und fachlichen Machbarkeitsprüfung bilden die Anforderungen.

Sind die technischen Anforderungen überhaupt erfüllbar? Ja  Nein

*Die separaten Teile des Projektes wurden in verschiedenen Übungen schon verwendet bzw. umgesetzt. Zudem wurden ähnliche Projekte bereits von anderen Entwicklern in einer ähnlichen Form umgesetzt.*

Gibt es Alternativen und was sind die jeweiligen Voraussetzungen? Ja  Nein

*Da dies ein einfaches Übungsprojekt ist wird kein reelles Problem gelöst dadurch ist die Frage nach einer alternativen unsinnig.*

Sind die fachlichen Annahmen und Anforderungen realistisch? Ja  Nein

*Da ähnliche Projekte bereits von zahlreichen anderen Entwicklern durchgeführt wurden, wird angenommen, dass es realistisch ist.*

Sind alle fachlichen Anforderungen der Nutzer berücksichtigt? Ja  Nein

*Der jeweilige Benutzer muss lediglich python3 auf dem Rechner installiert haben und der jeweilige Benutzer muss englisch Verstehen.*

Wo liegen die grössten technischen beziehungsweise fachlichen Risiken?

*Die Entwicklerin besitzt erst sehr wenig Erfahrung mit Python vor allem was das Testing angeht.*

1. **Ressourcen und Verfügbarkeit**

Wie viele Ressourcen stehen für die Umsetzung des gesamten Projektes zur Verfügung?

*Es steht ein Mitarbeiter für die Dauer von 5 Arbeitstagen zu 100% zur Verfügung.*

Wo liegen die grössten Risiken bezüglich der Ressourcen?

*Das Ausfallen der Arbeitskraft durch Krankheit oder Unfall.*

Welche sind die wichtigsten unterstützenden Massnahmen?

*Die Arbeitskraft schaut eigenständig auf die eigene Gesundheit, erkennt allfällige Symptome und gibt allenfalls nach Feststellung von Krankheitssymptomen umgehend dem Auftraggeber und dem Hauptexperten Bescheid, zudem wird bei einem Krankheitsfall ein Arztzeugnis verlangt.*

Gibt es bei der Verfügbarkeit des Produktes allfällige Risiken? Ja  Nein

Gibt es bei der Verfügbarkeit des Systems auf dem das Produkt entwickelt wird allfällige Risiken? Ja  Nein

*Es besteht immer das Risiko von technischem Versagen, sprich, dass der Rechner auf dem entwickelt wird, kaputtgeht. Aus diesem Grund werden als Massnahme die Dokumente auf Dropbox gespeichert und täglich ein Backup auf einem USB Stick erstellt.*

1. **Wirtschaftliche Machbarkeit**

Wie fällt die Wirtschaftlichkeitsbetrachtung (Kosten/Nutzen) aus (unter welchen Annahmen)?

*Da der Zeitrahmen und damit die Kosten fix sind können die Kosten für das Projekt in diesem Rahmen nicht steigen.*

Welche finanziellen Ressourcen werden wann benötigt? Ist dies machbar? Was gilt es zu beachten?

*Während 10 Arbeitstagen wird die Projektleiterin 100% nur an diesem Projekt arbeiten. Diese Ressourcen sind fix und sind nicht verhandelbar.*

Welches sind die grössten Kosten- und Terminrisiken?

*Die grössten Kosten sind die zwei Arbeitswochen an Ressourcen, die 100% für das Projekt freigestellt werden müssen.*

*Das grösste Terminrisiko ist der Abgabetermin.*

1. **Ressourcen und Verfügbarkeit**

Wie viele Ressourcen stehen für die Umsetzung des gesamten Projektes zur Verfügung?

*Es steht ein Mitarbeiter für die Dauer von 10 Arbeitstagen zu 100% zur Verfügung.*

Wo liegen die grössten Risiken bezüglich der Ressourcen?

*Das Ausfallen der Arbeitskraft durch Krankheit oder Unfall.*

Welche sind die wichtigsten unterstützenden Massnahmen?

*Die Arbeitskraft schaut eigenständig auf die eigene Gesundheit, erkennt allfällige Symptome und gibt allenfalls nach Feststellung von Krankheitssymptomen umgehend dem Auftraggeber und dem Hauptexperten Bescheid, zudem wird bei einem Krankheitsfall ein Arztzeugnis verlangt.*

Gibt es bei der Verfügbarkeit des Produktes allfällige Risiken? Ja  Nein

Gibt es bei der Verfügbarkeit des Systems auf dem das Produkt entwickelt wird allfällige Risiken? Ja  Nein

*Es besteht immer das Risiko von technischem Versagen, sprich, dass der Rechner auf dem entwickelt wird, kaputtgeht. Aus diesem Grund werden als Massnahme die Dokumente auf Dropbox gespeichert und täglich ein Backup auf einem USB Stick erstellt.*

**Fazit:** Ist das Projektziel realistisch? Ja Nein

Wenn sich Zweifel an der Machbarkeit ergeben gilt es, zwischen den drei folgenden Möglichkeiten zu entscheiden:

* Durchführung einer detaillierten Machbarkeitsstudie
* Projektabbruch
* Projektdurchführung trotz hohen Risikos

**Empfehlung**

Es wird empfohlen das Projekt durchzuführen.

## Risikoanalyse

Anschliessend ist eine Tabelle bezüglich der Risiken abgebildet, welche im schlimmsten Fall auftreten könnten.

Diese Risiken werden allgemein gehalten und sind nicht auf die spezifische Umsetzung, sprich das Programmieren, bezogen.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Risikobeschreibung | Auswirkung | Vor Massnahme | | | | Massnahmen/Erklärung | Nach Massnahme | | | |
| W | S | Risiko | Handlungsweise | **W** | **S** | **Risiko** | **Handlungsweise** |
| Verpassen des Abgabetermins | -Bewertung der Arbeit mit 0.5 Note Abzug  -Führt zum Nichtbestehen der Arbeit. | W3 | S4 | hoch | Risikominderung | -Erstellung und Einhaltung eines realistischen Zeitplans  -Einrichtung einer Terminerinnerung im Outlook  -Fertigstellungstermin vor dem Abgabetermin setzen. | W2 | S4 | mittel | Risikoakzeptanz |
| Umfang und Komplexität | -Die Arbeit kann nicht fertiggestellt werden. | W5 | S3 | hoch | Risikominderung | -Erstellung und Einhaltung eines realistischen Zeitplans  -Möglichst früher Fertigstellungstermin | W2 | S3 | gering | Risikoakzeptanz |
| Krankheitsfall | -Die Arbeit gerät massiv in den Rückstand.  -Verlorene Zeit muss nachgeholt werden oder die Arbeit kann nicht fertiggestellt werden. | W3 | S2 | mittel | Risikominderung | -Symptome frühzeitig erkennen.  -Bei eintreten umgehend ein Arztzeugnis besorgen, den Experten und den Auftraggeber informieren. | W2 | S2 | gering | Risikoakzeptanz |
| Begehung formaler Fehler | Abwertung der Endnote | W3 | S2 | mittel | Risikominderung | -Review der Arbeit durch Dritten mit Kenntnis der Formalitäten (z.B. Kollegen Familienmitglieder, etc.) | W2 | S2 | gering | Risikoakzeptanz |
| Verlust der Arbeit | -Die Arbeit kann nicht fertiggestellt werden.  -Teilbereiche müssen nochmals gemacht werden. | W3 | S3 | mittel | Risikominderung | -Versionisierung der Arbeit  -Regelmässiges Erstellen von Backups (täglich). | W1 | S3 | gering | Risikoakzeptanz |
| Ausfallen der Infrastruktur | -Blockiert die Arbeiten. Somit gerät die Arbeit in den Rückstand | W3 | S2 | gering | Risikominderung | -Tägliche Sicherung aller Dokumente erstellen  -Ersatzgeräte bereithalten  -Infrastruktur genügend testen | W1 | S2 | gering | Risikoakzeptanz |
| Höhere Gewalt | Die Arbeit kann im schlimmsten Fall nicht fertig gestellt werden. | W1 | S3 | gering | Risikoakzeptanz | -Massnahmen können bei Höherer Gewalt nicht getroffen werden.  -Da die Eintrittswahrscheinlichkeit jedoch sehr gering ist, wird das Risiko akzeptiert |  | | | |

Tabelle 9: Risikoanalyse

|  |
| --- |
| **Schadensausmaß:**  S1 = führt zu keiner Abwertung  S2 = geringe Abwertung bis 1.0 Notenpunkte  S3 = hohe Abwertung über 1,0 Notenpunkte  S4 = führt zu Nichtbestehen    **Eintrittswahrscheinlichkeit:**  W1 = unvorstellbar  W2 = unwahrscheinlich  W3 = eher vorstellbar  W4 = vorstellbar  W5 = Eintreffen hoch |

## Risikographen

Folgend sind die beiden Risikographen abgebildet. Die Nummern stehen in Bezug zu denen in der vorherigen Risikoanalyse (Kapitel 9.6).

### Risikograph vor den Massnahmen

|  |  |
| --- | --- |
| Nr. | **Risikobeschreibung** |
| 1 | Verpassen des Abgabetermins |
| 2 | Umfang und Komplexität |
| 3 | Krankheitsfall |
| 4 | Begehung formaler Fehler |
| 5 | Verlust der Arbeit |
| 6 | Ausfallen der Infrastruktur |
| 7 | Höhere Gewalt |

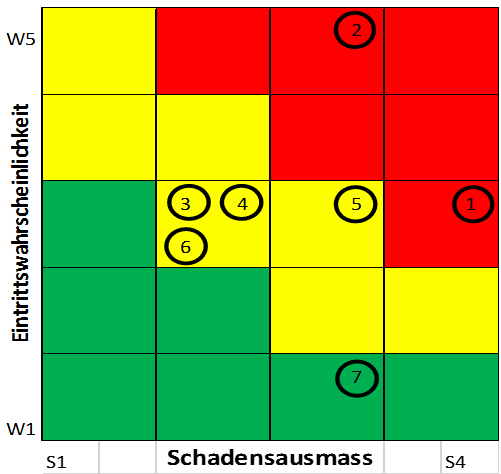


Abbildung 18: Risikograph

Tabelle 10: Risikobeschreibung

### Risikograph nach den Massnahmen

|  |  |
| --- | --- |
| Nr. | **Risikobeschreibung** |
| 1 | Verpassen des Abgabetermins |
| 2 | Umfang und Komplexität |
| 3 | Krankheitsfall |
| 4 | Begehung formaler Fehler |
| 5 | Verlust der Arbeit |
| 6 | Ausfallen der Infrastruktur |
| 7 | Höhere Gewalt |

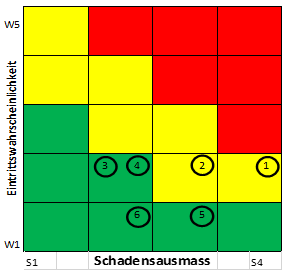


Abbildung 19: Risikograph

Tabelle 11: Risikobeschreibung

### Kurze Stellungnahmen zu den Risiken

Wie hier zu sehen ist, sind durch die getroffenen Massnahmen alle Risiken aus dem roten-Bereich herausgefallen. Während der Arbeit werden die gesamten genannten Massnahmen ausgeführt und das verbleibende Risiko wird akzeptiert.

## Lösungen suchen

Die Lösungssuche ist ein strukturierter und kreativer Prozess, in welchem nach möglichen Ansätzen gesucht wird, welche die Umsetzung der Systemziele unterstützen.

Der Entscheid über den Lösungsvorschlag (Variantenentscheid) schliesst die Lösungssuche ab.

### Aufsetzen der Chart Library

Bereits vor dem offiziellen Start der IPA wurde von der Projektleiterin entschieden, dass für das Projekt ein sogenanntes Template verwendet werden soll.

Da es eine hohe Anzahl an möglichen Templates gibt, wird dazu im nachfolgenden Kapitel ein Variantenentscheid erstellt.

### Aufruf

Als Lösungsansätze bezüglich der Übermittlung der Parameter stehen grundsätzlich folgende Möglichkeiten zur Verfügung.

* Ein simpler Funktionsaufruf, zu welchem ein Objekt als Parameter übergeben wird (Beispiel: Highcharts)
* Method chaining (Beispiel: D3.js, jQuery)

Die zwei oben genannten Möglichkeiten werden in der nachfolgenden Entscheidung berücksichtigt.

Aufgrund der Verwendung der D3.js Library und der dort eingesetzten Method chaining Variante wurde die Entscheidung zugunsten des Method chaining gefällt.

**Begründung: einheitliches Softwaremodell.**

## Varianten

Bereits vor dem offiziellen Start der IPA wurde von der Projektleiterin entschieden, dass für das Projekt ein sogenanntes Template verwendet werden soll.

Anbei eine kurze Auflistung, welche Eigenschaften dieses Template besitzen soll/muss.

* Das Testing mit Unit Tests soll möglich sein.
* Die Library soll in jedem momentanen Projekt integrierbar sein.
* Die Verwendung der Templates soll gut dokumentiert sein
* Es sollen keine überflüssigen Libraries bereits integriert sein
* ECMAScript Version: min ES5 oder ES6

Da es verschiedenste Templates gibt, werden in folgenden drei Unterkapiteln (9.9.1, 9.9.2 und 9.9.3) drei mögliche Varianten miteinander verglichen.

### Gewichtung der Kriterien

Die Kriterien werden mit einer Skala von 0-3 gewichtet. Dabei ist 0 unwichtig und 3 ein Muss Kriterium. Bei dem Kapitel 9.11, dem Variantenentscheid werden diese Punkte noch einmal aufgelistet. für die Berechnung jedoch wird 3 als 50%, 2 als 15% und 1 als 10% dargestellt.

|  |  |  |
| --- | --- | --- |
| **Nr.** | **Kriterium** | **Gewichtung** |
| **1** | Das Testing mit Unit Tests soll möglich sein. | 3 |
| **2** | Die Library soll in jedem momentanen Projekt integrierbar sein. | 3 |
| **3** | Die Verwendung der Templates soll gut dokumentiert sein | 2 |
| **4** | Es sollen keine überflüssigen Librarys bereits integriert sein | 2 |
| **5** | ECMAScript Version: ES5 oder ES6 | 1 |

Tabelle 12: Kriterien

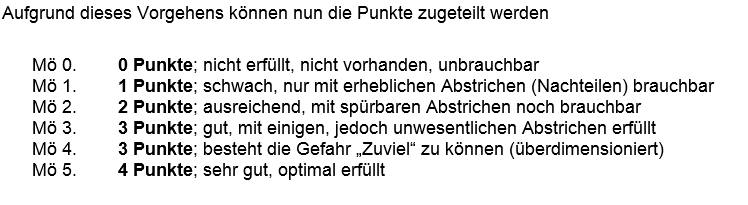
Aufgrund dieses Vorgehens können nun die Punkte zugeteilt werden.

Abbildung 13: Bewertungsmöglichkeiten

### React Redux Starter Kit

* react
* redux
* react-router
* webpack
* babel
* express
* karma
* eslint

|  |  |  |
| --- | --- | --- |
| **Kriterium** | **Beschreibung** | **Erfüllung** |
| **1** | Karma wird für das Testing verwendet | 4 |
| **2** | Babel wird zum Kompilieren verwendet | 4 |
| **3** | Sehr gut und sehr detailliert | 4 |
| **4** | React und redux sind eigentlich sehr nützlich jedoch für die Umsetzung nicht unbedingt nötig. | 3 |
| **5** | Ist gegeben | 4 |

Tabelle 13: React Redux Starter Kit

### ES6 Karma Jasmine Webpack Boilerplate

* Webpack
* Karma
  + Mocha, chai, sinon-chai, und chai-as-promised
  + Rewire enabled
  + PhantomJS und Chrome
  + Hochladen zu coveralls (https://github.com/caitp/karma-coveralls)
* Babel (^6.3.0)
  + babel-plugin-transform-runtime
  + babel-istanbul
* ESLint
  + Benutzt airbnb/base mit einigen zusätzlichen nützlichen Regeln

|  |  |  |
| --- | --- | --- |
| **Kriterium** | **Beschreibung** | **Erfüllung** |
| **1** | Karma wird für das Testing verwendet | 4 |
| **2** | Babel wird zum Kompilieren verwendet | 4 |
| **3** | Eine Dokumentation ist vorhanden jedoch nicht in dem gewünschten Detaillierungsgrad. | 2 |
| **4** | Nur das nötigste. | 4 |
| **5** | Ist gegeben | 4 |

Tabelle 14: ES6 Karma Jasmine Webpack Boilerplate

### Webpack library starter

* Webpack 2 basiert.
* ES6.
* Exports in einem umd format damit wird gegeben, dass die Library überall funktioniert.
* ES6 test setup mit Mocha und Chai.
* Linting mit ESLint

|  |  |  |
| --- | --- | --- |
| **Kriterium** | **Beschreibung** | **Erfüllung** |
| **1** | Chai und Mocha wird für das Testing verwendet | 4 |
| **2** | Babel wird zum Kompilieren verwendet | 4 |
| **3** | Die Dokumentation ist sehr gut, es besteht sogar eine eigene Webseite. | 4 |
| **4** | Nur das Nötigste. | 4 |
| **5** | Ist gegeben | 4 |

Tabelle 15: Webpack library starter

## Variantenentscheid

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Kriterien** | **Gewicht** | **React Redux Starter Kit** | | **ES6 Karma Jasmine Webpack Boilerplate** | | **Webpack library starter** | |
| Bewertung | Gesamt | Bewertung | Gesamt | Bewertung | Gesamt |
| **Das Testing mit Unit Tests soll möglich sein.** | 25% | 4 | 1 | 4 | 1 | 4 | 1 |
| **Die Library soll in jedem momentanen Projekt integrierbar sein.** | 25% | 4 | 1 | 4 | 1 | 4 | 1 |
| **Die Verwendung des Templates soll gut dokumentiert sein** | 15% | 4 | 0.6 | 2 | 0.3 | 4 | 0.6 |
| **Es sollen keine überflüssigen Librarys bereits integriert sein** | 15% | 3 | 0.45 | 4 | 0.45 | 4 | 0.6 |
| **ECMAScript Version: ES5 oder ES6** | 10% | 4 | 0.4 | 4 | 0.4 | 4 | 0.4 |
| **Webpack Version: Webpack 2** | 10% | 2 | 0.2 | 2 | 0.2 | 4 | 0.4 |
| **Total** | 100% | 21 | 3.65 | 20 | 3.35 | 24 | 4 |

Tabelle 16: Variantenentscheid

### Begründung

Wie der durchgeführte Variantenentscheid zeigt, deckt Variante 3 «Webpack library starter» die Anforderungen in optimalster Weise ab. Deswegen wird diese in der Umsetzung des Projektes angewendet.

# Konzept

In der Konzepterarbeitung werden die Grundlagen für die Realisierung und Einführung eines Informatiksystems entwickelt.

Das Konzept wird schrittweise mit folgender Gliederung entwickelt.

## Allgemein

Nachfolgend werden nochmals die allgemeinen Voraussetzungen, Anforderungen und die daraus entstehenden Konzepte einleitend erläutert.

### Grundlagen

In der Firma enersis werden zur Visualisierung von diversen Energiedaten regelmässig Charts mit Hilfe der Library D3.js (Data Driven Documents) erstellt. Dieser Code wird für jedes Projekt neu geschrieben und kostet die Entwicklung wiederholt vermeidbaren Aufwand. Aus diesem Grund soll eine Chart Library erstellt werden.

**Enersis Status-Quo**

* **Jedes Chart wird spezifisch für ein Projekt erstellt**
* **Für verschiedene Charts werden verschiedene Librarys verwendet**

**IPA-Ansatz**

* **Alle Charts sollen in einer Library gespeichert werden**
* **Die Umgesetzten Charts sollen in D3.js umgesetzt werden**
* **Basic Codeteile werden nicht mehr redundant sein**

### ISDS

ISDS wurde schon unter Punkt 9.4 Informationssicherheit und Datenschutz (ISDS) beschrieben. An dieser Stelle erfolgt nochmals die Aufzählung der wichtigsten Punkte.

## Architektur

### C:\Users\nicol\AppData\Local\Microsoft\Windows\INetCacheContent.Word\EdrawMax_2017-05-04_21-18-42.pngVom Source Code zur fertigen Library

Tabelle 17: Vom Source Code zur fertigen Library

Die Chart Library wird in ECMAScript 6 (ES6) umgesetzt. Da ES6 zum jetzigen Zeitpunkt nicht mit allen Browsern kompatibel ist wird Babel verwendet. Babel konvertiert die ES6 Files in ES5 Files und macht diese somit mit den momentan verwendeten Browsern kompatibel.

Um schlussendlich nur ein File einbinden zu müssen, werden die Source Files, mit all ihren Abhängigkeiten, mithilfe von Webpack in 2 Versionen eines Files gespeichert einer library.js Version und einer library.min.js Version.

### Einbindung der Charts auf einer Webseite

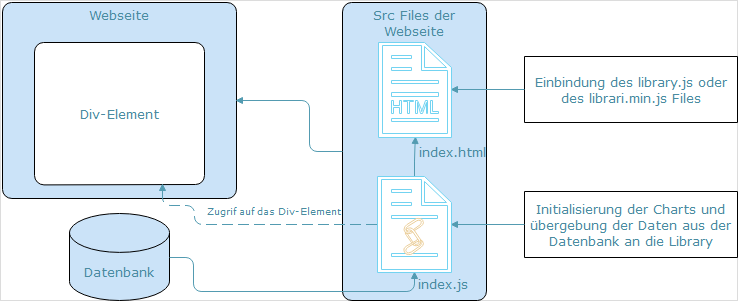


Abbildung 14: Einbindung der Charts auf einer Webseite

Die obige Abbildung zeigt eine grobe Struktur einer allgemeinen Webseite. Diese theoretische Webseite besteht aus einem index.html File, in welchem ein DIV-Element eingetragen ist und aus einem index.js File, welches eine Verbindung auf eine Datenbank besitzt.

Damit die fertige Library benutzt werden kann, muss als erstes entweder das Library.js File oder das Library.min.js File eingebunden werden. Anschliessend wird in dem index.js File die Library aufgerufen, das DIV-Element und die Daten übergeben. Daraufhin wird die Library ein Chart mit den mitgelieferten Daten erstellen und einen Zugriff auf das DIV-Element ausführen, in welchem das fertig gerenderte Chart schlussendlich angezeigt wird.

### Klassendiagramm

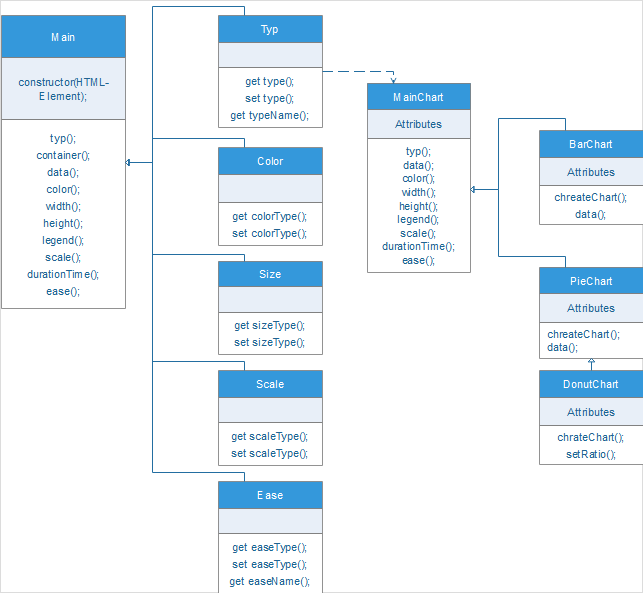


Abbildung 15: Klassendiagramm

Verschiedene Grundbausteine werden benötigt um den Code so modular, erweiterbar und nichtredundant wie möglich zu machen.

Diese sehr grob aufgelisteten Teile wären:

* allgemeine Zuweisung der Properties
* die Überprüfung der übermittelten Properties
* die Weiterleitung zum Erstellen des korrekten Charts
* die Erstellung des Charts

Die Library wird über die Main Klasse aufgerufen. Dabei muss die ID des DIVs, in welchem das Chart schlussendlich angezeigt werden soll, im String Format als Parameter übergeben werden.

Nach diesem Aufruf können die weiteren Properties via Method chaining hinzugefügt werden. Dabei müssen mindestens der Typ des Charts und dessen Daten in der korrekten Datenstruktur übergeben werden. In der Main Klasse werden zudem die ersten Validierungen durchgeführt. Die übergebenen Properties werden dabei auf ihren geforderten Datentyp überprüft. Wie in der zweiten Spalte des Diagramms zu sehen ist, wurden hierbei einzelne Funktionen in eigene Klassen ausgelagert. Bei diesen Klassen geschehen komplexere Validierungen. Dabei ist die Klasse Typ besonders zu beachten, da diese die Weiterleitung zum MainChart initialisiert.

In der MainChart Klasse werden nun die allgemeinen Funktionen mit den vorher definierten Eigenschaften ausgeführt. Dabei wurden das Barchart und das Donutchart in separate Klassen unterteilt in welchem die Charts schlussendlich erstellt werden.

Obwohl es nicht explizit zur detaillierten Aufgabenstellung gehört, wurde das Piechart zusätzlich zum Donutchart ergänzt. Die beiden Charts unterscheiden sich nur durch die Grösse des inneren Radius.

### C:\Users\nicol\AppData\Local\Microsoft\Windows\INetCacheContent.Word\EdrawMax_2017-05-04_22-32-09.pngDatenstruktur

Abbildung 23: Datenstruktur

Die allgemeine Datenstruktur besteht aus mehreren sogenannten Serien bzw. Objekten. In diesen Objekten werden als Eigenschaften mindestens einen Titel und ein Value angegeben.

Der Titel wird für die eventuell angezeigte Legende benötigt.

Die Values werden zur Erstellung des Charts benötigt, dabei ist zu beachten, dass der Value ein Zahlenwert sein muss.

Ein Beispiel wäre:

[{value: 9, title: ‘First Part’}, {value: 3, title: ‘Second Part’},

{value: 2, title: ‘Third Part’}]

## Anwendungsfall

Bisher wurden die Standard Charts spezifisch für jeden Kunden erstellt. Mit einer Chart Library werden die Standard Charts nicht mehr spezifisch für den Kunden entwickelt. Dadurch wird das sogenannte «look and feel» der Projekte einheitlich und die Corporate Identity wird ausgeprägter.

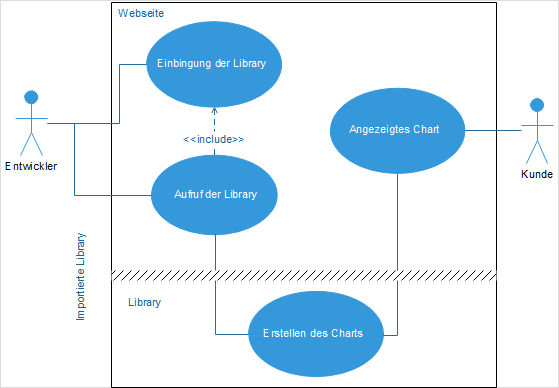


Abbildung 17: Anwendungsfälle

Bei diesem Anwendungsfall geht es im Wesentlichen darum wie der Entwickler ein Chart erstellt. Dafür muss er als erstes die Library in seinem Sourcecode einbinden. Anschliessend muss er die Library aufrufen und die korrekten Parameter bzw. Properties übergeben.

Daraufhin wird in der Chart Library das Chart erstellt. Das erstellte Chart wird nun auf der Webseite angezeigt, auf welcher der Kunde dieses sehen kann.

## Machbarkeitsprüfung Konzeptphase

1. Wirtschaftliche Machbarkeit
2. Technische Machbarkeit
3. Zeitliche Umsetzung
4. Ressourcen und Verfügbarkeit

**Empfehlung**

Im Vergleich zu der Initialisierungsphase hat sich an der Machbarkeit des Projektes nichts geändert. Aus diesem Grund wird weiterhin empfohlen das Projekt durchzuführen.

## Testkonzept

### Grundlagen

Das Testing besteht im Allgemeinen aus zwei Bereichen. Dem Frontend und dem Backend.

Das Frontend wird manuell kontrolliert und für das Backend wird Unit Testing verwendet.

### Test Szenario

Die Tests werden während des Programmierens gleichzeitig mit dem eigentlichen Code implementiert.

### Testrahmen

Die Tests werden lokal auf dem Arbeitsrechner der enersis ausgeführt.

Es bedarf keiner speziell gesicherten Testumgebung, weil die übermittelten Daten zum momentanen Zeitpunkt nicht produktionsrelevant sind.

### Testvorgehen

Die Unit Tests werden mithilfe von Mocca und Chai implementiert simultan mit dem eigentlichen Code implementiert.

Die Durchführung der Testsfälle geschieht automatisch. Dies geschieht durch die Eingabe des Befehls: npm run test.

### Testmethode

**Unit-Test**[[6]](#footnote-6): Mittels Unit-Test (auch Modul- oder Komponententest genannt) werden einzelne Komponenten einer Anwendung geprüft. Unit-Tests werden oft im Rahmen der Softwareentwicklung vorgenommen.

### Testfälle

|  |  |  |
| --- | --- | --- |
| **Nr.** | **Aktion** | **Erwartetes Ergebnis** |
| **1TF** | Aufruf des Barcharts. Nur mit Angabe des Typs. | Fehlermeldung |
| **2TF** | Aufruf des Barcharts. Mit Angabe des Typs und Daten mit falscher Datenstruktur. | Fehlermeldung |
| **3TF** | Aufruf des Barcharts. Mit Angabe des Typs und Daten mit korrekter Datenstruktur. | Anzeigen eines Barcharts |
| **4TF** | Aufruf eines Charts. Mit Angabe des eines falschen Typs. | Fehlermeldung |
| **5TF** | Properties des Titels : «Test Titel» (String). | Titel wurde im entsprechenden Property gespeichert. |
| **6TF** | Properties des Titels : 9 (Zahlenwert) | Fehlermeldung |
| **7TF** | Daten mit korrekter Datenstruktur | Daten wurde im entsprechenden Property gespeichert. |
| **8 TF** | Daten mit falscher Datenstruktur | Fehlermeldung |
| **9 TF** | Als Zielelement wird «#DIVId» (String) übergeben. | Wenn ein entsprechendes HTML Element gefunden wurde, wird das Element in dem entsprechenden Property gespeichert. |
| **10 TF** | Als Zielelement wird «.DIVClass» (String) übergeben. | Fehlermeldung |
| **11 TF** | Als Zielelement wird «#DIVId» (String) übergeben. | Wenn kein entsprechendes HTML Element gefunden wurde, bekommt man eine Fehlermeldung. |
| **12 TF** | Als Zielelement wird der Zahlenwert 9 übergeben. | Fehlermeldung |
| **13 TF** | Propertie des Color: [«#fff»] (String in einem Array) | Color wurde im entsprechenden Property gespeichert. |
| **14 TF** | Propertie des Color: «#fff» (String) | Fehlermeldung |
| **15 TF** | Propertie des Color: [«#fff», «#000», «#abcabc»](String in einem Array) | Color wurde im entsprechenden Property gespeichert. |
| **16 TF** | Propertie des height: 10 (Zahlenwert) | height wurde im entsprechenden Property gespeichert. |
| **17 TF** | Propertie des height: «test» (String) | Fehlermeldung |
| **18 TF** | Propertie des width: 10 (Zahlenwert) | width wurde im entsprechenden Property gespeichert. |
| **19 TF** | Propertie des width: «test» (String) | Fehlermeldung |
| **20 TF** | Propertie des TransitionTime: 100 (Zahlenwert) | TransitionTime wurde im entsprechenden Property gespeichert. |
| **21 TF** | Propertie des TransitionTime: «test» (String) | Fehlermeldung |
| **22 TF** | Propertie des TransitionType: sinus | TransitionType wurde im entsprechenden Property gespeichert. |
| **23 TF** | Propertie des TransitionTime: «test» (String) | Fehlermeldung |
| **24 TF** | Propertie des Legend: false | Legend wurde im entsprechenden Property gespeichert. |
| **25 TF** | Propertie des Legend: «test» (String) | Fehlermeldung |
| **26 TF** | Propertie des Scale: 10 (Zahlenwert) | Scale wurde im entsprechenden Property gespeichert. |
| **27 TF** | Propertie des Scale: «test» (String) | Fehlermeldung |
| **28 TF** | Propertie des Radius: false(Boolean) | Legend wurde im entsprechenden Property gespeichert. |
| **29 TF** | Propertie des Legend: «test» (String) | Fehlermeldung |

Tabelle 18: Testfälle

### Testziele

Alle Eingabeparameter wurden getestet. Und es ist sichergestellt, dass Veränderungen während der Laufzeit möglich sind.

# Realisierung

Die Phase Realisierung dient zur Erstellung des Systems und schafft die Voraussetzungen für die nachfolgende API.

## Template

In diesem Projekt wird das Wort Template zur Repräsentierung einer von einem anderen Entwickler aufgesetzten Projektstruktur verwendet.

Diese Projektstruktur enthält bereits die benötigten Libraries und mit diesem Template wird sichergestellt, dass mit diesen allgemeinen Arbeiten keine Zeit verschwendet wird. Das Projektstruktur – Template stellt ausserdem sicher, dass alle erforderlichen Dateien und Programme für die Kompilierung und zur Laufzeit bereitstehen.

In den folgenden Unterkapiteln wird auf die einzelnen Komponenten des Templates eingegangen.

### C:\Users\nicol\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Ecma_RVB-003.jpgECMAScript 6

ECMAScript 6 ist die neuste offizielle Version von ECMAScript[[7]](#footnote-7), der offizielle Name lautet ECMAScript 2015 und kurz ES6. ECMAScript ist ein Standard der Programmiersprache JavaScript.

Abbildung 25: ecma international

Andere Standards wären z.B. JScript oder ActionScript.

Entwickler benutzen ECMAScript vor allem zur clientseitigen Webentwicklung, jedoch wird es auch immer häufiger zur serverseitigen Programmierung mit Node.js verwendet.

**Neue Features in ES 6**

In der neuen ECMAScript Version wurden einige neue Features hinzugefügt, unter anderem wären das:

* Klassen
  + Getter/ Setter Methoden
* Module
* Generatoren
* Typed Arrays
* Destructuring Assignments

Die neue ECMAScript Version wird jedoch noch nicht von allen Browsern fehlerfrei unterstützt. Deswegen wird für die Verwendung von ECMAScript 6 Code ein Compiler benötigt, welcher den ES6 Code in ES5 Code kompiliert.

### C:\Users\nicol\AppData\Local\Microsoft\Windows\INetCacheContent.Word\babel-1.pngBabel

In diesem Projekt wird Babel[[8]](#footnote-8) als Compiler verwendet. Babel kompiliert ECMAScript 6 in ECMAScript 5.

In der nachfolgenden Abbildung sehen sie ein Beispiel, welches von der offiziellen Seite übernommen wurde.

Abbildung 26: BABEL



Abbildung 20: Babel Beispiel

Wie anhand des Beispiels erkennbar ist, transformiert Babel nur die Syntax, währendem die eigentliche Funktionalität bestehen bleibt.

Ausserdem wird die kompilierte Syntax gemapt, so dass bei der Fehlersuche ersichtlich ist, in welcher Zeile sich der Originalcode befindet.

### C:\Users\nicol\AppData\Local\Microsoft\Windows\INetCacheContent.Word\EsLint.pngESLint

ESLint[[9]](#footnote-9) ist ein Tool, welches bei der allgemeinen Code Formatierung und Problemsuche hilft. EsLint besitzt diverse Regelungen, welche alle separat an- und ausgeschaltet werden können. Diese Hinweise tragen zu einem besseren Code und damit zur Verhinderung von Fehlern bei.

Nach der Definition von Wikipedia[[10]](#footnote-10): «Lint ist ein Computerprogramm zur statischen Code-Analyse der Quelltexte von Computerprogrammen. Das Verb linten für   
das Durchführen der statischen Code-Analyse hat sich davon   
unabhängig als allgemeine Bezeichnung etabliert.»

Abbildung 28: ESLint

Hier noch ein paar übersetzte Angaben von der offiziellen Webseite bezüglich der Benutzung von ESLint[[11]](#footnote-11).

* Jede Regel kann: abgestellt, nur eine Warnmeldung oder eine Fehlermeldung ausgeben.
* Zusätzliche Regeln können während der Laufzeit hinzugefügt werden.
* ESLint fördert keinen bestimmten Codierungsstil

### Unit Testing

Nach der Definition von Wikipedia wird ein Modultest (auch von englisch unit test als Unittest oder als Komponententest bezeichnet) in der Softwareentwicklung angewendet, um die funktionalen Einzelteile (Units) von Computerprogrammen zu testen, d.h., sie auf korrekte Funktionalität zu prüfen.

#### C:\Users\nicol\AppData\Local\Microsoft\Windows\INetCacheContent.Word\mocha-chaijs.pngMocha und Chai

Mocha and Chai[[12]](#footnote-12) sind zwei der am häufigsten verwendeten Libraries in Bezug auf JavaScript basiertem Unit Testing.

Hierbei wird mit Mocha die eigentliche Testing Umgebung hinzugefügt und Chai enthält einige hilfreiche Funktionen, welche das Schreiben der Tests und dessen Auswertungen wesentlich einfacher machen. Die geschriebenen Tests können sowohl auf Node.js als auch im Browser ausgeführt werden.

Abbildung 29: mocha und chai

Ein Beispiel wie ein realisierter Chai Test aussehen könnte.

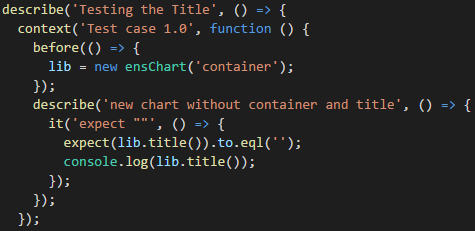
Wie auf der oberen Abbildung zu sehen ist, besteht ein Test aus 5 Hauptbausteinen, wobei einer davon optional ist.

Abbildung 30: Testing des Titels

|  |  |
| --- | --- |
| Describe | Wird zur Beschriftung des durchgeführten Tests verwendet. |
| Context (optional) | Context ist eine weitere Beschriftungsmöglichkeit und wird hier zur Strukturierung des Codes verwendet. Durch den Einsatz der context Funktion können mehrere Tests in einem describe ausgeführt werden. |
| Before | Wird ausgeführt bevor die eigentliche Überprüfung stattfindet. |
| It | Beschreibt was erwartet wird. |
| expect | Hier findet der eigentliche Test statt. Wobei erwartet wird, dass die ausgeführte Funktion true zurück liefert. |

Tabelle 19: Hauptbausteine von Chai[[13]](#footnote-13)

Hierbei ist zu beachten, dass der Vergleich der beiden Values, in diesem Fall ob der Titel wirklich den Standardwert zurückgibt, mit method chaining ungesetzt wurde. Die Funktion Namen von Chai wurden bewusst gewählt und die Leserlichkeit des Codes zu verbessern.

Die mit Chai geschriebenen Tests wurden mithilfe von Mocha getestet. Weil das Testing im Browser durchgeführt wurde, musste eine zusätzliche Dashboard Seite (test.html) umgesetzt werden.

### C:\Users\nicol\AppData\Local\Microsoft\Windows\INetCacheContent.Word\EdrawMax_2017-05-04_21-18-42.pngVom Source Code zur fertigen Library

Tabelle 20: Vom Source Code zur fertigen Library

Die Chart Library wird in ECMAScript 6 (ES6) umgesetzt. Da ES6 zum jetzigen Zeitpunkt nicht mit allen Browsern kompatibel ist wird Babel verwendet. Babel konvertiert die ES6 Files in ES5 Files und macht diese somit mit den momentan verwendeten Browsern kompatibel.

Um schlussendlich nur ein File einbinden zu müssen, werden die Source Files, mit all ihren Abhängigkeiten, mithilfe von Webpack in 2 Versionen eines Files gespeichert.

Einer library.js Version und einer library.min.js Version.

#### Webpack

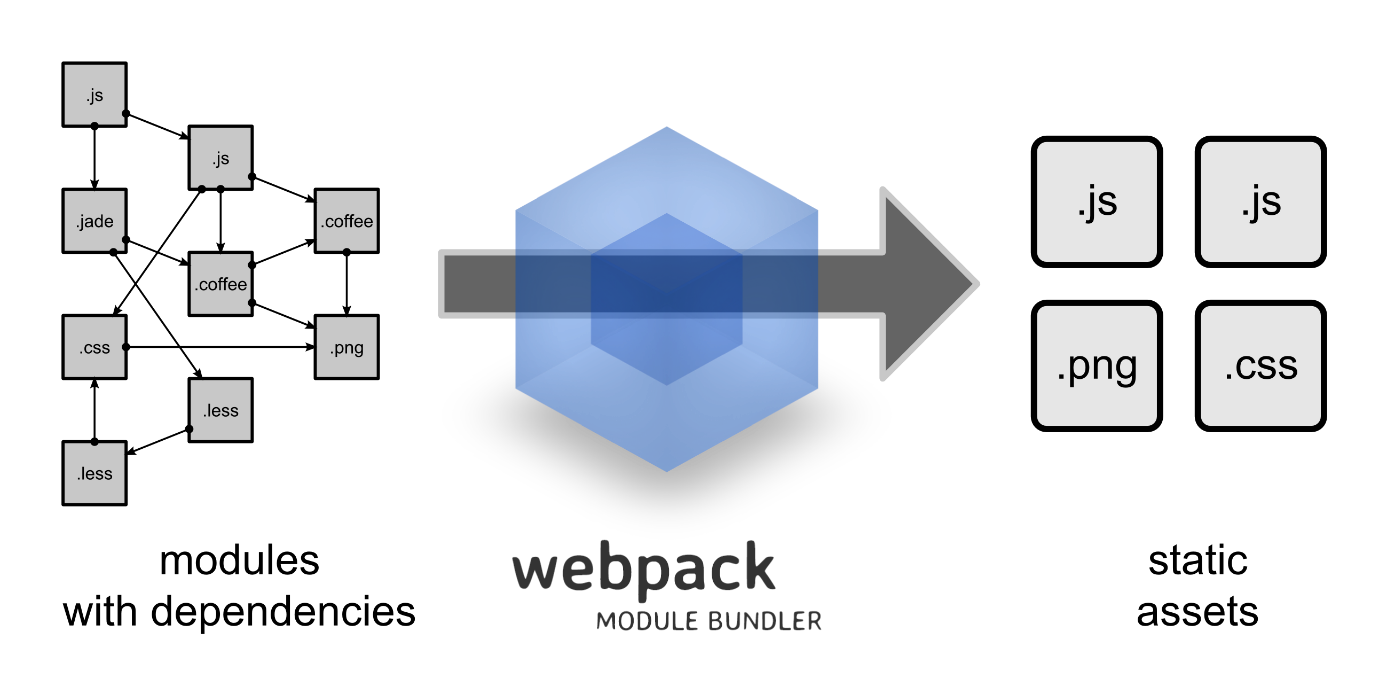


Tabelle 21: Was ist webpack[[14]](#footnote-14)

In normalen Projekten muss jedes einzelne js File einzeln nacheinander in das gewünschte File eingebunden werden. Da dies sehr umständlich ist und einige unnötige Fehler durch z.B. Einlesen der Files in der falschen Reihenfolge verursachen kann, werden sogenannte module bundeler, wie in diesem Projekt webpack, verwendet.

Webpack bundelt alle Files mit dessen Abhängigkeiten in ein paar wenige Files. Dadurch müssen deutlich weniger Files eingelesen werden. Dies macht die Verwendung wesentlich angenehmer, simpler und fehlerfreier. Der grösste Nachteil ist, dass sehr viele unnötige assets mitgeladen werden, die bei einem Projekt möglicherweise bereits eingebunden wurden.

## Klassendiagramm

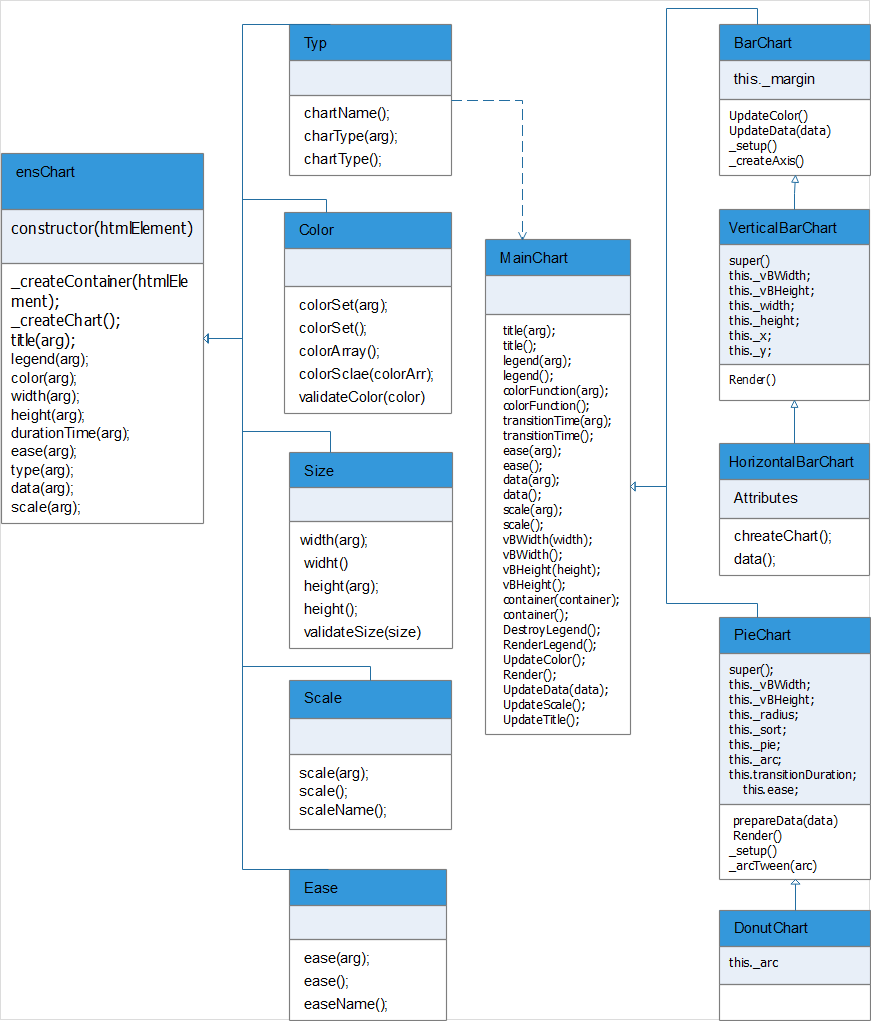


Abbildung 24: Klassendiagramm

Wie anhand der obigen Abbildung des Klassendiagrammes ersichtlich ist, blieb die Grundstruktur gleich, jedoch kamen einige Funktionen dazu bzw. die bereits beschriebenen Funktionen wurden verfeinert.

## Entwicklung

### ensChart Klasse

#### Zielelement (HTML-DIV-Element)

\_createContainer(htmlElement) {

if (htmlElement) {

if (typeof (htmlElement === 'string')) {

if(htmlElement.startsWith("#")){

let split = htmlElement.split("#");

htmlElement = split[1];

}

htmlElement = document.getElementById(htmlElement);

}

if (htmlElement instanceof HTMLElement) { … }

else {

console.error('Selection is neither ElementId nor HTMLElement');

return;

}

}

else {

console.error('You have to give a container');

return;

}

Diese Funktion sucht das HTML-DIV-Element, über dessen angegebene Id, in welchem das Chart angezeigt werden soll.

Der obige Code überprüft zuerst, ob der Parameter mitgeliefert wurde und ob es sich um den Typ String handelt. Anschliessend wird überprüft, ob der String mit einer Raute beginnt. Wenn dies der Fall ist, wird diese entfernt. Mit dem evtl. angepassten String wird anschliessend nach dem entsprechenden HTML-DIV-Element gesucht.

#### Data (array of objects)

data(arg) {

if (!arguments.length) {

return this.\_data ? this.\_data : [];

} else if (Array.isArray(arg)) {

this.\_data = arg;

this.\_chart.data = this.data();

return this;

}

console.error('Data is not an array');

return this;

}

Da die Datenstrukturen bei Charts sehr unterschiedlich sein können, wird in der Mainklasse nur überprüft, ob die gelieferten Daten einem Array entsprechen. Zusätzliche Validierungen werden erst in den betreffenden Charts durchgeführt.

#### Titel (string)

title(arg) {

if (!arguments.length) {

return this.\_title ? this.\_title : '';

} else if (typeof arg === 'string') {

this.\_title = arg;

this.\_chart.title = this.title();

return this;

}

console.error('Title not a String');

return this;

}

Mit der Funktion Titel wird ein Titel für das Chart erfasst. Dieser wird anschliessend über dem Chart angezeigt. Der Default Wert ist ein leerer String.

Zu Beginn wird überprüft, ob ein Parameter mitgeliefert wurde. Ist dies nicht der Fall, wird entweder der Default Value oder der zuvor gesetzte Value zurückgegeben.

Falls ein Parameter angegeben wurde, wird überprüft, ob dieser zum Typ String gehört. Wenn dies der Fall ist, wird der mitgelieferte Parameter als neuer Value gesetzt.

#### Legende (bool of visibility)

legend(arg) {

if (!arguments.length) {

return this.\_legend ? this.\_legend : false;

} else if (typeof arg === 'boolean') {

this.\_legend = arg;

this.\_chart.legend = this.legend();

return this;

}

console.error('Legend is not a Boolean');

return this;

}

Mit der Funktion Legende wird festgelegt, ob eine Legende zu den Datensätzen erstellt werden soll. Diese wird anschliessend unter dem Chart angezeigt. Der Default Wert ist der boolescher wert false.

Zu Beginn wird überprüft, ob ein Parameter mitgeliefert wurde. Ist dies nicht der Fall, wird entweder der Default Value oder der zuvor gesetzte Value zurückgegeben.

Falls ein Parameter angegeben wurde, wird überprüft, ob dieser zum Typ Boolean gehört. Wenn dies der Fall ist, wird der mitgelieferte Parameter als neuer Value gesetzt.

#### TransitionTime (number)

durationTime(arg) {

if (!arguments.length) {

return this.\_durationTime ? this.\_durationTime : 1000;

} else if (isFinite(arg)) {

this.\_durationTime = Math.abs(Number(arg));

this.\_chart.transitionTime = this.durationTime();

return this;

}

console.error('Legend is not a (positive) Number ');

return this;

}

Mit der Funktion durationTime wird festgelegt wie lange die Animation der Charts dauern soll. Der Default Wert ist der numerische Wert 1000. Hierbei ist zu beachten, dass der Value in Millisekunden angegeben wird.

Zu Beginn wird überprüft, ob ein Parameter mitgeliefert wurde. Ist dies nicht der Fall, wird entweder der Default Value oder der zuvor gesetzte Value zurückgegeben.

Falls ein Parameter angegeben wurde, wird überprüft, ob dieser endlich ist bzw. ob es eine reale Zahl ist. Wenn dies der Fall ist, wird der mitgelieferte Parameter als neuer Value gesetzt.

### Ausgelagerte Klassen

Da einige der Values weitere, komplexere Validierungen erfordern, wurden diese, wie bereits erwähnt, in separate Klassen ausgelagert, Jedoch erfolgt jeweils eine einfache Validierung sowie das Zurückgeben der Parameter über die Mainklasse.

#### Typ (string)

In der Main Klasse:

type(arg) {

if (!arguments.length) {

return this.\_cConfigType.chartName;

} else if (typeof arg === 'string') {

this.\_cConfigType.chartType = arg;

this.\_createChart();

return this;

}

console.error('Chart type is not a string');

return this;

}

#### Farben (string oder array mit Farbwerten)

In der Main Klasse:

color(arg) {

if (!arguments.length) {

return this.\_cConfigColor.colorArr;

} else if (typeof arg === 'string' || Array.isArray(arg)) {

this.\_cConfigColor.colorSet = arg;

this.\_chart.colorFunction = this.\_cConfigColor.colorSet;

return this;

}

console.error('Color is not a string or array');

return this;

}

#### Grösse (string in px oder number in px)

In der Main Klasse:

width(arg) {

if (!arguments.length) {

return this.\_cConfigSize.width;

} else if (isFinite(arg) || typeof arg === 'string') {

this.\_cConfigSize.width = arg;

if (this.\_chartContainer) {

this.styling.width = this.\_cConfigSize.width;

this.\_chartContainer.attr('style', "height: " + this.styling.height + ";" + " width: " + this.styling.width + ";");

}

return this;

}

console.error('width not a number or string');

return this;

}

#### Easing (string)

In der Main Klasse:

ease(arg) {

if (!arguments.length) {

return this.\_cConfigEase.easeName;

} else if (typeof arg === 'string') {

this.\_cConfigEase.ease = arg;

this.\_chart.ease = this.\_cConfigEase.ease;

return this;

}

console.error('ease not a string');

return this;

}

## Testprotokoll

Die Testprotokolle wurden in englischer Sprache verfasst gemäss dem Punkt sechs der «Richtlinien zur Softwareentwicklung» von enersis suisse AG: «Dokumentationen sind in englischer Sprache zu verfassen und sollen nach Möglichkeit mit grafischen Darstellungen zum besseren Verständnis ergänzt werden.»

Während der Realisierungsphase zeigte sich, dass zusätzliche Testfälle zu den in der Konzeptphase definierten Testfällen nötig waren. Diese wurden entsprechend ergänzt. Die folgenden Abbildungen stellen die Unit Tests dar, welche mit Mocha und Chai realisiert wurden.

### Constructor

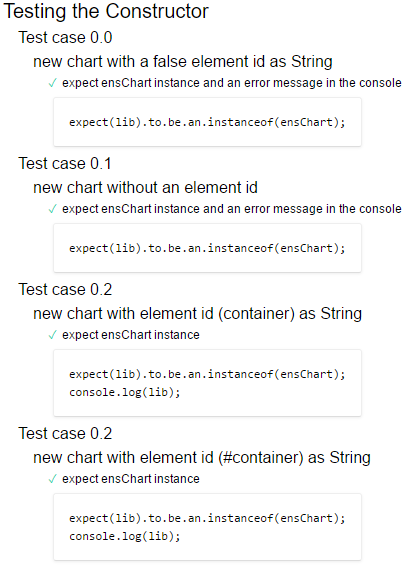


Abbildung 25: testing the Constructor

### Title

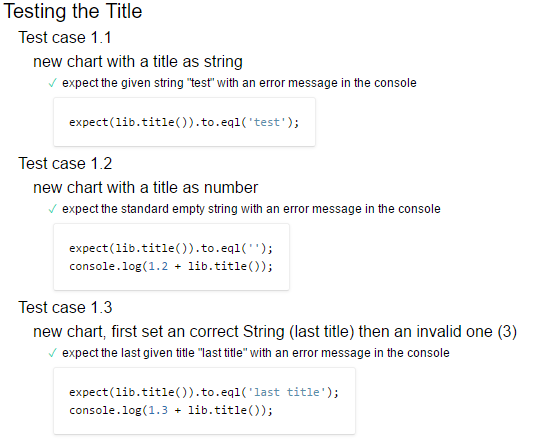


Abbildung 26: Testing the Title

### Legend

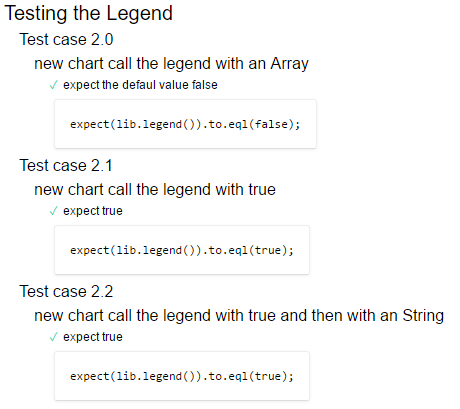


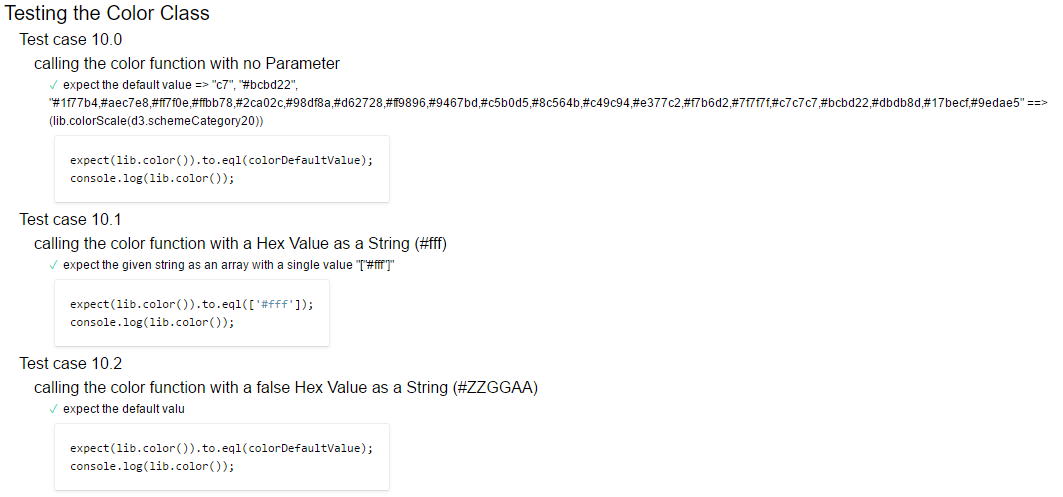
Abbildung 27: Testing the Legend

### DurationTime



Abbildung 28: Testing the DurationTime

### Color Klasse



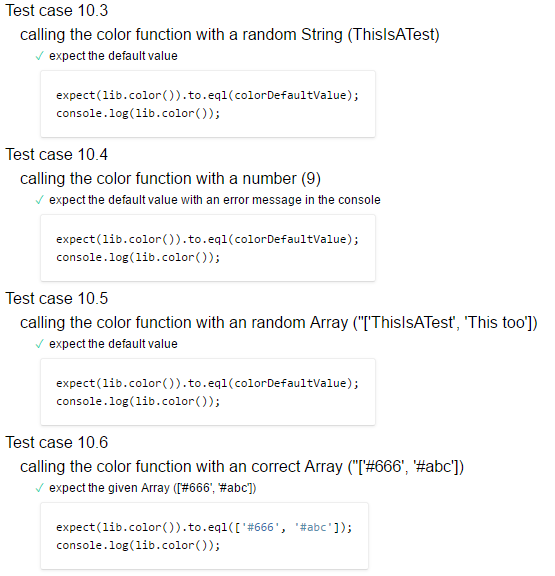


Abbildung 29: Testing the Color Class

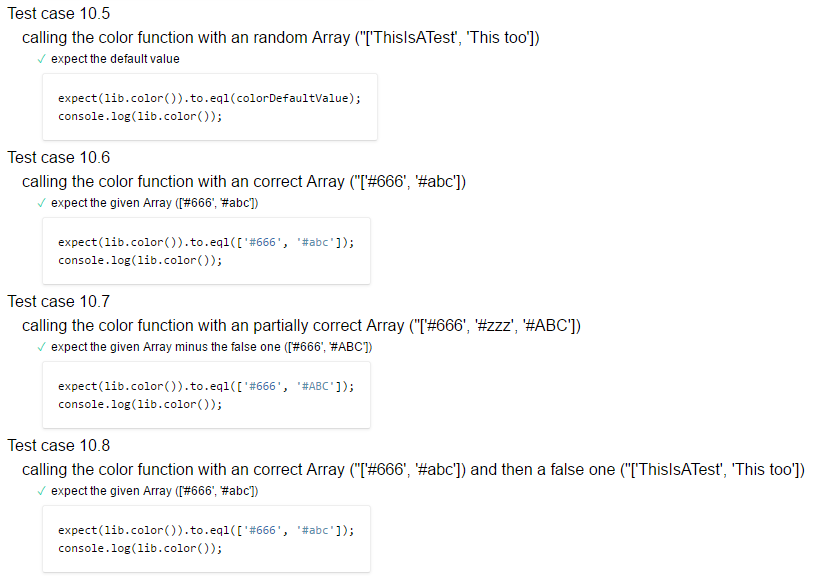


Abbildung 30: Testing the Color Class

### Ease Klasse

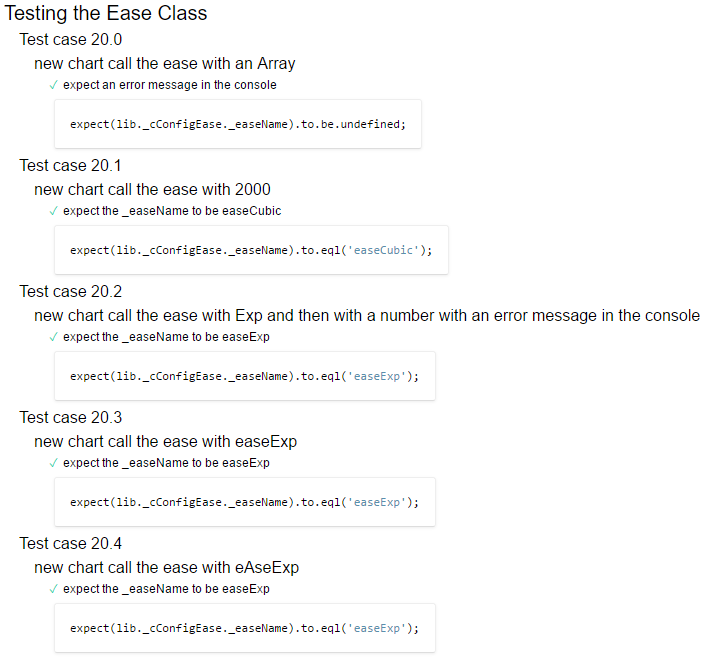


Abbildung 31: Testing the Ease Class

### Scale Klasse

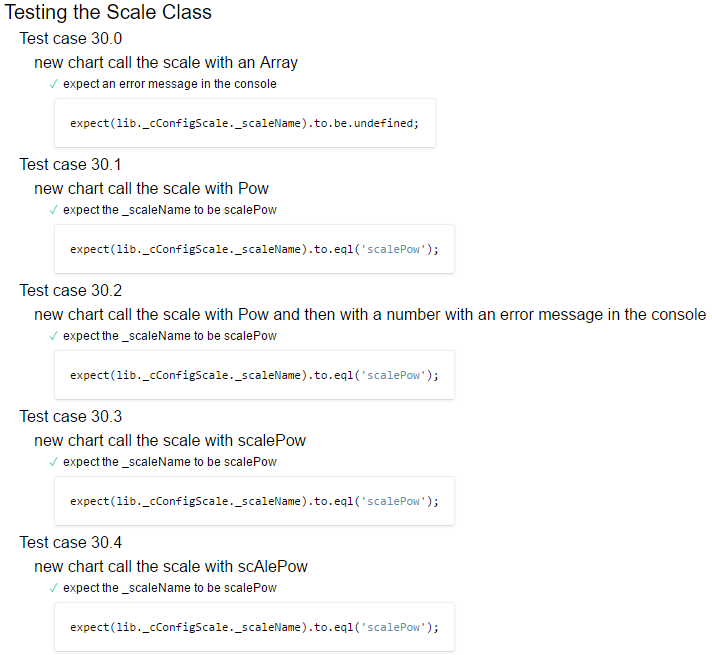


Abbildung 32: Testing the Scale Class

### Size Klasse

#### width



Abbildung 33: Testing the Size Class (width)

#### height

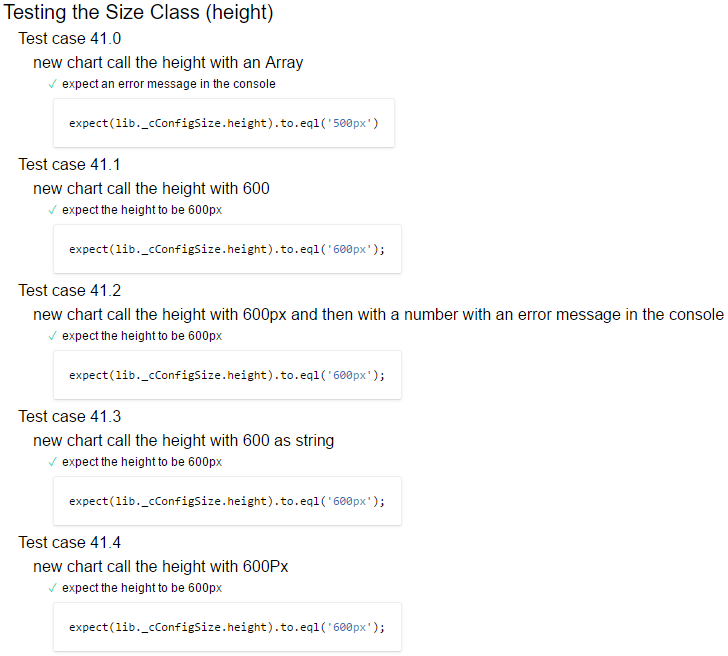


Abbildung 34: Testing the Size Class (height)

### Typ Klasse

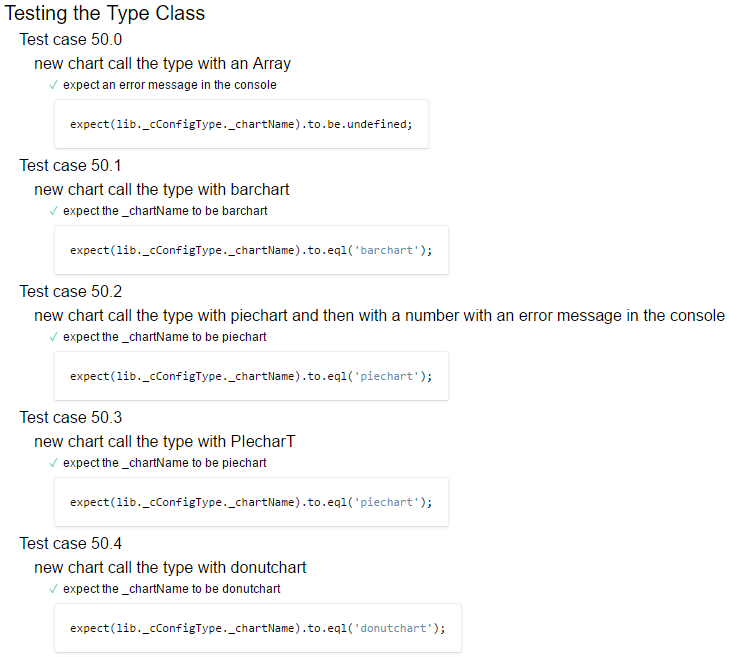


Abbildung 35: Testing Type Class

## Vergleich der Anforderung und der Umsetzung

### Funktionale Anforderungen

Funktionale Anforderungen beschreiben WAS das System bzw. Produkt leisten soll. Die funktionalen Anforderungen beschreiben somit die gewünschten Funktionalitäten des Systems, aber auch dessen Daten oder Verhalten.

* Die Chart Library soll folgende zwei Diagrammtypen zur Verfügung stellen:
  + Donut-Chart
  + Barchart
* Folgende Eigenschaften der beiden Charts sollen allgemein konfigurierbar sein:
  + Titel (string)
  + Typ (string)
  + Daten (array of objects oder analog den Anforderungen aus den bestehenden Charts)
  + Farben (string oder array mit Farbwerten)
  + Zielelement (HTML-DIV-Element)
  + Grösse (string in px oder number in px)
  + Legende (bool of visibility)
  + TransitionTime (number)
  + Easing (string)
* Die Chart Library soll als API zur Verfügung stehen, d.h. es gibt eine definierte Schnittstelle, wo entsprechende Parameter übergeben werden können.
* Die an die API übergebenen Parameter sollen validiert werden.
* Die Chart Library muss erweiterbar sein. Hierzu soll eine zusätzliche (frei definierbare) Eigenschaft, welche nur für eines der beiden Charts in der Library gilt, zur Erzeugung des Charts zur Verfügung stehen.

### Nicht funktionale Anforderungen

Nicht funktionale Anforderungen beschreiben WIE GUT das System etwas leisten soll und decken im Prinzip die qualitativen Aspekte und Bedingungen des Systems oder Funktionalitäten ab (z.B. Erweiterbarkeit).

* Die Chart Library soll als Java Script Dokument(e) zur Verfügung stehen, so dass diese zukünftig in Projekte von enersis integriert werden kann.
* Für die Darstellung und Manipulation der Daten in Form von Charts soll im Wesentlichen die D3.js Library genutzt werden.
* Für die Charts sind bestehende SVG-Templates zu nutzen.
* Die beiden oben genannten Charts
  + Donut-Chart und
  + Barchart

sollen mit Hilfe der API / Library als Beispielimplementation entwickelt und visualisiert werden.“.

## API Dokumentation vorbereiten

Eine API (ausgeschrieben[[15]](#footnote-15): application programming interface deutsch «Programmierschnittstelle» oder «Schnittstelle zur Anwendungs-Programmierung») in diesem Zusammenhang ist es die Schnittstelle zwischen dem Entwickler und der Library welche dokumentiert wird.

Diese API Dokumentation soll die Grundlage liefern, damit die enersis Entwickler die Chart Library ohne Probleme verwenden und erweitern können. Zudem soll die Dokumentation Klarheit bei möglichen Fehlern schaffen.

Die API wird ebenso wie die Kommentare im Source Code in englischer Sprache verfasst gemäss dem Punkt sechs der «Richtlinien zur Softwareentwicklung» der enersis suisse AG.

### Struktur der API Dokumentation

Für jeden Eingabe Parameter werden folgende Informationen beschrieben.

* Die Funktion die aufgerufen wird als Titel
* Eine kurze Beschreibung und die möglichen Eingabeparameter
* Ein simples Beispiel

Zusätzlich wird beschrieben, welche Schritte getätigt werden müssen, um ein neues Chart hinzuzufügen.

# API documentation ensChart

# General Information

This Library was realized with the “Webpack library starter”[[16]](#footnote-16)[[17]](#footnote-17) from Krasimir Tsonev. This Template features useful Libraries like:

* Webpack 2 based.
* ES6 as a source.
* Exports in a umd format so your library works everywhere.
* ES6 test setup with Mocha and Chai.
* Linting with ESLint.

# Getting started

Include the downloaded js File in your web page. With the script Tag. It should look something like this.

<script src="./lib/ensChart.js"></script>

<div id=”testId”></div>

<script src="./js/test.js"></script>

After you have done that you now can access the Library.  
Ensure the given path fits to the location of the Library file.

## Initializing the Library

Opening your test.js file you can now initialize the Library by writing something like this.

Lib = new ensChart(‘testId’);

Please pay attention to the parameter you’re giving. Because that’s the Id name of the div you want the chart to appear in.

You can give the Id with or without an #. But you must give an ID a class name won’t work.

# The different possible parameters you can set

All but a few of these are completely optional only the container parameter and the data are mandatory.

## Lib.title(*arg*)

The title function will be used to define the Title that will be shown at the top of the chart.

Lib.title('default Title');

The only type that is allowed is a string. If you try to set something else you will only get the default value: ‘’ and an error message in the console.

## Lib.data(*arg*)

The data function will be used to define the data that will be used in the chart.

Lib.data([{value: 6, title: ‘First Value’}, { value: 9, title: ‘Second Value’}, {…}]); || Lib.data([{ xAxis: 1, yAxis: 9, name: 'First Part' }, { xAxis: 2, yAxis: 4, name: 'Second Part' }, { xAxis: 3, yAxis: 9, name: 'Third Part' }])

The data is an array of objects. In general, it only needs the Title and the Value Properties to work correctly.

But for example, in the bar chart you need the xAxis value as well.

## Lib.legend(*arg*)

The legend function will be used to define if a Legend will be created.

Lib.legend(true);

The only type that is allowed is a boolean. If you try to set something else you will only get the default value: false and an error message in the console.

## Lib.color(*arg*)

The color function will be used to define what colors the charts will use. You can either give a String or an Array of Strings.

|  |
| --- |
| **Note: It is important that the strings are Hex values, but it is not important if you write an # or not. Also it is not case sensitive, that means it is not important if you write “#aabb00”, “aabb00”, “AB0”, “#AB0” or even “#Ab0”.** |

Lib.color(['#666', '#abc', ‘#4Ad’]); || Lib.color(‘a0a1a2’);

The only Type that is allowed is a String or an Array of Strings. If you try to set something else you will only get the default value:

d3.schemeCategory20  ["#1f77b4", "#aec7e8", "#ff7f0e", "#ffbb78", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5", "#8c564b", "#c49c94", "#e377c2", "#f7b6d2", "#7f7f7f", "#c7c7c7", "#bcbd22", "#dbdb8d", "#17becf", "#9edae5"]

and an error message in the console.

## Lib.ease(*arg*)

The ease function will be used to define what kind of animation the chart will have. The String you have to give will execute a d3 function, therefore you only can give the name of a easing d3 function.

On this Site, you have some Examples: http://bl.ocks.org/hunzy/9929724.

Lib. durationTime(3000);

The only Type that is allowed is a positive number. If you try to set something else you will only get the default value: 1000 and an error message in the console.

## Lib.durationTime(*arg*)

The durationTime function will be used to define how long the animation of the chart will take. The number’s unit is set to milliseconds (ms).

Lib. durationTime(3000);

The only Type that is allowed is a positive number. If you try to set something else you will only get the default value: 1000 and an error message in the console.

## Lib.scale(*arg*)

The durationTime function will be used to define how long the animation of the chart will take. The number’s unit is set to milliseconds (ms).

Lib. durationTime(3000);

The only Type that is allowed is a positive number if you try to set something else you will only get the default value: 1000 and an error message in the console.

## Lib.width(*arg*)

The width function will be used to define wide the chart is allowed to be. Since the Chart is responsive it will resize itself to the biggest possible version.

At the end your input will be a number in a String with the ending px. So, if your input is 90 the style will be set to “90px”. It doesn’t matter if you give the parameter as a number or a string.

Lib. width(90); || Lib. width(“90”); || Lib. width(“90px”);

## Lib.height(arg)

The height function will be used to define wide the chart is allowed to be. Since the Chart is responsive it will resize itself to the biggest possible version.

At the end your input will be a number in a String with the ending px. So, if your input is 90 the style will be set to “90px”. It doesn’t matter if you give the parameter as a number or a string.

Lib. height(90); || Lib. height(“90”); || Lib. height(“90px”);

## Lib.type(*arg*)

With the type function, you decide what Chart you want to use. At the moment, you can choose between:

|  |  |
| --- | --- |
| Nr. | Chart name |
| 1 | Piechart |
| 2 | Donutchart |
| 3 | HorizontalBarchart |
| 4 | VerticalBarchart |

Lib. type(Piechart);

|  |
| --- |
| **Note: It is important that you write it in one word.** |

# If something doesn’t work

First you should consider the console. If something went wrong, every function will give you an error message there and tell you what went wrong.

# If you want to include a new Chart into the Library

The steps for that are fairly simple. First of course you must get the Source Code and install it with Node in the project. After that follow these steps.

1. Save your new File in the Chart folder
   1. Make sure your New Chart class extends the MainChart
      1. Sample code: export default class PieChart extends MainChart {
2. Go to the type.config.js File
   1. Include your new Chart at the top of the file
      1. Sample code: import DonutChart from '../charts/pieChart/donut.chart';
   2. Write a new case in the setter method chartType
      1. Sample code: case 'piechart': this.\_chartType = new PieChart(); break;
   3. Test your new Chart
3. Done.

# Literatur und Quellenverzeichnis

Hinweis: An dieser Stelle muss ein Literatur- und Quellenverzeichnis eingefügt werden.

(es kann auch mit der Fussnote ein Hinweis auf die Quelle gemacht werden, diese muss aber im Quellen VZ ersichtlich sein

Method chaining Internet Quelle:

Name des Autors (falls erkennbar), „Titel der Seite“, Webadresse, Datum des letzten Zugriffs

Beispiel Internet

Quellenangabe „Koala“, http://de.wikipedia.org/wiki/Koala, 22.03.2008

Buch Quelle:

Erklärung Name des Autors: Titel. Verlag, Jahr, Seite, auf der der zitierte Text steht.

Beispiel Buch

Quellenangabe Ernst Walter Bauer: Humanbiologie. Cornelsen, 2006, S. 50.

# Glossar

|  |  |
| --- | --- |
| Begriff | Bedeutung |
| Achsenskalierung | Die Grössen Veränderung einer Achse eines Diagrammes (Beispiel Barchart) |
| Barchart | Eine Art von Diagramm (im deutschen ein Balkendiagramm) |
| Boilerplate (Code) | Sich immer wiederholender Codeteile. |
| custom controls | benutzerdefinierte Steuerelement |
| Donutchart | Eine Art von Diagramm (im deutschen ein Ringdiagramm) |
| ECMA-Script | Standardisierte Form von Java Script |
| Highchart-Charts | Highcharts ist ein Produckt der Firma Highsoft. Highcharts sind interaktive Java Script Diagramme |
| Java Script | Eine Programmiersprache die ursprünglich 1995 von Netscape für dynamisches HTML in Webbrowsern entwickelt wurde[[18]](#footnote-18) |
| Library | Bezeichnet in der Programmierung eine Sammlung von Unterprogrammen/-Routinen, die Lösungswege für thematisch zusammengehörende Problemstellungen anbieten.[[19]](#footnote-19) |
| Piechart | Eine Art von Diagramm (im deutschen ein Kreisdiagramm) |
| Pkorg | PkOrg unterstützt die verantwortlichen Prüfungsorganisationen bei der Durchführung der IPA (Individuelle Produktiv-Arbeit).[[20]](#footnote-20) |
| rendern | Software-Rendering bezeichnet die Methode der Grafikberechnung ohne spezialisierte Hardware, d. h. nur durch die CPU ohne Unterstützung durch eine Grafikkarte oder Vergleichbares.[[21]](#footnote-21) |
| stacked Barchart | Eine Art von Diagramm (im deutschen ein gestapeltes Balkendiagramm) |
| Template | Eine Art Vorlage die von einem anderen Programmierer übernimmt und mit seinen eigenen Änderungen anpasst. In diesem Projekt wäre das ein bereits konfiguriertes Projekt das jedoch noch keinerlei Inhalt hatte. |
| Unit Tests | Ein Modul Test auch als Komponententest bekannt. Welcher jede einzelne Funktion Testet. |

Tabelle 22: Glossar

# Unterschriften für Abnahme



Abbildung 36: Unterschrift für Abnahme

# Anhang

## Entwickler Richtlinien

Richtlinien für die Softwareentwicklung

Richtlinien zur Softwareentwicklung für das grids Framework

Autoren: Herr Andreas Jaus (enersis suisse AG)

Herr Mansur Esmann (enersis suisse AG)

Herr Berkay Üner (enersis suisse AG)

Verteiler: IT-Department

Version: Version 1.0,

Status: genehmigt

Letzte Änderung: 14.08.17 10:27:00

Klassifikation: **For Internal Use Only**© enersis Suisse AG 2016

Änderungsprotokoll

| **Version** | **Beschreibung; Bemerkung** | **Geändert durch:** | **Datum** |
| --- | --- | --- | --- |
| V 0.9 | Dokument eröffnet | EMA | 02.05.2016 |
| V 1.0 | Freigabe | JAN | 03.05.2016 |

Referenzierende Dokumente

|  |  |  |
| --- | --- | --- |
| **No** | **Dokument** | **Filename** |
|  |  |  |
|  |  |  |
|  |  |  |

Inhaltsverzeichnis

[1. Einleitung 3](#_Toc449998597)

[1.1. Ziel und Zweck dieses Dokuments 3](#_Toc449998598)

[1.2. Vorschriften und Standards 3](#_Toc449998599)

[2. Allgemeine Richtlinien und Verantwortlichkeiten 3](#_Toc449998600)

[2.1. Zuständigkeiten 3](#_Toc449998601)

[2.2. Technologische Verantwortlichkeiten 3](#_Toc449998602)

[3. Projektmanagement 4](#_Toc449998603)

[3.1. Methodik 4](#_Toc449998604)

[3.2. Projektmanagement Tool 4](#_Toc449998605)

[4. Softwareumgebung 5](#_Toc449998606)

[5. Entwicklungsrichtlinien 5](#_Toc449998607)

[5.1. Allgemein 5](#_Toc449998608)

[5.2. Grids 6](#_Toc449998609)

[5.3. Browserkompatibilität 6](#_Toc449998610)

[5.4. Coding Style 7](#_Toc449998611)

[5.4.1. Allgemein 7](#_Toc449998612)

[5.4.2. C# 7](#_Toc449998613)

[5.4.3. JavaScript 8](#_Toc449998614)

[5.4.4. PL-SQL 9](#_Toc449998615)

[6. Dokumentation 10](#_Toc449998616)

# Einleitung

## Ziel und Zweck dieses Dokuments

Dieser Katalog beschreibt die Richtlinien zur Softwareentwicklung bei enersis suisse AG. Er gilt nicht als Auftrag, sondern als Empfehlung. Gute Softwareentwicklung hat jedoch weitreichende Vorteile. Sie wird weniger Fehler enthalten und wird effektiver laufen als schlechte Programme. Da Software i.d.R. in seinem Lebenszyklus von mehr als einem Programmierer betreut wird, fällt es nachfolgenden Programmierern leichter die Entwicklung weiter zu führen, sofern sie bestimmte Standards vorfinden.

## Vorschriften und Standards

Dieses Dokument beinhaltet Richtlinien und Standards. Richtlinien sind nicht zwingend einzuhalten, stellen aber eine Empfehlung für eine gute und weitgehend akzeptierte Form der Softwareentwicklung dar. Standards sind in diesem Dokument entsprechend markiert.



Standard:

# Allgemeine Richtlinien und Verantwortlichkeiten

## Zuständigkeiten

Die Softwareentwicklung ist in enersis suisse AG der Abteilung IT/ Entwicklung unterstellt. Die Projektleiter stimmen auf Basis ihrer Projektplanung die notwendigen Entwickler-Ressourcen mit der Abteilungsleitung und/oder dem Teamleader ab.

Zuständigkeiten Entwicklung (Status per 02.05.2016):

|  |  |  |
| --- | --- | --- |
| **Zuständigkeit** | **Person** | **Aufgabenbeschreibung** |
| Abteilungsleitung | Andreas Jaus | Strategische Entscheidungen, Projekt- und Ressourcenplanung, Personalentwicklung |
| Teamleader | Mansur Esmann | Ressourcenplanung, strategische Beratung, SCRUM Master |
| Projektleitung | Projektabhängig verschiedene Personen | Product Owner, Projektplanung |
| Senior Developer | Mansur Esmann  Ivan Markovic | Software Architektur, Coaching der Developer, Detailkonzeption IT |
| Developer | Eric Kapitza  Berkay Üner  Nancy Lorena Ortiz  Stephan Plänitz  Daria Balkenende | Software Entwicklung |

## Technologische Verantwortlichkeiten

Die Softwareentwicklung ist nach den Kompetenzen der jeweiligen Entwickler in Verantwortlicher /- und stellvertretender Rolle definiert. Nachfolgende Matrix definiert dies.

|  |  |  |
| --- | --- | --- |
| **Bereich** | **Verantwortlich** | **Stellvertreter** |
| Frontend | Stephan Plänitz | Ivan Markovic |
| Backend | Ivan Markovic | Mansur Esmann |
| Datenbank | Mansur Esmann | Ivan Markovic |
| IT | Berkay Üner | Stephan Plänitz |
| Software Architektur | Mansur Esmann | Ivan Markovic |

# Projektmanagement

## Methodik

Innerhalb des Entwicklungsteams wird die SCRUM Methode angewendet. Soweit nicht anders vereinbart, beträgt die Laufzeit eines Sprints 10 Arbeitstage.

Das Sprint Planing Meeting wird regulär mit 6h (2 x 3h) angesetzt.

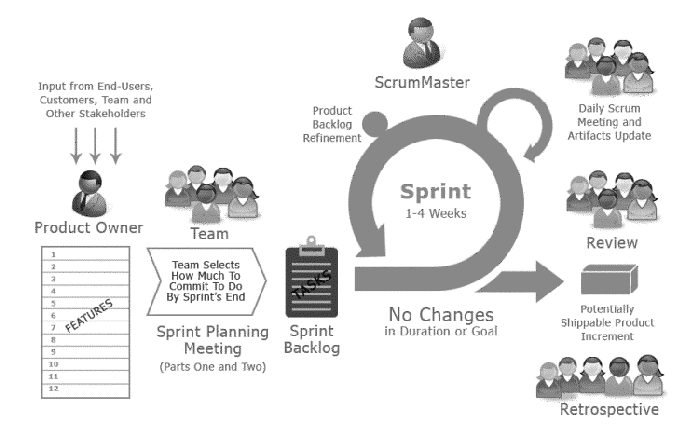
Täglich um 9:30 Uhr wird das Daily SCRUM Meeting abgehalten. Auf „zu spät Kommende“ wird keine Rücksicht genommen.

Der Review erfolgt am 11. Arbeitstag.

Die Publizierung erfolgt am 12. Arbeitstag.

Als Backlog Erfassungstool wird Team Foundation Server eingesetzt (Siehe 3.2 Projektmanagement Tool).

Folgende Inhalte sind im Sprint inkludiert (Visual):



## Projektmanagement Tool

Als Projektmanagement Tool wird Team Foundation Server eingesetzt. Er ist auffindbar unter <http://enersis.visualstudio.com>. Als Projekttemplate wird SCRUM eingesetzt. Nachfolgende Tabelle verdeutlicht die Verantwortlichkeit der WORK – Erfassung:

|  |  |  |
| --- | --- | --- |
| **Ebene** | **Verantwortung** | **Beschreibung** |
| Feature | Product Owner | Auf Featureebene soll ein vollständiger Themenblock erfasst werden. |
| Backlog | Product Owner / SCRUM Master | Backlogs beschreiben vollständig eine Teilanforderung eines Features und soll mit Beschreibung, Screenshots, Scribbles und einem erwarteten Aufwand versehen werden. Ein Backlog Item soll auf dieser Ebene bereits einem hauptverantwortlichen Entwickler zugeordnet werden. |
| Task | SCRUM Master / Developer | Tasks zu einem Backlog werden in Zusammenarbeit von Developer und SCRUM Master entwickelt. Vorrangig definieren die Entwickler welche detaillierten Tasks erforderlich sind. Der Backlog Verantwortliche ermittelt gemeinsam mit dem Team, welche Developer für die jeweiligen Tasks eingesetzt werden sollen. |

# Softwareumgebung

Den Developern steht ein definiertes Repertoire an Werkzeugen zur Verfügung. Grundsätzlich ist die Verwendung von individuellen Tools gestattet. Aktuell ausgewählte oder lizensierte Software soll jedoch nicht ersetzt werden. Lizenzpflichtige Software muss mit der Abteilungsleitung abgesprochen werden.

Nachfolgend ist die Liste der für Entwickler lizenzierten Softwareprodukte und deren Einsatzzweck aufgelistet:

|  |  |
| --- | --- |
| **Software** | **Zweck** |
| Visual Studio 2015 Community Edition | Entwicklungsumgebung für die Programmierung in C#, JavaScript und HTML |
| SQL Developer 3.x mit GeoRaptor | Datenbankentwicklung im Spatial Umfeld PL-SQL |
| SQL Developer 4.x | Allgemeine Datenbankentwicklung für Oracle PL-SQL |
| Microsoft Office 365 | Dokumentation (PPT, Word), Mail (Outlook), Datenaufbereitung (Excel / Access) |
| Notepad++ | Vielseitiges Werkzeug für Texteditierung |
| Eclipse Luna | SAP Hana Entwicklungen |
| WinSCP | FTP |
| SAP-DataService | ETL Umgebung |
| ReSharper | Erweiterung für VS für die Verbesserung des rapid development |
| Team Foundation Server | SourceControl der VisualStudio Projekte |

# Entwicklungsrichtlinien

## Allgemein

Bitte folge den folgenden, allgemeinen fünf Grundsätzen in der Softwareentwicklung bei enersis suisse:

1. Versuche nicht andere Softwareentwickler zu behindern
   1. Stelle sicher, dass dein Code kompiliert
   2. Stelle sicher, dass dein Code die automatischen Tests passiert
   3. Wenn du nicht compilierbaren Code einchecken musst, dann verwende Branch, oder versuche auf andere Weise den Einfluss auf andere Developer zu minimieren
2. Committe Code der gepflegt, wiederverwendbar und dokumentiert ist
   1. Formatiere Code so, dass er einheitlich ist und den nachfolgenden Richtlinien folgt
   2. Begrenze die Codezeilenlängen auf 120 Zeichen
   3. Verwende verständliche Bezeichnungen für Klassen, Variablen, etc. Verwende hierbei ausschließlich englische Ausdrücke
   4. Entferne Code vollständig, wenn er nicht benötigt wird und deaktiviere ihn nicht mittels Kommentarzeichen
3. Kommuniziere mit den anderen Entwickler
   1. Committe häufig und in logischen Einheiten und versehe das Commit mit einer sinnvollen Beschreibung
   2. Verwende .nuget für die Verwaltung von Abhängigkeiten / Assemblys
4. Vermeide Check-Ins von Dateien, die auf dein lokales System zugeschnitten sind
5. Vermeide Check-Ins von Code, der Outputs in die Konsole schreibt

## Grids

Das grids Framework besteht grundsätzlich aus einer 2-Schicht Architektur. Dem Frontend, das alle Funktionen der Applikation beinhaltet und der Datenbank-Ebene die neben dem Hosting der Daten auch die Business Logik beinhaltet. Die Komponenten folgen einer Servicearchitektur, wobei jede Komponente eigenständig betrieben wird, austauschbar ist und einen klar definierten Servicepoint anbietet. Vorrangiges Ziel ist die Vermeidung von serverseitigem Applikationscode. Grundsätzlich gilt: Ist eine serverseitige Interaktion erforderlich, so soll dies über einen definierten Servicepoint angeboten werden und unabhängig von anderen Komponenten laufen. Derzeit bestehen folgende Servicepoints:

1. Authentifizierung (EneAuth)(zur jetzigen Zeitpunkt DT benutzen)  
   Jegliche Authentifizierung muss über diesen Servicepoint erfolgen.
2. OData endpoint (DeepThought)  
   Jegliche Datenabfrage muss über diesen Service durchgeführt werden. Direkte Datenbankabfragen sind nicht zulässig
3. Product / Project configuration  
   Produktspezifische Konfigurationen werden hier abgefragt
4. Frontend (in Abhängigkeit zu den anderen Services)  
   Das Frontend ist immer in Abhängigkeit zu den Services

## Browserkompatibilität

Folgende Browser müssen von grids Framework Applikationen unterstützt werden. Vor jeder Publizierung muss auf diesen Browsern getestet werden:

|  |  |
| --- | --- |
| **Browser** | **Version** |
| Google Chrome | Immer die drei letzten Versionen von der aktuellen (d.zt. 50 49 48) |
| Mozilla Firefox | Immer die drei letzten Versionen von der aktuellen (d.zt. 45 44 43) |
| MS Internet Explorer | Immer die letzten 2 Versionen (d.zt. 11 10) |
| MS Edge | Derzeit Version 25 |
| Tablet Google Chrome 50 | Jeweils letzte Version |
| Tablet Safari 9 | Jeweils letzte Version |
| Tablet Mozilla Firefox | Jeweils letzte Version |

## Coding Style

### Allgemein

Alle Dateien der Softwareentwicklung müssen mit einem Kommentarblock versehen werden, in dem folgende Details aufgeführt sind:

* Name des Erstautors
* Datum der Erstellung
* Beschreibung über den Zweck des Files

Einrückungen sollen mit Tab und nicht mit Leerzeichen erfolgen.

Codekommentare sind im Besonderen dort erwünscht, wo sich der Zweck des Codes nicht unmittelbar ersehen lässt (Hack)

### C#

**Namensgebung**

* Lokale Variablen beginnen immer mit Kleinbuchstabe. Nachfolgewörter aus Zusammensetzungen werden groß geschrieben. Beispiel: currentPerformance
* Globale public Variablen werden groß geschrieben.

Beispiel: CurrentPerformance.

* Globale private und internal Variablen werden groß geschrieben und mit dem Prefix „\_“ (Underscore) versehen. Beispiel: \_CurrentPerformance.
* Parameter werden mit dem Prefix „p“ versehen. Beispiel: pCurrentPerformance.
* Vermeide Zahlen in Variablennamen

**Deklarationen**

Verwende nie var für Datentypen.

Beispiel: string currentPerformance = „This is a string value”;

**Using Direktiven**

Verwende immer using Direktiven und setze den Namespace nicht im Code ein.

Gut:

Using System.Diagnostics;

Schlecht:

Var currentPerfomanceCounterCategory = new System.Diagnostics.PerformanceCounterCategory();

**Layout**

Der Code soll in einem gut lesbaren Layout dargestellt werden. Folgende Layout Vorschläge machen den Code besser lesbar:

* Vier Leerzeichen als Einrückung
* Verwende Regions zur Strukturierung
* Ein Statement pro Zeile
* JSON Objekte und etwaige andere Inhaltsstrukturen sollen mit Hilfe von Zeilenumbrüchen lesbar gestaltet werden
* Eine Deklaration pro Zeile
* Bedingungsergebnisse um 4 Zeichen einrücken
* Eine Leerzeile zwischen Methodendefinition und Eigenschaften
* Klammern für Bedingungen, auch wenn der Ausdruck eine Zeile lang ist. Beispiel:

if((val1 > val2) && (val1 > val3))

{

// Some code

}

**Kommentare**

* Keine Kommentare am Ende der Zeile
* Punkt am Satzende
* Ein Leerzeichen zwischen // und dem Textbeginn

// Das ist ein Kommentar einer bestimmten Länge mit nutzlosen

// und recht einfältigem Text in ihm.

Jede Klasse und jede Public Methode muss mit der Code-Kommentarfunktion versehen werden. Beispiel:

#region web methods

/// <summary>

/// Check the login/pass of the user

/// </summary>

/// <param name="login">User's login</param>

/// <param name="password">User's password</param>

/// <param name="url">The url from where user aceess the login page</param>

/// <returns>json object of list of the porjects to which user has an access or 401 error if no access.</returns>

[WebMethod]

public string Login(string login, string password, string url)

**Try-catch**

Verwende immer try-catch-finally Anweisungen, bzw. try-catch-throw

**&& und || Operatoren**

Zur Vermeidung von Exceptions und zur Steigerung der Performance nutze && anstelle von & und || anstelle von |

### JavaScript

Als Richtlinie dient hier die Coding Convention folgender Zusammenstellung: <http://javascript.crockford.com/code.html>

**JS-Dateien**

JavaScript Code soll nicht in die HTML Datei eingebettet werden.

Es gelten die gleichen Regeln wie bei C#. Nachfolgend spezielle JS-Anforderungen.

**Layout**

* Verwende Leerzeilen zwischen Funktionen
* Trenne Schlüsselwörter vom rechts folgenden Rest mit einem Leerzeichen
* Verwende 4 Leerzeichen als Einrückung

**Variablen**

* Erkläre jede Variable mit einem Kommentar am Ende der Zeile
* Definiere Variablen immer nur am Anfang der Funktion, oder am Anfang des JS-Files (Globale)
* Keine Zahlen in Variablennamen
* If-Statements verwenden immer geschweifte Klammern. Die öffnende Klammer steht in der Folgezeile

### PL-SQL

**Pakete**

Alle Funktionen und Prozeduren werden als Teil eines Packages compiliert.

Pakete werden mit dem Prefix „PACK\_“ versehen.

Alle SQL Querys werden in Pakete gekapselt.

Die Prefixe „get\_“ und „set\_“ werden bei Prozedurnamen eingesetzt soweit möglich.

Quelldateien

Package Header und Body werden in einem File definiert. Jedes Package muss als Quelldatei im jeweiligen Projektordner gesichert werden. Maximales Alter einer Package Sicherung ist 14 Tage.

**Kommentar**

Alle Packages müssen im Package Header folgendes Kommentar aufweisen:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\* Filename: TopCoder\_Unit\_Test.pkg

/\* Component: Unit\_Test\_Framework

/\* Package: Unit\_Test\_Frmwrk

/\* Designer: TCSDesigner

/\* Developer: TCSDeveloper /\* Version: 1.0

/\* Copyright (c) 2006, TopCoder, Inc. All rights reserved.

/\*

/\* Description: Description of PL/SQL package...

/\*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Funktionen und Prozeduren werden mit folgendem Header kommentiert:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*\*

/\*\* Function: get\_employee\_ssn Confidential ©TopCoder Software, Inc. 2002 Page 2

/\*\* In: p\_employee\_id – the id of the employee to search for

/\*\* Returns: the Social Security Number for the employee

/\*\*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Jede Prozedur wird im Package Header folgendermaßen kommentiert:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*\* /\*\* Procedure: ins\_employee

/\*\* Out: p\_employee\_id – the id of the newly created employee.

/\*\* In: p\_ssn – the Social Security Number of the employee to

/\*\* insert.

/\*\* In: p\_name – the name of the employee to insert.

/\*\*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Layout**

* Maximale Zeilenlänge ist 120 Zeichen
* 4 Leerzeichen als Einrückung
* Komma Separierte Listen (Bspw. Feldnamen) werden um eine weitere Einrückung eingerückt
* In If-Blöcken wird das Then in eine neue Zeile geschrieben und eingerückt
* Reservierte Wörter werden nicht verwendet
* Mehrzeilige Stringwerte werden mit logischen und verständlichen Zeilenumbrüchen versehen
* Die Skripte werden grundsätzlich klein geschrieben.

**Variablen**

* Variablen werden ausschließlich im DECLARE Bereich definiert
* Parameter werden mit dem Prefix „p\_“ versehen
* Lokale Variablen werden mit dem Prefix „v\_“ versehen
* Variablen sollen nach Möglichkeit im DECLARE Bereich mit einem Default Wert versehen werden

**Sonstiges**

* Goto Statements werden nicht eingesetzt.
* Zu jedem Package / Anwendungsfall gibt es ein PL-SQL File zur Ausführung des Package / Prozedur – zusammen mit allen erforderlichen Parameter und funktionierenden Beispielwerten. Diese SQL-Programme müssen an einem zentralen Solution-Speicherplatz für alle relevanten Developer zugänglich sein (Nicht die lokale Festplatte).

# Dokumentation

Dokumentationen sind in englischer Sprache zu verfassen und sollen nach Möglichkeit mit grafischen Darstellungen zum besseren Verständnis ergänzt werden.

Codedokumentation wurde in den vorherigen Kapiteln behandelt und ist Pflicht.

Etwaige erforderliche Änderungen am Code oder an Konfigurationsparameter sind mit dem Text // TODO: zu versehen.

## Protokolle

### Erster Besuchstag

**Protokoll - Erster Besuchstag**

**Entwicklung einer Chart Library zur internen Verwendung**

| Auftraggeber | Andreas Jaus |
| --- | --- |
| Berufsbildner | Roland Dardel |
| Haupt-Experte | Maciej Styczynski |
| Zweit-Experte | Enrico Buchs |
| Projektleiterin | Nicole Sager |
| Autor | Nicole Sager |
| Klassifizierung | Nicht klassifiziert |
| Status | In Prüfung |

Sitzung: Erster Besuchstag

Datum: 27.04.2017

Zeit: 16:00-17:00

Ort: enersis suisse AG – Morgenstrasse, 131 3018 Bern; Sitzungszimmer

Anwesend: Andreas Jaus, Maciej Styczynski, Nicole Sager

Abwesend: Roland Dardel, Enrico Buchs

**Änderungsverzeichnis**

| Datum | Version | Änderung | Autor |
| --- | --- | --- | --- |
| 27.04.2017 | V1 | Erstellung der Vorlage | Nicole Sager |
| 27.04.2017 | V1.1 | Erster Besuchstag | Nicole Sager |
| 28.04.2017 | V1.2 | Textliche Anpassungen | Nicole Sager |

1. **Ablauf**

* Kennenlernen
* Projektleiterin beschreibt Aufgabenstellung IPA
  + Projektleiterin beschreibt den wichtigsten Teil einer Library (die Dokumentation)
* Projektleiterin beschreibt die Herausforderungen
* Besprechung des Zeitplans
* Besprechung allgemeine Punkte betreffend der IPA

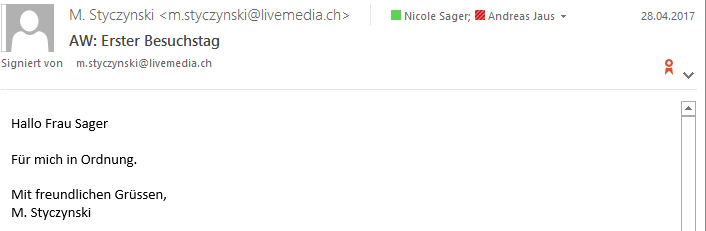
1. **Wichtige Punkte und Entscheidungen**

* Der Projektstart/ Initialisierungsauftrag ist kein Meilenstein und soll aus dem Zeitplan entfernt werden.
* Die eigenen Meilensteine sollen sich von den HERMES Meilensteinen unterscheiden.
* Im Arbeitsjournal werden die mündlichen Hilfestellungen auch erfasst. Beschrieben werden muss: Wer dabei war und was der Inhalt war.
* HERMES 5.1 IPA als Projektmethode wurde vom Hauptexperten akzeptiert.
* Der Hauptexperte will die IPA Dokumentation elektronisch. Endgültiger Entscheid wird beim zweiten Besuch gefällt.
* Zweit-Experte erhält die IPA Dokumentation via A-Post. Endgültiger Entscheid wird beim zweiten Besuch gefällt.
* Es müssen definierte Abnahmekriterien für die Meilensteine erstellt werden.

Zudem wurde die Projektleiterin darauf aufmerksam gemacht, sofern Fragen auftauchen, soll sie sich umgehend beim Fachvorgesetzten melden. Bei Fragen bezüglich der IPA soll sie sich an den Hauptexperten wenden.

Im Krankheitsfall oder Unfall soll sich die Projektleiterin umgehend beim Hauptexperten melden.

Bitte bestätigen Sie die Vollständigkeit dieses Dokuments (Bestätigung per E-Mail ausreichend).



### zweiter Besuchstag

**Protokoll - Zweiter Besuchstag**

**Entwicklung einer Chart Library zur internen Verwendung**

| Auftraggeber | Andreas Jaus |
| --- | --- |
| Berufsbildner | Roland Dardel |
| Haupt-Experte | Maciej Styczynski |
| Zweit-Experte | Enrico Buchs |
| Projektleiterin | Nicole Sager |
| Autor | Nicole Sager |
| Klassifizierung | Nicht klassifiziert |
| Status | In Prüfung |

Sitzung: Zweiter Besuchstag

Datum: 05.05.2017

Zeit: 8:30-9:00

Ort: enersis suisse AG – Morgenstrasse, 131 3018 Bern; Sitzungszimmer

Anwesend: Andreas Jaus, Maciej Styczynski, Nicole Sager

Abwesend: Roland Dardel, Enrico Buchs

**Änderungsverzeichnis**

| Datum | Version | Änderung | Autor |
| --- | --- | --- | --- |
| 27.04.2017 | V1 | Erstellung der Vorlage | Nicole Sager |
| 05.05.2017 | V1.1 | Zweiter Besuchstag | Nicole Sager |
|  |  |  |  |

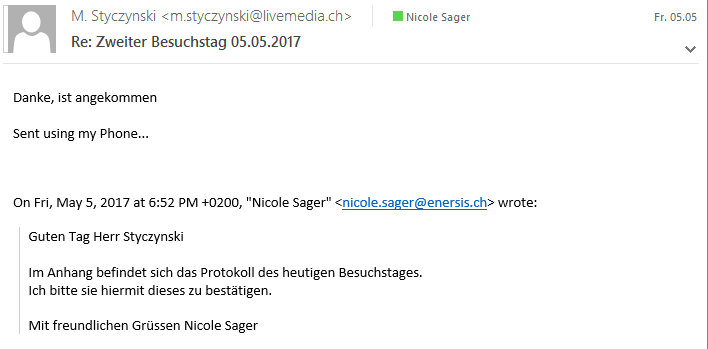
1. Ablauf

* Allgemeiner Stand des Projektes
  + Genauere Besprechung des Konzepts
  + Erklärung des Klassendiagramms
* Mini Demo des aktuellen Standes der Realisierung
* Informationen zur abschliessenden IPA Präsentation

1. Wichtige Punkte und Entscheidungen

* Der Anwendungsfall stimmt noch nicht mit dem UML Standard überein und muss angepasst werden.
* Bei der Ordnerstruktur soll der Inhalt eines ‘Tagesordners’ angezeigt werden.
* Die Dokumentation darf bis zum Schluss der IPA noch angepasst werden, mit Ausnahme der Aufgabenstellung und des Zeitplans.
* Der Code muss gut kommentiert werden.
* Das Websummary muss mindestens 24h vor der Präsentation hochgeladen werden. Ausserdem soll ein ausgedrucktes Exemplar für die Präsentation ausgedruckt werden.
* Der Hauptexperte möchte die fertiggestellte Dokumentation im PDF Format via E-Mail erhalten: [m.styczynski@livemedia.ch](mailto:m.styczynski@livemedia.ch)
* Der Zweitexperte möchte die fertiggestellte und ausgedruckte Dokumentation per Post erhalten: web4b GmbH, Buchenweg 5, 2552 Orpund

Bitte bestätigen Sie die Vollständigkeit dieses Dokuments (Bestätigung per E-Mail ausreichend).



## Source Code

### index.js

import \* as d3 from '../node\_modules/d3';

import ConfigColor from './config/color.config';

import ConfigEase from './config/ease.config';

import ConfigScale from './config/scale.config';

import ConfigSize from './config/size.config';

import ConfigType from './config/type.config';

export default class ensChart {

constructor(htmlElement) {

this.\_chart = {}; // defining variable to prevent 'undefined' errors

this.styling = { height: 'auto', width: 'auto' };

this.\_cConfigColor = new ConfigColor();

this.\_cConfigEase = new ConfigEase();

this.\_cConfigScale = new ConfigScale();

this.\_cConfigSize = new ConfigSize();

this.\_cConfigType = new ConfigType();

this.\_createContainer(htmlElement);

}

\_createContainer(htmlElement) {

if (htmlElement) {

if (typeof (htmlElement === 'string')) {

if (htmlElement.startsWith('#')) {

let split = htmlElement.split('#');

htmlElement = split[1];

}

htmlElement = document.getElementById(htmlElement);

}

if (htmlElement instanceof HTMLElement) {

this.\_chartContainer = d3

.select(htmlElement)

.append('div')

.attr('style', 'height: '+ this.styling.height + '; width: ' + this.styling.width + ';');

this.titleContainer = this.\_chartContainer.append('h1')

.attr('style', 'height: 10%; width: 100%;');

this.\_svgContainer = this.\_chartContainer

.append('svg')

.attr('viewBox', '0 0 0 0')

.attr('style', 'height: 90%; width: 100%;');

} else {

console.error('Selection is neither ElementId nor HTMLElement');

return;

}

} else {

console.error('You have to give a container');

return;

}

}

\_createChart() {

if (!this.\_svgContainer) {

console.error('Container not defined');

return;

} else if (this.\_cConfigType.chartType === undefined) {

console.error('Charttype undefined');

return;

}

this.\_chart = this.\_cConfigType.chartType;

this.\_svgContainer.html('');

this.\_chart.container = this.\_svgContainer;

this.\_chart.title = this.title();

this.\_chart.legend = this.legend();

this.\_chart.colorFunction = this.\_cConfigColor.colorSet;

this.\_chart.transitionTime = this.durationTime();

this.\_chart.ease = this.\_cConfigEase.ease;

this.\_chart.easeName = this.\_cConfigEase.\_easeName;

this.\_chart.scale = this.\_cConfigScale.scale;

this.\_chart.data = this.data();

this.\_chart.Render();

}

/\*\*

\* arg - stands for argument and is used to indicate a given Parameter

\*/

/\*\* The @function title(arg)

\* first it confirms that the Parameter is actually given.

\* If not you either get the default value ''

\* or the last set property.

\*

\* If you have given a Parameter it validates that it is of the type String.

\* If that is the case it sets the Parameter as the new value.

\*

\* If both validations fail. Say you have given an object as the parameter it gives you an

\* error message in the Console, describing what the datatype of the parameter should be.

\*

\* @param {any} arg

\* @returns this

\* @memberof ensChart

\*/

title(arg) {

if (!arguments.length) {

return this.\_title ? this.\_title : '';

} else if (typeof arg === 'string') {

this.\_title = arg;

this.\_chart.title = this.title();

this.\_titleContainer = this.titleContainer

.text(this.\_title);

return this;

}

console.error('Title not a String');

return this;

}

/\*\* The @function legend(arg)

\* first it confirms that the Parameter is actually given.

\* If not you either get the default value 'false'

\* or the last set property.

\*

\* If you have given a Parameter it validates that it is of the type Boolean.

\* If that is the case it sets the Parameter as the new value.

\*

\* If both validations fail. Say you have given an object as the parameter it gives you an

\* error message in the Console, describing what the datatype of the parameter should be.

\*

\* @param {any} arg

\* @returns this

\* @memberof ensChart

\*/

legend(arg) {

if (!arguments.length) {

return this.\_legend ? this.\_legend : false;

} else if (typeof arg === 'boolean') {

this.\_legend = arg;

this.\_chart.legend = this.legend();

return this;

}

console.error('Legend is not a Boolean');

return this;

}

/\*\* The @function color(arg)

\* first it confirms that the Parameter is actually given.

\* If not you either get the default value '['#72ecfa', '#dcffa0', '#fa8072', '#d20057', '#10c390']'

\* or the last set property.

\*

\* If you have given a Parameter of the type Number it validates that it is either a String or an Array.

\* If its just a String the string gets convertet into an Array with a single propertie

\* If that is the case it sets the Parameter as the new value.

\*

\* If both validations fail. Say you have given a string as the parameter it gives you an

\* error message in the Console, describing what the datatype of the parameter should be.

\*

\* @param {any} arg

\* @returns this

\* @memberof ensChart

\*/

color(arg) {

if (!arguments.length) {

return this.\_cConfigColor.colorArr;

} else if (typeof arg === 'string' || Array.isArray(arg)) {

this.\_cConfigColor.colorSet = arg;

this.\_chart.colorFunction = this.\_cConfigColor.colorSet;

return this;

}

console.error('Color is not a string or array');

return this;

}

width(arg) {

if (!arguments.length) {

return this.\_cConfigSize.width;

} else if (isFinite(arg) || typeof arg === 'string') {

this.\_cConfigSize.width = arg;

if (this.\_chartContainer) {

this.styling.width = this.\_cConfigSize.width;

this.\_chartContainer.attr(

'style',

'height: ' +

this.styling.height +

';' +

' width: ' +

this.styling.width +

';'

);

}

return this;

}

console.error('width not a number or string');

return this;

}

height(arg) {

if (!arguments.length) {

return this.\_cConfigSize.height;

} else if (isFinite(arg) || typeof arg === 'string') {

this.\_cConfigSize.height = arg;

if (this.\_chartContainer) {

this.styling.height = this.\_cConfigSize.height;

this.\_chartContainer.attr(

'style',

'height: ' +

this.styling.height +

';' +

' width: ' +

this.styling.width +

';'

);

}

return this;

}

console.error('height not a number or string');

return this;

}

/\*\* The @function durationTime(arg)

\* first it confirms that the Parameter is actually given.

\* If not you either get the default value '1000'

\* or the last set property.

\*

\* If you have given a Parameter of the type Number it validates that it is positive.

\* If that is the case it sets the Parameter as the new value.

\*

\* If both validations fail. Say you have given a string as the parameter it gives you an

\* error message in the Console, describing what the datatype of the parameter should be.

\*

\* @param {any} arg

\* @returns this

\* @memberof ensChart

\*/

durationTime(arg) {

if (!arguments.length) {

return this.\_durationTime ? this.\_durationTime : 1000;

} else if (isFinite(arg)) {

this.\_durationTime = Math.abs(Number(arg));

this.\_chart.transitionTime = this.durationTime();

return this;

}

console.error('Legend is not a (positive) Number ');

return this;

}

ease(arg) {

if (!arguments.length) {

return this.\_cConfigEase.easeName;

} else if (typeof arg === 'string') {

this.\_cConfigEase.ease = arg;

this.\_chart.ease = this.\_cConfigEase.ease;

return this;

}

console.error('ease not a string');

return this;

}

type(arg) {

if (!arguments.length) {

return this.\_cConfigType.chartName;

} else if (typeof arg === 'string') {

this.\_cConfigType.chartType = arg;

this.\_createChart();

return this;

}

console.error('Chart type is not a string');

return this;

}

data(arg) {

if (!arguments.length) {

return this.\_data ? this.\_data : [];

} else if (Array.isArray(arg)) {

this.\_data = arg;

this.\_chart.data = this.data();

return this;

}

console.error('Data is not an array');

return this;

}

scale(arg) {

if (!arguments.length) {

return this.\_cConfigScale.scaleName;

} else if (typeof arg === 'string') {

this.\_cConfigScale.scale = arg;

this.\_chart.scale = this.\_cConfigScale.scale;

return this;

}

console.error('Chart type is not a string');

return this;

}

}

### color.config.js

import \* as d3 from '../../node\_modules/d3';

export default class ConfigColor {

set colorSet(arg) {

let colors = [];

if (typeof arg === 'string') {

if (this.validateColor(arg)) {

colors.push(arg);

}

} else if (Array.isArray(arg)) {

for (let i = 0, x = arg.length; i < x; i++) {

if (typeof arg[i] === 'string' && this.validateColor(arg[i])) {

colors.push(arg[i]);

}

}

} else {

console.error('Color is not a string or array');

return;

}

if (!colors.length){

return;

}

this.\_colorArr = colors;

this.\_colors = this.colorScale(colors);

}

get colorSet() {

return this.\_colors ? this.\_colors : this.colorScale(d3.schemeCategory20);

}

get colorArr(){

return this.\_colorArr ? this.\_colorArr : d3.schemeCategory20;

}

colorScale(colorArr) {

return d3.scaleOrdinal().range(colorArr).domain(0, 1);

}

validateColor(color) {

let isHex = /(^#[0-9A-F]{6}$)|(^#[0-9A-F]{3}$)/i;

if (isHex.test(color)) {

return true;

}

return false;

}

}

### ease.config.js

import \* as d3 from '../../node\_modules/d3';

export default class ConfigEase {

set ease(arg) {

if (typeof arg === 'string') {

arg = arg.toLowerCase();

if (arg.startsWith('ease')) {

arg = arg.split('ease')[1];

}

let easing = 'ease' + arg[0].toUpperCase() + arg.slice(1).toLowerCase();

if (d3[easing]) {

this.\_ease = d3[easing];

this.\_easeName = easing;

} else {

console.error(easing + ' is not a d3 function');

return;

}

} else {

console.error('Ease not a string');

return;

}

}

get ease() {

return this.\_ease ? this.\_ease : d3.easeLinear;

}

get easeName() {

return this.\_easeName ? this.\_easeName : 'easeLinear';

}

}

### scale.config.js

import \* as d3 from '../../node\_modules/d3';

export default class ConfigScale {

set scale(arg) {

if (typeof arg === 'string') {

arg = arg.toLowerCase();

if (arg.startsWith('scale')) {

arg = arg.split('scale')[1];

}

let scaling = 'scale' + arg[0].toUpperCase() + arg.slice(1).toLowerCase();

if (d3[scaling]) {

this.\_scale = d3[scaling];

this.\_scaleName = scaling;

} else {

console.error(scaling + ' is not a d3 function');

return;

}

} else {

console.error('Scale not a string');

return;

}

}

get scale() {

return this.\_scale ? this.\_scale : d3.scaleLinear;

}

get scaleName() {

return this.\_scaleName ? this.\_scaleName : 'scaleLinear';

}

}

### size.config.js

export default class ConfigSize {

set width(arg) {

if (!this.validateSize(arg)){

return;

}

this.\_width = this.validateSize(arg);

}

get width() {

return this.\_width ? this.\_width : '500px';

}

set height(arg) {

if (!this.validateSize(arg)){

return;

}

this.\_height = this.validateSize(arg);

}

get height() {

return this.\_height ? this.\_height : '500px';

}

validateSize(size) {

if (typeof size === 'string') {

size = size.toLowerCase();

}

if (typeof size === 'string' && size.endsWith('px')) {

size = size.split('px')[0];

}

if (isFinite(size)) {

return size + 'px';

}

console.error('Size argument not correct');

return;

}

}

### type.config.js

import BarChart from '../charts/barChart/bar.chart';

import VerticalBarChart from '../charts/barChart/vertical';

import HorizontalBarChart from '../charts/barChart/horizontal';

import PieChart from '../charts/pieChart/pie.chart';

import DonutChart from '../charts/pieChart/donut.chart';

export default class ConfigType {

constructor() {}

get chartName() {

return this.\_chartName;

}

get chartType() {

return this.\_chartType;

}

set chartType(arg) {

if (typeof arg === 'string') {

switch (arg.toLowerCase()) {

case 'barchart':

this.\_chartType = new BarChart();

break;

case 'horizontalbarchart':

this.\_chartType = new BarChart();

break;

case 'verticalbarchart':

this.\_chartType = new BarChart();

break;

case 'piechart':

this.\_chartType = new PieChart();

break;

case 'donutchart':

this.\_chartType = new DonutChart();

break;

default:

console.error('This type is not available');

return;

}

this.\_chartName = arg.toLowerCase();

} else {

console.error('type not string');

return;

}

}

}

### main.chart.js

/\*\*

\*

\*

\* @export

\* @class MainChart

\*/

export default class MainChart {

constructor() {}

set title(arg) {

this.\_title = arg;

this.UpdateTitle();

}

get title() {

return this.\_title;

}

set legend(arg) {

if (arg === true) {

this.RenderLegend();

} else {

this.DestroyLegend();

}

this.\_legend = arg;

}

get legend() {

return this.\_legend;

}

set colorFunction(arg) {

this.\_colorFunction = arg;

this.UpdateColor();

}

get colorFunction() {

return this.\_colorFunction;

}

set transitionTime(arg) {

this.\_transitionTime = arg;

}

get transitionTime() {

return this.\_transitionTime;

}

set ease(arg) {

this.\_ease = arg;

}

get ease() {

return this.\_ease;

}

set data(arg) {

this.\_data = arg;

this.UpdateData(arg);

}

get data() {

return this.\_data;

}

set scale(arg) {

this.\_scale = arg;

this.UpdateScale();

}

get scale() {

return this.\_scale;

}

set vBWidth(width) {

this.\_vBWidth = width;

}

get vBWidth() {

return this.\_vBWidth ? this.\_vBWidth : 0;

}

set vBHeight(height) {

this.\_vBHeight = height;

}

get vBHeight() {

return this.\_vBHeight ? this.\_vBHeight : 0;

}

set container(container) {

this.\_container = container;

this.\_container.attr(

'viewBox',

'0 0 ' + this.vBWidth + ' ' + this.vBHeight

);

this.\_container;

}

get container() {

return this.\_container;

}

DestroyLegend() {

return;

}

RenderLegend() {

return;

}

UpdateColor() {

return;

}

Render() {

return;

}

UpdateData(data) {

return;

}

UpdateScale() {

return;

}

UpdateTitle() {

return;

}

}

### donut.chart.js

import PieChart from './pie.chart';

export default class DonutChart extends PieChart {

constructor() {

super();

this.\_arc.innerRadius(this.\_radius \* 0.6);

}

}

### pie.chart.js

import MainChart from '../main.chart';

import \* as d3 from '../../../node\_modules/d3';

export default class PieChart extends MainChart {

constructor() {

super();

this.\_vBWidth = 1000;

this.\_vBHeight = 1000;

this.\_radius = Math.min(this.\_vBWidth, this.\_vBHeight) / 2;

this.\_sort = null;

this.\_pie = d3.pie().sort(this.\_sort).value(function (d) {

return d.value;

});

this.\_arc = d3.arc().outerRadius(this.\_radius \* 0.8).innerRadius(0);

this.transitionDuration;

this.ease;

}

prepareData(data) {

for (let i = 0, x = data.length; i < x; i++) {

if (!data[i].value) {

data.splice(i, 1);

i--;

} else if (!data[i].name) {

data[i].name = 'unknown';

}

}

if (!data.length) {

console.error('the Data You have given is invalid or nonexistent.');

}

this.\_data = data;

return;

}

Render() {

if (!this.\_data || !this.data.length) {

return;

}

if (!this.\_setupDone) {

this.\_setup();

this.\_setupDone = true;

}

var that = this;

var arc = this.\_svg.selectAll('path')

.on("change", change)

.data(that.\_pie(that.\_data));

arc

.enter()

.append('path')

.attr('d', that.\_arc)

.attr('fill', function (d, i) {

return that.colorFunction(i);

});

arc.call(changeAngle);

function change() {

let value = that.\_data;

clearTimeout(timeout);

pie.value(function(d) { return d[value]; }); // change the value function

path = path.data(pie); // compute the new angles

path.transition().duration(that.transitionDuration).attrTween("d", arcTween); // redraw the arcs

}

function changeAngle(selection) {

selection

.transition()

.ease(that.ease)

.duration(that.transitionDuration)

.attrTween('d', that.\_arcTween(that.\_arc));

}

arc.exit().remove();

}

\_setup() {

this.\_svg = this.container.append('g');

this.\_svg.attr(

'transform',

'translate(' + this.\_vBWidth / 2 + ',' + this.\_vBHeight / 2 + ')'

);

}

\_arcTween(arc) {

return function (d) {

this.\_current = this.\_current || d;

var interpolate = d3.interpolate(this.\_current, d);

this.\_current = interpolate(0);

return function (t) {

return arc(interpolate(t));

};

};

}

}

### bar.chart.js

import \* as d3 from '../../../node\_modules/d3';

import MainChart from '../main.chart';

export default class BarChart extends MainChart {

constructor() {

super()

this.\_margin = { top: 20, right: 20, bottom: 30, left: 40 };

}

UpdateColor() {

if (!this.\_svg) {

return;

}

var that = this;

this.\_svg.selectAll(".bar").transition().ease(that.ease).duration(that.transitionDuration)

.attrTween("fill", function (d, i) {

debugger;

return d3.interpolate(d3.select(this).attr("fill"), that.colorSet(0));

});

}

UpdateData(data) {

for (let i = 0, x = data.length; i < x; i++) {

if (!data[i].xAxis || !data[i].yAxis) {

data.splice(i, 1);

i--;

} else if (!data[i].name) {

data[i].name = data[i].xAxis;

}

}

this.\_data = data;

this.Render();

//hack

this.Render();

}

\_setup() {

this.\_svg = this.container.append("g");

this.\_svg.attr("transform", "translate(" + this.\_margin.left + "," + this.\_margin.top + ")");

this.\_createAxis();

}

\_createAxis(){

this.\_svg.append("g")

.attr("class", "x axis")

.attr("transform", "translate(0," + this.\_height + ")");

this.\_svg.append("g")

.attr("class", "y axis")

.append("text")

.attr("transform", "rotate(-90)")

.attr("y", 6)

.attr("dy", "0.71em")

.attr("text-anchor", "end")

.text("yAxis");

}

}

### horizontal.js

import \* as d3 from '../../../node\_modules/d3';

import VerticalBarChart from './vertical';

export default class HorizontalBarChart extends VerticalBarChart {

constructor() {

super()

}

\_setup() {

this.\_svg = this.container.append("g");

this.\_svg.attr("transform", "translate(" + (this.\_vBWidth /1.5) + "," + (-this.\_margin.top\*9) + ") rotate(90)");

this.\_svg.append("g")

.attr("class", "x axis")

.attr("transform", "translate(0," + this.\_height + ")");

this.\_svg.append("g")

.attr("class", "y axis")

.append("text")

.attr("transform", "rotate(90)")

.attr("y", 6)

.attr("dy", "0.71em")

.attr("text-anchor", "end")

.text("yAxis");

}

}

### vertical.js

import \* as d3 from '../../../node\_modules/d3';

import BarChart from './barchart.js';

export default class VerticalBarChart extends BarChart {

constructor() {

super()

this.\_vBWidth = 960;

this.\_vBHeight = 500;

this.\_width = this.\_vBWidth - this.\_margin.left - this.\_margin.right;

this.\_height = this.\_vBHeight - this.\_margin.top - this.\_margin.bottom;

this.\_x = d3.scaleBand().rangeRound([0, this.\_width]).padding(0.1);

this.\_y = d3.scaleLinear().rangeRound([this.\_height, 0]);

}

Render() {

if (!this.\_data) {

return;

}

if (!this.\_setupDone) {

this.\_setup();

this.\_setupDone = true;

}

var that = this;

this.\_x.domain(this.\_data.map(function (d) { return d.xAxis; }));

this.\_y.domain([0, d3.max(this.\_data, function (d) { return d.yAxis; })]);

this.\_svg.select('.x.axis').transition().ease(that.ease).duration(that.transitionDuration).call(d3.axisBottom(that.\_x));

this.\_svg.select(".y.axis").transition().ease(that.ease).duration(that.transitionDuration).call(d3.axisLeft(that.\_y).ticks(10));

var bars = this.\_svg.selectAll(".bar").data(that.\_data, function (d) { return d.xAxis; })

bars.exit()

.transition()

.ease(that.ease)

.duration(that.transitionDuration)

.attr("y", this.\_y(0))

.attr("height", this.\_height - this.\_y(0))

.style('fill-opacity', 1e-6)

.remove();

bars.enter().append("rect")

.attr("class", "bar")

.attr("y", this.\_y(0))

.attr("height", this.\_height - this.\_y(0))

.attr("fill", that.colorSet(0))

bars.transition().ease(that.ease).duration(that.transitionDuration).attr("x", function (d) { return that.\_x(d.xAxis); })

.attr("width", that.\_x.bandwidth()) // constant, so no callback function(d) here

.attr("y", function (d) { return that.\_y(d.yAxis); })

.attr("height", function (d) { return that.\_height - that.\_y(d.yAxis); });

}

}

### index.html

<!DOCTYPE html>

<html>

<head>

<title>Mocha Tests</title>

<link rel="stylesheet" href="./node\_modules/mocha/mocha.css">

</head>

<body>

<div id="mocha"></div>

<script src="./node\_modules/mocha/mocha.js"></script>

<script src="./node\_modules/chai/chai.js"></script>

<script>

mocha.setup('bdd')

</script>

<!-- load code you want to test here -->

<script src="./lib/ensChart.js"></script>

<!-- load your test files here -->

<script src="./test/library.spec.js"></script>

<div id="container"></div>

<script src="./test/color.spec.js"></script>

<div id="colorContainer"></div>

<script src="./test/ease.spec.js"></script>

<div id="easeContainer"></div>

<script src="./test/scale.spec.js"></script>

<div id="scaleContainer"></div>

<script src="./test/size.spec.js"></script>

<div id="sizeContainer"></div>

<script src="./test/type.spec.js"></script>

<div id="typeContainer"></div>

<script>

mocha.run();

</script>

</body>

</html>

### library.spec.js

/\* global describe, it, before \*/

chai.expect();

const expect = chai.expect;

let lib;

describe('Testing the Constructor', () => {

context('Test case 0.0', function () {

before(() => {

lib = new ensChart('asd');

});

describe('new chart with a false element id as String', () => {

it('expect ensChart instance and an error message in the console', () => {

expect(lib).to.be.an.instanceof(ensChart);

});

});

});

context('Test case 0.1', function () {

before(() => {

lib = new ensChart();

});

describe('new chart without an element id', () => {

it('expect ensChart instance and an error message in the console', () => {

expect(lib).to.be.an.instanceof(ensChart);

});

});

});

context('Test case 0.2', function () {

before(() => {

lib = new ensChart('container');

});

describe('new chart with element id (container) as String', () => {

it('expect ensChart instance', () => {

expect(lib).to.be.an.instanceof(ensChart);

});

});

});

context('Test case 0.2', function () {

before(() => {

lib = new ensChart('#container');

});

describe('new chart with element id (#container) as String', () => {

it('expect ensChart instance', () => {

expect(lib).to.be.an.instanceof(ensChart);

});

});

});

});

describe('Testing the Title', () => {

context('Test case 1.1', function () {

before(() => {

lib = new ensChart('container').title('test');

});

describe('new chart with a title as string', () => {

it('expect the given string "test" with an error message in the console', () => {

expect(lib.title()).to.eql('test');

});

});

});

context('Test case 1.2', function () {

before(() => {

lib = new ensChart('container').title(3);

});

describe('new chart with a title as number', () => {

it('expect the standard empty string with an error message in the console', () => {

expect(lib.title()).to.eql('');

});

});

});

context('Test case 1.3', function () {

before(() => {

lib = new ensChart('container').title('last title').title(3);

});

describe('new chart, first set an correct String (last title) then an invalid one (3)', () => {

it('expect the last given title "last title" with an error message in the console', () => {

expect(lib.title()).to.eql('last title');

});

});

});

});

describe('Testing the Legend', () => {

context('Test case 2.0', function () {

before(() => {

lib = new ensChart('container').legend(['test']);

});

describe('new chart call the legend with an Array', () => {

it('expect the defaul value false', () => {

expect(lib.legend()).to.eql(false);

});

});

});

context('Test case 2.1', function () {

before(() => {

lib = new ensChart('container').legend(true);

});

describe('new chart call the legend with true', () => {

it('expect true', () => {

expect(lib.legend()).to.eql(true);

});

});

});

context('Test case 2.2', function () {

before(() => {

lib = new ensChart('container').legend(true).legend('test');

});

describe('new chart call the legend with true and then with an String', () => {

it('expect true', () => {

expect(lib.legend()).to.eql(true);

});

});

});

});

describe('Testing the durarionTime', () => {

context('Test case 3.0', function () {

before(() => {

lib = new ensChart('container').durationTime(['test']);

});

describe('new chart call the durationTime with an Array', () => {

it('expect the defaul value 1000', () => {

expect(lib.durationTime()).to.eql(1000);

});

});

});

context('Test case 3.1', function () {

before(() => {

lib = new ensChart('container').durationTime(2000);

});

describe('new chart call the durationTime with 2000', () => {

it('expect 2000', () => {

expect(lib.durationTime()).to.eql(2000);

});

});

});

context('Test case 3.2', function () {

before(() => {

lib = new ensChart('container').durationTime(2000).durationTime('test');

});

describe('new chart call the durationTime with 2000 and then with an String', () => {

it('expect 2000', () => {

expect(lib.durationTime()).to.eql(2000);

});

});

});

});

### color.spec.js

/\* global describe, it, before \*/

let colorDefaultValue = ["#1f77b4", "#aec7e8", "#ff7f0e", "#ffbb78", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5", "#8c564b", "#c49c94", "#e377c2", "#f7b6d2", "#7f7f7f", "#c7c7c7", "#bcbd22", "#dbdb8d", "#17becf", "#9edae5"];

describe('Testing the Color Class', () => {

context("Test case 10.0", function () {

before(() => {

lib = new ensChart('colorContainer');

});

describe('calling the color function with no Parameter', () => {

it('expect the default value => "c7", "#bcbd22", "'+ colorDefaultValue +'" ==> (lib.colorScale(d3.schemeCategory20))', () => {

expect(lib.color()).to.eql(colorDefaultValue);

console.log(lib.color());

});

});

})

context("Test case 10.1", function () {

before(() => {

lib = new ensChart('colorContainer')

.color('#fff');

});

describe('calling the color function with a Hex Value as a String (#fff)', () => {

it('expect the given string as an array with a single value "["#fff"]"', () => {

expect(lib.color()).to.eql(['#fff']);

console.log(lib.color());

});

});

})

context("Test case 10.2", function () {

before(() => {

lib = new ensChart('colorContainer')

.color('#ZZGGAA');

});

describe('calling the color function with a false Hex Value as a String (#ZZGGAA)', () => {

it('expect the default valu', () => {

expect(lib.color()).to.eql(colorDefaultValue);

console.log(lib.color());

});

});

})

context("Test case 10.3", function () {

before(() => {

lib = new ensChart('colorContainer')

.color("ThisIsATest");

});

describe('calling the color function with a random String (ThisIsATest)', () => {

it('expect the default value', () => {

expect(lib.color()).to.eql(colorDefaultValue);

console.log(lib.color());

});

});

})

context("Test case 10.4", function () {

before(() => {

lib = new ensChart('colorContainer')

.color(9);

});

describe('calling the color function with a number (9)', () => {

it('expect the default value with an error message in the console', () => {

expect(lib.color()).to.eql(colorDefaultValue);

console.log(lib.color());

});

});

})

context("Test case 10.5", function () {

before(() => {

lib = new ensChart('colorContainer')

.color(['ThisIsATest', 'This too']);

});

describe('calling the color function with an random Array ("[\'ThisIsATest\', \'This too\'])', () => {

it('expect the default value', () => {

expect(lib.color()).to.eql(colorDefaultValue);

console.log(lib.color());

});

});

})

context("Test case 10.6", function () {

before(() => {

lib = new ensChart('colorContainer')

.color(['#666', '#abc']);

});

describe('calling the color function with an correct Array ("[\'#666\', \'#abc\'])', () => {

it('expect the given Array ([\'#666\', \'#abc\'])', () => {

expect(lib.color()).to.eql(['#666', '#abc']);

console.log(lib.color());

});

});

})

context("Test case 10.7", function () {

before(() => {

lib = new ensChart('colorContainer')

.color(['#666', '#zzz', '#ABC']);

});

describe('calling the color function with an partially correct Array ("[\'#666\', \'#zzz\', \'#ABC\'])', () => {

it('expect the given Array minus the false one ([\'#666\', \'#ABC\'])', () => {

expect(lib.color()).to.eql(['#666', '#ABC']);

console.log(lib.color());

});

});

})

context("Test case 10.8", function () {

before(() => {

lib = new ensChart('colorContainer')

.color(['#666', '#abc'])

.color(['ThisIsATest', 'This too']);

});

describe('calling the color function with an correct Array ("[\'#666\', \'#abc\']) and then a false one ("[\'ThisIsATest\', \'This too\'])', () => {

it('expect the given Array ([\'#666\', \'#abc\'])', () => {

expect(lib.color()).to.eql(['#666', '#abc']);

console.log(lib.color());

});

});

})

});

### ease.spec.js

/\* global describe, it, before \*/

describe('Testing the Ease Class', () => {

context('Test case 20.0', function () {

before(() => {

lib = new ensChart('easeContainer').ease(['test']);

});

describe('new chart call the ease with an Array', () => {

it('expect an error message in the console', () => {

expect(lib.\_cConfigEase.\_easeName).to.be.undefined;

});

});

});

context('Test case 20.1', function () {

before(() => {

lib = new ensChart('easeContainer').ease("Cubic");

});

describe('new chart call the ease with 2000', () => {

it('expect the \_easeName to be easeCubic', () => {

expect(lib.\_cConfigEase.\_easeName).to.eql('easeCubic');

});

});

});

context('Test case 20.2', function () {

before(() => {

lib = new ensChart('easeContainer').ease('Exp').ease('test');

});

describe('new chart call the ease with Exp and then with a number with an error message in the console', () => {

it('expect the \_easeName to be easeExp', () => {

expect(lib.\_cConfigEase.\_easeName).to.eql('easeExp');

});

});

});

context('Test case 20.3', function () {

before(() => {

lib = new ensChart('easeContainer').ease('easeExp');

});

describe('new chart call the ease with easeExp', () => {

it('expect the \_easeName to be easeExp', () => {

expect(lib.\_cConfigEase.\_easeName).to.eql('easeExp');

});

});

});

context('Test case 20.4', function () {

before(() => {

lib = new ensChart('easeContainer').ease('eAseExp');

});

describe('new chart call the ease with eAseExp', () => {

it('expect the \_easeName to be easeExp', () => {

expect(lib.\_cConfigEase.\_easeName).to.eql('easeExp');

});

});

});

});

### scale.spec.js

/\* global describe, it, before \*/

describe('Testing the Scale Class', () => {

context('Test case 30.0', function () {

before(() => {

lib = new ensChart('scaleContainer').scale(['test']);

});

describe('new chart call the scale with an Array', () => {

it('expect an error message in the console', () => {

expect(lib.\_cConfigScale.\_scaleName).to.be.undefined;

});

});

});

context('Test case 30.1', function () {

before(() => {

lib = new ensChart('scaleContainer').scale("Pow");

});

describe('new chart call the scale with Pow', () => {

it('expect the \_scaleName to be scalePow', () => {

expect(lib.\_cConfigScale.\_scaleName).to.eql('scalePow');

});

});

});

context('Test case 30.2', function () {

before(() => {

lib = new ensChart('scaleContainer').scale('Pow').scale('test');

});

describe('new chart call the scale with Pow and then with a number with an error message in the console', () => {

it('expect the \_scaleName to be scalePow', () => {

expect(lib.\_cConfigScale.\_scaleName).to.eql('scalePow');

});

});

});

context('Test case 30.3', function () {

before(() => {

lib = new ensChart('scaleContainer').scale('scalePow');

});

describe('new chart call the scale with scalePow', () => {

it('expect the \_scaleName to be scalePow', () => {

expect(lib.\_cConfigScale.\_scaleName).to.eql('scalePow');

});

});

});

context('Test case 30.4', function () {

before(() => {

lib = new ensChart('scaleContainer').scale('scAlePow');

});

describe('new chart call the scale with scAlePow', () => {

it('expect the \_scaleName to be scalePow', () => {

expect(lib.\_cConfigScale.\_scaleName).to.eql('scalePow');

});

});

});

});

### size.spec.js

/\* global describe, it, before \*/

describe('Testing the Size Class (width)', () => {

context('Test case 40.0', function () {

before(() => {

lib = new ensChart('sizeContainer').width(['test']);

});

describe('new chart call the width with an Array', () => {

it('expect an error message in the console', () => {

expect(lib.\_cConfigSize.width).to.eql('500px')

});

});

});

context('Test case 40.1', function () {

before(() => {

lib = new ensChart('sizeContainer').width(600);

});

describe('new chart call the width with 600', () => {

it('expect the width to be 600px', () => {

expect(lib.\_cConfigSize.width).to.eql('600px');

});

});

});

context('Test case 40.2', function () {

before(() => {

lib = new ensChart('sizeContainer').width('600px').width('test');

});

describe('new chart call the width with 600px and then with a number with an error message in the console', () => {

it('expect the width to be 600px', () => {

expect(lib.\_cConfigSize.width).to.eql('600px');

});

});

});

context('Test case 40.3', function () {

before(() => {

lib = new ensChart('sizeContainer').width('600');

});

describe('new chart call the width with 600 as string', () => {

it('expect the width to be 600px', () => {

expect(lib.\_cConfigSize.width).to.eql('600px');

});

});

});

context('Test case 40.4', function () {

before(() => {

lib = new ensChart('sizeContainer').width('600Px');

});

describe('new chart call the width with 600Px', () => {

it('expect the width to be 600px', () => {

expect(lib.\_cConfigSize.width).to.eql('600px');

});

});

});

});

/\* global describe, it, before \*/

describe('Testing the Size Class (height)', () => {

context('Test case 41.0', function () {

before(() => {

lib = new ensChart('sizeContainer').height(['test']);

});

describe('new chart call the height with an Array', () => {

it('expect an error message in the console', () => {

expect(lib.\_cConfigSize.height).to.eql('500px')

});

});

});

context('Test case 41.1', function () {

before(() => {

lib = new ensChart('sizeContainer').height(600);

});

describe('new chart call the height with 600', () => {

it('expect the height to be 600px', () => {

expect(lib.\_cConfigSize.height).to.eql('600px');

});

});

});

context('Test case 41.2', function () {

before(() => {

lib = new ensChart('sizeContainer').height('600px').height('test');

});

describe('new chart call the height with 600px and then with a number with an error message in the console', () => {

it('expect the height to be 600px', () => {

expect(lib.\_cConfigSize.height).to.eql('600px');

});

});

});

context('Test case 41.3', function () {

before(() => {

lib = new ensChart('sizeContainer').height('600');

});

describe('new chart call the height with 600 as string', () => {

it('expect the height to be 600px', () => {

expect(lib.\_cConfigSize.height).to.eql('600px');

});

});

});

context('Test case 41.4', function () {

before(() => {

lib = new ensChart('sizeContainer').height('600Px');

});

describe('new chart call the height with 600Px', () => {

it('expect the height to be 600px', () => {

expect(lib.\_cConfigSize.height).to.eql('600px');

});

});

});

});

### type.spec.js

/\* global describe, it, before \*/

describe('Testing the Type Class', () => {

context('Test case 50.0', function () {

before(() => {

lib = new ensChart('typeContainer').type(['test']);

});

describe('new chart call the type with an Array', () => {

it('expect an error message in the console', () => {

expect(lib.\_cConfigType.\_chartName).to.be.undefined;

});

});

});

context('Test case 50.1', function () {

before(() => {

lib = new ensChart('typeContainer').type("barchart");

});

describe('new chart call the type with barchart', () => {

it('expect the \_chartName to be barchart', () => {

expect(lib.\_cConfigType.\_chartName).to.eql('barchart');

});

});

});

context('Test case 50.2', function () {

before(() => {

lib = new ensChart('typeContainer').type('piechart').type(6);

});

describe('new chart call the type with piechart and then with a number with an error message in the console', () => {

it('expect the \_chartName to be piechart', () => {

expect(lib.\_cConfigType.\_chartName).to.eql('piechart');

});

});

});

context('Test case 50.3', function () {

before(() => {

lib = new ensChart('typeContainer').type('PIecharT');

});

describe('new chart call the type with PIecharT', () => {

it('expect the \_chartName to be piechart', () => {

expect(lib.\_cConfigType.\_chartName).to.eql('piechart');

});

});

});

context('Test case 50.4', function () {

before(() => {

lib = new ensChart('typeContainer').type('donutchart');

});

describe('new chart call the type with donutchart', () => {

it('expect the \_chartName to be donutchart', () => {

expect(lib.\_cConfigType.\_chartName).to.eql('donutchart');

});

});

});

});

### ensChart.js & ensChart.js.map

Wäre das komprimierte File mitsamt den Angängen da jedes dieses Dokument separat über 1800 Zeilen beinhaltet wird es in diesem Dokument nicht eingefügt.

1. Leitfaden\_Hermes5\_1.pdf [↑](#footnote-ref-1)
2. Ausgeschrieben «Source Code» (deutsch Quelltext oder Quellcode) [↑](#footnote-ref-2)
3. Leitfaden\_Hermes5\_1.pdf(26.04.2017) [↑](#footnote-ref-3)
4. https://de.wikipedia.org/wiki/SMART\_(Projektmanagement)(26.04.2017) [↑](#footnote-ref-4)
5. https://www.fin.be.ch [↑](#footnote-ref-5)
6. https://www.computerwoche.de/a/tipps-zur-qualitaetssicherung,1867513,4(30.05.2017 [↑](#footnote-ref-6)
7. https://www.ecma-international.org/ecma-262/6.0/ [↑](#footnote-ref-7)
8. https://babeljs.io/(05.05.2017) [↑](#footnote-ref-8)
9. http://eslint.org/docs/about/(05.05.2017) [↑](#footnote-ref-9)
10. https://de.wikipedia.org/wiki/Lint\_(Programmierwerkzeug)(05.05.2017) [↑](#footnote-ref-10)
11. http://eslint.org/docs/about/(05.05.2017) [↑](#footnote-ref-11)
12. http://mherman.org/blog/2015/09/10/testing-node-js-with-mocha-and-chai/#.WRAoN9ryguU [↑](#footnote-ref-12)
13. http://chaijs.com/ [↑](#footnote-ref-13)
14. https://mantro.net/en/blog/what-is-webpack [↑](#footnote-ref-14)
15. https://de.wikipedia.org/wiki/Programmierschnittstelle(08.05.2017) [↑](#footnote-ref-15)
16. https://github.com/krasimir/webpack-library-starter(08.05.2017) [↑](#footnote-ref-16)
17. http://krasimirtsonev.com/blog/article/javascript-library-starter-using-webpack-es6(08.05.2017) [↑](#footnote-ref-17)
18. https://www.google.ch/search?q=donutchart+deutsch&rlz=1C1CHBF\_deCH736CH736&oq=donutchart+deutsch&aqs=chrome..69i57.4222j0j4&sourceid=chrome&ie=UTF-8#q=Java+Script(09.05.2017) [↑](#footnote-ref-18)
19. https://de.wikipedia.org/wiki/Programmbibliothek(09.05.2017) [↑](#footnote-ref-19)
20. https://www.pkorg.ch/informatik/(09.05.2017) [↑](#footnote-ref-20)
21. https://de.wikipedia.org/wiki/Software\_Rendering(09.05.2017) [↑](#footnote-ref-21)