# Struktur / Gliederung der Software

### Regeln für das Verwalten der Software

* Die gesammte Software wird im Ordner „Lupus\_Station“ gespeichert
* Der Quellcode wird in Lupus\_Station/source gesichert
* Git nutzen für die Versionsverwaltungssoftware
  + Bei Konflikten immer zunächst mit dem Autor des Konfliktcodes absprechen, damit keine wichtiger Code überschrieben wird
* Verwende mehrere Source Dateien, damit das gemeinsame Arbeiten erleichtert wird
  + Setup/Replacement File
  + Story File (für jede Szene eine eigene Datei) (alphabetisch)
  + Datei mit der Deklaration der Gegenstände (evtl. splitten) (alphabetisch)
  + Datei mit der Deklaration der Räume und Geräusche (alphabetisch)
  + Datei mit der Deklaration der Türen (alphabetisch)
  + Evtl. sonst nur Dateien pro Szene wo alle Gegenstände die auftreten drin sind?
* Alle Programmierarbeiten im Trello protokollieren
* Das Coden erfolgt nach folgendem Schema

1. Aufgabenpakete sind alle im Trello gesammelt, anfangs unter TODO
   1. Jedes Paket wird min. zwei Mitgliedern zugeordnet, dem Hauptentwickler und einem Stellvertreter, der übernimmt, falls der Hauptentwickler ausfällt/Probleme hat
   2. Bei komplizierten Aufgabenpaketen können auch mehr Entwickler zugeordnet werden
2. Wenn ein Entwickler ein Aufgabenpaket programmieren will, muss er zunächst das Paket unter PROGRESS speichern und dazu kommentieren, dass er sich daransetzt, damit sofort Autor und Zeitpunkt ersichtlich sind. Falls der Entwickler nicht am selben Tag fertig wird, muss das in den Kommentaren vermerkt werden.
   1. Falls beim Coden auffällt, dass etwas in der Spezifikation nicht bzw. anders umgesetzt werden muss, so muss dies immer mit dem Team besprochen werden!
   2. Falls es Probleme/Unklarheiten beim Coden gibt, muss das sofort dem Team mitgeteilt werden, damit schnell geholfen werden kann
3. Bei Abschluss wird das Aufgabenpaket unter TESTING abgelegt (und evtl. spezielle Testfälle werden dazu geschrieben)
   1. Falls für das Testen weitere Aufgabenpakete benötigt werden, bleibt das Paket solange unter TESTING gespeichert, bis alle nötigen Aufgabenpakete auch unter TESTING sind, sodass alle Pakete zusammen getestet werden können
4. das Aufgabenpaket wird bzw. (falls mehrere voneinander abhängig sind) die Aufgabenpakete werden von einem/mehreren Teammitgliedern getestet und anschließend unter DONE gespeichert
   1. beim Testen muss immer auch ein Teammitglied dabei sein, dass nicht an der Entwicklung des Pakets beteiligt war
   2. Falls der Code durch einen anderen Entwickler debuggt wird, muss der „fehlerhafte“ Code auskommentiert werden und nicht gelöscht werden und es muss immer im Trello kommentiert werden, was genau verändert wurde (bzw. wo), sodass der ursprüngliche Entwickler schnell einschreiten kann, falls der auskommentierte Code doch noch benötigt wird
5. Falls im weiteren Verlauf Änderungen am Code eines Pakets vorgenommen werden müssen, muss das Aufgabenpaket wieder unter TESTING abgelegt werden und muss erneut getestet werden (falls das Paket von anderen Paketen abhängt oder umgekehrt, müssen alle abhängigen Pakete getestet werden)

### Coding Richtlinien

* Es ist stets so zu programmieren, dass der Code schnell durch andere verstanden werden kann, also keine unnötige Komplexität
* Alle Variablennamen für Gegenstände/Räume/Türen werden aus der Spezifikation entnommen
  + Um Verwirrung zu vermeiden, sind komplizierte Namen (z.B. Räume des Solarmoduls, 2 Transporter, etc.) mit Kommentaren zu versehen, die kurz erläutern
  + Variablen die nicht in der Spezifikation stehen (z.B. temporäre Variablen in Schleifen) sollten sinnvoll benannt werden (sinnvoll, keine nichtssagenden Namen, nicht zu lang) und sind mit Kommentaren zu versehen
* Alle Dateien besitzen einen Datei-Header mit Namen des/der Autor/en, letztes Änderungsdatum und kurze Beschreibung wofür die Datei genutzt wird
* Jeder Code Abschnitt muss mit einem Kommentar versehen werden, in dem beschrieben ist, wer den Code schreibt, was der Code macht, wo in der Spezifikation die Funktion zu finden ist und evtl. auch das zugehörige Aufgabenpaket
  + Der Code muss immer mit der Spezifikation übereinstimmen
  + Das Ende des Code Abschnitts muss auch kommentiert werden mit einem „Ende von Code Abschnitt XY“
* Einzelne Zeilen sind zu kommentieren, falls der darin enthaltene Code verwirrend ist (bspw. Bei komplexen Schleifen mit mehreren Abbruchkriterien), damit eine kurze Beschreibung der Zeile geben ist
* Zeilen sollten nicht länger als 120 Zeichen sein
* Einrückungen sind immer zu nutzen, sobald sie sinnvoll erscheinen
  + Keine Tabs, sondern Leerzeichen