Bei zunehmender Größe können Programmierprojekte schnell unübersichtlich und schwierig zu verstehen sein. Um diesem Problem entgegen zu wirken, wird meist eine separate Dokumentation zum Code erstellt, die diesen genauer erklärt. Diese Trennung zwischen Code und Dokumentation kann jedoch dazu führen, dass es schwerfällt, ein wirkliches Verständnis für den Code zu entwickeln, da häufig nur wenige kurze Sätze über die Nutzung einer bestimmten Funktionalität in der Dokumentation stehen, wodurch Unklarheiten entstehen können. Um dieses Problem zu beheben, gibt es Ansätze, bei denen der Code und die Dokumentation in der gleichen Datei zu finden sind, sodass ein Programm übersichtlicher und schrittweise am tatsächlichen Quellcode erklärt werden kann. Einer dieser Ansätze ist die Notizbuchprogrammierung, an die u.a. das LiveViewProgramming angelehnt ist. Dabei können, in beliebiger Reihenfolge, Dokumentationstext und Quellcode in der gleichen Datei untergebracht werden, sodass geschriebener Code direkt in einer Markup-Sprache wie Markdown erklärt werden kann. Der Code kann dabei auch Schrittweise ausgeführt werden, sodass in Gebieten wie Data Science, in denenhäufig mit solchen Notizbüchern gearbeitet wird, große Datensätze direkt im Notizbuch geplottet werden können.
Ein anderer, aber ähnlicher Ansatz ist das sogenannte Literate Programming, das in den 80er Jahren von Donald E. Knuth erdacht wurde. Hierbei wird der Code als Teil eines größeren Dokuments betrachtet, das sowohl den Programmablauf und Code als auch Erläuterungen dazu enthält. Ein wichtiges Hilfsmittel im Literate Programming ist die Verwendung von "Makros". Diese dienen Dazu, einzelne Codeabschnitte, die zusammenhängen unter einer Überschrift zusammenzufassen und somit die Struktur des Codes übersichtlicher zu gestalten. Durch die Verwendung von Makros können komplizierte Prozesse in verständliche, kleinere Einheiten unterteilt werden, was Lesbarkeit und Wartbarkeit des Codes verbessert. Damit der Code auch ausführbar bleibt, werden aus dem geschriebenen Quellcode zwei unterschiedliche Dateien generiert: Eine vollständig dokumentierte Fassung, die über die Funktion "weave" entsteht, und eine Datei, die über den Befehl "tangle" generiert wird und nur den ausführbaren bzw. Compilierbaren Programmcode in der eigentlichen Programmiersprache enthält.

Ziel der Arbeit soll es sein, eine Möglichkeit zur Generierung dieser beiden Dateitypen zu implementieren und im Anschluss das Literate Programming mit der Idee im LiveViewProgramming zu vergleichen und zu untersuchen, inwiefern sich diese kombinieren lassen, sodass bspw. Die Datei, die durch *tangle* generiert wird, solch ein Format einnimmt, dass diese direkt im LiveViewProgramming angezeigt werden kann.