Notizen:

* Erstellen des Projekts
* Erstellen der GitRepository
* Informationsbeschaffung zu neuen Klassen
* Erstellen des JFrame
* Erstellen des JPanel
* Erstellen der Zeichenfläche mit paintComponent und Konstruktor Panel
* Ändern der Farbe mit Buttons
* MouseAdapterKlassen hinzugefügt
* Pinsel hinzugefügt
* Linie, Rechteck, Ellipse hinzugefügt
* Radierer überarbeitet
* Strichdicke hinzugefügt
* Anpassbare Blattgröße beim Starten
* Neues Blatt hinzugefügt
* Farben und Tools keine Arrays, um symbolisch programmieren zu können
* Erstellung der Buttons reorganisiert
* Laden und Speichern von Bildern 🡪 Standardverzeichnis im Projektordner (gitignore angepasst)
* Neu Funktion erzeugt kein neues panel mehr
* Tooltips hinzugefügt
* Neue Farben hinzugefügt
* Redo-Funktion gestrichen 🡪 neues Werkzeuge-Menü
* Neu-Funktion überarbeitet.. arbeitet jetzt auch bei kleinen Fenstern
* Scrollbars implementiert
* Kommentare überarbeitet
* Buttons geändert (keine Variable mehr)
* Klassenvariablen teilweise zu lokalen Variablen geändert
* Fehlermeldungen bearbeitet
* Symbolleiste überarbeitet
* Icons erstellt/hinzugefügt.. überflüssige icons gelöscht
* ButtonGroups zur markierung von buttons erstellt (JButton 🡪 JToggleButton)
* Vorschau für Linien, Rechtecke und Ellipsen hinzugefügt
* Sicherheitsabfrage beim Erstellen einer neuen Arbeitsfläche und beim Laden
* Neuer Button „Speichern unter“
* Testing und Fehlersuche

**1. Qualitätssicherungsdokumentation**

*Zweck:*  
Die Qualitätssicherungsdokumentation beschreibt die Maßnahmen, Strategien und Standards, die während der Entwicklung eingehalten werden, um eine hohe Softwarequalität sicherzustellen. Sie legt fest, wie die Einhaltung von Qualitätskriterien überwacht wird.

*Inhalt:*

* **Entwicklungsrichtlinien:** z.B. Code-Standards, Namenskonventionen, Dokumentationsrichtlinien (wie Kommentare im Code).
* **Designprinzipien:** z.B. Modularität, Wiederverwendbarkeit, Wartbarkeit (z.B. durch Clean Code).
* **Qualitätskriterien:** z.B. Funktionalität, Benutzerfreundlichkeit, Stabilität, Performance.
* **Code-Reviews:** Verfahren zur Überprüfung des Codes durch Teammitglieder oder automatisierte Tools.
* **Versionskontrolle:** Nutzung von Git oder ähnlichen Systemen inklusive Branching-Strategien.
* **Fehler- und Änderungsmanagement:** Prozesse zur Nachverfolgung und Behebung von Fehlern.
* **Werkzeuge und Methoden:** z.B. statische Codeanalyse, Linter, Continuous Integration (CI).

*Ziel:*  
Sicherstellen, dass das Projekt den Qualitätsanforderungen entspricht und systematisch kontrolliert wird.

**2. Testdokumentation**

*Zweck:*  
Die Testdokumentation dokumentiert die Planung, Durchführung und Auswertung der Tests am Programm. Sie zeigt auf, ob das Programm die Anforderungen erfüllt und fehlerfrei funktioniert.

*Inhalt:*

* **Testplan:** Zielsetzung der Tests, zu testende Funktionen (z.B. Zeichnen von Ellipsen), Testmethoden (z.B. manuelle Tests), Verantwortlichkeiten.
* **Testfälle:** Beschreibung einzelner Tests inklusive Eingaben, erwarteter Ergebnisse und tatsächlicher Ergebnisse. Beispiel: "Zeichnen einer Linie mit Maus – Ergebnis: Linie erscheint korrekt."
* **Testumgebung:** Hardware-, Software-, Betriebssystemeinstellungen sowie verwendete Tools (z.B. JUnit für Unit-Tests).
* **Testergebnisse:** Dokumentation der Testergebnisse inklusive gefundener Fehler oder Abweichungen vom Sollzustand.
* **Fehlerberichte:** Detaillierte Beschreibung gefundener Bugs mit Schritten zur Reproduktion und Status (offen/behoben).
* **Abnahmetests / Freigabe:** Dokumentation darüber, ob das Programm den Anforderungen entspricht und freigegeben wurde.

*Ziel:*  
Nachweis über die Funktionstüchtigkeit des Programms sowie Nachvollziehbarkeit der durchgeführten Tests.