ネットワークプログラミング応用演習

2132023番

内山　裕太

1. 目的

講義内で学んだ技術を再確認し，javaプログラミングの技術を向上させ，

本講義の総まとめをする．

1. 課題

講義中に配布されたプログラムをもとに，

第10回講義資料P140中の1,2,3,4,5,6,7,9を実装した．

また，オリジナル拡張として「すべてのオブジェクトの一括削除」

「サンプル描画の表示」を実装した．

1. 理論

楕円を描画する「Dotクラス」を作成するにあたり，第10回講義資料

P141,142を参照した．また，ファイル名を指定しての保存と読込を実装するにあたり，「FileDialogクラス」「Toolkitメソッド」について調査した．参考文献については本レポートの最後に明記する．

1. プログラム（ソースコード）

ソースコードは別途，zipファイルにて添付する．

今回用いた技術をソースコードの内容に合わせて記述する．

なお，配布プログラムの改良であるため，最後に変更点をまとめて記述する．また，本プログラム中に用いるSample描画3つもzipファイルにまとめる．

* 1. Java.awt.Toolkit

アプリケーションのウィンドウサイズを取得するために用いた．

Paint4クラス38行目にて，Toolkitの取得を行う．39,40行目にて，実行している端末のウィンドウサイズを取得し，縦横の大きさを変数に格納している．今回は4/5倍の大きさでプログラムが実行するようにしている．

加えて，縦横の変数を各ボタンの配置にも用いている．

* 1. チェックボックスグループ

今回は3つのチェックボックスを宣言した．

* + 1. 描画する図形選択（cbg1）

1つ目のチェックボックスは，選択されたものに応じて描画する図形を変更するものである．Paint4クラスの60~78行目で作成している．

同クラスの239行目にて，チェックボックスから取得した値に応じたクラスを呼び出すように設計している．

* + 1. 図形のカラー選択（cbg2）

2つ目のチェックボックスは，選択された色に応じて描画する図形の色を変更するものである．Paint4クラスの80~116行目で作成している．

同クラスの237~381行目で各クラスを呼び出す際に，cbg2にて選択された色に応じた数値を各クラスに第1引数として渡し，図形に色を付けるようにしている．

* + 1. 図形のタイプ選択（cbg3）

3つ目のチェックボックスは，選択されたタイプに応じて描画する図形の塗りつぶしを選択できる．Paint4クラスの119~125行目で作成している．同クラスの237~381行目で各クラスを呼び出す際に，cbg3にて選択されたタイプに応じた数値を各クラスに第2引数として渡し，図形の塗りつぶしを選択できるようにしている．ただし，「Lineクラス」

「Straightクラス」は塗りつぶし機能はないため，第1引数のみである．

* 1. FileDialogクラス

FileDialogクラスを用いることで，任意の名前のファイルの保存，

読込を可能にした．Paint4クラス208~215行目にファイルの

保存，読込のボタンが押された場合の処理を記載している．

* 1. その他の変更点
     1. Paint4クラス
        1. Undoボタン

「Undo」ボタンを作成し，描画した図形を1つずつ削除できるようにしている（196~201行目）．ボタンが押された際に，objListの最新要素のみを削除するようにしている．その際に，subListを作成し，図形を削除する直前にsubListに同じ図形を追加するようにしている．これは他の機能を実装するための準備である．

* + - 1. Redoボタン

「Redo」ボタンを作成し，直前に削除した図形をもとに戻すことが

できるようにしている（202~207行目）．ボタンが押された際に，objListにsubListの最新要素を追加するようにし，削除した図形を元に戻す機能として実装している．

* + - 1. Sampleボタン

「Sample」ボタンを作成し，3種類のサンプル描画をランダムで

表示するようにしている（219~223行目）．

あらかじめ「sample」「sample2」「sample3」を同じワークスペース

フォルダに保存しておく．1~3の数値を乱数で発生させ，結果に応じたサンプル描画が画面に表示される．ただし，サンプルボタンを押す行為は戻るボタンで戻ることができないため注意が必要である．

* + - 1. Deleteボタン

「Delete」ボタンを作成し，描画した図形すべてを削除することができるようにしている（217,218行目）．

* + 1. Dotクラス

楕円を描画できるクラスである．与えられた引数に応じて「色」

「塗りつぶし」を選択する．また，横長と縦長の楕円に対して場合分けを行った．

* + - 1. Math.abs（絶対値）

座標が負の値になることを考慮し，Math.absメソッドを用いた．

これにより，楕円の式の複雑化を防いだ．

* + 1. Circleクラス

円を描画できるクラスである．与えられた引数に応じて「色」

「塗りつぶし」を選択する．

* + 1. Rectクラス

四角形を描画できるクラスである．与えられた引数に応じて「色」

「塗りつぶし」を選択する．移動座標が負の値をとることを考慮し，

あらかじめ式の値にマイナスをかけるようにしている．

* + 1. Polygonクラス

三角形を描画できるクラスである．与えられた引数に応じて「色」

「塗りつぶし」を選択する．

* + 1. Lineクラス

折れ線を描画するクラスである．マウスがプレスされた場所と離された場所を対角とする長方形内に，折れ線を描画する．与えられた引数に応じて「色」を選択する．

* + 1. Straightクラス

直線を描画するクラスである．マウスがプレスされた場所と離された場所を結ぶ直線を描画する．与えられた引数に応じて「色」を選択する．

1. 動作確認手法・環境

今回の動作環境として，eclipseを用いた．

以下，各課題の確認方法を記述する．

* 1. 課題1，7

プログラムを実行した際に右にチェックボックスが表示されるかを確認する．選択した色とタイプに応じた描画がされているかを確認する．

* 1. 課題2，3，4

縦と横の移動座標がプラスとマイナスになる4パターンで描画し，すべてで描画できるかを確認する．課題2，3については，マウスカーソルが

図形の周上に来ているかを確認する．

* 1. 課題5，6，Delete（オリジナル）

各ボタンを押した際に処理が行われるかを確認する．

Undoボタンは，直前の描画が消えれば動作していることになる．

Redoボタンは，Undoボタンで直前に削除した描画が戻れば動作していることになる．

Deleteボタンは，画面上のすべての描画が削除されれば動作していることになる．

* 1. Sample（オリジナル）

Sampleボタンを押した際にサンプル描画が表示されるかを確認する．

3種類の描画がランダムに出現するかを確認する．今回は5回ボタンを押して検証する．

* 1. 課題9

Saveボタンを押した際に任意の名前でファイルを保存できることを確認する．また，Loadボタンを押した際に任意の名前のファイルを開くことができることを確認する．

1. 実行結果
   1. 課題1，7

背景パターン が含まれている画像

自動的に生成された説明図形, 円

自動的に生成された説明図形, 円

自動的に生成された説明図形, 円

自動的に生成された説明図形, 円

自動的に生成された説明

これらは，図形描画を行った際のスクリーンショットである．

選択された図形の描画ができていることが確認できる．また，選択した色と塗りつぶしタイプが適応していることがわかる．

* 1. 課題2，3，4

グラフ, 箱ひげ図

自動的に生成された説明

これは，中央の黒点を始点として4つの長方形を描画している様子の

スクリーンショットである．移動座標がマイナスになる場合も問題なく描画できていることがわかる．

* 1. 課題5，6，Delete（オリジナル）

図形, 円

自動的に生成された説明図形, 円

自動的に生成された説明図形, 円

自動的に生成された説明背景パターン が含まれている画像

自動的に生成された説明

これらは，Undo，Redo，Deleteボタンを各1回ずつ押した様子のスクリーンショットである．Undoを押すことにより，描画されている図形が

削除されていることを確認できる．また，Redoを押すことで，消えた図形が元に戻ることも確認できる．Deleteを押すと図形がすべて消えていることも確認できる．

* 1. Sample（オリジナル）

バブル チャート

自動的に生成された説明ダイアグラム が含まれている画像

自動的に生成された説明ロゴ

中程度の精度で自動的に生成された説明バブル チャート

自動的に生成された説明ロゴ

中程度の精度で自動的に生成された説明

これらは，Sampleボタンを5回押した実行結果のスクリーンショットである．ランダム表示ではあるが，今回は3種類とも確認することができた．Sampleボタンが動作していることを確認できた．

* 1. 課題9

グラフィカル ユーザー インターフェイス が含まれている画像

自動的に生成された説明グラフィカル ユーザー インターフェイス

低い精度で自動的に生成された説明コンピューターのスクリーンショット

自動的に生成された説明

これらは，Saveボタンを押した際のスクリーンショットである．

今回はSample描画を用いた．1枚目はSaveボタンを押した際のものである．今回は2枚目のように「practice」という名のファイルで保存する．

すると，3枚目のようにワークスペース内に「practice」ファイルが保存されたことを確認できた．任意の名前で描画をファイル保存できていることを確認できた．

グラフィカル ユーザー インターフェイス

自動的に生成された説明グラフィカル ユーザー インターフェイス

自動的に生成された説明ロゴ

中程度の精度で自動的に生成された説明

また，これらはLoadボタンを押した際のスクリーンショットである．

1枚目はLoadボタンを押した直後である．今回は，先ほど保存した

「practice」ファイルを読み込む．2枚目のようにファイル名を入力し，

開いた結果が3枚目である．先ほど保存した「practice」ファイルを再度開くことができた．任意の名前のファイルを開くことができることを確認できた．

1. まとめ

今回のプログラムは，本演習の総まとめとなるものであった．様々な未知のクラスやメソッドを用いたが，理解しきれずに用いたものも存在する．

しかし，レポート1では理解しきれていなかったものである

「オーバーライド」や「インスタンス」，今回初めて用いたものである

「チェックボックス」や「FileDialog」など，理解が大変深まったものが多い．本演習の序盤では「C言語より難しい，わかりづらい」と思っていたが，今では「基本的な構造は同じであり，文法などが異なるだけだ」と感じている．これは，プログラムの基礎能力の上昇を意味していると考えられ，本学習で得られた大きな成長であると考える．

1. 感想

Java言語は多くの場面で用いられており，これからも多くの場面で出会うことだろう．私自身が興味を持っているゲーム開発でも用いられている言語である．iPhoneユーザーである以上は別の言語の学習も必要となるが，冬季休業を用いて，javaプログラミングの知識をさらに深めようと感じた．

1. 参考文献（作者名敬称略）

題名：クラスjava.awt.Toolktの使用

URL：<http://www.kusa.ac.jp/~kajiura/javadocs/java/awt/class-> use/Toolkit.html

最終閲覧日：2022/12/13

作者：不明

題名：java.awt.FileDialog主なメソッド

URL：<http://memories.zal.jp/WP/blog/20110228_1967.html>

最終閲覧日：2022/12/13

作者：サイト構築日記

題名：レポートの書き方

URL：<https://medium.com/@geckyokawa/42a58a0a48b3>

最終閲覧日：2022/12/13

作者：大川　茂樹

題名：JavaDrive－IT技術全般の学習サイト

URL：<http://javadrive.s25.xrea.com/java2d/shape/index9.html>

最終閲覧日：2022/12/13

作者：Tatsuo Ikura