Examination
User Interaction (UI)

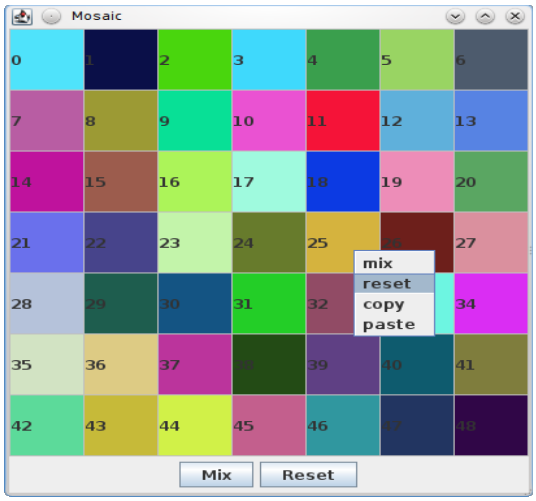
Prof. Dr.-Ing. Christian Heller <christian.heller@ba-leipzig.de>

Personal Data	
First and Last Name	Ihre Daten werden von der Klausuraufsichtsperson schriftlich auf der Anwesenheitsliste festgehalten.
Matriculation Number	
Subject and Year	CS 2013
Login	Das Login "klaus..." muss unbedingt schriftlich auf der Anwesenheitsliste festgehalten werden, da sonst keine Zuordnung des Logins zu Ihrem Namen und damit keine Korrektur der Klausur möglich ist!

Examination Data	
Date	2015-03-24
Duration [min]	100
Permitted Study Aids	Dokumentation im lokalen Netzwerkverzeichnis (Intranet); NICHT gestattet sind Kommunikationsmöglichkeiten (Internet) oder Anmeldung via SSH auf dem Rechner "fileserv", wo Ihr Homeverzeichnis liegt, oder eine Anmeldung mit Ihrem Klausur-Login nach Ende der Prüfung. Dies kann leicht geprüft werden (last cs12*, Server-Log-Dateien). Bitte unterlassen Sie also Täuschungsversuche in Ihrem eigenen Interesse.
Remarks	Hinterlegen Sie alle Programme und Antworten in elektronischer Form! Es wird kein Papier angenommen. Möchten Sie Lösungen erläutern, so nutzen Sie Quelltext-Kommentare oder legen eine Text-Datei an. Speichern Sie sämtliche Daten im HOME-Verzeichnis des Nutzers, d.h. unter Windows auf Laufwerk H:\ (NICHT auf C:\ oder "Eigene Dateien")! Idealerweise legen Sie dort ein Unterverzeichnis namens "klausur" an. Lesen Sie die Aufgaben komplett durch, bevor Sie sie lösen! Die Reihenfolge der Lösung ist Ihnen überlassen. Probieren Sie immer, eine Aufgabe zu lösen, da auch auf richtige Teile nicht vollständiger Lösungen Punkte vergeben werden! Falls vom Prinzip her richtig, so werden auch alternative Lösungen akzeptiert. Sie dürfen beliebig viele Bildschirmausgaben von Werten in den Quelltext einbauen, um ein Programm besser nachvollziehen zu können. Diese Aufgabenstellung in Papierform können Sie nach dem Ende der Klausur behalten.

Evaluation						
Task	1	2	3	4	5	Summe
Points	10	10	35	20	25	100

Im Verlauf der Klausur soll eine Anwendung erstellt werden, die ein dem folgenden Bildschirmschnappschuss ähnliches Aussehen hat. Sinn und Zweck sind Farbspielereien:



Task 1: Swing Framework [10]

Es wird eine Anwendung auf Basis der *Swing*-Klassenbibliothek erstellt, welche ein grafisches Rahmenfenster anzeigt.

a) Erstellen Sie eine Klasse namens *MainFrame*, welche von einer geeigneten *Swing*-Rahmenfenster-Klasse erbt! Fügen Sie eine Methode namens *initialise* ein! Geben Sie darin dem Fenster den Titel *Mosaic*! Lassen Sie seine Größe automatisch optimal so bestimmen, dass alle enthaltenen Komponenten dargestellt werden! Aktivieren Sie einen Schließvorgang, der bewirkt, dass sich die Anwendung beendet, sobald das Rahmenfenster geschlossen wird! Schalten Sie das Fenster sichtbar! [4]

b) Erstellen Sie zwei Tafeln namens *colourPanel* und *buttonPanel* vom Typ *JPanel*! Legen Sie die bevorzugte Größe des *colourPanel* quadratisch auf *400 Pixel* fest! Setzen Sie seine Hintergrundfarbe auf *hellgrau*! Fügen Sie beide Tafeln zur *content pane* des Rahmenfensters hinzu, wobei sich das *buttonPanel* am unteren Rand befinden soll! [4]

c) Erstellen Sie schließlich eine *main*-Methode, in welcher eine Instanz der *MainFrame*-Klasse erzeugt und woran ihre *initialise*-Methode aufgerufen wird! [2]

Task 2: Layout Manager [10]

Hier werden Kacheln mit zufälligen Farben platziert.

a) Erstellen Sie innerhalb der *initialise*-Methode eine Instanz eines *Layout Managers*, der Komponenten gitternetzartig anordnet, wobei jeweils sieben Zeilen und Spalten verwendet werden! Weisen Sie ihn dem *colourPanel* zu! [4]

b) Geben Sie der *MainFrame*-Klasse ein Attribut namens *labels*, welches ein *Array* mit Elementen vom Typ *JLabel* enthalten soll! Erstellen und befüllen Sie es innerhalb der *initialise*-Methode mit *49* gültigen Objekten! Weisen Sie dabei jedem Objekt als Textaufschrift eine eindeutige Zahl zwischen *0* und *48* zu! Deaktivieren Sie via *setOpaque*-Methode die Transparenz der Objekte! (Sonst werden farbige Hintergründe später nicht korrekt angezeigt.) Fügen Sie jedes neu erzeugte *JLabel*-Objekt zum *colourPanel* hinzu! [4]

c) Erstellen Sie innerhalb der *initialise*-Methode ebenfalls zwei Knöpfe, einen mit der Aufschrift *Mix* und den anderen mit der Aufschrift *Reset*! Fügen Sie beide dem *buttonPanel* hinzu! [2]

Task 3: Model View Controller (MVC) [35]

Die Datenverwaltung (hier lediglich Farbwerte) wird in eine *Model*-Klasse ausgelagert.

a) Erstellen Sie eine neue Klasse namens *Model* und geben Sie ihr folgende zwei Attribute:

```
private Color[] colours = null;
public Color storedColour = null;
```

Ergänzen Sie die Zugriffsmethode zum Holen des *colours*-Attributes! [2]

b) Erstellen Sie außerdem eine *getRandomColour*-Methode, welche ein Farbobjekt mit zufälligen *Rot-Grün-Blau*-Farbwerten erzeugt und zurück gibt! Erstellen Sie desweiteren eine Methode namens *mixOneColour*, welche einen Parameter vom Typ *int* entgegennimmt! Ordnen Sie durch Aufruf der *getRandomColour*-Methode dem *colours-Array* Element, auf das der als Parameter übergebene Index verweist, einen Farbwert zu! [8]

c) Erstellen Sie schließlich auch eine *mixAllColours*-Methode, in welcher die *mixOneColour*-Methode innerhalb einer Schleife für jedes der *49* Elemente des *colours*-Arrays aufgerufen wird! In ähnlicher Weise ist eine *resetAllColours*-Methode zu erstellen, in welcher alle Farbobjekte des *colours-Arrays* auf den Farbwert *weiß* gesetzt werden! [6]

d) Nun soll die *Model*-Klasse noch eine Methode namens *copyColour* erhalten, an welche bei Aufruf zwei Parameter zu übergeben sind. Der erste Parameter vom Typ *int* soll als Index dienen; der zweite vom Typ *Color* soll ein Farbobjekt enthalten. Innerhalb der Methode ist das Farbobjekt dem *colours-Array* an der übergebenen Indexposition zuzuweisen! [4]

e) Was noch fehlt, ist eine *initialise*-Methode für die *Model*-Klasse. In ihr soll der *colours Container* erzeugt werden. Anschließend ist die *mixAllColours*-Methode aufzurufen! [4]

f) Geben Sie der *MainFrame*-Klasse ein Attribut namens *model* vom Typ *Model* mit der Sichtbarkeit *public*! Überschreiben Sie die von *java.awt.Component* geerbte *paint*-Methode! Holen Sie sich darin via *model*-Attribut mittels *getColours*-Methode das *Array* mit Farbobjekten und setzen Sie die Hintergrundfarbe aller *49* Elemente des *labels*-Attributes entsprechend dieser *49* Farben! [6]

g) Instanziieren Sie die *Model*-Klasse in der *main*-Methode und weisen Sie die erhaltene Objektreferenz der *MainFrame*-Instanz' *model*-Eigenschaft zu! Vergessen Sie nicht, die *initialise*-Methode der *Model*-Instanz aufzurufen! [5]

Task 4: Event Handling [20]

Die Verarbeitung von Ereignissen soll durch eine spezielle *Controller*-Klasse geschehen.

a) Erstellen Sie eine Klasse namens *Controller* und geben Sie ihr zwei Attribute der Sichtbarkeit *public*: *model* vom Typ *Model* und *view* vom Typ *MainFrame*. Lassen Sie die Klasse von der *ActionListener*-Schnittstelle erben und implementieren Sie die geforderte Methode! [5]

b) Geben Sie der *MainFrame*-Klasse ein Attribut namens *controller* vom Typ *Controller*! [2]

c) Instanziieren Sie die *Controller*-Klasse in der *main*-Methode und weisen Sie die Instanz der entsprechenden Eigenschaft der *MainFrame*-Instanz zu! Weisen Sie auch den beiden Eigenschaften *model* und *view* der *Controller*-Instanz die jeweiligen Objektreferenzen zu! [3]

d) Ergänzen Sie die *initialise*-Methode der *MainFrame*-Klasse in der Weise, dass den beiden Knöpfen das *controller*-Objekt als *ActionListener* zugeordnet wird! Weisen Sie den beiden Knöpfen außerdem ein eindeutiges *ActionCommand* zu! [4]

e) Ergänzen Sie die *actionPerformed*-Methode der *Controller*-Klasse so, dass die von den Knöpfen versandten Ereignisse abgefangen werden und passend reagiert wird! Verwenden Sie dazu die Methoden *mixAllColours* und *resetAllColours* der *Model*-Klasse! Hinweis: Vergessen Sie nicht, am Ende die *repaint*-Methode am *view*-Attribut aufzurufen, damit sich das Fenster neu zeichnet! [6]

Zusatzaufgabe: Definieren Sie die *action commands* als Konstanten der *Controller*-Klasse! [2]

Task 5: Popup Menu [25]

Ein *Popup*-Menü soll als Alternative zu den Knöpfen angeboten werden. Außerdem sollen darüber Farbwerte von einem Feld zu anderen kopiert werden können. Desweiteren soll sich bei einem Klick mit der linken Maustaste der Farbwert des aktuellen Feldes ändern.

a) Geben Sie der *MainFrame*-Klasse ein Attribut vom Typ *JPopupMenu*! Fügen Sie eine neue Methode namens *buildMenu* hinzu, in welcher das *Popup*-Menü bestehend aus den vier Einträgen: *Mix, Reset, Copy, Paste* zusammengebaut wird!

Hinweis: Vergessen Sie nicht, den *action listener* und auch *action commands* zuzuordnen! [5]

b) Ergänzen Sie die *initialise*-Methode durch einen Aufruf der *buildMenu*-Methode! Ordnen Sie außerdem jeder erzeugten *JLabel*-Instanz das *controller*-Attribut als *mouse listener* hinzu, damit Ereignisse des *Popup*-Menüs später abgefangen werden können! [5]

c) Geben Sie der *Controller*-Klasse zwei neue Attribute: *index* vom Typ *int* und *colour* vom Typ *Color*, jeweils mit der Sichtbarkeit *private*. Fangen Sie in der *actionPerformed*-Methode auch die Ereignisse für *copy* und *paste* ab! Weisen Sie im Falle des Kopierens (*copy*) den aktuellen Wert des *colour*-Attributes der *Controller*-Klasse dem *storedColour*-Attribut der *Model*-Klasse zu! Dies ist wichtig, damit die Einfügen (*paste*)-Funktion später den vorherigen Farbwert zur Verfügung hat. Rufen Sie im Falle des Einfügens (*paste*) die *copyColour*-Methode der *Model*-Klasse auf! [5]

d) Lassen Sie die *Controller*-Klasse von der Klasse *MouseAdapter* erben und implementieren Sie die *mouseClicked*-Methode! Bestimmen Sie die Quelle des Ereignisses! Prüfen Sie mittels *instanceof*-Operator, ob es sich um eine Instanz von *JLabel* handelt! Falls ja, so bestimmen Sie zum Einen den Aufschriftentext des angeklickten *Labels* und zum Anderen seine aktuelle Hintergrundfarbe! Unterscheiden Sie nun, welcher Mausknopf gedrückt wurde! War es der linke, so soll sich der Farbwert des angeklickten Feldes ändern. (Hinweis: Methode *mixOneColour* und Neuzeichnen des Fensters.) War es der rechte, so soll das *Popup*-Menü angezeigt werden. Speichern Sie in letzterem Falle auch den (weiter oben bereits bestimmten) Index und Farbwert des aktuellen *Labels* in den Attributen *index* bzw. *colour* der *Controller*-Klasse! [10]

Viel Erfolg!