**שלב 5 - תמיכה בצבע\אור, הוספת סכמה, ובניית תמונה עם תאורת מילוי**

לפני הכל נקפיא את השלב הקודם באתר **github** ע"י יצירת **branch** חדש בשם **recitation4**. לאחר מכן נוודא שענף ברירת המחדל נשאר **master**.

***נ.ב.1*** *אין לבנות בנאי ברירת מחדל, בנאי העתקה, והעמסת* ***equals*** *במחלקות של השלב הזה.*

***נ.ב.2*** *חובה להקפיד על תיעוד בפורמט* ***javadoc*** *לכל המחלקות והפעולות החדשות ולעבוד ע"פ הסדר כפי שנלמד בקורס!*

**שלב 1 - הכנות**

קודם כל ניקח מתיבת ההגשה שני מודולים עם המחלקות המתאימות:

* המודול **Color.java** מכיל מחלקה **Color** שעוטפת (**Wrapper design pattern**) את החמלקה של ג'אוה פריימוורק **java.awt.Color** ומספקת עבורנו את הממשק הדרוש לנו במיני-פרויקט. את המודול הזה נוסיף בחבילת **primitives**.
* המודול **ImageWriter.java** מכיל מחלקה שאחראית על שלושה תחומים:
  + הגדרת תיקיה ושם הקובץ של התמונה ופעולה **writeToImage()** ליצירת הקובץ בסיום העבודה
  + צבירת הצבעים של הפיקסלים במטריצת הצבעים של פיקסלים בעזרת הפעולה **writePixel(int j, int i, java.awt.Color color)**. שני הפרמטרים הראשונים – אינדקס של הפיקסל – הראשון על ציר ה-**x** והשני על ציר ה-**y**. הפרמטר השלישי -צבע הפיקסל מהסוג שלצבע של ג'אוה פריימוורק.
  + החזקת הפרמטרים של **View Plane** הקשורים ליצירת התמונה – הגודל והרזולוציה.

עבור המודול הזה נוסיף חבילה חדשה: **renderer** ונוסיף את המודול **ImageWriter.java** בחבילה החדשה הזו

לאחר מכן בחבילה **elements** נוסיף מחלקה **AmbientLight** עבור "תאורה סביבתית". הבנאי של המחלקה יקבל שני פרמטרים:

* פרמטר מסוג **Color** – לאור מילוי מקורי (עוצמת האור לפי קומפוננטות **RGB**) –
* מסוג **double** – מקדם הנחתה של אור מילוי –

המחלקה תכיל שדה עצמת תאורת מילוי **\_intensity** מסוג **Color**, בנאי שמקבל ערכים כנ"ל ומחשב את העצמה הסופית של תאורת מילוי שנשמרת בשדה הנ"ל, ופונקציה **GetIntensity()** שתחזיר את ערך עצמת התאורה הסביבתית מסוג **Color**.

A close up of a logo

Description automatically generated**שלב 2 – בניית תמונה ראשונה בלי רינדור**

התפקיד של השלב הזה הוא לתרגל את השימוש במחלקה **ImageWriter**. בחבילת **unittests** נוסיף מודול בדיקה **ImageWriterTest** שיכיל בניית תמונה ראשונית – תמונה בצבע אחד עם רשת קוים בצבע שני. בטסט הזה נבנה רשת של **10x16** ריבועים במסך (**ViewPlane**) **1000** על **1600** יח"א ורזולוציה של **500** על **800**:

הצבע של הרקע ושל הרשת – לבחירתכם\ן.

**שלב 3 – בניית סצנה**

נוסיף חבילה חדשה **scene** ובתוכה מחלקה חדשה **Scene** שתכיל את השדות הפרטיים הבאים:

* שם הסצנה (**String \_name**)
* צבע רקע (**Color \_background**)
* תאורה סביבתית (**AmbientLight \_ambientLight**)
* אוסף הגופים של מודל תלת-המימד (**Geometries \_geometries**)
* מצלמה (**Camera \_camera**)
* מרחק מהמצלמה למסך המדומה (**double \_distance**)

במחלקה נממש את הפונקציות הבאות:

* בנאי המקבל את שם הסצנה (בלבד) שיבנה גם אוסף ריק של גופים עבור מודל **D3**.
* פעולות אחזור לכל התכונות
* פעולות עדכון עבור צבע רקע, תאורה סביבתית, מצלמה ומרחק ממצלמה עד **View Plane**
* פעולת הוספה של צורה\ות למודל תלת המימד:

**void addGeometries(Intersectable… geometries) {…}**

**שלב 4 – הוספת בדיקת מנוע רינדור בניית מנוע הרינדור**

1. ניקח מתיבת ההגשה מודול **RenderTests.java** ונוסיף אותו בחבילה **unittests**. פונקציית הבדיקה שבמודול תבנה מודל **D3** עם ארבעה משולשים וכדור, מצלמה, סצנה, אובייקט יצירת תמונה ואובייקט רינדור ותפעיל את בניית התמונה כפי שלמדנו. הפעם הבדיקה לא תודיע לנו האם היו תקלות או לאו אלא לאחר הרצת הבדיקה יש לעיין בתמונה שנוצרה בתיקיית הפרויקט ולוודא את תקינותה!
2. בחבילה **renderer** נבנה מחלקה **Render** (שתפקידה ליצור מהסצנה את מטריצת הצבעים של התמונה). המחלקה תכיל שדה של **ImageWriter** ושדה של סצנה. במחלקה נוסיף פונקציות ציבוריות:

* **renderImage()**
* **calcColor(Point3D p)**
* **getClosestPoint(List<Point3D> points): Point3D**
* **printGrid(int interval, java.awt.Color color)**

חובה להקפיד על הרשאות מתאימות כפי שנלמד בקורס התאורטי.

**שימו לב**: תפקיד הפונקציה **printGrid** הוא להוסיף רשת מעל התמונה אך לא לדרוס את התמונה לגמרי. הפעלת הפונקציה בטסט – לפי הצורך.

**שלב לבונוס – בניית סצנה מקובץ XML**

ניתן לקבל בונוס של עד 2 נק' ע"י:

* פעולת עזר לטסט מנוע רינדור המקבל שם קובץ **XML** ובונה ממנו את האובייקטים של סצנה ו-**imageWriter**
* חבילת ניתוח קובץ **XML** ובניית האובייקטים של סצנה ו-**imageWriter** בעזרת אחת הספריות של **java**: **DOM**, **SAX** או **JAXP**.

בהצלחה!